aws

Panduan Developer

AWS Elastic Beanstalk



Copyright © 2025 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

AWS Elastic Beanstalk: Panduan Developer

Copyright © 2025 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Merek dagang dan tampilan dagang Amazon tidak boleh digunakan sehubungan dengan produk atau layanan apa pun yang bukan milik Amazon, dengan cara apa pun yang dapat menyebabkan kebingungan di antara pelanggan, atau dengan cara apa pun yang merendahkan atau mendiskreditkan Amazon. Semua merek dagang lain yang tidak dimiliki oleh Amazon merupakan milik dari pemiliknya masing-masing, yang mungkin berafiliasi dengan, terhubung ke, atau disponsori oleh Amazon.

Table of Contents

Apa itu AWS Elastic Beanstalk? 1
Platform yang didukung 2
Alur kerja penerapan aplikasi 3
Harga 3
Langkah selanjutnya 4
Tutorial memulai 5
Apa yang akan Anda bangun 7
Langkah 1 - Buat aplikasi 7
Langkah 2 - Menyebarkan aplikasi Anda 10
Langkah 3 - Jelajahi lingkungan 13
Pemecahan masalah dengan catatan 16
Langkah 4 - Perbarui aplikasi Anda 18
Langkah 5 - Skala aplikasi Anda 19
Tingkatkan pengaturan kapasitas 20
Verifikasi peningkatan kapasitas 21
Bersihkan 21
Langkah selanjutnya 23
Menyiapkan CLI EB 24
Instal EB CLI 25
Pemasangan manual 26
Instal di virtualenv 28
Memasang EB CLI dengan homebrew 29
Mengonfigurasi EB CLI 30
Mengabaikan file menggunakan .ebignore 32
Menggunakan profil bernama 33
Men-deploy artifact bukan folder proyek 33
Pengaturan konfigurasi dan prioritas 34
Metadata instans 35
Menggunakan EB CLI dengan Git 35
Mengaitkan lingkungan Elastic Beanstalk dengan cabang Git 35
Men-deploy perubahan 36
Menggunakan submodul Git 36
Menetapkan tanda Git ke versi aplikasi Anda 37
Perintah EB CLI 38

Opsi umum 39
eb abort 40
Deskripsi 40
Sintaksis 40
Opsi 40
Output 40
Contoh 40
eb appversion 41
Deskripsi 41
Sintaksis 41
Opsi 41
Menggunakan perintah secara interaktif 43
Output 44
Contoh 44
eb clone 45
Deskripsi 45
Sintaksis 46
Opsi 46
Output 47
Contoh 47
eb codesource 48
Deskripsi 48
Sintaksis 48
Opsi 48
Output 49
Contoh 49
eb config 49
Deskripsi 49
Sintaks 50
Opsi 51
Output 54
Contoh 55
eb console 59
Deskripsi 59
Sintaksis 59
Opsi 59

eb create 59
Deskripsi 59
Sintaks 60
Opsi 61
Output 73
Contoh 74
eb deploy 76
Deskripsi 76
Sintaksis 76
Opsi 77
Output 78
Contoh 78
eb events 78
Deskripsi 78
Sintaksis 78
Opsi 79
Output 79
Contoh 79
eb health 80
Deskripsi 80
Sintaksis 80
Opsi 80
Output 80
Contoh 81
eb init 82
Deskripsi 82
Sintaksis 82
Opsi 82
CodeBuild dukungan 85
Output 85
Contoh 85
eb labs 86
Deskripsi 86
eb list 87
Deskripsi 87
Sintaksis 87

Opsi ... 87
Output ... 87
Contoh 1 ... 88
Contoh 2 ... 88
eb local ... 88
Deskripsi ... 88
Sintaksis ... 89
Opsi ... 89
Output ... 90
Contoh ... 90
eb logs ... 92
Deskripsi ... 92
Sintaks ... 92
Opsi ... 93
Output ... 96
Contoh ... 96
eb migrate ... 96
Deskripsi ... 96
Sintaksis ... 98
Subperintah ... 98
Opsi ... 100
Output ... 107
Contoh ... 107
eb open ... 111
Deskripsi ... 111
Sintaksis ... 112
Opsi ... 112
Output ... 112
eb platform ... 112
Deskripsi ... 112
Menggunakan platform eb untuk platform khusus ... 113
Menggunakan platform eb untuk lingkungan ... 120
eb printenv ... 122
Deskripsi ... 122
Sintaksis ... 122
Opsi ... 122

Output 123
Contoh 123
eb restore 123
Deskripsi 123
Sintaksis 123
Opsi 123
Output 123
Contoh 123
eb scale 124
Deskripsi 124
Sintaksis 124
Opsi 125
Output 125
Contoh 125
eb setenv 125
Deskripsi 125
Sintaks 125
Opsi 126
Output 126
Contoh 126
eb ssh 127
Deskripsi 127
Sintaksis 128
Opsi 128
Output 129
Contoh 129
eb status 130
Deskripsi 130
Sintaksis 130
Opsi 130
Output 130
Contoh 131
eb swap 131
Deskripsi 131
Sintaks 132
Opsi 132

Output 132
Contoh 132
eb tags 133
Deskripsi 133
Sintaks 133
Opsi 134
Output 135
Contoh 136
eb terminate 137
Deskripsi 137
Sintaksis 138
Opsi 138
Output 138
Contoh 138
eb upgrade 139
Deskripsi 139
Sintaksis 139
Opsi 140
Output 140
Contoh 140
eb use 141
Deskripsi 141
Sintaksis 141
Opsi 141
Konsep 142
Aplikasi 143
Versi aplikasi 143
Lingkungan 143
Tingkat lingkungan 143
Konfigurasi lingkungan 144
Konfigurasi tersimpan 144
Platform 144
Lingkungan server web 144
Lingkungan pekerja 146
Pertimbangan desain 147
Skalabilitas 148

Keamanan 149
Penyimpanan tetap 149
Toleransi kesalahan 151
Pengiriman konten 151
Pembaruan perangkat lunak dan patch 151
Konektivitas 152
Mengelola aplikasi 153
Konsol pengelola aplikasi 154
Mengelola versi aplikasi 155
Membuat versi aplikasi 155
Menghapus versi aplikasi 156
Siklus hidup versi 158
Melabeli versi aplikasi 159
Membuat sumber paket 162
Membuat sebuah paket sumber dari baris perintah 163
Membuat paket sumber dengan Git 163
zip file di Mac OS X Finder atau Windows explorer 164
Membuat bundel sumber untuk aplikasi NET 165
Menguji bundel sumber Anda 166
Membangun dengan CodeBuild 166
Membuat aplikasi 167
Membangun dan men-deploy kode aplikasi Anda 167
Aplikasi pemberian label 169
Menambahkan label selama pembuatan aplikasi 169
Mengelola label dari aplikasi yang ada 170
Pelabelan sumber daya 171
Sumber daya yang dapat Anda beri label 172
Tag propagasi untuk meluncurkan template 173
Membuat lingkungan 175
Konsol manajemen lingkungan 176
Mengakses konsol tersebut 177
Panel ikhtisar lingkungan 178
Detail lingkungan 179
Tindakan lingkungan 183
Membuat lingkungan 185
Wizard pembuatan lingkungan baru 191

Mengkloning lingkungan ... 213
Menghirkan lingkungan ... 215
Dengan AWS CLI ... 217
Dengan API ... 219
Meluncurkan URL Sekarang ... 223
Lingkungan pembuatan ... 227
Lingkungan penyusunan ... 230
Deployment ... 233
Memilih kebijakan deployment ... 234
Men-deploy versi aplikasi baru ... 236
Men-deploy ulang versi sebelumnya ... 237
Cara lain untuk men-deploy aplikasi Anda ... 237
Pilihan deployment ... 237
Deployment Biru/Hijau ... 246
Perubahan konfigurasi ... 248
Pembaruan bergulir ... 250
Pembaruan tetap ... 254
Pembaruan platform ... 258
Metode 1 – Perbarui versi platform lingkungan Anda ... 262
Metode 2 - Lakukan Blue/Green penyebaran ... 263
Pembaruan yang dikelola ... 265
Tingkatkan lingkungan legasi ... 272
Migrasi ke 023/ AL2 AL2 ... 274
FAQ Pensiun Platform ... 291
Membatalkan pembaruan ... 295
Membangun kembali lingkungan ... 297
Membangun kembali lingkungan yang sedang berjalan ... 297
Membangun kembali lingkungan yang diakhiri ... 298
Jenis lingkungan ... 300
Lingkungan yang dapat diskalakan dengan beban seimbang ... 300
Lingkungan instans tunggal ... 300
Mengubah jenis lingkungan ... 301
Lingkungan pekerja ... 302
Daemon SQS lingkungan pekerja ... 305
Antrean surat mati ... 306
Tugas periodik ... 307

Gunakan Amazon CloudWatch untuk penskalaan otomatis di tingkat lingkungan pekerja 309
Mengonfigurasi lingkungan pekerja 309
Tautan lingkungan 313
Memulihkan dari status tumpukan yang tidak valid 315
Mengatasi kesalahan 315
Mengapa kesalahan terjadi 315
Mengonfigurasi lingkungan 317
Sumber daya yang disediakan 317
Konfigurasi menggunakan konsol 319
Halaman konfigurasi 319
Tinjau halaman perubahan 321
EC2 Contoh Amazon 322
Jenis EC2 instans Amazon 323
Mengkonfigurasi dengan konsol 325
Mengelola kelompok EC2 keamanan 333
Mengkonfigurasi dengan AWS CLI 335
Mengkonfigurasi dengan namespace 345
IMDS 347
Grup Auto Scaling 349
Templat peluncuran 351
Dukungan Instans Spot 354
Pemicu 371
Tindakan terjadwal 373
Pengaturan pemeriksaan kondisi 378
Penyeimbang beban 379
Classic Load Balancer 381
Application Load Balancer 392
Application Load Balancer Bersama 412
Penyeimbang Beban Jaringan 430
Mengonfigurasi log akses 442
Basis data 442
Siklus hidup basis data 443
Menambahkan instans Amazon RDS DB ke lingkungan Anda menggunakan konsol 444
Menghubungkan ke basis data 446
Mengkonfigurasi instans RDS DB terintegrasi menggunakan konsol 446
Mengkonfigurasi instans RDS DB terintegrasi menggunakan file konfigurasi 447

Memisahkan instance RDS DB menggunakan konsol 448
Memisahkan instance RDS DB menggunakan file konfigurasi 451
Keamanan 453
Mengonfigurasi keamanan lingkungan Anda 453
Namespace konfigurasi keamanan lingkungan 456
Menandai lingkungan 456
Menambahkan tag selama pembuatan lingkungan 457
Mengelola tag dari lingkungan yang ada 458
Variabel lingkungan dan pengaturan perangkat lunak 460
Mengonfigurasi pengaturan spesifik platform 461
Mengkonfigurasi properti lingkungan (variabel lingkungan) 462
Namespace pengaturan perangkat lunak 464
Mengakses properti lingkungan 466
Debugging 467
Meninjau log 470
Pemberitahuan 472
Mengonfigurasi pemberitahuan menggunakan konsol Elastic Beanstalk 474
Mengonfigurasi pemberitahuan menggunakan opsi konfigurasi 474
Mengonfigurasi izin untuk mengirim pemberitahuan 476
Amazon VPC 478
Mengonfigurasi pengaturan VPC di konsol Elastic Beanstalk 479
Namespace aws:ec2:vpc 482
Migrasi dari EC2 -Classic ke VPC 483
Nama domain 488
Mengonfigurasi lingkungan (lanjutan) 490
Opsi konfigurasi 491
Precedence 491
Nilai yang disarankan 492
Sebelum pembuatan lingkungan 494
Selama pembuatan 500
Setelah pembuatan 507
Opsi umum 517
Opsi spesifik platform 608
Opsi khusus 620
.Ebextensions 621
Pengaturan opsi 624

Server Linux ... 626
Server Windows ... 643
Sumber daya khusus ... 653
Konfigurasi tersimpan ... 681
Menandai konfigurasi tersimpan ... 687
`env.yaml` ... 689
Citra kustom ... 692
Membuat AMI kustom ... 693
Mengelola AMI kustom ... 697
Membersihkan AMI kustom ... 697
AMI berdasarkan platform pensiunan ... 698
File statis ... 704
Konfigurasikan file statis menggunakan konsol tersebut ... 705
Konfigurasikan file statis menggunakan opsi konfigurasi ... 706
HTTPS ... 707
Sertifikat server ... 708
Hentikan HTTPS di penyeimbang beban ... 715
Hentikan HTTPS pada instance ... 719
End-to-end enkripsi ... 756
TCP Passthrough ... 760
Pengalihan HTTP ke HTTPS ... 761
Platform ... 764
Glosarium platform ... 764
Model tanggung jawab bersama ... 767
Kebijakan dukungan platform ... 769
Cabang platform pensiun ... 769
Melampaui masa tenggang 90 hari ... 770
Jadwal platform ... 770
Sumber daya perencanaan ... 771
Rilis cabang platform mendatang ... 771
Jadwal cabang platform yang pensiun ... 772
Sejarah cabang platform pensiunan ... 772
Riwayat server dan OS ... 777
Platform yang didukung ... 779
Platform dan riwayat komponen yang didukung ... 779
Platform Linux ... 780

Versi Amazon Linux yang didukung 780
Daftar platform Linux Elastic Beanstalk 782
Alur kerja deployment instans 782
Alur kerja penerapan instans untuk ECS di dan yang lebih baru AL2 785
Alat skrip platform 788
Memperluas platform Linux 798
Buildfile dan Procfile 798
Hook platform 800
File konfigurasi 802
Konfigurasi proksi terbalik 803
Contoh aplikasi dengan ekstensi 807
Menyebarkan .NET (Windows) 809
QuickStart untuk .NET Core di Windows 810
AWS Akun Anda 811
Prasyarat 813
Langkah 1: Buat .NET Core pada aplikasi Windows 813
Langkah 2: Jalankan aplikasi Anda secara lokal 814
Langkah 3: Menyebarkan .NET Core Anda pada aplikasi Windows dengan EB CLI 815
Langkah 4: Jalankan aplikasi Anda di Elastic Beanstalk 816
Langkah 5: Bersihkan 816
AWS sumber daya untuk aplikasi Anda 816
Langkah selanjutnya 817
Terapkan dengan konsol 818
QuickStart untuk ASP.NET 818
AWS Akun Anda 818
Prasyarat 821
Langkah 1: Buat aplikasi ASP.NET 821
Langkah 2: Jalankan aplikasi Anda secara lokal 822
Langkah 3: Menyebarkan aplikasi ASP.NET Anda dengan AWS Toolkit for Visual Studio 822
Langkah 4: Jalankan aplikasi Anda di Elastic Beanstalk 823
Langkah 5: Bersihkan 823
AWS sumber daya untuk aplikasi Anda 824
Langkah selanjutnya 824
Terapkan dengan konsol 825
Lingkungan pengembangan 825
Menginstal IDE 826

Memasang AWS Toolkit for Visual Studio ... 826
Platform .NET ... 826
Mengonfigurasi lingkungan .NET Anda di konsol Elastic Beanstalk ... 827
Namespace aws:elasticbeanstalk:container:dotnet:apppool ... 828
Migrasi versi utama ... 829
Manifes deployment ... 832
EC2 Peluncuran Cepat ... 851
Menambahkan basis data ... 852
Menambahkan instans DB ke lingkungan Anda ... 853
Mengunduh driver ... 854
Menghubungkan ke basis data ... 854
The AWS Toolkit for Visual Studio ... 855
Uji secara lokal ... 856
Membuat lingkungan Elastic Beanstalk ... 856
Mengakhiri lingkungan ... 866
Deploy ... 867
Mengelola lingkungan ... 870
Mengelola akun ... 884
Debug ... 885
Memantau ... 886
Alat deployment ... 888
Memigrasi aplikasi On-Premise ... 891
Rekomendasi komponen pensiun ... 891
Pensiunan cabang Windows 2012 R2 ... 891
TLS 1.2 Kompatibilitas ... 892
Menyebarkan inti .NET (Linux) ... 894
QuickStart untuk .NET Core di Linux ... 895
AWS Akun Anda ... 895
Prasyarat ... 897
Langkah 1: Buat .NET Core pada aplikasi Linux ... 898
Langkah 2: Jalankan aplikasi Anda secara lokal ... 898
Langkah 3: Menyebarkan .NET Core Anda pada aplikasi Linux dengan EB CLI ... 899
Langkah 4: Jalankan aplikasi Anda di Elastic Beanstalk ... 901
Langkah 5: Bersihkan ... 816
AWS sumber daya untuk aplikasi Anda ... 901
Langkah selanjutnya ... 902

Terapkan dengan konsol ... 902
Lingkungan pengembangan ... 902
Menginstal .NET Core SDK ... 903
Menginstal IDE ... 826
Instalasi AWS Toolkit for Visual Studio ... 826
Inti .NET pada platform Linux ... 903
.NET Core pada pertimbangan platform Linux ... 904
Mengonfigurasi .NET Core Anda di lingkungan Linux ... 905
.NET Core di namespace konfigurasi Linux ... 906
Aplikasi paketan ... 906
Procfile ... 908
Server proksi ... 909
The AWS Toolkit for Visual Studio ... 910
Prasyarat ... 910
Membuat proyek aplikasi baru ... 911
Membuat lingkungan Elastic Beanstalk dan men-deploy aplikasi Anda ... 912
Mengakhiri lingkungan ... 914
Mengelola lingkungan ... 914
Memantau ... 928
Migrasi dari Windows ke Linux ... 929
Pertimbangan untuk migrasi ke .NET Core di platform Linux ... 930
Menerapkan Go ... 931
QuickStart untuk Go ... 931
AWS Akun Anda ... 932
Prasyarat ... 934
Langkah 1: Buat aplikasi Go ... 934
Langkah 2: Terapkan aplikasi Go Anda dengan EB CLI ... 935
Langkah 3: Jalankan aplikasi Anda di Elastic Beanstalk ... 936
Langkah 4: Membersihkan ... 816
AWS sumber daya untuk aplikasi Anda ... 937
Langkah selanjutnya ... 938
Terapkan dengan konsol ... 938
Lingkungan pengembangan ... 938
Menginstal Go ... 938
Menginstal AWS SDK for Go ... 938
Platform Go ... 939

Mengonfigurasi lingkungan Go Anda ... 940
Namespace konfigurasi Go ... 941
Procfile ... 943
Buildfile ... 945
Konfigurasi proxy ... 945
Menerapkan Java ... 948
QuickStart untuk Java ... 949
AWS Akun Anda ... 950
Prasyarat ... 952
Langkah 1: Buat aplikasi Java ... 953
Langkah 2: Jalankan aplikasi Anda secara lokal ... 955
Langkah 3: Menyebarkan aplikasi Java Anda dengan EB CLI ... 956
Langkah 4: Jalankan aplikasi Anda di Elastic Beanstalk ... 957
Langkah 5: Bersihkan ... 958
AWS sumber daya untuk aplikasi Anda ... 958
Langkah selanjutnya ... 959
Terapkan dengan konsol ... 959
QuickStart untuk Java di Tomcat ... 959
AWS Akun Anda ... 960
Prasyarat ... 962
Langkah 1: Buat aplikasi Java JSP ... 962
Langkah 2: Menyebarkan aplikasi Java JSP Anda dengan EB CLI ... 963
Langkah 3: Jalankan aplikasi Anda di Elastic Beanstalk ... 964
Langkah 4: Membersihkan ... 816
AWS sumber daya untuk aplikasi Anda ... 964
Langkah selanjutnya ... 965
Terapkan dengan konsol ... 966
Lingkungan pengembangan ... 966
Menginstal kit pengembangan Java ... 966
Menginstal kontainer web ... 966
Mengunduh pustaka ... 966
Menginstal AWS SDK for Java ... 967
Menginstal IDE atau editor teks ... 967
Contoh aplikasi dan tutorial ... 968
Meluncurkan lingkungan dengan sampel aplikasi Java ... 968
Langkah selanjutnya ... 977

Platform Tomcat 977
Mengonfigurasi lingkungan Tomcat Anda 979
Namespace konfigurasi Tomcat 982
Paketan file WAR 985
Menata folder proyek Anda 986
Konfigurasi proxy 989
Platform Java SE 995
Mengonfigurasi lingkungan Java SE Anda 996
Namespace konfigurasi Java SE 997
Buildfile 999
Procfile 1000
Konfigurasi proksi 1001
Menambahkan basis data 1005
Mengunduh driver JDBC 1006
Menghubungkan ke basis data (platform Java SE) 1007
Menghubungkan ke basis data (platform Tomcat) 1008
Pemecahan masalah koneksi basis data 1011
Sumber daya 1013
Menerapkan Node.js 1014
QuickStart untuk Node.js 1014
AWS Akun Anda 1015
Prasyarat 1017
Langkah 1: Buat aplikasi Node.js 1018
Langkah 2: Jalankan aplikasi Anda secara lokal 1019
Langkah 3: Terapkan aplikasi Node.js Anda dengan EB CLI 1019
Langkah 4: Jalankan aplikasi Anda di Elastic Beanstalk 1020
Langkah 5: Bersihkan 816
AWS sumber daya untuk aplikasi Anda 1020
Langkah selanjutnya 1021
Terapkan dengan konsol 1022
Lingkungan pengembangan 1022
Instal Node.js 1022
Konfirmasi instalasi npm 1022
Instal AWS SDK untuk Node.js 1023
Pasang generator Express 1023
Siapkan kerangka kerja dan server Express 1024

Platform Node.js 1025
Mengkonfigurasi lingkungan Anda Node.js 1026
Node.jskonfigurasi namespace 1029
Procfile 1032
Mengonfigurasi dependensi 1033
File npm shrinkwrap 1037
Konfigurasi proxy 1038
Contoh aplikasi dan tutorial 1041
Meluncurkan lingkungan dengan sampel aplikasi Node.js 1042
Langkah selanjutnya 1043
Tutorial - Express 1043
Prasyarat 1043
Membuat lingkungan Elastic Beanstalk 1044
Perbarui aplikasi untuk menggunakan Express 1046
Perbarui aplikasi untuk menggunakan Amazon RDS 1050
Bersihkan 1055
Tutorial - Express dengan pengklasteran 1055
Prasyarat 1056
Membuat lingkungan Elastic Beanstalk 1056
Perbarui aplikasi untuk menggunakan Express 1059
Pembersihan 1073
Tutorial - Node.js w/ DynamoDB 1073
Prasyarat 1074
Membuat lingkungan Elastic Beanstalk 1074
Menambahkan izin ke instans lingkungan Anda 1077
Menyebarkan contoh aplikasi 1078
Buat tabel DynamoDB 1080
Perbarui file konfigurasi aplikasi 1081
Konfigurasikan lingkungan Anda untuk ketersediaan yang tinggi 1083
Pembersihan 1083
Langkah selanjutnya 1084
Menambahkan basis data 1084
Menambahkan instans DB ke lingkungan Anda 1085
Mengunduh driver 1086
Menyambungkan ke basis data 1087
Sumber daya 1088

Menerapkan PHP 1089
QuickStart untuk PHP 1089
AWS Akun Anda 1090
Prasyarat 1092
Langkah 1: Buat aplikasi PHP 1092
Langkah 2: Jalankan aplikasi Anda secara lokal 1092
Langkah 3: Inisialisasi dan gunakan aplikasi PHP Anda 1093
Langkah 4: Jelajahi aplikasi cloud Anda 1093
Langkah 5: Perbarui dan terapkan ulang aplikasi Anda 1093
Bersihkan 816
Langkah selanjutnya 1095
Platform PHP 1095
Menginstal AWS SDK for PHP 1096
Pertimbangan untuk PHP 8.1 di Amazon Linux 2 1097
Mengonfigurasi lingkungan PHP Anda 1099
Ruang nama untuk konfigurasi 1101
Menginstal dependensi 1102
Memperbarui Composer 1103
Memperluas php.ini 1105
Contoh lanjutan 1105
Menambahkan basis data 1106
Tutorial - Laravel 1109
Tutorial - CakePHP 1119
Tutorial - Symfony 1127
Tutorial - Produksi HA 1133
Tutorial - HA WordPress 1144
Tutorial - HA Drupal 1161
Menerapkan Python 1179
QuickStart untuk Python 1179
AWS Akun Anda 1180
Prasyarat 1182
Langkah 1: Buat aplikasi Python 1184
Langkah 2: Jalankan aplikasi Anda secara lokal 1184
Langkah 3: Menyebarkan aplikasi Python Anda dengan EB CLI 1185
Langkah 4: Jalankan aplikasi Anda di Elastic Beanstalk 1186
Langkah 5: Bersihkan 816

AWS sumber daya untuk aplikasi Anda 1186
Langkah selanjutnya 1187
Terapkan dengan konsol 1187
Lingkungan pengembangan 1188
Prasyarat 1188
Menggunakan lingkungan virtual 1189
Mengonfigurasi proyek Python pada Elastic Beanstalk 1190
Platform Python 1191
Mengonfigurasi lingkungan Python Anda 1192
Namespace konfigurasi Python 1194
Yang dapat `python3` dieksekusi 1196
Procfile 1196
Menentukan dependensi 1197
Tutorial - flask 1200
Prasyarat 1200
Menyiapkan lingkungan virtual Python dengan Flask 1201
Buat aplikasi Flask 1202
Men-deploy situs Anda dengan EB CLI 1204
Pembersihan 1208
Langkah selanjutnya 1208
Tutorial - Django 1208
Prasyarat 1209
Menyiapkan lingkungan virtual Python dan menginstal Django 1209
Buat proyek Django 1211
Konfigurasikan aplikasi Django Anda untuk Elastic Beanstalk 1213
Men-deploy situs Anda dengan EB CLI 1215
Perbarui aplikasi Anda 1218
Bersihkan 1222
Langkah selanjutnya 1222
Menambahkan basis data 1223
Menambahkan instans DB ke lingkungan Anda 1223
Mengunduh driver 1224
Menghubungkan ke basis data 1225
Sumber daya 1225
Menyebarkan Ruby 1227
Lingkungan pengembangan 1227

Menginstal Ruby 1228
Memasang AWS SDK for Ruby 1229
Menginstal IDE atau editor teks 1229
Platform Ruby 1230
Mengonfigurasi lingkungan Ruby Anda 1231
Namespace konfigurasi Ruby 1234
Gemfile 1235
Procfile 1236
Tutorial - rails 1237
Prasyarat 1238
Pengetahuan Elastic Beanstalk dasar 1238
Meluncurkan lingkungan Elastic Beanstalk 1238
Menginstal rails dan menghasilkan sebuah situs web 1240
Mengonfigurasi pengaturan rails 1244
Deploy aplikasi Anda 1245
Pembersihan 1246
Langkah selanjutnya 1246
Tutorial - sinatra 1246
Prasyarat 1247
Meluncurkan lingkungan Elastic Beanstalk 1247
Tulis situs web dasar sinatra 1249
Men-deploy aplikasi Anda 1251
Pembersihan 1251
Langkah selanjutnya 1252
Menambahkan basis data 1252
Menambahkan instans DB ke lingkungan Anda 1252
Mengunduh adaptor 1254
Menyambungkan ke basis data 1254
Menerapkan dengan Docker 1255
Cabang platform Docker 1255
Cabang platform pensiunan berjalan di Amazon Linux AMI () AL1 1256
Cabang platform Docker 1257
QuickStart untuk Docker 1257
Konfigurasi gambar Docker 1265
Cabang platform yang dikelola ECS 1271
Ikhtisar platform Docker yang dikelola ECS 1272

Sumber daya Amazon ECS dibuat oleh Elastic Beanstalk 1272
Dockerrun.aws.jsonberkas v2 1273
Gambar Docker 1274
Deployment kontainer gagal 1274
Memperluas platform Docker berbasis ECS 1274
Konfigurasi Docker terkelola ECS untuk Elastic Beanstalk 1275
Tutorial - ECS dikelola Docker 1287
Migrasi ke ECS berjalan pada 023 AL2 1296
Menggunakan gambar dari repositori pribadi 1298
Repositori Amazon ECR 1298
Toko Parameter SSM 1300
Berkas DockerRun.aws.json 1302
Konfigurasi lingkungan 1305
Mengkonfigurasi perangkat lunak di lingkungan Docker 1305
Referensi variabel lingkungan dalam kontainer 1307
Menggunakan fitur interpolasi untuk variabel lingkungan dengan Docker Compose 1309
Membuat log untuk pelaporan kesehatan yang ditingkatkan dengan Docker Compose 1310
Pencatatan khusus wadah Docker dengan Docker Compose 1311
Gambar Docker 1312
Mengkonfigurasi pembaruan terkelola untuk lingkungan Docker 1312
Ruang nama konfigurasi Docker 1313
Konfigurasi docker di Amazon Linux AMI (sebelumnya Amazon Linux 2) 1314
Platform warisan 1316
(Legacy) Migrasi ke Docker berjalan di Amazon Linux 2 1316
(Warisan) Kontainer Docker GlassFish 1321
Lingkungan pemantauan 1331
Memantau konsol 1331
Grafik pemantauan 1332
Menyesuaikan konsol pemantauan 1333
Memantau kesehatan dengan CLI 1334
Membaca output 1338
Tampilan kondisi interaktif 1340
Opsi tampilan kondisi interaktif 1342
Pelaporan kondisi dasar 1342
Warna kondisi 1343
Pemeriksaan kondisi Elastic Load Balancing 1344

Instans tunggal dan pemeriksaan kondisi lingkungan tingkat pekerja 1345
Pemeriksaan tambahan 1345
CloudWatch Metrik Amazon 1345
Pelaporan dan pemantauan kondisi yang ditingkatkan 1347
Agen kondisi Elastic Beanstalk 1349
Faktor dalam menentukan kondisi instans dan lingkungan 1350
Penyesuaian aturan pemeriksaan kondisi 1353
Peran kondisi yang ditingkatkan 1353
Otorisasi kondisi yang ditingkatkan 1354
Peristiwa kondisi yang ditingkatkan 1355
Perilaku pelaporan kondisi yang ditingkatkan selama pembaruan, deployment, dan penskalaan 1356
Aktifkan kondisi yang ditingkatkan 1356
Konsol kondisi 1360
Warna dan status kondisi 1365
Metrik instans 1368
Aturan kondisi yang ditingkatkan 1371
CloudWatch 1376
Pengguna API 1385
Format log kondisi yang ditingkatkan 1387
Pemberitahuan dan pemecahan masalah 1391
Mengelola alarm 1393
Lihat riwayat perubahan 1395
Melihat peristiwa 1396
Konsol 1397
Baris perintah 1397
Pemantauan instans 1398
Tampilkan log instans 1399
Log lokasi di EC2 instans Amazon 1401
Lokasi log di Amazon S3 1402
Pengaturan rotasi log pada Linux 1403
Memperluas konfigurasi tugas log default 1403
Streaming file log ke Amazon CloudWatch Logs 1407
Mengintegrasikan layanan AWS 1409
Gambaran umum arsitektur 1409
CloudFront 1410

CloudTrail 1411
Informasi Elastic Beanstalk di CloudTrail 1411
Memahami entri file log Elastic Beanstalk 1412
CloudWatch 1413
CloudWatch Log 1414
Prasyarat untuk melakukan streaming log ke Log CloudWatch 1415
Bagaimana Elastic Beanstalk mengatur Log CloudWatch 1416
Streaming log instance ke CloudWatch Log 1420
Pemecahan Masalah Integrasi Log CloudWatch 1423
Kondisi lingkungan streaming 1424
EventBridge 1427
Pantau sumber daya Elastic Beanstalk dengan EventBridge 1428
Contoh pola peristiwa Elastic Beanstalk 1431
Contoh peristiwa Elastic Beanstalk 1433
Pemetaan bidang peristiwa Elastic Beanstalk 1434
AWS Config 1437
Menyiapkan AWS Config 1438
Mengkonfigurasi AWS Config untuk merekam sumber daya Elastic Beanstalk 1438
Melihat detail konfigurasi Elastic Beanstalk di konsol AWS Config 1439
Mengevaluasi sumber daya Elastic Beanstalk menggunakan aturan AWS Config 1443
DynamoDB 1444
ElastiCache 1444
Amazon EFS 1445
File konfigurasi 1446
Sistem file terenkripsi 1447
Aplikasi sampel 1447
Membersihkan sistem file 1448
IAM 1448
Profil instans 1449
Peran layanan 1454
Menggunakan peran terkait layanan 1461
Kebijakan pengguna 1474
Format ARN 1482
Sumber daya dan kondisi 1484
Kontrol akses berbasis tag 1529
Contoh kebijakan terkelola 1533

Contoh kebijakan khusus sumber daya 1537
Akses bucket S3 lintas lingkungan 1547
Amazon RDS 1549
Amazon RDS dalam VPC default 1551
Kredensi Amazon RDS dan Secrets Manager 1557
Membersihkan instans Amazon RDS eksternal 1557
Amazon S3 1558
Bucket akun pelanggan Elastic Beanstalk Amazon S3 1558
Isi ember akun pelanggan Elastic Beanstalk Amazon S3 1558
Menghapus objek dalam bucket Elastic Beanstalk Amazon S3 1559
Menghapus bucket Elastic Beanstalk Amazon S3 1560
Secrets Manager & Systems Manager Parameter Store 1561
Ambil rahasia ke variabel lingkungan 1562
Izin IAM yang diperlukan 1569
Menggunakan Secrets Manager dan Systems Manager Parameter Store 1571
Memecahkan masalah rahasia dan variabel lingkungan 1573
Amazon VPC 1574
VPC publik 1576
VPC publik/privat 1577
VPC privat 1578
Host bastion 1580
Amazon RDS 1585
Titik akhir VPC 1592
Kebijakan VPC endpoint 1596
Keamanan 1607
Perlindungan data 1608
Enkripsi data 1609
Privasi antar jaringan 1610
Manajemen identitas dan akses 1610
AWS kebijakan terkelola 1611
Pencatatan dan pemantauan 1625
Pelaporan kondisi yang ditingkatkan 1625
Log EC2 instans Amazon 1625
Notifikasi lingkungan 1626
CloudWatch Alarm Amazon 1626
AWS CloudTrail log 1626

AWS X-Ray debugging 1626
Validasi kepatuhan 1626
Ketahanan 1627
Keamanan infrastruktur 1628
Model tanggung jawab bersama 1628
Praktik terbaik keamanan 1629
Praktik terbaik keamanan pencegahan 1629
Praktik terbaik keamanan detective 1630
Izin 1632
Peran yang dibutuhkan 1632
Kebijakan dan peran opsional 1632
Peran layanan 1633
AWSElasticBeanstalkEnhancedHealth 1455
AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy 1455
Profil instans 1635
Kebijakan terkelola 1636
Membuat profil EC2 instance 1636
Kebijakan pengguna 1636
Tutorial dan sampel 1638
Migrasi aplikasi IIS 1641
Prasyarat 1642
Glosarium migrasi 1644
Istilah Windows, IIS, dan .NET 1644
Istilah Elastic Beanstalk 1648
Istilah Python 1652
Migrasi IIS dasar 1653
Menjelajahi lingkungan IIS Anda 1653
Mempersiapkan migrasi 1655
Migrasi pertama Anda 1655
Mengontrol migrasi 1656
Memantau progres 1657
Memverifikasi migrasi 1657
Mengelola artefak migrasi 1658
Konfigurasi jaringan 1659
Konfigurasi VPC 1659
Penerapan multi-situs dengan konfigurasi port 1663

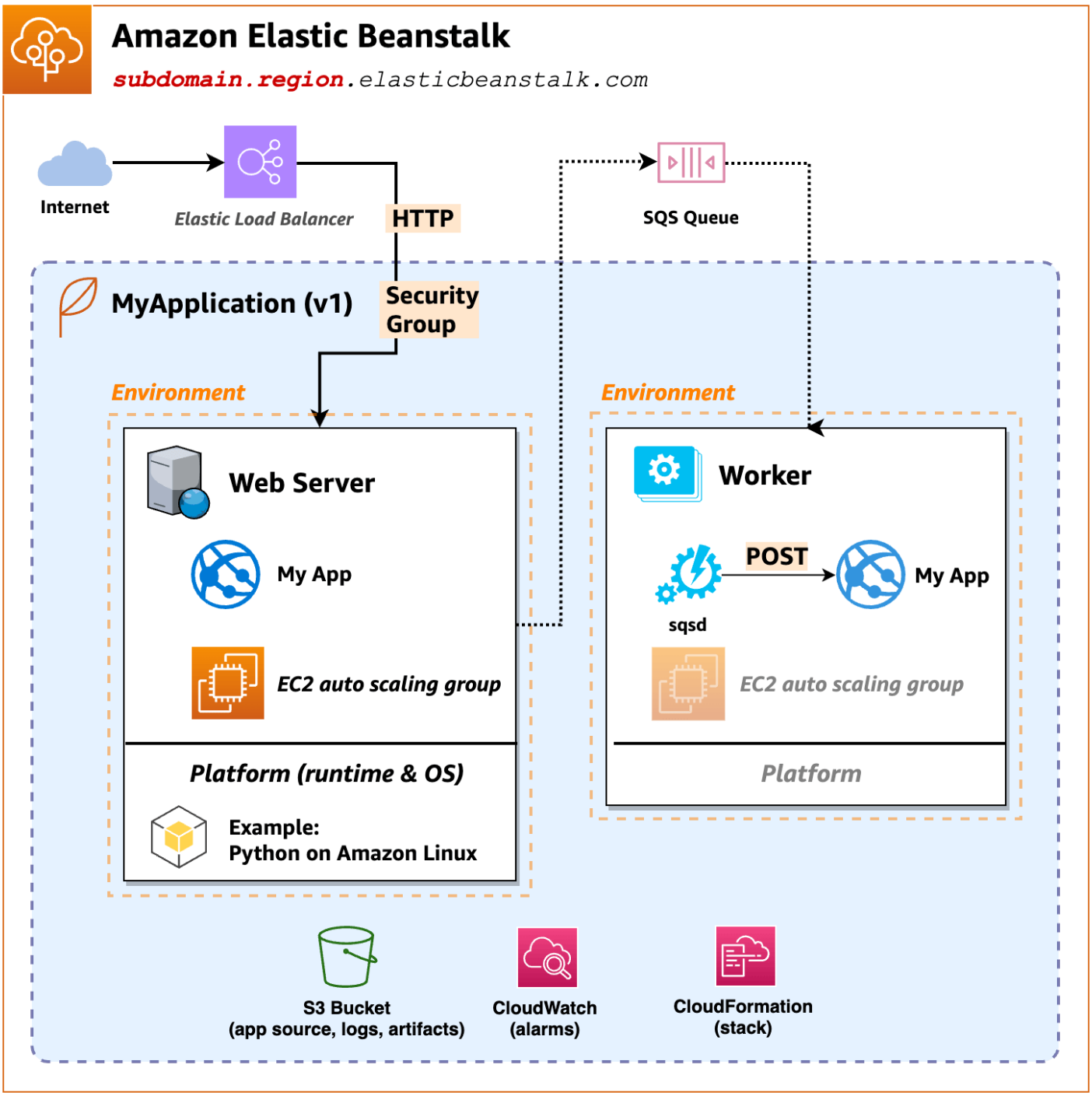
Konfigurasi dan dependensi bersama 1666
Praktik terbaik 1666
Pemecahan Masalah 1667
Konfigurasi keamanan 1668
Konfigurasi profil instance 1668
Manajemen peran layanan 1668
Konfigurasi grup keamanan 1669
Integrasi sertifikat SSL 1670
Otentikasi Windows 1671
Praktik terbaik dan pemecahan masalah 1672
Pemetaan migrasi IIS ke Elastic Beanstalk 1674
Situs dan aplikasi IIS di Elastic Beanstalk 1674
Direktori virtual dan manajemen jalur aplikasi 1675
Penulisan ulang URL dan perutean permintaan aplikasi (ARR) 1676
Struktur artefak migrasi 1677
Skenario migrasi lanjutan 1679
Migrasi multi-situs dengan Application Request Routing (ARR) 1679
Migrasi multi-situs tanpa ARR menggunakan perutean berbasis host 1682
Manajemen direktori virtual 1686
Pengaturan kolam aplikasi kustom 1690
Menerapkan versi sebelumnya 1692
Pemecahan masalah dan diagnostik 1693
Mengaitkan EC2 keypair dengan lingkungan Anda 1693
Mengakses log 1694
Mengakses artefak sisi klien 1694
Memantau kesehatan lingkungan 1694
EC2 optimasi kinerja 1695
Konfigurasi volume EBS 1695
Masalah dan solusi umum 1696
Mendapatkan Dukungan 1696
Opsi migrasi: EB CLI vs MGN 1697
Kapan menggunakan setiap opsi migrasi 1698
Perbandingan alur kerja migrasi 1699
Kesimpulan 1699
Pemecahan Masalah 1701
Menggunakan alat Systems Manager 1702

Panduan umum 1703
Variabel lingkungan untuk rahasia 1704
Pembuatan lingkungan 1705
Deployment 1706
Kondisi 1706
Konfigurasi 1707
Docker 1707
Pertanyaan yang Sering Diajukan 1708
Pemecahan masalah deployment 1709
Kesalahan penerapan 1710
Sumber Daya 1714
Aplikasi sampel 1715
AWS SDK untuk Java 1715
AWS SDK untuk .NET 1716
AWS Toolkit for Visual Studio 1716
AWS SDK untuk JavaScript di Node.js 1716
AWS SDK untuk PHP 1717
AWS SDK for Python (Boto) 1717
AWS SDK untuk Ruby 1717
Riwayat dokumen 1718
..... mdccxxiv

Apa itu AWS Elastic Beanstalk?

Dengan Elastic Beanstalk Anda dapat menyebarkan aplikasi web AWS ke Cloud pada berbagai platform yang didukung. Anda membangun dan menyebarkan aplikasi Anda. Elastic Beanstalk menyediakan instans Amazon EC2 , mengonfigurasi load balancing, mengatur pemantauan kesehatan, dan menskalakan lingkungan Anda secara dinamis.

Selain lingkungan server web, Elastic Beanstalk juga menyediakan lingkungan pekerja yang dapat Anda gunakan untuk memproses pesan dari antrian Amazon SQS, berguna untuk tugas asinkron atau yang berjalan lama. Untuk informasi selengkapnya, lihat Lingkungan pekerja Elastic Beanstalk.



Platform yang didukung

Elastic Beanstalk mendukung aplikasi yang dikembangkan di Go, Java, .NET, Node.js, PHP, Python, dan Ruby. Elastic Beanstalk juga mendukung wadah Docker, di mana Anda dapat memilih bahasa pemrograman dan dependensi aplikasi Anda sendiri. Saat Anda menerapkan aplikasi, Elastic

Beanstalk akan membuat versi platform yang didukung dan menyediakan satu atau AWS beberapa sumber daya, seperti instans EC2 Amazon, di akun Anda untuk menjalankan aplikasi. AWS

Anda dapat berinteraksi dengan Elastic Beanstalk melalui AWS Command Line Interface konsol Elastic Beanstalk, (), atau EB CLI AWS CLI, alat baris perintah tingkat tinggi yang dirancang khusus untuk Elastic Beanstalk.

Anda dapat melakukan sebagian besar tugas penerapan, seperti mengubah ukuran armada EC2 instans Amazon atau memantau aplikasi Anda, langsung dari antarmuka web Elastic Beanstalk (konsol).

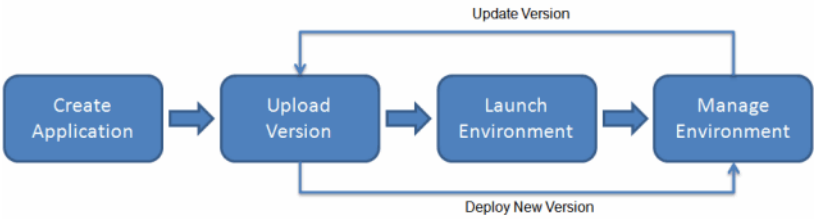
Untuk mempelajari lebih lanjut tentang cara menerapkan contoh aplikasi web menggunakan Elastic Beanstalk, lihat. Pelajari cara memulai dengan Elastic Beanstalk

Alur kerja penerapan aplikasi

Untuk menggunakan Elastic Beanstalk, Anda membuat aplikasi, lalu mengunggah bundel sumber aplikasi Anda ke Elastic Beanstalk. Selanjutnya, Anda memberikan informasi tentang aplikasi, dan Elastic Beanstalk secara otomatis meluncurkan lingkungan dan membuat serta AWS mengonfigurasi sumber daya yang diperlukan untuk menjalankan kode Anda.

Setelah Anda membuat dan menyebarkan aplikasi Anda dan lingkungan Anda diluncurkan, Anda dapat mengelola lingkungan Anda dan menerapkan versi aplikasi baru. Informasi tentang aplikasi—termasuk metrik, peristiwa, dan status lingkungan—tersedia melalui konsol Elastic Beanstalk, dan Antarmuka Baris Perintah. APIs

Diagram berikut menggambarkan alur kerja Elastic Beanstalk:



Harga

Tidak ada biaya tambahan untuk Elastic Beanstalk. Anda hanya membayar untuk AWS sumber daya dasar yang digunakan aplikasi Anda. Untuk detail tentang harga, lihat Halaman detail layanan Elastic Beanstalk.

Langkah selanjutnya

Kami merekomendasikan tutorial,Tutorial memulai, untuk mulai menggunakan Elastic Beanstalk. Tutorial ini membantu Anda membuat, melihat, dan memperbarui contoh aplikasi Elastic Beanstalk.

Pelajari cara memulai dengan Elastic Beanstalk

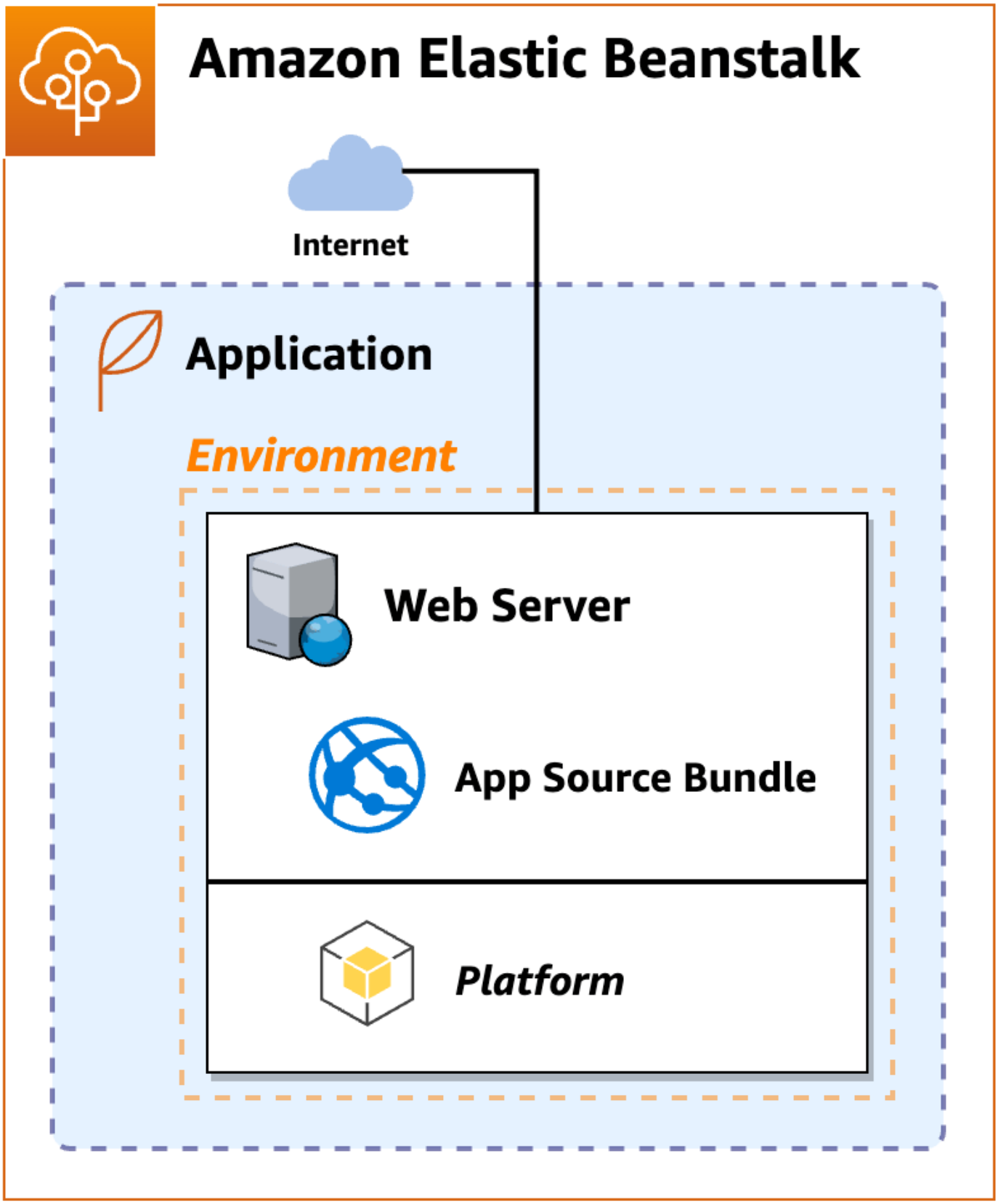
Dengan Elastic Beanstalk Anda dapat menyebarkan, memantau, dan menskalakan aplikasi dan layanan web. Biasanya, Anda akan mengembangkan kode Anda secara lokal kemudian menerapkannya ke instance EC2 server Amazon. Contoh ini, juga disebut lingkungan, berjalan pada platform yang dapat ditingkatkan melalui AWS konsol atau baris perintah.

Untuk memulai, sebaiknya gunakan aplikasi sampel pra-bangun langsung dari konsol. Kemudian, Anda dapat mempelajari cara mengembangkan secara lokal dan menyebarkan dari baris perintah di. the section called “QuickStart untuk PHP”

Tidak ada biaya untuk menggunakan Elastic Beanstalk, tetapi biaya standar AWS berlaku untuk sumber daya yang Anda buat selama tutorial ini sampai Anda menghapusnya di akhir. Total biaya biasanya kurang dari satu dolar. Untuk informasi tentang cara meminimalkan biaya, lihat tingkat gratis AWS.

Setelah menyelesaikan tutorial ini, Anda akan memahami dasar-dasar membuat, mengonfigurasi, menerapkan, memperbarui, dan memantau aplikasi Elastic Beanstalk dengan lingkungan yang berjalan pada instance Amazon. EC2

Perkiraan durasi: 35-45 menit


Amazon Elastic Beanstalk
Internet
Application
Environment
Web Server
App Source Bundle
Platform

Apa yang akan Anda bangun

Aplikasi Elastic Beanstalk pertama Anda akan terdiri dari satu lingkungan EC2 Amazon yang menjalankan sampel PHP pada platform yang dikelola PHP.

Aplikasi Elastic Beanstalk

: Aplikasi Elastic Beanstalk adalah wadah untuk komponen Elastic Beanstalk, termasuk lingkungan tempat kode aplikasi Anda berjalan pada platform yang disediakan dan dikelola oleh Elastic Beanstalk, atau dalam wadah khusus yang Anda sediakan.

Lingkungan

: Lingkungan Elastic Beanstalk adalah AWS kumpulan sumber daya yang berjalan bersama termasuk instans Amazon. EC2 Saat Anda membuat lingkungan, Elastic Beanstalk menyediakan sumber daya yang diperlukan ke akun Anda. AWS

Platform

: Platform adalah kombinasi dari sistem operasi, runtime bahasa pemrograman, server web, server aplikasi, dan komponen Elastic Beanstalk tambahan. Elastic Beanstalk menyediakan platform yang rusak, atau Anda dapat menyediakan platform Anda sendiri dalam sebuah wadah.

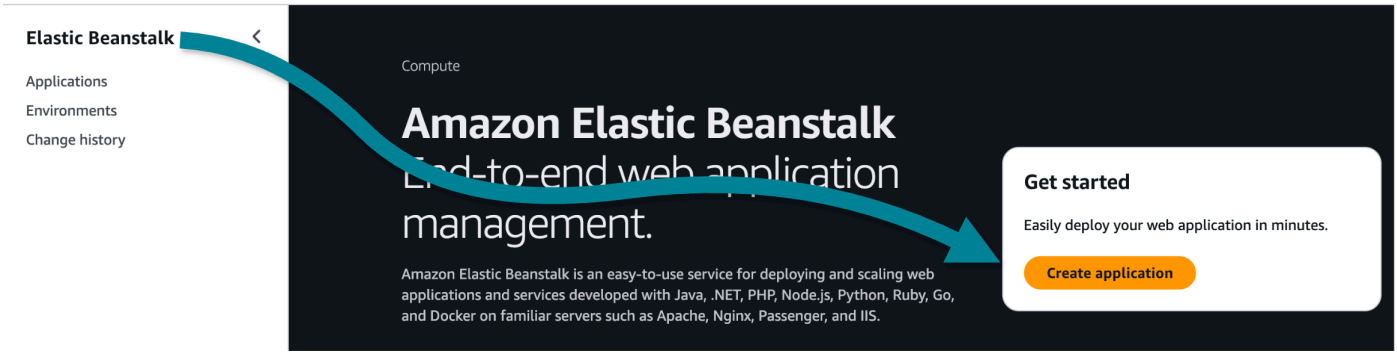
Elastic Beanstalk menyediakan platform untuk bahasa pemrograman yang berbeda, server aplikasi, dan kontainer Docker. Saat Anda membuat lingkungan, Anda harus memilih platform. Anda dapat meningkatkan platform, tetapi Anda tidak dapat mengubah platform untuk suatu lingkungan.

> **Beralih platform**
>
> Jika Anda perlu mengubah bahasa pemrograman, Anda harus membuat dan beralih ke lingkungan baru di platform yang berbeda.

Langkah 1 - Buat aplikasi

Untuk membuat aplikasi contoh Anda, Anda akan menggunakan wizard Buat konsol aplikasi. Ini membuat aplikasi Elastic Beanstalk dan meluncurkan lingkungan di dalamnya.

Pengingat: lingkungan adalah kumpulan AWS sumber daya yang diperlukan untuk menjalankan kode aplikasi Anda.



Untuk membuat aplikasi

1. Buka Konsol Elastic Beanstalk.
2. Pilih Create application (Buat aplikasi).
3. Untuk nama Aplikasi masukkan**`getting-started-app`**.

Konsol menyediakan proses enam langkah untuk membuat aplikasi dan mengkonfigurasi lingkungan. Untuk memulai cepat ini, Anda hanya perlu fokus pada dua langkah pertama, lalu Anda dapat melompat ke depan untuk meninjau dan membuat aplikasi dan lingkungan Anda.

Untuk mengkonfigurasi lingkungan

1. Dalam informasi Lingkungan, untuk nama Lingkungan masukkan:**`gs-app-web-env`**.
2. Untuk Platform, pilih platform PHP.
3. Untuk kode Aplikasi dan Preset, terima default (Contoh aplikasi dan contoh Tunggal), lalu pilih Berikutnya.

Untuk mengkonfigurasi akses layanan

Selanjutnya, Anda membutuhkan dua peran. Peran layanan memungkinkan Elastic Beanstalk EC2 memantau instans Anda dan meningkatkan platform lingkungan Anda. Peran profil EC2 instance memungkinkan tugas-tugas seperti menulis log dan berinteraksi dengan layanan lain.

Untuk membuat peran Layanan

1. Untuk peran Layanan, pilih Buat peran.
2. Untuk jenis entitas Tepercaya, pilih AWS layanan.

3. Untuk kasus Penggunaan, pilih Elastic Beanstalk — Environment.
4. Pilih Berikutnya.
5. Verifikasi bahwa kebijakan Izin menyertakan yang berikut ini, lalu pilih Berikutnya:
   - `AWSElasticBeanstalkEnhancedHealth`
   - `AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy`
6. Pilih Buat peran.
7. Kembali ke tab Konfigurasi akses layanan, segarkan daftar, lalu pilih peran layanan yang baru dibuat.

Untuk membuat profil EC2 instance

1. Pilih Buat peran.
2. Untuk jenis entitas Tepercaya, pilih AWS layanan.
3. Untuk kasus Penggunaan, pilih Elastic Beanstalk — Compute.
4. Pilih Berikutnya.
5. Verifikasi bahwa kebijakan Izin menyertakan yang berikut ini, lalu pilih Berikutnya:
   - `AWSElasticBeanstalkWebTier`
   - `AWSElasticBeanstalkWorkerTier`
   - `AWSElasticBeanstalkMulticontainerDocker`
6. Pilih Buat peran.
7. Kembali ke tab Konfigurasi akses layanan, segarkan daftar, lalu pilih profil EC2 instance yang baru dibuat.

Untuk menyelesaikan konfigurasi dan membuat aplikasi Anda

1. Lewati EC2 key pair.

   Kami akan menunjukkan cara lain untuk terhubung ke EC2 instans Amazon Anda melalui Konsol.
2. Pilih Skip to Review untuk memindahkan beberapa langkah opsional.

   Langkah-langkah opsional: jaringan, database, parameter penskalaan, konfigurasi lanjutan untuk pembaruan, pemantauan, dan pencatatan.

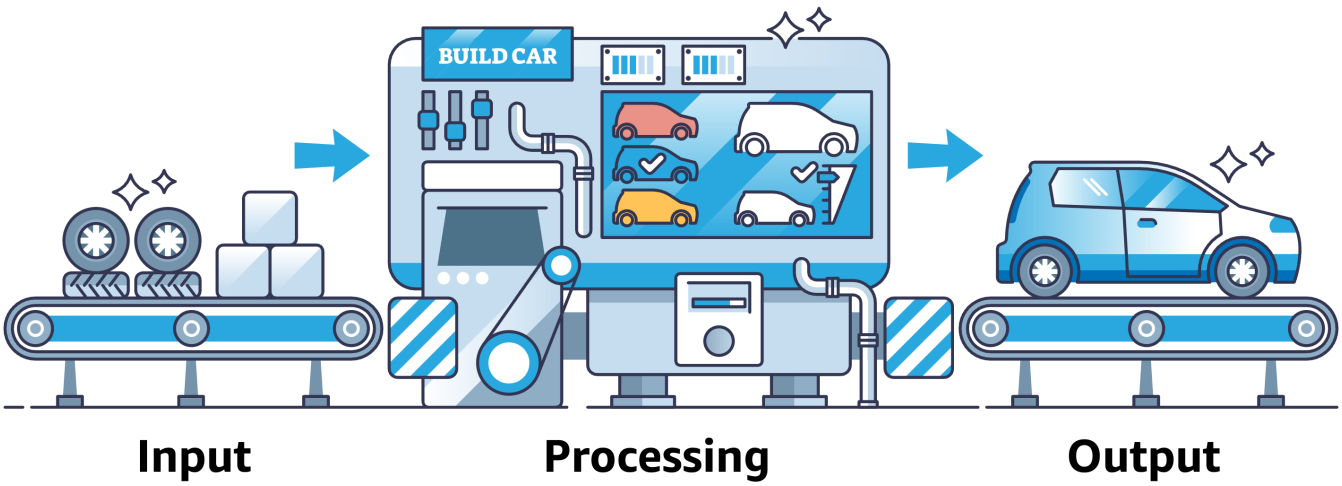
3. Pada halaman Ulasan yang menampilkan ringkasan pilihan Anda, pilih Kirim.

> **Selamat!**
>
> Anda telah membuat aplikasi dan mengkonfigurasi lingkungan! Sekarang Anda harus menunggu sumber daya untuk digunakan.

Langkah 2 - Menyebarkan aplikasi Anda

Saat Anda membuat aplikasi, Elastic Beanstalk mengatur lingkungan untuk Anda. Anda hanya perlu duduk dan menunggu.

Penyebaran awal dapat memakan waktu hingga lima menit untuk membuat sumber daya. Pembaruan akan memakan waktu lebih sedikit karena hanya perubahan yang akan diterapkan ke tumpukan Anda.



Ketika Anda membuat aplikasi contoh, Elastic Beanstalk membuat sumber daya berikut:

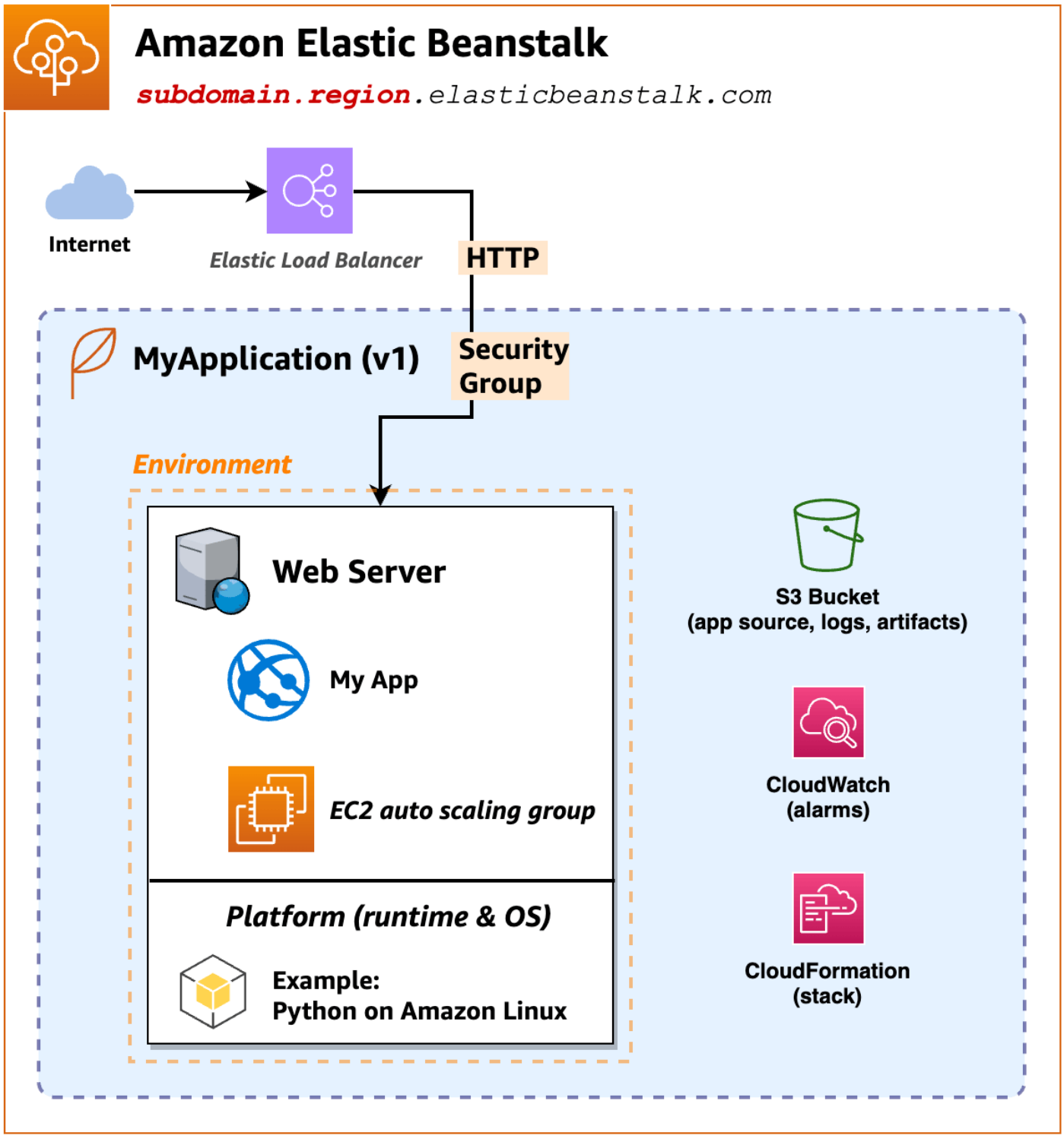
- EC2 instance — Mesin EC2 virtual Amazon yang dikonfigurasi untuk menjalankan aplikasi web pada platform yang Anda pilih.

  Setiap platform menjalankan serangkaian perangkat lunak, file konfigurasi, dan skrip yang berbeda untuk mendukung versi bahasa tertentu, kerangka kerja, wadah web, atau kombinasinya. Sebagian besar platform menggunakan Apache atau nginx sebagai proxy terbalik untuk meneruskan

lalu lintas web ke aplikasi web Anda, menyajikan aset statis, dan menghasilkan akses dan log kesalahan. Anda dapat terhubung ke EC2 instans Amazon Anda untuk melihat konfigurasi dan log.

- Grup keamanan instans — Grup EC2 keamanan Amazon akan dibuat untuk mengizinkan permintaan masuk pada port 80, sehingga lalu lintas masuk pada penyeimbang beban dapat mencapai aplikasi web Anda.
- Bucket Amazon S3 — Lokasi penyimpanan untuk kode sumber, log, dan artefak lainnya.
- CloudWatch Alarm Amazon — Dua CloudWatch alarm dibuat untuk memantau beban pada instans Anda dan menskalakannya ke atas atau ke bawah sesuai kebutuhan.
- AWS CloudFormation stack - Elastic AWS CloudFormation Beanstalk digunakan untuk menyebarkan sumber daya di lingkungan Anda dan membuat perubahan konfigurasi. Anda dapat melihat template definisi sumber daya di AWS CloudFormation konsol.
- Nama domain — Nama domain yang merutekan ke aplikasi web Anda dalam formulir:*`subdomain`*`.`*`region`*`.elasticbeanstalk.com`.

Elastic Beanstalk membuat aplikasi Anda, meluncurkan lingkungan, membuat versi aplikasi, lalu menyebarkan kode Anda ke lingkungan. Selama proses, konsol melacak kemajuan dan menampilkan status acara di tab Acara.



Setelah semua sumber daya dikerahkan, kesehatan lingkungan harus berubah menjadi**Ok**.

Aplikasi Anda sudah siap!

Setelah Anda melihat perubahan kesehatan aplikasi**Ok**, Anda dapat menelusuri situs web aplikasi web Anda.

Langkah 3 - Jelajahi lingkungan Elastic Beanstalk

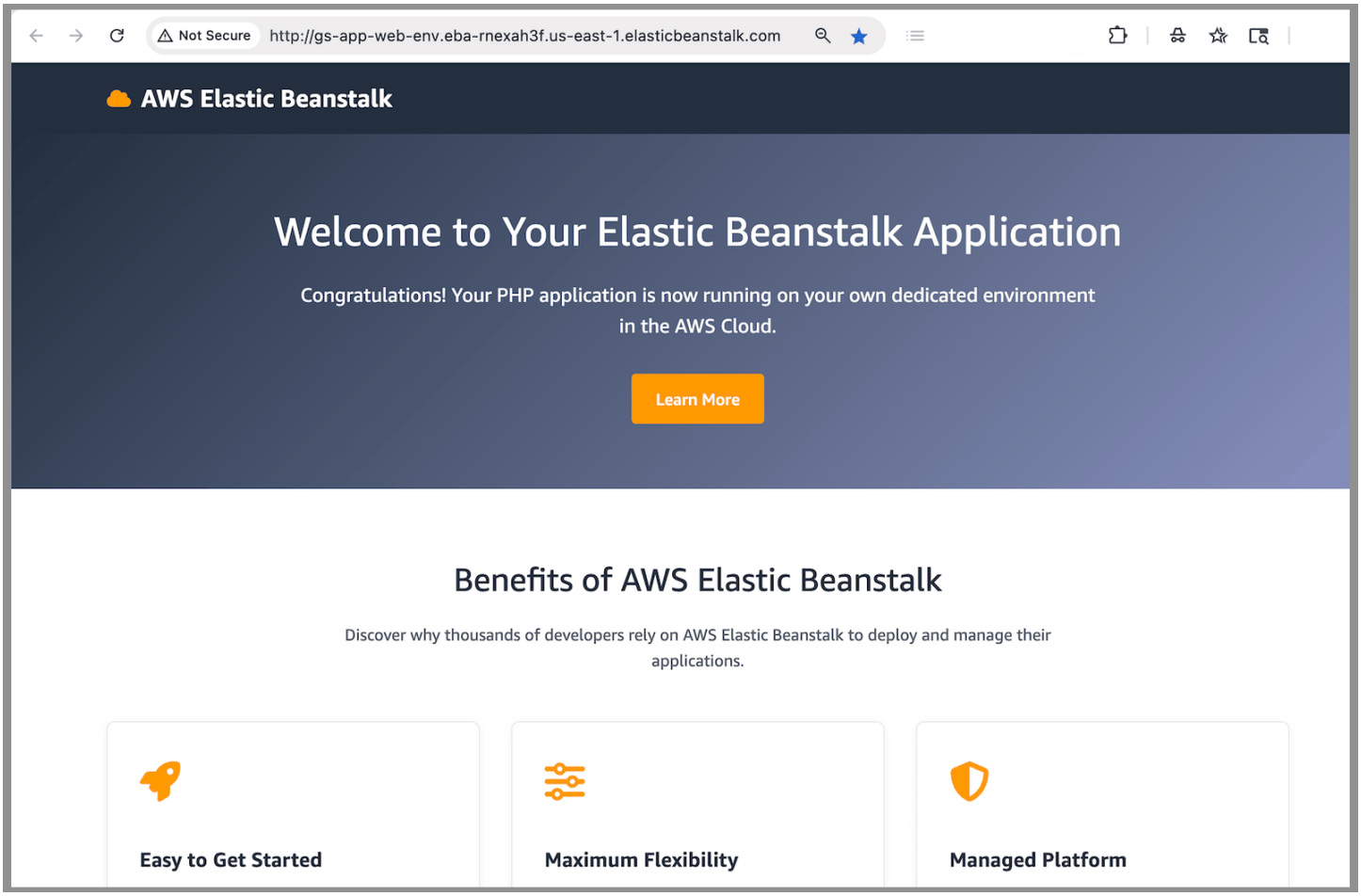
Anda akan mulai menjelajahi lingkungan aplikasi yang diterapkan dari halaman ikhtisar Lingkungan di konsol.

Untuk melihat lingkungan dan aplikasi Anda

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pilih Pergi ke lingkungan untuk menelusuri aplikasi Anda!

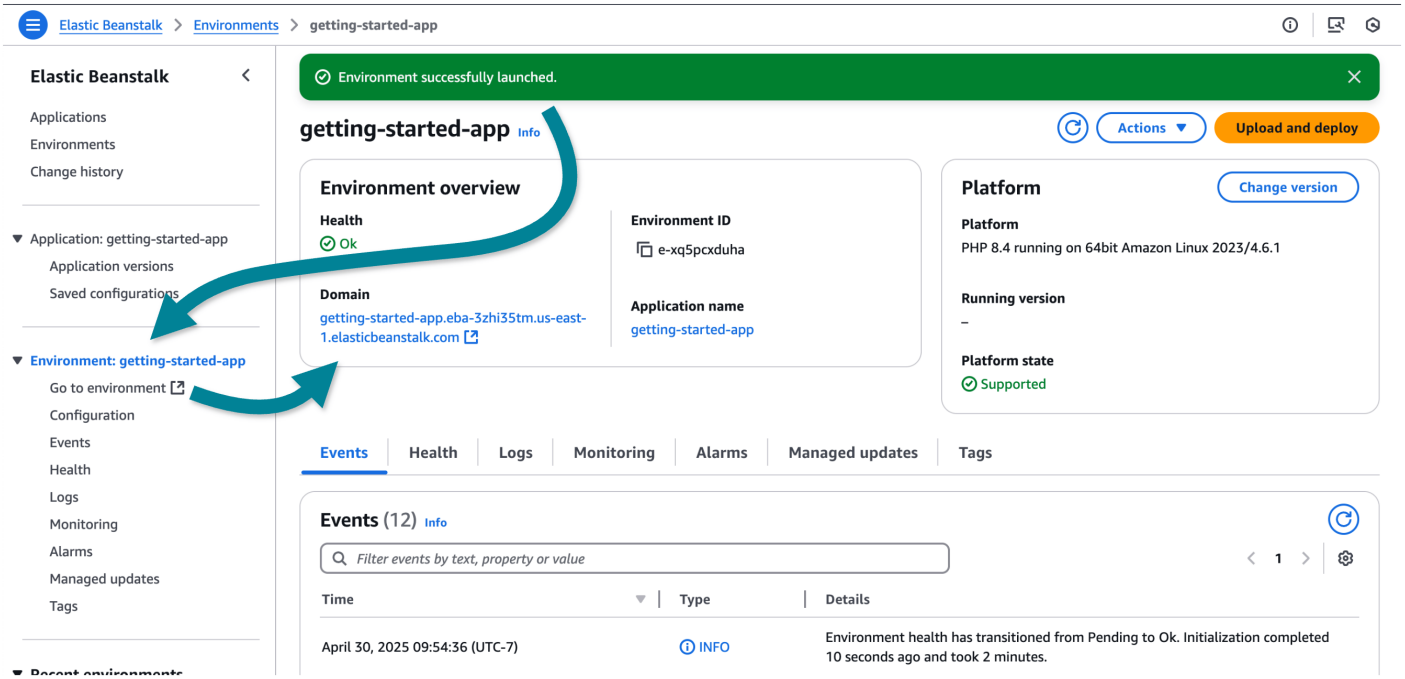
   (Anda juga dapat memilih tautan URL yang terdaftar untuk Domain untuk menelusuri aplikasi Anda.)

   Koneksi akan HTTP (bukan HTTPS), jadi Anda mungkin melihat peringatan di browser Anda.



Kembali ke konsol Elastic Beanstalk, bagian atas menampilkan ikhtisar Lingkungan dengan informasi tingkat atas tentang lingkungan Anda, termasuk nama, URL domain, status kesehatan saat ini, versi berjalan, dan platform tempat aplikasi berjalan. Versi dan platform yang sedang berjalan sangat penting untuk memecahkan masalah aplikasi yang Anda gunakan saat ini.

Setelah panel ikhtisar, Anda akan melihat aktivitas lingkungan terbaru di tab Acara.



Sementara Elastic Beanstalk AWS menciptakan sumber daya Anda dan meluncurkan aplikasi Anda, lingkungan dalam keadaan. `Pending` Pesan status tentang acara peluncuran terus ditambahkan ke daftar Acara.

Domain lingkungan adalah URL untuk aplikasi web yang Anda gunakan. Di panel navigasi kiri, Go to environment juga membawa Anda ke domain Anda. Demikian pula, panel navigasi kiri memiliki tautan yang sesuai dengan berbagai tab.

Perhatikan tautan Konfigurasi di panel navigasi kiri. yang menampilkan ringkasan nilai opsi konfigurasi lingkungan, dikelompokkan berdasarkan kategori.

> **Pengaturan konfigurasi lingkungan**
>
> Perhatikan tautan Konfigurasi di panel navigasi kiri. Anda dapat melihat dan mengedit pengaturan lingkungan terperinci, seperti peran layanan, jaringan, database, penskalaan, pembaruan platform terkelola, memori, pemantauan kesehatan, penyebaran bergulir, pencatatan, dan banyak lagi!

Berbagai tab berisi informasi rinci tentang lingkungan Anda:

- Acara - Lihat daftar informasi dan pesan kesalahan yang diperbarui dari layanan Elastic Beanstalk dan layanan lainnya untuk sumber daya di lingkungan Anda.
- Kesehatan - Lihat status dan informasi kesehatan terperinci untuk EC2 instans Amazon yang menjalankan aplikasi Anda.
- Log - Ambil dan unduh log dari Amazon EC2 di lingkungan Anda. Anda dapat mengambil log lengkap atau aktivitas terbaru. Log yang diambil tersedia selama 15 menit.
- Monitoring - Lihat statistik untuk lingkungan, seperti latensi rata-rata dan pemanfaatan CPU.
- Alarm — Lihat dan edit alarm yang dikonfigurasi untuk metrik lingkungan.
- Pembaruan terkelola - Lihat informasi tentang pembaruan platform terkelola yang akan datang dan selesai serta penggantian instans.
- Tag — Lihat dan edit pasangan kunci-nilai yang diterapkan ke lingkungan Anda.

> **Note**
>
> Tautan di panel navigasi konsol akan menampilkan tab yang sesuai.

Pemecahan masalah dengan catatan

Untuk memecahkan masalah perilaku tak terduga atau men-debug penerapan, Anda mungkin ingin memeriksa log di lingkungan Anda.

Anda dapat meminta 100 baris dari semua file log di bawah tab Log di konsol Elastic Beanstalk. Atau, Anda dapat terhubung langsung ke EC2 instans Amazon dan mengekor log secara realtime.

Untuk meminta log (konsol Elastic Beanstalk)

1. Arahkan ke lingkungan Anda di konsol Elastic Beanstalk.
2. Pilih tab Log atau navigasi kiri, lalu pilih Minta log.
3. Pilih 100 baris terakhir.
4. Setelah log dibuat, pilih tautan Unduh untuk melihat log di browser.

Di log, temukan log dan catat direktori untuk log akses nginx.

Menambahkan kebijakan untuk mengaktifkan koneksi ke Amazon EC2

Sebelum dapat terhubung, Anda harus menambahkan kebijakan yang memungkinkan koneksi ke Amazon EC2 dengan Session Manager.

1. Arahkan ke konsol IAM.
2. Temukan dan pilih **`aws-elasticbeanstalk-ec2-role`** peran.
3. Pilih Tambahkan izin, lalu Lampirkan kebijakan.
4. Cari kebijakan default yang dimulai dengan teks berikut:**`AmazonSSMManagedEC2Instance`**, lalu tambahkan ke peran.

Untuk terhubung ke Amazon Anda EC2 dengan Session Manager

1. Arahkan ke EC2 konsol Amazon.
2. Pilih Instans, lalu pilih `gs-app-web-env` instans Anda.
3. Pilih Connect, lalu Session Manager.
4. Pilih Hubungkan.

Setelah menghubungkan ke instance, mulai bash shell dan ekor log:

1. Jalankan perintah **`bash`**.
2. Jalankan perintah **`cd /var/log/nginx`**.
3. Jalankan perintah **`tail -f access.log`**.
4. Di browser Anda, buka URL domain aplikasi. Segarkan.

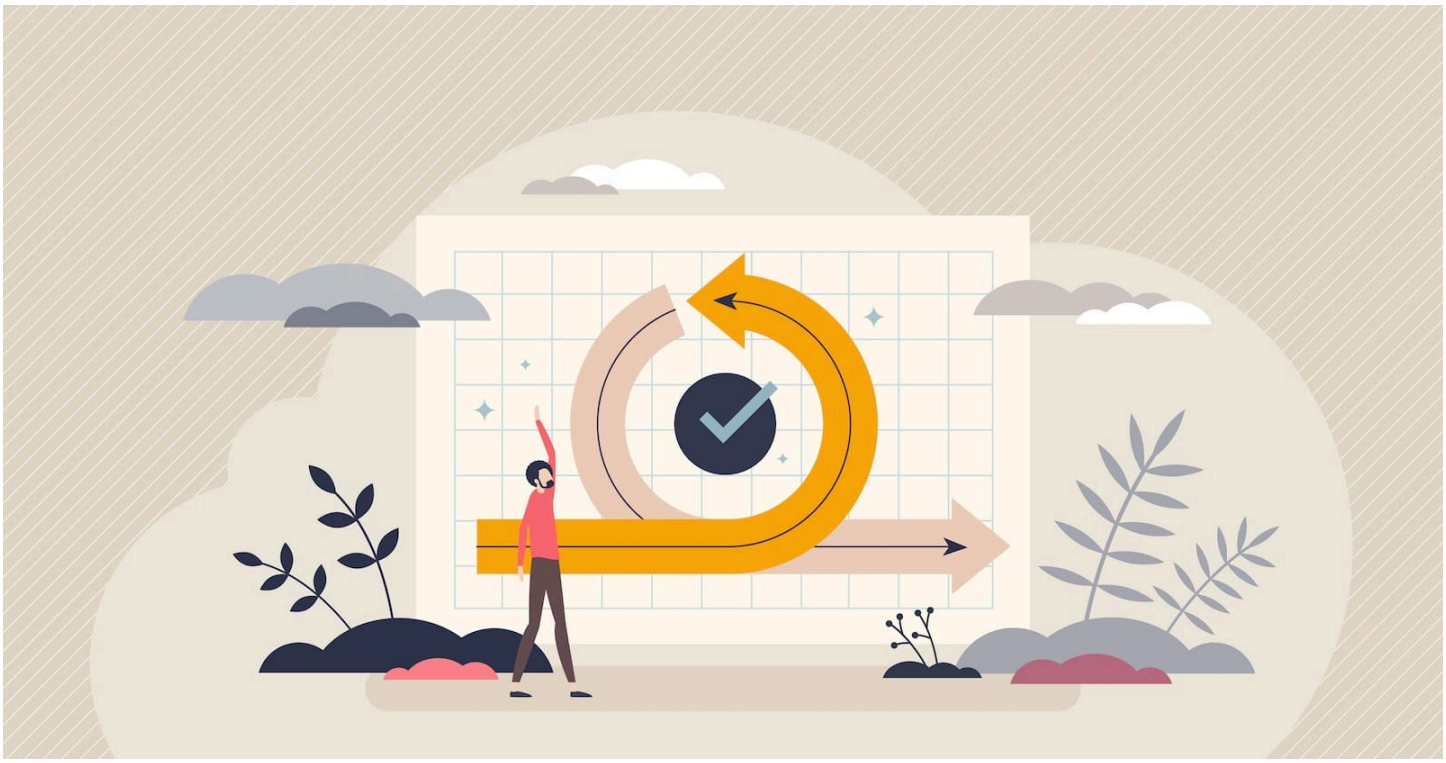
> Selamat, Anda terhubung!
>
> Anda akan melihat entri log dalam pembaruan instans Anda setiap kali Anda me-refresh halaman.

> Tombol Connect tidak berfungsi?
>
> Jika tombol sambungkan tidak tersedia, kembali ke IAM dan verifikasi bahwa Anda menambahkan kebijakan yang diperlukan ke peran tersebut.

Langkah 4 - Perbarui aplikasi Anda

Akhirnya, Anda akan ingin memperbarui aplikasi Anda. Anda dapat men-deploy versi baru setiap saat, selama tidak ada operasi pembaruan lain yang sedang berlangsung di lingkungan Anda.

Versi aplikasi yang Anda mulai tutorialnya disebut Aplikasi Sampel.



Untuk memperbarui versi aplikasi Anda

1. Unduh aplikasi contoh PHP berikut:

   PHP - php-v2.zip
2. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
3. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
4. Di halaman gambaran umum lingkungan, pilih Unggah dan deploy.
5. Pilih Pilih file, lalu unggah contoh bundel sumber aplikasi yang Anda unduh.

   Konsol secara otomatis mengisi label Versi dengan label unik baru, secara otomatis menambah bilangan bulat tambahan. Jika Anda memilih label versi Anda sendiri, pastikan itu unik.
6. Pilih Deploy.

Sementara Elastic Beanstalk menyebarkan file Anda ke instans Amazon, EC2 Anda dapat melihat status penerapan di halaman ikhtisar Lingkungan. Sementara versi aplikasi diperbarui, status Kesehatan lingkungan berwarna abu-abu. Setelah deployment selesai, Elastic Beanstalk melakukan pemeriksaan kondisi aplikasi. Aplikasi dianggap sehat ketika merespons pemeriksaan kondisi dan status kembali ke hijau. Gambaran umum lingkungan menunjukkan Versi Berjalan—nama yang Anda berikan sebagai Label versi.

Elastic Beanstalk juga mengunggah versi aplikasi baru Anda dan menambahkannya ke tabel versi aplikasi. Untuk melihat tabel, pilih Versi aplikasi di bawah getting-started-apppada panel navigasi.

Perbarui kesuksesan!

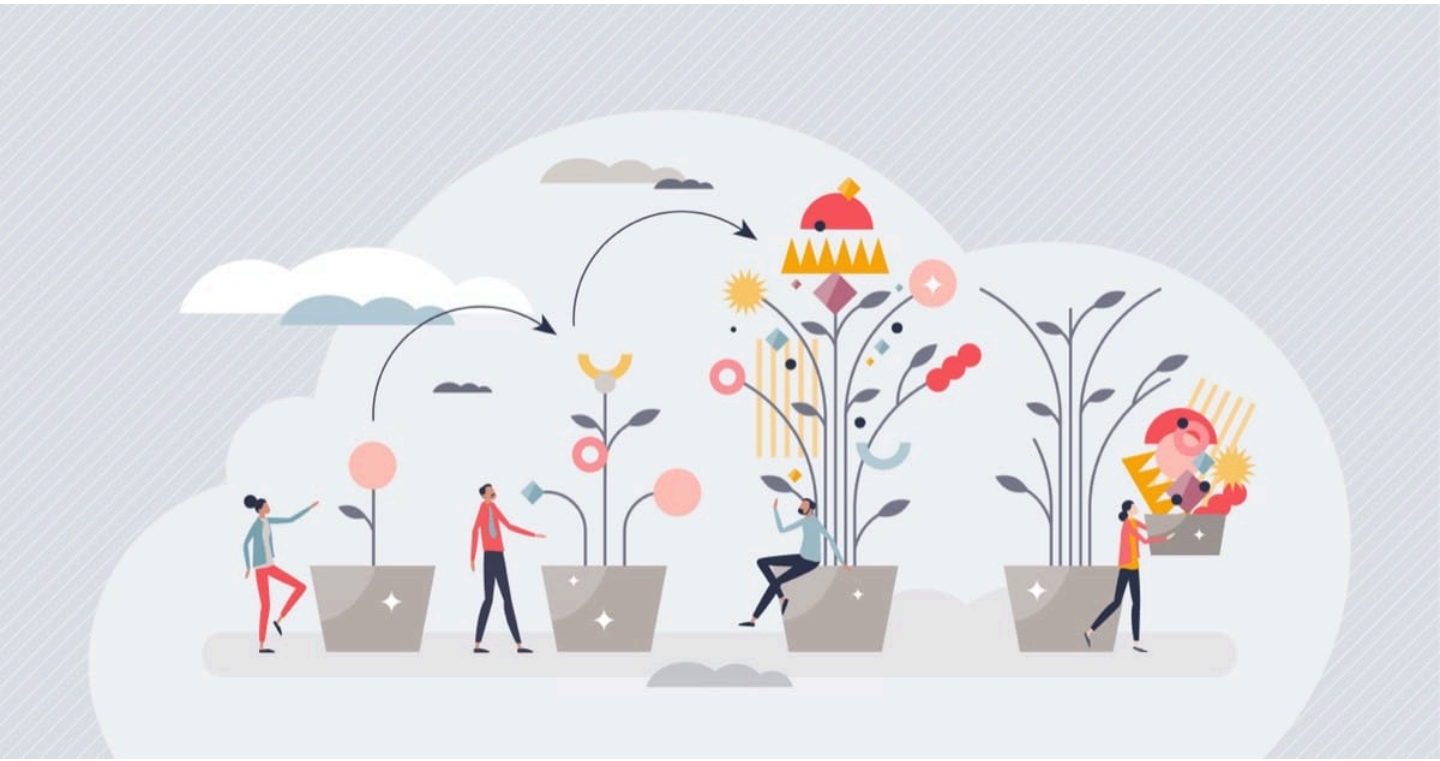
Anda akan melihat pesan “v2" yang diperbarui setelah menyegarkan browser Anda.
Jika Anda ingin mengedit sumbernya sendiri, unzip, edit, lalu zip ulang bundel sumber.
Di macOS, gunakan perintah berikut dari dalam direktori php Anda dengan -X untuk mengecualikan atribut file tambahan:

```
zip -X -r ../php-v2.zip .
```

Langkah 5 - Skala aplikasi Anda

Anda dapat mengonfigurasi lingkungan Anda agar lebih sesuai dengan aplikasi Anda. Misalnya, jika Anda memiliki aplikasi komputasi intensif, Anda dapat mengubah jenis instance Amazon Elastic Compute Cloud (Amazon EC2) yang menjalankan aplikasi Anda. Untuk menerapkan perubahan konfigurasi, Elastic Beanstalk melakukan pembaruan lingkungan.

Beberapa perubahan konfigurasi itu sederhana dan terjadi dengan cepat. Beberapa perubahan memerlukan penghapusan dan pembuatan ulang AWS sumber daya, yang dapat memakan waktu beberapa menit. Ketika Anda mengubah pengaturan konfigurasi, Elastic Beanstalk memperingatkan Anda tentang potensi downtime aplikasi.



Tingkatkan pengaturan kapasitas

Di contoh perubahan konfigurasi ini, Anda mengedit pengaturan kapasitas lingkungan Anda. Anda mengonfigurasi lingkungan yang seimbang beban dan dapat diskalakan yang memiliki antara dua dan empat EC2 instans Amazon dalam grup Auto Scaling, lalu Anda memverifikasi bahwa perubahan itu terjadi. Elastic Beanstalk membuat instance Amazon EC2 tambahan, menambah instance tunggal yang dibuat pada awalnya. Kemudian, Elastic Beanstalk mengaitkan kedua instans dengan penyeimbang beban lingkungan. Akibatnya, responsivitas aplikasi Anda dan ketersediaannya meningkat.

Untuk mengubah kapasitas lingkungan Anda

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Dalam kategori konfigurasi lalu lintas Instans dan penskalaan, pilih Edit.
5. Tutup bagian Instans, sehingga Anda dapat lebih mudah melihat bagian Kapasitas. Di bawah grup Auto Scaling, ubah jenis Lingkungan menjadi Beban seimbang.
6. Di baris Instances, ubah Min menjadi **2** dan Max menjadi**4**.

7. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

   Jika Anda diperingatkan bahwa pembaruan akan menggantikan semua instans Anda saat ini. Pilih Konfirmasi.

Pembaruan lingkungan memerlukan waktu beberapa menit. Anda akan melihat beberapa pembaruan dalam daftar acara. Perhatikan acara Berhasil menerapkan konfigurasi baru ke lingkungan.

Verifikasi peningkatan kapasitas

Setelah pembaruan lingkungan selesai dan lingkungan siap, Elastic Beanstalk secara otomatis meluncurkan instance kedua untuk memenuhi pengaturan kapasitas minimum baru Anda.

Untuk memverifikasi peningkatan kapasitas

1. Pilih Health dari daftar tab atau panel navigasi kiri.
2. Tinjau bagian kesehatan instans yang Ditingkatkan.

> Anda baru saja ditingkatkan!
>
> Dengan dua EC2 instans Amazon, kapasitas lingkungan Anda meningkat dua kali lipat, dan hanya butuh beberapa menit.

Membersihkan lingkungan Elastic Beanstalk Anda

Untuk memastikan bahwa Anda tidak dikenakan biaya untuk layanan apa pun yang tidak Anda gunakan, hapus semua versi aplikasi dan hentikan lingkungan, yang juga menghapus AWS sumber daya yang dibuat lingkungan untuk Anda.



Untuk menghapus aplikasi dan semua sumber daya terkait

1. Hapus semua versi aplikasi.
    a. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
    b. Di panel navigasi, pilih Aplikasi, lalu pilih getting-started-app.
    c. Di panel navigasi, temukan nama aplikasi Anda dan pilih Versi aplikasi.
    d. Di halaman Versi aplikasi, pilih semua versi aplikasi yang ingin Anda hapus.
    e. Pilih Tindakan, dan kemudian pilih Hapus.
    f. Nyalakan Hapus versi dari Amazon S3.
    g. Pilih Hapus, dan kemudian pilih Selesai.
2. Akhiri lingkungan.

a. Di panel navigasi, pilih getting-started-app, lalu pilih GettingStartedApp-env di daftar lingkungan.
b. Pilih Tindakan, lalu pilih Hentikan Lingkungan.
c. Konfirmasikan bahwa Anda ingin mengakhiri GettingStartedApp-env dengan mengetikkan nama lingkungan, lalu pilih Terminate.

3. Hapus getting-started-app aplikasi.

a. Di panel navigasi, pilih. getting-started-app
b. Pilih Tindakan, dan kemudian pilih Hapus aplikasi.
c. Konfirmasikan bahwa Anda ingin menghapus getting-started-appdengan mengetikkan nama aplikasi, lalu pilih Hapus.

Selamat!

Anda telah berhasil menerapkan contoh aplikasi ke AWS Cloud, mengunggah versi baru, memodifikasi konfigurasinya untuk menambahkan instance Auto Scaling kedua, dan membersihkan sumber daya Anda! AWS

Langkah selanjutnya

Untuk mempelajari cara menggunakan alat baris **eb** perintah untuk mengotomatiskan penerapan kode Anda ke Elastic Beanstalk, Kami sarankan untuk melanjutkan dengan. the section called “QuickStart untuk PHP”

Selanjutnya, Anda mungkin ingin meninjau cara mengatur koneksi HTTPS, lihatthe section called “HTTPS”.

Menyiapkan antarmuka baris perintah EB (EB CLI) untuk mengelola Elastic Beanstalk

EB CLI adalah antarmuka baris perintah yang menyediakan perintah interaktif untuk membuat, memperbarui, dan memantau lingkungan di. AWS Elastic Beanstalk Proyek open-source EB CLI ada di Github: aws/aws-elastic-beanstalk-cli

Setelah Anda mengintal EB CLI dan mengonfigurasi direktori proyek, Anda dapat membuat lingkungan dengan satu perintah:

```
$ eb create <my-beanstalk-environment>
```

Kami merekomendasikan menginstal dengan skrip pengaturan, pelajari caranyathe section called “Instal EB CLI”.

AWS CLI Ini menyediakan akses langsung ke Elastic Beanstalk APIs tingkat rendah. Meskipun kuat, itu juga bertele-tele dan kurang disukai daripada CLI EB. Misalnya, menciptakan lingkungan dengan AWS CLI memerlukan serangkaian perintah berikut:

```
$ aws elasticbeanstalk check-dns-availability \
    --cname-prefix my-cname
$ aws elasticbeanstalk create-application-version \
    --application-name my-application \
    --version-label v1 \
    --source-bundle S3Bucket=amzn-s3-demo-bucket,S3Key=php-proxy-sample.zip
$ aws elasticbeanstalk create-environment \
    --cname-prefix my-cname \
    --application-name my-app \
    --version-label v1 \
    --environment-name my-env \
    --solution-stack-name "64bit Amazon Linux 2023 v4.5.0 running Ruby 3.4"
```

Instal EB CLI dengan skrip pengaturan (disarankan)

> **Kami merekomendasikan skrip penginstal**
>
> Kami merekomendasikan menggunakan skrip installer untuk mengatur CLI EB dan dependensinya dan mencegah potensi konflik dengan paket Python lainnya.

Prasyarat: Git, Python, dan virtualenv

Untuk mengunduh dan menggunakan skrip penginstal

1. Kloning repositori.

   ```
   git clone https://github.com/aws/aws-elastic-beanstalk-cli-setup.git
   ```

2. Instal atau tingkatkan EB CLI.

   macOS/Linux di Bash atau Zsh

   ```
   python ./aws-elastic-beanstalk-cli-setup/scripts/ebcli_installer.py
   ```

   Windows di PowerShell atau jendela Perintah

   ```
   python .\aws-elastic-beanstalk-cli-setup\scripts\ebcli_installer.py
   ```

3. Verifikasi bahwa EB CLI terpasang dengan benar.

   ```
   $ eb --version
   EB CLI 3.21.0 (Python 3.12)
   ```

Untuk petunjuk penginstalan lengkap, lihat aws/aws-elastic-beanstalk-cli-setuprepositori aktif. GitHub

Memasang EB CLI secara manual

Anda dapat menginstal EB CLI di Linux, macOS, dan Windows dengan manajer `pip` paket untuk Python yang menyediakan instalasi, peningkatan, dan penghapusan paket Python dan dependensinya.

> **Kami merekomendasikan skrip penginstal**
>
> Kami merekomendasikan penggunaan Instal EB CLI untuk mengatur CLI EB dan mencegah konflik ketergantungan.

Prasyarat - Anda harus memiliki versi Python yang didukung diinstal. Anda dapat mengunduhnya dari halaman unduhan Python di situs web Python.

Untuk menginstal EB CLI (secara manual)

1. Jalankan perintah berikut.

   ```
   $ pip install awsebcli --upgrade --user
   ```

   Opsi `--upgrade` memberitahu `pip` untuk memperbarui persyaratan yang sudah terpasang. Opsi `--user` memberitahu `pip` untuk memasang program ke subdirektori direktori pengguna Anda untuk menghindari memodifikasi pustaka yang digunakan sistem operasi Anda.

   > **Memecahkan masalah**
   >
   > Jika Anda mengalami masalah saat Anda mencoba untuk memasang EB CLI dengan `pip`, Anda dapat memasang EB CLI di lingkungan virtual untuk mengisolasi alat dan dependensinya, atau menggunakan versi Python yang berbeda dari yang biasanya Anda lakukan.

2. Tambahkan jalur ke file yang dapat dieksekusi ke variabel `PATH`:

   - Di Linux dan macOS:

     Linux – `~/.local/bin`

     macOS – `~/Library/Python/3.12/bin`

Untuk memodifikasi variabel PATH Anda (Linux, Unix, atau macOS):

a. Temukan penulisan profil shell Anda di folder pengguna Anda. Jika Anda tidak yakin shell mana yang Anda miliki, jalankan `echo $SHELL`.

```
$ ls -a ~
.  ..  .bash_logout  .bash_profile  .bashrc  Desktop  Documents  Downloads
```

- Bash – `.bash_profile`, `.profile`, or `.bash_login`.
- Zsh – `.zshrc`
- Tcsh – `.tcshrc`, `.cshrc` or `.login`.

b. Tambahkan perintah ekspor ke penulisan profil Anda. Contoh berikut menambahkan jalur diwakili oleh *`LOCAL_PATH`* PATH variabel saat ini.

```
export PATH=LOCAL_PATH:$PATH
```

c. Muat penulisan profil yang dijelaskan di langkah pertama ke sesi Anda saat ini. Contoh berikut memuat skrip profil diwakili oleh*`PROFILE_SCRIPT`*.

```
$ source ~/PROFILE_SCRIPT
```

- Di Windows:

  Python 3.12 — `%USERPROFILE%\AppData\Roaming\Python\Python312\Scripts`

  Python versi terdahulu – `%USERPROFILE%\AppData\Roaming\Python\Scripts`

  Untuk memodifikasi variabel PATH (Windows):

  a. Tekan kunci Windows, dan kemudian masukkan **`environment variables`**.

  b. Pilih Edit variabel lingkungan untuk akun Anda.

  c. Pilih PATH, lalu pilih Edit.

  d. Tambahkan jalur ke bidang Nilai variabel, yang dipisahkan oleh titik koma. Sebagai contoh: *`C:\item1\path`***`;C:\item2\path`**

  e. Pilih OK dua kali untuk menerapkan pengaturan baru.

  f. Tutup jendela Command Prompt yang berjalan, dan kemudian bukalah kembali jendela tersebut.

3. Verifikasi bahwa EB CLI terpasang dengan benar dengan menjalankan eb --version.

```
$ eb --version
EB CLI 3.21.0 (Python 3.12)
```

EB CLI diperbarui secara teratur untuk menambahkan fungsionalitas yang mendukung fitur Elastic Beanstalk terbaru. Untuk memperbarui ke versi terbaru dari EB CLI, jalankan perintah pemasangan lagi.

```
$ pip install awsebcli --upgrade --user
```

Jika Anda harus menghapus pemasangan EB CLI, gunakan `pip uninstall`.

```
$ pip uninstall awsebcli
```

Pasang EB CLI di lingkungan virtual

Anda dapat menghindari konflik persyaratan versi dengan paket `pip` lainnya dengan memasang EB CLI di lingkungan virtual.

Untuk memasang EB CLI di lingkungan virtual

1. Pertama, instal `virtualenv` dengan`pip`.

```
$ pip install --user virtualenv
```

2. Buat lingkungan virtual.

```
$ virtualenv ~/eb-ve
```

Untuk menggunakan Python yang dapat dieksekusi selain yang default, gunakan opsi `-p`.

```
$ virtualenv -p /usr/bin/python3.12 ~/eb-ve
```

3. Aktifkan lingkungan virtual.

Linux, Unix, atau macOS

```
$ source ~/eb-ve/bin/activate
```

Windows

```
$ %USERPROFILE%\eb-ve\Scripts\activate
```

4. Pasang EB CLI.

```
(eb-ve)$ pip install awsebcli --upgrade
```

5. Verifikasi bahwa EB CLI terpasang dengan benar.

```
$ eb --version
EB CLI 3.23.0 (Python 3.12)
```

Anda dapat menggunakan perintah `deactivate` untuk keluar dari lingkungan virtual. Setiap kali Anda memulai sesi baru, jalankan perintah aktivasi lagi.

Untuk memperbarui ke versi terbaru, jalankan perintah pemasangan lagi.

```
(eb-ve)$ pip install awsebcli --upgrade
```

Memasang EB CLI dengan homebrew

Versi terbaru dari EB CLI biasanya tersedia dari `Homebrew` beberapa hari setelah muncul di `pip`.

> ⚠ Kami merekomendasikan skrip penginstal
>
> Kami merekomendasikan penggunaan Instal EB CLI untuk mengatur CLI EB dan mencegah konflik ketergantungan.

Untuk memasang EB CLI dengan **`Homebrew`**

1. Pastikan Anda memiliki versi terbaru `Homebrew`.

```
$ brew update
```

2. Jalankan `brew install awsebcli`.

```
$ brew install awsebcli
```

3. Verifikasi bahwa EB CLI terpasang dengan benar.

```
$ eb --version
EB CLI 3.21.0 (Python 3.12)
```

Mengonfigurasi EB CLI

Setelah menginstal CLI EB, Anda siap untuk mengkonfigurasi direktori proyek Anda dan CLI EB dengan eb init menjalankan.Contoh berikut menunjukkan langkah-langkah konfigurasi saat eb init berjalan untuk pertama kalinya dalam folder proyek bernama. `eb`

Untuk menginisialisasi proyek EB CLI

1. Pertama, EB CLI meminta Anda untuk memilih wilayah. Pilih wilayah pilihan Anda.

```
~/eb $ eb init
Select a default region
1) us-east-1 : US East (N. Virginia)
2) us-west-1 : US West (N. California)
3) us-west-2 : US West (Oregon)
4) eu-west-1 : Europe (Ireland)
5) eu-central-1 : Europe (Frankfurt)
6) ap-south-1 : Asia Pacific (Mumbai)
7) ap-southeast-1 : Asia Pacific (Singapore)
...
(default is 3): 3
```

2. Jika diminta, berikan kunci akses dan kunci rahasia Anda sehingga CLI EB dapat mengelola sumber daya untuk Anda. Kunci akses dibuat di AWS Identity and Access Management konsol. Jika Anda tidak memiliki kunci, lihat Bagaimana Saya Mendapatkan Kredensi Keamanan? di Referensi Umum Amazon Web.

```
You have not yet set up your credentials or your credentials are incorrect.
You must provide your credentials.
(aws-access-id): AKIAJOUAASEXAMPLE
```

```
(aws-secret-key): 5ZRIrtTM4ciIAvd4EXAMPLEDtm+PiPSzpoK
```

3. Aplikasi Elastic Beanstalk adalah sumber daya yang berisi serangkaian versi aplikasi (sumber), lingkungan, dan konfigurasi tersimpan yang terkait dengan satu aplikasi web. Setiap kali Anda men-deploy kode sumber Anda ke Elastic Beanstalk menggunakan EB CLI, versi aplikasi baru dibuat dan ditambahkan ke daftar.

```
Select an application to use
1) [ Create new Application ]
(default is 1): 1
```

4. Nama aplikasi default adalah nama folder tempat Anda menjalankan eb init. Masukkan nama yang menggambarkan proyek Anda.

```
Enter Application Name
(default is "eb"): eb
Application eb has been created.
```

5. Pilih platform yang cocok dengan bahasa atau kerangka kerja tempat aplikasi web Anda dikembangkan. Jika Anda belum mulai mengembangkan aplikasi, pilih platform yang Anda minati. Anda akan segera melihat cara meluncurkan aplikasi sampel, dan Anda selalu dapat mengubah pengaturan ini nanti.

```
Select a platform.
1) .NET Core on Linux
2) .NET on Windows Server
3) Docker
4) Go
5) Java
6) Node.js
7) PHP         <== select platform by number
8) Packer
9) Python
10) Ruby
11) Tomcat
(make a selection):7
```

6. Pilih cabang platform tertentu.

```
Select a platform branch.
1) PHP 8.4 running on 64bit Amazon Linux 2023
2) PHP 8.3 running on 64bit Amazon Linux 2023
```

```
3) PHP 8.2 running on 64bit Amazon Linux 2023
4) PHP 8.1 running on 64bit Amazon Linux 2023
5) PHP 8.1 running on 64bit Amazon Linux 2
(default is 1):1
```

7. Pilih ya untuk menetapkan pasangan kunci SSH untuk instans di lingkungan Elastic Beanstalk Anda. Ini mengizinkan Anda terhubung langsung ke mereka untuk pemecahan masalah.

```
Do you want to set up SSH for your instances?
(y/n): y
```

8. Anda dapat memilih pasangan kunci yang sudah ada atau membuat yang baru. Untuk menggunakan eb init untuk membuat pasangan kunci baru, Anda harus memasang ssh-keygen di komputer lokal Anda dan tersedia dari baris perintah. EB CLI mendaftarkan key pair baru dengan EC2 Amazon untuk Anda dan menyimpan kunci pribadi secara lokal dalam folder `.ssh` bernama di direktori pengguna Anda.

```
Select a keypair.
1) [ Create new KeyPair ]
(default is 1): 1
```

Pemasangan EB CLI Anda sekarang sudah dikonfigurasi dan siap digunakan.

Konfigurasi Lanjutan

- Mengabaikan file menggunakan .ebignore
- Menggunakan profil bernama
- Men-deploy artifact bukan folder proyek
- Pengaturan konfigurasi dan prioritas
- Metadata instans

Mengabaikan file menggunakan .ebignore

Anda dapat memberitahu EB CLI untuk mengabaikan file tertentu di direktori proyek Anda dengan menambahkan file `.ebignore` ke direktori. File ini bekerja seperti file `.gitignore`. Ketika Anda men-deploy direktori proyek Anda ke Elastic Beanstalk dan membuat versi aplikasi baru, EB CLI tidak menyertakan file yang ditentukan oleh `.ebignore` di paket sumber yang dibuatnya.

Jika `.ebignore` tidak ada, tetapi ada `.gitignore`, EB CLI mengabaikan file yang ditentukan di `.gitignore`. Jika `.ebignore` ada, EB CLI tidak membaca `.gitignore`.

Saat `.ebignore` ada, EB CLI tidak menggunakan perintah git untuk membuat paket sumber Anda. Ini berarti bahwa EB CLI mengabaikan file yang ditentukan di `.ebignore`, dan menyertakan semua file lainnya. Khususnya, ia termasuk file sumber yang tidak terikat.

> **Note**
>
> Di Windows, menambahkan `.ebignore` menyebabkan EB CLI mengikuti tautan simbolik dan menyertakan file bertautan saat membuat paket sumber. Ini adalah masalah yang diketahui dan akan diperbaiki di pembaruan masa mendatang.

Menggunakan profil bernama

Jika Anda menyimpan kredensial Anda sebagai profil bernama di file `credentials` atau `config`, Anda dapat menggunakan opsi `--profile` untuk secara eksplisit menentukan profil. Sebagai contoh, perintah berikut membuat aplikasi baru menggunakan profil `user2`.

```
$ eb init --profile user2
```

Anda juga dapat mengubah profil default dengan menetapkan variabel lingkungan `AWS_EB_PROFILE`. Ketika variabel ini diatur, EB CLI membaca kredensial dari profil yang ditentukan, bukan dari `default` atau eb-cli.

Linux, macOS, atau Unix

```
$ export AWS_EB_PROFILE=user2
```

Windows

```
> set AWS_EB_PROFILE=user2
```

Men-deploy artifact bukan folder proyek

Anda dapat memberitahu EB CLI untuk men-deploy file ZIP atau file WAR yang Anda hasilkan sebagai bagian dari proses pembangunan terpisah dengan menambahkan baris berikut ke `.elasticbeanstalk/config.yml` di folder proyek Anda.

```
deploy:
  artifact: path/to/buildartifact.zip
```

Jika Anda mengonfigurasi EB CLI di repositori git, dan anda tidak melakukan artifact ke sumber, gunakan opsi `--staged` untuk men-deploy pembangunan terbaru.

```
~/eb$ eb deploy --staged
```

Pengaturan konfigurasi dan prioritas

EB CLI menggunakan rantai penyedia AWS untuk mencari kredensi di sejumlah tempat berbeda, termasuk variabel sistem atau lingkungan pengguna dan file konfigurasi lokal. AWS

EB CLI mencari kredensial dan pengaturan konfigurasi dalam urutan berikut:

1. Opsi baris perintah – Tentukan profil bernama dengan menggunakan `--profile` untuk mengganti pengaturan default.
2. Variabel lingkungan – `AWS_ACCESS_KEY_ID` dan `AWS_SECRET_ACCESS_KEY`.
3. File AWS kredensial — Terletak `~/.aws/credentials` di sistem Linux dan OS X, atau `C:\Users\`*`USERNAME`*`\.aws\credentials` di sistem Windows. File ini dapat berisi beberapa profil bernama selain profil default.
4. File konfigurasi AWS CLI – Terletak pada `~/.aws/config` di sistem Linux dan OS X atau pada `C:\Users\`*`USERNAME`*`\.aws\config` di sistem Windows. File ini dapat berisi profil default, profil bernama, dan AWS CLI parameter konfigurasi spesifik untuk masing-masing.
5. File konfigurasi EB CLI warisan – Terletak pada `~/.elasticbeanstalk/config` di sistem Linux dan OS X atau pada `C:\Users\`*`USERNAME`*`\.elasticbeanstalk\config` di sistem Windows.
6. Kredensial profil instans — Kredensional ini dapat digunakan di EC2 instans Amazon dengan peran instans yang ditetapkan, dan dikirimkan melalui layanan metadata Amazon. EC2 Profil instans harus memiliki izin untuk menggunakan Elastic Beanstalk.

Jika file kredensial berisi profil bernama dengan nama "eb-cli", EB CLI akan lebih memilih profil tersebut daripada profil default. Jika profil tidak ditemukan, atau profil ditemukan tetapi tidak memiliki izin untuk menggunakan Elastic Beanstalk, EB CLI meminta Anda untuk memasukkan kunci.

Metadata instans

Untuk menggunakan CLI EB dari instans EC2 Amazon, buat peran yang memiliki akses ke sumber daya yang dibutuhkan dan tetapkan peran tersebut ke instance saat diluncurkan. Peluncuran instans dan memasang EB CLI dengan menggunakan `pip`.

```
~$ sudo pip install awsebcli
```

`pip` muncul dalam kondisi sudah terpasang di Amazon Linux.

EB CLI membaca kredensial dari metadata instans. Untuk informasi selengkapnya, lihat Memberikan Aplikasi yang Berjalan di EC2 Instans Amazon Akses ke AWS Sumber Daya di Panduan Pengguna IAM.

Menggunakan EB CLI dengan Git

EB CLI menyediakan integrasi Git sehingga Anda dapat mengaitkan cabang kode dengan lingkungan tertentu di Elastic Beanstalk untuk mengatur penyebaran Anda.

Untuk memasang Git dan menginisialisasi repositori Git Anda

1. Unduh versi terbaru dari Git dengan mengunjungi http://git-scm.com.
2. Inisialisasi repositori Git Anda dengan mengetik berikut ini:

   ```
   ~/eb$ git init
   ```

   EB CLI sekarang akan mengenali bahwa aplikasi Anda diatur dengan Git.
3. Jika Anda belum menjalankan eb init, sekarang lakukan:

   ```
   ~/eb$ eb init
   ```

Mengaitkan lingkungan Elastic Beanstalk dengan cabang Git

Anda dapat mengaitkan lingkungan yang berbeda dengan setiap cabang kode Anda. Ketika Anda checkout cabang, perubahan di-deploy ke lingkungan terkait. Misalnya, Anda dapat mengetik berikut ini untuk mengaitkan lingkungan produksi Anda dengan cabang utama Anda, dan lingkungan pengembangan terpisah dengan cabang pengembangan Anda:

```
~/eb$ git checkout mainline
~/eb$ eb use prod
~/eb$ git checkout develop
~/eb$ eb use dev
```

Men-deploy perubahan

Secara default, EB CLI men-deploy komitmen terbaru di cabang saat ini, menggunakan commit ID dan pesan masing-masing sebagai label versi aplikasi dan deskripsi. Jika Anda ingin men-deploy ke lingkungan Anda tanpa melakukan, Anda dapat menggunakan opsi `--staged` untuk men-deploy perubahan yang telah ditambahkan ke area persiapan.

Untuk men-deploy perubahan tanpa melakukan

1. Tambahkan file yang baru dan diubah ke area persiapan:

   ```
   ~/eb$ git add .
   ```

2. Men-deploy perubahan persiapan dengan eb deploy:

   ```
   ~/eb$ eb deploy --staged
   ```

Jika Anda mengonfigurasi EB CLI untuk men-deploy Artifact, dan Anda tidak melakukan Artifact ke repositori git Anda, gunakan opsi `--staged` untuk men-deploy pembangunan terbaru.

Menggunakan submodul Git

Beberapa proyek kode mendapatkan manfaat dari memiliki submodule Git — repositori dalam repositori tingkat atas. Ketika Anda men-deploy kode Anda menggunakan eb create atau eb deploy, EB CLI dapat menyertakan submodul di file zip versi aplikasi dan mengunggahnya dengan sisa kode.

Anda dapat mengontrol masuknya submodul dengan menggunakan opsi `include_git_submodules` di bagian `global` dari file konfigurasi EB CLI, `.elasticbeanstalk/config.yml` di folder proyek Anda.

Untuk menyertakan submodul, atur opsi ini ke `true`:

```
global:
```

```
  include_git_submodules: true
```

Saat opsi `include_git_submodules` hilang atau diatur ke `false`, EB CLI tidak menyertakan submodul di file zip yang diunggah.

Lihat Alat Git - Submodul untuk detail selengkapnya tentang submodule Git.

> **Perilaku default**
>
> Ketika Anda menjalankan eb init untuk mengonfigurasi proyek Anda, EB CLI menambahkan opsi `include_git_submodules` dan mengaturnya ke `true`. Hal ini memastikan bahwa setiap submodul yang Anda miliki di proyek Anda disertakan di deployment Anda.
> EB CLI tidak selalu mendukung, termasuk submodul. Untuk menghindari perubahan yang tidak disengaja dan tidak diinginkan pada proyek yang telah ada sebelum kami menambahkan dukungan submodul, EB CLI tidak menyertakan submodul ketika opsi `include_git_submodules` hilang. Jika Anda memiliki salah satu proyek yang ada dan Anda ingin menyertakan submodul di deployment Anda, tambahkan opsi dan atur ke `true` seperti yang dijelaskan di bagian ini.

> **CodeCommit perilaku**
>
> Integrasi Elastic Beanstalk dengan CodeCommit tidak mendukung submodul saat ini. Jika Anda mengaktifkan lingkungan untuk diintegrasikan CodeCommit, submodul tidak disertakan dalam penerapan Anda.

Menetapkan tanda Git ke versi aplikasi Anda

Anda dapat menggunakan tanda Git sebagai label versi Anda untuk mengidentifikasi versi aplikasi apa yang berjalan di lingkungan Anda. Misalnya, lihat yang berikut ini:

```
~/eb$ git tag -a v1.0 -m "My version 1.0"
```

Referensi perintah EB CLI

Anda dapat menggunakan antarmuka baris perintah Elastic Beanstalk (EB CLI) untuk melakukan berbagai operasi untuk men-deploy dan mengelola aplikasi Elastic Beanstalk dan lingkungan. EB CLI terintegrasi dengan Git jika Anda ingin men-deploy kode sumber aplikasi yang berada di bawah kontrol sumber Git. Untuk informasi selengkapnya, lihat Menyiapkan antarmuka baris perintah EB (EB CLI) untuk mengelola Elastic Beanstalk dan Menggunakan EB CLI dengan Git.

Commands

- Opsi umum
- eb abort
- eb appversion
- eb clone
- eb codesource
- eb config
- eb console
- eb create
- eb deploy
- eb events
- eb health
- eb init
- eb labs
- eb list
- eb local
- eb logs
- eb migrate
- eb open
- eb platform
- eb printenv
- eb restore

- eb scale
- eb setenv
- eb ssh
- eb status
- eb swap
- eb tags
- eb terminate
- eb upgrade
- eb use

Opsi umum

Anda dapat menggunakan opsi berikut dengan semua perintah EB CLI.

Nama	Deskripsi
`--debug`	Cetak informasi untuk debugging.
`-h`, `--help`	Tampilkan pesan Bantuan. Tipe: String Default: Tidak ada
`--no-verify-ssl`	Lewati verifikasi sertifikat SSL. Gunakan opsi ini jika Anda memiliki masalah menggunakan CLI dengan proksi.
`--profile`	Gunakan profil tertentu dari file AWS kredensi Anda.
`--quiet`	Tekan semua output dari perintah.
`--region`	Gunakan wilayah yang ditentukan.
`-v`, `--verbose`	Tampilkan informasi verbose.

eb abort

Deskripsi

Membatalkan peningkatan ketika perubahan konfigurasi lingkungan ke instans masih berlangsung.

> **Note**
>
> Jika Anda memiliki lebih dari dua lingkungan yang sedang mengalami pembaruan, Anda akan diminta untuk memilih nama lingkungan yang Anda ingin memutar kembali perubahan.

Sintaksis

eb abort

eb abort ***environment-name***

Opsi

Nama	Deskripsi
Opsi umum	

Output

Perintah menunjukkan daftar lingkungan yang saat ini sedang diperbarui dan meminta Anda untuk memilih pembaruan yang ingin Anda batalkan. Jika hanya satu lingkungan yang saat ini sedang diperbarui, Anda tidak perlu menentukan nama lingkungan. Jika berhasil, perintah mengalihkan perubahan konfigurasi lingkungan. Proses rollback berlanjut sampai semua instans di lingkungan memiliki konfigurasi lingkungan sebelumnya atau sampai proses pengembalian mengalami kegagalan.

Contoh

Contoh berikut membatalkan peningkatan platform.

```
$ eb abort
Aborting update to environment "tmp-dev".
```

```
<list of events>
```

eb appversion

Deskripsi

Perintah `appversion` EB CLI mengelola versi aplikasi Elastic Beanstalk Anda. Anda dapat membuat versi baru dari aplikasi tanpa men-deploy, menghapus versi aplikasi, atau membuat Kebijakan siklus hidup versi aplikasi. Jika Anda memanggil perintah tanpa opsi apapun, ini memasuki mode interaktif.

Gunakan opsi `--create` untuk membuat versi baru aplikasi.

Gunakan opsi `--delete` untuk menghapus versi aplikasi.

Ggunakan opsi `lifecycle` untuk menampilkan atau membuat kebijakan siklus hidup versi aplikasi. Untuk informasi selengkapnya, lihat the section called “Siklus hidup versi”.

Sintaksis

eb appversion

eb appversion [-c | --create]

eb appversion [-d | --delete] ***version-label***

eb appversion lifecycle [-p | --print]

Opsi

Nama	Deskripsi Tipe: String
-a *application-name* atau --application_name *application-name*	Nama aplikasi. Jika aplikasi dengan nama yang ditentukan tidak ditemukan, EB CLI membuat versi aplikasi untuk aplikasi baru. Hanya berlaku dengan opsi `--create`. Tipe: String
-c	Buat versi baru aplikasi.

Nama	Deskripsi Tipe: String
atau --create	
-d *version-label* atau --hapus *version-label*	Hapus versi aplikasi yang diberi label*version-label* .
`-l` *version_label* atau `--label` *version_label*	Tentukan label yang akan digunakan untuk versi yang dibuat oleh EB CLI. Jika Anda tidak menggunakan opsi ini, EB CLI menghasilkan label unik baru. Jika Anda memberikan label versi, pastikan bahwa label tersebut unik. Hanya berlaku dengan opsi `--create`. Tipe: String
siklus hidup	Memanggil editor default untuk membuat kebijakan siklus hidup versi aplikasi baru. Gunakan kebijakan ini untuk menghindari mencapai kuota versi aplikasi.
siklus hidup -p atau siklus hidup --print	Menampilkan kebijakan siklus hidup aplikasi saat ini.

`-m "`*version_description*` "`
 	
atau
 	
`--message "`*version_description*` "`
	Deskripsi untuk versi aplikasi. Ini tertutup dalam tanda kutip ganda.
	
	Hanya berlaku dengan opsi `--create`.
	
	Tipe: String

Nama	Deskripsi Tipe: String

`-p`
 	
atau
 	
`--process`
	Pra-pemrosesan dan validasi manifes lingkungan dan file konfigurasi di paket sumber. Memvalidasi file konfigurasi dapat mengidentifikasi masalah. Kami rekomendasikan Anda melakukan ini sebelum men-deploy versi aplikasi ke lingkungan.
	
	Hanya berlaku dengan opsi `--create`.

`--source codecommit/` *`repository-name`*`/`*`branch-name`*	CodeCommit repositori dan cabang. Hanya berlaku dengan opsi `--create`.
`--staged`	Gunakan file yang dipersiapkan di indeks git, bukan HEAD commit, untuk membuat versi aplikasi. Hanya berlaku dengan opsi `--create`.
`--timeout` *`minutes`*	Jumlah menit sebelum waktu perintah habis. Hanya berlaku dengan opsi `--create`.
Opsi umum	

Menggunakan perintah secara interaktif

Jika Anda menggunakan perintah tanpa argumen, output menampilkan versi aplikasi. Mereka terdaftar di urutan kronologis terbalik, dengan versi terakhir terdaftar pertama. Lihat bagian Contoh untuk contoh tampilan layar. Perhatikan bahwa baris status ditampilkan di bagian bawah. Baris status menampilkan informasi yang sensitif konteks.

Tekan d untuk menghapus versi aplikasi, tekan `l` untuk mengelola kebijakan siklus hidup aplikasi Anda, atau tekan q untuk berhenti tanpa membuat perubahan apa pun.

> **Note**
>
> Jika versi di-deploy ke lingkungan apa pun, Anda tidak dapat menghapus versi tersebut.

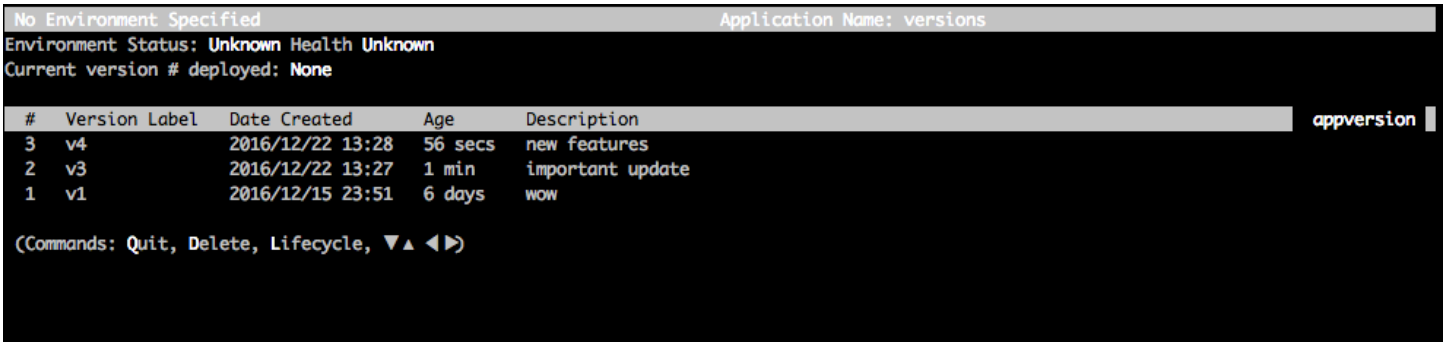
Output

Perintah dengan opsi `--create` menampilkan pesan yang mengonfirmasikan bahwa versi aplikasi telah dibuat.

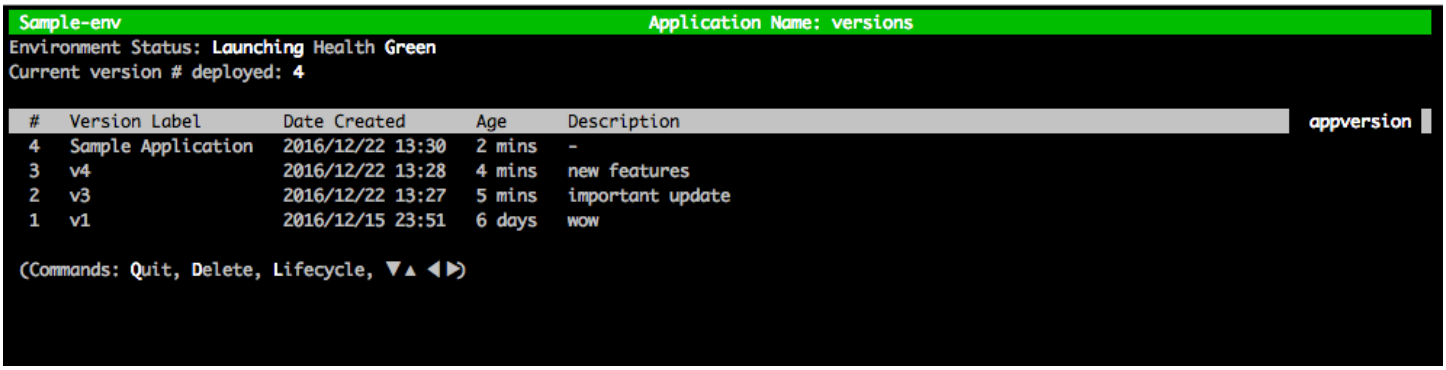
Perintah dengan `--delete` *`version-label`* opsi menampilkan pesan yang mengonfirmasi bahwa versi aplikasi telah dihapus.

Contoh

Contoh berikut menunjukkan jendela interaktif untuk aplikasi tanpa deployment.



Contoh berikut menunjukkan jendela interaktif untuk aplikasi dengan versi keempat, dengan label versi Aplikasi Sampel, di-deploy.



Contoh berikut menunjukkan output dari eb appversion lifecycle -p perintah, di *`ACCOUNT-ID`* mana ID akun pengguna:

```
Application details for: lifecycle
  Region: sa-east-1
  Description: Application created from the EB CLI using "eb init"
  Date Created: 2016/12/20 02:48 UTC
  Date Updated: 2016/12/20 02:48 UTC
```

```
  Application Versions: ['Sample Application']
  Resource Lifecycle Config(s):
    VersionLifecycleConfig:
      MaxCountRule:
        DeleteSourceFromS3: False
        Enabled: False
        MaxCount: 200
      MaxAgeRule:
        DeleteSourceFromS3: False
        Enabled: False
        MaxAgeInDays: 180
    ServiceRole: arn:aws:iam::ACCOUNT-ID:role/aws-elasticbeanstalk-service-role
```

eb clone

Deskripsi

Klon lingkungan ke lingkungan baru sehingga keduanya memiliki pengaturan lingkungan yang identik.

> **Note**
>
> Secara default, terlepas dari versi tumpukan solusi lingkungan dari tempat Anda membuat klon, perintah eb clone membuat klon lingkungan dengan tumpukan solusi terbaru. Anda dapat menekan ini dengan menyertakan opsi `--exact` ketika Anda menjalankan perintah.

> **Important**
>
> Lingkungan Elastic Beanstalk yang dikloning tidak membawa kelompok keamanan untuk masuk, meninggalkan lingkungan terbuka untuk semua lalu lintas internet. Anda harus membangun kembali grup keamanan ingress untuk lingkungan kloning.
> Anda dapat melihat sumber daya yang mungkin tidak dikloning dengan memeriksa status drift konfigurasi lingkungan Anda. Untuk informasi selengkapnya, lihat Mendeteksi drift di seluruh CloudFormation tumpukan di Panduan AWS CloudFormation Pengguna.

Sintaksis

eb clone

eb clone ***environment-name***

Opsi

Nama	Deskripsi
`-n` *`string`* atau `--clone_name` *`string`*	Nama yang diinginkan untuk lingkungan yang diklon.
`-c` *`string`* atau `--cname` *`string`*	Prefiks CNAME yang diinginkan untuk lingkungan yang diklon.
`--envvars`	Properti lingkungan dalam daftar yang dipisahkan koma dengan format *`name`* =. *`value`* Tipe: String Kendala: • Pasangan nilai kunci harus dipisahkan dengan koma. • Kunci dan nilai dapat berisi karakter abjad dalam bahasa apa pun, karakter numerik, spasi, pemisah tak terlihat, dan simbol berikut: _ . : / + \ - @ • Kunci dapat berisi hingga 128 karakter. Nilai dapat berisi hingga 256 karakter. • Kunci dan nilai peka huruf besar dan kecil. • Nilai tidak dapat cocok dengan nama lingkungan. • Nilai tidak dapat mencakup baik `aws:` atau `elasticbeanstalk:` .

Nama	Deskripsi
	• Ukuran gabungan semua properti lingkungan tidak dapat melebihi 4096 byte.
`--exact`	Mencegah Elastic Beanstalk memperbarui versi tumpukan solusi untuk lingkungan klon baru ke versi terbaru yang tersedia (untuk platform lingkungan asli).
`--scale` *number*	Jumlah instans untuk berjalan di lingkungan klon ketika diluncurkan.
`--tags` *name*=*value*	Tag untuk sumber daya di lingkungan Anda dalam daftar yang dipisahkan koma dengan format *name* =. *value*
`--timeout`	Jumlah menit sebelum waktu perintah habis.
Opsi umum	

Output

Jika berhasil, perintah membuat lingkungan yang memiliki pengaturan yang sama seperti lingkungan asli atau dengan modifikasi terhadap lingkungan seperti yang ditentukan oleh opsi eb clone.

Contoh

Contoh berikut mengkloning lingkungan tertentu.

```
$ eb clone
Enter name for Environment Clone
(default is tmp-dev-clone):
Enter DNS CNAME prefix
(default is tmp-dev-clone):
Environment details for: tmp-dev-clone
  Application name: tmp
  Region: us-west-2
  Deployed Version: app-141029_144740
  Environment ID: e-vjvrqnn5pv
  Platform: 64bit Amazon Linux 2014.09 v1.0.9 running PHP 5.5
  Tier: WebServer-Standard-1.0
```

```
  CNAME: tmp-dev-clone.elasticbeanstalk.com
  Updated: 2014-10-29 22:00:23.008000+00:00
Printing Status:
2018-07-11 21:04:20    INFO: createEnvironment is starting.
2018-07-11 21:04:21    INFO: Using elasticbeanstalk-us-west-2-888888888888 as Amazon S3
 storage bucket for environment data.
...
2018-07-11 21:07:10    INFO: Successfully launched environment: tmp-dev-clone
```

eb codesource

Deskripsi

Mengonfigurasi CLI EB untuk diterapkan dari CodeCommit repositori, atau menonaktifkan CodeCommit integrasi dan mengunggah bundel sumber dari mesin lokal Anda.

> **Note**
>
> Beberapa AWS daerah tidak menawarkan CodeCommit. Integrasi antara Elastic CodeCommit Beanstalk dan tidak berfungsi di Wilayah ini.
> Untuk informasi tentang AWS layanan yang ditawarkan di setiap Wilayah, lihat Tabel Wilayah.

Sintaksis

eb codesource

eb codesource codecommit

eb codesource local

Opsi

Nama	Deskripsi
Opsi umum	

Output

eb codesourcememinta Anda untuk memilih antara CodeCommit integrasi dan penerapan standar.

eb codesource codecommitmemulai konfigurasi repositori interaktif untuk integrasi. CodeCommit

eb codesource localmenunjukkan konfigurasi asli dan menonaktifkan CodeCommit integrasi.

Contoh

Gunakan eb codesource codecommit untuk mengkonfigurasi CodeCommit integrasi untuk cabang saat ini.

```
~/my-app$ eb codesource codecommit
Select a repository
1) my-repo
2) my-app
3) [ Create new Repository ]
(default is 1): 1

Select a branch
1) mainline
2) test
3) [ Create new Branch with local HEAD ]
(default is 1): 1
```

Gunakan eb codesource local untuk menonaktifkan CodeCommit integrasi untuk cabang saat ini.

```
~/my-app$ eb codesource local
Current CodeCommit setup:
  Repository: my-app
  Branch: mainline
Default set to use local sources
```

eb config

Deskripsi

Kelola pengaturan konfigurasi aktif dan konfigurasi tersimpan lingkungan Anda. Anda dapat menggunakan perintah ini untuk mengunggah, mengunduh, atau mencantumkan konfigurasi

tersimpan lingkungan Anda. Anda juga dapat menggunakannya untuk mengunduh, menampilkan, atau memperbarui pengaturan konfigurasi aktifnya.

Jika direktori root berisi file `platform.yaml` yang menentukan platform khusus, perintah ini juga mengubah pengaturan konfigurasi pembangun. Hal ini dilakukan berdasarkan nilai-nilai yang ditetapkan di `platform.yaml`.

> **Note**
>
> eb config tidak menampilkan properti lingkungan. Untuk mengatur properti lingkungan yang dapat Anda baca dari dalam aplikasi Anda, gunakan eb setenv sebagai gantinya.

Sintaks

Berikut ini adalah bagian dari sintaks yang digunakan untuk perintah eb config agar bekerja dengan pengaturan konfigurasi aktif lingkungan Anda. Untuk contoh spesifik, lihat bagian Contoh ini nanti di topik ini.

- eb config – Menampilkan pengaturan konfigurasi aktif lingkungan Anda di editor teks yang Anda konfigurasikan sebagai variabel lingkungan EDITOR. Ketika Anda menyimpan perubahan ke file dan menutup editor, lingkungan diperbarui dengan pengaturan opsi yang Anda simpan di file.

  > **Note**
  >
  > Jika Anda tidak mengonfigurasi variabel lingkungan EDITOR, EB CLI menampilkan pengaturan opsi Anda di editor default untuk file YAML.

- eb config ***environment-name*** – Menampilkan dan memperbarui konfigurasi untuk lingkungan bernama. Konfigurasi baik ditampilkan di editor teks yang Anda konfigurasikan atau file YAML editor default Anda.
- eb config save – Menyimpan pengaturan konfigurasi aktif untuk lingkungan saat ini ke `.elasticbeanstalk/saved_configs/` dengan nama file `[configuration-name].cfg.yml`. Secara default, CLI EB menyimpan pengaturan konfigurasi *configuration-name* dengan berdasarkan nama lingkungan. Anda dapat menentukan nama konfigurasi yang berbeda dengan menyertakan opsi `--cfg` dengan nama konfigurasi yang Anda inginkan ketika Anda menjalankan perintah.

Anda dapat memberi tanda pada konfigurasi tersimpan menggunakan opsi `--tags`.

- eb config **`--display`** – Menulis pengaturan konfigurasi aktif lingkungan ke stdout bukan sebuah file. Secara default ini menampilkan pengaturan konfigurasi ke terminal.
- eb config **`--update`** ***`configuration_string`*** **`|`** ***`file_path`***— Memperbarui pengaturan konfigurasi aktif untuk lingkungan saat ini dengan informasi yang ditentukan dalam *`configuration_string`* atau di dalam file yang diidentifikasi oleh*`file_path`*.

> **Note**
>
> Opsi `--display` dan `--update` menyediakan fleksibilitas untuk membaca dan merevisi pengaturan konfigurasi lingkungan secara terprogram.

Berikut ini menjelaskan sintaks untuk menggunakan perintah eb config untuk bekerja dengan konfigurasi tersimpan. Untuk contoh, lihat bagian Contoh ini nanti di topik ini.

- eb config get ***`config-name`*** – Mengunduh konfigurasi tersimpan bernama dari Amazon S3.
- eb config delete ***`config-name`*** – Menghapus konfigurasi tersimpan bernama dari Amazon S3. Juga hapus secara lokal, jika Anda sudah mengunduhnya.
- eb config list – Mencantumkan konfigurasi tersimpan yang Anda miliki di Amazon S3.
- eb config put ***`filename`*** – Mengunggah konfigurasi tersimpan bernama ke bucket Amazon S3. *`filename`*Harus memiliki ekstensi file `.cfg.yml`. Untuk menentukan nama file tanpa jalur, Anda dapat menyimpan file ke folder `.elasticbeanstalk` atau ke folder `.elasticbeanstalk/saved_configs/` sebelum Anda menjalankan perintah. Atau, Anda dapat menentukan *`filename`* dengan menyediakan jalur lengkap.

Opsi

Nama	Deskripsi
`--cfg` *`config-name`*	Nama yang digunakan untuk konfigurasi tersimpan. Opsi ini bekerja dengan eb config save saja.

Nama	Deskripsi

`-d`
 	
atau
 	
`--display`
	Tampilkan pengaturan konfigurasi untuk lingkungan saat ini (menulis ke stdout).
	
	Gunakan dengan opsi `--format` untuk menentukan output berada di JSON atau YAML. Jika Anda tidak menentukan, output dalam format YAML.
	
	Opsi ini hanya bekerja jika Anda menggunakan perintah eb config tanpa salah satu sub perintah lainnya.

`-f` *`format_type`* atau `--format` *`format_type`*	Tentukan format tampilan. Nilai yang valid adalah JSON atau YAML. Default untuk YAML. Opsi ini bekerja dengan opsi `--display` saja.
`--tags` *`key1`*`=`*`value1`*`[,`*`ke`	Tanda untuk ditambahkan ke konfigurasi tersimpan. Saat menentukan tanda di daftar, tentukan mereka sebagai pasangan kunci=nilai dan pisahkan masing-masing dengan koma. Untuk informasi selengkapnya, lihat Menandai konfigurasi tersimpan. Opsi ini bekerja dengan eb config save saja.
`--timeout` *`timeout`*	Jumlah menit sebelum waktu perintah habis.

Nama	Deskripsi

`-u configuration_string | file_path`
 	
atau
 	
`--update configuration_string | file_path`
	Perbarui pengaturan konfigurasi aktif untuk lingkungan saat ini.
	
	Opsi ini hanya bekerja jika Anda menggunakan perintah eb config tanpa salah satu sub perintah lainnya.
	
	Parameter *configuration_string | file_path* adalah tipe string. String menyediakan daftar namespace dan opsi yang sesuai untuk menambah, memperbarui, atau menghapus dari pengaturan konfigurasi untuk lingkungan Anda. Atau, string input dapat mewakili file yang berisi informasi yang sama.
	
	Untuk menentukan nama file, string input harus mengikuti format "file://< *path*><*filename*>". Untuk menentukan nama file tanpa *path*, simpan file ke folder tempat Anda menjalankan perintah. Atau, tentukan nama file dengan menyediakan jalur lengkap.
	
	Informasi konfigurasi harus memenuhi syarat berikut. Setidaknya salah satu bagian, OptionSettings atau OptionsToRemove, diperlukan. Gunakan OptionSettings untuk menambah atau mengubah opsi. Gunakan OptionsToRemove untuk menghapus opsi dari namespace. Untuk contoh spesifik, lihat bagian Contoh ini nanti di topik ini.
	
	Example
	
	Format YAMB
	
	`OptionSettings:`
	`  namespace1:`
	`    option-name-1: option-value-1`
	`    option-name-2: option-value-2`
	`    ...`
	`OptionsToRemove:`
	`  namespace1:`
	`    option-name-1`
	`    option-name-2`

Nama	Deskripsi

	...
	
	Example
	
	Format JSON
	
	{
	"OptionSettings": {
	"namespace1": {
	"option-name-1": " *option-value-1* ",
	"option-name-2": " *option-value-2* ",
	...
	}
	},
	"OptionsToRemove": {
	"namespace1": {
	"option-name-1",
	"option-name-2",
	...
	}
	}
	}

Opsi umum	

Output

Jikaperintah eb config atau eb config ***environment-name*** berhasil berjalan tanpa sub perintah atau opsi ditambahkan, perintah menampilkan pengaturan opsi Anda saat ini di editor teks yang Anda konfigurasikan sebagai variabel lingkungan EDITOR. Jika Anda tidak mengonfigurasi variabel lingkungan EDITOR, EB CLI menampilkan pengaturan opsi Anda di editor default untuk file YAML.

Ketika Anda menyimpan perubahan ke file dan menutup editor, lingkungan diperbarui dengan pengaturan opsi yang Anda simpan di file. Output berikut ditampilkan untuk mengonfirmasi pembaruan konfigurasi.

```
$ eb config myApp-dev
    Printing Status:
```

```
    2021-05-19 18:09:45    INFO    Environment update is starting.
    2021-05-19 18:09:55    INFO    Updating environment myApp-dev's configuration
settings.
    2021-05-19 18:11:20    INFO    Successfully deployed new configuration to
environment.
```

Jika perintah berhasil berjalan dengan opsi `--display`, ini menampilkan pengaturan konfigurasi untuk lingkungan saat ini (menulis ke stdout).

Jika perintah berhasil berjalan dengan parameter `get`, perintah menampilkan lokasi salinan lokal yang Anda unduh.

Jika perintah berhasil berjalan dengan parameter `save`, perintah menampilkan lokasi file tersimpan.

Contoh

Bagian ini menjelaskan cara mengubah editor teks yang Anda gunakan untuk melihat dan mengedit file pengaturan pilihan Anda.

Untuk Linux dan UNIX, contoh berikut mengubah editor menjadi vim:

```
$ export EDITOR=vim
```

Untuk Linux dan UNIX, contoh berikut mengubah editor menjadi apa pun yang dipasang di `/usr/bin/kate`.

```
$ export EDITOR=/usr/bin/kate
```

Untuk Windows, contoh berikut mengubah editor ke Notepad++.

```
> set EDITOR="C:\Program Files\Notepad++\Notepad++.exe
```

Bagian ini menyediakan contoh untuk perintah eb config ketika dijalankan dengan sub perintah.

Contoh berikut menghapus konfigurasi tersimpan bernama `app-tmp`.

```
$ eb config delete app-tmp
```

Contoh berikut mengunduh konfigurasi tersimpan dengan nama app-tmp dari bucket Amazon S3 Anda.

```
$ eb config get app-tmp
```

Contoh berikut mencantumkan nama konfigurasi tersimpan yang disimpan di bucket Amazon S3.

```
$ eb config list
```

Contoh berikut mengunduh salinan lokal konfigurasi tersimpan dengan bernama app-tmp ke bucket Amazon S3 Anda.

```
$ eb config put app-tmp
```

Contoh berikut menyimpan pengaturan konfigurasi dari lingkungan yang berjalan saat ini. Jika Anda tidak memberikan nama untuk digunakan untuk konfigurasi tersimpan, maka Elastic Beanstalk menamakan file konfigurasi sesuai dengan nama lingkungan. Sebagai contoh, sebuah lingkungan bernama tmp-dev akan disebut `tmp-dev.cfg.yml`. Elastic Beanstalk menyimpan file ke folder `/.elasticbeanstalk/saved_configs/`.

```
$ eb config save
```

Pada contoh berikut, opsi `--cfg` digunakan untuk menyimpan pengaturan konfigurasi dari lingkungan tmp-dev ke sebuah file bernama `v1-app-tmp.cfg.yml`. Elastic Beanstalk menyimpan file ke folder `/.elasticbeanstalk/saved_configs/`. Jika Anda tidak menentukan nama lingkungan, Elastic Beanstalk menyimpan pengaturan konfigurasi dari lingkungan yang berjalan saat ini.

```
$ eb config save tmp-dev --cfg v1-app-tmp
```

Bagian ini menyediakan contoh untuk perintah eb config ketika dijalankan tanpa sub perintah.

Perintah berikut menampilkan pengaturan opsi untuk lingkungan Anda saat ini di editor teks.

```
$ eb config
```

Perintah berikut menampilkan pengaturan opsi untuk lingkungan my-env di editor teks.

```
$ eb config my-env
```

Contoh berikut menampilkan pengaturan opsi untuk lingkungan saat ini. Ini mengeluarkan dalam format YAML karena tidak ada format spesifik yang ditentukan dengan opsi `--format`.

```
$ eb config --display
```

Contoh berikut memperbarui pengaturan opsi untuk lingkungan Anda saat ini dengan spesifikasi di file bernama `example.txt`. File ini dalam format YAML atau JSON. EB CLI secara otomatis mendeteksi format file.

- Opsi Minsize diatur ke 1 untuk namespace `aws:autoscaling:asg`.
- Ukuran batch untuk namespace `aws:elasticbeanstalk:command` diatur ke 30%.
- Ini menghapus pengaturan opsi IdleTimeout: Tidak ada dari namespace`AWSEBV2LoadBalancer.aws:elbv2:loadbalancer`.

```
$ eb config --update "file://example.txt"
```

Example - nama file: **`example.txt`** - format YAML

```
OptionSettings:
  'aws:elasticbeanstalk:command':
    BatchSize: '30'
    BatchSizeType: Percentage
  'aws:autoscaling:asg':
    MinSize: '1'
OptionsToRemove:
  'AWSEBV2LoadBalancer.aws:elbv2:loadbalancer':
    IdleTimeout
```

Example - nama file: **`example.txt`** - format JSON

```
{
    "OptionSettings": {
        "aws:elasticbeanstalk:command": {
            "BatchSize": "30",
            "BatchSizeType": "Percentage"
        },
        "aws:autoscaling:asg": {
            "MinSize": "1"
```

```
        }
    },
    "OptionsToRemove": {
        "AWSEBV2LoadBalancer.aws:elbv2:loadbalancer": {
            "IdleTimeout"
        }
    }
}
```

Contoh berikut memperbarui pengaturan opsi untuk lingkungan Anda saat ini. Perintah mengatur opsi Minsize ke 1 untuk namespace `aws:autoscaling:asg`.

> **Note**
>
> Contoh-contoh ini khusus untuk Windows PowerShell. Mereka melepaskan kejadian literal dari karakter kutipan ganda (") dengan mendahuluinya dengan karakter garis miring (\). Sistem operasi yang berbeda dan lingkungan baris perintah mungkin memiliki urutan keluar yang berbeda. Untuk alasan ini, sebaiknya gunakan opsi file yang ditampilkan di contoh sebelumnya. Menentukan opsi konfigurasi di file tidak memerlukan pengeluaran karakter dan konsisten di seluruh sistem operasi yang berbeda.

Contoh berikut dalam format JSON. EB CLI mendeteksi jika format dalam JSON atau YAML.

```
PS C:\Users\myUser\EB_apps\myApp-env>eb config --update '{\"OptionSettings\":
{\"aws:autoscaling:asg\":{\"MaxSize\":\"1\"}}}'
```

Contoh berikut dalam format YAML. Untuk memasukkan string YAMAL dalam format yang benar, perintah menyertakan spasi dan end-of-line pengembalian yang diperlukan dalam file YAMAL.

- Akhiri setiap baris dengan tombol "masuk" atau "kembali".
- Mulai baris kedua dengan dua spasi, dan mulai baris ketiga dengan empat spasi.

```
PS C:\Users\myUser\EB_apps\myApp-env>eb config --update 'OptionSettings:
>>  aws:autoscaling:asg:
>>    MinSize: \"1\"'
```

eb console

Deskripsi

Buka peramban untuk menampilkan dasbor konfigurasi lingkungan di Konsol Manajemen Elastic Beanstalk.

Jika direktori root berisi file `platform.yaml` yang menentukan sebuah platform khusus, perintah ini juga menampilkan konfigurasi lingkungan pembangun, sebagaimana ditentukan di `platform.yaml`, di Konsol Manajemen Elastic Beanstalk.

Sintaksis

eb console

eb console ***environment-name***

Opsi

Nama	Deskripsi
Opsi umum	

eb create

Deskripsi

Membuat lingkungan baru dan men-deploy versi aplikasi Anda ke lingkungan baru.

> **Note**
>
> - Untuk menggunakan eb create di aplikasi .NET, Anda harus membuat paket deployment seperti yang dijelaskan di Membuat bundel sumber untuk aplikasi NET, kemudian mengatur konfigurasi CLI untuk men-deploy paket sebagai Artifact seperti yang dijelaskan di Men-deploy artifact bukan folder proyek.

- Membuat lingkungan dengan EB CLI memerlukan peran layanan. Peran layanan dapat dibuat dengan membuat lingkungan di konsol Elastic Beanstalk. Jika Anda tidak memiliki peran layanan, EB CLI mencoba membuatnya saat Anda menjalankan `eb create`.

Anda dapat men-deploy versi aplikasi dari beberapa sumber:

- Secara default: Dari kode sumber aplikasi di direktori proyek lokal.
- Menggunakan opsi `--version`: Dari versi aplikasi yang sudah ada di aplikasi Anda.
- Ketika direktori proyek Anda tidak memiliki kode aplikasi, atau ketika menggunakan opsi `--sample`: Di-deploy dari aplikasi sampel, khusus untuk platform lingkungan Anda.

Sintaks

eb create

eb create ***environment-name***

Nama lingkungan harus antara 4 dan 40 karakter panjangnya. Hanya dapat berisi huruf, angka, dan tanda hubung (-). Nama lingkungan tidak dapat dimulai atau diakhiri dengan tanda hubung.

Jika Anda menyertakan nama lingkungan di perintah, EB CLI tidak meminta Anda untuk membuat pilihan atau membuat peran layanan.

Jika Anda menjalankan perintah tanpa argumen nama lingkungan, ini berjalan di aliran interaktif, dan meminta Anda untuk memasukkan atau memilih nilai untuk beberapa pengaturan. Di aliran interaktif ini, dalam kasus Anda men-deploy aplikasi sampel, EB CLI juga meminta Anda jika Anda ingin mengunduh aplikasi sampel ini ke direktori proyek lokal Anda. Dengan mengunduhnya, Anda dapat menggunakan EB CLI dengan lingkungan baru nanti untuk menjalankan operasi yang memerlukan kode aplikasi, seperti eb deploy.

Beberapa prompt aliran interaktif ditampilkan hanya dalam syarat tertentu. Misalnya, jika Anda memilih untuk menggunakan Application Load Balancer, dan akun Anda memiliki setidaknya satu Application Load Balancer yang dapat dibagikan, Elastic Beanstalk menampilkan prompt yang menanyakan apakah Anda ingin menggunakan penyeimbang beban bersama. Jika tidak ada Application Load Balancer yang dapat dibagikan di akun Anda, prompt ini tidak ditampilkan.

Opsi

Tak satu pun dari opsi ini diperlukan. Jika Anda menjalankan eb create tanpa opsi apa pun, EB CLI meminta Anda untuk memasukkan atau memilih nilai untuk setiap pengaturan.

Nama	Deskripsi

`-d`
atau
`--branch_default`
	Atur lingkungan sebagai lingkungan default untuk repositori saat ini.

`--cfg` *`config-name`*	Gunakan pengaturan platform dari konfigurasi tersimpan di `.elasticbeanstalk/saved_configs/` atau bucket Amazon S3. Tentukan hanya nama file, tanpa ekstensi `.cfg.yml`.
`-c` *`subdomain-name`* atau `--cname` *`subdomain-name`*	Nama subdomain untuk prefiks entri CNAME DNS yang merutekan ke situs web Anda. Tipe: String Default: Nama lingkungan

`-db`
atau
`--database`
	Lampirkan basis data untuk lingkungan. Jika Anda menjalankan eb create dengan opsi `--database` , tapi tanpa opsi `--database.username` dan `--database.password` , EB CLI meminta Anda untuk nama pengguna utama basis data dan kata sandi.

`-db.engine` *`engine`* atau `--database.engine` *`engine`*	Tipe mesin basis data. Jika Anda menjalankan eb create dengan opsi ini, maka EB CLI meluncurkan lingkungan dengan basis data yang dilampirkan. Hal ini terjadi bahkan jika Anda tidak menjalankan perintah dengan opsi `--database` . Tipe: String

Nama	Deskripsi
	Nilai valid: `mysql`, `oracle-se1` , `postgres`, `sqlserver-ex` , `sqlserver-web` , `sqlserver-se`
`-db.i` *`instance_type`* atau `--database.instance` *`instance_type`*	Jenis EC2 instans Amazon yang digunakan untuk database. Jika Anda menjalankan eb create dengan opsi ini, maka EB CLI meluncurkan lingkungan dengan basis data yang dilampirkan. Hal ini terjadi bahkan jika Anda tidak menjalankan perintah dengan opsi `--database` . Tipe: String Nilai valid: Amazon RDS mendukung seperangkat instans DB standar. Untuk memilih instans DB yang sesuai untuk mesin DB Anda, Anda harus mempertimbangkan beberapa pertimbangan khusus. Untuk informasi selengkapnya, lihat kelas instans DB di Panduan Pengguna Amazon RDS.
`-db.pass` *`password`* atau `--database.password` *`password`*	Kata sandi untuk basis data. Jika Anda menjalankan eb create dengan opsi ini, maka EB CLI meluncurkan lingkungan dengan basis data yang dilampirkan. Hal ini terjadi bahkan jika Anda tidak menjalankan perintah dengan opsi `--database` .

Nama	Deskripsi
`-db.size` *`number_of_gigabytes`* atau `--database.size` *`number_of_gigabytes`*	Jumlah gigabita (GB) untuk mengalokasikan untuk penyimpanan basis data. Jika Anda menjalankan eb create dengan opsi ini, maka EB CLI meluncurkan lingkungan dengan basis data yang dilampirkan. Hal ini terjadi bahkan jika Anda tidak menjalankan perintah dengan opsi `--database` . Tipe: Angka Nilai valid: • MySQL – `5` ke `1024`. Default-nya adalah `5`. • Postgres – `5` ke `1024`. Default-nya adalah `5`. • Oracle – `10` ke `1024`. Default-nya adalah `10`. • Microsoft SQL Server Express Edition – `30`. • Microsoft SQL Server Web Edition – `30`. • Microsoft SQL Server Standard Edition – `200`.
`-db.user` *`username`* atau `--database.username` *`username`*	Nama pengguna untuk basis data. Jika Anda menjalankan eb create dengan opsi ini, maka EB CLI meluncurkan lingkungan dengan basis data terlampir meskipun Anda tidak menjalankan perintah dengan opsi `--database` . Jika Anda menjalankan eb create dengan opsi `--database` , tapi tanpa opsi `--database.username` dan `--database.password` , kemudian EB CLI meminta Anda untuk nama pengguna basis data utama dan kata sandi.
`-db.version` *`version`* atau `--database.version` *`version`*	Digunakan untuk menentukan versi mesin basis data. Jika bendera ini hadir, lingkungan akan meluncurkan dengan basis data dengan nomor versi tertentu, bahkan jika bendera `--database` tidak ada.

Nama	Deskripsi
`--elb-type` *type*	tipe penyeimbang beban. Tipe: String Nilai valid: `classic`, `application` , `network` Default: `application`

`-es`
 	
atau
 	
`--enable-spot`
	Aktifkan permintaan Instans Spot untuk lingkungan Anda. Untuk informasi selengkapnya, lihat Grup Auto Scaling.
	
	Tindakan terkait:
	
	• `--instance-types`
	• `--on-demand-base-capacity`
	• `--on-demand-above-base-capacity`
	• `--spot-max-price`

`--env-group-suffix` *groupname*	Nama grup untuk menambahkan nama lingkungan. Hanya untuk digunakan dengan Penyusunan Lingkungan.
`--envvars`	Properti lingkungan dalam daftar yang dipisahkan koma dengan format *name* =. *value* Lihat Mengkonfigurasi properti lingkungan (variabel lingkungan) untuk batas.
`-ip` *profile_name* atau `--instance_profile` *profile_name*	Profil instans dengan peran IAM dengan kredenal keamanan sementara yang dibutuhkan aplikasi Anda untuk mengakses sumber daya. AWS

Nama	Deskripsi

`-it`
 	
atau
 	
`--instance-types` *`type1`*`[,`*`type2`* `...]`
	Daftar tipe EC2 instans Amazon yang dipisahkan koma yang Anda inginkan untuk digunakan oleh lingkungan Anda. Jika Anda tidak menentukan opsi ini, Elastic Beanstalk menyediakan tipe instans default.
	
	Untuk informasi selengkapnya, lihat EC2 Contoh Amazon dan Grup Auto Scaling.
	
	Important
	EB CLI hanya memberlakukan opsi ini untuk Instans Spot. Kecuali opsi ini digunakan dengan opsi `--enable-spot` , EB CLI mengabaikan itu. Untuk menentukan tipe instans untuk Instans Sesuai Permintaan, gunakan opsi `--intance-type` (tanpa "s") sebagai gantinya.

`-i`
 	
atau
 	
`--instance_type`
	Jenis EC2 instans Amazon yang Anda inginkan untuk digunakan oleh lingkungan Anda. Jika Anda tidak menentukan opsi ini, Elastic Beanstalk menyediakan tipe instans default.
	
	Untuk informasi selengkapnya, lihat EC2 Contoh Amazon.
	
	Important
	EB CLI hanya memberlakukan opsi ini untuk Instans Sesuai Permintaan. Jangan gunakan opsi ini dengan opsi `--enable-spot` , karena EB CLI mengabaikannya ketika Anda melakukannya. Untuk menentukan tipe instans untuk Instans Spot, gunakan opsi `--intance-types` (tanpa "s") sebagai gantinya.

Nama	Deskripsi
`-k` *`key_name`* atau `--keyname` *`key_name`*	Nama EC2 key pair Amazon untuk digunakan dengan klien Secure Shell (SSH) untuk masuk dengan aman ke EC2 instans Amazon yang menjalankan aplikasi Elastic Beanstalk Anda. Jika Anda menyertakan opsi ini dengan perintah eb create, nilai yang Anda berikan menimpa setiap nama kunci yang mungkin telah Anda ditentukan dengan eb init. Nilai yang valid: Nama kunci yang sudah ada yang terdaftar di Amazon EC2
`-im` *`number-of-instances`* atau `--min-instances` *`number-of-instances`*	Jumlah minimum EC2 instance Amazon yang Anda butuhkan untuk dimiliki lingkungan Anda. Tipe: Nomor (bilangan bulat) Default: 1 Nilai valid: 1 hingga `10000`
`-ix` *`number-of-instances`* atau `--max-instances` *`number-of-instances`*	Jumlah maksimum EC2 instans Amazon yang Anda izinkan untuk dimiliki lingkungan Anda. Tipe: Nomor (bilangan bulat) Default: `4` Nilai valid: 1 hingga `10000`
`--modules` *`component-a component-b`*	Daftar lingkungan komponen untuk membuat. Hanya untuk digunakan dengan Penyusunan Lingkungan.

Nama	Deskripsi

`-sb`
 	
atau
 	
`--on-demand-base-capacity`
	Jumlah minimum Instans Sesuai Permintaan yang disediakan grup Auto Scaling Anda sebelum mempertim bangkan Instans Spot saat lingkungan Anda menaikkan skala.
	
	Opsi ini hanya dapat ditentukan dengan opsi `--enable-spot` . Untuk informasi selengkapnya, lihat Grup Auto Scaling.
	
	Tipe: Nomor (bilangan bulat)
	
	Default: `0`
	
	Nilai valid: `0` hingga `--max-instances` (ketika tidak ada: opsi `MaxSize` di namespace `aws:autos caling:asg`)

`-sp`
 	
atau
 	
`--on-demand-above-base-capacity`
	Persentase Instans Sesuai Permintaan sebagai bagian dari kapasitas tambahan yang disediakan grup Auto Scaling lebih dari jumlah instans yang ditentukan oleh opsi `--on-demand-base-capacity` .
	
	Opsi ini hanya dapat ditentukan dengan opsi `--enable-spot` . Untuk detail selengkapnya, lihat Grup Auto Scaling.
	
	Tipe: Nomor (bilangan bulat)
	
	Default: `0` untuk lingkungan instans tunggal; `70` untuk lingkungan yang seimbang beban
	
	Nilai valid: `0` hingga `100`

Nama	Deskripsi
`-p` *`platform-version`* atau `--platform` *`platform-version`*	Versi platform untuk digunakan. Anda dapat menentukan platform, platform dan versi, cabang platform, nama tumpukan solusi, atau solusi tumpukan ARN. Sebagai contoh: • `php`, `PHP`, `node.js` – Versi platform terbaru untuk platform yang ditentukan • `php-7.2`, `"PHP 7.2"` – Versi platform PHP 7.2 yang direkomendasikan (biasanya terbaru) • `"PHP 7.2 running on 64bit Amazon Linux"` – Versi platform PHP yang direkomendasikan (biasanya terbaru) di cabang platform ini • `"64bit Amazon Linux 2017.09 v2.6.3 running PHP 7.1"` – Versi platform PHP yang ditentukan oleh nama tumpukan solusi ini • `"arn:aws:elasticbeanstalk:us-east-2::platform/PHP 7.1 running on 64bit Amazon Linux/2.6.3"` – Versi platform PHP yang ditentukan oleh tumpukan solusi ARN ini Gunakan `eb platform list` untuk mendapatkan daftar konfigurasi yang tersedia. Jika Anda menentukan opsi `--platform` , opsi ini menimpa nilai yang diberikan selama `eb init`.

`-pr`
 	
atau
 	
`--process`
	Pra-pemrosesan dan validasi manifes lingkungan dan file konfigurasi di paket sumber. Memvalidasi file konfigurasi dapat mengidentifikasi masalah sebelum men-deploy versi aplikasi ke lingkungan.

Nama	Deskripsi
`-r` *region* atau `--region` *region*	AWS Wilayah tempat Anda ingin menyebarkan aplikasi. Untuk daftar nilai yang dapat Anda tentukan untuk opsi ini, lihat AWS Elastic Beanstalk Titik akhir dan Kuota di. Referensi Umum AWS
`--sample`	Deploy aplikasi sampel ke lingkungan baru bukan ke kode di repositori Anda.
`--scale` *number-of-instances*	Meluncurkan dengan jumlah instans yang ditentukan
`--service-role` *servicerole*	Tetapkan peran layanan non-default ke lingkungan. **Note** Jangan masukkan ARN. Hanya masukkan nama peran. Elastic Beanstalk mengawali nama peran dengan nilai yang benar untuk membuat ARN yang dihasilkan secara internal.

Nama	Deskripsi
`-ls` *`load-balancer`* atau `--shared-lb` *`load-balancer`*	Konfigurasikan lingkungan untuk menggunakan penyeimbang beban bersama. Berikan nama atau ARN dari penyeimbang beban yang dapat dibagikan di akun Anda—Application Load Balancer yang secara eksplisit Anda buat, bukan yang dibuat oleh lingkungan Elastic Beanstalk lainnya. Untuk informasi selengkapnya, lihat Application Load Balancer Bersama. Contoh parameter: • `FrontEndLB` – Nama penyeimbang beban. • `arn:aws:elasticloadbalancing:us-east-2:123456789012:loadbalancer/app/FrontEndLB/0dbf78d8ad96abbc` – Application Load Balancer ARN. Anda dapat menentukan opsi ini hanya dengan `--elb-type application`. Jika Anda menentukan opsi tersebut dan tidak menentukan `--shared-lb`, Elastic Beanstalk membuat penyeimbang beban khusus untuk lingkungan.
`-lp` *`port`* atau `--shared-lb-port` *`port`*	Port listener default penyeimbang beban bersama untuk lingkungan ini. Elastic Beanstalk menambahkan aturan listener yang merutekan semua lalu lintas dari listener ini ke proses default lingkungan. Untuk informasi selengkapnya, lihat Application Load Balancer Bersama. Tipe: Nomor (bilangan bulat) Default: `80` Nilai valid: Setiap bilangan bulat yang mewakili port listener dari penyeimbang beban bersama.

Nama	Deskripsi
`--single`	Buat lingkungan dengan satu EC2 instans Amazon dan tanpa penyeimbang beban. **Warning** Lingkungan instans tunggal tidak siap produksi. Jika instans menjadi tidak stabil selama deployment, atau Elastic Beanstalk berakhir dan mengulang kembali instans selama pembaruan konfigurasi, aplikasi Anda dapat tidak tersedia untuk jangka waktu tertentu. Gunakan lingkungan instans tunggal untuk pengembangan, pengujian, atau pementasan. Gunakan lingkungan seimbang beban untuk produksi.

`-sm`
 	
atau
 	
`--spot-max-price`
	Harga maksimum per unit jam, dalam dolar A.S., yang bersedia Anda bayarkan untuk Instans Spot.
	
	Opsi ini hanya dapat ditentukan dengan opsi `--enable-spot` . Untuk detail selengkapnya, lihat Grup Auto Scaling.
	
	Tipe: Nomor (float)
	
	Default: Harga Sesuai Permintaan, untuk setiap jenis instans. Nilai opsi pada kasus ini adalah `null`.
	
	Nilai valid: `0.001` hingga `20.0`
	
	Untuk rekomendasi tentang opsi harga maksimum untuk Instans Spot, lihat riwayat harga Instans Spot di Panduan EC2 Pengguna Amazon.

Nama	Deskripsi
`--tags` *key1*`=`*value1*`[,`*key2*`=`*va*	Tandai sumber daya di lingkungan Anda. Tanda ditentuka n sebagai daftar yang dipisahkan koma pasangan `key=value` . Untuk informasi selengkapnya, lihat Menandai lingkunga n.

`-t worker`
 	
atau
 	
`--tier worker`
	Buat lingkungan pekerja. Abaikan opsi ini untuk membuat lingkungan web server.

`--timeout` *minutes*	Atur jumlah menit sebelum waktu perintah habis.
`--version` *version_label*	Menentukan versi aplikasi yang ingin Anda deploy ke lingkungan bukan kode sumber aplikasi di direktori proyek lokal. Tipe: String Nilai valid: Label versi aplikasi yang ada
`--vpc`	Konfigurasikan VPC untuk lingkungan Anda. Ketika Anda menyertakan opsi ini, EB CLI meminta Anda untuk memasukkan semua pengaturan yang diperlukan sebelum meluncurkan lingkungan.
`--vpc.dbsubnets` *subnet1,s ubnet2*	Menentukan subnet untuk instans basis data di VPC. Diperlukan ketika `--vpc.id` ditentukan.
`--vpc.ec2subnets` *subnet1,s ubnet2*	Menentukan subnet untuk EC2 instans Amazon dalam VPC. Diperlukan ketika `--vpc.id` ditentukan.

Nama	Deskripsi
`--vpc.elbpublic`	Luncurkan penyeimbang beban Elastic Load Balancing Anda di subnet publik di VPC Anda. Anda tidak dapat menentukan opsi ini dengan opsi `--tier worker` atau `--single`.
`--vpc.elbsubnets` *subnet1,subnet2*	Tentukan subnet untuk penyeimbang beban Elastic Load Balancing di VPC. Anda tidak dapat menentukan opsi ini dengan opsi `--tier worker` atau `--single`.
`--vpc.id` *ID*	Luncurkan lingkungan Anda di VPC yang ditentukan.
`--vpc.publicip`	Meluncurkan EC2 instans Amazon Anda di subnet publik di VPC Anda. Anda tidak dapat menentukan opsi ini dengan opsi `--tier worker`.
`--vpc.securitygroups` *securitygroup1,securitygroup2*	Menentukan kelompok IDs keamanan. Diperlukan ketika `--vpc.id` ditentukan.
Opsi umum	

Output

Jika berhasil, perintah meminta Anda dengan pertanyaan dan kemudian mengembalikan status operasi pembuatan. Jika ada masalah selama peluncuran, Anda dapat menggunakan operasi eb events untuk mendapatkan detail selengkapnya.

Jika Anda mengaktifkan CodeBuild dukungan dalam aplikasi Anda, eb create menampilkan informasi dari CodeBuild saat kode Anda dibuat. Untuk informasi tentang CodeBuild dukungan di Elastic Beanstalk, lihat. Menggunakan EB CLI dengan AWS CodeBuild

Contoh

Contoh berikut membuat lingkungan dalam mode interaktif.

```
$ eb create
Enter Environment Name
(default is tmp-dev): ENTER
Enter DNS CNAME prefix
(default is tmp-dev): ENTER
Select a load balancer type
1) classic
2) application
3) network
(default is 2): ENTER
Environment details for: tmp-dev
  Application name: tmp
  Region: us-east-2
  Deployed Version: app-141029_145448
  Environment ID: e-um3yfrzq22
  Platform: 64bit Amazon Linux 2014.09 v1.0.9 running PHP 5.5
  Tier: WebServer-Standard-1.0
  CNAME: tmp-dev.elasticbeanstalk.com
  Updated: 2014-10-29 21:54:51.063000+00:00
Printing Status:
...
```

Contoh berikut juga membuat lingkungan dalam mode interaktif. Di contoh ini, direktori proyek Anda tidak memiliki kode aplikasi. Perintah men-deploy aplikasi sampel dan mengunduhnya ke direktori proyek lokal Anda.

```
$ eb create
Enter Environment Name
(default is tmp-dev): ENTER
Enter DNS CNAME prefix
(default is tmp-dev): ENTER
Select a load balancer type
1) classic
2) application
3) network
(default is 2): ENTER
NOTE: The current directory does not contain any source code. Elastic Beanstalk is
 launching the sample application instead.
```

```
Do you want to download the sample application into the current directory?
(Y/n): ENTER
INFO: Downloading sample application to the current directory.
INFO: Download complete.
Environment details for: tmp-dev
  Application name: tmp
  Region: us-east-2
  Deployed Version: Sample Application
  Environment ID: e-um3yfrzq22
  Platform: 64bit Amazon Linux 2014.09 v1.0.9 running PHP 5.5
  Tier: WebServer-Standard-1.0
  CNAME: tmp-dev.elasticbeanstalk.com
  Updated: 2017-11-08 21:54:51.063000+00:00
Printing Status:
...
```

Perintah berikut membuat lingkungan tanpa menampilkan permintaanya.

```
$ eb create dev-env
Creating application version archive "app-160312_014028".
Uploading test/app-160312_014028.zip to S3. This may take a while.
Upload Complete.
Application test has been created.
Environment details for: dev-env
  Application name: test
  Region: us-east-2
  Deployed Version: app-160312_014028
  Environment ID: e-6fgpkjxyyi
  Platform: 64bit Amazon Linux 2015.09 v2.0.8 running PHP 5.6
  Tier: WebServer-Standard
  CNAME: UNKNOWN
  Updated: 2016-03-12 01:40:33.614000+00:00
Printing Status:
...
```

Perintah berikut membuat lingkungan di VPC khusus.

```
$ eb create dev-vpc --vpc.id vpc-0ce8dd99 --vpc.elbsubnets subnet-
b356d7c6,subnet-02f74b0c --vpc.ec2subnets subnet-0bb7f0cd,subnet-3b6697c1 --
vpc.securitygroup sg-70cff265
Creating application version archive "app-160312_014309".
Uploading test/app-160312_014309.zip to S3. This may take a while.
Upload Complete.
```

```
Environment details for: dev-vpc
  Application name: test
  Region: us-east-2
  Deployed Version: app-160312_014309
  Environment ID: e-pqkcip3mns
  Platform: 64bit Amazon Linux 2015.09 v2.0.8 running Java 8
  Tier: WebServer-Standard
  CNAME: UNKNOWN
  Updated: 2016-03-12 01:43:14.057000+00:00
Printing Status:
...
```

eb deploy

Deskripsi

Deploy paket sumber aplikasi dari direktori proyek yang diinisialisasi untuk aplikasi yang berjalan.

Jika git dipasang, EB CLI menggunakan perintah `git archive` untuk membuat file `.zip` dari konten perintah `git commit` terbaru.

Nmun, saat `.ebignore` ada di direktori proyek Anda, EB CLI tidak menggunakan perintah git untuk membuat paket sumber Anda. Ini berarti EB CLI mengabaikan file yang ditentukan di `.ebignore`, dan menyertakan semua file lainnya. Khususnya, ia termasuk file sumber yang tidak terikat.

> **Note**
>
> Anda dapat mengonfigurasi EB CLI untuk men-deploy Artifact dari proses membangun Anda bukan membuat file ZIP folder proyek Anda. Lihat Men-deploy artifact bukan folder proyek untuk detailnya.

Sintaksis

eb deploy

eb deploy ***environment-name***

Opsi

Nama	Deskripsi
`-l` *version_label* atau `--label` *version_label*	Tentukan label yang akan digunakan untuk versi yang dibuat oleh EB CLI. Jika label telah digunakan, EB CLI men-deploy ulang versi sebelumnya dengan label tersebut. Tipe: String
`--env-group-suffix` *groupname*	Nama grup untuk menambahkan nama lingkungan. Hanya untuk digunakan dengan Penyusunan Lingkungan.

`-m "`*version_description*` "`
 	
atau
 	
`--message "`*version_description*` "`
	Deskripsi untuk versi aplikasi, tertutup dalam tanda kutip ganda.
	
	Tipe: String

`--modules` *component-a component-b*	Daftar komponen untuk diperbarui. Hanya untuk digunakan dengan Penyusunan Lingkungan.

`-p`
 	
atau
 	
`--process`
	Pra-pemrosesan dan validasi manifes lingkungan dan file konfigurasi di paket sumber. Memvalidasi file konfigurasi dapat mengidentifikasi masalah sebelum men-deploy versi aplikasi ke lingkungan.

`--source codecommit/` *repository-name/branch-name*	CodeCommit repositori dan cabang.
`--staged`	Deploy persiapan file di indeks git bukan HEAD commit.
`--timeout` *minutes*	Jumlah menit sebelum waktu perintah habis.
`--version` *version_label*	Versi aplikasi yang ada untuk di-deploy.

Nama	Deskripsi
	Tipe: String
Opsi umum	

Output

Jika berhasil, perintah mengembalikan status operasi `deploy`.

Jika Anda mengaktifkan CodeBuild dukungan dalam aplikasi Anda, eb deploy menampilkan informasi dari CodeBuild saat kode Anda dibuat. Untuk informasi tentang CodeBuild dukungan di Elastic Beanstalk, lihat. Menggunakan EB CLI dengan AWS CodeBuild

Contoh

Contoh berikut men-deploy aplikasi saat ini.

```
$ eb deploy
2018-07-11 21:05:22    INFO: Environment update is starting.
2018-07-11 21:05:27    INFO: Deploying new version to instance(s).
2018-07-11 21:05:53    INFO: New application version was deployed to running EC2
 instances.
2018-07-11 21:05:53    INFO: Environment update completed successfully.
```

eb events

Deskripsi

Mengembalikan peristiwa terbaru untuk lingkungan.

Jika direktori root berisi `platform.yaml` file yang menentukan platform khusus, perintah ini juga mengembalikan peristiwa terbaru untuk lingkungan pembangun.

Sintaksis

eb events

eb events ***environment-name***

Opsi

Nama	Deskripsi

`-f`
 	
atau
 	
`--follow`
	Streaming peristiwa. Untuk membatalkan, tekan CTRL+C.

Output

Jika berhasil, perintah mengembalikan peristiwa terbaru.

Contoh

Contoh berikut mengembalikan peristiwa terbaru.

```
$ eb events
2014-10-29 21:55:39     INFO    createEnvironment is starting.
2014-10-29 21:55:40     INFO    Using elasticbeanstalk-us-west-2-111122223333 as Amazon
 S3 storage bucket for environment data.
2014-10-29 21:55:57     INFO    Created load balancer named: awseb-e-r-AWSEBLoa-
NSKUOK5X6Z9J
2014-10-29 21:56:16     INFO    Created security group named: awseb-e-rxgrhjr9bx-stack-
AWSEBSecurityGroup-1UUHU5LZ20ZY7
2014-10-29 21:57:18     INFO    Waiting for EC2 instances to launch. This may take a
 few minutes.
2014-10-29 21:57:18     INFO    Created Auto Scaling group named: awseb-e-rxgrhjr9bx-
stack-AWSEBAutoScalingGroup-1TE320ZCJ9RPD
2014-10-29 21:57:22     INFO    Created Auto Scaling group policy named:
 arn:aws:autoscaling:us-east-2:11122223333:scalingPolicy:2cced9e6-859b-421a-
be63-8ab34771155a:autoScalingGroupName/awseb-e-rxgrhjr9bx-stack-
AWSEBAutoScalingGroup-1TE320ZCJ9RPD:policyName/awseb-e-rxgrhjr9bx-stack-
AWSEBAutoScalingScaleUpPolicy-1I2ZSNVU4APRY
2014-10-29 21:57:22     INFO    Created Auto Scaling group policy named:
 arn:aws:autoscaling:us-east-2:11122223333:scalingPolicy:1f08b863-
bf65-415a-b584-b7fa3a69a0d5:autoScalingGroupName/awseb-e-rxgrhjr9bx-stack-
AWSEBAutoScalingGroup-1TE320ZCJ9RPD:policyName/awseb-e-rxgrhjr9bx-stack-
AWSEBAutoScalingScaleDownPolicy-1E3G7PZKZPSOG
```

```
2014-10-29 21:57:25     INFO    Created CloudWatch alarm named: awseb-e-rxgrhjr9bx-
stack-AWSEBCloudwatchAlarmLow-VF5EJ549FZBL
2014-10-29 21:57:25     INFO    Created CloudWatch alarm named: awseb-e-rxgrhjr9bx-
stack-AWSEBCloudwatchAlarmHigh-LA9YEW3O6WJO
2014-10-29 21:58:50     INFO    Added EC2 instance 'i-c7ee492d' to Auto ScalingGroup
 'awseb-e-rxgrhjr9bx-stack-AWSEBAutoScalingGroup-1TE320ZCJ9RPD'.
2014-10-29 21:58:53     INFO    Successfully launched environment: tmp-dev
2014-10-29 21:59:14     INFO    Environment health has been set to GREEN
2014-10-29 21:59:43     INFO    Adding instance 'i-c7ee492d' to your environment.
```

eb health

Deskripsi

Mengembalikan kondisi terbaru untuk lingkungan.

Jika direktori root berisi file `platform.yaml` yang menentukan platform khusus, perintah ini juga mengembalikan kondisi terbaru untuk lingkungan pembangun.

Sintaksis

eb health

eb health ***environment-name***

Opsi

Nama	Deskripsi

`-r`
 	
atau
 	
`--refresh`
	Tampilkan informasi kondisi secara interaktif dan perbarui setiap 10 detik saat informasi baru dilaporkan.

`--mono`	Jangan tampilkan warna di output.

Output

Jika berhasil, perintah mengembalikan kondisi terbaru.

Contoh

Contoh berikut mengembalikan informasi kondisi terbaru untuk lingkungan Linux.

```
~/project $ eb health
 elasticBeanstalkExa-env                                Ok
 2015-07-08 23:13:20
WebServer
 Ruby 2.1 (Puma)
  total      ok    warning  degraded  severe    info   pending  unknown
    5        5        0        0        0        0        0        0

  instance-id   status     cause
                                      health
    Overall     Ok
  i-d581497d    Ok
  i-d481497c    Ok
  i-136e00c0    Ok
  i-126e00c1    Ok
  i-8b2cf575    Ok

  instance-id   r/sec    %2xx   %3xx   %4xx   %5xx      p99      p90      p75     p50
   p10                              requests
    Overall     671.8   100.0    0.0    0.0    0.0    0.003    0.002    0.001   0.001
 0.000
  i-d581497d    143.0    1430      0      0      0    0.003    0.002    0.001   0.001
 0.000
  i-d481497c    128.8    1288      0      0      0    0.003    0.002    0.001   0.001
 0.000
  i-136e00c0    125.4    1254      0      0      0    0.004    0.002    0.001   0.001
 0.000
  i-126e00c1    133.4    1334      0      0      0    0.003    0.002    0.001   0.001
 0.000
  i-8b2cf575    141.2    1412      0      0      0    0.003    0.002    0.001   0.001
 0.000

  instance-id   type       az   running     load 1  load 5      user%  nice%  system%
 idle%   iowait%                            cpu
  i-d581497d    t2.micro   1a   12 mins       0.0    0.04        6.2    0.0      1.0
 92.5       0.1
  i-d481497c    t2.micro   1a   12 mins      0.01    0.09        5.9    0.0      1.6
 92.4       0.1
```

```
 i-136e00c0    t2.micro   1b   12 mins       0.15    0.07        5.5    0.0      0.9
93.2       0.0
 i-126e00c1    t2.micro   1b   12 mins       0.17    0.14        5.7    0.0      1.4
92.7       0.1
 i-8b2cf575    t2.micro   1c   1 hour        0.19    0.08        6.5    0.0      1.2
92.1       0.1

 instance-id   status     id   version             ago
                                 deployments
 i-d581497d    Deployed   1    Sample Application   12 mins
 i-d481497c    Deployed   1    Sample Application   12 mins
 i-136e00c0    Deployed   1    Sample Application   12 mins
 i-126e00c1    Deployed   1    Sample Application   12 mins
 i-8b2cf575    Deployed   1    Sample Application   1 hour
```

eb init

Deskripsi

Tetapkan nilai default untuk aplikasi Elastic Beanstalk yang dibuat dengan EB CLI dengan meminta Anda dengan serangkaian pertanyaan.

> **Note**
>
> Nilai yang Anda eb init tetapkan berlaku untuk direktori dan repositori saat ini di komputer saat ini.
> Perintah membuat aplikasi Elastic Beanstalk di akun Anda. Untuk membuat lingkungan Elastic Beanstalk, jalankan eb create setelah menjalankan eb init.

Sintaksis

eb init

eb init *application-name*

Opsi

Jika Anda menjalankan eb init tanpa menentukan opsi `--platform`, EB CLI meminta Anda untuk memasukkan nilai untuk setiap pengaturan.

> **Note**
>
> Untuk menggunakan eb init untuk membuat pasangan kunci baru, Anda harus memasang `ssh-keygen` di komputer lokal Anda dan tersedia dari baris perintah.

Nama	Deskripsi	

`-i`
 	
`--interactive`
	Memaksa EB CLI untuk meminta Anda untuk memberikan nilai untuk setiap opsi perintah eb init.
	
	> **Note**
	> Perintah `init` meminta Anda untuk memberikan nilai untuk opsi perintah eb init yang tidak memiliki nilai (default). Setelah pertama kali Anda menjalankan perintah eb init di direktori, EB CLI mungkin tidak meminta Anda tentang opsi perintah apa pun. Oleh karena itu, gunakan opsi `--interactive` ketika Anda ingin mengubah pengaturan yang sebelumnya Anda tetapkan.	

`-k` *`keyname`* `--keyname` *`keyname`*	Nama EC2 key pair Amazon untuk digunakan dengan klien Secure Shell (SSH) untuk masuk dengan aman ke EC2 instans Amazon yang menjalankan aplikasi Elastic Beanstalk Anda.	
`--modules` *`folder-1 folder-2`*	Daftar direktori anak untuk menginisialisasi. Hanya untuk digunakan dengan Penyusunan Lingkungan.	
`-p` *`platform-version`* `--platform` *`platform-version`*	Versi platform untuk digunakan. Anda dapat menentukan platform, platform dan versi, cabang platform, nama tumpukan solusi, atau solusi tumpukan ARN. Sebagai contoh:	

Nama	Deskripsi	
	• `php`, `PHP`, `node.js` – Versi platform terbaru untuk platform yang ditentukan • `php-7.2`, `"PHP 7.2"` – Versi platform PHP 7.2 yang direkomendasikan (biasanya terbaru) • `"PHP 7.2 running on 64bit Amazon Linux"` – Versi platform PHP yang direkomendasikan (biasanya terbaru) di cabang platform ini • `"64bit Amazon Linux 2017.09 v2.6.3 running PHP 7.1"` – Versi platform PHP yang ditentukan oleh nama tumpukan solusi ini • `"arn:aws:elasticbeanstalk:us-east-2::platform/PHP 7.1 running on 64bit Amazon Linux/2.6.3"` – Versi platform PHP yang ditentukan oleh tumpukan solusi ARN ini Gunakan `eb platform list` untuk mendapatkan daftar konfigurasi yang tersedia. Tentukan opsi `--platform` untuk melewati konfigurasi interaktif. **Note** Bila Anda menentukan opsi ini, maka EB CLI tidak meminta Anda untuk nilai untuk opsi lain. Sebaliknya, ini mengasumsikan nilai default untuk setiap opsi. Anda dapat menentukan opsi untuk apa pun yang Anda tidak ingin gunakan nilai default.	
`--source codecommit/` *`repository-name`*`/`*`branch-name`*	CodeCommit repositori dan cabang.	

Nama	Deskripsi	
`--tags` *key1*=*value1*	Beri tanda aplikasi Anda. Tanda ditentukan sebagai daftar yang dipisahkan koma pasangan `key=value` . Untuk detail selengkapnya, lihat Aplikasi pemberian label.	
Opsi umum		

CodeBuild dukungan

Jika Anda menjalankan eb init di folder yang berisi file buildspec.yml, Elastic Beanstalk menguraikan file untuk entri eb_codebuild_settings dengan opsi khusus untuk Elastic Beanstalk. Untuk informasi tentang CodeBuild dukungan di Elastic Beanstalk, lihat. Menggunakan EB CLI dengan AWS CodeBuild

Output

Jika berhasil, perintah memandu Anda menyiapkan aplikasi Elastic Beanstalk baru melalui serangkaian permintaan.

Contoh

Contoh permintaan berikut menginisialisasi EB CLI dan meminta Anda untuk memasukkan informasi tentang aplikasi Anda. Ganti *placeholder* teks dengan nilai Anda sendiri.

```
$ eb init -i
Select a default region
1) us-east-1 : US East (N. Virginia)
2) us-west-1 : US West (N. California)
3) us-west-2 : US West (Oregon)
4) eu-west-1 : Europe (Ireland)
5) eu-central-1 : Europe (Frankfurt)
6) ap-south-1 : Asia Pacific (Mumbai)
7) ap-southeast-1 : Asia Pacific (Singapore)
...
(default is 3): 3
```

```
Select an application to use
1) HelloWorldApp
2) NewApp
3) [ Create new Application ]
(default is 3): 3

Enter Application Name
(default is "tmp"):
Application tmp has been created.

It appears you are using PHP. Is this correct?
(y/n): y

Select a platform branch.
1) PHP 7.2 running on 64bit Amazon Linux
2) PHP 7.1 running on 64bit Amazon Linux (Deprecated)
3) PHP 7.0 running on 64bit Amazon Linux (Deprecated)
4) PHP 5.6 running on 64bit Amazon Linux (Deprecated)
5) PHP 5.5 running on 64bit Amazon Linux (Deprecated)
6) PHP 5.4 running on 64bit Amazon Linux (Deprecated)
(default is 1): 1
Do you want to set up SSH for your instances?
(y/n): y

Select a keypair.
1) aws-eb
2) [ Create new KeyPair ]
(default is 2): 1
```

eb labs

Deskripsi

Subperintah eb labs dukungan work-in-progress atau fungsionalitas eksperimental. Perintah-perintah ini dapat dihapus atau dikerjakan ulang di versi EB CLI yang akan datang dan tidak dijamin kompatibelnya.

Untuk daftar sub perintah dan deskripsi yang tersedia, jalankan eb labs --help.

eb list

Deskripsi

Cantumkan semua lingkungan di aplikasi saat ini atau semua lingkungan di semua aplikasi, sebagaimana ditentukan oleh opsi `--all`.

Jika direktori root berisi file `platform.yaml` yang menentukan platform khusus, perintah ini juga mencatumkan lingkungan pembangun.

Sintaksis

eb list

Opsi

Nama	Deskripsi

`-a`
atau
`--all`
	Cantumkan semua lingkungan dari semua aplikasi.

`-v`
atau
`--verbose`
	Berikan informasi detail selengkapnya tentang semua lingkungan, termasuk instans.

Opsi umum	

Output

Jika berhasil, perintah mengembalikan daftar nama lingkungan saat lingkungan Anda saat ini ditandai dengan tanda bintang (*).

Contoh 1

Contoh berikut mencantumkan lingkungan Anda dan menunjukkan bahwa tmp-dev adalah lingkungan default Anda.

```
$ eb list
* tmp-dev
```

Contoh 2

Contoh berikut mencantumkan lingkungan Anda dengan detail tambahan.

```
$ eb list --verbose
Region: us-west-2
Application: tmp
    Environments: 1
        * tmp-dev : ['i-c7ee492d']
```

eb local

Deskripsi

Gunakan eb local run untuk menjalankan kontainer aplikasi Anda secara lokal di Docker. Periksa status kontainer aplikasi dengan eb local status. Buka aplikasi di peramban web dengan eb local open. Ambil lokasi log aplikasi dengan eb local logs.

eb local setenv dan eb local printenv memungkinkan Anda mengatur dan melihat variabel lingkungan yang disediakan untuk kontainer Docker yang Anda jalankan secara lokal dengan eb local run.

Anda harus menjalankan semua perintah eb local di direktori proyek aplikasi Docker yang telah diinisialisasi sebagai repositori EB CLI dengan menggunakan eb init.

> **Note**
>
> Gunakan eb local di komputer lokal yang menjalankan Linux atau macOS. Perintah tidak mendukung Windows.
> Sebelum menggunakan perintah di macOS, pasang Docker untuk Mac, dan pastikan boot2docker tidak terpasang (atau tidak berada di jalur eksekusi). Perintah eb local mencoba menggunakan boot2docker jika ada, namun tidak berfungsi dengan baik di macOS.

Sintaksis

eb local run

eb local status

eb local open

eb local logs

eb local setenv

eb local printenv

Opsi

eb local run

Nama	Deskripsi
`--envvars` *`key1=value1,key2=value2`*	Atur variabel lingkungan yang EB CLI akan diteruskan ke kontainer Docker lokal. Di lingkungan banyak kontainer, semua variabel diteruskan ke semua kontainer.
`--port` *`hostport`*	Petakan port di host ke port terbuka di kontainer. Jika Anda tidak menentukan opsi ini, EB CLI menggunakan port yang sama di kedua host dan kontainer. Opsi ini hanya berfungsi dengan aplikasi platform Docker. Ini tidak berlaku untuk platform Multicontainer Docker.
Opsi umum	

eb local status

eb local open

eb local logs

eb local setenv

eb local printenv

Nama	Deskripsi
Opsi umum	

Output

eb local run

Pesan status dari Docker. Tetap aktif selama aplikasi berjalan. Tekan Ctrl+C untuk menghentikan aplikasi.

eb local status

Status setiap kontainer yang digunakan oleh aplikasi, berjalan atau tidak.

eb local open

Buka aplikasi di web peramban dan keluar.

eb local logs

Lokasi log yang dihasilkan di direktori proyek Anda oleh aplikasi yang berjalan secara lokal di bawah eb local run.

eb local setenv

Tidak ada

eb local printenv

Nama dan nilai-nilai variabel lingkungan diatur dengan eb local setenv.

Contoh

eb local run

```
~/project$ eb local run
Creating elasticbeanstalk_phpapp_1...
```

```
Creating elasticbeanstalk_nginxproxy_1...
Attaching to elasticbeanstalk_phpapp_1, elasticbeanstalk_nginxproxy_1
phpapp_1     | [23-Apr-2015 23:24:25] NOTICE: fpm is running, pid 1
phpapp_1     | [23-Apr-2015 23:24:25] NOTICE: ready to handle connections
```

eb local status

Lihat status kontainer lokal Anda:

```
~/project$ eb local status
Platform: 64bit Amazon Linux 2014.09 v1.2.1 running Multi-container Docker 1.3.3
 (Generic)
Container name: elasticbeanstalk_nginxproxy_1
Container ip: 127.0.0.1
Container running: True
Exposed host port(s): 80
Full local URL(s): 127.0.0.1:80

Container name: elasticbeanstalk_phpapp_1
Container ip: 127.0.0.1
Container running: True
Exposed host port(s): None
Full local URL(s): None
```

eb local logs

Lihat jalur log untuk proyek saat ini:

```
~/project$ eb local logs
Elastic Beanstalk will write logs locally to /home/user/project/.elasticbeanstalk/logs/
local.
Logs were most recently created 3 minutes ago and written to /home/user/
project/.elasticbeanstalk/logs/local/150420_234011665784.
```

eb local setenv

Atur variabel lingkungan untuk digunakan dengan eb local run.

```
~/project$ eb local setenv PARAM1=value
```

Cetak variabel lingkungan yang diatur dengan eb local setenv.

```
~/project$ eb local printenv
Environment Variables:
PARAM1=value
```

eb logs

Deskripsi

eb logsPerintah ini memiliki dua tujuan berbeda: untuk mengaktifkan atau menonaktifkan streaming CloudWatch log ke Log, dan untuk mengambil log instance atau CloudWatch log Log. Dengan opsi `--cloudwatch-logs` (`-cw`), perintah tersebut mengaktifkan atau menonaktifkan streaming log. Tanpa opsi ini, perintah mengambil log.

Saat mengambil log, tentukan opsi `--all`, `--zip`, atau `--stream` untuk mengambil log yang lengkap. Jika Anda tidak menentukan salah satu opsi ini, Elastic Beanstalk mengambil log ekor.

Perintah memproses log untuk lingkungan tertentu atau default. Log yang relevan bervariasi menurut jenis kontainer. Jika direktori root berisi file `platform.yaml` yang menentukan platform khusus, perintah ini juga memproses log untuk lingkungan pembangun.

Untuk informasi selengkapnya, lihat the section called “CloudWatch Log”.

Sintaks

Untuk mengaktifkan atau menonaktifkan streaming log ke CloudWatch Log:

```
eb logs --cloudwatch-logs [enable | disable] [--cloudwatch-log-source instance |
 environment-health | all] [environment-name]
```

Untuk mengambil log instans:

```
eb logs [-all | --zip | --stream] [--cloudwatch-log-source instance] [--
instance instance-id] [--log-group log-group] [environment-name]
```

Untuk mengambil log kondisi lingkungan:

```
eb logs [-all | --zip | --stream] --cloudwatch-log-source environment-health
 [environment-name]
```

Opsi

Nama	Deskripsi

`-cw [enable | disable]`
 	
atau
 	
`--cloudwatch-logs [enable | disable]`
	Mengaktifkan atau menonaktifkan streaming log ke CloudWatch Log. Jika tidak ada argumen yang diberikan, streaming log diaktifkan. Jika opsi `--cloudwatch-log-source (-cls)` tidak ditentukan sebagai tambahan, streaming log instans diaktifkan atau dinonaktifkan.

`-cls instance | environment-health | all`
 	
atau
 	
`--cloudwatch-log-source instance | environment-health | all`
	Menentukan sumber log saat bekerja dengan CloudWatch Log. Dengan mengaktifkan atau menonaktifkan bentuk perintah, ini adalah log untuk mengaktifkan atau menonaktifkan streaming CloudWatch Log. Dengan bentuk pengambilan perintah, ini adalah log untuk mengambil dari CloudWatch Log.
	
	Nilai valid:
	
	• Dengan `--cloudwatch-logs` (mengaktifkan atau menonaktifkan) – `instance | environment-health | all`
	• Tanpa `--cloudwatch-logs` (ambil) – `instance | environment-health`
	
	Arti nilai:
	
	• `instance` (default) – Log instans
	• `environment-health` – Log kondisi lingkungan (didukung hanya ketika peningkatan kondisi diaktifkan di lingkungan)
	• `all` – Kedua sumber log

`-a`
 	
atau
 	
`--all`
	Ambil log lengkap dan simpan ke direktori `.elasticbeanstalk/logs`.

Nama	Deskripsi

`-z`
 	
atau
 	
`--zip`
	Ambil log lengkap, kompres mereka menjadi file `.zip`, dan kemudian simpan file ke direktori `.elasticbeanstalk/logs` .

`--stream`	Pengaliran (output terus menerus) log lengkap. Dengan opsi ini, perintah terus berjalan sampai Anda menginterupsinya (tekan **`Ctrl+C`**).
`-i` *`instance-id`* atau `--instance` *`instance-id`*	Ambil log untuk instans tertentu saja.

Nama	Deskripsi
`-g` *`log-group`* atau `--log-group` *`log-group`*	Menentukan kelompok CloudWatch log Log dari mana untuk mengambil log. Opsi ini hanya valid ketika streaming log instance ke CloudWatch Log diaktifkan. Jika streaming log instans diaktifkan, dan Anda tidak menentukan opsi `--log-group` , grup log default adalah salah satu hal berikut: • Amazon Linux 2 - `/aws/elasticbeanstalk/` *`environment-name`* `/var/log/eb-engine.log` • Platform Windows – `/aws/elasticbeanstalk/` *`environment-name`* `/EBDeploy-Log` • Amazon Linux AMI (AL1) — `/aws/elasticbeanstalk/` *`environment-name`* `/var/log/eb-activity.log` **Note** Pada tanggal 18 Juli 2022, Elastic Beanstalk menetapkan status semua cabang platform berdasarkan Amazon Linux AMI () untuk pensiun. AL1 Untuk informasi selengkapnya tentang migrasi ke cabang platform Amazon Linux 2023 saat ini dan didukung penuh, lihat. Memigrasi aplikasi Elastic Beanstalk Linux Anda ke Amazon Linux 2023 atau Amazon Linux 2 Untuk informasi tentang grup log yang sesuai untuk setiap berkas log, lihat Bagaimana Elastic Beanstalk mengatur Log CloudWatch .
Opsi umum	

Output

Secara default, tampilkan log langsung di terminal. Gunakan program halaman untuk menampilkan output. Tekan **`Q`** atau **`q`** untuk keluar.

Dengan `--stream`, tunjukkan log yang ada di terminal dan terus berjalan. Tekan **`Ctrl+C`** untuk keluar.

Dengan `--all` dan `--zip`, simpan log ke file lokal dan tampilkan lokasi file.

Contoh

Contoh berikut memungkinkan streaming log instance ke CloudWatch Log.

```
$ eb logs -cw enable
Enabling instance log streaming to CloudWatch for your environment
After the environment is updated you can view your logs by following the link:
https://console.aws.amazon.com/cloudwatch/home?region=us-east-1#logs:prefix=/aws/
elasticbeanstalk/environment-name/
Printing Status:
2018-07-11 21:05:20    INFO: Environment update is starting.
2018-07-11 21:05:27    INFO: Updating environment environment-name's configuration
 settings.
2018-07-11 21:06:45    INFO: Successfully deployed new configuration to environment.
```

Contoh berikut mengambil log instans ke file `.zip`.

```
$ eb logs --zip
Retrieving logs...
Logs were saved to /home/workspace/environment/.elasticbeanstalk/logs/150622_173444.zip
```

eb migrate

Deskripsi

Memigrasi situs dan aplikasi Layanan Informasi Internet (IIS) dari server Windows ke Elastic Beanstalk. Perintah mengemas aplikasi Anda, mempertahankan konfigurasinya, dan menerapkannya ke lingkungan Elastic Beanstalk baru.

Untuk informasi selengkapnya tentang memigrasi situs dan aplikasi IIS Anda, lihat. Migrasi aplikasi IIS

> **Note**
>
> Sebelum menggunakan perintah ini, pastikan sistem Anda memenuhi persyaratan ini:
>
> - Internet Information Services (IIS) versi 7.0 atau yang lebih baru
> - Web Deploy 3.6 atau yang lebih baru diinstal
> - Hak istimewa administratif di server Windows
> - AWS kredensil yang dikonfigurasi dengan izin yang sesuai
> - Server sumber Anda memiliki akses internet keluar ke AWS layanan.

Langkah-langkah berikut merangkum proses migrasi:

1. Temukan situs IIS dan konfigurasinya.
2. Konten dan konfigurasi aplikasi Package.
3. Buat lingkungan dan aplikasi Elastic Beanstalk.
4. Menyebarkan aplikasi dengan pengaturan yang diawetkan.

Perintah membuat artefak migrasi dalam direktori terstruktur seperti yang ditunjukkan dalam daftar berikut:

```
C:\migration_workspace\
### .\migrations\latest\
    ### upload_target.zip
    ### upload_target\
        ### [SiteName].zip                    # One ZIP per default application of
 IIS site
        ### [SiteName-ApplicationName].zip     # One ZIP per additional application
        ### aws-windows-deployment-manifest.json
        ### ebmigrateScripts\
            ### site_installer.ps1             # Site installation scripts
            ### permission_handler.ps1         # Permission management
            ### other helper scripts
```

Gunakan eb migrate cleanup untuk mengelola artefak ini.

Sintaksis

eb migrate ***[options]***

eb migrate explore ***[options]***

eb migrate cleanup ***[options]***

Ketika dijalankan tanpa argumen, eb migrate beroperasi dalam mode non-interaktif. Untuk menjalankannya dalam mode interaktif, jalankaneb migrate --interactive.

Perintah mode interaktif meminta informasi berikut:

- Pemilihan situs IIS untuk bermigrasi
- Nama lingkungan dan aplikasi
- Pemilihan versi platform
- Jenis instans dan opsi konfigurasi lainnya

Subperintah

explore

eb migrate exploreSubperintah memeriksa server IIS Anda dan daftar situs yang tersedia.

Gunakan perintah ini untuk menampilkan informasi berikut:

- Lihat semua situs IIS di server
- Dengan`--verbose`, periksa konfigurasi terperinci termasuk:
  - Binding dan port situs
  - Kolam aplikasi
  - Direktori virtual dan jalur fisiknya
  - Pengaturan otentikasi

```
PS C:\migrations_workspace >  eb migrate explore
Default Web Site
Site2
site3
router
```

```
PS C:\migrations_workspace >  eb migrate explore --verbose
1: Default Web Site:
  - Bindings:
    - *:8083:
  - Application '/':
    - Application Pool: DefaultAppPool
    - Enabled Protocols: http
    - Virtual Directories:
      - /:
        - Physical Path: C:\inetpub\wwwroot
        - Logon Method: ClearText
  - Application '/dotnet-6-0':
    - Application Pool: DefaultAppPool
    - Enabled Protocols: http
    - Virtual Directories:
      - /:
        - Physical Path: C:\inetpub\AspNetCoreWebApps\CoreWebApp-6-0
        - Logon Method: ClearText
  - Application '/dotnet-8-0':
    - Application Pool: DefaultAppPool
    - Enabled Protocols: http
    - Virtual Directories:
      - /:
        - Physical Path: C:\inetpub\AspNetCoreWebApps\CoreWebApp-8-0
        - Logon Method: ClearText
2: Site2:
  - Bindings:
    - *:8081:
...
```

pembersihan

eb migrate cleanupSubperintah mengelola artefak migrasi dengan tindakan berikut:

- Melestarikan migrasi sukses terbaru di `./migrations/latest`
- Menghapus direktori migrasi lama
- Mempertahankan file konfigurasi kritis

```
PS C:\migrations_workspace >   eb migrate cleanup
Are you sure you would like to cleanup older artifacts within `./migrations/`? (y/N):
```

Gunakan `--force` untuk melewati permintaan konfirmasi selama pembersihan.

```
PS C:\migrations_workspace >   eb migrate cleanup --force
```

Opsi

Tak satu pun dari opsi ini diperlukan. Jika Anda menjalankan eb migrate tanpa opsi apa pun, CLI EB akan dijalankan dalam mode non-interaktif. Denganeb migrate --interactive, CLI EB meminta Anda untuk memasukkan atau memilih nilai untuk pengaturan yang diperlukan.

Nama	Deskripsi
`-a` *`application-name`* atau `--application-name` *`application-name`*	Nama untuk aplikasi Elastic Beanstalk baru. Tipe: String Default: EBMigrated Aplikasi
`--archive` *`directory-or-zip`*	Direktori atau file ZIP yang berisi kode sumber yang sebelumnya dihasilkan oleheb migrate --archive-only. Gunakan opsi ini untuk menerapkan paket migrasi yang dibuat sebelumnya. Contoh: `--archive .\migrations\latest\upload_target` atau `--archive .\migrations\latest\upload_target.zip`

`-ao`
 	
atau
 	
`--archive-only`
	Buat hanya direktori arsip tujuan tanpa penerapan.
	
	Direktori yang dihasilkan dapat digunakan secara manual menggunakan eb migrate dengan `archive` opsi, ataueb deploy.

`-c` *`subdomain-name`* atau `--cname` *`subdomain-name`*	Nama subdomain untuk awalan entri DNS CNAME untuk aplikasi yang dimigrasi. Tipe: String

Nama	Deskripsi
	Default: Nama lingkungan

`-cf`
 	
atau
 	
`--copy-firewall-config`
	Salin konfigurasi firewall server sumber ke tujuan untuk semua port HTTP dengan binding aktif.
	
	Membuat aturan grup keamanan yang sesuai di AWS.

`-es` *`snapshot-id`* [*`snapshot-id`* ...] atau `--ebs-snapshots` *`snapshot-id`* [*`snapshot-id`* ...]	Daftar snapshot Amazon EBS yang dipisahkan koma IDs untuk dikaitkan dengan lingkungan. Contoh: `--ebs-snapshots snap-1234567890abcdef0, snap-0987654321fedcba1`
`--encrypt-ebs-volumes`	Menerapkan enkripsi untuk semua volume Amazon EBS baru. **Important** Ini adalah pengaturan seluruh akun yang memengaruhi semua pembuatan volume Amazon EBS di masa mendatang.
`-e` *`environment-name`* atau `--environment-name` *`environment-name`*	Nama untuk lingkungan Elastic Beanstalk yang baru. Tipe: String Default: EBMigrated Env Kendala: Panjangnya harus antara 4 dan 40 karakter. Hanya dapat berisi huruf, angka, dan tanda hubung. Tidak dapat memulai atau mengakhiri dengan tanda hubung.

Nama	Deskripsi
`--force`	Lewati permintaan konfirmasi selama operasi. Ketika digunakan dengan cleanup subperintah, menghapus artefak migrasi tanpa konfirmasi.
`-ip` *`profile-name`* atau `--instance-profile` *`profile-name`*	Profil Instance untuk diasosiasikan dengan EC2 instans Amazon lingkungan. Jika tidak ditentukan, buat profil instance default dengan izin untuk mengakses sumber daya Elastic Beanstalk. Untuk informasi selengkapnya, lihat the section called “Profil instans”.
`-i` *`instance-type`* atau `--instance-type` *`instance-type`*	Jenis EC2 instans Amazon untuk lingkungan Elastic Beanstalk Anda. Tipe: String Default: c5.2xlarge Untuk jenis instans yang tersedia, lihat jenis EC2 instans Amazon di Panduan EC2 Pengguna Amazon.

`-in`
 	
atau
 	
`--interactive`
	Paksa mode interaktif untuk proses migrasi.
	
	Meminta nilai konfigurasi bahkan ketika default tersedia.

`-k` *`key-name`* atau `--keyname` *`key-name`*	Amazon EC2 key pair untuk mengaktifkan akses RDP ke instance lingkungan. Berguna untuk menyelidiki masalah tingkat instance yang tidak terlihat di log. Nilai yang valid: Nama key pair yang sudah ada terdaftar di Amazon EC2

Nama	Deskripsi
`-p` *`platform-version`* atau `--platform` *`platform-version`*	Runtime platform Elastic Beanstalk untuk lingkungan. Jika tidak ditentukan, secara otomatis terdeteksi dari host versi Windows Server. Contoh: `"64bit Windows Server 2016 v2.16.2 running IIS 10.0"` Untuk daftar versi platform yang tersedia, gunakan daftar platform eb.
`--remote`	Menunjukkan untuk menjalankan migrasi dalam mode jarak jauh. Opsi ini memungkinkan eksekusi dari host bastion, yang terhubung ke server target yang berisi aplikasi dan konfigurasi yang akan dimigrasikan ke Elastic Beanstalk. Berjalan dari server bastion, `eb migrate` menemukan konfigurasi, mentahapkan logika migrasi pada host bastion, lalu menyebarkan aplikasi Anda ke lingkungan Elastic Beanstalk baru. Opsi ini menghilangkan kebutuhan untuk menginstal EB CLI dan Python di server Windows yang perlu Anda migrasi. Anda menginstal Python dan EB CLI pada host bastion sebagai gantinya, di mana Anda menjalankan eb migrate perintah dengan opsi. `--remote` Gunakan `--target-ip` opsi untuk menentukan host dengan konfigurasi IIS untuk bermigrasi. Harus digunakan dengan`--target-ip` ,`--username` , dan`--password` .
`--target-ip` *`ip-address`*	Alamat IP publik dari mesin Windows jarak jauh yang berisi server IIS yang akan dimigrasikan. Diperlukan saat menggunakan`--remote`. Hanya dapat ditentukan saat menggunakan`--remote`.

Nama	Deskripsi
`--username` *`username`*	Nama pengguna profil pengguna untuk mengakses mesin Windows jarak jauh yang berisi server IIS yang akan dimigrasi. Diperlukan saat menggunakan`--remote`. Hanya dapat ditentukan saat menggunakan`--remote`.
`--password` *`password`*	Kata sandi profil pengguna untuk mengakses mesin Windows jarak jauh yang berisi server IIS yang akan dimigrasi. Diperlukan saat menggunakan`--remote`. Hanya dapat ditentukan saat menggunakan`--remote`.
`-sr` *`role-name`* atau `--service-role` *`role-name`*	Peran layanan IAM untuk Elastic Beanstalk untuk mengelola layanan terkait. AWS Jika tidak ditentukan, buat peran layanan default dengan izin yang diperlukan. Untuk informasi selengkapnya, lihat the section called “Peran layanan”. **Note** Tentukan hanya nama peran, bukan ARN lengkap. Elastic Beanstalk secara otomatis menciptakan ARN lengkap.
`-s` *`site-names`* atau `--sites` *`site-names`*	Daftar situs IIS yang dipisahkan koma untuk bermigrasi. Jika tidak ditentukan, memigrasikan semua situs yang tersedia di server. Contoh: `--sites "Default Web Site,Intranet,API"`

Nama	Deskripsi
`--ssl-certificates` *certificate-arn* [,*certificate-arn* ...]	Daftar sertifikat SSL ACM yang dipisahkan koma ARNs untuk dikaitkan dengan Application Load Balancer. Diperlukan saat memigrasi situs dengan binding HTTPS. Contoh: `--ssl-certificates arn:aws:acm:region:account:certificate/certificate-id`
`-t` *key1*=*value1*[,*key2*=*value2* .. atau `--tags` *key1*=*value1*[,*key2*=*va*	Daftar `key=value` pasangan yang dipisahkan koma untuk menandai sumber daya baru di lingkungan Anda: Lingkungan, aplikasi Elastic Beanstalk, versi Aplikasi. Untuk informasi selengkapnya, lihat Menandai lingkungan.
`--verbose`	Tampilkan informasi terperinci selama proses migrasi. Saat digunakan dengan explore subperintah, menampilkan detail konfigurasi situs yang komprehensif.

Nama	Deskripsi

`-vpc` *`config-file-or-string`*
 	
atau
 	
`--vpc-config` *`config-file-or-string`*
	Konfigurasi VPC untuk lingkungan, ditentukan baik sebagai jalur file JSON atau string JSON.
	
	Konfigurasi harus mencakup:
	
	`{`
	`"id": "vpc-1234567890abcdef0",`
	`"publicip": "true|false",`
	`"elbscheme": "public|private",`
	`"ec2subnets": ["subnet-a1b2c3d4",`
	`"subnet-e5f6g7h8"],`
	`"securitygroups": "sg-123456,sg-789012",`
	`"elbsubnets": ["subnet-a1b2c3d4",`
	`"subnet-e5f6g7h8"]`
	`}`
	
	• `id`: (Diperlukan) pengidentifikasi VPC
	• `publicip`: Apakah akan menetapkan publik IPs ke instance
	• `elbscheme` : Skema penyeimbang beban (publik atau pribadi)
	• `ec2subnets` : Daftar subnet IDs untuk instance EC2
	• `securitygroups` : Kelompok keamanan yang dipisahkan koma IDs
	• `elbsubnets` : Daftar subnet IDs untuk penyeimbang beban
	
	Important
	Migrasi akan mengabaikan pengaturan VPC yang ada dari lingkungan sumber saat Anda menentukan parameter. `--vpc-config` Saat Anda menggunakan parameter ini, migrasi hanya akan menggunakan pengaturan VPC yang ditentukan dalam file konfigurasi yang

Nama	Deskripsi
	Anda lewati. Menggunakan parameter ini mengesampingkan perilaku default untuk menemukan konfigurasi VPC instance sumber atau menggunakan VPC default.
Opsi umum	

Output

Perintah menyediakan pembaruan status selama proses migrasi:

1. Deteksi konfigurasi VPC (saat berjalan pada sebuah EC2 instance)
2. Kemajuan pembuatan bundel sumber untuk setiap situs
3. Status penciptaan lingkungan
4. Kemajuan penerapan

Jika berhasil, menampilkan detail lingkungan baru termasuk:

- Nama dan ID lingkungan
- Nama aplikasi
- Wilayah
- Versi platform
- Lingkungan CNAME

Untuk masalah selama migrasi, gunakan eb healthperintah eb eventsand untuk mendapatkan informasi terperinci.

Contoh

Penggunaan Dasar

Migrasi dasar dalam mode interaktif:

```
PS C:\migrations_workspace >  eb migrate
```

```
Identifying VPC configuration of this EC2 instance (i-0123456789abcdef0):
  id: vpc-1234567890abcdef0
  publicip: true
  elbscheme: public
  ec2subnets: subnet-123,subnet-456,subnet-789
  securitygroups: sg-123,sg-456
  elbsubnets: subnet-123,subnet-456,subnet-789

Using .\migrations\latest to contain artifacts for this migration run.
Generating source bundle for sites, applications, and virtual directories...
  Default Web Site/ -> .\migrations\latest\upload_target\DefaultWebSite.zip

Creating application version
Creating environment

Environment details for: EBMigratedEnv
  Application name: EBMigratedApp
  Region: us-west-2
  Deployed Version: app-230320_153045
  Environment ID: e-abcdef1234
  Platform: 64bit Windows Server 2019 v2.7.0 running IIS 10.0
  Tier: WebServer-Standard-1.0
  CNAME: ebmigratedenv.us-west-2.elasticbeanstalk.com
  Updated: 2023-03-20 15:30:45
```

Migrasi situs tertentu dengan konfigurasi khusus:

```
PS C:\migrations_workspace >  eb migrate `
    --sites "Default Web Site,InternalAPI" `
    --application-name "CorporateApp" `
    --environment-name "Production" `
    --instance-type "c5.xlarge" `
    --tags "Environment=Production,Team=WebOps" `
    --copy-firewall-config
```

Membuat arsip migrasi tanpa penerapan:

```
PS C:\migrations_workspace >  eb migrate --archive-only
Using .\migrations\latest to contain artifacts for this migration run.
Generating source bundle for sites, applications, and virtual directories...
  Default Web Site/ -> .\migrations\latest\upload_target\DefaultWebSite.zip

Generated destination archive directory at .\migrations\latest\upload_target
```

```
You can execute `eb init` and `eb create` from this directory to deploy to EB.
```

Contoh Konfigurasi Lanjutan

Migrasi dengan konfigurasi VPC kustom menggunakan file JSON:

```
PS C:\migrations_workspace >  cat vpc-config.json
{
    "id": "vpc-1234567890abcdef0",
    "publicip": "false",
    "elbscheme": "internal",
    "ec2subnets": [
        "subnet-private1",
        "subnet-private2"
    ],
    "securitygroups": [
        "sg-app",
        "sg-database",
        "sg-monitoring"
    ],
    "elbsubnets": [
        "subnet-private1",
        "subnet-private2"
    ]
}

PS C:\migrations_workspace eb migrate `
    --sites "InternalAPI" `
    --vpc-config vpc-config.json `
    --instance-type "r5.xlarge" `
    --tags "Environment=Internal,Security=High"
```

Migrasi situs dengan sertifikat SSL dan header host:

```
PS C:\migrations_workspace >  eb migrate `
    --sites "SecurePortal" `
    --ssl-certificates "arn:aws:acm:region:account:certificate/
cert1,arn:aws:acm:region:account:certificate/cert2" `
    --verbose
INFO: Detected HTTPS bindings:
  - www.example.com:443
  - api.example.com:443
```

```
INFO: Configuring Application Load Balancer with SSL certificates
INFO: Creating host-based routing rules:
  - www.example.com -> target group 1
  - api.example.com -> target group 2
```

Migrasi dengan konfigurasi snapshot EBS: fo

```
PS C:\migrations_workspace >  eb migrate `
    --sites "Default Web Site" `
    --ebs-snapshots "snap-1234567890abcdef0" "snap-0987654321fedcba1" `
    --encrypt-ebs-volumes
Using .\migrations\latest to contain artifacts for this migration run.
INFO: Enabling EBS encryption for all new volumes in us-west-2
INFO: Configuring environment with specified EBS snapshots
```

Contoh Konfigurasi Keamanan

Menangani situs dengan aturan firewall yang kompleks:

```
PS C:\migrations_workspace >  eb migrate `
    --sites "Default Web Site,ReportingService" `
    --copy-firewall-config `
    --verbose
INFO: Detected the following Windows Firewall rules:
  - Allow Web Traffic (TCP 80, 443)
  - Allow Reporting Traffic (TCP 8081)
INFO: Creating corresponding security group rules
```

Migrasi dengan peran IAM khusus:

```
PS C:\migrations_workspace >  eb migrate `
    --sites "SecureApp" `
    --instance-profile "CustomInstanceProfile" `
    --service-role "CustomServiceRole"
```

Contoh Eksekusi Jarak Jauh

Migrasi aplikasi IIS dari server Windows jarak jauh:

```
PS C:\migrations_workspace >  eb migrate `
    --remote `
```

```
    --target-ip "192.0.2.10" `
    --username "administrator" `
    --password "YourPassword123" `
    --application-name "RemoteApp" `
    --environment-name "RemoteEnv"
INFO: Establishing SSH connection to remote host 192.0.2.10...
INFO: Connection established
INFO: Discovering IIS sites on remote host...
INFO: Found 2 sites: Default Web Site, API
INFO: Extracting site configurations...
INFO: Generating source bundle for sites, applications, and virtual directories...
  Default Web Site/ -> .\migrations\latest\upload_target\DefaultWebSite.zip
  API/ -> .\migrations\latest\upload_target\API.zip

Creating application version
Creating environment

Environment details for: RemoteEnv
  Application name: RemoteAppstage mi
  Region: us-west-2
  Deployed Version: app-230320_153045
  Environment ID: e-abcdef1234
  Platform: 64bit Windows Server 2019 v2.7.0 running IIS 10.0
  Tier: WebServer-Standard-1.0
  CNAME: remoteenv.us-west-2.elasticbeanstalk.com
  Updated: 2023-03-20 15:30:45
```

Migrasi jarak jauh dengan pemilihan lokasi tertentu:

```
PS C:\migrations_workspace >  eb migrate `
    --remote `
    --target-ip "192.0.2.10" `
    --username "administrator" `
    --password "YourPassword123" `
    --sites "API" `
    --instance-type "c5.large"
```

eb open

Deskripsi

Buka URL publik situs web Anda di peramban default.

Sintaksis

eb open

eb open ***environment-name***

Opsi

Nama	Deskripsi
Opsi umum	

Output

Perintah eb open tidak memiliki output. Sebaliknya, ini membuka aplikasi di jendela peramban.

eb platform

Deskripsi

Perintah ini mendukung dua ruang kerja yang berbeda:

Platform

Gunakan ruang kerja ini untuk mengelola platform khusus.

Lingkungan

Gunakan ruang kerja ini untuk memilih platform default atau menampilkan informasi tentang platform saat ini.

Elastic Beanstalk menyediakan jalan pintas ebp untuk eb platform.

> **Note**
>
> Windows PowerShell menggunakan ebp sebagai alias perintah. Jika Anda menjalankan EB CLI di PowerShell Windows, gunakan bentuk panjang dari perintah ini —. eb platform

Menggunakan platform eb untuk platform khusus

Cantumkan versi platform saat ini dan memungkinkan Anda untuk mengelola platform khusus.

Sintaksis

eb platform create [***version***] [***options***]

eb platform delete [***version***] [***options***]

eb platform events [***version***] [***options***]

eb platform init [***platform***] [***options***]

eb platform list [***options***]

eb platform logs [***version***] [***options***]

eb platform status [***version***] [***options***]

eb platform use [***platform***] [***options***]

Opsi

Nama	Deskripsi
`create [version] [options]`	Bangun versi baru platform. Pelajari selengkapnya.
`delete version [options]`	Hapus versi platform. Pelajari selengkapnya.
`events [version] [options]`	Menampilkan peristiwa dari versi platform. Pelajari selengkapnya.
`init [platform] [options]`	Inisialisasi repositori platform. Pelajari selengkapnya.
`list [options]`	Cantumkan versi platform saat ini. Pelajari selengkapnya.

Nama	Deskripsi
`logs [version] [options]`	Tampilkan log dari lingkungan pembangun untuk versi platform. Pelajari selengkapnya.
`status [version] [options]`	Tampilkan status dari versi platform. Pelajari selengkapnya.
`use [platform] [options]`	Pilih platform yang berbeda dari versi baru yang dibangun. Pelajari selengkapnya.
Opsi umum	

Opsi umum

Semua perintah eb platform termasuk opsi umum berikut.

Nama	Deskripsi

`-h`
ATAU
`--help`
	Tampilkan pesan bantuan dan keluar.

`--debug`	Tampilkan output debugging tambahan.
`--quiet`	Tekan semua output.

`-v`
ATAU
`--verbose`
	Tampilkan output tambahan.

`--profile PROFILE`	Menggunakan yang ditentukan *PROFILE* dari kredensil Anda.
`-r REGION` ATAU	Gunakan wilayah tersebut*REGION*.

Nama	Deskripsi
`--region` *REGION*	
`--no-verify-ssl`	Jangan memverifikasi sertifikat AWS SSL.

Buat platform eb

Bangun versi baru dari platform dan kembalikan ARN untuk versi baru. Jika tidak ada lingkungan pembangun berjalan di wilayah saat ini, perintah ini meluncurkan satu. Opsi *`version`* dan kenaikan (`-M`,`-m`, dan`-p`) saling eksklusif.

Opsi

Nama	Penjelasan
`version`	Jika *`version`* tidak ditentukan, buat versi baru berdasarkan platform terbaru dengan versi patch (N di N.n.n) bertambah.

`-M`
ATAU
`--major-increment`
	Tambahkan nomor versi utama (N di N.n.n).

`-m`
ATAU
`--minor-increment`
	Tambahkan nomor versi minor (N di n.N.n).

`-p`
ATAU
`--patch-increment`
	Tambahkan nomor versi patch (N di n.n.N).

`-i` *INSTANCE_TYPE* ATAU	Gunakan *INSTANCE_TYPE* sebagai tipe instance, seperti**`t1.micro`**.

Nama	Penjelasan
--tipe-contoh *INSTANCE_TYPE*	
`-ip` *INSTANCE_PROFILE* ATAU `--instance-profile` *INSTANCE_PROFILE*	Gunakan *INSTANCE_PROFILE* sebagai profil instance saat membuat AMIs untuk platform khusus. Jika opsi `-ip` tidak ditentukan, buat profil instans `aws-elasticbeanstalk-custom-platforme-ec2-role` dan gunakan untuk platform khusus.
`--tags` *key1*=*value1*[,*ke*	Tanda ditentukan sebagai daftar yang dipisahkan koma pasangan `key=value` .
`--timeout` *minutes*	Atur jumlah menit sebelum waktu perintah habis.
`--vpc.id` *VPC_ID*	ID VPC tempat Packer dibangun.
`--vpc.subnets` *VPC_SUBNETS*	Subnet VPC tempat Packer membangun.
`--vpc.publicip`	Mengaitkan publik IPs ke EC2 instans diluncurkan.

Buat platform eb

Hapus versi platform. Versi tidak dihapus jika lingkungan menggunakan versi tersebut.

Opsi

Nama	Deskripsi
version	Versi untuk dihapus. Nilai ini diperlukan.
`--cleanup`	Hapus semua versi platform dengan status `Failed`.
`--all-platforms`	Jika `--cleanup` ditentukan, hapus semua versi platform dengan status `Failed` untuk semua platform.
`--force`	Konfirmasi tidak diperlukan saat menghapus versi.

Peristiwa platform eb

Menampilkan peristiwa dari versi platform. Jika *version* ditentukan, tampilkan peristiwa dari versi itu, jika tidak, tampilkan peristiwa dari versi saat ini.

Opsi

Nama	Penjelasan
version	Versi peristiwa yang ditampilkan. Nilai ini diperlukan.

`-f`
ATAU
`--follow`
	Lanjutkan untuk menampilkan peristiwa yang terjadi.

Unit platform eb

Inisialisasi repositori platform.

Opsi

Nama	Deskripsi
platform	Nama platform untuk menginisialisasi. Nilai ini diperlukan, kecuali `-i` (mode interaktif) diaktifkan.

`-i`
ATAU
`--interactive`
	Gunakan mode interaktif.

`-k` *KEYNAME* ATAU `--keyname` *KEYNAME*	Nama EC2 kunci default.

Anda dapat menjalankan perintah ini di direktori yang telah diinisialisasi sebelumnya, meskipun Anda tidak dapat mengubah tipe ruang kerja jika dijalankan di direktori yang sebelumnya telah diinisialisasi.

Untuk menginisialisasi ulang dengan opsi yang berbeda, gunakan opsi `-i`.

Daftar platform eb

Cantumkan versi platform yang terkait dengan ruang kerja (direktori) atau wilayah.

Perintah mengembalikan hasil yang berbeda tergantung pada tipe ruang kerja tempat Anda menjalankannya, sebagai berikut:

- Di ruang kerja platform (direktori yang diinisialisasi oleh `eb platform init`), perintah mengembalikan daftar semua versi platform dari platform khusus yang ditentukan di ruang kerja. Tambahkan opsi `--all-platforms` atau `--verbose` untuk mendapatkan daftar semua versi platform dari semua platform khusus yang dimiliki akun Anda di wilayah yang terkait dengan ruang kerja.
- Di ruang kerja aplikasi (direktori yang diinisialisasi oleh `eb init`), perintah mengembalikan daftar semua versi platform, baik untuk platform yang dikelola oleh Elastic Beanstalk dan untuk platform khusus akun Anda. Daftar ini menggunakan nama versi platform pendek, dan beberapa varian versi platform mungkin digabungkan. Tambahkan opsi `--verbose` untuk mendapatkan daftar detail dengan nama lengkap dan semua varian terdaftar secara terpisah.
- Di direktori yang tidak diinisialisasi, perintah hanya bekerja dengan opsi `--region`. Ini mengembalikan daftar semua versi platform yang dikelola Elastic Beanstalk yang didukung di wilayah tersebut. Daftar ini menggunakan nama versi platform pendek, dan beberapa varian versi platform mungkin digabungkan. Tambahkan opsi `--verbose` untuk mendapatkan daftar detail dengan nama lengkap dan semua varian terdaftar secara terpisah.

Opsi

Nama	Deskripsi

`-a`
 	
ATAU
 	
`--all-platforms`
	Hanya valid di ruang kerja terinisialisasi (direktori yang diinisialisasi oleh `eb platform init` atau `eb init`). Cantumkan versi platform dari semua platform khusus yang terkait dengan akun Anda.

Nama	Deskripsi

`-s STATUS`
 	
ATAU
 	
`--status STATUS`
	Cantumkan hanya platform yang cocok*STATUS*:
	
	• Siap
	• Gagal
	• Menghapus
	• Membuat

Log platform eb

Tampilkan log dari lingkungan pembangun untuk versi platform.

Opsi

Nama	Deskripsi
version	Versi platform yang ditampilkan log. Jika dihilangkan, tampilkan log dari versi saat ini.
`--stream`	Log penerapan streaming yang telah disiapkan dengan CloudWatch.

Status platform eb

Tampilkan status dari versi platform.

Opsi

Nama	Deskripsi
version	Versi platform yang statusnya diambil. Jika dihilangkan, tampilkan status dari versi saat ini.

Penggunaan platform eb

Pilih platform yang berbeda dari versi baru yang dibangun.

Opsi

Nama	Penjelasan
platform	Menentukan *platform* sebagai versi aktif untuk ruang kerja ini. Nilai ini diperlukan.

Menggunakan platform eb untuk lingkungan

Cantumkan platform yang didukung dan memungkinkan Anda untuk mengatur platform default dan versi platform untuk digunakan ketika Anda meluncurkan lingkungan. Gunakan eb platform list untuk melihat daftar semua platform yang didukung. Gunakan eb platform select untuk mengubah platform untuk proyek Anda. Gunakan eb platform show untuk melihat platform yang dipilih proyek Anda.

Sintaksis

eb platform list

eb platform select

eb platform show

Opsi

Nama	Deskripsi
`list`	Cantumkan versi platform saat ini.
`select`	Pilih platform default.
`show`	Tampilkan informasi tentang platform saat ini.

Contoh 1

Contoh berikut mencantumkan nama-nama semua konfigurasi untuk semua platform yang didukung Elastic Beanstalk.

```
$ eb platform list
```

```
docker-1.5.0
glassfish-4.0-java-7-(preconfigured-docker)
glassfish-4.1-java-8-(preconfigured-docker)
go-1.3-(preconfigured-docker)
go-1.4-(preconfigured-docker)
iis-7.5
iis-8
iis-8.5
multi-container-docker-1.3.3-(generic)
node.js
php-5.3
php-5.4
php-5.5
python
python-2.7
python-3.4
python-3.4-(preconfigured-docker)
ruby-1.9.3
ruby-2.0-(passenger-standalone)
ruby-2.0-(puma)
ruby-2.1-(passenger-standalone)
ruby-2.1-(puma)
ruby-2.2-(passenger-standalone)
ruby-2.2-(puma)
tomcat-6
tomcat-7
tomcat-7-java-6
tomcat-7-java-7
tomcat-8-java-8
```

Contoh 2

Contoh berikut meminta Anda untuk memilih dari daftar platform dan versi yang ingin Anda deploy untuk platform tertentu.

```
$ eb platform select
Select a platform.
1) PHP
2) Node.js
3) IIS
4) Tomcat
5) Python
6) Ruby
```

```
7) Docker
8) Multi-container Docker
9) GlassFish
10) Go
(default is 1): 5

Select a platform version.
1) Python 2.7
2) Python
3) Python 3.4 (Preconfigured - Docker)
```

Contoh 3

Contoh berikut menunjukkan informasi tentang platform default saat ini.

```
$ eb platform show
Current default platform: Python 2.7
New environments will be running:  64bit Amazon Linux 2014.09 v1.2.0 running Python 2.7

Platform info for environment "tmp-dev":
Current: 64bit Amazon Linux 2014.09 v1.2.0 running Python
Latest:  64bit Amazon Linux 2014.09 v1.2.0 running Python
```

eb printenv

Deskripsi

Cetak semua properti lingkungan di jendela perintah.

Sintaksis

eb printenv

eb printenv ***environment-name***

Opsi

Nama	Deskripsi
Opsi umum	

Output

Jika berhasil, perintah mengembalikan status operasi `printenv`.

Contoh

Contoh berikut mencetak properti lingkungan untuk lingkungan tertentu.

```
$ eb printenv
Environment Variables:
     PARAM1 = Value1
```

eb restore

Deskripsi

Bangun kembali lingkungan yang dihentikan, buat lingkungan baru dengan nama, ID, dan konfigurasi yang sama. Nama lingkungan, nama domain, dan versi aplikasi harus tersedia untuk digunakan agar pembangunan kembali berhasil.

Sintaksis

eb restore

eb restore ***environment_id***

Opsi

Nama	Deskripsi
Opsi umum	

Output

EB CLI menampilkan daftar lingkungan yang diakhiri yang tersedia untuk dipulihkan.

Contoh

```
$ eb restore
```

```
Select a terminated environment to restore

  #   Name          ID             Application Version      Date Terminated        Ago

  3   gamma         e-s7mimej8e9   app-77e3-161213_211138   2016/12/14 20:32 PST   13
 mins
  2   beta          e-sj28uu2wia   app-77e3-161213_211125   2016/12/14 20:32 PST   13
 mins
  1   alpha         e-gia8mphu6q   app-77e3-161213_211109   2016/12/14 16:21 PST   4
 hours

 (Commands: Quit, Restore, # #)

Selected environment alpha
Application:    scorekeep
Description:    Environment created from the EB CLI using "eb create"
CNAME:          alpha.h23tbtbm92.us-east-2.elasticbeanstalk.com
Version:        app-77e3-161213_211109
Platform:       64bit Amazon Linux 2016.03 v2.1.6 running Java 8
Terminated:     2016/12/14 16:21 PST
Restore this environment? [y/n]: y

2018-07-11 21:04:20    INFO: restoreEnvironment is starting.
2018-07-11 21:04:39    INFO: Created security group named: sg-e2443f72
...
```

eb scale

Deskripsi

Skalakan lingkungan agar selalu berjalan di sejumlah instans tertentu, mengatur kedua jumlah instans minimum dan maksimum ke jumlah yang ditentukan.

Sintaksis

eb scale ***number-of-instances***

eb scale ***number-of-instances environment-name***

Opsi

Nama	Deskripsi
--waktu habis	Jumlah menit sebelum waktu perintah habis.
Opsi umum	

Output

Jika berhasil, perintah memperbarui jumlah minimum dan maksimum instans untuk berjalan ke jumlah yang ditentukan.

Contoh

Contoh berikut menetapkan jumlah instans ke 2.

```
$ eb scale 2
2018-07-11 21:05:22    INFO: Environment update is starting.
2018-07-11 21:05:27    INFO: Updating environment tmp-dev's configuration settings.
2018-07-11 21:08:53    INFO: Added EC2 instance 'i-5fce3d53' to Auto Scaling Group
 'awseb-e-2cpfjbra9a-stack-AWSEBAutoScalingGroup-7AXY7U13ZQ6E'.
2018-07-11 21:08:58    INFO: Successfully deployed new configuration to environment.
2018-07-11 21:08:59    INFO: Environment update completed successfully.
```

eb setenv

Deskripsi

Atur properti lingkungan untuk lingkungan default.

Sintaks

eb setenv ***key*=*value***

Anda dapat menyertakan properti sebanyak yang Anda inginkan, tetapi ukuran total semua properti tidak dapat melebihi 4096 byte. Anda dapat menghapus variabel dengan membiarkan nilai kosong. Lihat Mengkonfigurasi properti lingkungan (variabel lingkungan) untuk batas.

> **Note**
>
> Jika `value` berisi karakter khusus, anda harus keluar dari karakter tersebut dengan mendahuluinya dengan karakter \.

Opsi

Nama	Deskripsi
--waktu habis	Jumlah menit sebelum waktu perintah habis.
Opsi umum	

Output

Jika berhasil, perintah menampilkan bahwa pembaruan lingkungan berhasil.

Contoh

Contoh berikut menetapkan variabel lingkungan ExampleVar.

```
$ eb setenv ExampleVar=ExampleValue
2018-07-11 21:05:25    INFO: Environment update is starting.
2018-07-11 21:05:29    INFO: Updating environment tmp-dev's configuration settings.
2018-07-11 21:06:50    INFO: Successfully deployed new configuration to environment.
2018-07-11 21:06:51    INFO: Environment update completed successfully.
```

Perintah berikut menetapkan beberapa properti lingkungan. Ini menambahkan properti lingkungan bernama `foo` dan menetapkan nilainya ke `bar`, mengubah nilai properti `JDBC_CONNECTION_STRING`, dan menghapus properti `PARAM4` dan `PARAM5`.

```
$ eb setenv foo=bar JDBC_CONNECTION_STRING=hello PARAM4= PARAM5=
```

eb ssh

Deskripsi

> **Note**
>
> Perintah ini tidak bekerja dengan lingkungan yang menjalankan instans Windows Server.

Connect ke EC2 instans Amazon Linux di lingkungan Anda menggunakan Secure Shell (SSH). Jika lingkungan memiliki beberapa instans berjalan, EB CLI meminta Anda untuk menentukan instans yang Anda ingin hubungkan. Untuk menggunakan perintah ini, SSH harus dipasang di mesin lokal Anda dan tersedia dari baris perintah. File kunci pribadi harus terletak di folder bernama `.ssh` di bawah direktori pengguna Anda, dan EC2 instance di lingkungan Anda harus memiliki alamat IP publik.

Jika direktori root berisi file `platform.yaml` yang menentukan platform khusus, perintah ini juga menghubungkan ke instans di lingkungan khusus.

> **Kunci SSH**
>
> Jika Anda belum mengonfigurasi SSH sebelumnya, Anda dapat menggunakan EB CLI untuk membuat kunci saat menjalankan eb init. Jika Anda sudah menjalankan eb init, jalankan lagi dengan opsi `--interactive` dan pilih Ya dan Buat Pasangan Kunci Baru saat diminta untuk mengatur SSH. Kunci yang dibuat selama proses ini akan disimpan di folder yang tepat oleh EB CLI.

Perintah ini sementara membuka port 22 di grup keamanan lingkungan Anda untuk lalu lintas masuk dari 0.0.0.0/0 (semua alamat IP) jika tidak ada aturan untuk port 22 sudah di tempat. Jika Anda telah mengonfigurasi grup keamanan lingkungan Anda untuk membuka port 22 ke kisaran CIDR terbatas untuk meningkatkan keamanan, EB CLI akan menghormati pengaturan tersebut dan melupakan perubahan apa pun pada grup keamanan. Untuk menimpa perilaku ini dan memaksa EB CLI untuk membuka port 22 untuk semua lalu lintas masuk, gunakan opsi `--force`.

Lihat EC2 kelompok keamanan untuk informasi tentang mengonfigurasi grup keamanan lingkungan Anda.

Sintaksis

eb ssh

eb ssh ***environment-name***

Opsi

Nama	Deskripsi

`-i`
atau
`--instance`
	Tentukan ID instans dari instans yang Anda hubungkan. Kami rekomendasikan agar Anda menggunakan opsi ini.

`-n`
atau
`--number`
	Tentukan instans untuk terhubung dengan nomor.

`-o`
atau
`--keep_open`
	Tinggalkan port 22 terbuka di grup keamanan setelah sesi SSH berakhir.

`--command`	Eksekusi perintah shell di instans tertentu, bukan memulai sesi SSH.
`--custom`	Tentukan perintah SSH untuk digunakan sebagai pengganti 'ssh -i keyfile'. Jangan sertakan pengguna jarak jauh dan nama host.
`--setup`	Ubah pasangan kunci yang ditugaskan untuk instans lingkungan (membutuhkan instans untuk diganti).
`--force`	Buka port 22 untuk lalu lintas masuk dari 0.0.0.0/0 di grup keamanan lingkungan, bahkan jika grup keamanan sudah dikonfigurasi untuk SSH.

Nama	Deskripsi
	Gunakan opsi ini jika grup keamanan lingkungan Anda dikonfigurasi untuk membuka port 22 ke kisaran CIDR terbatas yang tidak termasuk alamat IP yang Anda sedang coba hubungkan.
`--timeout` *minutes*	Atur jumlah menit sebelum waktu perintah habis. Hanya dapat digunakan dengan argumen `--setup`.
Opsi umum	

Output

Jika berhasil, perintah membuka koneksi SSH ke instans.

Contoh

Contoh berikut menghubungkan Anda ke lingkungan tertentu.

```
$ eb ssh
Select an instance to ssh into
1) i-96133799
2) i-5931e053
(default is 1): 1
INFO: Attempting to open port 22.
INFO: SSH port 22 open.
The authenticity of host '54.191.45.125 (54.191.45.125)' can't be established.
RSA key fingerprint is ee:69:62:df:90:f7:63:af:52:7c:80:60:1b:3b:51:a9.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '54.191.45.125' (RSA) to the list of known hosts.

       __|  __|_  )
       _|  (     /   Amazon Linux AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-ami/2014.09-release-notes/
No packages needed for security; 1 packages available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-8-185 ~]$ ls
[ec2-user@ip-172-31-8-185 ~]$ exit
```

```
logout
Connection to 54.191.45.125 closed.
INFO: Closed port 22 on ec2 instance security group
```

eb status

Deskripsi

Berikan informasi tentang status lingkungan.

Jika direktori root berisi file `platform.yaml` yang menentukan platform khusus, perintah ini juga memberikan informasi tentang lingkungan pembangun.

Sintaksis

eb status

eb status ***environment-name***

Opsi

Nama	Deskripsi

`-v`
 	
atau
 	
`--verbose`
	Berikan informasi selengkapnya tentang instans individu, seperti statusnya dengan penyeimbang beban Elastic Load Balancing.

Opsi umum	

Output

Jika berhasil, perintah mengembalikan informasi berikut tentang lingkungan:

- Nama lingkungan
- Nama aplikasi
- Versi aplikasi yang di-deploy
- ID Lingkungan

- Platform
- Tingkat lingkungan
- CNAME
- Waktu lingkungan terakhir diperbarui
- Status
- Kondisi

Jika Anda menggunakan mode verbose, EB CLI juga memberi Anda jumlah instans Amazon yang berjalan. EC2

Contoh

Contoh berikut menunjukkan status lingkungan tmp-dev.

```
$ eb status
Environment details for: tmp-dev
  Application name: tmp
  Region: us-west-2
  Deployed Version: None
  Environment ID: e-2cpfjbra9a
  Platform: 64bit Amazon Linux 2014.09 v1.0.9 running PHP 5.5
  Tier: WebServer-Standard-1.0
  CNAME: tmp-dev.elasticbeanstalk.com
  Updated: 2014-10-29 21:37:19.050000+00:00
  Status: Launching
  Health: Grey
```

eb swap

Deskripsi

Tukar CNAME lingkungan dengan CNAME lingkungan lain (misalnya, untuk menghindari waktu habis ketika Anda memperbarui versi aplikasi Anda).

> **Note**
>
> Jika Anda memiliki lebih dari dua lingkungan, Anda diminta untuk memilih nama lingkungan yang saat ini menggunakan CNAME yang Anda diinginkan dari daftar lingkungan.

Untuk menekan ini, Anda dapat menentukan nama lingkungan untuk digunakan dengan menyertakan opsi `-n` ketika Anda menjalankan perintah.

Sintaks

eb swap

eb swap ***environment-name***

> **Note**
>
> *environment-name*Ini adalah lingkungan di mana Anda menginginkan CNAME yang berbeda. Jika Anda tidak menentukan *environment-name* sebagai parameter baris perintah saat Anda menjalankaneb swap, EB CLI memperbarui CNAME dari lingkungan default.

Opsi

Nama	Deskripsi

`-n`
atau
`--destination_name`
	Menentukan nama lingkungan yang ingin Anda swap CNAMEs. Jika Anda menjalankan eb swap tanpa opsi ini, maka EB CLI meminta Anda untuk memilih dari daftar lingkungan Anda.

Opsi umum	

Output

Jika berhasil, perintah mengembalikan status operasi `swap`.

Contoh

Contoh berikut menukar lingkungan tmp-dev dengan live-env.

```
$ eb swap
Select an environment to swap with.
1) staging-dev
2) live-env
(default is 1): 2
2018-07-11 21:05:25    INFO: swapEnvironmentCNAMEs is starting.
2018-07-11 21:05:26    INFO: Swapping CNAMEs for environments 'tmp-dev' and 'live-env'.
2018-07-11 21:05:30    INFO: 'tmp-dev.elasticbeanstalk.com' now points to 'awseb-e-j-
AWSEBLoa-M7U21VXNLWHN-487871449.us-west-2.elb.amazonaws.com'.
2018-07-11 21:05:30    INFO: Completed swapping CNAMEs for environments 'tmp-dev' and
 'live-env'.
```

Contoh berikut menukar lingkungan tmp-dev dengan lingkungan live-env, tetapi tidak meminta Anda untuk memasukkan atau memilih nilai untuk pengaturan apapun.

```
$ eb swap tmp-dev --destination_name live-env
2018-07-11 21:18:12    INFO: swapEnvironmentCNAMEs is starting.
2018-07-11 21:18:13    INFO: Swapping CNAMEs for environments 'tmp-dev' and 'live-env'.
2018-07-11 21:18:17    INFO: 'tmp-dev.elasticbeanstalk.com' now points to 'awseb-e-j-
AWSEBLoa-M7U21VXNLWHN-487871449.us-west-2.elb.amazonaws.com'.
2018-07-11 21:18:17    INFO: Completed swapping CNAMEs for environments 'tmp-dev' and
 'live-env'.
```

eb tags

Deskripsi

Tambahkan, hapus, perbarui, dan cantumkan tanda sumber daya Elastic Beanstalk.

Untuk detail tentang penandaan sumber daya di Elastic Beanstalk, lihat Pelabelan sumber daya aplikasi Elastic Beanstalk.

Sintaks

eb tags [***environment-name***] [--resource ***ARN***] -l | --list

eb tags [***environment-name***] [--resource ***ARN***] -a | --add ***key1***=***value1***[,***key2***=***value2*** ...]

eb tags [***environment-name***] [--resource ***ARN***] -u | --update ***key1***=***value1***[,***key2***=***value2*** ...]

eb tags [*environment-name*] [--resource *ARN*] -d | --delete *key1*[,*key2* ...]

Anda dapat menggabungkan opsi sub perintah `--add`, `--update`, dan `--delete` dalam satu perintah. Setidaknya satu dari mereka diperlukan. Anda tidak dapat menggabungkan salah satu dari tiga opsi sub perintah ini dengan `--list`.

Tanpa argumen tambahan, semua perintah ini mencantumkan atau memodifikasi tanda dari lingkungan default di aplikasi direktori saat ini. Dengan *environment-name* argumen, perintah mencantumkan atau memodifikasi tag lingkungan itu. Dengan opsi `--resource`, perintah mencantumkan atau memodifikasi tanda dari sumber daya Elastic Beanstalk – aplikasi, lingkungan, versi aplikasi, konfigurasi tersimpan, atau versi platform khusus. Tentukan sumber daya dengan Amazon Resource Name (ARN)-nya.

Opsi

Tak satu pun dari opsi ini diperlukan. Jika Anda menjalankan eb create tanpa opsi apa pun, Anda diminta untuk memasukkan atau memilih nilai untuk setiap pengaturan.

Nama	Deskripsi

`-l`
atau
`--list`
	Cantumkan semua tanda yang saat ini diterapkan ke sumber daya.

`-a` *key1*`=`*value1*`[,`*key2*`=`*value2* atau `--add` *key1*`=`*value1*`[,`*key2*`=`*val*	Terapkan tanda baru ke sumber daya. Tentukan tanda sebagai daftar yang dipisahkan koma pasangan `key=value` . Anda tidak dapat menentukan kunci dari tanda yang ada. Nilai valid: Lihat Pelabelan sumber daya.
`-u` *key1*`=`*value1*`[,`*key2*`=`*value2* atau `--updat e` *key1*`=`*value1*`[,`*key2*`=`*value2* .	Perbarui nilai tanda sumber daya yang ada. Tentukan tanda sebagai daftar yang dipisahkan koma pasangan `key=value` . Anda harus menentukan kunci dari tanda yang ada. Nilai valid: Lihat Pelabelan sumber daya.

Nama	Deskripsi

`-d` *key1*`[,`*key2* `...]`
 	
atau
 	
`--delete` *key1*`[,`*key2* `...]`
	Hapus tanda sumber daya yang ada. Tentukan tanda sebagai daftar kunci yang dipisahkan koma pasangan. Anda harus menentukan kunci dari tanda yang ada.
	
	Nilai valid: Lihat Pelabelan sumber daya.

`-r` *region* atau `--region` *region*	Wilayah AWS Di mana sumber daya Anda ada. Default: wilayah default yang dikonfigurasi. Untuk daftar nilai yang dapat Anda tentukan untuk opsi ini, lihat AWS Elastic Beanstalk Titik akhir dan Kuota di. Referensi Umum AWS
`--resource` *ARN*	ARN sumber daya yang dimodifikasi atau dicantumkan tandanya oleh perintah. Jika tidak ditentukan, perintah mengacu ke lingkungan (default atau ditentukan) di aplikasi direktori saat ini. Nilai valid: Lihat salah satu sub topik Pelabelan sumber daya yang spesifik dengan sumber daya yang Anda minati. Topik-topik ini menunjukkan bagaimana ARN sumber daya dibangun dan menjelaskan cara mendapatk an daftar sumber daya ini ARNs yang ada untuk aplikasi atau akun Anda.

Output

Opsi sub perintah `--list` menampilkan daftar tanda sumber daya. Output menunjukkan kedua tanda yang Elastic Beanstalk terapkan secara default dan tanda khusus Anda.

```
$ eb tags --list
Showing tags for environment 'MyApp-env':

Key                                 Value

Name                                MyApp-env
```

```
elasticbeanstalk:environment-id     e-63cmxwjaut
elasticbeanstalk:environment-name   MyApp-env
mytag                               tagvalue
tag2                                2nd value
```

Opsi sub perintah `--add`, `--update`, dan `--delete`, ketika berhasil, tidak memiliki output apapun. Anda dapat menambahkan opsi `--verbose` untuk melihat output detail dari aktivitas perintah.

```
$ eb tags --verbose --update "mytag=tag value"
Updated Tags:

Key                                 Value

mytag                               tag value
```

Contoh

Perintah berikut berhasil menambahkan tanda dengan kunci `tag1` dan nilai `value1` ke lingkungan default aplikasi, dan pada saat yang sama menghapus tanda `tag2`.

```
$ eb tags --add tag1=value1 --delete tag2
```

Perintah berikut berhasil menambahkan tanda ke konfigurasi tersimpan di aplikasi.

```
$ eb tags --add tag1=value1 \
      --resource "arn:aws:elasticbeanstalk:us-east-2:my-account-
id:configurationtemplate/my-app/my-template"
```

Perintah berikut gagal karena mencoba untuk memperbarui tanda yang tidak ada.

```
$ eb tags --update tag3=newval
ERROR: Tags with the following keys can't be updated because they don't exist:

  tag3
```

Perintah berikut gagal karena mencoba untuk memperbarui dan menghapus kunci yang sama.

```
$ eb tags --update mytag=newval --delete mytag
ERROR: A tag with the key 'mytag' is specified for both '--delete' and '--update'. Each
 tag can be either deleted or updated in a single operation.
```

eb terminate

Deskripsi

Mengakhiri lingkungan berjalan sehingga Anda tidak dikenakan biaya untuk sumber daya yang tidak AWS digunakan.

Menggunakan opsi `--all`, hapus aplikasi yang direktori saat ini diinisialisasi untuk menggunakan eb init. Perintah mengakhiri semua lingkungan dalam aplikasi. Ini juga mengakhiri versi aplikasi dan konfigurasi yang disimpan untuk aplikasi, dan kemudian menghapus aplikasi.

Jika direktori root berisi file `platform.yaml` yang menentukan platform khusus, perintah ini mengakhiri lingkungan khusus yang sedang berjalan.

> **Note**
>
> Anda selalu dapat meluncurkan lingkungan baru menggunakan versi yang sama nantinya.

Jika Anda memiliki data dari lingkungan yang ingin Anda pertahankan, setel kebijakan penghapusan database `Retain` sebelum mengakhiri lingkungan. Ini membuat database tetap beroperasi di luar Elastic Beanstalk. Setelah ini, setiap lingkungan Elastic Beanstalk harus terhubung dengannya sebagai database eksternal. Jika Anda ingin mencadangkan data tanpa menjaga database tetap beroperasi, setel kebijakan penghapusan untuk mengambil snapshot database sebelum mengakhiri lingkungan. Untuk informasi selengkapnya, lihat Siklus hidup basis data di bagian Mengkonfigurasi lingkungan dari panduan ini.

> **Important**
>
> Jika Anda mengakhiri lingkungan, Anda juga harus menghapus pemetaan CNAME yang Anda buat, karena pelanggan lain dapat menggunakan kembali nama host yang tersedia. Pastikan untuk menghapus catatan DNS yang mengarah ke lingkungan Anda yang dihentikan untuk mencegah entri DNS yang menggantung. Entri DNS yang menggantung dapat mengekspos lalu lintas internet yang ditujukan untuk domain Anda ke kerentanan keamanan. Itu juga dapat menimbulkan risiko lain.
> Untuk informasi selengkapnya, lihat Perlindungan dari catatan delegasi yang menggantung di Route 53 di Panduan Pengembang Amazon Route 53. Anda juga dapat mempelajari

selengkapnya tentang menggantung entri DNS di Perlindungan Domain yang Ditingkatkan untuk Permintaan CloudFront Amazon di AWS Blog Keamanan.

Sintaksis

eb terminate

eb terminate ***environment-name***

Opsi

Nama	Deskripsi
`--all`	Akhiri semua lingkungan di aplikasi, versi aplikasi aplikasi, dan konfigurasi tersimpan, dan kemudian hapus aplikasi.
`--force`	Mengakhiri lingkungan tanpa meminta konfirmasi.
`--ignore-links`	Mengakhiri lingkungan bahkan jika ada lingkungan dependen dengan tautan ke sana. Lihat Penyusunan Lingkungan.
`--timeout`	Jumlah menit sebelum waktu perintah habis.

Output

Jika berhasil, perintah mengembalikan status operasi `terminate`.

Contoh

Contoh permintaan berikut mengakhiri lingkungan tmp-dev.

```
$ eb terminate
The environment "tmp-dev" and all associated instances will be terminated.
To confirm, type the environment name: tmp-dev
2018-07-11 21:05:25    INFO: terminateEnvironment is starting.
2018-07-11 21:05:40    INFO: Deleted CloudWatch alarm named: awseb-e-2cpfjbra9a-stack-
AWSEBCloudwatchAlarmHigh-16V08YOF2KQ7U
```

```
2018-07-11 21:05:41    INFO: Deleted CloudWatch alarm named: awseb-e-2cpfjbra9a-stack-
AWSEBCloudwatchAlarmLow-6ZAWH9F20P7C
2018-07-11 21:06:42    INFO: Deleted Auto Scaling group policy named:
 arn:aws:autoscaling:us-east-2:11122223333:scalingPolicy:5d7d3e6b-
d59b-47c5-b102-3e11fe3047be:autoScalingGroupName/awseb-e-2cpfjbra9a-stack-
AWSEBAutoScalingGroup-7AXY7U13ZQ6E:policyName/awseb-e-2cpfjbra9a-stack-AWSEBAutoSca
lingScaleUpPolicy-1876U27JEC34J
2018-07-11 21:06:43    INFO: Deleted Auto Scaling group policy named:
 arn:aws:autoscaling:us-east-2:11122223333:scalingPolicy:29c6e7c7-7ac8-46fc-91f5-
cfabb65b985b:autoScalingGroupName/awseb-e-2cpfjbra9a-stack-
AWSEBAutoScalingGroup-7AXY7U13ZQ6E:policyName/awseb-e-2cpfjbra9a-stack-AWSEBAutoSca
lingScaleDownPolicy-SL4LHODMOMU
2018-07-11 21:06:48    INFO: Waiting for EC2 instances to terminate. This may take a
 few minutes.
2018-07-11 21:08:55    INFO: Deleted Auto Scaling group named: awseb-e-2cpfjbra9a-
stack-AWSEBAutoScalingGroup-7AXY7U13ZQ6E
2018-07-11 21:09:10    INFO: Deleted security group named: awseb-e-2cpfjbra9a-stack-
AWSEBSecurityGroup-XT4YYGFL7I99
2018-07-11 21:09:40    INFO: Deleted load balancer named: awseb-e-2-AWSEBLoa-
AK6RRYFQVV3S
2018-07-11 21:09:42    INFO: Deleting SNS topic for environment tmp-dev.
2018-07-11 21:09:52    INFO: terminateEnvironment completed successfully.
```

eb upgrade

Deskripsi

Tingkatkan platform lingkungan Anda ke versi terbaru dari platform yang saat ini sedang berjalan.

Jika direktori root berisi file `platform.yaml` yang menentukan platform khusus, perintah ini meningkatkan lingkungan ke versi terbaru dari platform khusus yang saat ini sedang berjalan.

Sintaksis

eb upgrade

eb upgrade ***environment-name***

Opsi

Nama	Deskripsi
`--force`	Peningkatan tanpa mengharuskan Anda mengonfirmasi nama lingkungan sebelum memulai proses peningkatan.
`--noroll`	Perbarui semua instans tanpa menggunakan pembaruan bergulir untuk menjaga beberapa instans di layanan selama peningkatan.
Opsi umum	

Output

Perintah menunjukkan gambaran umum perubahan dan meminta Anda untuk mengonfirmasi peningkatan dengan mengetik nama lingkungan. Jika berhasil, lingkungan Anda diperbarui dan kemudian diluncurkan dengan versi terbaru dari platform.

Contoh

Contoh berikut meningkatkan versi platform saat ini dari lingkungan tertentu untuk versi platform yang paling baru tersedia.

```
$ eb upgrade
Current platform: 64bit Amazon Linux 2014.09 v1.0.9 running Python 2.7
Latest platform:  64bit Amazon Linux 2014.09 v1.2.0 running Python 2.7

WARNING: This operation replaces your instances with minimal or zero downtime. You may
 cancel the upgrade after it has started by typing "eb abort".
You can also change your platform version by typing "eb clone" and then "eb swap".

To continue, type the environment name:
```

eb use

Deskripsi

Tetapkan lingkungan yang ditentukan sebagai lingkungan default.

Saat menggunakan Git, eb use menetapkan lingkungan default untuk cabang saat ini. Jalankan perintah ini sekali di setiap cabang yang ingin Anda deploy ke Elastic Beanstalk.

Sintaksis

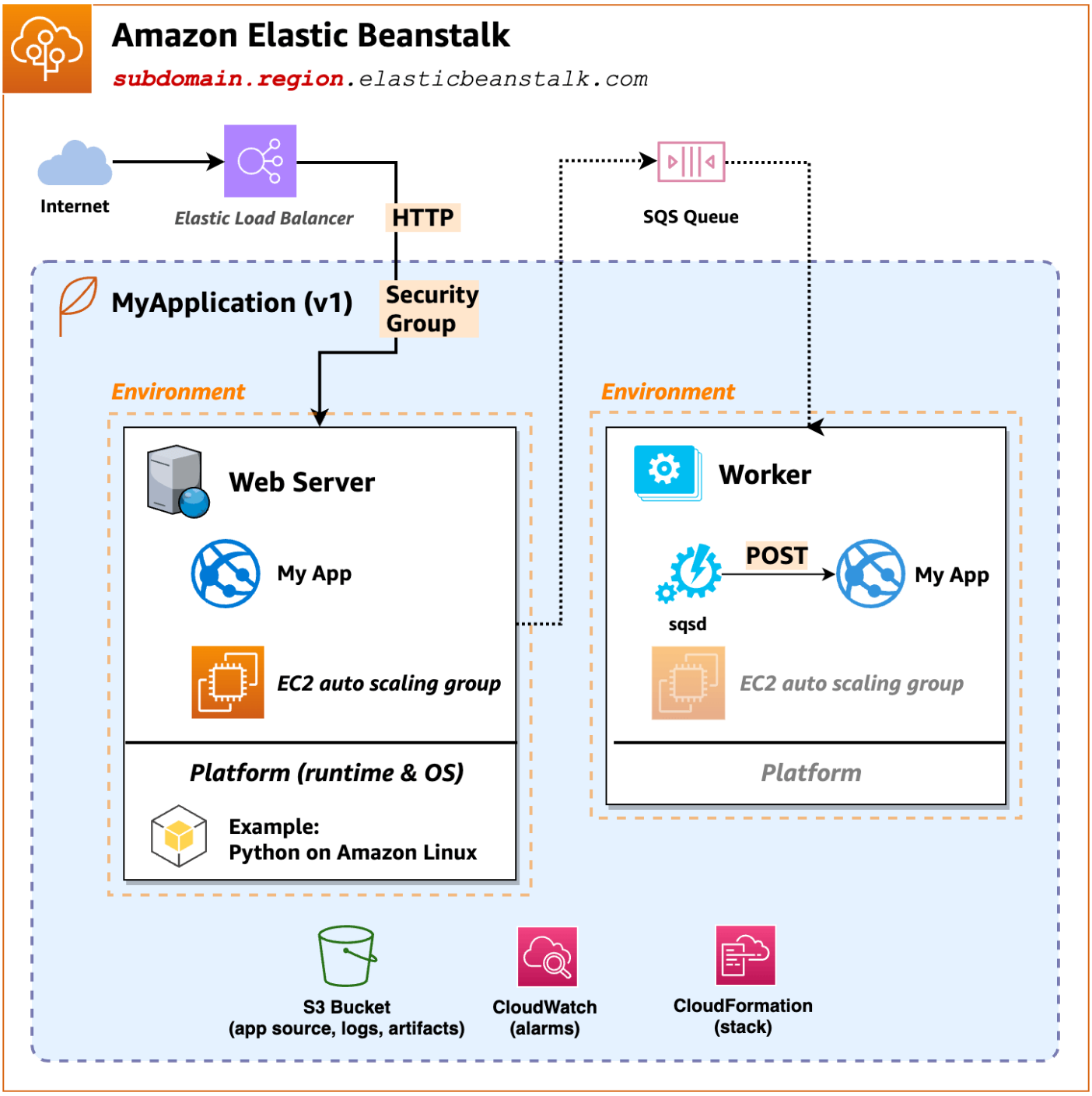
eb use ***environment-name***

Opsi

Nama	Penjelasan
`--source codecommit/` *repository-name*`/`*branch-name*	CodeCommit repositori dan cabang.
`-r` *region* `--region` *region*	Ubah wilayah tempat Anda membuat lingkungan.
Opsi umum	

Memahami konsep dalam Elastic Beanstalk

Menjadi akrab dengan konsep dan istilah akan membantu Anda mendapatkan pemahaman yang diperlukan untuk menerapkan aplikasi Anda dengan Elastic Beanstalk.



Aplikasi

Aplikasi Elastic Beanstalk adalah wadah untuk komponen Elastic Beanstalk, termasuk lingkungan, versi, dan konfigurasi lingkungan. Dalam aplikasi Elastic Beanstalk, Anda mengelola semua sumber daya yang relevan untuk menjalankan kode Anda.

Versi aplikasi

Dalam Elastic Beanstalk, sebuah versi aplikasi mengacu pada spesifik, iterasi berlabel kode deployable untuk aplikasi web. Versi aplikasi menunjuk ke sebuah objek Amazon Simple Storage Service (Amazon S3) yang berisi kode deployable, seperti file Java WAR.

Versi aplikasi adalah bagian dari aplikasi. Aplikasi dapat memiliki banyak versi dan setiap versi aplikasi adalah unik. Dalam lingkungan yang sedang berlangsung, Anda dapat menyebarkan versi aplikasi apa pun yang sudah Anda unggah ke aplikasi, atau Anda dapat mengunggah dan segera menyebarkan versi aplikasi baru. Misalnya, Anda dapat mengunggah beberapa versi aplikasi untuk menguji perbedaan di antara keduanya.

Lingkungan

Lingkungan adalah kumpulan AWS sumber daya yang menjalankan versi aplikasi. Setiap lingkungan hanya berlangsung satu versi aplikasi pada satu waktu, namun, Anda dapat menjalankan versi aplikasi yang sama atau versi aplikasi yang berbeda di banyak lingkungan secara bersamaan. Saat Anda membuat lingkungan, Elastic Beanstalk menyediakan sumber daya yang dibutuhkan AWS di akun Anda untuk menjalankan versi aplikasi yang Anda tentukan.

Tingkat lingkungan

Ketika Anda meluncurkan lingkungan Elastic Beanstalk, Anda terlebih dahulu memilih tingkat lingkungan. Tingkat lingkungan menunjuk jenis aplikasi yang dijalankan lingkungan dan menentukan sumber daya apa ketentuan Elastic Beanstalk untuk mendukungnya. Sebuah aplikasi yang melayani permintaan HTTP berjalan di Tingkat lingkungan server web. Lingkungan backend yang menarik tugas dari antrean Amazon Simple Queue Service (Amazon SQS) berjalan di tingkat lingkungan pekerja.

Konfigurasi lingkungan

Konfigurasi lingkungan mengidentifikasi kumpulan parameter dan pengaturan yang menentukan bagaimana lingkungan dan sumber daya yang terkait berperilaku. Ketika Anda memperbarui pengaturan konfigurasi lingkungan, Elastic Beanstalk secara otomatis menerapkan perubahan pada sumber daya yang ada atau menghapus dan menyebarkan sumber daya baru (tergantung pada jenis perubahan).

Konfigurasi tersimpan

sebuah konfigurasi yang disimpan adalah template yang dapat Anda gunakan sebagai titik awal untuk membuat konfigurasi lingkungan yang unik. Anda dapat membuat dan memodifikasi konfigurasi yang disimpan, dan menerapkannya ke lingkungan, menggunakan konsol Elastic Beanstalk, EB CLI, atau API. AWS CLI API dan AWS CLI merujuk ke konfigurasi yang disimpan sebagai templat konfigurasi.

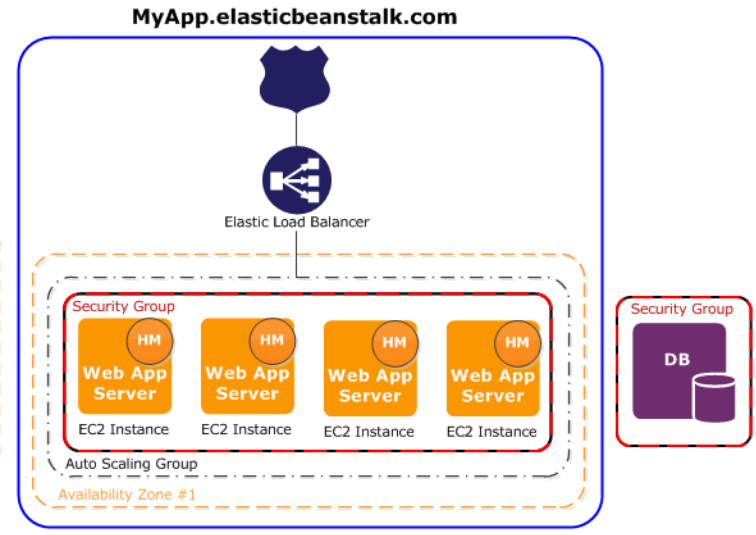
Platform

Sebuah platform adalah kombinasi dari sistem operasi, pemrograman bahasa runtime, web server, server aplikasi, dan komponen Elastic Beanstalk. Anda merancang dan menargetkan aplikasi web Anda ke sebuah platform. Elastic Beanstalk menyediakan berbagai platform di mana Anda dapat membangun aplikasi Anda.

Untuk detailnya, lihat Platform Elastic Beanstalk.

Lingkungan server web Elastic Beanstalk

Diagram berikut menunjukkan contoh arsitektur Elastic Beanstalk untuk tingkat lingkungan server web, dan menunjukkan cara komponen pada jenis lingkungan tersebut bekerja sama.



Lingkungan adalah nyawa dari aplikasi. Dalam diagram, lingkungan ditunjukkan dalam garis solid yang ada di posisi teratas. Saat Anda membuat lingkungan, Elastic Beanstalk menyediakan sumber daya yang diperlukan untuk menjalankan aplikasi Anda. AWS sumber daya yang dibuat untuk suatu lingkungan mencakup satu penyeimbang beban elastis (ELB dalam diagram), grup Auto Scaling, dan satu atau beberapa instans Amazon Elastic Compute Cloud (EC2Amazon).

Setiap lingkungan memiliki CNAME (URL) yang mengarah ke penyeimbang beban. Lingkungan memiliki URL, seperti `myapp.us-west-2.elasticbeanstalk.com`. URL ini dialiaskan di Amazon Route 53 ke URL Elastic Load Balancing—sesuatu seperti `abcdef-123456.us-west-2.elb.amazonaws.com`—dengan menggunakan catatan CNAME. Amazon Route 53 adalah layanan web Domain Name System (DNS) yang dapat diskalakan dan sangat tersedia. Amazon Route 53 menyediakan perutean yang aman dan dapat diandalkan untuk infrastruktur Anda. Nama domain yang Anda daftarkan dengan menggunakan penyedia DNS Anda akan meneruskan permintaan ke CNAME.

Load balancer berada di depan EC2 instans Amazon, yang merupakan bagian dari grup Auto Scaling. EC2 Auto Scaling Amazon secara otomatis memulai EC2 instans Amazon tambahan untuk mengakomodasi peningkatan beban pada aplikasi Anda. Jika pemuatan pada aplikasi Anda berkurang, Amazon EC2 Auto Scaling menghentikan instance, tetapi setidaknya satu instance tetap berjalan.

Tumpukan perangkat lunak yang berjalan pada EC2 instance Amazon bergantung pada jenis kontainer. Jenis kontainer menentukan topologi infrastruktur dan tumpukan perangkat lunak yang akan digunakan untuk lingkungan itu. Misalnya, lingkungan Elastic Beanstalk dengan kontainer Apache Tomcat menggunakan sistem pengoperasian Amazon Linux, server web Apache, dan

perangkat lunak Apache Tomcat. Untuk daftar jenis kontainer yang didukung, lihat Platform yang didukung Elastic Beanstalk. Setiap EC2 instans Amazon yang menjalankan aplikasi Anda menggunakan salah satu jenis penampung ini. Selain itu, komponen perangkat lunak yang disebut host manager (HM) berjalan pada setiap EC2 instans Amazon. Manajer host bertanggung jawab untuk hal berikut:

- Men-deploy aplikasi
- Menggabungkan peristiwa dan metrik untuk mendapatkannya kembali melalui konsol, API, atau baris perintah
- Menghasilkan peristiwa tingkat instans
- Memantau berkas log aplikasi untuk kesalahan kritis
- Memantau server aplikasi
- Patch komponen instans
- Memutar berkas log aplikasi Anda dan mempublikasikannya ke Amazon S3

Manajer host melaporkan metrik, kesalahan dan peristiwa, dan status instance server, yang tersedia melalui konsol Elastic APIs Beanstalk, dan. CLIs

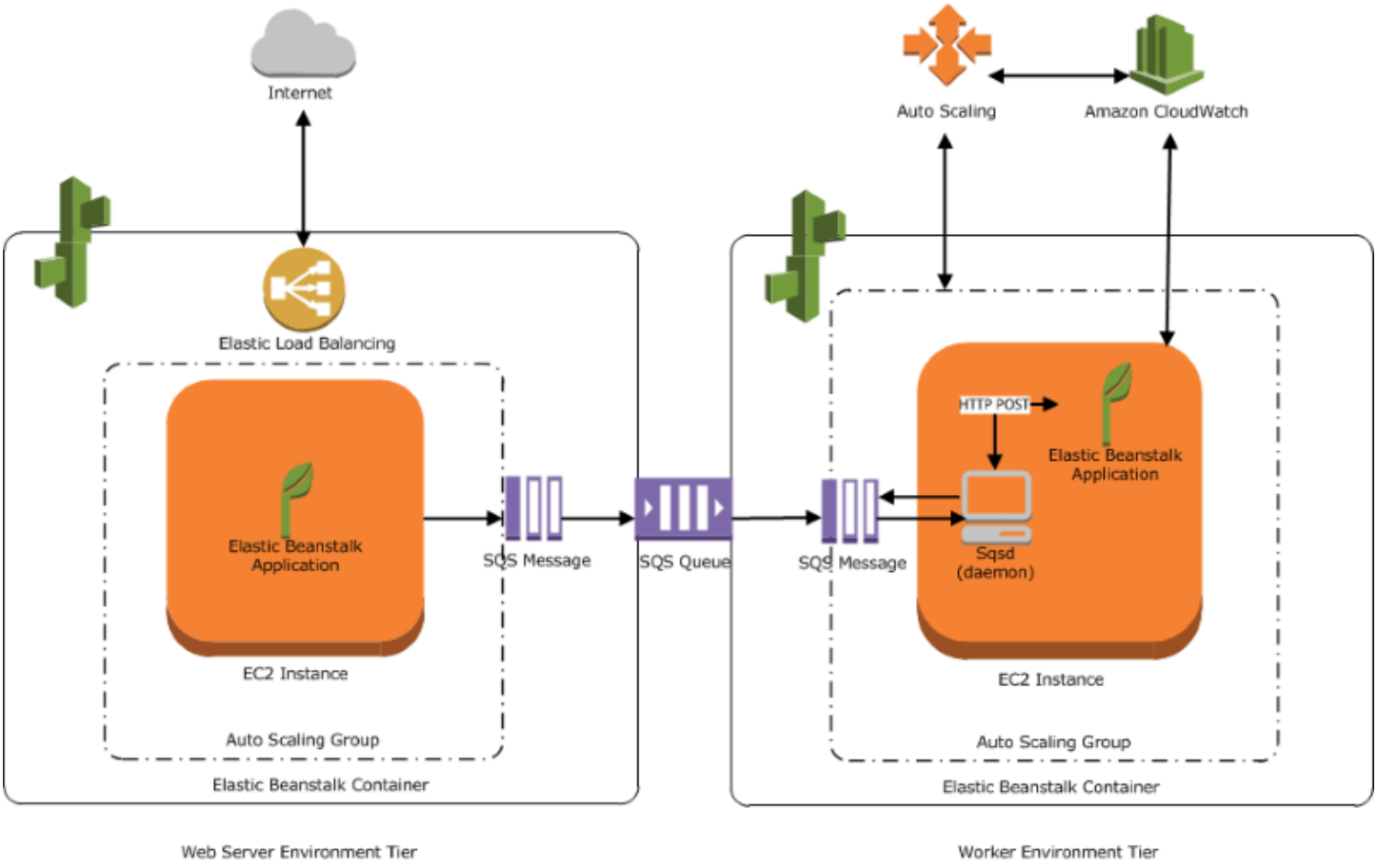
EC2 Contoh Amazon yang ditunjukkan dalam diagram adalah bagian dari satu grup keamanan. Grup keamanan menentukan aturan firewall untuk instans Anda. Secara default, Elastic Beanstalk menentukan grup keamanan, yang mengizinkan setiap orang terhubung menggunakan port 80 (HTTP). Anda dapat menentukan lebih dari satu grup keamanan. Misalnya, Anda dapat menentukan grup keamanan untuk server basis data Anda. Untuk informasi selengkapnya tentang grup EC2 keamanan Amazon dan cara mengonfigurasinya untuk aplikasi Elastic Beanstalk Anda, lihat. EC2 kelompok keamanan

Lingkungan pekerja Elastic Beanstalk

AWS resource yang dibuat untuk tingkat lingkungan pekerja mencakup grup Auto Scaling, satu atau EC2 beberapa instans Amazon, dan peran IAM. Untuk tingkat lingkungan pekerja, Elastic Beanstalk juga membuat dan menyediakan antrean Amazon SQS jika Anda belum memilikinya. Saat Anda meluncurkan lingkungan pekerja, Elastic Beanstalk menginstal file dukungan yang diperlukan untuk bahasa pemrograman pilihan Anda dan daemon pada EC2 setiap instance dalam grup Auto Scaling. Daemon membaca pesan dari antrean Amazon SQS. Daemon mengirimkan data dari setiap pesan yang dibacanya ke aplikasi web yang berjalan di lingkungan pekerja untuk diproses. Jika Anda

memiliki beberapa instans di lingkungan pekerja Anda, setiap instans memiliki daemon sendiri, tetapi mereka semua membaca dari antrean Amazon SQS yang sama.

Diagram berikut menunjukkan komponen yang berbeda dan interaksinya di seluruh lingkungan dan AWS layanan.



Amazon CloudWatch digunakan untuk alarm dan pemantauan kesehatan. Untuk informasi selengkapnya, kunjungi Pelaporan kondisi dasar.

Untuk informasi detail tentang cara tingkat lingkungan pekerja bekerja, lihat Lingkungan pekerja Elastic Beanstalk.

Pertimbangan desain untuk aplikasi Elastic Beanstalk Anda

Karena aplikasi yang digunakan menggunakan AWS Cloud sumber daya AWS Elastic Beanstalk berjalan, Anda harus mengingat beberapa faktor konfigurasi untuk mengoptimalkan aplikasi Anda: skalabilitas, keamanan, penyimpanan persisten, toleransi kesalahan, pengiriman konten, pembaruan dan penambalan perangkat lunak, dan konektivitas. Masing-masing dibahas secara terpisah dalam

topik ini. Untuk daftar lengkap AWS whitepaper teknis, yang mencakup topik-topik seperti arsitektur, serta keamanan dan ekonomi, lihat Whitepaper AWS Cloud Computing.

Skalabilitas

Saat beroperasi di lingkungan perangkat keras fisik, berbeda dengan lingkungan cloud, Anda dapat mendekati skalabilitas dengan salah satu dari dua cara. Entah Anda dapat meningkatkan skala melalui penskalaan vertikal atau Anda dapat menskalakan melalui penskalaan horizontal. Pendekatan scale-up mengharuskan Anda berinvestasi dalam perangkat keras yang kuat, yang dapat mendukung meningkatnya permintaan bisnis Anda. Pendekatan scale-out mengharuskan Anda mengikuti model investasi terdistribusi. Dengan demikian, akuisisi perangkat keras dan aplikasi Anda dapat lebih ditargetkan, kumpulan data Anda digabungkan, dan desain Anda berorientasi pada layanan. Pendekatan scale-up bisa mahal, dan ada juga risiko bahwa permintaan Anda bisa melebihi kapasitas Anda. Dalam hal ini, pendekatan scale-out biasanya lebih efektif. Namun, saat menggunakannya, Anda harus dapat memprediksi permintaan secara berkala dan menyebarkan infrastruktur dalam potongan untuk memenuhi permintaan itu. Akibatnya, pendekatan ini sering dapat menyebabkan kapasitas yang tidak digunakan dan mungkin memerlukan pemantauan yang cermat.

Dengan bermigrasi ke cloud, Anda dapat membuat infrastruktur Anda selaras dengan permintaan dengan memanfaatkan elastisitas cloud. Elastisitas membantu merampingkan akuisisi dan pelepasan sumber daya. Dengan itu, infrastruktur Anda dapat dengan cepat masuk dan skala keluar saat permintaan berfluktuasi. Untuk menggunakannya, konfigurasikan pengaturan Auto Scaling Anda untuk meningkatkan atau menurunkan skala berdasarkan metrik sumber daya di lingkungan Anda. Misalnya, Anda dapat mengatur metrik seperti pemanfaatan server atau jaringan I/O. Anda dapat menggunakan Auto Scaling agar kapasitas komputasi ditambahkan secara otomatis setiap kali penggunaan meningkat dan untuk dihapus setiap kali penggunaan turun. Anda dapat mempublikasikan metrik sistem (misalnya, CPU, memori, disk I/O, dan I/O jaringan) ke Amazon. CloudWatch Kemudian, Anda dapat menggunakan CloudWatch untuk mengonfigurasi alarm untuk memicu tindakan Auto Scaling atau mengirim notifikasi berdasarkan metrik ini. Untuk petunjuk tentang cara mengonfigurasi Auto Scaling, lihat. Auto Scaling instance lingkungan Elastic Beanstalk Anda

Kami juga menyarankan Anda mendesain semua aplikasi Elastic Beanstalk Anda tanpa kewarganegaraan mungkin, menggunakan komponen yang digabungkan secara longgar dan toleran terhadap kesalahan yang dapat ditingkatkan sesuai kebutuhan. Untuk informasi selengkapnya tentang merancang arsitektur aplikasi yang dapat diskalakan AWS, lihat AWS Well-Architected Framework.

Keamanan

Keamanan AWS adalah tanggung jawab bersama. Amazon Web Services melindungi sumber daya fisik di lingkungan Anda dan memastikan bahwa Cloud adalah tempat yang aman bagi Anda untuk menjalankan aplikasi. Anda bertanggung jawab atas keamanan data yang masuk dan keluar dari lingkungan Elastic Beanstalk Anda dan keamanan aplikasi Anda.

Konfigurasikan SSL untuk melindungi informasi yang mengalir antara aplikasi dan klien Anda. Untuk mengkonfigurasi SSL, Anda memerlukan sertifikat gratis dari AWS Certificate Manager (ACM). Jika Anda sudah memiliki sertifikat dari otoritas sertifikat eksternal (CA), Anda dapat menggunakan ACM untuk mengimpor sertifikat Anda. Jika tidak, Anda dapat mengimpornya menggunakan file AWS CLI.

Jika ACM tidak tersedia di Anda Wilayah AWS, Anda dapat membeli sertifikat dari CA eksternal, seperti VeriSign atau Entrust. Kemudian, gunakan AWS Command Line Interface (AWS CLI) untuk mengunggah sertifikat pihak ketiga atau yang ditandatangani sendiri dan kunci pribadi ke AWS Identity and Access Management (IAM). Kunci publik sertifikat mengotentikasi server Anda ke browser. Hal ini juga berfungsi sebagai dasar dalam membuat kunci sesi bersama yang membuat data terenkripsi di kedua arah. Untuk petunjuk tentang cara membuat, mengunggah, dan menetapkan sertifikat SSL ke lingkungan Anda, lihat. Mengonfigurasi HTTPS untuk lingkungan Elastic Beanstalk Anda

Saat Anda mengonfigurasi sertifikat SSL untuk lingkungan Anda, data dienkripsi antara klien dan penyeimbang beban Elastic Load Balancing untuk lingkungan Anda. Secara default, enkripsi dihentikan pada penyeimbang beban, dan lalu lintas antara penyeimbang beban dan EC2 instans Amazon tidak dienkripsi.

Penyimpanan tetap

Aplikasi Elastic Beanstalk berjalan EC2 di instans Amazon yang tidak memiliki penyimpanan lokal persisten. Ketika EC2 instans Amazon dihentikan, sistem file lokal tidak disimpan. EC2 Instans Amazon baru dimulai dengan sistem file default. Kami menyarankan Anda mengonfigurasi aplikasi Anda untuk menyimpan data dalam sumber data persisten. AWS menawarkan sejumlah layanan penyimpanan persisten yang dapat Anda gunakan untuk aplikasi Anda. Tabel berikut mencantumkan sejumlah layanannya.

Layanan penyimpanan	Dokumentasi layanan	Integrasi Elastic Beanstalk
Amazon S3	Dokumentasi Layanan Penyimpanan Sederhana Amazon	Menggunakan Elastic Beanstalk dengan Amazon S3
Sistem File Elastis Amazon	Dokumentasi Amazon Elastic File System	Menggunakan Elastic Beanstalk dengan Amazon Elastic File System
Amazon Elastic Block Store	Amazon Elastic Block Store Panduan Fitur: Toko Blok Elastis	
Amazon DynamoDB	Dokumentasi Amazon DynamoDB	Menggunakan Elastic Beanstalk dengan Amazon DynamoDB
Amazon Relational Database Service (RDS)	Dokumentasi Layanan Amazon Relational Database Service	Menggunakan Elastic Beanstalk dengan Amazon RDS

> **Note**
>
> Elastic Beanstalk menciptakan pengguna webapp untuk Anda atur sebagai pemilik direktori aplikasi pada instance. EC2 Untuk versi platform Amazon Linux 2 yang dirilis pada atau setelah 3 Februari 2022, Elastic Beanstalk memberi pengguna webapp nilai uid (id pengguna) dan gid (id grup) 900 untuk lingkungan baru. Itu melakukan hal yang sama untuk lingkungan yang ada setelah pembaruan versi platform. Pendekatan ini menjaga izin akses yang konsisten bagi pengguna webapp ke penyimpanan sistem file permanen.
> Dalam situasi yang tidak mungkin bahwa pengguna atau proses lain sudah menggunakan 900, sistem operasi default pengguna webapp uid dan gid ke nilai lain. Jalankan perintah Linux id webapp pada EC2 instance Anda untuk memverifikasi nilai uid dan gid yang ditetapkan ke pengguna webapp.

Toleransi kesalahan

Sebagai aturan praktis, Anda harus menjadi pesimis saat merancang arsitektur untuk cloud. Manfaatkan elastisitas yang ditawarkannya. Selalu rancang, terapkan, dan deploy untuk pemulihan otomatis dari kegagalan. Gunakan beberapa Availability Zone untuk EC2 instans Amazon Anda dan untuk Amazon RDS. Availability Zone secara konseptual seperti pusat data logis. Gunakan Amazon CloudWatch untuk mendapatkan lebih banyak visibilitas ke dalam kesehatan aplikasi Elastic Beanstalk Anda dan mengambil tindakan yang tepat jika terjadi kegagalan perangkat keras atau penurunan kinerja. Konfigurasikan pengaturan Auto Scaling Anda untuk mempertahankan armada EC2 instans Amazon Anda pada ukuran tetap sehingga EC2 instans Amazon yang tidak sehat diganti dengan yang baru. Jika Anda menggunakan Amazon RDS, atur periode retensi untuk pencadangan, sehingga Amazon RDS dapat melakukan pencadangan otomatis.

Pengiriman konten

Ketika pengguna terhubung ke situs web Anda, permintaan mereka mungkin diarahkan melalui sejumlah jaringan individu. Akibatnya, pengguna mungkin mengalami kinerja yang buruk karena latensi tinggi. Amazon CloudFront dapat membantu memperbaiki masalah latensi dengan mendistribusikan konten web Anda, seperti gambar dan video, di seluruh jaringan lokasi tepi di seluruh dunia. Permintaan pengguna dialihkan ke lokasi tepi terdekat, sehingga konten dikirimkan dengan kinerja terbaik. CloudFront bekerja mulus dengan Amazon S3, yang tahan lama menyimpan versi asli dan definitif file Anda. Untuk informasi selengkapnya tentang Amazon CloudFront, lihat Panduan CloudFront Pengembang Amazon.

Pembaruan perangkat lunak dan patch

AWS Elastic Beanstalk secara teratur merilis pembaruan platform untuk memberikan perbaikan, pembaruan perangkat lunak, dan fitur baru. Elastic Beanstalk menawarkan beberapa opsi untuk menangani pembaruan platform. Dengan pembaruan platform terkelola, lingkungan Anda secara otomatis meningkatkan ke versi terbaru platform selama jendela pemeliharaan terjadwal saat aplikasi Anda tetap dalam layanan. Untuk lingkungan yang dibuat pada 25 November 2019 atau lebih baru menggunakan konsol Elastic Beanstalk, pembaruan terkelola diaktifkan secara default bila memungkinkan. Anda juga dapat memulai pembaruan secara manual menggunakan konsol Elastic Beanstalk atau EB CLI.

Konektivitas

Elastic Beanstalk harus dapat terhubung ke instans di lingkungan Anda untuk menyelesaikan deployment. Ketika Anda men-deploy aplikasi Elastic Beanstalk di dalam VPC Amazon, konfigurasi yang diperlukan untuk mengaktifkan konektivitas tergantung pada jenis lingkungan Amazon VPC yang Anda buat:

- Untuk lingkungan instance tunggal, tidak diperlukan konfigurasi tambahan. Ini karena, dengan lingkungan ini, Elastic Beanstalk memberikan EC2 setiap instans Amazon alamat IP Elastis publik yang memungkinkan instance berkomunikasi langsung dengan internet.
- Untuk lingkungan yang dapat diskalakan dan seimbang beban di Amazon VPC dengan subnet publik dan privat, Anda harus melakukan hal berikut:
  - Buat penyeimbang beban di subnet publik untuk merutekan lalu lintas masuk dari internet ke instans Amazon. EC2
  - Buat perangkat terjemahan alamat jaringan (NAT) untuk merutekan lalu lintas keluar dari EC2 instans Amazon di subnet pribadi ke internet.
  - Buat aturan perutean masuk dan keluar untuk EC2 instans Amazon di dalam subnet pribadi.
  - Jika Anda menggunakan instans NAT, konfigurasikan grup keamanan untuk instans NAT dan EC2 instans Amazon untuk mengaktifkan komunikasi internet.
- Untuk lingkungan yang seimbang beban dan dapat diskalakan di VPC Amazon yang memiliki satu subnet publik, tidak diperlukan konfigurasi tambahan. Ini karena, dengan lingkungan ini, EC2 instans Amazon Anda dikonfigurasi dengan alamat IP publik yang memungkinkan instance berkomunikasi dengan internet.

Untuk informasi selengkapnya tentang menggunakan Elastic Beanstalk dengan Amazon VPC, lihat Menggunakan Elastic Beanstalk dengan Amazon VPC.

Mengelola aplikasi Elastic Beanstalk

Bab ini menjelaskan cara mengelola dan mengkonfigurasi aplikasi Elastic Beanstalk Anda. Langkah pertama dalam menggunakan AWS Elastic Beanstalk adalah membuat aplikasi, yang mewakili aplikasi web Anda di AWS. Dalam Elastic Beanstalk aplikasi berfungsi sebagai wadah untuk lingkungan yang menjalankan aplikasi web Anda dan untuk versi kode sumber aplikasi web Anda, konfigurasi yang disimpan, log, dan artefak lain yang Anda buat saat menggunakan Elastic Beanstalk.

Untuk membuat sebuah aplikasi

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Aplikasi, lalu pilih Buat aplikasi.
3. Gunakan formulir yang ada pada layar untuk memberikan nama aplikasi.
4. (Opsional) Berikan deskripsi, dan tambahkan kunci dan nilai tag.
5. Pilih Buat.

Setelah membuat aplikasi, konsol meminta Anda untuk membuat sebuah lingkungan untuk itu. Untuk informasi rinci tentang semua opsi yang tersedia, lihat Membuat lingkungan Elastic Beanstalk.

Jika Anda tidak lagi memerlukan aplikasi, Anda dapat menghapusnya.

> ⚠ Warning
>
> Menghapus aplikasi akan menghentikan semua lingkungan yang terkait dan menghapus semua versi aplikasi dan konfigurasi tersimpan yang dimilik aplikasi tersebut.

Untuk menghapus aplikasi

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Aplikasi, kemudian pilih aplikasi Anda pada daftar.
3. Pilih Tindakan, lalu pilih Hapus aplikasi.

Topik

- Konsol pengelola aplikasi Elastic Beanstalk
- Mengelola versi aplikasi
- Buat bundel sumber aplikasi Elastic Beanstalk
- Menggunakan EB CLI dengan AWS CodeBuild
- Aplikasi pemberian label
- Pelabelan sumber daya aplikasi Elastic Beanstalk

Konsol pengelola aplikasi Elastic Beanstalk

Topik ini menjelaskan bagaimana Anda dapat menggunakan AWS Elastic Beanstalk konsol untuk mengelola aplikasi, versi aplikasi, dan konfigurasi yang disimpan.

Untuk mengakses konsol manajemen aplikasi

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Pada panel navigasi, pilih Aplikasi, dan kemudian pilih nama aplikasi Anda dari daftar.

   Halaman ikhtisar aplikasi menampilkan daftar dengan ikhtisar semua lingkungan yang terkait dengan aplikasi.
3. Anda mempunyai beberapa cara untuk melanjutkan:

   a. Dari menu tarik-turun Tindakan, Anda dapat memilih salah satu tindakan manajemen aplikasi: Buat lingkungan, Hapus aplikasi, Lihat versi aplikasi, Lihat konfigurasi yang disimpan, Kembalikan lingkungan yang dihentikan.

      Untuk meluncurkan lingkungan di aplikasi ini, Anda dapat langsung memilih Buat lingkungan. Lihat perinciannya di the section called “Membuat lingkungan”.

   b. Halaman mencantumkan nama lingkungan di samping aplikasi yang digunakan ke lingkungan. Pilih nama lingkungan untuk pergi ke konsol pengaturan lingkungan untuk lingkungan tersebut, dimana Anda dapat mengkonfigurasi, memantau, atau mengelola lingkungan.

   c. Saat Anda memilih aplikasi dari daftar, panel navigasi kiri mencantumkan aplikasi.

      - Pilih Versi aplikasi mengikuti nama aplikasi di panel navigasi untuk melihat dan mengelola versi aplikasi untuk aplikasi Anda.

Versi aplikasi ini adalah versi yang terunggah dari kode aplikasi Anda. Anda dapat mengunggah versi yang baru, menyebarkan versi yang sudah ada ke salah satu lingkungan aplikasi, atau menghapus versi lama. Untuk informasi selengkapnya, lihat Mengelola versi aplikasi.

- PilihKonfigurasi tersimpanmengikuti nama aplikasi di panel navigasi untuk melihat dan mengelola konfigurasi yang disimpan dari lingkungan yang berjalan.

  Konfigurasi yang disimpan adalah kumpulan pengaturan yang dapat Anda gunakan untuk memulihkan pengaturan lingkungan ke pengaturan sebelumnya, atau untuk menciptakan lingkungan dengan pengaturan yang sama. Untuk informasi selengkapnya, lihat Menggunakan konfigurasi tersimpan Elastic Beanstalk.

Mengelola versi aplikasi

Topik ini menjelaskan versi aplikasi dan cara membuat dan mengelolanya.

Elastic Beanstalk membuat versi aplikasi setiap kali Anda mengunggah kode sumber. Hal ini biasanya terjadi ketika Anda membuat lingkungan atau mengunggah dan menyebarkan kode menggunakan konsol manajemen lingkungan atau EB CLI. Elastic Beanstalk menghapus versi aplikasi ini sesuai dengan kebijakan siklus hidup aplikasi dan ketika Anda menghapus aplikasi. Untuk detail tentang kebijakan siklus hidup aplikasi, lihat Mengkonfigurasi pengaturan siklus hidup versi aplikasi.

Anda juga dapat mengunggah bundel sumber tanpa menyebarkan dari konsol manajemen aplikasi atau dengan perintah EB CLI eb appversion. Elastic Beanstalk menyimpan bundel sumber di Amazon Simple Storage Service (Amazon S3) dan tidak secara otomatis menghapusnya.

Anda dapat memberikan label ke versi aplikasi saat Anda membuatnya, dan mengedit label dari versi aplikasi yang ada. Lihat perinciannya di Melabeli versi aplikasi.

Membuat versi aplikasi

Anda juga dapat membuat versi aplikasi baru menggunakan EB CLI. Untuk informasi selengkapnya, lihat eb appversion di perintah EB CLI bab.

> **Note**
>
> Seiring waktu, aplikasi Anda dapat mengumpulkan banyak versi aplikasi. Untuk menghemat ruang penyimpanan dan menghindari memukul Kuota versi aplikasi, sebaiknya hapus versi aplikasi yang tidak lagi Anda butuhkan.

File yang Anda tentukan dalam prosedur berikut dikaitkan dengan aplikasi Anda. Anda dapat menyebarkan versi aplikasi ke lingkungan baru atau lingkungan yang sudah ada.

Untuk membuat versi aplikasi yang baru

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Pada panel navigasi, pilih Aplikasi, dan kemudian pilih nama aplikasi Anda dari daftar.
3. Di panel navigasi, cari nama aplikasi Anda dan pilih Versi aplikasi.
4. Pilih Unggah. Gunakan formulir di layar untuk mengunggah aplikasi Anda paket sumber.

   > **Note**
   >
   > Batas ukuran file bundel sumber adalah 500 MB.

5. Opsional, memberikan deskripsi singkat, dan menambahkan kunci label dan nilai-nilai.
6. Pilih Unggah.

Menghapus versi aplikasi

Anda juga dapat menghapus versi aplikasi menggunakan EB CLI. Untuk informasi selengkapnya, lihat eb appversion di perintah EB CLI bab.

> **Note**
>
> Menghapus versi aplikasi tidak mempengaruhi lingkungan yang sedang menggunakan versi tersebut.

Anda juga dapat mengkonfigurasi Elastic Beanstalk untuk menghapus versi lama secara otomatis dengan mengkonfigurasi pengaturan siklus hidup versi aplikasi. Jika Anda mengonfigurasi setelan siklus hidup ini, setelan tersebut akan diterapkan saat Anda membuat versi aplikasi baru. Misalnya, jika Anda mengkonfigurasi maksimum 25 versi aplikasi, Elastic Beanstalk menghapus versi tertua ketika Anda mengunggah versi 26. Jika Anda menetapkan usia maksimum 90 hari, versi apa pun yang lebih tua dari 90 hari akan dihapus saat Anda mengunggah versi baru. Lihat perinciannya di the section called “Siklus hidup versi”.

Menghapus versi aplikasi

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Pada panel navigasi, pilih Aplikasi, dan kemudian pilih nama aplikasi Anda dari daftar.
3. Di panel navigasi, cari nama aplikasi Anda dan pilih versi Aplikasi.
4. Pilih satu atau lebih versi aplikasi yang ingin Anda hapus.
5. Pilih Tindakan, lalu pilih Hapus.
6. (Opsional) Untuk meninggalkan bundel sumber aplikasi untuk versi aplikasi ini di Amazon Simple Storage Service (Amazon S3) bucket Anda, hilangkan kotak untuk Menghapus versi dari Amazon S3.
7. Pilih Hapus.

Jika Anda tidak memilih untuk menghapus bundel sumber dari Amazon S3, Elastic Beanstalk masih menghapus versi dari catatannya. Namun, bundel sumber yang tersisa di penyimpanan Elastic Beanstalk bucket Anda. Kuota versi aplikasi hanya berlaku untuk versi Elastic Beanstalk trek. Oleh karena itu, Anda dapat menghapus versi untuk tetap berada dalam kuota, tetapi mempertahankan semua bundel sumber di Amazon S3.

> **Note**
>
> Kuota versi aplikasi tidak berlaku untuk bundel sumber, tetapi Anda mungkin masih dikenakan biaya Amazon S3, dan menyimpan informasi pribadi di luar waktu yang Anda butuhkan. Elastic Beanstalk tidak pernah menghapus bundel sumber secara otomatis. Anda harus menghapus bundel sumber saat tidak lagi membutuhkannya.

Mengkonfigurasi pengaturan siklus hidup versi aplikasi

Topik ini menjelaskan kebijakan dan kuota yang berlaku Elastic Beanstalk untuk versi aplikasi Anda di lingkungan tertentu, termasuk berapa lama versi aplikasi tetap berada di lingkungan.

Setiap kali Anda mengunggah versi baru dari aplikasi Anda dengan konsol Elastic Beanstalk atau EB CLI, Elastic Beanstalk menciptakan versi aplikasi. Jika Anda tidak menghapus versi yang tidak lagi Anda gunakan, Anda akhirnya akan mencapai kuota versi aplikasi dan tidak dapat membuat versi baru dari aplikasi tersebut.

Anda dapat menghindari batas jumlah kuota dengan menerapkan Kebijakan siklus hidup versi aplikasi ke aplikasi Anda. Kebijakan siklus hidup memberitahu Elastic Beanstalk untuk menghapus versi aplikasi yang lama, atau untuk menghapus versi aplikasi ketika jumlah total versi aplikasi melebihi jumlah tertentu.

Elastic Beanstalk menerapkan kebijakan siklus hidup aplikasi setiap kali Anda membuat versi aplikasi baru, dan menghapus hingga 100 versi setiap kali kebijakan siklus hidup diterapkan. Elastic Beanstalk menghapus versi lama setelah membuat versi baru, dan tidak menghitung versi baru menuju jumlah maksimum versi yang didefinisikan dalam kebijakan.

Elastic Beanstalk tidak menghapus versi aplikasi yang saat ini sedang digunakan oleh lingkungan, atau versi aplikasi yang digunakan untuk lingkungan yang dihentikan kurang dari sepuluh minggu sebelum kebijakan dicetuskan.

Kuota versi aplikasi berlaku di semua aplikasi di suatu wilayah. Jika Anda memiliki beberapa aplikasi, konfigurasikan setiap aplikasi dengan kebijakan siklus hidup yang sesuai agar tidak mencapai kuota. Misalnya, jika Anda memiliki 10 aplikasi di suatu wilayah dan kuota adalah 1.000 versi aplikasi, pertimbangkan untuk menetapkan kebijakan siklus hidup dengan kuota 99 versi aplikasi untuk semua aplikasi, atau tetapkan nilai lain di setiap aplikasi selama totalnya kurang dari 1.000 versi aplikasi. Elastic Beanstalk hanya menerapkan kebijakan jika pembuatan versi aplikasi berhasil, jadi jika Anda telah mencapai kuota, Anda harus menghapus beberapa versi secara manual sebelum membuat versi baru.

Secara default, Elastic Beanstalk meninggalkan versi aplikasi paket sumber di Amazon S3 untuk mencegah hilangnya data. Anda dapat menghapus bundel sumber untuk menghemat ruang.

Anda dapat mengatur pengaturan siklus hidup melalui Elastic Beanstalk CLI dan. APIs Lihateb appversion, CreateApplication(menggunakan `ResourceLifecycleConfig` parameter), dan UpdateApplicationResourceLifecycleuntuk detailnya.

Mengatur pengaturan siklus hidup aplikasi di konsol

Anda dapat menentukan pengaturan siklus hidup di konsol Elastic Beanstalk.

untuk menentukan setelan siklus hidup aplikasi

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Pada panel navigasi, pilih Aplikasi, dan kemudian pilih nama aplikasi Anda dari daftar.
3. Di panel navigasi, cari nama aplikasi Anda dan pilih versi aplikasi.
4. Pilih Pengaturan.
5. Gunakan formulir di layar untuk mengonfigurasi pengaturan siklus hidup aplikasi.
6. Pilih Simpan.

Di halaman pengaturan, Anda dapat melakukan hal berikut.

- Konfigurasikan pengaturan siklus hidup berdasarkan total jumlah versi aplikasi atau usia versi aplikasi.
- Tentukan apakah menghapus bundel sumber dari S3 saat versi aplikasi dihapus.
- Tentukan peran layanan di mana versi aplikasi dihapus. Untuk menyertakan semua izin yang diperlukan untuk penghapusan versi, pilih peran layanan default Elastic Beanstalk, bernama `aws-elasticbeanstalk-service-role`, atau peran layanan lain menggunakan kebijakan layanan terkelola Elastic Beanstalk. Lihat informasi yang lebih lengkap di Mengelola peran layanan Elastic Beanstalk.

Melabeli versi aplikasi

Topik ini menjelaskan manfaat menandai versi aplikasi Elastic Beanstalk Anda dan cara mengelola tag.

Anda dapat menerapkan tag ke versi AWS Elastic Beanstalk aplikasi Anda. Tag adalah pasangan nilai kunci yang terkait dengan AWS sumber daya. Untuk informasi tentang penandaan sumber daya Elastic Beanstalk, penggunaan kasus, kunci label dan batasan nilai, dan jenis sumber daya yang didukung, lihat Pelabelan sumber daya aplikasi Elastic Beanstalk.

Anda dapat menentukan tlabel ag saat Anda membuat versi aplikasi. Dalam versi aplikasi yang ada, Anda dapat menambahkan atau menghapus label, dan memperbarui nilai label yang ada. Anda dapat menambahkan hingga 50 label ke setiap versi aplikasi.

Menambahkan label selama pembuatan versi aplikasi

Bila Anda menggunakan konsol Elastic Beanstalk untuk menciptakan lingkungan, dan Anda memilih untuk mengunggah versi kode aplikasi, Anda dapat menentukan kunci dan nilai label untuk dikaitkan dengan versi aplikasi baru.

Anda juga dapat menggunakan konsol Elastic Beanstalk untukmengunggah versi aplikasitanpa segera menggunakannya dalam lingkungan. Anda dapat menentukan kunci label dan nilai ketika Anda meng-upload versi aplikasi.

Dengan AWS CLI atau klien berbasis API lainnya, tambahkan tag dengan menggunakan `--tags` parameter pada perintah. create-application-version

```
$ aws elasticbeanstalk create-application-version \
      --tags Key=mytag1,Value=value1 Key=mytag2,Value=value2 \
      --application-name my-app --version-label v1
```

Ketika Anda menggunakan EB CLI untuk membuat atau memperbarui lingkungan, versi aplikasi dibuat dari kode yang Anda menyebarkan. Tidak ada cara langsung untuk menandai versi aplikasi selama pembuatannya melalui EB CLI. Lihat bagian berikut untuk mempelajari tentang menambahkan tag ke versi aplikasi yang ada.

Mengelola label dari versi aplikasi yang ada

Anda dapat menambahkan, memperbarui, dan menghapus label dalam versi aplikasi Elastic Beanstalk yang ada.

Untuk mengelola label versi aplikasi menggunakan konsol Elastic Beanstalk

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Pada panel navigasi, pilih Aplikasi, dan kemudian pilih nama aplikasi Anda dari daftar.
3. Di panel navigasi, cari nama aplikasi Anda dan pilihVersi aplikasi.
4. Pilih versi aplikasi yang Anda ingin kelola.

5. PilihTindakan, lalu pilihKelola label.
6. Gunakan formulir di layar untuk menambahkan, memperbarui, atau menghapus tag.
7. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Jika Anda menggunakan EB CLI untuk memperbarui versi aplikasi Anda, gunakaneb tagsuntuk menambahkan, memperbarui, menghapus, atau daftar label.

Misalnya, perintah berikut mencantumkan label di versi aplikasi.

```
~/workspace/my-app$ eb tags --list --resource "arn:aws:elasticbeanstalk:us-east-2:my-account-id:applicationversion/my-app/my-version"
```

Perintah berikut memperbarui label `mytag1` dan menghapus label `mytag2`.

```
~/workspace/my-app$ eb tags --update mytag1=newvalue --delete mytag2 \
      --resource "arn:aws:elasticbeanstalk:us-east-2:my-account-id:applicationversion/my-app/my-version"
```

Untuk daftar lengkap opsi dan contoh lainnya, lihat `eb tags`.

Dengan AWS CLI atau klien berbasis API lainnya, gunakan list-tags-for-resource perintah untuk membuat daftar tag versi aplikasi.

```
$ aws elasticbeanstalk list-tags-for-resource --resource-arn
 "arn:aws:elasticbeanstalk:us-east-2:my-account-id:applicationversion/my-app/my-version"
```

Gunakan update-tags-for-resource Untuk menambahkan, memperbarui, atau menghapus label di versi aplikasi.

```
$ aws elasticbeanstalk update-tags-for-resource \
      --tags-to-add Key=mytag1,Value=newvalue --tags-to-remove mytag2 \
      --resource-arn "arn:aws:elasticbeanstalk:us-east-2:my-account-id:applicationversion/my-app/my-version"
```

Tentukan kedua label untuk menambahkan dan label untuk memperbarui dalam `--tags-to-add` parameter dari update-tags-for-resource. Tag yang tidak ada ditambahkan, dan nilai tag yang ada diperbarui.

> **Note**
>
> Untuk menggunakan beberapa CLI EB AWS CLI dan perintah dengan versi aplikasi Elastic Beanstalk, Anda memerlukan ARN versi aplikasi. Anda dapat mengambil ARN dengan menggunakan perintah berikut.
>
> ```
> $ aws elasticbeanstalk describe-application-versions --application-name my-app
>  --version-label my-version
> ```

Buat bundel sumber aplikasi Elastic Beanstalk

Topik ini menjelaskan cara mengunggah file sumber aplikasi Anda ke Elastic Beanstalk dalam bundel sumber. Ini menjelaskan persyaratan bundel sumber, struktur, dan pendekatan untuk membuatnya.

Saat Anda menggunakan AWS Elastic Beanstalk konsol untuk menyebarkan aplikasi baru atau versi aplikasi, Anda harus mengunggah file untuk aplikasi dalam bundel sumber. Sumber paket Anda harus memenuhi persyaratan berikut:

- Terdiri dari satu file `ZIP` atau file `WAR` (Anda dapat menyertakan beberapa file `WAR` dalam file `ZIP` Anda)
- Tidak melebihi 500 MB
- Tidak termasuk folder induk atau direktori tingkat atas (subdirektori baik-baik saja)

Jika Anda ingin menyebarkan aplikasi pekerja yang memproses tugas-tugas latar belakang periodik, paket sumber aplikasi Anda juga harus menyertakan file `cron.yaml`. Untuk informasi selengkapnya, lihat Tugas periodik.

Jika Anda menerapkan aplikasi Anda dengan Elastic Beanstalk Command Line Interface (EB CLI), Toolkit for Eclipse, AWS atau Toolkit for Visual Studio AWS , file ZIP atau WAR akan secara otomatis terstruktur dengan benar. Untuk informasi selengkapnya, lihat Menyiapkan antarmuka baris perintah EB (EB CLI) untuk mengelola Elastic Beanstalk, Menyebarkan aplikasi Java dengan Elastic Beanstalk, dan The AWS Toolkit for Visual Studio.

Bagian

- Membuat sebuah paket sumber dari baris perintah
- Membuat paket sumber dengan Git

- zip file di Mac OS X Finder atau Windows explorer
- Membuat bundel sumber untuk aplikasi NET
- Menguji bundel sumber Anda

Membuat sebuah paket sumber dari baris perintah

Buat sebuah paket sumber menggunakan perintah `zip`. Untuk menyertakan file dan folder tersembunyi, gunakan pola seperti berikut ini.

```
~/myapp$ zip ../myapp.zip -r * .[^.]*
  adding: app.js (deflated 63%)
  adding: index.js (deflated 44%)
  adding: manual.js (deflated 64%)
  adding: package.json (deflated 40%)
  adding: restify.js (deflated 85%)
  adding: .ebextensions/ (stored 0%)
  adding: .ebextensions/xray.config (stored 0%)
```

Hal ini memastikan bahwa file konfigurasi Elastic Beanstalk dan file dan folder lain yang dimulai dengan sebuah titik (.) disertakan dalam arsip.

Untuk aplikasi web Tomcat, gunakan `jar` untuk membuat arsip web.

```
~/myapp$ jar -cvf myapp.war .
```

Perintah di atas termasuk file tersembunyi yang dapat meningkatkan ukuran paket sumber Anda yang tidak perlu. Untuk pengaturan lebih lanjut, gunakan pola file yang lebih rinci, atau membuat paket sumber Anda dengan Git.

Membuat paket sumber dengan Git

Jika Anda menggunakan Git untuk mengelola kode sumber aplikasi Anda, gunakan perintah `git archive` untuk membuat paket sumber Anda.

```
$ git archive -v -o myapp.zip --format=zip HEAD
```

`git archive` hanya menyertakan file yang disimpan dalam git, dan tidak termasuk berkas yang diabaikan dan berkas git. Ini membantu menjaga paket sumber Anda sekecil mungkin. Untuk informasi lebih lanjut, buka situs web halaman manual git-archive.

zip file di Mac OS X Finder atau Windows explorer

Saat Anda membuat file `ZIP` di Mac OS X Finder atau Windows Explorer, pastikan Anda zip file dan subfolder itu sendiri, bukan hanya zip folder induk.

> **Note**
>
> Antarmuka pengguna grafis (GUI) pada Mac OS X dan sistem operasi berbasis Linux tidak menampilkan file dan folder dengan nama yang dimulai dengan sebuah titik (.). Gunakan baris perintah bukan GUI untuk mengkompres aplikasi Anda jika file `ZIP` harus menyertakan file dalam folder tersembunyi, seperti `.ebextensions`. Untuk prosedur baris perintah untuk membuat file `ZIP` pada Mac OS X atau sistem operasi berbasis Linux, lihat Membuat sebuah paket sumber dari baris perintah.

Example

Misalkan Anda memiliki folder berlabel proyek Python `myapp`, yang mencakup berkas dan subfolder berikut ini:

```
myapplication.py
README.md
static/
static/css
static/css/styles.css
static/img
static/img/favicon.ico
static/img/logo.png
templates/
templates/base.html
templates/index.html
```

Seperti yang tercantum dalam daftar persyaratan di atas, paket sumber Anda harus dikompresi tanpa folder induk, sehingga struktur yang dekompresi tidak memuat direktori top-level tambahan. Dalam contoh ini, tidak ada folder `myapp` yang harus dibuat ketika file didekompresi (atau, pada baris perintah, tidak ada segmen `myapp` yang harus ditambahkan ke lokasi file).

Struktur file contoh ini digunakan di seluruh topik ini untuk menggambarkan bagaimana cara zip file.

Membuat bundel sumber untuk aplikasi NET

Jika Anda menggunakan Visual Studio, Anda dapat menggunakan alat penyebaran yang disertakan dalam untuk menyebarkan aplikasi.NET Anda AWS Toolkit for Visual Studio ke Elastic Beanstalk. Untuk informasi selengkapnya, lihat Men-deploy aplikasi Elastic Beanstalk di .NET menggunakan alat deployment.

Jika Anda perlu secara manual membuat bundel sumber untuk aplikasi .NET Anda, Anda tidak dapat hanya membuat file `ZIP` yang berisi direktori proyek. Anda harus membuat paket deployment web untuk proyek Anda yang cocok untuk deployment ke Elastic Beanstalk. Ada beberapa metode yang dapat Anda gunakan untuk membuat paket deployment:

- Buat paket deployment menggunakan wizard Publikasikan Web dalam Visual Studio. Untuk informasi lebih lanjut, kunjungi Cara: Membuat Paket Deployment Web dalam Visual Studio.

  > **Important**
  >
  > Ketika membuat paket deployment web, Anda harus memulai Nama situs dengan `Default Web Site`.

- Jika Anda memiliki proyek.NET, Anda dapat membuat paket penyebaran menggunakan msbuild perintah seperti yang ditunjukkan pada contoh berikut.

  > **Important**
  >
  > Parameter `DeployIisAppPath` harus dimulai dengan `Default Web Site`.

  ```
  C:/> msbuild <web_app>.csproj /t:Package /p:DeployIisAppPath="Default Web Site"
  ```

- Jika Anda memiliki proyek situs web, Anda dapat menggunakan alat IIS Web Deploy untuk membuat paket deployment. Untuk informasi lebih lanjut, kunjungi Pengemasan dan Pemulihan situs Web.

  > **Important**
  >
  > Parameter `apphostconfig` harus dimulai dengan `Default Web Site`.

Jika Anda menyebarkan beberapa aplikasi atau ASP.NET Core aplikasi, tempatkan folder `.ebextensions` Anda di akar bundel sumber, berdampingan dengan bundel aplikasi dan bentuk file:

```
~/workspace/source-bundle/
|-- .ebextensions
|   |-- environmentvariables.config
|   `-- healthcheckurl.config
|-- AspNetCore101HelloWorld.zip
|-- AspNetCoreHelloWorld.zip
|-- aws-windows-deployment-manifest.json
`-- VS2015AspNetWebApiApp.zip
```

Menguji bundel sumber Anda

Anda mungkin ingin menguji bundel sumber Anda secara lokal sebelum Anda mengunggah ke Elastic Beanstalk. Karena Elastic Beanstalk pada dasarnya menggunakan baris perintah untuk mengekstrak file, yang terbaik untuk melakukan tes Anda dari baris perintah daripada dengan alat GUI.

Pastikan bahwa file dekompresi muncul di folder yang sama dengan arsip itu sendiri, bukan di folder tingkat atas baru atau direktori.

Menggunakan EB CLI dengan AWS CodeBuild

AWS CodeBuild mengumpulkan kode sumber Anda, menjalankan pengujian unit, dan menghasilkan Artifact yang siap di-deploy. Anda dapat menggunakan CodeBuild bersama dengan EB CLI untuk mengotomatiskan pembuatan aplikasi Anda dari kode sumbernya. Pembuatan lingkungan dan setiap deployment sesudahnya dimulai dengan langkah pembangunan, dan kemudian men-deploy aplikasi yang dihasilkan.

> **Note**
>
> Beberapa daerah tidak menawarkan CodeBuild. Integrasi antara Elastic CodeBuild Beanstalk dan tidak berfungsi di wilayah ini.
> Untuk informasi tentang AWS layanan yang ditawarkan di setiap wilayah, lihat Tabel Wilayah.

Membuat aplikasi

Untuk membuat aplikasi Elastic Beanstalk yang menggunakan CodeBuild

1. Sertakan file spesifikasi CodeBuild build `buildspec.yml`, di folder aplikasi Anda.
2. Tambahkan entri `eb_codebuild_settings` dengan opsi yang spesifik dengan Elastic Beanstalk ke file.
3. Jalankan eb init di folder.

> **Note**
>
> Jangan gunakan karakter period (`.`) atau spasi ( ) dalam nama Aplikasi saat Anda menggunakan EB CodeBuild CLI dengan.

Elastic Beanstalk CodeBuild memperluas format file spesifikasi build untuk menyertakan pengaturan tambahan berikut:

```
eb_codebuild_settings:
  CodeBuildServiceRole: role-name
  ComputeType: size
  Image: image
  Timeout: minutes
```

`CodeBuildServiceRole`

ARN atau nama peran layanan AWS Identity and Access Management (IAM) yang CodeBuild dapat digunakan untuk berinteraksi dengan AWS layanan dependen atas nama Anda. Nilai ini diperlukan. Jika Anda menghilangkan itu, setiap perintah eb create atau eb deploy berikutnya gagal.

Untuk mempelajari selengkapnya tentang membuat peran layanan CodeBuild, lihat Membuat Peran CodeBuild Layanan di Panduan AWS CodeBuild Pengguna.

> **Note**
>
> Anda juga memerlukan izin untuk melakukan tindakan CodeBuild itu sendiri. Kebijakan pengguna yang dikelola AdministratorAccessElastic AWSElastic Beanstalk - Beanstalk

mencakup semua izin tindakan yang diperlukan. CodeBuild Jika Anda tidak menggunakan kebijakan terkelola, pastikan untuk mengizinkan izin berikut di kebijakan pengguna Anda.

```
"codebuild:CreateProject",
"codebuild:DeleteProject",
"codebuild:BatchGetBuilds",
"codebuild:StartBuild"
```

Lihat perinciannya di Mengelola kebijakan pengguna Elastic Beanstalk.

`ComputeType`

Jumlah sumber daya yang digunakan oleh kontainer Docker di lingkungan CodeBuild build. Nilai yang valid adalah BUILD_ GENERAL1 _SMALL, BUILD_ _MEDIUM, dan BUILD_ GENERAL1 _LARGE. GENERAL1

`Image`

Nama image Docker Hub atau Amazon ECR yang CodeBuild digunakan untuk lingkungan build. Gambar Docker ini harus berisi semua alat dan pustaka runtime yang diperlukan untuk membangun kode Anda, dan harus sesuai dengan platform target aplikasi Anda. CodeBuild mengelola dan memelihara satu set gambar yang secara khusus dimaksudkan untuk digunakan dengan Elastic Beanstalk. Direkomendasikan agar Anda menggunakan salah satunya. Untuk detailnya, lihat Gambar Docker Disediakan oleh CodeBuild di Panduan AWS CodeBuild Pengguna.

Nilai `Image` bersifat opsional. Jika Anda menghilangkan itu, perintah eb init mencoba untuk memilih citra yang paling sesuai dengan platform target Anda. Selain itu, jika Anda menjalankan eb init dengan mode interaktif dan gagal untuk memilih citra untuk Anda, maka akan meminta Anda untuk memilih salah satu. Di akhir inisialisasi yang sukses, eb init menulis citra yang dipilih ke dalam file `buildspec.yml`.

`Timeout`

Durasi, dalam hitungan menit, CodeBuild build berjalan sebelum waktu habis. Nilai ini bersifat opsional. Untuk detail tentang nilai valid dan default, lihat Membuat Proyek Build di CodeBuild.

> **Note**
>
> Batas waktu ini mengontrol durasi maksimum untuk CodeBuild menjalankan, dan CLI EB juga menghormatinya sebagai bagian dari langkah pertama untuk membuat versi aplikasi. Ini berbeda dari nilai yang dapat Anda tentukan dengan opsi `--timeout` dari perintah eb create atau eb deploy. Nilai terakhir mengontrol durasi maksimum untuk EB CLI menunggu pembuatan lingkungan atau pembaruan.

Membangun dan men-deploy kode aplikasi Anda

Kapan pun kode aplikasi Anda perlu diterapkan, CLI EB menggunakan CodeBuild untuk menjalankan build, lalu menyebarkan artefak build yang dihasilkan ke lingkungan Anda. Hal ini terjadi ketika Anda membuat lingkungan Elastic Beanstalk untuk aplikasi Anda menggunakan perintah eb create, dan setiap kali Anda kemudian men-deploy perubahan kode ke lingkungan menggunakan perintah eb deploy.

Jika CodeBuild langkah gagal, pembuatan atau penerapan lingkungan tidak dimulai.

Aplikasi pemberian label

Topik ini menjelaskan manfaat menandai aplikasi Elastic Beanstalk Anda. Ini juga memberikan instruksi untuk membuat dan mengelola tag aplikasi. Tag adalah pasangan nilai kunci yang terkait dengan AWS sumber daya. Untuk informasi tentang pelabelan sumber daya Elastic Beanstalk, menggunakan kasus, kunci label dan batasan nilai, dan jenis sumber daya yang didukung, lihat Pelabelan sumber daya aplikasi Elastic Beanstalk.

Anda dapat menentukan label saat Anda membuat aplikasi. Dalam aplikasi yang ada, Anda dapat menambah atau menghapus label, dan memperbarui nilai label yang ada. Anda dapat menambahkan hingga 50 label ke setiap aplikasi.

Menambahkan label selama pembuatan aplikasi

Bila Anda menggunakan konsol Elastic Beanstalk untuk membuat aplikasi, Anda dapat menentukan kunci dan nilai label dalam Kotak dialog Buat aplikasi Baru.

Jika Anda menggunakan EB CLI untuk membuat aplikasi, gunakan opsi `--tags` dengan eb init untuk menambahkan label.

```
~/workspace/my-app$ eb init --tags mytag1=value1,mytag2=value2
```

Dengan AWS CLI atau klien berbasis API lainnya, tambahkan tag dengan menggunakan `--tags` parameter pada perintah. create-application

```
$ aws elasticbeanstalk create-application \
      --tags Key=mytag1,Value=value1 Key=mytag2,Value=value2 \
      --application-name my-app --version-label v1
```

Mengelola label dari aplikasi yang ada

Anda dapat menambahkan, memperbarui, dan menghapus label di aplikasi Elastic Beanstalk yang sudah ada.

Untuk mengelola label aplikasi di konsol Elastic Beanstalk

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Pada panel navigasi, pilih Aplikasi, dan kemudian pilih nama aplikasi Anda dari daftar.
3. Pilih Tindakan, lalu pilih Kelola tag.
4. Gunakan formulir di layar untuk menambahkan, memperbarui, atau menghapus tag.
5. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Jika Anda menggunakan EB CLI untuk memperbarui aplikasi Anda, gunakan eb tags untuk menambah, memperbarui, menghapus, atau mencantumkan label.

Misalnya, perintah berikut cantumkan label dalam aplikasi.

```
~/workspace/my-app$ eb tags --list --resource "arn:aws:elasticbeanstalk:us-east-2:my-account-id:application/my-app"
```

Perintah berikut perbarui label `mytag1` dan hapus label `mytag2`.

```
~/workspace/my-app$ eb tags --update mytag1=newvalue --delete mytag2 \
      --resource "arn:aws:elasticbeanstalk:us-east-2:my-account-id:application/my-app"
```

Untuk daftar lengkap opsi dan contoh lainnya, lihat `eb tags`.

Dengan AWS CLI atau klien berbasis API lainnya, gunakan list-tags-for-resource perintah untuk membuat daftar tag aplikasi.

```
$ aws elasticbeanstalk list-tags-for-resource --resource-arn
 "arn:aws:elasticbeanstalk:us-east-2:my-account-id:application/my-app"
```

Gunakan perintah update-tags-for-resource untuk menambah, memperbarui, atau menghapus label di aplikasi.

```
$ aws elasticbeanstalk update-tags-for-resource \
      --tags-to-add Key=mytag1,Value=newvalue --tags-to-remove mytag2 \
      --resource-arn "arn:aws:elasticbeanstalk:us-east-2:my-account-id:application/my-
app"
```

Tentukan kedua label untuk menambah dan label untuk memperbarui parameter `--tags-to-add` dari update-tags-for-resource. Tag yang tidak ada ditambahkan, dan nilai tag yang ada diperbarui.

> **Note**
>
> Untuk menggunakan beberapa CLI EB AWS CLI dan perintah dengan aplikasi Elastic Beanstalk, Anda memerlukan ARN aplikasi. Anda dapat mengambil ARN dengan menggunakan perintah berikut.
>
> ```
> $ aws elasticbeanstalk describe-applications --application-names my-app
> ```

Pelabelan sumber daya aplikasi Elastic Beanstalk

Topik ini menjelaskan manfaat menggunakan tag dengan sumber daya aplikasi Elastic Beanstalk Anda bersama dengan kendala untuk melakukannya. Ini juga menjelaskan cara membuat dan mengelola tag untuk sumber daya aplikasi.

Anda dapat menerapkan tag ke sumber daya AWS Elastic Beanstalk aplikasi Anda. Tag adalah pasangan nilai kunci yang terkait dengan AWS sumber daya. Label dapat membantu Anda mengkategorikan sumber daya. Mereka sangat berguna jika Anda mengelola banyak sumber daya sebagai bagian dari beberapa AWS aplikasi.

Berikut beberapa cara menggunakan label dengan sumber daya Elastic Beanstalk:

- Tahap Deployment— Identifikasi sumber daya yang terkait dengan berbagai tahapan aplikasi Anda, seperti pengembangan, beta, dan produksi.
- Alokasi biaya — Gunakan laporan alokasi biaya untuk melacak penggunaan AWS sumber daya yang terkait dengan berbagai akun pengeluaran. Laporan mencakup kedua sumber daya baik yang berlabel maupun tidak, dan mereka mengumpulkan biaya menurut label. Untuk informasi tentang bagaimana laporan alokasi biaya yang menggunakan label, lihat Gunakan Label Alokasi Biaya untuk Laporan Penagihan Khusus dalam AWS Panduan Pengguna Penagihan dan Manajemen Biaya.
- Kontrol akses— Gunakan label untuk mengelola izin permintaan dan sumber daya. Misalnya, pengguna yang hanya dapat membuat dan mengelola lingkungan beta seharusnya hanya dapat mengakses sumber daya tahap beta. Untuk rincian selengkapnya, lihat Menggunakan tag untuk mengontrol akses ke sumber Elastic Beanstalk.

Anda dapat menambahkan hingga 50 label per sumber daya. Lingkungan sedikit berbeda: Elastic Beanstalk menambahkan tiga label sistem default ke lingkungan, dan Anda tidak dapat mengedit atau menghapus label ini. Selain label default, Anda dapat menambahkan hingga 47 label tambahan ke setiap lingkungan.

Batasan berikut berlaku untuk label kunci dan nilai:

- Kunci dan nilai dapat berisi huruf, angka, spasi, dan simbol-simbol berikut: `_ . : / = + - @`
- Kunci dapat berisi hingga 127 karakter. Nilai dapat berisi hingga 255 karakter.

> **Note**
>
> Batas panjang ini ditujukan untuk karakter Unicode dalam UTF-8. Untuk pengkodean multibyte lainnya, batasnya mungkin lebih rendah.

- Kunci peka terhadap huruf besar dan kecil.
- Kunci tidak bisa dimulai dengan `aws:` atau `elasticbeanstalk:`

Sumber daya yang dapat Anda beri label

Berikut ini adalah jenis sumber daya Elastic Beanstalk yang dapat Anda beri label, dan tautan ke topik tertentu untuk mengelola masing-masing label:

- Aplikasi

- Lingkungan
- Versi aplikasi
- Konfigurasi tersimpan

Tag propagasi untuk meluncurkan template

Elastic Beanstalk menyediakan opsi untuk mengaktifkan propagasi tag lingkungan untuk meluncurkan templat. Opsi ini memberikan dukungan berkelanjutan untuk kontrol akses berbasis tag (TBAC) dengan templat peluncuran.

> **Note**
>
> Konfigurasi peluncuran sedang dihapus dan diganti dengan template peluncuran. Untuk informasi selengkapnya, lihat Meluncurkan konfigurasi di Panduan Pengguna EC2 Auto Scaling Amazon.

Untuk mencegah downtime menjalankan EC2 instance AWS CloudFormation tidak menyebarkan tag ke template peluncuran yang ada. Jika ada kasus penggunaan yang memerlukan tag untuk sumber daya lingkungan Anda, Anda dapat mengaktifkan Elastic Beanstalk untuk membuat template peluncuran dengan tag untuk sumber daya ini. Untuk melakukannya, atur `LaunchTemplateTagPropagationEnabled` opsi di aws:autoscaling:launchconfiguration namespace ke. `true` Nilai default-nya adalah `false`.

Contoh file konfigurasi berikut memungkinkan propagasi tag untuk meluncurkan template.

```
option_settings:
  aws:autoscaling:launchconfiguration:
    LaunchTemplateTagPropagationEnabled: true
```

Elastic Beanstalk hanya dapat menyebarkan tag untuk meluncurkan template untuk sumber daya berikut:

- Volume EBS
- EC2 contoh
- EC2 antarmuka jaringan
- AWS CloudFormation luncurkan template yang menentukan sumber daya

Kendala ini ada karena CloudFormation hanya mengizinkan tag pada pembuatan template untuk sumber daya tertentu. Untuk informasi selengkapnya lihat TagSpecificationdi Panduan AWS CloudFormation Pengguna.

> **Important**
>
> - Mengubah nilai opsi ini dari `false` ke `true` untuk lingkungan yang ada mungkin merupakan perubahan besar untuk tag yang sudah ada sebelumnya.
> - Ketika fitur ini diaktifkan, propagasi tag akan membutuhkan EC2 penggantian, yang dapat mengakibatkan downtime. Anda dapat mengaktifkan pembaruan bergulir untuk menerapkan perubahan konfigurasi dalam batch dan mencegah waktu henti selama proses pembaruan. Untuk informasi selengkapnya, lihat Perubahan konfigurasi.

Untuk informasi selengkapnya tentang template peluncuran, lihat berikut ini:

- Luncurkan template di Panduan Pengguna EC2 Auto Scaling Amazon
- Bekerja dengan template di Panduan AWS CloudFormation Pengguna
- Cuplikan template Elastic Beanstalk di Panduan Pengguna AWS CloudFormation

Menciptakan lingkungan di Elastic Beanstalk

Bab ini menjelaskan cara membuat dan mengelola lingkungan Elastic Beanstalk Anda. Halaman pengantar ini memberikan ikhtisar pembaruan, pemeliharaan, dan konfigurasi yang akan Anda terapkan seiring waktu seiring berkembangnya aplikasi dan lingkungan Anda.

Fungsi lingkungan

Anda dapat membuat dan mengelola lingkungan terpisah untuk pengembangan, pengujian, dan penggunaan produksi, dan Anda dapat men-deploy versi apa pun aplikasi Anda ke lingkungan apa pun. Lingkungan dapat berjalan lama atau sementara. Ketika Anda mengakhiri lingkungan, Anda dapat menyimpan konfigurasi untuk membuatnya kembali nanti.

Penerapan aplikasi

Ketika Anda mengembangkan aplikasi, Anda akan sering men-deploy-nya, mungkin untuk beberapa lingkungan yang berbeda dengan tujuan yang berbeda. Elastic Beanstalk memungkinkan Anda mengonfigurasi bagaimana deployment dilakukan. Anda dapat men-deploy ke semua instans di lingkungan Anda secara bersamaan, atau membagi deployment ke dalam batch-batch dengan deployment bergulir.

Perubahan konfigurasi

Perubahan konfigurasi diproses secara terpisah dari deployment, dan memiliki ruang lingkup mereka sendiri. Misalnya, jika Anda mengubah jenis EC2 instance yang menjalankan aplikasi Anda, semua instance harus diganti. Di sisi lain, jika Anda memodifikasi konfigurasi penyeimbang beban lingkungan, perubahan tersebut dapat dilakukan di tempat tanpa mengganggu layanan atau menurunkan kapasitas. Anda juga dapat menerapkan perubahan konfigurasi yang memodifikasi instans di lingkungan Anda dalam batch-batch dengan pembaruan konfigurasi bergulir.

> **Note**
>
> Modifikasi sumber daya di lingkungan Anda hanya dengan menggunakan Elastic Beanstalk. Jika Anda memodifikasi sumber daya menggunakan konsol layanan lain, perintah CLI, atau SDKs, Elastic Beanstalk tidak akan dapat memantau status sumber daya tersebut secara akurat, dan Anda tidak akan dapat menyimpan konfigurasi atau membuat ulang lingkungan dengan andal. Perubahan di luar band juga dapat menyebabkan masalah saat memperbarui atau mengakhiri lingkungan.

Pembaruan platform

Ketika Anda meluncurkan lingkungan, Anda memilih versi platform. Kami memperbarui platform secara berkala dengan versi platform baru untuk memberikan peningkatan performa dan fitur baru. Anda dapat memperbarui lingkungan Anda ke versi platform terbaru kapan pun. Lihat panduan AWS Elastic Beanstalk Platform untuk daftar platform yang didukung dan riwayat versi platform yang menyertakan rentang tanggal saat ini.

Pilihan arsitektur

Saat aplikasi Anda tumbuh dalam kompleksitas, Anda dapat membaginya menjadi beberapa komponen, masing-masing berjalan di lingkungan yang terpisah. Untuk beban kerja yang berjalan lama, Anda dapat meluncurkan lingkungan pekerja yang memproses tugas dari antrean Amazon Simple Queue Service (Amazon SQS).

Topik

- Menggunakan konsol manajemen lingkungan Elastic Beanstalk
- Membuat lingkungan Elastic Beanstalk
- Mengelola beberapa lingkungan Elastic Beanstalk sebagai grup dengan EB CLI
- Men-deploy aplikasi ke lingkungan Elastic Beanstalk
- Perubahan konfigurasi
- Memperbarui versi platform lingkungan Elastic Beanstalk Anda
- Membatalkan pembaruan konfigurasi lingkungan dan deployment aplikasi
- Membangun kembali lingkungan Elastic Beanstalk
- Jenis lingkungan
- Lingkungan pekerja Elastic Beanstalk
- Membuat tautan antara lingkungan Elastic Beanstalk
- Memulihkan lingkungan Elastic Beanstalk Anda dari keadaan tidak valid

Menggunakan konsol manajemen lingkungan Elastic Beanstalk

Bagian ini menjelaskan bagaimana Anda dapat mengelola lingkungan Elastic Beanstalk menggunakan konsol manajemen lingkungan. Konsol menyediakan kemampuan untuk mengelola konfigurasi lingkungan Anda dan melakukan tindakan umum. Tindakan ini termasuk memulai ulang

server web yang berjalan di lingkungan Anda, mengkloning lingkungan Anda, atau membangun kembali lingkungan Anda dari scratch.

Topik

- Mengakses konsol manajemen lingkungan
- Panel ikhtisar lingkungan
- Detail lingkungan
- Tindakan lingkungan

Mengakses konsol manajemen lingkungan

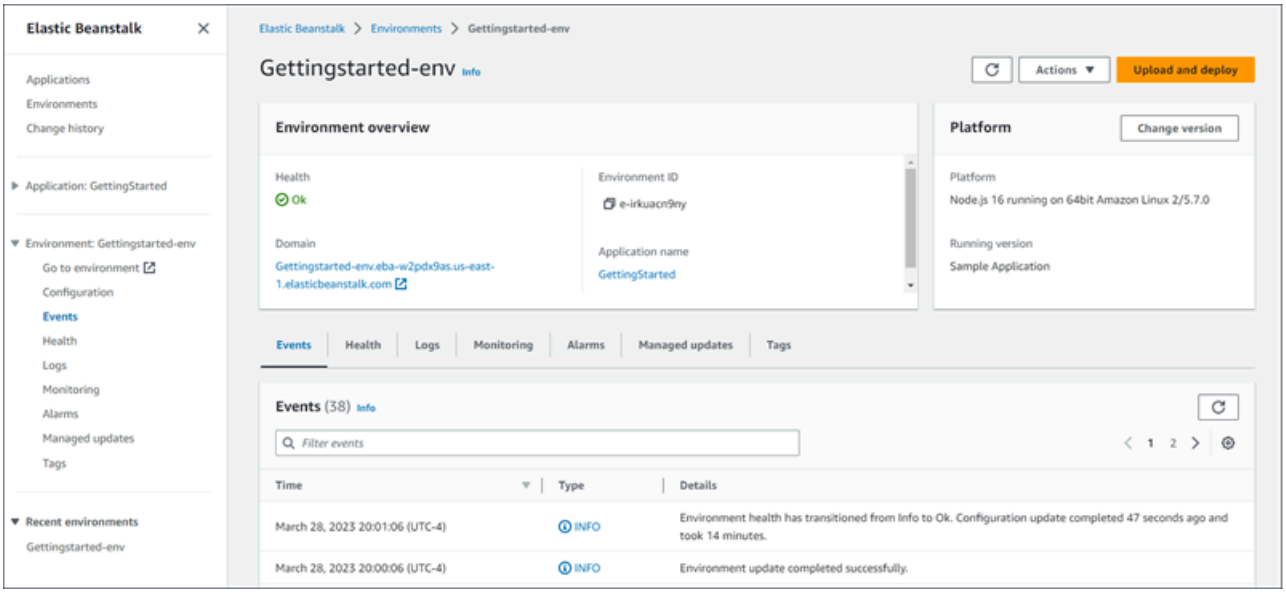
Prosedur berikut menyediakan langkah-langkah untuk meluncurkan konsol manajemen lingkungan.

Jika Anda sudah masuk ke konsol Elastic Beanstalk, Anda juga dapat meluncurkan halaman manajemen lingkungan dari. Konsol pengelola aplikasi Pilih lingkungan dari daftar untuk menampilkan detail konsol manajemen untuk lingkungan yang dipilih.

Untuk mengakses konsol manajemen lingkungan

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.

Gambar berikut mengilustrasikan konsol manajemen lingkungan.



Panel atas adalah halaman ikhtisar Lingkungan. Ini menunjukkan informasi tingkat atas tentang lingkungan Anda.

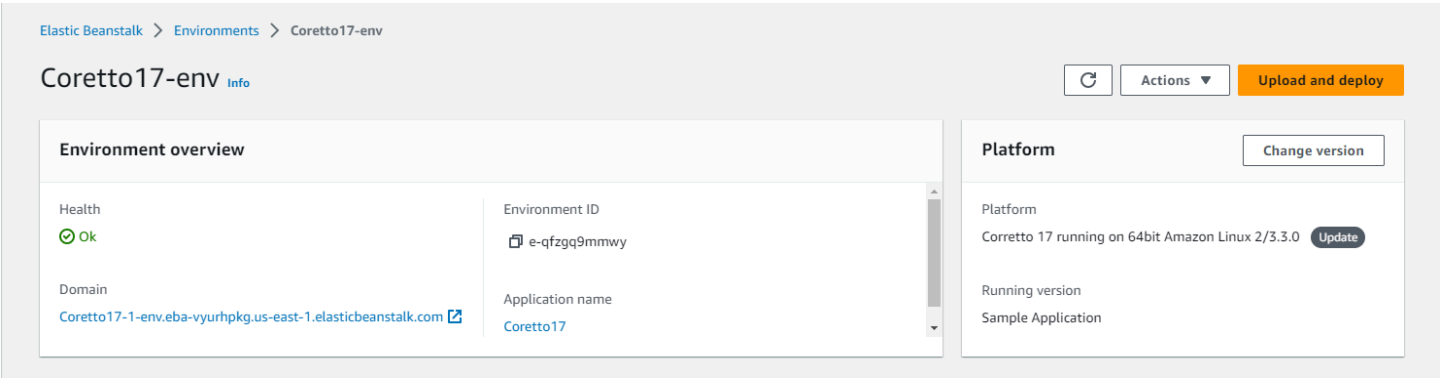
Bagian bawah halaman menampilkan tab yang memberikan informasi lebih rinci. Tab Acara ditampilkan secara default. Halaman yang ditautkan ke tab, juga tercantum di panel navigasi kiri di bawah lingkungan.

Panel navigasi konsol menunjukkan nama aplikasi yang diterapkan ke lingkungan, dengan halaman manajemen aplikasi terkait. Nama lingkungan juga ditampilkan pada halaman navigasi, diikuti oleh halaman manajemen lingkungan. Tautan yang tercantum di bawah nama lingkungan juga menyertakan Pergi ke lingkungan dan Konfigurasi, selain halaman tab yang disebutkan sebelumnya.

Panel ikhtisar lingkungan

Topik ini menjelaskan informasi yang disediakan panel ikhtisar Lingkungan. Ini menunjukkan informasi tingkat atas tentang lingkungan Anda dan terletak di bagian atas konsol manajemen lingkungan.

Gambar berikut menampilkan panel Ikhtisar Lingkungan.



Kondisi

Kondisi lingkungan secara keseluruhan. Jika kesehatan lingkungan Anda menurun, tampilan menyebabkan tautan ditampilkan di sebelah kesehatan lingkungan. Pilih tautan ini untuk melihat tab Kesehatan dengan detail selengkapnya.

Domain

Domain lingkungan, atau URL, terletak di bagian atas halaman ikhtisar Lingkungan, di bawah Kesehatan lingkungan. Ini adalah URL dari aplikasi web yang dijalankan lingkungan. Anda dapat meluncurkan aplikasi dengan memilih URL.

Id lingkungan

ID lingkungan. Ini adalah ID internal yang dihasilkan saat lingkungan dibuat.

Nama aplikasi

Nama aplikasi yang digunakan dan berjalan di lingkungan Anda.

Versi berjalan

Nama versi aplikasi yang digunakan dan di-deploy di lingkungan Anda. Pilih Unggah dan deploy untuk mengunggahan paket sumber dan men-deploy ke lingkungan Anda. Opsi ini menciptakan versi aplikasi baru.

Platform

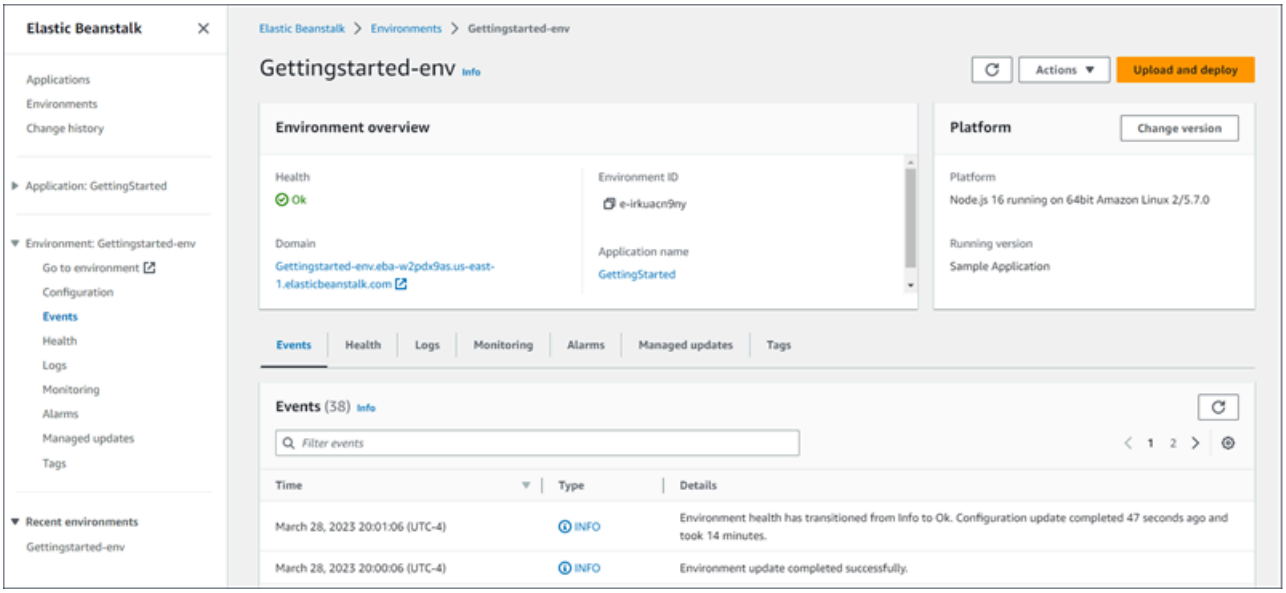
Nama versi platform yang berjalan di lingkungan Anda. Biasanya, ini terdiri dari arsitektur, sistem operasi (OS), bahasa, dan server aplikasi (secara kolektif dikenal sebagai cabang platform), dengan nomor versi platform tertentu.

Jika versi platform Anda bukan yang terbaru tersedia, maka label status akan ditampilkan di sebelahnya di bagian Platform. Label Pembaruan menunjukkan bahwa meskipun versi platform didukung, versi yang lebih baru tersedia. Versi platform juga dapat diberi label sebagai Usang atau Pensiun. Pilih Ubah versi untuk memperbarui cabang platform Anda ke versi yang lebih baru. Untuk informasi selengkapnya tentang status versi platform, lihat bagian Cabang Platform di bagianGlosarium Platform Elastic Beanstalk. Gambar sebelumnya di halaman ini menggambarkan label status Perbarui untuk platform yang diberikan.

Detail lingkungan

Topik ini menjelaskan informasi tambahan yang disediakan konsol manajemen lingkungan dari panel navigasi kiri dan halaman tab.

Gambar berikut mengilustrasikan konsol manajemen lingkungan.



Bagian bawah konsol manajemen lingkungan mencantumkan tab yang memberikan informasi yang lebih rinci dan beragam tentang lingkungan. Anda dapat memilih halaman tab atau label halaman dari panel navigasi kiri.

Dari panel navigasi kiri konsol dengan nama lingkungan, ada dua pilihan yang tidak ada di halaman tab. Ini adalah Pergi ke lingkungan dan Konfigurasi.

> **Note**
>
> Pilih Pergi ke lingkungan untuk meluncurkan aplikasi Anda.

Konfigurasi

Gunakan halaman Konfigurasi di panel navigasi kiri untuk melihat dan memperbarui pengaturan konfigurasi saat ini untuk lingkungan Anda dan sumber dayanya. Ini termasuk konfigurasi jaringan, konfigurasi basis data, penyeimbangan beban, pemberitahuan, pengaturan pemantauan kesehatan, konfigurasi pembaruan platform terkelola, konfigurasi penerapan, streaming log instance, CloudWatch integrasi AWS X-Ray, pengaturan server proxy, properti lingkungan, dan opsi khusus platform. Gunakan pengaturan di halaman ini untuk menyesuaikan perilaku lingkungan Anda selama deployment, mengaktifkan fitur tambahan, dan mengubah jenis instans dan pengaturan lain yang Anda pilih selama pembuatan lingkungan.

Untuk informasi selengkapnya, lihat Mengonfigurasi lingkungan Elastic Beanstalk.

Peristiwa

Halaman Peristiwa menampilkan alur kejadian untuk lingkungan Anda. Elastic Beanstalk mengeluarkan pesan acara setiap kali Anda berinteraksi dengan lingkungan, dan ketika salah satu sumber daya lingkungan Anda dibuat atau dimodifikasi sebagai hasilnya.

Untuk informasi selengkapnya, lihat Melihat alur kejadian lingkungan Elastic Beanstalk.

Kondisi

Jika pemantauan kesehatan yang ditingkatkan diaktifkan, halaman ini mencantumkan EC2 contoh di lingkungan Anda dan informasi kesehatan langsung untuk setiap contoh.

Halaman kesehatan keseluruhan menunjukkan data kesehatan sebagai rata-rata untuk semua contoh lingkungan Anda digabungkan.

Panel kesehatan instans yang Ditingkatkan menampilkan informasi kesehatan langsung untuk setiap EC2 instans individu di lingkungan Anda. Pemantauan kondisi yang ditingkatkan memungkinkan Elastic Beanstalk untuk memantau sumber daya di lingkungan Anda secara ketat sehingga dapat menilai kondisi aplikasi Anda dengan lebih akurat.

Ketika pemantauan kondisi yang ditingkatkan diaktifkan, halaman ini menampilkan informasi tentang permintaan yang disajikan oleh instans di lingkungan Anda dan metrik dari sistem operasi, termasuk latensi, beban, dan utilisasi CPU.

Untuk informasi selengkapnya, lihat Peningkatan pelaporan dan pemantauan kesehatan di Elastic Beanstalk.

Log

Halaman Log memungkinkan Anda mengambil log dari EC2 instance di lingkungan Anda. Ketika Anda meminta log, Elastic Beanstalk mengirimkan perintah ke instans, yang kemudian mengunggah log ke bucket penyimpanan Elastic Beanstalk Anda di Amazon S3. Ketika Anda meminta log pada halaman ini, Elastic Beanstalk secara otomatis menghapusnya dari Amazon S3 setelah 15 menit.

Anda juga dapat mengonfigurasi instans lingkungan Anda untuk mengunggah log ke Amazon S3 untuk penyimpanan permanen setelah instans telah dirotasi secara lokal.

Untuk informasi selengkapnya, lihat Melihat log dari EC2 instans Amazon di lingkungan Elastic Beanstalk Anda.

Pemantauan

Halaman Pemantauan menampilkan gambaran umum informasi kondisi untuk lingkungan Anda. Ini termasuk set metrik default yang disediakan oleh Elastic Load Balancing dan EC2 Amazon, dan grafik yang menunjukkan bagaimana kesehatan lingkungan telah berubah dari waktu ke waktu.

Untuk informasi selengkapnya, lihat Memantau kesehatan lingkungan di konsol AWS manajemen.

Alarm

Alarm yang sudah ada menampilkan informasi tentang alarm apa pun yang telah Anda konfigurasikan untuk lingkungan Anda. Anda dapat menggunakan opsi di halaman ini untuk membuat atau menghapus alarm.

Untuk informasi selengkapnya, lihat Mengelola alarm.

Pembaruan yang terkelola

Halaman Pembaruan terkelola menampilkan informasi tentang pembaruan platform terkelola yang akan datang dan selesai serta penggantian instans.

Fitur pembaruan terkelola memungkinkan Anda mengonfigurasi lingkungan Anda untuk memperbarui ke versi platform terbaru secara otomatis selama jendela pemeliharaan mingguan yang Anda pilih. Di sela-sela rilis platform, Anda dapat memilih agar lingkungan Anda mengganti semua EC2 instans Amazonnya selama jendela pemeliharaan. Hal ini dapat mengurangi masalah yang terjadi ketika aplikasi Anda berjalan untuk jangka waktu yang lama.

Untuk informasi selengkapnya, lihat Pembaruan platform yang dikelola.

Tanda

Halama Tag menunjukkan tag yang Elastic Beanstalk terapkan ke lingkungan saat Anda membuatnya, dan setiap tag yang Anda tambahkan. Anda dapat menambahkan, menyunting, dan menghapus tag kustom. Anda tidak dapat menyunting atau menghapus tag yang Elastic Beanstalk diterapkan.

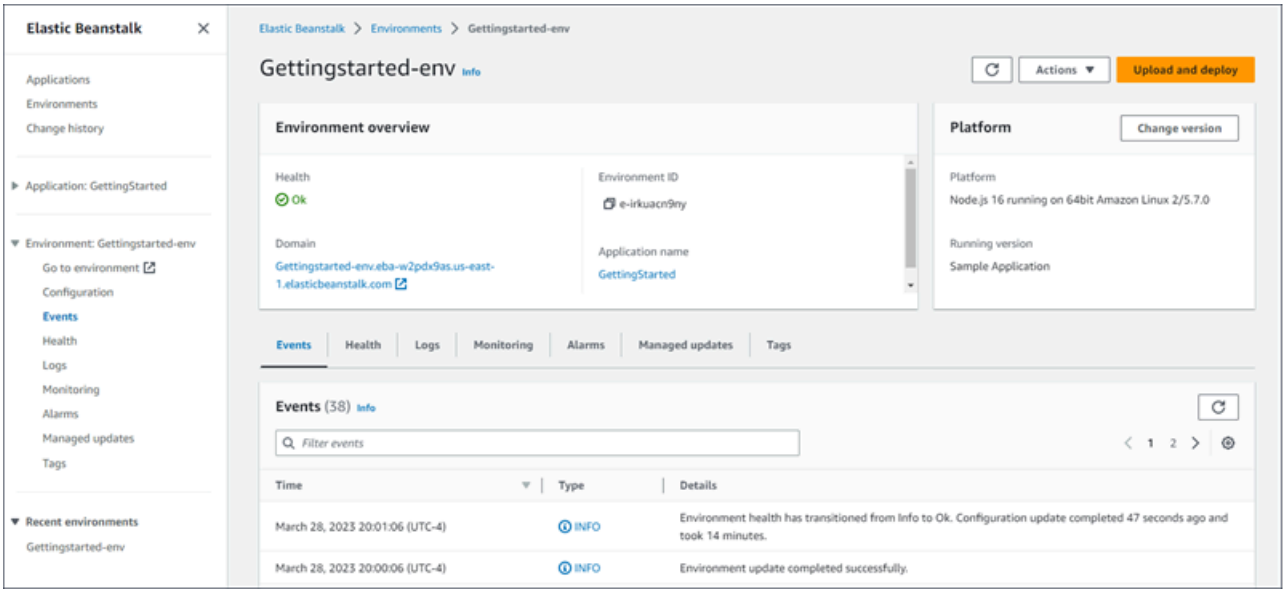
Tag lingkungan diterapkan untuk setiap sumber daya yang Elastic Beanstalk menciptakan untuk mendukung aplikasi Anda.

Untuk informasi selengkapnya, lihat Menandai sumber daya di lingkungan Elastic Beanstalk Anda.

Tindakan lingkungan

Topik ini menjelaskan operasi umum yang dapat Anda pilih untuk dilakukan di lingkungan Anda dari menu tarik-turun Tindakan di konsol manajemen lingkungan.

Gambar berikut mengilustrasikan konsol manajemen lingkungan. Menu tarik-turun Tindakan ada di sisi kanan header yang menampilkan nama lingkungan, di sebelah tombol Refresh.



> **Note**
>
> Beberapa tindakan hanya tersedia dalam kondisi tertentu, tetap dinonaktifkan sampai kondisi yang tepat terpenuhi.

Memuat konfigurasi

Memuat konfigurasi yang telah disimpan sebelumnya. Konfigurasi disimpan ke aplikasi Anda dan dapat dimuat oleh lingkungan yang terkait. Jika Anda telah membuat perubahan pada konfigurasi lingkungan, Anda dapat memuat konfigurasi yang disimpan untuk membatalkan perubahan tersebut. Anda juga dapat memuat konfigurasi yang Anda simpan dari lingkungan yang berbeda yang menjalankan aplikasi yang sama untuk menyebarkan perubahan konfigurasi antara mereka.

Simpan konfigurasi

Simpan konfigurasi saat ini dari lingkungan untuk aplikasi Anda. Sebelum Anda membuat perubahan pada konfigurasi lingkungan Anda, simpan konfigurasi saat ini sehingga Anda dapat memutar

kembali nanti, jika diperlukan. Anda juga dapat menerapkan konfigurasi yang tersimpan saat meluncurkan lingkungan baru.

Swap lingkungan Domain () URLs

Tukar CNAME lingkungan saat ini dengan lingkungan baru. Setelah tukar CNAME, semua lalu lintas ke aplikasi menggunakan URL lingkungan masuk ke lingkungan baru. Ketika Anda siap untuk men-deploy versi baru dari aplikasi Anda, Anda dapat meluncurkan lingkungan yang terpisah di bawah versi baru. Ketika lingkungan baru siap untuk mulai mengambil permintaan, lakukan tukar CNAME untuk memulai perutean lalu lintas ke lingkungan baru. Melakukan hal ini tidak mengganggu layanan Anda. Untuk informasi selengkapnya, lihat Deployment Biru/Hijau dengan Elastic Beanstalk.

Lingkungan pengkloning

Meluncurkan lingkungan baru dengan konfigurasi yang sama dengan lingkungan Anda yang sedang berjalan.

Klon dengan platform terbaru

Kloning lingkungan Anda saat ini dengan versi terbaru dari platform Elastic Beanstalk yang digunakan. Opsi ini tersedia hanya ketika versi yang lebih baru dari platform lingkungan saat ini tersedia untuk digunakan.

Membatalkan operasi saat ini

Menghentikan pembaruan lingkungan yang sedang berlangsung. Menghentikan operasi dapat menyebabkan beberapa instans di lingkungan Anda berada dalam keadaan yang berbeda dari yang lain, tergantung pada seberapa jauh operasi berlangsung. Opsi ini hanya tersedia bila lingkungan Anda sedang diperbarui.

Mulai ulang server aplikasi

Mulai ulang server web yang berjalan pada instans lingkungan Anda. Opsi ini tidak menghentikan atau memulai ulang AWS sumber daya apa pun. Jika lingkungan Anda bertindak aneh dalam menanggapi beberapa permintaan buruk, memulai ulang server aplikasi dapat memulihkan fungsionalitas sementara ketika Anda memecahkan masalah akar penyebabnya.

Lingkungan membangun ulang

Mengakhiri semua sumber daya di lingkungan berjalan dan membangun lingkungan baru dengan pengaturan yang sama. Operasi ini memakan waktu beberapa menit, sama dengan jumlah waktu

yang dibutuhkan untuk men-deploy lingkungan baru dari scratch. Setiap instans Amazon RDS yang berjalan di tingkat data lingkungan Anda akan dihapus selama membangun kembali. Jika Anda membutuhkan data, buat snapshot. Anda dapat membuat snapshot secara manual di konsol RDS atau konfigurasikan Kebijakan Penghapusan tingkat data Anda untuk membuat snapshot secara otomatis sebelum menghapus instans. Ini adalah pengaturan default saat Anda membuat tingkat data.

Mengakhiri lingkungan

Mengakhiri semua sumber daya di lingkungan berjalan dan menghapus lingkungan dari aplikasi. Jika Anda memiliki instance RDS yang berjalan di tingkat data dan Anda perlu menyimpan datanya, pastikan kebijakan penghapusan database disetel ke salah satu atau. `Snapshot` `Retain` Untuk informasi selengkapnya, lihat Siklus hidup basis data di bagian Mengkonfigurasi lingkungan dari panduan ini.

Membuat lingkungan Elastic Beanstalk

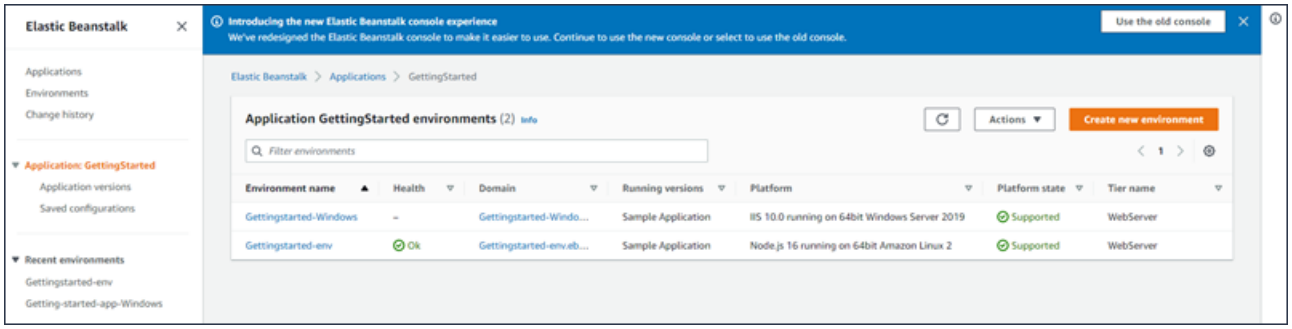
Prosedur berikut meluncurkan lingkungan baru yang menjalankan aplikasi default. Langkah-langkah ini disederhanakan untuk mengaktifkan lingkungan Anda dan berjalan dengan cepat, menggunakan nilai pilihan default.

> **Catatan tentang izin**
>
> Mmebuat lingkungan memerlukan izin di kebijakan pengelolaan akses penuh Elastic Beanstalk. Lihat Kebijakan pengguna Elastic Beanstalk untuk detail selengkapnya.

Untuk meluncurkan lingkungan dengan aplikasi sampel (konsol)

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Pada panel navigasi, pilih Aplikasi. Pilih aplikasi yang ada dalam daftar. Anda juga dapat memilih untuk membuatnya, mengikuti instruksi diMengelola aplikasi.
3. Pada halaman ikhtisar aplikasi, pilih Buat lingkungan baru.

   Gambar berikut menampilkan halaman ikhtisar aplikasi.



Ini meluncurkan wizard Create environment. Wizard menyediakan serangkaian langkah bagi Anda untuk membuat lingkungan baru.

4. Untuk tingkat Lingkungan, pilih lingkungan server Web atau tingkat lingkungan Worker. Anda tidak dapat mengubah tingkat lingkungan setelah pembuatan.

> **Note**
>
> .NET di platform Windows Server tidak mendukung tingkat lingkungan pekerja.

Bidang informasi Aplikasi default, berdasarkan aplikasi yang Anda pilih sebelumnya.

Dalam informasi Lingkungan mengelompokkan nama Environment default, berdasarkan nama aplikasi. Jika Anda lebih suka nama lingkungan yang berbeda, Anda dapat memasukkan nilai lain di bidang. Anda dapat secara opsional memasukkan nama Domain; jika tidak, Elastic Beanstalk menghasilkan nilai secara otomatis. Anda juga dapat secara opsional memasukkan deskripsi Lingkungan.

5. Untuk Platform, pilih platform dan cabang platform yang sesuai dengan bahasa yang digunakan aplikasi Anda.

> **Note**
>
> Elastic Beanstalk mendukung beberapa versi untuk sebagian besar platform yang tercantum. Secara default, konsol tersebut memilih versi yang direkomendasikan untuk platform dan cabang platform yang Anda pilih. Jika aplikasi Anda memerlukan versi yang berbeda, Anda dapat memilihnya di sini. Untuk informasi tentang versi platform yang didukung, lihat the section called “Platform yang didukung”.

6. Untuk kode Aplikasi, Anda memiliki beberapa pilihan untuk meluncurkan aplikasi sampel.

- Untuk meluncurkan aplikasi sampel default tanpa memasok kode sumber, pilih Contoh aplikasi. Tindakan ini memilih aplikasi satu halaman yang disediakan Elastic Beanstalk untuk platform yang Anda pilih sebelumnya.
- Jika Anda mengunduh contoh aplikasi dari panduan ini atau sumber lain, lakukan langkah-langkah berikut.

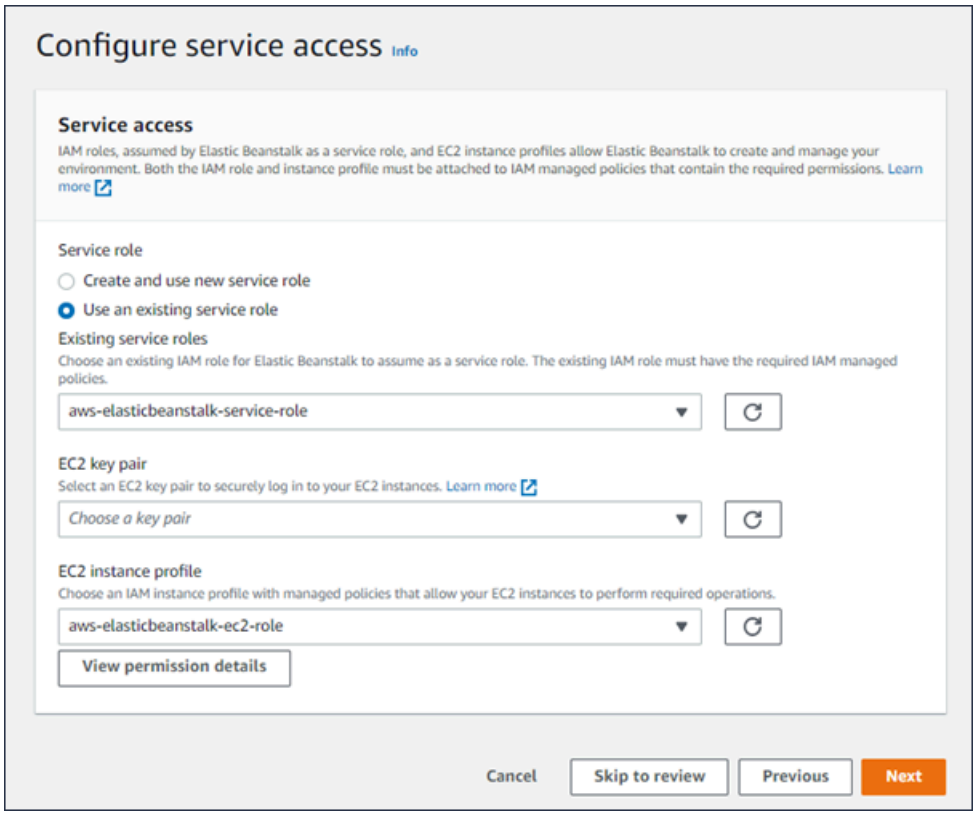
  a. Pilih Unggah kode Anda.

  b. Selanjutnya pilih File lokal, lalu di bawah Unggah aplikasi, pilih Pilih file.

  c. Sistem operasi komputer Anda akan memberi Anda antarmuka untuk memilih file lokal yang Anda unduh. Pilih file bundel sumber dan lanjutkan.

7. Untuk Preset, pilih Single instance.
8. Pilih Berikutnya.
9. Halaman akses layanan Konfigurasi ditampilkan.

   Gambar berikut mengilustrasikan halaman akses layanan Konfigurasi.



10. Pilih nilai dari dropdown Peran Layanan yang Ada.
11. (Opsional) Jika sebelumnya Anda membuat EC2 key pair, Anda dapat memilihnya dari dropdown bidang EC2 key pair. Anda akan menggunakannya untuk masuk dengan aman ke EC2 instans

Amazon yang disediakan Elastic Beanstalk untuk aplikasi Anda. Jika Anda melewati langkah ini, Anda selalu dapat membuat dan menetapkan EC2 key pair setelah lingkungan dibuat. Untuk informasi selengkapnya, lihat EC2 key pair.

12. Selanjutnya, kita akan fokus pada daftar dropdown profil EC2 instance. Nilai yang ditampilkan dalam daftar dropdown ini dapat bervariasi, tergantung pada apakah akun Anda sebelumnya telah membuat lingkungan baru.

    Pilih salah satu item berikut, berdasarkan nilai yang ditampilkan dalam daftar Anda.

    - Jika `aws-elasticbeanstalk-ec2-role` ditampilkan dalam daftar dropdown, pilih dari daftar dropdown.
    - Jika nilai lain ditampilkan dalam daftar, dan itu adalah profil EC2 instance default yang ditujukan untuk lingkungan Anda, pilih dari daftar dropdown.
    - Jika daftar dropdown profil EC2 instance tidak mencantumkan nilai apa pun, Anda harus membuat profil instance.

      > **Buat profil instance**
      >
      > Untuk membuat profil instance, kita akan mengambil jalan memutar ke prosedur lain di halaman yang sama ini. Pergi ke akhir prosedur ini dan memperluas prosedur yang berikut, Buat IAM Role EC2 misalnya profil.
      > Selesaikan langkah-langkah di Buat Peran IAM EC2 misalnya profil untuk membuat Peran IAM yang selanjutnya dapat Anda pilih untuk profil EC2 instans. Kemudian kembali ke langkah ini.

      Sekarang Anda telah membuat Peran IAM, dan menyegarkan daftar, itu ditampilkan sebagai pilihan dalam daftar dropdown. Pilih Peran IAM yang baru saja Anda buat dari daftar dropdown profil EC2 instance.

13. Pilih Lewati untuk Meninjau di halaman Konfigurasi akses layanan.

    Ini akan memilih nilai default untuk langkah ini dan melewati langkah-langkah opsional.

14. Halaman Review menampilkan ringkasan dari semua pilihan Anda.

    Untuk lebih menyesuaikan lingkungan Anda, pilih Edit di samping langkah yang menyertakan item apa pun yang ingin Anda konfigurasikan. Anda dapat mengatur opsi berikut hanya selama pembuatan lingkungan:

- Nama lingkungan
- Nama domain
- Versi platform
- Prosesor
- VPC
- Tingkat

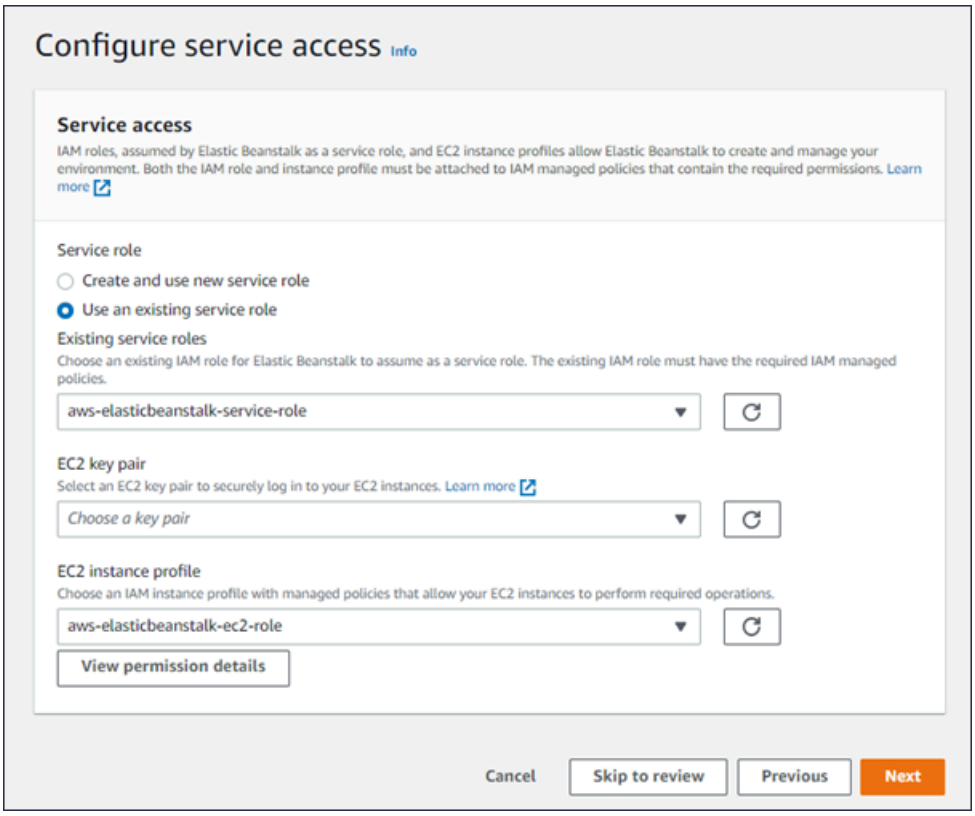
Anda dapat mengubah pengaturan berikut setelah pembuatan lingkungan, tetapi mereka memerlukan instans baru atau sumber daya lain untuk disediakan dan dapat memakan waktu lama untuk menerapkan:

- Jenis instans, volume root, key pair, dan peran AWS Identity and Access Management (IAM)
- Basis data Amazon RDS internal
- Penyeimbang beban

Untuk detail di semua pengaturan yang tersedia, lihat Wizard pembuatan lingkungan baru.

15. Pilih Kirim di bagian bawah halaman untuk menginisialisasi pembuatan lingkungan baru Anda.

Buat Peran IAM EC2 misalnya profil



Untuk membuat profil EC2 instance

1. Pilih Buat peran.
2. Untuk jenis entitas Tepercaya, pilih AWS layanan.
3. Untuk kasus Penggunaan, pilih Elastic Beanstalk — Compute.
4. Pilih Berikutnya.
5. Verifikasi bahwa kebijakan Izin menyertakan yang berikut ini, lalu pilih Berikutnya:
   - `AWSElasticBeanstalkWebTier`
   - `AWSElasticBeanstalkWorkerTier`
   - `AWSElasticBeanstalkMulticontainerDocker`
6. Pilih Buat peran.
7. Kembali ke tab Konfigurasi akses layanan, segarkan daftar, lalu pilih profil EC2 instance yang baru dibuat.

Sementara Elastic Beanstalk membuat lingkungan Anda, Anda dialihkan ke konsol Elastic Beanstalk. Ketika kondisi lingkungan berubah menjadi hijau, pilih URL di samping nama lingkungan untuk melihat aplikasi yang sedang berjalan. URL ini umumnya dapat diakses dari internet kecuali Anda mengonfigurasi lingkungan Anda untuk menggunakan VPC kustom dengan penyeimbang beban internal.

Topik

- Wizard pembuatan lingkungan baru
- Mengkloning lingkungan Elastic Beanstalk
- Mengakhiri lingkungan Elastic Beanstalk
- Buat lingkungan Elastic Beanstalk dengan AWS CLI
- Buat lingkungan Elastic Beanstalk dengan API
- Membangun Meluncurkan URL Sekarang
- Membuat dan memperbarui grup lingkungan Elastic Beanstalk

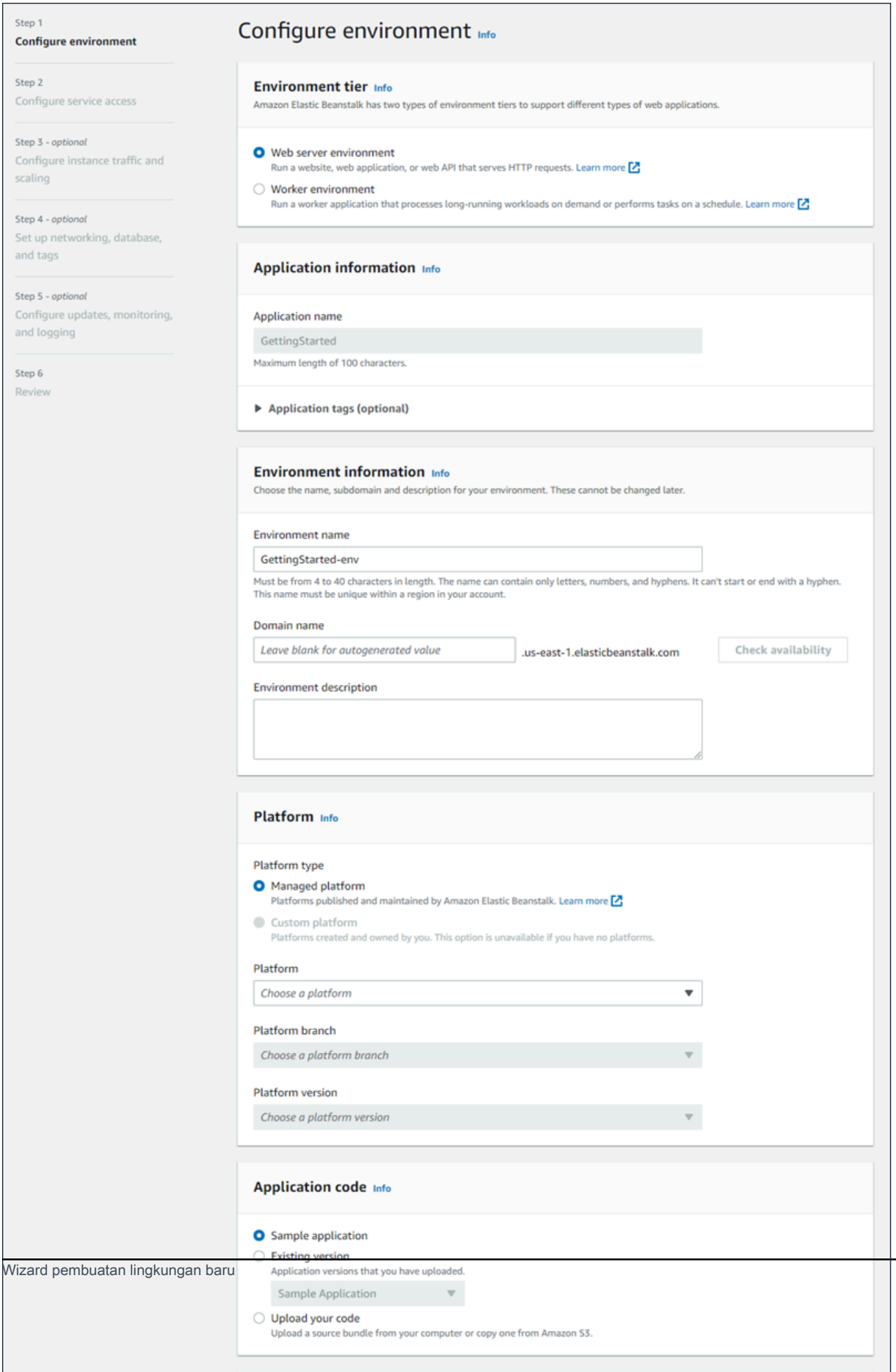
Wizard pembuatan lingkungan baru

Topik ini menjelaskan panduan Buat lingkungan dan semua cara Anda dapat menggunakannya untuk mengonfigurasi lingkungan yang ingin Anda buat.

> **Note**
>
> Di Membuat lingkungan Elastic Beanstalk kami menunjukkan cara meluncurkan wizard Buat lingkungan dan dengan cepat membuat lingkungan dengan nilai default dan pengaturan yang disarankan. Topik saat ini akan memandu Anda melalui semua opsi.

Halaman Wizard

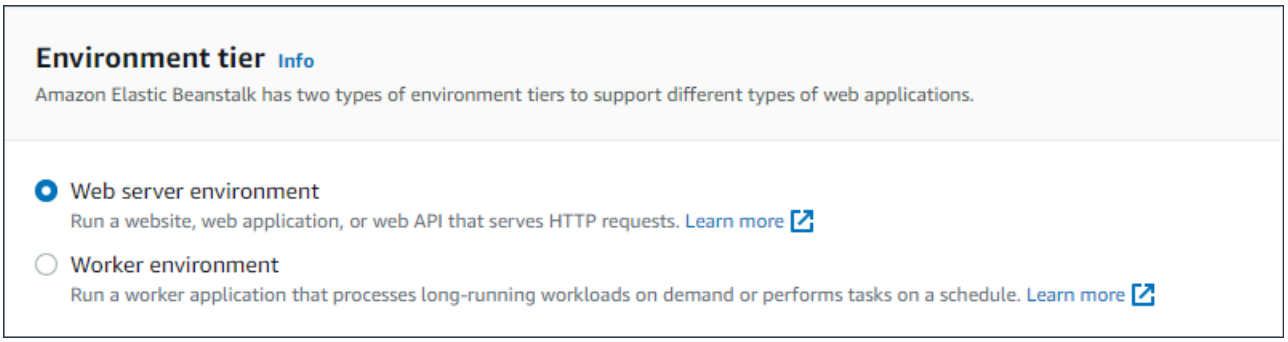
Wizard Create environment menyediakan serangkaian langkah bagi Anda untuk membuat lingkungan baru.



Wizard pembuatan lingkungan baru

Tingkat lingkungan

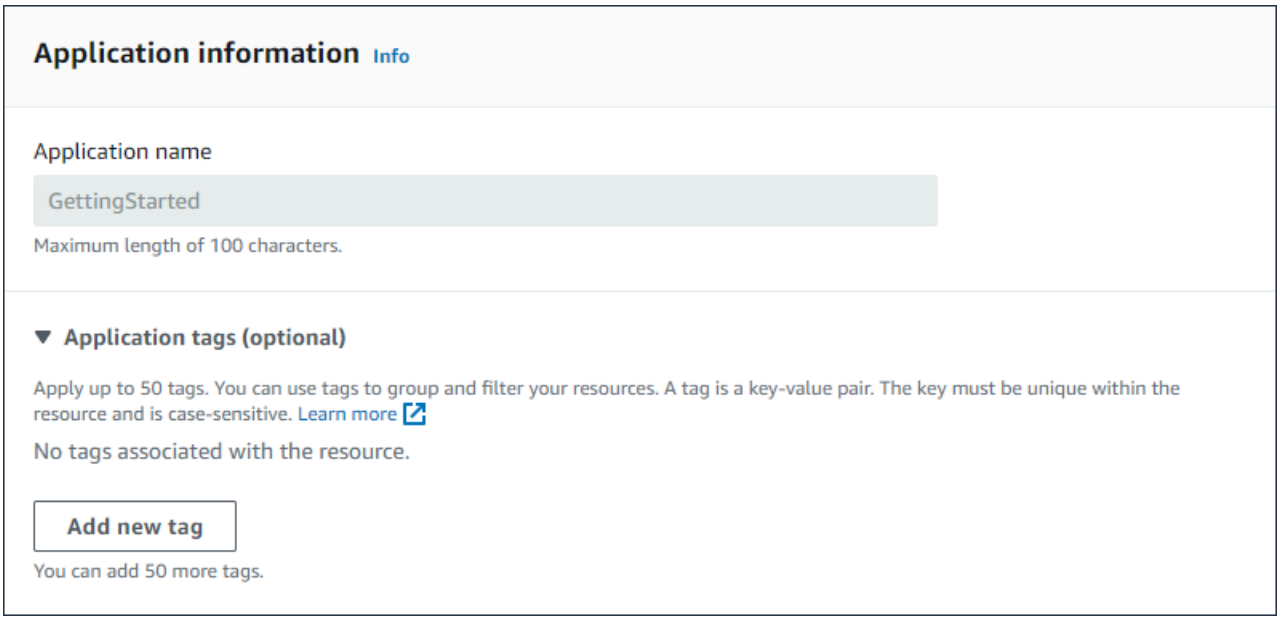
Untuk tingkat lingkungan, pilih lingkungan server Web atau tingkat lingkungan Worker. Anda tidak dapat mengubah tingkat lingkungan setelah pembuatan.



> **Note**
>
> .NET di platform Windows Server tidak mendukung tingkat lingkungan pekerja.

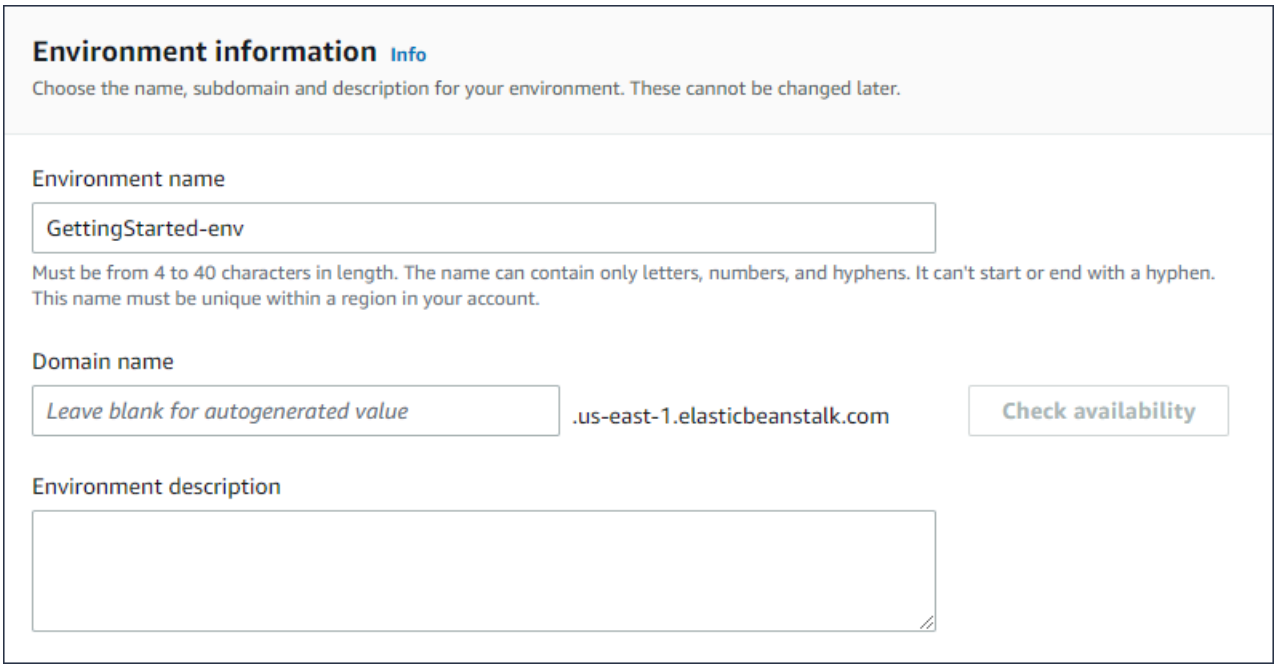
Informasi aplikasi

Jika Anda meluncurkan wizard dengan memilih Buat lingkungan baru dari halaman Ikhtisar aplikasi, maka nama Aplikasi sudah diisi sebelumnya. Jika tidak, masukkan nama aplikasi. Secara opsional, tambahkan tag aplikasi.



Informasi lingkungan

Tetapkan nama dan domain lingkungan, dan buat deskripsi untuk lingkungan Anda. Perhatikan bahwa pengaturan lingkungan ini tidak dapat berubah setelah lingkungan dibuat.



- Nama – Masukkan nama untuk lingkungan. Formulir menyediakan nama yang dihasilkan.
- Domain – (lingkungan server web) Masukkan nama domain yang unik untuk lingkungan Anda. Nama default adalah nama lingkungan. Anda dapat memasukkan nama domain yang berbeda. Elastic Beanstalk menggunakan nama ini untuk membuat CNAME yang unik untuk lingkungan. Untuk memeriksa apakah nama domain yang Anda inginkan tersedia, pilih Periksa Ketersediaan.
- Deskripsi – Masukkan deskripsi untuk lingkungan ini.

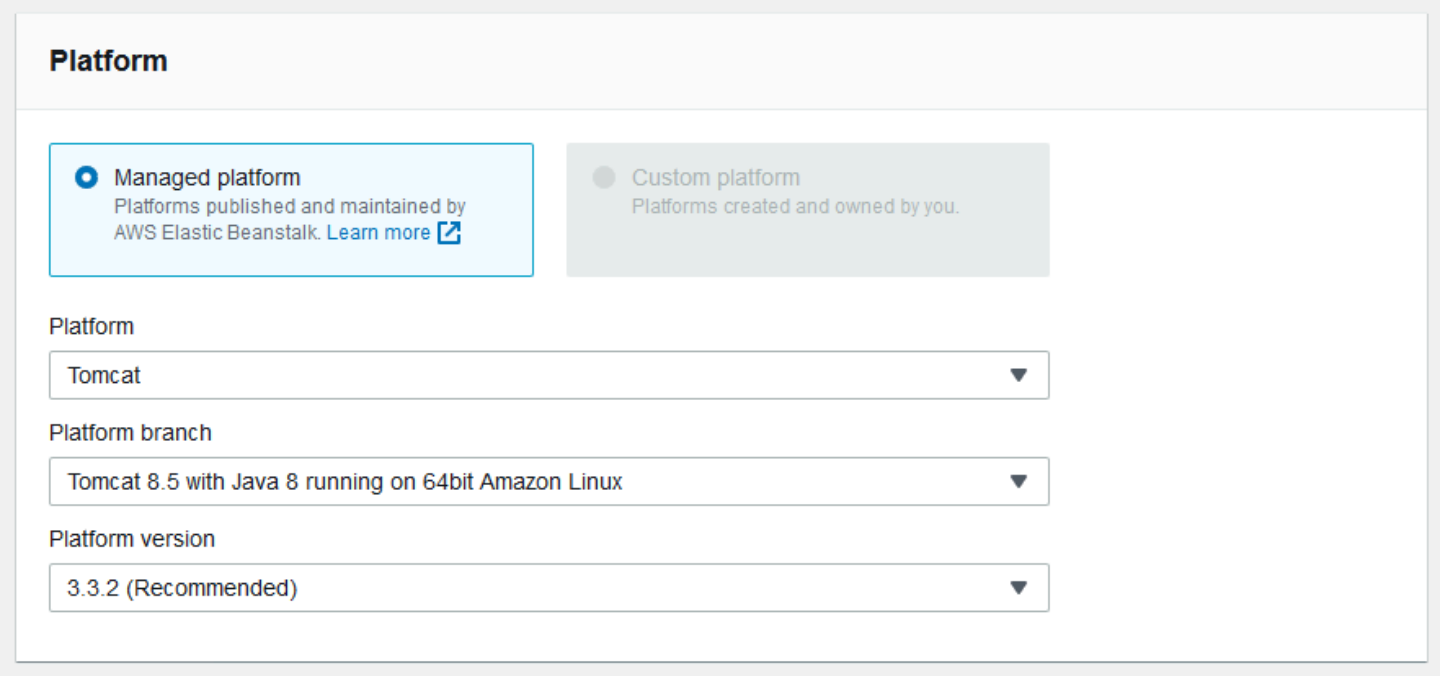
Pilih platform untuk lingkungan baru

Anda dapat membuat lingkungan baru dari dua jenis platform:

- Platform yang dikelola
- Platform kustom

Platform terkelola

Dalam kebanyakan kasus, Anda menggunakan platform yang dikelola Elastic Beanstalk untuk lingkungan baru Anda. Ketika wizard lingkungan baru dimulai, pilihan Platform yang dikelola terpilih secara default.



Pilih platform, cabang platform dalam platform tersebut, dan versi platform tertentu di cabang. Bila Anda memilih cabang platform, versi yang direkomendasikan dalam cabang dipilih secara default. Selain itu, Anda dapat memilih versi platform apa pun yang pernah Anda gunakan sebelumnya.

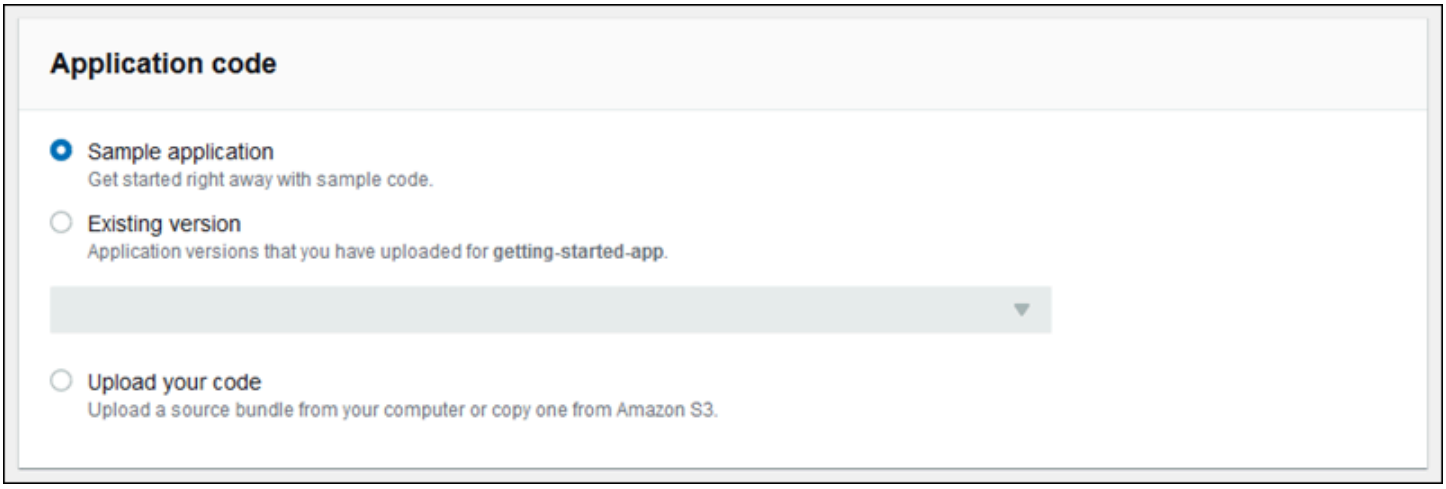
> **Note**
>
> Untuk lingkungan produksi, kami rekomendasikan Anda memilih versi platform di cabang platform yang didukung. Untuk rincian tentang status cabang platform, lihat definisi Cabang platform dalam the section called “Glosarium platform”.

Platform kustom

Jika off-the-shelf platform tidak memenuhi kebutuhan Anda, Anda dapat membuat lingkungan baru dari platform khusus. Untuk menentukan platform khusus, pilih pilihan Platform kustom, dan kemudian pilih salah satu platform kustom yang tersedia. Jika tidak ada platform kustom yang tersedia, opsi ini diredupkan.

Berikan kode aplikasi

Setelah Anda telah memilih platform untuk digunakan, langkah berikutnya adalah memberikan kode aplikasi Anda.

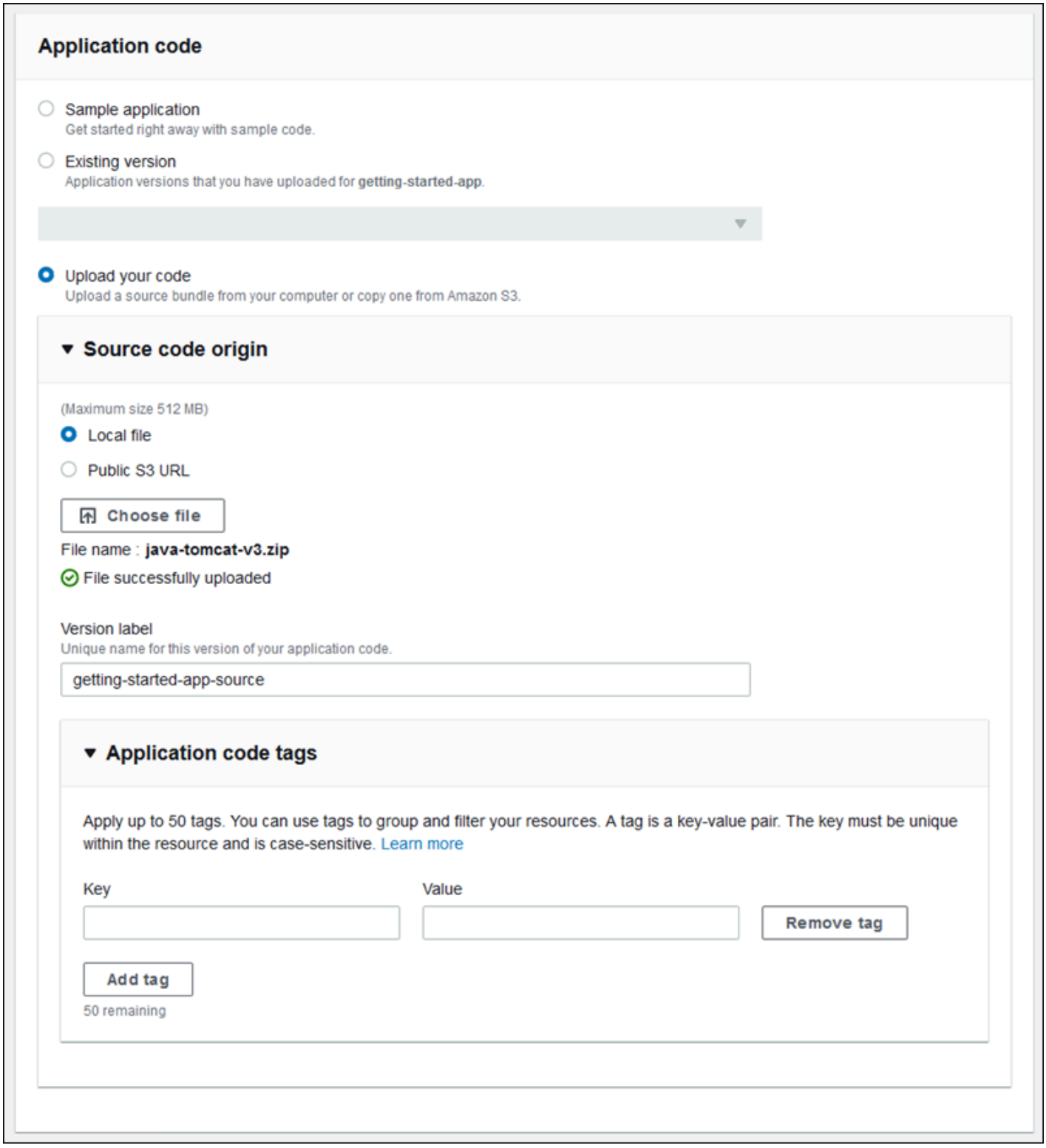


Anda memiliki beberapa pilihan:

- Anda dapat menggunakan aplikasi sampel yang Elastic Beanstalk sediakan untuk setiap platform.
- Anda dapat menggunakan kode yang sudah di-deploy untuk Elastic Beanstalk. Pilih Versi yang sudah dan aplikasi Anda di bagian Kode aplikasi.
- Anda dapat mengunggah kode baru. Pilih Unggah kode Anda, lalu pilih Unggah. Anda dapat mengunggah kode aplikasi baru dari file lokal, atau Anda dapat menentukan URL untuk bucket Amazon S3 yang berisi kode aplikasi Anda.

> **Note**
>
> Tergantung pada versi platform yang Anda pilih, Anda dapat mengunggah aplikasi Anda dalam ZIP paket sumber, file WAR, atau konfigurasi Docker plaintext. Batas ukuran file adalah 500 MB.

Ketika Anda memilih untuk mengunggah kode baru, Anda juga dapat memberikan tanda untuk dikaitkan dengan kode baru Anda. Untuk informasi selengkapnya tentang pemberian penandaan pada versi aplikasi, lihat the section called “Melabeli versi aplikasi”.


Application code
Sample application
Get started right away with sample code.
Existing version
Application versions that you have uploaded for getting-started-app.
Upload your code
Upload a source bundle from your computer or copy one from Amazon S3.
Source code origin
(Maximum size 512 MB)
Local file
Public S3 URL
Choose file
File name : java-tomcat-v3.zip
File successfully uploaded
Version label
Unique name for this version of your application code.
getting-started-app-source
Application code tags
Apply up to 50 tags. You can use tags to group and filter your resources. A tag is a key-value pair. The key must be unique within the resource and is case-sensitive. Learn more
Key
Value
Remove tag
Add tag
50 remaining

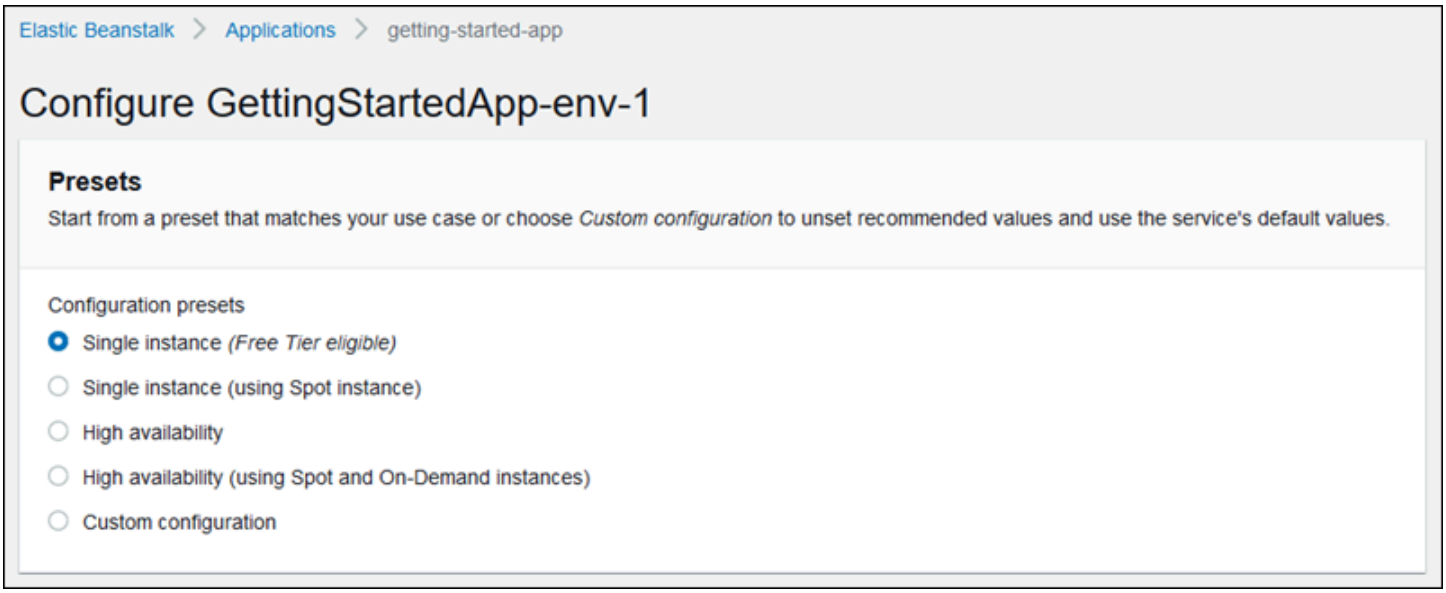
Untuk pembuatan lingkungan yang cepat menggunakan pilihan konfigurasi default, sekarang Anda dapat memilih Buat lingkungan. Pilih Konfigurasi pilihan lainnya untuk perubahan konfigurasi tambahan, seperti yang dijelaskan di bagian berikut.

Halaman konfigurasi wizard

Bila Anda memilih Konfigurasi pilihan lainnya, wizard menunjukkan halaman Konfigurasi. Pada halaman ini Anda dapat memilih pengaturan konfigurasi, mengubah versi platform yang ingin digunakan lingkungan Anda, atau membuat pilihan konfigurasi khusus untuk lingkungan baru.

Pilih konfigurasi prasetel

Pada bagian Prasetel dari halaman, Elastic Beanstalk menyediakan beberapa prasetel konfigurasi untuk kasus penggunaan yang berbeda. Setiap preset mencakup nilai yang direkomendasikan untuk beberapa Pilihan konfigurasi.



Preset Ketersediaan yang tinggi mencakup penyeimbang beban, dan direkomendasikan untuk lingkungan produksi. Pilih preset tersebut jika Anda ingin lingkungan yang dapat menjalankan beberapa instans untuk ketersediaan dan skala yang tinggi dalam menanggapi beban. Preset Satu instans direkomendasikan terutama untuk pengembangan. Dua dari prasetel memungkinkan permintaan instans Spot. Untuk detail tentang konfigurasi kapasitas Elastic Beanstalk, lihat Grup Auto Scaling.

Prasetel terakhir, Konfigurasi khusus, menghapus semua nilai yang direkomendasikan kecuali pengaturan peran dan penggunaan default API. Pilih pilihan ini jika Anda men-deploy paket sumber

dengan file konfigurasi yang mengatur pilihan konfigurasi. Konfigurasi khusus juga dipilih secara otomatis jika Anda mengubah preset konfigurasi Biaya rendah atau Ketersediaan yang tinggi.

Kustomisasi konfigurasi Anda

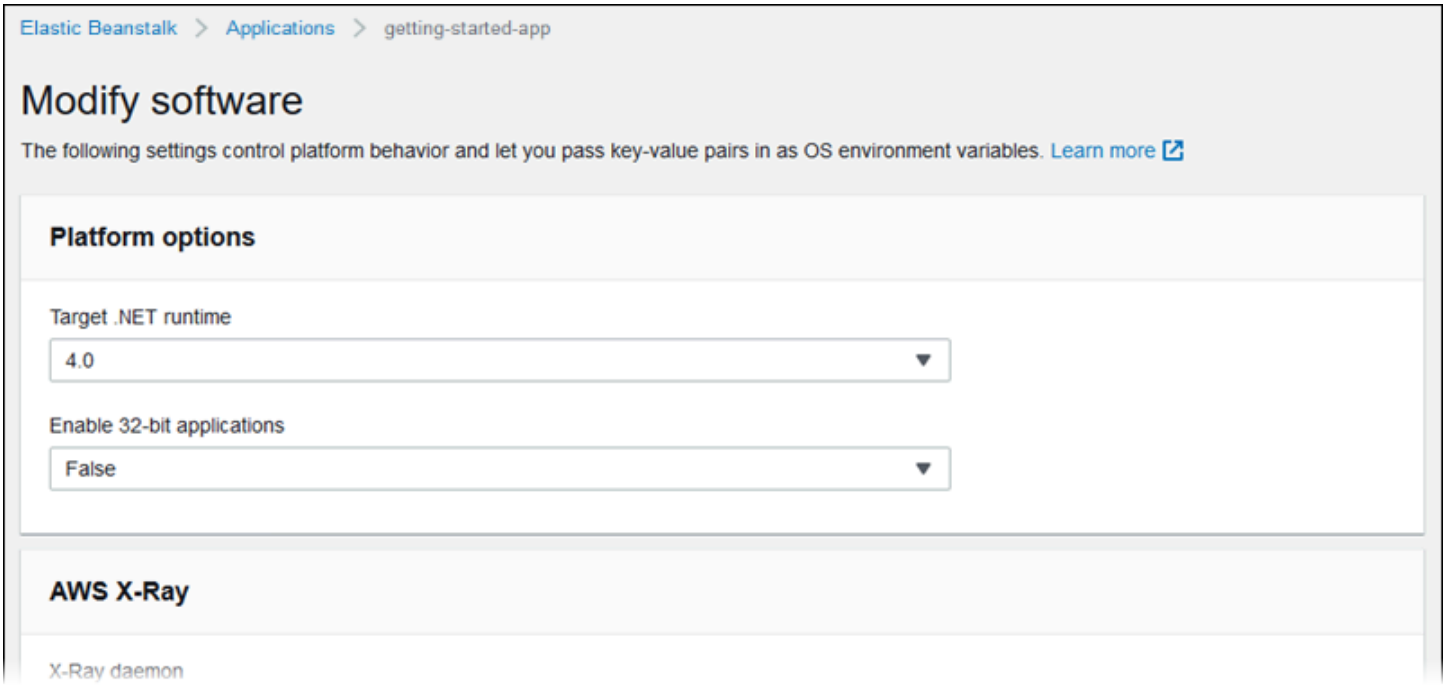
Selain (atau bukan) memilih prasetel konfigurasi, Anda dapat menyempurnakan pilihan konfigurasi di lingkungan Anda. Wizard Konfigurasi wizard menunjukkan beberapa kategori konfigurasi. Setiap kategori konfigurasi menampilkan ringkasan nilai untuk segrup pengaturan konfigurasi. Pilih Edit untuk mengedit grup pengaturan ini.

Kategori konfigurasi

- Pengaturan perangkat lunak
- Instans
- Kapasitas
- Penyeimbang beban
- Pembaruan bergulir dan deployment
- Keamanan
- Pemantauan
- Pembaruan terkelola
- Notifikasi
- Jaringan
- Basis Data
- Tanda
- Lingkungan pekerja

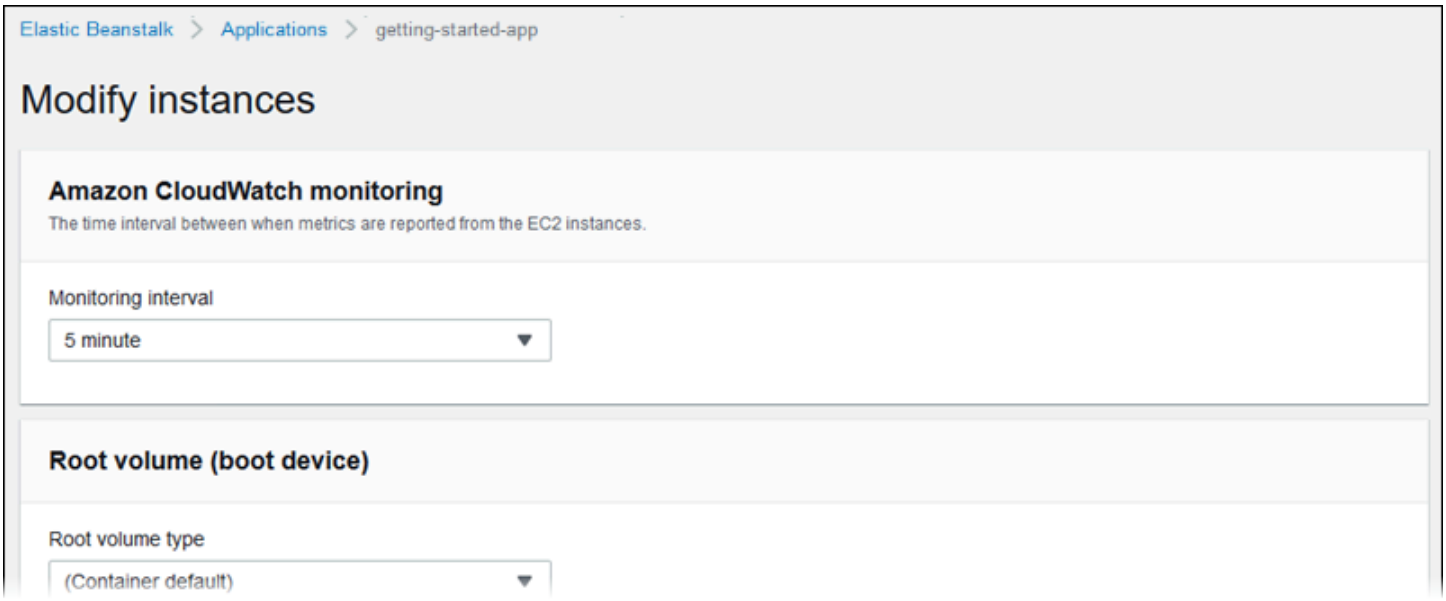
Pengaturan perangkat lunak

Gunakan halaman Modify software configuration untuk mengonfigurasi perangkat lunak pada instans Amazon Elastic Compute Cloud (Amazon EC2) yang menjalankan aplikasi Anda. Anda dapat mengonfigurasi properti lingkungan, AWS X-Ray debugging, penyimpanan dan streaming log instance, dan pengaturan khusus platform. Lihat perinciannya di the section called “Variabel lingkungan dan pengaturan perangkat lunak”.



Instans

Gunakan halaman konfigurasi Modify instance untuk mengonfigurasi EC2 instans Amazon yang menjalankan aplikasi Anda. Lihat perinciannya di the section called “ EC2 Contoh Amazon”.



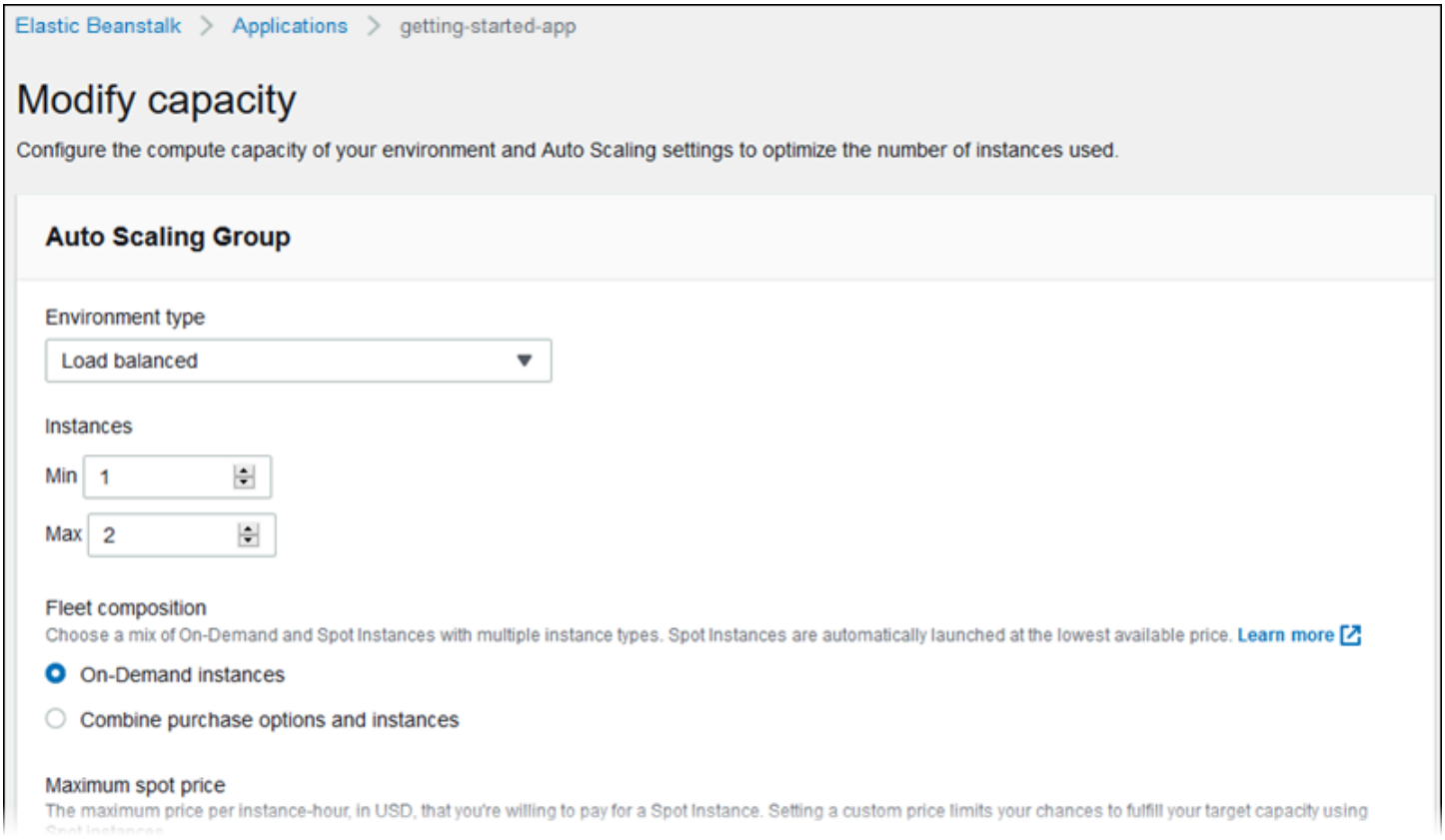
Kapasitas

Gunakan Modifikasi kapasitas untuk mengonfigurasi kapasitas komputasi lingkungan Anda dan pengaturan Grup Auto Scaling untuk mengoptimalkan jumlah dan jenis instans yang Anda gunakan. Anda juga dapat mengubah kapasitas lingkungan Anda berdasarkan pemicu atau jadwal.

Lingkungan dengan beban yang seimbang dapat menjalankan beberapa instans untuk ketersediaan tinggi dan mencegah waktu henti selama pembaruan konfigurasi dan deployment. Dalam lingkungan dengan beban yang seimbang, nama domain dipetakan ke penyeimbang beban. Dalam lingkungan instans tunggal, dipetakan ke alamat IP elastis pada instans.

> **Warning**
>
> Lingkungan instans tunggal tidak siap produksi. Jika instans menjadi tidak stabil selama deployment, atau Elastic Beanstalk berakhir dan mengulang kembali instans selama pembaruan konfigurasi, aplikasi Anda dapat tidak tersedia untuk jangka waktu tertentu. Gunakan lingkungan instans tunggal untuk pengembangan, pengujian, atau pementasan. Gunakan lingkungan dengan beban yang seimbang untuk produksi.

Untuk informasi selengkapnya tentang pengaturan kapasitas lingkungan, lihat the section called “Grup Auto Scaling” danthe section called “ EC2 Contoh Amazon”.

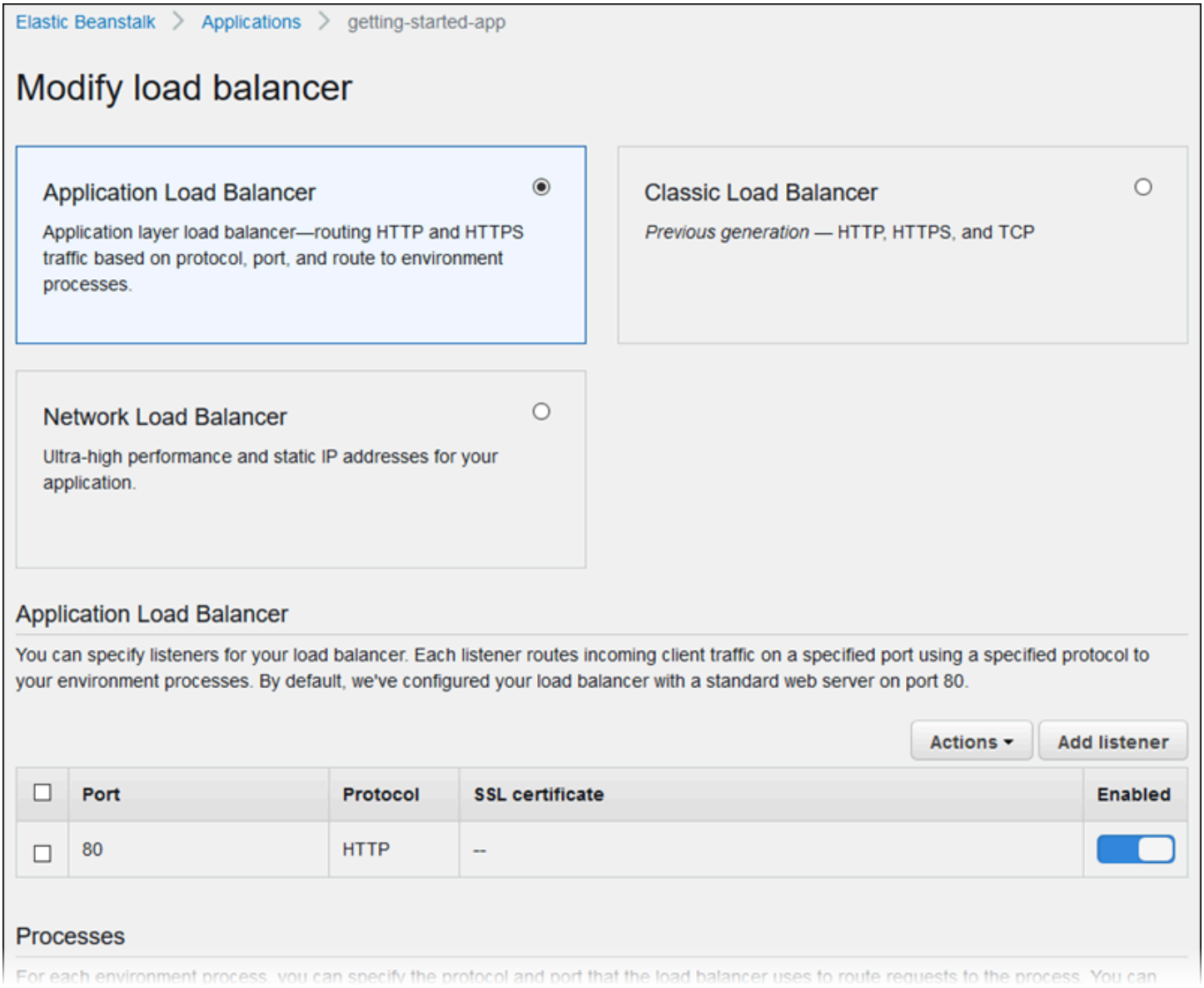


Penyeimbang beban

Gunakan Modifikasi penyeimbang beban untuk memilih jenis penyeimbang beban dan mengonfigurasi pengaturan untuk itu. Dalam lingkungan dengan beban yang seimbang, penyeimbang beban lingkungan Anda adalah titik masuk untuk semua lalu lintas yang menuju aplikasi Anda. Elastic Beanstalk mendukung beberapa jenis penyeimbang beban. Secara default, konsol Elastic Beanstalk menciptakan Application Load Balancer dan mengonfigurasinya untuk melayani lalu lintas HTTP di port 80.

> **Note**
>
> Anda hanya dapat memilih jenis penyeimbang beban lingkungan Anda selama pembuatan lingkungan.

Untuk informasi selengkapnya tentang jenis dan pengaturan penyeimbang beban, lihat the section called “Penyeimbang beban” dan the section called “HTTPS”.

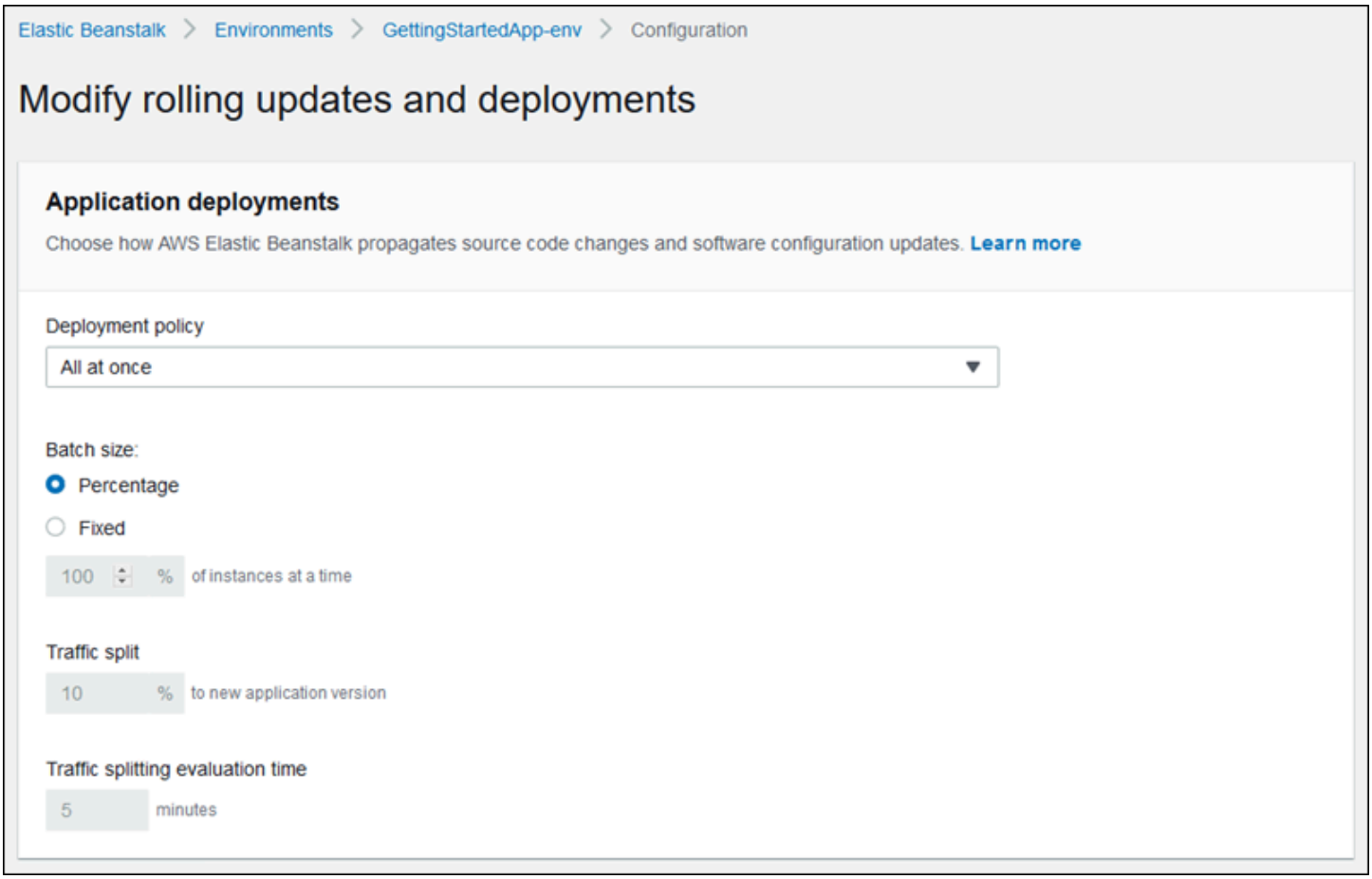


> **Note**
>
> Opsi Classic Load Balancer (CLB) dinonaktifkan pada wizard konsol Create Environment. Jika Anda memiliki lingkungan yang sudah ada yang dikonfigurasi dengan Classic Load Balancer, Anda dapat membuat yang baru dengan mengkloning lingkungan yang ada menggunakan konsol Elastic Beanstalk atau EB CLI. Anda juga memiliki opsi untuk menggunakan EB CLI atau untuk membuat lingkungan baru AWS CLIyang dikonfigurasi dengan Classic Load Balancer. Alat baris perintah ini akan menciptakan lingkungan baru dengan CLB meskipun belum ada di akun Anda.

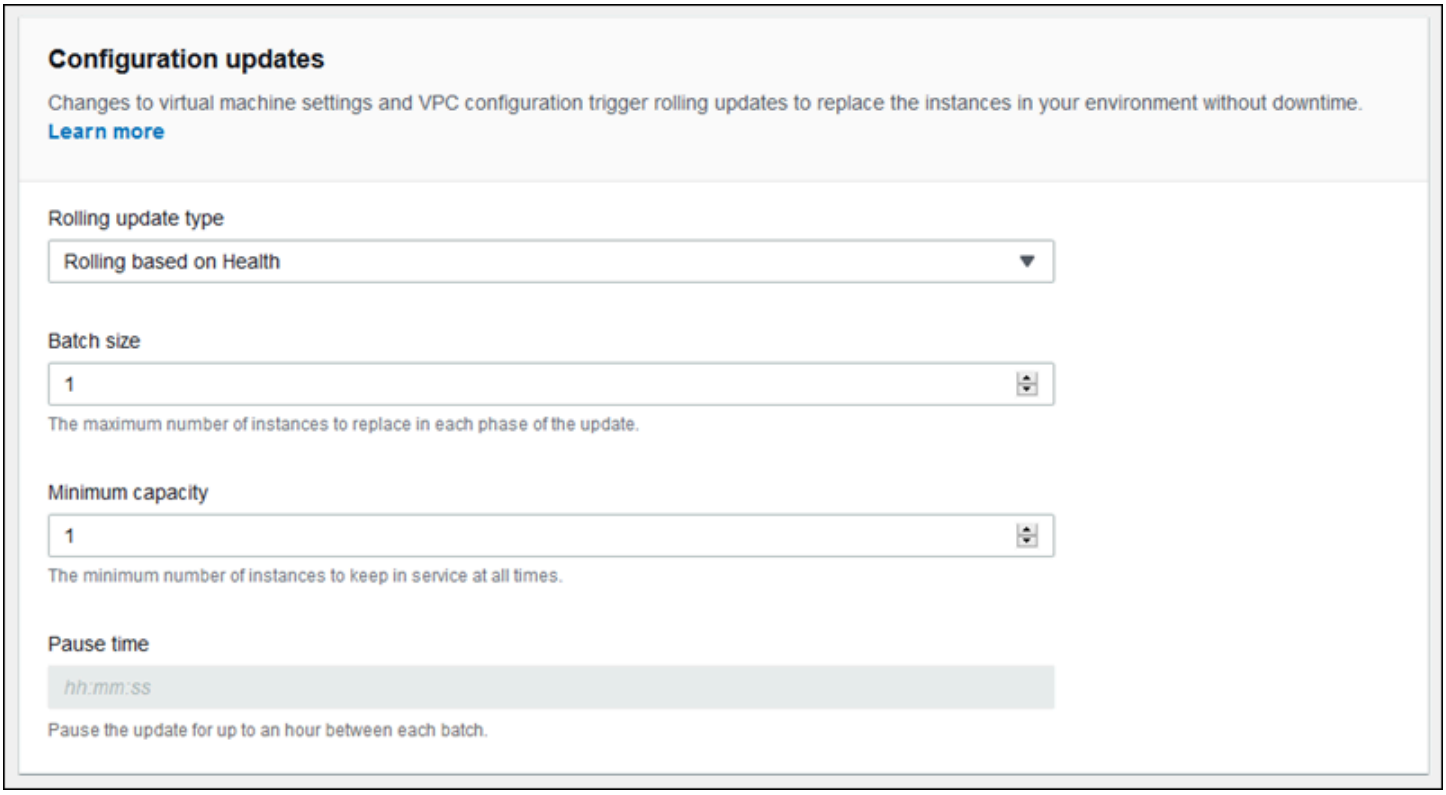
Pembaruan bergulir dan deployment

Gunakan halaman Modifikasi pembaruan bergulir dan deployment untuk mengonfigurasi bagaimana Elastic Beanstalk memproses deployment aplikasi dan pembaruan konfigurasi untuk lingkungan Anda.

Deployment aplikasi terjadi ketika Anda mengunduh paket sumber aplikasi yang diperbarui dan men-deploy ke lingkungan Anda. Untuk informasi selengkapnya tentang konfigurasi deployment, lihat the section called “Pilihan deployment”.



Perubahan konfigurasi yang memodifikasi konfigurasi peluncuran atau pengaturan VPC memerlukan diakhirinya semua instans di lingkungan Anda dan menggantinya. Untuk informasi selengkapnya tentang cara mengatur jenis pembaruan dan pilihan lainnya, lihat the section called “Perubahan konfigurasi”.



Keamanan

Gunakan halaman Konfigurasi akses layanan untuk mengonfigurasi pengaturan keamanan layanan dan instans.

Untuk deskripsi konsep keamanan Elastic Beanstalk, lihat Izin.

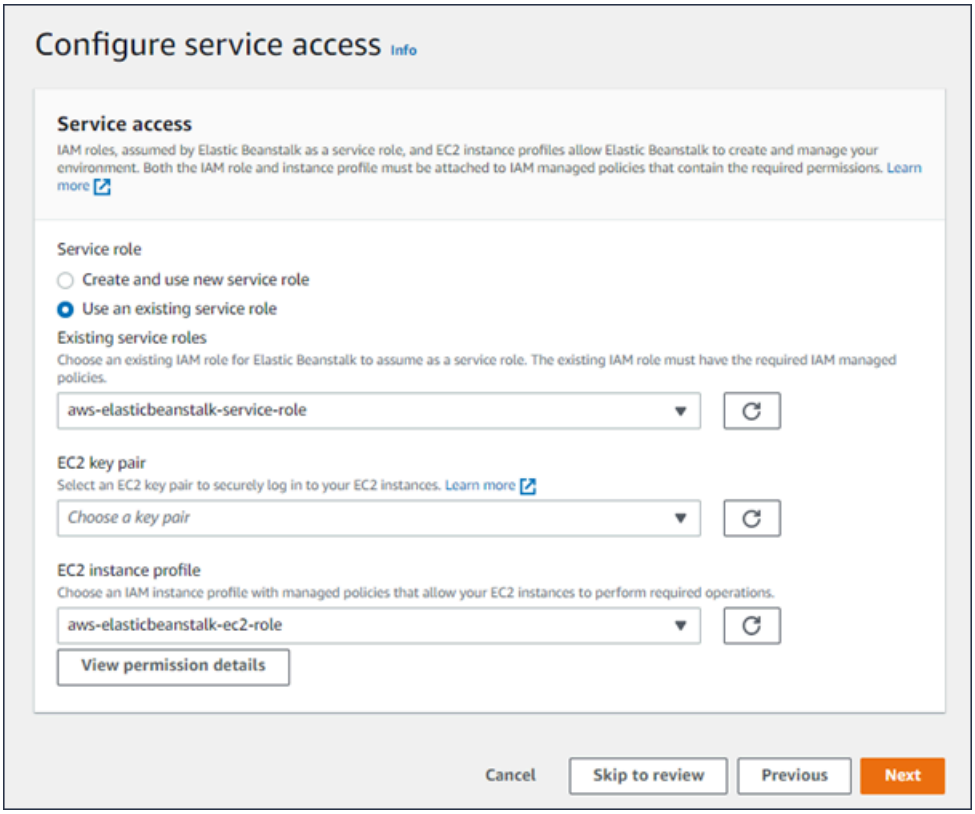
Saat pertama kali membuat lingkungan di konsol Elastic Beanstalk, Anda harus EC2 membuat profil instance dengan seperangkat izin default. Jika daftar dropdown profil EC2 instance tidak menampilkan nilai apa pun untuk dipilih, perluas prosedur berikut. Ini menyediakan langkah-langkah untuk membuat Peran yang selanjutnya dapat Anda pilih untuk profil EC2 instance.

Buat Peran IAM EC2 misalnya profil

Untuk membuat profil EC2 instance

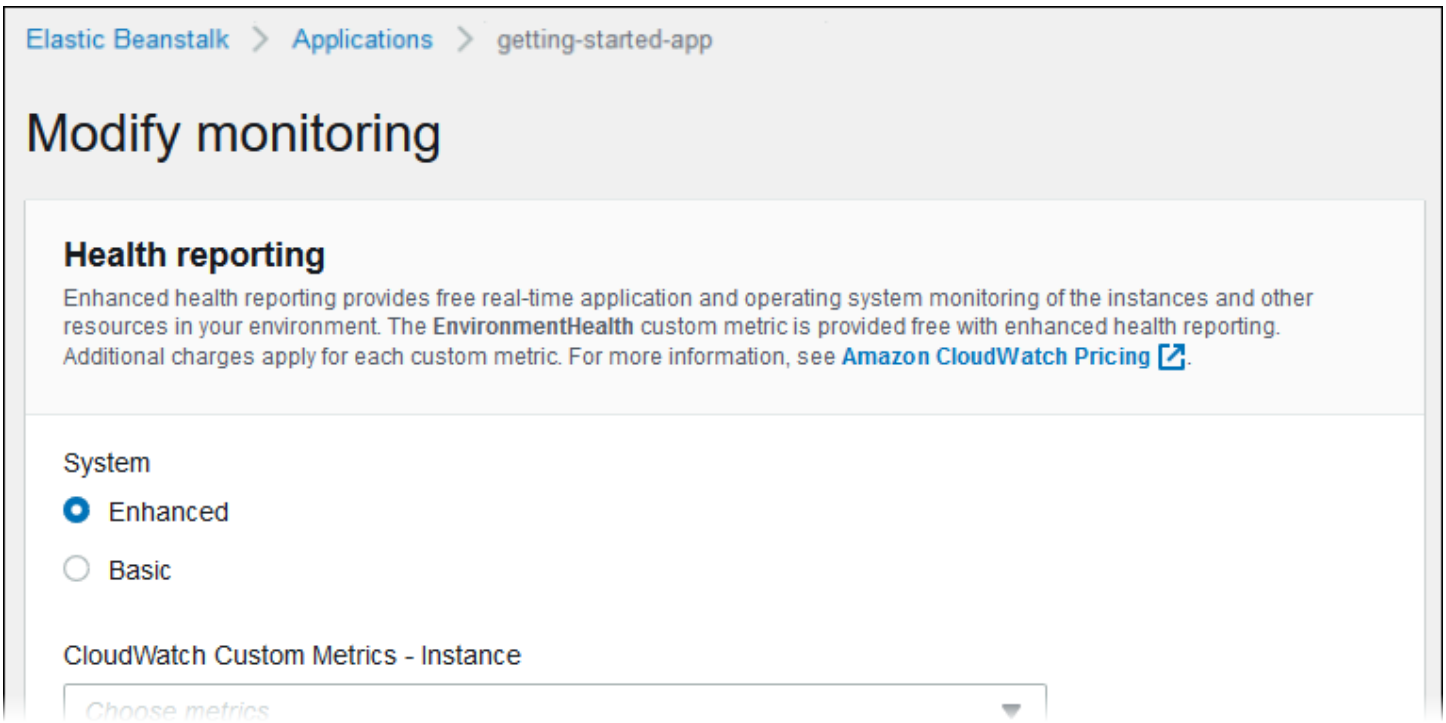
1. Pilih Buat peran.
2. Untuk jenis entitas Tepercaya, pilih AWS layanan.
3. Untuk kasus Penggunaan, pilih Elastic Beanstalk — Compute.
4. Pilih Berikutnya.

5. Verifikasi bahwa kebijakan Izin menyertakan yang berikut ini, lalu pilih Berikutnya:
   - `AWSElasticBeanstalkWebTier`
   - `AWSElasticBeanstalkWorkerTier`
   - `AWSElasticBeanstalkMulticontainerDocker`
6. Pilih Buat peran.
7. Kembali ke tab Konfigurasi akses layanan, segarkan daftar, lalu pilih profil EC2 instance yang baru dibuat.



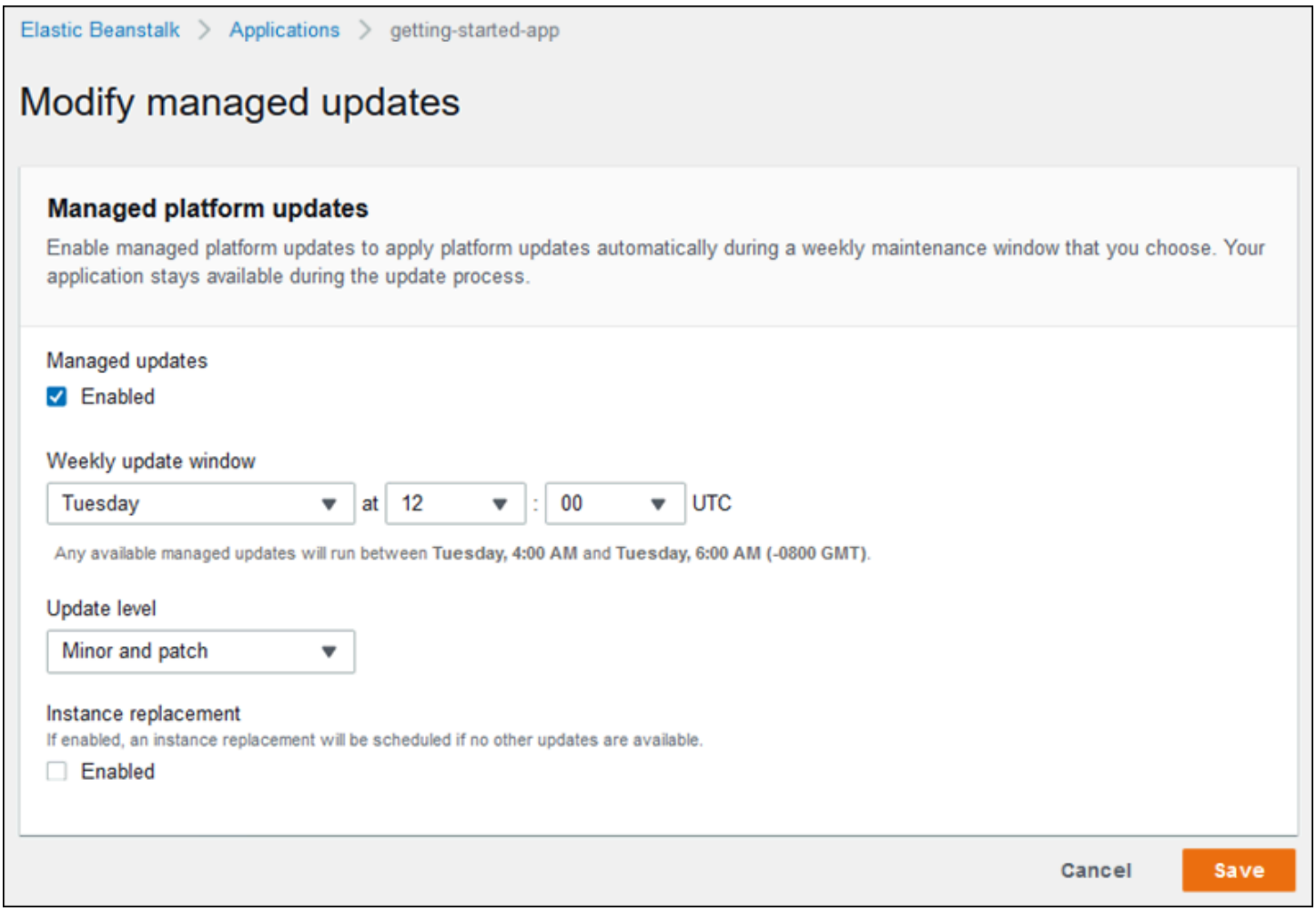
Pemantauan

Gunakan halaman konfigurasi Modifikasi pemantauan untuk mengonfigurasi pelaporan kondisi, aturan pemantauan, dan kondisi alur kejadian. Untuk informasi selengkapnya, lihat the section called “Aktifkan kondisi yang ditingkatkan”, the section called “Aturan kondisi yang ditingkatkan”, and the section called “Kondisi lingkungan streaming”.



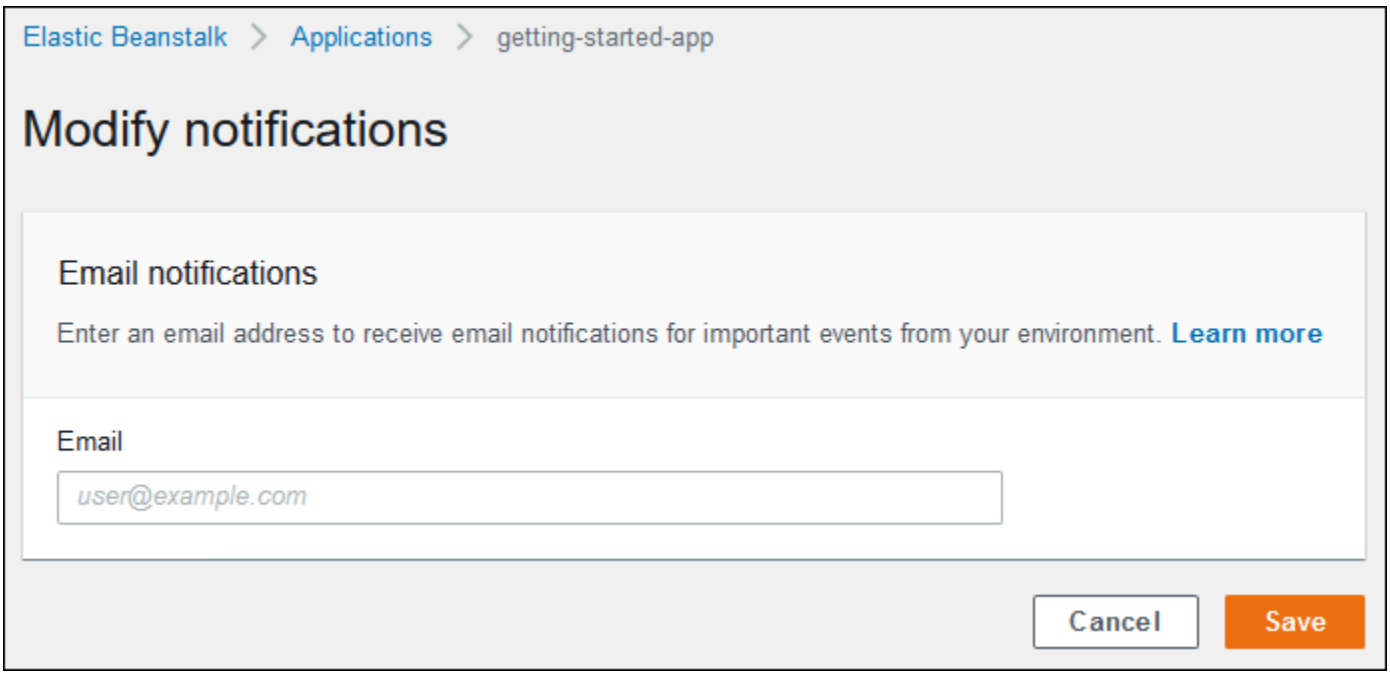
Pembaruan terkelola

Gunakan Modifikasi pembaruan terkelola untuk mengonfigurasi pembaruan platform terkelola. Anda dapat memutuskan apakah Anda ingin mereka diaktifkan, mengatur jadwal, dan mengonfigurasi properti lainnya. Untuk detail selengkapnya, lihat the section called “Pembaruan yang dikelola”.



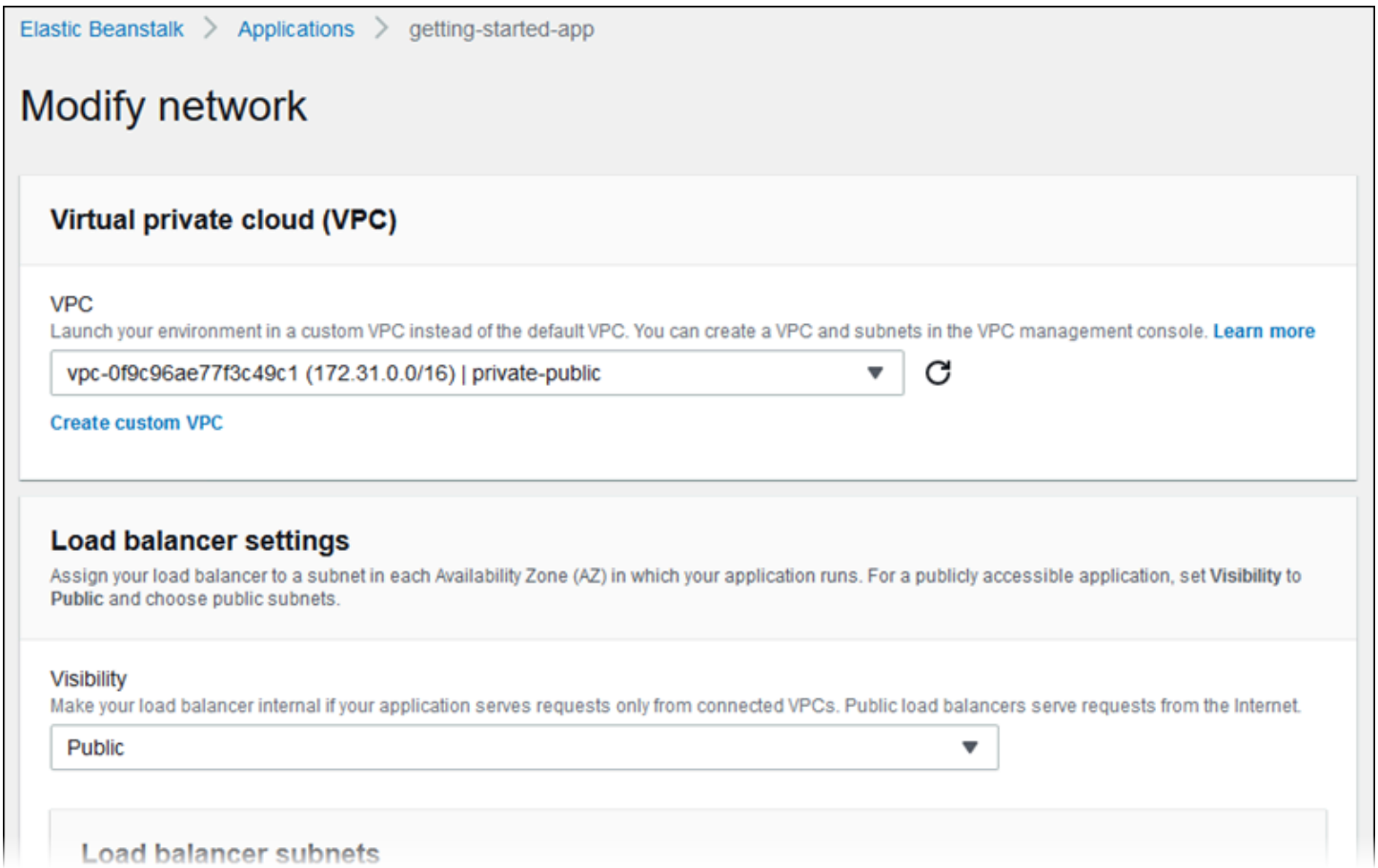
Notifikasi

Gunakan halaman konfigurasi Modifikasi pemberitahuan untuk menentukan alamat email yang akan menerima notifikasi email untuk peristiwa-peristiwa penting dari lingkungan Anda.



Jaringan

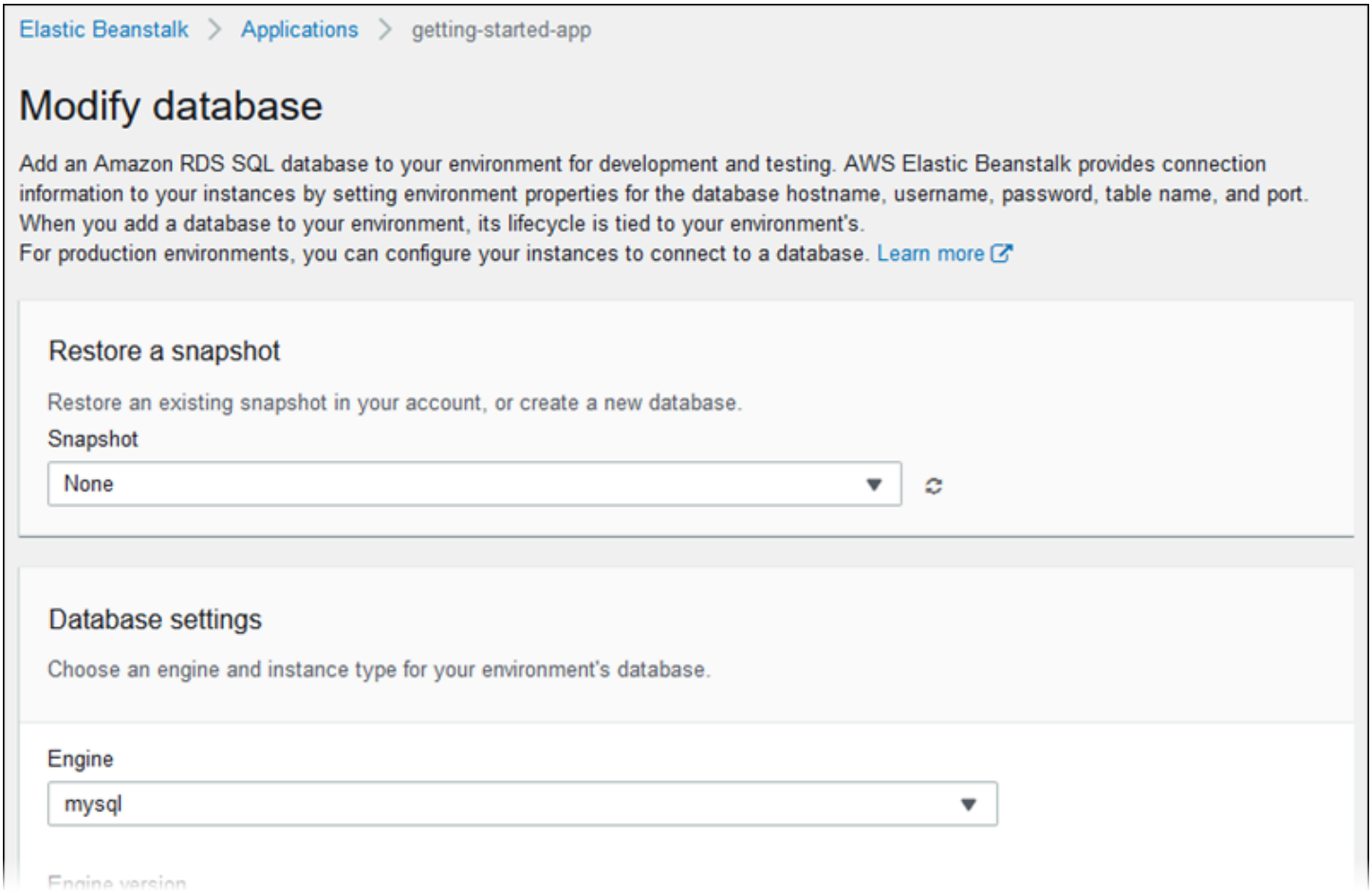
Jika Anda telah membuat VPC kustom, halaman konfigurasi Modifikasi jaringan untuk mengonfigurasi lingkungan Anda untuk menggunakannya. Jika Anda tidak memilih VPC, Elastic Beanstalk menggunakan VPC dan subnet default.



Basis Data

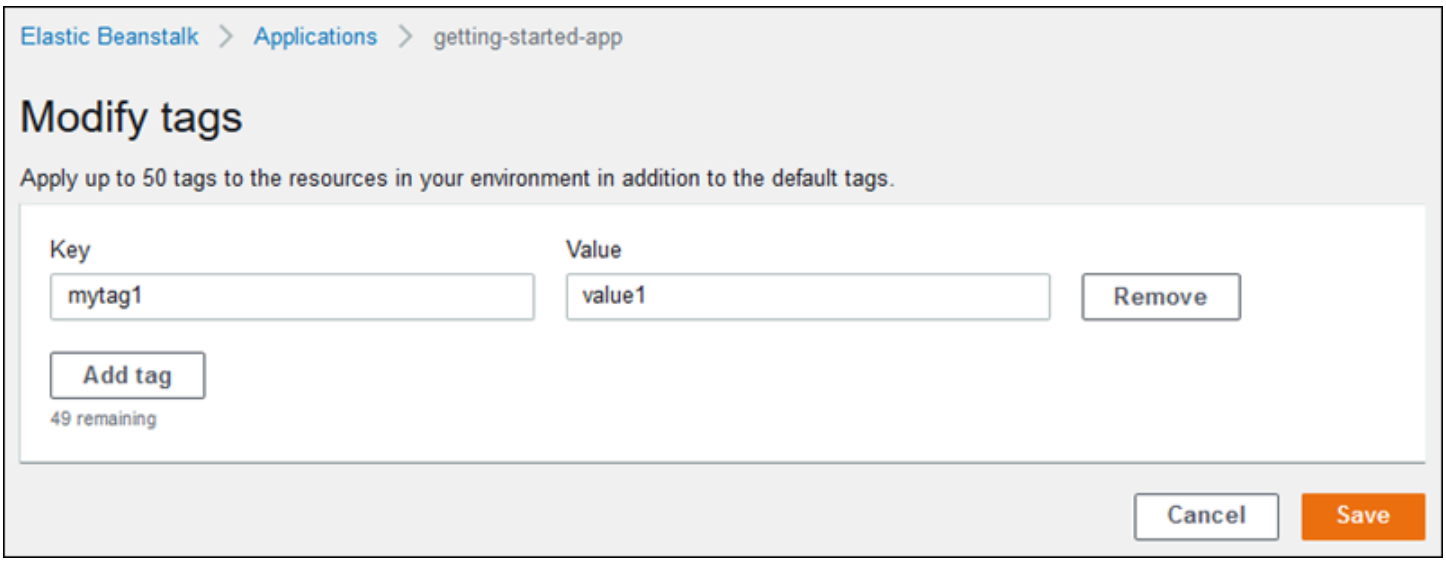
Gunakan halaman konfigurasi Modifikasi basis data untuk menambahkan basis data Amazon Relational Database Service (Amazon RDS) ke lingkungan Anda untuk pengembangan dan pengujian. Elastic Beanstalk memberikan informasi koneksi ke instans Anda dengan mengatur properti lingkungan untuk nama host basis data, nama pengguna, kata sandi, nama tabel, dan port.

Lihat perinciannya di the section called “Basis data”.



Tanda

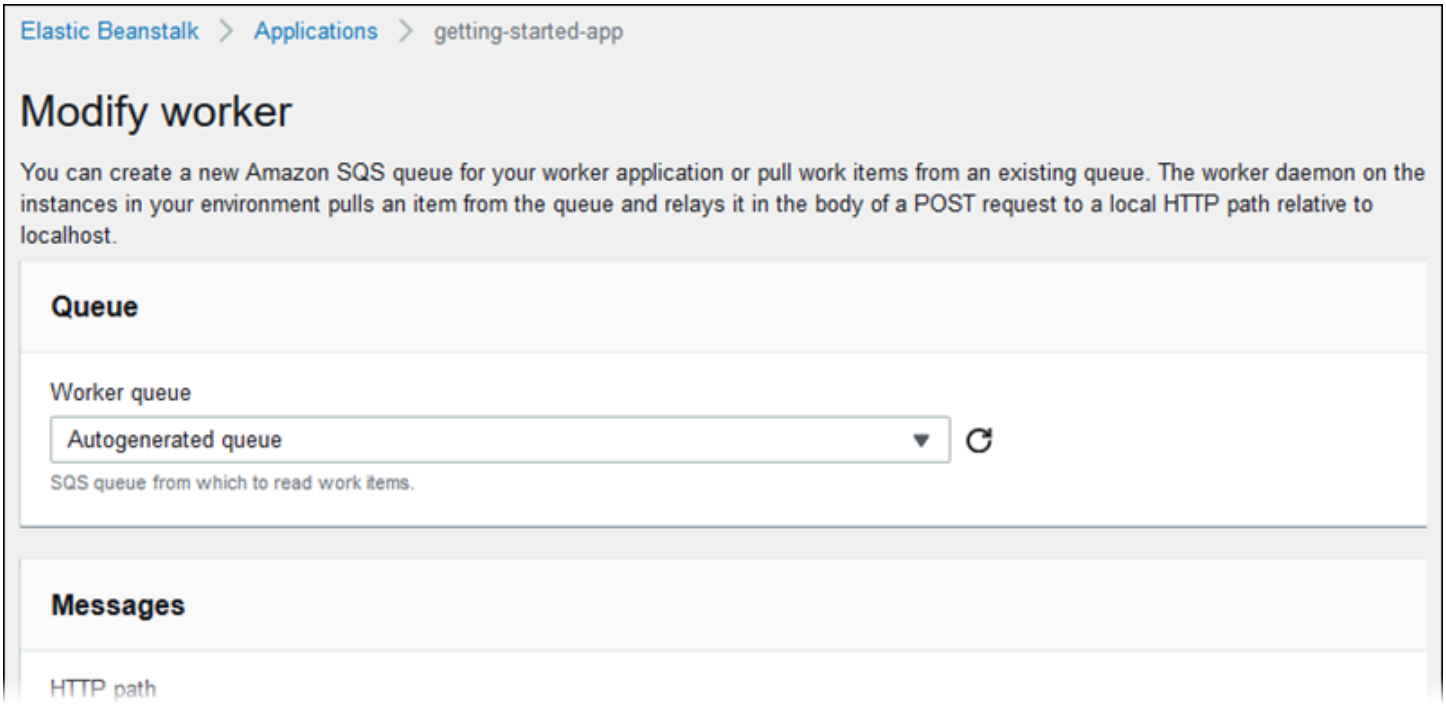
Gunakan halaman konfigurasi Modifikasi tanda untuk menambahkan tanda ke sumber daya di lingkungan Anda. Untuk informasi lebih lanjut tentang pemberian penandaan pada lingkungan, lihat Menandai sumber daya di lingkungan Elastic Beanstalk Anda.



Lingkungan pekerja

Jika Anda membuat lingkungan pekerja, gunakan halaman konfigurasi Modifikasi pekerja untuk mengonfigurasi lingkungan pekerja. Daemon pekerja pada instans di lingkungan Anda menarik item dari antrean Amazon Simple Queue Service (Amazon SQS) dan menyampaikannya sebagai pesan posting ke aplikasi pekerja Anda. Anda dapat memilih antrean Amazon SQS yang dibaca oleh daemon pekerja dari (dibuat otomatis atau sudah ada). Anda juga dapat mengonfigurasi pesan yang daemon pekerja kirim ke aplikasi Anda.

Untuk informasi selengkapnya, lihat the section called “Lingkungan pekerja”.



Mengkloning lingkungan Elastic Beanstalk

Anda dapat menggunakan lingkungan Elastic Beanstalk yang ada sebagai dasar untuk lingkungan baru dengan mengkloning lingkungan yang ada. Misalnya, Anda mungkin ingin membuat klon sehingga Anda dapat menggunakan versi yang lebih baru dari cabang platform yang digunakan oleh platform lingkungan asli. Elastic Beanstalk mengkonfigurasi klon dengan pengaturan lingkungan yang digunakan oleh lingkungan asli. Dengan mengkloning lingkungan yang ada alih-alih membuat lingkungan baru, Anda tidak perlu mengonfigurasi pengaturan opsi, variabel lingkungan, dan pengaturan lain secara manual yang Anda buat dengan layanan Elastic Beanstalk. Elastic Beanstalk juga membuat salinan sumber daya apa pun yang terkait AWS dengan lingkungan asli.

Penting untuk menyadari situasi berikut:

- Selama proses kloning, Elastic Beanstalk tidak menyalin data dari Amazon RDS ke klon.
- Elastic Beanstalk tidak memasukkan perubahan yang tidak terkelola ke sumber daya dalam klon. Perubahan pada sumber daya AWS yang Anda buat menggunakan alat selain konsol Elastic Beanstalk, alat baris perintah, atau API dianggap sebagai perubahan tidak terkelola.
- Grup keamanan untuk masuk dianggap sebagai perubahan yang tidak terkelola. Lingkungan Elastic Beanstalk yang dikloning tidak membawa kelompok keamanan untuk masuk, meninggalkan lingkungan terbuka untuk semua lalu lintas internet. Anda harus membangun kembali grup keamanan ingress untuk lingkungan kloning.

Anda hanya dapat mengkloning lingkungan ke versi platform yang berbeda dari cabang platform yang sama. Cabang platform yang berbeda tidak dijamin kompatibel. Untuk menggunakan cabang platform yang berbeda, Anda harus secara manual membuat lingkungan baru, men-deploy kode aplikasi Anda, dan membuat perubahan yang diperlukan dalam kode dan pilihan untuk memastikan aplikasi Anda bekerja dengan benar pada cabang platform baru.

AWS konsol manajemen

> **Important**
>
> Lingkungan Elastic Beanstalk yang dikloning tidak membawa kelompok keamanan untuk masuk, meninggalkan lingkungan terbuka untuk semua lalu lintas internet. Anda harus membangun kembali grup keamanan ingress untuk lingkungan kloning.

Anda dapat melihat sumber daya yang mungkin tidak dikloning dengan memeriksa status drift konfigurasi lingkungan Anda. Untuk informasi selengkapnya, lihat Mendeteksi drift di seluruh CloudFormation tumpukan di Panduan AWS CloudFormation Pengguna.

Untuk mengkloning lingkungan

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada halaman ikhtisar lingkungan, pilih Tindakan.
4. Pilih lingkungan Klon.
5. Pada halaman Klon lingkungan, tinjau informasi di bagian Lingkungan Asli untuk memverifikasi bahwa Anda telah memilih lingkungan yang ingin Anda buat klon.
6. Di bagian Lingkungan Baru, Anda dapat secara opsional mengubah nila-nilai Nama lingkungan, URL lingkungan, Deskripsi, Versi platform, dan Peran layanan yang secara otomatis ditetapkan oleh Elastic Beanstalk berdasarkan lingkungan asli.

   Note

   Jika versi platform yang digunakan di lingkungan asli bukan yang direkomendasikan untuk digunakan di cabang platform, Anda diperingatkan bahwa versi platform yang berbeda direkomendasikan. Pilih Versi platform, dan Anda dapat melihat versi platform yang direkomendasikan di daftar—misalnya, 3.3.2 (Direkomendasikan).

7. Saat Anda siap, pilih Klon.

Antarmuka baris Elastic Beanstalk (EB CLI)

Important

Lingkungan Elastic Beanstalk yang dikloning tidak membawa kelompok keamanan untuk masuk, meninggalkan lingkungan terbuka untuk semua lalu lintas internet. Anda harus membangun kembali grup keamanan ingress untuk lingkungan kloning.

Anda dapat melihat sumber daya yang mungkin tidak dikloning dengan memeriksa status drift konfigurasi lingkungan Anda. Untuk informasi selengkapnya, lihat Mendeteksi drift di seluruh CloudFormation tumpukan di Panduan AWS CloudFormation Pengguna.

Gunakan perintah eb clone untuk mengkloning lingkungan yang sedang berjalan, sebagai berikut.

```
~/workspace/my-app$ eb clone my-env1
Enter name for Environment Clone
(default is my-env1-clone): my-env2
Enter DNS CNAME prefix
(default is my-env1-clone): my-env2
```

Anda dapat menentukan nama lingkungan sumber dalam perintah klon, atau membiarkannya mengkloning lingkungan default untuk folder proyek saat ini. EB CLI meminta Anda untuk memasukkan nama dan prefiks DNS untuk lingkungan baru.

Secara default, eb clone membuat lingkungan baru dengan versi terbaru yang tersedia dari platform lingkungan sumber. Untuk memaksa EB CLI untuk menggunakan versi yang sama, bahkan jika ada versi yang lebih baru yang tersedia, gunakan pilihan `--exact`.

```
~/workspace/my-app$ eb clone --exact
```

Untuk informasi selengkapnya tentang perintah ini, lihat eb clone.

Mengakhiri lingkungan Elastic Beanstalk

Anda dapat mengakhiri AWS Elastic Beanstalk lingkungan berjalan menggunakan konsol Elastic Beanstalk. Dengan melakukan ini, Anda menghindari biaya untuk sumber daya yang tidak AWS digunakan.

> **Note**
>
> Anda selalu dapat meluncurkan lingkungan baru menggunakan versi yang sama nantinya.

Jika Anda memiliki data dari lingkungan yang ingin Anda pertahankan, setel kebijakan penghapusan database `Retain` sebelum mengakhiri lingkungan. Ini membuat database tetap beroperasi di luar

Elastic Beanstalk. Setelah ini, setiap lingkungan Elastic Beanstalk harus terhubung dengannya sebagai database eksternal. Jika Anda ingin mencadangkan data tanpa menjaga database tetap beroperasi, setel kebijakan penghapusan untuk mengambil snapshot database sebelum mengakhiri lingkungan. Untuk informasi selengkapnya, lihat Siklus hidup basis data di bagian Mengkonfigurasi lingkungan dari panduan ini.

Elastic Beanstalk mungkin gagal mengakhiri lingkungan Anda. Salah satu alasan umum adalah bahwa kelompok keamanan lingkungan lain memiliki ketergantungan pada kelompok keamanan lingkungan yang ingin Anda hentikan. Untuk petunjuk tentang cara menghindari masalah ini, lihat EC2 kelompok keamanan di halaman EC2Instans panduan ini.

> **Important**
>
> Jika Anda mengakhiri lingkungan, Anda juga harus menghapus pemetaan CNAME apa pun yang Anda buat, karena pelanggan lain dapat menggunakan kembali nama host yang tersedia. Pastikan untuk menghapus catatan DNS yang mengarah ke lingkungan Anda yang dihentikan untuk mencegah entri DNS yang menggantung. Entri DNS yang menggantung dapat mengekspos lalu lintas internet yang ditujukan untuk domain Anda ke kerentanan keamanan. Itu juga dapat menimbulkan risiko lain.
> Untuk informasi selengkapnya, lihat Perlindungan dari catatan delegasi yang menggantung di Route 53 di Panduan Pengembang Amazon Route 53. Anda juga dapat mempelajari selengkapnya tentang menggantung entri DNS di Perlindungan Domain yang Ditingkatkan untuk Permintaan CloudFront Amazon di AWS Blog Keamanan.

Konsol Elastic Beanstalk

Untuk mengakhiri lingkungan

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pilih Tindakan, lalu pilih Terminate environment.
4. Gunakan kotak dialog di layar untuk mengonfirmasi pengakhiran lingkungan.

> **Note**
>
> Ketika Anda mengakhiri lingkungan Anda, CNAME yang terkait dengan lingkungan yang dihentikan dibebaskan untuk digunakan oleh siapa pun.

Dibutuhkan beberapa menit bagi Elastic Beanstalk untuk AWS menghentikan sumber daya yang berjalan di lingkungan.

AWS CLI

Untuk mengakhiri lingkungan

- Jalankan perintah berikut.

```
$ aws elasticbeanstalk terminate-environment --environment-name my-env
```

API

Untuk mengakhiri lingkungan

- Hubungi `TerminateEnvironment` dengan parameter berikut ini:

  `EnvironmentName` = `SampleAppEnv`

```
https://elasticbeanstalk.us-west-2.amazon.com/?EnvironmentName=SampleAppEnv
&Operation=TerminateEnvironment
&AuthParams
```

Buat lingkungan Elastic Beanstalk dengan AWS CLI

Untuk detail tentang AWS CLI perintah untuk Elastic Beanstalk, lihat Referensi Perintah.AWS CLI

1. Periksa apakah CNAME untuk lingkungan tersedia.

```
$ aws elasticbeanstalk check-dns-availability --cname-prefix my-cname
{
```

```
    "Available": true,
    "FullyQualifiedCNAME": "my-cname.elasticbeanstalk.com"
}
```

2. Pastikan versi aplikasi Anda ada.

```
$ aws elasticbeanstalk describe-application-versions --application-name my-app --
version-label v1
```

Jika Anda belum memiliki versi aplikasi untuk sumber Anda, buatlah. Misalnya, perintah berikut membuat versi aplikasi dari paket sumber Amazon Simple Storage Service (Amazon S3).

```
$ aws elasticbeanstalk create-application-version --application-name my-app --
version-label v1 --source-bundle S3Bucket=amzn-s3-demo-bucket,S3Key=my-source-
bundle.zip
```

3. Buat templat konfigurasi untuk aplikasi.

```
$ aws elasticbeanstalk create-configuration-template --application-name my-app --
template-name v1 --solution-stack-name "64bit Amazon Linux 2015.03 v2.0.0 running
 Ruby 2.2 (Passenger Standalone)"
```

4. Buat lingkungan.

```
$ aws elasticbeanstalk create-environment --cname-prefix my-cname --application-
name my-app --template-name v1 --version-label v1 --environment-name v1clone --
option-settings file://options.txt
```

Pengaturan Pilihan didefinisikan dalam file options.txt:

```
[
    {
        "Namespace": "aws:autoscaling:launchconfiguration",
        "OptionName": "IamInstanceProfile",
        "Value": "aws-elasticbeanstalk-ec2-role"
    }
]
```

Pengaturan pilihan di atas mendefinisikan profil instans IAM. Anda dapat menentukan ARN atau nama profil.

5. Menentukan apakah lingkungan baru Hijau dan Siap.

```
$ aws elasticbeanstalk describe-environments --environment-names my-env
```

Jika lingkungan baru tidak muncul sebagai Hijau dan Siap, Anda harus memutuskan apakah Anda ingin mencoba lagi operasi atau meninggalkan lingkungan dalam keadaan saat ini untuk diselidiki. Pastikan untuk mengakhiri lingkungan setelah Anda selesai, dan bersihkan sumber daya yang tidak digunakan.

> **Note**
>
> Anda dapat menyesuaikan periode waktu habis jika lingkungan tidak diluncurkan dalam waktu yang wajar.

Buat lingkungan Elastic Beanstalk dengan API

1. Hubungi `CheckDNSAvailability` dengan parameter berikut ini:

   - `CNAMEPrefix` = `SampleApp`

   Example

```
https://elasticbeanstalk.us-east-2.amazonaws.com/?CNAMEPrefix=sampleapplication
&Operation=CheckDNSAvailability
&AuthParams
```

2. Hubungi `DescribeApplicationVersions` dengan parameter berikut ini:

   - `ApplicationName` = `SampleApp`
   - `VersionLabel` = `Version2`

   Example

```
https://elasticbeanstalk.us-east-2.amazonaws.com/?ApplicationName=SampleApp
&VersionLabel=Version2
&Operation=DescribeApplicationVersions
&AuthParams
```

3. Hubungi `CreateConfigurationTemplate` dengan parameter berikut ini:

   - `ApplicationName` = `SampleApp`
   - `TemplateName` = `MyConfigTemplate`
   - `SolutionStackName` = `64bit%20Amazon%20Linux%202015.03%20v2.0.0%20running%20Ruby%202.2%20(Passenger%20Standalone)`

   Example

   ```
   https://elasticbeanstalk.us-east-2.amazonaws.com/?ApplicationName=SampleApp
   &TemplateName=MyConfigTemplate
   &Operation=CreateConfigurationTemplate
   &SolutionStackName=64bit%20Amazon%20Linux%202015.03%20v2.0.0%20running%20Ruby
   %202.2%20(Passenger%20Standalone)
   &AuthParams
   ```

4. Hubungi `CreateEnvironment` dengan salah satu set parameter berikut ini.

   a. Gunakan berikut ini untuk tingkat lingkungan server web:

      - `EnvironmentName` = `SampleAppEnv2`
      - `VersionLabel` = `Version2`
      - `Description` = `description`
      - `TemplateName` = `MyConfigTemplate`
      - `ApplicationName` = `SampleApp`
      - `CNAMEPrefix` = `sampleapplication`
      - `OptionSettings.member.1.Namespace` = `aws:autoscaling:launchconfiguration`
      - `OptionSettings.member.1.OptionName` = `IamInstanceProfile`
      - `OptionSettings.member.1.Value` = `aws-elasticbeanstalk-ec2-role`

      Example

      ```
      https://elasticbeanstalk.us-east-2.amazonaws.com/?ApplicationName=SampleApp
      &VersionLabel=Version2
      &EnvironmentName=SampleAppEnv2
      &TemplateName=MyConfigTemplate
      ```

```
&CNAMEPrefix=sampleapplication
&Description=description
&Operation=CreateEnvironment
&OptionSettings.member.1.Namespace=aws%3Aautoscaling%3Alaunchconfiguration
&OptionSettings.member.1.OptionName=IamInstanceProfile
&OptionSettings.member.1.Value=aws-elasticbeanstalk-ec2-role
&AuthParams
```

b. Gunakan berikut ini untuk tingkat lingkungan pekerja:

- `EnvironmentName` = `SampleAppEnv2`
- `VersionLabel` = `Version2`
- `Description` = `description`
- `TemplateName` = `MyConfigTemplate`
- `ApplicationName` = `SampleApp`
- `Tier` = `Worker`
- `OptionSettings.member.1.Namespace` = `aws:autoscaling:launchconfiguration`
- `OptionSettings.member.1.OptionName` = `IamInstanceProfile`
- `OptionSettings.member.1.Value` = `aws-elasticbeanstalk-ec2-role`
- `OptionSettings.member.2.Namespace` = `aws:elasticbeanstalk:sqsd`
- `OptionSettings.member.2.OptionName` = `WorkerQueueURL`
- `OptionSettings.member.2.Value` = `sqsd.elasticbeanstalk.us-east-2.amazonaws.com`
- `OptionSettings.member.3.Namespace` = `aws:elasticbeanstalk:sqsd`
- `OptionSettings.member.3.OptionName` = `HttpPath`
- `OptionSettings.member.3.Value` = `/`
- `OptionSettings.member.4.Namespace` = `aws:elasticbeanstalk:sqsd`
- `OptionSettings.member.4.OptionName` = `MimeType`
- `OptionSettings.member.4.Value` = `application/json`
- `OptionSettings.member.5.Namespace` = `aws:elasticbeanstalk:sqsd`
- `OptionSettings.member.5.OptionName` = `HttpConnections`
- `OptionSettings.member.5.Value` = `75`
- `OptionSettings.member.6.Namespace` = `aws:elasticbeanstalk:sqsd`

- OptionSettings.member.6.OptionName = ConnectTimeout
- OptionSettings.member.6.Value = 10
- OptionSettings.member.7.Namespace = aws:elasticbeanstalk:sqsd
- OptionSettings.member.7.OptionName = InactivityTimeout
- OptionSettings.member.7.Value = 10
- OptionSettings.member.8.Namespace = aws:elasticbeanstalk:sqsd
- OptionSettings.member.8.OptionName = VisibilityTimeout
- OptionSettings.member.8.Value = 60
- OptionSettings.member.9.Namespace = aws:elasticbeanstalk:sqsd
- OptionSettings.member.9.OptionName = RetentionPeriod
- OptionSettings.member.9.Value = 345600

Example

```
https://elasticbeanstalk.us-east-2.amazonaws.com/?ApplicationName=SampleApp
&VersionLabel=Version2
&EnvironmentName=SampleAppEnv2
&TemplateName=MyConfigTemplate
&Description=description
&Tier=Worker
&Operation=CreateEnvironment
&OptionSettings.member.1.Namespace=aws%3Aautoscaling%3Alaunchconfiguration
&OptionSettings.member.1.OptionName=IamInstanceProfile
&OptionSettings.member.1.Value=aws-elasticbeanstalk-ec2-role
&OptionSettings.member.2.Namespace=aws%3Aelasticbeanstalk%3Asqsd
&OptionSettings.member.2.OptionName=WorkerQueueURL
&OptionSettings.member.2.Value=sqsd.elasticbeanstalk.us-east-2.amazonaws.com
&OptionSettings.member.3.Namespace=aws%3elasticbeanstalk%3sqsd
&OptionSettings.member.3.OptionName=HttpPath
&OptionSettings.member.3.Value=%2F
&OptionSettings.member.4.Namespace=aws%3Aelasticbeanstalk%3Asqsd
&OptionSettings.member.4.OptionName=MimeType
&OptionSettings.member.4.Value=application%2Fjson
&OptionSettings.member.5.Namespace=aws%3Aelasticbeanstalk%3Asqsd
&OptionSettings.member.5.OptionName=HttpConnections
&OptionSettings.member.5.Value=75
&OptionSettings.member.6.Namespace=aws%3Aelasticbeanstalk%3Asqsd
&OptionSettings.member.6.OptionName=ConnectTimeout
```

```
&OptionSettings.member.6.Value=10
&OptionSettings.member.7.Namespace=aws%3Aelasticbeanstalk%3Asqsd
&OptionSettings.member.7.OptionName=InactivityTimeout
&OptionSettings.member.7.Value=10
&OptionSettings.member.8.Namespace=aws%3Aelasticbeanstalk%3Asqsd
&OptionSettings.member.8.OptionName=VisibilityTimeout
&OptionSettings.member.8.Value=60
&OptionSettings.member.9.Namespace=aws%3Aelasticbeanstalk%3Asqsd
&OptionSettings.member.9.OptionName=RetentionPeriod
&OptionSettings.member.9.Value=345600
&AuthParams
```

Membangun Meluncurkan URL Sekarang

Anda dapat membuat URL khusus sehingga siapa pun dapat dengan cepat menyebarkan dan menjalankan aplikasi web yang telah ditentukan sebelumnya. AWS Elastic Beanstalk URL ini disebut Meluncurkan URL Sekarang. Anda mungkin memerlukan URL Launch Now, misalnya, untuk mendemonstrasikan aplikasi web yang dibuat untuk berjalan di Elastic Beanstalk. Dengan Meluncurkan URL Sekarang, Anda dapat menggunakan parameter untuk menambahkan informasi yang diperlukan ke wizard Buat Aplikasi terlebih dahulu. Setelah Anda menambahkan informasi ini ke wizard, siapa pun dapat menggunakan tautan URL untuk meluncurkan lingkungan Elastic Beanstalk dengan sumber aplikasi web Anda hanya dalam beberapa langkah. Ini berarti pengguna tidak perlu mengunggah atau menentukan lokasi bundel sumber aplikasi secara manual. Mereka juga tidak perlu memberikan informasi tambahan apa pun kepada wizard.

URL Launch Now memberi Elastic Beanstalk informasi minimum yang diperlukan untuk membuat aplikasi: nama aplikasi, tumpukan solusi, jenis instance, dan jenis lingkungan. Elastic Beanstalk menggunakan nilai default untuk detail konfigurasi lain yang tidak secara eksplisit ditentukan dalam URL Launch Now kustom Anda.

Meluncurkan URL Sekarang menggunakan sintaks URL standar. Untuk informasi selengkapnya, lihat RFC 3986 - Uniform Resource Identifier (URI): Generic Syntax.

Parameter URL

URL harus berisi parameter berikut, yang peka huruf besar/kecil:

- wilayah — Tentukan AWS Wilayah. Untuk daftar Wilayah yang didukung oleh Elastic Beanstalk AWS Elastic Beanstalk , lihat Titik Akhir dan Kuota di. Referensi Umum AWS

- applicationName – Tentukan nama aplikasi Anda. Elastic Beanstalk menampilkan nama aplikasi di konsol Elastic Beanstalk untuk membedakannya dari aplikasi lain. Secara default, nama aplikasi juga membentuk dasar dari nama lingkungan dan URL lingkungan.
- Platform – Tentukan versi platform yang akan digunakan untuk lingkungan. Gunakan salah satu metode berikut, lalu berikan kode-URL pada pilihan Anda:
  - Tentukan ARN platform tanpa versi. Elastic Beanstalk memilih versi platform terbaru dari versi utama platform yang sesuai. Misalnya, untuk memilih versi platform Python 3.6 terbaru, tentukan. `Python 3.6 running on 64bit Amazon Linux`
  - Tentukan nama platform. Elastic Beanstalk memilih versi terbaru dari runtime bahasa terbaru platform (misalnya,). `Python`

  Untuk deskripsi mengenai semua platform yang tersedia dan versi mereka, lihat Platform yang didukung Elastic Beanstalk.

  Anda dapat menggunakan AWS Command Line Interface(AWS CLI) untuk mendapatkan daftar semua versi platform yang tersedia dengan masing-masing ARNs. `list-platform-versions`Perintah ini mencantumkan informasi terperinci tentang semua versi platform yang tersedia. Gunakan `--filters` argumen untuk cakupan daftar. Misalnya, Anda dapat membuat cakupan daftar untuk hanya menampilkan versi platform dari bahasa tertentu.

  Contoh berikut menanyakan semua versi platform Python, dan menyalurkan output melalui serangkaian perintah. Hasilnya adalah daftar versi platform ARNs (tanpa */version* ekor), dalam format yang dapat dibaca manusia, tanpa pengkodean URL.

```
$ aws elasticbeanstalk list-platform-versions --filters
 'Type="PlatformName",Operator="contains",Values="Python"' | grep PlatformArn | awk -
F '"' '{print $4}' | awk -F '/' '{print $2}'
Preconfigured Docker - Python 3.4 running on 64bit Debian
Preconfigured Docker - Python 3.4 running on 64bit Debian
Python 2.6 running on 32bit Amazon Linux
Python 2.6 running on 32bit Amazon Linux 2014.03
...
Python 3.6 running on 64bit Amazon Linux
```

  Contoh berikut menambahkan perintah Perl untuk contoh terakhir untuk URL-encode output.

```
$ aws elasticbeanstalk list-platform-versions --filters
 'Type="PlatformName",Operator="contains",Values="Python"' | grep PlatformArn | awk
```

```
 -F '"' '{print $4}' | awk -F '/' '{print $2}' | perl -MURI::Escape -ne 'chomp;print
 uri_escape($_),"\n"'
Preconfigured%20Docker%20-%20Python%203.4%20running%20on%2064bit%20Debian
Preconfigured%20Docker%20-%20Python%203.4%20running%20on%2064bit%20Debian
Python%202.6%20running%20on%2032bit%20Amazon%20Linux
Python%202.6%20running%20on%2032bit%20Amazon%20Linux%202014.03
...
Python%203.6%20running%20on%2064bit%20Amazon%20Linux
```

Meluncurkan URL Sekarang secara opsional dapat berisi parameter berikut. Jika Anda tidak menyertakan parameter pilihan dalam Meluncurkan URL Sekarang Anda, Elastic Beanstalk menggunakan nilai default untuk membuat dan menjalankan aplikasi Anda. Bila Anda tidak menyertakan sourceBundleUrlparameter, Elastic Beanstalk menggunakan aplikasi sampel default untuk platform yang ditentukan.

- sourceBundleUrl— Tentukan lokasi bundel sumber aplikasi web Anda dalam format URL. Misalnya, jika Anda mengunggah bundel sumber ke bucket Amazon S3, Anda dapat menentukan nilai parameter sebagai sourceBundleUrl. `https://amzn-s3-demo-bucket.s3.amazonaws.com/myobject`

  > **Note**
  >
  > Anda dapat menentukan nilai sourceBundleUrlparameter sebagai URL HTTP, tetapi browser web pengguna akan mengonversi karakter sesuai kebutuhan dengan menerapkan pengkodean URL HTML.

- environmentType – Tentukan apakah lingkungan memiliki beban yang seimbang dan dapat diskalakan atau hanya instans tunggal. Untuk informasi selengkapnya, lihat Jenis lingkungan. Anda dapat menentukan salah satu `LoadBalancing` atau `SingleInstance` sebagai nilai untuk parameter.
- tierName – Tentukan apakah lingkungan mendukung aplikasi web yang memproses permintaan web atau aplikasi web yang menjalankan pekerjaan latar belakang. Untuk informasi selengkapnya, lihat Lingkungan pekerja Elastic Beanstalk. Anda dapat menentukan baik `WebServer` atau `Worker`,
- instanceType – Tentukan server dengan karakteristik (termasuk ukuran memori dan daya CPU) yang paling sesuai untuk aplikasi Anda. Untuk informasi selengkapnya tentang keluarga dan jenis EC2 instans Amazon, lihat Jenis instans di Panduan EC2 Pengguna Amazon. Untuk informasi

selengkapnya tentang jenis instans yang tersedia di seluruh Wilayah, lihat Jenis instans yang tersedia di Panduan EC2 Pengguna Amazon.

- withVpc – Tentukan apakah akan membuat lingkungan dalam Amazon VPC. Anda dapat menentukan baik `true` atau `false`. Untuk informasi selengkapnya tentang penggunaan Elastic Beanstalk dengan Amazon VPC, lihat Menggunakan Elastic Beanstalk dengan Amazon VPC.
- withRds – Tentukan apakah akan membuat instans basis data Amazon RDS dengan lingkungan ini. Untuk informasi selengkapnya, lihat Menggunakan Elastic Beanstalk dengan Amazon RDS. Anda dapat menentukan baik `true` atau `false`.
- rds DBEngine - Tentukan mesin database yang ingin Anda gunakan untuk EC2 instans Amazon Anda di lingkungan ini. Anda dapat menentukan `mysql`, `oracle-sel`, `sqlserver-ex`, `sqlserver-web`, atau `sqlserver-se`. Nilai default-nya adalah `mysql`.
- rds DBAllocated Storage - Tentukan ukuran penyimpanan database yang dialokasikan dalam gigabyte (GB). Anda dapat menentukan salah satu nilai berikut:
  - MySQL – `5` ke `1024`. Default-nya adalah `5`.
  - Oracle – `10` ke `1024`. Default-nya adalah `10`.
  - Microsoft SQL Server Express Edition – `30`.
  - Microsoft SQL Server Web Edition – `30`.
  - Microsoft SQL Server Standard Edition – `200`.
- rds DBInstance Class - Tentukan jenis instance database. Nilai `db.m1.large` defaultnya adalah `db.t2.micro` (untuk lingkungan yang tidak berjalan di VPC Amazon). Untuk daftar kelas instance database yang didukung oleh Amazon RDS, lihat Kelas Instance DB di Panduan Pengguna Layanan Amazon Relational Database Service.
- RDSMulti AZDatabase — Tentukan apakah Elastic Beanstalk perlu membuat instance database di beberapa Availability Zone. Anda dapat menentukan `true` atau `false`. Untuk informasi selengkapnya tentang beberapa penerapan Availability Zone dengan Amazon RDS, lihat Wilayah dan Availability Zone di Panduan Pengguna Layanan Amazon Relational Database Service.
- DBDeletionKebijakan rds - Tentukan apakah akan menghapus atau memotret instance database pada penghentian lingkungan. Anda dapat menentukan `Delete` atau `Snapshot`.

Contoh

Berikut ini adalah contoh Meluncurkan URL Sekarang. Setelah Anda membangun sendiri, Anda dapat memberikannya kepada pengguna Anda. Misalnya, Anda dapat menyematkan URL di halaman

web atau dalam materi pelatihan. Ketika pengguna membuat aplikasi menggunakan Meluncurkan URL Sekarang, wizard Elastic Beanstalk Buat Aplikasi tidak memerlukan masukan tambahan.

```
https://console.aws.amazon.com/elasticbeanstalk/home?region=us-west-2#/newApplication?
applicationName=YourCompanySampleApp
      &platform=PHP%207.3%20running%20on%2064bit%20Amazon%20Linux&sourceBundleUrl=
        http://s3.amazonaws.com/amzn-s3-demo-bucket/
myobject&environmentType=SingleInstance&tierName=WebServer
      &instanceType=m1.small&withVpc=true&withRds=true&rdsDBEngine=

 postgres&rdsDBAllocatedStorage=6&rdsDBInstanceClass=db.m1.small&rdsMultiAZDatabase=
          true&rdsDBDeletionPolicy=Snapshot
```

Untuk menggunakan Meluncurkan URL Sekarang

1. Pilih Meluncurkan URL Sekarang.
2. Setelah konsol Elastic Beanstalk terbuka, pada halaman Buat aplikasi web, pilih Tinjau dan luncurkan untuk melihat pengaturan yang digunakan Elastic Beanstalk untuk membuat aplikasi dan meluncurkan lingkungan tempat aplikasi berjalan.
3. Pada halaman Konfigurasi, pilih Buat aplikasi untuk membuat aplikasi.

Membuat dan memperbarui grup lingkungan Elastic Beanstalk

Dengan AWS Elastic Beanstalk ComposeEnvironmentsAPI, Anda dapat membuat dan memperbarui grup lingkungan Elastic Beanstalk dalam satu aplikasi. Setiap lingkungan dalam grup dapat menjalankan komponen terpisah dari aplikasi arsitektur berorientasi layanan. API `Compose Environments` mengambil daftar versi aplikasi dan nama grup pilihan. Elastic Beanstalk membuat lingkungan untuk setiap versi aplikasi, atau, jika lingkungan sudah ada, men-deploy versi aplikasi kepada mereka.

Buat tautan antara lingkungan Elastic Beanstalk untuk menunjuk satu lingkungan sebagai ketergantungan dari yang lain. Ketika Anda membuat grup lingkungan dengan API `Compose Environments`, Elastic Beanstalk membuat lingkungan dependen hanya setelah dependensi mereka berfungsi. Untuk informasi lebih lanjut terkait tautan lingkungan, lihat Membuat tautan antara lingkungan Elastic Beanstalk.

API `Compose Environments` menggunakan manifes lingkungan untuk menyimpan detail konfigurasi yang dibagi oleh grup lingkungan. Setiap aplikasi komponen harus memiliki file

konfigurasi `env.yaml` dalam paket sumber aplikasi yang menentukan parameter yang digunakan untuk membuat lingkungannya.

`Compose Environments` membutuhkan `EnvironmentName` dan `SolutionStack` yang akan ditentukan dalam manifes lingkungan untuk setiap aplikasi komponen.

Anda dapat menggunakan `Compose Environments` API dengan antarmuka baris perintah Elastic Beanstalk (EB CLI), the, atau SDK. AWS CLI Lihat Mengelola beberapa lingkungan Elastic Beanstalk sebagai grup dengan EB CLI untuk instruksi EB CLI.

Menggunakan API **`Compose Environments`**

Misalnya, Anda dapat membuat aplikasi bernama `Media Library` yang memungkinkan pengguna mengunggah dan mengelola citra dan video yang tersimpan di Amazon Simple Storage Service (Amazon S3). Aplikasi ini memiliki lingkungan front-end, `front`, yang menjalankan aplikasi web yang memungkinkan pengguna mengunggah dan mengunduh masing-masing file, melihat pustaka mereka, dan memulai tugas pemrosesan batch.

Alih-alih memproses tugas secara langsung, aplikasi front-end menambahkan pekerjaan untuk antrean Amazon SQS. Lingkungan kedua, `worker`, menarik tugas dari antrean dan memprosesnya. `worker` menggunakan tipe instans G2 yang memiliki GPU berperfoma tinggi, sementara `front` dapat berjalan pada jenis instans generik yang lebih hemat biaya.

Anda akan mengatur folder proyek, `Media Library`, ke dalam direktori terpisah untuk setiap komponen, dengan setiap direktori yang berisi file definisi lingkungan (`env.yaml`) dengan kode sumber untuk masing-masing:

```
~/workspace/media-library
|-- front
|   `-- env.yaml
`-- worker
    `-- env.yaml
```

Daftar berikut menunjukkan file `env.yaml` untuk setiap aplikasi komponen.

`~/workspace/media-library/front/env.yaml`

```
EnvironmentName: front+
EnvironmentLinks:
  "WORKERQUEUE" : "worker+"
```

```
AWSConfigurationTemplateVersion: 1.1.0.0
EnvironmentTier:
  Name: WebServer
  Type: Standard
SolutionStack: 64bit Amazon Linux 2015.09 v2.0.4 running Java 8
OptionSettings:
  aws:autoscaling:launchconfiguration:
    InstanceType: m4.large
```

~/workspace/media-library/worker/env.yaml

```
EnvironmentName: worker+
AWSConfigurationTemplateVersion: 1.1.0.0
EnvironmentTier:
  Name: Worker
  Type: SQS/HTTP
SolutionStack: 64bit Amazon Linux 2015.09 v2.0.4 running Java 8
OptionSettings:
  aws:autoscaling:launchconfiguration:
    InstanceType: g2.2xlarge
```

Setelah membuat versi aplikasi untuk komponen aplikasi front-end (`front-v1`) dan pekerja (`worker-v1`), Anda memanggil API `Compose Environments` dengan nama versi. Dalam contoh ini, kami menggunakan AWS CLI untuk memanggil API.

```
# Create application versions for each component:
~$ aws elasticbeanstalk create-application-version --application-name media-
library --version-label front-v1 --process --source-bundle S3Bucket="amzn-s3-demo-
bucket",S3Key="front-v1.zip"
  {
    "ApplicationVersion": {
        "ApplicationName": "media-library",
        "VersionLabel": "front-v1",
        "Description": "",
        "DateCreated": "2015-11-03T23:01:25.412Z",
        "DateUpdated": "2015-11-03T23:01:25.412Z",
        "SourceBundle": {
            "S3Bucket": "amzn-s3-demo-bucket",
            "S3Key": "front-v1.zip"
        }
    }
  }
```

```
~$ aws elasticbeanstalk create-application-version --application-name media-library
 --version-label worker-v1 --process --source-bundle S3Bucket="amzn-s3-demo-
bucket",S3Key="worker-v1.zip"
  {
    "ApplicationVersion": {
        "ApplicationName": "media-library",
        "VersionLabel": "worker-v1",
        "Description": "",
        "DateCreated": "2015-11-03T23:01:48.151Z",
        "DateUpdated": "2015-11-03T23:01:48.151Z",
        "SourceBundle": {
            "S3Bucket": "amzn-s3-demo-bucket",
            "S3Key": "worker-v1.zip"
        }
    }
  }
# Create environments:
~$ aws elasticbeanstalk compose-environments --application-name media-library --group-
name dev --version-labels front-v1 worker-v1
```

Panggilan ketiga membuat dua lingkungan, `front-dev` dan `worker-dev`. API membuat nama-nama lingkungan dengan menggabungkan `EnvironmentName` yang ditentukan dalam file `env.yaml` dengan pilihan `group name` yang ditentukan dalam panggilan `Compose Environments`, dipisahkan oleh tanda hubung. Panjang total dua pilihan ini dan tanda hubung tidak boleh melebihi maksimum yang diizinkan panjang nama lingkungan 23 karakter.

Aplikasi yang berjalan di lingkungan `front-dev` dapat mengakses nama antrean Amazon SQS yang terlampir pada lingkungan `worker-dev` dengan membaca variabel `WORKERQUEUE`. Untuk informasi lebih lanjut terkait tautan lingkungan, lihat Membuat tautan antara lingkungan Elastic Beanstalk.

Mengelola beberapa lingkungan Elastic Beanstalk sebagai grup dengan EB CLI

Anda dapat menggunakan EB CLI untuk membuat grup lingkungan, masing-masing menjalankan komponen terpisah AWS Elastic Beanstalk dari aplikasi arsitektur berorientasi layanan. CLI EB mengelola grup tersebut dengan menggunakan ComposeEnvironmentsAPI.

> **Note**
>
> Grup lingkungan berbeda dari beberapa kontainer di lingkungan Multicontainer Docker. Dengan grup lingkungan, setiap komponen aplikasi Anda berjalan di lingkungan Elastic Beanstalk yang terpisah, dengan rangkaian instans Amazon khusus. EC2 Setiap komponen dapat menskalakan secara terpisah. Dengan Multicontainer Docker, Anda menggabungkan beberapa komponen aplikasi ke dalam satu lingkungan. Semua komponen berbagi set EC2 instans Amazon yang sama, dengan setiap instance menjalankan beberapa container Docker. Pilih salah satu dari arsitektur ini sesuai dengan kebutuhan aplikasi Anda.
> Untuk detail tentang Multicontainer Docker, lihat Menggunakan cabang platform Docker yang dikelola ECS di Elastic Beanstalk.

Mengatur komponen aplikasi Anda ke dalam struktur folder berikut:

```
~/project-name
|-- component-a
|   `-- env.yaml
`-- component-b
    `-- env.yaml
```

Setiap subfolder berisi kode sumber untuk komponen independen dari aplikasi yang akan berjalan di lingkungannya sendiri dan file definisi lingkungan bernama `env.yaml`. Untuk detail di format `env.yaml`, lihat Manifes lingkungan (`env.yaml`).

Untuk menggunakan API `Compose Environments`, jalankan eb init dari folder proyek terlebih dahulu, menentukan setiap komponen dengan nama folder yang berisi dengan opsi `--modules`:

```
~/workspace/project-name$ eb init --modules component-a component-b
```

EB CLI meminta Anda untuk mengonfigurasi setiap komponen, dan kemudian membuat direktori `.elasticbeanstalk` di setiap folder komponen. EB CLI tidak membuat file konfigurasi di direktori induk.

```
~/project-name
|-- component-a
|   |-- .elasticbeanstalk
|   `-- env.yaml
```

```
`-- component-b
    |-- .elasticbeanstalk
    `-- env.yaml
```

Selanjutnya, jalankan perintah eb create dengan daftar lingkungan untuk membuat, satu untuk setiap komponen:

```
~/workspace/project-name$ eb create --modules component-a component-b --env-group-suffix group-name
```

Perintah ini membuat lingkungan untuk setiap komponen. Nama-nama lingkungan yang dibuat dengan menggabungkan `EnvironmentName` yang ditentukan di file `env.yaml` dengan nama grup, dipisahkan dengan tanda hubung. Panjang total dua opsi ini dan tanda hubung tidak boleh melebihi maksimum panjang nama lingkungan 23 karakter yang diizinkan.

Untuk memperbarui lingkungan, gunakan perintah eb deploy:

```
~/workspace/project-name$ eb deploy --modules component-a component-b
```

Anda dapat memperbarui setiap komponen secara individu atau Anda dapat memperbaruinya sebagai grup. Tentukan komponen yang ingin Anda perbarui dengan opsi `--modules`.

EB CLI menyimpan nama grup yang digunakan dengan eb create di bagian `branch-defaults` dari file konfigurasi EB CLI di bawah `/.elasticbeanstalk/config.yml`. Untuk men-deploy aplikasi Anda ke grup yang berbeda, gunakan opsi `--env-group-suffix` ketika Anda menjalankan eb deploy. Jika grup sudah tidak ada, EB CLI akan membuat grup baru lingkungan:

```
~/workspace/project-name$ eb deploy --modules component-a component-b --env-group-suffix group-2-name
```

Untuk mengakhiri lingkungan, jalankan eb terminate di folder untuk setiap modul. Secara default, EB CLI akan menunjukkan kesalahan jika Anda mencoba untuk mengakhiri lingkungan yang lingkungan berjalan lain dependen padanya. Mengakhiri lingkungan dependen pertama, atau menggunakan opsi `--ignore-links` untuk mengganti perilaku default:

```
~/workspace/project-name/component-b$ eb terminate --ignore-links
```

Men-deploy aplikasi ke lingkungan Elastic Beanstalk

Anda dapat menggunakan AWS Elastic Beanstalk konsol untuk mengunggah bundel sumber yang diperbarui dan menerapkannya ke lingkungan Elastic Beanstalk Anda, atau menerapkan ulang versi yang diunggah sebelumnya.

Setiap deployment diidentifikasi oleh ID deployment. Penerapan IDs dimulai 1 dan bertambah satu per satu dengan setiap penerapan dan perubahan konfigurasi instance. Jika Anda mengaktifkan pelaporan kondisi yang ditingkatkan, Elastic Beanstalk menampilkan ID deployment di konsol kondisi dan EB CLI ketika melaporkan status kondisi instans. ID deployment membantu Anda menentukan keadaan lingkungan Anda ketika pembaruan bergulir mengalami kegagalan.

Elastic Beanstalk menyediakan beberapa kebijakan dan pengaturan deployment. Untuk detail tentang cara mengonfigurasi kebijakan dan pengaturan tambahan, lihat the section called “Pilihan deployment”. Tabel berikut mencantumkan kebijakan dan jenis lingkungan yang mendukung.

Kebijakan deployment yang didukung

Kebijakan deployment	Lingkungan dengan beban yang seimbang	Lingkungan instans tunggal	Lingkungan Server Windows Legasi†
Sekaligus	✓ Ya	✓ Ya	✓ Ya
Bergulir	✓ Ya	✗ Tidak	✓ Ya
Bergulir dengan batch tambahan	✓ Ya	✗ Tidak	✗ Tidak
Tidak dapat diubah	✓ Ya	✓ Ya	✗ Tidak
Pemisahan lalu lintas	✓ Ya (Application Load Balancer)	✗ Tidak	✗ Tidak

† Dalam tabel ini, Lingkungan Server Windows warisan adalah lingkungan yang didasarkan pada Konfigurasi platform Server Windows yang menggunakan versi IIS lebih awal dari IIS 8.5.

> **Warning**
>
> Beberapa kebijakan mengganti semua instans selama deployment atau pembaruan. Hal ini menyebabkan semua akumulasi saldo EC2 burst Amazon hilang. Hal ini terjadi dalam kasus berikut:
>
> - Pembaruan platform terkelola dengan penggantian instans diaktifkan
> - Pembaruan tetap
> - Deployment dengan pembaruan tetap atau pembagian lalu lintas diaktifkan

Memilih kebijakan deployment

Memilih kebijakan deployment yang tepat untuk aplikasi Anda adalah tradeoff dari beberapa pertimbangan, dan tergantung pada kebutuhan khusus Anda. Halaman the section called “Pilihan deployment” memiliki informasi lebih lanjut tentang setiap kebijakan, dan penjelasan detail tentang cara kerja beberapa dari mereka.

Daftar berikut memberikan informasi ringkasan tentang kebijakan deployment yang berbeda dan menambahkan pertimbangan terkait.

- Semua sekaligus – Metode deployment tercepat. Cocok jika Anda dapat menerima kehilangan layanan yang singkat, dan jika deployment yang cepat penting bagi Anda. Dengan metode ini, Elastic Beanstalk men-deploy versi aplikasi baru untuk setiap instans. Kemudian, proksi web atau server aplikasi mungkin perlu memulai kembali. Akibatnya, aplikasi Anda mungkin tidak tersedia untuk pengguna (atau memiliki ketersediaan rendah) untuk waktu yang singkat.
- Bergulir – Menghindari waktu henti dan meminimalkan berkurangnya ketersediaan, dengan biaya waktu deployment yang lebih lama. Cocok jika Anda tidak dapat menerima periode layanan yang hilang sepenuhnya. Dengan metode ini, aplikasi Anda di-deploy ke lingkungan Anda dengan satu batch instans pada satu waktu. Sebagian besar bandwidth dipertahankan selama deployment.
- Bergulir dengan batch tambahan – Menghindari ketersediaan yang berkurang, dengan biaya waktu deployment yang lebih lama dibandingkan dengan metode Bergulir. Cocok jika Anda harus mempertahankan bandwidth yang sama selama deployment. Dengan metode ini, Elastic Beanstalk meluncurkan batch ekstra instans, kemudian melakukan deployment bergulir. Peluncuran batch ekstra membutuhkan waktu, dan pastikan bahwa bandwidth yang sama dipertahankan selama deployment.

- Tetap – Metode deployment yang lebih lambat, yang memastikan versi aplikasi baru Anda selalu di-deploy untuk instans baru, bukan memperbarui instans yang ada. Hal ini juga memiliki keuntungan tambahan dari rollback yang cepat dan aman dalam kasus kegagalan deployment. Dengan metode ini, Elastic Beanstalk melakukan pembaruan tetap untuk men-deploy aplikasi Anda. Dalam pembaruan tetap, grup Auto Scaling kedua diluncurkan di lingkungan Anda dan versi baru melayani lalu lintas bersama versi lama sampai instans baru lolos pemeriksaan kondisi.
- Pembagian lalu lintas – Metode deployment pengujian canary. Cocok jika Anda ingin menguji kondisi versi aplikasi baru Anda menggunakan sebagian lalu lintas masuk, sambil menjaga sisa lalu lintas yang dilayani oleh versi aplikasi lama.

Tabel berikut membandingkan sifat metode deployment.

Metode deployment

Metode	Dampak penerapan yang gagal	Menyebarkan waktu	Nol downtim	Tidak ada perubah DNS	Proses rollback	Kode dikerahka n ke
Sekaligu	Waktu henti	◷	✕ Tidak	✓ Ya	Men-deplo y ulang secara manual	Untuk instans yang ada
Bergulir	Satu batch keluar dari layanan; setiap batch yang sukses sebelum kegagalan menjalankan versi aplikasi baru	◷	✓ Ya	✓ Ya	Men-deplo y ulang secara manual	Untuk instans yang ada
Bergulir dengan batch tambah	Minimal jika batch pertama gagal; sebaliknya, mirip dengan Bergulir	◷	✓ Ya	✓ Ya	Men-deplo y ulang	Instans yang baru dan

Metode	Dampak penerapan yang gagal	Menyebarkan waktu	Nol downtin	Tidak ada perubah DNS	Proses rollback	Kode dikerahka n ke
					secara manual	yang ada
Tidak dapat diubah	Minimal	🕒	🕒	✓ Ya	✓ Ya	Akhiri instans yang baru	Instans baru
Pemisa lalu lintas	Persentase lalu lintas klien diarahkan ke versi baru sementara terkena dampak	🕒	🕒	✓ Ya	✓ Ya	Mengub rute lalu lintas dan mengak i instans baru	Instans baru
Biru/ hijau	Minimal	🕒	🕒	✓ Ya	✕ Tidak	Ganti URL	Instans baru

† Bervariasi tergantung pada ukuran batch.

†† Bervariasi tergantung pada pengaturan pilihan waktu evaluasi.

Men-deploy versi aplikasi baru

Anda dapat melakukan deployment dari dashboard lingkungan Anda.

Untuk men-deploy versi aplikasi baru ke lingkungan Elastic Beanstalk

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.

3. Pilih Unggah dan deploy.
4. Gunakan formulir di layar untuk mengunggah paket sumber aplikasi.
5. Pilih Men-deploy.

Men-deploy ulang versi sebelumnya

Anda juga dapat men-deploy versi aplikasi Anda yang sebelumnya telah diunggah ke salah satu lingkungannya dari halaman versi aplikasi.

Untuk men-deploy versi aplikasi yang ada ke lingkungan yang ada

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Pada panel navigasi, pilih Aplikasi, dan kemudian pilih nama aplikasi Anda dari daftar.
3. Di panel navigasi, temukan nama aplikasi dan pilih Versi aplikasi.
4. Pilih versi aplikasi untuk digunakan.
5. Pilih Tindakan, lalu pilih Deploy.
6. Pilih lingkungan, dan kemudian pilih Deploy.

Cara lain untuk men-deploy aplikasi Anda

Jika Anda sering men-deploy, pertimbangkan untuk menggunakan Antarmuka Baris Perintah Elastic Beanstalk (EB CLI) untuk mengelola lingkungan Anda. EB CLI membuat repositori bersama dengan kode sumber Anda. Hal ini juga dapat membuat bundel sumber, unggah ke Elastic Beanstalk, dan deploy dengan satu perintah.

Untuk deployment yang bergantung pada perubahan konfigurasi sumber daya atau versi baru yang tidak dapat berjalan bersama versi lama, Anda dapat meluncurkan lingkungan baru dengan versi baru dan melakukan perubahan CNAME untuk deployment biru/hijau.

Kebijakan dan pengaturan deployment

AWS Elastic Beanstalk menyediakan beberapa opsi untuk cara penerapan diproses, termasuk kebijakan penerapan (Semua sekaligus, Rolling, Rolling dengan batch tambahan, Immutable, dan Traffic splitting) dan opsi yang memungkinkan Anda mengonfigurasi ukuran batch dan perilaku pemeriksaan kesehatan selama penerapan. Secara default, lingkungan Anda menggunakan all-at-

once penerapan. Jika Anda membuat lingkungan dengan EB CLI dan itu adalah lingkungan yang dapat diskalakan (Anda tidak menentukan pilihan `--single`), lingkungan tersebut menggunakan deployment bergulir.

Dengan penerapan bergulir, Elastic Beanstalk membagi instans EC2 Amazon lingkungan menjadi beberapa batch dan menyebarkan versi baru aplikasi ke satu batch pada satu waktu. Elastic Beanstalk meninggalkan sisa instans di lingkungan yang menjalankan versi lama dari aplikasi. Selama deployment bergulir, beberapa instans melayani permintaan dengan versi lama aplikasi, sementara instans dalam batch yang sudah selesai melayani permintaan lain dengan versi baru. Untuk rincian selengkapnya, lihat the section called “Cara kerja deployment bergulir”.

Untuk mempertahankan kapasitas penuh selama deployment, Anda dapat mengonfigurasi lingkungan Anda untuk meluncurkan batch baru instans sebelum mengeluarkan setiap instans dari layanan. Pilihan ini dikenal sebagai deployment bergulir dengan batch tambahan. Ketika deployment selesai, Elastic Beanstalk mengakhiri batch tambahan instans.

Deployment tetap melakukan pembaruan tetap untuk meluncurkan serangkaian lengkap instans baru yang menjalankan versi baru aplikasi dalam grup Auto Scaling terpisah, bersama dengan instans yang menjalankan versi lama. Deployment tetap dapat mencegah masalah yang disebabkan oleh deployment bergilir yang sebagian telah selesai. Jika instans baru tidak lolos pemeriksaan kondisi, Elastic Beanstalk mengakhiri mereka, meninggalkan instans asli tidak tersentuh.

Deployment pembagian lalu lintas memungkinkan Anda melakukan pengujian canary sebagai bagian dari deployment aplikasi Anda. Dalam deployment pembagian lalu lintas, Elastic Beanstalk meluncurkan satu set lengkap instans baru seperti selama deployment tetap. Kemudian meneruskan persentase tertentu dari lalu lintas klien masuk ke versi aplikasi baru untuk periode evaluasi tertentu. Jika instans baru tetap sehat, Elastic Beanstalk meneruskan semua lalu lintas ke mereka dan mengakhiri yang lama. Jika instans baru tidak lolos pemeriksaan kondisi, atau jika Anda memilih untuk membatalkan deployment, Elastic Beanstalk memindahkan lalu lintas kembali ke instans lama dan mengakhiri yang baru. Tidak pernah ada gangguan layanan. Untuk rincian selengkapnya, lihat the section called “Cara kerja deployment pembagian lalu lintas”.

> **Warning**
>
> Beberapa kebijakan mengganti semua instans selama deployment atau pembaruan. Hal ini menyebabkan semua akumulasi saldo EC2 burst Amazon hilang. Hal ini terjadi dalam kasus berikut:
>
> - Pembaruan platform terkelola dengan penggantian instans diaktifkan

- Pembaruan tetap
- Deployment dengan pembaruan yang tidak berubah atau pembagian lalu lintas diaktifkan

Jika aplikasi Anda tidak lolos semua pemeriksaan kondisi, namun tetap beroperasi dengan benar pada status kondisi yang lebih rendah, Anda dapat mengizinkan instans untuk lolos pemeriksaan kondisi dengan status lebih rendah, seperti `Warning`, dengan memodifikasi pilihan Ambang batas sehat. Jika deployment Anda gagal karena tidak lolos pemeriksaan kondisi dan Anda perlu untuk memaksa pembaruan terlepas dari status kondisi, tentukan pilihan Abaikan pemeriksaan kondisi.

Ketika Anda menentukan ukuran batch untuk pembaruan bergulir, Elastic Beanstalk juga menggunakan nilai tersebut untuk memulai kembali aplikasi bergulir. Gunakan mulai kembali bergulir ketika Anda perlu untuk memulai kembali server proksi dan aplikasi yang berjalan pada instans lingkungan Anda tanpa waktu henti.

Mengonfigurasi deployment aplikasi

Di konsol manajemen lingkungan, aktifkan dan konfigurasikan deployment versi aplikasi batch dengan mengedit Pembaruan dan Deployment di halaman Konfigurasi lingkungan.

Untuk mengonfigurasi deployment (konsol)

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Di panel navigasi, pilih Konfigurasi.
4. Di kategori konfigurasi Pembaruan dan deployment bergulir, pilih Edit.
5. Di bagian Deployment Aplikasi, pilih Kebijakan deployment, pengaturan batch, dan pilihan pemeriksaan kondisi.
6. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Bagian Deployment aplikasi dari Pembaruan dan deployment bergulir memiliki pilihan berikut untuk deployment aplikasi:

- Kebijakan deployment – Pilih salah satu opsi deployment berikut:

- Semua sekaligus – Men-deploy versi baru untuk semua instans secara bersamaan. Semua instans di lingkungan Anda keluar dari layanan untuk waktu yang singkat sementara deployment terjadi.
- Bergulir – Men-deploy versi baru dalam batch. Setiap batch dikeluarkan dari layanan selama tahap deployment, mengurangi kapasitas lingkungan Anda dengan jumlah instans dalam batch.
- Bergulir dengan batch tambahan – Men-deploy versi baru dalam batch, tetapi pertama-tama luncurkan batch baru dari instans untuk memastikan kapasitas penuh selama proses deployment.
- Tidak berubah – Men-deploy versi baru ke kelompok baru instans dengan melakukan pembaruan yang tidak dapat diubah.
- Pembagian lalu lintas – Men-deploy versi baru ke kelompok baru instans dan membagi lalu lintas klien masuk dengan sementara antara versi aplikasi yang sudah ada dan yang baru.

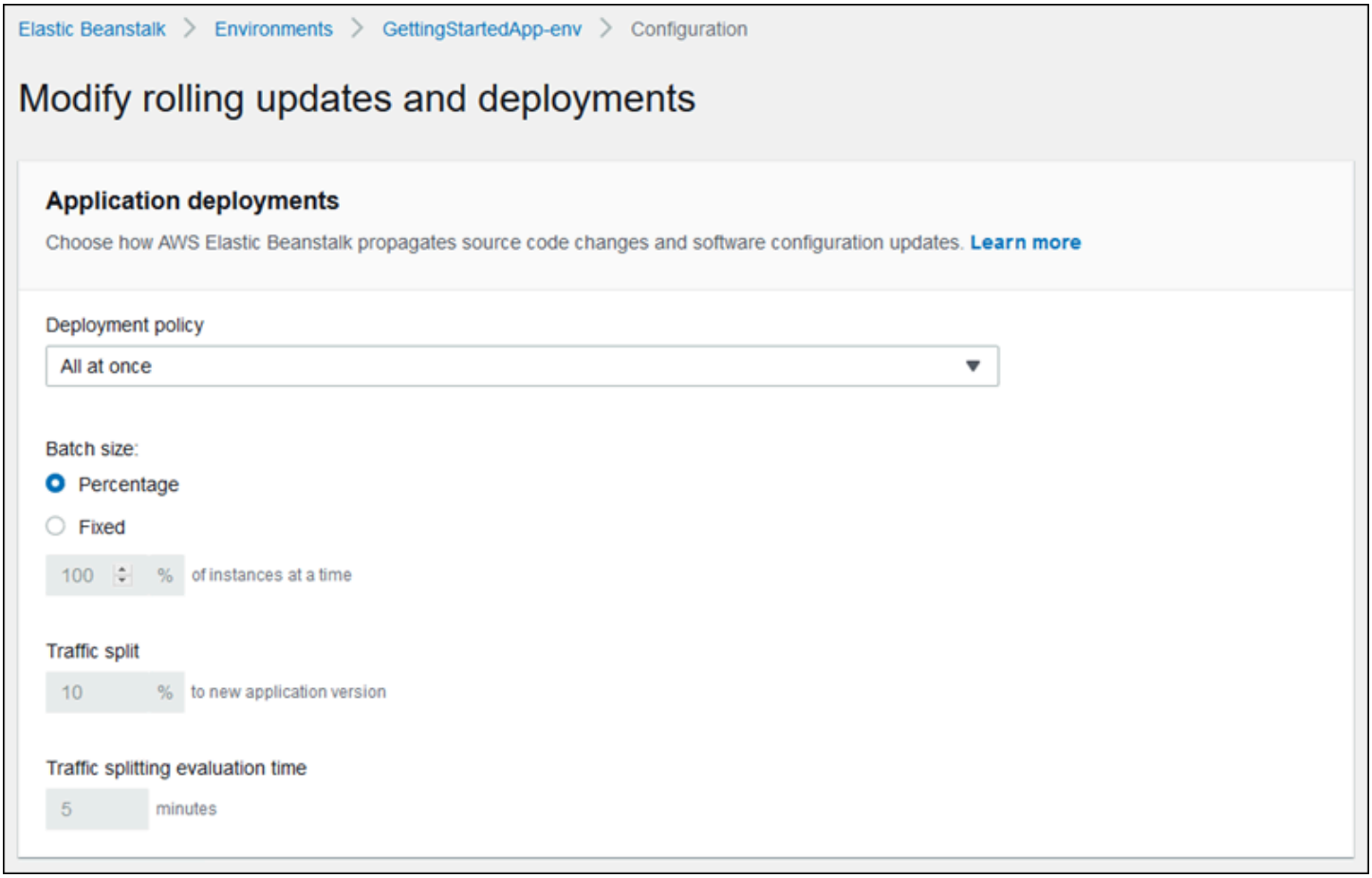
Untuk kebijakan deployment Bergulir dan Bergulir dengan batch tambahan Anda dapat mengonfigurasi:

- Ukuran Batch – Ukuran set instans untuk di-deploy di setiap batch.

  Pilih Persentase untuk mengonfigurasi persentase dari jumlah total EC2 instans dalam grup Auto Scaling (hingga 100 persen), atau pilih Tetap untuk mengonfigurasi jumlah instans tetap (hingga jumlah instans maksimum dalam konfigurasi Auto Scaling lingkungan Anda).

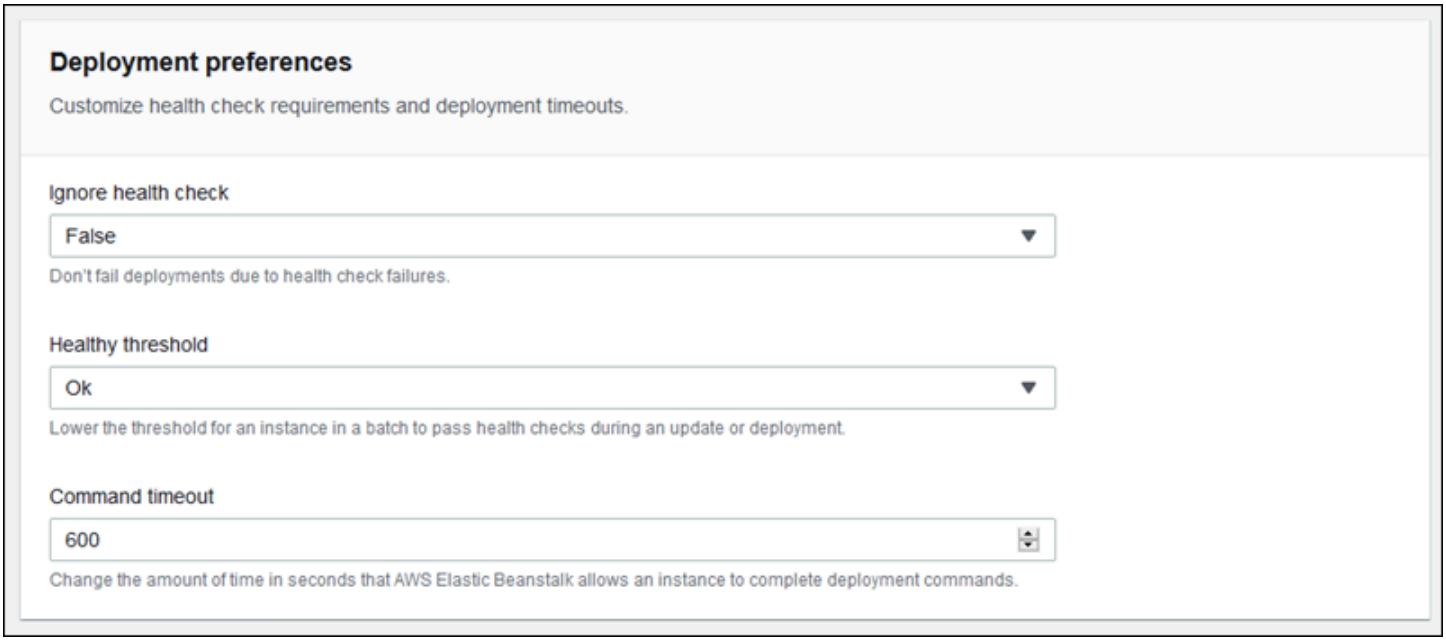
Untuk kebijakan deployment Pembagian lalu lintas Anda dapat mengonfigurasi berikut ini:

- Pembagian lalu lintas – Persentase awal lalu lintas klien masuk yang digeser oleh Elastic Beanstalk ke instans lingkungan yang menjalankan versi aplikasi baru yang Anda deploy.
- Waktu evaluasi pembagian lalu lintas – Periode waktu, dalam menit, di mana Elastic Beanstalk menunggu setelah deployment awal yang sehat sebelum melanjutkan untuk menggeser semua lalu lintas klien masuk ke versi aplikasi baru yang Anda deploy.



Bagian Preferensi deployment berisi pilihan yang berkaitan dengan pemeriksaan kondisi.

- Abaikan pemeriksaan kondisi – Mencegah deployment dari pengguliran kembali ketika batch gagal untuk menjadi sehat dalam Perintah waktu habis.
- Ambang batas sehat – Menurunkan ambang batas di mana instans dianggap sehat selama deployment bergulir, pembaruan bergulir, dan pembaruan yang tidak berubah.
- Perintah waktu habis – Jumlah detik untuk menunggu instans menjadi sehat sebelum membatalkan deployment atau, jika Abaikan pemeriksaan kondisi diatur, untuk melanjutkan ke batch berikutnya.



Cara kerja deployment bergulir

Saat memproses batch, Elastic Beanstalk melepaskan semua instans dalam batch dari penyeimbang beban, men-deploy versi aplikasi baru, dan kemudian memasangkan ulang instans. Jika Anda mengaktifkan pengurasan koneksi, Elastic Beanstalk menguras koneksi yang ada dari instans EC2 Amazon di setiap batch sebelum memulai penerapan.

Setelah memasangkan ulang instans dalam batch ke load balancer, Elastic Load Balancing menunggu sampai mereka melewati jumlah minimum pemeriksaan kondisi Elastic Load Balancing (nilai Ambang batas hitungan pemeriksaan kesehatan), dan kemudian mulai perutean lalu lintas ke mereka. Jika tidak ada URL pemeriksaan kondisi yang dikonfigurasi, hal ini dapat terjadi sangat cepat, karena instans akan lolos pemeriksaan kondisi segera setelah dapat menerima koneksi TCP. Jika URL pemeriksaan kondisi dikonfigurasi, penyeimbang beban tidak merutekan lalu lintas ke instans yang diperbarui sampai mereka mengembalikan kode status `200 OK` dalam menanggapi permintaan `HTTP GET` ke URL pemeriksaan kondisi.

Elastic Beanstalk menunggu sampai semua instans dalam batch sehat sebelum pindah ke batch berikutnya. Dengan pelaporan kondisi dasar, kondisi instans bergantung pada status pemeriksaan kondisi Elastic Load Balancing. Ketika semua instans dalam batch lolos pemeriksaan kondisi yang cukup untuk dianggap sehat oleh Elastic Load Balancing, batch tersebut telah selesai. Jika pelaporan kondisi yang ditingkatkan diaktifkan, Elastic Beanstalk mempertimbangkan beberapa faktor lain, termasuk hasil permintaan yang masuk. Dengan pelaporan kondisi yang ditingkatkan, semua instans harus melewati 12 pemeriksaan kondisi berturut-turut dengan Status OK dalam waktu dua menit

untuk lingkungan server web, dan 18 pemeriksaan kondisi dalam waktu tiga menit untuk lingkungan pekerja.

Jika batch instans tidak menjadi sehat dalam perintah waktu habis, deployment gagal. Setelah deployment gagal, periksa kondisi instans di lingkungan Anda untuk informasi tentang penyebab kegagalan. Kemudian lakukan deployment lain dengan versi tetap atau diketahui baik dari aplikasi Anda untuk memutar kembali.

Jika deployment mengalami kegagalan setelah satu atau lebih batch telah berhasil terselesaikan, batch yang telah selesai menjalankan versi baru aplikasi Anda sementara batch yang tertunda tetap menjalankan versi lama. Anda dapat mengidentifikasi versi yang berjalan pada instans di lingkungan Anda pada halaman kondisi di konsol. Halaman ini menampilkan ID deployment terbaru dari deployment yang terbaru yang dijalankan pada setiap instans di lingkungan Anda. Jika Anda mengakhiri instans dari deployment yang gagal, Elastic Beanstalk menggantikan instans tersebut dengan instans yang menjalankan versi aplikasi dari deployment yang sukses terbaru.

Cara kerja deployment pembagian lalu lintas

Deployment pembagian lalu lintas memungkinkan Anda melakukan pengujian canary. Anda mengarahkan beberapa lalu lintas klien masuk ke versi aplikasi baru Anda untuk memverifikasi kondisi aplikasi sebelum melakukan ke versi baru dan mengarahkan semua lalu lintas ke sana.

Selama deployment pembagian lalu lintas, Elastic Beanstalk membuat serangkaian instans baru dalam grup Auto Scaling terpisah yang sementara. Elastic Beanstalk kemudian menginstruksikan penyeimbang beban untuk mengarahkan persentase tertentu dari lalu lintas masuk lingkungan Anda ke instans baru. Kemudian, untuk jumlah waktu yang dikonfigurasi, Elastic Beanstalk melacak kondisi rangkaian instans baru. Jika semuanya baik-baik saja, Elastic Beanstalk menggeser lalu lintas yang tersisa ke instans baru dan melampirkannya ke grup Auto Scaling lingkungan asli, menggantikan instans lama. Kemudian Elastic Beanstalk membersihkan—mengakhiri instans lama dan menghapus grup Auto Scaling sementara.

> **Note**
>
> Kapasitas lingkungan tidak berubah selama deployment pembagian lalu lintas. Elastic Beanstalk meluncurkan jumlah instans yang sama dalam grup Auto Scaling sementara karena ada dalam grup Auto Scaling asli pada saat deployment dimulai. Kemudian mempertahankan sejumlah konstan instans di kedua grup Auto Scaling untuk durasi

deployment. Perhatikan fakta ini saat mengonfigurasi waktu evaluasi pembagian lalu lintas lingkungan.

Menggulir kembali deployment ke versi aplikasi sebelumnya berlangsung dengan cepat dan tidak mempengaruhi layanan untuk lalu lintas klien. Jika instans baru tidak lolos pemeriksaan kondisi, atau jika Anda memilih untuk membatalkan deployment, Elastic Beanstalk memindahkan lalu lintas kembali ke instans lama dan mengakhiri yang baru. Anda dapat membatalkan deployment apa pun dengan menggunakan halaman gambaran umum lingkungan di konsol Elastic Beanstalk, dan pilih Batalkan operasi saat ini di Tindakan lingkungan. Anda juga dapat memanggil AbortEnvironmentUpdateAPI atau AWS CLI perintah yang setara.

Deployment pembagian lalu lintas memerlukan Application Load Balancer. Elastic Beanstalk menggunakan jenis penyeimbang beban ini secara default saat Anda membuat lingkungan Anda menggunakan konsol Elastic Beanstalk atau EB CLI.

Namespace pilihan deployment

Anda dapat menggunakan pilihan konfigurasi di namespace `aws:elasticbeanstalk:command` untuk mengonfigurasi deployment Anda. Jika Anda memilih kebijakan pembagian lalu lintas, pilihan tambahan untuk kebijakan ini tersedia di namespace `aws:elasticbeanstalk:trafficsplitting`.

Gunakan pilihan `DeploymentPolicy` untuk mengatur jenis deployment. Nilai berikut didukung:

- `AllAtOnce` – Menonaktifkan deployment bergulir dan selalu men-deploy ke semua instans secara bersamaan.
- `Rolling` – Mengaktifkan deployment bergulir standar.
- `RollingWithAdditionalBatch` – Meluncurkan batch ekstra instans, sebelum memulai deployment, untuk mempertahankan kapasitas penuh.
- `Immutable` – Melakukan pembaruan tetap untuk setiap deployment.
- `TrafficSplitting` – Melakukan deployment pembagian lalu lintas untuk menguji canary deployment aplikasi Anda.

Bila Anda mengaktifkan deployment bergulir, atur `BatchSize` dan pilihan `BatchSizeType` untuk mengonfigurasi ukuran setiap batch. Misalnya, untuk men-deploy 25 persen dari semua instans di setiap batch, tentukan pilihan dan nilai berikut.

Example .ebextensions/rolling-updates.config

```
option_settings:
  aws:elasticbeanstalk:command:
    DeploymentPolicy: Rolling
    BatchSizeType: Percentage
    BatchSize: 25
```

Untuk men-deploy ke lima instans di setiap batch, terlepas dari jumlah instans yang sedang berjalan, dan untuk memunculkan batch tambahan lima instans menjalankan versi baru sebelum mengeluarkan setiap instans dari layanan, tentukan pilihan dan nilai berikut.

Example .ebextensions/rolling-additionalbatch.config

```
option_settings:
  aws:elasticbeanstalk:command:
    DeploymentPolicy: RollingWithAdditionalBatch
    BatchSizeType: Fixed
    BatchSize: 5
```

Untuk melakukan pembaruan yang tidak berubah untuk setiap deployment dengan ambang batas pemeriksaan kondisi Peringatan, dan lanjutkan dengan deployment bahkan jika instans dalam batch tidak lolos pemeriksaan kondisi dalam batas waktu 15 menit, tentukan pilihan dan nilai berikut.

Example .ebextensions/immutable-ignorehealth.config

```
option_settings:
  aws:elasticbeanstalk:command:
    DeploymentPolicy: Immutable
    HealthCheckSuccessThreshold: Warning
    IgnoreHealthCheck: true
    Timeout: "900"
```

Untuk melakukan deployment pembagian lalu lintas, teruskan 15 persen lalu lintas klien ke versi aplikasi baru dan mengevaluasi kondisi selama 10 menit, tentukan pilihan dan nilai berikut.

Example .ebextensions/traffic-splitting.config

```
option_settings:
  aws:elasticbeanstalk:command:
```

```
    DeploymentPolicy: TrafficSplitting
  aws:elasticbeanstalk:trafficsplitting:
    NewVersionPercent: "15"
    EvaluationTime: "10"
```

Konsol EB CLI dan Elastic Beanstalk menerapkan nilai yang direkomendasikan untuk pilihan sebelumnya. Anda harus menghapus pengaturan ini jika Anda ingin menggunakan file konfigurasi untuk mengonfigurasi hal yang sama. Lihat Nilai yang disarankan untuk rincian selengkapnya.

Deployment Biru/Hijau dengan Elastic Beanstalk

Karena AWS Elastic Beanstalk melakukan pembaruan di tempat saat Anda memperbarui versi aplikasi, aplikasi Anda mungkin menjadi tidak tersedia bagi pengguna untuk waktu yang singkat. Untuk menghindari hal ini, lakukan penyebaran biru/hijau. Untuk melakukan ini, terapkan versi baru ke lingkungan yang terpisah, dan kemudian tukar CNAMEs dari dua lingkungan untuk mengarahkan lalu lintas ke versi baru secara instan.

Penerapan biru/hijau juga diperlukan jika Anda ingin memperbarui lingkungan ke versi platform yang tidak kompatibel. Untuk informasi selengkapnya, lihat the section called “Pembaruan platform”.

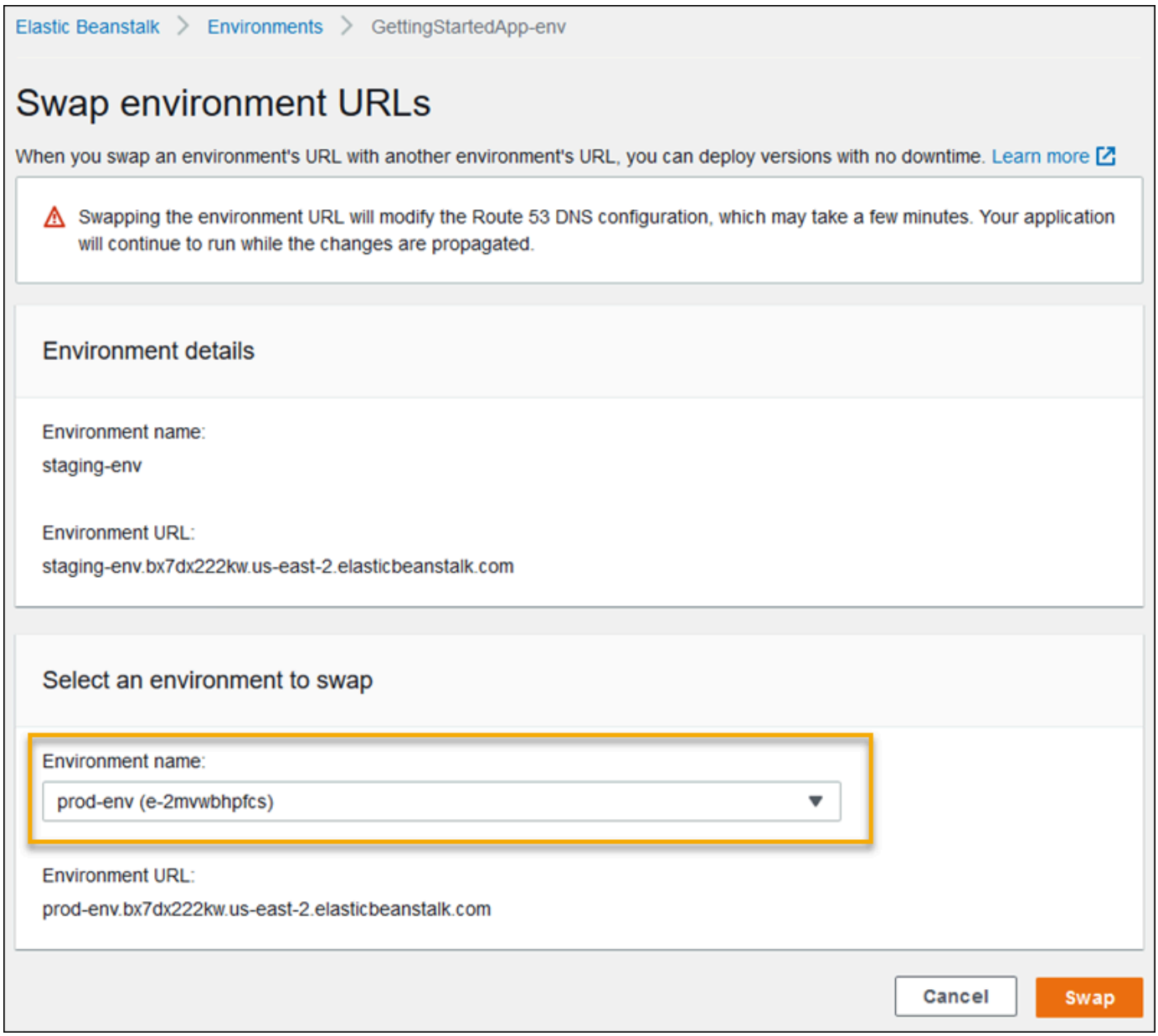
Deployment biru/hijau mengharuskan lingkungan Anda berjalan secara independen dari basis data produksi Anda, jika aplikasi Anda menggunakannya. Jika lingkungan Anda menyertakan database yang Elastic Beanstalk dibuat atas nama Anda, database dan koneksi lingkungan tidak dipertahankan kecuali Anda mengambil tindakan tertentu. Jika Anda memiliki database yang ingin Anda pertahankan, gunakan salah satu opsi siklus hidup database Elastic Beanstalk. Anda dapat memilih opsi Retain untuk menjaga database dan lingkungan tetap beroperasi setelah memisahkan database. Untuk informasi selengkapnya lihat Siklus hidup basis data di bagian Mengkonfigurasi lingkungan dari panduan ini.

Untuk petunjuk tentang cara mengonfigurasi aplikasi agar tersambung ke instans Amazon RDS yang tidak dikelola oleh Elastic Beanstalk, lihat. Menggunakan Elastic Beanstalk dengan Amazon RDS

Untuk melakukan deployment biru/hijau

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Kloning lingkungan Anda saat ini, atau luncurkan lingkungan baru untuk menjalankan versi platform yang Anda inginkan.
3. Men-deploy versi aplikasi baru ke lingkungan baru.

4. Uji versi baru pada lingkungan baru.
5. Pada halaman ikhtisar lingkungan, pilih Tindakan, lalu pilih Swap lingkungan URLs.
6. Untuk Nama lingkungan, pilih lingkungan saat ini.



7. Pilih Ubah.

Elastic Beanstalk menukar catatan CNAME dari lingkungan lama dan baru, mengarahkan lalu lintas dari versi lama ke versi baru.

Setelah Elastic Beanstalk menyelesaikan operasi perubahan, verifikasi bahwa lingkungan baru merespon ketika Anda mencoba untuk terhubung ke URL lingkungan lama. Namun, jangan akhiri

lingkungan lama Anda sampai perubahan DNS disebarkan dan catatan DNS lama kedaluwarsa. Server DNS tidak selalu menghapus catatan lama dari cache mereka berdasarkan waktu untuk hidup (TTL) yang Anda tetapkan pada catatan DNS Anda.

Perubahan konfigurasi

Saat Anda mengubah pengaturan opsi konfigurasi di bagian Konfigurasi konsol manajemen lingkungan, AWS Elastic Beanstalk menyebarkan perubahan ke semua sumber daya yang terpengaruh. Sumber daya ini mencakup penyeimbang beban yang mendistribusikan lalu lintas ke EC2 instans Amazon yang menjalankan aplikasi Anda, grup Auto Scaling yang mengelola instance tersebut, dan instans itu sendiri. EC2

Banyak perubahan konfigurasi dapat diterapkan ke lingkungan yang sedang berjalan tanpa mengganti instans yang ada. Sebagai contoh, menetapkan URL pemeriksaan kondisi memicu pembaruan lingkungan untuk memodifikasi pengaturan penyeimbang beban, tetapi tidak menyebabkan waktu henti apapun karena instans yang menjalankan aplikasi Anda terus melayani permintaan sementara pembaruan disebarkan.

Perubahan konfigurasi yang memodifikasi konfigurasi peluncuran atau pengaturan VPC memerlukan diakhirinya semua instans di lingkungan Anda dan menggantinya. Misalnya, ketika Anda mengubah jenis instans atau setelan kunci SSH untuk lingkungan Anda, EC2 instance harus dihentikan dan diganti. Elastic Beanstalk menyediakan beberapa kebijakan yang menentukan bagaimana penggantian ini dilakukan.

- Pembaruan bergulir – Elastic Beanstalk menerapkan perubahan konfigurasi Anda dalam batch, menjaga jumlah minimum instans yang berjalan dan melayani lalu lintas setiap saat. Pendekatan ini mencegah waktu henti selama proses pembaruan. Untuk detail selengkapnya, lihat Pembaruan bergulir.
- Pembaruan yang tidak berubah – Elastic Beanstalk meluncurkan grup Auto Scaling sementara di luar lingkungan Anda dengan serangkaian instans terpisah yang berjalan dengan konfigurasi baru. Kemudian Elastic Beanstalk menempatkan instans ini di belakang load balancer lingkungan Anda. Instans lama dan baru melayani lalu lintas sampai instans baru lolos dalam pemeriksaan kondisi. Pada saat itu, Elastic Beanstalk menggerakkan instans baru ke dalam grup Auto Scaling lingkungan Anda dan mengakhiri grup sementara dan instans lama. Untuk detail selengkapnya, lihat Pembaruan tetap.
- Nonaktif – Elastic Beanstalk tidak berusaha untuk menghindari waktu henti. Elastic Beanstalk mengakhiri instans lingkungan Anda yang ada dan menggantikannya dengan instans baru yang berjalan dengan konfigurasi baru.

> **Warning**
>
> Beberapa kebijakan mengganti semua instans selama deployment atau pembaruan. Hal ini menyebabkan semua akumulasi saldo EC2 burst Amazon hilang. Hal ini terjadi dalam kasus berikut:
>
> - Pembaruan platform terkelola dengan penggantian instans diaktifkan
> - Pembaruan tetap
> - Deployment dengan pembaruan yang tidak berubah atau pembagian lalu lintas diaktifkan

Tipe pembaruan yang didukung

Pengaturan pembaruan bergulir	Lingkungan dengan beban yang seimbang	Lingkungan instans tunggal	Lingkungan server Windows legasi†
Nonaktif	✓ Ya	✓ Ya	✓ Ya
Bergulir Berdasarkan Kondisi	✓ Ya	✗ Tidak	✓ Ya
Bergulir Berdasarkan Waktu	✓ Ya	✗ Tidak	✓ Ya
Tetap	✓ Ya	✓ Ya	✗ Tidak

† Untuk tujuan tabel ini, Lingkungan Server Windows Legasi adalah sebuah lingkungan yang didasarkan pada Konfigurasi platform Server Windows yang menggunakan versi IIS lebih awal dari IIS 8.5.

Topik

- Pembaruan konfigurasi lingkungan bergulir Elastic Beanstalk
- Pembaruan lingkungan tetap

Pembaruan konfigurasi lingkungan bergulir Elastic Beanstalk

Saat perubahan konfigurasi memerlukan penggantian instans, Elastic Beanstalk dapat melakukan pembaruan dalam batch untuk menghindari waktu henti sementara perubahan disebarkan. Selama pembaruan bergulir, kapasitas hanya dikurangi dengan ukuran satu batch, yang dapat Anda konfigurasikan. Elastic Beanstalk mengambil satu batch instans dari layanan, mengakhirinya, kemudian meluncurkan batch dengan konfigurasi baru. Setelah batch baru mulai melayani permintaan, Elastic Beanstalk melanjutkan ke batch berikutnya.

Batch pembaruan konfigurasi bergulir dapat diproses secara berkala (berbasis waktu), dengan penundaan antara setiap batch, atau berdasarkan kondisi. Untuk pembaruan bergulir berbasis waktu, Anda dapat mengonfigurasi jumlah waktu tunggu Elastic Beanstalk setelah menyelesaikan peluncuran batch instans sebelum melanjutkan ke batch berikutnya. Waktu jeda ini mengizinkan aplikasi Anda untuk melakukan bootstrap dan mulai melayani permintaan.

Dengan pembaruan bergulir berbasis kondisi, Elastic Beanstalk menunggu sampai instans dalam batch lolos dalam pemeriksaan kondisi sebelum beralih ke batch berikutnya. Kondisi sebuah instans ditentukan oleh sistem pelaporan kondisi, bisa dasar atau yang ditingkatkan. Dengan kondisi dasar, batch dianggap sehat segera setelah semua instans di dalamnya lolos dalam pemeriksaan kondisi Elastic Load Balancing (ELB).

Dengan laporan kondisi yang ditingkatkan, semua instans dalam batch harus melewati beberapa pemeriksaan kondisi berturut-turut sebelum Elastic Beanstalk akan beralih ke batch berikutnya. Selain pemeriksaan kondisi ELB, yang hanya memeriksa instans Anda, kondisi yang ditingkatkan memantau log aplikasi dan status sumber daya lingkungan Anda yang lain. Dalam lingkungan server web dengan kondisi yang ditingkatkan, semua instans harus lolos dalam 12 pemeriksaan kondisi selama dua menit (18 pemeriksaan selama tiga menit untuk lingkungan pekerja). Jika setiap instans gagal dalam satu pemeriksaan kondisi, hitungan akan kembali ke awal.

Jika batch tidak menjadi sehat dalam waktu habis pembaruan bergulir (default adalah 30 menit), pembaruan dibatalkan. Waktu habis pembaruan bergulir adalah pilihan konfigurasi yang tersedia di namespace `aws:autoscaling:updatepolicy:rollingupdate`. Jika aplikasi Anda tidak lolos pemeriksaan kondisi dengan status Ok tetapi stabil pada tingkat yang berbeda, Anda dapat mengatur pilihan `HealthCheckSuccessThreshold` pada namespace `aws:elasticbeanstalk:healthreporting:system` untuk mengubah tingkat di mana Elastic Beanstalk menganggap sebuah instans sehat.

Jika proses pembaruan bergulir gagal, Elastic Beanstalk memulai pembaruan bergulir lain untuk kembali ke konfigurasi sebelumnya. Pembaruan bergulir dapat gagal karena pemeriksaan kondisi

yang gagal atau jika meluncurkan instans baru menyebabkan Anda melebihi kuota di akun Anda. Jika Anda menekan kuota pada jumlah EC2 instans Amazon, misalnya, pembaruan bergulir dapat gagal saat mencoba menyediakan batch instance baru. Dalam kasus ini, rollback juga gagal.

Rollback yang gagal mengakhiri proses pembaruan dan meninggalkan lingkungan Anda dalam keadaan tidak sehat. Batch yang belum diproses masih menjalankan instans dengan konfigurasi lama, sementara batch yang berhasil diselesaikan memiliki konfigurasi baru. Untuk memperbaiki lingkungan setelah rollback gagal, pertama selesaikan masalah yang menyebabkan pembaruan gagal, lalu mulai pembaruan lingkungan lain.

Metode alternatif adalah dengan men-deploy versi baru dari aplikasi Anda ke lingkungan yang berbeda dan kemudian melakukan perubahan CNAME untuk mengarahkan lalu lintas dengan waktu henti nol. Lihat Deployment Biru/Hijau dengan Elastic Beanstalk untuk informasi selengkapnya.

Pembaruan bergulir versus deployment bergulir

Pembaruan bergulir terjadi ketika Anda mengubah setelan yang memerlukan EC2 instans Amazon baru untuk disediakan untuk lingkungan Anda. Ini termasuk perubahan konfigurasi grup Auto Scaling, seperti jenis instans dan pengaturan pasangan kunci, dan perubahan pengaturan VPC. Dalam pembaruan bergulir, setiap batch instans diakhiri sebelum batch baru ditetapkan untuk menggantikannya.

Deployment bergulir terjadi setiap kali Anda men-deploy aplikasi Anda dan biasanya dapat dilakukan tanpa mengganti instans di lingkungan Anda. Elastic Beanstalk mengambil setiap batch dari layanan, men-deploy versi aplikasi baru, dan kemudian menempatkannya kembali dalam layanan.

Pengecualian untuk ini adalah jika Anda mengubah pengaturan yang memerlukan penggantian instans pada saat yang sama Anda men-deploy versi aplikasi baru. Sebagai contoh, jika Anda mengubah pengaturan nama kunci di file konfigurasi dalam paket sumber Anda dan deploy ke lingkungan Anda, Anda memicu pembaruan bergulir. Alih-alih men-deploy versi aplikasi baru Anda untuk setiap batch instans yang ada, batch baru instans ditetapkan dengan konfigurasi baru. Dalam kasus ini, deployment terpisah tidak terjadi karena instans baru dinaikkan dengan versi aplikasi baru.

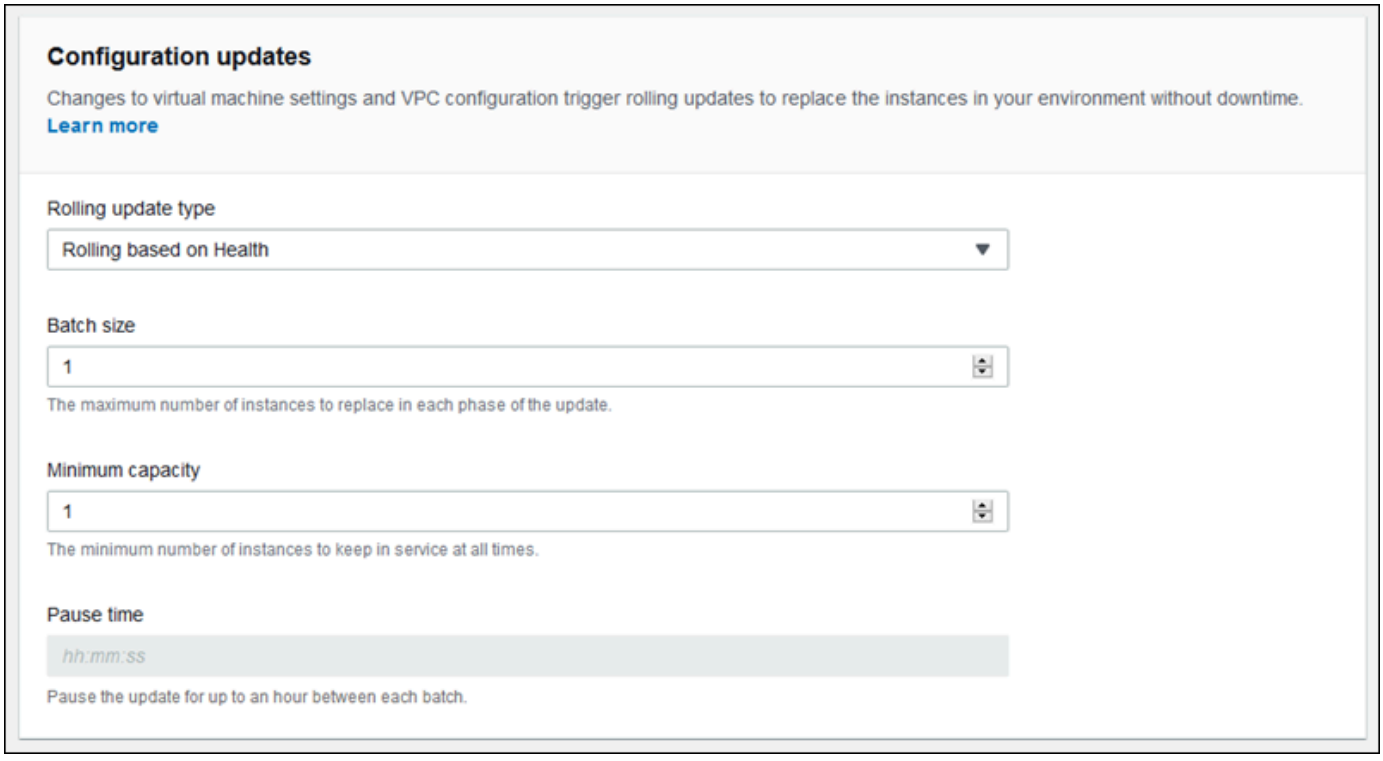
Kapan saja instans baru ditetapkan sebagai bagian dari pembaruan lingkungan, ada fase deployment di mana kode sumber aplikasi Anda di-deploy ke instans baru dan setiap pengaturan konfigurasi yang memodifikasi sistem operasi atau perangkat lunak pada instans diterapkan. Pengaturan pemeriksaan kondisi deployment (Abaikan pemeriksaan kondisi, Ambang batas sehat, dan Batas waktu perintah) juga berlaku untuk pembaruan bergulir berbasis kondisi dan pembaruan tetap selama fase deployment.

Mengonfigurasi pembaruan bergulir

Anda dapat mengaktifkan dan mengonfigurasi pembaruan bergulir di konsol Elastic Beanstalk.

Untuk mengaktifkan pembaruan bergulir

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Di panel navigasi, pilih Konfigurasi.
4. Di kategori konfigurasi Pembaruan dan deployment bergulir, pilih Edit.
5. Di bagian Pembaruan konfigurasi, untuk Jenis pembaruan bergulir, pilih salah satu pilihan Bergulir.



6. Pilih pengaturan Ukuran batch, Kapasitas minimum, dan Jeda waktu.
7. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Bagian Pembaruan konfigurasi di halaman Pembaruan dan deployment bergulir memiliki pilihan berikut untuk pembaruan bergulir:

- Jenis pembaruan bergulir – Elastic Beanstalk menunggu setelah selesai memperbarui batch instans sebelum berpindah ke batch berikutnya, untuk mengizinkan instans tersebut menyelesaikan bootstrapping dan mulai melayani lalu lintas. Pilih salah satu pilihan berikut:
  - Bergulir berdasarkan Kondisi – Tunggu sampai instans dalam batch saat ini menjadi sehat sebelum menempatkan instans dalam layanan dan memulai batch berikutnya.
  - Bergulir berdasarkan Waktu – Tentukan jumlah waktu untuk menunggu antara meluncurkan instans baru dan menempatkannya di layanan sebelum memulai batch berikutnya.
  - Tidak berubah – Terapkan versi baru ke grup baru instans dengan melakukan pembaruan tetap.
- Ukuran Batch – Jumlah instans untuk diganti di setiap batch, antara **1** dan **10000**. Secara default, nilai ini adalah sepertiga dari ukuran minimum grup Auto Scaling, dibulatkan ke seluruh nomor.
- Kapasitas minimum – Jumlah minimum dari instans untuk tetap berjalan sementara instans lain diperbarui, antara **0** dan **9999**. Nilai default adalah ukuran minimum grup Auto Scaling atau kurang dari ukuran maksimum grup Auto Scaling, nomor mana yang lebih rendah.
- Jeda waktu (hanya berbasis waktu) – Jumlah waktu untuk menunggu setelah batch diperbarui sebelum berpindah ke batch berikutnya, untuk mengizinkan aplikasi Anda untuk mulai menerima lalu lintas. Antara 0 detik dan satu jam.

Namespace aws:autoscaling:updatepolicy:rollingupdate

Anda dapat menggunakan pilihan konfigurasi di namespace `aws:autoscaling:updatepolicy:rollingupdate` untuk mengonfigurasi deployment Anda.

Gunakan pilihan `RollingUpdateEnabled` untuk mengaktifkan pembaruan bergulir, dan `RollingUpdateType` untuk memilih jenis pembaruan. Nilai berikut didukung untuk `RollingUpdateType`:

- `Health` – Tunggu sampai instans dalam batch saat ini sehat sebelum menempatkan instans dalam layanan dan memulai batch berikutnya.
- `Time` – Tentukan jumlah waktu untuk menunggu antara meluncurkan instans baru dan menempatkannya di layanan sebelum memulai batch berikutnya.
- `Immutable` – Terapkan perubahan konfigurasi untuk kelompok baru instans dengan melakukan pembaruan tetap.

Bila Anda mengaktifkan deployment bergulir, atur pilihan `MaxBatchSize` dan `MinInstancesInService` untuk mengonfigurasi ukuran setiap batch. Untuk pembaruan bergulir

berbasis waktu dan berbasis kondisi, Anda juga dapat mengonfigurasi `PauseTime` dan `Timeout`, masing-masing.

Misalnya, untuk meluncurkan hingga lima instans sekaligus, sambil mempertahankan setidaknya dua instans dalam layanan, dan menunggu lima menit dan 30 detik di antara batch, tentukan pilihan dan nilai berikut.

Example .ebextensions/timebased.config

```
option_settings:
  aws:autoscaling:updatepolicy:rollingupdate:
    RollingUpdateEnabled: true
    MaxBatchSize: 5
    MinInstancesInService: 2
    RollingUpdateType: Time
    PauseTime: PT5M30S
```

Untuk mengaktifkan pembaruan bergulir berbasis kondisi, dengan batas waktu 45 menit untuk setiap batch, tentukan pilihan dan nilai berikut.

Example .ebextensions/healthbased.config

```
option_settings:
  aws:autoscaling:updatepolicy:rollingupdate:
    RollingUpdateEnabled: true
    MaxBatchSize: 5
    MinInstancesInService: 2
    RollingUpdateType: Health
    Timeout: PT45M
```

`Timeout`dan `PauseTime` nilai harus ditentukan dalam durasi ISO86 01:`PT#H#M#S`, di mana masing-masing # adalah jumlah jam, menit, atau detik, masing-masing.

Konsol EB CLI dan Elastic Beanstalk menerapkan nilai yang direkomendasikan untuk pilihan sebelumnya. Anda harus menghapus pengaturan ini jika Anda ingin menggunakan file konfigurasi untuk mengonfigurasi hal yang sama. Lihat Nilai yang disarankan untuk rincian selengkapnya.

Pembaruan lingkungan tetap

Pembaruan lingkungan tetap adalah alternatif untuk pembaruan bergulir. Pembaruan lingkungan tetap memastikan bahwa perubahan konfigurasi yang memerlukan penggantian instans diterapkan

secara efisien dan aman. Jika pembaruan lingkungan tetap gagal, proses rollback perlu hanya mengakhiri grup Auto Scaling. Pembaruan bergulir gagal, di sisi lain, memerlukan melakukan pembaruan bergulir tambahan untuk mengembalikan perubahan.

Untuk melakukan pembaruan lingkungan tetap, Elastic Beanstalk membuat grup Auto Scaling kedua sementara di belakang penyeimbang beban lingkungan Anda untuk menampung instans baru. Pertama, Elastic Beanstalk meluncurkan satu instans dengan konfigurasi baru dalam kelompok baru. Instans ini melayani lalu lintas bersama semua instans di grup Auto Scaling asli yang menjalankan konfigurasi sebelumnya.

Ketika instans pertama melewati pemeriksaan kondisi, Elastic Beanstalk meluncurkan instans tambahan dengan konfigurasi baru, cocok dengan jumlah instans yang berjalan di grup Auto Scaling asli. Ketika semua instans baru melewati pemeriksaan kondisi, Elastic Beanstalk mentransfernya ke grup Auto Scaling asli, dan mengakhiri grup Scaling Otomatis sementara dan instans lama.

> **Note**
>
> Selama pembaruan lingkungan tetap, kapasitas lingkungan Anda menjadi ganda untuk waktu yang singkat ketika instans di grup Auto Scaling baru mulai melayani permintaan dan sebelum instans di grup Auto Scaling asli diakhiri. Jika lingkungan Anda memiliki banyak instans, atau Anda memiliki kuota instans sesuai permintaan yang rendah, pastikan bahwa Anda memiliki kapasitas yang cukup untuk melakukan pembaruan lingkungan yang tidak berubah. Jika Anda mendekati kuota, pertimbangkan untuk menggunakan pembaruan bergulir sebagai gantinya.

Pembaruan yang tidak berubah memerlukan laporan kondisi yang ditingkatkan untuk mengevaluasi kondisi lingkungan Anda selama pembaruan berlangsung. Pelaporan kondisi yang ditingkatkan menggabungkan pemeriksaan kondisi load balancer standar dengan pemantauan instans untuk memastikan bahwa instans yang menjalankan konfigurasi baru melayani permintaan dengan sukses.

Anda juga dapat menggunakan pembaruan tetap untuk men-deploy versi baru dari aplikasi Anda, sebagai alternatif untuk penyebaran bergulir. Saat Anda mengonfigurasi Elastic Beanstalk untuk menggunakan pembaruan tetap untuk deployment aplikasi, Elastic Beanstalk mengganti semua instans di lingkungan Anda setiap kali Anda men-deploy versi baru dari aplikasi Anda. Jika deployment aplikasi tetap gagal, Elastic Beanstalk dengan segera mengembalikan perubahan dengan mengakhiri grup Auto Scaling baru. Hal ini dapat mencegah deployment armada parsial, yang dapat terjadi ketika deployment bergulir gagal setelah beberapa batch telah selesai.

Warning

Beberapa kebijakan mengganti semua instans selama deployment atau pembaruan. Hal ini menyebabkan semua akumulasi saldo EC2 burst Amazon hilang. Hal ini terjadi dalam kasus berikut:

- Pembaruan platform terkelola dengan penggantian instans diaktifkan
- Pembaruan tetap
- Deployment dengan pembaruan tetap atau pembagian lalu lintas diaktifkan

Jika pembaruan yang tidak berubah gagal, instans baru mengunggah log paket ke Amazon S3 sebelum Elastic Beanstalk mengakhirinya. Elastic Beanstalk meninggalkan log dari pembaruan tetap yang gagal di Amazon S3 selama satu jam sebelum menghapusnya, alih-alih standar 15 menit untuk paket dan ekor log.

Note

Jika Anda menggunakan pembaruan tidak berubah untuk deployment versi aplikasi, tetapi tidak untuk konfigurasi, Anda mungkin mengalami kesalahan jika Anda mencoba untuk men-deploy versi aplikasi yang berisi perubahan konfigurasi yang biasanya akan memicu pembaruan bergulir (misalnya, konfigurasi yang mengubah tipe instans). Untuk menghindari hal ini, buat perubahan konfigurasi di pembaruan terpisah, atau mengonfigurasi pembaruan tetap untuk deployment dan perubahan konfigurasi.

Anda tidak dapat melakukan pembaruan tetap bersama dengan perubahan konfigurasi sumber daya. Misalnya, Anda tidak dapat mengubah pengaturan yang memerlukan penggantian instans sementara juga memperbarui pengaturan lainnya, atau melakukan deployment tetap dengan file konfigurasi yang mengubah pengaturan konfigurasi atau sumber daya tambahan dalam kode sumber Anda. Jika Anda mencoba untuk mengubah pengaturan sumber daya (misalnya, pengaturan penyeimbang beban) dan secara bersamaan melakukan pembaruan tetap, Elastic Beanstalk mengembalikannya dengan kesalahan.

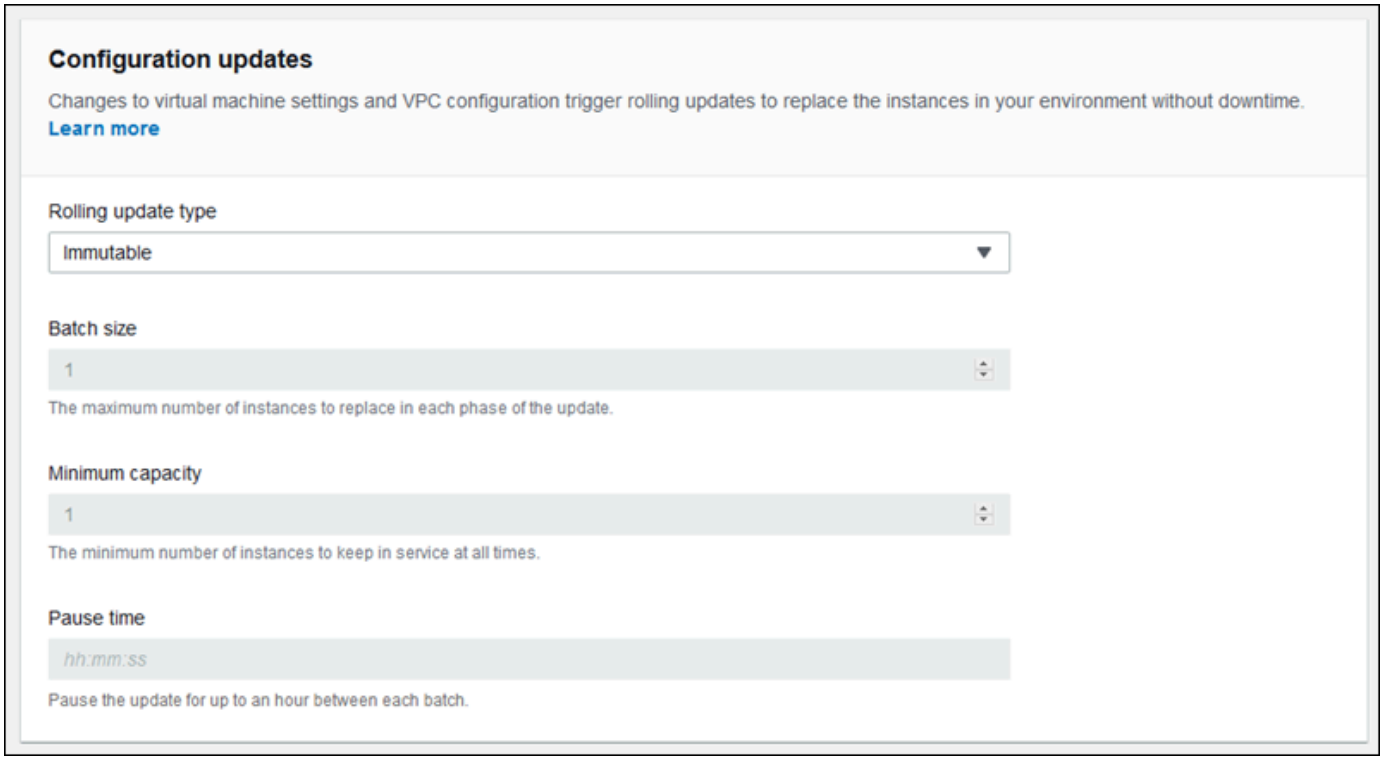
Jika perubahan konfigurasi sumber daya Anda tidak bergantung pada perubahan kode sumber atau konfigurasi instans, lakukan dalam dua pembaruan. Jika mereka bergantung, lakukan deployment biru/hijau sebagai gantinya.

Mengonfigurasi pembaruan tetap

Anda dapat mengaktifkan dan mengonfigurasi pembaruan tetap di konsol Elastic Beanstalk.

Untuk mengaktifkan pembaruan tetap (konsol)

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Di panel navigasi, pilih Konfigurasi.
4. Di kategori konfigurasi Pembaruan dan deployment bergulir, pilih Edit.
5. Di bagian Pembaruan konfigurasi, atur Jenis pembaruan bergulir menjadi Tetap.



6. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Namespace aws:autoscaling:updatepolicy:rollingupdate

Anda juga dapat menggunakan pilihan di namespace `aws:autoscaling:updatepolicy:rollingupdate` untuk mengonfigurasi pembaruan tetap. Contoh berikut file konfigurasi mengaktifkan pembaruan tetap untuk perubahan konfigurasi.

Example .ebextensions/immutable-updates.config

```
option_settings:
  aws:autoscaling:updatepolicy:rollingupdate:
    RollingUpdateType: Immutable
```

Contoh berikut mengaktifkan pembaruan tetap untuk perubahan konfigurasi dan deployment.

Example .ebextensions/immutable-all.config

```
option_settings:
  aws:autoscaling:updatepolicy:rollingupdate:
    RollingUpdateType: Immutable
  aws:elasticbeanstalk:command:
    DeploymentPolicy: Immutable
```

Konsol EB CLI dan Elastic Beanstalk menerapkan nilai yang direkomendasikan untuk pilihan sebelumnya. Anda harus menghapus pengaturan ini jika Anda ingin menggunakan file konfigurasi untuk mengonfigurasi hal yang sama. Lihat Nilai yang disarankan untuk rincian selengkapnya.

Memperbarui versi platform lingkungan Elastic Beanstalk Anda

Elastic Beanstalk secara teratur merilis versi platform baru untuk memperbarui semua platform berbasis Linux dan berbasis Server Windows. Versi platform baru menyediakan pembaruan untuk komponen perangkat lunak yang ada dan dukungan untuk fitur baru dan pilihan konfigurasi. Untuk mempelajari tentang platform dan versi platform, lihat Glosarium Platform Elastic Beanstalk.

Anda dapat menggunakan konsol Elastic Beanstalk atau EB CLI untuk memperbarui versi platform lingkungan Anda. Tergantung pada versi platform yang ingin Anda perbarui, Elastic Beanstalk merekomendasikan salah satu dari dua metode untuk melakukan pembaruan platform.

- Metode 1 – Perbarui versi platform lingkungan Anda. Kami merekomendasikan metode ini ketika Anda memperbarui ke versi platform terbaru dalam cabang platform—dengan waktu pengoperasian, server web, server aplikasi, dan sistem operasi yang sama, dan tanpa perubahan dalam versi platform utama. Ini adalah pembaruan platform yang paling umum dan rutin.
- Metode 2 - Lakukan Blue/Green penyebaran. Kami merekomendasikan metode ini ketika Anda memperbarui ke versi platform terbaru dalam cabang platform—dengan waktu pengoperasian, server web, server aplikasi, dan sistem operasi yang berbeda, atau ke versi platform utama yang

berbeda. Hal ini merupakan pendekatan yang baik ketika Anda ingin mengambil keuntungan dari kemampuan waktu pengoperasian yang baru atau fungsionalitas Elastic Beanstalk terbaru, atau ketika Anda ingin memindahkan cabang platform yang telah usang atau pensiun.

Migrasi dari versi platform lama memerlukan blue/green penerapan, karena versi platform ini tidak kompatibel dengan versi yang didukung saat ini.

Migrasi aplikasi Linux ke Amazon Linux 2 memerlukan blue/green penerapan, karena versi platform Amazon Linux 2 tidak kompatibel dengan versi platform AMI Amazon Linux sebelumnya.

Untuk bantuan lebih lanjut dengan pemilihan metode pembaruan platform terbaik, perluas bagian untuk platform lingkungan Anda.

Docker

Gunakan Metode 1 untuk melakukan pembaruan platform.

Docker Multikontainer

Gunakan Metode 1 untuk melakukan pembaruan platform.

Docker yang Telah Dikonfigurasi

Pertimbangkan kasus berikut:

- Jika Anda memigrasi aplikasi ke platform lain, misalnya dari Go 1.4 (Docker) ke Go 1.11 atau dari Python 3.4 (Docker) ke Python 3.6, gunakan Metode 2.
- Jika Anda memigrasi aplikasi ke versi kontainer Docker yang berbeda, misalnya dari Glassfish 4.1 (Docker) ke Glassfish 5.0 (Docker), gunakan Metode 2.
- Jika Anda memperbarui ke versi platform terbaru tanpa perubahan dalam versi kontainer atau versi utama, gunakan Metode 1.

Go

Gunakan Metode 1 untuk melakukan pembaruan platform.

Java SE

Pertimbangkan kasus berikut:

- Jika Anda memigrasi aplikasi Anda ke versi waktu pengoperasian Java yang berbeda, misalnya dari Java 7 ke Java 8, gunakan Metode 2.
- Jika Anda memperbarui ke versi platform terbaru tanpa perubahan dalam versi waktu pengoperasian, gunakan Metode 1.

Java dengan Tomcat

Pertimbangkan kasus berikut:

- Jika Anda memigrasi aplikasi Anda ke versi waktu pengoperasian Java yang berbeda atau versi server aplikasi Tomcat, misalnya dari Java 7 dengan Tomcat 7 ke Java 8 dengan Tomcat 8.5, gunakan Metode 2.
- Jika Anda memigrasi aplikasi Anda di versi platform Java dengan Tomcat utama (v1.x.x, v2.x.x, dan v3.x.x), gunakan Metode 2.
- Jika Anda memperbarui ke versi platform terbaru tanpa perubahan dalam versi waktu pengoperasian, versi server aplikasi, atau versi utama, gunakan Metode 1.

.NET pada server Windows dengan IIS

Pertimbangkan kasus berikut:

- Jika Anda memigrasi aplikasi Anda ke versi sistem operasi Windows yang berbeda, misalnya dari Windows Server 2008 R2 ke Server Windows 2016, gunakan Metode 2.
- Jika Anda memigrasi aplikasi Anda di seluruh versi platform Server Windows utama, lihat Migrasi dari versi utama sebelumnya dari platform server Windows, dan gunakan Metode 2.
- Jika aplikasi Anda saat ini berjalan pada platform Server Windows V2.x.x dan Anda memperbarui ke versi platform terbaru, gunakan Metode 1.

> **Note**
>
> Versi platform Server Windows lebih awal dari v2 tidak secara semantis berversi. Anda hanya dapat meluncurkan versi terbaru dari masing-masing Server Windows versi platform utama dan tidak dapat mengembalikannya setelah upgrade.

Node.js

Gunakan Metode 2 untuk melakukan pembaruan platform.

PHP

Pertimbangkan kasus berikut:

- Jika Anda memigrasi aplikasi Anda ke versi waktu pengoperasian PHP yang berbeda, misalnya dari PHP 5.6 ke PHP 7.2, gunakan Metode 2.
- Jika Anda memigrasi aplikasi Anda di versi platform PHP utama (v1.x.x and v2.x.x), gunakan Metode 2.
- Jika Anda memperbarui ke versi platform terbaru tanpa perubahan dalam versi waktu pengoperasian atau versi utama, gunakan Metode 1.

Python

Pertimbangkan kasus berikut:

- Jika Anda memigrasi aplikasi Anda ke versi waktu pengoperasian Python yang berbeda, misalnya dari Python 2.7 ke Python 3.6, gunakan Metode 2.
- Jika Anda memigrasi aplikasi Anda di versi platform Python utama (v1.x.x and v2.x.x), gunakan Metode 2.
- Jika Anda memperbarui ke versi platform terbaru tanpa perubahan dalam versi waktu pengoperasian atau versi utama, gunakan Metode 1.

Ruby

Pertimbangkan kasus berikut:

- Jika Anda memigrasi aplikasi Anda ke versi waktu pengoperasian Ruby yang berbeda atau versi server aplikasi, misalnya dari Ruby 2.3 dengan Puma ke Ruby 2.6 dengan Puma, gunakan Metode 2.
- Jika Anda memigrasi aplikasi Anda di versi platform Ruby utama (v1.x.x and v2.x.x), gunakan Metode 2.
- Jika Anda memperbarui ke versi platform terbaru tanpa perubahan dalam versi waktu pengoperasian, versi server aplikasi, atau versi utama, gunakan Metode 1.

Metode 1 – Perbarui versi platform lingkungan Anda

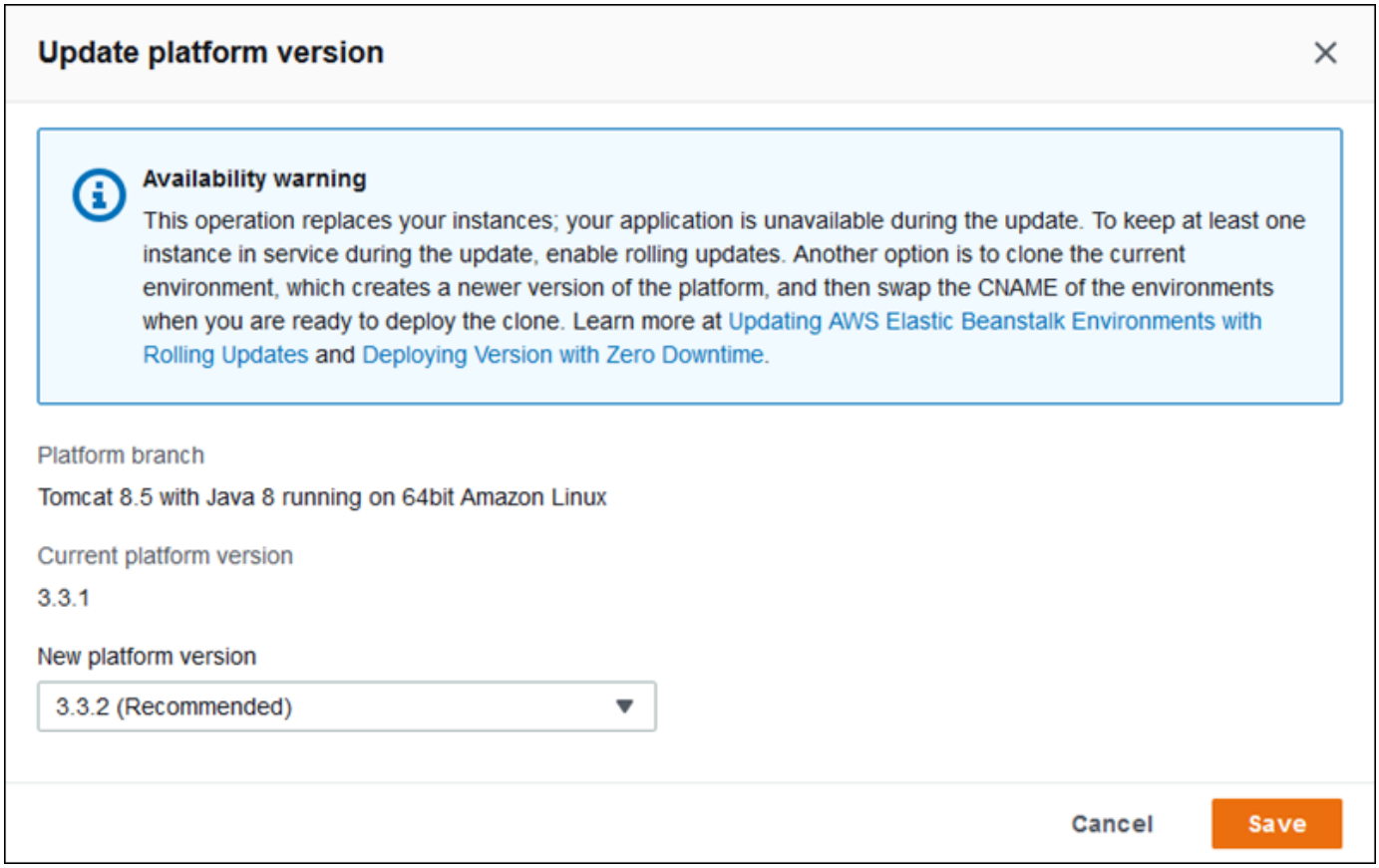
Gunakan metode ini untuk memperbarui ke versi terbaru dari cabang platform lingkungan Anda. Jika sebelumnya Anda telah membuat lingkungan menggunakan versi platform yang lebih lama, atau meningkatkan lingkungan Anda dari versi lama, Anda juga dapat menggunakan metode ini untuk kembali ke versi platform sebelumnya, asalkan berada di cabang platform yang sama.

Untuk memperbarui versi platform lingkungan Anda

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada halaman gambaran umum lingkungan, di bawah Platform, pilih Ubah.



4. Dalam dialog Perbarui versi platform, pilih versi platform. Versi platform terbaru (disarankan) di cabang dipilih secara otomatis. Anda dapat memperbarui ke versi apa pun yang telah Anda gunakan sebelumnya.



5. Pilih Simpan.

Untuk lebih menyederhanakan pembaruan platform, Elastic Beanstalk dapat mengelolanya untuk Anda. Anda dapat mengonfigurasi lingkungan Anda untuk menerapkan pembaruan versi minor dan patch secara otomatis selama jendela pemeliharaan mingguan dikonfigurasi. Elastic Beanstalk menerapkan pembaruan terkelola tanpa waktu henti atau pengurangan kapasitas, dan membatalkan pembaruan segera jika instans yang menjalankan aplikasi Anda pada versi baru gagal dalam pemeriksaan kondisi. Lihat perinciannya di Pembaruan platform yang dikelola.

Metode 2 - Lakukan Blue/Green penyebaran

Gunakan metode ini untuk memperbarui ke cabang platform yang berbeda—dengan waktu pengoperasian, server web, server aplikasi, atau sistem operasi yang berbeda, atau ke versi platform utama yang berbeda. Hal ini biasanya diperlukan ketika Anda ingin mengambil keuntungan dari kemampuan waktu pengoperasian yang baru atau fungsionalitas Elastic Beanstalk terbaru. Hal ini juga diperlukan ketika Anda memigrasi cabang platform yang sudah usang atau pensiun.

Ketika Anda memigrasi di seluruh versi platform utama atau ke versi platform dengan pembaruan komponen utama, ada kemungkinan besar bahwa aplikasi Anda, atau beberapa aspeknya, mungkin

tidak berfungsi seperti yang diharapkan pada versi platform baru, dan mungkin memerlukan perubahan.

Sebelum melakukan migrasi, perbarui mesin pengembangan lokal Anda ke versi waktu pengoperasian yang lebih baru dan komponen lain dari platform yang Anda rencanakan untuk dimigrasi. Verifikasi bahwa aplikasi Anda masih bekerja seperti yang diharapkan, dan buat perbaikan kode dan perubahan yang diperlukan. Kemudian gunakan prosedur praktik terbaik berikut untuk memindahkan lingkungan Anda dengan aman ke versi platform baru.

Untuk memigrasikan lingkungan Anda ke versi platform dengan pembaruan utama

1. Buat lingkungan baru, menggunakan versi platform target yang baru, dan deploy kode aplikasi Anda ke versi platform tersebut. Lingkungan baru harus berada dalam aplikasi Elastic Beanstalk yang berisi lingkungan yang Anda migrasikan. Jangan dulu mengakhiri lingkungan yang ada.
2. Gunakan lingkungan yang baru untuk memigrasi aplikasi Anda. Khususnya:

   - Temukan dan perbaiki masalah kompatibilitas aplikasi yang tidak dapat Anda temukan selama tahap pengembangan.
   - Pastikan bahwa setiap penyesuaian yang dibuat aplikasi Anda menggunakan file konfigurasi bekerja dengan benar di lingkungan baru. Ini mungkin termasuk pilihan pengaturan, paket tambahan yang diinstal, kebijakan keamanan kustom, dan file skrip atau konfigurasi yang diinstal pada instans lingkungan.
   - Jika aplikasi Anda menggunakan Amazon Machine Image (AMI) khusus, buat AMI kustom yang baru berdasarkan AMI versi platform yang baru. Untuk mempelajari informasi lebih lanjut, lihat Menggunakan image mesin Amazon kustom (AMI) di lingkungan Elastic Beanstalk Anda. Khususnya, ini diperlukan jika aplikasi Anda menggunakan platform Server Windows dengan AMI kustom, dan Anda bermigrasi ke versi platform Server Windows V2. Dalam hal ini, lihat juga Migrasi dari versi utama sebelumnya dari platform server Windows.

   Iterasikan pengujian dan men-deploy perbaikan Anda sampai Anda puas dengan aplikasi di lingkungan yang baru.
3. Ubah lingkungan baru ke lingkungan produksi Anda dengan mengubah CNAME-nya dengan CNAME lingkungan produksi yang ada. Untuk detail selengkapnya, lihat Deployment Biru/Hijau dengan Elastic Beanstalk.
4. Ketika Anda puas dengan keadaan lingkungan baru dalam produksi Anda, akhiri lingkungan lama. Lihat perinciannya di Mengakhiri lingkungan Elastic Beanstalk.

Pembaruan platform yang dikelola

AWS Elastic Beanstalk secara teratur merilis pembaruan platform untuk memberikan perbaikan, pembaruan perangkat lunak, dan fitur baru. Dengan pembaruan platform terkelola, Anda dapat mengonfigurasi lingkungan Anda untuk secara otomatis memperbarui ke versi terbaru dari platform selama jendela pemeliharaan yang dijadwalkan. Aplikasi Anda tetap beroperasi selama proses pembaruan tanpa pengurangan kapasitas. Pembaruan terkelola tersedia di lingkungan instans tunggal dan beban yang seimbang.

> **Note**
>
> Fitur ini tidak tersedia di versi platform Server Windows yang lebih awal dari versi 2 (v2).

Anda dapat mengonfigurasi lingkungan Anda untuk secara otomatis menerapkan pembaruan versi patch, atau pembaruan versi patch dan minor. Pembaruan platform terkelola tidak mendukung pembaruan di seluruh cabang platform (pembaruan ke versi utama yang berbeda dari komponen platform seperti sistem operasi, waktu aktif pengoperasian, atau komponen Elastic Beanstalk), karena ini dapat memperkenalkan perubahan yang tidak kompatibel ke belakang.

Anda juga dapat mengonfigurasi Elastic Beanstalk untuk menggantikan semua instans di lingkungan Anda selama jendela pemeliharaan, bahkan jika pembaruan platform tidak tersedia. Mengganti semua instans di lingkungan Anda sangat membantu jika aplikasi Anda menemukan bug atau masalah memori saat berjalan dalam waktu lama.

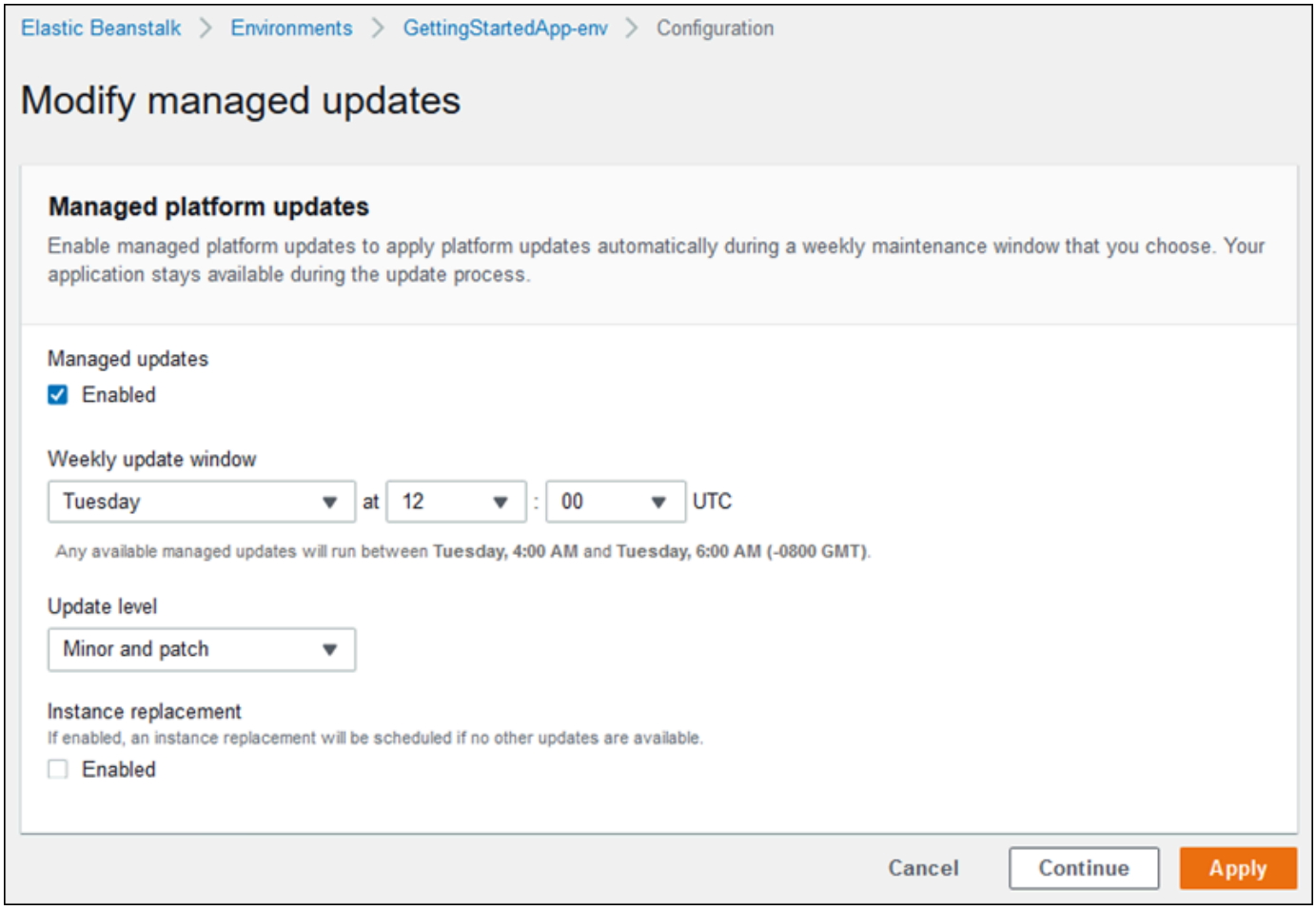
Pada lingkungan yang dibuat pada 25 November 2019 atau yang lebih baru menggunakan konsol Elastic Beanstalk, pembaruan terkelola diaktifkan secara default bila memungkinkan. Pembaruan terkelola memerlukan kondisi yang ditingkatkan untuk diaktifkan. Kondisi yang ditingkatkan diaktifkan secara default ketika Anda memilih salah satu prasetel konfigurasi, dan dinonaktifkan saat Anda memilih Konfigurasi kustom. Konsol tidak dapat mengaktifkan pembaruan terkelola untuk versi platform lama yang tidak mendukung kondisi yang ditingkatkan, atau ketika kondisi yang ditingkatkan dinonaktifkan. Ketika konsol memungkinkan pembaruan terkelola untuk lingkungan baru, Jendela pembaruan mingguan diatur ke hari acak dalam seminggu secara acak. Tingkat pembaruan diatur untuk Minor dan patch, dan Pengganti instans dinonaktifkan. Anda dapat menonaktifkan atau mengonfigurasi ulang pembaruan terkelola sebelum langkah terakhir pembuatan lingkungan.

Untuk lingkungan yang ada, gunakan konsol Elastic Beanstalk kapan saja untuk mengonfigurasi pembaruan platform terkelola.

> **Important**
>
> Sejumlah besar lingkungan Beanstalk dalam satu AWS akun dapat menimbulkan risiko masalah pelambatan selama pembaruan terkelola. Angka tinggi adalah jumlah relatif yang bergantung pada seberapa dekat Anda menjadwalkan pembaruan terkelola untuk lingkungan Anda. Lebih dari 200 lingkungan dalam satu akun yang dijadwalkan secara ketat dapat menyebabkan masalah pelambatan, meskipun angka yang lebih rendah mungkin juga bermasalah.
>
> Untuk menyeimbangkan beban sumber daya untuk pembaruan terkelola, kami menyarankan Anda menyebarkan jendela pemeliharaan terjadwal untuk lingkungan dalam satu akun. Juga, pertimbangkan strategi multi-akun. Untuk informasi selengkapnya, lihat Mengatur AWS Lingkungan Anda Menggunakan Beberapa Akun di situs web AWS Whitepaper & Guides.

Untuk mengonfigurasi pembaruan platform terkelola

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Di panel navigasi, pilih Konfigurasi.
4. Di kategori Pembaruan yang dikelola, pilih Edit.
5. Nonaktifkan atau aktifkan Pembaruan yang dikelola.
6. Jika pembaruan terkelola diaktifkan, pilih jendela pemeliharaan, dan kemudian pilih Tingkat pembaruan.
7. (Opsional) Pilih Pengganti instans untuk mengaktifkan penggantian instans mingguan.



8. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Pembaruan platform terkelola bergantung pada pelaporan kondisi yang ditingkatkan untuk menentukan bahwa aplikasi Anda cukup sehat untuk mempertimbangkan keberhasilan pembaruan platform. Lihat Mengaktifkan pelaporan kondisi yang ditingkatkan Elastic Beanstalk untuk instruksi.

Bagian

- Izin yang diperlukan untuk melakukan pembaruan platform terkelola
- Jendela pemeliharaan pembaruan terkelola
- Pembaruan versi minor dan patch
- Pembaruan lingkungan yang tidak berubah
- Mengelola pembaruan terkelola
- Namespace pilihan tindakan terkelola

Izin yang diperlukan untuk melakukan pembaruan platform terkelola

Elastic Beanstalk membutuhkan izin untuk memulai pembaruan platform atas nama Anda. Untuk mendapatkan izin ini, Elastic Beanstalk mengasumsikan peran layanan pembaruan yang dikelola. Bila Anda menggunakan pengaturan peran layanan default untuk lingkungan Anda, konsol Elastic Beanstalk menggunakannya sebagai peran layanan pembaruan terkelola juga. Konsol menetapkan kebijakan terkelola `AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy` untuk peran layanan Anda. Kebijakan ini memiliki semua izin yang Elastic Beanstalk perlukan melakukan pembaruan platform terkelola.

Untuk rincian tentang cara lain untuk menetapkan peran layanan pembaruan terkelola, lihat the section called “Peran layanan”.

> **Note**
>
> Jika Anda menggunakan file konfigurasi untuk memperluas lingkungan Anda untuk menyertakan sumber daya tambahan, Anda mungkin perlu menambahkan izin ke peran layanan pembaruan terkelola lingkungan Anda. Biasanya Anda perlu menambahkan izin ketika Anda merujuk sumber daya ini dengan nama di bagian atau file lain.

Jika pembaruan gagal, Anda dapat menemukan alasan dari kegagalan pada halaman Pembaruan terkelola.

Jendela pemeliharaan pembaruan terkelola

Saat AWS merilis versi baru platform lingkungan Anda, Elastic Beanstalk menjadwalkan pembaruan platform terkelola selama jendela pemeliharaan mingguan berikutnya. Windows pemeliharaan terjadi selama dua jam. Elastic Beanstalk memulai pembaruan yang dijadwalkan selama jendela pemeliharaan. Pembaruan mungkin tidak selesai sampai setelah jendela berakhir.

> **Note**
>
> Dalam kebanyakan kasus, Elastic Beanstalk menjadwalkan pembaruan terkelola Anda untuk terjadi selama jendela pemeliharaan mingguan mendatang. Sistem mempertimbangkan berbagai aspek keamanan pembaruan dan ketersediaan layanan saat menjadwalkan pembaruan terkelola. Dalam kasus yang jarang terjadi, pembaruan mungkin tidak dijadwalkan untuk jendela pemeliharaan yang datang pertama. Jika ini terjadi, sistem mencoba lagi pada jendela pemeliharaan berikutnya. Untuk secara manual menerapkan

pembaruan terkelola, pilih Terapkan sekarang seperti yang dijelaskan dalam Mengelola pembaruan terkelola di halaman ini.

Pembaruan versi minor dan patch

Anda dapat mengaktifkan pembaruan platform terkelola untuk menerapkan pembaruan versi patch saja, atau untuk pembaruan versi minor dan patch. Pembaruan versi patch menyediakan perbaikan bug dan peningkatan kinerja, dan dapat mencakup perubahan konfigurasi minor pada perangkat lunak, skrip, dan pilihan konfigurasi pada instans. Pembaruan versi minor memberikan dukungan untuk fitur Elastic Beanstalk yang baru. Anda tidak dapat menerapkan pembaruan versi utama, yang mungkin membuat perubahan yang tidak kompatibel, dengan pembaruan platform terkelola.

Dalam nomor versi platform, nomor kedua adalah versi pembaruan minor, dan nomor ketiga adalah versi patch. Sebagai contoh, versi platform versi 2.0.7 mempunyai versi minor 0 dan versi patch 7.

Pembaruan lingkungan yang tidak berubah

Pembaruan platform terkelola melakukan Pembaruan lingkungan tetap untuk meningkatkan lingkungan Anda ke versi platform yang baru. Pembaruan tetap memperbarui lingkungan Anda tanpa mengambil instans apa pun dari layanan atau memodifikasi lingkungan Anda, sebelum mengonfirmasi bahwa instans yang menjalankan versi baru lolos dalam pemeriksaan kondisi.

Dalam pembaruan tetap, Elastic Beanstalk men-deploy instans sebanyak seperti yang saat ini berjalan dengan versi platform yang baru. Instans baru mulai mengambil permintaan bersama instans yang menjalankan versi lama. Jika rangkaian instans baru melewati semua pemeriksaan kondisi, Elastic Beanstalk mengakhiri rangkaian instans lama, hanya menyisakan instans dengan versi baru.

Pembaruan platform terkelola selalu melakukan pembaruan yang tidak berubah, bahkan ketika Anda menerapkannya di luar jendela pemeliharaan. Jika Anda mengubah versi platform dari Dasbor, Elastic Beanstalk menerapkan kebijakan pembaruan yang telah Anda pilih untuk pembaruan konfigurasi.

⚠ Warning

Beberapa kebijakan mengganti semua instans selama deployment atau pembaruan. Hal ini menyebabkan semua akumulasi saldo EC2 burst Amazon hilang. Hal ini terjadi dalam kasus berikut:

- Pembaruan platform terkelola dengan penggantian instans diaktifkan

- Pembaruan tetap
- Deployment dengan pembaruan tetap atau pembagian lalu lintas diaktifkan

Mengelola pembaruan terkelola

Konsol Elastic Beanstalk menunjukkan informasi rinci tentang pembaruan terkelola pada halaman Gambaran umum pembaruan terkelola.

Untuk melihat informasi tentang pembaruan terkelola (konsol)

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pilih Pembaruan terkelola.

Bagian Gambaran umum pembaruan terkelola menyediakan informasi tentang pembaruan terkelola yang terjadwal dan tertunda. Bagian Riwayat membuat daftar pembaruan yang berhasil dan upaya yang gagal.

Anda dapat memilih untuk menerapkan pembaruan terjadwal segera, alih-alih menunggu hingga jendela pemeliharaan.

Untuk menerapkan pembaruan platform terkelola dengan segera (konsol)

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pilih Pembaruan terkelola.
4. Pilih Terapkan sekarang.
5. Verifikasi rincian pembaruan, dan kemudian pilih Terapkan.

Ketika Anda menerapkan pembaruan platform terkelola di luar jendela pemeliharaan, Elastic Beanstalk melakukan pembaruan tetap. Jika Anda memperbarui platform lingkungan dari Dasbor, atau dengan menggunakan klien yang berbeda, Elastic Beanstalk menggunakan jenis pembaruan yang Anda pilih untuk perubahan konfigurasi.

Jika Anda tidak memiliki jadwal pembaruan terkelola, lingkungan Anda mungkin sudah menjalankan versi terbaru. Alasan lain tidak memiliki pembaruan yang dijadwalkan meliputi:

- Pembaruan versi minor tersedia, tetapi lingkungan Anda dikonfigurasi untuk secara otomatis menerapkan hanya pembaruan versi patch.
- Lingkungan Anda belum dipindai sejak pembaruan dirilis. Elastic Beanstalk biasanya memeriksa pembaruan setiap jam.
- Pembaruan tertunda atau sudah berlangsung.

Saat jendela pemeliharaan dimulai atau saat Anda memilih Terapkan sekarang, pembaruan terjadwal berubah ke status tertunda sebelum eksekusi.

Namespace pilihan tindakan terkelola

Anda dapat menggunakan pilihan konfigurasi di namespace `aws:elasticbeanstalk:managedactions` dan `aws:elasticbeanstalk:managedactions:platformupdate` untuk mengaktifkan dan mengonfigurasi pembaruan platform terkelola.

Pilihan `ManagedActionsEnabled` mengaktifkan pembaruan platform terkelola. Tetapkan pilihan ini ke `true` untuk mengaktifkan pembaruan platform terkelola, dan gunakan pilihan lain untuk mengonfigurasi perilaku pembaruan.

Gunakan `PreferredStartTime` untuk mengonfigurasi awal jendela pemeliharaan mingguan dalam *`minute`* format *`dayhour`*::.

Tetapkan `UpdateLevel` ke `minor` atau `patch` untuk menerapkan pembaruan versi minor dan patch, atau hanya pembaruan versi patch, masing-masing.

Ketika pembaruan platform terkelola diaktifkan, Anda dapat mengaktifkan penggantian instans dengan menetapkan pilihan `InstanceRefreshEnabled` untuk `true`. Ketika pengaturan ini diaktifkan, Elastic Beanstalk menjalankan pembaruan yang tidak berubah pada lingkungan Anda setiap minggu, terlepas dari apakah ada versi platform baru yang tersedia.

Contoh berikut file konfigurasi memungkinkan pembaruan platform terkelola untuk pembaruan versi patch dengan jendela pemeliharaan mulai pukul 9:00 AM UTC setiap hari Selasa.

Example .ebextensions/ .config managed-platform-update

```
option_settings:
```

```
  aws:elasticbeanstalk:managedactions:
    ManagedActionsEnabled: true
    PreferredStartTime: "Tue:09:00"
  aws:elasticbeanstalk:managedactions:platformupdate:
    UpdateLevel: patch
    InstanceRefreshEnabled: true
```

Memigrasi aplikasi Anda dari versi platform legasi

Jika Anda telah men-deploy aplikasi Elastic Beanstalk yang menggunakan versi platform legasi, Anda harus memigrasi aplikasi Anda ke lingkungan baru menggunakan versi platform non-legasi sehingga Anda bisa mendapatkan akses ke fitur baru. Jika Anda tidak yakin apakah Anda menjalankan aplikasi Anda menggunakan versi platform legasi, Anda dapat memeriksa di konsol Elastic Beanstalk. Untuk instruksi, lihat Untuk memeriksa apakah Anda menggunakan versi platform legasi.

Fitur baru apa saja yang tidak ada di versi platform legasi?

Platform legasi tidak mendukung fitur-fitur berikut:

- File konfigurasi, seperti yang dijelaskan dalam topik Penyesuaian lingkungan lanjutan dengan file konfigurasi (`.ebextensions`)
- Pemeriksaan kondisi ELB, seperti yang dijelaskan dalam topik Pelaporan kondisi dasar
- Profil instans, seperti yang dijelaskan dalam topik Mengelola profil instans Elastic Beanstalk
- VPCs, seperti yang dijelaskan dalam Menggunakan Elastic Beanstalk dengan Amazon VPC topik
- Tingkat Data, seperti yang dijelaskan dalam topik Menambahkan basis data ke lingkungan Elastic Beanstalk Anda
- Tingkat Pekerja, seperti yang dijelaskan dalam topik Lingkungan pekerja Elastic Beanstalk
- Lingkungan Instans Tunggal, seperti yang dijelaskan dalam topik Jenis lingkungan
- Tag, seperti yang dijelaskan dalam topik Menandai sumber daya di lingkungan Elastic Beanstalk Anda
- Pembaruan Bergulir, seperti yang dijelaskan dalam topik Pembaruan konfigurasi lingkungan bergulir Elastic Beanstalk

Mengapa beberapa versi platform ditandai sebagai legasi?

Beberapa versi platform yang lebih lama tidak mendukung fitur Elastic Beanstalk terbaru. Versi ini ditandai (legasi) pada halaman gambaran umum lingkungan di konsol Elastic Beanstalk.

Untuk memeriksa apakah Anda menggunakan versi platform legasi

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada halaman gambaran umum lingkungan, lihat nama Platform.

   Aplikasi Anda menggunakan versi platform legasi jika Anda melihat (legasi) di samping nama platform.

Untuk memigrasi aplikasi Anda

1. Men-deploy aplikasi sampel ke lingkungan Anda. Untuk instruksi, buka Membuat lingkungan Elastic Beanstalk.
2. Jika Anda memiliki Instans Amazon RDS DB, perbarui grup keamanan database Anda untuk mengizinkan akses ke grup EC2 keamanan untuk lingkungan baru Anda. Untuk petunjuk tentang cara menemukan nama grup EC2 keamanan menggunakan AWS Management Console, lihatEC2 kelompok keamanan. Untuk informasi selengkapnya tentang mengonfigurasi grup EC2 keamanan Anda, buka bagian “Mengotorisasi Akses Jaringan ke Grup EC2 Keamanan Amazon” dari Bekerja dengan Grup Keamanan DB di Panduan Pengguna Layanan Amazon Relational Database Service.
3. Ubah URL lingkungan Anda. Untuk instruksi, buka Deployment Biru/Hijau dengan Elastic Beanstalk.
4. Akhiri lingkungan lama Anda. Untuk instruksi, buka Mengakhiri lingkungan Elastic Beanstalk.

> **Note**
>
> Jika Anda menggunakan AWS Identity and Access Management (IAM) maka Anda perlu memperbarui kebijakan Anda untuk menyertakan AWS CloudFormation dan Amazon RDS (jika berlaku). Lihat informasi yang lebih lengkap di Menggunakan Elastic Beanstalk dengan AWS Identity and Access Management.

Memigrasi aplikasi Elastic Beanstalk Linux Anda ke Amazon Linux 2023 atau Amazon Linux 2

Bagian ini menjelaskan cara memigrasi aplikasi Anda menggunakan salah satu jalur migrasi berikut.

- Bermigrasi dari cabang platform Amazon Linux 2 ke cabang platform Amazon Linux 2023.
- Bermigrasi dari cabang platform Amazon Linux AMI (AL1) ke cabang platform Amazon Linux 2023 (disarankan) atau cabang platform Amazon Linux 2.

Topik

- Migrasi dari Amazon Linux 2 ke Amazon Linux 2023
- Migrasi dari Amazon Linux AMI (AL1) ke AL2 atau AL2 023

Migrasi dari Amazon Linux 2 ke Amazon Linux 2023

Topik ini memberikan panduan untuk memigrasikan aplikasi Anda dari cabang platform Amazon Linux 2 ke cabang platform Amazon Linux 2023.

Perbedaan dan kompatibilitas

Antara platform Elastic Beanstalk dan AL2 AL2 023

Ada tingkat kompatibilitas yang tinggi antara platform Elastic Beanstalk Amazon Linux 2 dan Amazon Linux 2023. Meskipun ada beberapa perbedaan yang perlu diperhatikan:

- Layanan Metadata Instans Versi 1 (IMDSv1) - Pengaturan IMDSv1 opsi Nonaktifkan default ke platform 023. `true` AL2 Defaultnya ada `false` di AL2 platform.
- alat instans pkg-repo - Alat ini tidak tersedia untuk lingkungan yang berjalan pada platform 023. pkg-repo AL2 Namun, Anda dapat secara manual menerapkan paket dan pembaruan sistem operasi ke instans AL2 023. Untuk informasi selengkapnya, lihat Mengelola paket dan pembaruan sistem operasi di Panduan Pengguna Amazon Linux 2023.
- HTTPd Konfigurasi Apache - `httpd.conf` File Apache untuk platform AL2 023 memiliki beberapa pengaturan konfigurasi yang berbeda dari yang untuk: AL2
  - Tolak akses ke seluruh sistem file server secara default. Pengaturan ini dijelaskan dalam Lindungi File Server secara Default di halaman Tips Keamanan situs web Apache.

- Hentikan pengguna untuk mengganti fitur keamanan yang telah Anda konfigurasikan. Konfigurasi menolak akses untuk mengatur `.htaccess` di semua direktori, kecuali yang diaktifkan secara khusus. Pengaturan ini dijelaskan dalam Melindungi Pengaturan Sistem di halaman Tips Keamanan situs web Apache. Tutorial Server HTTP Apache: halaman file.htaccess menyatakan pengaturan ini dapat membantu meningkatkan kinerja.
- Tolak akses ke file dengan pola nama `.ht*`. Pengaturan ini mencegah klien web melihat `.htaccess` dan `.htpasswd` file.

Anda dapat mengubah salah satu pengaturan konfigurasi di atas untuk lingkungan Anda. Untuk informasi selengkapnya, lihat Mengonfigurasi Apache HTTPD.

Antara sistem operasi Amazon Linux

Untuk informasi selengkapnya tentang perbedaan antara sistem operasi Amazon Linux 2 dan Amazon Linux 2023, lihat Membandingkan Amazon Linux 2 dan Amazon Linux 2023 di Panduan Pengguna Amazon Linux 2023.

Untuk informasi selengkapnya tentang Amazon Linux 2023, lihat Apa itu Amazon Linux 2023? di Panduan Pengguna Amazon Linux 2023.

Proses migrasi umum

Ketika Anda siap untuk pergi ke produksi, Elastic Beanstalk blue/green memerlukan penerapan untuk melakukan upgrade. Berikut ini adalah langkah-langkah praktik terbaik umum yang kami rekomendasikan untuk migrasi dengan prosedur blue/green penerapan.

Bersiap untuk menguji migrasi Anda

Sebelum Anda menerapkan aplikasi Anda dan mulai menguji, tinjau informasi di bagian Perbedaan dan kompatibilitas sebelumnya. Tinjau juga referensi yang dikutip di bagian itu, Membandingkan Amazon Linux 2 dan Amazon Linux 2023 di Panduan Pengguna Amazon Linux 2023. Catat informasi spesifik dari konten ini yang berlaku atau mungkin berlaku untuk pengaturan aplikasi dan konfigurasi Anda.

Langkah migrasi tingkat tinggi

1. Buat lingkungan baru yang didasarkan pada cabang platform AL2 023.
2. Terapkan aplikasi Anda ke lingkungan target AL2 023.

Lingkungan produksi Anda yang ada akan tetap aktif dan tidak terpengaruh, sementara Anda mengulangi pengujian dan membuat penyesuaian pada lingkungan baru.

3. Uji aplikasi Anda secara menyeluruh di lingkungan baru.
4. Ketika lingkungan AL2 023 tujuan Anda siap untuk pergi ke produksi, tukar CNAMEs dari dua lingkungan untuk mengarahkan lalu lintas ke lingkungan AL2 023 yang baru.

Langkah-langkah migrasi yang lebih rinci dan praktik terbaik

Untuk prosedur blue/green penerapan yang lebih rinci, lihatDeployment Biru/Hijau dengan Elastic Beanstalk.

Untuk panduan yang lebih spesifik dan langkah-langkah praktik terbaik yang terperinci, lihat Metode Biru/Hijau.

Referensi lainnya untuk membantu merencanakan migrasi

Referensi berikut dapat menawarkan informasi tambahan untuk merencanakan migrasi Anda.

- Platform yang didukung Elastic Beanstalk di Platform AWS Elastic Beanstalk
- Sejarah cabang platform pensiunan
- the section called “Platform Linux”
- FAQ Pensiun Platform

Migrasi dari Amazon Linux AMI (AL1) ke AL2 atau AL2 023

Jika aplikasi Elastic Beanstalk Anda didasarkan pada cabang platform AMI Amazon Linux, gunakan bagian ini untuk mempelajari cara memigrasi lingkungan aplikasi Anda ke Amazon Linux 2 atau Amazon Linux 2023. Cabang platform generasi sebelumnya berbasis Amazon Linux AMI sekarang sudah pensiun.

Kami sangat menyarankan Anda bermigrasi ke Amazon Linux 2023, karena ini lebih baru daripada Amazon Linux 2. Sistem operasi Amazon Linux 2 akan mencapai akhir dukungan sebelum Amazon Linux 2023 melakukannya, jadi Anda akan mendapat manfaat dari kerangka waktu dukungan yang lebih lama jika Anda bermigrasi ke Amazon Linux 2023.

Perlu dicatat bahwa ada tingkat kompatibilitas yang tinggi antara platform Elastic Beanstalk Amazon Linux 2 dan Amazon Linux 2023. Meskipun beberapa area memiliki perbedaan: opsi Instance

Metadata Service Version 1 (IMDSv1) default, dukungan untuk alat instans pkg-repo, dan beberapa konfigurasi Apache. HTTPd Untuk informasi selengkapnya, lihat Amazon Linux 2023

Perbedaan dan kompatibilitas

Cabang platform AL2 berbasis AL2 023/ tidak dijamin kompatibel dengan aplikasi Anda yang ada. Penting juga untuk diperhatikan bahwa meskipun kode aplikasi Anda berhasil diterapkan ke versi platform baru, kode tersebut mungkin berperilaku atau berkinerja berbeda karena perbedaan sistem operasi dan waktu berjalan.

Meskipun Amazon Linux AMI dan AL2 023/ AL2 berbagi kernel Linux yang sama, mereka berbeda dalam aspek-aspek berikut: sistem inisialisasi mereka, `libc` versi, rantai alat kompiler, dan berbagai paket. Untuk informasi selengkapnya, lihat Amazon Linux 2 FAQs.

Layanan Elastic Beanstalk juga telah memperbarui versi spesifik platform runtime, alat build, dan dependensi lainnya.

Oleh karena itu kami merekomendasikan Anda untuk meluangkan waktu Anda, menguji aplikasi Anda secara menyeluruh dalam lingkungan pengembangan, dan membuat penyesuaian yang diperlukan.

Proses migrasi umum

Ketika Anda siap untuk pergi ke produksi, Elastic Beanstalk blue/green memerlukan penerapan untuk melakukan upgrade. Berikut ini adalah langkah-langkah praktik terbaik umum yang kami rekomendasikan untuk migrasi dengan prosedur blue/green penerapan.

Bersiap untuk menguji migrasi Anda

Sebelum Anda menerapkan aplikasi Anda dan mulai menguji, tinjau informasi diPertimbangan untuk semua platform Linux, yang mengikuti nanti dalam topik ini. Juga, tinjau informasi yang berlaku untuk platform Anda di Pertimbangan spesifik platform bagian berikut. Catat informasi spesifik dari konten ini yang berlaku atau mungkin berlaku untuk pengaturan aplikasi dan konfigurasi Anda.

Langkah migrasi tingkat tinggi

1. Buat lingkungan baru yang didasarkan pada cabang platform AL2 atau AL2 023. Kami menyarankan Anda bermigrasi ke cabang platform AL2 023.
2. Terapkan aplikasi Anda ke lingkungan AL2 AL2 023/target.

   Lingkungan produksi Anda yang ada akan tetap aktif dan tidak terpengaruh, sementara Anda mengulangi pengujian dan membuat penyesuaian pada lingkungan baru.

3. Uji aplikasi Anda secara menyeluruh di lingkungan baru.
4. Ketika AL2 lingkungan AL2 023/tujuan Anda siap untuk diproduksi, tukar CNAMEs dari dua lingkungan untuk mengarahkan lalu lintas ke lingkungan baru.

Langkah-langkah migrasi yang lebih rinci dan praktik terbaik

Untuk prosedur blue/green penerapan yang lebih rinci, lihatDeployment Biru/Hijau dengan Elastic Beanstalk.

Untuk panduan yang lebih spesifik dan langkah-langkah praktik terbaik yang terperinci, lihat Metode Biru/Hijau.

Referensi lainnya untuk membantu merencanakan migrasi

Referensi berikut dapat menawarkan informasi tambahan untuk merencanakan migrasi Anda.

- Membandingkan Panduan Pengguna Amazon Linux 2 dan Amazon Linux 2023 Amazon Linux 2023.
- Apa itu Amazon Linux 2023? di Panduan Pengguna Amazon Linux 2023
- Platform yang didukung Elastic Beanstalk di Platform AWS Elastic Beanstalk
- Sejarah cabang platform pensiunan
- the section called “Platform Linux”
- FAQ Pensiun Platform

Pertimbangan untuk semua platform Linux

Tabel berikut membahas pertimbangan yang harus Anda ketahui saat merencanakan migrasi aplikasi ke AL2 023/. AL2 Pertimbangan ini berlaku untuk setiap platform Linux Elastic Beanstalk, terlepas dari bahasa pemrograman atau server aplikasi tertentu.

Luas	Perubahan dan informasi
File konfigurasi	Pada AL2 023/ AL2 platform, Anda dapat menggunakan file konfigurasi seperti sebelumnya, dan semua bagian bekerja dengan cara yang sama. Namun, pengaturan tertentu mungkin tidak bekerja sama seperti yang mereka lakukan pada platform Amazon Linux AMI sebelumnya. Misalnya:

Luas	Perubahan dan informasi
	• Beberapa paket perangkat lunak yang Anda instal menggunakan file konfigurasi mungkin tidak tersedia pada AL2 023/AL2, atau namanya mungkin telah berubah. • Beberapa pilihan konfigurasi platform tertentu telah berpindah dari namespace platform tertentu mereka ke namespace platform agnostik yang berbeda. • File konfigurasi proksi yang disediakan di direktori `.ebextensions/nginx` harus dipindahkan ke direktori kaitan platform `.platform/nginx` . Lihat perinciannya di Konfigurasi proksi terbalik. Kami merekomendasikan untuk menggunakan kaitan platform untuk menjalankan kode khusus pada instans lingkungan Anda. Anda masih dapat menggunakan perintah dan perintah kontainer di file konfigurasi `.ebextensions` , tetapi tidak semudah itu untuk dapat bekerjasama. Misalnya, menulis skrip perintah di dalam file YAML dapat merepotkkan dan sulit untuk diuji. Anda masih perlu menggunakan file konfigurasi `.ebextensions` untuk setiap skrip yang membutuhkan referensi ke sumber daya AWS CloudFormation .
Kaitan platform	AL2 platform memperkenalkan cara baru untuk memperluas platform lingkungan Anda dengan menambahkan file yang dapat dieksekusi untuk menghubungkan direktori pada instance lingkungan. Dengan versi platform Linux sebelumnya, Anda mungkin telah menggunakan kaitan platform khusus. Kaitan ini tidak dirancang untuk platform terkelola dan tidak didukung, tetapi dapat bekerja dengan cara yang berguna dalam beberapa kasus. Dengan versi AL2 AL2 023/platform, kait platform khusus tidak berfungsi. Anda harus memigrasikan kaitan apa pun ke kaitan platform yang baru. Untuk detailnya lihatHook platform.

Luas	Perubahan dan informasi
Server proksi yang didukung	AL2Versi 023/ AL2 platform mendukung server proxy terbalik yang sama dengan setiap platform yang didukung dalam versi platform Amazon Linux AMI. Semua AL2 023/AL2; versi platform menggunakan nginx sebagai server proxy terbalik default mereka, dengan pengecualian platform ECS dan Docker. Platform Tomcat, Node.js, PHP, dan Python juga mendukung Apache HTTPD sebagai alternatif. Semua platform mengaktifkan konfigurasi server proksi dengan cara yang seragam, seperti yang dijelaskan di bagian ini. Namun, konfigurasi server proksi sedikit berbeda dari pada Amazon Linux AMI. Ini adalah perbedaan untuk semua platform: • Default adalah nginx — Server proxy default pada AL2 semua versi AL2 023/platform adalah nginx. Pada platform versi Amazon Linux AMI dari Tomcat, PHP, dan Python, server proksi default adalah Apache HTTPD. • Namespace yang konsisten — Semua versi AL2 023/ AL2 platform menggunakan `aws:elasticbeanstalk:environment:proxy` namespace untuk mengkonfigurasi server proxy. Pada versi platform Amazon Linux AMI hal ini merupakan keputusan per-platform, dan Node.js menggunakan namespace yang berbeda. • Lokasi file konfigurasi — Anda harus menempatkan file konfigurasi proxy di `.platform/httpd` direktori `.platform/nginx` dan pada semua versi AL2 023/ platformAL2 . Pada versi platform Amazon Linux AMI lokasi ini adalah `.ebextensions/nginx` dan `.ebextensions/httpd` , masing-masing. Untuk perubahan konfigurasi proksi spesifik platform, lihat the section called “Pertimbangan spesifik platform”. Untuk informasi tentang konfigurasi proxy pada AL2 023/ AL2 platform, lihat. Konfigurasi proksi terbalik
Perubahan Konfigurasi Proxy	Ada perubahan konfigurasi proxy yang berlaku seragam untuk semua platform selain perubahan konfigurasi proxy yang spesifik untuk setiap platform. Penting untuk merujuk keduanya untuk mengonfigurasi lingkungan Anda secara akurat. • Semua platform — lihat Konfigurasi proksi terbalik • Khusus platform — lihat. the section called “Pertimbangan spesifik platform”

Luas	Perubahan dan informasi
Profil instans	AL2AL2 023/platform memerlukan profil instance untuk dikonfigurasi. Pembuatan lingkungan mungkin berhasil sementara tanpa satu, tetapi lingkungan mungkin menunjukkan kesalahan segera setelah pembuatan ketika tindakan yang memerlukan profil instans mulai gagal. Untuk detail selengkapnya, lihat the section called “Profil instans”.
Kondisi yang ditingkatkan	AL2023/ versi AL2 platform memungkinkan peningkatan kesehatan secara default. Ini adalah perubahan jika Anda tidak menggunakan konsol Elastic Beanstalk untuk membuat lingkungan Anda. Konsol mengaktifkan kondisi yang ditingkatkan secara default bila memungkinkan, terlepas dari versi platform. Untuk detail selengkapnya, lihat the section called “Pelaporan dan pemantauan kondisi yang ditingkatkan”.
AMI khusus	Jika lingkungan Anda menggunakan AMI kustom, buat AMI baru berdasarkan AL2 023/ AL2 untuk lingkungan baru Anda menggunakan platform Elastic Beanstalk 023/. AL2 AL2
Platform kustom	Versi AL2 023/ AL2 platform yang dikelola AMIs tidak mendukung platform khusus.

Pertimbangan spesifik platform

Bagian ini membahas pertimbangan migrasi khusus untuk platform Elastic Beanstalk Linux tertentu.

Docker

Keluarga cabang platform Docker berbasis Amazon Linux AMI (AL1) mencakup tiga cabang platform. Kami merekomendasikan jalur migrasi yang berbeda untuk masing-masing.

AL1 Cabang platform	Jalur Migrasi ke AL2 023/ AL2
Docker multi-kontainer yang dikelola	Cabang platform Docker AL2 AL2 023/berbasis ECS

AL1 Cabang platform	Jalur Migrasi ke AL2 023/ AL2
oleh Amazon ECS yang berjalan di Amazon Linux AMI () AL1	Cabang platform Docker AL2 023/ berbasis ECS menawarkan jalur migrasi langsung untuk lingkungan yang berjalan di cabang AL2 platform Multi-container Docker. AL1 • Seperti cabang Multi-container Docker sebelumnya, AL1 cabang platform AL2 023/ AL2 menggunakan Amazon ECS untuk mengoordinasikan penyebara n beberapa kontainer Docker ke cluster Amazon ECS di lingkungan Elastic Beanstalk. • Cabang AL2 platform AL2 023/ mendukung semua fitur di cabang Multi-container Docker sebelumnya. AL1 • Cabang AL2 platform AL2 023/ juga mendukung file `Dockerrun.aws.json` v2 yang sama. Untuk informasi selengkapnya tentang memigrasi aplikasi yang berjalan di cabang platform Multi-container Docker Amazon Linux ke Amazon ECS yang berjalan AL2 di cabang platform AL2 023/, lihat. ???

<table>
<tr><th>AL1 Cabang platform</th><th>Jalur Migrasi ke AL2 023/ AL2</th></tr>
<tr><td>Docker berjalan di Amazon Linux AMI () AL1

Docker yang telah dikonfigurasi sebelumnya (Glassfish 5.0) yang menjalankan Amazon Linux AMI () AL1</td><td>
Docker Berjalan di AL2 023/ cabang platform AL2

Kami menyarankan Anda memigrasikan aplikasi yang berjalan di lingkungan berdasarkan Docker yang telah dikonfigurasi sebelumnya (Glassfish 5.0) atau Docker yang berjalan di Amazon Linux AMI (AL1) ke lingkungan yang didasarkan pada Docker Running di Amazon Linux 2 atau Docker Running pada cabang platform 023. AL2

Jika lingkungan Anda didasarkan pada cabang platform Docker (Glassfish 5.0) yang telah dikonfigurasi sebelumnya, lihat. the section called “Tutorial - GlassFish di Docker: jalur ke Amazon Linux 2023”

Tabel berikut mencantumkan informasi migrasi khusus untuk cabang platform Docker Running pada AL2 AL2 023/.

<table>
<tr><th>Luas</th><th>Perubahan dan informasi</th></tr>
<tr><td>Penyimpanan</td><td>Elastic Beanstalk mengonfigurasi Docker untuk menggunakan driver penyimpanan untuk menyimpan gambar Docker dan data kontainer. Pada Amazon Linux AMI, Elastic Beanstalk menggunakan Driver penyimpanan Device Mapper. Untuk meningkatkan kinerja, Elastic Beanstalk menetapkan volume Amazon EBS ekstra. Pada versi platform AL2 023/ AL2 Docker, Elastic Beanstalk menggunakan driver penyimpanan OverlayFS, dan mencapai kinerja yang lebih baik sementara tidak memerlukan volume terpisah lagi.

Dengan Amazon Linux AMI, jika Anda menggunakan pilihan <code>BlockDeviceMappings</code> dari namespace <code>aws:autoscaling:launchconfiguration</code> untuk menambahkan volume penyimpanan khusus ke lingkungan Docker, kami menyarankan Anda untuk menambahkan juga volume Amazon EBS <code>/dev/xvdcz</code> yang disediakan oleh Elastic Beanstalk. Elastic Beanstalk tidak menyediakan volume ini lagi, jadi Anda harus menghapusnya dari file konfigurasi Anda. Untuk detail selengkapnya, lihat the section</td></tr>
</table>
</td></tr>
</table>

AL1 Cabang platform	Jalur Migrasi ke AL2 023/ AL2

Luas	Perubahan dan informasi
	called “Konfigurasi docker di Amazon Linux AMI (sebelumnya Amazon Linux 2)”.
Autentikasi penyimpanan privat	Saat Anda menyediakan file otentikasi buatan Docker untuk terhubung ke repositori pribadi, Anda tidak perlu lagi mengonversinya ke format lama yang diperlukan oleh versi platform Amazon Linux AMI Docker. AL2023/ Versi platform AL2 Docker mendukung format baru. Lihat perinciannya di the section called “Menggunakan gambar dari repositori pribadi”.
Server proksi	AL2023/ Versi platform AL2 Docker tidak mendukung kontainer mandiri yang tidak berjalan di belakang server proxy. Pada versi platform Amazon Linux AMI Docker, hal ini pernah memungkan untuk terjadi melalui nilai none dari pilihan `ProxyServer` di namespace `aws:elasticbeanstalk:environment:proxy` .

Go

Tabel berikut mencantumkan informasi migrasi untuk versi AL2 platform AL2 023/ di platform Go.

Luas	Perubahan dan informasi
Melewati port	Pada AL2 platform AL2 023/, Elastic Beanstalk tidak meneruskan nilai port ke proses aplikasi Anda melalui variabel lingkungan. `PORT` Anda dapat menyimulasikan perilaku ini untuk proses Anda dengan mengonfigurasi properti lingkungan `PORT` sendiri. Namun, jika Anda memiliki beberapa proses, dan Anda mengandalkan pada Elastic Beanstalk untuk melewati nilai port tambahan untuk proses Anda (5000, 5100, 5200 dll.), Anda harus memodifikasi implementasi Anda. Untuk detailnya lihatKonfigurasi proksi terbalik.

Amazon Corretto

Tabel berikut mencantumkan informasi migrasi untuk cabang platform Corretto di Platform Java SE.

Luas	Perubahan dan informasi
Corretto vs OpenJDK	Untuk mengimplementasikan Platform Java, Standard Edition (Java SE), AL2 023/ cabang AL2 platform menggunakan Amazon Corretto, AWS distribusi Open Java Development Kit (OpenJDK). Sebelumnya cabang platform Elastic Beanstalk Java SE menggunakan paket OpenJDK yang disertakan dengan Amazon Linux AMI.
Alat pembangunan	AL2023/ AL2 platform memiliki versi alat build yang lebih baru:`gradle`,, `maven` dan. `ant`
Penanganan file JAR	Di AL2 platform AL2 023/, jika bundel sumber Anda (file ZIP) berisi satu file JAR dan tidak ada file lain, Elastic Beanstalk tidak lagi mengganti nama file JAR menjadi. `application.jar` Mengganti nama terjadi hanya jika Anda mengirimkan file JAR sendiri, tidak dalam file ZIP.
Melewati port	Pada AL2 platform AL2 023/, Elastic Beanstalk tidak meneruskan nilai port ke proses aplikasi Anda melalui variabel lingkungan. `PORT` Anda dapat menyimula sikan perilaku ini untuk proses Anda dengan mengonfigurasi properti lingkungan `PORT` sendiri. Namun, jika Anda memiliki beberapa proses, dan Anda mengandal kan pada Elastic Beanstalk untuk melewati nilai port tambahan untuk proses Anda (5000, 5100, 5200 dll.), Anda harus memodifikasi implementasi Anda. Untuk detailnya lihatKonfigurasi proksi terbalik.
Java 7	Elastic Beanstalk tidak AL2 mendukung AL2 cabang platform 023/Java 7. Jika Anda memiliki aplikasi Java 7, migrasikan ke Corretto 8 atau Corretto 11.

Tomcat

Tabel berikut mencantumkan informasi migrasi untuk versi AL2 023/ AL2 platform di platform Tomcat.

<table>
<tr><th>Luas</th><th>Perubahan dan informasi</th></tr>
<tr><td>Pilihan Konfigurasi</td><td>Pada versi AL2 023/ AL2 platform, Elastic Beanstalk hanya mendukung sebagian dari opsi konfigurasi dan nilai opsi di namespace. <code>aws:elasticbeanstalk:environment:proxy</code> Berikut informasi migrasi untuk setiap pilihan.

<table><tr><th>Opsi</th><th>Informasi migrasi</th></tr><tr><td><code>GzipCompression</code></td><td>Tidak didukung pada versi AL2 023/ platformAL2 .</td></tr><tr><td><code>ProxyServer</code></td><td>AL2Versi platform 023/ AL2 Tomcat mendukung server proxy nginx dan Apache HTTPD versi 2.4. Namun, Apache versi 2.2 tidak didukung.

Pada versi platform Amazon Linux AMI, proksi default adalah Apache 2.4. Jika Anda menggunakan pengaturan proxy default dan menambahkan file konfigurasi proxy kustom, konfigurasi proxy Anda akan tetap berfungsi pada AL2 023/AL2. Namun, jika Anda menggunakan nilai pilihan <code>apache/2.2</code> , Anda sekarang harus memigrasi konfigurasi proksi Anda ke Apache versi 2.4.</td></tr></table>
<code>XX:MaxPermSize</code> Opsi di <code>aws:elasticbeanstalk:container:tomcat:jvmoptions</code> namespace tidak didukung pada versi AL2 AL2 023/ platform. Pengaturan JVM untuk memodifikasi ukuran generasi permanen hanya berlaku untuk Java 7 dan sebelumnya, dan karena itu tidak berlaku untuk versi AL2 023/ platformAL2 .</td></tr>
<tr><td>Jalur aplikasi</td><td>Pada AL2 platform AL2 023/, jalur ke direktori aplikasi di EC2 instans Amazon di lingkungan Anda adalah. <code>/var/app/current</code> <code>/var/lib/tomcat8/webapps</code> di platform Amazon Linux AMI.</td></tr>
</table>

Node.js

Tabel berikut mencantumkan informasi migrasi untuk versi AL2 platform AL2 023/ di platform Node.js.

<table>
<tr><th>Luas</th><th>Perubahan dan informasi</th></tr>
<tr><td>Versi Node.js yang dipasang</td><td>Pada AL2 platform AL2 023/, Elastic Beanstalk memelihara beberapa cabang platform Node.js, dan hanya menginstal versi terbaru dari versi mayor Node.js yang sesuai dengan cabang platform pada setiap versi platform. Sebagai contoh, setiap versi platform di cabang platform Node.js 12 hanya memiliki Node.js 12.x.y yang dipasang secara default. Pada versi platform Amazon Linux AMI, kami memasang beberapa versi dari beberapa versi Node.js pada setiap versi platform, dan hanya mempertahankan cabang platform tunggal.

Pilih cabang platform Node.js yang sesuai dengan versi utama Node.js yang aplikasi Anda butuhkan.</td></tr>
<tr><td>Nama file log Apache HTTPD</td><td>Pada AL2 023/ AL2 platform, jika Anda menggunakan server proxy Apache HTTPD, nama file log HTTPD adalah <code>access_log</code> dan<code>error_log</code> , yang konsisten dengan semua platform lain yang mendukung Apache HTTPD. Pada versi platform Amazon Linux AMI file log ini dinamakan <code>access.log</code> dan <code>error.log</code> , masing-masing.

Untuk rincian tentang nama file log dan lokasi untuk semua platform, lihat the section called “Bagaimana Elastic Beanstalk mengatur Log CloudWatch ”.</td></tr>
<tr><td>Pilihan konfigurasi</td><td>Pada AL2 platform AL2 023/, Elastic Beanstalk tidak mendukung opsi konfigurasi di namespace. <code>aws:elasticbeanstalk:container:nodejs</code> Beberapa pilihan memiliki alternatif. Berikut informasi migrasi untuk setiap pilihan.

<table>
<tr><th>Opsi</th><th>Informasi migrasi</th></tr>
<tr><td><code>NodeComm</code>
nd</td><td>Gunakan kata kunci <code>Procfile</code> atau <code>scripts</code> dalam file <code>package.json</code> untuk menentukan skrip awal.</td></tr>
<tr><td><code>NodeVers</code>
on</td><td>Gunakan kata kunci <code>engines</code> dalam file <code>package.json</code> untuk menentukan versi Node.js. Perhatikan bahwa Anda hanya dapat menentukan versi Node.js yang berhubungan dengan cabang platform Anda. Sebagai contoh, jika Anda menggunakan cabang platform Node.js 12, Anda hanya dapat menentukan versi 12.x.y</td></tr>
</table>
</td></tr>
</table>

Luas	Perubahan dan informasi

Opsi	Informasi migrasi
	Node.js. Lihat perinciannya di the section called “Menentukan Node.js dependensi dengan file package.json”.
`GzipCompression`	Tidak didukung pada versi AL2 023/ platformAL2 .

`ProxyServer`	Pada versi platform AL2 023/ AL2 Node.js, opsi ini dipindahkan ke namespace. `aws:elasticbeanstalk:environment:proxy` Anda dapat memilih antara `nginx` (default) dan `apache`.
	
	AL2Versi platform 023/ AL2 Node.js tidak mendukung aplikasi mandiri yang tidak berjalan di belakang server proxy. Pada versi platform Amazon Linux AMI Node.js, hal ini pernah memungkinkan untuk terjadi melalui nilai none dari pilihan `ProxyServer` di namespace `aws:elasticbeanstalk:container:nodejs` . Jika lingkungan Anda menjalankan aplikasi mandiri, perbarui kode Anda untuk mendengarkan port dii mana server proksi (nginx atau Apache) meneruskan lalu lintas.
	
	`var port = process.env.PORT || 5000;`
	
	`app.listen(port, function() {`
	`  console.log('Server running at http://127.0.0.1:%s',`
	` port);`
	`});`

PHP

Tabel berikut mencantumkan informasi migrasi untuk versi AL2 023/ AL2 platform di platform PHP.

Luas	Perubahan dan informasi
Pengolahan file PHP	Pada AL2 AL2 023/platform, file PHP diproses menggunakan PHP-FPM (manajer proses CGI). Pada platform Amazon Linux AMI kami menggunakan mod_php (modul Apache).
Server proksi	AL2Versi platform 023/ AL2 PHP mendukung server proxy nginx dan Apache HTTPD. Default nya adalah nginx. Versi platform Amazon Linux AMI PHP mendukung hanya Apache HTTPD. Jika Anda menambahkan file konfigurasi Apache khusus, Anda dapat mengatur pilihan `ProxyServer` di namespace `aws:elasticbeanstalk:environment:proxy` untuk `apache`.

Python

Tabel berikut mencantumkan informasi migrasi untuk versi AL2 023/ AL2 platform di platform Python.

Luas	Perubahan dan informasi
Server WSGI	Pada AL2 AL2 023/platform, Gunicorn adalah server WSGI default. Secara default, Gunicorn mendengarkan port 8000. Port mungkin berbeda dari apa yang aplikasi Anda gunakan pada platform Amazon Linux AMI. Jika Anda menyetel pilihan `WSGIPath` dari namespace `aws:elasticbeanstalk:container:python`, ganti nilai dengan sintaks Gunicorn. Untuk detail selengkapnya, lihat the section called “Namespace konfigurasi Python”. Alternatifnya, Anda dapat menggunakan `Procfile` untuk menentukan dan mengonfigurasi server WSGI. Untuk detail selengkapnya, lihat the section called “Procfile”.
Jalur aplikasi	Pada AL2 platform AL2 023/, jalur ke direktori aplikasi di EC2 instans Amazon di lingkungan Anda adalah. `/var/app/current` `/opt/python/current/app` di platform Amazon Linux AMI.
Server proksi	AL2Versi platform 023/ AL2 Python mendukung server proxy nginx dan Apache HTTPD. Default-nya adalah nginx.

Luas	Perubahan dan informasi
	Versi platform Amazon Linux AMI Python hanya mendukung Apache HTTPD. Jika Anda menambahkan file konfigurasi Apache khusus, Anda dapat mengatur pilihan `ProxyServer` di namespace `aws:elasticbeanstalk:environment:proxy` untuk `apache`.

Ruby

Tabel berikut mencantumkan informasi migrasi untuk versi AL2 platform AL2 023/ di platform Ruby.

Luas	Perubahan dan informasi
Versi Ruby yang dipasang	Pada AL2 platform AL2 023/, Elastic Beanstalk hanya menginstal versi terbaru dari satu versi Ruby, sesuai dengan cabang platform, pada setiap versi platform. Sebagai contoh, setiap versi platform di cabang platform Ruby 2.6 hanya memiliki Ruby 2.6.x yang terpasang. Pada versi platform Amazon Linux AMI, kami memasang versi terbaru dari beberapa versi Ruby, misalnya, 2.4.x, 2.5.x, dan 2.6.x. Jika aplikasi Anda menggunakan versi Ruby yang tidak sesuai dengan cabang platform yang Anda gunakan, kami merekomendasikan agar Anda beralih ke cabang platform yang memiliki versi Ruby yang benar untuk aplikasi Anda.
Server Aplikasi	Pada AL2 platform AL2 023/, Elastic Beanstalk hanya menginstal server aplikasi Puma di semua versi platform Ruby. Anda dapat menggunakan `Procfile` untuk memulai server aplikasi yang berbeda, dan `Gemfile` untuk memasangnya. Pada platform Amazon Linux AMI, kami mendukung dua rasa cabang platform untuk setiap versi Ruby—satu dengan server aplikasi Puma dan yang lainnya dengan server aplikasi Passenger. Jika aplikasi Anda menggunakan Passenger , Anda dapat mengonfigurasi lingkungan Ruby Anda untuk memasang dan menggunakan Passenger. Untuk informasi selengkapnya dan contoh tambahan, lihat the section called “Platform Ruby”.

FAQ Pensiun Platform

> **Note**
>
> Elastic Beanstalk menghentikan semua cabang platform berbasis Amazon Linux AL1 AMI () pada 18 Juli 2022.

Jawaban dalam FAQ ini mengacu pada topik-topik berikut:

- Kebijakan dukungan platform Elastic Beanstalk
- Sejarah cabang platform pensiunan
- Platform yang didukung Elastic Beanstalk di Platform AWS Elastic Beanstalk
- Memigrasi aplikasi Elastic Beanstalk Linux Anda ke Amazon Linux 2023 atau Amazon Linux 2
- Amazon Linux 2 FAQs.

1. Apa arti pensiun dari cabang platform?

Setelah tanggal pensiun yang diumumkan dari cabang platform, Anda tidak akan lagi dapat membuat lingkungan baru berdasarkan cabang platform yang sudah pensiun, kecuali jika Anda sudah memiliki lingkungan aktif berdasarkan cabang platform tersebut. Untuk informasi selengkapnya, lihat FAQ #11. Elastic Beanstalk akan berhenti memberikan pembaruan pemeliharaan baru untuk cabang platform ini. Cabang platform yang sudah pensiun tidak direkomendasikan untuk digunakan di lingkungan produksi. Untuk informasi selengkapnya, lihat FAQ #5.

2. Mengapa telah AWS pensiun dari cabang platform AL1 berbasis?

Elastic Beanstalk menghentikan cabang platform ketika komponen platform tidak digunakan lagi atau dihentikan oleh vendornya. Dalam hal ini, Amazon Linux AMI (AL1) telah mengakhiri dukungan standar pada 31 Desember 2020. Sementara Elastic Beanstalk terus menawarkan platform berbasis AL1 hingga 2022, kami AL2 telah AL2 merilis dan 023 dan platform berbasis yang memiliki fitur terbaru. Agar pelanggan dapat terus mendapatkan keuntungan dari keamanan dan fitur terbaru di masa depan, sangat penting bagi pelanggan untuk bermigrasi ke platform berbasis kami AL2 atau AL2 023.

3. Cabang platform mana yang sudah pensiun?

Untuk daftar komponen platform dan cabang platform yang telah pensiun, lihatSejarah cabang platform pensiunan.

4. Platform mana yang saat ini didukung?

Lihat platform yang didukung Elastic Beanstalk di Platform.AWS Elastic Beanstalk

5. Akankah Elastic Beanstalk menghapus atau menghentikan komponen lingkungan saya setelah pensiun?

Kebijakan kami untuk cabang platform yang sudah pensiun tidak menghapus akses ke lingkungan atau menghapus sumber daya. Namun, lingkungan yang didasarkan pada cabang platform yang sudah pensiun dapat berakhir dalam situasi yang tidak terduga, karena Elastic Beanstalk tidak dapat memberikan pembaruan keamanan, dukungan teknis, atau perbaikan terbaru untuk cabang platform yang sudah pensiun karena pemasok menandai komponen mereka sebagai End of Life (EOL). Misalnya, kerentanan keamanan yang merugikan dan kritis dapat muncul di lingkungan yang berjalan di cabang platform yang sudah pensiun. Atau tindakan EB API dapat berhenti bekerja untuk lingkungan jika menjadi tidak kompatibel dengan layanan Elastic Beanstalk dari waktu ke waktu. Peluang untuk jenis risiko ini meningkat semakin lama lingkungan berdasarkan cabang platform yang sudah pensiun tetap aktif.

Jika aplikasi Anda mengalami masalah saat berjalan di cabang platform yang sudah pensiun dan Anda tidak dapat memigrasikannya ke platform yang didukung, Anda harus mempertimbangkan alternatif lain. Solusinya termasuk merangkum aplikasi ke dalam gambar Docker untuk menjalankannya sebagai wadah Docker. Ini akan memungkinkan pelanggan untuk menggunakan salah satu solusi Docker kami, seperti platform Elastic AL2 Beanstalk 023/Docker kami, atau layanan berbasis Docker lainnya seperti Amazon AL2 ECS atau Amazon EKS. Alternatif non-Docker termasuk AWS CodeDeploy layanan kami, yang memungkinkan penyesuaian lengkap runtime yang Anda inginkan.

6. Bisakah saya mengajukan permintaan untuk memperpanjang tanggal pensiun?

Tidak. Setelah tanggal pensiun lingkungan yang ada akan terus berfungsi. Namun, Elastic Beanstalk tidak akan lagi menyediakan pemeliharaan platform dan pembaruan keamanan. Oleh karena itu, sangat penting untuk bermigrasi ke AL2 atau AL2 023 jika Anda masih menjalankan aplikasi pada platform AL1 berbasis. Untuk informasi selengkapnya tentang risiko dan solusi, lihat FAQ #5.

7. Apa solusinya jika saya tidak dapat menyelesaikan migrasi saya AL2 atau AL2 023 tepat waktu?

Pelanggan dapat terus menjalankan lingkungan, meskipun kami sangat menyarankan Anda untuk merencanakan untuk memigrasikan semua lingkungan Elastic Beanstalk Anda ke versi platform yang didukung. Melakukannya akan meminimalkan risiko dan memberikan manfaat berkelanjutan dari peningkatan keamanan, kinerja, dan fungsionalitas penting yang ditawarkan dalam rilis yang lebih baru. Untuk informasi selengkapnya tentang risiko dan solusi, lihat FAQ #5.

8. Apa proses yang disarankan untuk bermigrasi ke AL2 atau AL2 023 platform?

Untuk petunjuk AL2 migrasi AL1 yang komprehensif hingga AL2 023/, lihat. Memigrasi aplikasi Elastic Beanstalk Linux Anda ke Amazon Linux 2023 atau Amazon Linux 2 Topik ini menjelaskan bahwa Elastic Beanstalk memerlukan penerapan biru/hijau untuk melakukan peningkatan.

9. Jika saya memiliki lingkungan yang berjalan di platform pensiunan, apa dampaknya?

Lingkungan berdasarkan cabang platform yang sudah pensiun dapat berakhir dalam situasi yang tidak terduga, karena Elastic Beanstalk tidak dapat memberikan pembaruan keamanan, dukungan teknis, atau perbaikan terbaru untuk cabang platform yang sudah pensiun karena pemasok menandai komponen mereka sebagai End of Life (EOL). Misalnya, kerentanan keamanan yang merugikan dan kritis dapat muncul di lingkungan yang berjalan di cabang platform yang sudah pensiun. Atau tindakan EB API dapat berhenti bekerja untuk lingkungan jika menjadi tidak kompatibel dengan layanan Elastic Beanstalk dari waktu ke waktu. Peluang untuk jenis risiko ini meningkat semakin lama lingkungan di cabang platform yang sudah pensiun tetap aktif. Untuk informasi selengkapnya, lihat FAQ #5.

10. Apa yang terjadi 90 hari setelah tanggal pensiun?

Kebijakan kami untuk cabang platform yang sudah pensiun tidak menghapus akses ke lingkungan atau menghapus sumber daya. Namun, ketahuilah bahwa lingkungan yang didasarkan pada cabang platform yang sudah pensiun dapat berakhir dalam situasi yang tidak terduga, karena Elastic Beanstalk tidak dapat memberikan pembaruan keamanan, dukungan teknis, atau perbaikan terbaru untuk cabang platform yang sudah pensiun karena pemasok menandai komponen mereka sebagai End of Life (EOL). Misalnya, kerentanan keamanan yang merugikan dan kritis dapat muncul di lingkungan yang berjalan di cabang platform yang sudah pensiun. Atau tindakan EB API dapat berhenti bekerja untuk lingkungan jika menjadi tidak kompatibel dengan layanan Elastic Beanstalk

dari waktu ke waktu. Peluang untuk jenis risiko ini meningkat semakin lama lingkungan di cabang platform yang sudah pensiun tetap aktif. Untuk informasi lebih lanjut lihat FAQ #5.

11. Bisakah saya membuat lingkungan baru berdasarkan platform yang sudah pensiun?

Anda dapat membuat lingkungan baru berdasarkan cabang platform yang sudah pensiun, jika Anda telah menggunakan cabang platform tersebut untuk membuat lingkungan yang ada menggunakan akun yang sama dan di wilayah yang sama. Cabang platform yang sudah pensiun tidak akan tersedia di konsol Elastic Beanstalk. Namun, untuk pelanggan yang memiliki lingkungan yang ada berdasarkan cabang platform yang sudah pensiun, itu akan tersedia melalui EB CLI, EB API, dan. AWS CLI Selain itu, pelanggan yang sudah ada dapat menggunakan lingkungan Clone dan konsol lingkungan Rebuild. Namun, ketahuilah bahwa lingkungan yang didasarkan pada cabang platform yang sudah pensiun dapat berakhir dalam situasi yang tidak terduga. Untuk informasi selengkapnya, lihat FAQ #5.

12. Jika saya memiliki lingkungan yang ada yang berjalan di cabang platform yang sudah pensiun, sampai kapan saya dapat membuat lingkungan baru berdasarkan cabang platform yang sudah pensiun? Bisakah saya melakukannya menggunakan konsol, CLI atau API?

Anda dapat menciptakan lingkungan setelah tanggal pensiun. Namun, perlu diingat bahwa cabang platform yang sudah pensiun dapat berakhir dalam situasi yang tidak terduga. Semakin jauh dalam waktu lingkungan seperti itu diciptakan atau aktif, semakin tinggi risiko bagi lingkungan untuk menghadapi masalah yang tidak terduga. Untuk informasi selengkapnya tentang membuat lingkungan baru, lihat FAQ #11.

13. Dapatkah saya mengkloning atau membangun kembali lingkungan saya yang didasarkan pada platform yang sudah pensiun?

Ya. Anda dapat melakukannya menggunakan lingkungan Clone dan Rebuild konsol lingkungan. Anda juga dapat menggunakan EB CLI, EB API, dan. AWS CLI Untuk informasi selengkapnya tentang membuat lingkungan baru, lihat FAQ #11.

Namun, kami sangat menyarankan Anda untuk merencanakan untuk memigrasikan semua lingkungan Elastic Beanstalk Anda ke versi platform yang didukung. Melakukannya akan meminimalkan risiko dan memberikan manfaat berkelanjutan dari peningkatan keamanan,

kinerja, dan fungsionalitas penting yang ditawarkan dalam rilis yang lebih baru. Untuk informasi selengkapnya tentang risiko dan solusi, lihat FAQ #5.

14. Setelah tanggal pensiun, apa yang akan terjadi pada AWS sumber daya lingkungan Elastic Beanstalk saya yang didasarkan pada cabang platform yang sudah pensiun? Misalnya, jika EC2 instance yang sedang berjalan dihentikan, apakah Elastic Beanstalk dapat meluncurkan instance berbasis baru untuk mempertahankan kapasitas AL1 ? EC2

Sumber daya lingkungan akan tetap aktif dan terus berfungsi. Dan, ya, Elastic Beanstalk akan AL1 EC2 otomatis menskalakan untuk contoh di lingkungan. Namun, Elastic Beanstalk akan berhenti memberikan pembaruan pemeliharaan platform baru ke lingkungan, yang dapat menyebabkan lingkungan berakhir dalam situasi yang tidak terduga dari waktu ke waktu. Untuk informasi selengkapnya, lihat FAQ #5.

15. Apa perbedaan utama antara sistem operasi AL2 023/ dan AL2 Amazon Linux AMI (AL1)? Bagaimana cabang platform Elastic AL2 Beanstalk AL2 023/terpengaruh?

Meskipun Amazon Linux AMI dan AL2 023/ AL2 berbagi kernel Linux yang sama, mereka berbeda dalam sistem inisialisasi, `libc` versi, rantai alat kompiler, dan berbagai paket. Untuk informasi selengkapnya, lihat Amazon Linux 2 FAQs.

Layanan Elastic Beanstalk juga telah memperbarui versi spesifik platform runtime, alat build, dan dependensi lainnya. Cabang platform AL2 berbasis AL2 023/ tidak dijamin kompatibel dengan aplikasi Anda yang ada. Selain itu, bahkan jika kode aplikasi Anda berhasil di-deploy ke versi platform baru, mungkin dapat berperilaku atau melakukan hal berbeda karena perbedaan sistem operasi dan waktu pengoperasian. Untuk daftar dan deskripsi konfigurasi dan penyesuaian yang perlu Anda tinjau dan uji, lihat. Memigrasi aplikasi Elastic Beanstalk Linux Anda ke Amazon Linux 2023 atau Amazon Linux 2

Membatalkan pembaruan konfigurasi lingkungan dan deployment aplikasi

Anda dapat membatalkan pembaruan yang sedang berlangsung yang dipicu oleh perubahan konfigurasi lingkungan. Anda juga dapat membatalkan deployment versi aplikasi baru yang sedang berlangsung. Misalnya, Anda mungkin ingin membatalkan pembaruan jika Anda memutuskan untuk

tetap menggunakan konfigurasi lingkungan yang ada alih-alih menerapkan pengaturan konfigurasi lingkungan baru. Atau, Anda mungkin menyadari bahwa versi aplikasi yang baru yang Anda deploy memiliki masalah yang akan menyebabkannya tidak dapat memulai atau tidak berjalan dengan benar. Dengan membatalkan pembaruan lingkungan atau pembaruan versi aplikasi, Anda dapat menghindari menunggu hingga proses pembaruan atau deployment terselesaikan sebelum Anda memulai upaya baru untuk memperbarui versi lingkungan atau aplikasi.

> **Note**
>
> Selama fase pembersihan di mana sumber daya lama yang tidak lagi diperlukan dihapus, setelah batch instans terakhir telah diperbarui, Anda tidak dapat lagi membatalkan pembaruan.

Elastic Beanstalk melakukan rollback dengan cara yang sama dengan yang dilakukannya pada pembaruan terakhir yang berhasil. Misalnya, jika Anda memiliki pembaruan bergulir berbasis waktu diaktifkan di lingkungan Anda, maka Elastic Beanstalk akan menunggu waktu jeda yang telah ditentukan di antara pengembalian perubahan pada satu batch instans sebelum mengembalikan perubahan pada batch berikutnya. Atau, jika Anda baru saja mengaktifkan pembaruan bergulir, tetapi terakhir kali Anda berhasil memperbarui pengaturan konfigurasi lingkungan Anda dilakukan tanpa pembaruan bergulir, Elastic Beanstalk akan melakukan rollback pada semua instans secara bersamaan.

Anda tidak dapat menghentikan Elastic Beanstalk untuk mengembalikan ke konfigurasi lingkungan sebelumnya setelah mulai membatalkan pembaruan. Proses rollback berlanjut sampai semua instans di lingkungan memiliki konfigurasi lingkungan sebelumnya atau sampai proses pengembalian mengalami kegagalan. Untuk deployment versi aplikasi, membatalkan penerapan hanya menghentikan deployment; beberapa instans akan memiliki versi aplikasi baru dan instans lainnya akan terus menjalankan versi aplikasi yang ada. Anda dapat men-deploy versi aplikasi yang sama atau yang lain nanti.

Untuk informasi lebih lanjut tentang pembaruan bergulir, lihat Pembaruan konfigurasi lingkungan bergulir Elastic Beanstalk. Untuk informasi selengkapnya tentang deployment versi aplikasi batch, lihat Kebijakan dan pengaturan deployment.

Untuk membatalkan pembaruan

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS

2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada halaman ikhtisar lingkungan, pilih Tindakan, lalu pilih Batalkan operasi saat ini.

Membangun kembali lingkungan Elastic Beanstalk

AWS Elastic Beanstalk Lingkungan Anda dapat menjadi tidak dapat digunakan jika Anda tidak menggunakan fungsionalitas Elastic Beanstalk untuk memodifikasi atau menghentikan sumber daya yang mendasari lingkungan. AWS Jika ini terjadi, Anda dapat membangun kembali lingkungan untuk mencoba mengembalikannya ke keadaan yang bekerja. Membangun kembali lingkungan mengakhiri semua sumber daya dan menggantikannya dengan sumber daya baru dengan konfigurasi yang sama.

Anda juga dapat membangun kembali lingkungan yang diakhiri dalam waktu enam minggu (42 hari) dari waktu pengakhiran. Ketika Anda membangun kembali, Elastic Beanstalk mencoba untuk membuat lingkungan baru dengan nama, ID, dan konfigurasi yang sama.

Membangun kembali lingkungan yang sedang berjalan

Anda dapat membangun kembali lingkungan melalui konsol Elastic Beanstalk atau dengan menggunakan API `RebuildEnvironment`.

> **Warning**
>
> Jika lingkungan Anda memiliki database yang digabungkan, itu akan dihapus sebagai bagian dari pembangunan kembali, dan database baru di lingkungan yang dibangun kembali tidak akan berisi data sebelumnya. Jika Anda ingin mempertahankan database atau mengambil snapshot, pastikan Anda memiliki kebijakan penghapusan database yang dikonfigurasi dengan benar untuk hasil yang diinginkan setelah dibangun kembali. Untuk informasi selengkapnya, lihat Siklus hidup basis data.

Untuk membangun kembali lingkungan berjalan (konsol)

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pilih Actions, lalu pilih Rebuild environment.

4. Pilih Membangun kembali.

Untuk membangun kembali lingkungan berjalan dengan Elastic Beanstalk API, `RebuildEnvironment`gunakan tindakan dengan atau SDK. AWS CLI AWS

```
$ aws elasticbeanstalk rebuild-environment --environment-id e-vdnftxubwq
```

Membangun kembali lingkungan yang diakhiri

Anda dapat membangun kembali lingkungan dengan menggunakan konsol Elastic Beanstalk, EB CLI, atau API `RebuildEnvironment`.

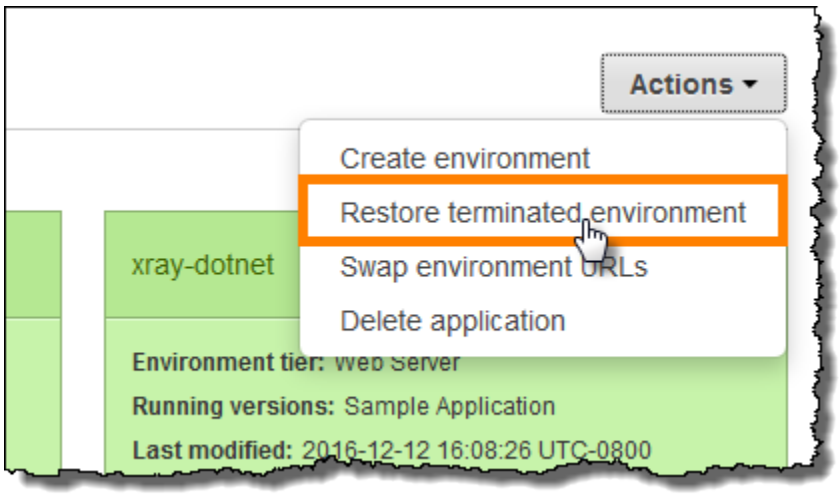
> **Note**
>
> Kecuali Anda menggunakan nama domain khusus Anda sendiri dengan lingkungan Anda yang telah diakhiri, lingkungan menggunakan subdomain dari elasticbeanstalk.com. Subdomain ini dibagi dalam wilayah Elastic Beanstalk. Oleh karena itu, mereka dapat digunakan oleh lingkungan yang dibuat oleh setiap pelanggan di wilayah yang sama. Sementara lingkungan Anda diakhiri, lingkungan lain bisa menggunakan subdomainnya. Dalam kasus ini, pembangunan kembali akan gagal.
> Anda dapat menghindari masalah ini dengan menggunakan domain kustom. Lihat Nama domain lingkungan Elastic Beanstalk Anda untuk detail selengkapnya.

Lingkungan yang baru diakhiri muncul dalam gambaran umum aplikasi hingga satu jam. Selama waktu ini, Anda dapat melihat peristiwa lingkungan di Dasbor, dan gunakan tindakan Memulihkan lingkungan untuk membangunnya kembali.

Untuk membangun kembali lingkungan yang tidak lagi terlihat, gunakan pilihan Memulihkan lingkungan yang diakhiri dari halaman aplikasi.

Untuk membangun kembali lingkungan yang telah diakhiri (konsol)

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Pada panel navigasi, pilih Aplikasi, dan kemudian pilih nama aplikasi Anda dari daftar.
3. Pilih Tindakan, lalu pilih Memulihkan lingkungan yang telah diakhiri.



4. Pilih lingkungan yang telah diakhiri.
5. Pilih Memulihkan.



Elastic Beanstalk mencoba membuat lingkungan baru dengan nama, ID, dan konfigurasi yang sama. Jika lingkungan dengan nama atau URL yang sama ada ketika Anda mencoba untuk membangun kembali, pembangunan kembali akan mengalami kegagalan. Menghapus versi aplikasi yang di-deploy ke lingkungan juga akan menyebabkan pembangunan kembali gagal.

Jika Anda menggunakan EB CLI untuk mengelola lingkungan Anda, gunakan perintah eb restore untuk membangun kembali lingkungan yang telah diakhiri.

```
$ eb restore e-vdnftxubwq
```

Lihat eb restore untuk informasi selengkapnya.

Untuk membangun kembali lingkungan yang dihentikan dengan Elastic Beanstalk API, gunakan `RebuildEnvironment`tindakan dengan atau SDK. AWS CLI AWS

```
$ aws elasticbeanstalk rebuild-environment --environment-id e-vdnftxubwq
```

Jenis lingkungan

Di AWS Elastic Beanstalk, Anda dapat membuat lingkungan load-balanced, scalable atau lingkungan single instance. Jenis lingkungan yang Anda butuhkan tergantung pada aplikasi yang Anda deploy. Misalnya, Anda dapat mengembangkan dan menguji aplikasi di lingkungan instans tunggal untuk menghemat biaya dan kemudian meningkatkan lingkungan tersebut menjadi lingkungan yang dapat diskalakan dengan beban seimbang ketika aplikasi siap untuk produksi.

> **Note**
>
> Tingkat lingkungan pekerja untuk aplikasi web yang memproses tugas latar belakang tidak termasuk penyeimbang beban. Namun, lingkungan pekerja tidak secara efektif menskalakan keluar dengan menambahkan instans untuk grup Auto Scaling untuk memproses data dari antrian Amazon SQS ketika beban memerlukan itu.

Lingkungan yang dapat diskalakan dengan beban seimbang

Lingkungan yang seimbang beban dan dapat diskalakan menggunakan layanan Elastic Load Balancing dan Amazon Auto EC2 Scaling untuk menyediakan instans EC2 Amazon yang diperlukan untuk aplikasi yang Anda gunakan. Amazon EC2 Auto Scaling secara otomatis memulai instans tambahan untuk mengakomodasi peningkatan beban pada aplikasi Anda. Jika beban pada aplikasi Anda berkurang, Amazon EC2 Auto Scaling menghentikan instance tetapi selalu membiarkan jumlah minimum instans yang Anda tentukan berjalan. Jika aplikasi Anda memerlukan skalabilitas dengan opsi untuk berjalan di beberapa Availability Zone, gunakan lingkungan yang dapat diskalakan dengan beban seimbang. Jika Anda tidak yakin jenis lingkungan mana yang akan dipilih, Anda dapat memilih salah satu dan, jika diperlukan, beralih jenis lingkungan nanti.

Lingkungan instans tunggal

Lingkungan instans tunggal berisi satu EC2 instans Amazon dengan alamat IP Elastis. Lingkungan instans tunggal tidak memiliki penyeimbang beban, yang dapat membantu Anda mengurangi biaya dibandingkan dengan lingkungan yang dapat diskalakan dengan beban seimbang. Meskipun lingkungan instans tunggal menggunakan layanan EC2 Auto Scaling Amazon, pengaturan untuk

jumlah instans minimum, jumlah instans maksimum, dan kapasitas yang diinginkan semuanya diatur ke 1. Akibatnya, instans baru tidak mulai mengakomodasi peningkatan beban pada aplikasi Anda.

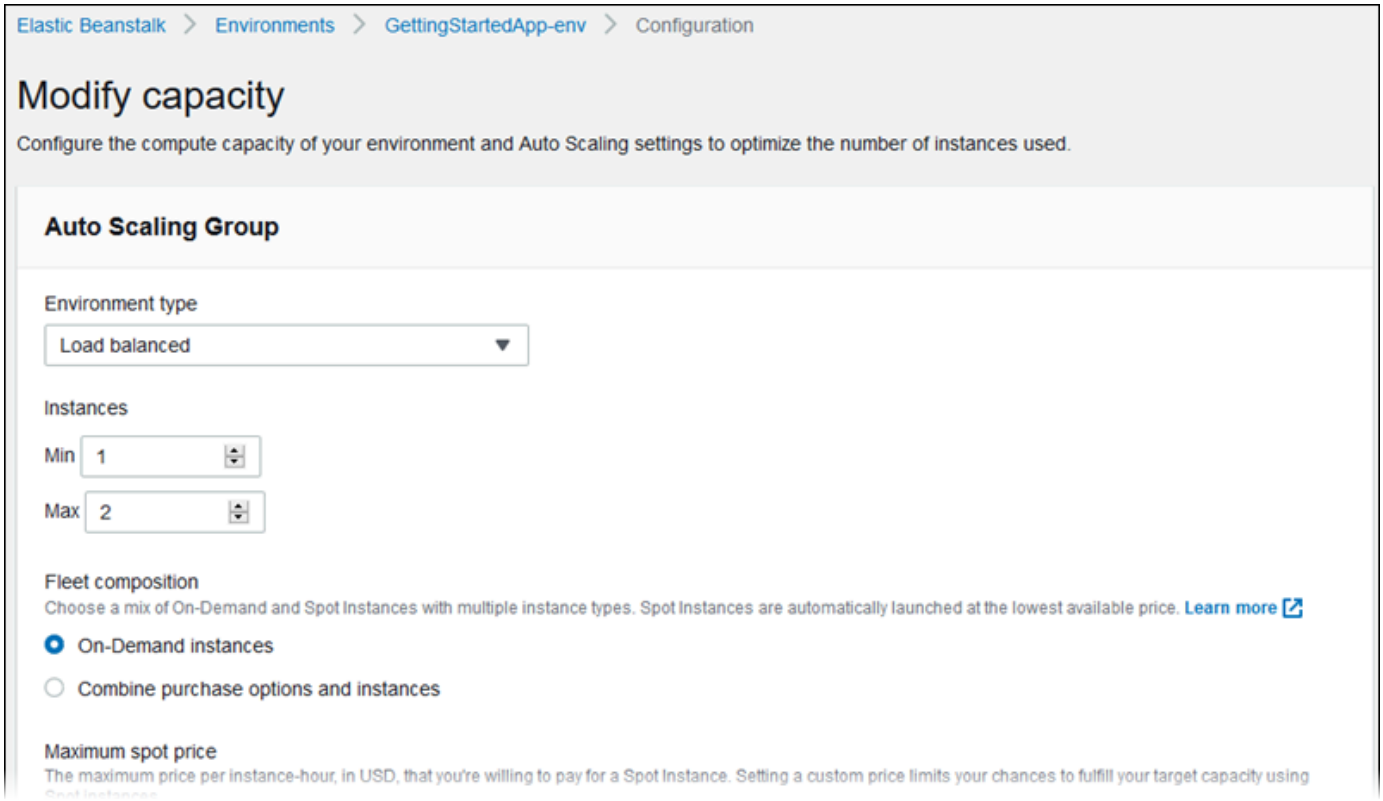
Gunakan lingkungan instans tunggal jika Anda mengharapkan aplikasi produksi Anda untuk memiliki lalu lintas rendah atau jika Anda melakukan pengembangan jarak jauh. Jika Anda tidak yakin jenis lingkungan mana yang akan dipilih, Anda dapat memilih salah satu dan, jika diperlukan, Anda dapat mengganti jenis lingkungannya nanti. Untuk informasi selengkapnya, lihat Mengubah jenis lingkungan.

Mengubah jenis lingkungan

Anda dapat mengubah jenis lingkungan Anda menjadi lingkungan yang dapat diskalakan dengan instans tunggal atau beban seimbang dengan mengedit konfigurasi lingkungan Anda. Dalam beberapa kasus, Anda mungkin ingin mengubah jenis lingkungan Anda dari satu jenis ke jenis lainnya. Misalnya, katakanlah Anda mengembangkan dan menguji aplikasi di lingkungan instans tunggal untuk menghemat biaya. Ketika aplikasi Anda siap untuk produksi, Anda dapat mengubah jenis lingkungan menjadi lingkungan yang dapat diskalakan dengan beban seimbang sehingga dapat disesuaikan untuk memenuhi permintaan pelanggan Anda.

Untuk mengubah jenis lingkungan

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Di kategori Kapasitas, pilih Edit.
5. Dari daftar Jenis Lingkungan, pilihan jenis lingkungan yang Anda inginkan.



6. Pilih Simpan.

   Diperlukan beberapa menit bagi lingkungan untuk memperbarui sementara Elastic Beanstalk menyediakan sumber daya AWS .

Jika lingkungan Anda berada dalam VPC, pilih subnet untuk menempatkan Elastic Load Balancing dan instans Amazon. EC2 Setiap Availability Zone tempat aplikasi Anda berjalan harus memiliki keduanya. Lihat Menggunakan Elastic Beanstalk dengan Amazon VPC untuk detailnya.

Lingkungan pekerja Elastic Beanstalk

Jika AWS Elastic Beanstalk aplikasi Anda menjalankan operasi atau alur kerja yang membutuhkan waktu lama untuk diselesaikan, Anda dapat menurunkan tugas tersebut ke lingkungan pekerja khusus. Pemisahan aplikasi web front end Anda dari proses yang melakukan operasi pemblokiran adalah cara yang umum untuk memastikan bahwa aplikasi Anda tetap responsif di bawah beban.

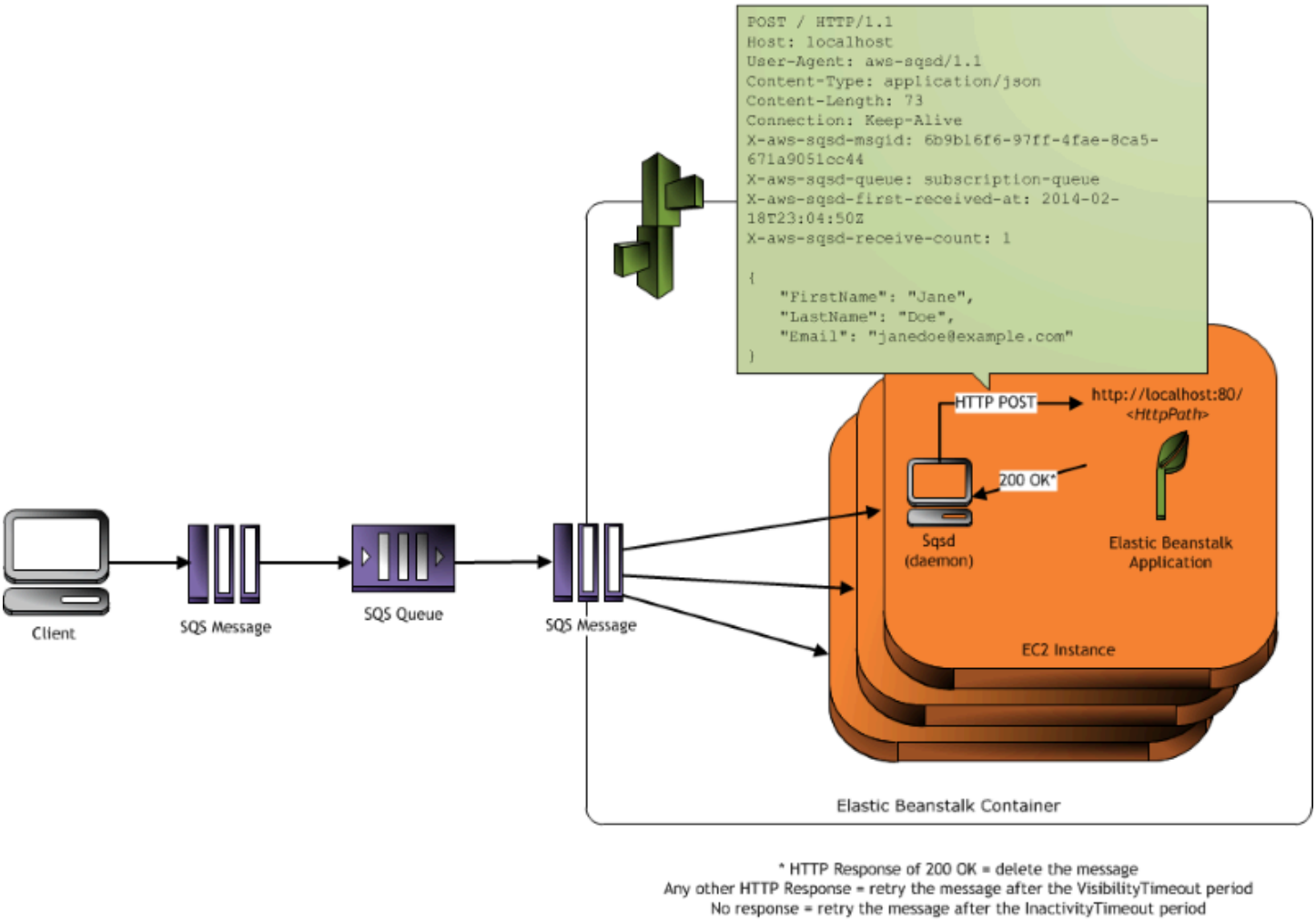
Tugas yang berjalan lama adalah sesuatu yang secara substansial meningkatkan waktu yang diperlukan untuk menyelesaikan permintaan, seperti memproses gambar atau video, mengirim email, atau membuat arsip ZIP. Operasi ini dapat mengambil hanya satu atau dua detik untuk

menyelesaikan, tetapi penundaan beberapa detik adalah banyak untuk permintaan web yang jika tidak akan selesai dalam waktu kurang dari 500 ms.

Salah satu pilihan adalah untuk memunculkan proses pekerja lokal, mengembalikan keberhasilan, dan memproses tugas secara asinkron. Ini bekerja jika instans Anda dapat mengikuti semua tugas yang dikirim ke sana. Di bawah beban tinggi, bagaimanapun, sebuah instans dapat menjadi kewalahan dengan tugas latar belakang dan menjadi tidak responsif terhadap permintaan prioritas yang lebih tinggi. Jika pengguna individu dapat menghasilkan beberapa tugas, peningkatan beban mungkin tidak sesuai dengan peningkatan pengguna, sehingga sulit untuk menskalakan keluar tingkat server web Anda secara efektif.

Untuk menghindari menjalankan tugas yang berjalan lama secara lokal, Anda dapat menggunakan AWS SDK untuk bahasa pemrograman Anda untuk mengirimnya ke antrean Amazon Simple Queue Service (Amazon SQS), dan menjalankan proses yang menjalankannya pada kumpulan instance terpisah. Anda kemudian merancang instans pekerja ini untuk mengambil item dari antrean hanya ketika mereka memiliki kapasitas untuk menjalankannya, mencegah mereka dari menjadi kewalahan.

Lingkungan pekerja Elastic Beanstalk menyederhanakan proses ini dengan mengelola antrean Amazon SQS dan menjalankan proses daemon pada setiap instans yang berbunyi dari antrean untuk Anda. Ketika daemon menarik item dari antrean, ia akan mengirimkan permintaan HTTP POST secara lokal ke `http://localhost/` pada port 80 dengan isi pesan antrean dalam tubuh. Semua yang aplikasi Anda perlu lakukan adalah melakukan tugas lama berjalan dalam menanggapi POST. Anda dapat mengonfigurasi daemon untuk mengirim ke jalur yang berbeda, menggunakan jenis MIME selain aplikasi/JSON, terhubung ke antrean yang ada, atau menyesuaikan koneksi (permintaan bersamaan maksimum), batas waktu, dan percobaan ulang.



Dengan tugas berkala, Anda juga dapat mengonfigurasi pekerja daemon untuk pesan antrean berdasarkan jadwal cron. Setiap tugas periodik dapat POST ke jalan yang berbeda. Mengaktifkan tugas-tugas periodik dengan memasukkan file YAML dalam kode sumber Anda yang mendefinisikan jadwal dan jalan untuk setiap tugas.

> **Note**
>
> .NET pada platform Windows Server tidak mendukung lingkungan pekerja.

Bagian

- Daemon SQS lingkungan pekerja
- Antrean surat mati
- Tugas periodik

- Gunakan Amazon CloudWatch untuk penskalaan otomatis di tingkat lingkungan pekerja
- Mengonfigurasi lingkungan pekerja

Daemon SQS lingkungan pekerja

Lingkungan pekerja menjalankan proses daemon yang disediakan oleh Elastic Beanstalk. Daemon ini diperbarui secara teratur untuk menambahkan fitur dan memperbaiki bug. Untuk mendapatkan versi terbaru daemon, perbarui ke versi platform terbaru.

Ketika aplikasi di lingkungan pekerja mengembalikan respons `200 OK` untuk mengakui bahwa ia telah menerima dan berhasil memproses permintaan, daemon mengirimkan panggilan `DeleteMessage` ke antrean Amazon SQS untuk menghapus pesan dari antrean. Jika aplikasi mengembalikan respons selain `200 OK`, Elastic Beanstalk menunggu untuk menempatkan pesan kembali antrean setelah periode `ErrorVisibilityTimeout` dikonfigurasi. Jika tidak ada respons, Elastic Beanstalk menunggu untuk menempatkan pesan kembali dalam antrean setelah periode `InactivityTimeout` sehingga pesan tersedia untuk upaya pemrosesan lainnya.

> **Note**
>
> Properti antrian Amazon SQS (urutan pesan, at-least-once pengiriman, dan pengambilan sampel pesan) dapat memengaruhi cara Anda mendesain aplikasi web untuk lingkungan pekerja. Untuk informasi selengkapnya, lihat Properti Antrean Terdistribusi di Panduan Developer Amazon Simple Queue Service.

Amazon SQS secara otomatis menghapus pesan yang telah dalam antrean selama lebih lama daripada `RetentionPeriod` yang dikonfigurasi.

Daemon menetapkan header HTTP berikut.

Header HTTP	
Nama	Nilai
`User-Agent`	`aws-sqsd` `aws-sqsd/1.1 1`

Header HTTP	
`X-Aws-Sqsd-Msgid`	ID pesan SQS, digunakan untuk mendeteksi strom pesan (jumlah pesan baru yang luar biasa tinggi).
`X-Aws-Sqsd-Queue`	Nama antrean SQS.
`X-Aws-Sqsd-First-Received-At`	Waktu UTC, di Format ISO 8601, saat pesan pertama kali diterima.
`X-Aws-Sqsd-Receive-Count`	Pesan SQS menerima hitungan.
`X-Aws-Sqsd-Attr-` *`message-attribute-name`*	Atribut pesan khusus yang ditetapkan ke pesan yang sedang diproses. `message-attribute-name` adalah nama atribut pesan yang sebenarnya. Semua string dan nomor atribut pesan ditambahkan ke header. Atribut biner dibuang dan tidak termasuk dalam header.
`Content-Type`	Konfigurasi tipe mime; secara default, `application/json` .

Antrean surat mati

Lingkungan pekerja Elastic Beanstalk mendukung antrean surat mati Amazon Simple Queue Service (Amazon SQS). Antrean surat mati adalah antrean di mana antrian (sumber) lainnya dapat mengirim pesan yang untuk beberapa alasan tidak dapat berhasil diproses. Manfaat utama menggunakan antrean surat mati adalah kemampuan untuk mengesampingkan dan mengisolasi pesan tidak berhasil diproses. Anda kemudian dapat menganalisis pesan yang dikirim ke antrean surat mati untuk mencoba menentukan mengapa pesan tersebut tidak berhasil diproses.

Jika Anda menentukan antrean Amazon SQS otomatis pada saat Anda membuat tingkat lingkungan pekerja Anda, antrean surat mati diaktifkan secara default untuk lingkungan pekerja. Jika Anda memilih antrean SQS yang ada untuk lingkungan pekerja Anda, Anda harus menggunakan SQS untuk mengonfigurasi antrean surat mati secara independen. Untuk informasi tentang cara

menggunakan SQS untuk mengonfigurasi antrean surat mati, lihat Menggunakan Antrean Surat Mati Amazon SQS.

Anda tidak dapat menonaktifkan antrean surat mati. Pesan yang tidak dapat dikirim selalu akhirnya dikirim ke antrean surat mati. Namun, Anda dapat menonaktifkan fitur ini secara efektif dengan menetapkan pilihan `MaxRetries` untuk nilai maksimum yang valid 100.

Jika antrean surat mati tidak dikonfigurasi untuk antrean Amazon SQS lingkungan pekerja Anda, Amazon SQS menyimpan pesan di antrean sampai periode penyimpanan berakhir. Untuk rincian tentang cara mengonfigurasi periode penyimpanan, lihat the section called “Mengonfigurasi lingkungan pekerja”.

> **Note**
>
> Pilihan Elastic Beanstalk `MaxRetries` setara dengan pilihan SQS `MaxReceiveCount`. Jika lingkungan pekerja Anda tidak menggunakan antrean SQS otomatis, gunakan pilihan `MaxReceiveCount` di SQS untuk secara efektif menonaktifkan antrean surat mati Anda. Untuk informasi selengkapnya, lihat Menggunakan antrean surat mati Amazon SQS.

Untuk informasi lebih lanjut tentang siklus hidup pesan SQS, silakan kunjungi Pesan Siklus hidup.

Tugas periodik

Anda dapat menentukan tugas-tugas periodik dalam sebuah file bernama `cron.yaml` dalam paket sumber Anda untuk menambahkan lowongan ke antrean lingkungan pekerja Anda secara otomatis pada interval reguler.

Misalnya, yang berikut ini file `cron.yaml` membuat dua tugas periodik. Yang pertama berjalan setiap 12 jam dan yang kedua berjalan pada pukul 11 PM UTC setiap hari.

Example cron.yaml

```
version: 1
cron:
 - name: "backup-job"
   url: "/backup"
   schedule: "0 */12 * * *"
 - name: "audit"
   url: "/audit"
```

```
  schedule: "0 23 * * *"
```

`name` harus unik untuk setiap tugas. URL adalah jalur dimana permintaan POST dikirim untuk memicu pekerjaan. Jadwalnya adalah ekspresi CRON yang menentukan kapan tugas berjalan.

Ketika tugas berjalan, daemon memposting pesan ke lingkungan antrean SQS dengan header menunjukkan pekerjaan yang perlu dilakukan. Setiap instans di lingkungan dapat mengambil pesan dan memproses pekerjaan.

> **Note**
>
> Jika Anda mengonfigurasi lingkungan pekerja Anda dengan antrean SQS yang ada dan memilih antrean Amazon SQS FIFO, tugas periodik tidak didukung.

Elastic Beanstalk menggunakan pemilihan pemimpin untuk menentukan instans di lingkungan pekerja Anda yang mengantre tugas periodik. Setiap instans mencoba untuk menjadi pemimpin dengan menulis ke tabel Amazon DynamoDB. Intans pertama yang berhasil adalah pemimpin, dan harus terus menulis ke meja untuk mempertahankan status pemimpin. Jika pemimpin keluar dari layanan, instans lainnya dengan cepat mengambil tempatnya.

Untuk tugas berkala, daemon pekerja menetapkan header tambahan berikut.

Header HTTP	
Nama	Nilai
`X-Aws-Sqsd-Taskname`	Untuk tugas berkala, nama tugas yang harus dilakukan.
`X-Aws-Sqsd-Scheduled-At`	Waktu di mana tugas periodik dijadwalkan
`X-Aws-Sqsd-Sender-Id`	AWS nomor rekening pengirim pesan

Gunakan Amazon CloudWatch untuk penskalaan otomatis di tingkat lingkungan pekerja

Bersama-sama, Amazon EC2 Auto Scaling dan CloudWatch memantau pemanfaatan CPU dari instance yang berjalan di lingkungan pekerja. Bagaimana Anda mengonfigurasi batas penskalaan otomatis untuk kapasitas CPU menentukan berapa banyak instans grup Auto Scaling berjalan dengan tepat mengelola throughput pesan dalam antrean Amazon SQS. Setiap EC2 instance menerbitkan metrik pemanfaatan CPU-nya ke. CloudWatch Amazon EC2 Auto Scaling mengambil dari penggunaan CPU rata-rata CloudWatch di semua instans di lingkungan pekerja. Anda mengonfigurasi ambang atas dan bawah serta berapa banyak instans untuk menambah atau mengakhiri sesuai dengan kapasitas CPU. Saat Amazon EC2 Auto Scaling mendeteksi bahwa Anda telah mencapai ambang batas atas yang ditentukan pada kapasitas CPU, Elastic Beanstalk membuat instance baru di lingkungan pekerja. Intans dihapus ketika beban CPU turun kembali di bawah ambang batas.

> **Note**
>
> Pesan yang belum diproses pada saat sebuah instans yang dihentikan dikembalikan ke antrean di mana mereka dapat diproses oleh daemon lain pada instans yang masih berjalan.

Anda juga dapat mengatur CloudWatch alarm lain, sesuai kebutuhan, dengan menggunakan konsol Elastic Beanstalk, CLI, atau file opsi. Untuk informasi lebih lanjut, lihat Menggunakan Elastic Beanstalk dengan Amazon CloudWatch dan Membuat grup Auto Scaling dengan Kebijakan Penskalaan Langkah.

Mengonfigurasi lingkungan pekerja

Anda dapat mengelola konfigurasi lingkungan pekerja dengan mengedit kategori Pekerja pada halaman Konfigurasi di konsol manajemen lingkungan.

Elastic Beanstalk > Environments > GettingStartedApp-env > Configuration

Modify worker

You can create a new Amazon SQS queue for your worker application or pull work items from an existing queue. The worker daemon on the instances in your environment pulls an item from the queue and relays it in the body of a POST request to a local HTTP path relative to localhost.

Queue

Worker queue

Autogenerated queue

SQS queue from which to read work items.

Messages

HTTP path

/

The daemon pulls items from the Amazon SQS queue and posts them locally to this path.

MIME type

application/json

Change the MIME type of the POST requests that the worker daemon sends to your application.

HTTP connections

50

Maximum number of concurrent connections to the application.

Visibility timeout

300 seconds

The amount of time to lock an incoming message for processing before returning it to the queue.

Error visibility timeout

seconds

The amount of time to wait before resending a message after an error response from the application.

Advanced options

The following settings control advanced behavior of the worker tier daemon. Learn more

Max retries

10

Maximum number of retries after which the message is discarded.

Connection timeout

5

Inactivity timeout

> **Note**
>
> Anda dapat mengonfigurasi jalur URL untuk mengirimkan pesan antrean pekerja, namun Anda tidak dapat mengonfigurasi port IP. Elastic Beanstalk selalu memposting pesan antrean pekerja di port 80. Aplikasi lingkungan pekerja atau proksi harus mendengarkan port 80.

Untuk mengonfigurasikan daemon worker

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Dalam kategori konfigurasi Pekerja, pilih Edit.

Halaman konfigurasi Memodifikasi pekerja memiliki opsi berikut.

Di bagian Antre:

- Antrean Pekerja – Tentukan antrean Amazon SQS dari mana daemon membaca. Jika Anda memilikinya, Anda dapat memilih antrean yang ada. Jika Anda memilih Antrean otomatis, Elastic Beanstalk membuat antrean Amazon SQS baru dan sesuai URL antrean pekerja.

  > **Note**
  >
  > Bila Anda memilih Antrean otomatis, antrean yang Elastic Beanstalk buat adalah antrean Amazon SQS standar. Bila Anda memilih antrean yang ada, Anda dapat memberikan standar atau antrean Amazon SQS FIFO. Ketahuilah bahwa jika Anda memberikan antrean FIFO, tugas berkala tidak didukung.

- URL antrean pekerja – Jika Anda memilihAntrean pekerja yang ada, pengaturan ini menampilkan URL yang terkait dengan antrean Amazon SQS.

Di bagian Pesan:

- Jalur HTTP – Tentukan jalur relatif ke aplikasi yang akan menerima data dari antrean Amazon SQS. Data dimasukkan ke dalam badan pesan dari pesan HTTP POST. Nilai default-nya adalah `/`.

- Jenis MIME – menunjukkan tipe MIME yang digunakan oleh pesan HTTP POST. Nilai default-nya adalah `application/json`. Namun, nilai apa pun berlaku karena Anda dapat membuat dan kemudian menentukan jenis MIME Anda sendiri.
- Koneksi HTTP - Tentukan jumlah maksimum koneksi bersamaan yang dapat dibuat daemon ke aplikasi apa pun dalam instance Amazon. EC2 Nilai default-nya **`50`**. Anda dapat menentukan **`1`** ke **`100`**.
- Waktu habis – Tunjukkan jumlah waktu, dalam hitungan detik, pesan masuk dari antrean Amazon SQS terkunci untuk diproses. Setelah jumlah waktu yang dikonfigurasi telah berlalu, pesan kembali dibuat terlihat dalam antrean untuk membaca daemon lain. Memilih nilai yang lebih panjang dari yang Anda harapkan aplikasi Anda butuhkan untuk memproses pesan, hingga **`43200`** detik.
- Batas waktu visibilitas kesalahan – Menunjukkan jumlah waktu, dalam detik, yang berlalu sebelum Elastic Beanstalk mengembalikan pesan ke antrean Amazon SQS setelah upaya untuk memproses gagal dengan kesalahan eksplisit. Anda dapat menentukan **`0`** ke **`43200`** detik.

Di bagian Pilihan lanjutan:

- Coba lagi Maks – Tentukan jumlah maksimum kali Elastic Beanstalk mencoba untuk mengirim pesan ke antrean Amazon SQS sebelum memindahkan pesan ke antrean surat mati. Nilai default-nya adalah **`10`**. Anda dapat menentukan **`1`** ke **`100`**.

> **Note**
>
> Opsi percobaan ulang Max hanya berlaku untuk antrian Amazon SQS yang dikonfigurasi dengan antrian huruf mati. Untuk antrian Amazon SQS apa pun yang tidak dikonfigurasi dengan antrean huruf mati, Amazon SQS menyimpan pesan dalam antrian dan memprosesnya hingga periode yang ditentukan oleh opsi Periode penyimpanan berakhir.

- Koneksi waktu habis – Tunjukkan jumlah waktu, dalam hitungan detik, untuk menunggu koneksi sukses ke aplikasi. Nilai default-nya adalah **`5`**. Anda dapat menentukan **`1`** ke **`60`** detik.
- Waktu habis tidak aktif – Tunjukkan jumlah waktu, dalam detik, untuk menunggu respons pada koneksi yang ada ke aplikasi. Nilai default-nya adalah **`180`**. Anda dapat menentukan **`1`** ke **`36000`** detik.
- Periode penahanan – Tunjukkan jumlah waktu, dalam hitungan detik, pesan valid dan diproses secara aktif. Nilai default-nya adalah **`345600`**. Anda dapat menentukan **`60`** ke **`1209600`** detik.

Jika Anda menggunakan antrean Amazon SQS yang ada, pengaturan yang Anda mengonfigurasi ketika Anda membuat lingkungan pekerja dapat konflik dengan pengaturan Anda dikonfigurasi secara langsung di Amazon SQS. Sebagai contoh, jika Anda mengonfigurasi lingkungan pekerja dengan nilai `RetentionPeriod` yang lebih tinggi dari nilai `MessageRetentionPeriod` yang Anda tetapkan di Amazon SQS, Amazon SQS menghapus pesan ketika melebihi `MessageRetentionPeriod`.

Sebaliknya, jika nilai `RetentionPeriod` yang Anda mengonfigurasi dalam pengaturan lingkungan pekerja lebih rendah dari nilai `MessageRetentionPeriod` yang Anda tetapkan di Amazon SQS, daemon menghapus pesan sebelum Amazon SQS dapat menghapusnya. Untuk `VisibilityTimeout`, nilai yang Anda mengonfigurasi untuk daemon dalam pengaturan lingkungan pekerja menimpa pengaturan Amazon SQS `VisibilityTimeout`. Pastikan bahwa pesan dihapus tepat dengan membandingkan pengaturan Elastic Beanstalk Anda untuk pengaturan Amazon SQS Anda.

Membuat tautan antara lingkungan Elastic Beanstalk

Karena aplikasi Anda tumbuh dalam ukuran dan kompleksitas, Anda mungkin ingin membaginya menjadi komponen yang memiliki perkembangan dan siklus operasional yang berbeda. Dengan menjalankan layanan yang lebih kecil yang berinteraksi satu sama lain melalui antarmuka yang terdefinisi dengan baik, tim dapat bekerja secara independen dan penerapan dapat mengurangi resiko. AWS Elastic Beanstalk memungkinkan Anda menautkan lingkungan Anda untuk berbagi informasi antar komponen yang bergantung satu sama lain.

> **Note**
>
> Elastic Beanstalk saat ini mendukung tautan lingkungan untuk semua platform kecuali Multicontainer Docker.

Dengan tautan lingkungan, Anda dapat menentukan hubungan antara lingkungan komponen aplikasi Anda sebagai referensi bernama. Ketika Anda membuat lingkungan yang mendefinisikan tautan, Elastic Beanstalk menetapkan variabel lingkungan dengan nama yang sama dengan tautan. Nilai variabel adalah titik akhir yang dapat Anda gunakan untuk terhubung ke komponen lain, yang dapat menjadi server web atau lingkungan pekerja.

Misalnya, jika aplikasi Anda terdiri dari frontend yang mengumpulkan alamat email dan pekerja yang mengirimkan email selamat datang ke alamat email yang dikumpulkan oleh frontend, Anda dapat

membuat tautan ke pekerja tersebut di frontend Anda dan memiliki frontend yang secara otomatis menemukan titik akhir (URL antrean) untuk pekerja Anda.

Tentukan tautan ke lingkungan lain dalam manifes lingkungan, sebuah file berformat YAML yang dinamai `env.yaml` di akar sumber aplikasi Anda. Manifes berikut mendefinisikan tautan ke lingkungan bernama pekerja:

`~/workspace/my-app/frontend/env.yaml`

```
AWSConfigurationTemplateVersion: 1.1.0.0
EnvironmentLinks:
  "WORKERQUEUE": "worker"
```

Ketika Anda membuat lingkungan dengan versi aplikasi yang mencakup manifes lingkungan di atas, Elastic Beanstalk mencari lingkungan bernama `worker` milik aplikasi yang sama. Jika lingkungan itu ada, Elastic Beanstalk membuat properti lingkungan bernama `WORKERQUEUE`. Nilai dari `WORKERQUEUE` adalah URL antrean Amazon SQS. Aplikasi frontend dapat membaca properti ini dengan cara yang sama sebagai variabel lingkungan. Lihat Manifes lingkungan (`env.yaml`) untuk rincian selengkapnya.

Untuk menggunakan tautan lingkungan, tambahkan manifes lingkungan ke sumber aplikasi Anda dan unggah dengan EB CLI AWS CLI , atau SDK. Jika Anda menggunakan AWS CLI atau SDK, setel `process` tanda saat Anda memanggil`CreateApplicationVersion`:

```
$ aws elasticbeanstalk create-application-version --process --application-name
 my-app --version-label frontend-v1 --source-bundle S3Bucket="amzn-s3-demo-
bucket",S3Key="front-v1.zip"
```

Opsi ini memberitahu Elastic Beanstalk untuk memvalidasi manifes lingkungan dan file konfigurasi dalam paket sumber Anda ketika Anda membuat versi aplikasi. EB CLI mengatur penanda ini secara otomatis ketika Anda memiliki manifes lingkungan pada direktori proyek Anda.

Buat lingkungan Anda secara normal menggunakan klien apa pun. Ketika Anda perlu untuk mengakhiri lingkungan, akhiri lingkungan dengan tautan pertama. Jika lingkungan ditautkan dengan lingkungan lain, Elastic Beanstalk akan mencegah lingkungan yang ditautkan tersebut dihentikan. Untuk mengesampingkan perlindungan ini, gunakan penanda `ForceTerminate`. Parameter ini tersedia di AWS CLI sebagai `--force-terminate`:

```
$ aws elasticbeanstalk terminate-environment --force-terminate --environment-name
 worker
```

Memulihkan lingkungan Elastic Beanstalk Anda dari keadaan tidak valid

Topik ini memberikan beberapa informasi latar belakang dan sumber daya yang menjelaskan cara memecahkan masalah lingkungan Elastic Beanstalk dalam keadaan tidak valid.

Mengatasi kesalahan

Operasi standar pada lingkungan dalam keadaan tidak valid tidak akan berhasil diselesaikan. Operasi yang gagal akan mengembalikan kesalahan yang mencakup teks berikut:

```
The stack stack_id associated with environment environment-ID is in stack-status state.
```

Untuk memecahkan masalah dan mengatasi kesalahan ini, lihat artikel Pusat Pengetahuan Mengapa lingkungan Elastic Beanstalk saya dalam keadaan tidak valid? .

> **Note**
>
> Sebelum 16 Desember 2024, operasi yang gagal mengembalikan kesalahan berikut: `Environment is in an invalid state for this operation. Must be ready.` Dalam hal ini Anda harus menghubungi AWS Support untuk mengatur ulang status lingkungan setelah Anda menyelesaikan tindakan korektif.
> Hari ini Anda masih harus menyelesaikan masalah tumpukan mengikuti instruksi dalam artikel Pusat Pengetahuan yang direferensikan. Namun, setelah Anda berhasil menyelesaikan tindakan korektif, Elastic Beanstalk secara otomatis memperbarui status lingkungan dari tidak valid menjadi tersedia, dan Anda dapat melanjutkan operasi standar di lingkungan Anda tanpa penundaan lebih lanjut.

Mengapa kesalahan terjadi

Saat Anda menerapkan aplikasi di Elastic Beanstalk, layanan akan membuat tumpukan yang mendasarinya. AWS CloudFormation Elastic Beanstalk AWS CloudFormation memanggil layanan untuk meluncurkan sumber daya di lingkungan Anda dan menyebarkan perubahan konfigurasi.

Jika Elastic Beanstalk melakukan operasi pada lingkungan tanpa memiliki akses ke sumber daya yang diperlukan, tumpukan yang mendasari lingkungan dapat memasuki CloudFormation status gagal. Masalah lain juga dapat menyebabkan keadaan ini, meskipun masalah izin adalah penyebab utamanya. Sebagai hasil dari status gagal tumpukan, AWS CloudFormation memblokir permintaan operasi Elastic Beanstalk agar tidak melakukan pembaruan tumpukan lebih lanjut, menyebabkan kegagalan operasi Elastic Beanstalk, seperti dan. UpdateEnvironment RetrieveEnvironmentInfo

Pada titik ini Anda harus terlebih dahulu memperbaiki akar penyebab masalah mendasar untuk memperbaiki CloudFormation tumpukan. Layanan Elastic Beanstalk kemudian CloudFormation mendeteksi perubahan status tumpukan dan menindaklanjuti untuk mengatur ulang lingkungan Anda ke status yang tersedia. Pada titik ini operasi lebih lanjut dapat diselesaikan dengan sukses.

Masalah izin biasanya menyebabkan efek ini pada CloudFormation tumpukan dan lingkungan Elastic Beanstalk out-of-band, meskipun perubahan juga dapat menyebabkan masalah.

> **Important**
>
> Untuk menghindari gangguan pada lingkungan Anda, kami sangat menyarankan agar Anda hanya memulai operasi untuk mengelola dan mengonfigurasi lingkungan Anda dari layanan Elastic Beanstalk. Modifikasi sumber daya dengan menggunakan konsol, perintah CLI, atau SDK layanan selain Elastic Beanstalk adalah perubahan, yang menyebabkan penyimpangan sumber daya. out-of-band Penyimpangan sumber daya mempengaruhi status CloudFormation tumpukan, yang pada gilirannya menyebabkan lingkungan Elastic Beanstalk masuk ke dalam keadaan tidak valid.
> Untuk informasi selengkapnya tentang penyimpangan sumber daya, lihat Apa itu drift? dalam AWS CloudFormation User Guide.

Mengonfigurasi lingkungan Elastic Beanstalk

Topik ini berfokus pada opsi konfigurasi yang tersedia di konsol Elastic Beanstalk. AWS Elastic Beanstalk menyediakan berbagai pilihan untuk menyesuaikan sumber daya di lingkungan Anda, bersama dengan perilaku Elastic Beanstalk dan pengaturan platform.

Topik berikut menunjukkan cara mengonfigurasi lingkungan Anda di konsol. Mereka juga mendeskripsikan namespace yang mendasari yang sesuai dengan pilihan konsol untuk digunakan dengan file konfigurasi atau opsi konfigurasi API. Untuk mempelajari tentang metode konfigurasi lanjutan, lihat Mengonfigurasi lingkungan (lanjutan).

Topik

- Sumber daya yang disediakan
- Konfigurasi lingkungan menggunakan konsol Elastic Beanstalk
- EC2 Contoh Amazon untuk lingkungan Elastic Beanstalk Anda
- Auto Scaling instance lingkungan Elastic Beanstalk Anda
- Penyeimbang beban untuk lingkungan Elastic Beanstalk Anda
- Menambahkan basis data ke lingkungan Elastic Beanstalk Anda
- Keamanan AWS Elastic Beanstalk lingkungan Anda
- Menandai sumber daya di lingkungan Elastic Beanstalk Anda
- Variabel lingkungan dan pengaturan perangkat lunak lainnya
- Pemberitahuan lingkungan Elastic Beanstalk dengan Amazon SNS
- Mengonfigurasi Amazon Virtual Private Cloud (Amazon VPC) dengan Elastic Beanstalk
- Nama domain lingkungan Elastic Beanstalk Anda

Sumber daya yang disediakan

Saat Anda membuat lingkungan server web, Elastic Beanstalk menciptakan beberapa sumber daya untuk mendukung pengoperasian aplikasi Anda. Bab ini menjelaskan cara menyesuaikan sumber daya ini untuk lingkungan Elastic Beanstalk Anda.

- EC2 instance — Mesin virtual Amazon Elastic Compute Cloud (Amazon EC2) yang dikonfigurasi untuk menjalankan aplikasi web pada platform yang Anda pilih.

Setiap platform menjalankan satu set perangkat lunak, file konfigurasi dan penulisan tertentu untuk mendukung versi bahasa, kerangka kerja, kontainer web tertentu, atau kombinasi dari semua ini. Sebagian besar platform menggunakan Apache atau NGINX sebagai proksi terbalik yang ada di depan aplikasi web Anda, meneruskan permintaan ke aplikasi web, menyajikan aset statis, dan menghasilkan log akses dan kesalahan.

- Grup keamanan instans — Grup EC2 keamanan Amazon yang dikonfigurasi untuk memungkinkan lalu lintas masuk pada port 80. Sumber daya ini memungkinkan lalu lintas HTTP dari penyeimbang beban mencapai EC2 instance yang menjalankan aplikasi web Anda. Secara default, lalu lintas tidak diizinkan di port lain.
- Penyeimbang beban – Penyeimbang beban Elastic Load Balancing yang dikonfigurasi untuk mendistribusikan permintaan ke instans yang menjalankan aplikasi Anda. Penyeimbang beban juga menghilangkan kebutuhan untuk mengekspos instans Anda langsung ke internet.
- Grup keamanan penyeimbang beban - Grup EC2 keamanan Amazon yang dikonfigurasi untuk memungkinkan lalu lintas masuk di port 80. Sumber daya ini memungkinkan lalu lintas HTTP dari internet mencapai penyeimbang beban. Secara default, lalu lintas tidak diizinkan di port lain.
- Grup Auto Scaling – Grup Auto Scaling yang dikonfigurasi untuk menggantikan instans jika diakhiri atau menjadi tidak tersedia.
- Bucket Amazon S3 – Lokasi penyimpanan untuk kode sumber, log, dan artifact lainnya yang dibuat saat Anda menggunakan Elastic Beanstalk.
- CloudWatch Alarm Amazon — Dua CloudWatch alarm yang memantau beban pada instans di lingkungan Anda dan yang dipicu jika beban terlalu tinggi atau terlalu rendah. Saat alarm terpicu, grup Auto Scaling Anda akan menaikkan atau menurunkan skala sebagai respons.
- AWS CloudFormation stack - Elastic AWS CloudFormation Beanstalk digunakan untuk meluncurkan sumber daya di lingkungan Anda dan menyebarkan perubahan konfigurasi. Sumber daya ditentukan di sebuah templat yang dapat Anda lihat di Konsol AWS CloudFormation.
- Nama domain — Nama domain yang merutekan ke aplikasi web Anda dalam formulir *`subdomain`*.*`region`*.elasticbeanstalk.com.

> **Keamanan domain**
>
> Untuk meningkatkan keamanan aplikasi Elastic Beanstalk Anda, domain elasticbeanstalk.com terdaftar di Daftar Akhiran Publik (PSL).
> Jika Anda perlu mengatur cookie sensitif di nama domain default untuk aplikasi Elastic Beanstalk Anda, kami sarankan Anda menggunakan cookie `__Host-` dengan awalan untuk meningkatkan keamanan. Praktik ini melindungi domain Anda dari upaya pemalsuan

permintaan lintas situs (CSRF). Untuk informasi selengkapnya, lihat halaman Set-Cookie di Jaringan Pengembang Mozilla.

Konfigurasi lingkungan menggunakan konsol Elastic Beanstalk

Topik ini menguraikan opsi konfigurasi yang tersedia melalui konsol Elastic Beanstalk dan menjelaskan cara menavigasi halaman konfigurasi.

Untuk melihat ringkasan konfigurasi lingkungan Anda

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.

Halaman konfigurasi

Halaman Gambaran umum konfigurasi menampilkan satu set kategori konfigurasi. Setiap kategori konfigurasi mengelompokkan satu set opsi terkait.

Akses layanan

Opsi dalam kategori ini memilih peran layanan dan profil EC2 instance yang digunakan Elastic Beanstalk untuk mengelola lingkungan Anda. Secara opsional pilih EC2 key pair untuk login dengan aman ke instans Anda EC2 .

Jaringan dan database

Opsi dalam kategori ini mengonfigurasi pengaturan VPC, dan subnet untuk EC2 instance lingkungan dan penyeimbang beban. Mereka juga menyediakan opsi untuk menyiapkan database Amazon RDS yang terintegrasi dengan lingkungan Anda.

Lalu lintas dan penskalaan instans

Opsi ini menyesuaikan kapasitas, penskalaan, dan penyeimbangan beban untuk instans lingkungan. EC2 Anda juga dapat mengonfigurasi Elastic Load Balancing untuk menangkap log dengan informasi rinci tentang permintaan yang dikirim ke penyeimbang beban.

Opsi berikut untuk EC2 instans Anda juga tersedia untuk konfigurasi:

- Jenis volume root, ukuran, laju input/output operasi (IOPS), dan throughput.
- Mengaktifkan layanan metadata instance (IMDS).
- Pemilihan kelompok EC2 keamanan untuk mengontrol lalu lintas instance.
- CloudWatch interval pemantauan metrik.
- Interval waktu untuk pencatatan metrik.

Pembaruan, pemantauan, dan pencatatan

Kategori ini mengonfigurasi opsi berikut:

- Pelaporan kesehatan lingkungan, termasuk opsi untuk memilih pelaporan kesehatan yang ditingkatkan.
- Pembaruan platform terkelola yang menentukan kapan dan bagaimana Elastic Beanstalk menyebarkan perubahan ke lingkungan.
- Mengaktifkan layanan X-Ray untuk mengumpulkan data tentang perilaku aplikasi Anda untuk mengidentifikasi masalah dan peluang pengoptimalan.
- Opsi khusus platform, termasuk server proxy dan properti lingkungan OS.

Menavigasi halaman konfigurasi

Pilih Edit dalam kategori konfigurasi untuk meluncurkan halaman konfigurasi terkait, di mana Anda dapat melihat nilai opsi lengkap dan membuat perubahan.

Navigasi dalam kategori konfigurasi

Navigasikan di halaman kategori konfigurasi dengan salah satu tindakan berikut:

- Batal — Kembali ke halaman ikhtisar Konfigurasi tanpa menerapkan perubahan konfigurasi Anda. Saat Anda memilih Batal, konsol kehilangan setiap perubahan tertunda yang Anda buat pada setiap kategori konfigurasi.

  Anda juga dapat membatalkan perubahan konfigurasi dengan memilih item lain di halaman navigasi kiri, seperti Peristiwa atau Log.
- Lanjutkan – Kembali ke halaman Gambaran umum konfigurasi. Anda kemudian dapat lanjut membuat perubahan atau menerapkan yang tertunda.

- Terapkan - Terapkan perubahan yang Anda buat di salah satu kategori konfigurasi ke lingkungan Anda. Dalam beberapa hal, Anda diminta untuk mengonfirmasi konsekuensi dari salah satu keputusan konfigurasi Anda.

Navigasi pada halaman ikhtisar Konfigurasi

Pilih Edit dalam kategori konfigurasi untuk meluncurkan halaman konfigurasi terkait, di mana Anda dapat melihat nilai opsi lengkap dan membuat perubahan. Setelah selesai melihat dan memodifikasi opsi, Anda dapat memilih salah satu tindakan berikut dari halaman ikhtisar Konfigurasi:

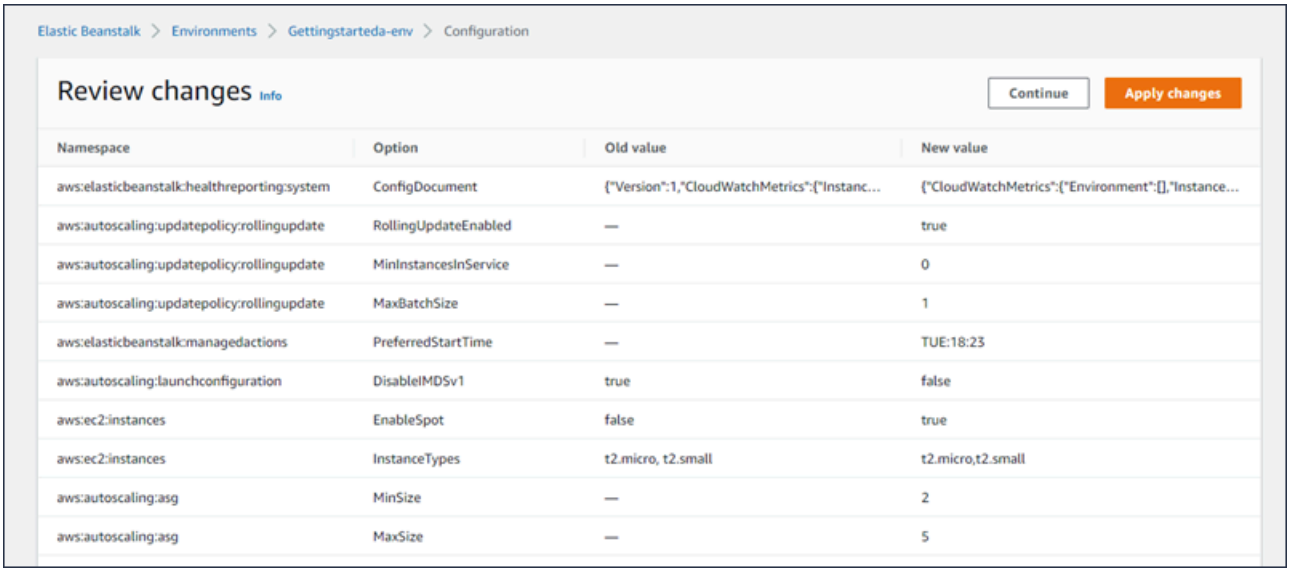
- Cancel – Kembali ke dasbor lingkungan tanpa menerapkan perubahan konfigurasi Anda. Saat Anda memilih Batal, konsol kehilangan setiap perubahan tertunda yang Anda buat pada setiap kategori konfigurasi.

  Anda juga dapat membatalkan perubahan konfigurasi dengan memilih item lain di halaman navigasi kiri, seperti Peristiwa atau Log.
- Tinjau perubahan – Dapatkan ringkasan tentang semua perubahan tertunda yang Anda buat di salah satu kategori konfigurasi. Lihat perinciannya di Tinjau halaman perubahan.
- Terapkan perubahan — Terapkan perubahan yang Anda buat di salah satu kategori konfigurasi ke lingkungan Anda. Dalam beberapa hal, Anda diminta untuk mengonfirmasi konsekuensi dari salah satu keputusan konfigurasi Anda.

Tinjau halaman perubahan

Halaman Tinjau Perubahan menampilkan tabel yang menunjukkan semua perubahan opsi tertunda yang Anda buat di salah satu kategori konfigurasi dan belum diterapkan ke lingkungan Anda.

Tabel mencantumkan setiap opsi sebagai kombinasi dari Namespace dan Option yang dengannya Elastic Beanstalk mengidentifikasinya. Lihat perinciannya di Opsi konfigurasi.



Setelah selesai meninjau perubahan, Anda dapat memilih salah satu tindakan berikut:

- Lanjutkan – Kembali ke halaman Gambaran umum konfigurasi. Anda kemudian dapat lanjut membuat perubahan atau menerapkan yang tertunda.
- Terapkan perubahan — Terapkan perubahan yang Anda buat di salah satu kategori konfigurasi ke lingkungan Anda. Dalam beberapa hal, Anda diminta untuk mengonfirmasi konsekuensi dari salah satu keputusan konfigurasi Anda.

EC2 Contoh Amazon untuk lingkungan Elastic Beanstalk Anda

Topik ini menjelaskan EC2 instans Amazon dan opsi konfigurasi yang memengaruhi lingkungan Elastic Beanstalk Anda.

Saat Anda membuat lingkungan server web, AWS Elastic Beanstalk buat satu atau beberapa mesin virtual Amazon Elastic Compute Cloud (Amazon EC2), yang dikenal sebagai Instans.

Instans di lingkungan Anda dikonfigurasi untuk menjalankan aplikasi web pada platform yang Anda pilih. Anda dapat membuat perubahan pada berbagai properti dan perilaku instance lingkungan Anda ketika Anda membuat lingkungan Anda atau setelah itu sudah berjalan. Atau, Anda sudah dapat membuat perubahan ini dengan memodifikasi kode sumber yang Anda terapkan ke lingkungan. Untuk informasi lebih lanjut, lihatthe section called “Opsi konfigurasi”.

> **Note**
>
> Grup Auto Scaling di lingkungan Anda mengelola EC2 instans Amazon yang menjalankan aplikasi Anda. Saat Anda membuat perubahan konfigurasi yang dijelaskan dalam topik ini, konfigurasi peluncuran juga berubah. Konfigurasi peluncuran adalah template EC2 peluncuran Amazon atau sumber daya konfigurasi peluncuran grup Auto Scaling. Perubahan ini membutuhkan penggantian semua instance. Ini juga memicu pembaruan bergulir atau pembaruan yang tidak dapat diubah, tergantung pada mana yang dikonfigurasi.

EC2 opsi pembelian contoh

Elastic Beanstalk mendukung EC2 beberapa opsi pembelian instans Amazon:

- On-Demand — Instans On-Demand adalah pay-as-you-go sumber daya—tidak ada komitmen jangka panjang yang diperlukan saat Anda menggunakannya.
- Cadangan — Instans Cadangan adalah diskon penagihan pra-pembelian yang diterapkan secara otomatis untuk mencocokkan instans Sesuai Permintaan di lingkungan Anda.
- Spot — Instans Spot adalah instans Amazon EC2 yang tidak digunakan yang tersedia dengan harga kurang dari harga On-Demand. Anda dapat mengaktifkan dan mengonfigurasi alokasi Instans Spot di lingkungan Anda. Untuk informasi selengkapnya, lihat Auto Scaling instance lingkungan Elastic Beanstalk Anda.

Topik

- Jenis EC2 instans Amazon
- Mengonfigurasi EC2 instans Amazon menggunakan konsol Elastic Beanstalk
- Mengelola kelompok EC2 keamanan
- Mengonfigurasi grup EC2 keamanan Amazon dan jenis instans menggunakan AWS CLI
- Mengonfigurasi EC2 instans Amazon dengan opsi namespace
- Mengonfigurasi IMDS pada instans lingkungan Elastic Beanstalk Anda

Jenis EC2 instans Amazon

Topik ini menjelaskan istilah tipe instance. Saat Anda membuat lingkungan baru, Elastic Beanstalk menyediakan instans EC2 Amazon yang didasarkan pada jenis instans Amazon yang Anda EC2

pilih. Jenis instans yang Anda pilih menentukan perangkat keras host yang menjalankan instance Anda. EC2 tipe instance dapat dikategorikan berdasarkan arsitektur prosesor mana masing-masing didasarkan. Elastic Beanstalk mendukung jenis instans berdasarkan AWS arsitektur prosesor berikut: Arsitektur Graviton 64-bit Arm (arm64), arsitektur 64-bit (x86), dan arsitektur 32-bit (i386). Elastic Beanstalk memilih arsitektur prosesor x86 secara default saat Anda membuat lingkungan baru.

> **Note**
>
> Arsitektur i386 32-bit tidak lagi didukung oleh sebagian besar platform Elastic Beanstalk. Kami menyarankan Anda memilih jenis arsitektur x86 atau arm64 sebagai gantinya. Elastic Beanstalk menyediakan opsi konfigurasi untuk tipe instans prosesor i386 di namespace. `aws:ec2:instances`

Semua tipe instance dalam konfigurasi untuk lingkungan Elastic Beanstalk tertentu harus memiliki tipe arsitektur prosesor yang sama. Asumsikan Anda menambahkan tipe instance baru ke lingkungan yang sudah ada yang sudah memiliki tipe instans t2.medium, yang didasarkan pada arsitektur x86. Anda hanya dapat menambahkan jenis instance lain dari arsitektur yang sama, seperti t2.small. Jika Anda ingin mengganti jenis instance yang ada dengan yang berasal dari arsitektur yang berbeda, Anda dapat melakukannya. Tetapi pastikan bahwa semua jenis instance dalam perintah didasarkan pada jenis arsitektur yang sama.

Elastic Beanstalk secara teratur menambahkan dukungan untuk jenis instans baru yang kompatibel setelah Amazon memperkenalkannya. EC2 Untuk informasi tentang jenis instans yang tersedia, lihat Jenis instans di Panduan EC2 Pengguna Amazon.

> **Note**
>
> Elastic Beanstalk sekarang menawarkan dukungan untuk Graviton di semua platform Amazon Linux 2 terbaru di semua Wilayah yang didukung Graviton. AWS Untuk informasi selengkapnya tentang membuat lingkungan Elastic Beanstalk dengan tipe instance berbasis arm64, lihat. Mengonfigurasi EC2 instans Amazon menggunakan konsol Elastic Beanstalk Buat lingkungan baru yang menjalankan EC2 instans Amazon pada arsitektur arm64 dan memigrasikan aplikasi yang ada ke dalamnya dengan opsi penerapan di Elastic Beanstalk. Untuk mempelajari lebih lanjut tentang prosesor berbasis Graviton arm64, lihat sumber daya ini: AWS
>
> - Manfaat - Prosesor AWS Graviton

- Memulai dan topik lainnya, seperti pertimbangan khusus bahasa — Memulai artikel Graviton AWS GitHub

Mengonfigurasi EC2 instans Amazon menggunakan konsol Elastic Beanstalk

Anda dapat membuat atau memodifikasi konfigurasi instans Amazon lingkungan Elastic Beanstalk di konsol EC2 Elastic Beanstalk.

> **Note**
>
> Meskipun konsol Elastic Beanstalk tidak menyediakan opsi untuk mengubah arsitektur prosesor dari lingkungan yang ada, Anda dapat melakukannya dengan. AWS CLI Misalnya perintah, lihatMengonfigurasi grup EC2 keamanan Amazon dan jenis instans menggunakan AWS CLI.

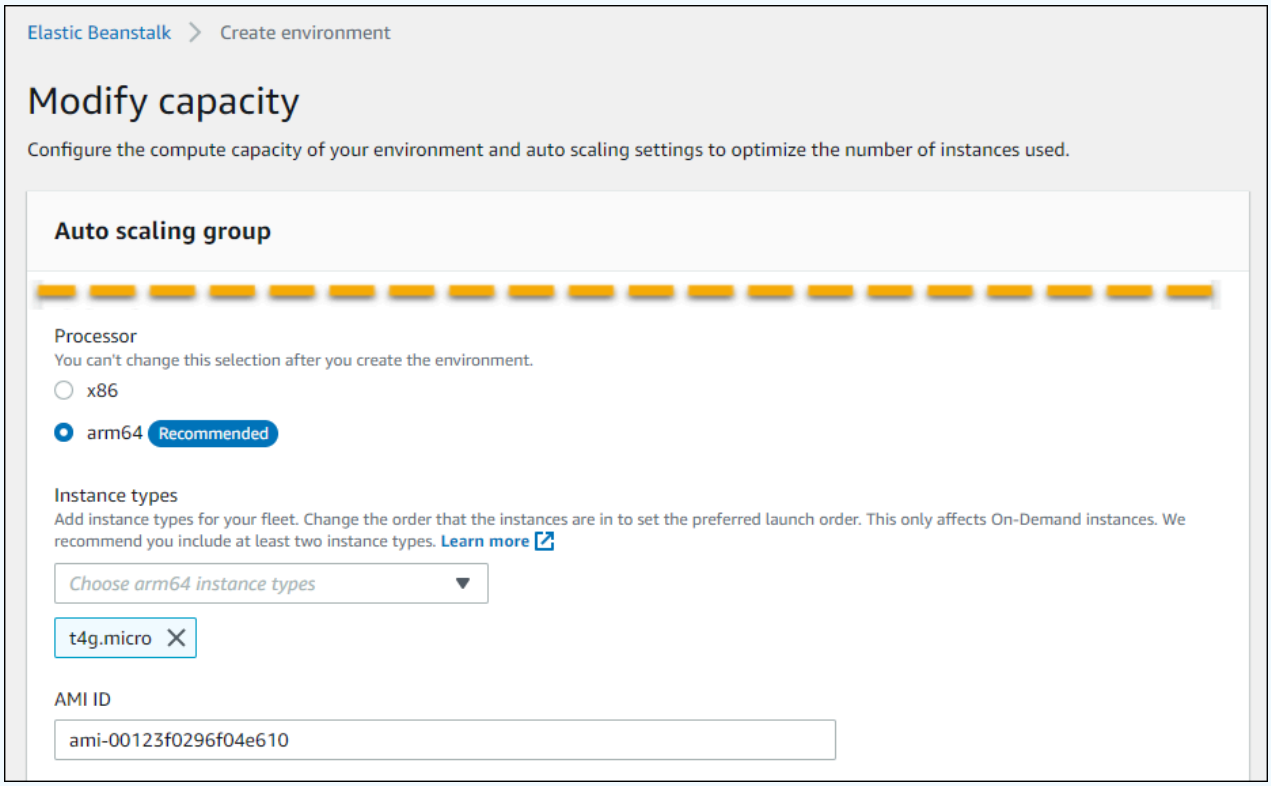
Untuk mengonfigurasi EC2 instans Amazon di konsol Elastic Beanstalk selama pembuatan lingkungan

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Pada panel navigasi, pilih Lingkungan.
3. Pilih Buat lingkungan baru untuk mulai membuat lingkungan Anda.
4. Pada halaman utama wizard, sebelum memilih Buat lingkungan, pilih Konfigurasikan opsi lainnya.
5. Pada kategori konfigurasi Instans, pilih Edit. Buat perubahan pada pengaturan dalam kategori ini, dan kemudian pilih Terapkan. Untuk mengatur deskripsi, lihat bagian the section called “Pengaturan kategori instans” di halaman ini.
6. Pada kategori konfigurasi Kapasitas, pilih Edit. Buat perubahan pada pengaturan dalam kategori ini, dan kemudian pilih Lanjutkan. Untuk mengatur deskripsi, lihat bagian the section called “Pengaturan kategori kapasitas” di halaman ini.

Memilih arsitektur prosesor

Gulir ke bawah ke Processor untuk memilih arsitektur prosesor untuk EC2 instance Anda. Konsol mencantumkan arsitektur prosesor yang didukung oleh platform yang Anda pilih sebelumnya di panel Create environment.

Jika Anda tidak melihat arsitektur prosesor yang Anda butuhkan, kembali ke daftar kategori konfigurasi untuk memilih platform yang mendukungnya. Dari panel Modify Capacity, pilih Cancel. Kemudian, pilih Ubah versi platform untuk memilih pengaturan platform baru. Selanjutnya, dalam kategori konfigurasi Kapasitas pilih Edit untuk melihat pilihan arsitektur prosesor lagi.



7. Pilih Simpan, dan kemudian buat perubahan konfigurasi lain yang diperlukan lingkungan Anda.
8. Pilih Buat lingkungan.

Untuk mengonfigurasi EC2 instans Amazon lingkungan berjalan di konsol Elastic Beanstalk

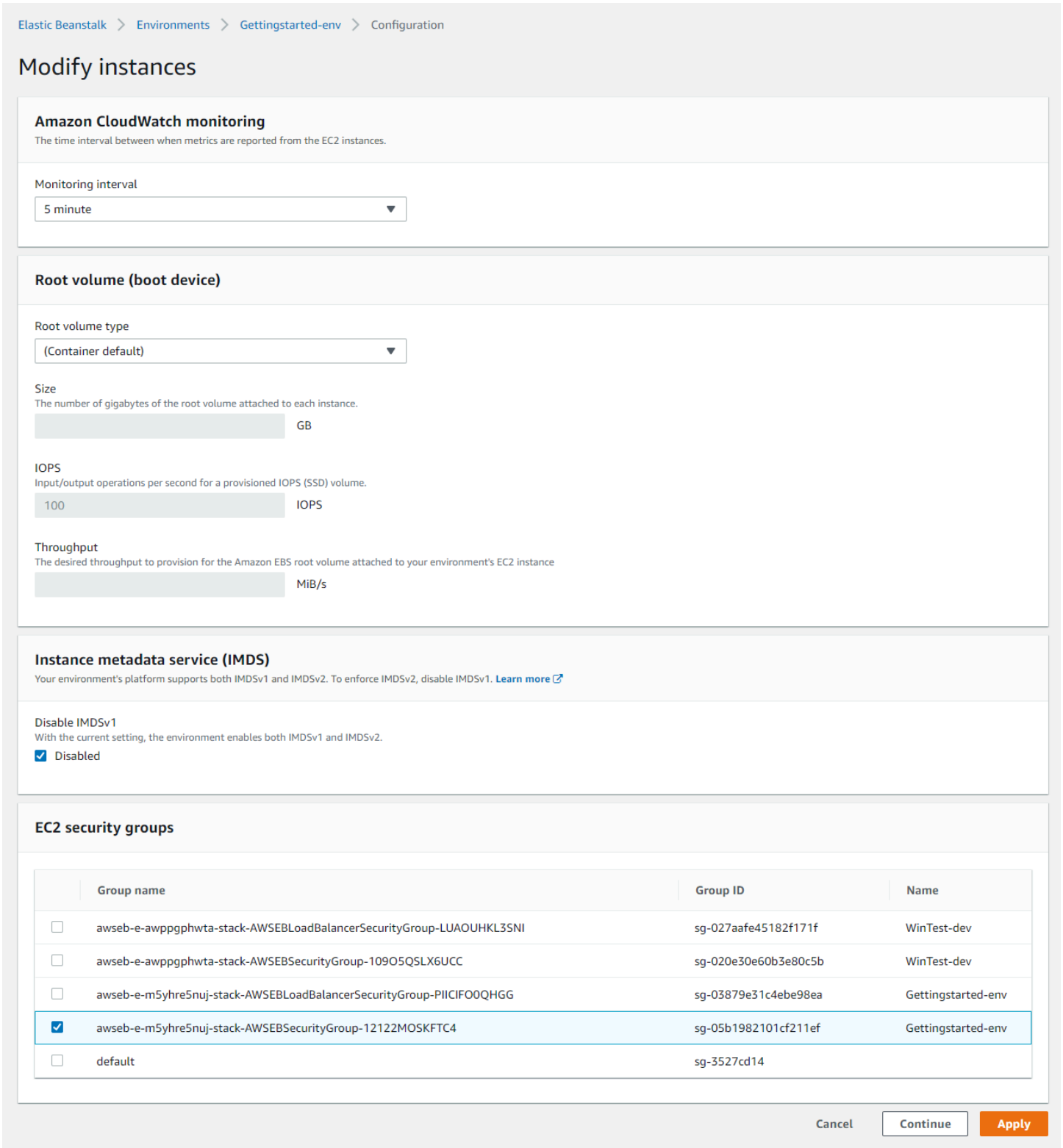
1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Pada kategori konfigurasi Instans, pilih Edit. Buat perubahan pada pengaturan dalam kategori ini, dan kemudian pilih Terapkan. Untuk mengatur deskripsi, lihat bagian the section called “Pengaturan kategori instans” di halaman ini.
5. Pada kategori konfigurasi Kapasitas, pilih Edit. Buat perubahan pada pengaturan dalam kategori ini, dan kemudian pilih Lanjutkan. Untuk mengatur deskripsi, lihat bagian the section called “Pengaturan kategori kapasitas” di halaman ini.

Pengaturan kategori instans

Setelan berikut yang terkait dengan EC2 instans Amazon tersedia di kategori konfigurasi Instans.

Opsi

- Interval pemantauan
- Volume root (perangkat boot)
- Layanan metadata instans
- EC2 kelompok keamanan


Elastic Beanstalk > Environments > Gettingstarted-env > Configuration
Modify instances
Amazon CloudWatch monitoring
The time interval between when metrics are reported from the EC2 instances.
Monitoring interval
5 minute
Root volume (boot device)
Root volume type
(Container default)
Size
The number of gigabytes of the root volume attached to each instance.
GB
IOPS
Input/output operations per second for a provisioned IOPS (SSD) volume.
100
IOPS
Throughput
The desired throughput to provision for the Amazon EBS root volume attached to your environment's EC2 instance
MiB/s
Instance metadata service (IMDS)
Your environment's platform supports both IMDSv1 and IMDSv2. To enforce IMDSv2, disable IMDSv1. Learn more
Disable IMDSv1
With the current setting, the environment enables both IMDSv1 and IMDSv2.
Disabled
EC2 security groups
Group name
Group ID
Name
awseb-e-awppgphwta-stack-AWSEBLoadBalancerSecurityGroup-LUAOUHKL3SNI
sg-027aafe45182f171f
WinTest-dev
awseb-e-awppgphwta-stack-AWSEBSecurityGroup-109O5QSLX6UCC
sg-020e30e60b3e80c5b
WinTest-dev
awseb-e-m5yhre5nuj-stack-AWSEBLoadBalancerSecurityGroup-PIICIFO0QHGG
sg-03879e31c4ebe98ea
Gettingstarted-env
awseb-e-m5yhre5nuj-stack-AWSEBSecurityGroup-12122MOSKFTC4
sg-05b1982101cf211ef
Gettingstarted-env
default
sg-3527cd14
Cancel
Continue
Apply

Interval pemantauan

Secara default, instans di lingkungan Anda mempublikasikan metrik kesehatan dasar ke Amazon dengan CloudWatch interval lima menit tanpa biaya tambahan.

Untuk pelaporan yang lebih rinci, Anda dapat mengatur interval Pemantauan menjadi 1 menit untuk meningkatkan frekuensi sumber daya di lingkungan Anda menerbitkan metrik kesehatan dasar. CloudWatch CloudWatch Biaya layanan berlaku untuk metrik interval satu menit. Untuk informasi lebih lanjut, lihat Amazon CloudWatch.

Volume root (perangkat boot)

Setiap instans di lingkungan Anda dikonfigurasi dengan volume root. Volume root adalah perangkat blok Amazon EBS yang terpasang pada instans untuk menyimpan sistem operasi, perpustakaan, skrip, dan kode sumber aplikasi Anda. Secara default, semua platform menggunakan perangkat blok SSD tujuan umum untuk penyimpanan.

Anda dapat memodifikasi Jenis volume root untuk menggunakan penyimpanan magnetik atau jenis volume IOPS SSD yang disediakan dan, jika perlu, menambah ukuran volume. Untuk volume IOPS yang disediakan, Anda juga harus memilih jumlah IOPS untuk persediaan. Throughput hanya berlaku untuk tipe volume gp3 SSD. Anda dapat memasukkan throughput yang diinginkan ke ketentuan. Ini dapat berkisar antara 125 dan 1000 mebibytes per detik (MIB/s). Pilih jenis volume yang memenuhi persyaratan performa dan harga Anda.

> **Important**
>
> Pengaturan `RootVolumeType` opsi dapat menyebabkan Elastic Beanstalk memigrasikan lingkungan yang ada dengan konfigurasi peluncuran untuk meluncurkan templat. Melakukannya memerlukan izin yang diperlukan untuk mengelola templat peluncuran. Izin ini disertakan dalam kebijakan terkelola kami. Jika Anda menggunakan kebijakan khusus alih-alih kebijakan terkelola kami, pembuatan atau pembaruan lingkungan mungkin gagal saat Anda memperbarui konfigurasi lingkungan. Untuk informasi lebih lanjut dan pertimbangan lainnya, lihatMemigrasi lingkungan Elastic Beanstalk Anda untuk meluncurkan template .

Untuk informasi selengkapnya, lihat Jenis Volume Amazon EBS di Panduan EC2 Pengguna Amazon dan Detail Produk Amazon EBS.

Layanan metadata instans

Layanan metadata instans (IMDS) adalah komponen pada instans yang menggunakan kode pada instans untuk mengakses metadata instans dengan aman. Kode dapat mengakses metadata instance dari instance yang sedang berjalan menggunakan salah satu dari dua metode. Mereka adalah Layanan Metadata Instance Versi 1 (IMDSv1) atau Layanan Metadata Instance Versi 2 (). IMDSv2 IMDSv2 lebih aman. Nonaktifkan IMDSv1 untuk menegakkan IMDSv2. Untuk informasi selengkapnya, lihat the section called “IMDS”.

> **Note**
>
> Bagian IMDS pada halaman konfigurasi ini hanya muncul untuk versi platform yang mendukung IMDSv2.

EC2 kelompok keamanan

Grup keamanan yang dilampirkan ke instans Anda menentukan lalu lintas mana yang diizinkan untuk mencapai dan keluar dari instans.

Grup EC2 keamanan default yang dibuat Elastic Beanstalk memungkinkan semua lalu lintas masuk dari internet atau penyeimbang beban pada port standar untuk HTTP (80) dan SSH (22). Anda juga dapat menentukan grup keamanan kustom Anda sendiri untuk menetapkan aturan firewall untuk EC2 instance. Kelompok keamanan dapat mengizinkan lalu lintas di port lain atau dari sumber lain. Misalnya, Anda dapat membuat grup keamanan untuk akses SSH yang memungkinkan lalu lintas masuk pada port 22 dari rentang alamat IP terbatas. Atau untuk keamanan tambahan, Anda dapat membuat satu yang memungkinkan lalu lintas dari host benteng yang hanya dapat Anda akses.

Anda dapat memilih untuk memilih keluar lingkungan Anda dari grup EC2 keamanan default dengan menyetel `DisableDefaultEC2SecurityGroup` opsi di aws:autoscaling:launchconfiguration namespace ke. `true` Opsi ini tidak tersedia di konsol, tetapi Anda dapat mengaturnya dengan AWS CLI. Untuk informasi selengkapnya, lihat Mengelola kelompok EC2 keamanan.

Untuk informasi selengkapnya tentang grup EC2 keamanan Amazon, lihat Grup EC2 Keamanan Amazon di Panduan EC2 Pengguna Amazon.

> **Note**
>
> Untuk mengizinkan lalu lintas antara instans lingkungan A dan instans lingkungan B, Anda dapat menambahkan aturan ke grup keamanan yang Elastic Beanstalk melekat pada

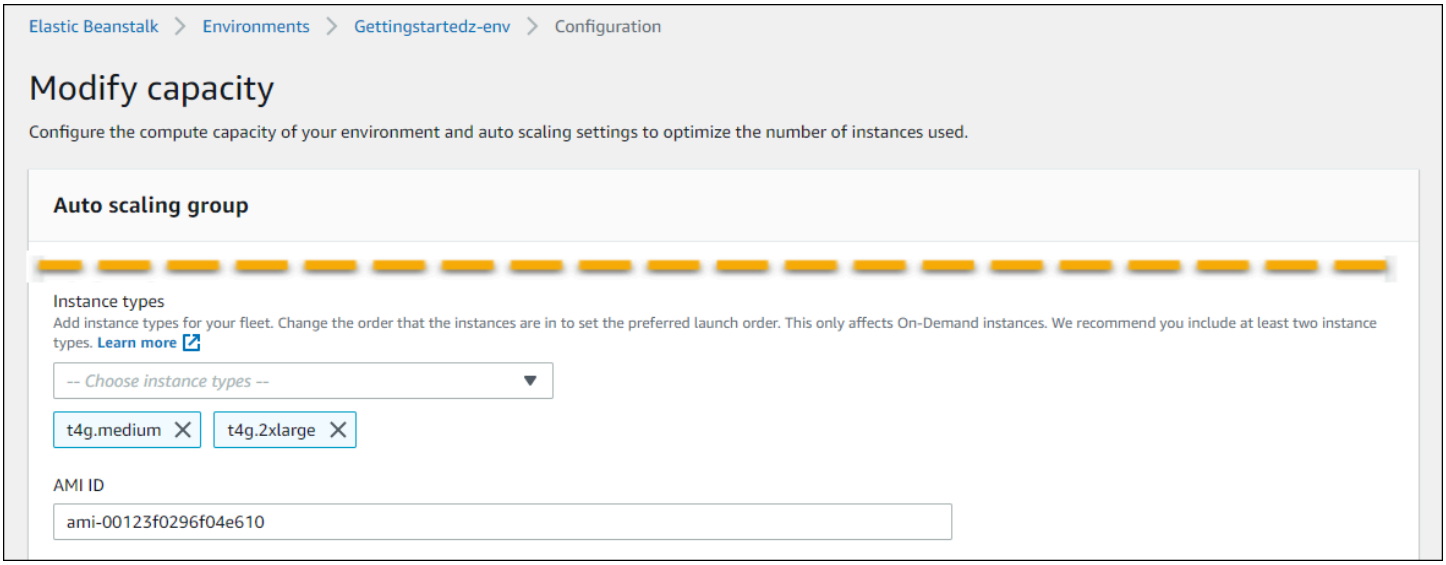
lingkungan B. Kemudian, Anda dapat menentukan grup keamanan yang Elastic Beanstalk melekat pada lingkungan A. Ini memungkinkan lalu lintas masuk dari, atau lalu lintas keluar ke, instance lingkungan A. Namun, melakukannya menciptakan ketergantungan antara dua kelompok keamanan. Jika nanti Anda mencoba menghentikan lingkungan A, Elastic Beanstalk tidak dapat menghapus grup keamanan lingkungan, karena grup keamanan lingkungan B bergantung padanya.
Oleh karena itu, kami sarankan Anda terlebih dahulu membuat grup keamanan terpisah. Kemudian, lampirkan ke lingkungan A, dan tentukan dalam aturan grup keamanan lingkungan B.

Pengaturan kategori kapasitas

Pengaturan berikut yang terkait dengan EC2 instans Amazon tersedia dalam kategori konfigurasi Kapasitas.

Opsi

- Tipe instans
- ID AMI



Tipe instans

Pengaturan tipe Instance menentukan jenis EC2 instans Amazon yang diluncurkan untuk menjalankan aplikasi Anda. Halaman konfigurasi ini menampilkan daftar jenis Instance. Anda dapat

memilih satu atau beberapa jenis instance. Konsol Elastic Beanstalk hanya menampilkan tipe instans berdasarkan arsitektur prosesor yang dikonfigurasi untuk lingkungan Anda. Oleh karena itu, Anda hanya dapat menambahkan jenis instance dari arsitektur prosesor yang sama.

> **Note**
>
> Meskipun konsol Elastic Beanstalk tidak menyediakan opsi untuk mengubah arsitektur prosesor dari lingkungan yang ada, Anda dapat melakukannya dengan. AWS CLI Misalnya perintah, lihatMengonfigurasi grup EC2 keamanan Amazon dan jenis instans menggunakan AWS CLI.

Pilih instance yang cukup kuat untuk menjalankan aplikasi Anda di bawah beban, tetapi tidak begitu kuat sehingga menganggur sebagian besar waktu. Untuk tujuan pengembangan, keluarga t2 instans menyediakan daya dalam jumlah sedang dengan kemampuan untuk meledak dalam waktu singkat. Untuk aplikasi skala besar dan ketersediaan tinggi, gunakan kumpulan instance untuk memastikan bahwa kapasitas tidak terlalu terpengaruh jika ada satu instance yang turun. Mulailah dengan jenis instans yang dapat Anda gunakan untuk menjalankan lima instance di bawah beban moderat selama jam normal. Jika ada instans yang gagal, instans lainnya dapat menyerap sisa lalu lintas. Buffer kapasitas juga memungkinkan menambah waktu bagi lingkungan saat lalu lintas mulai meningkat selama jam sibuk.

Untuk informasi selengkapnya tentang keluarga dan jenis EC2 instans Amazon, lihat Jenis instans di Panduan EC2 Pengguna Amazon. Untuk menentukan jenis instans yang memenuhi persyaratan dan Wilayah yang didukung, lihat Jenis instans yang tersedia di Panduan EC2 Pengguna Amazon.

ID AMI

Amazon Machine Image (AMI) adalah image mesin Amazon Linux atau Windows Server yang digunakan Elastic Beanstalk untuk EC2 meluncurkan instans Amazon di lingkungan Anda. Elastic Beanstalk menyediakan citra mesin yang berisi alat dan sumber daya yang diperlukan untuk menjalankan aplikasi Anda.

Elastic Beanstalk memilih AMI default untuk lingkungan Anda berdasarkan Wilayah, versi platform, dan arsitektur prosesor yang Anda pilih. Jika Anda telah membuat AMI kustom, ganti ID AMI default dengan ID kustom default Anda sendiri.

Mengelola kelompok EC2 keamanan

Ketika Elastic Beanstalk menciptakan lingkungan, ia menetapkan grup keamanan default EC2 ke instance yang diluncurkan dengannya. Grup keamanan yang dilampirkan ke instans Anda menentukan lalu lintas mana yang diizinkan untuk mencapai dan keluar dari instans.

Grup EC2 keamanan default yang dibuat Elastic Beanstalk memungkinkan semua lalu lintas masuk dari internet atau penyeimbang beban pada port standar untuk HTTP (80) dan SSH (22). Anda juga dapat menentukan grup keamanan kustom Anda sendiri untuk menetapkan aturan firewall untuk EC2 instance. Kelompok keamanan dapat mengizinkan lalu lintas di port lain atau dari sumber lain. Misalnya, Anda dapat membuat grup keamanan untuk akses SSH yang memungkinkan lalu lintas masuk pada port 22 dari rentang alamat IP terbatas. Atau untuk keamanan tambahan, Anda dapat membuat satu yang memungkinkan lalu lintas dari host benteng yang hanya dapat Anda akses.

Anda dapat memilih untuk memilih keluar lingkungan Anda dari grup EC2 keamanan default dengan menyetel `DisableDefaultEC2SecurityGroup` opsi di aws:autoscaling:launchconfiguration namespace ke. `true` Gunakan file konfigurasi AWS CLIatau untuk menerapkan opsi ini ke lingkungan Anda dan untuk melampirkan grup keamanan khusus ke EC2 instance.

Mengelola grup EC2 keamanan di lingkungan multi-instance

Jika Anda membuat grup EC2 keamanan khusus di lingkungan multi-instans, Anda juga harus mempertimbangkan bagaimana penyeimbang beban dan peraturan lalu lintas yang masuk menjaga instans Anda tetap aman dan dapat diakses.

Lalu lintas masuk ke lingkungan dengan beberapa EC2 instance dikelola oleh penyeimbang beban, yang mengarahkan lalu lintas masuk di antara semua instance. EC2 Ketika Elastic Beanstalk membuat EC2 grup keamanan default, itu juga mendefinisikan aturan masuk yang memungkinkan lalu lintas masuk dari penyeimbang beban. Tanpa aturan masuk ini di grup keamanan, lalu lintas yang masuk tidak akan diizinkan memasuki instans. Kondisi ini pada dasarnya akan memblokir instance dari permintaan eksternal.

Jika Anda menonaktifkan grup EC2 keamanan default untuk lingkungan yang seimbang beban, Elastic Beanstalk memvalidasi beberapa aturan konfigurasi. Jika konfigurasi tidak memenuhi pemeriksaan validasi, konfigurasi akan mengeluarkan pesan yang menginstruksikan Anda untuk menyediakan konfigurasi yang diperlukan. Pemeriksaan validasi adalah sebagai berikut:

- Setidaknya satu grup keamanan harus ditugaskan ke penyeimbang beban menggunakan `SecurityGroups` opsi aws:elbv2:loadbalancer atauaws:elb:loadbalancer, tergantung pada

apakah itu penyeimbang beban aplikasi atau penyeimbang beban klasik, masing-masing. Sebagai AWS CLI contoh lihatMengkonfigurasi dengan AWS CLI.

- Aturan lalu lintas masuk harus ada yang memungkinkan EC2 instans Anda menerima lalu lintas dari penyeimbang beban. Baik grup EC2 keamanan Anda dan grup keamanan penyeimbang beban Anda harus mereferensikan aturan masuk ini. Untuk informasi lebih lanjut, lihat Aturan masuk untuk lalu lintas bagian berikut.

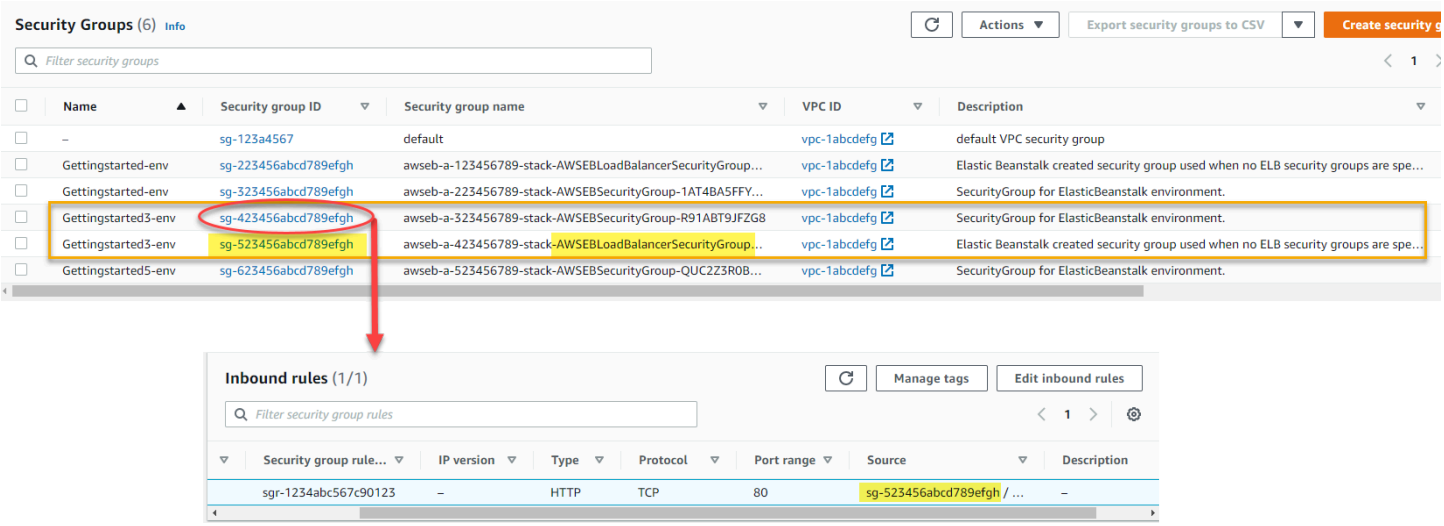
Aturan masuk untuk lalu lintas

Grup EC2 keamanan untuk lingkungan multi-instance, harus menyertakan aturan masuk yang mereferensikan grup keamanan penyeimbang beban. Ini berlaku untuk lingkungan dengan semua jenis penyeimbang beban, khusus atau bersama, dan dengan grup keamanan penyeimbang beban khusus atau default.

Anda dapat melihat semua grup keamanan yang dilampirkan ke komponen lingkungan Anda di EC2 konsol. Gambar berikut menunjukkan daftar EC2 konsol grup keamanan yang dibuat Elastic Beanstalk secara default selama operasi create environment.

Layar Grup Keamanan menampilkan lingkungan dan grup keamanan terkait. Baik GettingStarted-env dan GettingStarted3-env adalah lingkungan multi-instance dengan penyeimbang beban khusus. Masing-masing lingkungan ini memiliki dua grup keamanan yang terdaftar, satu untuk EC2 instance dan satu lagi untuk penyeimbang beban. Elastic Beanstalk menciptakan grup keamanan ini saat menciptakan lingkungan. GettingStarted5-env tidak memiliki grup keamanan penyeimbang beban, karena hanya memiliki satu EC2 instance, dan karenanya tidak ada penyeimbang beban.

Layar aturan Inbound menelusuri ke grup EC2 keamanan untuk instance 3-env. GettingStarted Contoh ini mendefinisikan aturan masuk untuk grup EC2 keamanan. Perhatikan bahwa kolom Sumber dalam Aturan masuk mencantumkan id grup keamanan grup keamanan penyeimbang beban yang tercantum dalam gambar sebelumnya. Aturan ini memungkinkan EC2 instance GettingStarted3-env untuk menerima lalu lintas masuk dari penyeimbang beban tertentu pada port 80.



Untuk informasi selengkapnya, lihat Mengubah grup keamanan untuk instans dan aturan Elastic Load Balancing di EC2 Panduan Pengguna Amazon.

Mengonfigurasi grup EC2 keamanan Amazon dan jenis instans menggunakan AWS CLI

Anda dapat menggunakan AWS Command Line Interface (AWS CLI) untuk mengonfigurasi EC2 instans Amazon di lingkungan Elastic Beanstalk Anda.

Mengkonfigurasi grup EC2 keamanan menggunakan AWS CLI

Topik ini memberikan contoh konfigurasi grup EC2 keamanan yang berbeda untuk lingkungan single instance dan load balanced (multi-instance). Untuk informasi selengkapnya tentang opsi dalam contoh ini, lihat aws:autoscaling:launchconfiguration.

> **Catatan**
>
> Operasi create environment menyediakan grup EC2 keamanan secara default. Ini juga menciptakan lingkungan dengan penyeimbang beban aplikasi secara default. Operasi lingkungan pembaruan dapat digunakan untuk menonaktifkan atau mengaktifkan grup EC2 keamanan default untuk lingkungan Anda dengan opsi `DisableDefaultEC2SecurityGroup` boolean. Contoh 5 menunjukkan cara mengatur lingkungan Anda kembali ke konfigurasi keamanan default jika Anda sebelumnya telah memodifikasinya.

Contoh berikut menunjukkan perintah create-environment yang memilih keluar dari grup EC2 keamanan default dan menyediakan grup keamanan khusus sebagai gantinya. Karena `DisableDefaultEC2SecurityGroup` opsi disetel ke`true`, grup EC2 keamanan default yang biasanya dikaitkan dengan Elastic Beanstalk ke instance EC2 tidak dibuat. Oleh karena itu, Anda harus memberikan `SecurityGroups` opsi kepada grup keamanan lain.

Perhatikan bahwa aws:elasticbeanstalk:environment `EnvironmentType` opsi diatur ke`SingleInstance`. Untuk membuat lingkungan instance tunggal, Anda harus menentukan opsi ini, karena `LoadBalanced` adalah default`EnvironmentType`. Karena lingkungan ini tidak menyertakan penyeimbang beban, kita tidak perlu menentukan grup keamanan penyeimbang beban.

Example 1 - Lingkungan instance tunggal baru dengan grup EC2 keamanan khusus (opsi namespace sebaris)

```
aws elasticbeanstalk create-environment \
--region us-east-1 \
--application-name my-app \
--environment-name my-env \
--solution-stack-name "64bit Amazon Linux 2023 v6.5.0 applrunning Node.js 22" \
--option-settings \
Namespace=aws:elasticbeanstalk:environment,OptionName=EnvironmentType,Value=SingleInstance
 \
Namespace=aws:autoscaling:launchconfiguration,OptionName=IamInstanceProfile,Value=aws-
elasticbeanstalk-ec2-role \
Namespace=aws:autoscaling:launchconfiguration,OptionName=DisableDefaultEC2SecurityGroup,Value=t
 \
Namespace=aws:autoscaling:launchconfiguration,OptionName=SecurityGroups,Value=sg-
abcdef01, sg-abcdef02 \
Namespace=aws:autoscaling:launchconfiguration,OptionName=EC2KeyName,Value=my-keypair
```

Sebagai alternatif, gunakan `options.json` file untuk menentukan opsi namespace alih-alih memasukkannya sebaris.

Example 2 - Lingkungan instance tunggal baru dengan grup EC2 keamanan khusus (opsi namespace dalam file) **`options.json`**

```
aws elasticbeanstalk create-environment \
--region us-east-1 \
--application-name my-app \
```

```
--environment-name my-env \
--solution-stack-name "64bit Amazon Linux 2023 v6.5.0 running Node.js 22" \
--option-settings file://options.json
```

Example

```
### example options.json ###
[
  { "Namespace" : "aws:elasticbeanstalk:environment",
    "OptionName" : "EnvironmentType",
    "Value" : "SingleInstance"
  },
  {
    "Namespace": "aws:autoscaling:launchconfiguration",
    "OptionName": "IamInstanceProfile",
    "Value": "aws-elasticbeanstalk-ec2-role"
  },
  {
    "Namespace": "aws:autoscaling:launchconfiguration",
    "OptionName": "DisableDefaultEC2SecurityGroup",
    "Value": "true"
  },
  {
    "Namespace": "aws:autoscaling:launchconfiguration",
    "OptionName": "SecurityGroups",
    "Value": "sg-abcdef01, sg-abcdef02"
  },
  {
    "Namespace": "aws:autoscaling:launchconfiguration",
    "OptionName": "EC2KeyName",
    "Value": "my-keypair"
  }
]
```

Contoh berikut menciptakan lingkungan load-balanced. Ini menentukan opsi aws:elasticbeanstalk:environment `LoadBalancerType` namespace diatur ke. `application` Karena kami menonaktifkan grup EC2 keamanan default dengan `DisableDefaultEC2SecurityGroup` opsi, kami perlu menyediakan grup keamanan khusus kami sendiri untuk EC2 instance lagi, dengan aws:autoscaling:launchconfiguration `SecurityGroups` opsi, seperti contoh sebelumnya. Karena lingkungan ini memiliki penyeimbang beban untuk merutekan lalu lintas, kami harus menyediakan grup keamanan untuk penyeimbang beban juga.

Untuk membuat lingkungan dengan penyeimbang beban klasik, tetapi konfigurasi yang sama, perbarui konfigurasi untuk opsi aws:elasticbeanstalk:environment namespace ke. `LoadBalancerType` `classic`

Jenis penyeimbang beban yang berbeda memiliki ruang nama berbeda yang menyimpan opsi untuk menentukan grup keamanan:

- penyeimbang beban aplikasi - opsi aws:elbv2:loadbalancer `SecurityGroups`
- penyeimbang beban klasik - opsi aws:elb:loadbalancer `SecurityGroups`
- penyeimbang beban jaringan — karena penyeimbang beban jaringan tidak memiliki grup keamanan, konfigurasikan grup EC2 keamanan dengan pengidentifikasi VPC. Untuk informasi selengkapnya, lihat Memperbarui grup keamanan untuk Network Load Balancer di Panduan Pengguna untuk Network Load Balancer.

Example 3 - Lingkungan multi-instance baru dengan grup EC2 keamanan khusus (opsi namespace dalam file) **`options.json`**

```
aws elasticbeanstalk create-environment \
--region us-east-1 \
--application-name my-app \
--environment-name my-env \
--solution-stack-name "64bit Amazon Linux 2023 v6.5.0 running Node.js 22" \
--option-settings file://options.json
```

Example

```
### example options.json ###
[
  {
    "Namespace" : "aws:elasticbeanstalk:environment",
    "OptionName" : "EnvironmentType",
    "Value" : "LoadBalanced"
  },
  {
  "Namespace" : "aws:elasticbeanstalk:environment",
    "OptionName" : "LoadBalancerType",
    "Value" : "application"
  },
  {
```

```
    "Namespace" : "aws:elbv2:loadbalancer",
    "OptionName" : "SecurityGroups",
    "Value" : "sg-abcdefghikl012345"
  },
  {
    "Namespace": "aws:autoscaling:launchconfiguration",
    "OptionName": "IamInstanceProfile",
    "Value": "aws-elasticbeanstalk-ec2-role"
  },
  {
    "Namespace": "aws:autoscaling:launchconfiguration",
    "OptionName": "DisableDefaultEC2SecurityGroup",
    "Value": "true"
  },
  {
    "Namespace": "aws:autoscaling:launchconfiguration",
    "OptionName": "SecurityGroups",
    "Value": "sg-abcdef01, sg-abcdef02"
  },
  {
    "Namespace": "aws:autoscaling:launchconfiguration",
    "OptionName": "EC2KeyName",
    "Value": "my-keypair"
  }
]
```

Anda dapat menonaktifkan grup EC2 keamanan default untuk lingkungan yang ada dengan perintah update-environment. Contoh perintah berikut menonaktifkan grup EC2 keamanan default dan menetapkan EC2 instans lingkungan grup keamanan kustom EC2 .

Gunakan `options.jason` file contoh dalam contoh 4 (a), 4 (b), atau 4 (c), tergantung pada apakah lingkungan seimbang beban dan jenis penyeimbang beban. File konfigurasi 4 (a) menentukan grup keamanan untuk lingkungan satu instance. Karena tidak memerlukan penyeimbang beban, kami hanya menyediakan grup keamanan untuk EC2 instans. File konfigurasi 4 (b) dan 4 (c) menentukan grup keamanan untuk penyeimbang beban aplikasi dan penyeimbang beban klasik. Untuk kasus ini kita juga perlu menentukan grup keamanan untuk penyeimbang beban.

Example 4 - Perbarui lingkungan yang ada untuk menonaktifkan grup EC2 keamanan default (opsi namespace dalam **`options.json`** file)

```
aws elasticbeanstalk update-environment \
--region us-east-1 \
```

```
--application-name my-app \
--environment-name my-env \
--solution-stack-name "64bit Amazon Linux 2023 v6.5.0 running Node.js 22" \
--option-settings file://options.json
```

Example 4 (a) - File konfigurasi untuk lingkungan instance tunggal (tidak ada penyeimbang beban)

```
### example options.json ###
[
  {
    "Namespace": "aws:autoscaling:launchconfiguration",
    "OptionName": "DisableDefaultEC2SecurityGroup",
    "Value": "true"
  },
  {
    "Namespace": "aws:autoscaling:launchconfiguration",
    "OptionName": "SecurityGroups",
    "Value": "sg-abcdef01, sg-abcdef02"
  }
]
```

Untuk memperbarui lingkungan yang menggunakan penyeimbang beban aplikasi, gunakan `aws:elbv2:loadbalancer` namespace untuk menentukan grup keamanan untuk penyeimbang beban.

Example 4 (b) - File konfigurasi untuk lingkungan dengan penyeimbang beban aplikasi

```
### example options.json ###
[
  {
    "Namespace" : "aws:elbv2:loadbalancer",
    "OptionName" : "SecurityGroups",
    "Value" : "sg-abcdefghikl012345"
  },
  {
    "Namespace": "aws:autoscaling:launchconfiguration",
    "OptionName": "DisableDefaultEC2SecurityGroup",
    "Value": "true"
  },
  {
    "Namespace": "aws:autoscaling:launchconfiguration",
    "OptionName": "SecurityGroups",
```

```
    "Value": "sg-abcdef01, sg-abcdef02"
  }
]
```

Untuk memperbarui lingkungan yang menggunakan penyeimbang beban klasik, gunakan `aws:elb:loadbalancer` namespace untuk menentukan grup keamanan untuk penyeimbang beban.

Example 4 (c) - File konfigurasi untuk lingkungan dengan penyeimbang beban klasik

```
### example options.json ###
[
  {
    "Namespace" : "aws:elb:loadbalancer",
    "OptionName" : "SecurityGroups",
    "Value" : "sg-abcdefghikl012345"
  },
  {
    "Namespace": "aws:autoscaling:launchconfiguration",
    "OptionName": "DisableDefaultEC2SecurityGroup",
    "Value": "true"
  },
  {
    "Namespace": "aws:autoscaling:launchconfiguration",n
    "OptionName": "SecurityGroups",
    "Value": "sg-abcdef01, sg-abcdef02"
  }
]
```

Untuk mengembalikan lingkungan Anda ke perilaku dan konfigurasi default dengan grup keamanan default yang ditetapkan Elastic Beanstalk, gunakan perintah update-environment untuk menyetel ke. `DisableDefaultEC2SecurityGroup` `false` Untuk lingkungan multi-instance, Elastic Beanstalk juga menangani grup keamanan dan aturan lalu lintas jaringan untuk penyeimbang beban lingkungan Anda.

Contoh berikut berlaku untuk lingkungan single instance atau multi-instance (load balanced):

Example 5 - Perbarui lingkungan kembali menggunakan grup keamanan default (opsi namespace dalam **`options.json`** file)

```
aws elasticbeanstalk update-environment \
```

```
--region us-east-1 \
--application-name my-app \
--environment-name my-env \
--solution-stack-name "64bit Amazon Linux 2023 v6.5.0 running Node.js 22" \
--option-settings file://options.json
```

Example

```
### example options.json ###
[
  {
    "Namespace": "aws:autoscaling:launchconfiguration",
    "OptionName": "DisableDefaultEC2SecurityGroup",
    "Value": "false"
  }
]
```

Mengkonfigurasi EC2 dengan tipe instance menggunakan AWS CLI

Topik ini memberikan contoh untuk mengonfigurasi jenis instance EC2 instance di lingkungan Anda.

Dua contoh pertama menciptakan lingkungan baru. Perintah menentukan jenis EC2 instans Amazon, t4g.small, yang didasarkan pada arsitektur prosesor arm64. Elastic Beanstalk me-default Image ID (AMI EC2) untuk instance berdasarkan Region, versi platform, dan jenis instans. Jenis instance sesuai dengan arsitektur prosesor. `solution-stack-name`Parameter ini berlaku untuk versi platform.

Example 1 - buat lingkungan berbasis arm64 baru (opsi namespace inline)

```
aws elasticbeanstalk create-environment \
--region us-east-1 \
--application-name my-app \
--environment-name my-env \
--solution-stack-name "64bit Amazon Linux 2 v3.4.7 running Docker" \
--option-settings \
Namespace=aws:autoscaling:launchconfiguration,OptionName=IamInstanceProfile,Value=aws-elasticbeanstalk-ec2-role \
Namespace=aws:ec2:instances,OptionName=InstanceTypes,Value=t4g.small
```

Sebagai alternatif, gunakan `options.json` file untuk menentukan opsi namespace alih-alih memasukkannya sebaris.

Example 2 - buat lingkungan berbasis arm64 baru (opsi namespace dalam file) **`options.json`**

```
aws elasticbeanstalk create-environment \
--region us-east-1 \
--application-name my-app \
--environment-name my-env \
--solution-stack-name "64bit Amazon Linux 2 v3.4.7 running Docker" \
--option-settings file://options.json
```

Example

```
### example options.json ###
[
  {
    "Namespace": "aws:autoscaling:launchconfiguration",
    "OptionName": "IamInstanceProfile",
    "Value": "aws-elasticbeanstalk-ec2-role"
  },
  {
    "Namespace": "aws:ec2:instances",
    "OptionName": "InstanceTypes",
    "Value": "t4g.small"
  }
]
```

Dua contoh berikutnya memperbarui konfigurasi untuk lingkungan yang ada dengan perintah update-environment. Dalam contoh ini kita menambahkan jenis instance lain yang juga didasarkan pada arsitektur prosesor arm64. Untuk lingkungan yang ada, semua jenis instance yang ditambahkan harus memiliki arsitektur prosesor yang sama. Jika Anda ingin mengganti jenis instance yang ada dengan yang berasal dari arsitektur yang berbeda, Anda dapat melakukannya. Tetapi pastikan bahwa semua jenis instance dalam perintah memiliki tipe arsitektur yang sama.

Example 3 - perbarui lingkungan berbasis arm64 yang ada (opsi namespace sebaris)

```
aws elasticbeanstalk update-environment \
--region us-east-1 \
--application-name my-app \
--environment-name my-env \
--solution-stack-name "64bit Amazon Linux 2 v3.4.7 running Docker" \
--option-settings \
```

```
Namespace=aws:autoscaling:launchconfiguration,OptionName=IamInstanceProfile,Value=aws-
elasticbeanstalk-ec2-role \
Namespace=aws:ec2:instances,OptionName=InstanceTypes,Value=t4g.small,t4g.micro
```

Sebagai alternatif, gunakan `options.json` file untuk menentukan opsi namespace alih-alih memasukkannya sebaris.

Example 4 - memperbarui lingkungan berbasis arm64 yang ada (opsi namespace dalam file) **`options.json`**

```
aws elasticbeanstalk update-environment \
--region us-east-1 \
--application-name my-app \
--environment-name my-env \
--solution-stack-name "64bit Amazon Linux 2 v3.4.7 running Docker" \
--option-settings file://options.json
```

Example

```
### example options.json ###
[
  {
    "Namespace": "aws:autoscaling:launchconfiguration",
    "OptionName": "IamInstanceProfile",
    "Value": "aws-elasticbeanstalk-ec2-role"
  },
  {
    "Namespace": "aws:ec2:instances",
    "OptionName": "InstanceTypes",
    "Value": "t4g.small, t4g.micro"
  }
]
```

Dua contoh berikutnya menunjukkan lebih banyak perintah create-environment. Contoh-contoh ini tidak memberikan nilai untuk`InstanceTypes`. Ketika `InstanceTypes` nilai tidak ditentukan, Elastic Beanstalk default ke arsitektur prosesor berbasis x86. ID Gambar (AMI) untuk EC2 instance lingkungan akan default sesuai dengan Wilayah, versi platform, dan jenis instans default. Jenis instance sesuai dengan arsitektur prosesor.

Example 5 - buat lingkungan berbasis x86 baru (opsi namespace inline)

```
aws elasticbeanstalk create-environment \
--region us-east-1 \
--application-name my-app \
--environment-name my-env \
--solution-stack-name "64bit Amazon Linux 2 v3.4.7 running Docker" \
--option-settings \
Namespace=aws:autoscaling:launchconfiguration,OptionName=IamInstanceProfile,Value=aws-elasticbeanstalk-ec2-role
```

Sebagai alternatif, gunakan `options.json` file untuk menentukan opsi namespace alih-alih memasukkannya sebaris.

Example 6 - buat lingkungan berbasis x86 baru (opsi namespace dalam file) **`options.json`**

```
aws elasticbeanstalk create-environment \
--region us-east-1 \
--application-name my-app \
--environment-name my-env \
--solution-stack-name "64bit Amazon Linux 2 v3.4.7 running Docker" \
--option-settings file://options.json
```

Example

```
### example options.json ###
[
  {
    "Namespace": "aws:autoscaling:launchconfiguration",
    "OptionName": "IamInstanceProfile",
    "Value": "aws-elasticbeanstalk-ec2-role"
  }
]
```

Mengonfigurasi EC2 instans Amazon dengan opsi namespace

Anda dapat menggunakan opsi konfigurasi di `aws:autoscaling:launchconfiguration` namespace untuk mengonfigurasi instance untuk lingkungan Anda, termasuk opsi tambahan yang tidak tersedia di konsol.

> **Important**
>
> Pengaturan`DisableIMDSv1`,`RootVolumeType`, atau `BlockDeviceMappings` opsi dapat menyebabkan Elastic Beanstalk memigrasikan lingkungan yang ada dengan konfigurasi peluncuran untuk meluncurkan templat. Melakukannya memerlukan izin yang diperlukan untuk mengelola templat peluncuran. Izin ini disertakan dalam kebijakan terkelola kami. Jika Anda menggunakan kebijakan khusus alih-alih kebijakan terkelola kami, pembuatan atau pembaruan lingkungan mungkin gagal saat Anda memperbarui konfigurasi lingkungan. Untuk informasi lebih lanjut dan pertimbangan lainnya, lihatMemigrasi lingkungan Elastic Beanstalk Anda untuk meluncurkan template .

Contoh file konfigurasi berikut menggunakan opsi konfigurasi dasar yang dijelaskan dalam topik ini. Untuk melihat contoh opsi konfigurasi tambahan saat Anda perlu menentukan grup keamanan untuk penyeimbang beban, lihat. Mengkonfigurasi dengan AWS CLI

```
option_settings:
  aws:autoscaling:launchconfiguration:
    SecurityGroups: my-securitygroup
    MonitoringInterval: "1 minute"
    DisableIMDSv1: false
    DisableDefaultEC2SecurityGroup: true
    SecurityGroups: "sg-abcdef01, sg-abcdef02"
    EC2KeyName: my-keypair
    IamInstanceProfile: "aws-elasticbeanstalk-ec2-role"
    BlockDeviceMappings: "/dev/sdj=:100,/dev/sdh=snap-51eef269,/dev/sdb=ephemeral0"
  aws:elasticbeanstalk:environment:
    EnvironmentType: SingleInstance
```

`DisableDefaultEC2SecurityGroup`Dan tidak `BlockDeviceMappings` tersedia di konsol.

Anda dapat menggunakan `BlockDeviceMappings` untuk mengonfigurasi perangkat blok tambahan untuk instans Anda. Untuk informasi selengkapnya, lihat Memblokir Pemetaan Perangkat di Panduan EC2 Pengguna Amazon.

Konsol EB CLI dan Elastic Beanstalk menerapkan nilai yang direkomendasikan untuk pilihan sebelumnya. Anda harus menghapus pengaturan ini jika Anda ingin menggunakan file konfigurasi untuk mengonfigurasi hal yang sama. Lihat Nilai yang disarankan untuk rincian selengkapnya.

Mengonfigurasi IMDS pada instans lingkungan Elastic Beanstalk Anda

Topik ini menjelaskan Layanan Metadata Instance (IMDS).

Metadata instans adalah data yang terkait dengan instans Amazon Elastic Compute Cloud (Amazon EC2) yang dapat digunakan aplikasi untuk mengonfigurasi atau mengelola instance yang sedang berjalan. Layanan metadata instans (IMDS) adalah komponen pada instans yang menggunakan kode pada instans untuk mengakses metadata instans dengan aman. Kode ini dapat berupa kode platform Elastic Beanstalk pada instance lingkungan Anda AWS , SDK yang mungkin digunakan aplikasi Anda, atau bahkan kode aplikasi Anda sendiri. Untuk informasi selengkapnya, lihat Metadata instans dan data pengguna di EC2 Panduan Pengguna Amazon.

Kode dapat mengakses metadata instance dari instance yang sedang berjalan menggunakan salah satu dari dua metode: Instance Metadata Service Version 1 (IMDSv1) atau Instance Metadata Service Version 2 (). IMDSv2 IMDSv2 menggunakan permintaan berorientasi sesi dan mengurangi beberapa jenis kerentanan yang dapat digunakan untuk mencoba mengakses IMDS. Untuk informasi tentang kedua metode ini, lihat Mengonfigurasi layanan metadata instance di Panduan Pengguna Amazon EC2 .

Topik

- Dukungan platform untuk IMDS
- Memilih metode IMDS
- Mengonfigurasi IMDS menggunakan konsol Elastic Beanstalk
- Namespace aws:autoscaling:launchconfiguration

Dukungan platform untuk IMDS

Platform Elastic Beanstalk yang berjalan di Amazon Linux 2 dan Amazon Linux 2023 dan server Windows semuanya mendukung keduanya dan. IMDSv1 IMDSv2 Untuk informasi selengkapnya, lihat Mengonfigurasi IMDS menggunakan konsol Elastic Beanstalk

Memilih metode IMDS

Saat membuat keputusan tentang metode IMDS yang Anda ingin lingkungan Anda dukung, pertimbangkan kasus penggunaan berikut:

- AWS SDK — Jika aplikasi Anda menggunakan AWS SDK, pastikan Anda menggunakan SDK versi terbaru. AWS SDKsMembuat panggilan IMDS, dan versi SDK yang lebih baru digunakan

IMDSv2 bila memungkinkan. Jika Anda pernah menonaktifkan IMDSv1, atau jika aplikasi Anda menggunakan versi SDK lama, panggilan IMDS mungkin gagal.

- Kode aplikasi Anda — Jika aplikasi Anda membuat panggilan IMDS, pertimbangkan untuk menggunakan AWS SDK sehingga Anda dapat melakukan panggilan alih-alih membuat permintaan HTTP langsung. Dengan cara ini, Anda tidak perlu membuat perubahan kode untuk beralih di antara metode IMDS. AWS SDK menggunakan IMDSv2 bila memungkinkan.
- Kode platform Elastic Beanstalk — Kode kami membuat panggilan IMDS AWS melalui SDK, dan IMDSv2 karenanya digunakan pada semua versi platform pendukung. Jika kode Anda menggunakan up-to-date AWS SDK dan membuat semua panggilan IMDS melalui SDK, Anda dapat menonaktifkannya dengan aman. IMDSv1

Mengonfigurasi IMDS menggunakan konsol Elastic Beanstalk

Anda dapat memodifikasi konfigurasi instans Amazon lingkungan Elastic Beanstalk di konsol EC2 Elastic Beanstalk.

> **Important**
>
> Pengaturan `DisableIMDSv1` opsi dapat menyebabkan Elastic Beanstalk memigrasikan lingkungan yang ada dengan konfigurasi peluncuran untuk meluncurkan templat. Melakukannya memerlukan izin yang diperlukan untuk mengelola templat peluncuran. Izin ini disertakan dalam kebijakan terkelola kami. Jika Anda menggunakan kebijakan khusus alih-alih kebijakan terkelola kami, pembuatan atau pembaruan lingkungan mungkin gagal saat Anda memperbarui konfigurasi lingkungan. Untuk informasi lebih lanjut dan pertimbangan lainnya, lihatMemigrasi lingkungan Elastic Beanstalk Anda untuk meluncurkan template .

Untuk mengonfigurasi IMDS pada EC2 instans Amazon Anda di konsol Elastic Beanstalk

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Dalam kategori konfigurasi lalu lintas Instans dan penskalaan, pilih Edit.
5. Setel Nonaktifkan IMDSv1 untuk menegakkan IMDSv2. Hapus Nonaktifkan IMDSv1 untuk mengaktifkan keduanya IMDSv1 dan IMDSv2.

6. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Namespace aws:autoscaling:launchconfiguration

Anda dapat menggunakan opsi konfigurasi di namespace `aws:autoscaling:launchconfiguration` untuk mengonfigurasi IMDS pada instans lingkungan Anda.

> **Important**
>
> Pengaturan `DisableIMDSv1` opsi dapat menyebabkan Elastic Beanstalk memigrasikan lingkungan yang ada dengan konfigurasi peluncuran untuk meluncurkan templat. Melakukannya memerlukan izin yang diperlukan untuk mengelola templat peluncuran. Izin ini disertakan dalam kebijakan terkelola kami. Jika Anda menggunakan kebijakan khusus alih-alih kebijakan terkelola kami, pembuatan atau pembaruan lingkungan mungkin gagal saat Anda memperbarui konfigurasi lingkungan. Untuk informasi lebih lanjut dan pertimbangan lainnya, lihatMemigrasi lingkungan Elastic Beanstalk Anda untuk meluncurkan template .

Contoh file konfigurasi berikut menonaktifkan IMDSv1 menggunakan `DisableIMDSv1` opsi.

```
option_settings:
  aws:autoscaling:launchconfiguration:
    DisableIMDSv1: true
```

Setel Nonaktifkan IMDSv1 `true` untuk menonaktifkan IMDSv1 dan menegakkan IMDSv2.

Setel Nonaktifkan IMDSv1 `false` untuk mengaktifkan keduanya IMDSv1 dan IMDSv2.

Auto Scaling instance lingkungan Elastic Beanstalk Anda

Topik ini menjelaskan bagaimana Anda dapat menyesuaikan fitur Auto Scaling untuk mengelola beban kerja lingkungan Elastic Beanstalk Anda. Anda dapat mengonfigurasi Auto Scaling untuk lingkungan Anda menggunakan konsol Elastic Beanstalk, opsi konfigurasi namespace, CLI, atau EB. AWS CLI

Lingkungan instans beban seimbang atau tunggal

AWS Elastic Beanstalk Lingkungan Anda menyertakan grup Auto Scaling yang mengelola EC2 instans Amazon di lingkungan Anda. Dalam lingkungan instans tunggal, grup Auto Scaling memastikan bahwa selalu ada satu instans yang berjalan. Di lingkungan dengan beban seimbang, Anda mengonfigurasi grup dengan berbagai instans untuk dijalankan, dan Auto Scaling menambahkan atau menghapus instans yang diperlukan, berdasarkan beban.

EC2 Konfigurasi instans

Grup Auto Scaling juga menerapkan pilihan konfigurasi Anda untuk menyediakan dan mengelola EC2 instance di lingkungan Anda. Anda dapat memodifikasi EC2 konfigurasi untuk mengubah jenis instans, key pair, penyimpanan Amazon Elastic Block Store (Amazon EBS), dan pengaturan lain yang hanya dapat dikonfigurasi saat meluncurkan instance.

Instans Sesuai Permintaan dan Spot

Sebagai pilihan, Elastic Beanstalk dapat menyertakan Instans Spot di lingkungan Anda dan mengelolanya dalam kombinasi dengan instans Sesuai Permintaan. Anda dapat mengonfigurasi EC2 Auto Scaling Amazon untuk memantau dan secara otomatis merespons perubahan yang memengaruhi ketersediaan Instans Spot Anda dengan mengaktifkan Penyeimbangan Kembali Kapasitas. Anda juga dapat mengonfigurasi layanan Auto Scaling Strategi alokasi spot yang digunakan untuk menyediakan Instans Spot ke lingkungan Anda.

Izin yang diperlukan saat mengaktifkan Instans Spot

Mengaktifkan permintaan Instans Spot memerlukan penggunaan templat EC2 peluncuran Amazon. Saat Anda mengonfigurasi fitur ini selama pembuatan atau pembaruan lingkungan, Elastic Beanstalk mencoba mengonfigurasi lingkungan Anda untuk menggunakan templat peluncuran EC2 Amazon (jika lingkungan belum menggunakannya). Dalam kasus ini, jika kebijakan pengguna Anda tidak memiliki izin yang diperlukan, pembuatan atau pembaruan lingkungan mungkin gagal. Oleh karena itu, kami merekomendasikan Anda untuk menggunakan kebijakan pengguna terkelola kami atau menambahkan izin yang diperlukan untuk kebijakan khusus Anda. Untuk detail tentang izin yang diperlukan, lihat Izin yang diperlukan untuk template peluncuran.

Pemicu Auto Scaling

Grup Auto Scaling menggunakan dua CloudWatch alarm Amazon untuk memicu operasi penskalaan. Pemicu default menskalakan ketika lalu lintas jaringan keluar rata-rata dari setiap instans lebih tinggi dari 6 MiB atau lebih rendah dari 2 MiB selama periode lima menit. Untuk menggunakan Auto Scaling secara efektif, konfigurasikan pemicu yang sesuai untuk aplikasi Anda, tipe instans, dan persyaratan

layanan. Anda dapat menskalakan berdasar beberapa statistik termasuk latensi, I/O disk, utilisasi CPU, dan jumlah permintaan.

Jadwalkan tindakan Auto Scaling

Untuk mengoptimalkan penggunaan EC2 instans Amazon di lingkungan Anda melalui periode lalu lintas puncak yang dapat diprediksi, konfigurasikan grup Auto Scaling Anda untuk mengubah jumlah instance-nya sesuai jadwal. Anda dapat menjadwalkan perubahan pada konfigurasi grup Anda yang berulang setiap hari atau setiap minggu, atau menjadwalkan perubahan satu-kali untuk mempersiapkan acara pemasaran yang akan mengarahkan banyak lalu lintas ke situs Anda.

Pemeriksaan kesehatan Auto Scaling

Auto Scaling memantau kesehatan setiap EC2 instans Amazon yang diluncurkan. Jika ada instans yang berakhir tiba-tiba, Auto Scaling mendeteksi penghentian dan meluncurkan instans pengganti. Untuk mengonfigurasi grup agar menggunakan mekanisme pemeriksaan kondisi penyeimbang beban, lihat Pengaturan pemeriksaan kesehatan Auto Scaling untuk lingkungan Elastic Beanstalk Anda.

Topik

- Memigrasi lingkungan Elastic Beanstalk Anda untuk meluncurkan template
- Dukungan Spot Instance untuk lingkungan Elastic Beanstalk Anda
- Pemicu Auto Scaling untuk lingkungan Elastic Beanstalk Anda
- Tindakan Auto Scaling Terjadwal untuk lingkungan Elastic Beanstalk Anda
- Pengaturan pemeriksaan kesehatan Auto Scaling untuk lingkungan Elastic Beanstalk Anda

Memigrasi lingkungan Elastic Beanstalk Anda untuk meluncurkan template

Mulai 1 Oktober 2024, Amazon EC2 Auto Scaling tidak lagi mendukung konfigurasi peluncuran untuk akun baru. Akun yang dibuat sebelum tanggal tersebut mungkin memiliki konfigurasi peluncuran.

Kami merekomendasikan migrasi untuk meluncurkan template untuk manfaat berikut:

- Peningkatan ketersediaan untuk aplikasi Anda
- Optimalisasi beban kerja yang lebih baik di grup Auto Scaling
- Akses ke fitur Auto EC2 Scaling terbaru

Untuk informasi selengkapnya, lihat konfigurasi peluncuran Auto Scaling di Panduan Pengguna Amazon EC2 Auto Scaling.

Pengaturan opsi untuk templat peluncuran

Untuk memigrasikan lingkungan Anda dari konfigurasi peluncuran ke templat peluncuran, tetapkan salah satu opsi konfigurasi berikut:

- `RootVolumeType`opsi diatur ke gp3. Anda dapat mengatur opsi ini dengan konsol atau namespace.
- `BlockDeviceMappings`opsi berisi gp3. Anda dapat mengatur opsi ini dengan konsol atau namespace.
- `DisableIMDSv1`opsi disetel ke true. Kami menyarankan Anda mengatur opsi ini menggunakan namespace.
- `EnableSpot`opsi disetel ke true. Untuk informasi selengkapnya, lihat Mengaktifkan Instans Spot.

> **Important**
>
> Setelah lingkungan mulai menggunakan template peluncuran, Elastic Beanstalk tidak kembali ke konfigurasi peluncuran, bahkan jika Anda menghapus opsi konfigurasi yang awalnya memicu penggunaan template peluncuran.

Konfirmasikan apakah lingkungan Anda memiliki konfigurasi peluncuran atau templat peluncuran

Anda dapat mengonfirmasi apakah lingkungan Anda sudah menggunakan template peluncuran, atau jika menggunakan konfigurasi peluncuran, dengan memeriksa template CloudFormation tumpukan.

Untuk memeriksa template CloudFormation tumpukan lingkungan Anda

1. Buka AWS CloudFormation konsol di https://console.aws.amazon.com/cloudformation.
2. Pada bilah navigasi di bagian atas layar, pilih AWS Wilayah tempat Anda membuat lingkungan.
3. Pada halaman Stacks CloudFormation konsol, periksa kolom Deskripsi.

   Cari dan pilih tumpukan untuk lingkungan Elastic Beanstalk. CloudFormation menampilkan detail tumpukan untuk lingkungan.

4. Di Detail Stack pilih tab Template.

   Menggunakan pencarian halaman browser Anda, Anda dapat mencari teks template untuk launchtemplate atau launchconfiguration.

Untuk informasi selengkapnya, lihat Melihat informasi tumpukan di Panduan AWS CloudFormation Pengguna.

Izin yang diperlukan untuk template peluncuran

Kebijakan AWSElasticBeanstalkManagedUpdatesCustomerRolePolicyperan layanan terkelola Elastic Beanstalk default memberikan izin yang diperlukan untuk membuat dan mengelola templat peluncuran. Elastic Beanstalk harus mengelola template peluncuran untuk menyelesaikan banyak operasi lingkungan, termasuk menciptakan lingkungan.

Jika Anda melampirkan kebijakan kustom ke peran layanan Elastic Beanstalk, verifikasi bahwa peran layanan menyertakan izin berikut untuk membuat templat peluncuran. Izin ini memungkinkan Elastic Beanstalk berhasil membuat dan memperbarui lingkungan di akun Anda:

Izin yang diperlukan untuk templat EC2 peluncuran Amazon

- `ec2:RunInstances`
- `ec2:CreateLaunchTemplate`
- `ec2:CreateLaunchTemplateVersions`
- `ec2:DeleteLaunchTemplate`
- `ec2:DeleteLaunchTemplateVersions`
- `ec2:DescribeLaunchTemplate`
- `ec2:DescribeLaunchTemplateVersions`

Contoh berikut kebijakan IAM mencakup izin ini.

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:RunInstances",
        "ec2:CreateLaunchTemplate",
```

```
        "ec2:CreateLaunchTemplateVersions",
        "ec2:DeleteLaunchTemplate",
        "ec2:DeleteLaunchTemplateVersions",
        "ec2:DescribeLaunchTemplate",
        "ec2:DescribeLaunchTemplateVersions"
      ],
      "Resource": [
                "*"
            ]
    }
  ]
}
```

Untuk informasi selengkapnya, lihat Mengelola peran layanan Elastic Beanstalk dan Mengelola kebijakan pengguna Elastic Beanstalk.

Lebih lanjut tentang template peluncuran

Untuk mempelajari lebih lanjut tentang template peluncuran, lihat templat peluncuran Auto Scaling di Panduan Pengguna Amazon Auto EC2 Scaling.

Untuk mempelajari lebih lanjut tentang AWS transisi untuk meluncurkan template dan manfaat yang mereka tawarkan, lihat Amazon EC2 Auto Scaling tidak akan lagi menambahkan dukungan untuk EC2 fitur baru ke Luncurkan Konfigurasi di Blog Komputasi.AWS

> **Important**
>
> Anda tidak perlu mengikuti prosedur yang dirujuk dalam artikel blog ini untuk transisi lingkungan yang lebih lama untuk meluncurkan template. Untuk memigrasikan lingkungan Elastic Beanstalk yang ada untuk meluncurkan template, tetapkan salah satu opsi yang tercantum di dalamnya. Pengaturan opsi untuk templat peluncuran

Dukungan Spot Instance untuk lingkungan Elastic Beanstalk Anda

Topik ini menjelaskan opsi konfigurasi yang tersedia bagi Anda untuk mengelola kapasitas dan load balancing Instans Spot di lingkungan Elastic Beanstalk Anda. Ini juga memberikan detail dan contoh untuk metode yang dapat Anda gunakan untuk mengonfigurasi opsi ini. Anda dapat menggunakan konsol Elastic Beanstalk, opsi konfigurasi namespace, CLI, atau AWS CLIEB untuk mengelola opsi konfigurasi.

Minimalkan interupsi instans Spot dengan Penyeimbangan Kembali Kapasitas

Untuk membantu meminimalkan dampak interupsi Instans Spot ke aplikasi, Anda dapat mengaktifkan opsi Penyeimbangan Kembali Kapasitas yang disertakan dengan Amazon Auto Scaling. EC2

> **Important**
>
> Permintaan untuk Instans Spot dapat bervariasi secara signifikan dari waktu ke waktu, dan ketersediaan Instans Spot juga dapat bervariasi secara signifikan tergantung pada berapa banyak instans EC2 Amazon yang tidak digunakan yang tersedia. Selalu ada kemungkinan Instans Spot Anda diinterupsi.

Saat Anda mengaktifkan Rebalancing Kapasitas, EC2 secara otomatis mencoba mengganti Instans Spot di grup Auto Scaling sebelum terputus. Untuk mengaktifkan fitur ini, gunakan konsol Elastic Beanstalk untuk mengonfigurasi grup Auto Scaling. Atau, Anda dapat mengatur `true` opsi konfigurasi Elastic Beanstalk ke dalam `EnableCapacityRebalancing` namespace aws:autoscaling:asg.

Untuk informasi selengkapnya, lihat Penyeimbangan Kembali Kapasitas di Panduan Pengguna Penskalaan Otomatis Amazon EC2 dan Gangguan Instans Spot di Panduan Pengguna Amazon. EC2

Jenis Instance Lama dan Dukungan Instance Spot

Beberapa AWS akun lama mungkin menyediakan Elastic Beanstalk dengan tipe instans default yang tidak mendukung Instans Spot. Jika Anda mengaktifkan permintaan Instans Spot dan melihat kesalahan Tidak ada jenis instans yang Anda tentukan mendukung Spot, perbarui konfigurasi Anda dengan jenis instans yang mendukung Instans Spot. Untuk memilih tipe Instans Spot, gunakan Penasihat Instans Spot.

Topik

- Mengaktifkan Instans Spot untuk lingkungan Anda
- Strategi alokasi Instans Spot
- Mengelola instans On-Demand dan instans Spot
- Konfigurasi kapasitas untuk lingkungan Elastic Beanstalk Anda

Mengaktifkan Instans Spot untuk lingkungan Anda

Untuk memanfaatkan Instans EC2 Spot Amazon, tetapkan `EnableSpot` opsi untuk lingkungan Anda. Grup Auto Scaling lingkungan Anda kemudian menggabungkan opsi EC2 pembelian Amazon dan mempertahankan campuran Instans Sesuai Permintaan dan Spot.

Anda dapat menggunakan konsol Elastic Beanstalk, opsi konfigurasi namespace, CLI, atau EB AWS CLIuntuk mengaktifkan permintaan Instans Spot untuk lingkungan Anda.

Sebelum Anda mengaktifkan Instans Spot untuk lingkungan Anda, kenali opsi konfigurasi Auto Scaling, kapasitas, dan load balancing yang tersedia. Persyaratan aplikasi Anda yang terkait dengan beban kerja, dampak interupsi instans, dan harga, merupakan pertimbangan penting dalam perencanaan Anda untuk mengaktifkan Instans Spot.

Topik berikut memberikan detail tentang Auto Scaling dan opsi manajemen kapasitas dan bagaimana penggunaan gabungannya memengaruhi lingkungan Anda. Ada prosedur dan contoh konfigurasi untuk menginformasikan dan memandu Anda tentang berbagai opsi dan cara mengonfigurasinya. Kami juga menawarkan alat dan fitur untuk membantu Anda mengelola konfigurasi dan menanggapi peristiwa. Anda dapat menjadwalkan perubahan otomatis pada konfigurasi berdasarkan periode lalu lintas yang dapat diprediksi, mengonfigurasi pemicu untuk merespons faktor-faktor seperti volume lalu lintas, dan mengonfigurasi pemantauan Auto Scaling dan pemeriksaan kesehatan.

Untuk informasi selengkapnya tentang Instans Spot, termasuk penjelasan tentang konsep utama dan praktik terbaik, lihat Instans Spot di EC2 Panduan Pengguna Amazon.

> **Important**
>
> Pengaturan `EnableSpot` opsi dapat menyebabkan Elastic Beanstalk memigrasikan lingkungan yang ada dengan konfigurasi peluncuran untuk meluncurkan templat. Melakukannya memerlukan izin yang diperlukan untuk mengelola templat peluncuran. Izin ini disertakan dalam kebijakan terkelola kami. Jika Anda menggunakan kebijakan khusus alih-alih kebijakan terkelola kami, pembuatan atau pembaruan lingkungan mungkin gagal saat Anda memperbarui konfigurasi lingkungan. Untuk informasi lebih lanjut dan pertimbangan lainnya, lihatMemigrasi lingkungan Elastic Beanstalk Anda untuk meluncurkan template .

Strategi alokasi Instans Spot

Anda dapat memilih salah satu strategi alokasi yang tercantum dalam topik ini untuk lingkungan Elastic Beanstalk Anda. Gunakan konsol Elastic Beanstalk, opsi konfigurasi namespace, AWS

CLIatau, untuk mengatur dan mengonfigurasi strategi alokasi Instans Spot dan atribut terkait untuk lingkungan Anda.

Amazon EC2 menerapkan strategi alokasi untuk mengelola dan menyediakan instans Spot untuk lingkungan Anda. Setiap strategi alokasi mengoptimalkan instans yang dialokasikan berdasarkan bagaimana hal itu didefinisikan untuk menangani kapasitas, harga, dan pemilihan jenis instans yang tersedia.

Amazon EC2 Auto Scaling menyediakan strategi alokasi berikut untuk Instans Spot.

- Kapasitas dioptimalkan (default)
  - Meminta Instans Spot dari kumpulan, dengan kapasitas optimal untuk jumlah instans yang diluncurkan.
  - Strategi ini bekerja dengan baik untuk beban kerja di mana kemungkinan gangguan layanan harus diminimalkan.
- Kapasitas harga dioptimalkan
  - Meminta Instans Spot dari kumpulan yang memiliki peluang interupsi terendah dan harga serendah mungkin.
  - Ini adalah pilihan yang lebih disukai untuk sebagian besar beban kerja Spot.
- Kapasitas dioptimalkan diprioritaskan
  - Meminta Instans Spot berdasarkan ketersediaan kapasitas terlebih dahulu, sambil menghormati prioritas jenis instans pilihan Anda dengan upaya terbaik. Anda dapat memberikan daftar jenis instans, yang diurutkan berdasarkan prioritas, saat mengonfigurasi opsi Instance Spot untuk Elastic Beanstalk.
  - Strategi ini bagus untuk beban kerja yang memerlukan gangguan layanan minimal, dan prioritas jenis instance tertentu penting.
- Harga terendah
  - Meminta Instans Spot dari kumpulan harga terendah dengan instans yang tersedia.
  - Penting untuk mengambil tindakan pencegahan saat menggunakan strategi ini, karena hanya mempertimbangkan harga instans dan bukan ketersediaan kapasitas, yang akan menghasilkan tingkat interupsi yang tinggi.

Untuk detail selengkapnya tentang setiap strategi alokasi, lihat Strategi alokasi untuk beberapa jenis instans di Panduan Pengguna Amazon Auto EC2 Scaling.

Untuk membantu Anda memahami strategi alokasi mana yang paling cocok untuk memenuhi persyaratan lingkungan Anda, lihat Memilih strategi alokasi Spot yang sesuai di EC2 Panduan Pengguna Amazon.

Mengelola instans On-Demand dan instans Spot

Anda dapat meluncurkan dan secara otomatis menyesuaikan armada Instans Sesuai Permintaan dan Instans Spot dalam satu grup Auto Scaling. Opsi berikut dapat digunakan bersama-sama untuk mengonfigurasi cara layanan Auto Scaling mengelola Instans Spot dan Instans Sesuai Permintaan di lingkungan Anda.

Anda dapat mengonfigurasi opsi ini untuk lingkungan Anda menggunakan konsol Elastic Beanstalk, opsi konfigurasi namespace, the, atau EB CLI. AWS CLI

Opsi ini adalah bagian dari namespace aws:ec2:instances:

- `EnableSpot`- Ketika disetel ke pengaturan `true` ini memungkinkan permintaan Instans Spot untuk lingkungan Anda.
- `SpotFleetOnDemandBase`- Menetapkan jumlah minimum Instans Sesuai Permintaan yang disediakan grup Auto Scaling Anda sebelum mempertimbangkan Instans Spot saat lingkungan Anda meningkat.
- `SpotFleetOnDemandAboveBasePercentage`- Persentase Instans Sesuai Permintaan sebagai bagian dari kapasitas tambahan yang disediakan grup Auto Scaling Anda di luar instans. `SpotOnDemandBase`

Opsi yang tercantum sebelumnya berkorelasi dengan opsi berikut di ruang nama aws:autoscaling:asg:

- `MinSize`- Jumlah minimum instans yang Anda inginkan dalam grup Auto Scaling Anda.
- `MaxSize`- Jumlah maksimum instans yang Anda inginkan dalam grup Auto Scaling Anda.

> **Important**
>
> Pengaturan `EnableSpot` opsi dapat menyebabkan Elastic Beanstalk memigrasikan lingkungan yang ada dengan konfigurasi peluncuran untuk meluncurkan templat. Melakukannya memerlukan izin yang diperlukan untuk mengelola templat peluncuran. Izin ini disertakan dalam kebijakan terkelola kami. Jika Anda menggunakan kebijakan khusus alih-

alih kebijakan terkelola kami, pembuatan atau pembaruan lingkungan mungkin gagal saat Anda memperbarui konfigurasi lingkungan. Untuk informasi lebih lanjut dan pertimbangan lainnya, lihatMemigrasi lingkungan Elastic Beanstalk Anda untuk meluncurkan template .

Menerapkan kedua set opsi namespace

Poin-poin berikut menjelaskan bagaimana kombinasi pengaturan opsi ini memengaruhi penskalaan untuk lingkungan Anda.

- Hanya `MinSize` menentukan kapasitas awal lingkungan Anda—jumlah instans yang ingin Anda jalankan seminimal mungkin.
- `SpotFleetOnDemandBase`tidak mempengaruhi kapasitas awal. Saat Spot diaktifkan, opsi ini menentukan berapa banyak Instans Sesuai Permintaan yang disediakan sebelum Instans Spot dipertimbangkan.
- Pertimbangkan kapan `SpotFleetOnDemandBase` kurang dari`MinSize`. Anda masih akan mendapatkan `MinSize` instance persis sebagai kapasitas awal. Setidaknya `SpotFleetOnDemandBase` dari mereka harus Instans On-Demand.
- Pertimbangkan kapan `SpotFleetOnDemandBase` lebih besar dari`MinSize`. Saat lingkungan Anda meningkat, Anda dijamin mendapatkan setidaknya jumlah instance tambahan yang sama dengan perbedaan antara kedua nilai tersebut. Dengan kata lain, Anda dijamin mendapatkan setidaknya `(SpotFleetOnDemandBase - MinSize)` contoh tambahan yang Sesuai Permintaan sebelum memenuhi persyaratan. `SpotFleetOnDemandBase`

Lingkungan instans tunggal

Dalam lingkungan produksi, Instans Spot sangat berguna sebagai bagian dari lingkungan dengan beban seimbang yang dapat diskalakan. Kami tidak menyarankan penggunaan Spot di lingkungan instans tunggal. Jika Instans Spot tidak tersedia, Anda mungkin kehilangan seluruh kapasitas (instans tunggal) lingkungan Anda. Anda mungkin masih ingin menggunakan Instans Spot di lingkungan instans tunggal untuk pengembangan atau pengujian. Saat Anda melakukannya, pastikan untuk mengatur `SpotFleetOnDemandBase` dan `SpotFleetOnDemandAboveBasePercentage` ke nol. Pengaturan lainnya menghasilkan Instans Sesuai Permintaan.

Contoh pengaturan opsi penskalaan

Contoh berikut menunjukkan berbagai skenario dari pengaturan berbagai pilihan penskalaan. Semua contoh mengasumsikan lingkungan seimbang beban dengan permintaan Instans Spot diaktifkan.

Example 1: Instans Sesuai Permintaan dan Spot sebagai bagian dari kapasitas awal

Pengaturan opsi

Opsi	Namespace	Nilai
`MinSize`	`aws:autoscaling:asg`	`10`
`MaxSize`	`aws:autoscaling:asg`	`24`
`SpotFleetOnDemandBase`	`aws:ec2:instances`	`4`
`SpotFleetOnDemandAboveBasePercentage`	`aws:ec2:instances`	`50`

Dalam contoh ini, lingkungan dimulai dengan sepuluh instans, tujuh di antaranya adalah Sesuai Permintaan (empat dasar, dan 50% dari enam di atas dasar) dan yang tiga adalah Spot. Lingkungan dapat menskalakan keluar hingga 24 instans. Ketika lingkungan menskalakan keluar, porsi Sesuai Permintaan pada bagian armada di atas instans Sesuai Permintaan empat dasar dipertahankan di 50%, hingga maksimum 24 instans secara keseluruhan, 14 di antaranya adalah Sesuai Permintaan (empat dasar, dan 50% dari 20 di atas dasar) dan yang sepuluh adalah Spot.

Example 2: Semua kapasitas awal Sesuai Permintaan

Pengaturan opsi

Opsi	Namespace	Nilai
`MinSize`	`aws:autoscaling:asg`	`4`
`MaxSize`	`aws:autoscaling:asg`	`24`
`SpotFleetOnDemandBase`	`aws:ec2:instances`	`4`
`SpotFleetOnDemandAboveBasePercentage`	`aws:ec2:instances`	`50`

Dalam contoh ini, lingkungan dimulai dengan empat instans, dan semuanya adalah Sesuai Permintaan. Lingkungan dapat menskalakan keluar hingga 24 instans. Ketika lingkungan menskalakan keluar, porsi Sesuai Permintaan pada bagian armada di atas instans Sesuai Permintaan empat dasar dipertahankan di 50%, hingga maksimum 24 instans secara keseluruhan, 14 di antaranya adalah Sesuai Permintaan (empat dasar, dan 50% dari 20 di atas dasar) dan yang sepuluh adalah Spot.

Example 3: Dasar Sesuai Permintaan Tambahan di luar kapasitas awal

Pengaturan opsi

Opsi	Namespace	Nilai
`MinSize`	`aws:autoscaling:asg`	`3`
`MaxSize`	`aws:autoscaling:asg`	`24`
`SpotFleetOnDemandBase`	`aws:ec2:instances`	`4`
`SpotFleetOnDemandAboveBasePercentage`	`aws:ec2:instances`	`50`

Dalam contoh ini, lingkungan dimulai dengan tiga instans, dan semuanya adalah Sesuai Permintaan. Lingkungan dapat menskalakan keluar hingga 24 instans. Instans tambahan pertama di atas tiga di awal adalah Sesuai Permintaan, untuk menyelesaikan instans Sesuai Permintaan empat dasar. Ketika semakin menskalakan keluar, porsi Sesuai Permintaan di bagian armada di atas instans Sesuai Permintaan empat dasar dipertahankan di 50%, hingga maksimum 24 instans secara keseluruhan, 14 di antaranya adalah Sesuai Permintaan (empat dasar, dan 50% dari 20 di atas dasar) dan yang sepuluh adalah Spot.

Konfigurasi kapasitas untuk lingkungan Elastic Beanstalk Anda

Topik ini menjelaskan berbagai pendekatan untuk mengonfigurasi kapasitas Auto Scaling untuk lingkungan Elastic Beanstalk Anda. Anda dapat menggunakan konsol Elastic Beanstalk, EB CLI, opsi, atau namespace. AWS CLI

> **Important**
>
> Pengaturan `EnableSpot` opsi dapat menyebabkan Elastic Beanstalk memigrasikan lingkungan yang ada dengan konfigurasi peluncuran untuk meluncurkan templat. Melakukannya memerlukan izin yang diperlukan untuk mengelola templat peluncuran. Izin ini disertakan dalam kebijakan terkelola kami. Jika Anda menggunakan kebijakan khusus alih-alih kebijakan terkelola kami, pembuatan atau pembaruan lingkungan mungkin gagal saat Anda memperbarui konfigurasi lingkungan. Untuk informasi lebih lanjut dan pertimbangan lainnya, lihatMemigrasi lingkungan Elastic Beanstalk Anda untuk meluncurkan template .

Konfigurasi menggunakan konsol

Anda dapat mengonfigurasi manajemen kapasitas grup Auto Scaling dengan mengedit Kapasitas pada halaman Konfigurasi lingkungan di konsol Elastic Beanstalk.

Untuk mengonfigurasi kapasitas grup Auto Scaling di konsol Elastic Beanstalk

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Pada kategori konfigurasi Kapasitas, pilih Edit.
5. Di bagian Grup Auto Scaling, konfigurasikan pengaturan berikut.

   - Tipe lingkungan – Pilih Beban yang Diseimbangkan.
   - Min instance — Jumlah minimum EC2 instance yang harus dikandung grup kapan saja. Grup dimulai dengan jumlah minimum dan menambahkan instans ketika kondisi pemicu menaikkan skala tersebut terpenuhi.
   - Instance Max — Jumlah maksimum EC2 instance yang harus dikandung grup kapan saja.

     > **Note**
     >
     > Jika Anda menggunakan pembaruan bergulir, pastikan bahwa jumlah instans maksimum lebih tinggi daripada Instans minimum dalam pengaturan layanan untuk pembaruan bergulir.

- Komposisi armada — Standarnya adalah Instans Sesuai Permintaan. Untuk mengaktifkan permintaan Instans Spot, pilih Opsi pembelian gabungan dan instance.

> **Important**
>
> Pengaturan `EnableSpot` opsi dapat menyebabkan Elastic Beanstalk memigrasikan lingkungan yang ada dengan konfigurasi peluncuran untuk meluncurkan templat. Melakukannya memerlukan izin yang diperlukan untuk mengelola templat peluncuran. Izin ini disertakan dalam kebijakan terkelola kami. Jika Anda menggunakan kebijakan khusus alih-alih kebijakan terkelola kami, pembuatan atau pembaruan lingkungan mungkin gagal saat Anda memperbarui konfigurasi lingkungan. Untuk informasi lebih lanjut dan pertimbangan lainnya, lihatMemigrasi lingkungan Elastic Beanstalk Anda untuk meluncurkan template .

Opsi berikut diaktifkan jika Anda memilih untuk mengaktifkan permintaan Instans Spot:

- Strategi alokasi spot — Menentukan metode yang digunakan untuk mengelola dan menyediakan Instans Spot di lingkungan Anda, berdasarkan kapasitas, harga, dan pemilihan jenis instans yang tersedia. Pilih dari Kapasitas yang dioptimalkan (default), Kapasitas harga dioptimalkan, Kapasitas dioptimalkan diprioritaskan, atau Harga terendah. Untuk deskripsi setiap strategi alokasi dan informasi lebih lanjut, lihatthe section called “Strategi alokasi spot”.
- Harga spot maksimum — Untuk rekomendasi tentang opsi harga maksimum untuk Instans Spot, lihat riwayat harga Instans Spot di Panduan EC2 Pengguna Amazon.
- Basis Sesuai Permintaan — Jumlah minimum Instans Sesuai Permintaan yang disediakan oleh grup Auto Scaling Anda sebelum mempertimbangkan Instans Spot sebagai skala lingkungan Anda.
- Sesuai Permintaan di atas basis — Persentase Instans Sesuai Permintaan sebagai bagian dari kapasitas tambahan apa pun yang disediakan grup Auto Scaling Anda di luar instans dasar On-Demand.

> **Note**
>
> Basis opsi On-Demand dan On-Demand di atas berkorelasi dengan opsi Instans Min dan Maks yang tercantum sebelumnya. Untuk informasi selengkapnya tentang opsi dan contoh ini, lihatthe section called “Dukungan Instans Spot”.

- Penyeimbangan Kembali Kapasitas — Opsi ini hanya relevan jika ada setidaknya satu Instance Spot di grup Auto Scaling Anda. Saat fitur ini diaktifkan, EC2 secara otomatis mencoba mengganti Instans Spot di grup Auto Scaling sebelum terputus, meminimalkan interupsi Instans Spot ke aplikasi Anda. Untuk informasi selengkapnya, lihat Penyeimbangan Kembali Kapasitas di Panduan Pengguna Amazon Auto EC2 Scaling

- Arsitektur — Arsitektur prosesor untuk EC2 instans Anda. Arsitektur prosesor menentukan jenis EC2 Instance yang tersedia di bidang berikutnya.
- Jenis instans — Jenis EC2 instans Amazon yang diluncurkan untuk menjalankan aplikasi Anda. Lihat perinciannya di the section called “Tipe instans”.
- ID AMI - Gambar mesin yang digunakan Elastic Beanstalk untuk EC2 meluncurkan instans Amazon di lingkungan Anda. Lihat perinciannya di the section called “ID AMI”.
- Availability Zone – Memilih jumlah Availability Zone untuk menyebarkan instans lingkungan Anda. Secara default, grup Auto Scaling meluncurkan instans secara merata di semua zona yang dapat digunakan. Untuk memusatkan instans Anda di zona yang lebih sedikit, pilih jumlah zona yang akan digunakan. Untuk lingkungan produksi, gunakan setidaknya dua zona untuk memastikan bahwa aplikasi Anda tersedia jika seandainya satu Availability Zone tidak berfungsi.
- Penempatan (opsional) – Pilih Availability Zone yang akan digunakan. Gunakan pengaturan ini jika instans Anda perlu terhubung ke sumber daya di zona tertentu, atau jika Anda telah membeli instans cadangan, yang merupakan spesifik zona. Jika Anda meluncurkan lingkungan Anda di VPC khusus, Anda tidak dapat mengonfigurasi opsi ini. Dalam VPC khusus, Anda memilih Availability Zone untuk subnet yang Anda tetapkan untuk lingkungan Anda.
- Menskalakan pendinginan – Jumlah waktu, dalam detik, untuk menunggu instans diluncurkan atau diakhiri setelah penskalaan, sebelum melanjutkan mengevaluasi pemicu. Untuk informasi lebih lanjut, lihat Penskalaan Pendinginan.

6. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Konfigurasi menggunakan opsi namespace

Elastic Beanstalk menyediakan opsi konfigurasi untuk pengaturan Auto Scaling dalam dua namespace: `aws:autoscaling:asg` dan `aws:ec2:instances`.

Namespace aws:autoscaling:asg

Namespace `aws:autoscaling:asg` menyediakan pilihan untuk penskalaan keseluruhan dan ketersediaan.

Contoh file konfigurasi berikut mengonfigurasi grup Auto Scaling untuk menggunakan dua sampai empat instans, availability zone tertentu, dan periode pendinginan 12 menit (720 detik). Ini memungkinkan Penyeimbangan Kembali Kapasitas untuk Instans Spot. `EnableCapacityRebalancing`Opsi ini hanya berlaku jika `EnableSpot` diatur ke `true` dalam `aws:ec2:instances`namespace, seperti yang ditunjukkan dalam contoh file konfigurasi berikut ini.

```
option_settings:
  aws:autoscaling:asg:
    Availability Zones: Any
    Cooldown: '720'
    Custom Availability Zones: 'us-west-2a,us-west-2b'
    MaxSize: '4'
    MinSize: '2'
    EnableCapacityRebalancing: true
```

Namespace aws:ec2:instances

> **Note**
>
> Saat memperbarui konfigurasi lingkungan dan menghapus satu atau beberapa jenis instans dari `InstanceTypes` opsi, Elastic Beanstalk menghentikan semua instans Amazon yang berjalan pada EC2 salah satu jenis instans yang dihapus. Grup Auto Scaling lingkungan Anda kemudian meluncurkan instans baru, yang diperlukan untuk melengkapi kapasitas yang diinginkan, menggunakan tipe instans yang ditentukan saat ini.

`aws:ec2:instances`Namespace menyediakan opsi yang terkait dengan instans lingkungan Anda, termasuk manajemen Instans Spot. Ini melengkapi `aws:autoscaling:launchconfiguration`dan `aws:autoscaling:asg`.

Contoh file konfigurasi berikut mengonfigurasi grup Auto Scaling untuk mengaktifkan permintaan Instans Spot untuk lingkungan Anda. Ini menunjuk tiga jenis contoh yang mungkin yang dapat digunakan. Setidaknya satu Instans Sesuai Permintaan digunakan untuk kapasitas dasar, dan 33% Instans Sesuai Permintaan yang berkelanjutan digunakan untuk kapasitas tambahan.

Konfigurasi menetapkan strategi alokasi spot `kecapacity-optimized-prioritized`. Strategi alokasi khusus ini memprioritaskan peluncuran instance dari kumpulan berdasarkan urutan jenis instance yang ditentukan dalam opsi. `InstanceTypes` Jika tidak `SpotAllocationStrategy` ditentukan, defaultnya. `capacity-optimized`

```
option_settings:
  aws:ec2:instances:
    EnableSpot: true
    InstanceTypes: 't2.micro,t3.micro,t3.small'
    SpotAllocationStrategy: capacity-optimized-prioritized
    SpotFleetOnDemandBase: '1'
    SpotFleetOnDemandAboveBasePercentage: '33'
```

Untuk memilih tipe Instans Spot, gunakan Penasihat Instans Spot.

> **Important**
>
> Pengaturan `EnableSpot` opsi dapat menyebabkan Elastic Beanstalk memigrasikan lingkungan yang ada dengan konfigurasi peluncuran untuk meluncurkan templat. Melakukannya memerlukan izin yang diperlukan untuk mengelola templat peluncuran. Izin ini disertakan dalam kebijakan terkelola kami. Jika Anda menggunakan kebijakan khusus alih-alih kebijakan terkelola kami, pembuatan atau pembaruan lingkungan mungkin gagal saat Anda memperbarui konfigurasi lingkungan. Untuk informasi lebih lanjut dan pertimbangan lainnya, lihatMemigrasi lingkungan Elastic Beanstalk Anda untuk meluncurkan template .

Konfigurasi menggunakan AWS CLI

Bagian ini memberikan contoh bagaimana Anda dapat menggunakan perintah AWS CLI create-environment untuk mengonfigurasi lingkungan Anda dengan opsi Auto Scaling dan Kapasitas yang dijelaskan di bagian ini. Anda akan melihat pengaturan namespace untuk `aws:autoscaling:asg`dan `aws:ec2:instances`, seperti yang dijelaskan di bagian opsi konfigurasi namespace sebelumnya juga dikonfigurasi dengan contoh ini.

AWS Command Line Interface menyediakan perintah untuk membuat dan mengkonfigurasi lingkungan Elastic Beanstalk. Dengan `--option-settings` opsi ini, Anda dapat meneruskan opsi namespace yang didukung oleh Elastic Beanstalk. Ini berarti bahwa opsi konfigurasi namespace yang dijelaskan sebelumnya dapat diteruskan ke AWS CLI perintah yang berlaku untuk mengonfigurasi lingkungan Elastic Beanstalk Anda.

> **Note**
>
> Anda juga dapat menggunakan perintah update-environment `--option-settings` untuk menambah atau memperbarui opsi namespace. Jika Anda perlu menghapus opsi namespace dari lingkungan Anda, gunakan update-environment perintah dengan. `--options-to-remove`

Contoh berikut menciptakan lingkungan baru. Lihat opsi konfigurasi namespace topik sebelumnya untuk konteks lebih lanjut tentang opsi yang diteruskan.

Opsi tinju yang tercantum, `IamInstanceProfile` di aws:autoscaling:launchconfiguration namespace, adalah profil instance Elastic Beanstalk. Ini diperlukan saat Anda membuat lingkungan baru.

Example — menciptakan lingkungan dengan opsi Auto Scaling (opsi namespace sebaris)

```
aws elasticbeanstalk create-environment \
--region us-east-1 \
--application-name my-app \
--environment-name my-env \
--solution-stack-name "64bit Amazon Linux 2023 v4.3.0 running Python 3.12" \
--option-settings \
Namespace=aws:autoscaling:launchconfiguration,OptionName=IamInstanceProfile,Value=aws-elasticbeanstalk-ec2-role
Namespace=aws:autoscaling:asg,OptionName=Availability Zones,Value=Any \
Namespace=aws:autoscaling:asg,OptionName=Cooldown,Value=720 \
Namespace=aws:autoscaling:asg,OptionName=Custom Availability Zones,Value=us-west-2a,us-west-2b \
Namespace=aws:autoscaling:asg,OptionName=MaxSize,Value=4 \
Namespace=aws:autoscaling:asg,OptionName=MinSize,Value=2 \
Namespace=aws:autoscaling:asg,OptionName=EnableCapacityRebalancing,Value=true \
Namespace=aws:ec2:instances,OptionName=EnableSpot,Value=true \
Namespace=aws:ec2:instances,OptionName=InstanceTypes,Value=t2.micro,t3.micro,t3.small \
```

```
Namespace=aws:ec2:instances,OptionName=SpotAllocationStrategy,Value=capacity-optimized-
prioritized \
Namespace=aws:ec2:instances,OptionName=SpotFleetOnDemandBase,Value=1 \
Namespace=aws:ec2:instances,OptionName=SpotFleetOnDemandAboveBasePercentage,Value=33
```

> **Important**
>
> Pengaturan `EnableSpot` opsi dapat menyebabkan Elastic Beanstalk memigrasikan lingkungan yang ada dengan konfigurasi peluncuran untuk meluncurkan templat. Melakukannya memerlukan izin yang diperlukan untuk mengelola templat peluncuran. Izin ini disertakan dalam kebijakan terkelola kami. Jika Anda menggunakan kebijakan khusus alih-alih kebijakan terkelola kami, pembuatan atau pembaruan lingkungan mungkin gagal saat Anda memperbarui konfigurasi lingkungan. Untuk informasi lebih lanjut dan pertimbangan lainnya, lihatMemigrasi lingkungan Elastic Beanstalk Anda untuk meluncurkan template .

Sebagai alternatif, gunakan `options.json` file untuk menentukan opsi namespace alih-alih memasukkannya sebaris.

Example —create-environment dengan opsi Auto Scaling (opsi namespace dalam file) **`options.json`**

```
aws elasticbeanstalk create-environment \
--region us-east-1 \
--application-name my-app \
--environment-name my-env \
--solution-stack-name "64bit Amazon Linux 2023 v4.3.0 running Python 3.12"
--option-settings file://options.json
```

Example

```
### example options.json ###
[
    {
        "Namespace": "aws:autoscaling:launchconfiguration",
        "OptionName": "IamInstanceProfile",
        "Value": "aws-elasticbeanstalk-ec2-role"
    },
```

```
    {
        "Namespace": "aws:autoscaling:asg",
        "OptionName": "Availability Zones",
        "Value": "Any"
    },
    {
        "Namespace": "aws:autoscaling:asg",
        "OptionName": "Cooldown",
        "Value": "720"
    },
    {
        "Namespace": "aws:autoscaling:asg",
        "OptionName": "Custom Availability Zones",
        "Value": "us-west-2a,us-west-2b"
    },
    {
        "Namespace": "aws:autoscaling:asg",
        "OptionName": "MaxSize",
        "Value": "4"
    },
    {
        "Namespace": "aws:autoscaling:asg",
        "OptionName": "MinSize",
        "Value": "2"
    },
    {
        "Namespace": "aws:autoscaling:asg",
        "OptionName": "EnableCapacityRebalancing",
        "Value": "true"
    },
    {
        "Namespace": "aws:ec2:instances",
        "OptionName": "EnableSpot",
        "Value": "true"
    },
    {
        "Namespace": "aws:ec2:instances",
        "OptionName": "InstanceTypes",
        "Value": "t2.micro,t3.micro,t3.small"
    },
    {
        "Namespace": "aws:ec2:instances",
        "OptionName": "SpotAllocationStrategy",
        "Value": "capacity-optimized-prioritized"
```

```
    },
    {
        "Namespace": "aws:ec2:instances",
        "OptionName": "SpotFleetOnDemandBase",
        "Value": "1"
    },
    {
        "Namespace": "aws:ec2:instances",
        "OptionName": "SpotFleetOnDemandAboveBasePercentage",
        "Value": "33"
    }
]
```

Konfigurasi menggunakan EB CLI

Saat membuat lingkungan menggunakan eb createperintah, Anda dapat menentukan beberapa opsi yang terkait dengan grup Auto Scaling lingkungan Anda. Ini adalah beberapa opsi yang membantu Anda mengontrol kapasitas lingkungan Anda.

`--single`

Membuat lingkungan dengan satu EC2 instans Amazon dan tanpa penyeimbang beban. Jika Anda tidak menggunakan opsi ini, penyeimbang beban ditambahkan ke lingkungan yang dibuat.

`--enable-spot`

Mengaktifkan permintaan Instans Spot untuk lingkungan Anda.

> **Important**
>
> Pengaturan `enable-spot` opsi dapat menyebabkan Elastic Beanstalk memigrasikan lingkungan yang ada dengan konfigurasi peluncuran untuk meluncurkan templat. Melakukannya memerlukan izin yang diperlukan untuk mengelola templat peluncuran. Izin ini disertakan dalam kebijakan terkelola kami. Jika Anda menggunakan kebijakan khusus alih-alih kebijakan terkelola kami, pembuatan atau pembaruan lingkungan mungkin gagal saat Anda memperbarui konfigurasi lingkungan. Untuk informasi lebih lanjut dan pertimbangan lainnya, lihatMemigrasi lingkungan Elastic Beanstalk Anda untuk meluncurkan template .

Opsi berikut untuk eb createperintah hanya dapat digunakan dengan`--enable-spot`.

`--instance-types`

Daftar jenis EC2 instans Amazon yang Anda ingin lingkungan Anda gunakan.

`--spot-max-price`

Harga maksimum per unit jam, dalam dolar A.S., yang bersedia Anda bayarkan untuk Instans Spot. Untuk rekomendasi tentang opsi harga maksimum untuk Instans Spot, lihat riwayat harga Instans Spot di Panduan EC2 Pengguna Amazon.

`--on-demand-base-capacity`

Jumlah minimum Instans Sesuai Permintaan yang disediakan grup Auto Scaling Anda sebelum mempertimbangkan Instans Spot saat lingkungan Anda bertambah besar.

`--on-demand-above-base-capacity`

Persentase Instans Sesuai Permintaan sebagai bagian dari kapasitas tambahan yang disediakan grup Auto Scaling lebih dari jumlah instans yang ditentukan oleh opsi `--on-demand-base-capacity`.

Contoh berikut membuat lingkungan dan mengonfigurasi grup Auto Scaling untuk mengaktifkan permintaan Instans Spot untuk lingkungan baru. Untuk contoh ini, tiga jenis instance yang mungkin dapat digunakan.

```
$ eb create --enable-spot --instance-types "t2.micro,t3.micro,t3.small"
```

Important

Ada opsi lain dengan nama serupa yang disebut `--instance-type` (tanpa “s”) yang hanya dikenali oleh EB CLI saat memproses Instans Sesuai Permintaan. Jangan gunakan `--instance-type` (tanpa "s") dengan opsi `--enable-spot`. Jika Anda melakukannya, EB CLI mengabaikannya. Alih-alih menggunakan `--instance-types` (dengan "s") dengan opsi `--enable-spot`.

Pemicu Auto Scaling untuk lingkungan Elastic Beanstalk Anda

Grup Auto Scaling di lingkungan Elastic Beanstalk Anda menggunakan dua CloudWatch alarm Amazon untuk memicu operasi penskalaan. Pemicu default menskalakan ketika lalu lintas jaringan keluar rata-rata dari setiap instans lebih tinggi dari 6 MB atau lebih rendah dari 2 MB selama periode

lima menit. Untuk menggunakan Amazon EC2 Auto Scaling secara efektif, konfigurasikan pemicu yang sesuai untuk aplikasi, jenis instans, dan persyaratan layanan Anda. Anda dapat menskalakan berdasar beberapa statistik termasuk latensi, I/O disk, utilisasi CPU, dan jumlah permintaan.

Untuk informasi selengkapnya tentang CloudWatch metrik dan alarm, lihat CloudWatch Konsep Amazon di CloudWatch Panduan Pengguna Amazon.

Mengonfigurasi pemicu Auto Scaling

Anda dapat mengonfigurasi pemicu yang menyesuaikan jumlah instans di grup Auto Scaling lingkungan Anda di konsol Elastic Beanstalk.

Untuk mengonfigurasi pemicu di konsol Elastic Beanstalk

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Pada kategori konfigurasi Kapasitas, pilih Edit.
5. Di bagian Pemicu penskalaan, konfigurasikan pengaturan berikut:

   - Metrik – Metrik yang digunakan untuk pemicu Auto Scaling.
   - Statistik – Perhitungan statistik yang harus digunakan pemicu, seperti `Average`.
   - Unit – Unit untuk metrik pemicu, seperti Byte.
   - Periode — Menentukan seberapa sering Amazon CloudWatch mengukur metrik untuk pemicu Anda.
   - Durasi pelanggaran – Jumlah waktu, dalam hitungan menit, metrik bisa saja berada di luar ambang batas atas dan bawah sebelum memicu operasi penskalaan.
   - Ambang batas atas – Jika metrik melebihi jumlah durasi pelanggaran, operasi penskalaan akan dipicu.
   - Peningkatan skala — Jumlah EC2 instans Amazon yang akan ditambahkan saat melakukan aktivitas penskalaan.
   - Ambang batas bawah – Jika metrik berada di bawah angka ini selama durasi pelanggaran, operasi penskalaan akan dipicu.
   - Peningkatan skala - Jumlah EC2 instans Amazon yang akan dihapus saat melakukan aktivitas penskalaan.

6. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Namespace aws:autoscaling:trigger

Elastic Beanstalk menyediakan opsi konfigurasi untuk pengaturan Auto Scaling dalam dua namespace `aws:autoscaling:trigger`. Pengaturan dalam namespace ini diatur oleh sumber daya yang diterapkan.

```
option_settings:
  AWSEBAutoScalingScaleDownPolicy.aws:autoscaling:trigger:
    LowerBreachScaleIncrement: '-1'
  AWSEBAutoScalingScaleUpPolicy.aws:autoscaling:trigger:
    UpperBreachScaleIncrement: '1'
  AWSEBCloudwatchAlarmHigh.aws:autoscaling:trigger:
    UpperThreshold: '6000000'
  AWSEBCloudwatchAlarmLow.aws:autoscaling:trigger:
    BreachDuration: '5'
    EvaluationPeriods: '1'
    LowerThreshold: '2000000'
    MeasureName: NetworkOut
    Period: '5'
    Statistic: Average
    Unit: Bytes
```

Tindakan Auto Scaling Terjadwal untuk lingkungan Elastic Beanstalk Anda

Untuk mengoptimalkan penggunaan EC2 instans Amazon di lingkungan Anda melalui periode lalu lintas puncak yang dapat diprediksi, konfigurasikan grup EC2 Auto Scaling Amazon Anda untuk mengubah jumlah instance-nya sesuai jadwal. Anda dapat mengonfigurasi lingkungan Anda dengan tindakan berulang untuk menaikkan skala setiap pagi hari, dan menurunkan skala pada malam hari ketika lalu lintas rendah. Misalnya, jika Anda memiliki acara pemasaran yang akan mengarahkan lalu lintas ke situs Anda untuk jangka waktu terbatas, Anda dapat menjadwalkan acara yang akan diadakan sekali untuk menaikkan skala saat dimulai, dan yang lainnya untuk menurunkan skala saat berakhir.

Anda dapat menentukan hingga 120 tindakan terjadwal yang aktif per lingkungan. Elastic Beanstalk juga mempertahankan hingga 150 tindakan terjadwal yang kedaluwarsa, yang dapat digunakan kembali dengan memperbarui pengaturannya.

Mengonfigurasi tindakan terjadwal

Anda dapat membuat tindakan terjadwal untuk grup Auto Scaling lingkungan Anda di konsol Elastic Beanstalk.

Untuk mengonfigurasi tindakan terjadwal di konsol Elastic Beanstalk

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Pada kategori konfigurasi Kapasitas, pilih Edit.
5. Di bagian Penskalaan berbasis waktu, pilih Tambahkan tindakan terjadwal.
6. Isi pengaturan tindakan terjadwal berikut:

   - Name – Tentukan nama unik maksimal 255 karakter alfanumerik, tanpa spasi.
   - Instans – Pilih jumlah instans minimum dan maksimum untuk diterapkan ke grup Auto Scaling.
   - Kapasitas yang diinginkan (opsional) – Atur kapasitas awal yang diinginkan pada grup Auto Scaling. Setelah tindakan terjadwal diterapkan, pemicu menyesuaikan kapasitas yang diinginkan berdasarkan pengaturannya.
   - Kejadian – Memilih Berulang untuk mengulangi tindakan penskalaan dalam jadwal.
   - Waktu mulai – Untuk tindakan yang hanya dilakukan sekali, pilih tanggal dan waktu untuk menjalankan tindakan.

     Untuk tindakan berulang, waktu mulai adalah opsional. Tentukan waktu mulai guna memilih waktu paling awal untuk dilakukan tindakan. Setelahnya, tindakan akan berulang sesuai dengan ekspresi Pengulangan.
   - Pengulangan – Gunakan ekspresi Cron untuk menentukan frekuensi yang Anda inginkan agar tindakan terjadwal terjadi. Sebagai contoh, `30 6 * * 2` menjalankan tindakan setiap hari Selasa UTC 6:30.
   - Waktu akhir (opsional) – Opsional untuk tindakan berulang. Jika ditentukan, tindakan akan berulang sesuai dengan ekspresi Pengulangan, dan tidak dilakukan lagi setelah saat ini.

     Saat tindakan terjadwal berakhir, Auto Scaling tidak secara otomatis kembali ke pengaturan sebelumnya. Konfigurasikan tindakan terjadwal kedua untuk mengembalikan Auto Scaling ke pengaturan asli sesuai kebutuhan.

7. Pilih Tambahkan.
8. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

> **Note**
>
> Tindakan terjadwal tidak akan disimpan sampai tindakan itu diterapkan.

Namespace aws:autoscaling:scheduledaction

Jika Anda perlu mengonfigurasi sejumlah besar tindakan terjadwal, Anda dapat menggunakan file konfigurasi atau API Elastic Beanstalk untuk menerapkan perubahan opsi konfigurasi dari file JSON atau YAML. Metode ini juga memungkinkan Anda mengakses opsi `Suspend` untuk menonaktifkan sementara tindakan terjadwal berulang.

> **Note**
>
> Ketika bekerja dengan opsi konfigurasi tindakan terjadwal di luar konsol, gunakan format waktu ISO 8601 untuk menentukan waktu mulai dan akhir dalam UTC. Sebagai contoh, 2015-04-28T04:07:02Z. Untuk informasi selengkapnya tentang format waktu ISO 8601, lihat Format Tanggal dan Waktu. Tanggal pada semua tindakan terjadwal harus unik.

Elastic Beanstalk menyediakan opsi konfigurasi untuk pengaturan tindakan terjadwal pada namespace `aws:autoscaling:scheduledaction`. Gunakan bidang `resource_name` untuk menentukan nama tindakan terjadwal.

Example S cheduled-scale-up-specific -time-long.config

File konfigurasi ini menginstruksikan Elastic Beanstalk untuk menskalakan keluar dari lima instans ke 10 instans pada 2015-12-12T00:00:00Z.

```
option_settings:
  - namespace: aws:autoscaling:scheduledaction
    resource_name: ScheduledScaleUpSpecificTime
    option_name: MinSize
    value: '5'
  - namespace: aws:autoscaling:scheduledaction
    resource_name: ScheduledScaleUpSpecificTime
```

```
    option_name: MaxSize
    value: '10'
  - namespace: aws:autoscaling:scheduledaction
    resource_name: ScheduledScaleUpSpecificTime
    option_name: DesiredCapacity
    value: '5'
  - namespace: aws:autoscaling:scheduledaction
    resource_name: ScheduledScaleUpSpecificTime
    option_name: StartTime
    value: '2015-12-12T00:00:00Z'
```

Example S cheduled-scale-up-specific -waktu.config

Untuk menggunakan sintaksis singkatan dengan EB CLI atau file konfigurasi, tambahkan nama sumber daya ke namespace.

```
option_settings:
  ScheduledScaleUpSpecificTime.aws:autoscaling:scheduledaction:
    MinSize: '5'
    MaxSize: '10'
    DesiredCapacity: '5'
    StartTime: '2015-12-12T00:00:00Z'
```

Example S cheduled-scale-down-specific -waktu.config

File konfigurasi ini menginstruksikan Elastic Beanstalk untuk menskalakan ke dalam pada 2015-12-12T07:00:00Z.

```
option_settings:
  ScheduledScaleDownSpecificTime.aws:autoscaling:scheduledaction:
    MinSize: '1'
    MaxSize: '1'
    DesiredCapacity: '1'
    StartTime: '2015-12-12T07:00:00Z'
```

Example S cheduled-periodic-scale-up .config

File konfigurasi ini menginstruksikan Elastic Beanstalk untuk menskalakan keluar setiap hari pukul 9 pagi. Tindakan ini dijadwalkan akan dimulai 14 Mei 2015 dan berakhir 12 Januari 2016.

```
option_settings:
```

```
  ScheduledPeriodicScaleUp.aws:autoscaling:scheduledaction:
    MinSize: '5'
    MaxSize: '10'
    DesiredCapacity: '5'
    StartTime: '2015-05-14T07:00:00Z'
    EndTime: '2016-01-12T07:00:00Z'
    Recurrence: 0 9 * * *
```

Example S cheduled-periodic-scale-down .config

File konfigurasi ini menginstruksikan Elastic Beanstalk untuk menskalakan ke dalam untuk tidak menjalankan instans setiap hari pukul 6 pagi. Jika Anda tahu bahwa aplikasi Anda sebagian besar tidak digunakan di luar jam kerja, Anda dapat membuat tindakan terjadwal serupa. Jika aplikasi Anda harus dimatikan di luar jam kerja, ubah `MaxSize` ke `0`.

```
option_settings:
  ScheduledPeriodicScaleDown.aws:autoscaling:scheduledaction:
    MinSize: '0'
    MaxSize: '1'
    DesiredCapacity: '0'
    StartTime: '2015-05-14T07:00:00Z'
    EndTime: '2016-01-12T07:00:00Z'
    Recurrence: 0 18 * * *
```

Example S cheduled-weekend-scale-down .config

File konfigurasi ini menginstruksikan Elastic Beanstalk untuk menskalakan ke dalam setiap hari Jumat pukul 6 sore. Jika Anda tahu bahwa aplikasi Anda tidak menerima banyak lalu lintas selama akhir pekan, Anda dapat membuat tindakan terjadwal serupa.

```
option_settings:
  ScheduledWeekendScaleDown.aws:autoscaling:scheduledaction:
    MinSize: '1'
    MaxSize: '4'
    DesiredCapacity: '1'
    StartTime: '2015-12-12T07:00:00Z'
    EndTime: '2016-01-12T07:00:00Z'
    Recurrence: 0 18 * * 5
```

Pengaturan pemeriksaan kesehatan Auto Scaling untuk lingkungan Elastic Beanstalk Anda

Amazon EC2 Auto Scaling memantau kesehatan setiap instans Amazon Elastic Compute Cloud (Amazon EC2) yang diluncurkan. Jika ada instans yang berakhir tiba-tiba, Auto Scaling mendeteksi penghentian dan meluncurkan instans pengganti. Secara default, grup Auto Scaling yang dibuat untuk lingkungan Anda menggunakan pemeriksaan EC2 status Amazon. Jika instance di lingkungan Anda gagal dalam pemeriksaan EC2 status Amazon, Auto Scaling akan menurunkannya dan menggantinya.

Pemeriksaan EC2 status Amazon hanya mencakup kesehatan instans, bukan kesehatan aplikasi, server, atau container Docker apa pun yang berjalan pada instance. Jika aplikasi Anda crash, namun instans yang menjalankannya masih dalam kondisi baik, mungkin aplikasi tersebut akan dikeluarkan dari penyeimbang beban, namun Auto Scaling tidak akan menggantikannya secara otomatis. Perilaku default baik untuk pemecahan masalah. Jika Auto Scaling mengganti instans segera setelah aplikasi crash, Anda mungkin tidak akan menyadari bahwa ada yang tidak beres, bahkan jika crash terjadi segera setelah memulai.

Jika Anda ingin Auto Scaling untuk menggantikan instans aplikasi yang telah berhenti merespons, Anda dapat menggunakan file konfigurasi untuk mengonfigurasi grup Auto Scaling untuk menggunakan pemeriksaan kondisi Elastic Load Balancing. Contoh berikut menetapkan grup untuk menggunakan pemeriksaan kesehatan penyeimbang beban, selain pemeriksaan EC2 status Amazon, untuk menentukan kesehatan instans.

Example .ebextensions/autoscaling.config

```
Resources:
  AWSEBAutoScalingGroup:
    Type: "AWS::AutoScaling::AutoScalingGroup"
    Properties:
      HealthCheckType: ELB
      HealthCheckGracePeriod: 300
```

Untuk informasi selengkapnya tentang `HealthCheckGracePeriod` properti `HealthCheckType` dan, lihat AWS::AutoScaling::AutoScalingKelompokkan dalam Panduan AWS CloudFormation Pengguna dan Pemeriksaan Kesehatan untuk Instans Penskalaan Otomatis di Panduan Pengguna Penskalaan EC2 Otomatis Amazon.

Secara default, pemeriksaan kondisi Elastic Load Balancing dikonfigurasi untuk mencoba koneksi TCP ke instans Anda melalui port 80. Ini mengonfirmasi bahwa server web yang berjalan pada instans menerima koneksi. Namun, Anda mungkin ingin menyesuaikan pemeriksaan kondisi penyeimbang beban untuk memastikan bahwa aplikasi Anda, dan bukan hanya web server, dalam keadaan baik. Pengaturan masa tenggang menyediakan kesempatan waktu dalam jumlah detik saat instans bisa saja gagal dalam pemeriksaan kondisi tanpa diakhiri dan diganti. Instans dapat pulih setelah dikeluarkan dari penyeimbang beban, jadi berikanlah instans sejumlah waktu yang sesuai untuk aplikasi Anda.

Penyeimbang beban untuk lingkungan Elastic Beanstalk Anda

Penyeimbang beban mendistribusikan lalu lintas di antara instans lingkungan Anda. Saat Anda mengaktifkan load balancing, AWS Elastic Beanstalk buat penyeimbang beban Elastic Load Balancing yang didedikasikan untuk lingkungan Anda. Elastic Beanstalk sepenuhnya mengelola penyeimbang beban ini, menjaga pengaturan keamanan dan menghentikan penyeimbang beban saat Anda menghentikan lingkungan Anda.

Atau, Anda dapat memilih untuk berbagi penyeimbang beban di beberapa lingkungan Elastic Beanstalk. Dengan penyeimbang beban bersama, Anda menghemat biaya operasional dengan menghindari penyeimbang beban khusus untuk setiap lingkungan. Anda juga menanggung lebih banyak tanggung jawab manajemen untuk penyeimbang beban bersama yang digunakan lingkungan Anda.

Elastic Load Balancing memiliki tipe penyeimbang beban ini:

- Classic Load Balancer — Penyeimbang beban generasi sebelumnya. Merutekan lalu lintas permintaan HTTP, HTTPS, atau TCP ke port yang berbeda pada instans lingkungan.
- Application Load Balancer — Penyeimbang beban lapisan aplikasi. Merutekan lalu lintas permintaan HTTP atau HTTPS ke port yang berbeda pada instans lingkungan berdasarkan jalur permintaan.
- Network Load Balancer — Penyeimbang beban lapisan jaringan. Merutekan lalu lintas permintaan TCP ke port yang berbeda pada instans lingkungan. Mendukung pemeriksaan kondisi aktif dan pasif.

Elastic Beanstalk mendukung ketiga tipe penyeimbang beban. Tabel berikut menunjukkan tipe apa yang dapat Anda gunakan dengan dua pola penggunaan:

Tipe penyeimbang beban	Khusus	Bersama
Classic Load Balancer	✓ Ya	X Tidak
Application Load Balancer	✓ Ya	✓ Ya
Network Load Balancer	✓ Ya	X Tidak

> **Note**
>
> Opsi Classic Load Balancer (CLB) dinonaktifkan pada wizard konsol Create Environment. Jika Anda memiliki lingkungan yang sudah ada yang dikonfigurasi dengan Classic Load Balancer, Anda dapat membuat yang baru dengan mengkloning lingkungan yang ada menggunakan konsol Elastic Beanstalk atau EB CLI. Anda juga memiliki opsi untuk menggunakan EB CLI atau untuk membuat lingkungan baru AWS CLIyang dikonfigurasi dengan Classic Load Balancer. Alat baris perintah ini akan menciptakan lingkungan baru dengan CLB meskipun belum ada di akun Anda.

Secara default, Elastic Beanstalk membuat Application Load Balancer untuk lingkungan Anda saat Anda mengaktifkan penyeimbangan beban dengan konsol Elastic Beanstalk atau EB CLI. EB CLI mengonfigurasi penyeimbang beban untuk mendengarkan lalu lintas HTTP pada port 80 dan meneruskan lalu lintas ini ke instans pada port yang sama. Anda dapat memilih tipe penyeimbang beban yang hanya digunakan lingkungan Anda selama pembuatan lingkungan. Nantinya, Anda dapat mengubah pengaturan untuk mengelola perilaku penyeimbang beban lingkungan berjalan Anda, tetapi Anda tidak dapat mengubah tipenya.

> **Note**
>
> Lingkungan Anda harus berada dalam VPC dengan subnet di setidaknya dua Availability Zone untuk membuat Application Load Balancer. Semua AWS akun baru menyertakan default VPCs yang memenuhi persyaratan ini.

Lihat topik berikut untuk mempelajari tentang setiap tipe penyeimbang beban yang didukung Elastic Beanstalk, fungsinya, cara mengonfigurasi dan mengelolanya di lingkungan Elastic Beanstalk, dan cara mengonfigurasi penyeimbang beban untuk mengunggah log akses ke Amazon S3.

Topik

- Mengonfigurasi Classic Load Balancer
- Mengonfigurasi Application Load Balancer
- Mengonfigurasi Application Load Balancer bersama
- Mengonfigurasi Network Load Balancer
- Mengonfigurasi log akses

Mengonfigurasi Classic Load Balancer

Saat Anda mengaktifkan load balancing, AWS Elastic Beanstalk lingkungan Anda dilengkapi dengan penyeimbang beban Elastic Load Balancing untuk mendistribusikan lalu lintas antar instans di lingkungan Anda. Elastic Load Balancing mendukung beberapa tipe penyeimbang beban. Untuk mempelajarinya, lihat Panduan Pengguna Elastic Load Balancing. Elastic Beanstalk dapat membuat penyeimbang beban untuk Anda, atau memungkinkan Anda menentukan penyeimbang beban bersama yang telah Anda buat.

Topik ini menjelaskan konfigurasi Classic Load Balancer yang dibuat Elastic Beanstalk dan didedikasikan untuk lingkungan Anda. Untuk informasi tentang cara mengonfigurasi semua tipe penyeimbang beban yang didukung Elastic Beanstalk, lihat Penyeimbang beban untuk lingkungan Elastic Beanstalk Anda.

> **Note**
>
> Anda dapat memilih tipe penyeimbang beban yang hanya digunakan lingkungan Anda selama pembuatan lingkungan. Nantinya, Anda dapat mengubah pengaturan untuk mengelola perilaku penyeimbang beban lingkungan berjalan Anda, tetapi Anda tidak dapat mengubah tipenya.

Pengantar

Classic Load Balancer adalah penyeimbang beban Elastic Load Balancing generasi sebelumnya. Classic Load Balancer mendukung perutean lalu lintas permintaan HTTP, HTTPS, atau TCP ke port yang berbeda pada instans lingkungan.

Saat lingkungan Anda menggunakan Classic Load Balancer, Elastic Beanstalk mengonfigurasinya secara default untuk mendengarkan lalu lintas HTTP pada port 80 dan meneruskannya ke instans

pada port yang sama. Meskipun Anda tidak dapat menghapus pendengar default port 80, Anda dapat menonaktifkannya, yang mencapai fungsionalitas yang sama dengan memblokir lalu lintas. Perhatikan bahwa Anda dapat menambah atau menghapus pendengar lain. Untuk mendukung koneksi yang aman, Anda dapat mengonfigurasi penyeimbang beban Anda dengan listener pada port 443 dan sertifikat TLS.

Penyeimbang beban menggunakan pemeriksaan kesehatan untuk menentukan apakah EC2 instans Amazon yang menjalankan aplikasi Anda sehat. Pemeriksaan kondisi membuat permintaan ke URL tertentu pada interval yang ditetapkan. Jika URL mengembalikan pesan kesalahan, atau gagal untuk kembali dalam jangka waktu tertentu, pemeriksaan kondisi gagal.

Jika aplikasi Anda melakukan lebih baik dengan melayani beberapa permintaan dari klien yang sama pada satu server, Anda dapat mengonfigurasi penyeimbang beban Anda untuk menggunakan sesi lekat. Dengan sesi lengket, penyeimbang beban menambahkan cookie ke respons HTTP yang mengidentifikasi EC2 instans Amazon yang melayani permintaan. Ketika permintaan berikutnya diterima dari klien yang sama, penyeimbang beban menggunakan cookie untuk mengirim permintaan ke instans yang sama.

Dengan penyeimbangan beban lintas zona, setiap simpul penyeimbang beban untuk Classic Load Balancer Anda mendistribusikan permintaan secara merata ke seluruh instans terdaftar di semua Availability Zone yang diaktifkan. Jika penyeimbangan beban lintas zona dinonaktifkan, setiap simpul penyeimbang beban mendistribusikan permintaan secara merata di seluruh instans terdaftar di Availability Zone saja.

Ketika sebuah instans dihapus dari penyeimbang beban karena instans menjadi tidak sehat atau lingkungan menurun, pengurasan koneksi memberikan waktu instans untuk menyelesaikan permintaan sebelum menutup koneksi antara instans dan penyeimbang beban. Anda dapat mengubah jumlah waktu yang diberikan kepada instans untuk mengirim respons, atau menonaktifkan pengurasan koneksi sepenuhnya.

> **Note**
>
> Pengurasan koneksi diaktifkan secara default saat Anda membuat lingkungan dengan konsol Elastic Beanstalk atau EB CLI. Untuk klien lain, Anda dapat mengaktifkan pengurasan koneksi dengan opsi konfigurasi.

Anda dapat menggunakan pengaturan penyeimbang beban lanjutan untuk mengonfigurasi pendengar pada port arbitrer, memodifikasi pengaturan sesi lengket tambahan, dan mengonfigurasi

penyeimbang beban untuk terhubung ke instance dengan aman. EC2 Pengaturan ini tersedia melalui opsi konfigurasi yang dapat Anda atur menggunakan file konfigurasi dalam kode sumber Anda, atau langsung pada lingkungan dengan menggunakan API Elastic Beanstalk. Banyak dari pengaturan ini juga tersedia di konsol Elastic Beanstalk. Selain itu, Anda dapat mengonfigurasi penyeimbang beban untuk mengunggah log akses ke Amazon S3.

Mengonfigurasi Classic Load Balancer menggunakan konsol Elastic Beanstalk

Anda dapat menggunakan konsol Elastic Beanstalk untuk mengonfigurasi port Classic Load Balancer, sertifikat HTTPS, dan pengaturan lainnya, selama pembuatan lingkungan atau nanti saat lingkungan Anda berjalan.

> **Note**
>
> Opsi Classic Load Balancer (CLB) dinonaktifkan pada wizard konsol Create Environment. Jika Anda memiliki lingkungan yang sudah ada yang dikonfigurasi dengan Classic Load Balancer, Anda dapat membuat yang baru dengan mengkloning lingkungan yang ada menggunakan konsol Elastic Beanstalk atau EB CLI. Anda juga memiliki opsi untuk menggunakan EB CLI atau untuk membuat lingkungan baru AWS CLIyang dikonfigurasi dengan Classic Load Balancer. Alat baris perintah ini akan menciptakan lingkungan baru dengan CLB meskipun belum ada di akun Anda.

Untuk mengonfigurasi Classic Load Balancer di lingkungan yang sedang berjalan di konsol Elastic Beanstalk

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Pada kategori konfigurasi Penyeimbang beban, pilih Edit.

> **Note**
>
> Jika kategori konfigurasi Penyeimbang beban tidak memiliki tombol Edit, lingkungan Anda tidak memiliki penyeimbang beban. Untuk mempelajari cara menyiapkan penyeimbang beban, lihat Mengubah jenis lingkungan.

5. Buat perubahan konfigurasi Classic Load Balancer yang diperlukan lingkungan Anda.
6. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Pengaturan Classic Load Balancer

- Listener
- Sesi
- Penyeimbangan beban lintas zona
- Pengurasan koneksi
- Pemeriksaan kondisi

Listener

Gunakan daftar ini untuk menentukan pendengar untuk penyeimbang beban Anda. Setiap pendengar merutekan lalu lintas klien yang masuk pada port tertentu menggunakan protokol tertentu untuk instans Anda. Awalnya, daftar menunjukkan pendengar default, yang merutekan lalu lintas HTTP masuk pada port 80 untuk server instans lingkungan Anda yang mendengarkan HTTP lalu lintas pada port 80.

> **Note**
>
> Meskipun Anda tidak dapat menghapus pendengar default port 80, Anda dapat menonaktifkannya, yang mencapai fungsionalitas yang sama dengan memblokir lalu lintas.

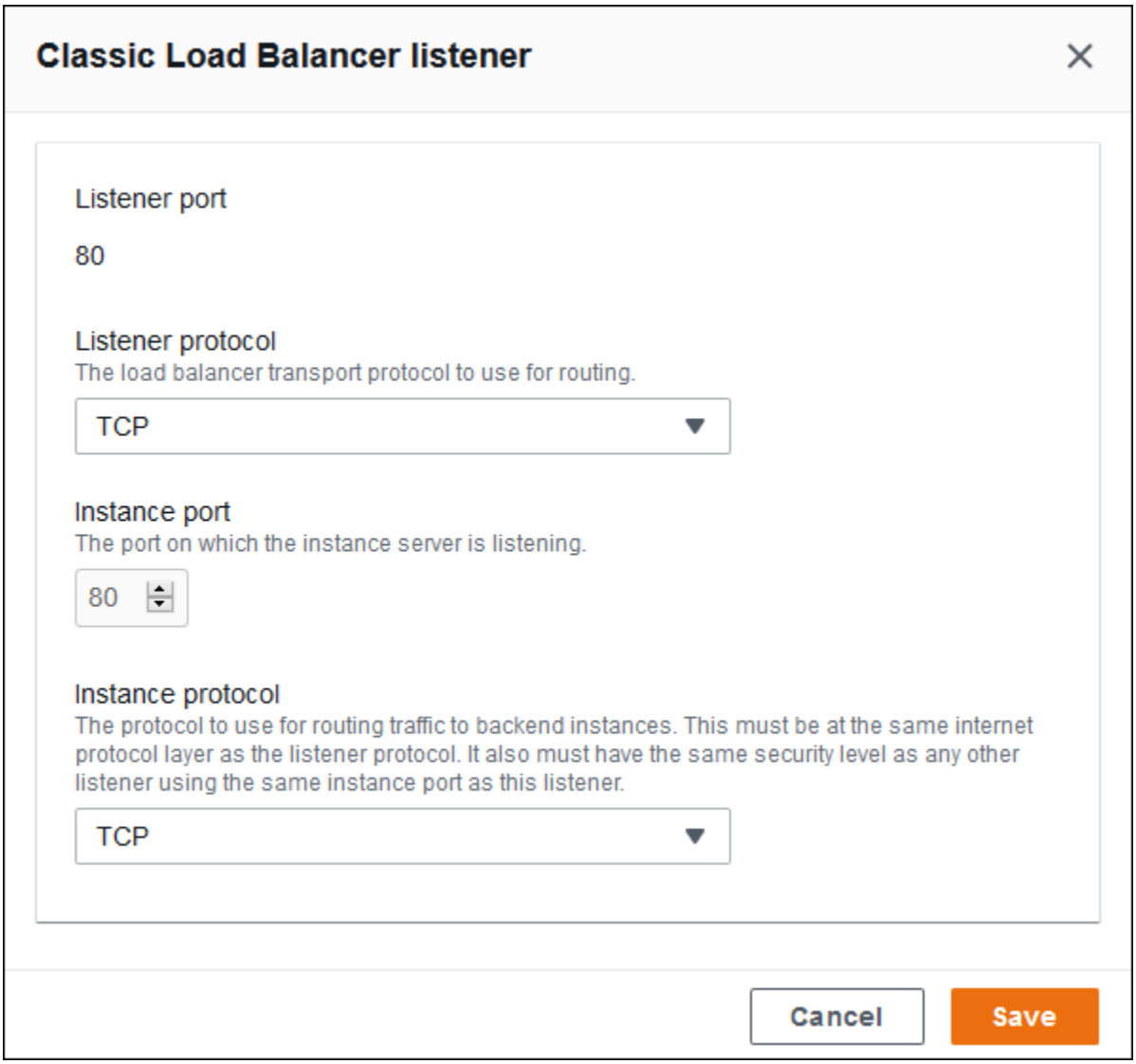


Untuk mengonfigurasi pendengar yang ada

1. Pilih kotak centang di samping entri tabel, pilih Tindakan, lalu pilih tindakan yang Anda inginkan.

2. Jika Anda memilih Edit, gunakan kotak dialog pendengar Classic Load Balancer untuk mengedit pengaturan, dan kemudian pilih Simpan.

Misalnya, Anda dapat mengedit pendengar default dan mengubah Protokol dari HTTP ke TCP jika Anda ingin penyeimbang beban meneruskan permintaan sebagaimana adanya. Hal ini mencegah penyeimbang beban menulis ulang header (termasuk `X-Forwarded-For`). Teknik ini tidak bekerja pada sesi lekat.



Untuk menambahkan pendengar

1. Pilih Tambahkan pendengar.
2. Di kotak dialog Classic Load Balancer, konfigurasikan pengaturan yang Anda inginkan, dan kemudian pilih Tambahkan.

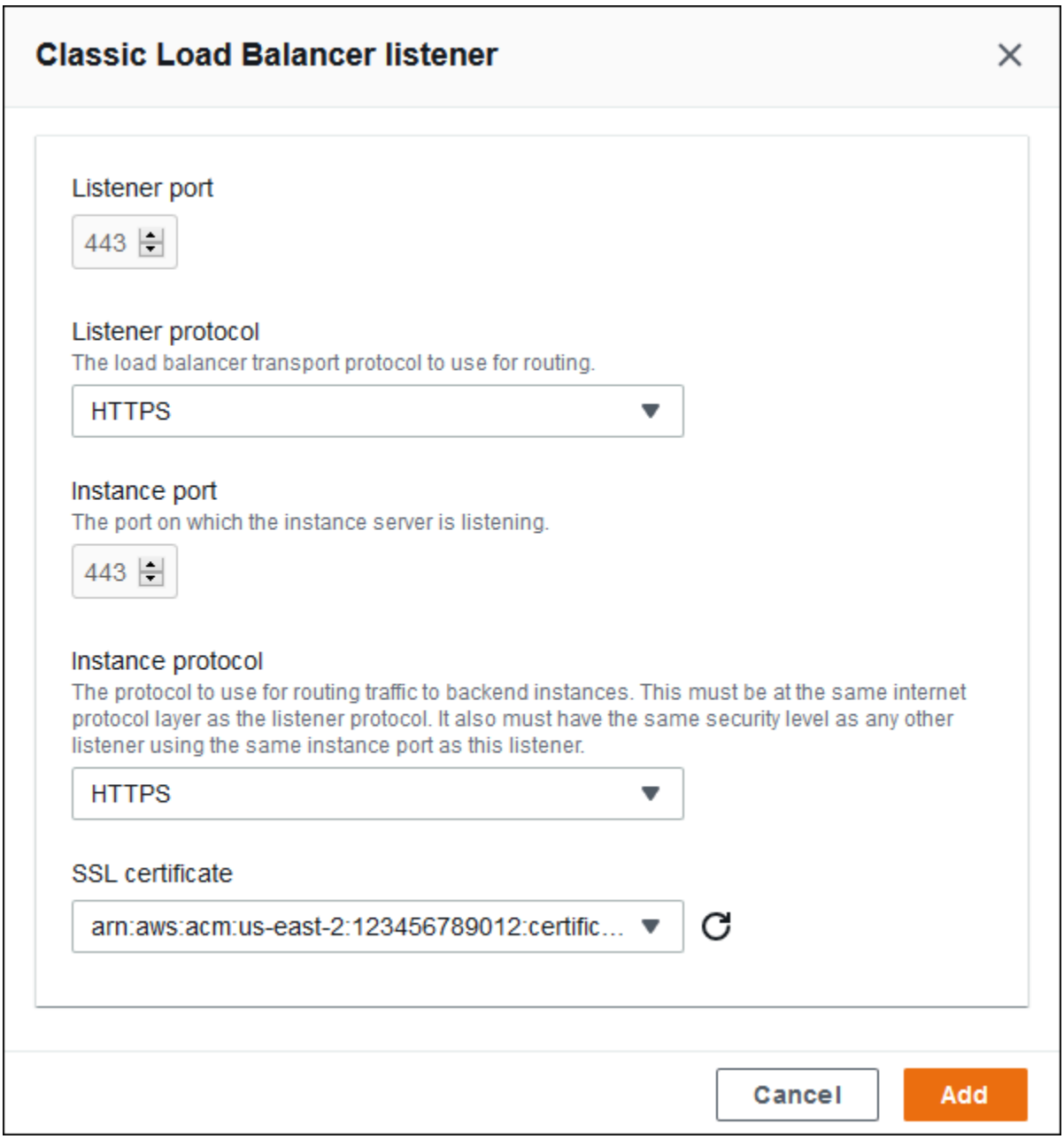
Menambahkan pendengar yang aman adalah kasus penggunaan umum. Contoh pada gambar berikut menambahkan pendengar untuk lalu lintas HTTPS pada port 443. Pendengar ini merutekan lalu lintas masuk ke server isntans lingkungan yang mendengarkan lalu lintas HTTPS pada port 443.

Sebelum Anda dapat mengonfigurasi pendengar HTTPS, pastikan bahwa Anda memiliki sertifikat SSL yang valid. Lakukan salah satu tindakan berikut:

- Jika AWS Certificate Manager (ACM) tersedia di AWS Wilayah Anda, buat atau impor sertifikat menggunakan ACM. Untuk informasi selengkapnya tentang permintaan sertifikat ACM, lihat Meminta Sertifikat di AWS Certificate Manager Panduan Pengguna. Untuk informasi tentang mengimpor sertifikat pihak ke tiga ke ACM, lihat Mengimpor Sertifikat di Panduan Pengguna AWS Certificate Manager .
- Jika ACM tidak tersedia di AWS Wilayah Anda, unggah sertifikat dan kunci yang ada ke IAM. Untuk informasi selengkapnya tentang pembuatan dan pengunggahan sertifikat ke IAM, lihat Bekerja dengan Sertifikat Server di Panduan Pengguna IAM.

Untuk detail lebih lanjut tentang mengonfigurasi HTTPS dan bekerja dengan sertifikat di Elastic Beanstalk, lihat Mengonfigurasi HTTPS untuk lingkungan Elastic Beanstalk Anda.

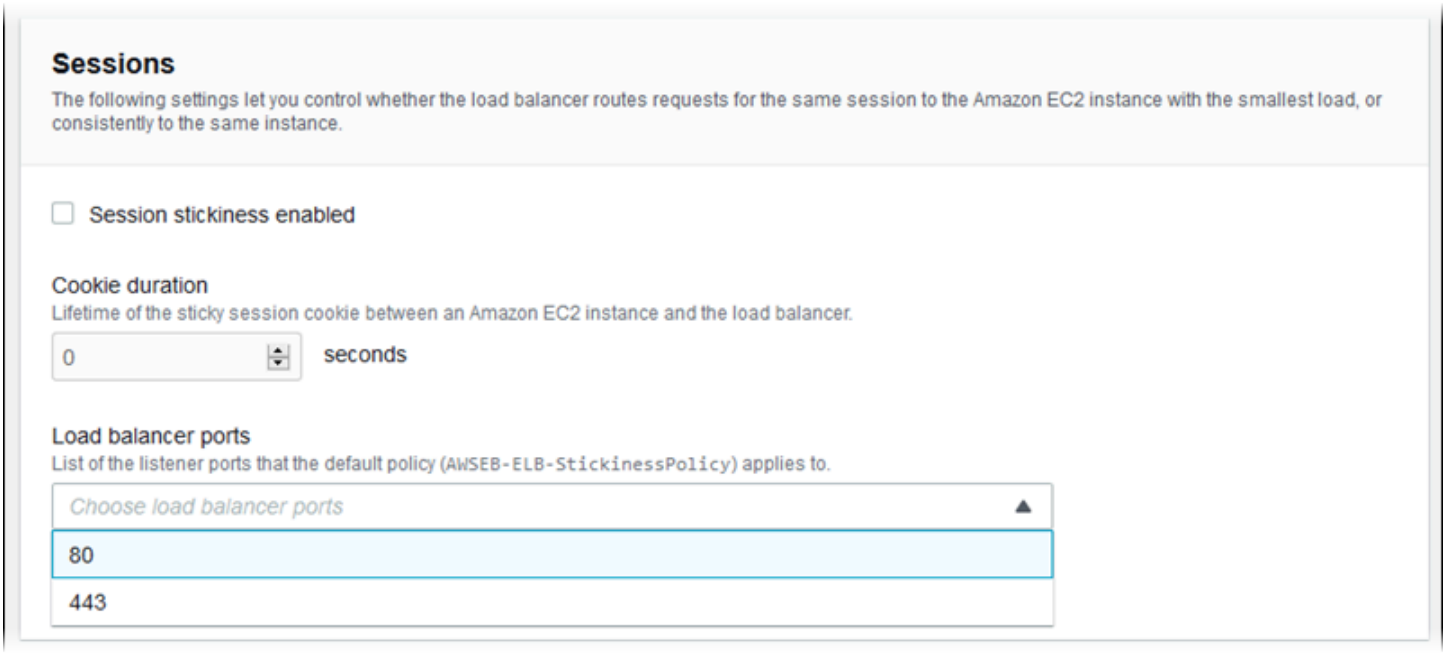
Untuk Sertifikat SSL, pilih ARN sertifikat SSL Anda. Misalnya, `arn:aws:iam::123456789012:server-certificate/abc/certs/build`, atau `arn:aws:acm:us-east-2:123456789012:certificate/12345678-12ab-34cd-56ef-12345678`.



Untuk rincian tentang konfigurasi HTTPS dan bekerja dengan sertifikat di Elastic Beanstalk, lihat Mengonfigurasi HTTPS untuk lingkungan Elastic Beanstalk Anda.

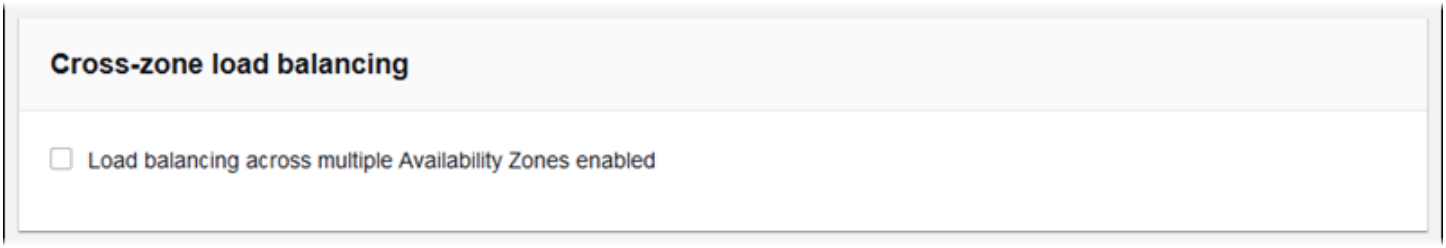
Sesi

Pilih atau kosongkan kotak centang Sesi lekat diaktifkan untuk mengaktifkan atau menonaktifkan sesi lekat. Gunakan durasi cookie untuk mengonfigurasi durasi sesi lekat, hingga **`1000000`** detik. Pada daftar Port penyeimbang beban, pilih pendengar port yang berlaku untuk kebijakan default (`AWSEB-ELB-StickinessPolicy`).



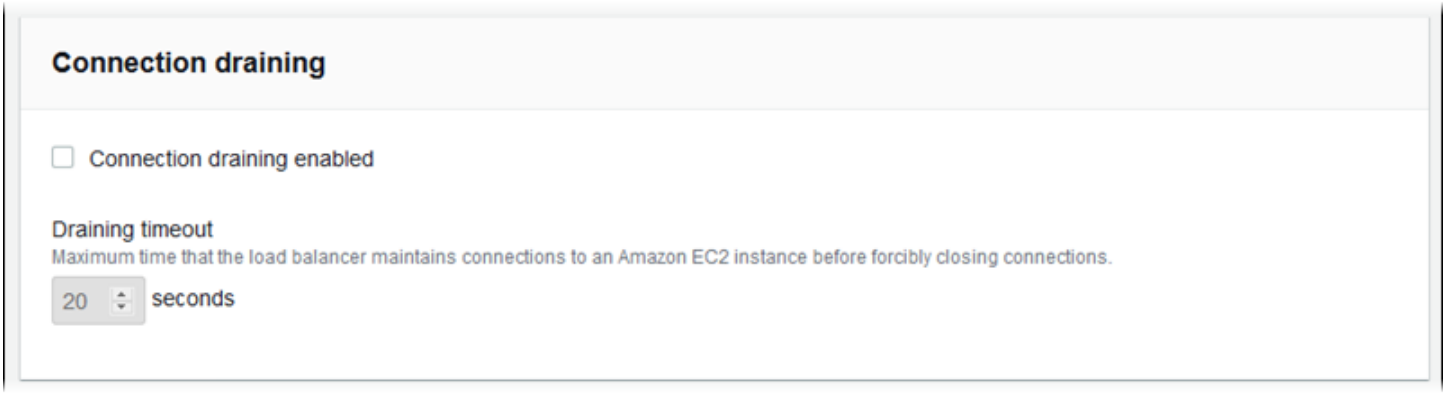
Penyeimbangan beban lintas zona

Pilih atau kosongkan kotak centang Penyeimbangan beban di beberapa Availability Zones yang diaktifkan untuk mengaktifkan atau menonaktifkan penyeimbangan beban lintas zona.



Pengurasan koneksi

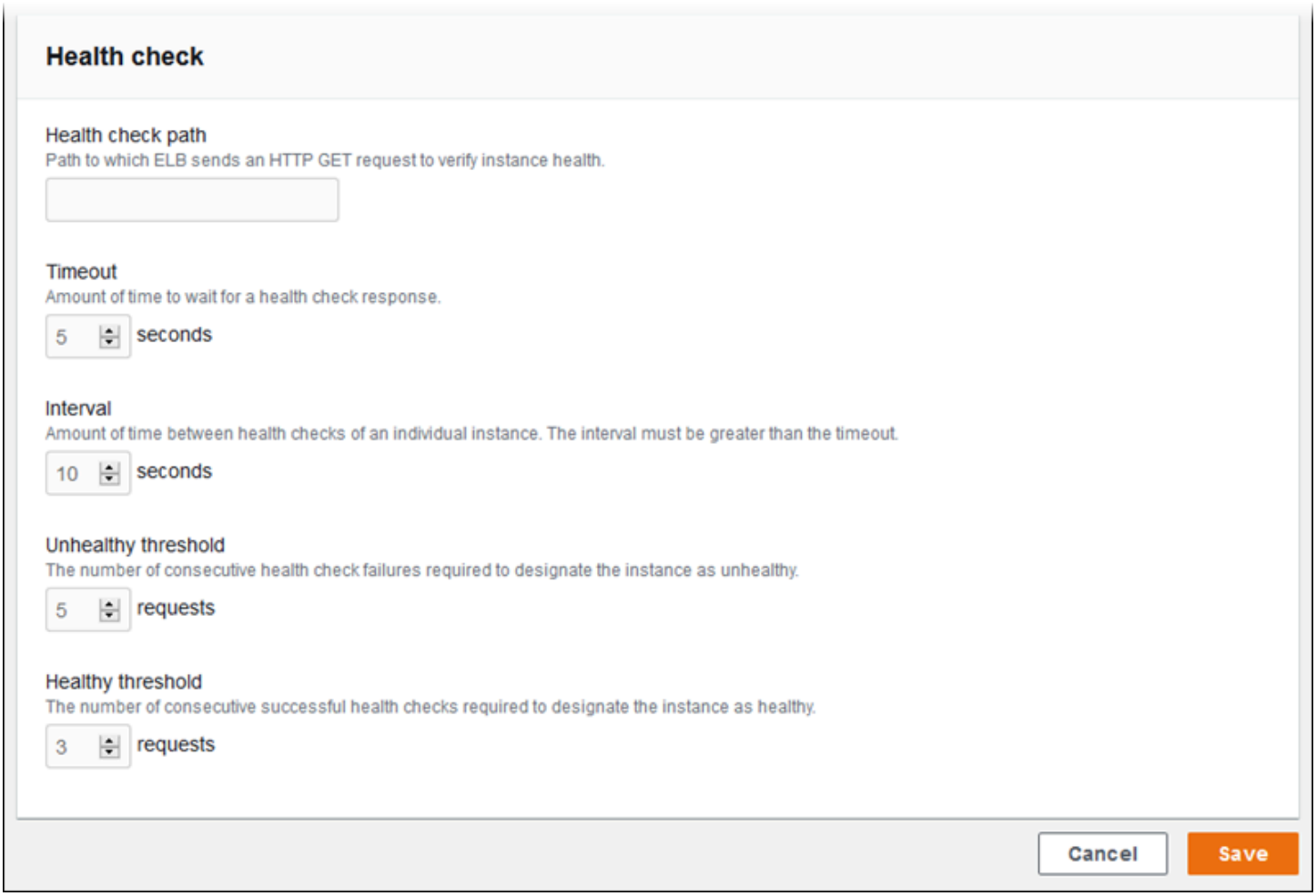
Pilih atau hapus kotak Pengurasan koneksi diaktifkan untuk mengaktifkan atau menonaktifkan pengurasan koneksi. Atur Batas waktu pengurasan, hingga **`3600`** detik.



Pemeriksaan kondisi

Gunakan pengaturan berikut untuk mengonfigurasi pemeriksaan kondisi penyeimbang beban:

- Jalur pemeriksaan kondisi — Jalur tempat penyeimbang beban mengirimkan permintaan pemeriksaan kondisi. Jika Anda tidak menetapkan jalur, penyeimbang beban mencoba untuk membuat koneksi TCP pada port 80 untuk memverifikasi kondisi.
- Batas waktu — Jumlah waktu, dalam detik, untuk menunggu respons pemeriksaan kondisi.
- Interval — Jumlah waktu, dalam detik, antara pemeriksaan kondisi dari instans individu. Interval harus lebih besar dari batas waktu.
- Ambang batas tidak sehat, Ambang batas sehat — Jumlah pemeriksaan kondisi yang harus gagal atau lulus, masing-masing, sebelum Elastic Load Balancing mengubah status kondisi instans.



> **Note**
>
> Pemeriksaan kondisi Elastic Load Balancing tidak memengaruhi perilaku pemeriksaan kondisi grup Auto Scaling lingkungan. Instans yang gagal dalam pemeriksaan kesehatan Elastic Load Balancing tidak secara otomatis digantikan oleh Amazon Auto EC2 Scaling kecuali Anda mengonfigurasi Amazon Auto Scaling secara manual untuk EC2 melakukannya. Lihat Pengaturan pemeriksaan kesehatan Auto Scaling untuk lingkungan Elastic Beanstalk Anda untuk rincian selengkapnya.

Untuk informasi selengkapnya tentang pemeriksaan kondisi dan pengaruhnya terhadap kondisi lingkungan Anda secara keseluruhan, lihat Pelaporan kondisi dasar.

Mengonfigurasi Classic Load Balancer menggunakan EB CLI

EB CLI meminta Anda untuk memilih tipe penyeimbang beban saat Anda menjalankan eb create.

```
$ eb create
Enter Environment Name
(default is my-app): test-env
Enter DNS CNAME prefix
(default is my-app): test-env-DLW24ED23SF

Select a load balancer type
1) classic
2) application
3) network
(default is 1):
```

Tekan Enter untuk memilih `classic`.

Anda juga dapat menentukan tipe penyeimbang beban dengan menggunakan `--elb-type` opsi.

```
$ eb create test-env --elb-type classic
```

Namespace konfigurasi Classic Load Balancer

Anda dapat menemukan pengaturan yang terkait dengan Classic Load Balancer di namespace berikut:

- `aws:elb:healthcheck` — Konfigurasikan ambang batas, periksa interval, dan batas waktu untuk pemeriksaan kondisi penyeimbang beban.
- `aws:elasticbeanstalk:application` — Konfigurasikan URL pemeriksaan kondisi.
- `aws:elb:loadbalancer` — Aktifkan penyeimbangan beban lintas zona. Tetapkan grup keamanan ke penyeimbang beban dan ganti grup keamanan default yang dibuat Elastic Beanstalk. Namespace ini juga termasuk pilihan usang untuk mengonfigurasi pendengar standar dan aman yang telah digantikan oleh opsi pada namespace `aws:elb:listener`.
- `aws:elb:listener` — Konfigurasikan pendengar default pada port 80, pendengar aman pada port 443, atau pendengar tambahan untuk protokol apa pun pada port mana pun. Jika Anda menentukan `aws:elb:listener` sebagai namespace, pengaturan berlaku untuk pendengar default pada port 80. Jika Anda menentukan port (misalnya, `aws:elb:listener:443`), pendengar dikonfigurasi pada port tersebut.
- `aws:elb:policies` — Konfigurasikan pengaturan tambahan untuk penyeimbang beban Anda. Gunakan opsi di namespace ini untuk mengonfigurasi listener pada port arbitrer, memodifikasi pengaturan sesi lengket tambahan, dan mengonfigurasi penyeimbang beban untuk terhubung ke instans Amazon dengan aman. EC2

Konsol EB CLI dan Elastic Beanstalk menerapkan nilai yang direkomendasikan untuk pilihan sebelumnya. Anda harus menghapus pengaturan ini jika Anda ingin menggunakan file konfigurasi untuk mengonfigurasi hal yang sama. Lihat Nilai yang disarankan untuk rincian selengkapnya.

Example .ebextensions/loadbalancer-terminatehttps.config

File konfigurasi contoh berikut membuat pendengar HTTPS pada port 443, menetapkan sertifikat yang digunakan penyeimbang beban untuk mengakhiri koneksi yang aman, dan menonaktifkan pendengar default pada port 80. Penyeimbang beban meneruskan permintaan yang didekripsi ke EC2 instance di lingkungan Anda di HTTP:80.

```
option_settings:
  aws:elb:listener:443:
    ListenerProtocol: HTTPS
    SSLCertificateId: arn:aws:acm:us-
east-2:123456789012:certificate/12345678-12ab-34cd-56ef-12345678
    InstancePort: 80
    InstanceProtocol: HTTP
  aws:elb:listener:
    ListenerEnabled: false
```

Mengonfigurasi Application Load Balancer

Saat Anda mengaktifkan load balancing, AWS Elastic Beanstalk lingkungan Anda dilengkapi dengan penyeimbang beban Elastic Load Balancing untuk mendistribusikan lalu lintas antar instans di lingkungan Anda. Elastic Load Balancing mendukung beberapa tipe penyeimbang beban. Untuk mempelajarinya, lihat Panduan Pengguna Elastic Load Balancing. Elastic Beanstalk dapat membuat penyeimbang beban untuk Anda, atau memungkinkan Anda menentukan penyeimbang beban bersama yang telah Anda buat.

Topik ini menjelaskan konfigurasi Application Load Balancer yang dibuat Elastic Beanstalk dan didedikasikan untuk lingkungan Anda. Lihat juga the section called “Application Load Balancer Bersama”. Untuk informasi tentang cara mengonfigurasi semua tipe penyeimbang beban yang didukung Elastic Beanstalk, lihat the section called “Penyeimbang beban”.

> **Note**
>
> Anda dapat memilih tipe penyeimbang beban yang hanya digunakan lingkungan Anda selama pembuatan lingkungan. Anda dapat mengubah pengaturan untuk mengelola perilaku penyeimbang beban lingkungan berjalan Anda, tetapi Anda tidak dapat mengubah tipenya. Anda juga tidak dapat beralih dari penyeimbang beban bersama khusus ke penyeimbang beban bersama atau sebaliknya.

Pengantar

Application Load Balancer memeriksa lalu lintas di lapisan protokol jaringan aplikasi untuk mengidentifikasi jalur permintaan sehingga lalu lintas tersebut dapat mengarahkan permintaan ke jalur yang berbeda ke tujuan yang berbeda.

Bila lingkungan Anda menggunakan Application Load Balancer, Elastic Beanstalk mengonfigurasinya secara default untuk melakukan fungsi yang sama dengan Classic Load Balancer. Pendengar default menerima permintaan HTTP pada port 80 dan mendistribusikannya ke instans di lingkungan Anda. Anda dapat menambahkan pendengar yang aman pada port 443 dengan sertifikat untuk mendekripsi lalu lintas HTTPS, mengonfigurasi perilaku pemeriksaan kondisi, dan mendorong log akses dari penyeimbang beban ke bucket Amazon Simple Storage Service (Amazon S3).

> **Note**
>
> Tidak seperti Classic Load Balancer atau Network Load Balancer, Application Load Balancer tidak dapat memiliki lapisan transport (layer 4) TCP atau pendengar. SSL/TLS Application Load Balancer hanya mendukung pendengar HTTP dan HTTPS. Selain itu, ia tidak dapat menggunakan autentikasi backend untuk mengotentikasi koneksi HTTPS antara penyeimbang beban dan instans backend.

Dalam lingkungan Elastic Beanstalk, Anda dapat menggunakan Application Load Balancer untuk mengarahkan lalu lintas untuk jalur tertentu ke proses yang berbeda pada instas server web Anda. Dengan Classic Load Balancer, semua lalu lintas ke pendengar diarahkan ke satu proses pada instans backend. Dengan Application Load Balancer, Anda dapat mengonfigurasi beberapa aturan pada pendengar untuk merutekan permintaan ke jalur tertentu ke proses backend yang berbeda. Anda mengonfigurasi setiap proses dengan port yang mendengarkan proses.

Misalnya, Anda dapat menjalankan proses login secara terpisah dari aplikasi utama Anda. Sementara aplikasi utama pada instans lingkungan Anda menerima sebagian besar permintaan dan mendengarkan pada port 80, proses login Anda berkonsentrasi pada port 5000 dan menerima permintaan ke jalur `/login`. Semua permintaan masuk dari klien masuk pada port 80. Dengan Application Load Balancer, Anda dapat mengonfigurasi satu pendengar untuk lalu lintas masuk di port 80, dengan dua aturan yang merutekan lalu lintas ke dua proses terpisah, tergantung pada jalur dalam permintaan. Anda menambahkan aturan khusus yang merutekan lalu lintas `/login` untuk proses login yang berkonsentrasi pada port 5000. Aturan default merutekan semua lalu lintas lain untuk proses aplikasi utama yang berkonsentrasi pada port 80.

Aturan Application Load Balancer memetakan permintaan ke grup target. Dalam Elastic Beanstalk, grup target diwakili oleh proses. Anda dapat mengonfigurasi proses dengan protokol, port, dan pengaturan pemeriksaan kondisi. Proses ini merupakan proses yang berjalan pada instans di lingkungan Anda. Proses default adalah pendengar pada port 80 dari proksi terbalik (nginx atau Apache) yang berjalan di depan aplikasi Anda.

> **Note**
>
> Di luar Elastic Beanstalk, grup target memetakan ke sekelompok instans. Seorang pendengar dapat menggunakan aturan dan grup target untuk merutekan lalu lintas ke instans yang

berbeda berdasarkan jalur. Dalam Elastic Beanstalk, semua instans di lingkungan Anda identik, sehingga perbedaan dibuat antara proses mendengarkan pada port yang berbeda.

Classic Load Balancer menggunakan satu jalur pemeriksaan kondisi untuk seluruh lingkungan. Dengan Application Load Balancer, setiap proses memiliki jalur pemeriksaan kondisi terpisah yang dipantau oleh penyeimbang beban dan pemantauan kesehatan yang disempurnakan dengan Elastic Beanstalk.

Untuk menggunakan Application Load Balancer, lingkungan Anda harus berada di VPC default atau khusus, dan harus memiliki peran layanan dengan seperangkat izin standar. Jika Anda memiliki peran layanan lama, Anda mungkin harus memperbarui izin pada peran tersebut untuk memasukkan `elasticloadbalancing:DescribeTargetHealth` dan `elasticloadbalancing:DescribeLoadBalancers`. Untuk informasi selengkapnya tentang Application Load Balancers, lihat Apa itu Application Load Balancer?.

> **Note**
>
> Pemeriksaan kondisi Application Load Balancer tidak menggunakan jalur pemeriksaan kondisi Elastic Beanstalk. Sebaliknya, pemeriksaan tersebut menggunakan jalur tertentu yang dikonfigurasi untuk setiap proses secara terpisah.

Mengonfigurasi Application Load Balancer menggunakan konsol Elastic Beanstalk

Anda dapat menggunakan konsol Elastic Beanstalk untuk mengonfigurasi pendengar, proses, dan aturan Application Load Balancer, selama pembuatan lingkungan atau nanti saat lingkungan Anda berjalan.

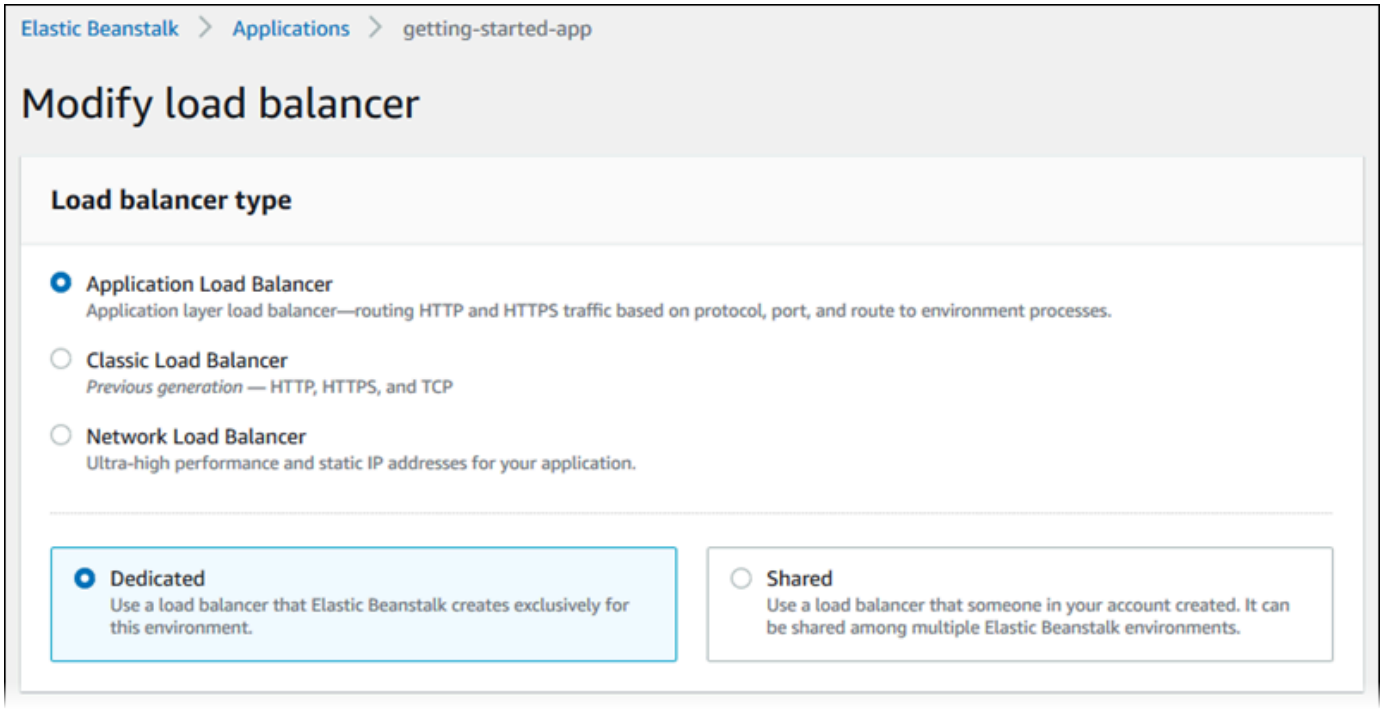
Untuk mengonfigurasi Application Load Balancer di konsol Elastic Beanstalk selama pembuatan lingkungan

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Pada panel navigasi, pilih Lingkungan.
3. Pilih Buat lingkungan baru untuk mulai membuat lingkungan Anda.
4. Pada halaman utama wizard, sebelum memilih Buat lingkungan, pilih Konfigurasikan opsi lainnya.

5. Pilih konfigurasi Ketersediaan yang tinggi yang telah ditetapkan.

   Atau, pada kategori konfigurasi Kapasitas, konfigurasikan tipe lingkungan dengan beban seimbang. Untuk rincian selengkapnya, lihat Kapasitas.

6. Pada kategori konfigurasi Penyeimbang beban, pilih Edit.
7. Pilih opsi Application Load Balancer dan Khusus, jika belum dipilih.



8. Buat perubahan konfigurasi Application Load Balancer yang dibutuhkan lingkungan Anda.
9. Pilih Simpan, dan kemudian buat perubahan konfigurasi lain yang diperlukan lingkungan Anda.
10. Pilih Buat lingkungan.

Untuk mengonfigurasi Application Load Balancer lingkungan berjalan di konsol Elastic Beanstalk

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Pada kategori konfigurasi Penyeimbang beban, pilih Edit.

> **Note**
>
> Jika kategori konfigurasi Penyeimbang beban tidak memiliki tombol Edit, lingkungan Anda tidak memiliki penyeimbang beban. Untuk mempelajari cara menyiapkan penyeimbang beban, lihat Mengubah jenis lingkungan.

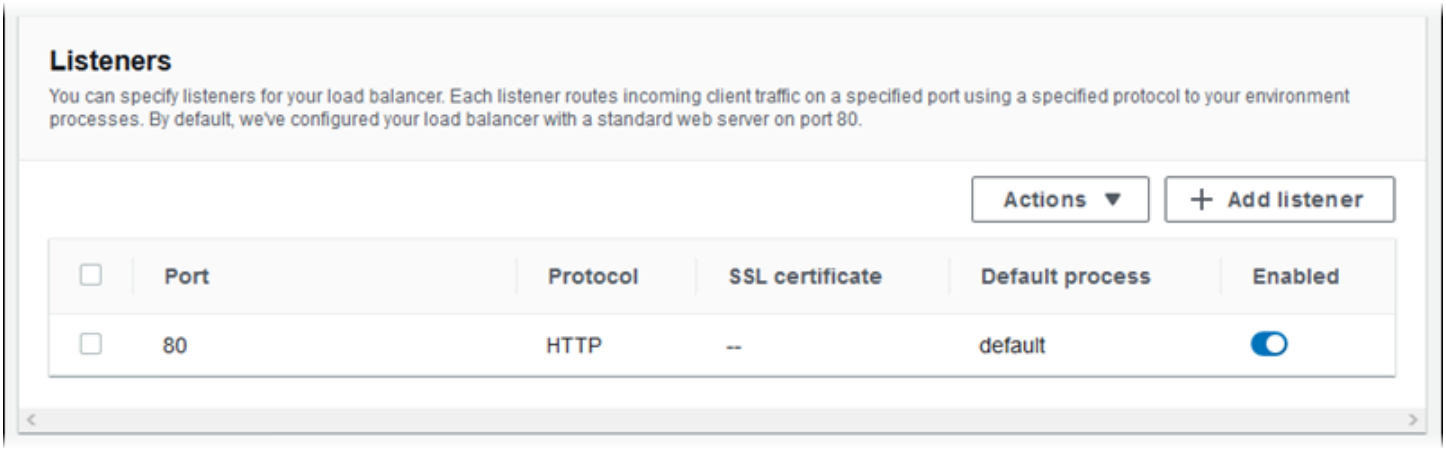
5. Buat perubahan konfigurasi Application Load Balancer yang diperlukan lingkungan Anda.
6. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Pengaturan Application Load Balancer

- Listener
- Proses
- Aturan
- Penangkapan log akses

Listener

Gunakan daftar ini untuk menentukan pendengar untuk penyeimbang beban Anda. Setiap pendengar merutekan lalu lintas klien masuk pada port tertentu menggunakan protokol tertentu untuk satu atau lebih proses pada instans Anda. Awalnya, daftar berikut menunjukkan pendengar default, yang merutekan lalu lintas HTTP masuk pada port 80 untuk proses bernama default.



Untuk mengonfigurasi pendengar yang ada

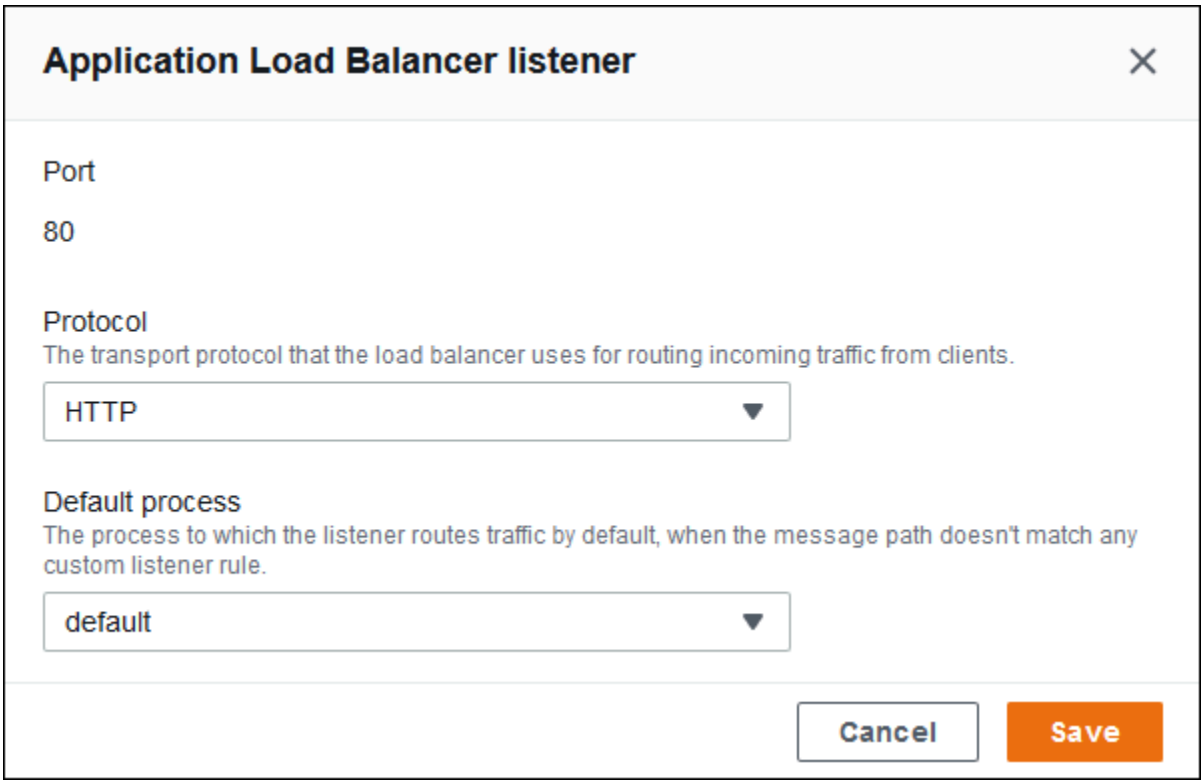
1. Pilih kotak centang di samping entri tabel, dan kemudian pilih Tindakan, Edit.

2. Gunakan kotak dialog pendengar Application Load Balancer untuk mengedit pengaturan, dan kemudian pilih Simpan.

Untuk menambahkan pendengar

1. Pilih Tambahkan pendengar.
2. Di kotak dialog Pendengar Application Load Balancer, konfigurasikan pengaturan yang Anda inginkan, dan kemudian pilih Tambahkan.

Gunakan pengaturan kotak dialog pendengar Application Load Balancer untuk memilih port dan protokol tempat pendengar mendengarkan lalu lintas, dan proses untuk merutekan lalu lintas. Jika Anda memilih protokol HTTPS, konfigurasikan pengaturan SSL.



Sebelum Anda dapat mengonfigurasi pendengar HTTPS, pastikan bahwa Anda memiliki sertifikat SSL yang valid. Lakukan salah satu tindakan berikut:

- Jika AWS Certificate Manager (ACM) tersedia di AWS Wilayah Anda, buat atau impor sertifikat menggunakan ACM. Untuk informasi selengkapnya tentang permintaan sertifikat ACM, lihat Meminta Sertifikat di AWS Certificate Manager Panduan Pengguna. Untuk informasi tentang

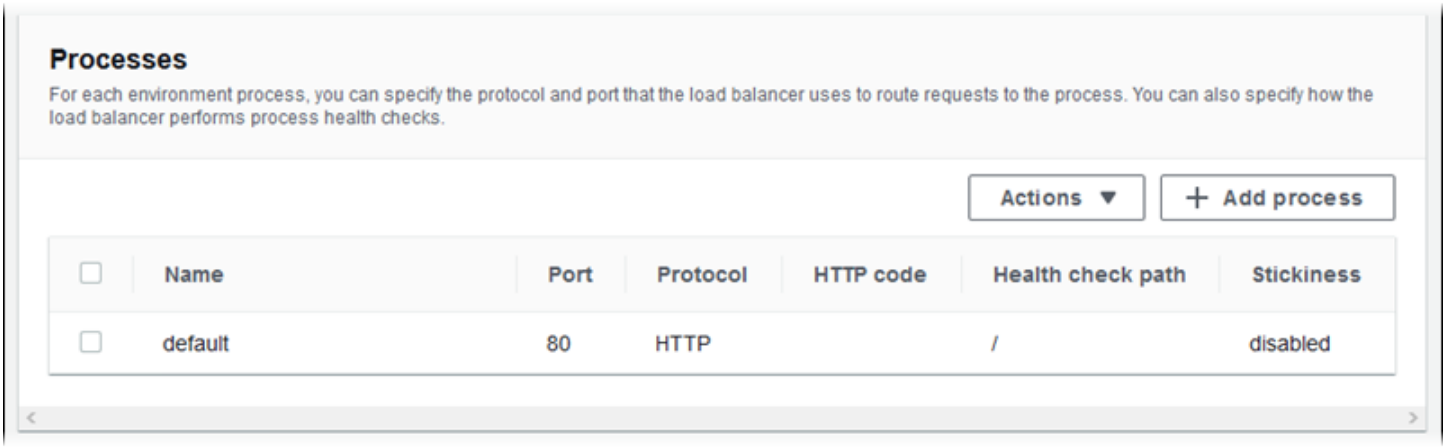
mengimpor sertifikat pihak ke tiga ke ACM, lihat Mengimpor Sertifikat di Panduan Pengguna AWS Certificate Manager .

- Jika ACM tidak tersedia di AWS Wilayah Anda, unggah sertifikat dan kunci yang ada ke IAM. Untuk informasi selengkapnya tentang pembuatan dan pengunggahan sertifikat ke IAM, lihat Bekerja dengan Sertifikat Server di Panduan Pengguna IAM.

Untuk detail lebih lanjut tentang mengonfigurasi HTTPS dan bekerja dengan sertifikat di Elastic Beanstalk, lihat Mengonfigurasi HTTPS untuk lingkungan Elastic Beanstalk Anda.

Proses

Gunakan daftar ini untuk menentukan proses penyeimbang beban Anda. Sebuah proses adalah target bagi pendengar untuk merutekan lalu lintas. Setiap pendengar merutekan lalu lintas klien masuk pada port tertentu menggunakan protokol tertentu untuk satu atau lebih proses pada instans Anda. Awalnya, daftar menunjukkan proses default, yang mendengarkan lalu lintas HTTP masuk pada port 80.



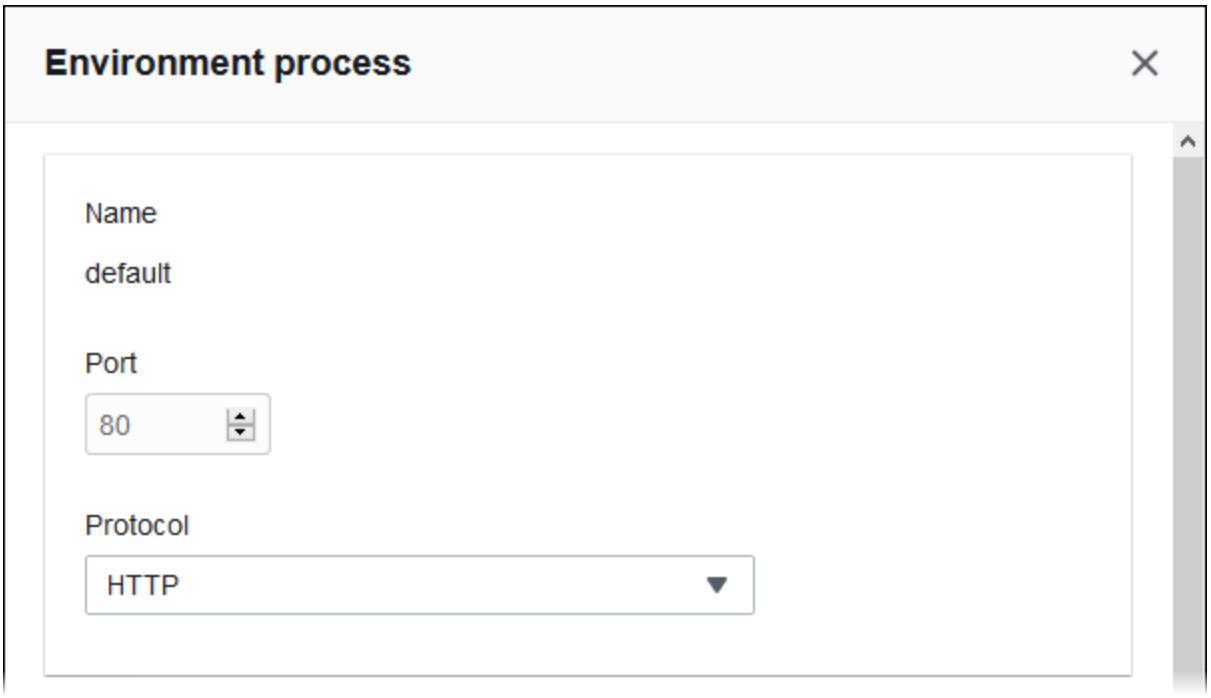
Anda dapat mengedit pengaturan proses yang ada, atau menambahkan proses baru. Untuk mulai mengedit proses pada daftar atau menambahkan proses untuk itu, gunakan langkah yang sama yang tercantum untuk daftar pendengar. Kotak dialog Proses lingkungan terbuka.

Pengaturan kotak dialog proses lingkungan Application Load Balancer

- Definisi
- Pemeriksaan kondisi
- Sesi

Definisi

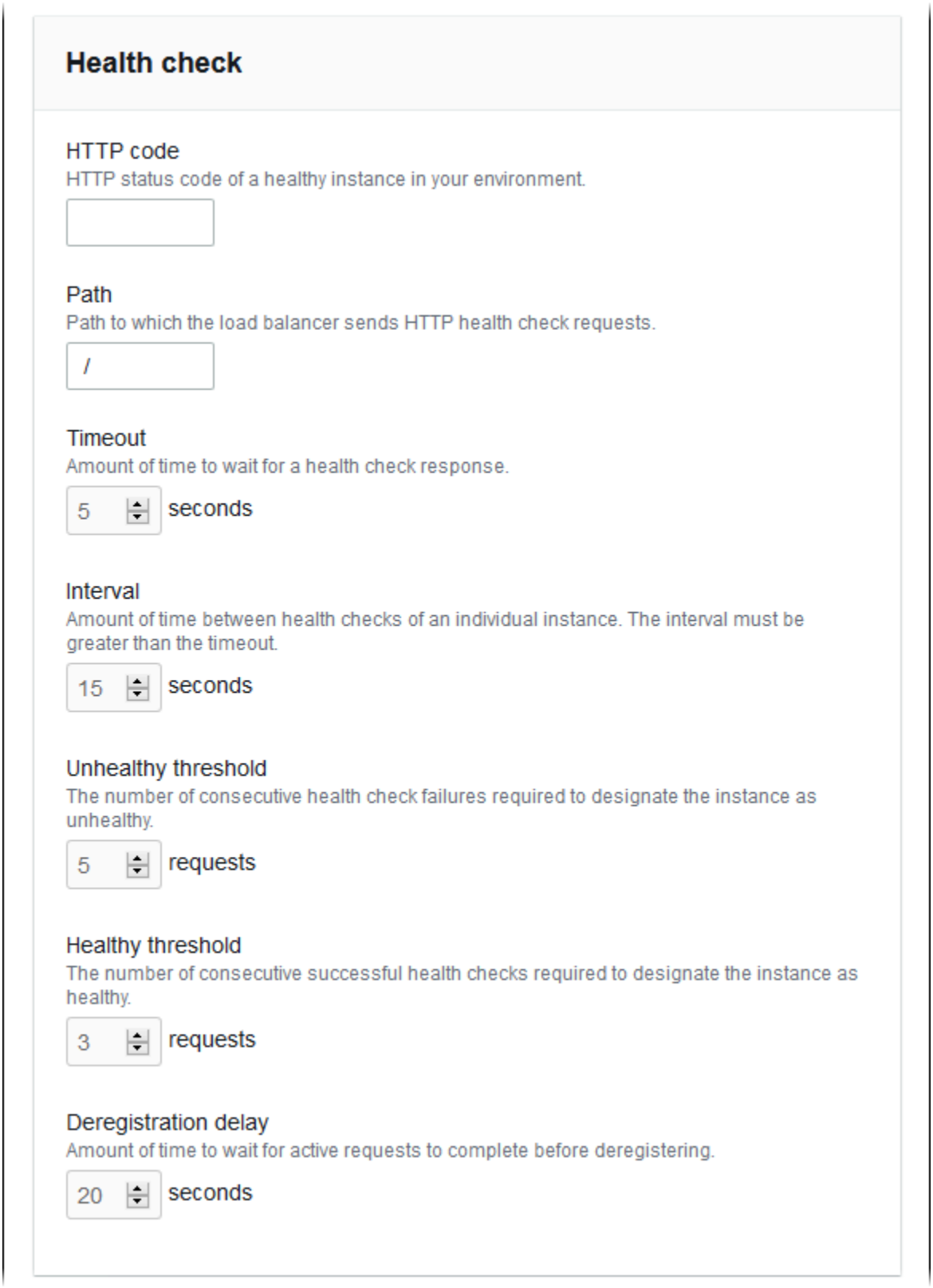
Gunakan pengaturan ini untuk menentukan proses: Nama, dan Port dan Protokol yang mendengarkan permintaan.



Pemeriksaan kondisi

Gunakan pengaturan berikut untuk mengonfigurasi pemeriksaan kondisi proses:

- Kode HTTP — Kode status HTTP menunjuk proses yang sehat.
- Jalur — Jalur permintaan pemeriksaan kondisi untuk proses tersebut.
- Batas waktu — Jumlah waktu, dalam detik, untuk menunggu respons pemeriksaan kondisi.
- Interval — Jumlah waktu, dalam detik, antara pemeriksaan kondisi dari instans individu. Interval harus lebih besar dari batas waktu.
- Ambang batas tidak sehat, Ambang batas sehat — Jumlah pemeriksaan kondisi yang harus gagal atau lulus, masing-masing, sebelum Elastic Load Balancing mengubah status kondisi instans.
- Penundaan pendaftaran — Jumlah waktu, dalam detik, untuk menunggu permintaan aktif selesai sebelum membatalkan pendaftaran.

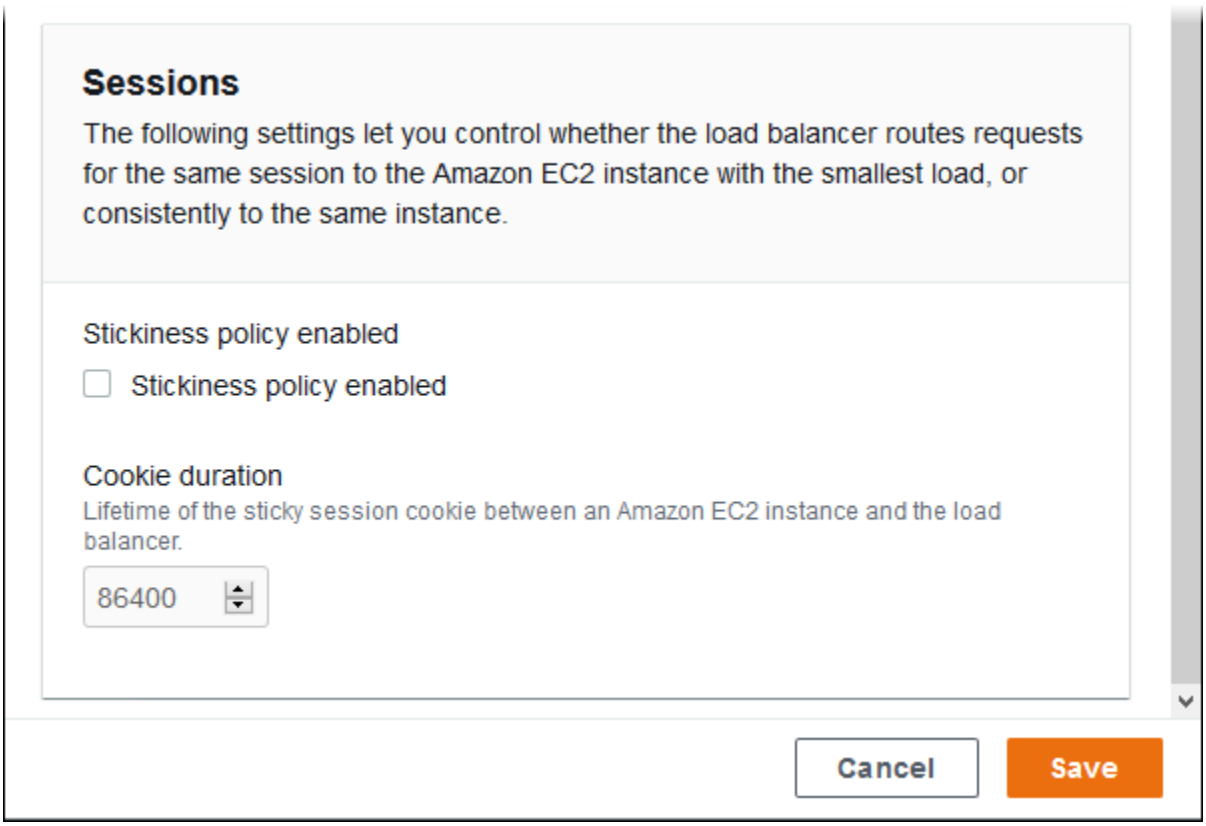

Health check
HTTP code
HTTP status code of a healthy instance in your environment.
Path
Path to which the load balancer sends HTTP health check requests.
/
Timeout
Amount of time to wait for a health check response.
5
seconds
Interval
Amount of time between health checks of an individual instance. The interval must be greater than the timeout.
15
seconds
Unhealthy threshold
The number of consecutive health check failures required to designate the instance as unhealthy.
5
requests
Healthy threshold
The number of consecutive successful health checks required to designate the instance as healthy.
3
requests
Deregistration delay
Amount of time to wait for active requests to complete before deregistering.
20
seconds

> **Note**
>
> Pemeriksaan kondisi Elastic Load Balancing tidak memengaruhi perilaku pemeriksaan kondisi grup Auto Scaling lingkungan. Instans yang gagal dalam pemeriksaan kesehatan Elastic Load Balancing tidak secara otomatis digantikan oleh Amazon Auto EC2 Scaling kecuali Anda mengonfigurasi Amazon Auto Scaling secara manual untuk EC2 melakukannya. Lihat Pengaturan pemeriksaan kesehatan Auto Scaling untuk lingkungan Elastic Beanstalk Anda untuk rincian selengkapnya.

Untuk informasi selengkapnya tentang pemeriksaan kondisi dan pengaruhnya terhadap kondisi lingkungan Anda secara keseluruhan, lihat Pelaporan kondisi dasar.

Sesi

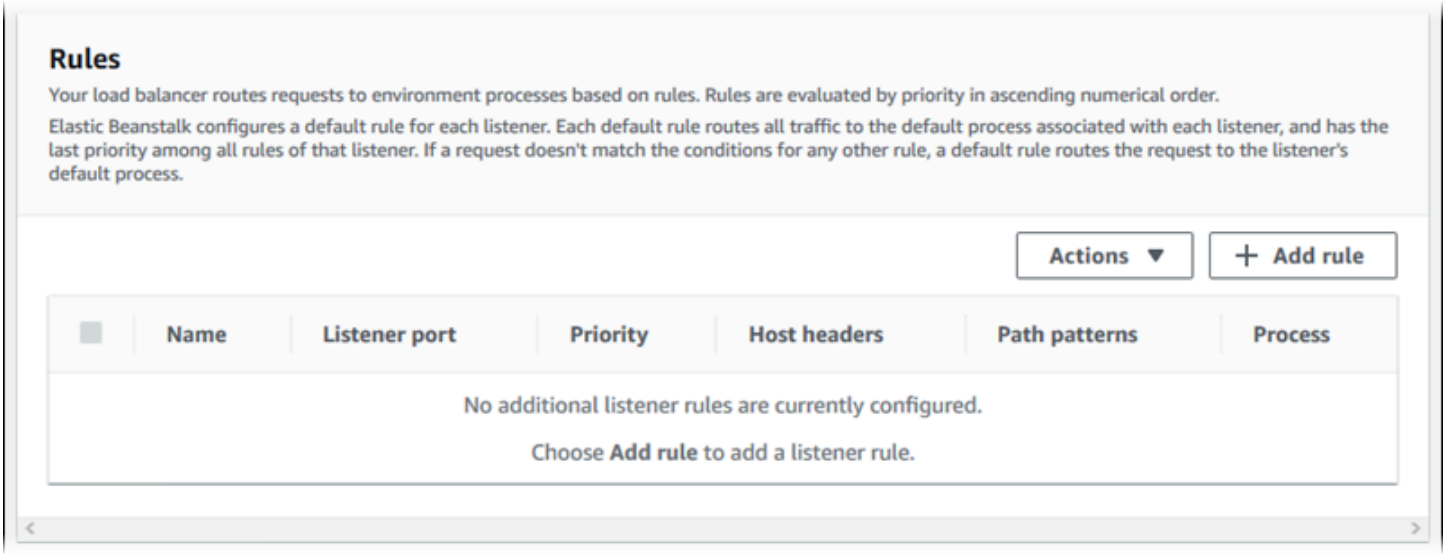
Pilih atau kosongkan kotak Kebijakan kelekatan diaktifkan untuk mengaktifkan atau menonaktifkan sesi lekat. Gunakan durasi cookie untuk mengonfigurasi durasi sesi lekat, hingga **`604800`** detik.



Aturan

Gunakan daftar ini untuk menentukan aturan pendengar khusus untuk penyeimbang beban Anda. Sebuah aturan memetakan permintaan yang diterima pendengar pada pola jalur tertentu untuk proses target. Setiap pendengar dapat memiliki beberapa aturan, merutekan permintaan pada jalur yang berbeda untuk proses yang berbeda pada instans Anda.

Aturan memiliki prioritas numerik yang menentukan prioritas penerapannya pada permintaan yang masuk. Untuk setiap pendengar baru yang Anda menambahkan, Elastic Beanstalk menambahkan aturan default yang merutekan semua lalu lintas pendengar ke proses default. Prioritas aturan default adalah yang terendah; itu diterapkan jika tidak ada aturan lain untuk pendengar yang sama cocok dengan permintaan masuk. Awalnya, jika Anda belum menambahkan aturan khusus, daftar tersebut kosong. Aturan default semua pendengar tidak ditampilkan.



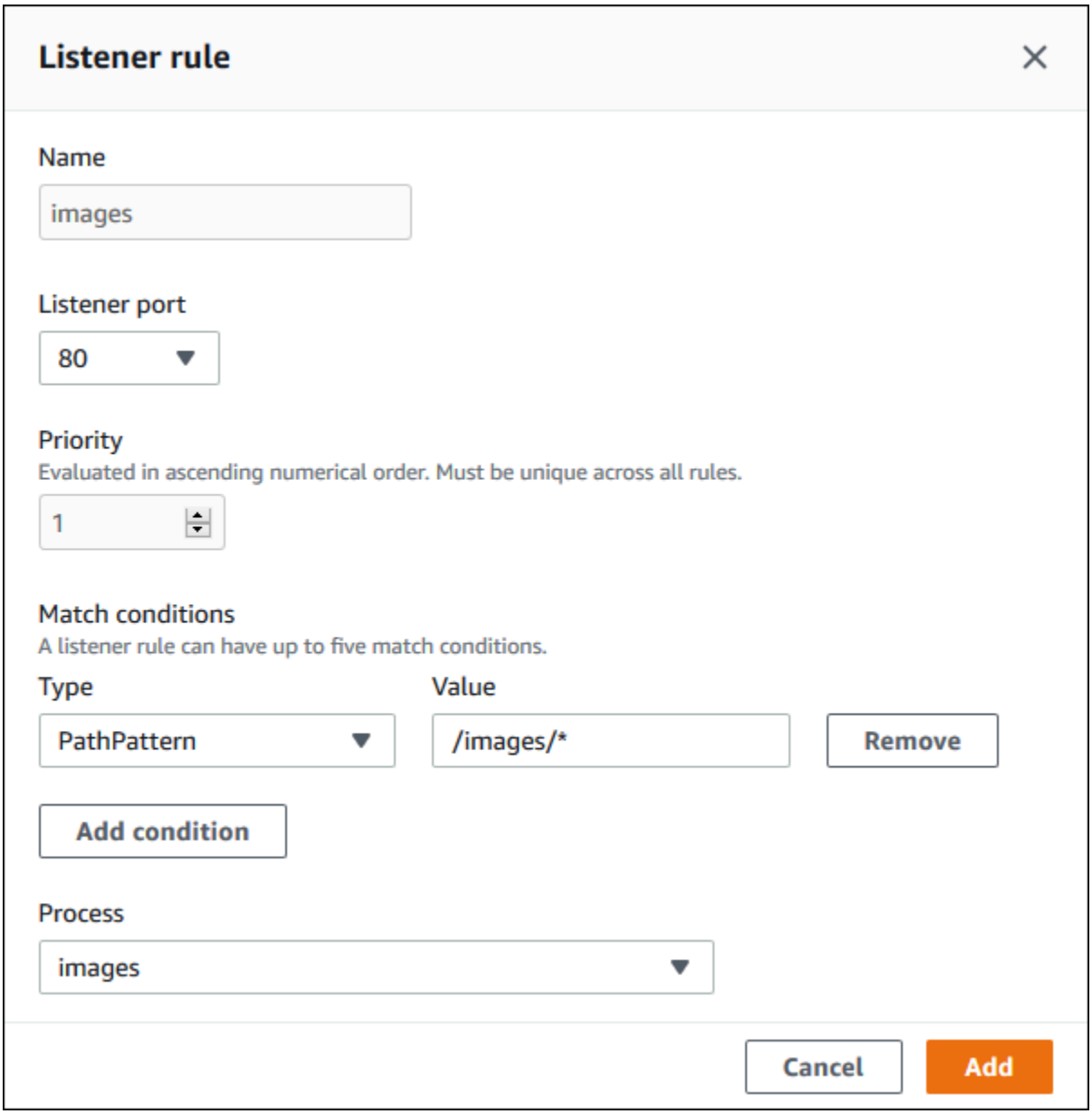
Anda dapat mengedit pengaturan aturan yang ada, atau menambahkan aturan baru. Untuk mulai mengedit aturan pada daftar atau menambahkan aturan padanya, gunakan langkah yang sama yang tercantum untuk daftar pendengar. Kotak dialog Aturan pendengar terbuka, dengan pengaturan berikut:

- Nama - Nama aturan.
- port Pendengar — port Pendengar yang aturannya berlaku.
- Prioritas - Prioritas aturan. Sebuah nomor prioritas yang lebih rendah memiliki prioritas yang lebih tinggi. Prioritas aturan pendengar harus unik.
- Kondisi yang cocok — Daftar kondisi URL permintaan tempat aturan berlaku. Ada dua jenis kondisi: HostHeader(bagian domain URL), dan PathPattern(bagian jalur URL). Anda dapat

menambahkan hingga lima kondisi. Setiap nilai kondisi mencapai 128 karakter, dan dapat mencakup karakter wildcard.

- Proses — Proses dimana penyeimbang beban merutekan permintaan yang sesuai dengan aturan.

Saat mengedit aturan yang sudah ada, Anda tidak dapat mengubah Nama dan port Pendengar.



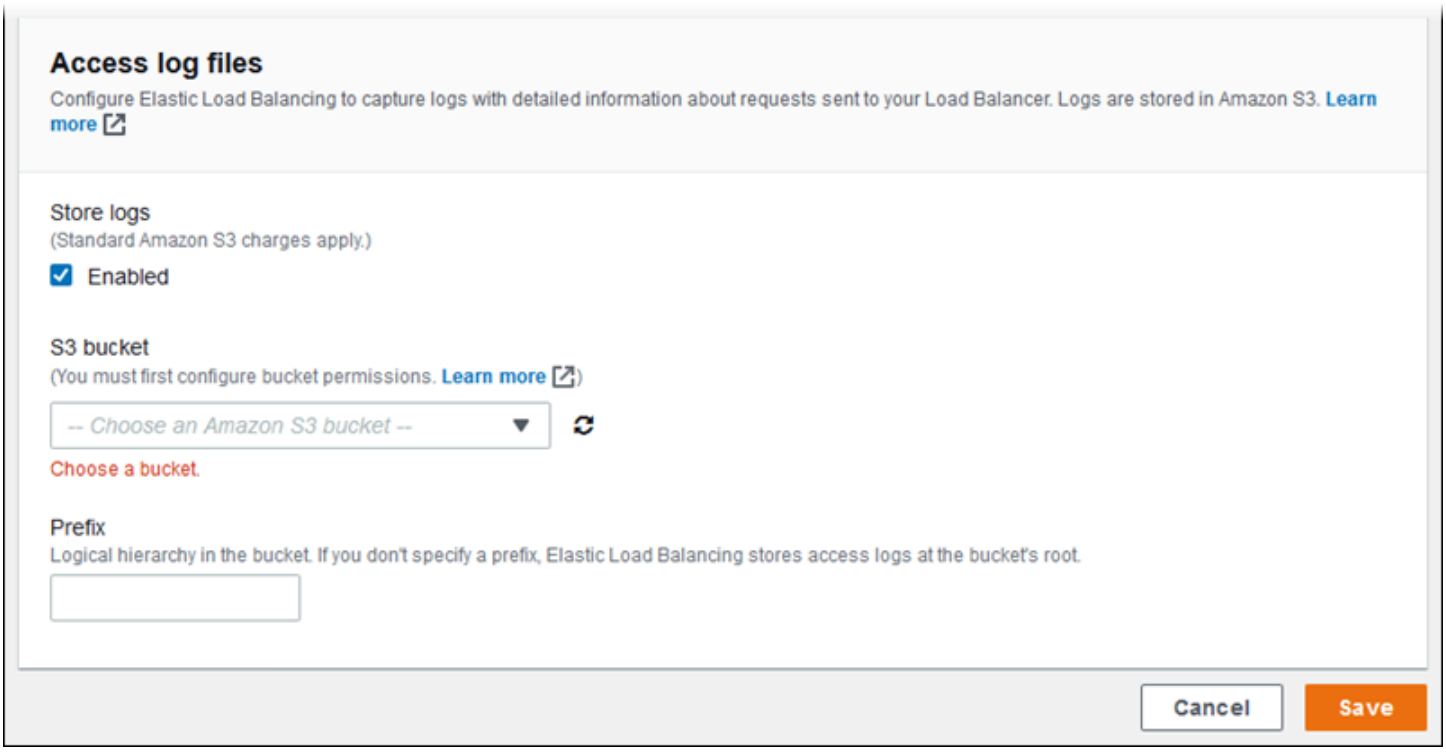
Penangkapan log akses

Gunakan pengaturan ini untuk mengonfigurasi Elastic Load Balancing untuk menangkap log dengan informasi terperinci tentang permintaan yang dikirim ke Application Load Balancer Anda. Akses tangkapan log dinonaktifkan secara default. Saat Log penyimpanan diaktifkan, Elastic Load Balancing

menyimpan log di bucket S3 yang Anda konfigurasikan. Pengaturan Prefiks menetapkan folder tingkat atas dalam bucket untuk log. Elastic Load Balancing menempatkan log dalam folder bernama `AWSLogs` di bawah prefiks Anda. Jika Anda tidak menentukan awalan, Elastic Load Balancing menempatkan foldernya di tingkat akar bucket.

> **Note**
>
> Jika bucket Amazon S3 yang Anda konfigurasikan untuk pengambilan log akses bukanlah bucket yang dibuat Elastic Beanstalk untuk akun Anda, pastikan untuk menambahkan kebijakan pengguna dengan izin yang sesuai untuk pengguna (IAM) Anda. AWS Identity and Access Management Kebijakan pengguna terkelola yang disediakan Elastic Beanstalk hanya mencakup izin ke sumber daya terkelola Elastic Beanstalk.

Untuk rincian tentang log akses, termasuk izin dan persyaratan lainnya, lihat Log akses untuk Application Load Balancer.



Contoh: Application Load Balancer dengan pendengar yang aman dan dua proses

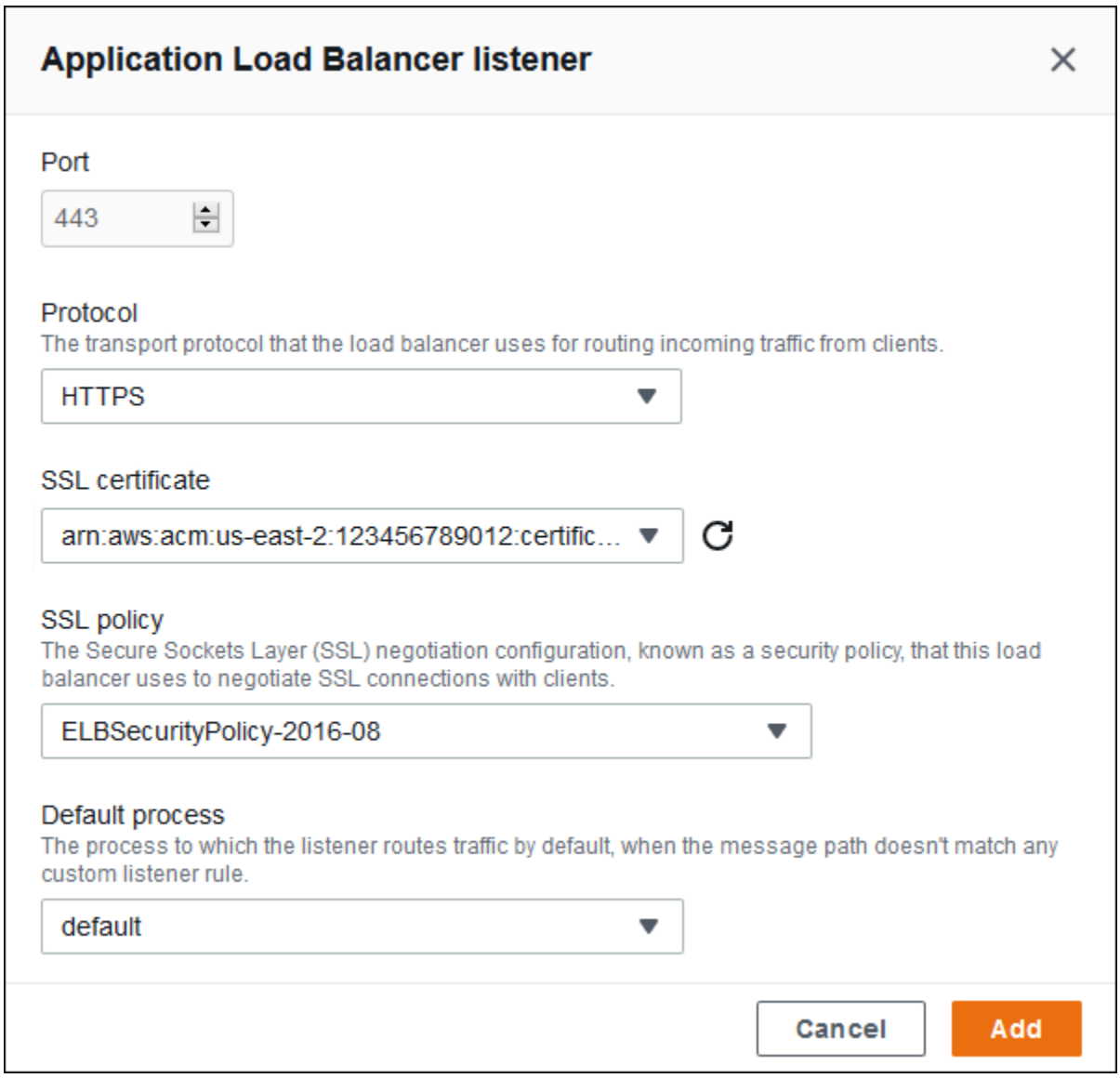
Dalam contoh ini, aplikasi Anda memerlukan enkripsi end-to-end lalu lintas dan proses terpisah untuk menangani permintaan administratif.

Untuk mengonfigurasi lingkungan Application Load Balancer Anda memenuhi persyaratan ini, Anda menghapus pendengar default, menambahkan pendengar HTTPS, menunjukkan bahwa proses default mendengarkan port 443 pada HTTPS, dan menambahkan proses dan aturan pendengar untuk lalu lintas admin di jalan yang berbeda.

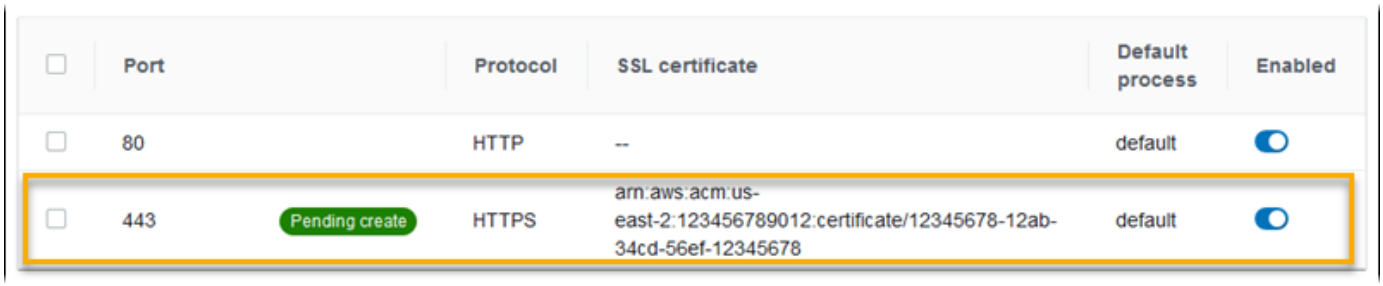
Untuk mengonfigurasi penyeimbang beban pada contoh ini

1. Tambahkan pendengar yang aman. Untuk Port, ketik **`443`**. Untuk Protokol, pilih **`HTTPS`**. Untuk Sertifikat SSL, pilih ARN sertifikat SSL Anda. Misalnya, **`arn:aws:iam::123456789012:server-certificate/abc/certs/build`**, atau **`arn:aws:acm:us-east-2:123456789012:certificate/12345678-12ab-34cd-56ef-12345678`**.

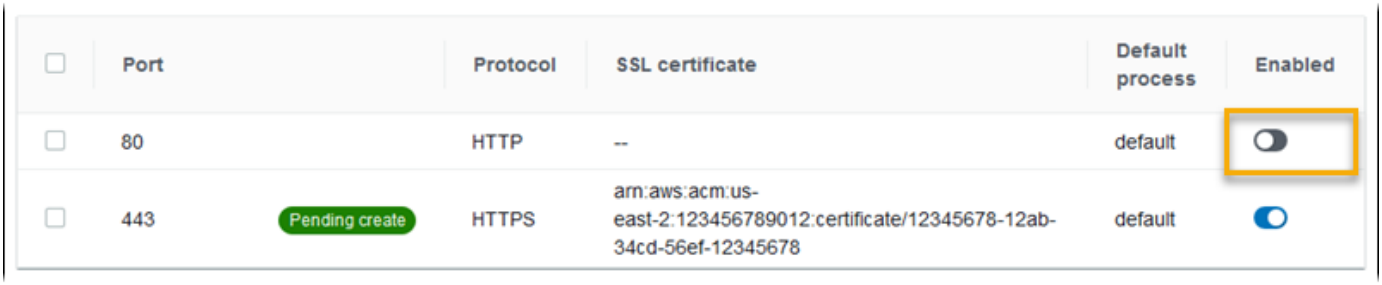
   Untuk Proses default, **`default`** tetap dipilih.



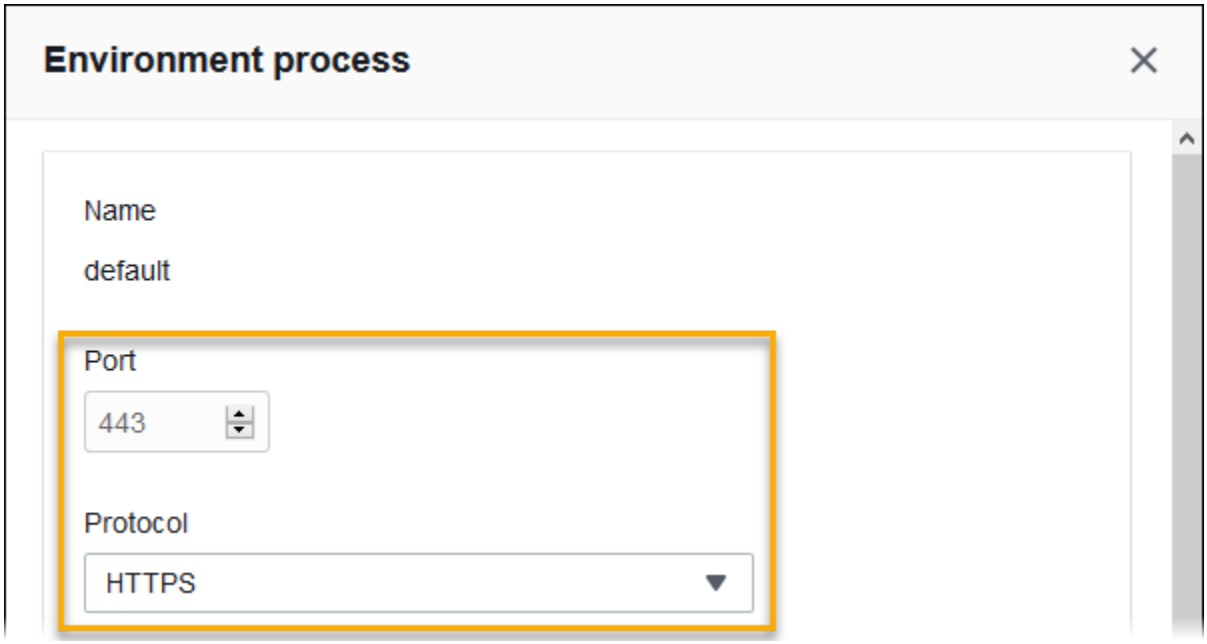
Anda sekarang dapat melihat pendengar tambahan Anda pada daftar.



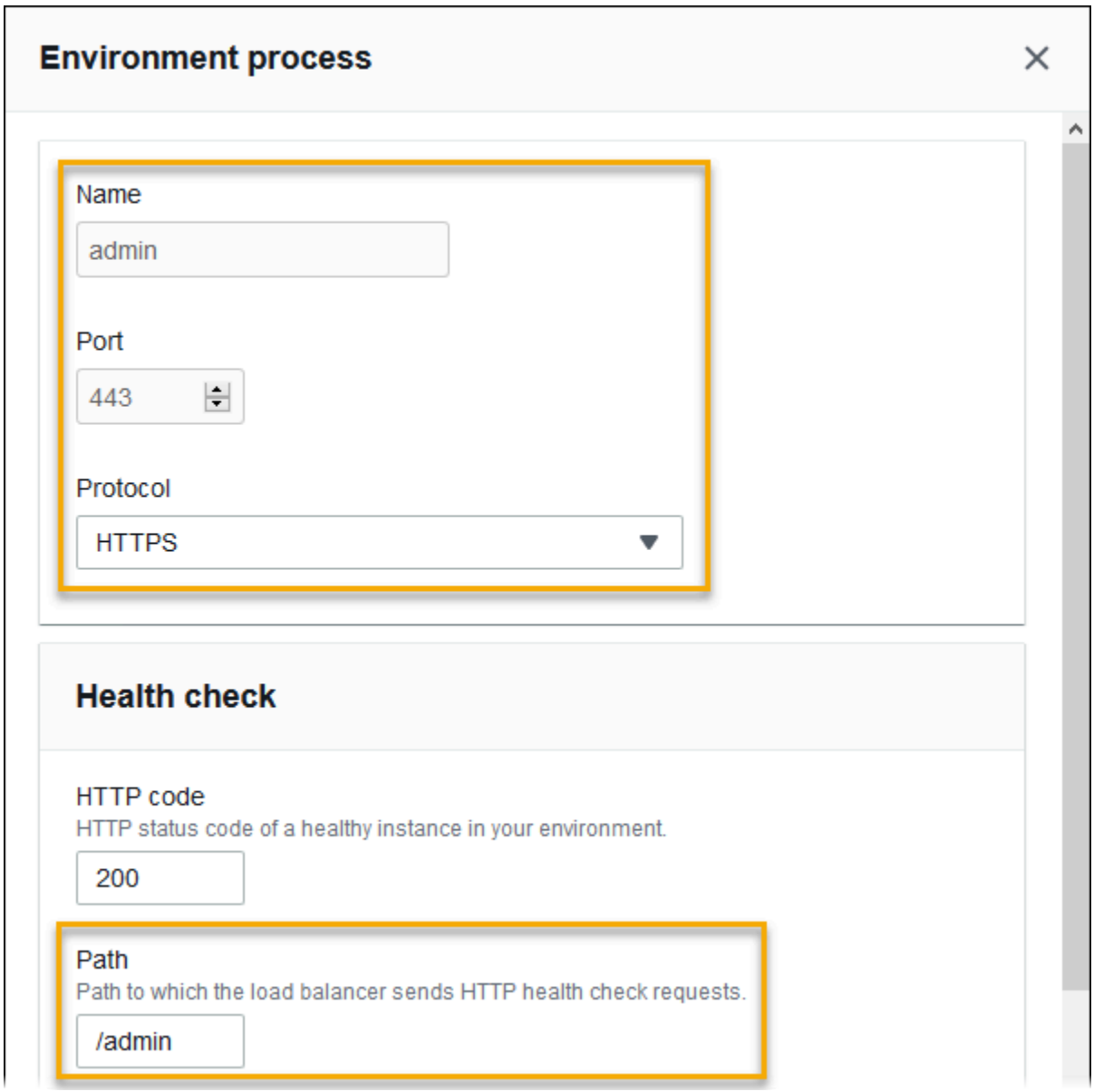
2. Nonaktifkan port default 80 HTTP listener. Untuk pendengar default, matikan opsi Diaktifkan.



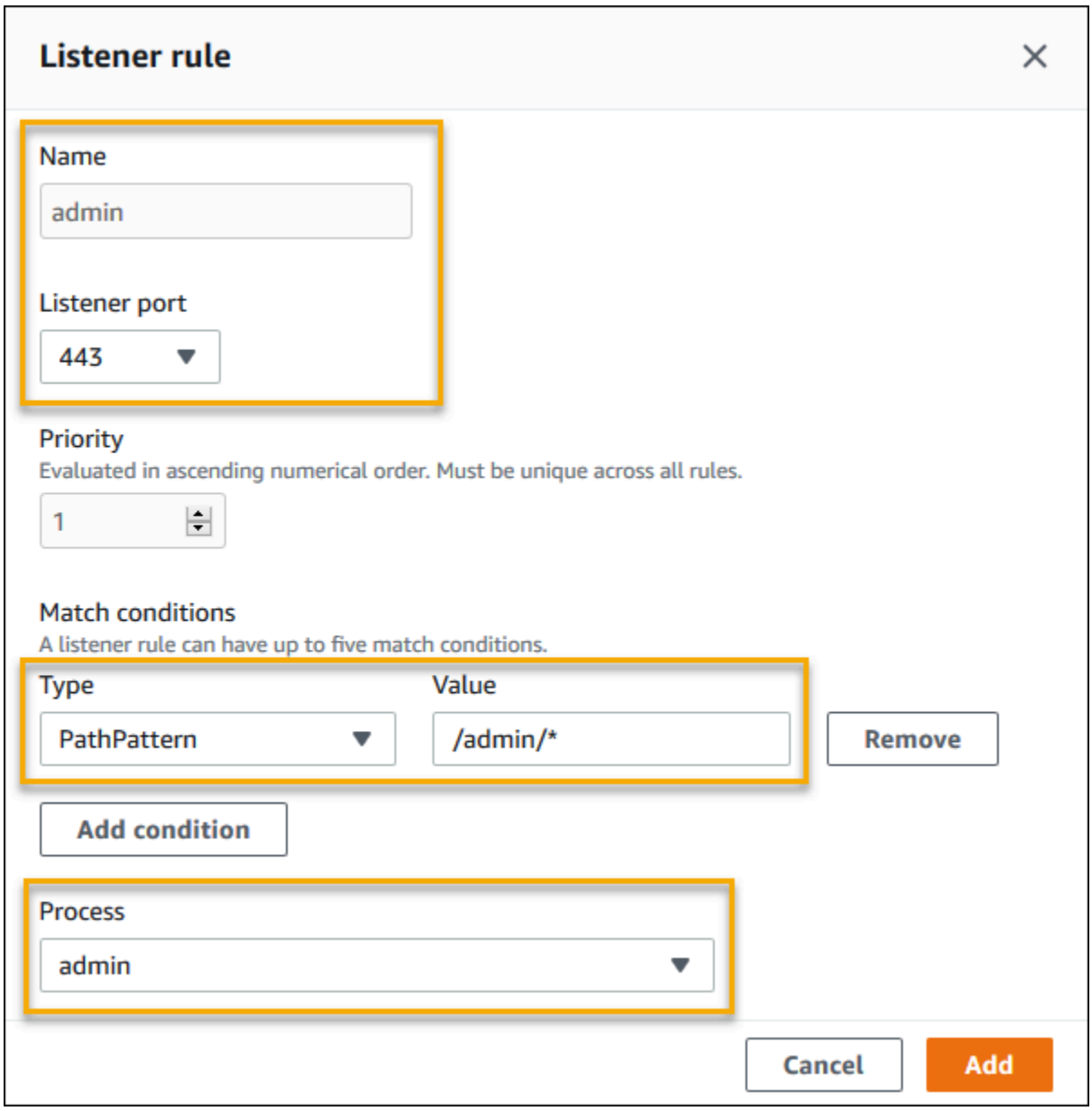
3. Konfigurasikan proses default ke HTTPS. Pilih proses default, dan kemudian untuk Tindakan, pilih Edit. Untuk Port, ketik **`443`**. Untuk Protokol, pilih **`HTTPS`**.



4. Tambahkan proses admin. Untuk Nama, ketik **`admin`**. Untuk Port, ketik **`443`**. Untuk Protokol, pilih **`HTTPS`**. Di bawah Pemeriksaan kondisi, untuk tipe Jalur **`/admin`**.



5. Tambahkan aturan untuk lalu lintas admin. Untuk Nama, ketik **`admin`**. Untuk port Pendengar, ketik **`443`**. Untuk kondisi Match, tambahkan a PathPatterndengan nilai**`/admin/*`**. Untuk Proses, pilih **`admin`**.



Mengonfigurasi Application Load Balancer menggunakan EB CLI

EB CLI meminta Anda untuk memilih tipe penyeimbang beban saat Anda menjalankan eb create.

```
$ eb create
Enter Environment Name
(default is my-app): test-env
Enter DNS CNAME prefix
(default is my-app): test-env-DLW24ED23SF

Select a load balancer type
```

```
1) classic
2) application
3) network
(default is 2):
```

Anda juga dapat menentukan tipe penyeimbang beban dengan opsi `--elb-type`.

```
$ eb create test-env --elb-type application
```

Namespace Application Load Balancer

Anda dapat menemukan pengaturan yang terkait dengan Application Load Balancers di namespace berikut:

- `aws:elasticbeanstalk:environment` — Pilih tipe penyeimbang beban untuk lingkungan. Nilai untuk Application Load Balancer adalah `application`.

  Anda tidak dapat mengatur opsi ini di file konfigurasi (.Ebextensions).
- `aws:elbv2:loadbalancer` — Konfigurasikan log akses dan pengaturan lain yang berlaku untuk Application Load Balancer secara keseluruhan.
- `aws:elbv2:listener` — Konfigurasikan pendengar pada Application Load Balancer. Pengaturan ini memetakan ke pengaturan `aws:elb:listener` untuk Classic Load Balancer.
- `aws:elbv2:listenerrule` — Konfigurasikan aturan yang merutekan lalu lintas ke proses yang berbeda, tergantung pada jalur permintaan. Aturan unik untuk Application Load Balancers.
- `aws:elasticbeanstalk:environment:process` — Konfigurasikan pemeriksaan kondisi dan tentukan port dan protokol untuk proses yang berjalan pada instans lingkungan Anda. Pengaturan port dan protokol memetakan ke port instans dan pengaturan protokol instans `aws:elb:listener` untuk pendengar di Classic Load Balancer. Peta pengaturan pemeriksaan kondisi ke pengaturan di namespace `aws:elb:healthcheck` dan `aws:elasticbeanstalk:application`.

Example .ebextensions/ .config alb-access-logs

File konfigurasi berikut mengizinkan unggah log akses untuk lingkungan dengan Application Load Balancer.

```
option_settings:
  aws:elbv2:loadbalancer:
```

```
    AccessLogsS3Bucket: amzn-s3-demo-bucket
    AccessLogsS3Enabled: 'true'
    AccessLogsS3Prefix: beanstalk-alb
```

Example .ebextensions/ .config alb-default-process

File konfigurasi berikut memodifikasi pemeriksaan kondisi dan pengaturan lekat pada proses default.

```
option_settings:
  aws:elasticbeanstalk:environment:process:default:
    DeregistrationDelay: '20'
    HealthCheckInterval: '15'
    HealthCheckPath: /
    HealthCheckTimeout: '5'
    HealthyThresholdCount: '3'
    UnhealthyThresholdCount: '5'
    Port: '80'
    Protocol: HTTP
    StickinessEnabled: 'true'
    StickinessLBCookieDuration: '43200'
```

Example .ebextensions/ .config alb-secure-listener

File konfigurasi berikut menambahkan pendengar yang aman dan proses pencocokan pada port 443.

```
option_settings:
  aws:elbv2:listener:443:
    DefaultProcess: https
    ListenerEnabled: 'true'
    Protocol: HTTPS
    SSLCertificateArns: arn:aws:acm:us-
east-2:123456789012:certificate/21324896-0fa4-412b-bf6f-f362d6eb6dd7
  aws:elasticbeanstalk:environment:process:https:
    Port: '443'
    Protocol: HTTPS
```

Example .ebextensions/ .config alb-admin-rule

File konfigurasi berikut menambahkan pendengar yang aman dengan aturan yang merutekan lalu lintas dengan jalur permintaan `/admin` ke proses yang bernama `admin` yang mendengarkan pada port 4443.

```
option_settings:
```

```
  aws:elbv2:listener:443:
    DefaultProcess: https
    ListenerEnabled: 'true'
    Protocol: HTTPS
    Rules: admin
    SSLCertificateArns: arn:aws:acm:us-
east-2:123456789012:certificate/21324896-0fa4-412b-bf6f-f362d6eb6dd7
  aws:elasticbeanstalk:environment:process:https:
    Port: '443'
    Protocol: HTTPS
  aws:elasticbeanstalk:environment:process:admin:
    HealthCheckPath: /admin
    Port: '4443'
    Protocol: HTTPS
  aws:elbv2:listenerrule:admin:
    PathPatterns: /admin/*
    Priority: 1
    Process: admin
```

Mengonfigurasi Application Load Balancer bersama

Saat Anda mengaktifkan load balancing, AWS Elastic Beanstalk lingkungan Anda dilengkapi dengan penyeimbang beban Elastic Load Balancing untuk mendistribusikan lalu lintas antar instans di lingkungan Anda. Elastic Load Balancing mendukung beberapa tipe penyeimbang beban. Untuk mempelajarinya, lihat Panduan Pengguna Elastic Load Balancing. Elastic Beanstalk dapat membuat penyeimbang beban untuk Anda, atau memungkinkan Anda menentukan penyeimbang beban bersama yang telah Anda buat.

Topik ini menjelaskan konfigurasi Application Load Balancer bersama yang Anda buat dan asosiasikan dengan lingkungan Anda. Lihat juga the section called “Application Load Balancer”. Untuk informasi tentang cara mengonfigurasi semua tipe penyeimbang beban yang didukung Elastic Beanstalk, lihat Penyeimbang beban untuk lingkungan Elastic Beanstalk Anda.

> **Note**
>
> Anda dapat memilih tipe penyeimbang beban yang hanya digunakan lingkungan Anda selama pembuatan lingkungan. Anda dapat mengubah pengaturan untuk mengelola perilaku penyeimbang beban lingkungan berjalan Anda, tetapi Anda tidak dapat mengubah tipenya. Anda juga tidak dapat beralih dari penyeimbang beban bersama khusus ke penyeimbang beban bersama atau sebaliknya.

Pengantar

Penyeimbang beban bersama adalah penyeimbang beban yang Anda buat dan kelola sendiri menggunakan layanan Amazon Elastic Compute Cloud (Amazon EC2), lalu gunakan di beberapa lingkungan Elastic Beanstalk.

Saat Anda membuat lingkungan penskalaan yang seimbang beban dan memilih untuk menggunakan Application Load Balancer, Elastic Beanstalk membuat penyeimbang beban yang didedikasikan untuk lingkungan Anda secara default. Untuk mempelajari apa itu Application Load Balancer dan cara kerjanya di lingkungan Elastic Beanstalk, lihat pengantar untuk mengonfigurasi Application Load Balancer untuk Elastic Beanstalk.

Dalam beberapa situasi Anda mungkin ingin menghemat biaya dari kepemilikan beberapa penyeimbang beban khusus. Hal ini dapat membantu ketika Anda memiliki beberapa lingkungan, misalnya, jika aplikasi Anda adalah rangkaian layanan mikro bukan layanan monolitik. Dalam kasus seperti itu, Anda dapat memilih untuk menggunakan penyeimbang beban bersama.

Untuk menggunakan penyeimbang beban bersama, buat terlebih dahulu di Amazon EC2 dan tambahkan satu atau beberapa pendengar. Selama pembuatan lingkungan Elastic Beanstalk, Anda kemudian menyediakan penyeimbang beban dan memilih port pendengar. Elastic Beanstalk mengaitkan pendengar dengan proses default di lingkungan Anda. Anda dapat menambahkan aturan pendengar khusus untuk merutekan lalu lintas dari header host tertentu dan jalur untuk proses lingkungan lainnya.

Elastic Beanstalk menambahkan tag ke penyeimbang beban bersama. Nama tag adalah `elasticbeanstalk:shared-elb-environment-count`, dan nilainya adalah jumlah lingkungan yang berbagi penyeimbang beban ini.

Menggunakan penyeimbang beban bersama berbeda dengan menggunakan penyeimbana beban khusus dalam beberapa hal.

Mengenai	Application Load Balancer Khusus	Application Load Balancer Bersama
Manajemer	Elastic Beanstalk membuat dan mengelola penyeimbang beban, pendengar, aturan pendengar, dan proses (grup target). Elastic Beanstalk juga menghapusnya saat Anda mengakhiri lingkunga	Anda membuat dan mengelola penyeimba ng beban dan pendengar di luar Elastic Beanstalk. Elastic Beanstalk membuat dan mengelola aturan default dan proses default, dan Anda dapat menambahkan aturan dan proses. Elastic Beanstalk menghapus aturan

Mengenai	Application Load Balancer Khusus	Application Load Balancer Bersama
	n Anda. Elastic Beanstalk dapat mengatur pengambilan log akses penyeimbang beban, jika Anda memilih opsi itu.	pendengar dan proses yang ditambahkan selama pembuatan lingkungan.
Aturan pendengar	Elastic Beanstalk membuat aturan default untuk setiap pendengar, untuk merutekan semua lalu lintas ke proses default pendengar.	Elastic Beanstalk mengaitkan aturan default hanya dengan port 80 pendengar, jika ada. Jika Anda memilih port pendengar default yang berbeda, Anda harus mengasosiasikan aturan default dengan port pendengar (konsol Elastic Beanstalk dan EB CLI melakukan ini untuk Anda). Untuk mengatasi konflik kondisi aturan pendengar di lingkungan berbagi penyeimba ng beban, Elastic Beanstalk menambahkan CNAME lingkungan untuk aturan pendengar sebagai kondisi header host. Elastic Beanstalk memperlakukan pengatura n prioritas aturan sebagai relatif di lingkunga n yang berbagi penyeimbang beban, dan memetakannya ke prioritas mutlak selama pembuatan.
Grup keamanan	Elastic Beanstalk membuat grup keamanan default dan melampirk annya ke penyeimbang beban.	Anda dapat mengonfigurasi satu atau lebih grup keamanan untuk digunakan sebagai penyeimbang beban. Jika tidak, Elastic Beanstalk memeriksa apakah grup keamanan yang ada yang dikelola Elastic Beanstalk sudah terpasang pada penyeimbang beban. Jika belum, Elastic Beanstalk membuat grup keamanan dan melampirkannya ke penyeimba ng beban. Elastic Beanstalk menghapus grup keamanan ini ketika lingkungan terakhir yang berbagi penyeimbang beban berakhir.

Mengenai	Application Load Balancer Khusus	Application Load Balancer Bersama
Pembaruan	Anda dapat memperbarui Application Load Balancer Anda setelah pembuatan lingkungan. Anda dapat mengedit pendengar, aturan pendengar, dan proses. Anda dapat mengonfigurasi pengambilan log akses penyeimbang beban.	Anda tidak dapat menggunakan Elastic Beanstalk untuk mengonfigurasi pengambilan log akses di Application Load Balancer Anda, dan Anda tidak dapat memperbarui pendengar dan aturan pendengar setelah pembuatan lingkungan. Anda hanya dapat memperbarui proses (grup target). Untuk mengonfigurasi pengambilan log akses, dan untuk memperbarui aturan pendengar dan pendengar, gunakan Amazon. EC2

Mengonfigurasi Application Load Balancer bersama menggunakan konsol Elastic Beanstalk

Anda dapat menggunakan konsol Elastic Beanstalk untuk mengonfigurasi Application Load Balancer bersama selama pembuatan lingkungan. Anda dapat memilih salah satu penyeimbang beban akun yang dapat dibagi untuk digunakan di lingkungan, pilih port pendengar default, dan konfigurasikan proses tambahan dan aturan pendengar.

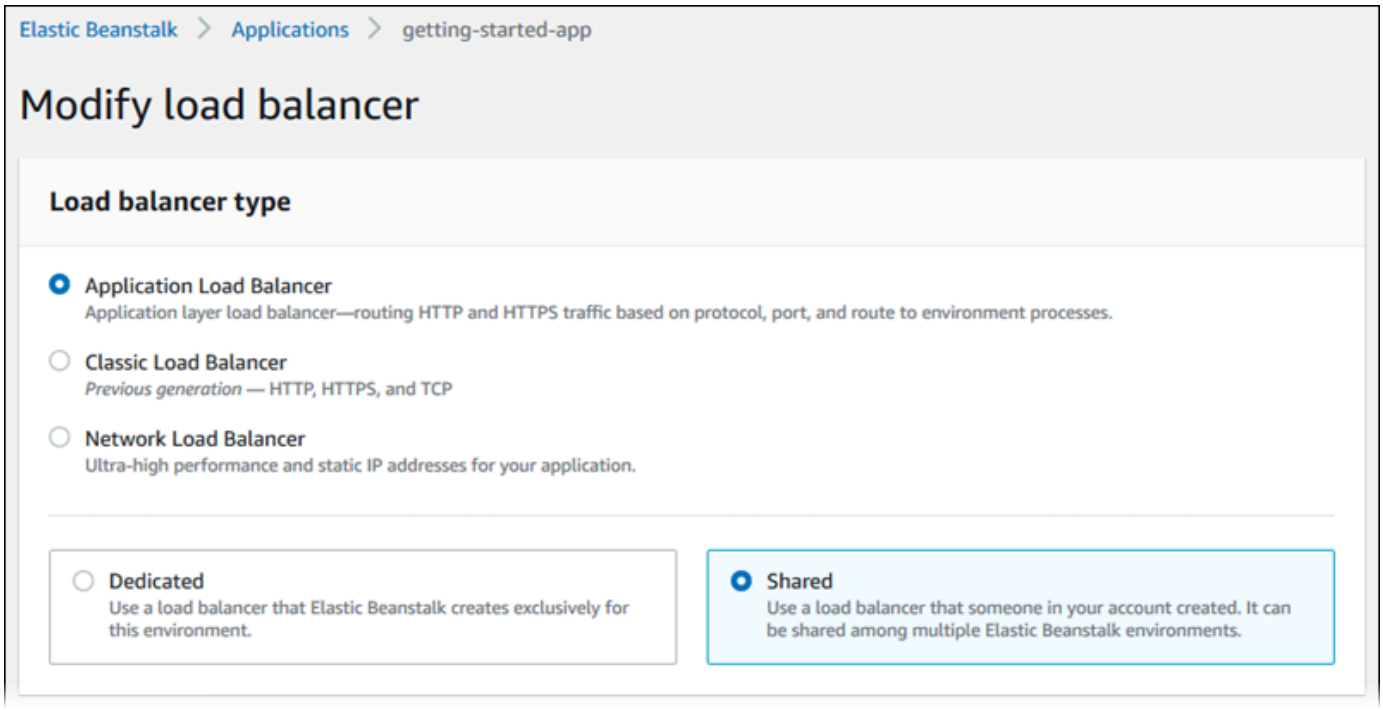
Anda tidak dapat mengedit konfigurasi Application Load Balancer bersama Anda di konsol Application Load Balancer setelah lingkungan Anda dibuat. Untuk mengonfigurasi listener, aturan listener, proses (grup target), dan mengakses penangkapan log, gunakan Amazon. EC2

Untuk mengonfigurasi Application Load Balancer di konsol Elastic Beanstalk selama pembuatan lingkungan

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Pada panel navigasi, pilih Lingkungan.
3. Pilih Buat lingkungan baru untuk mulai membuat lingkungan Anda.
4. Pada halaman utama wizard, sebelum memilih Buat lingkungan, pilih Konfigurasikan opsi lainnya.
5. Pilih konfigurasi Ketersediaan yang tinggi yang telah ditetapkan.

Atau, pada kategori konfigurasi Kapasitas, konfigurasikan tipe lingkungan dengan beban seimbang. Untuk rincian selengkapnya, lihat Kapasitas.

6. Pada kategori konfigurasi Penyeimbang beban, pilih Edit.
7. Pilih opsi Application Load Balancer, jika penyeimbang beban belum dipilih, dan kemudian pilih opsi Dibagikan.



8. Buat perubahan konfigurasi Application Load Balancer bersama yang dibutuhkan lingkungan Anda.
9. Pilih Simpan, dan kemudian buat perubahan konfigurasi lain yang diperlukan lingkungan Anda.
10. Pilih Buat lingkungan.

Pengaturan Application Load Balancer Bersama

- Application Load Balancer Bersama
- Proses
- Aturan

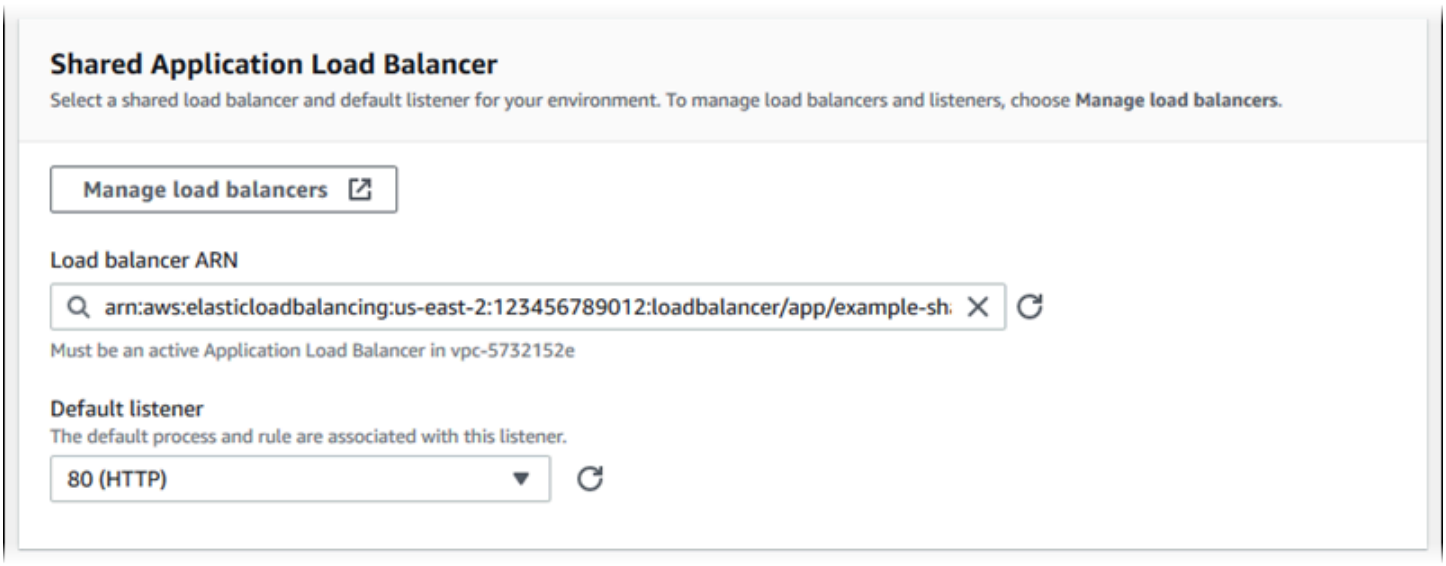
Application Load Balancer Bersama

Gunakan bagian ini untuk memilih Application Load Balancer bersama untuk lingkungan Anda dan konfigurasikan perutean lalu lintas default.

Sebelum Anda dapat mengonfigurasi Application Load Balancer bersama di sini, gunakan Amazon EC2 untuk menentukan setidaknya satu Application Load Balancer untuk dibagikan, dengan setidaknya satu pendengar, di akun Anda. Jika Anda belum melakukannya, Anda dapat memilih Mengelola penyeimbang beban. Elastic Beanstalk membuka konsol Amazon EC2 di tab browser baru.

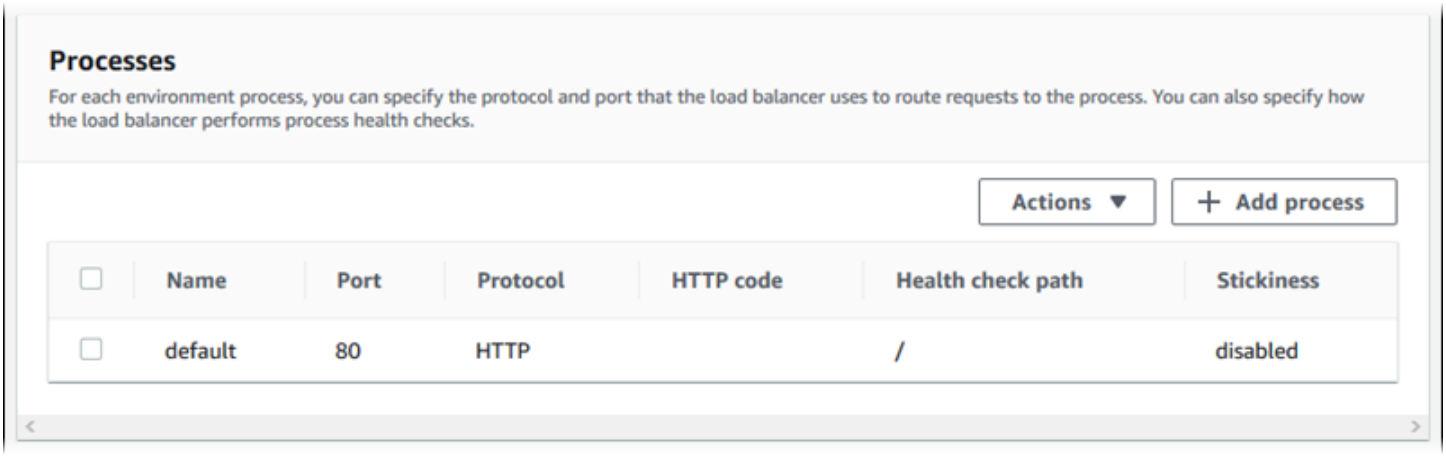
Setelah selesai mengonfigurasi penyeimbang beban bersama di luar Elastic Beanstalk, konfigurasikan pengaturan berikut di bagian konsol ini:

- ARN penyeimbang beban — Penyeimbang beban bersama yang akan digunakan di lingkungan ini. Pilih dari daftar penyeimbang beban atau masukkan penyeimbang beban Amazon Resource Name (ARN).
- Port pendengar default — Port pendengar yang diperhatikan oleh penyeimbang beban bersama. Pilih dari daftar port pendengar yang sudah ada. Lalu lintas dari pendengar ini dengan CNAME lingkungan di header host dirutekan ke proses default di lingkungan ini.



Proses

Gunakan daftar ini untuk menentukan proses penyeimbang beban bersama Anda. Sebuah proses adalah target bagi pendengar untuk merutekan lalu lintas. Awalnya, daftar menunjukkan proses default, yang menerima lalu lintas dari pendengar default.



Untuk mengonfigurasi proses yang ada

1. Pilih kotak centang di samping entri tabel, dan kemudian pilih Tindakan, Edit.
2. Gunakan kotak dialog Proses lingkungan untuk mengedit pengaturan, dan kemudian pilih Simpan.

Untuk menambahkan proses

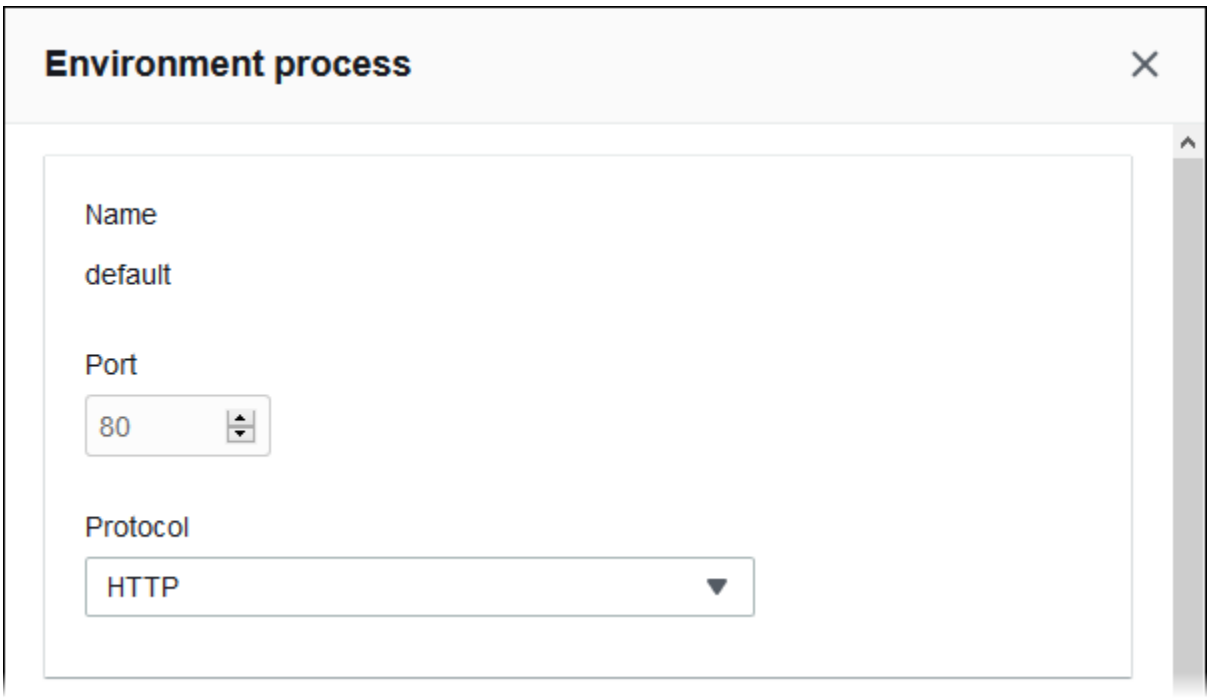
1. Pilih Tambah proses.
2. Di kotak dialog Proses lingkungan, konfigurasikan pengaturan yang Anda inginkan, dan kemudian pilih Tambahkan.

Pengaturan kotak dialog proses lingkungan Application Load Balancer

- Definisi
- Pemeriksaan kondisi
- Sesi

Definisi

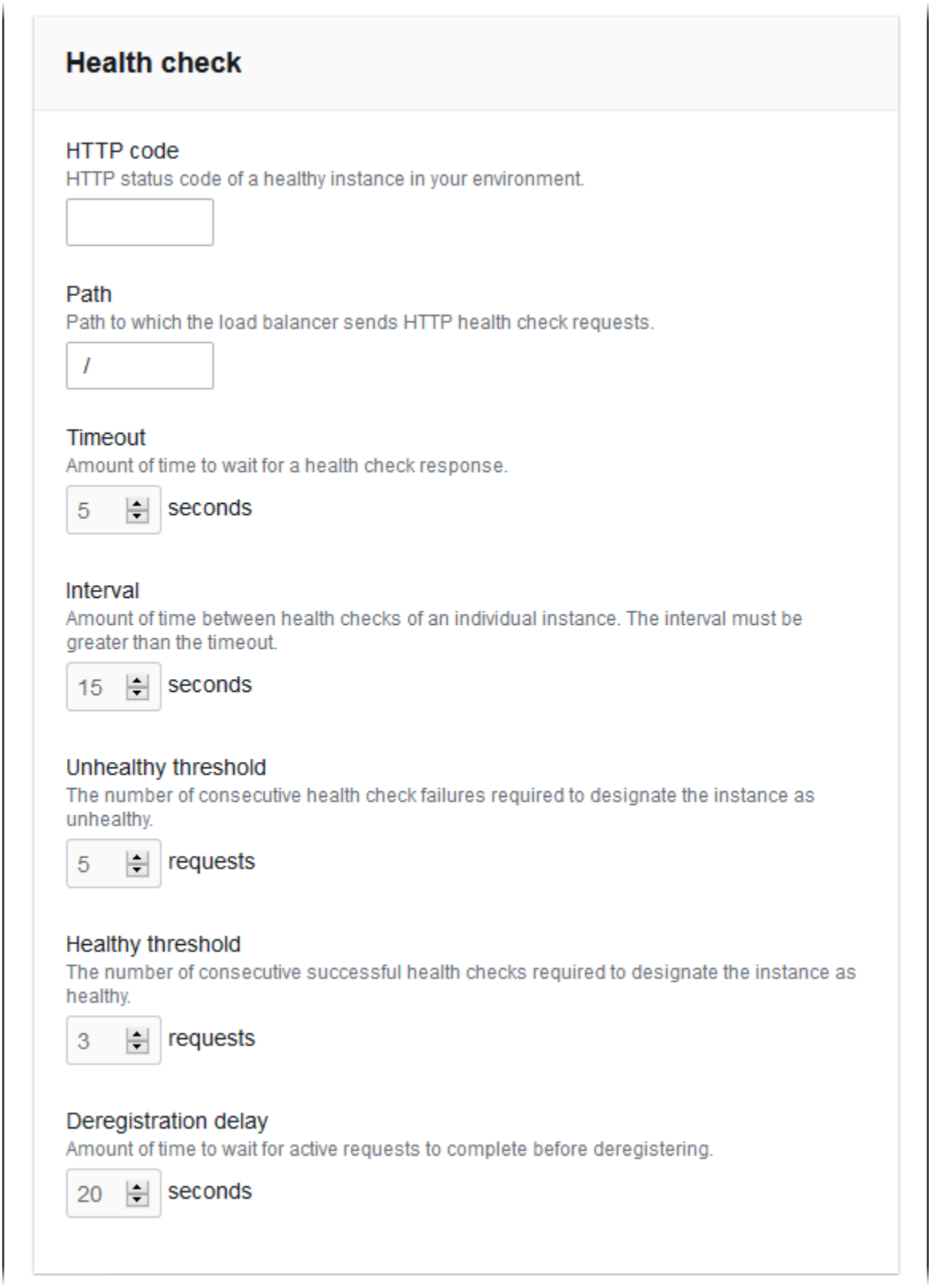
Gunakan pengaturan ini untuk menentukan proses: Nama, dan Port dan Protokol yang mendengarkan permintaan.



Pemeriksaan kondisi

Gunakan pengaturan berikut untuk mengonfigurasi pemeriksaan kondisi proses:

- Kode HTTP — Kode status HTTP menunjuk proses yang sehat.
- Jalur — Jalur permintaan pemeriksaan kondisi untuk proses tersebut.
- Batas waktu — Jumlah waktu, dalam detik, untuk menunggu respons pemeriksaan kondisi.
- Interval — Jumlah waktu, dalam detik, antara pemeriksaan kondisi dari instans individu. Interval harus lebih besar dari batas waktu.
- Ambang batas tidak sehat, Ambang batas sehat — Jumlah pemeriksaan kondisi yang harus gagal atau lulus, masing-masing, sebelum Elastic Load Balancing mengubah status kondisi instans.
- Penundaan pendaftaran — Jumlah waktu, dalam detik, untuk menunggu permintaan aktif selesai sebelum membatalkan pendaftaran.

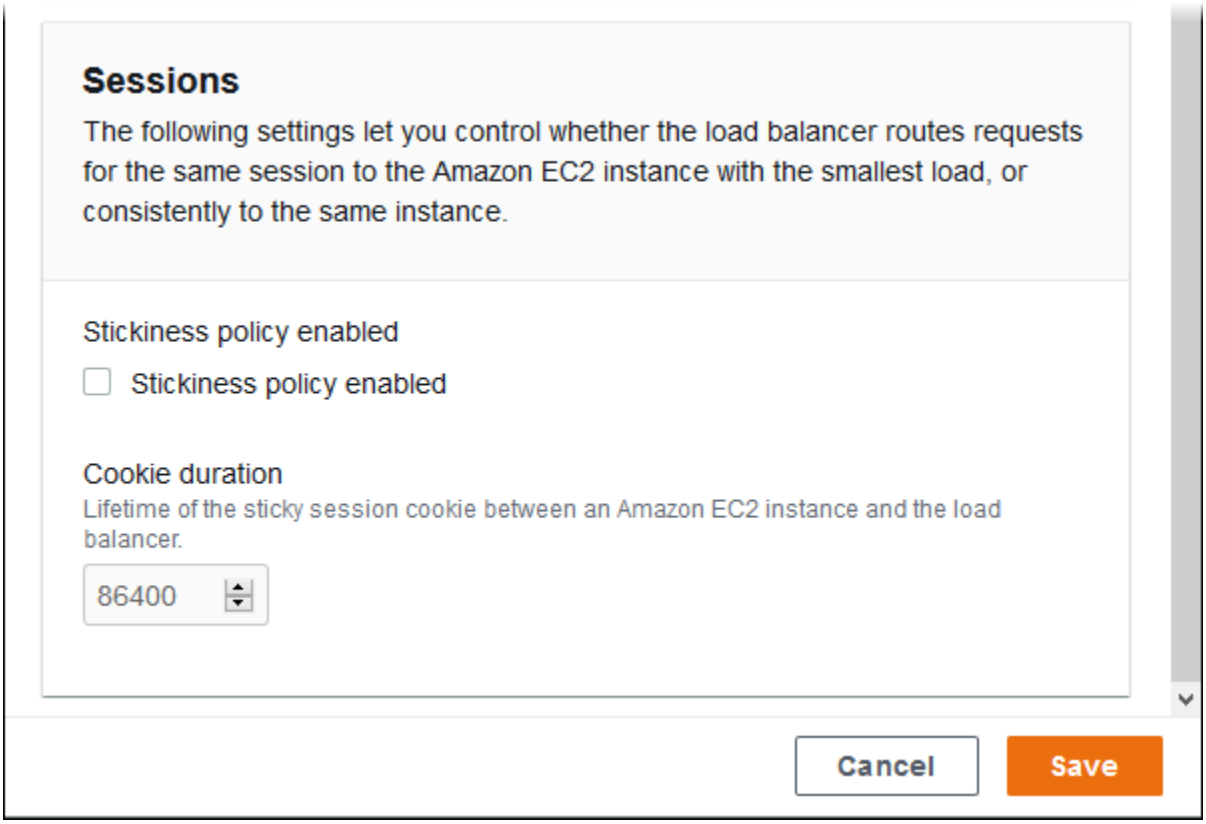

Health check
HTTP code
HTTP status code of a healthy instance in your environment.
Path
Path to which the load balancer sends HTTP health check requests.
/
Timeout
Amount of time to wait for a health check response.
5 seconds
Interval
Amount of time between health checks of an individual instance. The interval must be greater than the timeout.
15 seconds
Unhealthy threshold
The number of consecutive health check failures required to designate the instance as unhealthy.
5 requests
Healthy threshold
The number of consecutive successful health checks required to designate the instance as healthy.
3 requests
Deregistration delay
Amount of time to wait for active requests to complete before deregistering.
20 seconds

> **Note**
>
> Pemeriksaan kondisi Elastic Load Balancing tidak memengaruhi perilaku pemeriksaan kondisi grup Auto Scaling lingkungan. Instans yang gagal dalam pemeriksaan kesehatan Elastic Load Balancing tidak secara otomatis digantikan oleh Amazon Auto EC2 Scaling kecuali Anda mengonfigurasi Amazon Auto Scaling secara manual untuk EC2 melakukannya. Lihat Pengaturan pemeriksaan kesehatan Auto Scaling untuk lingkungan Elastic Beanstalk Anda untuk rincian selengkapnya.

Untuk informasi selengkapnya tentang pemeriksaan kondisi dan pengaruhnya terhadap kondisi lingkungan Anda secara keseluruhan, lihat Pelaporan kondisi dasar.

Sesi

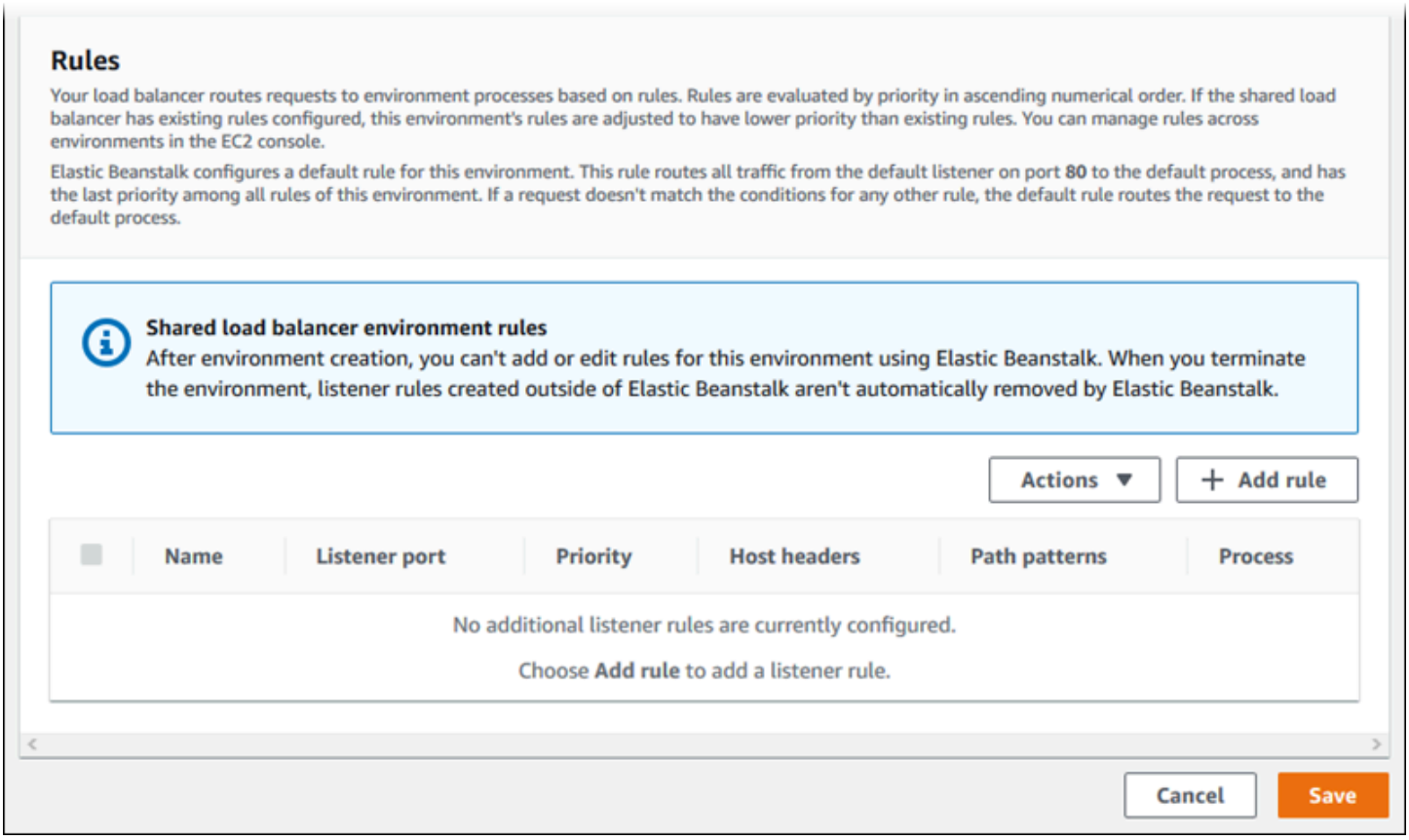
Pilih atau kosongkan kotak Kebijakan kelekatan diaktifkan untuk mengaktifkan atau menonaktifkan sesi lekat. Gunakan durasi cookie untuk mengonfigurasi durasi sesi lekat, hingga **`604800`** detik.



Aturan

Gunakan daftar ini untuk menentukan aturan pendengar khusus untuk penyeimbang beban bersama Anda. Sebuah aturan memetakan permintaan yang diterima pendengar pada pola jalur tertentu untuk proses target. Setiap pendengar dapat memiliki beberapa aturan, merutekan permintaan pada jalur yang berbeda untuk proses yang berbeda pada instans lingkungan yang berbeda berbagi pendengar.

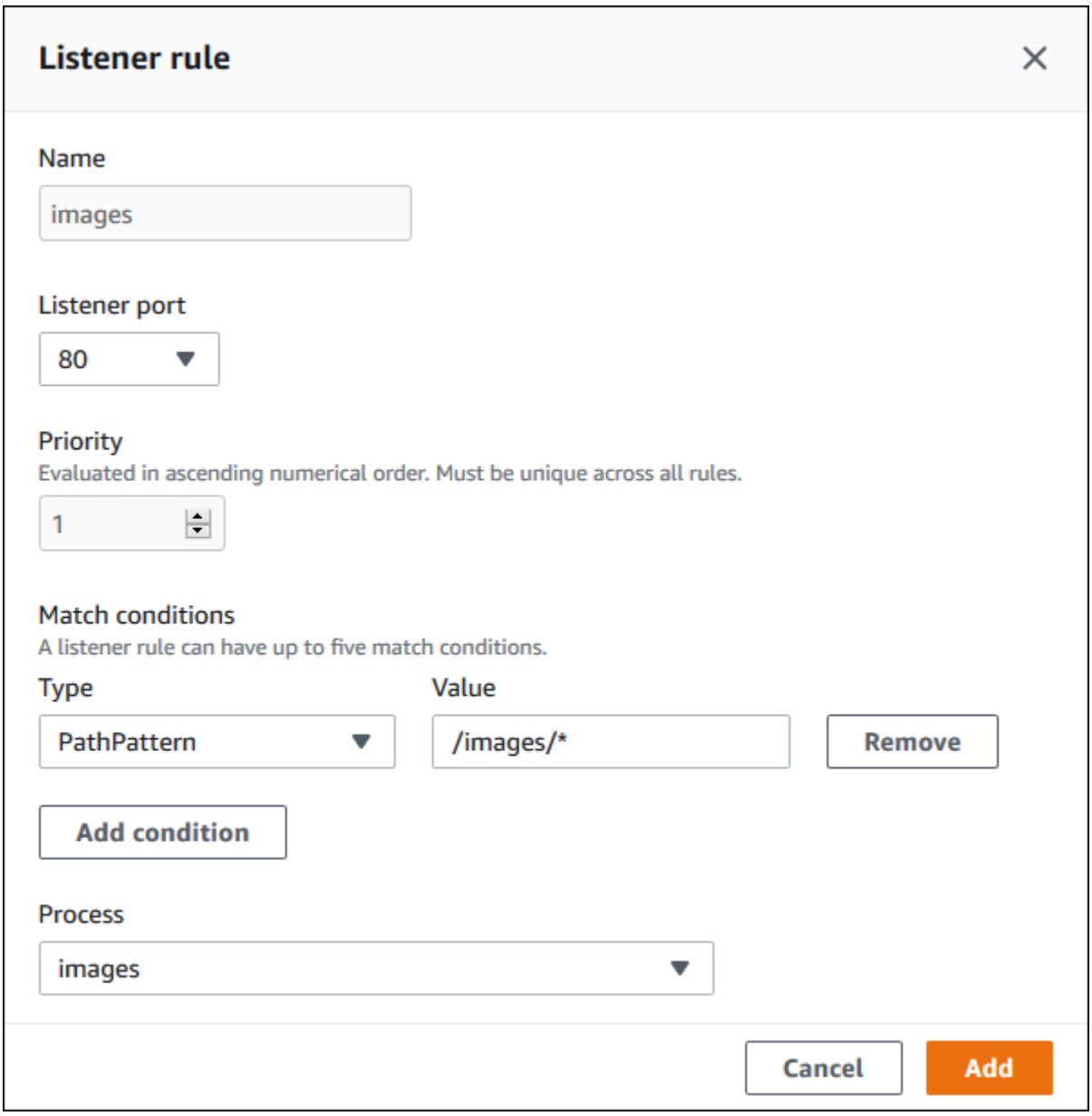
Aturan memiliki prioritas numerik yang menentukan prioritas penerapannya pada permintaan yang masuk. Elastic Beanstalk menambahkan aturan default yang merutekan semua lalu lintas pendengar default untuk proses default lingkungan baru Anda. Prioritas aturan default adalah yang terendah; itu diterapkan jika tidak ada aturan lain untuk pendengar yang sama cocok dengan permintaan masuk. Awalnya, jika Anda belum menambahkan aturan khusus, daftar tersebut kosong. Aturan default tidak ditampilkan.



Anda dapat mengedit pengaturan aturan yang ada, atau menambahkan aturan baru. Untuk mulai mengedit aturan pada daftar atau menambahkan aturan, gunakan langkah yang sama yang tercantum untuk Daftar proses. Kotak dialog Aturan pendengar terbuka, dengan pengaturan berikut:

- Nama - Nama aturan.
- port Pendengar — port Pendengar yang aturannya berlaku.

- Prioritas - Prioritas aturan. Sebuah nomor prioritas yang lebih rendah memiliki prioritas yang lebih tinggi. Prioritas aturan pendengar harus unik. Elastic Beanstalk memperlakukan prioritas aturan sebagai relatif di lingkungan berbagi, dan memetakan mereka ke prioritas mutlak selama pembuatan.
- Kondisi yang cocok — Daftar kondisi URL permintaan tempat aturan berlaku. Ada dua jenis kondisi: HostHeader(bagian domain URL), dan PathPattern(bagian jalur URL). Satu kondisi dicadangkan untuk subdomain lingkungan, dan Anda dapat menambahkan hingga empat kondisi. Setiap nilai kondisi mencapai 128 karakter panjangnya, dan dapat mencakup karakter wildcard.
- Proses — Proses dimana penyeimbang beban merutekan permintaan yang sesuai dengan aturan.



Contoh: menggunakan Application Load Balancer bersama untuk aplikasi yang aman micro-service-based

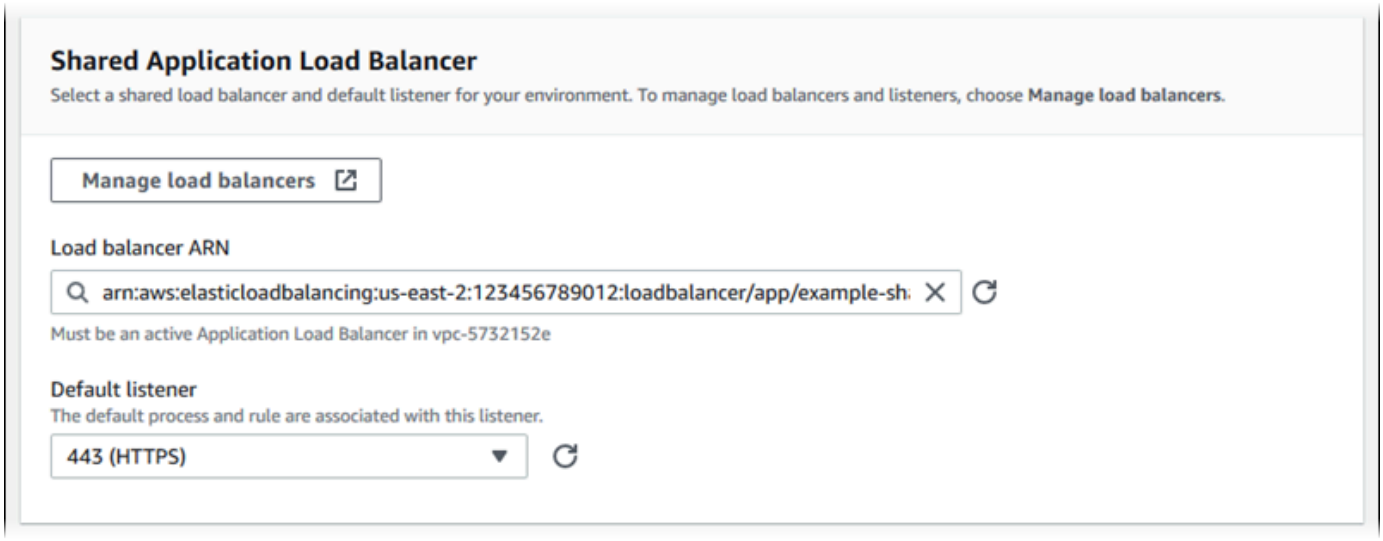
Dalam contoh ini, aplikasi Anda terdiri dari beberapa layanan mikro, masing-masing diimplementasikan sebagai lingkungan Elastic Beanstalk. Selain itu, Anda memerlukan enkripsi end-to-end lalu lintas. Kami akan menunjukkan salah satu lingkungan layanan mikro, yang memiliki proses utama untuk permintaan pengguna dan proses terpisah untuk menangani permintaan administratif.

Untuk memenuhi persyaratan ini, gunakan Amazon EC2 untuk membuat Application Load Balancer yang akan Anda bagikan di antara layanan mikro Anda. Tambahkan pendengar yang aman pada port 443 dan protokol HTTPS. Kemudian tambahkan beberapa sertifikat SSL ke pendengar—satu per domain layanan mikro. Untuk detail tentang membuat Application Load Balancer dan pendengar yang aman, lihat Buat Application Load Balancer dan Buat pendengar HTTPS untuk Application Load Balancer di Panduan pengguna untuk Application Load Balancers.

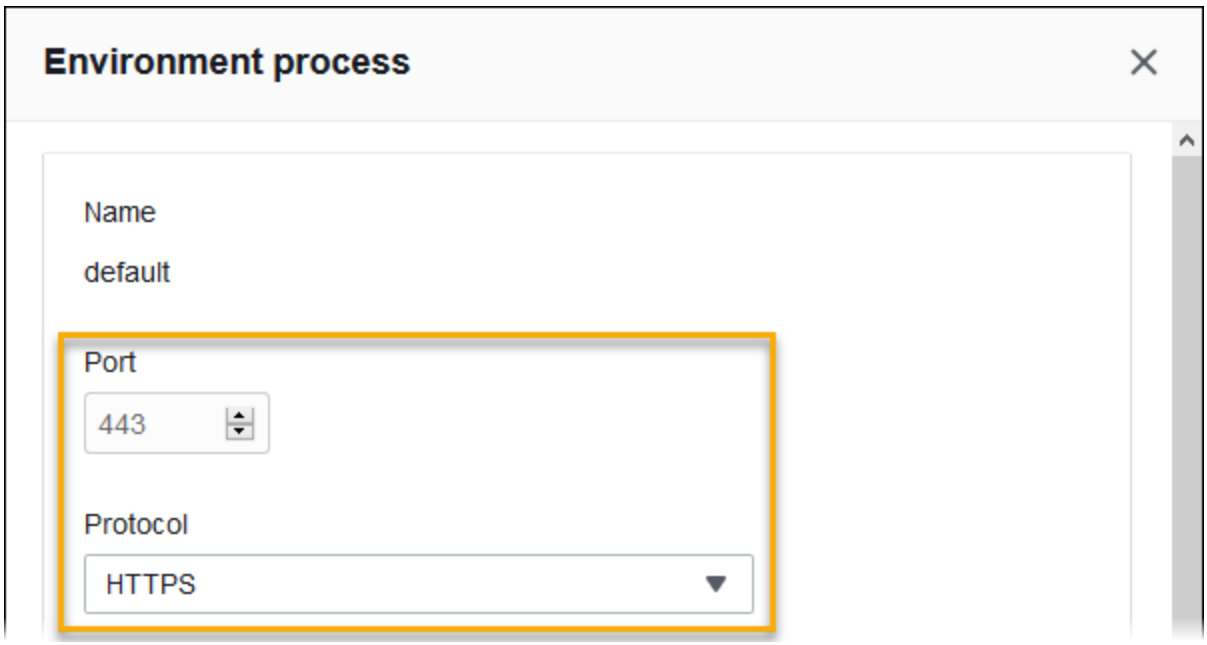
Dalam Elastic Beanstalk, konfigurasikan setiap lingkungan layanan mikro untuk menggunakan Application Load Balancer bersama dan atur port pendengar default ke 443. Dalam kasus lingkungan tertentu yang kami tunjukkan di sini, menunjukkan bahwa proses default mendengarkan port 443 pada HTTPS, dan menambahkan proses dan aturan pendengar untuk lalu lintas admin pada jalur yang berbeda.

Untuk mengonfigurasi penyeimbang beban bersama untuk contoh ini

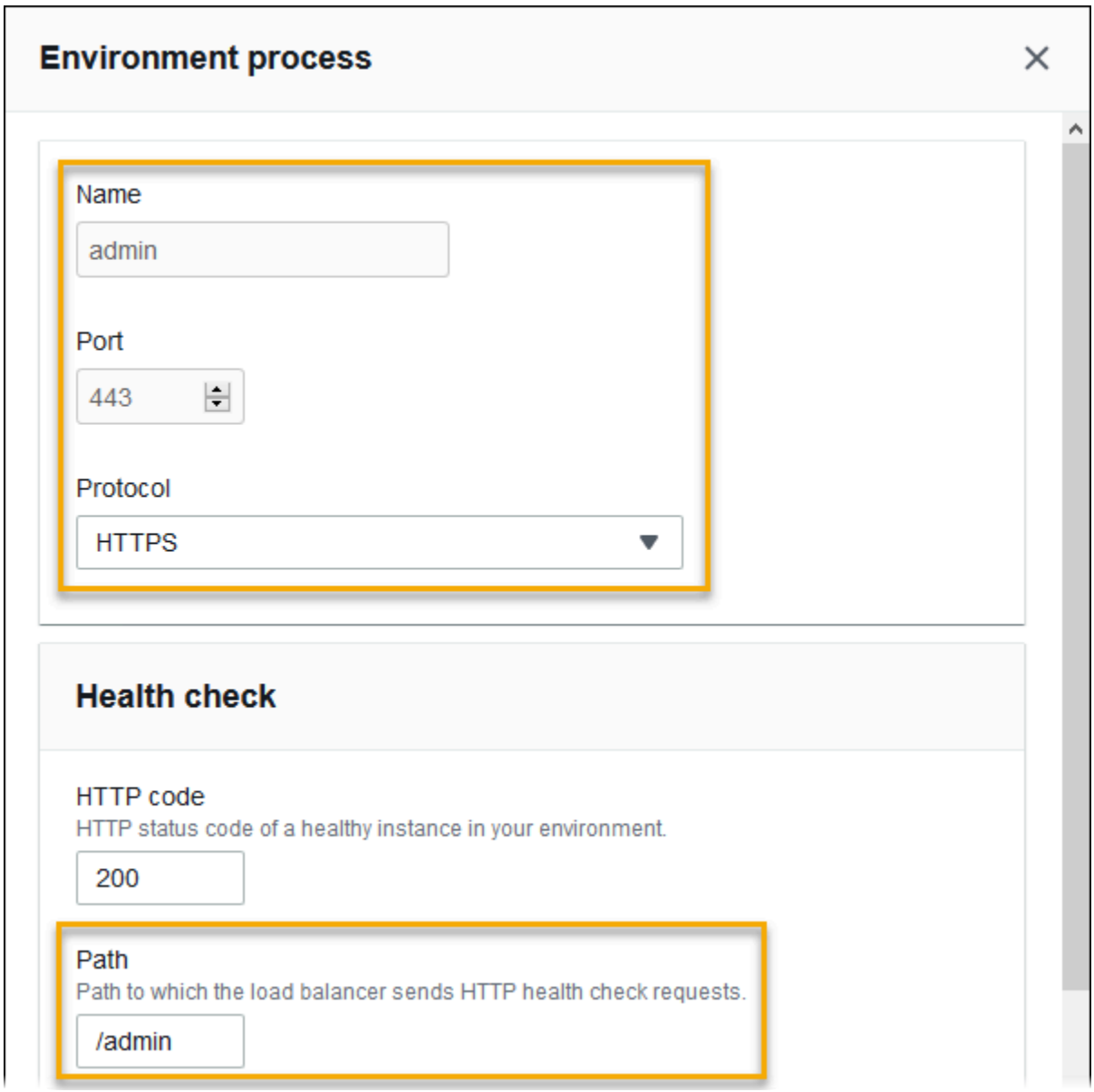
1. Di bagian Application Load Balancer Bersama, pilih penyeimbang beban Anda, dan kemudian, untuk Port pendengar default, pilih **443**. Port pendengar seharusnya sudah dipilih jika itu satu-satunya pendengar yang dimiliki penyeimbang beban.



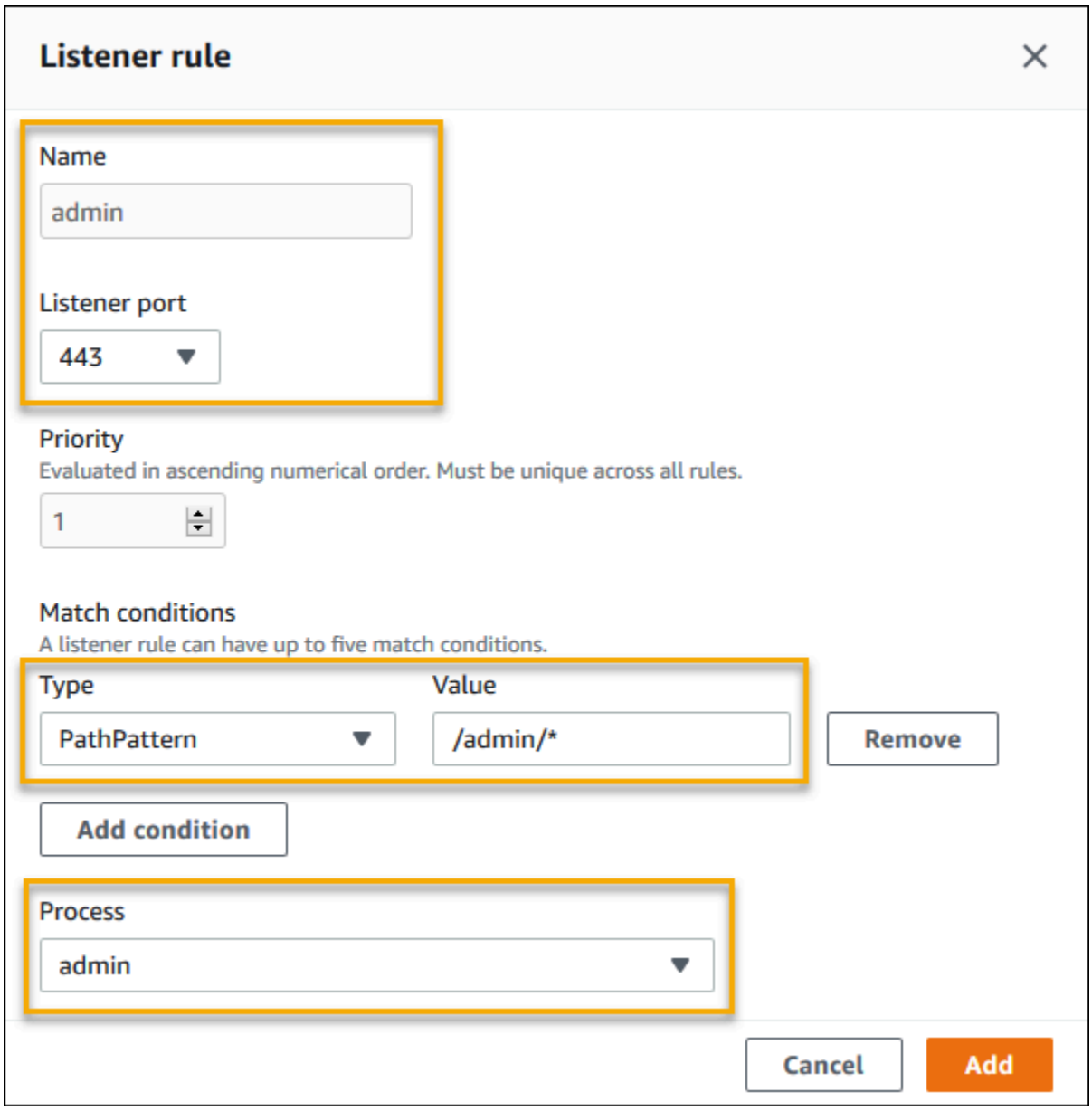
2. Konfigurasikan proses default ke HTTPS. Pilih proses default, dan kemudian untuk Tindakan, pilih Edit. Untuk Port, masukkan **`443`**. Untuk Protokol, pilih **`HTTPS`**.



3. Tambahkan proses admin. Untuk Nama, masukkan **`admin`**. Untuk Port, masukkan **`443`**. Untuk Protokol, pilih **`HTTPS`**. Di bawah Pemeriksaan kondisi, untuk Jalur masukkan **`/admin`**.



4. Tambahkan aturan untuk lalu lintas admin. Untuk Nama, masukkan **`admin`**. Untuk Port pendengar, masukkan **`443`**. Untuk kondisi Match, tambahkan a PathPatterndengan nilai**`/admin/*`**. Untuk Proses, pilih **`admin`**.



Mengonfigurasi Application Load Balancer bersama menggunakan EB CLI

EB CLI meminta Anda untuk memilih tipe penyeimbang beban saat Anda menjalankan eb create. Jika Anda memilih `application` (default), dan jika akun Anda memiliki setidaknya satu Application Load Balancer yang dapat dibagikan, EB CLI juga menanyakan apakah Anda ingin menggunakan Application Load Balancer bersama. Jika Anda menjawab **y**, Anda juga diminta untuk memilih penyeimbang beban dan port default.

```
$ eb create
Enter Environment Name
```

```
(default is my-app): test-env
Enter DNS CNAME prefix
(default is my-app): test-env-DLW24ED23SF

Select a load balancer type
1) classic
2) application
3) network
(default is 2):

Your account has one or more sharable load balancers. Would you like your new
 environment to use a shared load balancer?(y/N) y

Select a shared load balancer
1)MySharedALB1 - arn:aws:elasticloadbalancing:us-east-2:123456789012:loadbalancer/app/
MySharedALB1/6d69caa75b15d46e
2)MySharedALB2 - arn:aws:elasticloadbalancing:us-east-2:123456789012:loadbalancer/app/
MySharedALB2/e574ea4c37ad2ec8
(default is 1): 2

Select a listener port for your shared load balancer
1) 80
2) 100
3) 443
(default is 1): 3
```

Anda juga dapat menentukan penyeimbang beban bersama menggunakan opsi perintah.

```
$ eb create test-env --elb-type application --shared-lb MySharedALB2 --shared-lb-
port 443
```

Namespace Application Load Balancer Bersama

Anda dapat menemukan pengaturan yang terkait dengan Application Load Balancers bersama di namespace berikut:

- `aws:elasticbeanstalk:environment` — Pilih tipe penyeimbang beban untuk lingkungan, dan beri tahu Elastic Beanstalk bahwa Anda akan menggunakan penyeimbang beban bersama.

  Anda tidak dapat mengatur dua opsi ini di file konfigurasi (.Ebextensions).
- `aws:elbv2:loadbalancer` — Konfigurasikan ARN Application Load Balancer bersama dan grup keamanan.

- `aws:elbv2:listener` — Hubungkan pendengar Application Load Balancer bersama dengan proses lingkungan dengan mencantumkan aturan pendengar.
- `aws:elbv2:listenerrule` — Konfigurasikan aturan pendengar yang merutekan lalu lintas ke proses yang berbeda, tergantung pada jalur permintaan. Aturan unik untuk Application Load Balancers—baik khusus maupun bersama.
- `aws:elasticbeanstalk:environment:process` — Konfigurasikan pemeriksaan kondisi dan tentukan port dan protokol untuk proses yang berjalan pada instans lingkungan Anda.

Example .ebextensions/ .config application-load-balancer-shared

Untuk memulai dengan Application Load Balancer bersama, gunakan konsol Elastic Beanstalk, EB CLI, atau API untuk mengatur tipe penyeimbang beban `application` dan memilih untuk menggunakan penyeimbang beban bersama. Gunakan file konfigurasi untuk mengonfigurasi penyeimbang beban bersama.

```
option_settings:
  aws:elbv2:loadbalancer:
    SharedLoadBalancer: arn:aws:elasticloadbalancing:us-
east-2:123456789012:loadbalancer/app/MySharedALB2/e574ea4c37ad2ec8
```

> **Note**
>
> Anda dapat mengonfigurasi opsi ini hanya selama pembuatan lingkungan.

Example .ebextensions/ .config alb-shared-secure-listener

File konfigurasi berikut memilih pendengar aman default pada port 443 untuk penyeimbang beban bersama, dan menetapkan proses default untuk mendengarkan port 443.

```
option_settings:
  aws:elbv2:loadbalancer:
    SharedLoadBalancer: arn:aws:elasticloadbalancing:us-
east-2:123456789012:loadbalancer/app/MySharedALB2/e574ea4c37ad2ec8
  aws:elbv2:listener:443:
    rules: default
  aws:elasticbeanstalk:environment:process:default:
    Port: '443'
```

```
    Protocol: HTTPS
```

Example .ebextensions/ .config alb-shared-admin-rule

File konfigurasi berikut dibangun pada contoh sebelumnya dan menambahkan aturan yang merutekan lalu lintas dengan jalur permintaan `/admin` ke proses yang bernama `admin` yang mendengarkan port 4443.

```
option_settings:
  aws:elbv2:loadbalancer:
    SharedLoadBalancer: arn:aws:elasticloadbalancing:us-
east-2:123456789012:loadbalancer/app/MySharedALB2/e574ea4c37ad2ec8
  aws:elbv2:listener:443:
    rules: default,admin
  aws:elasticbeanstalk:environment:process:default:
    Port: '443'
    Protocol: HTTPS
  aws:elasticbeanstalk:environment:process:admin:
    HealthCheckPath: /admin
    Port: '4443'
    Protocol: HTTPS
  aws:elbv2:listenerrule:admin:
    PathPatterns: /admin/*
    Priority: 1
    Process: admin
```

Mengonfigurasi Network Load Balancer

Saat Anda mengaktifkan load balancing, AWS Elastic Beanstalk lingkungan Anda dilengkapi dengan penyeimbang beban Elastic Load Balancing untuk mendistribusikan lalu lintas antar instans di lingkungan Anda. Elastic Load Balancing mendukung beberapa tipe penyeimbang beban. Untuk mempelajarinya, lihat Panduan Pengguna Elastic Load Balancing. Elastic Beanstalk dapat membuat penyeimbang beban untuk Anda, atau memungkinkan Anda menentukan penyeimbang beban bersama yang telah Anda buat.

Topik ini menjelaskan konfigurasi Network Load Balancer yang dibuat Elastic Beanstalk dan didedikasikan untuk lingkungan Anda. Untuk informasi tentang cara mengonfigurasi semua tipe penyeimbang beban yang didukung Elastic Beanstalk, lihat Penyeimbang beban untuk lingkungan Elastic Beanstalk Anda.

> **Note**
>
> Anda dapat memilih tipe penyeimbang beban yang hanya digunakan lingkungan Anda selama pembuatan lingkungan. Anda dapat mengubah pengaturan untuk mengelola perilaku penyeimbang beban lingkungan berjalan Anda, tetapi Anda tidak dapat mengubah tipenya.

Pengantar

Dengan Network Load Balancer, pendengar default menerima permintaan TCP pada port 80 dan mendistribusikannya ke instans di lingkungan Anda. Anda dapat mengonfigurasi perilaku pemeriksaan kondisi, mengonfigurasi port pendengar, atau menambahkan pendengar pada port lain.

> **Note**
>
> Tidak seperti Classic Load Balancer atau Application Load Balancer, Network Load Balancer tidak bisa memiliki pendengar HTTP atau HTTPS lapisan aplikasi (layer 7). Network Load Balancer hanya mendukung pendengar TCP.lapisan transport (lapisan 4). Lalu lintas HTTP dan HTTPS dapat dirutekan ke lingkungan Anda melalui TCP. Untuk membuat koneksi HTTPS yang aman antara klien web dan lingkungan Anda, instal sertifikat yang ditandatangani sendiri pada instans lingkungan, dan konfigurasikan instas tersebut untuk mendengarkan pada port yang sesuai (biasanya 443) dan mengakhiri koneksi HTTPS. Konfigurasi bervariasi per platform. Lihat Mengkonfigurasi Pengakhiran HTTPS pada instance untuk instruksi. Kemudian konfigurasikan Network Load Balancer Anda untuk menambahkan pendengar yang memetakan ke proses mendengarkan pada port yang sesuai.

Network Load Balancer mendukung pemeriksaan kondisi aktif. Pemeriksaan ini didasarkan pada pesan ke jalur (/) root. Selain itu, Network Load Balancer mendukung pemeriksaan kondisi pasif. Pemeriksaan ini secara otomatis mendeteksi instans backend yang rusak dan merutekan lalu lintas hanya untuk instans yang sehat.

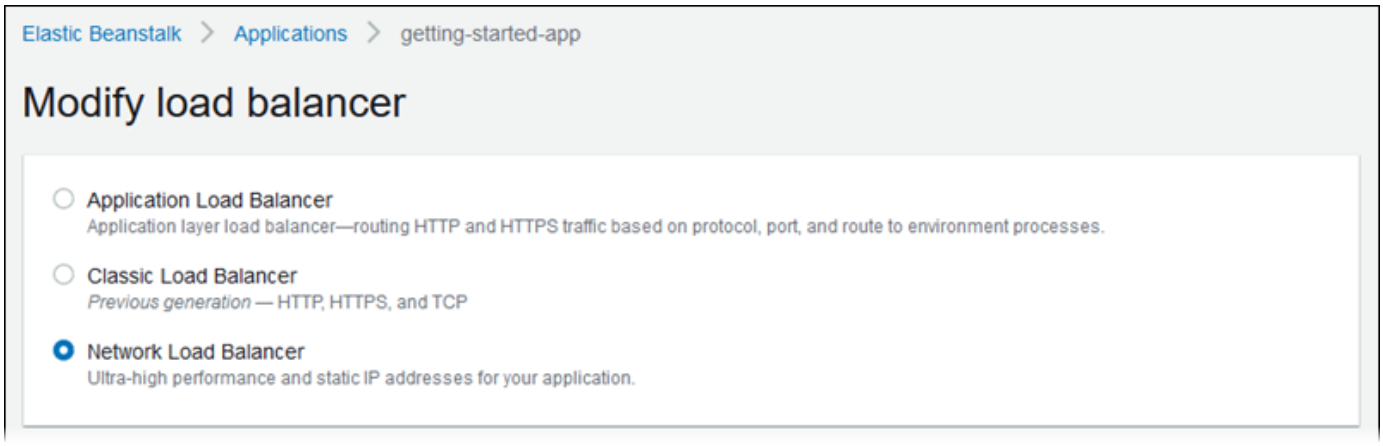
Mengonfigurasi Network Load Balancer menggunakan konsol Elastic Beanstalk

Anda dapat menggunakan konsol Elastic Beanstalk untuk mengonfigurasi pendengar dan proses Network Load Balancer selama pembuatan lingkungan, atau nanti saat lingkungan Anda berjalan.

Untuk menkonfigurasi Network Load Balancer di konsol Elastic Beanstalk selama pembuatan lingkungan

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Pada panel navigasi, pilih Lingkungan.
3. Pilih Buat lingkungan baru untuk mulai membuat lingkungan Anda.
4. Pada halaman utama wizard, sebelum memilih Buat lingkungan, pilih Konfigurasikan opsi lainnya.
5. Pilih konfigurasi Ketersediaan yang tinggi yang telah ditetapkan.

   Atau, pada kategori konfigurasi Kapasitas, konfigurasikan tipe lingkungan dengan beban seimbang. Untuk rincian selengkapnya, lihat Kapasitas.
6. Pada kategori konfigurasi Penyeimbang beban, pilih Edit.
7. Pilih opsi Network Load Balancer, jika penyeimbang beban belum dipilih.



8. Buat perubahan konfigurasi Network Load Balancer yang diperlukan lingkungan Anda.
9. Pilih Simpan, dan kemudian buat perubahan konfigurasi lain yang diperlukan lingkungan Anda.
10. Pilih Buat lingkungan.

Untuk mengonfigurasi Network Load Balancer lingkungan yang sedang berjalan di konsol Elastic Beanstalk

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.

3. Pada panel navigasi, pilih Konfigurasi.
4. Pada kategori konfigurasi Penyeimbang beban, pilih Edit.

> **Note**
>
> Jika kategori konfigurasi Penyeimbang beban tidak memiliki tombol Edit, lingkungan Anda tidak memiliki penyeimbang beban. Untuk mempelajari cara menyiapkan penyeimbang beban, lihat Mengubah jenis lingkungan.

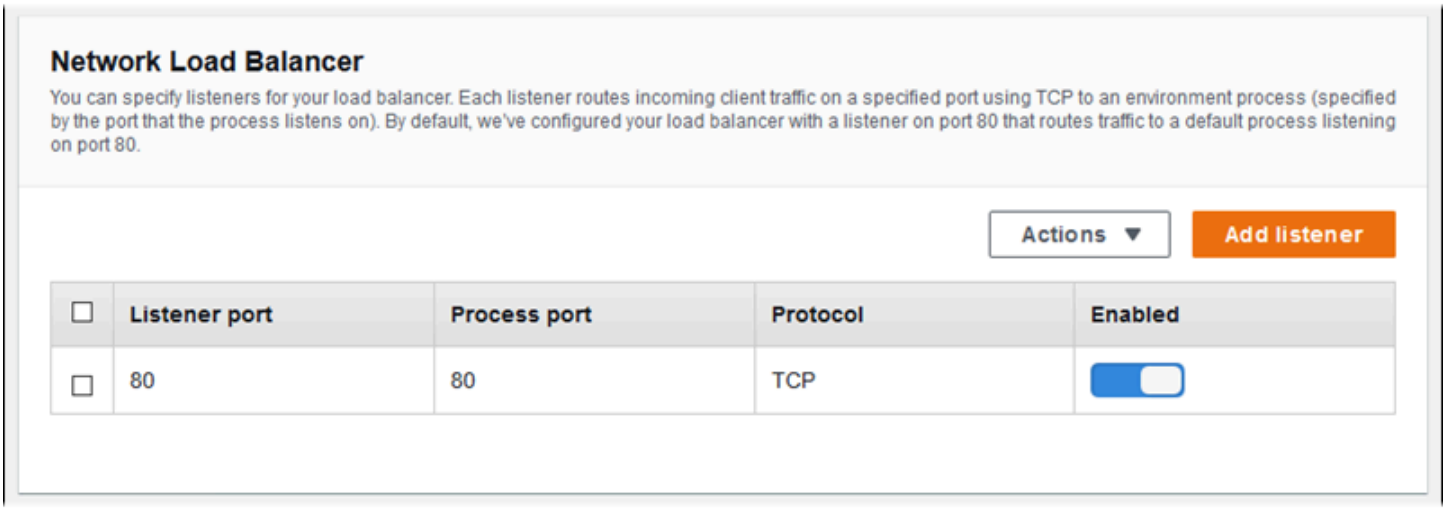
5. Buat perubahan konfigurasi Network Load Balancer yang diperlukan lingkungan Anda.
6. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Pengaturan Network Load Balancer

- Listener
- Proses

Listener

Gunakan daftar ini untuk menentukan pendengar untuk penyeimbang beban Anda. Setiap pendengar merutekan lalu lintas klien yang masuk pada port tertentu ke proses di instans Anda. Awalnya, daftar menunjukkan pendengar default, yang merutekan lalu lintas masuk pada port 80 untuk proses bernama default, yang mendengarkan port 80.



Untuk mengonfigurasi pendengar yang ada

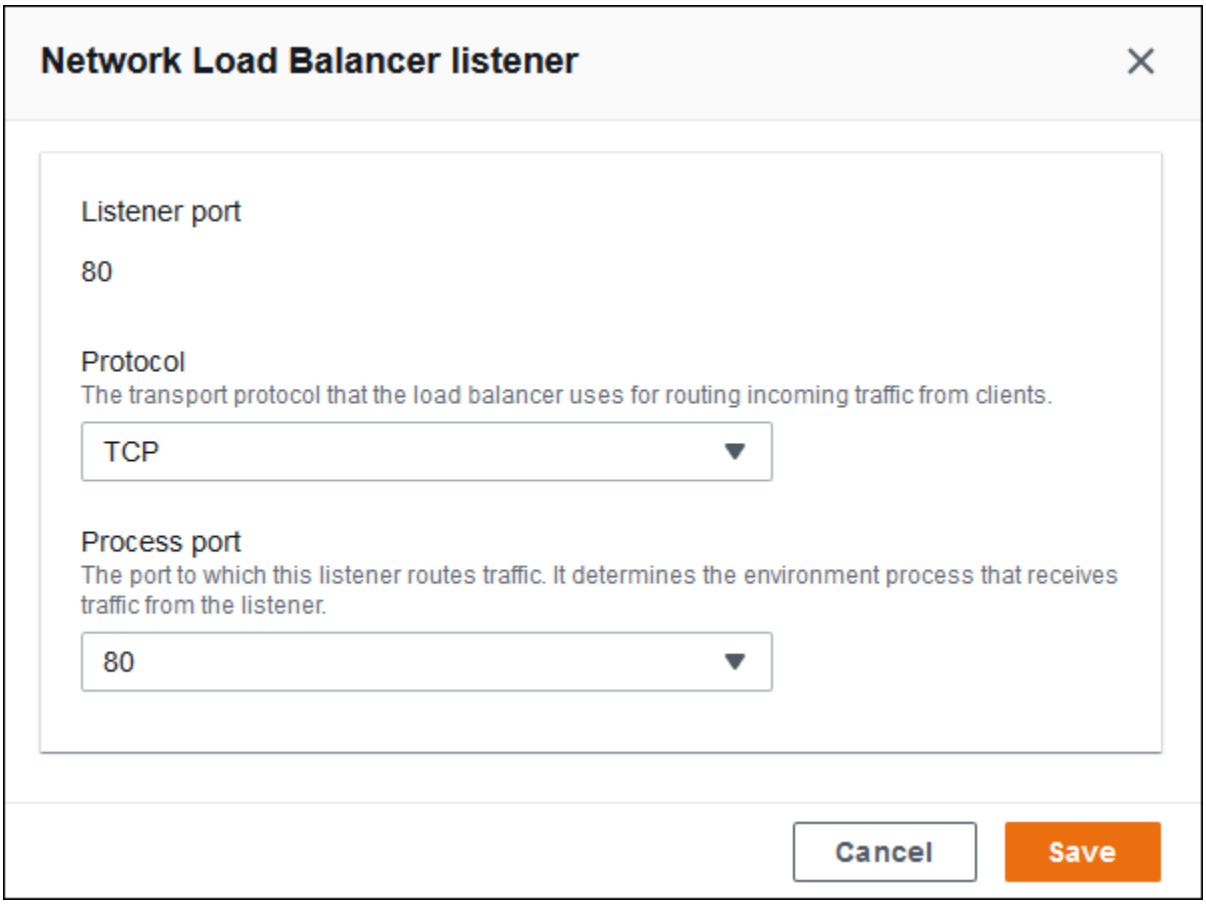
1. Pilih kotak centang di samping entri tabel, dan kemudian pilih Tindakan, Edit.

2. Gunakan kotak dialog pendengar Load Balancer untuk mengedit pengaturan, dan kemudian pilih Simpan.

Untuk menambahkan pendengar

1. Pilih Tambahkan pendengar.
2. Di kotak dialog pendengar Network Load Balancer, konfigurasikan pengaturan yang diperlukan, dan kemudian pilih Tambahkan.

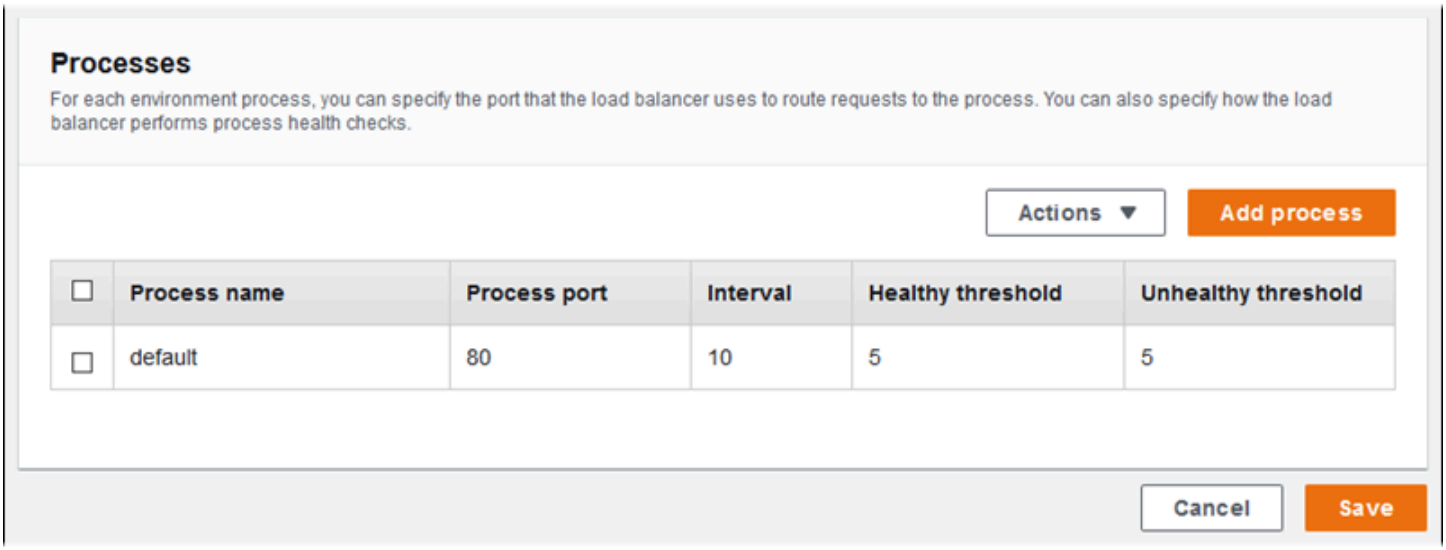
Gunakan kotak dialog Pendengar Network Load Balancer untuk mengonfigurasi port tempat pendengar mendengarkan lalu lintas, dan memilih proses tempat Anda ingin merutekan lalu lintas (ditentukan oleh port yang didengarkan oleh proses).



Proses

Gunakan daftar ini untuk menentukan proses penyeimbang beban Anda. Sebuah proses adalah target bagi pendengar untuk merutekan lalu lintas. Setiap pendengar merutekan lalu lintas klien yang

masuk pada port tertentu ke proses di instans Anda. Awalnya, daftar menunjukkan proses default, yang mendengarkan lalu lintas masuk di port 80.



Anda dapat mengedit pengaturan proses yang ada, atau menambahkan proses baru. Untuk mulai mengedit proses pada daftar atau menambahkan proses untuk itu, gunakan langkah yang sama yang tercantum untuk daftar pendengar. Kotak dialog Proses lingkungan terbuka.

Pengaturan kotak dialog proses lingkungan Network Load Balancer

- Definisi
- Pemeriksaan kondisi

Definisi

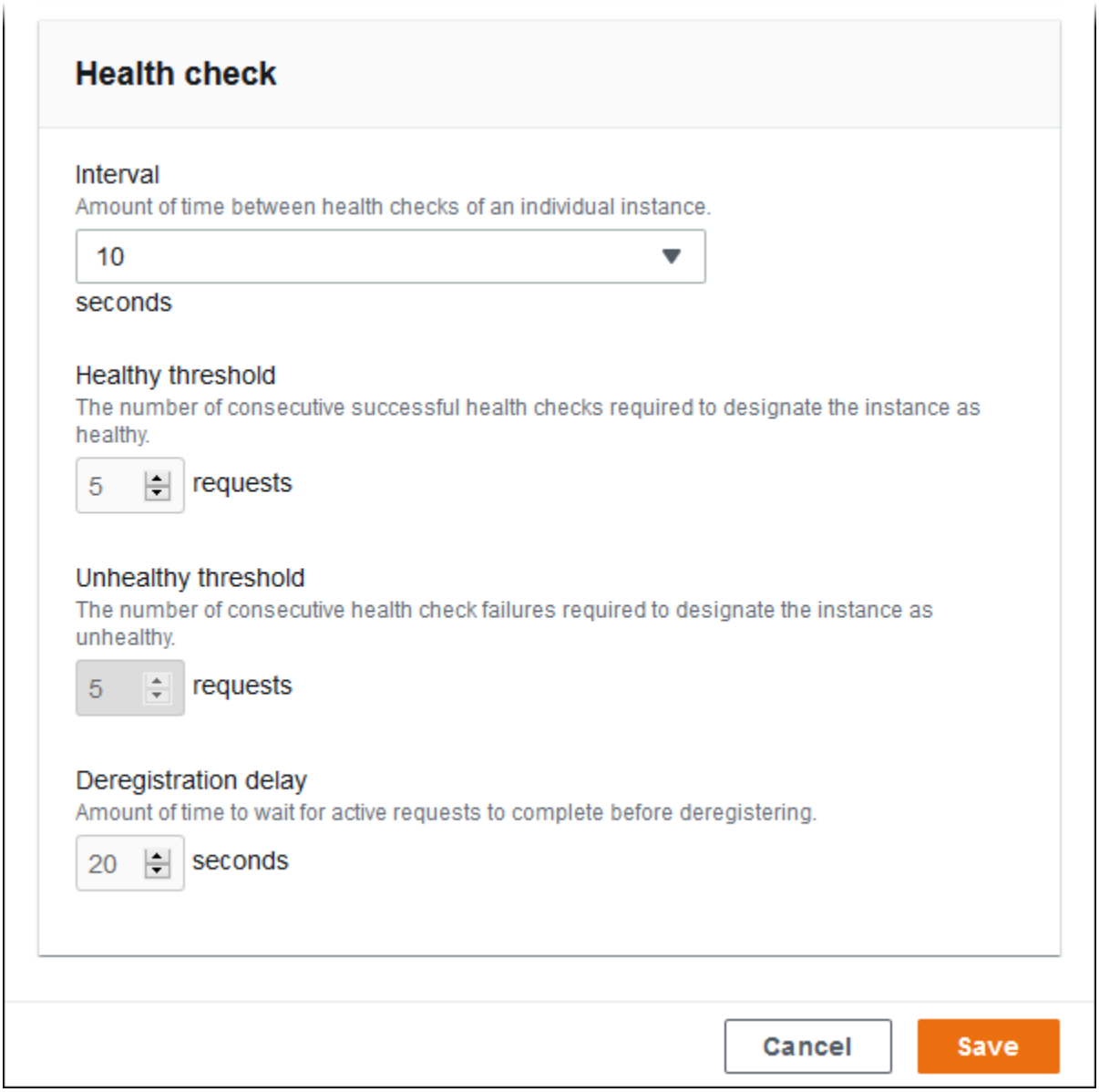
Gunakan pengaturan ini untuk menentukan proses: Nama dan Port proses yang mendengarkan permintaan.



Pemeriksaan kondisi

Gunakan pengaturan berikut untuk mengonfigurasi pemeriksaan kondisi proses:

- Interval — Jumlah waktu, dalam detik, antara pemeriksaan kondisi dari instans individu.
- Ambang batas sehat — Jumlah pemeriksaan kondisi yang harus dilalui sebelum Elastic Load Balancing mengubah status kondisi instans. (Untuk Network Load Balancer, Ambang batas tidak sehat adalah pengaturan hanya-baca yang selalu sama dengan nilai ambang sehat.)
- Penundaan pendaftaran — Jumlah waktu, dalam detik, untuk menunggu permintaan aktif selesai sebelum membatalkan pendaftaran.



> **Note**
>
> Pemeriksaan kondisi Elastic Load Balancing tidak memengaruhi perilaku pemeriksaan kondisi grup Auto Scaling lingkungan. Instans yang gagal dalam pemeriksaan kesehatan Elastic Load Balancing tidak akan otomatis diganti oleh Amazon Auto EC2 Scaling kecuali Anda mengonfigurasi Amazon Auto Scaling secara manual untuk EC2 melakukannya. Lihat Pengaturan pemeriksaan kesehatan Auto Scaling untuk lingkungan Elastic Beanstalk Anda untuk rincian selengkapnya.

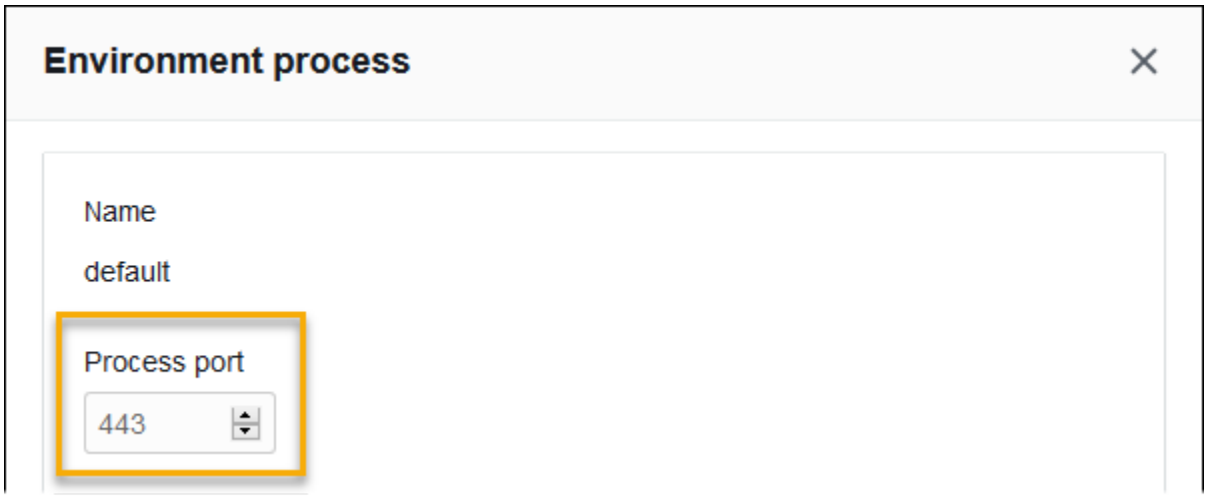
Untuk informasi selengkapnya tentang pemeriksaan kondisi dan pengaruhnya terhadap kondisi lingkungan Anda secara keseluruhan, lihat Pelaporan kondisi dasar.

Contoh: Network Load Balancer untuk lingkungan dengan enkripsi end-to-end

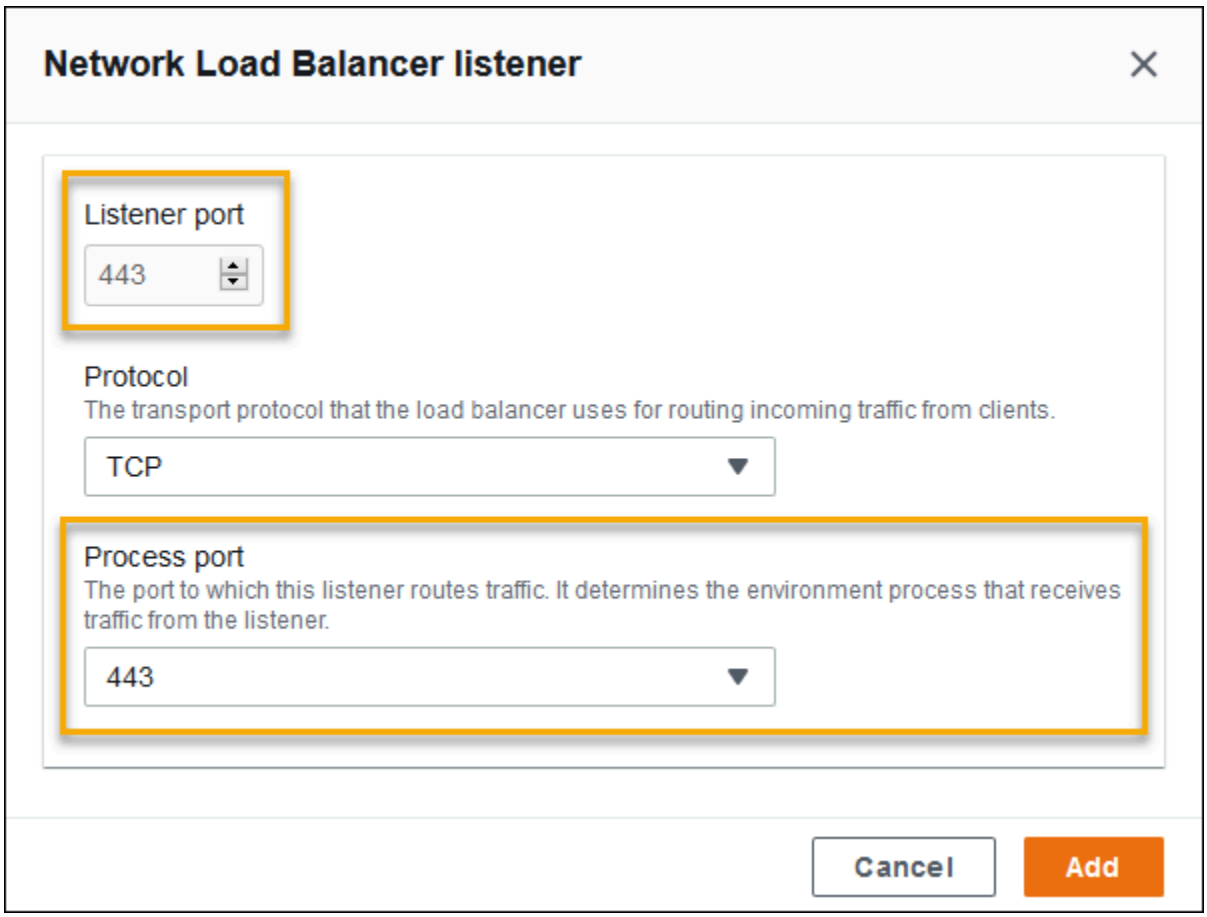
Dalam contoh ini, aplikasi Anda memerlukan enkripsi end-to-end lalu lintas. Untuk mengonfigurasi Network Load Balancer lingkungan Anda memenuhi persyaratan ini, Anda mengonfigurasi proses default untuk mendengarkan port 443, menambahkan pendengar ke port 443 yang merutekan lalu lintas ke proses default, dan menonaktifkan pendengar default.

Untuk mengonfigurasi penyeimbang beban pada contoh ini

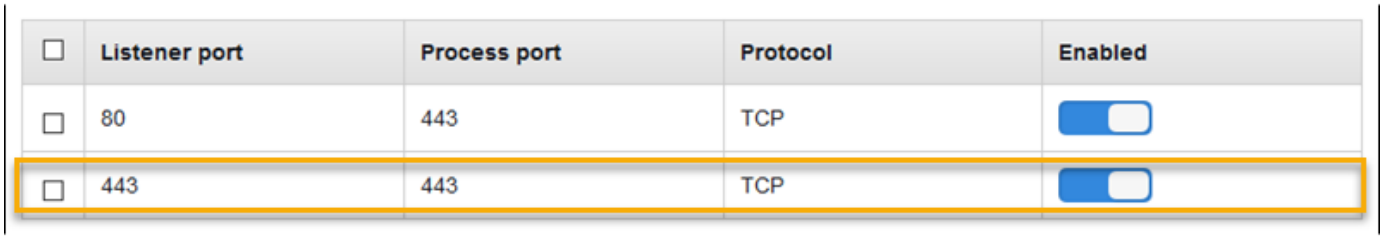
1. Konfigurasikan proses default. Pilih proses default, dan kemudian, untuk Tindakan, pilih Edit. Untuk port Proses, ketik `443`.



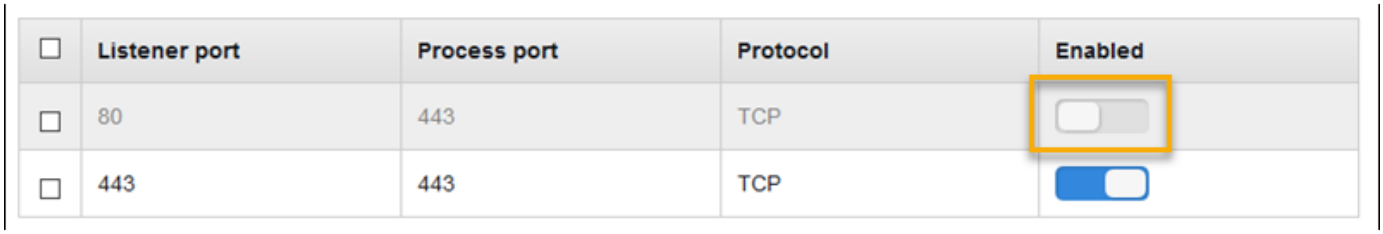
2. Tambahkan pendengar port 443. Tambahkan pendengar baru. Untuk port Pendengar, ketik `443`. Untuk port Proses, pastikan bahwa `443` dipilih.



Anda sekarang dapat melihat pendengar tambahan Anda pada daftar.



3. Nonaktifkan pendengar port 80 default. Untuk pendengar default, matikan opsi Diaktifkan.



Mengonfigurasi Network Load Balancer menggunakan EB CLI

EB CLI meminta Anda untuk memilih tipe penyeimbang beban saat Anda menjalankan eb create.

```
$ eb create
Enter Environment Name
(default is my-app): test-env
Enter DNS CNAME prefix
(default is my-app): test-env-DLW24ED23SF

Select a load balancer type
1) classic
2) application
3) network
(default is 1): 3
```

Anda juga dapat menentukan tipe penyeimbang beban dengan opsi `--elb-type`.

```
$ eb create test-env --elb-type network
```

Namespace Network Load Balancer

Anda dapat menemukan pengaturan yang terkait dengan Network Load Balancers di namespace berikut:

- `aws:elasticbeanstalk:environment` — Pilih tipe penyeimbang beban untuk lingkungan. Nilai untuk Network Load Balancer adalah `network`.
- `aws:elbv2:listener` — Konfigurasikan pendengar pada Network Load Balancer. Pengaturan ini memetakan ke pengaturan `aws:elb:listener` untuk Classic Load Balancer.
- `aws:elasticbeanstalk:environment:process` — Konfigurasikan pemeriksaan kondisi dan tentukan port dan protokol untuk proses yang berjalan pada instans lingkungan Anda. Pengaturan port dan protokol memetakan ke port instans dan pengaturan protokol instans `aws:elb:listener` untuk pendengar di Classic Load Balancer. Peta pengaturan pemeriksaan kondisi ke pengaturan di namespace `aws:elb:healthcheck` dan `aws:elasticbeanstalk:application`.

Example .ebextensions/ .config network-load-balancer

Untuk memulai dengan Network Load Balancer, gunakan file konfigurasi untuk mengatur tipe penyeimbang beban ke `network`.

```
option_settings:
```

```
  aws:elasticbeanstalk:environment:
    LoadBalancerType: network
```

> **Note**
>
> Anda dapat mengatur tipe penyeimbang beban hanya selama pembuatan lingkungan.

Example .ebextensions/ .config nlb-default-process

File konfigurasi berikut memodifikasi pengaturan pemeriksaan kondisi pada proses default.

```
option_settings:
  aws:elasticbeanstalk:environment:process:default:
    DeregistrationDelay: '20'
    HealthCheckInterval: '10'
    HealthyThresholdCount: '5'
    UnhealthyThresholdCount: '5'
    Port: '80'
    Protocol: TCP
```

Example .ebextensions/ .config nlb-secure-listener

File konfigurasi berikut menambahkan pendengar untuk lalu lintas aman pada port 443 dan pencocokan target proses yang mendengarkan port 443.

```
option_settings:
  aws:elbv2:listener:443:
    DefaultProcess: https
    ListenerEnabled: 'true'
  aws:elasticbeanstalk:environment:process:https:
    Port: '443'
```

Opsi `DefaultProcess` diberi nama demikian karena Application Load Balancers, yang dapat memiliki pendengar non-default pada port yang sama untuk lalu lintas ke jalur tertentu (lihat Application Load Balancer untuk detailnya). Untuk Network Load Balancer, opsi menentukan satu-satunya proses target untuk pendengar ini.

Pada contoh ini, kami menamai proses `https` karena proses ini mendengarkan lalu lintas yang aman (HTTPS). Pendengar mengirimkan lalu lintas ke proses pada port yang ditunjuk menggunakan

protokol TCP, karena Network Load Balancer hanya bekerja dengan TCP. Tidak apa-apa, karena lalu lintas jaringan untuk HTTP dan HTTPS diimplementasikan di atas TCP.

Mengonfigurasi log akses

Anda dapat menggunakan file konfigurasi untuk mengonfigurasi penyeimbang beban lingkungan Anda untuk mengunggah log akses ke bucket Amazon S3. Lihat contoh file konfigurasi berikut GitHub untuk petunjuk:

- `loadbalancer-accesslogs-existingbucket.config` — Konfigurasikan penyeimbang beban untuk mengunggah log akses ke bucket Amazon S3 yang ada.
- `loadbalancer-accesslogs-newbucket.config` — Konfigurasikan penyeimbang beban untuk mengunggah log akses ke bucket baru.

Menambahkan basis data ke lingkungan Elastic Beanstalk Anda

Elastic Beanstalk menyediakan integrasi database yang digabungkan dengan Amazon Relational Database Service (Amazon RDS). Anda dapat menggunakan Elastic Beanstalk untuk menambahkan database MySQL, PostgreSQL, Oracle, atau SQL Server ke lingkungan yang ada atau yang baru saat Anda membuatnya. Saat Anda menambahkan instance database yang digabungkan ke lingkungan, Elastic Beanstalk menyediakan informasi koneksi ke aplikasi Anda. Hal ini dilakukan dengan mengatur properti lingkungan untuk database hostname, port, nama pengguna, password, dan nama database.

Keuntungan menggunakan database yang digabungkan

Jika Anda belum pernah menggunakan instance database dengan aplikasi Anda sebelumnya, sebaiknya Anda terlebih dahulu menggunakan proses yang dijelaskan dalam topik ini untuk menambahkan database ke lingkungan pengujian menggunakan layanan Elastic Beanstalk. Dengan melakukan ini, Anda dapat memverifikasi bahwa aplikasi Anda dapat membaca properti lingkungan, membuat string koneksi, dan terhubung ke instance database, tanpa pekerjaan konfigurasi tambahan yang diperlukan untuk database eksternal untuk Elastic Beanstalk.

Pertimbangan saat akan produksi

Setelah Anda memverifikasi bahwa aplikasi Anda bekerja dengan benar dengan database, Anda dapat mempertimbangkan untuk beralih ke lingkungan produksi. Pada titik ini Anda memiliki opsi untuk memisahkan database dari lingkungan Elastic Beanstalk Anda untuk bergerak menuju konfigurasi yang menawarkan fleksibilitas yang lebih besar. Database yang dipisahkan dapat

tetap beroperasi sebagai instance database Amazon RDS eksternal. Kesehatan lingkungan tidak terpengaruh oleh decoupling database. Jika Anda perlu menghentikan lingkungan, Anda dapat melakukannya dan juga memilih opsi untuk menjaga database tetap tersedia dan beroperasi di luar Elastic Beanstalk.

Keuntungan pindah ke database terpisah

Menggunakan database eksternal memiliki beberapa keunggulan. Anda dapat terhubung ke database eksternal dari beberapa lingkungan, menggunakan tipe database yang tidak didukung dengan database terintegrasi, dan melakukan blue/green penerapan. Sebagai alternatif untuk menggunakan database terpisah yang dibuat Elastic Beanstalk, Anda juga dapat membuat instance database di luar lingkungan Elastic Beanstalk Anda. Kedua opsi menghasilkan instance database yang berada di luar lingkungan Elastic Beanstalk Anda dan akan memerlukan grup keamanan tambahan dan konfigurasi string koneksi. Untuk informasi selengkapnya, lihat Menggunakan Elastic Beanstalk dengan Amazon RDS.

Bagian-bagian

- Siklus hidup basis data
- Menambahkan instans Amazon RDS DB ke lingkungan Anda menggunakan konsol
- Menghubungkan ke basis data
- Mengkonfigurasi instans RDS DB terintegrasi menggunakan konsol
- Mengkonfigurasi instans RDS DB terintegrasi menggunakan file konfigurasi
- Memisahkan instance RDS DB menggunakan konsol
- Memisahkan instance RDS DB menggunakan file konfigurasi

Siklus hidup basis data

Anda dapat memilih apa yang Anda inginkan terjadi pada database setelah Anda memisahkannya dari lingkungan Elastic Beanstalk Anda. Opsi yang dapat Anda pilih secara kolektif disebut sebagai kebijakan penghapusan. Kebijakan penghapusan berikut berlaku untuk database setelah Anda memisahkannya dari lingkungan Elastic Beanstalk atau mengakhiri lingkungan Elastic Beanstalk.

- Snapshot — Sebelum Elastic Beanstalk mengakhiri database, ia menyimpan snapshot dari itu. Anda dapat memulihkan database dari snapshot saat menambahkan instans DB ke lingkungan Elastic Beanstalk atau saat Anda membuat database mandiri. Untuk informasi selengkapnya tentang membuat instans DB mandiri baru dari snapshot, lihat Memulihkan dari snapshot DB di

Panduan Pengguna Amazon RDS. Anda mungkin dikenakan biaya untuk menyimpan snapshot basis data. Untuk informasi lebih lanjut, lihat bagian Penyimpanan Backup dari Harga Amazon RDS.

- Hapus - Elastic Beanstalk mengakhiri database. Setelah dihentikan, instance database tidak lagi tersedia untuk operasi apa pun.
- Pertahankan — Instance database tidak dihentikan. Itu tetap tersedia dan operasional, meskipun dipisahkan dari Elastic Beanstalk. Anda kemudian dapat mengonfigurasi satu atau beberapa lingkungan untuk terhubung ke database sebagai instance database Amazon RDS eksternal. Untuk informasi selengkapnya, lihat Menggunakan Elastic Beanstalk dengan Amazon RDS.

Menambahkan instans Amazon RDS DB ke lingkungan Anda menggunakan konsol

Anda dapat menambahkan instans DB ke lingkungan Anda dengan menggunakan konsol Elastic Beanstalk.

Untuk menambahkan instans DB ke lingkungan Anda

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Di kategori konfigurasi Basis data, pilih Edit.
5. Pilih mesin DB, dan masukkan nama pengguna dan kata sandi.
6. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Anda dapat mengonfigurasi opsi berikut:

- Snapshot – Pilih snapshot basis data yang ada. Elastic Beanstalk mengembalikan snapshot dan menambahkannya ke lingkungan Anda. Nilai defaultnya adalah None. Ketika nilainya Tidak Ada, Anda dapat mengonfigurasi database baru menggunakan pengaturan lain di halaman ini.
- Mesin – Pilih mesin basis data.
- Versi mesin – Pilih versi tertentu dari mesin basis data.
- Kelas instans – Pilih kelas instans DB. Untuk informasi tentang kelas instans DB, lihat https://aws.amazon.com/rds/.

- Penyimpanan – Pilih jumlah penyimpanan yang akan disediakan untuk basis data Anda. Anda dapat meningkatkan penyimpanan yang dialokasikan nanti, tetapi Anda tidak dapat menguranginya. Untuk informasi tentang alokasi penyimpanan, lihat Fitur.
- Nama pengguna — Masukkan nama pengguna pilihan Anda menggunakan kombinasi hanya angka dan huruf.
- Kata Sandi – Masukkan kata sandi pilihan Anda yang berisi 8–16 karakter ASCII yang dapat dicetak (tidak termasuk `/`, `\`, dan `@`).
- Ketersediaan – Pilih Tinggi (Multi-AZ) untuk menjalankan warm backup pada Availability Zone kedua untuk ketersediaan tinggi.
- Kebijakan penghapusan database — Kebijakan penghapusan menentukan apa yang terjadi pada database setelah dipisahkan dari lingkungan Anda. Hal ini dapat diatur ke nilai-nilai berikut:`Create Snapshot`,`Retain`, atau`Delete`. Nilai-nilai ini dijelaskan Siklus hidup basis data dalam topik yang sama ini.

> **Note**
>
> Elastic Beanstalk membuat pengguna utama untuk basis data menggunakan nama pengguna dan kata sandi yang Anda berikan. Untuk mempelajari selengkapnya tentang pengguna utama dan hak istimewanya, lihat Hak Istimewa Akun Pengguna Utama.

Dibutuhkan sekitar 10 menit untuk menambahkan instans DB. Ketika pembaruan selesai, database baru digabungkan ke lingkungan Anda. Nama host dan informasi koneksi lainnya untuk instans DB tersedia untuk aplikasi Anda melalui properti lingkungan berikut.

Nama properti	Deskripsi	Nilai properti
`RDS_HOSTNAME`	Nama host instans DB.	Di tab Konektivitas & keamanan di konsol Amazon RDS: Titik akhir.
`RDS_PORT`	Port tempat instans DB menerima koneksi. Nilai default bervariasi di antara mesin DB.	Di tab Konektivitas & keamanan di konsol Amazon RDS: Port.

Nama properti	Deskripsi	Nilai properti
RDS_DB_NAME	Nama basis data, **ebdb**.	Di tab Konfigurasi di konsol Amazon RDS: Nama DB.
RDS_USERNAME	Nama pengguna yang Anda konfigurasi untuk basis data Anda.	Di tab Konfigurasi di konsol Amazon RDS: Nama pengguna utama.
RDS_PASSWORD	Kata sandi yang Anda konfigurasi untuk basis data Anda.	Tidak tersedia untuk referensi di konsol Amazon RDS.

Menghubungkan ke basis data

Gunakan informasi konektivitas untuk terhubung ke database Anda dari dalam aplikasi Anda melalui variabel lingkungan. Untuk informasi selengkapnya tentang menggunakan Amazon RDS dengan aplikasi Anda, lihat topik berikut.

- Java SE – Menghubungkan ke basis data (platform Java SE)
- Java dengan Tomcat – Menghubungkan ke basis data (platform Tomcat)
- Node.js – Menyambungkan ke basis data
- .NET – Menghubungkan ke basis data
- PHP – Menghubungkan ke database dengan PDO atau My SQLi
- Python – Menghubungkan ke basis data
- Ruby – Menyambungkan ke basis data

Mengkonfigurasi instans RDS DB terintegrasi menggunakan konsol

Anda dapat melihat dan memodifikasi pengaturan konfigurasi untuk instance database Anda di bagian Database pada halaman Konfigurasi lingkungan di konsol Elastic Beanstalk.

Untuk mengonfigurasi instans DB lingkungan Anda pada konsol Elastic Beanstalk

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS

2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Di kategori konfigurasi Basis data, pilih Edit.

Anda dapat mengubah kelas Instance, Storage, Password, Availability, dan pengaturan kebijakan penghapusan Database setelah pembuatan database. Jika Anda mengubah kelas instans, Elastic Beanstalk menyediakan kembali instans DB.

Jika Anda tidak lagi membutuhkan Elastic Beanstalk untuk mengaitkan database dengan lingkungan, Anda dapat memilih untuk memisahkannya dengan memilih database Decouple. Penting untuk memahami opsi dan pertimbangan yang terlibat dengan operasi ini. Untuk informasi selengkapnya, lihat the section called “Memisahkan instance RDS DB menggunakan konsol”.

> **Peringatan**
>
> Jangan mengubah pengaturan pada instance database yang digabungkan di luar fungsionalitas yang disediakan oleh Elastic Beanstalk (misalnya, di konsol Amazon RDS). Jika Anda mengubahnya, konfigurasi DB Amazon RDS Anda mungkin tidak sinkron dengan ketentuan lingkungan Anda. Ketika Anda memperbarui atau me-restart lingkungan Anda, pengaturan yang ditentukan di lingkungan mengganti pengaturan yang Anda buat di luar Elastic Beanstalk.
>
> Jika Anda perlu mengubah pengaturan yang tidak didukung secara langsung oleh Elastic Beanstalk, gunakan file konfigurasi Elastic Beanstalk.

Mengkonfigurasi instans RDS DB terintegrasi menggunakan file konfigurasi

Anda dapat mengonfigurasi instance database lingkungan Anda menggunakan file konfigurasi. Gunakan opsi di namespace `aws:rds:dbinstance`. Contoh berikut memodifikasi ukuran penyimpanan basis data yang dialokasikan menjadi 100 GB.

Example .ebextensions/ .config db-instance-options

```
option_settings:
  aws:rds:dbinstance:
    DBAllocatedStorage: 100
```

Jika Anda ingin mengonfigurasi properti instans DB yang tidak didukung Elastic Beanstalk, Anda masih dapat menggunakan file konfigurasi, dan menentukan pengaturan Anda menggunakan kunci. `resources` Contoh berikut menetapkan nilai untuk `StorageType` dan `Iops` properti Amazon RDS.

Example .ebextensions/ .config db-instance-properties

```
Resources:
  AWSEBRDSDatabase:
    Type: AWS::RDS::DBInstance
    Properties:
      StorageType:io1
      Iops: 1000
```

Memisahkan instance RDS DB menggunakan konsol

Anda dapat memisahkan database Anda dari lingkungan Elastic Beanstalk tanpa mempengaruhi kesehatan lingkungan. Pertimbangkan persyaratan berikut sebelum Anda memisahkan database:

- Apa yang harus terjadi pada database setelah dipisahkan?

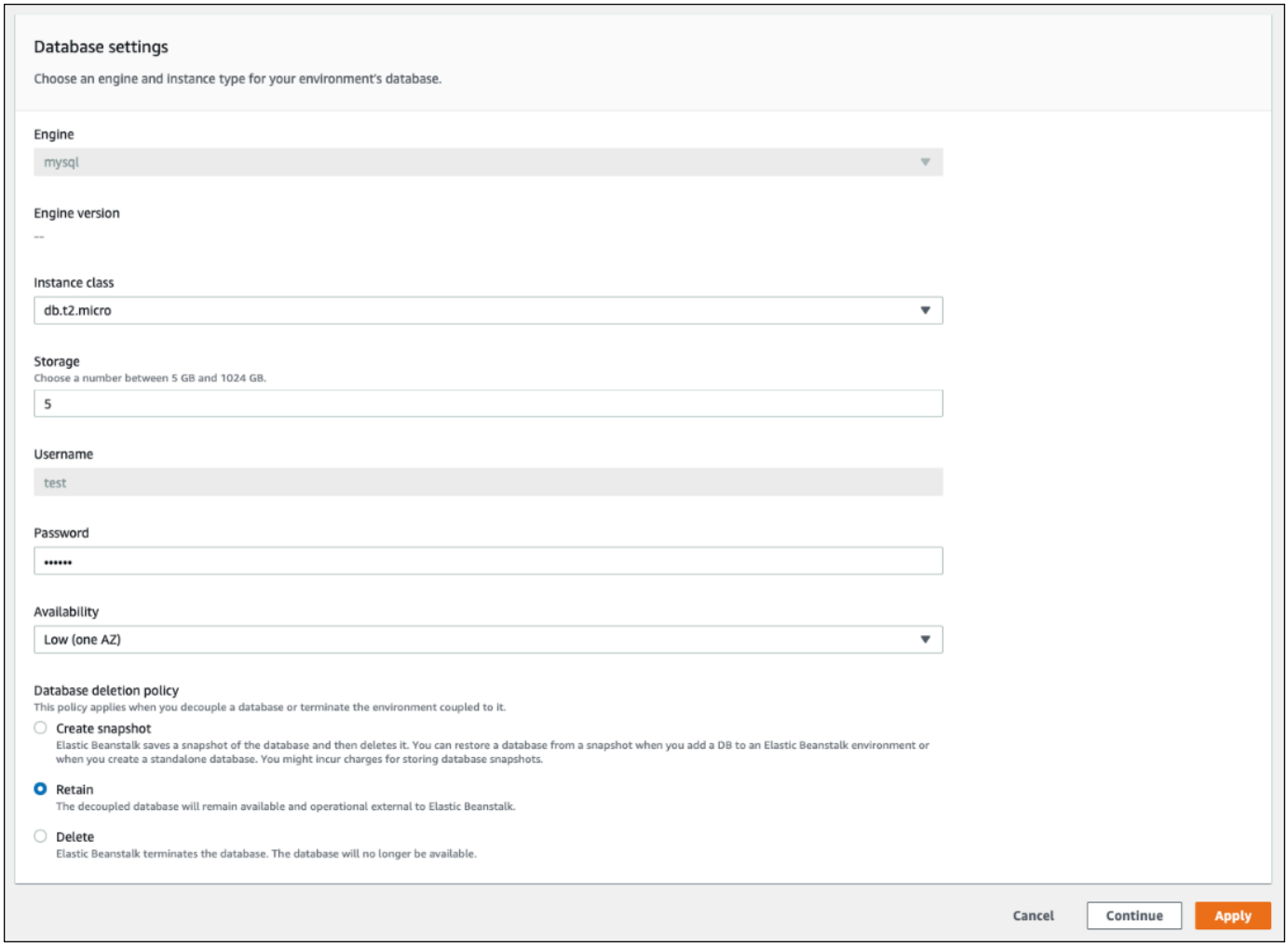
  Anda dapat memilih untuk membuat snapshot dari database dan kemudian menghentikannya, mempertahankan operasional database sebagai database mandiri di luar Elastic Beanstalk, atau menghapus database secara permanen. Setelan kebijakan penghapusan Database menentukan hasil ini. Untuk penjelasan rinci tentang kebijakan penghapusan, lihat Siklus hidup basis data di topik yang sama ini.
- Apakah Anda perlu membuat perubahan pada pengaturan konfigurasi database sebelum memisahkannya?

  Jika Anda perlu membuat perubahan konfigurasi apa pun pada database, Anda harus menerapkannya sebelum memisahkan database. Ini termasuk perubahan pada kebijakan penghapusan Database. Setiap perubahan yang tertunda yang dikirimkan bersamaan dengan pengaturan database Decouple akan diabaikan, sementara hanya pengaturan decouple yang diterapkan.

Untuk memisahkan instance DB dari lingkungan

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS

2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Di kategori konfigurasi Basis data, pilih Edit.
5. Tinjau semua nilai konfigurasi di bagian Setelan database, terutama kebijakan penghapusan Database, yang menentukan apa yang terjadi pada database setelah dipisahkan.



Jika semua pengaturan konfigurasi lainnya sudah benar, lewati ke Langkah 6 untuk memisahkan database.

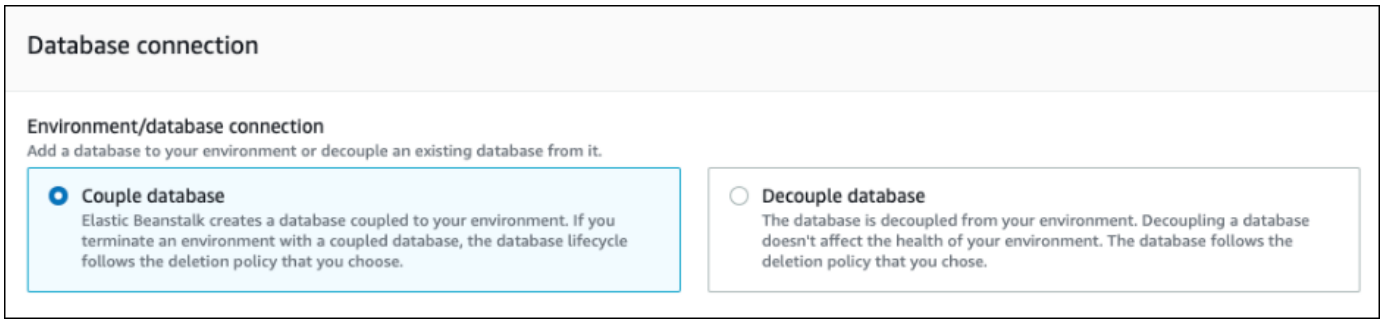
> **Warning**
>
> Penting untuk menerapkan pengaturan kebijakan penghapusan Database secara terpisah dari database Decouple. Jika Anda memilih Terapkan dengan maksud untuk menyimpan database Decouple dan kebijakan penghapusan Database yang baru dipilih,

kebijakan penghapusan baru yang Anda pilih akan diabaikan. Elastic Beanstalk akan memisahkan database mengikuti kebijakan penghapusan yang ditetapkan sebelumnya. Jika kebijakan penghapusan yang ditetapkan sebelumnya adalah `Delete` atau`Create Snapshot`, Anda berisiko kehilangan database alih-alih mengikuti kebijakan tertunda yang dimaksud.

Jika salah satu pengaturan konfigurasi memerlukan pembaruan lakukan hal berikut:

1. Buat modifikasi yang diperlukan di panel pengaturan Database.
2. Pilih Terapkan. Ini akan memakan waktu beberapa menit untuk menyimpan perubahan konfigurasi untuk database Anda.
3. Kembali ke Langkah 3 dan pilih Konfigurasi dari panel navigasi.

6. Pergi ke bagian koneksi database panel.



7. Pilih database Decouple.
8. Pilih Terapkan untuk memulai operasi decoupling database.

Setelan kebijakan penghapusan menentukan hasil untuk database dan lamanya waktu yang diperlukan untuk memisahkan database.

- Jika kebijakan penghapusan disetel ke`Delete`, database akan dihapus. Operasi dapat memakan waktu sekitar 10-20 menit, tergantung pada ukuran database.
- Jika kebijakan penghapusan disetel ke`Snapshot`, snapshot database akan dibuat. Kemudian, database dihapus. Lamanya waktu yang dibutuhkan untuk proses ini bervariasi sesuai dengan ukuran database.
- Jika kebijakan penghapusan disetel ke`Retain`, database tetap beroperasi di luar lingkungan Elastic Beanstalk. Biasanya dibutuhkan waktu kurang dari lima menit untuk memisahkan database.

Jika Anda memutuskan untuk mempertahankan database di luar lingkungan Elastic Beanstalk Anda, Anda harus mengambil langkah tambahan untuk mengonfigurasinya. Untuk informasi selengkapnya, lihat Menggunakan Elastic Beanstalk dengan Amazon RDS. Jika Anda berencana untuk menggunakan database yang Anda pisahkan untuk lingkungan produksi, verifikasi jenis penyimpanan yang digunakan database sesuai dengan beban kerja Anda. Untuk informasi selengkapnya, lihat Penyimpanan Instans DB dan Memodifikasi instans DB di Panduan Pengguna Amazon RDS.

Memisahkan instance RDS DB menggunakan file konfigurasi

Anda dapat memisahkan instans DB Anda dari lingkungan Elastic Beanstalk tanpa mempengaruhi kesehatan lingkungan. Instance database mengikuti kebijakan penghapusan database yang diterapkan ketika database dipisahkan.

Kedua opsi yang diperlukan untuk memisahkan database ada di namespace. the section called "aws:rds:dbinstance" Mereka adalah sebagai berikut:

- `DBDeletionPolicy`Opsi menetapkan kebijakan penghapusan. Hal ini dapat diatur ke nilai-nilai berikut:`Snapshot`,`Delete`, atau`Retain`. Nilai-nilai ini dijelaskan Siklus hidup basis data dalam topik yang sama ini.
- `HasCoupledDatabase`Opsi menentukan apakah lingkungan Anda memiliki database yang digabungkan.
  - Jika diaktifkan`true`, Elastic Beanstalk membuat instans DB baru yang digabungkan ke lingkungan Anda.
  - Jika diaktifkan`false`, Elastic Beanstalk mulai memisahkan instans DB dari lingkungan Anda.

Jika Anda ingin mengubah konfigurasi database Anda sebelum Anda memisahkannya, terapkan perubahan konfigurasi apa pun terlebih dahulu, dalam operasi terpisah. Ini termasuk mengubah `DBDeletionPolicy` konfigurasi. Setelah perubahan Anda diterapkan, jalankan perintah terpisah untuk mengatur opsi decoupling. Jika Anda mengirimkan pengaturan konfigurasi lain dan pengaturan decouple pada saat yang sama, pengaturan opsi konfigurasi lainnya diabaikan saat pengaturan decouple diterapkan.

> **Warning**
>
> Penting bagi Anda untuk menjalankan perintah untuk menerapkan `DBDeletionPolicy` dan `HasCoupledDatabase` pengaturan sebagai dua operasi terpisah. Jika kebijakan

penghapusan aktif sudah disetel ke `Delete` atau`Snapshot`, Anda berisiko kehilangan database. Basis data mengikuti kebijakan penghapusan yang saat ini aktif, bukan kebijakan penghapusan tertunda yang Anda inginkan.

Untuk memisahkan instance DB dari lingkungan

Ikuti langkah-langkah ini untuk memisahkan database dari lingkungan Elastic Beanstalk Anda. Anda dapat menggunakan EB CLI atau untuk menyelesaikan AWS CLI langkah-langkahnya. Untuk informasi selengkapnya, lihat Kustomisasi lingkungan lanjutan dengan file konfigurasi.

1. Jika Anda ingin mengubah kebijakan penghapusan, siapkan file konfigurasi dalam format berikut. Dalam contoh ini, kebijakan penghapusan diatur untuk mempertahankan.

   Example

   ```
   option_settings:
     aws:rds:dbinstance:
       DBDeletionPolicy: Retain
   ```

2. Jalankan perintah menggunakan alat pilihan Anda untuk menyelesaikan pembaruan konfigurasi.
3. Siapkan file konfigurasi untuk disetel `HasCoupledDatabase` ke`false`.

   Example

   ```
   option_settings:
     aws:rds:dbinstance:
       HasCoupledDatabase: false
   ```

4. Jalankan perintah menggunakan alat pilihan Anda untuk menyelesaikan pembaruan konfigurasi.

Setelan kebijakan penghapusan menentukan hasil untuk database dan lamanya waktu yang diperlukan untuk memisahkan database.

- Jika kebijakan penghapusan disetel ke`Delete`, database akan dihapus. Operasi dapat memakan waktu sekitar 10-20 menit, tergantung pada ukuran database.
- Jika kebijakan penghapusan disetel ke`Snapshot`, snapshot database akan dibuat. Kemudian, database dihapus. Lamanya waktu yang dibutuhkan untuk proses ini bervariasi sesuai dengan ukuran database.

- Jika kebijakan penghapusan disetel ke`Retain`, database tetap beroperasi di luar lingkungan Elastic Beanstalk. Biasanya dibutuhkan waktu kurang dari lima menit untuk memisahkan database.

Jika Anda memutuskan untuk mempertahankan database di luar lingkungan Elastic Beanstalk Anda, Anda harus mengambil langkah tambahan untuk mengonfigurasinya. Untuk informasi selengkapnya, lihat Menggunakan Elastic Beanstalk dengan Amazon RDS. Jika Anda berencana untuk menggunakan database yang Anda pisahkan untuk lingkungan produksi, verifikasi jenis penyimpanan yang digunakan database sesuai dengan beban kerja Anda. Untuk informasi selengkapnya, lihat Penyimpanan Instans DB dan Memodifikasi instans DB di Panduan Pengguna Amazon RDS.

Keamanan AWS Elastic Beanstalk lingkungan Anda

Elastic Beanstalk menyediakan beberapa opsi yang mengontrol akses layanan (keamanan) lingkungan Anda dan instance Amazon di EC2 dalamnya. Topik ini membahas konfigurasi opsi ini.

Bagian

- Mengonfigurasi keamanan lingkungan Anda
- Namespace konfigurasi keamanan lingkungan

Mengonfigurasi keamanan lingkungan Anda

Anda dapat mengubah konfigurasi keamanan lingkungan Elastic Beanstalk Anda di konsol Elastic Beanstalk.

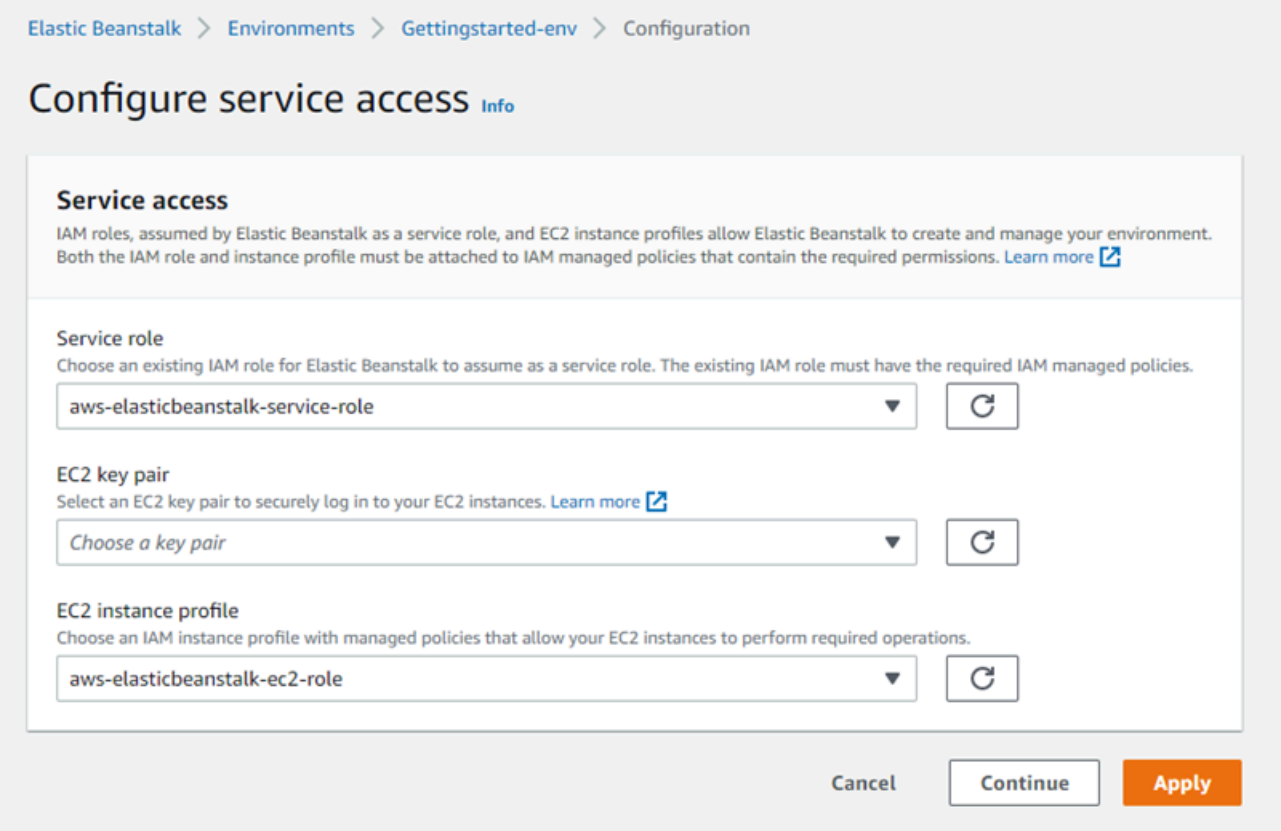
Untuk mengonfigurasi akses layanan lingkungan (keamanan) di konsol Elastic Beanstalk

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Dalam kategori konfigurasi akses layanan, pilih Edit.

Pengaturan berikut tersedia.

Pengaturan

- Peran layanan
- EC2 key pair
- Profil instans IAM



Peran layanan

Pilih peran layanan untuk mengasosiasikan dengan lingkungan Elastic Beanstalk Anda. Elastic Beanstalk mengasumsikan peran layanan saat mengakses layanan lain atas nama Anda. AWS Lihat perinciannya di Mengelola peran layanan Elastic Beanstalk.

EC2 key pair

Anda dapat masuk dengan aman ke instans Amazon Elastic Compute Cloud EC2 (Amazon) yang disediakan untuk aplikasi Elastic Beanstalk Anda dengan key pair Amazon. EC2 Untuk petunjuk cara membuat key pair, lihat Membuat Pasangan Kunci Menggunakan Amazon EC2 di Panduan EC2 Pengguna Amazon.

> **Note**
>
> Saat Anda membuat key pair, Amazon EC2 menyimpan salinan kunci publik Anda. Jika Anda tidak perlu lagi menggunakannya untuk terhubung ke instance lingkungan apa pun, Anda dapat menghapusnya dari Amazon EC2. Untuk detailnya, lihat Menghapus Pasangan Kunci Anda di Panduan EC2 Pengguna Amazon.

Pilih EC2 key pair dari menu drop-down untuk menetapkannya ke instance lingkungan Anda. Ketika Anda menetapkan pasangan kunci, kunci publik disimpan di instans untuk mengautentikasi kunci privat, yang Anda simpan secara lokal. Kunci pribadi tidak pernah disimpan AWS.

Untuk informasi selengkapnya tentang menghubungkan ke EC2 instans Amazon, lihat Connect to Your Instance dan Connecting to Linux/UNIX Instans dari Windows menggunakan PuTTY di Panduan Pengguna Amazon. EC2

Profil instans IAM

Profil EC2 instans adalah peran IAM yang diterapkan pada instans yang diluncurkan di lingkungan Elastic Beanstalk Anda. EC2 Instans Amazon mengasumsikan peran profil instans untuk menandatangani permintaan AWS dan mengakses APIs, misalnya, untuk mengunggah log ke Amazon S3.

Pertama kali Anda membuat lingkungan di konsol Elastic Beanstalk, Elastic Beanstalk meminta Anda untuk membuat profil instans dengan set izin default. Anda dapat menambahkan izin ke profil ini untuk memberikan akses instans Anda ke layanan lain AWS . Lihat perinciannya di Mengelola profil instans Elastic Beanstalk.

> **Note**
>
> Sebelumnya Elastic Beanstalk membuat `aws-elasticbeanstalk-ec2-role` profil instance default EC2 bernama pertama AWS kali akun membuat lingkungan. Profil instance ini menyertakan kebijakan terkelola default. Jika akun Anda sudah memiliki profil instans ini, itu akan tetap tersedia bagi Anda untuk menetapkan ke lingkungan Anda.
> Namun, pedoman AWS keamanan terbaru tidak mengizinkan AWS layanan untuk secara otomatis membuat peran dengan kebijakan kepercayaan ke AWS layanan lain, EC2 dalam hal ini. Karena pedoman keamanan ini, Elastic Beanstalk tidak lagi membuat profil instance default. `aws-elasticbeanstalk-ec2-role`

> **Note**
>
> Ada aspek lain dari keamanan EC2 instance yang menunjuk aturan firewall untuk EC2 instance. Ini dikendalikan oleh kelompok EC2 keamanan. Untuk informasi selengkapnya, lihat EC2 Contoh Amazon untuk lingkungan Elastic Beanstalk Anda.

Namespace konfigurasi keamanan lingkungan

Elastic Beanstalk menyediakan opsi konfigurasi pada ruang nama berikut untuk memungkinkan Anda menyesuaikan keamanan lingkungan Anda:

- `aws:elasticbeanstalk:environment` — Konfigurasikan peran layanan lingkungan menggunakan `ServiceRole` opsi tersebut.
- `aws:autoscaling:launchconfiguration`— Konfigurasikan izin untuk EC2 instans Amazon lingkungan menggunakan`EC2KeyName`,, `IamInstanceProfile`,`DisableDefaultEC2SecurityGroup`, dan `SecurityGroups` opsi.

Konsol EB CLI dan Elastic Beanstalk menerapkan nilai yang direkomendasikan untuk pilihan sebelumnya. Anda harus menghapus pengaturan ini jika Anda ingin menggunakan file konfigurasi untuk mengonfigurasi hal yang sama. Lihat Nilai yang disarankan untuk rincian selengkapnya.

Menandai sumber daya di lingkungan Elastic Beanstalk Anda

Anda dapat menerapkan tag ke AWS Elastic Beanstalk lingkungan Anda. Tag adalah pasangan nilai kunci yang terkait dengan AWS sumber daya. Untuk informasi tentang penandaan sumber daya Elastic Beanstalk, kasus penggunaan, kunci tag dan batasan nilai, dan jenis sumber daya yang didukung, lihat Pelabelan sumber daya aplikasi Elastic Beanstalk.

Elastic Beanstalk menerapkan tag lingkungan ke sumber daya lingkungan itu sendiri, serta AWS sumber daya lain yang diciptakan Elastic Beanstalk untuk lingkungan. Anda dapat menggunakan tag untuk mengelola izin pada tingkat sumber daya tertentu dalam lingkungan. Untuk informasi selengkapnya, lihat Menandai EC2 Sumber Daya Amazon Anda di Panduan EC2 Pengguna Amazon.

Secara default, Elastic Beanstalk menerapkan beberapa tag ke lingkungan Anda:

- `elasticbeanstalk:environment-name` — Nama lingkungan.
- `elasticbeanstalk:environment-id` — ID lingkungan.

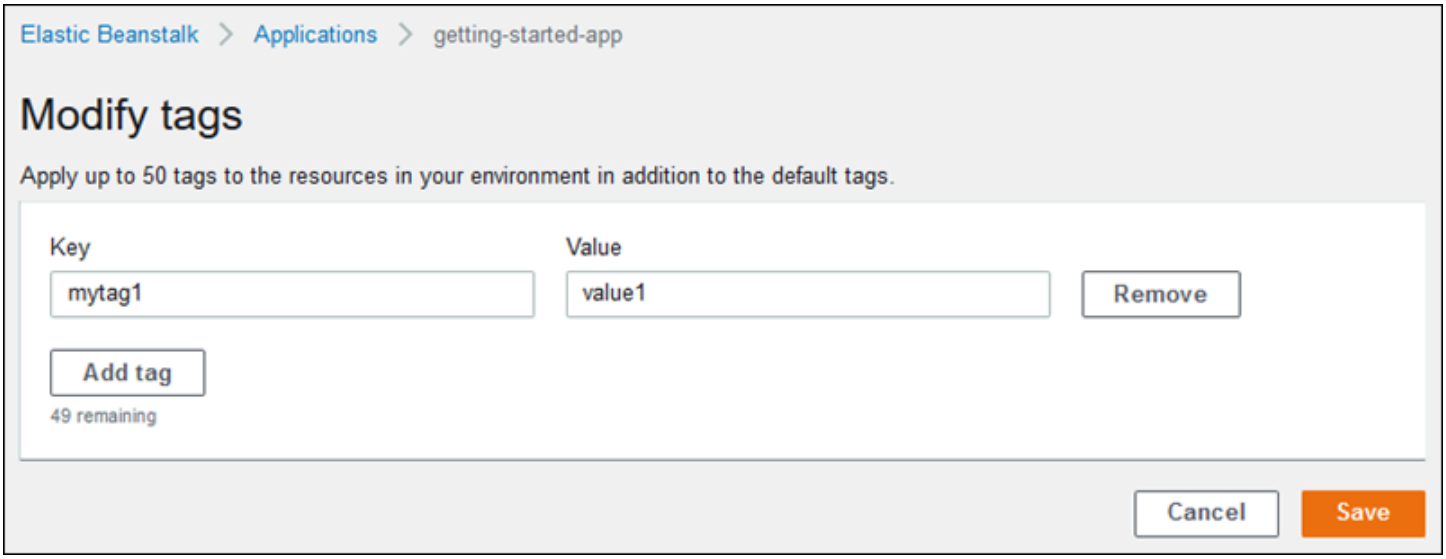
- `Name`— Juga nama lingkungan. `Name`digunakan di EC2 dasbor Amazon untuk mengidentifikasi dan mengurutkan sumber daya.

Anda tidak dapat mengedit tag default ini.

Anda dapat menentukan tag saat Anda membuat lingkungan Elastic Beanstalk. Di lingkungan yang sudah ada, Anda dapat menambahkan atau menghapus tag, dan memperbarui nilai-nilai tag yang ada. Lingkungan dapat memiliki hingga 50 tag termasuk tag default.

Menambahkan tag selama pembuatan lingkungan

Saat Anda menggunakan konsol Elastic Beanstalk untuk membuat lingkungan, Anda dapat menentukan kunci dan nilai tag pada halaman konfigurasi Ubah tag dari Wizard buat Lingkungan Baru.



Jika Anda menggunakan EB CLI untuk membuat lingkungan, gunakan opsi `--tags` dengan eb create untuk menambahkan tag.

```
~/workspace/my-app$ eb create --tags mytag1=value1,mytag2=value2
```

Dengan AWS CLI atau klien berbasis API lainnya, gunakan `--tags` parameter pada perintah. create-environment

```
$ aws elasticbeanstalk create-environment \
      --tags Key=mytag1,Value=value1 Key=mytag2,Value=value2 \
```

```
      --application-name my-app --environment-name my-env --cname-prefix my-app --
version-label v1 --template-name my-saved-config
```

Konfigurasi tersimpan menyertakan tag yang ditentukan pengguna. Ketika Anda menerapkan konfigurasi tersimpan yang berisi tag selama pembuatan lingkungan, tag tersebut diterapkan ke lingkungan baru, selama Anda tidak menentukan tag baru. Jika Anda menambahkan tag ke lingkungan menggunakan salah satu metode sebelumnya, setiap tag yang ditentukan dalam konfigurasi tersimpan akan dibuang.

Mengelola tag dari lingkungan yang ada

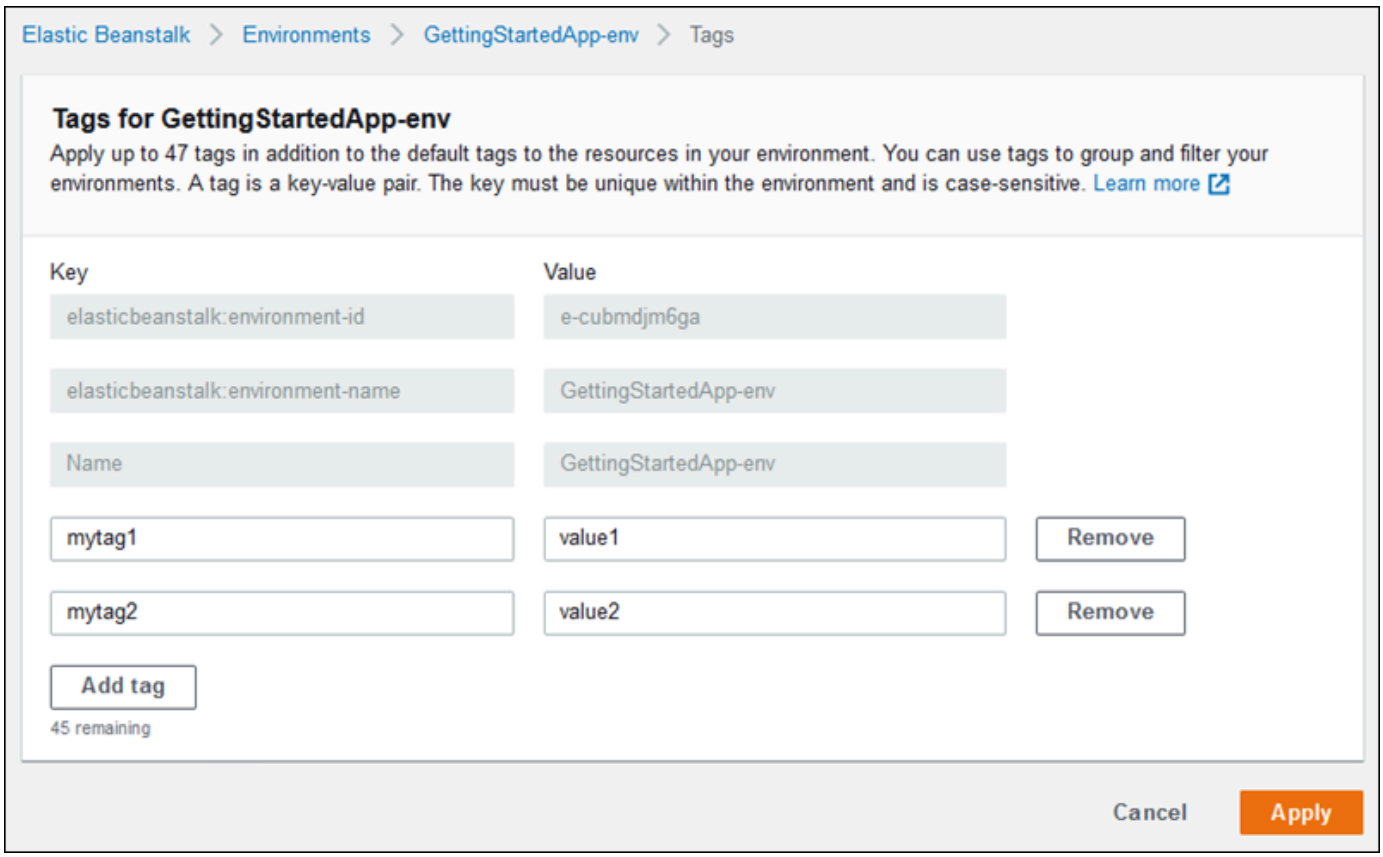
Anda dapat menambahkan, memperbarui, dan menghapus tag di lingkungan Elastic Beanstalk yang sudah ada. Elastic Beanstalk menerapkan perubahan pada sumber daya lingkungan Anda.

Namun, Anda tidak dapat mengedit tag default yang diterapkan Elastic Beanstalk ke lingkungan Anda.

Untuk mengelola tag lingkungan di konsol Elastic Beanstalk

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Tag.

   Halaman manajemen tag menunjukkan daftar tag yang saat ini ada di lingkungan.



4. Tambahkan, perbarui, atau hapus tag:

   - Untuk menambahkan tag, masukkan tag ke dalam kotak kosong di bagian bawah daftar. Untuk menambahkan tag lain, pilih Tambahkan tag dan Elastic Beanstalk menambahkan sepasang kotak kosong.
   - Untuk memperbarui kunci atau nilai tag, edit masing-masing kotak pada baris tag.
   - Untuk menghapus tag, pilih Hapus di samping tag yang akan dihapus.

5. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Jika Anda menggunakan EB CLI untuk memperbarui lingkungan Anda, gunakan eb tags untuk menambahkan, memperbarui, menghapus, atau mencantumkan tag.

Misalnya, perintah berikut mencantumkan tag di lingkungan default Anda.

```
~/workspace/my-app$ eb tags --list
```

Perintah berikut memperbarui tag `mytag1` dan menghapus tag `mytag2`.

```
~/workspace/my-app$ eb tags --update mytag1=newvalue --delete mytag2
```

Untuk daftar lengkap opsi dan contoh lainnya, lihat `eb tags`.

Dengan AWS CLI atau klien berbasis API lainnya, gunakan list-tags-for-resource perintah untuk membuat daftar tag lingkungan.

```
$ aws elasticbeanstalk list-tags-for-resource --resource-arn
 "arn:aws:elasticbeanstalk:us-east-2:my-account-id:environment/my-app/my-env"
```

Gunakan perintah update-tags-for-resource untuk menambahkan, memperbarui, atau menghapus tag di lingkungan.

```
$ aws elasticbeanstalk update-tags-for-resource \
      --tags-to-add Key=mytag1,Value=newvalue --tags-to-remove mytag2 \
      --resource-arn "arn:aws:elasticbeanstalk:us-east-2:my-account-id:environment/my-
app/my-env"
```

Tentukan kedua tag untuk ditambahkan dan tag untuk diperbarui dalam parameter `--tags-to-add` dari update-tags-for-resource. Tag yang tidak ada ditambahkan, dan nilai tag yang ada diperbarui.

> **Note**
>
> Untuk menggunakan dua AWS CLI perintah ini dengan lingkungan Elastic Beanstalk, Anda memerlukan ARN lingkungan. Anda dapat mengambil ARN dengan menggunakan perintah berikut.
>
> ```
> $ aws elasticbeanstalk describe-environments
> ```

Variabel lingkungan dan pengaturan perangkat lunak lainnya

Halaman konfigurasi konfigurasi pembaruan, pemantauan, dan pencatatan memungkinkan Anda mengonfigurasi perangkat lunak pada instans Amazon Elastic Compute Cloud (Amazon EC2) yang menjalankan aplikasi Anda. Anda dapat mengonfigurasi variabel lingkungan, AWS X-Ray debugging, penyimpanan dan streaming log instance, dan pengaturan khusus platform.

Topik

- Mengonfigurasi pengaturan spesifik platform
- Mengkonfigurasi properti lingkungan (variabel lingkungan)
- Namespace pengaturan perangkat lunak
- Mengakses properti lingkungan
- Mengkonfigurasi debugging AWS X-Ray
- Melihat log lingkungan Elastic Beanstalk Anda

Mengonfigurasi pengaturan spesifik platform

Selain serangkaian opsi standar yang tersedia untuk semua lingkungan, sebagian besar platform Elastic Beanstalk memungkinkan Anda menentukan pengaturan khusus bahasa atau kerangka kerja. Ini muncul di bagian perangkat lunak Platform pada halaman Konfigurasi pembaruan, pemantauan, dan pencatatan, dan dapat mengambil formulir berikut.

- Properti lingkungan yang telah ditetapkan — platform Ruby menggunakan properti lingkungan untuk pengaturan kerangka kerja, seperti `RACK_ENV` dan `BUNDLE_WITHOUT`.
- Properti lingkungan placeholder — platform Tomcat menetapkan properti lingkungan bernama `JDBC_CONNECTION_STRING` yang tidak diatur ke nilai apa pun. Jenis pengaturan ini lebih umum pada versi platform yang lebih lama.
- Opsi konfigurasi — Sebagian besar platform menetapkan opsi konfigurasi di namespace spesifik platform atau bersama, seperti `aws:elasticbeanstalk:xray` atau `aws:elasticbeanstalk:container:python`.

Untuk mengonfigurasi pengaturan spesifik platform di konsol Elastic Beanstalk

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Dalam kategori konfigurasi Pembaruan, pemantauan, dan pencatatan, pilih Edit.
5. Di bawah perangkat lunak Platform, buat perubahan pengaturan opsi yang diperlukan.
6. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Untuk informasi tentang opsi spesifik platform, dan tentang mendapatkan nilai properti lingkungan pada kode Anda, lihat topik platform untuk bahasa atau kerangka kerja Anda:

- Docker — the section called “Konfigurasi lingkungan”
- Go — Menggunakan platform Go Elastic Beanstalk
- Java SE — Menggunakan platform Java SE Elastic Beanstalk
- Tomcat — Menggunakan platform Elastic Beanstalk Tomcat
- .NET Core on Linux — Menggunakan inti Elastic Beanstalk .NET pada platform Linux
- .NET – Menggunakan platform Elastic Beanstalk .NET Windows
- Node.js – Menggunakan platform Node.js Elastic Beanstalk
- PHP — Menggunakan platform PHP Elastic Beanstalk
- Python – Menggunakan platform Python Elastic Beanstalk
- Ruby – Menggunakan platform Ruby Elastic Beanstalk

Mengkonfigurasi properti lingkungan (variabel lingkungan)

Anda dapat menggunakan properti lingkungan, (juga dikenal sebagai variabel lingkungan), untuk meneruskan titik akhir, pengaturan debug, dan informasi lainnya ke aplikasi Anda. Variabel lingkungan membantu Anda menjalankan aplikasi di beberapa lingkungan untuk tujuan yang berbeda, seperti pengembangan, pengujian, pementasan, dan produksi.

Selain itu, ketika Anda menambahkan database ke lingkungan Anda, Elastic Beanstalk menetapkan variabel lingkungan, `RDS_HOSTNAME` seperti, yang dapat Anda baca dalam kode aplikasi Anda untuk membangun objek koneksi atau string.

Untuk mengonfigurasi variabel lingkungan di konsol Elastic Beanstalk

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Dalam kategori konfigurasi Pembaruan, pemantauan, dan pencatatan, pilih Edit.
5. Gulir ke bawah ke variabel lingkungan Runtime.
6. Pilih Tambahkan variabel lingkungan.

7. Untuk Sumber pilih Teks biasa.

> **Note**
>
> Nilai Secrets Manager dan SSM Parameter Store di drop-down adalah untuk mengonfigurasi variabel lingkungan sebagai rahasia untuk menyimpan data sensitif, seperti kredensial dan kunci API. Untuk informasi selengkapnya, lihat Menggunakan Elastic Beanstalk dengan dan AWS Secrets Manager Parameter Store AWS Systems Manager.

8. Masukkan nama variabel Lingkungan dan pasangan nilai variabel Lingkungan.
9. Jika Anda perlu menambahkan lebih banyak variabel ulangi Langkah 6 hingga Langkah 8.
10. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Batas properti lingkungan

- Kunci dapat berisi karakter alfanumerik dan simbol-simbol berikut: `_ . : / + \ - @`

  Simbol yang tercantum berlaku untuk kunci properti lingkungan, tetapi mungkin tidak valid untuk nama variabel lingkungan pada platform lingkungan Anda. Untuk kompatibilitas dengan semua platform, batasi properti lingkungan untuk pola berikut: `[A-Z_][A-Z0-9_]*`
- Nilai dapat berisi karakter alfanumerik, spasi, dan simbol-simbol berikut: `_ . : / = + \ - @ ' "`

> **Note**
>
> Beberapa karakter dalam nilai properti lingkungan harus diloloskan. Gunakan karakter garis miring terbalik (\) untuk mewakili beberapa karakter khusus dan karakter kontrol. Daftar berikut mencakup contoh untuk mewakili beberapa karakter yang perlu diloloskan:
>
> - backslash (\) - untuk mewakili penggunaan `\\`
> - kutipan tunggal (') - untuk mewakili penggunaan `\'`
> - kutipan ganda (") — untuk mewakili penggunaan `\"`

- Kunci dan nilai peka huruf besar dan kecil.
- Ukuran gabungan dari semua properti lingkungan tidak dapat melebihi 4.096 byte saat disimpan sebagai string dengan format =. *key value*

Namespace pengaturan perangkat lunak

Anda dapat menggunakan file konfigurasi untuk menetapkan opsi konfigurasi dan melakukan tugas konfigurasi instans lain selama penerapan. Opsi konfigurasi dapat berupa platform khusus atau berlaku untuk semua platform di layanan Elastic Beanstalk secara keseluruhan. Opsi konfigurasi diatur ke dalam ruang nama.

Anda bisa menggunakan file konfigurasi Elastic Beanstalk untuk menetapkan properti lingkungan dan opsi konfigurasi dalam kode sumber Anda. Gunakan `aws:elasticbeanstalk:application:environment` namespace untuk menentukan properti lingkungan.

Example .ebextensions/options.config

```
option_settings:
  aws:elasticbeanstalk:application:environment:
    API_ENDPOINT: www.example.com/api
```

Jika Anda menggunakan file konfigurasi atau AWS CloudFormation templat untuk membuat sumber daya khusus, Anda dapat menggunakan AWS CloudFormation fungsi untuk mendapatkan informasi tentang sumber daya dan menetapkannya ke properti lingkungan secara dinamis selama penerapan. Contoh berikut dari elastic-beanstalk-samples GitHub repositori menggunakan fungsi Ref untuk mendapatkan ARN dari topik Amazon SNS yang dibuatnya, dan menetapkannya ke properti lingkungan bernama. `NOTIFICATION_TOPIC`

> **Catatan**
>
> - Jika Anda menggunakan AWS CloudFormation fungsi untuk menentukan properti lingkungan, konsol Elastic Beanstalk menampilkan nilai properti sebelum fungsi dievaluasi. Anda dapat menggunakan `get-config` skrip platform untuk mengonfirmasi nilai properti lingkungan yang tersedia untuk aplikasi Anda.
> - Platform Multicontainer Docker tidak digunakan AWS CloudFormation untuk membuat sumber daya kontainer. Akibatnya, platform ini tidak mendukung mendefinisikan properti lingkungan menggunakan AWS CloudFormation fungsi.

Example .Ebextensions/sns-topic.config

```
Resources:
  NotificationTopic:
    Type: AWS::SNS::Topic

option_settings:
  aws:elasticbeanstalk:application:environment:
    NOTIFICATION_TOPIC: ``{"Ref" : "NotificationTopic"}``
```

Anda juga dapat menggunakan fitur ini untuk menyebarkan informasi dari parameter pseudo AWS CloudFormation. Contoh ini mendapat wilayah saat ini dan menetapkannya ke properti bernama `AWS_REGION`.

Example .Ebextensions/idv-regionname.config

```
option_settings:
  aws:elasticbeanstalk:application:environment:
    AWS_REGION: ``{"Ref" : "AWS::Region"}``
```

Sebagian besar platform Elastic Beanstalk menentukan namespace tambahan dengan opsi untuk mengonfigurasi perangkat lunak yang berjalan pada instans, seperti proksi terbalik yang menyampaikan permintaan untuk aplikasi Anda. Untuk informasi selengkapnya tentang namespace yang tersedia untuk platform Anda, lihat berikut ini:

- Pergi — Namespace konfigurasi Go
- Java SE — Namespace konfigurasi Java SE
- Tomcat — Namespace konfigurasi Tomcat
- .NET Core on Linux — .NET Core di namespace konfigurasi Linux
- .NET – Namespace aws:elasticbeanstalk:container:dotnet:apppool
- Node.js – Node.jskonfigurasi namespace
- PHP — Ruang nama untuk konfigurasi
- Python – Namespace konfigurasi Python
- Ruby – Namespace konfigurasi Ruby

Elastic Beanstalk memberikan banyak opsi konfigurasi untuk menyesuaikan lingkungan Anda. Selain file konfigurasi, Anda juga dapat mengatur opsi konfigurasi menggunakan konsol tersebut, konfigurasi tersimpan, EB CLI, atau AWS CLI. Lihat Opsi konfigurasi untuk informasi selengkapnya.

Mengakses properti lingkungan

Pada sebagian besar kasus, Anda mengakses properti lingkungan dalam kode aplikasi Anda seperti variabel lingkungan. Secara umum, bagaimanapun, properti lingkungan diteruskan hanya ke aplikasi dan tidak dapat dilihat dengan menghubungkan sebuah instans di lingkungan Anda dan menjalankan `env`.

- Go — `os.Getenv`

  ```
  endpoint := os.Getenv("API_ENDPOINT")
  ```

- Java SE — `System.getenv`

  ```
  String endpoint = System.getenv("API_ENDPOINT");
  ```

- Tomcat — `System.getProperty` dan `System.getenv`

  Versi platform Tomcat yang dirilis pada atau setelah 26 Maret 2025, juga dapat digunakan `System.getenv` untuk mengakses variabel lingkungan teks biasa. Anda dapat terus menggunakan `System.getProperty` untuk mengakses variabel lingkungan plaintext. Namun, variabel lingkungan yang disimpan sebagai rahasia hanya tersedia menggunakan`System.getenv`.

  ```
  String endpoint = System.getProperty("API_ENDPOINT");
  ```

  ```
  String endpoint = System.getenv("API_ENDPOINT");
  ```

  Important

  Penambahan `System.getenv` akses untuk variabel lingkungan dalam versi platform Tomcat yang dirilis pada atau setelah 26 Maret 2025 dapat menyebabkan perilaku tak terduga dalam aplikasi yang memberikan prioritas variabel lingkungan daripada properti sistem Java atau ketika secara eksplisit beralih dari ke. `System.getProperty`

`System.getenv` Untuk informasi lebih lanjut dan tindakan yang disarankan, lihat Menggunakan platform Elastic Beanstalk Tomcat.

- .NET Core di Linux — `Environment.GetEnvironmentVariable`

```
string endpoint = Environment.GetEnvironmentVariable("API_ENDPOINT");
```

- .NET – `appConfig`

```
NameValueCollection appConfig = ConfigurationManager.AppSettings;
string endpoint = appConfig["API_ENDPOINT"];
```

- Node.js – `process.env`

```
var endpoint = process.env.API_ENDPOINT
```

- PHP — `$_SERVER`

```
$endpoint = $_SERVER['API_ENDPOINT'];
```

- Python – `os.environ`

```
import os
endpoint = os.environ['API_ENDPOINT']
```

- Ruby – ENV

```
endpoint = ENV['API_ENDPOINT']
```

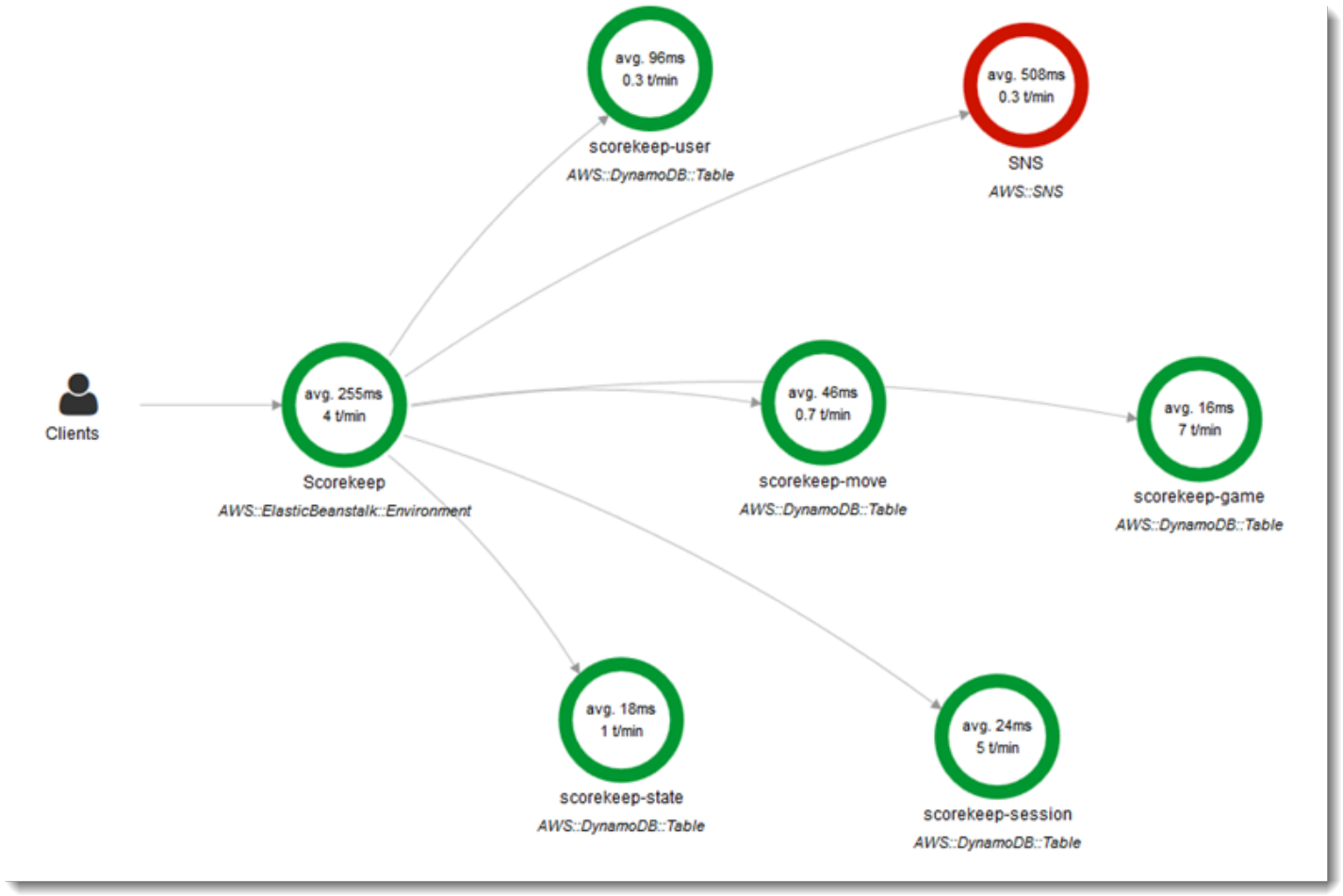
Di luar kode aplikasi, seperti dalam skrip yang berjalan selama deployment, Anda dapat mengakses properti lingkungan dengan skrip platform `get-config`. Lihat elastic-beanstalk-samples GitHub repositori misalnya konfigurasi yang menggunakan. `get-config`

Mengkonfigurasi debugging AWS X-Ray

Anda dapat menggunakan AWS Elastic Beanstalk konsol atau file konfigurasi untuk menjalankan AWS X-Ray daemon pada instance di lingkungan Anda. X-Ray adalah AWS layanan yang mengumpulkan data tentang permintaan yang dilayani aplikasi Anda, dan menggunakannya untuk

membuat peta layanan yang dapat Anda gunakan untuk mengidentifikasi masalah dengan aplikasi Anda dan peluang untuk pengoptimalan.

> **Note**
>
> Beberapa wilayah tidak menawarkan X-Ray. Jika Anda membuat lingkungan di salah satu wilayah ini, Anda tidak dapat menjalankan daemon X-Ray pada instans di lingkungan Anda. Untuk informasi tentang AWS layanan yang ditawarkan di setiap Wilayah, lihat Tabel Wilayah.



X-Ray menyediakan SDK yang dapat Anda gunakan untuk melengkapi kode aplikasi Anda, dan aplikasi daemon yang menyampaikan informasi debugging dari SDK ke API X-Ray.

Platform yang didukung

Anda dapat menggunakan SDK X-Ray dengan platform Elastic Beanstalk berikut ini:

- Pergi - versi 2.9.1 dan yang lebih baru
- Java 8 - versi 2.3.0 dan yang lebih baru
- Java 8 dengan Tomcat 8 - versi 2.4.0 dan yang lebih baru
- Node.js - versi 3.2.0 dan yang lebih baru
- Server Windows - semua versi platform yang dirilis pada atau setelah 18 Desember 2016
- Python - versi 2.5.0 dan yang lebih baru

Pada platform yang didukung, Anda dapat menggunakan opsi konfigurasi untuk menjalankan daemon X-Ray pada instans di lingkungan Anda. Anda dapat mengaktifkan daemon di konsol Elastic Beanstalk atau dengan menggunakan file konfigurasi.

Untuk mengunggah data ke X-Ray, daemon X-Ray memerlukan izin IAM dalam kebijakan terkelola. AWSXrayWriteOnlyAccess Izin ini disertakan dalam profil instans Elastic Beanstalk. Jika Anda tidak menggunakan profil instans default, lihat Memberikan Izin Daemon untuk Mengirim Data ke X-Ray di AWS X-Ray Panduan Developer.

Debugging dengan X-Ray memerlukan penggunaan X-Ray SDK. Lihat Memulai dengan AWS X-Ray di Panduan Developer AWS X-Ray untuk instruksi dan aplikasi sampel.

Jika Anda menggunakan versi platform yang tidak menyertakan daemon, Anda masih dapat menjalankannya dengan skrip di file konfigurasi. Untuk informasi selengkapnya, lihat Mengunduh dan Menjalankan Daemon X-Ray (Lanjutan) di Panduan Developer AWS X-Ray .

Bagian

- Mengonfigurasi debugging
- Namespace aws:elasticbeanstalk:xray

Mengonfigurasi debugging

Anda dapat mengaktifkan daemon X-Ray pada lingkungan yang berjalan di konsol Elastic Beanstalk.

Untuk mengaktifkan debugging di konsol Elastic Beanstalk

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.

3. Pada panel navigasi, pilih Konfigurasi.
4. Dalam kategori konfigurasi Pembaruan, pemantauan, dan pencatatan, pilih Edit.
5. Di bagian Amazon X-Ray, pilih Diaktifkan.
6. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Anda juga dapat mengaktifkan opsi ini selama pembuatan lingkungan. Untuk informasi selengkapnya, lihat Wizard pembuatan lingkungan baru.

Namespace aws:elasticbeanstalk:xray

Anda dapat menggunakan opsi `XRayEnabled` pada namespace `aws:elasticbeanstalk:xray` untuk mengaktifkan debugging.

Untuk mengaktifkan debugging secara otomatis ketika Anda menerapkan aplikasi Anda, atur opsi dalam file konfigurasi pada kode sumber Anda, sebagai berikut.

Example .ebextensions/debugging.config

```
option_settings:
  aws:elasticbeanstalk:xray:
    XRayEnabled: true
```

Melihat log lingkungan Elastic Beanstalk Anda

AWS Elastic Beanstalk menyediakan dua cara untuk secara teratur melihat log dari EC2 instans Amazon yang menjalankan aplikasi Anda:

- Konfigurasikan lingkungan Elastic Beanstalk Anda untuk mengunggah log instans yang dirotasi ke bucket Amazon S3 lingkungan.
- Konfigurasikan lingkungan untuk mengalirkan log instans ke Amazon CloudWatch Logs.

Saat Anda mengonfigurasi streaming log instance ke CloudWatch Log, Elastic CloudWatch Beanstalk membuat grup log Log untuk log proxy dan penerapan di instans EC2 Amazon, dan mentransfer file log ini ke Log secara real time. CloudWatch Untuk informasi selengkapnya tentang log instans, lihat Melihat log dari EC2 instans Amazon di lingkungan Elastic Beanstalk Anda.

Selain log instance, jika Anda mengaktifkan peningkatan kesehatan untuk lingkungan Anda, Anda dapat mengonfigurasi lingkungan untuk mengalirkan informasi kesehatan ke CloudWatch Log. Ketika

status kondisi lingkungan berubah, Elastic Beanstalk menambahkan catatan ke grup log kondisi, dengan status baru dan deskripsi penyebab perubahan. Untuk informasi tentang streaming kondisi lingkungan, lihat Streaming informasi kesehatan lingkungan Elastic Beanstalk ke Amazon Logs CloudWatch .

Mengonfigurasi tampilan log instans

Untuk melihat log instans, Anda dapat mengaktifkan rotasi log instans dan log streaming di konsol Elastic Beanstalk.

Untuk mengonfigurasi rotasi log instans dan log streaming di konsol Elastic Beanstalk

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Dalam kategori konfigurasi Pembaruan, pemantauan, dan pencatatan, pilih Edit.
5. Di bagian penyimpanan log S3, pilih Diaktifkan di bawah Putar log untuk mengaktifkan pengunggahan log yang diputar ke Amazon S3.
6. di bagian Streaming log Instans ke CloudWatch Log, konfigurasikan pengaturan berikut:
   - Streaming log - Pilih Diaktifkan untuk mengaktifkan streaming log.
   - Retensi - Tentukan jumlah hari untuk menyimpan log di CloudWatch Log.
   - Siklus Hidup — Setel ke Hapus log setelah penghentian untuk menghapus CloudWatch log dari Log segera jika lingkungan dihentikan, alih-alih menunggu mereka kedaluwarsa.
7. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Setelah Anda mengaktifkan streaming log, Anda dapat kembali ke kategori atau halaman konfigurasi Perangkat Lunak dan menemukan tautan Grup Log. Klik tautan ini untuk melihat log instans Anda di CloudWatch konsol.

Mengonfigurasi tampilan log kondisi lingkungan

Untuk melihat log kondisi lingkungan, Anda dapat mengaktifkan streaming log kondisi lingkungan di konsol Elastic Beanstalk.

Untuk mengonfigurasi streaming log kondisi lingkungan di konsol Elastic Beanstalk

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Dalam kategori konfigurasi Pembaruan, pemantauan, dan pencatatan, pilih Edit.
5. Pergi ke bagian Pemantauan.
6. Di bawah streaming acara Kesehatan ke CloudWatch Log, konfigurasikan pengaturan berikut:
   - Log streaming - Pilih untuk Diaktifkan untuk mengaktifkan streaming log.
   - Retensi - Tentukan jumlah hari untuk menyimpan log di CloudWatch Log.
   - Siklus Hidup — Setel ke Hapus log setelah penghentian untuk menghapus CloudWatch log dari Log segera jika lingkungan dihentikan, alih-alih menunggu mereka kedaluwarsa.
7. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Log menampilkan namespace

Namespace berikut berisi pengaturan untuk tampilan log:

- `aws:elasticbeanstalk:hostmanager` — Konfigurasikan pengunggahan log yang dirotasi ke Amazon S3.
- `aws:elasticbeanstalk:cloudwatch:logs`— Konfigurasikan streaming log instance ke CloudWatch.
- `aws:elasticbeanstalk:cloudwatch:logs:health`— Konfigurasikan streaming kesehatan lingkungan ke CloudWatch.

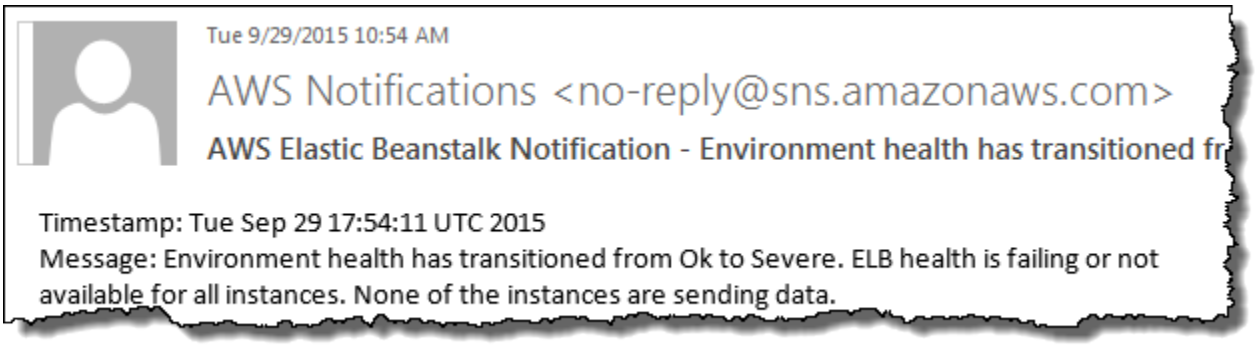
Pemberitahuan lingkungan Elastic Beanstalk dengan Amazon SNS

Anda dapat mengonfigurasi AWS Elastic Beanstalk lingkungan untuk menggunakan Amazon Simple Notification Service (Amazon SNS) untuk memberi tahu Anda tentang peristiwa penting yang memengaruhi aplikasi Anda. Untuk menerima email dari AWS setiap kali terjadi kesalahan atau kesehatan lingkungan Anda berubah, tentukan alamat email saat Anda membuat lingkungan atau nanti.

> **Note**
>
> Elastic Beanstalk menggunakan Amazon SNS untuk pemberitahuan. Untuk informasi tentang harga Amazon SNS, lihat https://aws.amazon.com/sns/ harga/.

Ketika Anda mengonfigurasi pemberitahuan untuk lingkungan Anda, Elastic Beanstalk membuat topik Amazon SNS untuk lingkungan Anda atas nama Anda. Untuk mengirim pesan ke topik Amazon SNS, Elastic Beanstalk harus memiliki izin yang diperlukan. Untuk informasi selengkapnya, lihat Mengonfigurasi izin untuk mengirim pemberitahuan.

Saat peristiwa penting terjadi, Elastic Beanstalk mengirimkan pesan ke topik. Kemudian, Amazon SNS menyampaikan pesan yang diterima ke pelanggan topik. Peristiwa penting mencakup kesalahan pembuatan lingkungan dan semua perubahan pada lingkungan dan kondisi instans. Peristiwa untuk operasi EC2 Auto Scaling Amazon (seperti menambahkan dan menghapus instance dari lingkungan) dan peristiwa informasi lainnya tidak memicu pemberitahuan.



Anda dapat memasukkan alamat email di konsol Elastic Beanstalk ketika Anda membuat lingkungan atau setelahnya. Proses ini akan membuat topik Amazon SNS dan berlangganan topik tersebut. Elastic Beanstalk mengelola siklus hidup topik, dan menghapusnya saat lingkungan Anda dihentikan atau saat Anda menghapus alamat email di konsol manajemen lingkungan.

Namespace `aws:elasticbeanstalk:sns:topics` tersebut memberikan opsi untuk mengonfigurasi topik Amazon SNS dengan menggunakan file konfigurasi, CLI, atau SDK. Dengan menggunakan salah satu metode ini, Anda dapat mengonfigurasi jenis pelanggan dan titik akhir. Untuk jenis pelanggan, Anda dapat memilih antrean Amazon SQS atau URL HTTP.

Anda hanya dapat mengaktifkan atau menonaktifkan pemberitahuan Amazon SNS. Frekuensi pemberitahuan yang dikirim ke topik bisa tinggi, tergantung dari ukuran dan komposisi lingkungan Anda. Untuk mengonfigurasi pemberitahuan yang akan dikirim pada keadaan tertentu, Anda memiliki

opsi lain. Anda dapat mengatur aturan berbasis peristiwa dengan Amazon EventBridge yang memberi tahu Anda saat Elastic Beanstalk memancarkan peristiwa yang memenuhi kriteria tertentu. Atau, sebagai alternatif, Anda dapat mengonfigurasi lingkungan Anda untuk menerbitkan metrik khusus dan menyetel CloudWatch alarm Amazon untuk memberi tahu Anda ketika metrik tersebut mencapai ambang batas yang tidak sehat.

Mengonfigurasi pemberitahuan menggunakan konsol Elastic Beanstalk

Anda dapat memasukkan alamat email di konsol Elastic Beanstalk untuk membuat topik Amazon SNS untuk lingkungan Anda.

Untuk mengonfigurasi pemberitahuan di konsol Elastic Beanstalk

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Dalam kategori konfigurasi Pembaruan, pemantauan, dan pencatatan, pilih Edit.
5. Gulir ke bawah ke bagian Pemberitahuan email.
6. Masukkan alamat email.
7. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Ketika Anda memasukkan alamat email untuk pemberitahuan, Elastic Beanstalk membuat topik Amazon SNS untuk lingkungan Anda dan menambahkan langganan. Amazon SNS mengirimkan email ke alamat berlangganan untuk mengonfirmasi langganan. Anda harus mengklik tautan di email konfirmasi untuk mengaktifkan langganan dan menerima pemberitahuan.

Mengonfigurasi pemberitahuan menggunakan opsi konfigurasi

Gunakan opsi di namespace `aws:elasticbeanstalk:sns:topics` untuk mengonfigurasi pemberitahuan Amazon SNS untuk lingkungan Anda. Anda dapat mengatur opsi ini dengan menggunakan file konfigurasi, CLI, atau SDK.

- Titik Akhir Pemberitahuan — Alamat email, antrean Amazon SQS, atau URL untuk mengirim pemberitahuan. Jika Anda menetapkan opsi ini, maka antrian SQS dan langganan untuk titik akhir yang ditentukan akan dibuat. Jika titik akhir bukan alamat email, Anda juga harus mengatur opsi `Notification Protocol`. SNS memvalidasi nilai `Notification Endpoint` berdasarkan nilai

dari `Notification Protocol`. Mengatur opsi ini beberapa kali membuat langganan tambahan untuk topik. Jika Anda menghapus opsi ini, topik akan dihapus.

- Protokol Pemberitahuan — Protokol yang digunakan untuk mengirim pemberitahuan ke `Notification Endpoint`. Opsi ini default ke `email`. Atur opsi ini menjadi `email-json` untuk mengirim email berformat JSON, `http` atau `https` untuk memposting pemberitahuan berformat JSON ke titik akhir HTTP, atau `sqs` untuk mengirim pemberitahuan ke antrean SQS.

  > **Note**
  >
  > AWS Lambda notifikasi tidak didukung.

- ARN Topik Pemberitahuan — Setelah mengatur titik akhir pemberitahuan untuk lingkungan Anda, baca pengaturan ini untuk mendapatkan ARN topik SNS. Anda juga dapat mengatur opsi ini agar menggunakan topik SNS yang ada untuk pemberitahuan. Topik yang Anda lampirkan ke lingkungan Anda meskipun opsi ini tidak dihapus ketika Anda mengubah opsi ini atau mengakhiri lingkungan.

  Untuk mengonfigurasi pemberitahuan Amazon SNS, Anda harus memiliki izin yang diperlukan. Jika pengguna IAM Anda menggunakan kebijakan pengguna terkelola AdministratorAccessElastic AWSElastic Beanstalk - Beanstalk, maka Anda seharusnya sudah memiliki izin yang diperlukan untuk mengonfigurasi topik Amazon SNS default yang dibuat Elastic Beanstalk untuk lingkungan Anda. Namun, jika Anda mengonfigurasi topik Amazon SNS yang tidak kelola Elastic Beanstalk, maka Anda perlu menambahkan kebijakan berikut untuk peran pengguna Anda.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "sns:SetTopicAttributes",
        "sns:GetTopicAttributes",
        "sns:Subscribe",
        "sns:Unsubscribe",
        "sns:Publish"
      ],
      "Resource": [
        "arn:aws:sns:us-east-2:123456789012:sns_topic_name"
      ]
    }
```

```
  ]
}
```

- Nama Topik Pemberitahuan — Atur opsi ini untuk menyesuaikan nama topik Amazon SNS yang digunakan untuk pemberitahuan lingkungan. Jika topik dengan nama yang sama sudah ada, Elastic Beanstalk melampirkan topik itu ke lingkungan.

  > **Warning**
  >
  > Jika Anda melampirkan topik SNS yang ada ke lingkungan `Notification Topic Name`, Elastic Beanstalk akan menghapus topik tersebut jika Anda mengakhiri lingkungan atau mengubah pengaturan ini di masa depan.

  Jika Anda mengubah opsi ini, `Notification Topic ARN` juga diubah. Jika topik sudah dilampirkan ke lingkungan, Elastic Beanstalk menghapus topik lama dan membuat topik dan langganan baru.

  Dengan menggunakan nama topik khusus, Anda juga harus menyediakan ARN dari topik khusus yang dibuat secara eksternal. Kebijakan pengguna terkelola tidak secara otomatis mendeteksi topik dengan nama khusus, sehingga Anda harus memberikan izin khusus Amazon SNS untuk pengguna IAM Anda. Gunakan kebijakan yang serupa dengan yang digunakan untuk topik khusu ARN, tetapi sertakan tambahan berikut:

  - Sertakan dua tindakan lagi dalam daftar `Actions`, khususnya: `sns:CreateTopic`, `sns:DeleteTopic`
  - Jika Anda mengubah `Notification Topic Name` dari satu nama topik kustom ke yang lain, Anda juga harus menyertakan kedua topik dalam `Resource` daftar. ARNs Sebagai alternatif, sertakan ekspresi reguler yang mencakup kedua topik tersebut. Dengan cara ini Elastic Beanstalk memiliki izin untuk menghapus topik lama dan membuat yang baru.

Konsol EB CLI dan Elastic Beanstalk menerapkan nilai yang direkomendasikan untuk pilihan sebelumnya. Anda harus menghapus pengaturan ini jika Anda ingin menggunakan file konfigurasi untuk mengonfigurasi hal yang sama. Lihat Nilai yang disarankan untuk rincian selengkapnya.

Mengonfigurasi izin untuk mengirim pemberitahuan

Bagian ini membahas pertimbangan keamanan yang berkaitan dengan pemberitahuan yang menggunakan Amazon SNS. Ada dua kasus yang berbeda:

- Gunakan topik Amazon SNS default yang dibuat Elastic Beanstalk untuk lingkungan Anda.
- Sediakan topik Amazon SNS eksternal melalui opsi konfigurasi.

Kebijakan akses default untuk topik Amazon SNS mengizinkan hanya pemilik topik yang mempublikasikan atau berlangganan topik tersebut. Namun, melalui konfigurasi kebijakan yang tepat, Elastic Beanstalk dapat diberikan izin untuk mempublikasikan topik Amazon SNS di salah satu dari dua kasus yang dijelaskan dalam bagian ini. Subbagian berikut memberikan informasi lebih lanjut.

Izin untuk topik default

Ketika Anda mengonfigurasi pemberitahuan untuk lingkungan Anda, Elastic Beanstalk membuat topik Amazon SNS untuk lingkungan Anda. Untuk mengirim pesan ke topik Amazon SNS, Elastic Beanstalk harus memiliki izin yang diperlukan. Jika lingkungan Anda menggunakan peran layanan yang dihasilkan oleh konsol Elastic Beanstalk atau EB CLI untuknya, atau peran yang terhubung ke layanan pemantauan akun Anda, maka Anda tidak perlu melakukan hal lain. Peran terkelola mencakup izin yang diperlukan yang mengizinkan Elastic Beanstalk untuk mengirim pesan ke topik Amazon SNS.

Namun, jika Anda memberikan peran layanan khusus ketika Anda membuat lingkungan Anda, pastikan bahwa peran layanan khusus ini mencakup kebijakan berikut.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "sns:Publish"
      ],
      "Resource": [
        "arn:aws:sns:us-east-2:123456789012:ElasticBeanstalkNotifications*"
      ]
    }
  ]
}
```

Izin untuk topik eksternal

Mengonfigurasi pemberitahuan menggunakan opsi konfigurasi menjelaskan bagaimana Anda dapat mengganti topik Amazon SNS yang disediakan oleh Elastic Beanstalk dengan topik Amazon

SNS lain. Jika Anda mengganti topik, Elastic Beanstalk harus memverifikasi bahwa Anda memiliki izin untuk menerbitkan topik SNS ini agar Anda dapat mengasosiasikan topik SNS dengan lingkungan. Anda harus memiliki `sns:Publish`. Peran layanan menggunakan izin yang sama. Untuk memverifikasi bahwa hal ini terjadi, Elastic Beanstalk mengirimkan pemberitahuan tes ke SNS sebagai bagian dari tindakan Anda untuk membuat atau memperbarui lingkungan. Jika tes ini gagal, maka usaha Anda untuk membuat atau memperbarui lingkungan juga gagal. Elastic Beanstalk menampilkan pesan yang menjelaskan alasan kegagalan ini.

Jika Anda memberikan peran layanan khusus untuk lingkungan Anda, pastikan bahwa peran layanan khusus Anda mencakup kebijakan berikut untuk mengizinkan Elastic Beanstalk mengirim pesan ke topik Amazon SNS. Pada kode berikut, ganti *sns_topic_name* dengan nama topik Amazon SNS yang Anda berikan pada opsi konfigurasi.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "sns:Publish"
      ],
      "Resource": [
        "arn:aws:sns:us-east-2:123456789012:sns_topic_name"
      ]
    }
  ]
}
```

Untuk informasi selengkapnya tentang kontrol akses Amazon SNS, lihat Contoh kasus untuk kontrol akses Amazon SNS di Panduan Developer Amazon Simple Notification Service.

Mengonfigurasi Amazon Virtual Private Cloud (Amazon VPC) dengan Elastic Beanstalk

Amazon Virtual Private Cloud (Amazon VPC) adalah layanan jaringan yang merutekan lalu lintas dengan aman ke EC2 instance yang menjalankan aplikasi Anda di Elastic Beanstalk. Jika Anda tidak mengonfigurasi VPC ketika Anda meluncurkan lingkungan Anda, Elastic Beanstalk menggunakan VPC default.

Anda dapat meluncurkan lingkungan Anda di VPC khusus untuk menyesuaikan pengaturan jaringan dan keamanan. Elastic Beanstalk memungkinkan Anda memilih subnet mana yang akan digunakan untuk sumber daya Anda, dan cara mengonfigurasi alamat IP untuk instans dan penyeimbang beban di lingkungan Anda. Lingkungan terkunci untuk VPC ketika Anda membuatnya, tetapi Anda dapat mengubah subnet dan pengaturan alamat IP di lingkungan yang sedang berjalan.

Mengonfigurasi pengaturan VPC di konsol Elastic Beanstalk

Jika Anda memilih VPC khusus ketika Anda membuat lingkungan Anda, Anda dapat mengubah pengaturan VPC di konsol Elastic Beanstalk.

Untuk mengonfigurasi pengaturan VPC lingkungan

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Pada kategori konfigurasi Jaringan, pilih Edit.

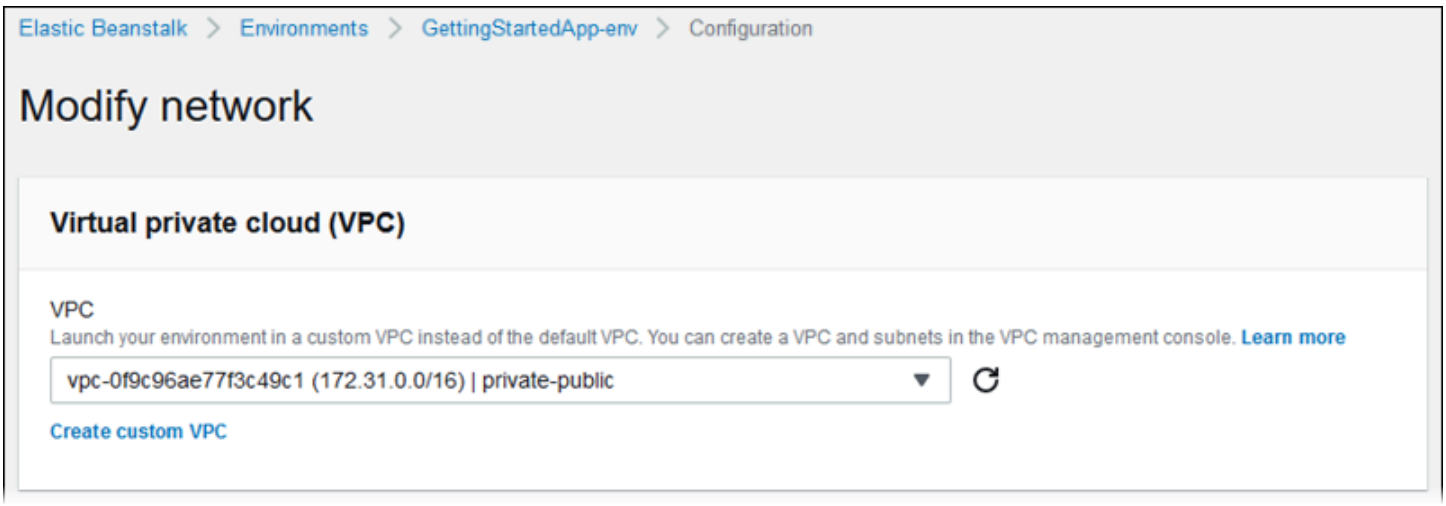
Pengaturan berikut tersedia.

Opsi

- VPC
- Visibilitas penyeimbang beban
- Subnet penyeimbang beban
- Alamat IP publik
- Subnet instans
- Subnet basis data

VPC

Pilih VPC untuk lingkungan Anda. Anda hanya dapat mengubah pengaturan ini selama pembuatan lingkungan.

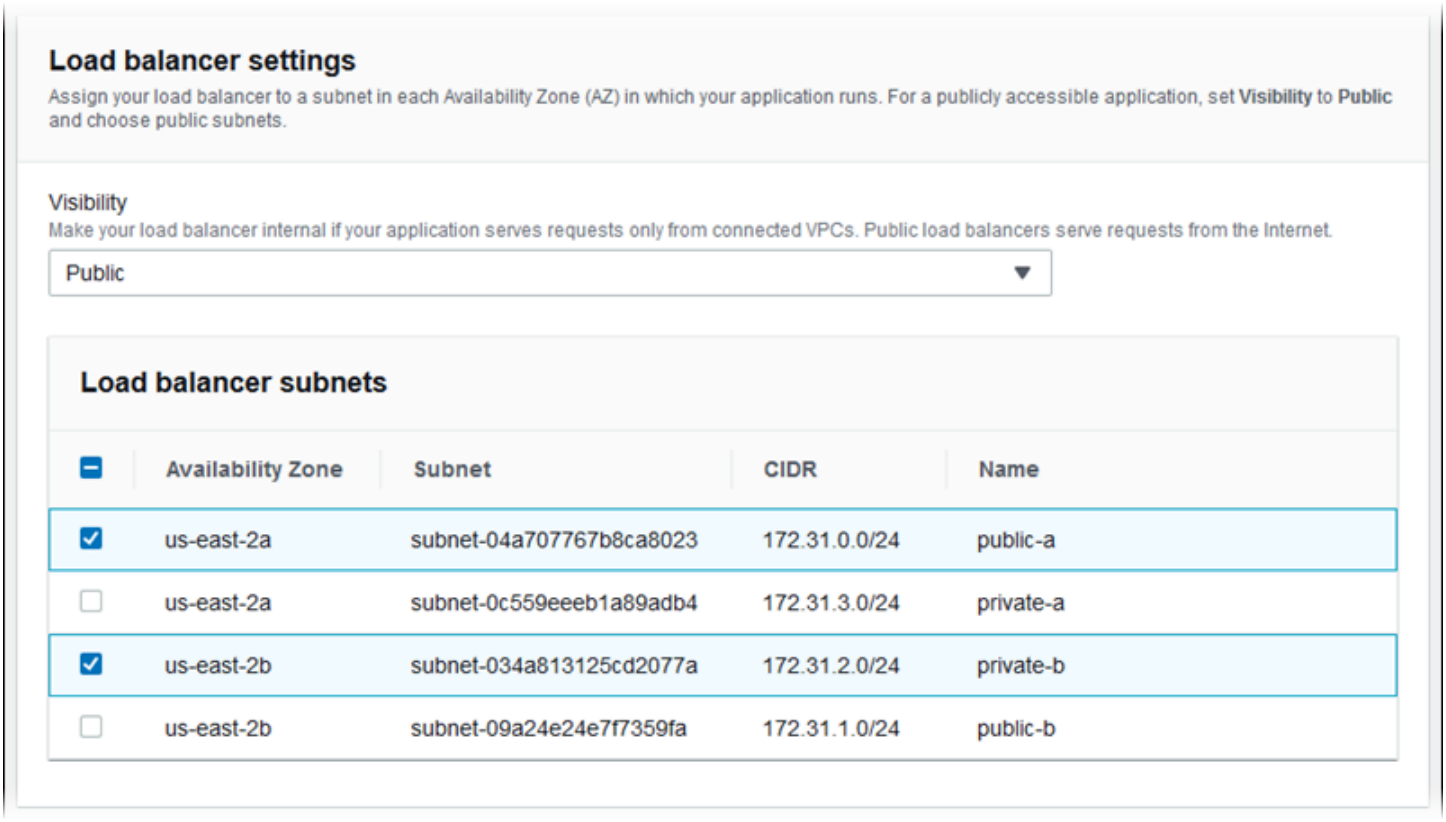


Visibilitas penyeimbang beban

Untuk lingkungan yang seimbang dengan beban, pilih skema penyeimbang beban. Secara default, penyeimbang beban bersifat publik, dengan alamat IP publik dan nama domain. Jika aplikasi Anda hanya melayani lalu lintas dari dalam VPC atau VPN yang terhubung, batalkan pilihan ini dan pilih subnet pribadi untuk penyeimbang beban Anda untuk menjadikan penyeimbang beban internal dan menonaktifkan akses dari Internet.

Subnet penyeimbang beban

Untuk lingkungan yang seimbang dengan beban, pilih subnet yang digunakan penyeimbang beban untuk melayani lalu lintas. Untuk aplikasi publik, pilih subnet publik. Gunakan subnet di beberapa zona ketersediaan untuk ketersediaan tinggi. Untuk aplikasi internal, pilih subnet pribadi dan nonaktifkan visibilitas penyeimbang beban.

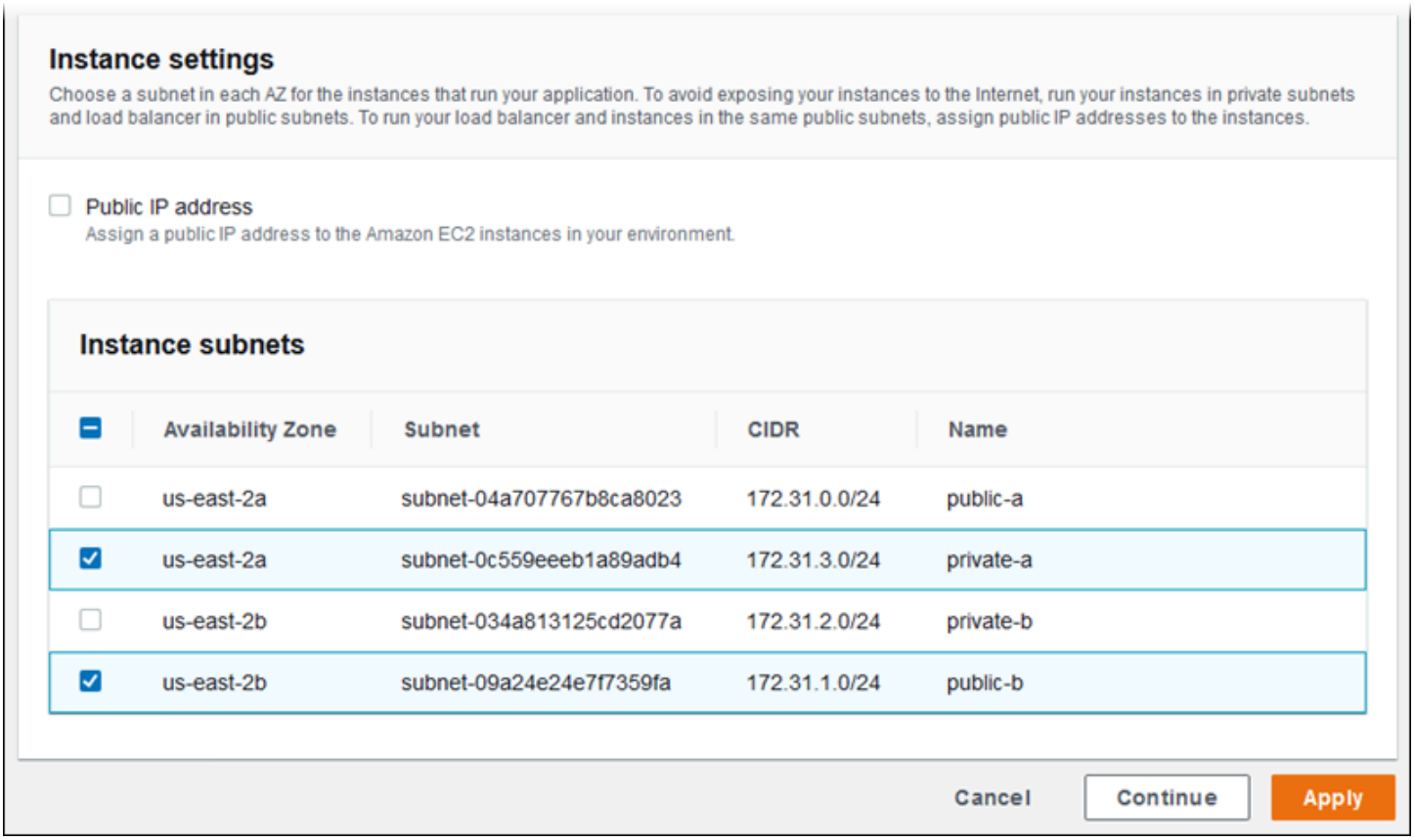


Alamat IP publik

Jika Anda memilih subnet publik untuk instans aplikasi Anda, aktifkan alamat IP publik untuk membuatnya dapat dirutekan dari Internet.

Subnet instans

Pilih subnet untuk instans aplikasi Anda. Pilih setidaknya satu subnet untuk setiap availability zone yang digunakan penyeimbang beban Anda. Jika Anda memilih subnet pribadi untuk instans Anda, VPC Anda harus memiliki gateway NAT di subnet publik yang dapat digunakan instans untuk mengakses Internet.



Subnet basis data

Ketika Anda menjalankan basis data Amazon RDS yang terlampir di lingkungan Elastic Beanstalk Anda, pilih subnet untuk instans basis data Anda. Untuk ketersediaan tinggi, buat basis data multi-AZ dan pilih subnet untuk setiap availability zone. Untuk memastikan bahwa aplikasi Anda dapat terhubung ke basis data Anda, jalankan keduanya di subnet yang sama.

Namespace aws:ec2:vpc

Anda dapat menggunakan opsi konfigurasi di namespace `aws:ec2:vpc` untuk mengonfigurasi pengaturan jaringan lingkungan Anda.

File konfigurasi berikut menggunakan pilihan pada namespace ini untuk mengatur lingkungan VPC dan subnet untuk konfigurasi publik-pribadi. Untuk mengatur ID VPC dalam file konfigurasi, file harus disertakan dalam paket sumber aplikasi selama pembuatan lingkungan. Lihat Menetapkan opsi konfigurasi selama pembuatan lingkungan untuk metode lain untuk mengonfigurasi pengaturan ini selama pembuatan lingkungan.

Example .ebextensions/vpc.config — Publik-pribadi

```
option_settings:
   aws:ec2:vpc:
      VPCId: vpc-087a68c03b9c50c84
      AssociatePublicIpAddress: 'false'
      ELBScheme: public
      ELBSubnets: subnet-0fe6b36bcb0ffc462,subnet-032fe3068297ac5b2
      Subnets: subnet-026c6117b178a9c45,subnet-0839e902f656e8bd1
```

Contoh ini menunjukkan konfigurasi publik-publik, di mana penyeimbang beban dan EC2 instance berjalan di subnet publik yang sama.

Example .ebextensions/vpc.config — Publik-publik

```
option_settings:
   aws:ec2:vpc:
      VPCId: vpc-087a68c03b9c50c84
      AssociatePublicIpAddress: 'true'
      ELBScheme: public
      ELBSubnets: subnet-0fe6b36bcb0ffc462,subnet-032fe3068297ac5b2
      Subnets: subnet-0fe6b36bcb0ffc462,subnet-032fe3068297ac5b2
```

Migrasi lingkungan Elastic EC2 Beanstalk dari -Classic ke VPC

Topik ini menjelaskan berbagai opsi tentang cara memigrasikan lingkungan Elastic Beanstalk Anda EC2 dari platform jaringan -Classic ke jaringan Amazon Virtual Private Cloud (Amazon VPC).

Jika Anda membuat AWS akun sebelum 4 Desember 2013, Anda mungkin memiliki lingkungan yang menggunakan konfigurasi jaringan EC2 -Classic di beberapa Wilayah AWS. Semua AWS akun yang dibuat pada atau setelah 4 Desember 2013 sudah hanya VPC di setiap Wilayah. AWS Satu-satunya pengecualian adalah jika Amazon EC2 -Classic diaktifkan sebagai hasil dari permintaan dukungan.

> **Note**
>
> Anda dapat melihat pengaturan konfigurasi jaringan untuk lingkungan Anda dalam kategori konfigurasi Jaringan pada halaman ikhtisar Konfigurasi konsol Elastic Beanstalk.

Mengapa Anda harus bermigrasi

Amazon EC2 -Classic akan mencapai akhir dukungan standar pada 15 Agustus 2022. Untuk menghindari gangguan pada beban kerja Anda, sebaiknya Anda bermigrasi dari EC2 Amazon - Classic ke VPC sebelum 15 Agustus 2022. Kami juga meminta Anda untuk tidak meluncurkan AWS sumber daya apa pun di Amazon EC2 -Classic di masa depan dan menggunakan Amazon VPC sebagai gantinya.

Saat Anda memigrasikan lingkungan Elastic Beanstalk dari EC2 Amazon -Classic ke Amazon VPC, Anda harus membuat akun baru. AWS Anda juga harus membuat ulang lingkungan AWS EC2 -Classic Anda di akun baru AWS Anda. Tidak ada pekerjaan konfigurasi tambahan untuk lingkungan Anda yang diperlukan untuk menggunakan VPC default. Jika VPC default tidak memenuhi persyaratan Anda, buat VPC kustom secara manual dan kaitkan dengan lingkungan Anda.

Atau, jika AWS akun Anda yang ada memiliki sumber daya yang tidak dapat Anda migrasi ke AWS akun baru, tambahkan VPC ke akun Anda saat ini. Kemudian, konfigurasikan lingkungan Anda untuk menggunakan VPC.

Untuk informasi lebih lanjut, lihat EC2-Classic Networking is Retiring - Inilah Cara Mempersiapkan posting blog.

Migrasikan lingkungan dari EC2 -Classic ke AWS akun baru (disarankan)

Jika Anda belum memiliki AWS akun yang dibuat pada atau setelah 4 Desember 2013, buat akun baru. Anda akan memigrasikan lingkungan Anda ke akun baru ini.

1. AWS Akun baru Anda menyediakan VPC default ke lingkungannya. Jika Anda tidak perlu membuat VPC khusus, lewati ke langkah 2.

   Anda dapat membuat VPC khusus dengan salah satu cara berikut:

   - Buat VPC dengan cepat menggunakan wizard konsol VPC Amazon dengan salah satu opsi konfigurasi yang tersedia. Untuk informasi selengkapnya, lihat konfigurasi wizard konsol Amazon VPC.
   - Buat VPC khusus di konsol VPC Amazon jika Anda memiliki persyaratan yang lebih spesifik untuk VPC Anda. Kami menyarankan Anda melakukan ini, misalnya, jika kasus penggunaan Anda memerlukan sejumlah subnet tertentu. Untuk informasi lebih lanjut, lihat VPCs dan subnet.

- Buat VPC menggunakan elastic-beanstalk-samplesrepositori di GitHub situs web jika Anda lebih suka menggunakan AWS CloudFormation templat dengan lingkungan Elastic Beanstalk Anda. Repositori ini mencakup AWS CloudFormation template. Untuk informasi selengkapnya, lihat Menggunakan Elastic Beanstalk dengan Amazon VPC.

> **Note**
>
> Anda juga dapat membuat VPC kustom pada saat yang sama Anda membuat ulang lingkungan di AWS akun baru Anda menggunakan wizard buat lingkungan baru. Jika Anda menggunakan wizard dan memilih untuk membuat VPC khusus, wizard mengarahkan Anda ke konsol VPC Amazon.

2. Di AWS akun baru Anda, buat lingkungan baru. Sebaiknya lingkungan menyertakan konfigurasi yang sama dengan lingkungan yang ada di AWS akun tempat Anda bermigrasi. Anda dapat melakukan ini dengan menggunakan salah satu metode berikut.

> **Note**
>
> Jika lingkungan baru Anda harus menggunakan CNAME yang sama setelah Anda bermigrasi, hentikan lingkungan asli pada platform -Classic. EC2 Ini merilis CNAME untuk digunakan. Namun, hal itu dapat mengakibatkan downtime untuk lingkungan itu dan juga dapat berisiko bahwa pelanggan lain mungkin memilih CNAME Anda antara Anda mengakhiri lingkungan EC2 -Classic Anda dan membuat yang baru. Untuk informasi selengkapnya, lihat Mengakhiri lingkungan Elastic Beanstalk.
> Untuk lingkungan yang memiliki nama domain milik mereka sendiri, CNAME tidak memiliki masalah ini. Anda hanya dapat memperbarui Domain Name System (DNS) untuk meneruskan permintaan ke CNAME baru.

   - Gunakan wizard membuat lingkungan baru pada Konsol Elastic Beanstalk. Wizard tersebut menyediakan pilihan untuk membuat VPC khusus. Jika Anda tidak memilih untuk membuat VPC kustom, VPC default ditetapkan.
   - Gunakan Elastic Beanstalk Command Line Interface (EB CLI) untuk menciptakan kembali lingkungan Anda di akun baru Anda. AWS Salah satu contoh pada deskripsi perintah eb create menunjukkan pembuatan lingkungan di VPC khusus. Jika Anda tidak memberikan ID VPC, lingkungan menggunakan VPC default.

Dengan menggunakan pendekatan ini, Anda dapat menggunakan file konfigurasi yang disimpan di kedua AWS akun. Akibatnya, Anda tidak perlu memasukkan semua informasi konfigurasi secara manual. Namun, Anda harus menyimpan pengaturan konfigurasi untuk lingkungan EC2 -Classic yang Anda migrasi dengan perintah simpan konfigurasi eb. Salin file konfigurasi tersimpan ke direktori baru untuk lingkungan akun baru.

> **Note**
>
> Anda harus mengedit beberapa data dalam file konfigurasi yang disimpan sebelum Anda dapat menggunakannya di akun baru. Anda juga harus memperbarui informasi yang berkaitan dengan akun Anda sebelumnya dengan data yang benar untuk akun baru Anda. Misalnya, Anda harus mengganti Amazon Resource Name (ARN) dari (IAM) role AWS Identity and Access Management dengan ARN IAM role untuk akun baru.

Jika Anda menggunakan perintah eb create dengan`cfg`, lingkungan baru dibuat menggunakan file konfigurasi tersimpan yang ditentukan. Untuk informasi selengkapnya, lihat Menggunakan konfigurasi tersimpan Elastic Beanstalk.

Migrasikan lingkungan dari EC2 -Classic dalam akun Anda yang sama AWS

AWS Akun Anda yang sudah ada mungkin memiliki sumber daya yang tidak dapat Anda migrasi ke AWS akun baru. Dalam hal ini Anda harus membuat ulang lingkungan Anda dan secara manual mengkonfigurasi VPC untuk setiap lingkungan yang Anda buat.

Migrasikan lingkungan Anda ke VPC khusus

Prasyarat

Sebelum memulai, Anda harus memiliki VPC. Anda dapat membuat VPC non-default (khusus) dengan salah satu cara berikut:

- Buat VPC dengan cepat menggunakan wizard konsol VPC Amazon dengan salah satu opsi konfigurasi yang tersedia. Untuk informasi selengkapnya, lihat konfigurasi wizard konsol Amazon VPC.

- Buat VPC khusus di konsol VPC Amazon jika Anda memiliki persyaratan yang lebih spesifik untuk VPC Anda. Kami menyarankan Anda melakukan ini, misalnya, jika kasus penggunaan Anda memerlukan sejumlah subnet tertentu. Untuk informasi lebih lanjut, lihat VPCs dan subnet.
- Buat VPC menggunakan elastic-beanstalk-samplesrepositori di GitHub situs web jika Anda lebih suka menggunakan AWS CloudFormation templat dengan lingkungan Elastic Beanstalk Anda. Repositori ini mencakup AWS CloudFormation template. Untuk informasi selengkapnya, lihat Menggunakan Elastic Beanstalk dengan Amazon VPC.

Dalam langkah-langkah berikut, Anda menggunakan ID VPC dan subnet yang dihasilkan IDs saat Anda mengonfigurasi VPC di lingkungan baru.

1. Buat lingkungan baru yang mencakup konfigurasi yang sama seperti lingkungan Anda yang sudah ada. Anda dapat melakukan ini dengan menggunakan salah satu metode berikut.

   > **Note**
   >
   > Fitur Konfigurasi Tersimpan dapat membantu Anda membuat ulang lingkungan di akun baru. Fitur ini dapat menyimpan konfigurasi lingkungan, sehingga Anda dapat menerapkannya saat membuat atau memperbarui lingkungan lain. Untuk informasi selengkapnya, lihat Menggunakan konfigurasi tersimpan Elastic Beanstalk.

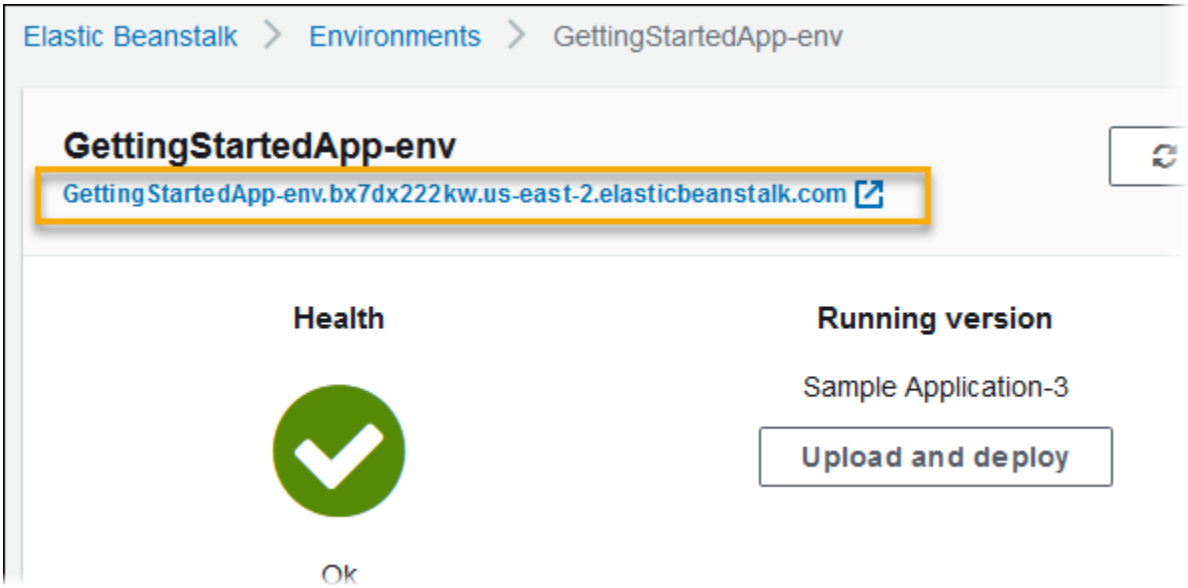
   - Menggunakan konsol Elastic Beanstalk, terapkan konfigurasi tersimpan EC2 dari lingkungan -Classic saat Anda mengonfigurasi lingkungan baru. Konfigurasi ini akan menggunakan VPC. Untuk informasi selengkapnya, lihat Menggunakan konfigurasi tersimpan Elastic Beanstalk.
   - Dengan menggunakan Elastic Beanstalk Command Line Interface (EB CLI), jalankan perintah eb create untuk membuat ulang lingkungan Anda. Berikan parameter lingkungan asli Anda dan pengidentifikasi VPC. Salah satu contoh dalam deskripsi eb create perintah menunjukkan cara membuat lingkungan di VPC kustom.
   - Gunakan AWS Command Line Interface (AWS CLI), dan buat ulang lingkungan Anda menggunakan elasticbeanstalk create-environment perintah. Berikan parameter lingkungan asli Anda dengan pengidentifikasi VPC. Untuk instruksi, lihat Buat lingkungan Elastic Beanstalk dengan AWS CLI.

2. Tukar lingkungan CNAMEs yang ada dengan lingkungan baru. Dengan cara ini, lingkungan baru yang Anda buat dapat direferensikan dengan alamat yang sudah dikenal. Anda dapat menggunakan EB CLI atau AWS CLI.
   - Menggunakan EB CLI, tukar CNAMEs lingkungan dengan menjalankan eb swap perintah. Untuk informasi selengkapnya, lihat Menyiapkan antarmuka baris perintah EB (EB CLI) untuk mengelola Elastic Beanstalk.
   - Menggunakan AWS CLI, tukar lingkungan CNAMEs dengan perintah swap-environment-cnameselasticbeanstalk. Untuk informasi selengkapnya, lihat AWS CLI Referensi Perintah.

Nama domain lingkungan Elastic Beanstalk Anda

Secara default, lingkungan Anda tersedia untuk pengguna di subdomain `elasticbeanstalk.com`. Saat Anda membuat lingkungan, Anda dapat memilih nama host untuk aplikasi Anda. Subdomain dan domain terisi secara otomatis ke *`region`*`.elasticbeanstalk.com`.

Untuk merutekan pengguna ke lingkungan Anda, Elastic Beanstalk mendaftarkan catatan CNAME yang mengarah ke penyeimbang beban lingkungan Anda. Anda dapat melihat URL aplikasi lingkungan Anda dengan nilai CNAME saat ini di halaman gambaran umum konsol Elastic Beanstalk.



Pilih URL pada halaman gambaran umum, atau pilih Masuk ke lingkungan pada panel navigasi, untuk mengarahkan ke halaman web aplikasi Anda.

Anda dapat mengubah CNAME di lingkungan Anda dengan menukarnya dengan CNAME lingkungan lain. Untuk instruksi, lihat Deployment Biru/Hijau dengan Elastic Beanstalk.

Jika Anda memiliki nama domain, Anda dapat menggunakan Amazon Route 53 untuk menyelesaikannya ke lingkungan Anda. Anda dapat membeli nama domain dengan Amazon Route 53, atau menggunakan nama domain yang Anda beli dari penyedia lain.

Untuk membeli nama domain dengan Route 53, lihat Mendaftarkan Domain Baru di Panduan Developer Amazon Route 53.

Untuk mempelajari selengkapnya tentang penggunaan domain khusus, lihat Merutekan Lalu Lintas ke Lingkungan AWS Elastic Beanstalk di Panduan Developer Amazon Route 53.

> **Important**
>
> Jika Anda mengakhiri lingkungan, Anda juga harus menghapus pemetaan CNAME yang Anda buat, karena pelanggan lain dapat menggunakan kembali nama host yang tersedia. Pastikan untuk menghapus catatan DNS yang mengarah ke lingkungan Anda yang dihentikan untuk mencegah entri DNS yang menggantung. Entri DNS yang menggantung dapat mengekspos lalu lintas internet yang ditujukan untuk domain Anda ke kerentanan keamanan. Itu juga dapat menimbulkan risiko lain.
> Untuk informasi selengkapnya, lihat Perlindungan dari catatan delegasi yang menggantung di Route 53 di Panduan Pengembang Amazon Route 53. Anda juga dapat mempelajari selengkapnya tentang menggantung entri DNS di Perlindungan Domain yang Ditingkatkan untuk Permintaan CloudFront Amazon di AWS Blog Keamanan.

Mengonfigurasi lingkungan Elastic Beanstalk (lanjutan)

Saat Anda membuat AWS Elastic Beanstalk lingkungan, Elastic Beanstalk menyediakan dan mengonfigurasi semua AWS sumber daya yang diperlukan untuk menjalankan dan mendukung aplikasi Anda. Selain mengonfigurasi metadata lingkungan dan perilaku pembaruan, Anda dapat menyesuaikan sumber daya ini dengan memberikan nilai untuk opsi konfigurasi. Misalnya, Anda mungkin ingin menambahkan antrean Amazon SQS dan alarm pada kedalaman antrian, atau Anda mungkin ingin menambahkan kluster Amazon. ElastiCache

Sebagian besar opsi konfigurasi memiliki nilai default yang diterapkan secara otomatis oleh Elastic Beanstalk. Anda dapat mengganti default ini dengan file konfigurasi, konfigurasi tersimpan, opsi baris perintah, atau dengan langsung memanggil Elastic Beanstalk API. Konsol EB CLI dan Elastic Beanstalk juga menerapkan nilai yang direkomendasikan untuk beberapa opsi.

Anda dapat dengan mudah menyesuaikan lingkungan Anda pada saat yang sama saat Anda men-deploy versi aplikasi Anda dengan menyertakan file konfigurasi dengan paket sumber Anda. Saat menyesuaikan perangkat lunak pada instans Anda, lebih menguntungkan menggunakan file konfigurasi daripada membuat AMI khusus karena Anda tidak perlu mempertahankan satu set. AMIs

Ketika men-deploy aplikasi Anda, Anda mungkin ingin menyesuaikan dan mengonfigurasi perangkat lunak yang diandalkan aplikasi Anda. File-file ini dapat berupa dependensi yang dibutuhkan oleh aplikasi—sebagai contoh, paket tambahan dari repositori yum—atau mereka dapat berupa file konfigurasi seperti pengganti untuk httpd.conf untuk mengambil alih pengaturan tertentu yang di-default oleh AWS Elastic Beanstalk.

Topik

- Opsi konfigurasi
- Penyesuaian lingkungan lanjutan dengan file konfigurasi (.ebextensions)
- Menggunakan konfigurasi tersimpan Elastic Beanstalk
- Manifes lingkungan (env.yaml)
- Menggunakan image mesin Amazon kustom (AMI) di lingkungan Elastic Beanstalk Anda
- Menyajikan file statis
- Mengonfigurasi HTTPS untuk lingkungan Elastic Beanstalk Anda

Opsi konfigurasi

Elastic Beanstalk mendefinisikan sejumlah besar opsi konfigurasi yang dapat Anda gunakan untuk mengonfigurasi perilaku lingkungan Anda dan sumber daya yang ada di dalamnya. Opsi konfigurasi diatur ke dalam namespace seperti `aws:autoscaling:asg`, yang mendefinisikan opsi untuk grup Auto Scaling lingkungan.

Konsol Elastic Beanstalk dan EB CLI mengatur opsi konfigurasi ketika Anda membuat lingkungan, termasuk opsi yang Anda tetapkan secara eksplisit, dan nilai yang disarankan yang ditentukan oleh klien. Anda juga dapat mengatur opsi konfigurasi pada konfigurasi tersimpan dan file konfigurasi. Jika opsi yang sama diatur di beberapa lokasi, nilai yang digunakan ditentukan oleh urutan prioritas.

Pengaturan opsi konfigurasi dapat disusun dalam format teks dan disimpan sebelum pembuatan lingkungan, diterapkan selama pembuatan lingkungan menggunakan klien yang didukung, dan ditambahkan, dimodifikasi atau dihapus setelah penciptaan lingkungan. Untuk rincian dari semua metode yang tersedia untuk bekerja dengan opsi konfigurasi pada masing-masing dari tiga tahap ini, baca topik berikut:

- Menetapkan opsi konfigurasi sebelum pembuatan lingkungan
- Menetapkan opsi konfigurasi selama pembuatan lingkungan
- Menetapkan opsi konfigurasi setelah pembuatan lingkungan

Untuk daftar lengkap namespace dan pilihan, termasuk nilai default dan didukung untuk masing-masing tahap, lihat Opsi umum untuk semua lingkungan dan Opsi spesifik platform.

Precedence

Selama pembuatan lingkungan, opsi konfigurasi diterapkan dari berbagai sumber dengan prioritas sebagai berikut, dari ysng tertinggi ke terendah:

- Pengaturan diterapkan langsung ke lingkungan — Pengaturan yang ditentukan selama membuat lingkungan atau memperbarui operasi lingkungan pada Elastic Beanstalk API oleh klien mana pun, termasuk konsol Elastic Beanstalk, EB CLI, dan. AWS CLI SDKs Konsol Elastic Beanstalk dan EB CLI juga menerapkan nilai yang disarankan untuk beberapa opsi yang berlaku pada tingkat ini kecuali jika diganti.
- Konfigurasi tersimpan — Pengaturan untuk opsi apa pun yang tidak diterapkan secara langsung ke lingkungan dimuat dari konfigurasi tersimpan, jika ditentukan.

- File Konfigurasi (.ebextensions) — Pengaturan untuk opsi apa pun yang tidak diterapkan secara langsung ke lingkungan, dan juga tidak ditentukan dalam konfigurasi tersimpan, dimuat dari file konfigurasi di folder `.ebextensions` pada root paket sumber aplikasi.

  File konfigurasi dijalankan dalam urutan abjad. Misalnya, `.ebextensions/01run.config` dijalankan sebelum `.ebextensions/02do.config`.
- Nilai default — Jika opsi konfigurasi memiliki nilai default, itu hanya berlaku ketika opsi tidak diatur pada salah satu tingkat di atas.

Jika opsi konfigurasi yang sama ditentukan di lebih dari satu lokasi, pengaturan dengan prioritas tertinggi akan diterapkan. Ketika pengaturan diterapkan dari konfigurasi tersimpan atau pengaturan diterapkan langsung ke lingkungan, pengaturan disimpan sebagai bagian dari konfigurasi lingkungan. Pengaturan ini dapat dihapus dengan AWS CLI atau dengan EB CLI.

Pengaturan dalam file konfigurasi tidak diterapkan langsung ke lingkungan dan tidak dapat dihapus tanpa memodifikasi file konfigurasi dan men-deploy versi aplikasi baru. Jika pengaturan yang diterapkan dengan salah satu metode lain dihapus, pengaturan yang sama akan dimuat dari file konfigurasi pada paket sumber.

Misalnya, Anda menetapkan jumlah minimum instans di lingkungan Anda menjadi 5 selama pembuatan lingkungan, baik menggunakan konsol Elastic Beanstalk, opsi baris perintah, atau konfigurasi tersimpan. Paket sumber untuk aplikasi Anda juga termasuk file konfigurasi yang menetapkan jumlah minimum instans menjadi 2.

Ketika Anda membuat lingkungan, Elastic Beanstalk menetapkan opsi `MinSize` di namespace `aws:autoscaling:asg` menjadi 5. Jika Anda kemudian menghapus opsi dari konfigurasi lingkungan, nilai dalam file konfigurasi tersebut dimuat, dan jumlah minimum instans diatur menjadi 2. Jika Anda kemudian menghapus file konfigurasi dari paket sumber dan memindahkannya, Elastic Beanstalk menggunakan pengaturan default 1.

Nilai yang disarankan

Konsol Elastic Beanstalk Command Line Interface (EB CLI) dan Elastic Beanstalk memberikan nilai yang direkomendasikan untuk beberapa opsi konfigurasi. Nilai ini bisa berbeda dari nilai default dan ditetapkan pada tingkat API ketika lingkungan Anda dibuat. Nilai yang disarankan mengizinkan Elastic Beanstalk untuk meningkatkan konfigurasi lingkungan default tanpa membuat perubahan yang tidak kompatibel ke API.

Misalnya, konsol EB CLI dan Elastic Beanstalk menetapkan opsi EC2 konfigurasi untuk tipe instance (di namespace). `InstanceType` `aws:autoscaling:launchconfiguration` Setiap klien menyediakan cara yang berbeda untuk mengganti pengaturan default. Di konsol tersebut Anda dapat memilih jenis instans yang berbeda dari menu menurun di halaman Detail Konfigurasi pada wizard Buat Lingkungan Baru. Dengan EB CLI, Anda dapat menggunakan parameter `--instance_type` untuk eb create.

Karena nilai yang direkomendasikan ditetapkan pada tingkat API, mereka akan mengganti nilai untuk opsi yang sama yang Anda tetapkan dalam file konfigurasi atau konfigurasi tersimpan. Opsi berikut ditetapkan:

Konsol Elastic Beanstalk

- Namespace: `aws:autoscaling:launchconfiguration`

  Nama opsi: `IamInstanceProfile`, `EC2KeyName`, `InstanceType`
- Namespace: `aws:autoscaling:updatepolicy:rollingupdate`

  Nama opsi: `RollingUpdateType` dan `RollingUpdateEnabled`
- Namespace: `aws:elasticbeanstalk:application`

  Nama opsi: `Application Healthcheck URL`
- Namespace: `aws:elasticbeanstalk:command`

  Nama opsi: `DeploymentPolicy`, `BatchSize` dan `BatchSizeType`
- Namespace: `aws:elasticbeanstalk:environment`

  Nama opsi: `ServiceRole`
- Namespace: `aws:elasticbeanstalk:healthreporting:system`

  Nama opsi: `SystemType` dan `HealthCheckSuccessThreshold`
- Namespace: `aws:elasticbeanstalk:sns:topics`

  Nama opsi: `Notification Endpoint`
- Namespace: `aws:elasticbeanstalk:sqsd`

  Nama opsi: `HttpConnections`
- Namespace: `aws:elb:loadbalancer`

Nama opsi: `CrossZone`

- Namespace: `aws:elb:policies`

  Nama opsi: `ConnectionDrainingTimeout` dan `ConnectionDrainingEnabled`

EB CLI

- Namespace: `aws:autoscaling:launchconfiguration`

  Nama opsi: `IamInstanceProfile`, `InstanceType`
- Namespace: `aws:autoscaling:updatepolicy:rollingupdate`

  Nama opsi: `RollingUpdateType` dan `RollingUpdateEnabled`
- Namespace: `aws:elasticbeanstalk:command`

  Nama opsi: `BatchSize` dan `BatchSizeType`
- Namespace: `aws:elasticbeanstalk:environment`

  Nama opsi: `ServiceRole`
- Namespace: `aws:elasticbeanstalk:healthreporting:system`

  Nama opsi: `SystemType`
- Namespace: `aws:elb:loadbalancer`

  Nama opsi: `CrossZone`
- Namespace: `aws:elb:policies`

  Nama opsi: `ConnectionDrainingEnabled`

Menetapkan opsi konfigurasi sebelum pembuatan lingkungan

AWS Elastic Beanstalk mendukung sejumlah besar opsi konfigurasi yang memungkinkan Anda memodifikasi pengaturan yang diterapkan ke sumber daya di lingkungan Anda. Beberapa opsi ini memiliki nilai default yang dapat diganti untuk menyesuaikan lingkungan Anda. Opsi lain dapat dikonfigurasi untuk mengaktifkan fitur tambahan.

Elastic Beanstalk mendukung dua metode penyimpanan pengaturan opsi konfigurasi. File konfigurasi dalam format YAML atau JSON dapat disertakan dalam kode sumber aplikasi Anda pada direktori bernama `.ebextensions` dan di-deploy sebagai bagian dari paket sumber aplikasi Anda. Anda membuat dan mengelola file konfigurasi lokal.

Konfigurasi tersimpan adalah templat yang Anda buat dari lingkungan berjalan atau file opsi JSON dan simpan di Elastic Beanstalk. Konfigurasi tersimpan yang sudah ada juga dapat diperpanjang untuk membuat konfigurasi baru.

> **Note**
>
> Pengaturan yang ditentukan dalam file konfigurasi dan konfigurasi tersimpan memiliki prioritas lebih rendah daripada pengaturan yang dikonfigurasi selama atau setelah pembuatan lingkungan, termasuk nilai yang direkomendasikan yang diterapkan oleh konsol Elastic Beanstalk dan EB CLI. Lihat Precedence untuk rincian selengkapnya.

Opsi juga dapat ditentukan dalam dokumen JSON dan diberikan langsung ke Elastic Beanstalk ketika Anda membuat atau memperbarui lingkungan dengan EB CLI atau AWS CLI. Opsi yang diberikan langsung ke Elastic Beanstalk dengan cara ini mengganti semua metode lainnya.

Untuk daftar lengkap opsi yang tersedia, lihat Opsi konfigurasi.

Metode

- File konfigurasi (.ebextensions)
- Konfigurasi tersimpan
- Dokumen JSON
- Konfigurasi EB CLI

File konfigurasi (`.ebextensions`)

Gunakan `.ebextensions` untuk mengonfigurasi opsi yang diperlukan untuk membuat aplikasi Anda berfungsi, dan memberikan nilai default untuk opsi lain yang dapat diganti pada tingkat prioritasyang lebih tinggi. Opsi yang ditentukan di `.ebextensions` memiliki tingkat prioritas terendah dan diganti oleh pengaturan pada tingkat lainnya.

Untuk menggunakan file konfigurasi, buat folder dengan nama `.ebextensions` di tingkat atas kode sumber proyek Anda. Tambahkan file dengan ekstensi `.config` dan tentukan opsi dengan cara berikut:

```
option_settings:
  - namespace:  namespace
    option_name:  option name
    value:  option value
  - namespace:  namespace
    option_name:  option name
    value:  option value
```

Sebagai contoh, file konfigurasi berikut menetapkan url pemeriksaan kondisi aplikasi ke `/health`:

`healthcheckurl.config`

```
option_settings:
  - namespace:  aws:elasticbeanstalk:application
    option_name:  Application Healthcheck URL
    value:  /health
```

Di JSON:

```
{
 "option_settings" :
    [
      {
        "namespace" : "aws:elasticbeanstalk:application",
        "option_name" : "Application Healthcheck URL",
        "value" : "/health"
      }
    ]
}
```

Ini mengonfigurasi penyeimbang beban Elastic Load Balancing di lingkungan Elastic Beanstalk Anda untuk membuat permintaan HTTP ke `/health` jalur ke EC2 setiap instance untuk menentukan apakah itu sehat atau tidak.

> **Note**
>
> YAML bergantung pada indentasi yang konsisten. Cocokkan tingkat indentasi saat mengganti konten dalam file konfigurasi contoh dan pastikan bahwa editor teks Anda menggunakan spasi, bukan karakter tab, untuk indentasi.

Sertakan direktori `.ebextensions` di Paket Sumber Aplikasi Anda dan terapkan paket tersebut ke lingkungan Elastic Beanstalk baru atau yang sudah ada.

File konfigurasi mendukung beberapa bagian selain `option_settings` untuk menyesuaikan perangkat lunak dan file yang berjalan di server di lingkungan Anda. Untuk informasi selengkapnya, lihat .Ebextensions.

Konfigurasi tersimpan

Buat konfigurasi tersimpan untuk menyimpan pengaturan yang telah diterapkan ke lingkungan yang ada selama atau setelah pembuatan lingkungan dengan menggunakan konsol Elastic Beanstalk, EB CLI, atau AWS CLI. Konfigurasi tersimpan adalah milik aplikasi dan dapat diterapkan ke lingkungan baru atau yang sudah ada untuk aplikasi itu.

Klien

- Konsol Elastic Beanstalk
- EB CLI
- AWS CLI

Konsol Elastic Beanstalk

Untuk membuat konfigurasi tersimpan (konsol Elastic Beanstalk)

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pilih Tindakan, lalu pilih Simpan konfigurasi.
4. Gunakan kotak dialog pada layar untuk menyelesaikan tindakan.

Konfigurasi tersimpan yang disimpan di bucket Elastic Beanstalk S3 dalam folder yang dinamai berdasarkan aplikasi Anda. Sebagai contoh, konfigurasi untuk aplikasi yang dinamai `my-app` di wilayah us-barat-2 untuk nomor akun 123456789012 dapat ditemukan di `s3://elasticbeanstalk-us-west-2-123456789012/resources/templates/my-app`.

EB CLI

EB CLI juga menyediakan sub perintah untuk berinteraksi dengan konfigurasi tersimpan di bawah eb config:

Untuk membuat konfigurasi tersimpan (EB CLI)

1. Simpan konfigurasi lingkungan yang terlampir saat ini:

   ```
   ~/project$ eb config save --cfg my-app-v1
   ```

   EB CLI menyimpan konfigurasi untuk `~/project/.elasticbeanstalk/saved_configs/my-app-v1.cfg.yml`
2. Ubah konfigurasi tersimpan secara lokal jika diperlukan.
3. Muat konfigurasi tersimpan ke S3:

   ```
   ~/project$ eb config put my-app-v1
   ```

AWS CLI

Buat konfigurasi tersimpan dari lingkungan berjalan dengan `aws elasticbeanstalk create-configuration-template`

Untuk membuat konfigurasi tersimpan (AWS CLI)

1. Identifikasi ID lingkungan Elastic Beanstalk lingkungan Anda dengan `describe-environments`:

   ```
   $ aws elasticbeanstalk describe-environments --environment-name my-env
   {
       "Environments": [
           {
               "ApplicationName": "my-env",
               "EnvironmentName": "my-env",
   ```

```
            "VersionLabel": "89df",
            "Status": "Ready",
            "Description": "Environment created from the EB CLI using \"eb create
\"",
            "EnvironmentId": "e-vcghmm2zwk",
            "EndpointURL": "awseb-e-v-AWSEBLoa-1JUM8159RA11M-43V6ZI1194.us-
west-2.elb.amazonaws.com",
            "SolutionStackName": "64bit Amazon Linux 2015.03 v2.0.2 running Multi-
container Docker 1.7.1 (Generic)",
            "CNAME": "my-env-nfptuqaper.elasticbeanstalk.com",
            "Health": "Green",
            "AbortableOperationInProgress": false,
            "Tier": {
                "Version": " ",
                "Type": "Standard",
                "Name": "WebServer"
            },
            "HealthStatus": "Ok",
            "DateUpdated": "2015-10-01T00:24:04.045Z",
            "DateCreated": "2015-09-30T23:27:55.768Z"
        }
    ]
}
```

2. Simpan konfigurasi lingkungan saat ini dengan `create-configuration-template`:

```
$ aws elasticbeanstalk create-configuration-template --environment-id e-vcghmm2zwk
 --application-name my-app --template-name v1
```

Elastic Beanstalk menyimpan konfigurasi ke bucket Elastic Beanstalk Anda di Amazon S3.

Dokumen JSON

Jika Anda menggunakan AWS CLI untuk membuat dan memperbarui lingkungan, Anda juga dapat memberikan opsi konfigurasi dalam format JSON. Pustaka file konfigurasi di JSON berguna jika Anda menggunakan AWS CLI untuk membuat dan mengelola lingkungan.

Sebagai contoh, dokumen JSON berikut menetapkan url pemeriksaan kondisi aplikasi untuk `/health`:

~/ebconfigs/healthcheckurl.json

```
[
  {
    "Namespace": "aws:elasticbeanstalk:application",
    "OptionName": "Application Healthcheck URL",
    "Value": "/health"
  }
]
```

Konfigurasi EB CLI

Selain mendukung konfigurasi tersimpan dan konfigurasi lingkungan langsung dengan eb config perintah, EB CLI memiliki file konfigurasi dengan opsi bernama `default_ec2_keyname` yang dapat Anda gunakan untuk menentukan EC2 key pair Amazon untuk akses SSH ke instance di lingkungan Anda. EB CLI menggunakan opsi ini untuk mengatur opsi konfigurasi `EC2KeyName` di namespace `aws:autoscaling:launchconfiguration`.

~/ .yml workspace/my-app/.elasticbeanstalk/config

```
branch-defaults:
  master:
    environment: my-env
  develop:
    environment: my-env-dev
deploy:
  artifact: ROOT.war
global:
  application_name: my-app
  default_ec2_keyname: my-keypair
  default_platform: Tomcat 8 Java 8
  default_region: us-west-2
  profile: null
  sc: git
```

Menetapkan opsi konfigurasi selama pembuatan lingkungan

Saat Anda membuat AWS Elastic Beanstalk lingkungan dengan menggunakan konsol Elastic Beanstalk, EB AWS CLI CLI, SDK, atau Elastic Beanstalk API, Anda dapat memberikan nilai untuk opsi konfigurasi untuk menyesuaikan lingkungan Anda dan sumber daya yang diluncurkan di dalamnya. AWS

Untuk apa pun selain perubahan konfigurasi satu kali, Anda dapat menyimpan file konfigurasi secara lokal, di paket sumber Anda, atau di Amazon S3.

Topik ini mencakup prosedur untuk semua metode dalam mengatur opsi konfigurasi selama pembuatan lingkungan.

Klien

- Di konsol Elastic Beanstalk
- Menggunakan EB CLI
- Menggunakan AWS CLI

Di konsol Elastic Beanstalk

Ketika Anda membuat lingkungan Elastic Beanstalk di konsol Elastic Beanstalk, Anda dapat memberikan opsi konfigurasi menggunakan file konfigurasi, konfigurasi tersimpan, dan formulir di wizard Buat Lingkungan Baru.

Metode

- Menggunakan file konfigurasi (.ebextensions)
- Menggunakan konfigurasi tersimpan
- Menggunakan wizard lingkungan baru

Menggunakan file konfigurasi (`.ebextensions`)

Sertakan file `.config` di paket sumber aplikasi Anda di folder yang dinamai `.ebextensions`.

Untuk detail tentang file konfigurasi, lihat .Ebextensions.

```
~/workspace/my-app-v1.zip
|-- .ebextensions
|   |-- environmentvariables.config
|   `-- healthcheckurl.config
|-- index.php
`-- styles.css
```

Unggah paket sumber ke Elastic Beanstalk secara normal, selama Pembuatan lingkungan.

Konsol Elastic Beanstalk menerapkan nilai yang disarankan untuk beberapa opsi konfigurasi dan memiliki kolom formulir untuk orang lain. Opsi yang dikonfigurasi oleh konsol Elastic Beanstalk diterapkan langsung ke lingkungan dan mengganti pengaturan dalam file konfigurasi.

Menggunakan konfigurasi tersimpan

Saat Anda membuat lingkungan baru menggunakan konsol Elastic Beanstalk, salah satu langkah awalnya adalah memilih konfigurasi. Konfigurasi dapat berupa konfigurasi yang telah ditentukan sebelumnya, biasanya versi terbaru dari platform seperti PHP atau Tomcat, atau dapat berupa konfigurasi tersimpan.

Untuk menerapkan konfigurasi tersimpan selama pembuatan lingkungan (konsol Elastic Beanstalk)

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Pada panel navigasi, pilih Aplikasi, dan kemudian pilih nama aplikasi Anda dari daftar.
3. Pada panel navigasi, cari nama aplikasi dan pilih Konfigurasi tersimpan.
4. Pilih konfigurasi tersimpan yang ingin Anda terapkan, lalu pilih Luncurkan lingkungan.
5. Lanjutkan melalui wizard untuk membuat lingkungan Anda.

Konfigurasi tersimpan hanya khusus aplikasi. Lihat Konfigurasi tersimpan untuk detail tentang cara membuat konfigurasi tersimpan.

Menggunakan wizard lingkungan baru

Sebagian besar opsi konfigurasi standar disajikan pada halaman Konfigurasikan opsi lainnya di wizard Buat Lingkungan Baru. Jika Anda membuat basis data Amazon RDS atau mengonfigurasi VPC untuk lingkungan Anda, opsi konfigurasi tambahan tersedia untuk sumber daya tersebut.

Untuk mengatur opsi konfigurasi selama pembuatan lingkungan (konsol Elastic Beanstalk)

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Pada panel navigasi, pilih Aplikasi.
3. Pilih atau buat Aplikasi.
4. Pilih Tindakan, lalu pilih Buat lingkungan.
5. Lanjutkan melalui wizard, dan pilih Konfigurasi opsi lainnya.

6. Pilih salah satu dari preset konfigurasi, lalu pilih Edit di satu atau lebih kategori konfigurasi untuk mengubah grup opsi konfigurasi terkait.
7. Setelah selesai membuat pilihan opsi, pilih Buat lingkungan.

Pilihan apa pun yang Anda tetapkan di wizard lingkungan baru ditetapkan secara langsung pada lingkungan dan mengganti pengaturan opsi apa pun dalam konfigurasi tersimpan atau file konfigurasi (`.ebextensions`) yang Anda terapkan. Anda dapat menghapus pengaturan setelah lingkungan dibuat menggunakan EB CLI atau AWS CLI untuk mengizinkan pengaturan dalam konfigurasi tersimpan atau file konfigurasi muncul.

Untuk detail tentang wizard lingkungan baru, lihat Wizard pembuatan lingkungan baru.

Menggunakan EB CLI

Metode

- Menggunakan file konfigurasi (.ebextensions)
- Menggunakan konfigurasi tersimpan
- Menggunakan opsi baris perintah

Menggunakan file konfigurasi (`.ebextensions`)

Sertakan file `.config` dalam folder proyek Anda di bawah `.ebextensions` untuk men-deploy filenya dengan kode aplikasi Anda.

Untuk detail tentang file konfigurasi, lihat .Ebextensions.

```
~/workspace/my-app/
|-- .ebextensions
|   |-- environmentvariables.config
|   `-- healthcheckurl.config
|-- .elasticbeanstalk
|   `-- config.yml
|-- index.php
`-- styles.css
```

Buat lingkungan Anda dan terapkan kode sumber Anda ke lingkungan tersebut dengan eb create.

```
~/workspace/my-app$ eb create my-env
```

Menggunakan konfigurasi tersimpan

Untuk menerapkan konfigurasi tersimpan saat Anda membuat lingkungan dengan eb create, gunakan opsi `--cfg` tersebut.

```
~/workspace/my-app$ eb create --cfg savedconfig
```

Anda dapat menyimpan konfigurasi tersimpan di folder proyek Anda atau di lokasi penyimpanan Elastic Beanstalk Anda di Amazon S3. Pada contoh sebelumnya, EB CLI pertama mencari file konfigurasi tersimpan yang dinamai `savedconfig.cfg.yml` di folder `.elasticbeanstalk/saved_configs/`. Jangan sertakan ekstensi nama file (`.cfg.yml`) saat menerapkan konfigurasi tersimpan dengan `--cfg`.

```
~/workspace/my-app/
|-- .ebextensions
|   `-- healthcheckurl.config
|-- .elasticbeanstalk
|   |-- saved_configs
|   |   `-- savedconfig.cfg.yml
|   `-- config.yml
|-- index.php
`-- styles.css
```

Jika EB CLI tidak menemukan konfigurasi lokal, EB CLI akan terlihat di lokasi penyimpanan Elastic Beanstalk di Amazon S3. Untuk detail tentang cara membuat, mengedit, dan mengunggah konfigurasi tersimpan, lihat Konfigurasi tersimpan.

Menggunakan opsi baris perintah

Perintah eb create EB CLI memiliki beberapa opsi yang dapat Anda gunakan untuk mengatur opsi konfigurasi selama pembuatan lingkungan. Anda dapat menggunakan opsi ini untuk menambahkan basis data RDS untuk lingkungan Anda, mengonfigurasi VPC, atau mengganti nilai yang disarankan.

Sebagai contoh, EB CLI menggunakan tipe instans `t2.micro` secara default. Untuk memilih tipe instans yang berbeda, gunakan opsi `--instance_type` tersebut.

```
$ eb create my-env --instance_type t2.medium
```

Untuk membuat instans basis data Amazon RDS dan melampirkannya ke lingkungan Anda, gunakan opsi `--database` tersebut.

```
$ eb create --database.engine postgres --database.username dbuser
```

Jika Anda menghapus nama lingkungan, kata sandi basis data atau parameter lain yang diperlukan untuk membuat lingkungan Anda, EB CLI meminta Anda untuk memasukkannya.

Lihat eb create untuk daftar lengkap opsi dan contoh penggunaan yang tersedia.

Menggunakan AWS CLI

Saat Anda menggunakan `create-environment` perintah untuk membuat lingkungan Elastic Beanstalk AWS CLI dengan AWS CLI , tidak menerapkan nilai yang disarankan. Semua opsi konfigurasi ditentukan di file konfigurasi dalam paket sumber yang Anda tentukan.

Metode

- Menggunakan file konfigurasi (.ebextensions)
- Menggunakan konfigurasi tersimpan
- Menggunakan opsi baris perintah

Menggunakan file konfigurasi (**`.ebextensions`**)

Untuk menerapkan file konfigurasi ke lingkungan yang Anda buat dengan AWS CLI, sertakan dalam bundel sumber aplikasi yang Anda unggah ke Amazon S3.

Untuk detail tentang file konfigurasi, lihat .Ebextensions.

```
~/workspace/my-app-v1.zip
|-- .ebextensions
|   |-- environmentvariables.config
|   `-- healthcheckurl.config
|-- index.php
`-- styles.css
```

Untuk mengunggah bundel sumber aplikasi dan membuat lingkungan dengan AWS CLI

1. Jika Anda belum memiliki bucket Elastic Beanstalk di Amazon S3, buat satu dengan `create-storage-location`.

   ```
   $ aws elasticbeanstalk create-storage-location
   {
   ```

```
    "S3Bucket": "elasticbeanstalk-us-west-2-123456789012"
}
```

2. Unggah paket sumber aplikasi Anda ke Amazon S3.

```
$ aws s3 cp sourcebundle.zip s3://elasticbeanstalk-us-west-2-123456789012/my-app/
sourcebundle.zip
```

3. Buat versi aplikasi.

```
$ aws elasticbeanstalk create-application-version --application-name my-app --
version-label v1 --description MyAppv1 --source-bundle S3Bucket="elasticbeanstalk-
us-west-2-123456789012",S3Key="my-app/sourcebundle.zip" --auto-create-application
```

4. Buat lingkungan.

```
$ aws elasticbeanstalk create-environment --application-name my-app --environment-
name my-env --version-label v1 --solution-stack-name "64bit Amazon Linux 2015.03
 v2.0.0 running Tomcat 8 Java 8"
```

Menggunakan konfigurasi tersimpan

Untuk menerapkan konfigurasi tersimpan ke lingkungan selama pembuatan, gunakan parameter `--template-name`.

```
$ aws elasticbeanstalk create-environment --application-name my-app --environment-name
 my-env --template-name savedconfig --version-label v1
```

Bila Anda menentukan konfigurasi tersimpan, jangan menentukan nama tumpukan solusi. Konfigurasi tersimpan sudah menentukan tumpukan solusi dan Elastic Beanstalk akan mengembalikan kesalahan jika Anda mencoba untuk menggunakan kedua pilihan.

Menggunakan opsi baris perintah

Gunakan parameter `--option-settings` untuk menentukan opsi konfigurasi dalam format JSON.

```
$ aws elasticbeanstalk create-environment --application-name my-app --environment-name
 my-env --version-label v1 --template-name savedconfig --option-settings '[
  {
    "Namespace": "aws:elasticbeanstalk:application",
```

```
    "OptionName": "Application Healthcheck URL",
    "Value": "/health"
  }
]
```

Untuk memuat JSON dari file, gunakan prefiks `file://` tersebut.

```
$ aws elasticbeanstalk create-environment --application-name my-app --environment-
name my-env --version-label v1 --template-name savedconfig --option-settings file://
healthcheckurl.json
```

Elastic Beanstalk menerapkan pengaturan opsi yang Anda tentukan dengan opsi `--option-settings` secara langsung ke lingkungan Anda. Jika pilihan yang sama ditentukan dalam konfigurasi disimpan atau file konfigurasi, `--option-settings` mengganti nilai-nilai tersebut.

Menetapkan opsi konfigurasi setelah pembuatan lingkungan

Anda dapat mengubah pengaturan opsi pada lingkungan berjalan dengan menerapkan konfigurasi tersimpan, mengunggah paket sumber baru dengan file konfigurasi (`.ebextensions`), atau menggunakan dokumen JSON. Konsol EB CLI dan Elastic Beanstalk juga memiliki fungsi klien khusus untuk mengatur dan memperbarui opsi konfigurasi.

Ketika Anda mengatur atau mengubah opsi konfigurasi, Anda dapat memicu pembaruan lingkungan penuh, tergantung pada tingkat kepelikan perubahan. Misalnya, perubahan pada opsi di `aws:autoscaling:launchconfiguration`, sepertiInstanceType, mengharuskan EC2 instans Amazon di lingkungan Anda direvisi. Ini memicu pembaruan bergulir. Perubahan konfigurasi lainnya dapat diterapkan tanpa gangguan atau penyediaan ulang.

Anda dapat menghapus pengaturan opsi dari lingkungan dengan EB CLI AWS CLI atau perintah. Menghapus opsi yang telah ditetapkan secara langsung di lingkungan pada tingkat API memungkinkan pengaturan dalam file konfigurasi, yang jika tidak tertutup oleh pengaturan yang diterapkan langsung ke lingkungan, akan muncul dan terpengaruh.

Pengaturan dalam konfigurasi tersimpan dan file konfigurasi dapat diganti dengan menetapkan opsi yang sama secara langsung pada lingkungan dengan salah satu metode konfigurasi lainnya. Namun, ini hanya dapat dihapus sepenuhnya dengan menerapkan konfigurasi tersimpan atau file konfigurasi yang diperbarui. Ketika opsi tidak diatur dalam konfigurasi tersimpan, dalam file konfigurasi, atau langsung pada lingkungan, nilai default berlaku, jika ada. Lihat Precedence untuk rincian selengkapnya.

Klien

- Konsol Elastic Beanstalk
- EB CLI
- The AWS CLI

Konsol Elastic Beanstalk

Anda dapat memperbarui pengaturan opsi konfigurasi di konsol Elastic Beanstalk dengan men-deploy paket sumber aplikasi yang berisi file konfigurasi, menerapkan konfigurasi tersimpan, atau memodifikasi lingkungan secara langsung dengan halaman Konfigurasi di konsol manajemen lingkungan.

Metode

- Menggunakan file konfigurasi (.ebextensions)
- Menggunakan konfigurasi tersimpan
- Menggunakan konsol Elastic Beanstalk

Menggunakan file konfigurasi (`.ebextensions`)

Memperbarui file konfigurasi di direktori sumber Anda, buat paket sumber baru, dan deploy versi baru ke lingkungan Elastic Beanstalk untuk menerapkan perubahan.

Untuk detail tentang file konfigurasi, lihat .Ebextensions.

Untuk men-deploy paket sumber

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Di halaman gambaran umum lingkungan, pilih Unggah dan deploy.
4. Gunakan kotak dialog di layar untuk mengunggah paket sumber.
5. Pilih Deploy.
6. Ketika deployment selesai, Anda dapat memilih URL situs untuk membuka situs web Anda di tab baru.

Perubahan yang dibuat pada file konfigurasi tidak akan mengganti pengaturan opsi dalam konfigurasi tersimpan atau pengaturan yang diterapkan langsung ke lingkungan pada tingkat API. Lihat Prioritas untuk detailnya.

Menggunakan konfigurasi tersimpan

Terapkan konfigurasi tersimpan ke lingkungan berjalan untuk menerapkan pengaturan opsi yang ditentukan.

Untuk menerapkan konfigurasi tersimpan untuk lingkungan berjalan (konsol Elastic Beanstalk)

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Pada panel navigasi, pilih Aplikasi, dan kemudian pilih nama aplikasi Anda dari daftar.
3. Pada panel navigasi, cari nama aplikasi dan pilih Konfigurasi tersimpan.
4. Pilih konfigurasi tersimpan yang ingin Anda terapkan, lalu pilih Muatan.
5. Pilih lingkungan, dan kemudian pilih Muatan.

Pengaturan yang ditentukan dalam konfigurasi tersimpan mengganti pengaturan dalam file konfigurasi, dan diganti oleh pengaturan yang dikonfigurasi menggunakan konsol manajemen lingkungan.

Lihat Konfigurasi tersimpan untuk detail tentang cara membuat konfigurasi tersimpan.

Menggunakan konsol Elastic Beanstalk

Konsol Elastic Beanstalk menyajikan banyak pilihan konfigurasi pada halaman Konfigurasi untuk setiap lingkungan.

Untuk mengubah opsi konfigurasi pada lingkungan berjalan (konsol Elastic Beanstalk)

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Temukan halaman konfigurasi yang ingin Anda edit:
   - Jika Anda melihat opsi yang Anda minati, atau Anda tahu kategori konfigurasinya, pilih Edit di kategori konfigurasi untuknya.

- Untuk mencari opsi, aktifkan Tampilan Tabel, lalu masukkan istilah pencarian ke dalam kotak pencarian. Saat Anda mengetik, daftar akan semakin sedikit dan hanya menampilkan opsi yang sesuai dengan istilah penelusuran Anda.

  Bila Anda melihat pilihan yang Anda cari, pilih Edit di kategori konfigurasi yang berisi opsi tersebut.



5. Ubah pengaturan, dan kemudian pilih Simpan.
6. Ulangi dua langkah sebelumnya pada kategori konfigurasi tambahan, sesuai kebutuhan.
7. Pilih Terapkan.

Perubahan yang dibuat untuk opsi konfigurasi di konsol manajemen lingkungan diterapkan langsung ke lingkungan. Perubahan ini mengganti pengaturan untuk opsi yang sama dalam file konfigurasi atau konfigurasi tersimpan. Untuk detailnya, lihat Prioritas.

Untuk detail tentang mengubah opsi konfigurasi pada lingkungan berjalan menggunakan konsol Elastic Beanstalk, lihat topik di bawah Mengonfigurasi lingkungan Elastic Beanstalk.

EB CLI

Anda dapat memperbarui pengaturan opsi konfigurasi dengan EB CLI dengan men-deploy kode sumber yang berisi file konfigurasi, menerapkan pengaturan dari konfigurasi tersimpan, atau memodifikasi konfigurasi lingkungan secara langsung dengan perintah eb config.

Metode

- Menggunakan file konfigurasi (.ebextensions)
- Menggunakan konfigurasi tersimpan

- Menggunakan eb config
- Menggunakan eb setenv

Menggunakan file konfigurasi (`.ebextensions`)

Sertakan file `.config` dalam folder proyek Anda di bawah `.ebextensions` untuk men-deploy filenya dengan kode aplikasi Anda.

Untuk detail tentang file konfigurasi, lihat .Ebextensions.

```
~/workspace/my-app/
|-- .ebextensions
|   |-- environmentvariables.config
|   `-- healthcheckurl.config
|-- .elasticbeanstalk
|   `-- config.yml
|-- index.php
`-- styles.css
```

Terapkan kode sumber Anda dengan eb deploy.

```
~/workspace/my-app$ eb deploy
```

Menggunakan konfigurasi tersimpan

Anda dapat menggunakan perintah eb config untuk menerapkan konfigurasi tersimpan ke lingkungan berjalan. Gunakan opsi `--cfg` dengan nama konfigurasi yang tersimpan untuk menerapkan pengaturan ke lingkungan Anda.

```
$ eb config --cfg v1
```

Pada contoh ini, `v1` adalah nama file konfigurasi yang telah dibuat dan disimpan sebelumnya.

Pengaturan yang diterapkan ke lingkungan dengan perintah ini mengganti pengaturan yang diterapkan selama pembuatan lingkungan, dan pengaturan ditentukan di file konfigurasi pada paket sumber aplikasi Anda.

Menggunakan eb config

Perintah eb config EB CLI memungkinkan Anda mengatur dan menghapus pengaturan opsi secara langsung pada lingkungan dengan menggunakan editor teks.

Ketika Anda menjalankan eb config, EB CLI menampilkan pengaturan yang diterapkan ke lingkungan Anda dari semua sumber, termasuk file konfigurasi, konfigurasi tersimpan, nilai yang disarankan, opsi yang diatur langsung di lingkungan, dan default API.

> **Note**
>
> eb config tidak menampilkan properti lingkungan. Untuk mengatur properti lingkungan yang dapat Anda baca dari dalam aplikasi Anda, gunakan eb setenv.

Contoh berikut menunjukkan pengaturan yang diterapkan di namespace `aws:autoscaling:launchconfiguration`. Pengaturan ini meliputi:

- Dua nilai yang direkomendasikan, untuk `IamInstanceProfile` dan `InstanceType`, diterapkan oleh EB CLI selama pembuatan lingkungan.
- Opsi `EC2KeyName`, diatur langsung pada lingkungan selama pembuatan berdasarkan konfigurasi repositori.
- Nilai default API untuk opsi lainnya.

```
ApplicationName: tomcat
DateUpdated: 2015-09-30 22:51:07+00:00
EnvironmentName: tomcat
SolutionStackName: 64bit Amazon Linux 2015.03 v2.0.1 running Tomcat 8 Java 8
settings:
...
aws:autoscaling:launchconfiguration:
    BlockDeviceMappings: null
    EC2KeyName: my-key
    IamInstanceProfile: aws-elasticbeanstalk-ec2-role
    ImageId: ami-1f316660
    InstanceType: t2.micro
...
```

Untuk mengatur atau mengubah opsi konfigurasi dengan eb config

1. Jalankan eb config untuk melihat konfigurasi lingkungan Anda.

   ```
   ~/workspace/my-app/$ eb config
   ```

2. Ubah salah satu nilai pengaturan menggunakan editor teks default.

```
aws:autoscaling:launchconfiguration:
    BlockDeviceMappings: null
    EC2KeyName: my-key
    IamInstanceProfile: aws-elasticbeanstalk-ec2-role
    ImageId: ami-1f316660
    InstanceType: t2.medium
```

3. Simpan file konfigurasi sementara dan keluar.
4. EB CLI memperbarui konfigurasi lingkungan Anda.

Menetapkan opsi konfigurasi dengan eb config mengganti pengaturan dari semua sumber lainnya.

Anda juga dapat menghapus opsi dari lingkungan Anda dengan eb config.

Untuk menghapus opsi konfigurasi (EB CLI)

1. Jalankan eb config untuk melihat konfigurasi lingkungan Anda.

```
~/workspace/my-app/$ eb config
```

2. Ganti nilai apa pun yang ditampilkan dengan `null` string. Anda juga dapat menghapus seluruh baris yang berisi opsi yang ingin Anda hapus.

```
aws:autoscaling:launchconfiguration:
    BlockDeviceMappings: null
    EC2KeyName: my-key
    IamInstanceProfile: aws-elasticbeanstalk-ec2-role
    ImageId: ami-1f316660
    InstanceType: null
```

3. Simpan file konfigurasi sementara dan keluar.
4. EB CLI memperbarui konfigurasi lingkungan Anda.

Menghapus opsi dari lingkungan Anda dengan eb config mengizinkan pengaturan untuk pilihan yang sama muncul dari file konfigurasi dalam paket sumber aplikasi Anda. Lihat Prioritas untuk detailnya.

Menggunakan eb setenv

Untuk mengatur properti lingkungan dengan EB CLI, gunakan eb setenv.

```
~/workspace/my-app/$ eb setenv ENVVAR=TEST
INFO: Environment update is starting.
INFO: Updating environment my-env's configuration settings.
INFO: Environment health has transitioned from Ok to Info. Command is executing on all
 instances.
INFO: Successfully deployed new configuration to environment.
```

Perintah ini menetapkan properti lingkungan di namespace `aws:elasticbeanstalk:application:environment`. Properti lingkungan yang diatur dengan eb setenv tersedia untuk aplikasi Anda setelah proses pembaruan singkat.

Lihat properti lingkungan yang ditetapkan di lingkungan Anda dengan eb printenv.

```
~/workspace/my-app/$ eb printenv
 Environment Variables:
     ENVVAR = TEST
```

The AWS CLI

Anda dapat memperbarui pengaturan opsi konfigurasi dengan menggunakan bundel sumber yang berisi file konfigurasi, menerapkan konfigurasi tersimpan yang disimpan dari jarak jauh, atau memodifikasi lingkungan secara langsung dengan perintah. AWS CLI `aws elasticbeanstalk update-environment`

Metode

- Menggunakan file konfigurasi (.ebextensions)
- Menggunakan konfigurasi tersimpan
- Menggunakan opsi baris perintah

Menggunakan file konfigurasi (`.ebextensions`)

Untuk menerapkan file konfigurasi ke lingkungan yang sedang berjalan dengan AWS CLI, sertakan dalam bundel sumber aplikasi yang Anda unggah ke Amazon S3.

Untuk detail tentang file konfigurasi, lihat .Ebextensions.

```
~/workspace/my-app-v1.zip
|-- .ebextensions
|   |-- environmentvariables.config
```

```
|   `-- healthcheckurl.config
|-- index.php
`-- styles.css
```

Untuk mengunggah paket sumber aplikasi dan menerapkannya ke lingkungan berjalan (AWS CLI)

1. Jika Anda belum memiliki bucket Elastic Beanstalk di Amazon S3, buat satu dengan `create-storage-location`:

   ```
   $ aws elasticbeanstalk create-storage-location
   {
       "S3Bucket": "elasticbeanstalk-us-west-2-123456789012"
   }
   ```

2. Unggah paket sumber aplikasi Anda ke Amazon S3.

   ```
   $ aws s3 cp sourcebundlev2.zip s3://elasticbeanstalk-us-west-2-123456789012/my-app/
   sourcebundlev2.zip
   ```

3. Buat versi aplikasi.

   ```
   $ aws elasticbeanstalk create-application-version --application-
   name my-app --version-label v2 --description MyAppv2 --source-bundle
    S3Bucket="elasticbeanstalk-us-west-2-123456789012",S3Key="my-app/
   sourcebundlev2.zip"
   ```

4. Perbarui lingkungan.

   ```
   $ aws elasticbeanstalk update-environment --environment-name my-env --version-label
    v2
   ```

Menggunakan konfigurasi tersimpan

Anda dapat menerapkan konfigurasi tersimpan ke lingkungan berjalan dengan opsi `--template-name` pada perintah `aws elasticbeanstalk update-environment`.

Konfigurasi tersimpan harus berada di bucket Elastic Beanstalk Anda di jalur yang dinamai berdasarkan aplikasi Anda di bawah `resources/templates`. Misalnya, templat `v1` untuk aplikasi `my-app` di Wilayah US West (Oregon) (us-west-2) untuk akun 123456789012 terletak di `s3://elasticbeanstalk-us-west-2-123456789012/resources/templates/my-app/v1`

Untuk menerapkan konfigurasi tersimpan ke lingkungan berjalan (AWS CLI)

- Tentukan konfigurasi yang simpan dalam panggilan `update-environment` dengan opsi `--template-name`.

```
$ aws elasticbeanstalk update-environment --environment-name my-env --template-name v1
```

Elastic Beanstalk menempatkan konfigurasi tersimpan di lokasi ini ketika Anda membuatnya dengan `aws elasticbeanstalk create-configuration-template`. Anda juga dapat memodifikasi konfigurasi tersimpan secara lokal dan menempatkannya di lokasi ini sendiri.

Menggunakan opsi baris perintah

Untuk mengubah opsi konfigurasi dengan dokumen JSON (AWS CLI)

1. Tentukan pengaturan opsi Anda dalam format JSON dalam file lokal.
2. Jalankan `update-environment` dengan opsi `--option-settings`.

```
$ aws elasticbeanstalk update-environment --environment-name my-env --option-settings file://~/ebconfigs/as-zero.json
```

Pada contoh ini, `as-zero.json` menentukan opsi yang mengonfigurasi lingkungan dengan minimum dan maksimum dari nol instans. Ini menghentikan instans di lingkungan tanpa mengakhiri lingkungan.

`~/ebconfigs/as-zero.json`

```
[
    {
        "Namespace": "aws:autoscaling:asg",
        "OptionName": "MinSize",
        "Value": "0"
    },
    {
        "Namespace": "aws:autoscaling:asg",
        "OptionName": "MaxSize",
        "Value": "0"
    },
```

```
    {
        "Namespace": "aws:autoscaling:updatepolicy:rollingupdate",
        "OptionName": "RollingUpdateEnabled",
        "Value": "false"
    }
]
```

> **Note**
>
> Menetapkan opsi konfigurasi dengan `update-environment` mengganti pengaturan dari semua sumber lainnya.

Anda juga dapat menghapus opsi dari lingkungan Anda dengan `update-environment`.

Untuk menghapus opsi konfigurasi (AWS CLI)

- Jalankan perintah `update-environment` dengan opsi `--options-to-remove`.

```
$ aws elasticbeanstalk update-environment --environment-name my-env --options-to-
remove Namespace=aws:autoscaling:launchconfiguration,OptionName=InstanceType
```

Menghapus opsi dari lingkungan Anda dengan `update-environment` mengizinkan pengaturan untuk pilihan yang sama muncul dari file konfigurasi dalam paket sumber aplikasi Anda. Jika opsi tidak dikonfigurasi menggunakan salah satu metode ini, nilai default API akan berlaku, jika ada. Lihat Prioritas untuk detailnya.

Opsi umum untuk semua lingkungan

Namespace

- aws:autoscaling:asg
- aws:autoscaling:launchconfiguration
- aws:autoscaling:scheduledaction
- aws:autoscaling:trigger
- aws:autoscaling:updatepolicy:rollingupdate
- aws:ec2:instances
- aws:ec2:vpc

- aws:elasticbeanstalk:application
- aws:elasticbeanstalk:application:environment
- aws:elasticbeanstalk:aplikasi:environmentsecrets
- aws:elasticbeanstalk:cloudwatch:logs
- aws:elasticbeanstalk:cloudwatch:logs:health
- aws:elasticbeanstalk:command
- aws:elasticbeanstalk:environment
- aws:elasticbeanstalk:environment:process:default
- aws:elasticbeanstalk:environment:process:process_name
- aws:elasticbeanstalk:environment:proxy:staticfiles
- aws:elasticbeanstalk:healthreporting:system
- aws:elasticbeanstalk:hostmanager
- aws:elasticbeanstalk:managedactions
- aws:elasticbeanstalk:manageactions:platformupdate
- aws:elasticbeanstalk:monitoring
- aws:elasticbeanstalk:sns:topic
- aws:elasticbeanstalk:sqsd
- aws:elasticbeanstalk:trafficsplitting
- aws:elasticbeanstalk:xray
- aws:elb:healthcheck
- aws:elb:loadbalancer
- aws:elb:listener
- aws:elb:listener:listener_port
- aws:elb:policies
- aws:elb:policies:policy_name
- aws:elbv2:listener:default
- aws:elbv2:listener:listener_port
- aws:elbv2:listener:rule_name
- aws:elbv2:loadbalancer
- aws:rds:dbinstance

aws:autoscaling:asg

Konfigurasikan grup Auto Scaling lingkungan Anda. Untuk informasi selengkapnya, lihat the section called “Grup Auto Scaling”.

Namespace: **`aws:autoscaling:asg`**

Nama	Deskripsi	Default	Nilai yang valid

Availability Zones	Availability Zones (AZs) adalah lokasi berbeda dalam AWS Wilayah yang direkayasa untuk diisolasi dari kegagalan di wilayah lain AZs. Mereka menyediakan konektivitas jaringan latensi rendah yang murah ke yang lain AZs di Wilayah yang sama. Pilih jumlah AZs untuk instans Anda.	`Any`	`Any`
			`Any 1`
			`Any 2`
			`Any 3`

Cooldown	Periode Cooldown membantu mencegah Amazon EC2 Auto Scaling memulai aktivitas penskalaan tambahan sebelum efek aktivitas sebelumnya terlihat. Periode jeda adalah jumlah waktu, dalam detik, setelah aktivitas penskalaan masuk selesai sebelum aktivitas penskalaan masuk lainnya dapat dimulai.	`360`	`0` untuk `10000`

Custom Availability Zones	Tentukan AZs untuk instance Anda.	Tidak ada	`us-east-1a`
			`us-east-1b`
			`us-east-1c`
			`us-east-1d`
			`us-east-1e`
			`eu-central-1`

Nama	Deskripsi	Default	Nilai yang valid

EnableCapacityRebalancing	Menentukan apakah akan mengaktifkan fitur Rebalancing Kapasitas untuk Instans Spot di Grup Auto Scaling Anda. Untuk informasi selengkapnya, lihat Penyeimbangan Kembali Kapasitas di Panduan Pengguna Auto EC2 Scaling Amazon.
	
	Opsi ini hanya relevan jika `EnableSpot` disetel ke `true` dalam `aws:ec2:instances` namespace, dan setidaknya ada satu Instance Spot di grup Auto Scaling Anda.
		`false`	`true`
			
			`false`

MinSize	Jumlah minimum instans yang diinginkan dalam grup Auto Scaling Anda.	1	1 untuk `10000`
MaxSize	Jumlah maksimum instans yang diinginkan dalam grup Auto Scaling Anda.	4	1 untuk `10000`

aws:autoscaling:launchconfiguration

Konfigurasikan instans Amazon Elastic Compute Cloud (Amazon EC2) untuk lingkungan Anda.

Instans yang digunakan untuk lingkungan Anda dibuat menggunakan templat EC2 peluncuran Amazon atau sumber daya konfigurasi peluncuran grup Auto Scaling. Opsi berikut bekerja dengan kedua jenis sumber daya ini.

Untuk informasi selengkapnya, lihat the section called “ EC2 Contoh Amazon”. Anda juga dapat mereferensikan informasi selengkapnya tentang Amazon Elastic Block Store (EBS) di Amazon EBS chapter di Panduan Pengguna Amazon. EC2

Namespace: **`aws:autoscaling:launchconfiguration`**

Nama	Deskripsi	Default	Nilai yang valid

DisableDefaultEC2SecurityGroup	Ketika diatur ke nilai default`false`, Elastic Beanstalk membuat grup keamanan default yang memungkinkan lalu lintas dari internet atau penyeimbang beban pada port standar untuk HTTP (80). Ini melampirkan grup keamanan ini ke EC2 contoh lingkungan ketika menciptakan lingkungan.
	
	Ketika diatur ke `true` Elastic Beanstalk tidak akan menetapkan grup keamanan default ke instance EC2 untuk lingkungan baru. Untuk lingkungan yang ada, Elastic Beanstalk akan membatalkan penetapan grup keamanan default EC2 dari instance lingkungan Anda. EC2 Akibatnya, Anda juga harus mengatur konfigurasi berikut:
	
	• `SecurityGroups` Opsi namespace ini akan memerlukan setidaknya satu nilai untuk menentukan grup keamanan kustom Anda.
	• Untuk lingkungan dengan penyeimbang beban, Anda juga perlu mengatur `SecurityGroups` opsi di namespace lain untuk mengonfigurasi grup keamanan khusus untuk penyeimbang beban. Untuk penyeimbang beban aplikasi, atur opsi di namespace. aws:elbv2:loadbalancer Untuk penyeimbang
		`false`	`true`
			
			`false`

Nama	Deskripsi	Default	Nilai yang valid
	beban klasik, atur opsi di namespace . aws:elb:loadbalancer • Untuk informasi selengkapnya, lihat Mengelola kelompok EC2 keamanan. Jika nilai ditentukan untuk EC2KeyName dalam lingkungan yang telah `DisableDefaultEC2SecurityGroup` disetel ke `true` grup keamanan default tidak akan dikaitkan dengan EC2 instance.		

Nama	Deskripsi	Default	Nilai yang valid

DisableIMDSv1	Setel `true` untuk menonaktifkan Layanan Metadata Instance Versi 1 (IMDSv1) dan menegakkan. IMDSv2
	
	Setel `false` untuk mengaktifkan keduanya IMDSv1 dan IMDSv2.
	
	Instans untuk lingkungan Anda default sebagai berikut, berdasarkan sistem operasi platform:
	
	• Server Windows, AL2 dan sebelumnya - aktifkan keduanya IMDSv1 dan IMDSv2 (DisableI MDSv1default ke) `false`
	• AL2023 - aktifkan saja IMDSv2 (DisableIMDSv1default ke) `true`
	
	Untuk informasi selengkapnya, lihat Mengonfigurasi layanan metadata instans.
	
	Important
	Pengaturan opsi ini dapat menyebabkan Elastic Beanstalk memigrasikan lingkungan yang ada dengan konfigurasi peluncuran untuk meluncurkan templat. Melakukannya memerluka n izin yang diperlukan untuk mengelola templat peluncura n. Izin ini disertakan dalam kebijakan terkelola kami. Jika
		`false`— platform berbasis server Windows, Amazon Linux 2 dan sebelumnya
		
		`true`— platform berbasis Amazon Linux 2023
			`true`
			
			`false`

Nama	Deskripsi	Default	Nilai yang valid
	Anda menggunakan kebijakan khusus alih-alih kebijakan terkelola kami, pembuatan atau pembaruan lingkunga n mungkin gagal saat Anda memperbarui konfigurasi lingkungan. Untuk informasi lebih lanjut dan pertimban gan lainnya, lihatMemigrasi lingkungan Elastic Beanstalk Anda untuk meluncurkan template .		
EC2KeyNam e	Anda dapat menggunakan key pair untuk masuk ke EC2 instans Anda dengan aman. Jika nilai ditentukan untuk `EC2KeyNam e` dalam lingkungan yang telah `DisableDefaultEC2SecurityGr oup` disetel ke `true` grup keamanan default tidak akan dikaitkan dengan EC2 instance. **Note** Jika Anda menggunakan konsol Elastic Beanstalk untuk membuat lingkungan, Anda tidak dapat menetapkan opsi ini di file konfigurasi. Konsol tersebut mengganti pilihan ini dengan nilai yang disarankan.	Tidak ada	

Nama	Deskripsi	Default	Nilai yang valid
IamInstanceProfile	Profil instans memungkinkan pengguna dan AWS layanan AWS Identity and Access Management (IAM) untuk mengakses kredensil keamanan sementara untuk melakukan AWS panggilan API. Tentukan nama profil instans atau ARN-nya. Contoh: • `aws-elasticbeanstalk-ec2-role` • `arn:aws:iam::123456789012:instance-profile/aws-elasticbeanstalk-ec2-role` **Note** Jika Anda menggunakan konsol Elastic Beanstalk atau EB CLI untuk membuat lingkungan, Anda tidak dapat menetapkan opsi ini di file konfigurasi. Konsol dan EB CLI mengganti opsi ini dengan nilai yang disarankan.	Tidak ada	Nama profil instans atau ARN.
ImageId	Anda dapat mengganti Amazon Machine Image (AMI) default dengan menentukan ID AMI khusus milik Anda sendiri. Contoh: `ami-1f316660`	Tidak ada	

Nama	Deskripsi	Default	Nilai yang valid
InstanceType	Jenis instance yang digunakan untuk menjalankan aplikasi Anda di lingkungan Elastic Beanstalk. **Important** Opsi `InstanceType` tersebut sudah usang. Opsi ini digantikan oleh opsi `InstanceTypes` yang lebih baru dan lebih kuat di namespace `aws:ec2:instances`. Anda dapat menggunakan opsi baru ini untuk menentukan daftar satu atau beberapa jenis instance untuk lingkungan Anda. Nilai pertama pada daftar itu setara dengan nilai `InstanceType` opsi yang disertakan dalam `aws:autoscaling:launchconfiguration` namespace yang dijelaskan di sini. Kami menyarankan Anda menentukan jenis instance dengan menggunakan opsi baru. Jika ditentukan, opsi baru lebih diutamakan daripada yang sebelumnya. Untuk informasi selengkapnya, lihat the section called “Namespace aws:ec2:instances”. Jenis instans yang tersedia bergantung pada Availability Zones dan Region	Bervariasi berdasarkan akun dan Wilayah.	Satu jenis EC2 contoh. Bervariasi berdasarkan akun, Wilayah, dan Availability Zone. Anda dapat memperoleh daftar jenis EC2 instans Amazon yang difilter berdasarkan nilai-nilai ini. Untuk informasi selengkapnya, lihat Jenis instans yang tersedia di Panduan EC2 Pengguna Amazon.

Nama	Deskripsi	Default	Nilai yang valid
	yang digunakan. Jika Anda memilih subnet, Availability Zone yang berisi subnet tersebut menentukan jenis instance yang tersedia. • Elastic Beanstalk tidak EC2 mendukung jenis instans Amazon Mac. • Untuk informasi selengkapnya tentang keluarga dan jenis EC2 instans Amazon, lihat Jenis instans di Panduan EC2 Pengguna Amazon. • Untuk informasi selengkapnya tentang jenis instans yang tersedia di seluruh Wilayah, lihat Jenis instans yang tersedia di Panduan EC2 Pengguna Amazon. **Note** Jika Anda menggunakan konsol Elastic Beanstalk atau EB CLI untuk membuat lingkungan, Anda tidak dapat menetapkan opsi ini di file konfigurasi. Konsol dan EB CLI mengganti opsi ini dengan nilai yang disarankan.		

Nama	Deskripsi	Default	Nilai yang valid

LaunchTemplateTagPropagationEnabled	Setel `true` untuk mengaktifkan penyebaran tag lingkungan ke templat peluncuran untuk sumber daya tertentu yang disediakan ke lingkungan.
	
	Elastic Beanstalk hanya dapat menyebarkan tag untuk meluncurkan template untuk sumber daya berikut:
	
	• Volume EBS
	• EC2 contoh
	• EC2 antarmuka jaringan
	• AWS CloudFormation luncurkan template yang menentukan sumber daya
	
	Kendala ini ada karena CloudFormation hanya mengizinkan tag pada pembuatan template untuk sumber daya tertentu. Untuk informasi selengkapnya lihat TagSpecificationdi Panduan AWS CloudFormation Pengguna.
	
	⚠ Important
	
	• Mengubah nilai opsi ini dari `false` ke `true` untuk lingkungan yang ada mungkin merupakan perubahan besar untuk tag yang sudah ada sebelumnya.
		`false`	`true`
			
			`false`

Nama	Deskripsi	Default	Nilai yang valid
	• Ketika fitur ini diaktifkan, propagasi tag akan membutuhkan EC2 penggantian, yang dapat mengakibatkan downtime. Anda dapat mengaktifkan pembaruan bergulir untuk menerapkan perubahan konfigurasi dalam batch dan mencegah waktu henti selama proses pembaruan. Untuk informasi selengkapnya, lihat Perubahan konfigurasi. Untuk informasi selengkapnya tentang template peluncuran, lihat berikut ini: • Luncurkan template di Panduan Pengguna EC2 Auto Scaling Amazon • Bekerja dengan template di Panduan AWS CloudFormation Pengguna • Cuplikan template Elastic Beanstalk di Panduan Pengguna AWS CloudFormation Untuk informasi selengkapnya tentang metrik ini, lihat Tag propagasi untuk meluncurkan template.		

Nama	Deskripsi	Default	Nilai yang valid
MonitoringInterval	Interval (dalam hitungan menit) yang Anda inginkan CloudWatch metrik Amazon dikembalikan.	`5 minute`	`1 minute` `5 minute`

Nama	Deskripsi	Default	Nilai yang valid
SecurityGroups	Daftar grup EC2 keamanan Amazon IDs untuk ditetapkan ke EC2 instans dalam grup Auto Scaling untuk menentukan aturan firewall untuk instance. Gunakan opsi ini bersama dengan `DisableDefaultEC2SecurityGroup` untuk melampirkan grup keamanan kustom Anda sendiri yang menentukan aturan firewall untuk EC2 instance. Untuk informasi selengkapnya, lihat Memuat lingkungan seimbang (multi-instance). **Important** Anda mungkin perlu menyelesaikan beberapa konfigurasi tambahan untuk mencegah lalu lintas masuk ke EC2 instans Anda diblokir. Ini hanya berlaku untuk lingkungan multi-instance dengan grup EC2 keamanan khusus. Kelompok EC2 keamanan harus menyertakan aturan masuk yang memberikan akses ke lalu lintas yang diarahkan dari penyeimbang beban. Untuk informasi selengkapnya, lihat Mengelola grup EC2 keamanan di lingkungan multi-instance.	`elasticbeanstalk-default`	

Nama	Deskripsi	Default	Nilai yang valid
	Anda dapat memberikan satu string nilai yang dipisahkan koma yang berisi grup EC2 keamanan Amazon yang ada IDs atau referensi ke AWS::EC2::SecurityGroup sumber daya yang dibuat dalam templat. Anda harus memberikan setidaknya satu nilai untuk opsi ini jika `DisableDefaultEC2SecurityGroup` untuk namespace ini diatur ke. `true`		

Nama	Deskripsi	Default	Nilai yang valid
SSHSource Restriction	Digunakan untuk mengunci akses SSH ke lingkungan. Misalnya, Anda dapat mengunci akses SSH ke EC2 instance sehingga hanya host bastion yang dapat mengakses instance di subnet pribadi. String ini mengambil bentuk sebagai berikut: *protocol*, *fromPort*, *toPort*, *source_restriction* *protocol* Protokol untuk aturan masuk. *fromPort* Nomor port awal. *toPort* Nomor port akhir. *source_restriction* Rentang Classless Inter-Domain Routing (CIDR) atau grup keamanan yang harus dilalui lalu lintas. Tentukan grup keamanan dengan ID grup keamanan. Untuk menentukan grup keamanan dari akun lain, sertakan ID AWS akun sebelum ID grup keamanan, dipisahkan dengan garis miring ke depan. Akun lain harus berada di AWS Wilayah yang sama.	Tidak ada	

Nama	Deskripsi	Default	Nilai yang valid
	Perhatikan sintaksnya:*aws-account-id* /*security-group-id* . Misalnya: **123456789012 /sg-99999999** Contoh: • `tcp, 22, 22, 54.240.196.185/32` • `tcp, 22, 22, my-security-group-id` • `tcp, 22, 22, 123456789012/their-security-group-id`		

Nama	Deskripsi	Default	Nilai yang valid
BlockDeviceMappings	Lampirkan volume Amazon EBS tambahan atau volume penyimpanan instans di semua instans dalam grup Auto Scaling. **Important** Pengaturan opsi ini dapat menyebabkan Elastic Beanstalk memigrasikan lingkungan yang ada dengan konfigurasi peluncuran untuk meluncurkan templat. Melakukannya memerlukan izin yang diperlukan untuk mengelola templat peluncuran. Izin ini disertakan dalam kebijakan terkelola kami. Jika Anda menggunakan kebijakan khusus alih-alih kebijakan terkelola kami, pembuatan atau pembaruan lingkungan mungkin gagal saat Anda memperbarui konfigurasi lingkungan. Untuk informasi lebih lanjut dan pertimbangan lainnya, lihat Memigrasi lingkungan Elastic Beanstalk Anda untuk meluncurkan template . Saat memetakan volume penyimpanan instance, Anda hanya perlu memetakan nama perangkat ke nama	Tidak ada	• ukuran - harus antara 500 dan 16384 GiB • throughput — harus antara 125 dan 1000 mebibytes per detik (MIB/s)

Nama	Deskripsi	Default	Nilai yang valid
	volume. Namun, kami menyarankan, saat memetakan volume Amazon EBS, Anda juga menentukan beberapa atau semua bidang berikut (setiap bidang harus dipisahkan oleh titik dua): • ID snapshot • ukuran, dalam GB • Hapus saat pengakhiran (`true` or `false`) • tipe penyimpanan (hanya untuk `gp3`, `gp2`, `standard`, `st1`, `sc1`, atau `io1`) • IOPS (hanya untuk `gp3` atau `io1`) • throughput (hanya untuk `gp3`) Contoh berikut melampirkan tiga volume Amazon EBS, satu volume 100GB gp2 kosong dan satu snapshot, satu volume io1 20GB kosong dengan 2000 provisioned IOPS, dan volume penyimpanan instans `ephemeral 0` . Beberapa volume penyimpanan instans dapat dilampirkan jika tipe instans mendukungnya. `/dev/sdj=:100:true:gp2,/dev /sdh=snap-51eef269,/dev/ sdi=:20:true:io1:2000,/ dev/sdb=ephemeral0`		

Nama	Deskripsi	Default	Nilai yang valid
RootVolumeType	Jenis volume (magnetik, SSD tujuan umum, atau SSD IOPS yang disediakan) untuk digunakan untuk volume Amazon EBS root yang terpasang pada EC2 instans untuk lingkungan Anda. **Important** Pengaturan opsi ini dapat menyebabkan Elastic Beanstalk memigrasikan lingkungan yang ada dengan konfigurasi peluncuran untuk meluncurkan templat. Melakukannya memerlukan izin yang diperlukan untuk mengelola templat peluncuran. Izin ini disertakan dalam kebijakan terkelola kami. Jika Anda menggunakan kebijakan khusus alih-alih kebijakan terkelola kami, pembuatan atau pembaruan lingkungan mungkin gagal saat Anda memperbarui konfigurasi lingkungan. Untuk informasi lebih lanjut dan pertimbangan lainnya, lihat Memigrasi lingkungan Elastic Beanstalk Anda untuk meluncurkan template .	Bervariasi berdasarkan platform.	`standard` untuk penyimpanan magnetik. `gp2` atau `gp3` untuk tujuan umum SSD. `io1` untuk SSD provisioned IOPS.

Nama	Deskripsi	Default	Nilai yang valid
RootVolumeSize	Kapasitas penyimpanan volume root Amazon EBS di seluruh GB. Diperlukan jika Anda mengatur `RootVolumeType` untuk SSD provisioned IOPS. Misalnya, `"64"`.	Bervariasi per platform untuk penyimpanan magnetik dan SSD tujuan umum. Tidak ada untuk SSD provisioned IOPS.	`10` ke `16384` GB untuk tujuan umum dan SSD provisioned IOPS. `8` ke `1024` GB untuk magnetik.
RootVolumeIOPS	input/output Operasi per detik yang diinginkan (IOPS) untuk volume root IOPS SSD yang disediakan atau untuk volume root SSD tujuan `gp3` umum. Rasio maksimum IOPS terhadap ukuran volume adalah 500 banding 1. Misalnya, volume dengan 3000 IOPS harus minimal 6 GiB.	Tidak ada	`100` ke `20000` untuk volume root SSD io1 provisioned IOPS. `3000` ke `16000` untuk volume akar SSD `gp3` tujuan umum.
RootVolumeThroughput	Throughput mebibytes per detik (MIB/s) yang diinginkan untuk menyediakan volume root Amazon EBS yang dilampirkan ke instans lingkungan Anda. EC2 **Note** Opsi ini hanya berlaku untuk tipe penyimpanan `gp3`.	Tidak ada	`125` untuk `1000`

aws:autoscaling:scheduledaction

Konfigurasikan tindakan terjadwal untuk grup Auto Scaling lingkungan Anda. Untuk setiap tindakan, tentukan `resource_name` selain nama opsi, namespace, dan nilai untuk setiap pengaturan. Lihat Namespace aws:autoscaling:scheduledaction sebagai contoh.

Namespace: **`aws:autoscaling:scheduledaction`**

Nama	Deskripsi	Default	Nilai yang valid
StartTime	Untuk tindakan satu kali, pilih tanggal dan waktu untuk menjalankan tindakan. Untuk tindakan berulang, pilih waktu untuk mengaktifkan tindakan.	Tidak ada	Stempel waktu ISO-8601 unik di semua tindakan penskalaan terjadwal.
EndTime	Tanggal dan waktu di masa depan (di zona UTC/GMT waktu) saat Anda ingin tindakan penskalaan terjadwal berhenti berulang. Jika Anda tidak menentukan EndTime, tindakan akan berulang sesuai dengan `Recurrence` ekspresi. Contoh: `2015-04-28T04:07:2Z` Ketika tindakan terjadwal berakhir, Amazon EC2 Auto Scaling tidak secara otomatis kembali ke pengaturan sebelumnya. Konfigurasikan tindakan terjadwal kedua untuk kembali ke pengaturan asli sesuai kebutuhan.	Tidak ada	Stempel waktu ISO-8601 unik di semua tindakan penskalaan terjadwal.
MaxSize	Jumlah instans maksimum yang diterapkan ketika tindakan berjalan.	Tidak ada	`0` untuk `10000`
MinSize	Jumlah instans minimum yang diterapkan ketika tindakan berjalan.	Tidak ada	`0` untuk `10000`

Nama	Deskripsi	Default	Nilai yang valid
DesiredCapacity	Atur kapasitas awal yang diinginkan untuk grup Auto Scaling. Setelah tindakan terjadwal diterapkan, pemicu menyesuaikan kapasitas yang diinginkan berdasarkan pengaturannya.	Tidak ada	`0` untuk `10000`
Recurrence	Frekuensi yang Anda inginkan untuk terjadinya tindakan terjadwal. Jika Anda tidak menentukan pengulangan, maka tindakan penskalaan terjadi hanya sekali, sebagaimana ditentukan oleh `StartTime` .	Tidak ada	Ekspresi Cron.
Suspend	Atur ke `true` untuk menonaktifkan tindakan terjadwal berulang untuk sementara.	`false`	`true` `false`

aws:autoscaling:trigger

Konfigurasikan pemicu penskalaan untuk grup Auto Scaling lingkungan Anda.

> **Note**
>
> Tiga opsi di namespace ini menentukan berapa lama metrik untuk pemicu dapat tetap melampaui batas yang ditentukan sebelum pemicu dimulai. Opsi tersebut adalah sebagai berikut:
> `BreachDuration = Period * EvaluationPeriods`
> Nilai default untuk pilihan ini (5, 5, dan 1, berturut-turut) memenuhi persamaan ini. Jika Anda menentukan nilai yang tidak konsisten, Elastic Beanstalk mungkin memodifikasi salah satu nilai sehingga persamaan tetap terpenuhi.

Namespace: **`aws:autoscaling:trigger`**

Nama	Deskripsi	Default	Nilai yang valid
BreachDuration	Jumlah waktu, dalam menit, metrik dapat melampaui batas yang ditetapkan (sebagaimana ditentukan dalam `UpperThreshold` dan `LowerThreshold`) sebelum pemicu dipanggil.	`5`	`1` untuk `600`
LowerBreachScaleIncrement	Berapa banyak EC2 instans Amazon yang harus dihapus saat melakukan aktivitas penskalaan.	`-1`	
LowerThreshold	Jika pengukuran turun di bawah angka ini selama durasi pelanggaran, pemicu dipanggil.	`2000000`	`0` untuk `20000000`

MeasureName	Metrik yang digunakan untuk pemicu Auto Scaling Anda.
	
	Note
	`HealthyHostCount`, `UnhealthyHostCount` dan `TargetResponseTime` hanya berlaku untuk lingkungan dengan penyeimbang beban khusus. Ini bukan nilai metrik yang valid untuk lingkungan yang dikonfigurasi dengan penyeimbang beban bersama. Untuk informasi selengkapnya tentang tipe penyeimbang beban, lihat Penyeimbang beban untuk
		`NetworkOut`	`CPUUtilization`
			
			`NetworkIn`
			
			`NetworkOut`
			
			`DiskWriteOps`
			
			`DiskReadBytes`
			
			`DiskReadOps`
			
			`DiskWriteBytes`
			
			`Latency`
			
			`RequestCount`

Nama	Deskripsi	Default	Nilai yang valid

	lingkungan Elastic Beanstalk Anda.		`HealthyHostCount`
			`UnhealthyHostCount`
			`TargetResponseTime`

Period	Menentukan seberapa sering Amazon CloudWatch mengukur metrik untuk pemicu Anda. Nilai adalah jumlah menit antara dua periode berturut-turut.	`5`	1 untuk `600`
EvaluationPeriods	Jumlah periode evaluasi berturut-turut yang digunakan untuk menentukan apakah pelanggaran terjadi.	`1`	1 untuk `600`

Statistic	Statistik yang digunakan pemicu, seperti`Average`.	`Average`	`Minimum`
			`Maximum`
			`Sum`
			`Average`

Nama	Deskripsi	Default	Nilai yang valid

Unit	Unit untuk pengukuran pemicu, seperti `Bytes`.	`Bytes`	`Seconds`
			`Percent`
			`Bytes`
			`Bits`
			`Count`
			`Bytes/Second`
			`Bits/Second`
			`Count/Second`
			`None`

UpperBreachScaleIncrement	Menentukan berapa banyak EC2 instans Amazon yang akan ditambahkan saat melakukan aktivitas penskalaan.	1	
UpperThreshold	Jika pengukuran lebih tinggi dari angka ini selama durasi pelanggaran, pemicu dipanggil.	`6000000`	`0` untuk `20000000`

aws:autoscaling:updatepolicy:rollingupdate

Konfigurasikan pembaruan bergulir grup Auto Scaling lingkungan Anda.

Namespace: **`aws:autoscaling:updatepolicy:rollingupdate`**

Nama	Deskripsi	Default	Nilai yang valid
MaxBatchSize	Jumlah instans yang disertakan	Sepertiga dari ukuran minimum grup Auto Scaling,	1 untuk `10000`

Nama	Deskripsi	Default	Nilai yang valid
	dalam setiap batch pembaruan bergulir.	dibulatkan ke bilangan bulat tertinggi berikutnya.	
MinInstancesInService	Jumlah minimum instans yang harus ada di layanan dalam grup Auto Scaling sementara instans lain dihentikan.	Ukuran minimum grup Auto Scaling atau kurang satu dari ukuran maksimum grup Auto Scaling, mana pun yang lebih rendah.	`0` untuk 9999

Nama	Deskripsi	Default	Nilai yang valid

RollingUpdateEnabled	Jika `true`, Auto Scalling memungkinkan pembaruan bergulir untuk lingkungan. Pembaruan bergulir berguna saat Anda perlu melakukan pembaruan kecil dan sering pada aplikasi perangkat lunak Elastic Beanstalk Anda dan Anda ingin menghindari waktu henti aplikasi.
	
	Menetapkan nilai ini ke true secara otomatis mengaktifkan opsi `MaxBatchSize` , `MinInstancesInService` , dan `PauseTime` . Menetapkan salah satu opsi tersebut juga secara otomatis menetapkan nilai opsi `RollingUpdateEnabled` ke `true`. Menetapkan opsi ini ke `false` menonaktifkan pembaruan bergulir.
		`false`	`true`
			
			`false`

Nama	Deskripsi	Default	Nilai yang valid
	Note Jika Anda menggunakan konsol Elastic Beanstalk atau EB CLI untuk membuat lingkungan, Anda tidak dapat menetapkan opsi ini di file konfigurasi. Konsol dan EB CLI mengganti opsi ini dengan nilai yang disarankan.		

Nama	Deskripsi	Default	Nilai yang valid

RollingUpdateType	Ini mencakup tiga tipe: pembaruan bergulir berbasis waktu, pembaruan bergulir berbasis kondisi, dan pembaruan yang tidak berubah.
	
	Pembaruan bergulir berbasis waktu menerapkan PauseTime antar batch. Pembaruan bergulir berbasis kondisi menunggu instans baru untuk lulus pemeriksaan kondisi sebelum beralih ke batch berikutnya. Pembaruan tetap meluncurkan serangkaian instans lengkap di grup Auto Scaling baru.
	
	Note
	Jika Anda menggunakan konsol Elastic Beanstalk atau EB CLI untuk
		`Time`	`Time`
			
			`Health`
			
			`Immutable`

Nama	Deskripsi	Default	Nilai yang valid
	membuat lingkungan, Anda tidak dapat menetapkan opsi ini di file konfigurasi. Konsol dan EB CLI mengganti opsi ini dengan nilai yang disarankan.		
PauseTime	Jumlah waktu (dalam detik, menit, atau jam) layanan Elastic Beanstalk menunggu setelah menyelesaikan pembaruan ke satu batch instance dan sebelum melanjutkan ke batch berikutnya.	Secara otomatis dihitung berdasarkan tipe instans dan kontainer.	`PT0S`* (0 detik) hingga `PT1H` (1 jam)

Nama	Deskripsi	Default	Nilai yang valid
Timeout	Jumlah maksimum waktu (dalam menit atau jam) untuk menunggu semua instance dalam batch instance untuk lulus pemeriksaan kesehatan sebelum membatalkan pembaruan.	`PT30M` (30 menit)	`PT5M`* (5 menit) sampai `PT1H` (1 jam) * Format durasi ISO86 01: `PT#H#M#S` di mana setiap # adalah jumlah jam, menit, and/or detik, masing-masing.

aws:ec2:instances

Konfigurasikan instans lingkungan Anda, termasuk opsi Spot. Namespace ini melengkapi `aws:autoscaling:launchconfiguration` dan `aws:autoscaling:asg`.

Untuk informasi selengkapnya, lihat the section called “Grup Auto Scaling”.

Namespace: **`aws:ec2:instances`**

Nama	Deskripsi	Defaul	Nilai yang valid

EnableSpot	Aktifkan permintaan Instans Spot untuk lingkungan Anda. Saat `false`, beberapa opsi di namespace ini tidak berlaku.
	
	Important
	Pengaturan opsi ini dapat menyebabkan Elastic Beanstalk memigrasikan lingkungan yang ada dengan konfigurasi peluncuran untuk meluncurkan templat.
		`false`	`true`
			
			`false`

Nama	Deskripsi	Defaul	Nilai yang valid
	Melakukannya memerlukan izin yang diperlukan untuk mengelola templat peluncuran. Izin ini disertakan dalam kebijakan terkelola kami. Jika Anda menggunakan kebijakan khusus alih-alih kebijakan terkelola kami, pembuatan atau pembaruan lingkungan mungkin gagal saat Anda memperbarui konfigurasi lingkungan. Untuk informasi lebih lanjut dan pertimbangan lainnya, lihatMemigrasi lingkungan Elastic Beanstalk Anda untuk meluncurkan template .		

Nama	Deskripsi	Defaul	Nilai yang valid
InstanceTypes	Daftar tipe instance yang dipisahkan koma yang Anda ingin lingkungan Anda gunakan (misalnya,`t2.micro, t3.micro`). Kapan `true` dan `EnableSpot` `SpotAllocationStrategy` diatur ke`capacity-optimized -prioritized` , daftar nilai yang ditentukan dalam opsi ini menentukan prioritas tipe instance untuk strategi alokasi Instans Spot. Ketika Instans Spot tidak diaktifkan (`EnableSpot` `isfalse`), hanya jenis instans pertama pada daftar yang digunakan. Tipe instans pertama pada daftar dalam pilihan ini setara dengan nilai opsi `InstanceType` di namespace `aws:autoscaling:launchconfiguration` . Kami tidak menyarankan penggunaan opsi terakhir karena opsi sudah usang. Jika Anda menentukan keduanya, tipe instans pertama pada daftar di opsi `InstanceTypes` digunakan, dan `InstanceType` diabaikan. Jenis instans yang tersedia bergantung pada Availability Zones dan Region yang digunakan. Jika Anda memilih subnet, Availability Zone yang berisi subnet tersebut menentukan jenis instance yang tersedia.	Daftar dua tipe instans Bervari berdas an akun dan Wilaya	Satu hingga empat puluh jenis EC2 contoh. Kami merekomendasikan setidaknya dua. Bervariasi berdasarkan akun, Wilayah, dan Availability Zone. Anda dapat memperoleh daftar jenis EC2 instans Amazon yang difilter berdasarkan nilai-nilai ini. Untuk informasi selengkapnya, lihat Jenis instans yang tersedia di Panduan EC2 Pengguna Amazon. Tipe instance semuanya harus menjadi bagian dari arsitektur yang sama (`arm64`,`x86_64`,`i386`). `Supported Architectures` juga merupakan bagian dari namespace ini. Jika Anda memberikan nilai apa pun`Supported Architectures` , nilai yang Anda masukkan `InstanceTypes` harus milik satu, dan hanya satu,

Nama	Deskripsi	Defaul	Nilai yang valid
	• Elastic Beanstalk tidak EC2 mendukung jenis instans Amazon Mac. • Untuk informasi selengkapnya tentang keluarga dan jenis EC2 instans Amazon, lihat Jenis instans di Panduan EC2 Pengguna Amazon. • Untuk informasi selengkapnya tentang jenis instans yang tersedia di seluruh Wilayah, lihat Jenis instans yang tersedia di Panduan EC2 Pengguna Amazon. **Note** Beberapa AWS akun lama mungkin menyediakan Elastic Beanstalk dengan tipe instans default yang tidak mendukung Instans Spot (misalnya, t1.micro). Jika Anda mengaktif kan permintaan Instans Spot dan mendapatkan kesalahan tentang jenis instans yang tidak mendukung Spot, pastikan untuk mengonfig urasi jenis instans yang mendukung Spot. Untuk memilih tipe Instans Spot, gunakan Penasihat Instans Spot.		dari arsitektur yang Anda `Supported Architectures` sediakan.

Nama	Deskripsi	Defaul	Nilai yang valid
	Saat memperbarui konfigurasi lingkungan dan menghapus satu atau beberapa jenis instans dari `InstanceTypes` opsi, Elastic Beanstalk menghentikan semua instans Amazon yang berjalan pada EC2 salah satu jenis instans yang dihapus. Grup Auto Scaling lingkungan Anda kemudian meluncurkan instans baru, yang diperlukan untuk menyelesaikan kapasitas yang diinginkan, menggunakan tipe instans yang ditentukan saat ini.		

SpotAllocationStrategy	Menentukan strategi alokasi instans spot yang menentukan bagaimana Instans Spot dialokasikan dari kumpulan kapasitas spot yang tersedia.
	
	Jika disetel ke`capacity-optimized-prioritized`, urutan nilai dalam `InstanceTypes` menetapkan prioritas tipe instance untuk alokasi.
	
	Pilihan ini hanya relevan bila `EnableSpot` adalah `true`.
		`capac optim`	`capacity-optimized`
			
			`price-capacity-optimized`
			
			`capacity-optimized-prioritized`
			
			`lowest-price`

Nama	Deskripsi	Defaul	Nilai yang valid
SpotFleet OnDemandBase	Jumlah minimum Instans Sesuai Permintaan yang disediakan grup Auto Scaling Anda sebelum mempertim bangkan Instans Spot saat lingkungan Anda bertambah besar. Pilihan ini hanya relevan bila `EnableSpot` adalah `true`.	`0`	`0` ke opsi `MaxSize` di namespace `aws:autos caling:asg`
SpotFleet OnDemandA boveBaseP ercentage	Persentase Instans Sesuai Permintaa n sebagai bagian dari kapasitas tambahan yang disediakan grup Auto Scaling di luar `SpotOnDemandBase` instans. Pilihan ini hanya relevan bila `EnableSpot` adalah `true`.	`0` untuk lingkur n instans tungga `70` untuk lingkur n yang seimba dengai beban	`0` untuk `100`

Nama	Deskripsi	Defaul	Nilai yang valid

SpotMaxPrice	Harga maksimum per unit jam, dalam dollar AS, yang bersedia Anda bayarkan untuk Instans Spot. Untuk rekomendasi tentang opsi harga maksimum untuk Instans Spot, lihat riwayat harga Instans Spot di Panduan EC2 Pengguna Amazon.
	
	Pilihan ini hanya relevan bila `EnableSpot` adalah `true`.
		Harga sesuai permin n, untuk setiap tipe instans Nilai opsi pada kasus ini adalah `null`.	`0.001` untuk `20.0`
			
			`null`

Nama	Deskripsi	Defaul	Nilai yang valid

SupportedArchitectures	Daftar tipe arsitektur EC2 instance yang dipisahkan koma yang akan Anda gunakan untuk lingkungan Anda.
	
	Elastic Beanstalk mendukung jenis instans berdasarkan arsitektur prosesor berikut:
	
	• AWS Arsitektur Arm Graviton 64-bit (arm64)
	• Arsitektur 64-bit (x86_64)
	• Arsitektur 32-bit (i386)
	
	Untuk informasi selengkapnya tentang arsitektur prosesor dan jenis EC2 instans Amazon, lihatthe section called “Jenis EC2 instans Amazon”.
		Tidak ada	`arm64`
			
			`x86_64`
			
			`i386`
			
			Note
			Arsitektur 32-bit tidak `i386` didukung oleh sebagian besar platform Elastic Beanstalk. Kami menyarankan Anda memilih `x86_64` atau jenis `arm64` arsitektur sebagai gantinya.

aws:ec2:vpc

Konfigurasikan lingkungan Anda untuk meluncurkan sumber daya di Amazon Virtual Private Cloud (Amazon VPC) khusus. Jika Anda tidak mengonfigurasi pengaturan dalam namespace ini, Elastic Beanstalk meluncurkan sumber daya di VPC default.

Namespace: **`aws:ec2:vpc`**

Nama	Deskripsi	Default	Nilai yang valid
VPCId	ID untuk Amazon VPC Anda.	Tidak ada	

Nama	Deskripsi	Default	Nilai yang valid
Subnets	IDs Subnet atau subnet grup Auto Scaling. Jika Anda memiliki beberapa subnet, tentukan nilainya sebagai string subnet yang dipisahkan koma tunggal IDs (misalnya,). `"subnet-11111111,subnet-22222222"`	Tidak ada	
ELBSubnets	Subnet atau subnet untuk penyeimbang beban elastis. IDs Jika Anda memiliki beberapa subnet, tentukan nilainya sebagai string subnet yang dipisahkan koma tunggal IDs (misalnya,). `"subnet-11111111,subnet-22222222"`	Tidak ada	

ELBScheme	Tentukan `internal` jika Anda ingin membuat penyeimbang beban internal di Amazon VPC Anda maka aplikasi Elastic Beanstalk Anda tidak dapat diakses dari luar Amazon VPC Anda. Jika Anda menentukan nilai selain dari `public` atau `internal`, Elastic Beanstalk mengabaikan nilainya.	`public`	`public`
			
			`internal`

DBSubnets	IDs Berisi subnet database. Ini hanya digunakan jika Anda ingin menambahkan Instans Amazon RDS DB sebagai bagian dari aplikasi Anda. Jika Anda memiliki beberapa subnet, tentukan nilainya sebagai string subnet yang dipisahkan koma tunggal IDs (misalnya,). `"subnet-11111111,subnet-22222222"`	Tidak ada	

Nama	Deskripsi	Default	Nilai yang valid

AssociatePublicIpAddress	Menentukan apakah akan meluncurkan instans dengan alamat IP publik di Amazon VPC Anda. Instans dengan alamat IP publik tidak memerlukan perangkat NAT untuk berkomunikasi dengan Internet. Anda harus menetapkan nilai `true` jika Anda ingin menyertakan penyeimbang beban dan instans Anda dalam satu subnet publik.
	
	Opsi ini tidak berpengaruh pada lingkungan instans tunggal, yang selalu memiliki satu EC2 instans Amazon dengan alamat IP Elastis. Pilihan ini relevan untuk lingkungan beban seimbang dan terukur.
		Tidak ada	`true`
			
			`false`

aws:elasticbeanstalk:application

Konfigurasikan jalur pemeriksaan kondisi untuk aplikasi Anda. Untuk informasi selengkapnya, lihat Pelaporan kondisi dasar.

Namespace: **`aws:elasticbeanstalk:application`**

Nama	Deskripsi	Defaul	Nilai yang valid

URL pemeriksaan kondisi aplikasi	Jalur tempat permintaan pemeriksaan kondisi dikirim. Jika jalur ini tidak ditetapkan, penyeimbang beban mencoba untuk membuat koneksi TCP pada port 80 untuk memverifikasi status kondisi aplikasi Anda. Atur ke jalur yang dimulai dengan / untuk mengirim permintaan HTTP GET ke jalur itu. Anda juga dapat menyertakan protokol (HTTP, HTTPS, TCP, atau SSL) dan port sebelum jalur itu untuk	Tidak ada	Nilai yang valid meliputi:
			
			/ (HTTP GET ke jalur root)
			
			`/health`
			
			`HTTPS:443/`
			
			`HTTPS:443/health`

Nama	Deskripsi	Defaul	Nilai yang valid
	memeriksa konektivitas HTTPS atau menggunakan port non-default. **Note** Jika Anda menggunakan konsol Elastic Beanstalk untuk membuat lingkungan, Anda tidak dapat menetapkan opsi ini di file konfigurasi. Konsol tersebut mengganti pilihan ini dengan nilai yang disarankan.		

Konsol EB CLI dan Elastic Beanstalk menerapkan nilai yang direkomendasikan untuk pilihan sebelumnya. Anda harus menghapus pengaturan ini jika Anda ingin menggunakan file konfigurasi untuk mengonfigurasi hal yang sama. Lihat Nilai yang disarankan untuk rincian selengkapnya.

aws:elasticbeanstalk:application:environment

Konfigurasikan properti lingkungan untuk aplikasi Anda.

Namespace: **`aws:elasticbeanstalk:application:environment`**

Nama	Deskripsi	Default	Nilai yang valid
Nama variabel lingkungan apa pun.	Lulus di pasangan nilai kunci.	Tidak ada	Nilai variabel lingkungan apa pun.

Lihat Variabel lingkungan dan pengaturan perangkat lunak lainnya untuk informasi selengkapnya.

aws:elasticbeanstalk:aplikasi:environmentsecrets

Konfigurasikan variabel lingkungan untuk berfungsi sebagai rahasia lingkungan untuk aplikasi Anda. Rahasia lingkungan menyimpan AWS Secrets Manager rahasia atau AWS Systems Manager parameter Parameter Store.

Namespace: **`aws:elasticbeanstalk:application:environmentsecrets`**

Nama	Deskripsi	Defaul	Nilai yang valid
OptionName	Menentukan nama variabel lingkungan untuk memegang toko rahasia atau nilai penyimpanan parameter.	Tidak ada	Nama variabel lingkungan apa pun.
Nilai	Menentukan ARN untuk nilai yang disimpan AWS Secrets Manager di AWS Systems Manager atau Parameter Store. Selama bootstrap instance Elastic Beanstalk memulai variabel lingkungan ke nilai yang disimpan dalam sumber daya ARN ini. **Note** Pastikan bahwa izin yang diperlukan tersedia untuk peran profil EC2 instans lingkungan Anda untuk mengakses rahasia dan parameter ARNs. Untuk informasi selengkapnya, lihat Izin IAM yang diperlukan.	Tidak ada	Nilai ARN yang valid untuk nilai parameter AWS Secrets Manager rahasia atau AWS Systems Manager Parameter Store.

Untuk informasi selengkapnya, lihat Mengkonfigurasi rahasia sebagai variabel lingkungan.

aws:elasticbeanstalk:cloudwatch:logs

Konfigurasikan streaming log instans untuk aplikasi Anda.

Namespace: **`aws:elasticbeanstalk:cloudwatch:logs`**

Nama	Deskripsi	Default	Nilai yang valid
StreamLogs	Menentukan apakah akan membuat grup di CloudWatch Log untuk log proxy dan penyebaran, dan streaming log dari setiap instance di lingkungan Anda.	`false`	`true` `false`
DeleteOnTerminate	Menentukan apakah akan menghapus grup log ketika lingkungan dihentikan. Jika `false`, log disimpan `RetentionInDays` hari.	`false`	`true` `false`
RetentionInDays	Jumlah hari untuk menyimpan peristiwa log sebelum mereka berakhir.	7	1, 3, 5, 7, 14, 30, 60, 90, 120, 150, 180, 365, 400, 545, 731, 1827, 3653

aws:elasticbeanstalk:cloudwatch:logs:health

Konfigurasikan streaming log kondisi lingkungan untuk aplikasi Anda.

Namespace: **`aws:elasticbeanstalk:cloudwatch:logs:health`**

Nama	Deskripsi	Default	Nilai yang valid
HealthStreamingEnabled	Untuk lingkungan dengan pelaporan kesehatan yang ditingkatkan diaktifkan, tentukan apakah akan membuat grup di CloudWatch Log untuk kesehatan lingkungan dan mengarsipkan data kesehatan lingkungan Elastic Beanstalk. Untuk informasi tentang mengaktifkan peningkatan kondisi, lihat `aws:elasticbeanstalk:healthreporting:system`.	`false`	`true` `false`
DeleteOnTerminate	Menentukan apakah akan menghapus grup log ketika lingkungan dihentikan. Jika `false`, data kondisi disimpan `RetentionInDays` hari.	`false`	`true` `false`
RetentionInDays	Jumlah hari untuk menyimpan data kondisi yang diarsipkan sebelum data kondisi kedaluwarsa.	7	1, 3, 5, 7, 14, 30, 60, 90, 120, 150, 180, 365, 400, 545, 731, 1827, 3653

aws:elasticbeanstalk:command

Konfigurasikan kebijakan deployment untuk kode aplikasi Anda. Untuk informasi selengkapnya, lihat the section called “Pilihan deployment”.

Namespace: **`aws:elasticbeanstalk:command`**

Nama	Deskripsi	Default	Nilai yang valid

DeploymentPolicy	Pilih kebijakan deployment untuk deployment versi aplikasi.
	
	Note
	Jika Anda menggunakan konsol Elastic Beanstalk untuk membuat lingkungan, Anda tidak dapat menetapkan opsi ini di file konfigurasi. Konsol tersebut mengganti pilihan ini dengan nilai yang disarankan.
		`AllAtOnce`	`AllAtOnce`
			
			`Rolling`
			
			`RollingWithAdditionalBatch`
			
			`Immutable`
			
			`TrafficSplitting`

Timeout	Jumlah waktu, dalam detik, untuk menunggu instans menyelesaikan perintah eksekusi. Elastic Beanstalk secara internal menambahkan 240 detik (empat menit) ke nilai `Timeout`. Misalnya, batas waktu efektif secara default adalah 840 detik (600 + 240), atau 14 menit.	`600`	1 untuk `3600`

BatchSizeType	Jenis nomor yang ditentukan dalam BatchSize.
	
	Note
	Jika Anda menggunakan konsol Elastic Beanstalk atau EB CLI untuk membuat lingkungan, Anda tidak dapat menetapkan opsi ini di file konfigurasi. Konsol dan EB CLI
		`Percentage`	`Percentage`
			
			`Fixed`

Nama	Deskripsi	Default	Nilai yang valid
	mengganti opsi ini dengan nilai yang disarankan.		
BatchSize	Persentase atau jumlah EC2 instans Amazon yang tetap dalam grup Auto Scaling untuk menjalankan penerapan secara bersamaan. Nilai yang valid bervariasi tergantung pada BatchSizeTypepengaturan yang digunakan. **Note** Jika Anda menggunakan konsol Elastic Beanstalk atau EB CLI untuk membuat lingkungan, Anda tidak dapat menetapkan opsi ini di file konfigurasi. Konsol dan EB CLI mengganti opsi ini dengan nilai yang disarankan.	`100`	1 hingga `100` (`Percentage`). 1ke aws:autoscaling:asg::() MaxSize `Fixed`

IgnoreHealthCheck	Jangan membatalkan penerapan karena pemeriksaan kesehatan yang gagal.	`false`	`true`
			
			`false`

aws:elasticbeanstalk:environment

Konfigurasikan peran arsitektur dan layanan lingkungan Anda.

Namespace: **`aws:elasticbeanstalk:environment`**

Nama	Deskripsi	Default	Nilai yang valid

EnvironmentType	Setel `SingleInstance` untuk meluncurkan satu EC2 instance tanpa penyeimbang beban.	`LoadBalanced`	`SingleInstance`
			
			`LoadBalanced`

ServiceRole	Nama IAM role yang menggunakan Elastic Beanstalk untuk mengelola sumber daya untuk lingkungan. Tentukan nama peran (opsional diawali dengan jalur khusus) atau ARN -nya. Contoh: • `aws-elasticbeanstalk-service-role` • *`custom-path /custom-role`* • `arn:aws:iam::123456789012:role/aws-elasticbeanstalk-service-role` **Note** Jika Anda menggunakan konsol Elastic Beanstalk atau EB CLI untuk membuat lingkungan, Anda tidak dapat menetapkan opsi ini di file konfigurasi. Konsol dan EB CLI mengganti opsi ini dengan nilai yang disarankan.	Tidak ada	Nama IAM role, jalur/nama, atau ARN

Nama	Deskripsi	Default	Nilai yang valid

LoadBalancerType	Tipe penyeimbang beban untuk lingkungan Anda. Untuk informasi selengkapnya, lihat the section called “Penyeimbang beban”.	`classic`	`classic`
			`application`
			`network`

LoadBalancerIsShared	Menentukan apakah penyeimbang beban lingkungan dikhususkan atau dibagi. Opsi ini hanya dapat diatur untuk Application Load Balancer. Itu tidak dapat diubah setelah lingkungan dibuat. Ketika`false`, lingkungan memiliki penyeimbang beban khusus sendiri, dibuat, dan dikelola oleh Elastic Beanstalk. Saat `true`, lingkungan menggunakan penyeimbang beban bersama, yang Anda buat dan tentukan di opsi `SharedLoadBalancer` dari namespace aws:elbv2:loadbalancer.	`false`	`true` `false`

aws:elasticbeanstalk:environment:process:default

Konfigurasikan proses default lingkungan Anda.

Namespace: **`aws:elasticbeanstalk:environment:process:default`**

Nama	Deskripsi	Default	Nilai yang valid
DeregistrationDelay	Jumlah waktu, dalam detik, untuk menunggu permintaan aktif selesai sebelum membatalkan pendaftaran.	`20`	`0` untuk `3600`

Nama	Deskripsi	Default	Nilai yang valid
HealthCheckInterval	Interval waktu, dalam hitungan detik, Elastic Load Balancing memeriksa kesehatan EC2 instans Amazon aplikasi Anda.	Dengan classic load balancer atau application load balancer: `15` Dengan penyeimbang beban jaringan: `30`	Dengan classic load balancer atau application load balancer: `5` ke `300` Dengan penyeimbang beban jaringan: `10`, `30`
HealthCheckPath	Jalur pengiriman permintaan HTTP untuk pemeriksaan kondisi.	`/`	Jalur yang dapat dirutekan.
HealthCheckTimeout	Jumlah waktu, dalam detik, untuk menunggu respons selama pemeriksaan kondisi. Opsi ini hanya berlaku untuk lingkungan dengan application load balancer.	`5`	`1` untuk `60`
HealthyThresholdCount	Jumlah permintaan sukses berturut-turut sebelum Elastic Load Balancing mengubah status kondisi instans.	Dengan classic load balancer atau application load balancer: `3` Dengan penyeimbang beban jaringan: `5`	`2` untuk `10`

Nama	Deskripsi	Default	Nilai yang valid
MatcherHTTPCode	Daftar kode HTTP (s) dipisahkan koma yang menunjukkan bahwa instans sehat. Opsi ini hanya berlaku untuk lingkungan dengan network load balancer atau application load balancer.	`200`	Dengan application load balancer: `200` ke `499` Dengan penyeimbang beban jaringan: `200` ke `399`
Port	Port yang didengarkan proses.	`80`	1 untuk `65535`

Protocol	Protokol yang digunakan proses.
	
	Dengan application load balancer, Anda hanya dapat mengatur opsi ini untuk `HTTP` atau `HTTPS`.
	
	Dengan penyeimbang beban jaringan, Anda hanya dapat mengatur opsi ini untuk `TCP`.
		Dengan classic load balancer atau application load balancer: `HTTP`
		
		Dengan penyeimbang beban jaringan: `TCP`
			`TCP`
			
			`HTTP`
			
			`HTTPS`

Nama	Deskripsi	Default	Nilai yang valid

StickinessEnabled	Atur ke true untuk mengaktifkan sesi lekat.
	
	Opsi ini hanya berlaku untuk lingkungan dengan application load balancer.
		`'false'`	`'false'`
			
			`'true'`

StickinessLBCookie Duration	Masa hidup, dalam hitungan detik, dari cookie sesi lekat. Opsi ini hanya berlaku untuk lingkungan dengan application load balancer.	`86400` (satu hari)	1 untuk `604800`
StickinessType	Atur `lb_cookie` agar menggunakan cookie untuk sesi lekat. Opsi ini hanya berlaku untuk lingkungan dengan application load balancer.	`lb_cookie`	`lb_cookie`
UnhealthyThreshold Count	Jumlah permintaan gagal berturut-turut sebelum Elastic Load Balancing mengubah status kondisi instans.	`5`	2 untuk `10`

aws:elasticbeanstalk:environment:process:process_name

Konfigurasikan proses tambahan untuk lingkungan Anda.

Namespace: **`aws:elasticbeanstalk:environment:process:`*`process_name`***

Nama	Deskripsi	Default	Nilai yang valid
DeregistrationDelay	Jumlah waktu, dalam detik, untuk menunggu permintaan aktif selesai sebelum membatalkan pendaftaran.	`20`	`0` untuk `3600`
HealthCheckInterval	Interval, dalam hitungan detik, Elastic Load Balancing memeriksa kesehatan EC2 instans Amazon untuk aplikasi Anda.	Dengan classic load balancer atau application load balancer: `15` Dengan penyeimbang beban jaringan: `30`	Dengan classic load balancer atau application load balancer: `5` ke `300` Dengan penyeimbang beban jaringan: `10`, `30`
HealthCheckPath	Jalur pengiriman permintaan HTTP untuk pemeriksaan kondisi.	`/`	Jalur yang dapat dirutekan.
HealthCheckTimeout	Jumlah waktu, dalam detik, untuk menunggu respons selama pemeriksaan kondisi. Opsi ini hanya berlaku untuk lingkungan dengan application load balancer.	`5`	`1` untuk `60`

Nama	Deskripsi	Default	Nilai yang valid
HealthyThresholdCount	Jumlah permintaan sukses berturut-turut sebelum Elastic Load Balancing mengubah status kondisi instans.	Dengan classic load balancer atau application load balancer: `3` Dengan penyeimbang beban jaringan: `5`	2 untuk `10`
MatcherHTTPCode	Daftar kode HTTP (s) yang dipisahkan koma yang menunjukkan instans sehat. Opsi ini hanya berlaku untuk lingkungan dengan network load balancer atau application load balancer.	`200`	Dengan application load balancer: `200` ke `499` Dengan penyeimbang beban jaringan: `200` ke `399`
Port	Port yang didengarkan proses.	`80`	1 untuk `65535`

Nama	Deskripsi	Default	Nilai yang valid

Protocol	Protokol yang digunakan proses.
	
	Dengan application load balancer, Anda hanya dapat mengatur opsi ini untuk `HTTP` atau `HTTPS`.
	
	Dengan penyeimbang beban jaringan, Anda hanya dapat mengatur opsi ini untuk `TCP`.
		Dengan classic load balancer atau application load balancer: `HTTP`
		
		Dengan penyeimbang beban jaringan: `TCP`
			`TCP`
			
			`HTTP`
			
			`HTTPS`

StickinessEnabled	Atur ke true untuk mengaktifkan sesi lekat.
	
	Opsi ini hanya berlaku untuk lingkungan dengan application load balancer.
		`'false'`	`'false'`
			
			`'true'`

StickinessLBCookieDuration	Masa hidup, dalam hitungan detik, dari cookie sesi lekat. Opsi ini hanya berlaku untuk lingkungan dengan application load balancer.	`86400` (satu hari)	1 untuk `604800`

Nama	Deskripsi	Default	Nilai yang valid
StickinessType	Atur `lb_cookie` agar menggunakan cookie untuk sesi lekat. Opsi ini hanya berlaku untuk lingkungan dengan application load balancer.	`lb_cookie`	`lb_cookie`
UnhealthyThreshold Count	Jumlah permintaan gagal berturut-turut sebelum Elastic Load Balancing mengubah status kondisi instans.	5	2 untuk `10`

aws:elasticbeanstalk:environment:proxy:staticfiles

Anda dapat menggunakan namespace berikut untuk mengonfigurasi server proksi agar melayani file statis. Ketika server proksi menerima permintaan untuk file di bawah jalur yang ditentukan, server menyajikan file langsung daripada merutekan permintaan ke aplikasi Anda. Hal ini mengurangi jumlah permintaan yang harus diproses oleh aplikasi Anda harus.

Memetakan jalur yang dilayani oleh server proksi ke folder dalam kode sumber Anda yang berisi aset statis. Setiap pilihan yang Anda tetapkan dalam namespace ini memetakan jalan yang berbeda.

> **Note**
>
> Namespace ini berlaku untuk cabang platform berbasis Amazon Linux 2 dan yang lebih baru. Jika lingkungan Anda menggunakan versi platform berbasis Amazon Linux AMI (sebelumnya Amazon Linux 2), lihat the section called “Opsi spesifik platform” untuk namespace file statis spesifik platform.

Namespace: **`aws:elasticbeanstalk:environment:proxy:staticfiles`**

Nama	Nilai
Jalur tempat server proksi melayani file. Mulai nilai dengan `/`. Misalnya, tentukan `/images` untuk melayani file di*`subdomain`* `.elasticbeanstalk.com/images` .	Nama folder yang berisi file. Misalnya, tentukan `staticimages` untuk menyajikan file dari folder bernama `staticimages` di tingkat atas bundel sumber Anda.

aws:elasticbeanstalk:healthreporting:system

Konfigurasikan pelaporan kondisi yang ditingkatkan untuk lingkungan Anda.

Namespace: **`aws:elasticbeanstalk:healthreporting:system`**

Nama	Deskripsi	Default	Nilai yang valid

SystemType	Sistem pelaporan kondisi (dasar atau ditingkatkan). Pelaporan kondisi yang ditingkatkan memerlukan peran layanan dan versi 2 atau versi platform yang lebih baru.
	
	Note
	Jika Anda menggunakan konsol Elastic Beanstalk atau EB CLI untuk membuat lingkungan, Anda tidak dapat menetapkan opsi ini di file konfigurasi. Konsol dan EB CLI mengganti opsi ini dengan nilai yang disarankan.
		`basic`	`basic`
			
			`enhanced`

ConfigDocument	Dokumen JSON yang menjelaskan lingkungan dan metrik instance untuk dipublikasikan. CloudWatch	Tidak ada	

Nama	Deskripsi	Default	Nilai yang valid

EnhancedHealthAuthEnabled	Memungkinkan otorisasi untuk API internal yang digunakan Elastic Beanstalk untuk mengkomunikasikan informasi kondisi yang ditingkatkan dari instans lingkungan Anda ke layanan Elastic Beanstalk.
	
	Untuk informasi selengkapnya, lihat the section called “Peran kondisi yang ditingkatkan”.
	
	Note
	Opsi ini hanya berlaku untuk pelaporan kesehatan yang ditingkatkan (seperti kapan `SystemType` disetel `keenhanced`).
		`true`	`true`
			
			`false`

HealthCheckSuccessThreshold	Menurunkan ambang batas untuk instans agar lulus pemeriksaan kondisi.
	
	Note
	Jika Anda menggunakan konsol Elastic Beanstalk untuk membuat lingkungan, Anda tidak dapat menetapkan opsi ini di file konfigurasi. Konsol tersebut mengganti pilihan ini dengan nilai yang disarankan.
		`Ok`	`Ok`
			
			`Warning`
			
			`Degraded`
			
			`Severe`

aws:elasticbeanstalk:hostmanager

Konfigurasikan EC2 instance di lingkungan Anda untuk mengunggah log yang diputar ke Amazon S3.

Namespace: **`aws:elasticbeanstalk:hostmanager`**

Nama	Deskripsi	Default	Nilai valid
LogPublicationControl	Salin file log EC2 instans Amazon untuk aplikasi Anda ke bucket Amazon S3 yang terkait dengan aplikasi Anda.	`false`	`true` `false`

aws:elasticbeanstalk:managedactions

Konfigurasikan pembaruan platform terkelola untuk lingkungan Anda.

Namespace: **`aws:elasticbeanstalk:managedactions`**

Nama	Deskripsi	Default	Nilai yang valid
ManagedActionsEnabled	Aktifkan pembaruan platform terkelola. Saat Anda menetapkan ini ke `true`, Anda juga harus menentukan `PreferredStartTime` dan `UpdateLevel`.	`false`	`true` `false`
PreferredStartTime	Mengonfigurasi jendela pemeliharaan untuk tindakan terkelola di UTC. Misalnya, `"Tue:09:00"`.	Tidak ada	Hari dan waktu di *day*:*hour*:*minute* format.
ServiceRoleForManagedUpdates	Nama IAM role yang digunakan Elastic Beanstalk untuk melakukan pembaruan platform terkelola untuk lingkungan Anda.	Tidak ada	Sama seperti `ServiceRole` atau

Nama	Deskripsi	Default	Nilai yang valid
	Anda dapat menggunakan peran yang sama yang Anda tentukan untuk opsi `ServiceRole` dari namespace `aws:elasticbeanstalk:environment`, atau peran yang terhubung layanan terkelola akun Anda. Pada kasus terakhir, jika akun tidak memiliki peran yang terhubung layanan pembaruan terkelola, Elastic Beanstalk akan membuatnya.		`AWSServiceRoleForElasticBeanstalkManagedUpdates`

aws:elasticbeanstalk:manageactions:platformupdate

Konfigurasikan pembaruan platform terkelola untuk lingkungan Anda.

Namespace: **`aws:elasticbeanstalk:managedactions:platformupdate`**

Nama	Deskripsi	Default	Nilai yang valid
UpdateLevel	Tingkat pembaruan tertinggi yang diterapkan dengan pembaruan platform terkelola. Platform berversi *`major`*. *`minor`*. *`patch`*. Sebagai contoh, 2.0.8 mempunyai versi major 2, versi minor 0, dan versi patch 8.	Tidak ada	`patch` hanya untuk pembaruan versi patch. `minor` untuk pembaruan minor dan versi patch.
InstanceRefreshEnabled	Aktifkan penggantian instans mingguan.	`false`	`true`

Nama	Deskripsi	Default	Nilai yang valid
	Hal ini memerlukan `ManagedActionsEnabled` untuk ditetapkan ke `true`.		`false`

aws:elasticbeanstalk:monitoring

Konfigurasikan lingkungan Anda untuk menghentikan EC2 instance yang gagal dalam pemeriksaan kesehatan.

Namespace: **`aws:elasticbeanstalk:monitoring`**

Nama	Deskripsi	Default	Nilai yang valid

Automatically Terminate Unhealthy Instances	Hentikan instans jika gagal dalam pemeriksaan kondisi.
	
	Note
	Opsi ini hanya didukung pada lingkungan warisan. Opsi ini menentukan kondisi sebuah instans berdasarkan kemampuan untuk mencapainya dan metrik berbasis instans lainnya. Elastic Beanstalk tidak menyediakan cara untuk secara otomatis mengakhiri instans
		`true`	`true`
			
			`false`

Nama	Deskripsi	Default	Nilai yang valid
	berdasarkan kondisi aplikasi.		

aws:elasticbeanstalk:sns:topic

Konfigurasikan notifikasi untuk lingkungan Anda.

Namespace: **`aws:elasticbeanstalk:sns:topics`**

Nama	Deskripsi	Default	Nilai yang valid
Notification Endpoint	Titik akhir tempat Anda ingin diberi tahu tentang peristiwa penting yang memengaruhi aplikasi Anda. **Note** Jika Anda menggunakan konsol Elastic Beanstalk untuk membuat lingkungan, Anda tidak dapat menetapkan opsi ini di file konfigurasi. Konsol tersebut mengganti pilihan ini dengan nilai yang disarankan.	Tidak ada	

Notification Protocol	Protokol yang digunakan untuk mengirim notifikasi ke titik akhir Anda.	`email`	`http`
			
			`https`
			
			`email`
			
			`email-json`

Nama	Deskripsi	Default	Nilai yang valid
			`sqs`
Notification Topic ARN	Amazon Resource Name (ARN) untuk topik langganan Anda.	Tidak ada	
Notification Topic Name	Nama topik yang langganan Anda.	Tidak ada	

aws:elasticbeanstalk:sqsd

Konfigurasikan antrean Amazon SQS untuk lingkungan pekerja.

Namespace: **`aws:elasticbeanstalk:sqsd`**

Nama	Deskripsi	Default	Nilai yang valid
WorkerQueueURL	URL antrean tempat daemon di tingkat lingkungan pekerja membaca pesan. **Note** Saat Anda tidak menentukan nilai, antrean yang dibuat Elastic Beanstalk secara otomatis adalah antrean Amazon SQS standar. Ketika Anda memberikan nilai, Anda dapat memberikan URL baik standar atau antrean Amazon SQS FIFO. Sadarilah bahwa jika Anda memberikan antrean	Secara otomatis dihasilkan	Jika Anda tidak menentukan nilai, maka Elastic Beanstalk secara otomatis membuat antrean.

Nama	Deskripsi	Default	Nilai yang valid
	FIFO, tugas berkala tidak didukung.		
HttpPath	Jalur relatif ke aplikasi yang mengirim pesan HTTP POST.	/	

MimeType	Jenis MIME dari pesan yang dikirim dalam permintaan HTTP POST.	`application/json`	`application/json`
			
			`application/x-www-form-urlencoded`
			
			`application/xml`
			
			`text/plain`
			
			Tipe MIME khusus.

HttpConnections	Jumlah maksimum koneksi bersamaan ke aplikasi apa pun yang ada dalam EC2 instans Amazon. **Note** Jika Anda menggunakan konsol Elastic Beanstalk untuk membuat lingkungan, Anda tidak dapat menetapkan opsi ini di file konfigurasi. Konsol tersebut mengganti pilihan ini dengan nilai yang disarankan.	`50`	1 untuk `100`

Nama	Deskripsi	Default	Nilai yang valid
ConnectTimeout	Jumlah waktu, dalam detik, untuk menunggu koneksi yang berhasil ke aplikasi.	5	1 untuk 60
InactivityTimeout	Jumlah waktu, dalam detik, untuk menunggu respons pada koneksi yang ada ke aplikasi. Pesan diproses ulang sampai daemon menerima respon 200 (OK) dari aplikasi di tingkat lingkungan pekerja atau `RetentionPeriod` kedaluwarsa.	299	1 untuk 36000
VisibilityTimeout	Jumlah waktu, dalam detik, pesan masuk dari antrean Amazon SQS terkunci untuk diproses. Setelah jumlah waktu yang dikonfigurasi terlampaui, maka pesan tersebut kembali ditampilkan dalam antrean untuk dibaca daemon lain.	300	0 untuk 43200
ErrorVisibilityTimeout	Jumlah waktu, dalam detik, yang berlalu sebelum Elastic Beanstalk mengembalikan pesan ke antrean Amazon SQS setelah upaya pemrosesan gagal dengan kesalahan eksplisit.	2 detik	0 hingga 43200 detik
Retention Period	Jumlah waktu, dalam detik, pesan valid dan diproses secara aktif.	345600	60 untuk 1209600

Nama	Deskripsi	Default	Nilai yang valid
MaxRetries	Jumlah maksimum upaya yang Elastic Beanstalk coba untuk mengirim pesan ke aplikasi web yang akan memprosesnya sebelum memindahkan pesan ke antrean surat mati.	`10`	1 untuk `100`

aws:elasticbeanstalk:trafficsplitting

Konfigurasikan penerapan pemisahan lalu lintas untuk lingkungan Anda.

Namespace ini berlaku ketika Anda menetapkan opsi `DeploymentPolicy` dari namespace aws:elasticbeanstalk:command ke `TrafficSplitting`. Untuk informasi selengkapnya tentang paket deployment, lihat the section called “Pilihan deployment”.

Namespace: **`aws:elasticbeanstalk:trafficsplitting`**

Nama	Deskripsi	Default	Nilai yang valid
NewVersionPercent	Persentase awal lalu lintas klien masuk yang dialihkan Elastic Beanstalk ke instans lingkungan menjalankan versi aplikasi baru yang Anda terapkan.	`10`	1 untuk `100`
EvaluationTime	Periode waktu, dalam menit, Elastic Beanstalk menunggu setelah deployment awal kondisi sebelum lanjut mengalihkan semua lalu lintas klien masuk ke versi aplikasi baru yang Anda terapkan.	`5`	3 untuk `600`

aws:elasticbeanstalk:xray

Jalankan AWS X-Ray daemon untuk menyampaikan informasi jejak dari aplikasi terintegrasi X-Ray Anda.

Namespace: **`aws:elasticbeanstalk:xray`**

Nama	Deskripsi	Default	Nilai yang valid

`XRayEnabled`	Tetapkan ke `true` untuk menjalankan daemon X-Ray pada instans di lingkungan Anda.	`false`	`true`
			
			`false`

aws:elb:healthcheck

Konfigurasikan pemeriksaan kondisi untuk Classic Load Balancer.

Namespace: **`aws:elb:healthcheck`**

Nama	Deskripsi	Default	Nilai yang valid
HealthyThreshold	Jumlah permintaan sukses berturut-turut sebelum Elastic Load Balancing mengubah status kondisi instans.	`3`	2 untuk `10`
Interval	Interval di mana Elastic Load Balancing memeriksa kesehatan EC2 instans Amazon aplikasi Anda.	`10`	5 untuk `300`
Timeout	Jumlah waktu, dalam detik, saat Elastic Load Balancing menunggu respons sebelum Elastic Beanstalk merespon instans nonresponsif.	`5`	2 untuk `60`
UnhealthyThreshold	Jumlah permintaan gagal berturut-turut sebelum Elastic Load Balancing mengubah status kondisi instans.	`5`	2 untuk `10`
(tidak digunakan lagi) Target	Tujuan pada instans backend tempat pemeriksaan kondisi dikirim. Gunakan `Application Healthcheck URL` di namespace `aws:elasticbeanstalk:application` sebagai gantinya.	`TCP:80`	Target dalam format *`PROTOCOL`*`:`*`PORT`*`/`*`PATH`

aws:elb:loadbalancer

Konfigurasikan Classic Load Balancer lingkungan Anda.

Beberapa opsi di namespace ini tidak lagi didukung demi opsi khusus pendengar di namespace. aws:elb:listener Dengan opsi ini yang tidak didukung lagi, Anda hanya dapat mengonfigurasi dua pendengar (satu aman dan satu tidak aman) pada port standar.

Namespace: **`aws:elb:loadbalancer`**

Nama	Deskripsi	Default	Nilai valid

CrossZone	Mengonfigurasi penyeimbang beban untuk merutekan lalu lintas secara merata di semua instans di semua Availability Zones bukan hanya di setiap zona.
	
	Note
	Jika Anda menggunakan konsol Elastic Beanstalk atau EB CLI untuk membuat lingkungan, Anda tidak dapat menetapkan opsi ini di file konfigurasi. Konsol dan EB CLI mengganti opsi ini dengan nilai yang disarankan.
		`false`	`true`
			
			`false`

SecurityGroups	Tetapkan satu atau lebih grup keamanan yang Anda buat untuk penyeimbang beban. Diperlukan jika `DisableDefaultEC2SecurityGroup` (aws:autoscaling:launchconfiguration) diatur `ketrue`. Memuat lingkungan seimbang yang telah memilih keluar dari grup keamanan Elastic Beanstalk EC2 default harus menyediakan satu atau beberapa grup keamanan dengan	Tidak ada	Satu atau lebih kelompok keamanan IDs.

Nama	Deskripsi	Default	Nilai valid
	opsi ini. Untuk informasi selengkapnya, lihat Mengelola kelompok EC2 keamanan.		
ManagedSecurityGroup	Tetapkan grup keamanan yang ada ke penyeimbang beban untuk lingkungan Anda, alih-alih membuat grup yang baru. Untuk menggunakan pengaturan ini, perbarui pengaturan `SecurityGroups` di namespace ini untuk menyertakan ID grup keamanan Anda, dan hapus ID grup keamanan yang dibuat secara otomatis, jika dibuat. Untuk mengizinkan lalu lintas dari penyeimbang beban ke EC2 instans lingkungan Anda, Elastic Beanstalk menambahkan aturan ke grup keamanan instans yang memungkinkan lalu lintas masuk dari grup keamanan terkelola.	Tidak ada	ID grup keamanan.

(tidak digunakan lagi) LoadBalancerHTTPPort	Port untuk mendengarkan pendengar yang tidak aman.	`80`	`OFF`
			
			`80`

(tidak digunakan lagi) LoadBalancerPortProtocol	Protokol yang digunakan pada pendengar yang tidak aman.	`HTTP`	`HTTP`
			
			`TCP`

(tidak digunakan lagi) LoadBalancerHTTPSPort	Port untuk mendengarkan pendengar yang aman.	`OFF`	`OFF`
			
			`443`
			
			`8443`

(tidak digunakan lagi) LoadBalancerSSLPortProtocol	Protokol untuk digunakan pada pendengar yang aman.	`HTTPS`	`HTTPS`
			
			`SSL`

Nama	Deskripsi	Default	Nilai valid
(tidak digunakan lagi) SSLCertificateId	Sertifikat SSL Amazon Resource Name (ARN) untuk mengikat pendengar yang aman.	Tidak ada	

aws:elb:listener

Konfigurasikan pendengar default (port 80) pada Classic Load Balancer.

Namespace: **`aws:elb:listener`**

Nama	Deskripsi	Default	Nilai valid
ListenerProtocol	Protokol yang digunakan oleh pendengar.	`HTTP`	`HTTP TCP`
InstancePort	Port yang digunakan pendengar ini untuk berkomunikasi dengan EC2 instance.	80	1 untuk `65535`

InstanceProtocol	Protokol yang digunakan pendengar ini untuk berkomunikasi dengan EC2 instance.
	
	Instans ini harus berada di lapisan protokol internet yang sama dengan `ListenerProtocol` . Instans ini juga harus memiliki tingkat keamanan yang sama seperti pendengar lain yang menggunakan `InstancePort` yang sama dengan pendengar ini.
	
	Misalnya, jika `ListenerProtocol` adalah `HTTPS` (lapisan aplikasi, yang menggunakan koneksi yang aman), Anda dapat menetapkan `InstanceProtocol` ke `HTTP` (juga pada lapisan aplikasi, yang menggunakan koneksi yang tidak aman). Jika, selain itu, Anda menetapkan `InstancePort` ke `80`, Anda harus
		`HTTP` saat `ListenerProtocol H`
		
		`TCP` saat `ListenerProtocol TC`
			`HTTP` atau `HTTPS` saat `ListenerProtocol` `HTTP` atau `HTTPS`
			
			`TCP` atau `SSL` saat `ListenerProtocol` `TCP` atau `SSL`

Nama	Deskripsi	Default	Nilai valid
	menetapkan `InstanceProtocol` ke `HTTP` di semua pendengar lainnya dengan `InstancePort` ditetapkan ke `80`.		
PolicyNames	Daftar nama kebijakan yang dipisahkan koma yang diterapkan ke port untuk pendengar ini. Kami menyarankan Anda menggunakan LoadBalancerPorts opsi aws:elb:policies namespace sebagai gantinya.	Tidak ada	
ListenerEnabled	Menentukan apakah pendengar ini diaktifkan. Jika Anda menentukan `false`, pendengar tidak disertakan dalam penyeimbang beban.	`true`	`true` `false`

aws:elb:listener:listener_port

Konfigurasikan pendengar tambahan pada Classic Load Balancer.

Namespace: **`aws:elb:listener:`*`listener_port`***

Nama	Deskripsi	Default	Nilai valid
ListenerProtocol	Protokol yang digunakan oleh pendengar.	`HTTP`	`HTTP` `HTTPS` `TCP` `SSL`
InstancePort	Port yang digunakan pendengar ini untuk berkomunikasi dengan EC2 instance.	Sama seperti*list port* .	1 untuk 65535
InstanceProtocol	Protokol yang digunakan pendengar ini untuk berkomunikasi dengan EC2 instance. Instans ini harus berada di lapisan protokol internet yang sama dengan `ListenerP`	`HTTP` saat `ListenerProtocol`	`HTTP` atau `HTTPS` saat `ListenerProtocol`

Nama	Deskripsi	Default	Nilai valid

	`rotocol` . Instans ini juga harus memiliki tingkat keamanan yang sama seperti pendengar lain yang menggunakan `InstancePort` yang sama dengan pendengar ini.
	
	Misalnya, jika `ListenerProtocol` adalah `HTTPS` (lapisan aplikasi, yang menggunakan koneksi yang aman), Anda dapat menetapkan `InstanceProtocol` ke `HTTP` (juga pada lapisan aplikasi, yang menggunakan koneksi yang tidak aman). Jika, selain itu, Anda menetapkan `InstancePort` ke `80`, Anda harus menetapkan `InstanceProtocol` ke `HTTP` di semua pendengar lainnya dengan `InstancePort` ditetapkan ke `80`.
		`HTTP` atau `HTTPS`
		
		`TCP` saat `ListenerProtocol` `TCP` atau `SSL`
			`HTTP` atau `HTTPS`
			
			`TCP` atau `SSL` saat `ListenerProtocol` `TCP` atau `SSL`

PolicyNames	Daftar nama kebijakan yang dipisahkan koma yang diterapkan ke port untuk pendengar ini. Kami menyarankan Anda menggunakan LoadBalancerPorts opsi aws:elb:policies namespace sebagai gantinya.	Tidak ada	
SSLCertificateId	Sertifikat SSL Amazon Resource Name (ARN) yang mengikat pendengar.	Tidak ada	
ListenerEnabled	Menentukan apakah pendengar ini diaktifkan. Jika Anda menentukan `false`, pendengar tidak disertakan dalam penyeimbang beban.	`true` jika pilihan lain diatur. `false` sebaliknya.	`true` `false`

aws:elb:policies

Ubah kebijakan kelekatan default dan kebijakan penyeimbang global untuk Classic Load Balancer.

Namespace: **`aws:elb:policies`**

Nama	Deskripsi	Default	Nilai yang valid

ConnectionDraining Enabled	Menentukan apakah penyeimbang beban mempertahankan koneksi yang ada untuk instans yang menjadi tidak sehat atau yang dibatalkan untuk menyelesaikan permintaan dalam proses.
	
	Note
	Jika Anda menggunakan konsol Elastic Beanstalk atau EB CLI untuk membuat lingkungan, Anda tidak dapat menetapkan opsi ini di file konfigurasi. Konsol dan EB CLI mengganti opsi ini dengan nilai yang disarankan.
		`false`	`true`
			
			`false`

ConnectionDraining Timeout	Jumlah maksimum detik saat penyeimbang beban mempertahankan koneksi yang ada ke instans selama pengurasan koneksi sebelum menutup koneksi secara paksa. **Note** Jika Anda menggunakan konsol Elastic Beanstalk untuk membuat lingkungan, Anda tidak dapat menetapkan opsi ini di file konfigurasi. Konsol tersebut	`20`	1 untuk `3600`

Nama	Deskripsi	Default	Nilai yang valid
	mengganti pilihan ini dengan nilai yang disarankan.		
ConnectionSettingIdleTimeout	Jumlah waktu, dalam detik, penyeimbang beban menunggu data yang akan dikirim atau diterima melalui koneksi. Jika tidak ada data yang dikirim atau diterima setelah periode waktu ini berlalu, penyeimbang beban menutup koneksi.	`60`	1 untuk `3600`
LoadBalancerPorts	Daftar port pendengar yang dipisahkan dengan koma yang diterapkan (`AWSEB-ELB-StickinessPolicy`) kebijakan default.	Tidak ada	Anda dapat menggunakan `:all` untuk mengindikasikan semua port pendengar
Stickiness Cookie Expiration	Jumlah waktu, dalam detik, setiap cookie valid. Menggunakan kebijakan default (`AWSEB-ELB-StickinessPolicy`).	`0`	`0` untuk `1000000`
Stickiness Policy	Mengikat sesi pengguna ke instans server tertentu sehingga semua permintaan yang datang dari pengguna selama sesi dikirim ke instans server yang sama. Menggunakan kebijakan default (`AWSEB-ELB-StickinessPolicy`).	`false`	`true false`

aws:elb:policies:policy_name

Buat kebijakan penyeimbang beban tambahan untuk Classic Load Balancer.

Namespace: **`aws:elb:policies:`*`policy_name`***

Nama	Deskripsi	Default	Nilai yang valid
CookieName	Nama cookie yang dihasilkan aplikasi yang mengontrol masa pakai sesi kebijakan `AppCookieStickinessPolicyType` . Kebijakan ini hanya dapat dikaitkan dengan HTTP/HTTPS pendengar.	Tidak ada	
InstancePorts	Daftar port instans yang dipisahkan dengan koma yang berlaku untuk kebijakan ini.	Tidak ada	Daftar port, atau `:all`
LoadBalancerPorts	Daftar port pendengar yang dipisahkan koma yang berlaku untuk kebijakan ini.	Tidak ada	Daftar port, atau `:all`
ProxyProtocol	Untuk kebijakan `ProxyProtocolPolicyType` , menentukan apakah akan menyertakan alamat IP dan port permintaan asal untuk pesan TCP. Kebijakan ini hanya dapat dikaitkan dengan TCP/SSL pendengar.	Tidak ada	`true false`
PublicKey	Isi kunci publik untuk `PublicKeyPolicyType` kebijakan yang akan digunakan saat mengautentikasi server backend atau server. Kebijakan ini tidak dapat diterapkan langsung ke server backend atau pendengar. Server ini harus menjadi bagian dari kebijakan `BackendServerAuthenticationPolicyType` .	Tidak ada	
PublicKeyPolicyNames	Daftar nama kebijakan yang dipisahkan koma (dari kebijakan) untuk `PublicKeyPolicyType` `BackendServerAuthenticationPolicyType` kebijakan	Tidak ada	

Nama	Deskripsi	Default	Nilai yang valid
	yang mengontrol otentikasi ke server backend atau server. Kebijakan ini hanya dapat dikaitkan dengan server backend yang menggunakan HTTPS/SSL.		
SSLProtocols	Daftar protokol SSL yang dipisahkan dengan koma yang akan diaktifkan untuk kebijakan `SSLNegotiationPolicyType` yang menentukan cipher dan protokol yang diterima oleh penyeimbang beban. Kebijakan ini hanya dapat dikaitkan dengan HTTPS/SSL pendengar.	Tidak ada	
SSLReferencePolicy	Nama kebijakan keamanan yang telah ditentukan sebelumnya yang mematuhi praktik terbaik AWS keamanan dan yang ingin Anda aktifkan untuk `SSLNegotiationPolicyType` kebijakan yang mendefinisikan sandi dan protokol yang diterima oleh penyeimbang beban. Kebijakan ini hanya dapat dikaitkan dengan HTTPS/SSL pendengar.	Tidak ada	
Stickiness Cookie Expiration	Jumlah waktu, dalam detik, setiap cookie valid.	`0`	`0` untuk `1000000`
Stickiness Policy	Mengikat sesi pengguna ke instans server tertentu sehingga semua permintaan yang datang dari pengguna selama sesi dikirim ke instans server yang sama.	`false`	`true false`

aws:elbv2:listener:default

Konfigurasikan pendengar default (port 80) pada Application Load Balancer atau Network Load Balancer.

Namespace ini tidak berlaku untuk lingkungan yang menggunakan penyeimbang beban bersama. Penyeimban beban bersama tidak memiliki pendengar default.

Namespace: **`aws:elbv2:listener:default`**

Nama	Deskripsi	Default	Nilai yang valid
DefaultProcess	Nama proses untuk meneruskan lalu lintas ke saat tidak ada aturan yang cocok.	`default`	Nama proses.

ListenerEnabled	Atur ke `false` untuk menonaktifkan pendengar. Anda dapat menggunakan opsi ini untuk menonaktifkan pendengar default pada port 80.	`true`	`true`
			
			`false`

Protocol	Protokol lalu lintas untuk memproses.	Dengan application load balancer: `HTTP` Dengan penyeimbang beban jaringan: `TCP`	Dengan application load balancer: `HTTP`, `HTTPS` Dengan penyeimbang beban jaringan: `TCP`
Rules	Daftar aturan yang diterapkan ke pendengar Opsi ini hanya berlaku untuk lingkungan dengan Application Load Balancer.	Tidak ada	Daftar nama aturan yang dipisahkan dengan koma.

Nama	Deskripsi	Default	Nilai yang valid
SSLCertificateArns	Sertifikat SSL Amazon Resource Name (ARN) yang mengikat pendengar. Opsi ini hanya berlaku untuk lingkungan dengan Application Load Balancer.	Tidak ada	ARN sertifikat yang disimpan dalam IAM atau ACM.
SSLPolicy	Tentukan kebijakan keamanan yang diterapkan ke pendengar. Opsi ini hanya berlaku untuk lingkungan dengan Application Load Balancer.	Tidak ada (default ELB)	Nama kebijakan keamanan penyeimbang beban.

aws:elbv2:listener:listener_port

Konfigurasikan pendengar tambahan di Application Load Balancer atau Network Load Balancer.

> **Note**
>
> Untuk Application Load Balancer bersama, Anda hanya dapat menentukan opsi `Rule`. Opsi lainnya tidak berlaku untuk penyeimbang beban bersama.

Namespace: **`aws:elbv2:listener:`*`listener_port`***

Nama	Deskripsi	Default	Nilai yang valid
DefaultProcess	Nama proses tempat lalu lintas diteruska	`default`	Nama proses.

Nama	Deskripsi	Default	Nilai yang valid
	n ketika tidak ada aturan yang cocok.		
ListenerEnabled	Atur ke `false` untuk menonaktifkan pendengar. Anda dapat menggunakan opsi ini untuk menonaktifkan pendengar default pada port 80.	`true`	`true` `false`
Protocol	Protokol lalu lintas untuk memproses.	Dengan application load balancer: `HTTP` Dengan penyeimbang beban jaringan: `TCP`	Dengan application load balancer: `HTTP`, `HTTPS` Dengan penyeimbang beban jaringan: `TCP`

Nama	Deskripsi	Default	Nilai yang valid
Rules	Daftar aturan yang diterapkan ke pendengar Opsi ini hanya berlaku untuk lingkungan dengan Application Load Balancer. Jika lingkungan Anda menggunakan Application Load Balancer bersama, dan Anda tidak menentukan opsi ini untuk pendengar apa pun, Elastic Beanstalk secara otomatis menghubungkan aturan `default` dengan port 80 pendengar.	Tidak ada	Daftar nama aturan yang dipisahkan dengan koma.
SSLCertificateArns	Sertifikat SSL Amazon Resource Name (ARN) yang mengikat pendengar. Opsi ini hanya berlaku untuk lingkungan dengan Application Load Balancer.	Tidak ada	ARN sertifikat yang disimpan dalam IAM atau ACM.

Nama	Deskripsi	Default	Nilai yang valid
SSLPolicy	Tentukan kebijakan keamanan yang diterapkan ke pendengar. Opsi ini hanya berlaku untuk lingkungan dengan Application Load Balancer.	Tidak ada (default ELB)	Nama kebijakan keamanan penyeimbang beban.

aws:elbv2:listener:rule_name

Tentukan aturan pendengar untuk Application Load Balancer. Jika permintaan cocok dengan nama host atau jalur dalam aturan, penyeimbang beban meneruskannya ke proses yang ditentukan. Untuk menggunakan aturan, tambahkan penyeimbang beban ke pendengar dengan opsi `Rules` di namespace `aws:elbv2:listener:listener_port`.

> **Note**
>
> Namespace ini tidak berlaku untuk lingkungan dengan penyeimbang beban jaringan.

Namespace: **`aws:elbv2:listenerrule:rule_name`**

Nama	Deskripsi	Default	Nilai yang valid
HostHeade rs	Daftar nama host yang cocok. Misalnya, `my.example.com` .	Penyeimbang beban khusus: Tidak ada Penyeimbang beban bersama: CNAME lingkungan	Setiap nama dapat berisi hingga 128 karakter. Pola dapat mencakup huruf besar dan huruf kecil, angka, tanda hubung (-), dan hingga tiga karakter wildcard (* cocok dengan nol

Nama	Deskripsi	Default	Nilai yang valid
			karakter atau lebih; ? sangat cocok dengan satu karakter). Anda dapat mencantum kan lebih dari satu nama, masing-masing dipisahkan dengan koma. Applicati on Load Balancer mendukung hingga lima gabungan aturan `HostHeader` dan `PathPattern` . Untuk informasi selengkapnya, lihat Kondisi host di Panduan Pengguna untuk Application Load Balancers.

Nama	Deskripsi	Default	Nilai yang valid
PathPatterns	Pola jalur untuk mencocokkan (misalnya,`/img/*`). Opsi ini hanya berlaku untuk lingkungan dengan application load balancer.	Tidak ada	Setiap pola dapat berisi hingga 128 karakter. Pola dapat mencakup huruf besar dan huruf kecil, angka, tanda hubung (—), dan hingga tiga karakter wildcard (`*` cocok dengan nol karakter atau lebih; `?` sangat cocok dengan satu karakter). Anda dapat menambahkan beberapa pola jalur dipisahkan koma. Application Load Balancer mendukung hingga lima gabungan aturan `HostHeader` dan `PathPattern` . Untuk informasi selengkapnya, lihat Kondisi jalur di Panduan Pengguna untuk Application Load Balancers.

Nama	Deskripsi	Default	Nilai yang valid
Priority	Prioritas aturan ini ketika beberapa aturan cocok. Jumlah yang lebih rendah diutamakan. Tidak ada dua aturan yang bisa memiliki prioritas yang sama. Dengan penyeimbang beban bersama, Elastic Beanstalk memperlakukan prioritas aturan sebagai relatif di seluruh lingkungan berbagi, dan memetakannya ke prioritas mutlak selama pembuatan.	1	1 untuk `1000`
Process	Nama proses untuk meneruskan lalu lintas ketika aturan ini cocok dengan permintaan.	`default`	Nama proses.

aws:elbv2:loadbalancer

Konfigurasikan Application Load Balancer.

Untuk penyeimbang beban bersama, hanya opsi `SharedLoadBalancer` dan `SecurityGroups` valid.

> **Note**
>
> Namespace ini tidak berlaku untuk lingkungan dengan Network Load Balancer.

Namespace: **`aws:elbv2:loadbalancer`**

Nama	Deskripsi	Default	Nilai yang valid
AccessLogsS3Bucket	Bucket Amazon S3 tempat log akses disimpan. Bucket tersebut harus berada dalam Wilayah yang sama dengan lingkungan dan mengizinkan akses tulis penyeimbang beban.	Tidak ada	Nama bucket.

Nama	Deskripsi	Default	Nilai yang valid
AccessLogsS3Enabled	Aktifkan penyimpanan log akses.	`false`	`true` `false`
AccessLogsS3Prefix	Prefiks untuk menambahkan nama log akses. Secara default, penyeimbang beban mengunggah log ke direktori bernama AWSLogs di bucket yang Anda tentukan. Tentukan awalan untuk menempatkan AWSLogs direktori di dalam direktori lain.	Tidak ada	
IdleTimeout	Jumlah waktu, dalam detik, untuk menunggu permintaan selesai sebelum menutup koneksi ke klien dan instans.	Tidak ada	1 untuk `3600`
ManagedSecurityGroup	Tetapkan grup keamanan yang ada ke penyeimbang beban lingkungan Anda, alih-alih membuat grup keamanan baru. Untuk menggunakan pengaturan ini, perbarui `SecurityGroups` di namespace ini untuk menyertakan ID grup keamanan Anda, dan hapus ID grup keamanan yang dibuat secara otomatis, jika ada. Untuk mengizinkan lalu lintas dari penyeimbang beban ke EC2 instans untuk lingkungan Anda, Elastic Beanstalk menambahkan aturan ke grup keamanan instans Anda yang memungkinkan lalu lintas masuk dari grup keamanan terkelola.	Grup keamana yang dibuat Elastic Beanstal s untuk penyeim ng beban Anda.	ID grup keamanan.

Nama	Deskripsi	Default	Nilai yang valid
SecurityGroups	Daftar grup keamanan yang dilampirkan ke penyeimbang beban. Diperlukan jika `DisableDefaultEC2SecurityGroup` (aws:autoscaling:launchconfiguration) diatur `ketrue`. Memuat lingkungan seimbang yang telah memilih keluar dari grup keamanan Elastic Beanstalk EC2 default harus menyediakan satu atau beberapa grup keamanan dengan opsi ini. Untuk informasi selengkapnya, lihat Mengelola kelompok EC2 keamanan. Untuk penyeimbang beban bersama, jika Anda tidak menentukan nilai ini, Elastic Beanstalk memeriksa apakah grup keamanan yang dikelolanya sudah terpasang pada penyeimbang beban. Jika salah satunya tidak dilampirkan pada penyeimbang beban, Elastic Beanstalk membuat grup keamanan dan melampirkannya ke penyeimbang beban. Elastic Beanstalk menghapus grup keamanan ini ketika lingkungan terakhir yang berbagi penyeimbang beban berakhir. Grup keamanan penyeimbang beban digunakan untuk mengatur aturan masuknya grup keamanan EC2 instans Amazon.	Grup keamana yang dibuat Elastic Beanstal untuk penyeim ng beban Anda.	Daftar kelompok keamanan yang dipisahkan koma. IDs

Nama	Deskripsi	Default	Nilai yang valid
SharedLoadBalancer	Amazon Resource Name (ARN) dari penyeimbang beban bersama. Opsi ini hanya relevan untuk Application Load Balancer. Opsi ini diperlukan ketika opsi `LoadBalancerIsShared` dari namespace aws:elasticbeanstalk:environment diatur ke `true`. Anda tidak dapat mengubah ARN penyeimbang beban bersama setelah lingkungan dibuat. Kriteria untuk nilai yang valid: • Ini harus merupakan penyeimbang beban aktif yang valid di AWS Wilayah tempat lingkungan berada. • ARN harus berada dalam Amazon Virtual Private Cloud (Amazon VPC) yang sama dengan lingkungan. • ARN tidak bisa menjadi penyeimbang beban yang dibuat oleh Elastic Beanstalk sebagai penyeimbang beban khusus untuk lingkungan lain. Anda dapat mengidentifikasi penyeimbang beban khusus ini dengan menggunakan prefiks `awseb-`. Contoh: `arn:aws:elasticloadbalancing:us-east-2:123456789012:lo`	Tidak ada	ARN dari penyeimbang beban valid yang memenuhi semua kriteria yang dijelaskan di sini.

Nama	Deskripsi	Default	Nilai yang valid
	`adbalancer/app/Fro ntEndLB/0dbf78d8ad96abbc`		

aws:rds:dbinstance

Konfigurasikan instans DB Amazon RDS terlampir.

Namespace: **`aws:rds:dbinstance`**

Nama	Deskripsi	Default	Nilai yang valid
DBAllocat edStorage	Ukuran penyimpanan basis data yang dialokasikan, ditentukan dalam gigabyte.	MySQL: `5` Oracle: `10` sqlserver-se: `200` sqlserver-ex: `30` sqlserver web: `30`	MySQL: `5-1024` Oracle: `10-1024` sqlserver: tidak dapat diubah

DBDeletio nPolicy	Menentukan apakah akan mempertah ankan, menghapus, atau membuat snapshot dari contoh DB ketika lingkungan dihentikan.
	Opsi ini bekerja bersama dengan`HasCoupledDatabase` , juga opsi namespace ini.
		`Delete`	`Delete`
			`Retain`
			`Snapshot`

Nama	Deskripsi	Default	Nilai yang valid
	Warning Menghapus hasil instans DB yang menyebabkan hilangnya data secara permanen.		

DBEngine	Nama mesin basis data yang akan digunakan untuk instans ini.	`mysql`	`mysql`
			`oracle-se1`
			`sqlserver-ex`
			`sqlserver-web`
			`sqlserver-se`
			`postgres`

DBEngineVersion	Nomor versi mesin basis data.	`5.5`	
DBInstanceClass	Tipe instans basis data.	`db.t2.micro` (`db.m1.large` untuk lingkungan yang tidak berjalan di Amazon VPC)	Untuk informasi selengkapnya, lihat Kelas Instans DB di Panduan Pengguna Amazon Relational Database Service.
DBPassword	Nama sandi pengguna master untuk instans basis data.	Tidak ada	

Nama	Deskripsi	Default	Nilai yang valid
DBSnapshotIdentifier	Pengidentifikasi untuk snapshot DB yang akan dipulihkan.	Tidak ada	
DBUser	Nama pengguna master untuk Instans DB.	ebroot	

HasCoupledDatabase	Menentukan apakah instance DB digabungkan ke lingkungan Anda. Jika diaktifkan`true`, Elastic Beanstalk membuat instans DB baru yang digabungkan ke lingkungan Anda. Jika diaktifkan`false`, Elastic Beanstalk memulai decoupling instance DB dari lingkungan Anda.
	
	Opsi ini bekerja bersama dengan`DBDeletionPolicy` , juga opsi namespace ini.
	
	Note
	Catatan: Jika Anda mengaktifkan nilai ini kembali `true` setelah memisahkan database sebelumnya, Elastic Beanstalk membuat database baru dengan pengaturan opsi database sebelumnya. Namun, untuk menjaga keamanan lingkungan Anda, itu tidak mempertahankan yang ada `DBUser` dan `DBPassword` pengaturan. Anda perlu menentukan `DBUser` dan `DBPassword` lagi.
		`false`	`true`
			
			`false`

Nama	Deskripsi	Default	Nilai yang valid

MultiAZDatabase	Menentukan apakah deployment instans basis data Multi-AZ perlu dibuat. Untuk informasi selengkapnya tentang penerapan Multi-AZ dengan Amazon Relational Database Service (RDS), lihat Wilayah dan Availability Zone di Panduan Pengguna Amazon Relational Database Service.	`false`	`true`
			
			`false`

Opsi spesifik platform

Beberapa platform Elastic Beanstalk menentukan namespace opsi yang khusus untuk platform. Namespace dan opsinya tercantum di bawah untuk setiap platform.

> **Note**
>
> Sebelumnya, pada versi platform berbasis Amazon Linux AMI (sebelumnya Amazon Linux 2), dua fitur berikut dan namespace masing-masing dianggap sebagai fitur spesifik platform, dan terdaftar di sini per platform:
>
> - Konfigurasi proksi untuk file statis — `aws:elasticbeanstalk:environment:proxy:staticfiles`
> - AWS X-Ray dukungan - `aws:elasticbeanstalk:xray`
>
> Di versi platform Amazon Linux 2, Elastic Beanstalk mengimplementasikan fitur ini secara konsisten di semua platform pendukung. Namespace terkait sekarang tercantum di halaman the section called “Opsi umum”. Kami terus menyebutkannya di halaman ini untuk platform yang memiliki namespace berbeda.

Platform

- Opsi platform Docker
- Buka opsi platform
- Pilihan platform Java SE

- Java dengan opsi platform Tomcat
- .NET Core pada opsi platform Linux
- Opsi platform .NET
- Opsi platform Node.js
- Opsi platform PHP
- Opsi platform Python
- Opsi platform Ruby

Opsi platform Docker

Opsi konfigurasi khusus Docker berikut ini berlaku untuk platform Docker dan Docker yang telah dikonfigurasi sebelumnya.

> **Note**
>
> Opsi konfigurasi ini tidak berlaku untuk
>
> - Platform Docker (Amazon Linux 2) dengan Docker Compose
> - Platform Multicontainer Docker (Amazon Linux AMI AL1) - platform ini sudah pensiun

Namespace: **`aws:elasticbeanstalk:environment:proxy`**

Nama	Deskripsi	Default	Nilai yang valid
ProxyServer	Menentukan server web yang digunakan sebagai proksi.	`nginx`	`nginx` none — Amazon Linux AM dan Docker dengan DC saja

Buka opsi platform

Opsi platform Amazon Linux AMI (pra-Amazon Linux 2)

Namespace: **`aws:elasticbeanstalk:container:golang:staticfiles`**

Anda dapat menggunakan namespace berikut untuk mengonfigurasi server proksi agar melayani file statis. Ketika server proksi menerima permintaan untuk file di bawah jalur yang ditentukan, server menyajikan file langsung daripada merutekan permintaan ke aplikasi Anda. Hal ini mengurangi jumlah permintaan yang harus diproses oleh aplikasi Anda harus.

Memetakan jalur yang dilayani oleh server proksi ke folder dalam kode sumber Anda yang berisi aset statis. Setiap pilihan yang Anda tetapkan dalam namespace ini memetakan jalan yang berbeda.

Nama	Nilai
Jalur tempat server proksi akan melayani file. Contoh: `/images` untuk melayani file di *`subdomain`* `.eleasticbeanstalk.com/images` .	Nama folder yang berisi file. Contoh: `staticimages` untuk melayani file dari folder yang dinamai `staticimages` di tingkat atas paket sumber Anda.

Pilihan platform Java SE

Opsi platform Amazon Linux AMI (pra-Amazon Linux 2)

Namespace: **`aws:elasticbeanstalk:container:java:staticfiles`**

Anda dapat menggunakan namespace berikut untuk mengonfigurasi server proksi agar melayani file statis. Ketika server proksi menerima permintaan untuk file di bawah jalur yang ditentukan, server menyajikan file langsung daripada merutekan permintaan ke aplikasi Anda. Hal ini mengurangi jumlah permintaan yang harus diproses oleh aplikasi Anda harus.

Memetakan jalur yang dilayani oleh server proksi ke folder dalam kode sumber Anda yang berisi aset statis. Setiap pilihan yang Anda tetapkan dalam namespace ini memetakan jalan yang berbeda.

Nama	Nilai
Jalur tempat server proksi akan melayani file.	Nama folder yang berisi file.

Nama	Nilai
Contoh: `/images` untuk melayani file di *`subdomain`* `.elasticbeanstalk.com/images` .	Contoh: `staticimages` untuk melayani file dari folder yang dinamai `staticimages` di tingkat atas paket sumber Anda.

Java dengan opsi platform Tomcat

Namespace: **`aws:elasticbeanstalk:application:environment`**

Nama	Deskripsi	Default	Nilai yang valid
JDBC_CONNECTION_STRING	Koneksi string ke basis data eksternal.	tidak ada	tidak ada

Lihat Variabel lingkungan dan pengaturan perangkat lunak lainnya untuk informasi selengkapnya.

Namespace: **`aws:elasticbeanstalk:container:tomcat:jvmoptions`**

Nama	Deskripsi	Default	Nilai yang valid
JVM Options	Lewati opsi baris perintah ke JVM saat memulai.	tidak ada	tidak ada
Xmx	Ukuran tumpukan JVM maksimum.	`256m`	tidak ada
XX:MaxPermSize	Bagian dari tumpukan JVM yang digunakan untuk menyimpan definisi kelas dan metadata terkait. **Note** Opsi ini hanya berlaku untuk versi Java lebih awal dari Java 8, dan tidak didukung pada platform Elastic	`64m`	T/A

Nama	Deskripsi	Default	Nilai yang valid
	Beanstalk Tomcat berbasis Amazon Linux 2 dan yang lebih baru.		
Xms	Ukuran tumpukan JVM awal.	`256m`	T/A
`optionName`	Tentukan pilihan JVM arbitrer selain yang ditentukan oleh platform Tomcat.	tidak ada	tidak ada

Namespace: **`aws:elasticbeanstalk:environment:proxy`**

Nama	Deskripsi	Default	Nilai yang valid

GzipCompression	Atur ke `false` untuk menonaktifkan kompresi respons.
	
	Hanya berlaku di Amazon Linux AMI (sebelumnya Amazon Linux 2) versi platform.
		`true`	`true`
			
			`false`

ProxyServer	Tetapkan proksi yang digunakan pada instans lingkungan Anda. Jika Anda menetapkan opsi `apache`, Elastic Beanstalk menggunakan Apache 2.4.
	
	Atur ke `apache/2.2` jika aplikasi Anda belum siap untuk bermigrasi dari Apache 2.2 karena pengaturan konfigurasi proksi yang tidak kompatibel. Nilai ini hanya berlaku di Amazon Linux AMI (sebelumnya Amazon Linux 2) versi platform.
	
	Atur ke `nginx` untuk menggunakan nginx. Ini adalah default yang dimulai dengan versi platform Amazon Linux 2.
		`nginx` (Amazon Linux 2)
		
		`apache` (Amazon Linux AMI)
			`apache`
			
			`apache/2.2` — Amazon Linux AMI saja
			
			`nginx`

Nama	Deskripsi	Default	Nilai yang valid
	Untuk informasi selengkapnya, lihat Mengonfigurasi server proksi.		

.NET Core pada opsi platform Linux

Namespace: **`aws:elasticbeanstalk:environment:proxy`**

Nama	Deskripsi	Default	Nilai yang valid
ProxyServer	Menentukan server web yang digunakan sebagai proksi.	`nginx`	`nginx` `none`

Opsi platform .NET

Namespace: **`aws:elasticbeanstalk:container:dotnet:apppool`**

Nama	Deskripsi	Default	Nilai yang valid
Target Runtime	Pilih versi .NET Framework untuk aplikasi Anda.	`4.0`	`2.0` `4.0`
Enable 32-bit Applications	Atur ke `True` untuk menjalankan aplikasi 32-bit.	`False`	`True` `False`

Opsi platform Node.js

Namespace: **`aws:elasticbeanstalk:environment:proxy`**

Nama	Deskripsi	Default	Nilai yang valid
ProxyServer	Tetapkan proksi yang digunakan pada instans lingkungan Anda.	`nginx`	`apache` `nginx`

Opsi platform Amazon Linux AMI (pra-Amazon Linux 2)

Namespace: **`aws:elasticbeanstalk:container:nodejs`**

Nama	Deskripsi	Default	Nilai yang valid
NodeCommand	Perintah yang digunakan untuk memulai aplikasi Node.js. Jika string kosong ditentukan, `app.js` digunakan, maka `server.js` , kemudian `npm start` dalam urutan itu.	""	tidak ada
NodeVersion	Versi Node.js. Misalnya, `4.4.6` Versi Node.js yang didukung bervariasi antara versi platform Node.js. Lihat Node.js di dokumen AWS Elastic Beanstalk Platform untuk daftar versi yang didukung saat ini. **Note** Ketika dukungan untuk versi Node.js yang Anda gunakan dihapus dari platform, Anda harus mengubah atau menghapus pengaturan versi sebelum melakukan pembaruan platform. Hal ini mungkin terjadi ketika kelemahan keamanan diidentifikasi untuk satu atau beberapa versi Node.js. Ketika ini terjadi, mencoba memperbarui ke versi baru platform yang tidak mendukung konfigurasi NodeVersiongagal. Untuk menghindari perlunya membuat lingkungan baru, ubah opsi NodeVersionkonfigurasi ke versi Node.js yang didukung oleh versi	bervariasi	bervariasi

Nama	Deskripsi	Default	Nilai yang valid
	platform lama dan yang baru, atau hapus pengaturan opsi, lalu lakukan pembaruan platform.		
GzipCompression	Menentukan apakah kompresi gzip diaktifkan. Jika ProxyServer diatur kenone, maka kompresi gzip dinonaktifkan.	`false`	`true` `false`

ProxyServer	Menentukan server web yang harus digunakan untuk koneksi proksi ke Node.js. Jika ProxyServer disetel kenone, maka pemetaan file statis tidak berlaku dan kompresi gzip dinonaktifkan.	`nginx`	`apache`
			`nginx`
			`none`

Namespace: **`aws:elasticbeanstalk:container:nodejs:staticfiles`**

Anda dapat menggunakan namespace berikut untuk mengonfigurasi server proksi agar melayani file statis. Ketika server proksi menerima permintaan untuk file di bawah jalur yang ditentukan, server menyajikan file langsung daripada merutekan permintaan ke aplikasi Anda. Hal ini mengurangi jumlah permintaan yang harus diproses oleh aplikasi Anda harus.

Memetakan jalur yang dilayani oleh server proksi ke folder dalam kode sumber Anda yang berisi aset statis. Setiap pilihan yang Anda tetapkan dalam namespace ini memetakan jalan yang berbeda.

> **Note**
>
> Pengaturan file statis tidak berlaku jika `aws:elasticbeanstalk:container:nodejs::ProxyFiles` ditetapkan ke `none`.

Nama	Nilai
Jalur tempat server proksi akan melayani file.	Nama folder yang berisi file.

Nama	Nilai
Contoh: `/images` untuk melayani file di *`subdomain`* `.eleasticbeanstalk.com/images` .	Contoh: `staticimages` untuk melayani file dari folder yang dinamai `staticimages` di tingkat atas paket sumber Anda.

Opsi platform PHP

Namespace: **`aws:elasticbeanstalk:container:php:phpini`**

Nama	Deskripsi	Default	Nilai yang valid
document_root	Tentukan direktori anak proyek Anda yang diperlakukan sebagai root web yang dapat diakses publik.	`/`	String kosong diperlakukan sebagai `/`, atau menentukan string yang dimulai dengan `/`
memory_limit	Jumlah memori yang dialokasikan ke lingkungan PHP.	`256M`	tidak ada

zlib.output_compression	Menentukan apakah PHP harus menggunakan kompresi untuk output atau tidak.	`Off`	`On`
			`Off`
			`true`
			`false`

allow_url_fopen	Menentukan apakah fungsi file PHP diizinkan untuk mengambil data dari lokasi terpencil, seperti situs web atau server FTP.	`On`	`On`
			`Off`
			`true`
			`false`

display_errors	Menentukan apakah pesan kesalahan harus menjadi bagian dari output.	`Off`	`On` `Off`

Nama	Deskripsi	Default	Nilai yang valid
max_execution_time	Menetapkan waktu maksimum, dalam detik, skrip diizinkan berjalan sebelum diakhiri oleh lingkungan.	`60`	`0` ke `9223372036854775807` (PHP_INT_MAX)
composer_options	Menetapkan opsi kustom untuk digunakan saat menginstal dependensi menggunakan Composer melalui perintah. composer.phar install Untuk informasi selengkapnya, lihat menginstal di situs getcomposer.org.	tidak berlaku	tidak ada

Namespace: **`aws:elasticbeanstalk:environment:proxy`**

Nama	Deskripsi	Default	Nilai yang valid
ProxyServer	Tetapkan proksi yang digunakan pada instans lingkungan Anda.	`nginx`	`apache` `nginx`

> **Note**
>
> Untuk informasi selengkapnya tentang platform PHP, lihat Menggunakan platform PHP Elastic Beanstalk.

Opsi platform Python

Namespace: **`aws:elasticbeanstalk:application:environment`**

Nama	Deskripsi	Default	Nilai yang valid
DJANGO_SETTINGS_MODULE	Menentukan file pengaturan yang akan digunakan.	tidak ada	tidak ada

Lihat Variabel lingkungan dan pengaturan perangkat lunak lainnya untuk informasi selengkapnya.

Namespace: **`aws:elasticbeanstalk:container:python`**

Nama	Deskripsi	Default	Nilai yang valid
WSGIPath	File yang berisi aplikasi WSGI. File ini harus memiliki `application` yang dapat dipanggil.	Pada versi platform Amazon Linux 2 Python: `application` Pada versi platform Amazon Linux AMI Python: `application.py`	tidak ada
NumProcesses	Jumlah proses daemon yang harus dimulai untuk grup proses ketika menjalankan aplikasi WSGI.	1	tidak ada
NumThreads	Jumlah utas yang akan dibuat untuk menangani permintaan pada setiap proses daemon dalam grup proses ketika menjalankan aplikasi WSGI.	15	tidak ada

Namespace: **`aws:elasticbeanstalk:environment:proxy`**

Nama	Deskripsi	Default	Nilai yang valid

ProxyServer	Tetapkan proksi yang digunakan pada instans lingkungan Anda.	`nginx`	`apache`
			
			`nginx`

Opsi platform Amazon Linux AMI (pra-Amazon Linux 2)

Namespace: **`aws:elasticbeanstalk:container:python:staticfiles`**

Anda dapat menggunakan namespace berikut untuk mengonfigurasi server proksi agar melayani file statis. Ketika server proksi menerima permintaan untuk file di bawah jalur yang ditentukan, server menyajikan file langsung daripada merutekan permintaan ke aplikasi Anda. Hal ini mengurangi jumlah permintaan yang harus diproses oleh aplikasi Anda harus.

Memetakan jalur yang dilayani oleh server proksi ke folder dalam kode sumber Anda yang berisi aset statis. Setiap pilihan yang Anda tetapkan dalam namespace ini memetakan jalan yang berbeda.

Secara default, server proksi di lingkungan Python melayani file apa pun dalam folder yang dinamai `static` di jalur `/static`.

Namespace: **`aws:elasticbeanstalk:container:python:staticfiles`**

Nama	Nilai
Jalur tempat server proksi akan melayani file. Contoh: `/images` untuk melayani file di *`subdomain`* `.eleasticbeanstalk.com/images` .	Nama folder yang berisi file. Contoh: `staticimages` untuk melayani file dari folder yang dinamai `staticimages` di tingkat atas paket sumber Anda.

Opsi platform Ruby

Namespace: **`aws:elasticbeanstalk:application:environment`**

Nama	Deskripsi	Default	Nilai yang valid

RAILS_SKIP_MIGRATIONS	Menentukan apakah akan dijalankan `` `rake db:migrate` `` atas nama aplikasi pengguna; atau apakah opsi harus dilewati. Ini hanya berlaku untuk aplikasi Rails 3.	`false`	`true`
			
			`false`

Nama	Deskripsi	Default	Nilai yang valid
RAILS_SKIP_ASSET_COMPILATION	Menentukan apakah kontainer harus menjalankan `rake assets:precompile` atas nama aplikasi pengguna; atau apakah opsi harus dilewati. Ini juga hanya berlaku untuk aplikasi Rails 3.	false	true false
BUNDLE_WITHOUT	Daftar grup yang dipisahkan (:) titik dua akan diabaikan ketika menginstal dependensi dari Gemfile.	test:development	tidak ada
RACK_ENV	Menentukan tahap lingkungan apa yang dapat menjalankan aplikasi. Contoh lingkungan umum meliputi pengembangan, produksi, pengujian.	production	tidak ada

Lihat Variabel lingkungan dan pengaturan perangkat lunak lainnya untuk informasi selengkapnya.

Opsi khusus

Gunakan namespace `aws:elasticbeanstalk:customoption` untuk menentukan pilihan dan nilai yang dapat dibaca di blok `Resources` dalam file konfigurasi lainnya. Gunakan opsi khusus untuk mengumpulkan pengaturan yang ditentukan pengguna dalam file konfigurasi tunggal.

Misalnya, Anda mungkin memiliki file konfigurasi kompleks yang menentukan sumber daya yang dapat dikonfigurasi oleh pengguna yang meluncurkan lingkungan. Jika Anda menggunakan `Fn::GetOptionSetting` untuk mengambil nilai opsi khusus, Anda dapat menempatkan definisi opsi tersebut di file konfigurasi yang berbeda, di mana itu akan lebih mudah ditemukan dan dimodifikasi oleh pengguna.

Juga, karena mereka adalah opsi konfigurasi, opsi khusus dapat diatur pada tingkat API untuk mengganti nilai yang ditetapkan dalam file konfigurasi. Lihat Prioritas untuk informasi selengkapnya.

Opsi khusus ditetapkan seperti opsi lain:

```
option_settings:
  aws:elasticbeanstalk:customoption:
    option name: option value
```

Misalnya, file konfigurasi berikut membuat opsi yang dinamai `ELBAlarmEmail` dan menetapkan nilai untuk `someone@example.com`:

```
option_settings:
  aws:elasticbeanstalk:customoption:
    ELBAlarmEmail: someone@example.com
```

Di tempat lain, file konfigurasi menentukan topik SNS yang membaca opsi dengan `Fn::GetOptionSetting` untuk mengisi nilai atribut `Endpoint`:

```
Resources:
  MySNSTopic:
    Type: AWS::SNS::Topic
    Properties:
      Subscription:
        - Endpoint:
            Fn::GetOptionSetting:
              OptionName: ELBAlarmEmail
              DefaultValue: nobody@example.com
          Protocol: email
```

Anda dapat menemukan lebih banyak cuplikan contoh menggunakan `Fn::GetOptionSetting` di Menambahkan dan menyesuaikan sumber daya lingkungan Elastic Beanstalk.

Penyesuaian lingkungan lanjutan dengan file konfigurasi (`.ebextensions`)

Anda dapat menambahkan file AWS Elastic Beanstalk konfigurasi (`.ebextensions`) ke kode sumber aplikasi web Anda untuk mengonfigurasi lingkungan Anda dan menyesuaikan AWS sumber daya yang dikandungnya. File konfigurasi adalah dokumen berformat YAMB atau JSON dengan ekstensi `.config` file yang Anda tempatkan di folder bernama `.ebextensions` dan terapkan di bundel sumber aplikasi Anda.

Example .ebextensions/ .config network-load-balancer

Contoh ini membuat perubahan konfigurasi sederhana. Contoh ini memodifikasi opsi konfigurasi untuk mengatur jenis penyeimbang beban lingkungan Anda ke Network Load Balancer.

```
option_settings:
  aws:elasticbeanstalk:environment:
    LoadBalancerType: network
```

Kami merekomendasikan menggunakan YAML untuk file konfigurasi Anda, karena YAML lebih mudah dibaca daripada JSON. YAML mendukung komentar, perintah multi-baris, beberapa alternatif untuk menggunakan tanda kutip, dan banyak lagi. Namun, Anda dapat membuat perubahan konfigurasi dalam file konfigurasi Elastic Beanstalk secara identik menggunakan YAML atau JSON.

> **Kiat**
>
> Ketika Anda mengembangkan atau menguji file konfigurasi baru, luncurkan lingkungan bersih yang menjalankan aplikasi default dan terapkan ke sana. File konfigurasi yang diformat dengan buruk akan menyebabkan peluncuran lingkungan baru gagal dan tidak dapat dipulihkan.

Bagian file konfigurasi `option_settings` menentukan nilai untuk opsi konfigurasi. Opsi konfigurasi memungkinkan Anda mengonfigurasi lingkungan Elastic Beanstalk, sumber daya di dalamnya AWS , dan perangkat lunak yang menjalankan aplikasi Anda. File konfigurasi hanya salah satu dari beberapa cara untuk mengatur opsi konfigurasi.

`Resources`Bagian ini memungkinkan Anda menyesuaikan sumber daya lebih lanjut di lingkungan aplikasi Anda, dan menentukan AWS sumber daya tambahan di luar fungsionalitas yang disediakan oleh opsi konfigurasi. Anda dapat menambahkan dan mengonfigurasi sumber daya apa pun yang didukung oleh AWS CloudFormation, yang digunakan Elastic Beanstalk untuk membuat lingkungan.

Bagian lain dari file konfigurasi (`packages`,, `sourcesfiles`,`users`,`groups`, `commandscontainer_commands`, dan`services`) memungkinkan Anda mengonfigurasi EC2 instance yang diluncurkan di lingkungan Anda. Setiap kali server diluncurkan di lingkungan Anda, Elastic Beanstalk menjalankan operasi yang ditentukan di bagian ini untuk mempersiapkan sistem operasi dan sistem penyimpanan untuk aplikasi Anda.

Untuk contoh yang umum digunakan .ebextensions, lihat Repositori File Konfigurasi Elastic Beanstalk.

Persyaratan

- Lokasi - Elastic Beanstalk akan `.ebextensions` memproses semua folder yang ada dalam penyebaran Anda. Namun, kami menyarankan Anda menempatkan semua file konfigurasi Anda dalam satu folder, bernama `.ebextensions`, di root bundel sumber Anda. Folder yang dimulai dengan titik dapat disembunyikan oleh peramban file, jadi pastikan folder ditambahkan ketika Anda membuat paket sumber Anda. Untuk informasi selengkapnya, lihat Buat bundel sumber aplikasi Elastic Beanstalk.
- Penamaan — File konfigurasi harus memiliki ekstensi file `.config`.
- Pemformatan — File konfigurasi harus sesuai dengan spesifikasi YAML atau JSON.

  Bila menggunakan YAML, selalu gunakan spasi untuk kunci indentasi pada tingkat bersarang yang berbeda. Untuk informasi selengkapnya tentang YAML, lihat YAML A'int Markup Language (YAML™) Versi 1.1.
- Keunikan — Gunakan setiap kunci hanya sekali dalam setiap file konfigurasi.

> **Peringatan**
>
> Jika Anda menggunakan kunci (misalnya, `option_settings`) dua kali dalam file konfigurasi yang sama, salah satu bagian akan dihapus. Gabungkan bagian duplikat ke dalam satu bagian, atau letakkan di file konfigurasi terpisah.

Proses untuk men-deploy sedikit berbeda tergantung pada klien yang Anda gunakan untuk mengelola lingkungan Anda. Lihat bagian berikut untuk detailnya:

- Konsol Elastic Beanstalk
- CLI EB
- AWS CLI

Topik

- Pengaturan opsi
- Menyesuaikan perangkat lunak pada server Linux
- Menyesuaikan perangkat lunak pada server Windows
- Menambahkan dan menyesuaikan sumber daya lingkungan Elastic Beanstalk

Pengaturan opsi

Anda dapat menggunakan kunci `option_settings` untuk memodifikasi konfigurasi Elastic Beanstalk dan menentukan variabel yang dapat diambil dari aplikasi Anda menggunakan variabel lingkungan. Beberapa namespace mengizinkan Anda untuk memperpanjang jumlah parameter, dan menentukan nama parameter. Untuk daftar namespace dan opsi konfigurasi, lihat Opsi konfigurasi.

Pengaturan opsi juga dapat diterapkan langsung ke lingkungan selama pembuatan lingkungan atau pembaruan lingkungan. Pengaturan yang diterapkan langsung ke lingkungan menimpa pengaturan untuk pilihan yang sama dalam file konfigurasi. Jika Anda menghapus pengaturan dari konfigurasi lingkungan, pengaturan dalam file konfigurasi akan terpengaruh. Lihat Precedence untuk rincian selengkapnya.

Sintaks

Sintaks standar untuk pengaturan opsi adalah susunan objek, masing-masing memiliki kunci `namespace`, `option_name` dan `value`.

```
option_settings:
  - namespace:  namespace
    option_name:  option name
    value:  option value
  - namespace:  namespace
    option_name:  option name
    value:  option value
```

Kunci `namespace` tersebut opsional. Jika Anda tidak menentukan namespace, nama default yang digunakan adalah `aws:elasticbeanstalk:application:environment`:

```
option_settings:
  - option_name:  option name
    value:  option value
  - option_name:  option name
    value:  option value
```

Elastic Beanstalk juga mendukung sintaks singkat untuk pengaturan opsi yang memungkinkan Anda menentukan opsi sebagai pasangan kunci-nilai di bawah namespace:

```
option_settings:
  namespace:
```

```
    option name: option value
    option name: option value
```

Contoh

Contoh berikut menetapkan opsi khusus platform Tomcat di `aws:elasticbeanstalk:container:tomcat:jvmoptions` namespace dan properti lingkungan yang dinamai `MYPARAMETER`.

Dalam format YAML standar:

Example .ebextensions/options.config

```
option_settings:
  - namespace:  aws:elasticbeanstalk:container:tomcat:jvmoptions
    option_name:  Xmx
    value:  256m
  - option_name: MYPARAMETER
    value: parametervalue
```

Dalam format singkat:

Example .ebextensions/options.config

```
option_settings:
  aws:elasticbeanstalk:container:tomcat:jvmoptions:
    Xmx: 256m
  aws:elasticbeanstalk:application:environment:
    MYPARAMETER: parametervalue
```

Di JSON:

Example .ebextensions/options.config

```
{
  "option_settings": [
    {
      "namespace": "aws:elasticbeanstalk:container:tomcat:jvmoptions",
      "option_name": "Xmx",
      "value": "256m"
    },
    {
```

```
      "option_name": "MYPARAMETER",
      "value": "parametervalue"
    }
  ]
}
```

Menyesuaikan perangkat lunak pada server Linux

Bagian ini menjelaskan jenis informasi yang dapat Anda sertakan dalam file konfigurasi untuk menyesuaikan perangkat lunak pada EC2 instance Anda yang menjalankan Linux. Untuk informasi umum tentang menyesuaikan dan mengonfigurasi lingkungan Elastic Beanstalk Anda, lihat Mengonfigurasi lingkungan Elastic Beanstalk. Untuk informasi tentang menyesuaikan perangkat lunak pada EC2 instans yang menjalankan Windows, lihat. Menyesuaikan perangkat lunak pada server Windows

Anda mungkin ingin menyesuaikan dan mengonfigurasi perangkat lunak yang diandalkan oleh aplikasi Anda. Anda dapat menambahkan perintah yang akan dieksekusi selama penyediaan instans; menentukan pengguna dan grup Linux; dan mengunduh atau langsung membuat file pada instans lingkungan Anda. File ini mungkin berupa dependensi yang diperlukan oleh aplikasi—misalnya, paket tambahan dari repositori yum—atau mungkin file konfigurasi seperti pengganti file konfigurasi proksi untuk menimpa pengaturan tertentu yang ditetapkan oleh Elastic Beanstalk.

> **Catatan**
>
> - Pada platform Amazon Linux 2, alih-alih menyediakan file dan perintah dalam file konfigurasi.ebextensions, kami sangat menyarankan agar Anda menggunakan Buildfile. Procfile, dan hook platform bila memungkinkan untuk mengonfigurasi dan menjalankan kode khusus pada instans lingkungan Anda selama penyediaan instans. Untuk detail tentang mekanisme ini, lihat the section called “Memperluas platform Linux”.
> - YAML bergantung pada indentasi yang konsisten. Cocokkan tingkat indentasi saat mengganti konten dalam file konfigurasi contoh dan pastikan bahwa editor teks Anda menggunakan spasi, bukan karakter tab, untuk indentasi.

File konfigurasi mendukung kunci berikut yang mempengaruhi server Linux tampat aplikasi Anda berjalan.

Kunci

- Paket
- Grup
- Pengguna
- Sumber
- Berkas
- Commands
- Layanan
- Perintah kontainer
- Contoh: Menggunakan CloudWatch metrik Amazon khusus

Kunci diproses dalam urutan yang tercantum di sini.

Perhatikan peristiwa lingkungan Anda saat mengembangkan dan menguji file konfigurasi. Elastic Beanstalk mengabaikan file konfigurasi yang berisi kesalahan validasi, seperti kunci yang tidak valid, dan tidak memproses kunci lain dalam file yang sama. Ketika ini terjadi, Elastic Beanstalk menambahkan peristiwa peringatan log peristiwa.

Paket

Anda dapat menggunakan kunci `packages` untuk mengunduh dan memasang aplikasi dan komponen yang telah dikemas sebelumnya.

Sintaks

```
packages:
  name of package manager:
    package name: version
    ...
  name of package manager:
    package name: version
    ...
  ...
```

Anda dapat menentukan beberapa paket di bawah kunci setiap manajer paket.

Format paket yang didukung

Elastic Beanstalk saat ini mendukung manajer paket berikut: yum, rubygems, python, dan rpm. Paket diproses dengan urutan sebagai berikut: rpm, yum, dan kemudian rubygems dan python. Tidak ada

pengurutan antara rubygems dan python. Dalam setiap manajer paket, pesanan instalasi paket tidak dijamin. Gunakan manajer paket yang didukung oleh sistem operasi Anda.

> **Note**
>
> Elastic Beanstalk mendukung dua manajer paket yang mendasari untuk Python, pip dan easy_install. Namun, dalam sintaks file konfigurasi, Anda harus menentukan nama manajer paket sebagai `python`. Bila Anda menggunakan file konfigurasi untuk menentukan manajer paket Python, Elastic Beanstalk menggunakan Python 2.7. Jika aplikasi Anda bergantung pada versi Python yang berbeda, Anda dapat menentukan paket yang akan dipasang di file `requirements.txt`. Untuk informasi selengkapnya, lihat Menentukan dependensi menggunakan file persyaratan pada Elastic Beanstalk.

Menentukan versi

Dalam setiap manajer paket, setiap paket ditetapkan sebagai nama paket dan daftar versi. Versi ini bisa berupa string, daftar versi, atau string atau daftar kosong. String atau daftar kosong menunjukkan bahwa Anda menginginkan versi terbaru. Untuk manajer rpm, versi ditentukan sebagai jalur menuju file pada disk atau URL. Jalur relatif tidak didukung.

Jika Anda menentukan versi paket, Elastic Beanstalk mencoba untuk memasang versi tersebut meskipun versi paket yang lebih baru sudah dipasang pada instans. Jika versi yang lebih baru sudah dipasang, deployment gagal. Beberapa manajer paket mendukung beberapa versi, tetapi yang lain mungkin tidak. Periksa dokumentasi manajer paket Anda untuk informasi lebih lanjut. Jika Anda tidak menentukan versi dan versi paket sudah terinstal, Elastic Beanstalk tidak menginstal versi baru—ini mengasumsikan bahwa Anda ingin menyimpan dan menggunakan versi yang ada.

Contoh snippet

Potongan berikut menentukan URL versi untuk rpm, meminta versi terbaru dari yum, dan versi 0.10.2 chef dari rubygems.

```
packages:
  yum:
    libmemcached: []
    ruby-devel: []
    gcc: []
  rpm:
    epel: http://download.fedoraproject.org/pub/epel/5/i386/epel-release-5-4.noarch.rpm
```

```
  rubygems:
    chef: '0.10.2'
```

Grup

Anda dapat menggunakan `groups` kunci untuk membuat grup Linux/Unix dan menetapkan grup. IDs Untuk membuat grup, tambahkan pasangan nilai kunci baru yang memetakan nama grup baru untuk ID grup opsional. Kunci grup dapat berisi satu nama grup atau beberapa. Tabel berikut menjelaskan kunci yang tersedia.

Sintaksis

```
groups:
  name of group: {}
  name of group:
    gid: "group id"
```

Opsi

`gid`

Nomor ID grup.

Jika ID grup ditentukan, dan grup sudah ada berdasarkan nama, pembuatan grup akan gagal. Jika grup lain memiliki ID grup tertentu, sistem operasi dapat menolak pembuatan grup.

Contoh snippet

Potongan berikut menentukan grup bernama groupOne tanpa menetapkan ID grup dan grup bernama groupTwo yang menentukan nilai ID grup 45.

```
groups:
  groupOne: {}
  groupTwo:
    gid: "45"
```

Pengguna

Anda dapat menggunakan `users` kunci untuk membuat pengguna Linux/Unix pada instance. EC2

Sintaksis

```
users:
  name of user:
    groups:
      - name of group
    uid: "id of the user"
    homeDir: "user's home directory"
```

Opsi

`uid`

ID pengguna. Proses pembuatan gagal jika nama pengguna ada dengan ID pengguna yang berbeda. Jika ID pengguna sudah ditetapkan untuk pengguna yang sudah ada, sistem operasi dapat menolak permintaan pembuatan.

`groups`

Daftar nama grup. Pengguna ditambahkan ke setiap grup dalam daftar.

`homeDir`

Direktori beranda pengguna.

Pengguna dibuat sebagai pengguna sistem noninteraktif dengan shell `/sbin/nologin`. Ini adalah disengaja dan tidak dapat dimodifikasi.

Contoh potongan

```
users:
  myuser:
    groups:
      - group1
      - group2
    uid: "50"
    homeDir: "/tmp"
```

Sumber

Anda dapat menggunakan `sources` kunci untuk mengunduh file arsip dari URL publik dan membongkarnya di direktori target pada EC2 instance.

Sintaks

```
sources:
  target directory: location of archive file
```

Format yang didukung

Format yang didukung adalah tar, tar+gzip, tar+bz2, dan zip. Anda dapat mereferensi lokasi eksternal seperti Amazon Simple Storage Service (Amazon S3) (misalnya, `https://amzn-s3-demo-bucket.s3.amazonaws.com/myobject`) selama URL dapat diakses publik.

Contoh snippet

Contoh berikut mengunduh file .zip publik dari bucket Amazon S3 dan membukanya ke dalam `/etc/myapp`:

```
sources:
  /etc/myapp: https://amzn-s3-demo-bucket.s3.amazonaws.com/myobject
```

> **Note**
>
> Beberapa ekstraksi tidak boleh menggunakan kembali jalur target yang sama. Mengekstrak sumber lain ke jalur target yang sama akan mengganti alih-alih menambah konten.

Berkas

Anda dapat menggunakan `files` kunci untuk membuat file pada EC2 instance. Konten dapat berupa inline dalam file konfigurasi, atau konten dapat ditarik dari URL. File ditulis ke disk dalam urutan leksikografis.

Anda dapat menggunakan kunci `files` untuk mengunduh file pribadi dari Amazon S3 dengan menyediakan profil instans untuk otorisasi.

Jika jalur file yang Anda tentukan sudah ada di instans, file yang ada dipertahankan dengan ekstensi `.bak` yang ditambahkan ke namanya.

Sintaksis

```
files:
```

```
  "target file location on disk":
     mode: "six-digit octal value"
     owner: name of owning user for file
     group: name of owning group for file
     source: URL
     authentication: authentication name:

  "target file location on disk":
     mode: "six-digit octal value"
     owner: name of owning user for file
     group: name of owning group for file
     content: |
      # this is my
      # file content
     encoding: encoding format
     authentication: authentication name:
```

Opsi

`content`

Konten string untuk ditambahkan ke file. Tentukan salah satu `content` atau `source`, tidak keduanya.

`source`

URL file yang akan diunduh. Tentukan salah satu `content` atau `source`, tidak keduanya.

`encoding`

Format pengodean dari string yang ditentukan dengan opsi `content`.

Nilai yang valid: `plain` | `base64`

`group`

Grup Linux yang memiliki file.

`owner`

Pengguna Linux yang memiliki file.

`mode`

Nilai oktal enam digit yang mewakili mode untuk file ini. Tidak mendukung sistem Windows. Gunakan tiga digit pertama untuk symlink dan tiga digit terakhir untuk pengaturan izin. Untuk

membuat symlink, tentukan `120xxx`, tempat `xxx` mendefinisikan izin file target. Untuk menentukan izin file, gunakan tiga digit terakhir, seperti `000644`.

`authentication`

Nama AWS CloudFormation metode autentikasi yang akan digunakan. Anda dapat menambahkan metode autentikasi ke metadata grup Auto Scaling dengan kunci Sumber Daya. Lihat di bawah untuk contoh.

Contoh snippet

```
files:
  "/home/ec2-user/myfile" :
    mode: "000755"
    owner: root
    group: root
    source: http://foo.bar/myfile

  "/home/ec2-user/myfile2" :
    mode: "000755"
    owner: root
    group: root
    content: |
      this is my
      file content
```

Contoh penggunaan symlink. Hal ini membuat tautan `/tmp/myfile2.txt` yang mengarah ke file yang ada `/tmp/myfile1.txt`.

```
files:
  "/tmp/myfile2.txt" :
    mode: "120400"
    content: "/tmp/myfile1.txt"
```

Contoh berikut menggunakan kunci Sumber daya untuk menambahkan metode autentikasi bernama S3Auth dan menggunakannya untuk mengunduh file pribadi dari bucket Amazon S3:

```
Resources:
  AWSEBAutoScalingGroup:
    Metadata:
      AWS::CloudFormation::Authentication:
```

```
        S3Auth:
          type: "s3"
          buckets: ["amzn-s3-demo-bucket2"]
          roleName:
            "Fn::GetOptionSetting":
              Namespace: "aws:autoscaling:launchconfiguration"
              OptionName: "IamInstanceProfile"
              DefaultValue: "aws-elasticbeanstalk-ec2-role"

files:
  "/tmp/data.json" :
    mode: "000755"
    owner: root
    group: root
    authentication: "S3Auth"
    source: https://elasticbeanstalk-us-west-2-123456789012.s3-us-west-2.amazonaws.com/
data.json
```

Commands

Anda dapat menggunakan `commands` kunci untuk menjalankan perintah pada EC2 instance. Perintah berjalan sebelum server aplikasi dan web diatur dan file versi aplikasi diekstrak.

Perintah yang ditentukan dijalankan sebagai pengguna root, dan diproses dalam urutan abjad berdasarkan nama. Secara default, perintah berjalan di direktori root. Untuk menjalankan perintah dari direktori lain, gunakan opsi `cwd` tersebut.

Untuk memecahkan masalah dengan perintah Anda, Anda dapat menemukan outputnya di log instans.

Sintaksis

```
commands:
  command name:
    command: command to run
    cwd: working directory
    env:
      variable name: variable value
    test: conditions for command
    ignoreErrors: true
```

Opsi

`command`

Array (koleksi urutan blok dalam sintaks YAML) atau string akan menentukan perintah untuk dijalankan. Beberapa catatan penting:

- Jika Anda menggunakan string, Anda tidak perlu menyertakan seluruh string dalam tanda kutip. Jika Anda menggunakan tanda kutip, hindari kejadian literal dari jenis kutipan yang sama.
- Jika Anda menggunakan array, Anda tidak perlu menghindari karakter spasi atau melampirkan parameter perintah pada kutipan. Setiap elemen array adalah argumen perintah tunggal. Jangan gunakan array untuk menentukan beberapa perintah.

Contoh berikut semuanya setara:

```
commands:
  command1:
    command: git commit -m "This is a comment."
  command2:
    command: "git commit -m \"This is a comment.\""
  command3:
    command: 'git commit -m "This is a comment."'
  command4:
    command:
      - git
      - commit
      - -m
      - This is a comment.
```

Untuk menentukan beberapa perintah, gunakan skalar blok literal, seperti yang ditunjukkan pada contoh berikut.

```
commands:
  command block:
    command: |
      git commit -m "This is a comment."
      git push
```

`env`

(Opsional) Atur variabel lingkungan untuk perintah. Properti ini menimpa, bukan menambahkan, lingkungan yang ada.

`cwd`

(Opsional) Direktori kerja. Jika tidak ditentukan, perintah dijalankan dari direktori root (/).

`test`

(Opsional) Perintah yang harus mengembalikan nilai `true` (kode keluar 0) agar Elastic Beanstalk memproses perintah, seperti skrip shell, yang terdapat dalam kunci `command`.

`ignoreErrors`

(Opsional) Nilai boolean yang menentukan apakah perintah lain harus dijalankan jika perintah yang terdapat dalam kunci `command` gagal (mengembalikan nilai bukan nol). Tetapkan nilai ini ke `true` jika Anda ingin terus menjalankan perintah meskipun perintah gagal. Atur nilai ke `false` jika Anda ingin berhenti menjalankan perintah jika perintah gagal. Nilai default-nya adalah `false`.

Contoh snippet

Contoh potongan berikut menjalankan skrip Python.

```
commands:
  python_install:
    command: myscript.py
    cwd: /home/ec2-user
    env:
      myvarname: myvarvalue
    test: "[ -x /usr/bin/python ]"
```

Layanan

Anda dapat menggunakan kunci `services` untuk menentukan layanan yang harus dimulai atau berhenti ketika instans diluncurkan. Kunci `services` tersebut juga mengizinkan Anda untuk menentukan dependensi pada sumber, paket, dan file sehingga jika mulai ulang diperlukan karena file yang diinstal, Elastic Beanstalk mengurus layanan mulai ulang.

Sintaksis

```
services:
  sysvinit:
    name of service:
      enabled: "true"
      ensureRunning: "true"
```

```
    files:
      - "file name"
    sources:
      - "directory"
    packages:
      name of package manager:
        "package name[: version]"
    commands:
      - "name of command"
```

Opsi

`ensureRunning`

Atur ke `true` untuk memastikan layanan berjalan setelah Elastic Beanstalk selesai.

Atur ke `false` untuk memastikan layanan tidak berjalan setelah Elastic Beanstalk selesai.

Abaikan kunci ini untuk tidak membuat perubahan pada status layanan.

`enabled`

Atur ke `true` untuk memastikan layanan dimulai secara otomatis setelah boot.

Atur ke `false` untuk memastikan layanan tidak dimulai secara otomatis setelah boot.

Abaikan kunci ini untuk tidak membuat perubahan pada properti ini.

`files`

Daftar file. Jika Elastic Beanstalk mengubah file secara langsung melalui blok file, layanan dimulai ulang.

`sources`

Daftar direktori. Jika Elastic Beanstalk memperluas arsip ke salah satu direktori ini, layanan akan dimulai ulang.

`packages`

Peta manajer paket ke daftar nama paket. Jika Elastic Beanstalk menginstal atau memperbarui salah satu paket ini, layanan akan dimulai ulang.

`commands`

Daftar nama perintah. Jika Elastic Beanstalk menjalankan perintah yang ditentukan, layanan akan dimulai ulang.

Contoh snippet

Berikut ini adalah contoh snippet:

```
services:
  sysvinit:
    myservice:
      enabled: true
      ensureRunning: true
```

Perintah kontainer

Anda dapat menggunakan kunci `container_commands` untuk menjalankan perintah yang mempengaruhi kode sumber aplikasi Anda. Perintah kontainer berjalan setelah aplikasi dan server web diatur dan arsip versi aplikasi telah diekstraksi, tetapi sebelum versi aplikasi di-deploy. Perintah non-kontainer dan operasi penyesuaian lainnya dilakukan sebelum kode sumber aplikasi diekstraksi.

Perintah yang ditentukan dijalankan sebagai pengguna root, dan diproses dalam urutan abjad berdasarkan nama. Perintah kontainer dijalankan dari direktori pementasan, di mana kode sumber Anda diekstrak sebelum di-deploy ke server aplikasi. Setiap perubahan yang Anda buat untuk kode sumber Anda di direktori pementasan dengan perintah kontainer akan disertakan ketika sumber di-deploy ke lokasi akhir.

> **Note**
>
> Output dari perintah container Anda dicatat di log `cfn-init-cmd.log` instance. Untuk informasi selengkapnya tentang mengambil dan melihat log instans, lihat Melihat log dari EC2 instans Amazon.

Anda dapat menggunakan `leader_only` hanya untuk menjalankan perintah pada satu instans, atau mengonfigurasi `test` hanya untuk menjalankan perintah ketika perintah pengujian dievaluasi `true`. Perintah kontainer khusus pemimpin hanya dijalankan selama pembuatan dan penerapan lingkungan, sedangkan perintah dan operasi penyesuaian server lain dilakukan setiap kali sebuah instans ditetapkan atau diperbarui. Perintah kontainer khusus pemimpin tidak dijalankan untuk memulai perubahan konfigurasi, seperti perubahan pada Id AMI atau jenis instans.

Sintaksis

```
container_commands:
```

```
  name of container_command:
    command: "command to run"
    leader_only: true
  name of container_command:
    command: "command to run"
```

Opsi

`command`

Sebuah string atau array string yang akan dijalankan.

`env`

(Opsional) Atur variabel lingkungan sebelum menjalankan perintah, menimpa nilai yang ada.

`cwd`

(Opsional) Direktori kerja. Secara default, ini adalah direktori persiapan aplikasi unzip.

`leader_only`

(Opsional) Hanya jalankan perintah pada satu instans yang dipilih oleh Elastic Beanstalk. Perintah kontainer khusus pemimpin dijalankan sebelum perintah kontainer lainnya. Sebuah perintah bisa ditujukan khusus untuk pemimpin atau memiliki `test`, tetapi tidak keduanya (`leader_only` diutamakan).

`test`

(Opsional) Jalankan perintah pengujian yang harus mengembalikan `true` untuk menjalankan perintah kontainer. Sebuah perintah bisa ditujukan khusus untuk pemimpin atau memiliki `test`, tetapi tidak keduanya (`leader_only` diutamakan).

`ignoreErrors`

(Opsional) Jangan gagalkan penerapan jika perintah kontainer mengembalikan nilai selain 0 (sukses). Atur ke `true` untuk mengaktifkan.

Contoh snippet

Berikut adalah contoh snippet.

```
container_commands:
```

```
  collectstatic:
    command: "django-admin.py collectstatic --noinput"
  01syncdb:
    command: "django-admin.py syncdb --noinput"
    leader_only: true
  02migrate:
    command: "django-admin.py migrate"
    leader_only: true
  99customize:
    command: "scripts/customize.sh"
```

Contoh: Menggunakan CloudWatch metrik Amazon khusus

Topik ini memberikan contoh konfigurasi yang mengintegrasikan metrik Elastic Beanstalk CloudWatch dengan agen Amazon untuk platform berbasis Amazon Linux 2 dan yang lebih baru. Contoh konfigurasi menggunakan file dan perintah dalam file `.ebextensions` konfigurasi.

Amazon CloudWatch adalah layanan web yang memungkinkan Anda memantau, mengelola, dan mempublikasikan berbagai metrik, serta mengonfigurasi tindakan alarm berdasarkan data dari metrik. Anda dapat menentukan metrik khusus untuk penggunaan Anda sendiri, dan Elastic Beanstalk akan mendorong metrik tersebut ke Amazon. CloudWatch Setelah Amazon CloudWatch berisi metrik khusus Anda, Anda dapat melihatnya di CloudWatch konsol Amazon.

CloudWatch Agen Amazon

CloudWatch Agen Amazon mengaktifkan pengumpulan CloudWatch metrik dan log dari EC2 instans Amazon dan server lokal di seluruh sistem operasi. Agen mendukung metrik yang dikumpulkan di tingkat sistem. Ini juga mendukung log kustom dan koleksi metrik dari aplikasi atau layanan Anda. Untuk informasi selengkapnya tentang CloudWatch agen Amazon, lihat Mengumpulkan metrik dan log dengan CloudWatch agen di Panduan CloudWatch Pengguna Amazon.

> **Note**
>
> Elastic Beanstalk Enhanced Health Reporting memiliki dukungan asli untuk menerbitkan berbagai instans dan metrik lingkungan. CloudWatch Lihat Menerbitkan metrik CloudWatch kustom Amazon untuk suatu lingkungan untuk rincian selengkapnya.

Topik

- file konfigurasi .Ebextensions

- Izin
- Melihat metrik di konsol CloudWatch

file konfigurasi .Ebextensions

Contoh ini menggunakan file dan perintah dalam file konfigurasi.ebextensions untuk mengonfigurasi dan menjalankan agen Amazon CloudWatch di platform Amazon Linux 2. Agen ini dikemas dengan Amazon Linux 2. Jika Anda menggunakan sistem operasi yang berbeda, langkah-langkah tambahan untuk menginstal agen mungkin diperlukan. Untuk informasi selengkapnya, lihat Menginstal CloudWatch agen di Panduan CloudWatch Pengguna Amazon.

Untuk menggunakan contoh ini, simpan contoh ke file bernama `cloudwatch.config` dalam direktori bernama `.ebextensions` di tingkat atas direktori proyek Anda, kemudian terapkan aplikasi Anda menggunakan konsol Elastic Beanstalk (termasuk direktori .ebextensions di paket sumber) atau EB CLI.

Untuk informasi lebih lanjut tentang file konfigurasi, lihat Penyesuaian lingkungan lanjutan dengan file konfigurasi (`.ebextensions`).

File ini memiliki dua bagian:

- `files`— Bagian ini menambahkan file konfigurasi agen. Ini menunjukkan metrik dan log mana yang harus dikirim agen ke Amazon CloudWatch. Dalam contoh ini, kami hanya mengirimkan metrik mem_used_percent. Untuk daftar lengkap metrik tingkat sistem yang didukung oleh CloudWatch agen Amazon, lihat Metrik yang dikumpulkan oleh CloudWatch agen di CloudWatch Panduan Pengguna Amazon.
- `container_commands`— Bagian ini berisi perintah yang memulai agen, meneruskan file konfigurasi sebagai parameter. Untuk detail lebih lanjut tentang`container_commands`, lihatPerintah kontainer.

.ebextensions/cloudwatch.config

```
files:
  "/opt/aws/amazon-cloudwatch-agent/bin/config.json":
    mode: "000600"
    owner: root
    group: root
    content: |
      {
```

```
        "agent": {
          "metrics_collection_interval": 60,
          "run_as_user": "root"
        },
        "metrics": {
          "namespace": "System/Linux",
          "append_dimensions": {
            "AutoScalingGroupName": "${aws:AutoScalingGroupName}"
          },
          "metrics_collected": {
            "mem": {
              "measurement": [
                "mem_used_percent"
              ]
            }
          }
        }
      }
container_commands:
  start_cloudwatch_agent:
    command: /opt/aws/amazon-cloudwatch-agent/bin/amazon-cloudwatch-agent-ctl -a
 append-config -m ec2 -s -c file:/opt/aws/amazon-cloudwatch-agent/bin/config.json
```

Izin

Instans di lingkungan Anda memerlukan izin IAM yang tepat untuk mempublikasikan CloudWatch metrik Amazon khusus menggunakan agen Amazon. CloudWatch Anda memberikan izin ke instance lingkungan Anda dengan menambahkannya ke profil instans lingkungan. Anda dapat menambahkan izin ke profil instans sebelum atau setelah men-deploy aplikasi Anda.

Untuk memberikan izin untuk mempublikasikan CloudWatch metrik

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Pada panel navigasi, pilih Peran.
3. Pilih peran profil instans lingkungan Anda. Secara default, ketika Anda membuat lingkungan dengan konsol Elastic Beanstalk atau EB CLI, ini adalah `aws-elasticbeanstalk-ec2-role`.
4. Pilih tab Izin.
5. Di bawah Kebijakan Izin, di bagian Izin, pilih Lampirkan kebijakan.
6. Di bawah Lampirkan Izin, pilih kebijakan AWS CloudWatchAgentServerPolicyterkelola. Kemudian klik Lampirkan kebijakan.

Untuk informasi selengkapnya tentang mengelola kebijakan, lihat Bekerja dengan Kebijakan di Panduan Pengguna IAM.

Melihat metrik di konsol CloudWatch

Setelah menerapkan file CloudWatch konfigurasi ke lingkungan Anda, periksa CloudWatchkonsol Amazon untuk melihat metrik Anda. Metrik khusus akan ditempatkan di CWAgentnamespace.

Untuk informasi selengkapnya, lihat Melihat metrik yang tersedia di Panduan CloudWatch Pengguna Amazon.

Menyesuaikan perangkat lunak pada server Windows

Anda mungkin ingin menyesuaikan dan mengonfigurasi perangkat lunak yang diandalkan oleh aplikasi Anda. File ini dapat berupa dependensi yang dibutuhkan oleh aplikasi—misalnya, paket tambahan atau layanan yang perlu dijalankan. Untuk informasi umum tentang menyesuaikan dan mengonfigurasi lingkungan Elastic Beanstalk Anda, lihat Mengonfigurasi lingkungan Elastic Beanstalk.

> **Note**
>
> YAML bergantung pada indentasi yang konsisten. Cocokkan tingkat indentasi saat mengganti konten dalam file konfigurasi contoh dan pastikan bahwa editor teks Anda menggunakan spasi, bukan karakter tab, untuk indentasi.

File konfigurasi mendukung kunci berikut yang mempengaruhi server Windows tempat aplikasi Anda berjalan.

Kunci

- Paket
- Sumber
- Berkas
- Commands
- Layanan
- Perintah kontainer

Kunci diproses dalam urutan yang tercantum di sini.

> **Note**
>
> Versi platform .NET yang lebih lama (tidak-berversi) tidak memproses file konfigurasi dalam urutan yang benar. Pelajari lebih lanjut di Migrasi di versi utama dari platform server Elastic Beanstalk Windows.

Perhatikan peristiwa lingkungan Anda saat mengembangkan dan menguji file konfigurasi. Elastic Beanstalk mengabaikan file konfigurasi yang berisi kesalahan validasi, seperti kunci yang tidak valid, dan tidak memproses kunci lain dalam file yang sama. Ketika ini terjadi, Elastic Beanstalk menambahkan peristiwa peringatan log peristiwa.

Paket

Gunakan kunci `packages` tersebut untuk mengunduh dan menginstal aplikasi dan komponen yang telah dikemas sebelumnya.

Di lingkungan Windows, Elastic Beanstalk mendukung pengunduhan dan penginstalan paket MSI. (Lingkungan Linux mendukung manajer paket tambahan. Untuk detailnya, lihat Paket pada halaman Menyesuaikan Perangkat Lunak pada Server Linux.)

Anda dapat mereferensikan lokasi eksternal, seperti objek Amazon Simple Storage Service (Amazon S3), selama URL dapat diakses publik.

Jika Anda menentukan beberapa paket `msi:`, instalasinya tidak dijamin.

Sintaksis

Tentukan nama pilihan Anda sebagai nama paket, dan URL ke lokasi file MSI sebagai nilai. Anda dapat menentukan beberapa paket di bawah kunci `msi:`.

```
packages:
  msi:
    package name: package url
    ...
```

Contoh

Contoh berikut menentukan URL untuk mengunduh mysql dari `https://dev.mysql.com/`.

```
packages:
  msi:
    mysql: https://dev.mysql.com/get/Downloads/Connector-Net/mysql-connector-
net-8.0.11.msi
```

Contoh berikut menentukan objek Amazon S3 sebagai lokasi file MSI.

```
packages:
  msi:
    mymsi: https://amzn-s3-demo-bucket.s3.amazonaws.com/myobject.msi
```

Sumber

Gunakan `sources` kunci untuk mengunduh file arsip dari URL publik dan membongkarnya di direktori target pada EC2 instance.

Sintaksis

```
sources:
  target directory: location of archive file
```

Format yang didukung

Di lingkungan Windows, Elastic Beanstalk mendukung format .zip. (Lingkungan Linux mendukung format tambahan. Untuk detailnya, lihat Sumber pada halaman Menyesuaikan Perangkat Lunak di Server Linux.)

Anda dapat mereferensikan lokasi eksternal, seperti objek Amazon Simple Storage Service (Amazon S3), selama URL dapat diakses publik.

Contoh

Contoh berikut mengunduh file.zip publik dari Amazon S3 dan membongkarnya. `c:/myproject/myapp`

```
sources:
  "c:/myproject/myapp": https://amzn-s3-demo-bucket.s3.amazonaws.com/myobject.zip
```

Berkas

Gunakan `files` tombol untuk membuat file pada EC2 instance. Konten dapat berupa inline dalam file konfigurasi, atau dari URL. File ditulis ke disk dalam urutan leksikografis. Untuk mengunduh file pribadi dari Amazon S3, berikan profil instans untuk autorisasi.

Sintaksis

```
files:
  "target file location on disk":
    source: URL
    authentication: authentication name:

  "target file location on disk":
    content: |
      this is my content
    encoding: encoding format
```

Opsi

`content`

(Opsional) String.

`source`

(Opsional) URL tempat file dimuat. Opsi ini tidak dapat ditentukan dengan kunci konten.

`encoding`

(Opsional) Format pengodean. Opsi ini hanya digunakan untuk nilai kunci konten yang disediakan. Nilai default-nya adalah `plain`.

Nilai yang valid: `plain` | `base64`

`authentication`

(Opsional) Nama metode autentikasi AWS CloudFormation yang akan digunakan. Anda dapat menambahkan metode autentikasi ke metadata grup Auto Scaling dengan kunci Sumber Daya.

Contoh

Contoh berikut menunjukkan dua cara untuk menyediakan konten file: dari URL, atau inline dalam file konfigurasi.

```
files:
  "c:\\targetdirectory\\targetfile.txt":
    source: http://foo.bar/myfile

  "c:/targetdirectory/targetfile.txt":
    content: |
      # this is my file
      # with content
```

> **Note**
>
> Jika Anda menggunakan backslash (\) di jalur file Anda, Anda harus mendahuluinya dengan backslash lain (karakter escape) seperti yang ditunjukkan pada contoh sebelumnya.

Contoh berikut menggunakan kunci Resources untuk menambahkan metode otentikasi bernama S3Auth dan menggunakannya untuk mengunduh file pribadi dari Amazon S3:

```
files:
  "c:\\targetdirectory\\targetfile.zip":
    source: https://elasticbeanstalk-us-east-2-123456789012.s3.amazonaws.com/prefix/
myfile.zip
    authentication: S3Auth

Resources:
  AWSEBAutoScalingGroup:
    Metadata:
      AWS::CloudFormation::Authentication:
        S3Auth:
          type: "s3"
          s: ["amzn-s3-demo-bucket"]
          roleName:
            "Fn::GetOptionSetting":
              Namespace: "aws:autoscaling:launchconfiguration"
              OptionName: "IamInstanceProfile"
              DefaultValue: "aws-elasticbeanstalk-ec2-role"
```

Commands

Gunakan `commands` tombol untuk menjalankan perintah pada EC2 instance. Perintah diproses dalam urutan abjad berdasarkan nama, dan mereka berjalan sebelum aplikasi dan server web diatur dan file versi aplikasi diekstrak.

Perintah yang ditentukan dijalankan sebagai pengguna Administrator.

Untuk memecahkan masalah dengan perintah Anda, Anda dapat menemukan outputnya di log instans.

Sintaksis

```
commands:
  command name:
    command: command to run
```

Opsi

`command`

Baik array atau string yang menentukan perintah yang akan dijalankan. Jika Anda menggunakan array, Anda tidak perlu menghindari karakter spasi atau melampirkan parameter perintah dalam tanda kutip.

`cwd`

(Opsional) Direktori kerja. Secara default, Elastic Beanstalk mencoba untuk menemukan lokasi direktori proyek Anda. Jika tidak ditemukan, Elastic Beanstalk menggunakan `c:\Windows\System32` sebagai default.

`env`

(Opsional) Atur variabel lingkungan untuk perintah. Properti ini menimpa, bukan menambahkan, lingkungan yang ada.

`ignoreErrors`

(Opsional) Nilai Boolean yang menentukan apakah perintah lain harus dijalankan jika perintah yang terdapat dalam kunci `command` gagal (mengembalikan nilai bukan nol). Tetapkan nilai ini ke `true` jika Anda ingin terus menjalankan perintah meskipun perintah gagal. Atur nilai ke `false` jika Anda ingin berhenti menjalankan perintah jika perintah gagal. Nilai default-nya adalah `false`.

`test`

(Opsional) Perintah yang harus mengembalikan nilai `true` (kode keluar 0) agar Elastic Beanstalk memproses perintah yang terdapat dalam kunci `command`.

`waitAfterCompletion`

(Opsional) Menunggu beberapa detik setelah perintah selesai sebelum menjalankan perintah berikutnya. Nilai default adalah **`60`** detik. Anda juga dapat menentukan**`forever`**.

> **Important**
>
> Reboot sistem selama penerapan tidak didukung. Jika sistem reboot sebagai akibat dari perintah, inisialisasi instance akan gagal, menyebabkan penerapan gagal.
> Sebagai solusinya, Anda dapat menggunakan ini `.ebextensions` `configuration`untuk menjadwalkan reboot setelah penerapan selesai.

Contoh

Contoh berikut menyimpan output dari perintah `set` ke file yang ditentukan. Jika ada perintah berikutnya, Elastic Beanstalk menjalankan perintah itu segera setelah perintah ini selesai. Jika perintah ini membutuhkan reboot, Elastic Beanstalk melakukan reboot instans segera setelah perintah selesai.

```
commands:
  test:
    command: set > c:\\myapp\\set.txt
    waitAfterCompletion: 0
```

Layanan

Gunakan kunci `services` untuk menentukan layanan yang harus dimulai atau dihentikan ketika instans diluncurkan. Kunci `services` tersebut juga memungkinkan Anda untuk menentukan dependensi pada sumber, paket, dan file sehingga jika restart diperlukan akibat dari file yang diinstal, Elastic Beanstalk mengurus restart layanan.

Sintaksis

```
services:
```

```
  windows:
    name of service:
      files:
        - "file name"
      sources:
        - "directory"
      packages:
        name of package manager:
            "package name[: version]"
      commands:
        - "name of command"
```

Opsi

`ensureRunning`

(Opsional) Atur ke `true` untuk memastikan layanan berjalan setelah Elastic Beanstalk selesai.

Atur ke `false` untuk memastikan layanan tidak berjalan setelah Elastic Beanstalk selesai.

Abaikan kunci ini untuk tidak membuat perubahan pada status layanan.

`enabled`

(Opsional) Atur ke `true` untuk memastikan layanan dimulai secara otomatis setelah boot.

Atur ke `false` untuk memastikan layanan tidak dimulai secara otomatis setelah boot.

Abaikan kunci ini untuk tidak membuat perubahan pada properti ini.

`files`

Daftar file. Jika Elastic Beanstalk mengubah file secara langsung melalui blok file, layanan dimulai ulang.

`sources`

Daftar direktori. Jika Elastic Beanstalk memperluas arsip ke salah satu direktori ini, layanan akan dimulai ulang.

`packages`

Peta manajer paket ke daftar nama paket. Jika Elastic Beanstalk menginstal atau memperbarui salah satu paket ini, layanan akan dimulai ulang.

`commands`

Daftar nama perintah. Jika Elastic Beanstalk menjalankan perintah yang ditentukan, layanan akan dimulai ulang.

Contoh

```
services:
  windows:
    myservice:
      enabled: true
      ensureRunning: true
```

Perintah kontainer

Gunakan kunci `container_commands` untuk menjalankan perintah yang mempengaruhi kode sumber aplikasi Anda. Perintah kontainer berjalan setelah aplikasi dan server web diatur dan arsip versi aplikasi telah diekstraksi, tetapi sebelum versi aplikasi di-deploy. Perintah non-kontainer dan operasi penyesuaian lainnya dilakukan sebelum kode sumber aplikasi diekstraksi.

Perintah kontainer dijalankan dari direktori pementasan, di mana kode sumber Anda diekstrak sebelum di-deploy ke server aplikasi. Setiap perubahan yang Anda buat untuk kode sumber Anda di direktori pementasan dengan perintah kontainer akan disertakan ketika sumber di-deploy ke lokasi akhir.

Untuk memecahkan masalah dengan perintah kontainer, Anda dapat menemukan output-nya di log instans.

Gunakan opsi `leader_only` hanya untuk menjalankan perintah pada satu instans, atau mengonfigurasi `test` hanya untuk menjalankan perintah ketika perintah pengujian dievaluasi `true`. Perintah kontainer khusus pemimpin hanya dijalankan selama pembuatan dan penerapan lingkungan, sedangkan perintah dan operasi penyesuaian server lain dilakukan setiap kali sebuah instans ditetapkan atau diperbarui. Perintah kontainer khusus pemimpin tidak dijalankan untuk memulai perubahan konfigurasi, seperti perubahan pada Id AMI atau jenis instans.

Sintaksis

```
container_commands:
  name of container_command:
    command: command to run
```

Opsi

`command`

Sebuah string atau array string yang akan dijalankan.

`env`

(Opsional) Atur variabel lingkungan sebelum menjalankan perintah, menimpa nilai yang ada.

`cwd`

(Opsional) Direktori kerja. Secara default, ini adalah direktori persiapan aplikasi unzip.

`leader_only`

(Opsional) Hanya jalankan perintah pada satu instans yang dipilih oleh Elastic Beanstalk. Perintah kontainer khusus pemimpin dijalankan sebelum perintah kontainer lainnya. Sebuah perintah bisa ditujukan khusus untuk pemimpin atau memiliki `test`, tetapi tidak keduanya (`leader_only` diutamakan).

`test`

(Opsional) Jalankan perintah pengujian yang harus mengembalikan `true` untuk menjalankan perintah kontainer. Sebuah perintah bisa ditujukan khusus untuk pemimpin atau memiliki `test`, tetapi tidak keduanya (`leader_only` diutamakan).

`ignoreErrors`

(Opsional) Jangan gagalkan penerapan jika perintah kontainer mengembalikan nilai selain 0 (sukses). Atur ke `true` untuk mengaktifkan.

`waitAfterCompletion`

(Opsional) Menunggu beberapa detik setelah perintah selesai sebelum menjalankan perintah berikutnya. Nilai default adalah **`60`** detik. Anda juga dapat menentukan**`forever`**.

> **Important**
>
> Reboot sistem selama penerapan tidak didukung. Jika sistem reboot sebagai akibat dari perintah, inisialisasi instance akan gagal, menyebabkan penerapan gagal.
> Sebagai solusinya, Anda dapat menggunakan ini `.ebextensions` `configuration`untuk menjadwalkan reboot setelah penerapan selesai.

Contoh

Contoh berikut menyimpan output dari perintah `set` ke file yang ditentukan. Elastic Beanstalk menjalankan perintah pada satu instans, dan melakukan reboot pada instans segera setelah perintah selesai.

```
container_commands:
  foo:
    command: set > c:\\myapp\\set.txt
    leader_only: true
    waitAfterCompletion: 0
```

Menambahkan dan menyesuaikan sumber daya lingkungan Elastic Beanstalk

Anda mungkin ingin menyesuaikan sumber daya lingkungan Anda yang merupakan bagian dari lingkungan Elastic Beanstalk Anda. Misalnya, Anda mungkin ingin menambahkan antrean Amazon SQS dan alarm pada kedalaman antrian, atau Anda mungkin ingin menambahkan kluster Amazon. ElastiCache Anda dapat dengan mudah menyesuaikan lingkungan Anda pada saat yang sama saat Anda men-deploy versi aplikasi Anda dengan menyertakan file konfigurasi dengan paket sumber Anda.

Anda dapat menggunakan `Resources` kunci dalam file konfigurasi untuk membuat dan menyesuaikan AWS sumber daya di lingkungan Anda. Sumber daya yang didefinisikan dalam file konfigurasi ditambahkan ke AWS CloudFormation template yang digunakan untuk meluncurkan lingkungan Anda. Semua jenis AWS CloudFormation sumber daya didukung.

> **Note**
>
> Setiap kali Anda menambahkan sumber daya yang tidak dikelola oleh Elastic Beanstalk, pastikan untuk menambahkan kebijakan pengguna dengan izin AWS Identity and Access Management yang sesuai untuk pengguna (IAM) Anda. Kebijakan pengguna terkelola yang disediakan Elastic Beanstalk hanya mencakup izin ke sumber daya terkelola Elastic Beanstalk.

Sebagai contoh, file konfigurasi berikut menambahkan hook siklus hidup Auto Scaling ke grup Auto Scaling default yang dibuat oleh Elastic Beanstalk:

~/my-app/.ebextensions/as-hook.config

```
Resources:
  hookrole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument: {
               "Version" : "2012-10-17",
               "Statement": [ {
                  "Effect": "Allow",
                  "Principal": {
                     "Service": [ "autoscaling.amazonaws.com" ]
                  },
                  "Action": [ "sts:AssumeRole" ]
               } ]
            }
      Policies: [ {
               "PolicyName": "SNS",
               "PolicyDocument": {
                     "Version": "2012-10-17",
                     "Statement": [{
                         "Effect": "Allow",
                         "Resource": "*",
                         "Action": [
                             "sqs:SendMessage",
                             "sqs:GetQueueUrl",
                             "sns:Publish"
                         ]
                       }
                     ]
                  }
               } ]
  hooktopic:
    Type: AWS::SNS::Topic
    Properties:
      Subscription:
        - Endpoint: "my-email@example.com"
          Protocol: email
  lifecyclehook:
    Type: AWS::AutoScaling::LifecycleHook
    Properties:
      AutoScalingGroupName: { "Ref" : "AWSEBAutoScalingGroup" }
      LifecycleTransition: autoscaling:EC2_INSTANCE_TERMINATING
      NotificationTargetARN: { "Ref" : "hooktopic" }
```

```
      RoleARN: { "Fn::GetAtt" : [ "hookrole", "Arn"] }
```

Contoh ini mendefinisikan tiga sumber daya, `hookrole`, `hooktopic` dan `lifecyclehook`. Dua sumber daya pertama adalah peran IAM, yang memberikan izin Amazon Auto EC2 Scaling untuk mempublikasikan pesan ke Amazon SNS, dan topik SNS, yang menyampaikan pesan dari grup Auto Scaling ke alamat email. Elastic Beanstalk membuat sumber daya ini dengan sifat dan jenis yang ditentukan.

Sumber daya akhir, `lifecyclehook`, adalah hook siklus hidup itu sendiri:

```
lifecyclehook:
  Type: AWS::AutoScaling::LifecycleHook
  Properties:
    AutoScalingGroupName: { "Ref" : "AWSEBAutoScalingGroup" }
    LifecycleTransition: autoscaling:EC2_INSTANCE_TERMINATING
    NotificationTargetARN: { "Ref" : "hooktopic" }
    RoleARN: { "Fn::GetAtt" : [ "hookrole", "Arn"] }
```

Definisi hook siklus hidup menggunakan dua fungsi untuk mengisi nilai properti hook. `{ "Ref" : "AWSEBAutoScalingGroup" }` mengambil nama grup Auto Scaling yang dibuat oleh Elastic Beanstalk untuk lingkungan. `AWSEBAutoScalingGroup` adalah salah satu nama sumber daya standar disediakan oleh Elastic Beanstalk.

Untuk `AWS::IAM::Role`, `Ref` hanya mengembalikan nama peran, bukan ARN. Untuk mendapatkan ARN untuk parameter `RoleARN`, Anda menggunakan fungsi intrinsik lain, `Fn::GetAtt` sebagai gantinya, yang bisa mendapatkan atribut apa pun dari sumber daya. `RoleARN: { "Fn::GetAtt" : [ "hookrole", "Arn"] }` mendapatkan atribut `Arn` dari sumber daya `hookrole`.

`{ "Ref" : "hooktopic" }` mendapatkan ARN dari topik Amazon SNS yang dibuat sebelumnya dalam file konfigurasi. Nilai yang dikembalikan oleh `Ref` bervariasi per jenis sumber daya dan dapat ditemukan di topik Panduan AWS CloudFormation Pengguna untuk jenis AWS::SNS::Topic sumber daya.

Memodifikasi sumber daya yang dibuat Elastic Beanstalk untuk lingkungan Anda

Sumber daya yang dibuat Elastic Beanstalk untuk lingkungan Anda memiliki nama. Anda dapat menggunakan nama ini untuk mendapatkan informasi tentang sumber daya dengan fungsi, atau memodifikasi properti pada sumber daya untuk menyesuaikan perilaku mereka. Topik ini menjelaskan AWS sumber daya yang digunakan Elastic Beanstalk di berbagai jenis lingkungan.

> **Note**
>
> Topik sebelumnya Sumber daya kustom menyediakan beberapa kasus penggunaan dan contoh untuk menyesuaikan sumber daya lingkungan. Anda juga dapat menemukan lebih banyak contoh file konfigurasi di topik selanjutnyaContoh sumber daya khusus.

Lingkungan server web memiliki sumber daya berikut.

Lingkungan server web

- `AWSEBAutoScalingGroup`(AWS::AutoScaling::AutoScalingGrup) - Grup Auto Scaling yang melekat pada lingkungan Anda.
- Salah satu dari dua sumber daya berikut.
  - `AWSEBAutoScalingLaunchConfiguration`(AWS::AutoScaling::LaunchConfiguration) - Konfigurasi peluncuran yang dilampirkan ke grup Auto Scaling lingkungan Anda.
  - `AWSEBEC2LaunchTemplate`(AWS::EC2::LaunchTemplate) - Template EC2 peluncuran Amazon yang digunakan oleh grup Auto Scaling lingkungan Anda.

  > **Note**
  >
  > Jika lingkungan Anda menggunakan fungsionalitas yang memerlukan templat EC2 peluncuran Amazon, dan kebijakan pengguna Anda tidak memiliki izin yang diperlukan, membuat atau memperbarui lingkungan mungkin gagal. Gunakan kebijakan pengguna yang dikelola AdministratorAccess- AWSElastic Beanstalk, atau tambahkan izin yang diperlukan ke kebijakan kustom Anda.

- `AWSEBEnvironmentName`(AWS::ElasticBeanstalk::Environment) — Lingkungan Anda.
- `AWSEBSecurityGroup`(AWS::EC2::SecurityGroup) - Grup keamanan yang dilampirkan ke grup Auto Scaling Anda.
- `AWSEBRDSDatabase`(AWS: :RDS:: DBInstance) — Instans Amazon RDS DB yang dilampirkan ke lingkungan Anda (jika ada).

Di lingkungan yang seimbang dengan beban, Anda dapat mengakses sumber daya tambahan yang terkait dengan penyeimbang beban. Classic load balancers memiliki sumber daya untuk penyeimbang beban dan satu untuk grup keamanan yang melekat padanya. Penyeimbang beban

aplikasi dan jaringan memiliki sumber daya tambahan bagi listener default penyeimbang beban, aturan listener, dan grup target.

Lingkungan yang seimbang dengan beban

- `AWSEBLoadBalancer`(AWS::ElasticLoadBalancing::LoadBalancer) - penyeimbang beban klasik lingkungan Anda.
- `AWSEBV2LoadBalancer`(AWS::ElasticLoadBalancingV2::LoadBalancer) — Aplikasi lingkungan Anda atau penyeimbang beban jaringan.
- `AWSEBLoadBalancerSecurityGroup`(AWS::EC2::SecurityGroup) - Hanya dalam Amazon Virtual Private Cloud (Amazon VPC) khusus, nama grup keamanan yang dibuat Elastic Beanstalk untuk penyeimbang beban. Dalam VPC default atau EC2 klasik, Elastic Load Balancing menetapkan grup keamanan default ke penyeimbang beban.
- `AWSEBV2LoadBalancerListener`(AWS::ElasticLoadBalancingV2::Listener) - Pendengar yang memungkinkan penyeimbang beban untuk memeriksa permintaan koneksi dan meneruskannya ke satu atau beberapa grup target.
- `AWSEBV2LoadBalancerListenerRule`(AWS::ElasticLoadBalancingV2::ListenerRule) — Mendefinisikan permintaan mana pendengar Elastic Load Balancing mengambil tindakan dan tindakan yang diperlukan.
- `AWSEBV2LoadBalancerTargetGroup`(AWS::ElasticLoadBalancingV2::TargetGroup) — Grup target Elastic Load Balancing yang merutekan permintaan ke satu atau beberapa target terdaftar, seperti instans Amazon EC2 .

Lingkungan pekerja memiliki sumber daya untuk antrean SQS yang menahan permintaan masuk, dan tabel Amazon DynamoDB yang digunakan oleh instans untuk pemilihan pemimpin.

Lingkungan pekerja

- `AWSEBWorkerQueue`(AWS::SQS::Queue) - Antrian Amazon SQS dari mana daemon menarik permintaan yang perlu diproses.
- `AWSEBWorkerDeadLetterQueue`(AWS::SQS::Queue) - Antrian Amazon SQS yang menyimpan pesan yang tidak dapat dikirim atau tidak berhasil diproses oleh daemon.
- `AWSEBWorkerCronLeaderRegistry`(AWS::DynamoDB::Table) - Tabel Amazon DynamoDB yang merupakan registri internal yang digunakan oleh daemon untuk tugas berkala.

Kunci AWS CloudFormation template lainnya

Kami telah memperkenalkan kunci file konfigurasi dari AWS CloudFormation sepertiResources,files, danpackages. Elastic Beanstalk menambahkan konten file konfigurasi AWS CloudFormation ke template yang mendukung lingkungan Anda, sehingga Anda dapat AWS CloudFormation menggunakan bagian lain untuk melakukan tugas-tugas lanjutan dalam file konfigurasi Anda.

Kunci

- Parameter
- Output
- Pemetaan

Parameter

Parameter adalah alternatif untuk opsi khusus Elastic Beanstalk sendiri yang dapat Anda gunakan untuk menentukan nilai yang Anda gunakan di tempat lain dalam file konfigurasi Anda. Seperti opsi khusus, Anda dapat menggunakan parameter untuk mengumpulkan nilai yang dapat dikonfigurasi pengguna di satu tempat. Tidak seperti opsi khusus, Anda tidak dapat menggunakan API Elastic Beanstalk untuk menetapkan nilai parameter, dan jumlah parameter yang dapat Anda tentukan dalam templat dibatasi oleh. AWS CloudFormation

Salah satu alasan Anda mungkin ingin menggunakan parameter adalah untuk membuat file konfigurasi Anda berfungsi ganda sebagai AWS CloudFormation templat. Jika Anda menggunakan parameter alih-alih opsi khusus, Anda dapat menggunakan file konfigurasi untuk membuat sumber daya yang sama AWS CloudFormation sebagai tumpukannya sendiri. Misalnya, Anda bisa memiliki file konfigurasi yang menambahkan sistem file Amazon EFS ke lingkungan Anda untuk pengujian, dan kemudian menggunakan file yang sama untuk membuat sistem file independen yang tidak terikat ke siklus hidup lingkungan Anda untuk penggunaan produksi.

Contoh berikut menunjukkan penggunaan parameter untuk mengumpulkan nilai yang dapat dikonfigurasi pengguna di bagian atas file konfigurasi.

Example L oadbalancer-accesslogs-existingbucket .config — Parameter

```
Parameters:
  bucket:
    Type: String
```

```
    Description: "Name of the Amazon S3 bucket in which to store load balancer logs"
    Default: "amzn-s3-demo-bucket"
  bucketprefix:
    Type: String
    Description: "Optional prefix. Can't start or end with a /, or contain the word
 AWSLogs"
    Default: ""
```

Output

Anda dapat menggunakan blok `Outputs` untuk mengekspor informasi tentang sumber daya yang dibuat ke AWS CloudFormation. Anda kemudian dapat menggunakan `Fn::ImportValue` fungsi untuk menarik nilai ke dalam AWS CloudFormation template di luar Elastic Beanstalk.

Contoh berikut membuat topik Amazon SNS dan mengekspor ARN ke dengan nama. AWS CloudFormation `NotificationTopicArn`

Example sns-topic.config

```
Resources:
  NotificationTopic:
    Type: AWS::SNS::Topic

Outputs:
  NotificationTopicArn:
    Description: Notification topic ARN
    Value: { "Ref" : "NotificationTopic" }
    Export:
      Name: NotificationTopicArn
```

Dalam file konfigurasi untuk lingkungan yang berbeda, atau AWS CloudFormation template di luar Elastic Beanstalk, Anda dapat `Fn::ImportValue` menggunakan fungsi untuk mendapatkan ARN yang diekspor. Contoh ini memberikan nilai diekspor ke properti lingkungan bernama `TOPIC_ARN`.

Example env.config

```
option_settings:
  aws:elasticbeanstalk:application:environment:
    TOPIC_ARN: '`{ "Fn::ImportValue" : "NotificationTopicArn" }`'
```

Pemetaan

Anda dapat menggunakan pemetaan untuk menyimpan pasangan nilai kunci yang diatur oleh namespace. Pemetaan dapat membantu Anda mengatur nilai yang Anda gunakan di seluruh konfigurasi Anda, atau mengubah nilai parameter tergantung pada nilai yang lain. Misalnya, konfigurasi berikut menetapkan nilai parameter ID akun berdasarkan wilayah saat ini.

Example L oadbalancer-accesslogs-newbucket .config - Pemetaan

```
Mappings:
  Region2ELBAccountId:
    us-east-1:
      AccountId: "111122223333"
    us-west-2:
      AccountId: "444455556666"
    us-west-1:
      AccountId: "123456789012"
    eu-west-1:
      AccountId: "777788889999"
...
            Principal:
              AWS:
                ? "Fn::FindInMap"
                :
                  - Region2ELBAccountId
                  -
                    Ref: "AWS::Region"
                  - AccountId
```

Fungsi

Anda dapat menggunakan fungsi dalam file konfigurasi Anda untuk mengisi nilai untuk properti sumber daya dengan informasi dari sumber daya lain atau dari pengaturan opsi konfigurasi Elastic Beanstalk. Elastic AWS CloudFormation Beanstalk mendukung `Ref` fungsi `Fn::GetAtt` (,`Fn::Join`,), dan satu fungsi khusus Elastic Beanstalk,. `Fn::GetOptionSetting`

Fungsi

- Ref
- Fn:: GetAtt
- Fn::Join

- Fn:: GetOptionSetting

Ref

Gunakan `Ref` untuk mengambil representasi string default dari AWS sumber daya. Nilai yang dikembalikan `Ref` tergantung pada jenis sumber daya, dan kadang-kadang tergantung pada faktor lain juga. Misalnya, grup keamanan (AWS::EC2::SecurityGroup) menampilkan nama atau ID grup keamanan, tergantung pada apakah grup keamanan berada di Amazon Virtual Private Cloud default (Amazon VPC), EC2 klasik, atau VPC kustom.

```
{ "Ref" : "resource name" }
```

> **Note**
>
> Untuk detail di setiap jenis sumber daya, termasuk nilai kembali dari `Ref`, lihat Referensi Jenis Sumber Daya AWS di Panduan Pengguna AWS CloudFormation .

Dari sampel hook siklus hidup Auto Scaling:

```
Resources:
  lifecyclehook:
    Type: AWS::AutoScaling::LifecycleHook
    Properties:
      AutoScalingGroupName: { "Ref" : "AWSEBAutoScalingGroup" }
```

Anda juga dapat menggunakan `Ref` untuk mengambil nilai AWS CloudFormation parameter yang ditentukan di tempat lain dalam file yang sama atau dalam file konfigurasi yang berbeda.

Fn:: GetAtt

Gunakan `Fn::GetAtt` untuk mengambil nilai atribut pada AWS sumber daya.

```
{ "Fn::GetAtt" : [ "resource name", "attribute name"] }
```

Dari sampel hook siklus hidup Auto Scaling:

```
Resources:
```

```
  lifecyclehook:
    Type: AWS::AutoScaling::LifecycleHook
    Properties:
      RoleARN: { "Fn::GetAtt" : [ "hookrole", "Arn"] }
```

Lihat Fn:: GetAtt untuk informasi lebih lanjut.

Fn::Join

Gunakan `Fn::Join` untuk menggabungkan string dengan pembatas. String dapat di-hardcode atau menggunakan output dari `Fn::GetAtt` atau `Ref`.

```
{ "Fn::Join" : [ "delimiter", [ "string1", "string2" ] ] }
```

Lihat Fn::Join untuk informasi selengkapnya.

Fn:: GetOptionSetting

Gunakan `Fn::GetOptionSetting` untuk mengambil nilai pengaturan opsi konfigurasi yang diterapkan ke lingkungan.

```
"Fn::GetOptionSetting":
  Namespace: "namespace"
  OptionName: "option name"
  DefaultValue: "default value"
```

Dari contoh penyimpanan kunci pribadi:

```
Resources:
  AWSEBAutoScalingGroup:
    Metadata:
      AWS::CloudFormation::Authentication:
        S3Auth:
          type: "s3"
          buckets: ["elasticbeanstalk-us-west-2-123456789012"]
          roleName:
            "Fn::GetOptionSetting":
              Namespace: "aws:autoscaling:launchconfiguration"
              OptionName: "IamInstanceProfile"
              DefaultValue: "aws-elasticbeanstalk-ec2-role"
```

Contoh sumber daya khusus

Berikut ini adalah daftar contoh file konfigurasi yang dapat Anda gunakan untuk menyesuaikan lingkungan Elastic Beanstalk Anda:

- DynamoDB,, dan CloudWatch SNS
- Elastic Load Balancing dan CloudWatch
- ElastiCache
- RDS dan CloudWatch
- SQS, SNS, dan CloudWatch

Subtopik halaman ini memberikan beberapa contoh tambahan untuk menambahkan dan mengonfigurasi sumber daya khusus di lingkungan Elastic Beanstalk.

Contoh

- Contoh: ElastiCache
- Contoh: SQS, CloudWatch, dan SNS
- Contoh: DynamoDB,, dan CloudWatch SNS

Contoh: ElastiCache

Contoh berikut menambahkan ElastiCache kluster Amazon ke platform EC2 -Classic dan EC2 -VPC (baik Amazon Virtual Private Cloud (Amazon VPC) default maupun kustom. Untuk informasi selengkapnya tentang platform ini dan bagaimana Anda dapat menentukan mana yang EC2 mendukung wilayah dan AWS akun Anda, lihathttps://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-supported-platforms.html. Kemudian lihat bagian dalam topik ini yang berlaku untuk platform Anda.

- EC2-platform klasik
- EC2-VPC (default)
- EC2-VPC (kustom)

EC2-platform klasik

Contoh ini menambahkan ElastiCache klaster Amazon ke lingkungan dengan instance yang diluncurkan ke platform EC2 -Classic. Semua properti yang tercantum dalam contoh ini adalah

properti minimum yang diperlukan yang harus ditetapkan untuk setiap jenis sumber daya. Anda dapat mengunduh contoh di ElastiCachecontoh.

> **Note**
>
> Contoh ini menciptakan AWS sumber daya, yang mungkin dikenakan biaya untuk Anda. Untuk informasi lebih lanjut tentang AWS harga, lihathttps://aws.amazon.com/pricing/. Beberapa layanan adalah bagian dari Tingkat Penggunaan AWS Gratis. Jika Anda adalah pelanggan baru, Anda dapat menguji layanan ini secara gratis. Lihat https://aws.amazon.com/free/ untuk informasi selengkapnya.

Untuk menggunakan contoh ini, lakukan hal berikut:

1. Buat direktori `.ebextensions` di tingkat atas dari paket sumber Anda.
2. Buat dua file konfigurasi dengan ekstensi `.config` dan tempatkan mereka di direktori `.ebextensions` Anda. Satu file konfigurasi menentukan sumber daya, dan file konfigurasi lainnya menentukan pilihan.
3. Terapkan aplikasi Anda ke Elastic Beanstalk.

   YAML bergantung pada indentasi yang konsisten. Cocokkan tingkat indentasi saat mengganti konten dalam file konfigurasi contoh dan pastikan bahwa editor teks Anda menggunakan spasi, bukan karakter tab, untuk indentasi.

Buat file konfigurasi (misalnya, `elasticache.config`) yang ditentukan sumber daya. Dalam contoh ini, kita membuat ElastiCache cluster dengan menentukan nama sumber daya ElastiCache cluster (`MyElastiCache`), mendeklarasikan jenisnya, dan kemudian mengkonfigurasi properti untuk cluster. Contoh referensi nama sumber daya grup ElastiCache keamanan yang akan dibuat dan didefinisikan dalam file konfigurasi ini. Selanjutnya, kami membuat grup ElastiCache keamanan. Kami menentukan nama untuk sumber daya ini, menyatakan jenisnya, dan menambahkan deskripsi untuk grup keamanan. Akhirnya, kami menetapkan aturan ingress untuk grup ElastiCache keamanan untuk mengizinkan akses hanya dari instance di dalam grup keamanan (`MyCacheSecurityGroup`) dan grup ElastiCache keamanan Elastic Beanstalk (). `AWSEBSecurityGroup` Nama parameter, `AWSEBSecurityGroup`, adalah nama sumber daya tetap yang disediakan oleh Elastic Beanstalk. Anda `AWSEBSecurityGroup` harus menambahkan aturan masuk grup ElastiCache keamanan Anda agar aplikasi Elastic Beanstalk Anda terhubung ke instance di cluster Anda. ElastiCache

```
#This sample requires you to create a separate configuration file that defines the
 custom option settings for CacheCluster properties.

Resources:
  MyElastiCache:
    Type: AWS::ElastiCache::CacheCluster
    Properties:
      CacheNodeType:
         Fn::GetOptionSetting:
             OptionName : CacheNodeType
             DefaultValue: cache.m1.small
      NumCacheNodes:
           Fn::GetOptionSetting:
             OptionName : NumCacheNodes
             DefaultValue: 1
      Engine:
           Fn::GetOptionSetting:
             OptionName : Engine
             DefaultValue: memcached
      CacheSecurityGroupNames:
        - Ref: MyCacheSecurityGroup
  MyCacheSecurityGroup:
    Type: AWS::ElastiCache::SecurityGroup
    Properties:
      Description: "Lock cache down to webserver access only"
  MyCacheSecurityGroupIngress:
    Type: AWS::ElastiCache::SecurityGroupIngress
    Properties:
      CacheSecurityGroupName:
        Ref: MyCacheSecurityGroup
      EC2SecurityGroupName:
        Ref: AWSEBSecurityGroup
```

Untuk informasi selengkapnya tentang sumber daya yang digunakan dalam contoh file konfigurasi ini, lihat referensi berikut ini:

- AWS::ElastiCache::CacheCluster
- AWS::ElastiCache::SecurityGroup
- AWS::ElastiCache: SecurityGroupIngress

Buat file konfigurasi terpisah yang disebut `options.config` dan tentukan pengaturan opsi khusus.

```
option_settings:
  "aws:elasticbeanstalk:customoption":
     CacheNodeType : cache.m1.small
     NumCacheNodes : 1
     Engine : memcached
```

Baris-baris ini memberi tahu Elastic Beanstalk untuk mendapatkan nilai CacheNodeTypeuntuk NumCacheNodes properti,, dan Engine CacheNodeTypedari NumCacheNodes,, dan nilai Engine dalam file konfigurasi (options.config dalam contoh kita) yang berisi bagian option_settings dengan bagian aws:elasticbeanstalk:customoption yang berisi pasangan nama-nilai yang berisi nilai aktual yang akan digunakan. Pada contoh di atas, ini berarti cache.m1.small, 1, dan memcached akan digunakan untuk nilainya. Untuk informasi selengkapnya tentang `Fn::GetOptionSetting`, lihat Fungsi.

EC2-VPC (default)

Contoh ini menambahkan ElastiCache klaster Amazon ke lingkungan dengan instance yang diluncurkan ke platform EC2 -VPC. Secara khusus, informasi di bagian ini berlaku untuk skenario di mana EC2 meluncurkan instance ke VPC default. Semua properti pada contoh ini adalah properti minimum yang diperlukan yang harus ditetapkan untuk setiap jenis sumber daya. Untuk informasi selengkapnya tentang default VPCs, lihat VPC dan Subnet Default Anda.

> **Note**
>
> Contoh ini menciptakan AWS sumber daya, yang mungkin dikenakan biaya untuk Anda. Untuk informasi lebih lanjut tentang AWS harga, lihathttps://aws.amazon.com/pricing/. Beberapa layanan adalah bagian dari Tingkat Penggunaan AWS Gratis. Jika Anda adalah pelanggan baru, Anda dapat menguji layanan ini secara gratis. Lihat https://aws.amazon.com/free/ untuk informasi selengkapnya.

Untuk menggunakan contoh ini, lakukan hal berikut:

1. Buat direktori `.ebextensions` di tingkat atas dari paket sumber Anda.
2. Buat dua file konfigurasi dengan ekstensi `.config` dan tempatkan mereka di direktori `.ebextensions` Anda. Satu file konfigurasi menentukan sumber daya, dan file konfigurasi lainnya menentukan pilihan.
3. Terapkan aplikasi Anda ke Elastic Beanstalk.

YAML bergantung pada indentasi yang konsisten. Cocokkan tingkat indentasi saat mengganti konten dalam file konfigurasi contoh dan pastikan bahwa editor teks Anda menggunakan spasi, bukan karakter tab, untuk indentasi.

Sekarang beri nama file konfigurasi sumber daya `elasticache.config`. Untuk membuat ElastiCache cluster, contoh ini menentukan nama sumber daya ElastiCache cluster (`MyElastiCache`), mendeklarasikan jenisnya, dan kemudian mengkonfigurasi properti untuk cluster. Contoh ini merujuk ID sumber daya grup keamanan yang kami buat dan tentukan dalam file konfigurasi ini.

Selanjutnya, kami membuat grup EC2 keamanan. Kami menentukan nama untuk sumber daya ini, menyatakan jenisnya, menambahkan deskripsi, dan menetapkan aturan masuk ke grup keamanan untuk mengizinkan akses hanya dari instans di dalam grup keamanan Elastic Beanstalk (`AWSEBSecurityGroup`). (Nama parameter,`AWSEBSecurityGroup`, adalah nama sumber daya tetap yang disediakan oleh Elastic Beanstalk. Anda `AWSEBSecurityGroup` harus menambahkan aturan masuknya grup ElastiCache keamanan agar aplikasi Elastic Beanstalk Anda dapat terhubung ke instance di cluster Anda.) ElastiCache

Aturan ingress untuk grup EC2 keamanan juga menentukan protokol IP dan nomor port di mana node cache dapat menerima koneksi. Untuk Redis, nomor port default-nya adalah 6379.

```
#This sample requires you to create a separate configuration file that defines the
 custom option settings for CacheCluster properties.

Resources:
  MyCacheSecurityGroup:
    Type: "AWS::EC2::SecurityGroup"
    Properties:
      GroupDescription: "Lock cache down to webserver access only"
      SecurityGroupIngress :
        - IpProtocol : "tcp"
          FromPort :
            Fn::GetOptionSetting:
              OptionName : "CachePort"
              DefaultValue: "6379"
          ToPort :
            Fn::GetOptionSetting:
              OptionName : "CachePort"
              DefaultValue: "6379"
          SourceSecurityGroupName:
```

```
            Ref: "AWSEBSecurityGroup"
  MyElastiCache:
    Type: "AWS::ElastiCache::CacheCluster"
    Properties:
      CacheNodeType:
        Fn::GetOptionSetting:
          OptionName : "CacheNodeType"
          DefaultValue : "cache.t2.micro"
      NumCacheNodes:
        Fn::GetOptionSetting:
          OptionName : "NumCacheNodes"
          DefaultValue : "1"
      Engine:
        Fn::GetOptionSetting:
          OptionName : "Engine"
          DefaultValue : "redis"
      VpcSecurityGroupIds:
        -
          Fn::GetAtt:
            - MyCacheSecurityGroup
            - GroupId

Outputs:
  ElastiCache:
    Description : "ID of ElastiCache Cache Cluster with Redis Engine"
    Value :
      Ref : "MyElastiCache"
```

Untuk informasi selengkapnya tentang sumber daya yang digunakan dalam contoh file konfigurasi ini, lihat referensi berikut ini:

- AWS::ElastiCache::CacheCluster
- AWS::EC2::SecurityGroup

Selanjutnya, beri nama file konfigurasi opsi `options.config` dan tentukan pengaturan opsi khusus.

```
option_settings:
  "aws:elasticbeanstalk:customoption":
    CacheNodeType : cache.t2.micro
    NumCacheNodes : 1
    Engine : redis
```

```
    CachePort : 6379
```

Garis-garis ini memerintahkan Elastic Beanstalk untuk mendapatkan nilai untuk properti `CacheNodeType`, `NumCacheNodes`, `Engine`, dan `CachePort` dari nilai `CacheNodeType`, `NumCacheNodes`, `Engine`, dan `CachePort` pada file konfigurasi (`options.config` dalam contoh kami). File tersebut mencakup bagian `aws:elasticbeanstalk:customoption` (di bawah `option_settings`) yang berisi pasangan nilai-nama dengan nilai sebenarnya yang akan digunakan. Pada contoh sebelumnya, `cache.t2.micro`, `1`, `redis`, dan `6379` akan digunakan untuk nilai tersebut. Untuk informasi selengkapnya tentang `Fn::GetOptionSetting`, lihat Fungsi.

EC2-VPC (kustom)

Jika Anda membuat VPC kustom di platform EC2 -VPC dan menentukannya sebagai VPC tempat EC2 peluncuran instance, proses penambahan ElastiCache cluster Amazon ke lingkungan Anda berbeda dari VPC default. Perbedaan utamanya adalah Anda harus membuat grup subnet untuk ElastiCache cluster. Semua properti pada contoh ini adalah properti minimum yang diperlukan yang harus ditetapkan untuk setiap jenis sumber daya.

> **Note**
>
> Contoh ini menciptakan AWS sumber daya, yang mungkin dikenakan biaya untuk Anda. Untuk informasi lebih lanjut tentang AWS harga, lihathttps://aws.amazon.com/pricing/. Beberapa layanan adalah bagian dari Tingkat Penggunaan AWS Gratis. Jika Anda adalah pelanggan baru, Anda dapat menguji layanan ini secara gratis. Lihat https://aws.amazon.com/free/ untuk informasi selengkapnya.

Untuk menggunakan contoh ini, lakukan hal berikut:

1. Buat direktori `.ebextensions` di tingkat atas dari paket sumber Anda.
2. Buat dua file konfigurasi dengan ekstensi `.config` dan tempatkan mereka di direktori `.ebextensions` Anda. Satu file konfigurasi menentukan sumber daya, dan file konfigurasi lainnya menentukan pilihan.
3. Terapkan aplikasi Anda ke Elastic Beanstalk.

   YAML bergantung pada indentasi yang konsisten. Cocokkan tingkat indentasi saat mengganti konten dalam file konfigurasi contoh dan pastikan bahwa editor teks Anda menggunakan spasi, bukan karakter tab, untuk indentasi.

Sekarang beri nama file konfigurasi sumber daya `elasticache.config`. Untuk membuat ElastiCache cluster, contoh ini menentukan nama sumber daya ElastiCache cluster (`MyElastiCache`), mendeklarasikan jenisnya, dan kemudian mengkonfigurasi properti untuk cluster. Properti dalam contoh referensi nama grup subnet untuk ElastiCache cluster serta ID sumber daya grup keamanan yang kita buat dan tentukan dalam file konfigurasi ini.

Selanjutnya, kami membuat grup EC2 keamanan. Kami menentukan nama untuk sumber daya ini, menyatakan jenisnya, menambahkan deskripsi, ID VPC, dan menetapkan aturan masuk grup keamanan untuk mengizinkan akses hanya dari insyans di dalam grup keamanan Elastic Beanstalk (`AWSEBSecurityGroup`). (Nama parameter,`AWSEBSecurityGroup`, adalah nama sumber daya tetap yang disediakan oleh Elastic Beanstalk. Anda `AWSEBSecurityGroup` harus menambahkan aturan masuknya grup ElastiCache keamanan agar aplikasi Elastic Beanstalk Anda dapat terhubung ke instance di cluster Anda.) ElastiCache

Aturan ingress untuk grup EC2 keamanan juga menentukan protokol IP dan nomor port di mana node cache dapat menerima koneksi. Untuk Redis, nomor port default-nya adalah 6379. Akhirnya, contoh ini membuat grup subnet untuk ElastiCache cluster. Kami menentukan nama untuk sumber daya ini, menyatakan jenisnya, dan menambahkan deskripsi dan ID subnet pada grup subnet.

> **Note**
>
> Kami menyarankan Anda menggunakan subnet pribadi untuk ElastiCache cluster. Untuk informasi lebih lanjut tentang VPC dengan subnet pribadi, lihat https://docs.aws.amazon.com/vpc/latest/userguide/VPC_Scenario2.html.

```
#This sample requires you to create a separate configuration file that defines the
 custom option settings for CacheCluster properties.

Resources:
  MyElastiCache:
    Type: "AWS::ElastiCache::CacheCluster"
    Properties:
      CacheNodeType:
        Fn::GetOptionSetting:
          OptionName : "CacheNodeType"
          DefaultValue : "cache.t2.micro"
      NumCacheNodes:
        Fn::GetOptionSetting:
          OptionName : "NumCacheNodes"
```

```
          DefaultValue : "1"
      Engine:
        Fn::GetOptionSetting:
          OptionName : "Engine"
          DefaultValue : "redis"
      CacheSubnetGroupName:
        Ref: "MyCacheSubnets"
      VpcSecurityGroupIds:
        - Ref: "MyCacheSecurityGroup"
  MyCacheSecurityGroup:
    Type: "AWS::EC2::SecurityGroup"
    Properties:
      GroupDescription: "Lock cache down to webserver access only"
      VpcId:
        Fn::GetOptionSetting:
          OptionName : "VpcId"
      SecurityGroupIngress :
        - IpProtocol : "tcp"
          FromPort :
            Fn::GetOptionSetting:
              OptionName : "CachePort"
              DefaultValue: "6379"
          ToPort :
            Fn::GetOptionSetting:
              OptionName : "CachePort"
              DefaultValue: "6379"
          SourceSecurityGroupId:
            Ref: "AWSEBSecurityGroup"
  MyCacheSubnets:
    Type: "AWS::ElastiCache::SubnetGroup"
    Properties:
      Description: "Subnets for ElastiCache"
      SubnetIds:
        Fn::GetOptionSetting:
          OptionName : "CacheSubnets"
Outputs:
  ElastiCache:
    Description : "ID of ElastiCache Cache Cluster with Redis Engine"
    Value :
      Ref : "MyElastiCache"
```

Untuk informasi selengkapnya tentang sumber daya yang digunakan dalam contoh file konfigurasi ini, lihat referensi berikut ini:

- AWS::ElastiCache::CacheCluster
- AWS::EC2::SecurityGroup
- AWS::ElastiCache::SubnetGroup

Selanjutnya, beri nama file konfigurasi opsi `options.config` dan tentukan pengaturan opsi khusus.

> **Note**
>
> Pada contoh berikut, ganti contoh `CacheSubnets` dan nilai `VpcId` dengan subnet dan VPC Anda sendiri.

```
option_settings:
  "aws:elasticbeanstalk:customoption":
    CacheNodeType : cache.t2.micro
    NumCacheNodes : 1
    Engine : redis
    CachePort : 6379
    CacheSubnets:
      - subnet-1a1a1a1a
      - subnet-2b2b2b2b
      - subnet-3c3c3c3c
    VpcId: vpc-4d4d4d4d
```

Garis-garis ini memerintahkan Elastic Beanstalk untuk mendapatkan nilai untuk properti `CacheNodeType`, `NumCacheNodes`, `Engine`, `CachePort`, `CacheSubnets`, dan `VpcId` dari nilai `CacheNodeType`, `NumCacheNodes`, `Engine`, `CachePort`, `CacheSubnets`, dan `VpcId` dalam file konfigurasi (`options.config` di contoh kami). File tersebut mencakup bagian `aws:elasticbeanstalk:customoption` (di bawah `option_settings`) yang berisi pasangan nilai-nama dengan nilai sampel. Pada contoh di atas, `cache.t2.micro`, `1`, `redis`, `6379`, `subnet-1a1a1a1a`, `subnet-2b2b2b2b`, `subnet-3c3c3c3c`, dan `vpc-4d4d4d4d` akan digunakan untuk nilai. Untuk informasi selengkapnya tentang `Fn::GetOptionSetting`, lihat Fungsi.

Contoh: SQS, CloudWatch, dan SNS

Contoh ini menambahkan antrean Amazon SQS dan alarm pada kedalaman antrean lingkungan. Properti yang Anda lihat dalam contoh ini adalah properti minimum diperlukan yang harus Anda

tetapkan untuk masing-masing sumber daya ini. Anda dapat mengunduh contoh di SQS, SNS, dan. CloudWatch

> **Note**
>
> Contoh ini menciptakan AWS sumber daya, yang mungkin dikenakan biaya untuk Anda. Untuk informasi lebih lanjut tentang AWS harga, lihathttps://aws.amazon.com/pricing/. Beberapa layanan adalah bagian dari Tingkat Penggunaan AWS Gratis. Jika Anda adalah pelanggan baru, Anda dapat menguji layanan ini secara gratis. Lihat https://aws.amazon.com/free/ untuk informasi selengkapnya.

Untuk menggunakan contoh ini, lakukan hal berikut:

1. Buat direktori `.ebextensions` di tingkat atas dari paket sumber Anda.
2. Buat dua file konfigurasi dengan ekstensi `.config` dan tempatkan mereka di direktori `.ebextensions` Anda. Satu file konfigurasi menentukan sumber daya, dan file konfigurasi lainnya menentukan pilihan.
3. Terapkan aplikasi Anda ke Elastic Beanstalk.

   YAML bergantung pada indentasi yang konsisten. Cocokkan tingkat indentasi saat mengganti konten dalam file konfigurasi contoh dan pastikan bahwa editor teks Anda menggunakan spasi, bukan karakter tab, untuk indentasi.

Buat file konfigurasi (misalnya, sqs.config) yang mendefinisikan sumber daya. Pada contoh ini, kami membuat antrean SQS dan menentukan properti `VisbilityTimeout` di sumber daya `MySQSQueue`. Selanjutnya, kami membuat SNS `Topic` dan menetapkan bahwa email akan dikirim ke `someone@example.com` saat alarm diaktifkan. Akhirnya, kami membuat CloudWatch alarm jika antrian tumbuh melampaui 10 pesan. Di properti `Dimensions` tersebut, kami menentukan nama dimensi dan nilai yang mewakili pengukuran dimensi. Kami menggunakan `Fn::GetAtt` untuk mengembalikan nilai `QueueName` dari `MySQSQueue`.

```
#This sample requires you to create a separate configuration file to define the custom
 options for the SNS topic and SQS queue.
Resources:
  MySQSQueue:
    Type: AWS::SQS::Queue
    Properties:
```

```
      VisibilityTimeout:
        Fn::GetOptionSetting:
          OptionName: VisibilityTimeout
          DefaultValue: 30
  AlarmTopic:
    Type: AWS::SNS::Topic
    Properties:
      Subscription:
        - Endpoint:
            Fn::GetOptionSetting:
              OptionName: AlarmEmail
              DefaultValue: "nobody@amazon.com"
          Protocol: email
  QueueDepthAlarm:
    Type: AWS::CloudWatch::Alarm
    Properties:
      AlarmDescription: "Alarm if queue depth grows beyond 10 messages"
      Namespace: "AWS/SQS"
      MetricName: ApproximateNumberOfMessagesVisible
      Dimensions:
        - Name: QueueName
          Value : { "Fn::GetAtt" : [ "MySQSQueue", "QueueName"] }
      Statistic: Sum
      Period: 300
      EvaluationPeriods: 1
      Threshold: 10
      ComparisonOperator: GreaterThanThreshold
      AlarmActions:
        - Ref: AlarmTopic
      InsufficientDataActions:
        - Ref: AlarmTopic

Outputs :
  QueueURL:
    Description : "URL of newly created SQS Queue"
    Value : { Ref : "MySQSQueue" }
  QueueARN :
    Description : "ARN of newly created SQS Queue"
    Value : { "Fn::GetAtt" : [ "MySQSQueue", "Arn"]}
  QueueName :
    Description : "Name newly created SQS Queue"
    Value : { "Fn::GetAtt" : [ "MySQSQueue", "QueueName"]}
```

Untuk informasi selengkapnya tentang sumber daya yang digunakan dalam contoh file konfigurasi ini, lihat referensi berikut ini:

- AWS::SQS::Queue
- AWS::SNS::Topic
- AWS::CloudWatch::Alarm

Buat file konfigurasi terpisah yang disebut `options.config` dan tentukan pengaturan opsi khusus.

```
option_settings:
  "aws:elasticbeanstalk:customoption":
     VisibilityTimeout : 30
     AlarmEmail : "nobody@example.com"
```

Baris ini memberi tahu Elastic Beanstalk untuk mendapatkan nilai untuk properti dan Subscription Endpoint VisibilityTimeout dari nilai VisibilityTimeout dan Subscription Endpoint dalam file konfigurasi (options.config dalam contoh kita) yang berisi bagian option_settings dengan bagian aws:elasticbeanstalk:customoption yang berisi pasangan nama-nilai yang berisi nilai aktual yang akan digunakan. Pada contoh di atas, ini berarti 30 dan "nobody@amazon.com" akan digunakan untuk nilai tersebut. Untuk informasi selengkapnya tentang `Fn::GetOptionSetting`, lihat the section called “Fungsi”.

Contoh: DynamoDB,, dan CloudWatch SNS

File konfigurasi ini mengatur tabel DynamoDB sebagai pengendali sesi untuk aplikasi berbasis PHP menggunakan SDK for PHP 2. AWS Untuk menggunakan contoh ini, Anda harus memiliki profil instans IAM, yang ditambahkan ke instans di lingkungan Anda dan digunakan untuk mengakses tabel DynamoDB.

Anda dapat men-download sampel yang akan kita gunakan dalam langkah ini di DynamoDB session Support contoh. Sampel berisi file berikut:

- Contoh aplikasi, `index.php`
- File konfigurasi,`dynamodb.config`, untuk membuat dan mengkonfigurasi tabel DynamoDB dan sumber daya AWS lainnya dan menginstal perangkat lunak pada instance yang meng-host aplikasi EC2 di lingkungan Elastic Beanstalk
- File konfigurasi, `options.config`, yang menimpa file default di `dynamodb.config` dengan pengaturan khusus untuk pemasangan khusus ini

index.php

```
<?php

// Include the SDK using the Composer autoloader
require '../vendor/autoload.php';

use Aws\DynamoDb\DynamoDbClient;

// Grab the session table name and region from the configuration file
list($tableName, $region) = file(__DIR__ . '/../sessiontable');
$tableName = rtrim($tableName);
$region = rtrim($region);

// Create a DynamoDB client and register the table as the session handler
$dynamodb = DynamoDbClient::factory(array('region' => $region));
$handler = $dynamodb->registerSessionHandler(array('table_name' => $tableName,
 'hash_key' => 'username'));

// Grab the instance ID so we can display the EC2 instance that services the request
$instanceId = file_get_contents("http://169.254.169.254/latest/meta-data/instance-id");
?>
<h1>Elastic Beanstalk PHP Sessions Sample</h1>
<p>This sample application shows the integration of the Elastic Beanstalk PHP
container and the session support for DynamoDB from the AWS SDK for PHP 2.
Using DynamoDB session support, the application can be scaled out across
multiple web servers. For more details, see the
<a href="https://aws.amazon.com/php/">PHP Developer Center</a>.</p>

<form id="SimpleForm" name="SimpleForm" method="post" action="index.php">
<?php
echo 'Request serviced from instance ' . $instanceId . '<br/>';
echo '<br/>';

if (isset($_POST['continue'])) {
  session_start();
  $_SESSION['visits'] = $_SESSION['visits'] + 1;
  echo 'Welcome back ' . $_SESSION['username'] . '<br/>';
  echo 'This is visit number ' . $_SESSION['visits'] . '<br/>';
  session_write_close();
  echo '<br/>';
  echo '<input type="Submit" value="Refresh" name="continue" id="continue"/>';
  echo '<input type="Submit" value="Delete Session" name="killsession"
 id="killsession"/>';
```

```
} elseif (isset($_POST['killsession'])) {
  session_start();
  echo 'Goodbye ' . $_SESSION['username'] . '<br/>';
  session_destroy();
  echo 'Username: <input type="text" name="username" id="username" size="30"/><br/>';
  echo '<br/>';
  echo '<input type="Submit" value="New Session" name="newsession" id="newsession"/>';
} elseif (isset($_POST['newsession'])) {
  session_start();
  $_SESSION['username'] = $_POST['username'];
  $_SESSION['visits'] = 1;
  echo 'Welcome to a new session ' . $_SESSION['username'] . '<br/>';
  session_write_close();
  echo '<br/>';
  echo '<input type="Submit" value="Refresh" name="continue" id="continue"/>';
  echo '<input type="Submit" value="Delete Session" name="killsession"
 id="killsession"/>';
} else {
  echo 'To get started, enter a username.<br/>';
  echo '<br/>';
  echo 'Username: <input type="text" name="username" id="username" size="30"/><br/>';
  echo '<input type="Submit" value="New Session" name="newsession" id="newsession"/>';
}
?>
</form>
```

.ebextensions/dynamodb.config

```
Resources:
  SessionTable:
    Type: AWS::DynamoDB::Table
    Properties:
      KeySchema:
        HashKeyElement:
          AttributeName:
            Fn::GetOptionSetting:
              OptionName : SessionHashKeyName
              DefaultValue: "username"
          AttributeType:
            Fn::GetOptionSetting:
              OptionName : SessionHashKeyType
              DefaultValue: "S"
      ProvisionedThroughput:
```

```
        ReadCapacityUnits:
          Fn::GetOptionSetting:
            OptionName : SessionReadCapacityUnits
            DefaultValue: 1
        WriteCapacityUnits:
          Fn::GetOptionSetting:
            OptionName : SessionWriteCapacityUnits
            DefaultValue: 1

  SessionWriteCapacityUnitsLimit:
    Type: AWS::CloudWatch::Alarm
    Properties:
      AlarmDescription: { "Fn::Join" : ["", [{ "Ref" : "AWSEBEnvironmentName" }, "
write capacity limit on the session table." ]]}
      Namespace: "AWS/DynamoDB"
      MetricName: ConsumedWriteCapacityUnits
      Dimensions:
        - Name: TableName
          Value: { "Ref" : "SessionTable" }
      Statistic: Sum
      Period: 300
      EvaluationPeriods: 12
      Threshold:
          Fn::GetOptionSetting:
            OptionName : SessionWriteCapacityUnitsAlarmThreshold
            DefaultValue: 240
      ComparisonOperator: GreaterThanThreshold
      AlarmActions:
        - Ref: SessionAlarmTopic
      InsufficientDataActions:
        - Ref: SessionAlarmTopic

  SessionReadCapacityUnitsLimit:
    Type: AWS::CloudWatch::Alarm
    Properties:
      AlarmDescription: { "Fn::Join" : ["", [{ "Ref" : "AWSEBEnvironmentName" }, " read
capacity limit on the session table." ]]}
      Namespace: "AWS/DynamoDB"
      MetricName: ConsumedReadCapacityUnits
      Dimensions:
        - Name: TableName
          Value: { "Ref" : "SessionTable" }
      Statistic: Sum
      Period: 300
```

```
      EvaluationPeriods: 12
      Threshold:
          Fn::GetOptionSetting:
            OptionName : SessionReadCapacityUnitsAlarmThreshold
            DefaultValue: 240
      ComparisonOperator: GreaterThanThreshold
      AlarmActions:
        - Ref: SessionAlarmTopic
      InsufficientDataActions:
        - Ref: SessionAlarmTopic

  SessionThrottledRequestsAlarm:
    Type: AWS::CloudWatch::Alarm
    Properties:
      AlarmDescription: { "Fn::Join" : ["", [{ "Ref" : "AWSEBEnvironmentName" }, ":
requests are being throttled." ]]}
      Namespace: AWS/DynamoDB
      MetricName: ThrottledRequests
      Dimensions:
        - Name: TableName
          Value: { "Ref" : "SessionTable" }
      Statistic: Sum
      Period: 300
      EvaluationPeriods: 1
      Threshold:
        Fn::GetOptionSetting:
          OptionName: SessionThrottledRequestsThreshold
          DefaultValue: 1
      ComparisonOperator: GreaterThanThreshold
      AlarmActions:
        - Ref: SessionAlarmTopic
      InsufficientDataActions:
        - Ref: SessionAlarmTopic

  SessionAlarmTopic:
    Type: AWS::SNS::Topic
    Properties:
      Subscription:
        - Endpoint:
            Fn::GetOptionSetting:
              OptionName: SessionAlarmEmail
              DefaultValue: "nobody@amazon.com"
          Protocol: email
```

```
files:
  "/var/app/sessiontable":
    mode: "000444"
    content: |
      `{"Ref" : "SessionTable"}`
      `{"Ref" : "AWS::Region"}`

  "/var/app/composer.json":
    mode: "000744"
    content:
      {
        "require": {
           "aws/aws-sdk-php": "*"
        }
      }

container_commands:
 "1-install-composer":
   command: "cd /var/app; curl -s http://getcomposer.org/installer | php"
 "2-install-dependencies":
   command: "cd /var/app; php composer.phar install"
 "3-cleanup-composer":
   command: "rm -Rf /var/app/composer.*"
```

Dalam file konfigurasi sampel, pertama kami membuat tabel DynamoDB dan mengonfigurasi struktur kunci utama untuk tabel dan unit kapasitas untuk mengalokasikan sumber daya yang cukup untuk menyediakan throughput yang diminta. Selanjutnya, kami membuat CloudWatch alarm untuk `WriteCapacity` dan`ReadCapacity`. Kami membuat topik SNS yang mengirim email ke "nobody@amazon.com" jika ambang alarm dilanggar.

Setelah kita membuat dan mengkonfigurasi AWS sumber daya kita untuk lingkungan kita, kita perlu menyesuaikan EC2 instance. Kami menggunakan `files` kunci untuk meneruskan rincian tabel DynamoDB ke instance EC2 di lingkungan kami serta menambahkan “require” dalam `composer.json` file untuk SDK AWS for PHP 2. Terakhir, kami menjalankan perintah kontainer untuk menginstal komposer, dependensi yang diperlukan, dan kemudian menghapus penginstal.

`.ebextensions/options.config`

```
option_settings:
  "aws:elasticbeanstalk:customoption":
     SessionHashKeyName                    : username
     SessionHashKeyType                    : S
```

```
    SessionReadCapacityUnits                : 1
    SessionReadCapacityUnitsAlarmThreshold  : 240
    SessionWriteCapacityUnits               : 1
    SessionWriteCapacityUnitsAlarmThreshold : 240
    SessionThrottledRequestsThreshold       : 1
    SessionAlarmEmail                       : me@example.com
```

Ganti SessionAlarmEmail nilainya dengan email tempat Anda ingin pemberitahuan alarm dikirim. File `options.config` tersebut berisi nilai-nilai yang digunakan untuk beberapa variabel yang ditentukan dalam `dynamodb.config`. Misalnya, `dynamodb.config` berisi baris berikut:

```
Subscription:
  - Endpoint:
      Fn::GetOptionSetting:
        OptionName: SessionAlarmEmail
        DefaultValue: "nobody@amazon.com"
```

Baris-baris ini yang memberi tahu Elastic Beanstalk untuk mendapatkan nilai untuk properti Endpoint SessionAlarmEmaildari nilai dalam **`options.config`** file konfigurasi (dalam aplikasi contoh kami) yang berisi bagian option_settings dengan bagian aws:elasticbeanstalk:customoption yang berisi pasangan nama-nilai yang berisi nilai aktual yang akan digunakan. Dalam contoh di atas, ini berarti SessionAlarmEmailakan diberi nilai`nobody@amazon.com`.

Untuk informasi selengkapnya tentang CloudFormation sumber daya yang digunakan dalam contoh ini, lihat referensi berikut:

- AWS::DynamoDB::Table
- AWS::CloudWatch::Alarm
- AWS::SNS::Topic

Menggunakan konfigurasi tersimpan Elastic Beanstalk

Anda dapat menyimpan konfigurasi lingkungan Anda sebagai objek di Amazon Simple Storage Service (Amazon S3) yang dapat diterapkan ke lingkungan lain selama pembuatan lingkungan, atau diterapkan ke lingkungan berjalan. Konfigurasi tersimpan adalah templat berformat YAML yang menentukan versi platform, tingkat, pengaturan opsi konfigurasi, dan tag.

Anda dapat menerapkan tag ke konfigurasi tersimpan saat Anda membuatnya, dan mengedit tag konfigurasi tersimpan yang ada. Tag yang diterapkan pada konfigurasi tersimpan tidak terkait

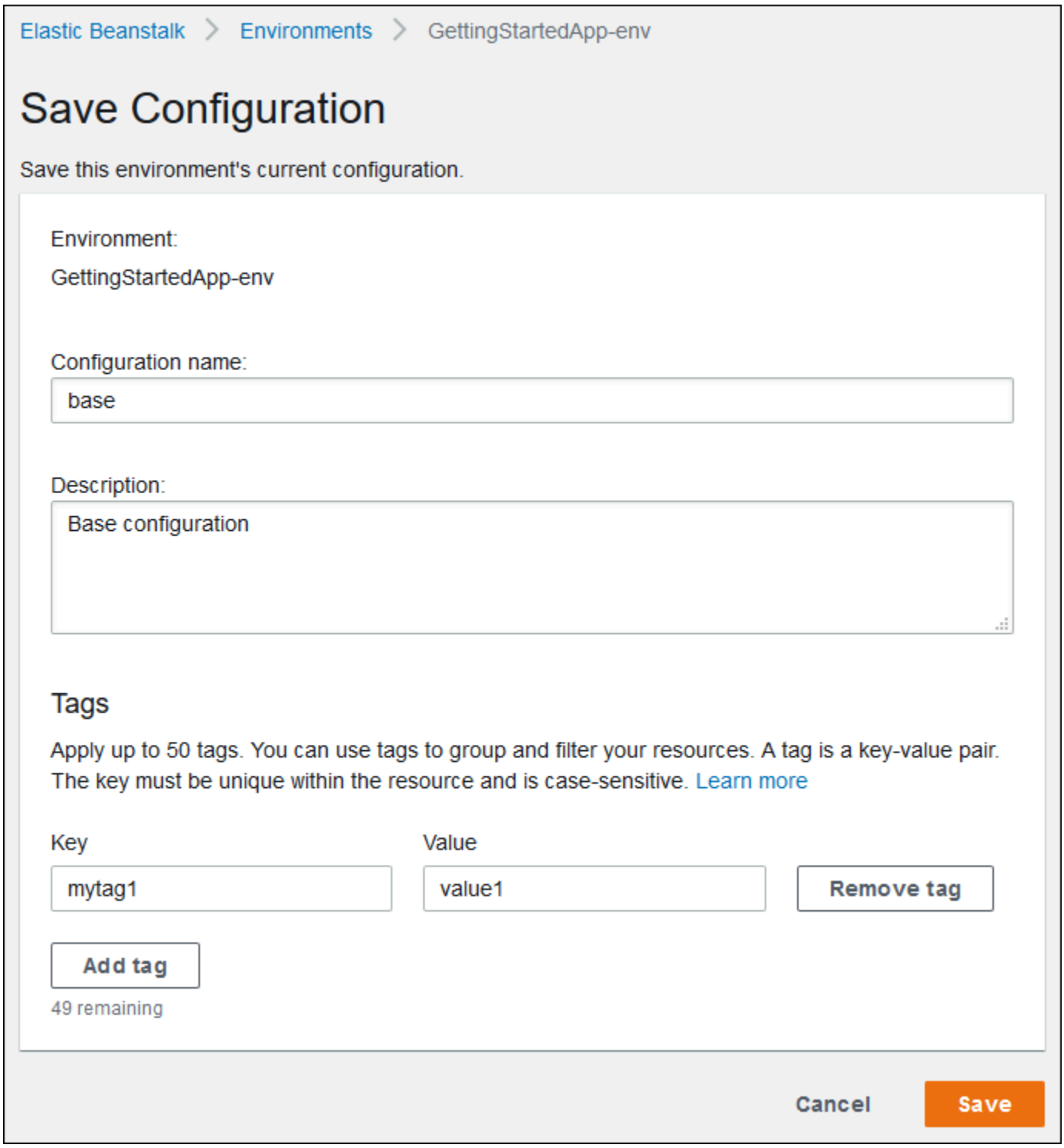
dengan tag yang ditentukan dalam konfigurasi tersimpan menggunakan kunci `Tags:`. Yang terakhir diterapkan ke lingkungan ketika Anda menerapkan konfigurasi terisimpan ke lingkungan. Lihat perinciannya di Menandai konfigurasi tersimpan.

> **Note**
>
> Anda dapat membuat dan menerapkan konfigurasi tersimpan ke lingkungan Elastic Beanstalk Anda menggunakan beberapa metode. Ini termasuk konsol Elastic Beanstalk, EB CLI, dan. AWS CLI
> Lihat topik berikut untuk contoh metode alternatif untuk membuat dan menerapkan konfigurasi tersimpan:
>
> - Menetapkan opsi konfigurasi sebelum pembuatan lingkungan
> - Menetapkan opsi konfigurasi selama pembuatan lingkungan
> - Menetapkan opsi konfigurasi setelah pembuatan lingkungan

Buat konfigurasi terisimpan dari status lingkungan Anda saat ini di konsol manajemen Elastic Beanstalk.

Untuk menyimpan konfigurasi lingkungan

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pilih Tindakan, lalu pilih Simpan konfigurasi.
4. Gunakan formulir di layar untuk memberi nama konfigurasi tersimpan. Opsional, berikan deskripsi singkat, dan tambahkan kunci dan nilai tag.
5. Pilih Simpan.



Konfigurasi tersimpan mencakup pengaturan apa pun yang telah Anda terapkan ke lingkungan dengan konsol tersebut atau klien lain yang menggunakan API Elastic Beanstalk. Anda kemudian dapat menerapkan konfigurasi tersimpan ke lingkungan Anda di kemudian hari untuk memulihkannya ke keadaan sebelumnya, atau menerapkannya ke lingkungan baru selama pembuatan lingkungan.

Anda dapat mengunduh konfigurasi menggunakan perintah EB the section called “eb config” CLI, seperti yang ditunjukkan pada contoh berikut. *NAME*adalah nama konfigurasi Anda yang disimpan.

```
eb config get NAME
```

Untuk menerapkan konfigurasi tersimpan selama pembuatan lingkungan (konsol Elastic Beanstalk)

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Pada panel navigasi, pilih Aplikasi, dan kemudian pilih nama aplikasi Anda dari daftar.
3. Pada panel navigasi, cari nama aplikasi dan pilih Konfigurasi tersimpan.
4. Pilih konfigurasi tersimpan yang ingin Anda terapkan, lalu pilih Luncurkan lingkungan.
5. Lanjutkan melalui wizard untuk membuat lingkungan Anda.

Konfigurasi tersimpan tidak menyertakan pengaturan yang diterapkan dengan file konfigurasi pada kode sumber aplikasi Anda. Jika pengaturan yang sama diterapkan di file konfigurasi dan konfigurasi tersimpan, pengaturan dalam konfigurasi tersimpan akan diutamakan. Demikian juga, opsi yang ditentukan pada konsol Elastic Beanstalk menimpa opsi dalam konfigurasi tersimpan. Untuk informasi selengkapnya, lihat Precedence.

Konfigurasi tersimpan yang disimpan di bucket Elastic Beanstalk S3 dalam folder yang dinamai berdasarkan aplikasi Anda. Misalnya, konfigurasi untuk aplikasi bernama `my-app` di wilayah us-west-2 untuk nomor akun 123456789012 dapat ditemukan di `s3://elasticbeanstalk-us-west-2-123456789012/resources/templates/my-app/`.

Lihat konten konfigurasi tersimpan dengan membukanya di editor teks. Contoh konfigurasi berikut menunjukkan konfigurasi lingkungan server web yang diluncurkan dengan konsol manajemen Elastic Beanstalk.

```
EnvironmentConfigurationMetadata:
  Description: Saved configuration from a multicontainer Docker environment created
 with the Elastic Beanstalk Management Console
  DateCreated: '1520633151000'
  DateModified: '1520633151000'
Platform:
  PlatformArn: arn:aws:elasticbeanstalk:us-east-2::platform/Java 8 running on 64bit
 Amazon Linux/2.5.0
OptionSettings:
```

```
  aws:elasticbeanstalk:command:
    BatchSize: '30'
    BatchSizeType: Percentage
  aws:elasticbeanstalk:sns:topics:
    Notification Endpoint: me@example.com
  aws:elb:policies:
    ConnectionDrainingEnabled: true
    ConnectionDrainingTimeout: '20'
  aws:elb:loadbalancer:
    CrossZone: true
  aws:elasticbeanstalk:environment:
    ServiceRole: aws-elasticbeanstalk-service-role
  aws:elasticbeanstalk:application:
    Application Healthcheck URL: /
  aws:elasticbeanstalk:healthreporting:system:
    SystemType: enhanced
  aws:autoscaling:launchconfiguration:
    IamInstanceProfile: aws-elasticbeanstalk-ec2-role
    InstanceType: t2.micro
    EC2KeyName: workstation-uswest2
  aws:autoscaling:updatepolicy:rollingupdate:
    RollingUpdateType: Health
    RollingUpdateEnabled: true
EnvironmentTier:
  Type: Standard
  Name: WebServer
AWSConfigurationTemplateVersion: 1.1.0.0
Tags:
  Cost Center: WebApp Dev
```

Anda dapat mengubah konten dari konfigurasi tersimpan dan menyimpannya di lokasi yang sama di Amazon S3. Konfigurasi tersimpan yang diformat dengan tepat dan disimpan dengan benar dapat diterapkan ke lingkungan dengan menggunakan konsol manajemen Elastic Beanstalk.

Kunci berikut didukung.

- AWSConfigurationTemplateVersion(wajib) - Versi template konfigurasi (1.1.0.0).

  ```
  AWSConfigurationTemplateVersion: 1.1.0.0
  ```

- Platform — Amazon Resource Name (ARN) dari versi platform lingkungan. Anda dapat menentukan platform dengan ARN atau solusi nama tumpukan.

```
Platform:
  PlatformArn: arn:aws:elasticbeanstalk:us-east-2::platform/Java 8 running on 64bit
 Amazon Linux/2.5.0
```

- SolutionStack— Nama lengkap dari tumpukan solusi yang digunakan untuk menciptakan lingkungan.

```
SolutionStack: 64bit Amazon Linux 2017.03 v2.5.0 running Java 8
```

- OptionSettings— Pengaturan opsi konfigurasi untuk diterapkan ke lingkungan. Sebagai contoh, entri berikut menetapkan jenis instans untuk t2.micro.

```
OptionSettings:
  aws:autoscaling:launchconfiguration:
    InstanceType: t2.micro
```

- Tag — Hingga 47 tag untuk diterapkan pada sumber daya yang dibuat dalam lingkungan.

```
Tags:
  Cost Center: WebApp Dev
```

- EnvironmentTier— Jenis lingkungan yang harus dibuat. Untuk lingkungan server web, Anda dapat mengecualikan bagian ini (server web adalah default). Untuk lingkungan pekerja, gunakan berikut ini.

```
EnvironmentTier:
  Name: Worker
  Type: SQS/HTTP
```

> **Note**
>
> Anda dapat membuat dan menerapkan konfigurasi tersimpan ke lingkungan Elastic Beanstalk Anda menggunakan beberapa metode. Ini termasuk konsol Elastic Beanstalk, EB CLI, dan. AWS CLI
> Lihat topik berikut untuk contoh metode alternatif untuk membuat dan menerapkan konfigurasi tersimpan:
>
> - Menetapkan opsi konfigurasi sebelum pembuatan lingkungan

- Menetapkan opsi konfigurasi selama pembuatan lingkungan
- Menetapkan opsi konfigurasi setelah pembuatan lingkungan

Menandai konfigurasi tersimpan

Anda dapat menerapkan tag ke konfigurasi yang AWS Elastic Beanstalk disimpan. Tag adalah pasangan nilai kunci yang terkait dengan AWS sumber daya. Untuk informasi tentang penandaan sumber daya Elastic Beanstalk, kasus penggunaan, kunci tag dan batasan nilai, dan jenis sumber daya yang didukung, lihat Pelabelan sumber daya aplikasi Elastic Beanstalk.

Anda dapat menentukan tag saat membuat konfigurasi tersimpan. Pada konfigurasi tersimpan yang ada, Anda dapat menambahkan atau menghapus tag, dan memperbarui nilai tag yang ada. Anda dapat menambahkan hingga 50 tag ke setiap konfigurasi tersimpan.

Menambahkan tag selama pembuatan konfigurasi tersimpan

Saat Anda menggunakan konsol Elastic Beanstalk untuk menyimpan konfigurasi, Anda dapat menentukan kunci dan nilai tag di halaman Simpan Konfigurasi.

Jika Anda menggunakan EB CLI untuk menyimpan konfigurasi, gunakan opsi `--tags` dengan eb config untuk menambahkan tag.

```
~/workspace/my-app$ eb config --tags mytag1=value1,mytag2=value2
```

Dengan AWS CLI atau klien berbasis API lainnya, tambahkan tag dengan menggunakan `--tags` parameter pada perintah. create-configuration-template

```
$ aws elasticbeanstalk create-configuration-template \
      --tags Key=mytag1,Value=value1 Key=mytag2,Value=value2 \
      --application-name my-app --template-name my-template --solution-stack-
name solution-stack
```

Mengelola tag dari konfigurasi tersimpan yang ada

Anda dapat menambahkan, memperbarui, dan menghapus tag dalam konfigurasi tersimpan Elastic Beanstalk yang ada.

Mengelola tag konfigurasi tersimpan menggunakan konsol Elastic Beanstalk

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Pada panel navigasi, pilih Aplikasi, dan kemudian pilih nama aplikasi Anda dari daftar.
3. Pada panel navigasi, cari nama aplikasi dan pilih Konfigurasi tersimpan.
4. Pilih konfigurasi tersimpan yang ingin Anda kelola.
5. Pilih Tindakan, lalu pilih Kelola tag.
6. Gunakan formulir di layar untuk menambahkan, memperbarui, atau menghapus tag.
7. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Jika Anda menggunakan EB CLI untuk memperbarui konfigurasi tersimpan Anda, gunakan eb tags untuk menambahkan, memperbarui, menghapus, atau mencantumkan tag.

Misalnya, perintah berikut mencantumkan tag dalam konfigurasi tersimpan.

```
~/workspace/my-app$ eb tags --list --resource "arn:aws:elasticbeanstalk:us-east-2:my-account-id:configurationtemplate/my-app/my-template"
```

Perintah berikut memperbarui tag `mytag1` dan menghapus tag `mytag2`.

```
~/workspace/my-app$ eb tags --update mytag1=newvalue --delete mytag2 \
      --resource "arn:aws:elasticbeanstalk:us-east-2:my-account-id:configurationtemplate/my-app/my-template"
```

Untuk daftar lengkap opsi dan contoh lainnya, lihat `eb tags`.

Dengan AWS CLI atau klien berbasis API lainnya, gunakan list-tags-for-resource perintah untuk membuat daftar tag konfigurasi yang disimpan.

```
$ aws elasticbeanstalk list-tags-for-resource --resource-arn
 "arn:aws:elasticbeanstalk:us-east-2:my-account-id:configurationtemplate/my-app/my-template"
```

Gunakan perintah update-tags-for-resource untuk menambahkan, memperbarui, atau menghapus tag dalam konfigurasi tersimpan.

```
$ aws elasticbeanstalk update-tags-for-resource \
```

```
      --tags-to-add Key=mytag1,Value=newvalue --tags-to-remove mytag2 \
      --resource-arn "arn:aws:elasticbeanstalk:us-east-2:my-account-
id:configurationtemplate/my-app/my-template"
```

Tentukan kedua tag untuk ditambahkan dan tag untuk diperbarui dalam parameter `--tags-to-add` dari update-tags-for-resource. Tag yang tidak ada ditambahkan, dan nilai tag yang ada diperbarui.

> **Note**
>
> Untuk menggunakan beberapa CLI EB AWS CLI dan perintah dengan konfigurasi tersimpan Elastic Beanstalk, Anda memerlukan ARN konfigurasi yang disimpan. Untuk membangun ARN, pertama gunakan perintah berikut untuk mengambil nama konfigurasi tersimpan tersebut.
>
> ```
> $ aws elasticbeanstalk describe-applications --application-names my-app
> ```
>
> Cari kunci `ConfigurationTemplates` di output perintah. Elemen ini menunjukkan nama konfigurasi tersimpan tersebut. Gunakan nama ini di mana *`my-template`* ditentukan dalam perintah yang disebutkan di halaman ini.

Manifes lingkungan (`env.yaml`)

Anda dapat menyertakan manifes lingkungan berformat YAML di akar paket sumber aplikasi Anda untuk mengonfigurasi nama lingkungan, tumpukan solusi dan tautan lingkungan untuk digunakan saat membuat lingkungan Anda.

Format file ini mencakup dukungan untuk grup lingkungan. Untuk menggunakan grup, tentukan nama lingkungan dalam manifes dengan simbol + di bagian akhir. Saat Anda membuat atau memperbarui lingkungan, tentukan nama grup dengan `--group-name` (AWS CLI) atau `--env-group-suffix` (EB CLI). Untuk informasi selengkapnya tentang grup, lihat Membuat dan memperbarui grup lingkungan Elastic Beanstalk.

Manifes contoh berikut mendefinisikan lingkungan server web dengan tautan ke komponen lingkungan pekerja yang bergantung padanya. Manifes tersebut menggunakan grup untuk mengizinkan pembuatan beberapa lingkungan dengan paket sumber yang sama:

`~/myapp/frontend/env.yaml`

```
AWSConfigurationTemplateVersion: 1.1.0.0
SolutionStack: 64bit Amazon Linux 2015.09 v2.0.6 running Multi-container Docker 1.7.1
 (Generic)
OptionSettings:
  aws:elasticbeanstalk:command:
    BatchSize: '30'
    BatchSizeType: Percentage
  aws:elasticbeanstalk:sns:topics:
    Notification Endpoint: me@example.com
  aws:elb:policies:
    ConnectionDrainingEnabled: true
    ConnectionDrainingTimeout: '20'
  aws:elb:loadbalancer:
    CrossZone: true
  aws:elasticbeanstalk:environment:
    ServiceRole: aws-elasticbeanstalk-service-role
  aws:elasticbeanstalk:application:
    Application Healthcheck URL: /
  aws:elasticbeanstalk:healthreporting:system:
    SystemType: enhanced
  aws:autoscaling:launchconfiguration:
    IamInstanceProfile: aws-elasticbeanstalk-ec2-role
    InstanceType: t2.micro
    EC2KeyName: workstation-uswest2
  aws:autoscaling:updatepolicy:rollingupdate:
    RollingUpdateType: Health
    RollingUpdateEnabled: true
Tags:
  Cost Center: WebApp Dev
CName: front-A08G28LG+
EnvironmentName: front+
EnvironmentLinks:
  "WORKERQUEUE" : "worker+"
```

Kunci berikut didukung.

- AWSConfigurationTemplateVersion(wajib) - Versi template konfigurasi (1.1.0.0).

  ```
  AWSConfigurationTemplateVersion: 1.1.0.0
  ```

- Platform — Amazon Resource Name (ARN) dari versi platform lingkungan. Anda dapat menentukan platform dengan ARN atau solusi nama tumpukan.

```
Platform:
  PlatformArn: arn:aws:elasticbeanstalk:us-east-2::platform/Java 8 running on 64bit
 Amazon Linux/2.5.0
```

- SolutionStack— Nama lengkap dari tumpukan solusi yang digunakan untuk menciptakan lingkungan.

```
SolutionStack: 64bit Amazon Linux 2017.03 v2.5.0 running Java 8
```

- OptionSettings— Pengaturan opsi konfigurasi untuk diterapkan ke lingkungan. Sebagai contoh, entri berikut menetapkan jenis instans untuk t2.micro.

```
OptionSettings:
  aws:autoscaling:launchconfiguration:
    InstanceType: t2.micro
```

- Tag — Hingga 47 tag untuk diterapkan pada sumber daya yang dibuat dalam lingkungan.

```
Tags:
  Cost Center: WebApp Dev
```

- EnvironmentTier— Jenis lingkungan yang harus dibuat. Untuk lingkungan server web, Anda dapat mengecualikan bagian ini (server web adalah default). Untuk lingkungan pekerja, gunakan berikut ini.

```
EnvironmentTier:
  Name: Worker
  Type: SQS/HTTP
```

- CName— CNAME untuk lingkungan. Sertakan karakter + di akhir nama untuk mengaktifkan grup.

```
CName: front-A08G28LG+
```

- EnvironmentName— Nama lingkungan yang akan dibuat. Sertakan karakter + di akhir nama untuk mengaktifkan grup.

```
EnvironmentName: front+
```

  Dengan grup yang diaktifkan, Anda harus menentukan nama grup ketika Anda membuat lingkungan. Elastic Beanstalk menambahkan nama grup untuk nama lingkungan dengan tanda

hubung. Sebagai contoh, dengan nama lingkungan `front+` dan nama grup `dev`, Elastic Beanstalk akan membuat lingkungan dengan nama tersebut `front-dev`.

- EnvironmentLinks— Peta nama variabel dan nama lingkungan dependensi. Contoh berikut membuat lingkungan `worker+` menjadi ketergantungan dan memberitahu Elastic Beanstalk untuk menyimpan informasi tautan ke variabel bernama `WORKERQUEUE`.

  ```
  EnvironmentLinks:
    "WORKERQUEUE" : "worker+"
  ```

  Nilai variabel tautan bervariasi tergantung pada jenis lingkungan yang ditautkan. Untuk lingkungan server web, tautannya adalah CNAME lingkungan. Untuk lingkungan pekerja, tautannya adalah nama antrean Amazon Simple Queue Service (Amazon SQS) lingkungan.

EnvironmentLinksKunci CName, EnvironmentNamedan dapat digunakan untuk membuat grup lingkungan dan tautan ke lingkungan lain. Fitur-fitur ini saat ini didukung saat menggunakan EB CLI AWS CLI , atau SDK.

Menggunakan image mesin Amazon kustom (AMI) di lingkungan Elastic Beanstalk Anda

Bagian ini menjelaskan kapan harus mempertimbangkan untuk menggunakan AMI kustom dan menyediakan prosedur untuk mengonfigurasi dan mengelola AMI kustom di lingkungan Anda. Saat membuat AWS Elastic Beanstalk lingkungan, Anda dapat menentukan Amazon Machine Image (AMI) untuk digunakan, bukan AMI Elastic Beanstalk standar yang disertakan dalam versi platform Anda. AMI khusus dapat meningkatkan waktu penyediaan saat instance diluncurkan di lingkungan Anda jika Anda perlu menginstal banyak perangkat lunak yang tidak disertakan dalam standar. AMIs

Penggunaan file konfigurasi efektif untuk menyesuaikan lingkungan Anda dengan cepat dan konsisten. Meskipun menerapkan konfigurasi dapat mulai memakan waktu lama selama pembuatan dan pembaruan lingkungan. Jika Anda melakukan banyak konfigurasi server dalam file konfigurasi, Anda dapat mengurangi waktunya dengan cara membuat AMI kustom yang sudah memiliki perangkat lunak dan konfigurasi yang dibutuhkan.

AMI kustom juga mengizinkan Anda untuk membuat perubahan pada komponen tingkat rendah, seperti kernel Linux, yang sulit diterapkan atau perlu waktu lama untuk menerapkannya dalam file konfigurasi. Untuk membuat AMI khusus, luncurkan platform Elastic Beanstalk AMI EC2 di Amazon,

sesuaikan perangkat lunak dan konfigurasi sesuai kebutuhan Anda, lalu hentikan instance dan simpan AMI darinya.

Membuat AMI kustom

Anda dapat menggunakan EC2 Image Builder untuk membuat dan mengelola kustom AMIs sebagai alternatif dari prosedur ini. Untuk informasi selengkapnya, lihat Panduan Pengguna Image Builder.

Untuk mengidentifikasi dasar AMI Elastic Beanstalk

1. Dalam jendela perintah, jalankan perintah seperti berikut. Untuk informasi selengkapnya, lihat describe-platform-version dalam AWS CLI Referensi Perintah.

   Tentukan AWS Wilayah tempat Anda ingin menggunakan AMI kustom Anda, dan ganti ARN platform dan nomor versi dengan platform Elastic Beanstalk yang menjadi dasar aplikasi Anda.

   Example - Mac OS / Linux OS

```
$ aws elasticbeanstalk describe-platform-version --region us-east-2 \
      --platform-arn "arn:aws:elasticbeanstalk:us-east-2::platform/Node.js 20
 running on 64bit Amazon Linux 2023/6.1.7" \
      --query PlatformDescription.CustomAmiList
[
    {
        "VirtualizationType": "pv",
        "ImageId": ""
    },
    {
        "VirtualizationType": "hvm",
        "ImageId": "ami-020ae06fdda6a0f66"
    }
]
```

   Example - Windows OS

```
C:\> aws elasticbeanstalk describe-platform-version --region us-east-2 --platform-
arn"arn:aws:elasticbeanstalk:us-east-2::platform/
IIS 10.0 running on 64bit Windows Server 2022/2.15.3" --query
 PlatformDescription.CustomAmiList
[
    {
```

```
        "VirtualizationType": "pv",
        "ImageId": ""
    },
    {
        "VirtualizationType": "hvm",
        "ImageId": "ami-020ae06fdda6a0f66"
    }
]
```

2. Perhatikan nilai `ImageId` yang terlihat seperti `ami-020ae06fdda6a0f66` sebagai hasilnya.

Nilainya adalah stok Elastic Beanstalk AMI untuk EC2 versi platform, arsitektur instans AWS , dan Wilayah yang relevan untuk aplikasi Anda. Jika Anda perlu membuat AMIs untuk beberapa platform, arsitektur, atau AWS Wilayah, ulangi proses ini untuk mengidentifikasi AMI dasar yang benar untuk setiap kombinasi.

> **Note**
>
> Jangan membuat AMI dari suatu instans yang telah diluncurkan di lingkungan Elastic Beanstalk. Elastic Beanstalk membuat perubahan pada instans selama penyediaan yang dapat menyebabkan masalah pada AMI yang disimpan. Menyimpan citra dari instans di lingkungan Elastic Beanstalk juga akan membuat versi aplikasi Anda yang di-deploy ke instans menjadi bagian tetap pada citra.

Untuk Linux, juga memungkinkan untuk membuat AMI kustom dari AMI komunitas yang tidak publikasikan oleh Elastic Beanstalk. Anda dapat menggunakan AMI Amazon Linux terbaru sebagai titik untuk awal. Ketika Anda meluncurkan lingkungan dengan AMI Linux yang tidak dikelola oleh Elastic Beanstalk, Elastic Beanstalk mencoba untuk menginstal perangkat lunak platform (bahasa, kerangka kerja, server proksi, dll.) dan komponen tambahan untuk mendukung fitur seperti Pelaporan Kondisi yang Ditingkatkan.

> **Note**
>
> Kustom AMIs berdasarkan Windows Server memerlukan stok Elastic Beanstalk AMI `describe-platform-version` dikembalikan dari, seperti yang ditunjukkan sebelumnya pada Langkah 1.

Meskipun Elastic Beanstalk dapat menggunakan AMI yang tidak dikelola oleh Elastic Beanstalk, peningkatan waktu penyediaan yang dihasilkan dari komponen yang hilang saat penginstalan Elastic Beanstalk dapat mengurangi atau menghilangkan manfaat pembuatan AMI kustom di awal. Distribusi Linux lainnya mungkin bekerja dengan beberapa pemecahan masalah tetapi tidak didukung secara resmi. Jika aplikasi Anda memerlukan distribusi Linux khusus, satu alternatifnya adalah dengan membuat citra Docker dan menjalankannya pada platform Docker Elastic Beanstalk atau platform Docker Multikontainer.

Untuk membuat AMI kustom

1. Buka EC2 konsol Amazon di https://console.aws.amazon.com/ec2/.
2. Pilih Luncurkan Instans.
3. Jika Anda menentukan dasar AMI Elastic Beanstalk (menggunakan `describe-platform-version`) atau AMI Amazon Linux, masukkan ID AMI di kotak pencarian. Lalu tekan Enter.

   Anda juga dapat mencari daftar untuk AMI komunitas lain yang sesuai dengan kebutuhan Anda.

   > **Note**
   >
   > Kami merekomendasikan Anda untuk memilih AMI yang menggunakan virtualisasi HVM. Ini AMIs menunjukkan jenis Virtualisasi: hvm dalam deskripsi mereka.
   > Untuk informasi selengkapnya, lihat Jenis virtualisasi di Panduan EC2 Pengguna Amazon.

4. Memilih Pilihan untuk memilih AMI.
5. Pilih tipe instans, lalu memilih Berikutnya: Konfigurasikan Detail Instans.
6. (Untuk platform AMI (AL1) Amazon Linux yang sudah pensiun) Lewati langkah ini jika lingkungan Anda berjalan pada platform berbasis Linux yang didukung atau pada platform Windows.

   Perluas bagian Detail Lanjutan dan tempel teks berikut di bidang Data Pengguna.

   ```
   #cloud-config
     repo_releasever: repository version number
     repo_upgrade: none
   ```

   Nomor versi repositori adalah versi tahun dan bulan dalam nama AMI. Misalnya, AMIs berdasarkan rilis Maret 2015 Amazon Linux memiliki nomor versi repositori. `2015.03` Untuk citra Elastic Beanstalk, nomor tersebut cocok dengan tanggal yang ditunjukkan dalam nama

tumpukan solusi untuk versi platform berdasarkan Amazon Linux AMI (Amazon Linux 2 yang terdahulu).

> **Note**
>
> `repo_releasever`Pengaturan mengonfigurasi lock-on-launch fitur untuk AMI Amazon Linux. Hal ini menyebabkan AMI menggunakan versi repositori tetap dan spesifik saat diluncurkan. Fitur ini tidak didukung di Amazon Linux 2—jangan tentukan jika lingkungan Anda menggunakan cabang platform Amazon Linux 2 saat ini. Pengaturan ini diperlukan jika Anda menggunakan AMI kustom dengan Elastic Beanstalk hanya pada cabang platform Amazon Linux AMI (Amazon Linux 2 yang terdahulu).
> Pengaturan `repo_upgrade` menonaktifkan instalasi otomatis pembaruan keamanan. Hal ini diperlukan untuk menggunakan AMI kustom dengan Elastic Beanstalk.

7. Lanjutkan melalui wizard untuk meluncurkan EC2 instance. Saat diminta, pilih pasangan kunci yang dapat Anda akses sehingga Anda dapat terhubung ke instans untuk langkah selanjutnya.
8. Connect ke instans dengan SSH atau RDP.
9. Lakukan penyesuaian apa pun yang Anda inginkan.
10. (Platform Windows) Jalankan layanan EC2 Config Sysprep. Untuk informasi tentang EC2 Config, lihat Mengonfigurasi Instans Windows Menggunakan Layanan Config EC2. Pastikan bahwa Sysprep dikonfigurasi untuk menghasilkan kata sandi acak yang dapat diambil kembali dari AWS Management Console.
11. Di EC2 konsol Amazon, hentikan EC2 instance. Kemudian pada menu Tindakan Instans, pilih Buat Citra (EBS AMI).
12. Untuk menghindari AWS biaya tambahan, hentikan instance. EC2

Untuk menggunakan AMI kustom Anda dalam lingkungan Elastic Beanstalk

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Pada kategori konfigurasi Kapasitas, pilih Edit.
5. Untuk ID AMI, masukkan ID AMI kustom Anda.
6. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Ketika Anda membuat lingkungan baru dengan AMI kustom, Anda harus menggunakan versi platform yang sama dengan yang Anda gunakan sebagai dasar untuk membuat AMI.

Mengelola lingkungan dengan AMI khusus

Pembaruan platform

Saat menggunakan AMI kustom, Elastic Beanstalk akan terus menggunakan AMI kustom yang sama di lingkungan saat versi platformnya diperbarui, terlepas dari apakah pembaruan diterapkan secara manual atau melalui pembaruan platform terkelola. Lingkungan tidak akan diatur ulang untuk menggunakan stok AMI versi platform baru.

Kami menyarankan Anda membuat AMI kustom baru berdasarkan stok AMI versi platform baru. Melakukannya akan menerapkan tambalan yang tersedia di versi platform baru dan juga akan meminimalkan kegagalan penerapan karena versi paket atau pustaka yang tidak kompatibel.

Untuk informasi selengkapnya tentang membuat AMI kustom baru, lihat yang Membuat AMI kustom lebih awal dalam topik ini.

Menghapus AMI kustom

Jika Anda ingin menghapus AMI kustom dari lingkungan dan mengatur ulang untuk menggunakan AMI stok untuk versi platform lingkungan, gunakan perintah CLI berikut.

```
aws elasticbeanstalk update-environment \
  --application-name my-application \
  --environment-name my-environment \
  --region us-east-1 \
  --options-to-remove Namespace=aws:autoscaling:launchconfiguration,OptionName=ImageId
```

> **Note**
>
> Untuk menghindari gangguan layanan Anda, uji aplikasi Anda dengan AMI stok sebelum menerapkan perubahan ini ke lingkungan produksi Anda.

Membersihkan AMI kustom

Ketika Anda selesai menggunakan AMI kustom dan tidak memerlukannya untuk meluncurkan lingkungan Elastic Beanstalk lagi, pertimbangkan untuk membersihkannya untuk meminimalkan

biaya penyimpanan. Membersihkan AMI khusus melibatkan membatalkan pendaftarannya dari Amazon EC2 dan menghapus sumber daya terkait lainnya. Untuk detailnya, lihat Pembatalan pendaftaran Linux AMI Anda atau Pembatalan pendaftaran Windows AMI Anda.

Mempertahankan akses ke Amazon Machine Image (AMI) untuk platform yang sudah pensiun

Elastic Beanstalk menetapkan status cabang platform untuk pensiun ketika sistem operasi atau komponen utama yang digunakan oleh cabang mencapai End of Life. Basis Elastic Beanstalk AMI untuk cabang platform juga dapat dibuat pribadi untuk mencegah penggunaan AMI ini. out-of-date Lingkungan AMIs yang menggunakan yang telah dibuat pribadi tidak akan lagi dapat meluncurkan instance.

Jika Anda tidak dapat memigrasikan aplikasi Anda ke lingkungan yang didukung sebelum dihentikan, lingkungan Anda mungkin berada dalam situasi ini. Kebutuhan untuk memperbarui lingkungan untuk cabang platform Beanstalk, di mana dasarnya Elastic Beanstalk AMI telah dibuat pribadi, mungkin timbul. Pendekatan alternatif tersedia. Anda dapat memperbarui lingkungan yang ada berdasarkan salinan dasar Elastic Beanstalk AMI yang digunakan oleh lingkungan Anda.

Topik ini menawarkan beberapa langkah dan skrip mandiri untuk memperbarui lingkungan yang ada berdasarkan salinan dasar Elastic Beanstalk AMI yang digunakan oleh lingkungan Anda. Setelah Anda dapat memigrasikan aplikasi Anda ke platform yang didukung, Anda dapat terus menggunakan prosedur standar untuk memelihara aplikasi dan lingkungan yang didukung.

Langkah-langkah manual

Untuk memperbarui lingkungan berdasarkan salinan AMI dari dasar Elastic Beanstalk AMI

1. Tentukan AMI mana yang digunakan lingkungan Anda. Perintah ini mengembalikan AMI yang digunakan oleh lingkungan Elastic Beanstalk yang Anda berikan dalam parameter. Nilai yang dikembalikan digunakan sebagai source-ami-idpada langkah berikutnya.

   Dalam jendela perintah, jalankan perintah seperti berikut. Untuk informasi selengkapnya, lihat describe-configuration-settings dalam AWS CLI Referensi Perintah.

   Tentukan AWS Wilayah yang menyimpan sumber AMI yang ingin Anda salin. Ganti nama aplikasi dan nama lingkungan dengan yang didasarkan pada sumber AMI. Masukkan teks untuk parameter kueri seperti yang ditunjukkan.

Example

```
>aws elasticbeanstalk describe-configuration-settings \
  --application-name my-application \
  --environment-name my-environment \
  --region us-east-2 \
  --query "ConfigurationSettings[0].OptionSettings[?OptionName=='ImageId'] |
 [0].Value"
```

2. Salin AMI ke akun Anda. Perintah ini mengembalikan AMI baru yang dihasilkan dari menyalin source-ami-idyang dikembalikan pada langkah sebelumnya.

> **Note**
>
> Pastikan untuk membuat catatan id AMI baru yang dihasilkan oleh perintah ini. Anda harus memasukkannya di langkah berikutnya, mengganti copied-ami-idperintah contoh.

Dalam jendela perintah, jalankan perintah seperti berikut. Untuk informasi selengkapnya, lihat copy-image di AWS CLI Command Reference.

Tentukan AWS Wilayah AMI sumber yang ingin Anda salin (--source-region) dan Wilayah tempat Anda ingin menggunakan AMI kustom baru Anda (--region). Ganti source-ami-id dengan AMI gambar yang Anda salin. source-ami-idItu dikembalikan oleh perintah pada langkah sebelumnya. Ganti new-ami-namedengan nama untuk menggambarkan AMI baru di Wilayah tujuan. Skrip yang mengikuti prosedur ini menghasilkan nama AMI baru dengan menambahkan string "Salinan" ke awal namasource-ami-id.

```
>aws ec2 copy-image \
    --region us-east-2 \
    --source-image-id source-ami-id \
    --source-region us-east-2 \
    --name new-ami-name
```

3. Perbarui lingkungan untuk menggunakan AMI yang disalin. Setelah perintah berjalan, ia mengembalikan status lingkungan.

Dalam jendela perintah, jalankan perintah seperti berikut. Untuk informasi selengkapnya, lihat update-environment di Command Reference.AWS CLI

Tentukan AWS Wilayah lingkungan dan aplikasi yang perlu Anda perbarui. Ganti nama aplikasi dan nama lingkungan dengan yang perlu Anda kaitkan dengan copied-ami-iddari langkah sebelumnya. Untuk parameter --option-setttings, ganti dengan id AMI *copied-ami-id* yang Anda catat dari output dari perintah sebelumnya.

```
>aws elasticbeanstalk update-environment \
  --application-name my-application \
  --environment-name my-environment \
  --region us-east-2 \
  --option-settings
 "Namespace=aws:autoscaling:launchconfiguration,OptionName=ImageId,Value=copied-
ami-id"
```

> **Note**
>
> Untuk meminimalkan biaya penyimpanan, pertimbangkan untuk membersihkan AMI khusus Anda saat Anda tidak membutuhkannya untuk meluncurkan lingkungan Elastic Beanstalk lagi. Untuk informasi selengkapnya, lihat Membersihkan AMI kustom.

Skrip mandiri

Skrip berikut memberikan hasil yang sama dengan langkah manual sebelumnya. Unduh skrip dengan memilih tautan ini: copy_ami_and_update_env.zip.

Sumber skrip: copy_ami_and_update_env.sh

```
#!/bin/bash

set -ue

USAGE="This script is used to copy an AMI used by your Elastic Beanstalk environment
 into your account to use in your environment.\n\n"
USAGE+="Usage:\n\n"
USAGE+="./$(basename $0) [OPTIONS]\n"
USAGE+="OPTIONS:\n"
```

```
USAGE+="\t--application-name <application-name>\tThe name of your Elastic Beanstalk
 application.\n"
USAGE+="\t--environment-name <environment-name>\tThe name of your Elastic Beanstalk
 environment.\n"
USAGE+="\t--region <region> \t\t\tThe AWS region your Elastic Beanstalk environment is
 deployed to.\n"
USAGE+="\n\n"
USAGE+="Script Usage Example(s):\n"
USAGE+="./$(basename $0) --application-name my-application --environment-name my-
environment --region us-east-1\n"

if [ $# -eq 0 ]; then
  echo -e $USAGE
  exit
fi

while [[ $# -gt 0 ]]; do
  case $1 in
    --application-name)       APPLICATION_NAME="$2"; shift ;;
    --environment-name)       ENVIRONMENT_NAME="$2"; shift ;;
    --region)                 REGION="$2"; shift ;;
    *)                        echo "Unknown option $1" ; echo -e $USAGE ; exit ;;
  esac
  shift
done

aws_cli_version="$(aws --version)"
if [ $? -ne 0 ]; then
  echo "aws CLI not found. Please install it: https://docs.aws.amazon.com/cli/latest/
userguide/getting-started-install.html. Exiting."
  exit 1
fi
echo "Using aws CLI version: ${aws_cli_version}"

account=$(aws sts get-caller-identity --query "Account" --output text)
echo "Using account ${account}"

environment_ami_id=$(aws elasticbeanstalk describe-configuration-settings \
  --application-name "$APPLICATION_NAME" \
  --environment-name "$ENVIRONMENT_NAME" \
  --region "$REGION" \
  --query "ConfigurationSettings[0].OptionSettings[?OptionName=='ImageId'] | [0].Value"
 \
  --output text)
```

```
echo "Image associated with environment ${ENVIRONMENT_NAME} is ${environment_ami_id}"

owned_image=$(aws ec2 describe-images \
  --owners self \
  --image-ids "$environment_ami_id" \
  --region "$REGION" \
  --query "Images[0]" \
  --output text)
if [ "$owned_image" != "None" ]; then
  echo "${environment_ami_id} is already owned by account ${account}. Exiting."
  exit
fi

source_image_name=$(aws ec2 describe-images \
  --image-ids "$environment_ami_id" \
  --region "$REGION" \
  --query "Images[0].Name" \
  --output text)
if [ "$source_image_name" = "None" ]; then
  echo "Cannot find ${environment_ami_id}. Please contact AWS support if you need
 additional help: https://aws.amazon.com/support."
  exit 1
fi

copied_image_name="Copy of ${source_image_name}"
copied_ami_id=$(aws ec2 describe-images \
  --owners self \
  --filters Name=name,Values="${copied_image_name}" \
  --region "$REGION" \
  --query "Images[0].ImageId" \
  --output text)
if [ "$copied_ami_id" != "None" ]; then
  echo "Detected that ${environment_ami_id} has already been copied by account
 ${account}. Skipping image copy."
else
  echo "Copying ${environment_ami_id} to account ${account} with name
 ${copied_image_name}"
  copied_ami_id=$(aws ec2 copy-image \
    --source-image-id "$environment_ami_id" \
    --source-region "$REGION" \
    --name "$copied_image_name" \
    --region "$REGION" \
    --query "ImageId" \
    --output text)
```

```
  echo "New AMI ID is ${copied_ami_id}"

  echo "Waiting for ${copied_ami_id} to become available"
  aws ec2 wait image-available \
    --image-ids "$copied_ami_id" \
    --region "$REGION"
  echo "${copied_ami_id} is now available"
fi

echo "Updating environment ${ENVIRONMENT_NAME} to use ${copied_ami_id}"
environment_status=$(aws elasticbeanstalk update-environment \
  --application-name "$APPLICATION_NAME" \
  --environment-name "$ENVIRONMENT_NAME" \
  --option-settings
 "Namespace=aws:autoscaling:launchconfiguration,OptionName=ImageId,Value=
${copied_ami_id}" \
  --region "$REGION" \
  --query "Status" \
  --output text)
echo "Environment ${ENVIRONMENT_NAME} is now ${environment_status}"

echo "Waiting for environment ${ENVIRONMENT_NAME} update to complete"
aws elasticbeanstalk wait environment-updated \
  --application-name "$APPLICATION_NAME" \
  --environment-names "$ENVIRONMENT_NAME" \
  --region "$REGION"
echo "Environment ${ENVIRONMENT_NAME} update complete"
```

> **Note**
>
> Anda harus AWS CLI menginstal untuk menjalankan skrip. Untuk petunjuk penginstalan, lihat Menginstal atau memperbarui versi terbaru dari Panduan AWS Command Line Interface Pengguna. AWS CLI
> Setelah menginstal AWS CLI, Anda juga harus mengkonfigurasinya untuk menggunakan AWS akun yang memiliki lingkungan. Untuk informasi selengkapnya, lihat Mengkonfigurasi AWS CLI dalam Panduan AWS Command Line Interface Pengguna. Akun juga harus memiliki izin untuk membuat AMI dan memperbarui lingkungan Elastic Beanstalk.

Langkah-langkah ini menggambarkan proses yang diikuti skrip.

1. Cetak akun yang sedang digunakan.
2. Tentukan AMI mana yang digunakan oleh lingkungan (sumber AMI).
3. Periksa apakah sumber AMI sudah dimiliki oleh akun. Jika ya, keluarlah.
4. Tentukan nama sumber AMI sehingga dapat digunakan dalam nama AMI baru. Ini juga berfungsi untuk mengkonfirmasi akses ke sumber AMI.
5. Periksa apakah sumber AMI telah disalin ke akun. Ini dilakukan dengan mencari AMIs dengan nama AMI yang disalin yang dimiliki oleh akun. Jika nama AMI telah diubah di antara eksekusi skrip, itu akan menyalin gambar lagi.
6. Jika sumber AMI belum disalin, salin sumber AMI ke akun dan tunggu AMI baru tersedia.
7. Perbarui konfigurasi lingkungan untuk menggunakan AMI baru.
8. Tunggu hingga pembaruan lingkungan selesai.

Setelah Anda mengekstrak skrip dari file copy_ami_and_update_env.zip, jalankan dengan mengeksekusi contoh berikut. Ganti nama aplikasi dan nama lingkungan dalam contoh dengan nilai Anda sendiri.

```
>sh copy_ami_and_update_env.sh \
  --application-name my-application \
  --environment-name my-environment \
  --region us-east-1
```

> **Note**
>
> Untuk meminimalkan biaya penyimpanan, pertimbangkan untuk membersihkan AMI khusus Anda saat Anda tidak membutuhkannya untuk meluncurkan lingkungan Elastic Beanstalk lagi. Untuk informasi selengkapnya, lihat Membersihkan AMI kustom.

Menyajikan file statis

Untuk meningkatkan performa, Anda dapat mengonfigurasi server proksi untuk menyajikan file statis (misalnya, HTML atau gambar) dari satu set direktori di dalam aplikasi web Anda. Ketika server proksi menerima permintaan untuk file di bawah jalur yang ditentukan, server menyajikan file langsung daripada merutekan permintaan ke aplikasi Anda.

Elastic Beanstalk mendukung konfigurasi proksi untuk menyajikan file statis pada sebagian besar cabang platform berdasarkan Amazon Linux 2. Satu-satunya pengecualian adalah Docker.

> **Note**
>
> Pada platform Python dan Ruby, Elastic Beanstalk mengonfigurasi beberapa folder file statis secara default. Untuk detailnya, lihat bagian konfigurasi file statis untuk Python dan Ruby. Anda dapat mengonfigurasi folder tambahan seperti yang dijelaskan pada halaman ini.

Konfigurasikan file statis menggunakan konsol tersebut

Untuk mengonfigurasi server proksi untuk menyajikan file statis

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Dalam kategori konfigurasi Pembaruan, pemantauan, dan pencatatan, pilih Edit.
5. Gulir ke bagian perangkat lunak Platform dan temukan grup file Statis.

    a. Untuk menambahkan pemetaan file statis, pilih Tambahkan file statis. Di baris tambahan yang muncul, Anda akan memasukkan jalur untuk menyajikan file statis dan direktori yang berisi file statis yang akan disajikan.

    - Di bidang Path, mulai nama jalur dengan garis miring (/) (misalnya, "/images “).
    - Di bidang Direktori, tentukan nama direktori yang terletak di root kode sumber aplikasi Anda. Jangan memulainya dengan garis miring (misalnya, "static/image-files “).

    > **Note**
    >
    > Jika Anda tidak melihat bagian File statis, Anda harus menambahkan setidaknya satu pemetaan dengan menggunakan file konfigurasi. Untuk detailnya, lihat the section called “Konfigurasikan file statis menggunakan opsi konfigurasi” di halaman ini.

b. Untuk menghapus pemetaan, pilih Hapus.

6. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Konfigurasikan file statis menggunakan opsi konfigurasi

Anda dapat menggunakan file konfigurasi untuk mengonfigurasi jalur file statis dan lokasi direktori menggunakan opsi konfigurasi. Anda dapat menambahkan file konfigurasi untuk paket sumber aplikasi Anda dan menerapkannya selama pembuatan lingkungan atau deployment selanjutnya.

Jika lingkungan Anda menggunakan cabang platform berbasis Amazon Linux 2, gunakan namespace `aws:elasticbeanstalk:environment:proxy:staticfiles`.

Contoh file konfigurasi berikut memberitahu server proksi untuk menyajikan file dalam folder `statichtml` pada jalur `/html`, dan file dalam `staticimages` folder pada jalur `/images`.

Example .ebextensions/static-files.config

```
option_settings:
  aws:elasticbeanstalk:environment:proxy:staticfiles:
    /html: statichtml
    /images: staticimages
```

Jika lingkungan Elastic Beanstalk Anda menggunakan versi platform Amazon Linux AMI (sebelumnya Amazon Linux 2), baca informasi tambahan berikut ini:

Namespace khusus platform Amazon Linux AMI

Pada cabang platform Amazon Linux AMI, namespace konfigurasi file statis bervariasi menurut platform. Untuk detailnya, lihat salah satu halaman berikut:

- Namespace konfigurasi Go
- Namespace konfigurasi Java SE
- Namespace konfigurasi Tomcat
- Node.jskonfigurasi namespace
- Namespace konfigurasi Python

Mengonfigurasi HTTPS untuk lingkungan Elastic Beanstalk Anda

Topik di bagian ini menjelaskan cara mengonfigurasi HTTPS untuk lingkungan Elastic Beanstalk Anda. HTTPS adalah suatu keharusan bagi setiap aplikasi yang mentransmisikan data pengguna atau informasi login.

Jika Anda telah membeli dan mengonfigurasi nama domain khusus untuk lingkungan Elastic Beanstalk Anda, Anda dapat menggunakan HTTPS untuk mengizinkan pengguna terhubung ke situs web Anda dengan aman.

Jika Anda tidak memiliki nama domain, Anda masih dapat menggunakan HTTPS dengan sertifikat yang ditandatangani sendiri untuk tujuan pengembangan dan pengujian. Untuk informasi selengkapnya, lihat Sertifikat server.

Mengkonfigurasi Penghentian HTTPS di penyeimbang beban

Load balancer mendistribusikan permintaan ke EC2 instance yang menjalankan aplikasi Anda. Penyeimbang beban juga menghilangkan kebutuhan untuk mengekspos instans Anda langsung ke internet. Cara termudah untuk menggunakan HTTPS dengan lingkungan multi-instance Elastic Beanstalk adalah dengan mengonfigurasi pendengar yang aman untuk penyeimbang beban. Koneksi antara klien dan penyeimbang beban tetap aman, sehingga Anda dapat mengonfigurasi penyeimbang beban untuk menghentikan HTTPS. Koneksi back end antara penyeimbang beban dan EC2 instance menggunakan HTTP, jadi tidak diperlukan konfigurasi tambahan instance. Untuk petunjuk mendetail untuk mengonfigurasi listenter aman, lihatMengkonfigurasi Penghentian HTTPS di penyeimbang beban.

Mengkonfigurasi Pengakhiran HTTPS pada instance EC2

Jika Anda menjalankan aplikasi dalam satu lingkungan instance, atau perlu mengamankan koneksi sampai ke EC2 instance di belakang penyeimbang beban, Anda dapat mengonfigurasi server proxy yang berjalan pada instance untuk menghentikan HTTPS. Mengonfigurasi instans Anda untuk mengakhiri koneksi HTTPS memerlukan penggunaan file konfigurasi untuk mengubah perangkat lunak yang berjalan pada instans, dan untuk mengubah grup keamanan agar yang memungkinkan koneksi aman. Untuk informasi selengkapnya, lihat Mengkonfigurasi Pengakhiran HTTPS pada instance.

Mengkonfigurasi HTTPS end-to-end

Untuk end-to-end HTTPS di lingkungan load-balanced, Anda dapat menggabungkan instance dan penghentian penyeimbang beban untuk mengenkripsi kedua koneksi. Secara default, jika

Anda mengonfigurasi penyeimbang beban untuk meneruskan lalu lintas menggunakan HTTPS, penyeimbang itu akan memercayai sertifikat apa pun yang disajikan oleh intans backend. Untuk keamanan maksimum, Anda dapat melampirkan kebijakan ke penyeimbang beban yang mencegahnya terhubung ke instans yang tidak menampilkan sertifikat publik yang dipercayainya. Untuk informasi selengkapnya, lihat Mengkonfigurasi end-to-end enkripsi dalam lingkungan Elastic Beanstalk Beanstalk yang seimbang.

Mengkonfigurasi HTTPS dengan TCP Passthrough

Anda juga dapat mengonfigurasi penyeimbang beban untuk menyampaikan lalu lintas HTTPS tanpa mendekripsinya. Untuk informasi selengkapnya, lihat Mengonfigurasi penyeimbang beban lingkungan Anda untuk TCP Passthrough.

> **Note**
>
> Apakah ia memiliki ular? contoh aplikasi pada GitHub menyertakan file konfigurasi dan instruksi untuk setiap metode konfigurasi HTTPS dengan aplikasi web Tomcat. Lihat file readme dan petunjuk HTTPS untuk detailnya.

Topik

- Sertifikat server
- Mengkonfigurasi Penghentian HTTPS di penyeimbang beban
- Mengkonfigurasi Pengakhiran HTTPS pada instance
- Mengkonfigurasi end-to-end enkripsi dalam lingkungan Elastic Beanstalk Beanstalk yang seimbang
- Mengonfigurasi penyeimbang beban lingkungan Anda untuk TCP Passthrough
- Mengonfigurasi HTTP ke pengalihan HTTPS

Sertifikat server

Topik ini menjelaskan berbagai jenis sertifikat yang dapat Anda gunakan untuk mengonfigurasi HTTPS dan kapan harus menerapkannya. Subtopik di bagian ini memberikan instruksi untuk membuat sertifikat Anda sendiri dan cara mengunggahnya.

AWS Certificate Manager (ACM)

ACM adalah alat pilihan untuk menyediakan, mengelola, dan menerapkan sertifikat server Anda. Anda dapat melakukannya secara terprogram atau menggunakan. AWS CLI Dengan ACM Anda dapat membuat sertifikat tepercaya untuk nama domain Anda secara gratis.

Sertifikat ACM hanya dapat digunakan dengan penyeimbang AWS beban dan CloudFront distribusi Amazon, dan ACM hanya tersedia di Wilayah tertentu. AWS Untuk menggunakan sertifikat ACM dengan Elastic Beanstalk, lihat Mengkonfigurasi Penghentian HTTPS di penyeimbang beban. Untuk informasi selengkapnya tentang ACM, lihat Panduan AWS Certificate Manager Pengguna.

> **Note**
>
> Untuk daftar wilayah di mana ACM tersedia, lihat titik akhir ACM dan kuota di. Referensi Umum Amazon Web Services

Jika ACM tidak tersedia di AWS Wilayah Anda, Anda dapat mengunggah sertifikat pihak ketiga atau yang ditandatangani sendiri dan kunci pribadi ke AWS Identity and Access Management (IAM). Anda dapat menggunakan AWS CLI untuk mengunggah sertifikat. Sertifikat yang disimpan dalam IAM dapat digunakan dengan penyeimbang beban dan distribusi. CloudFront Untuk informasi selengkapnya, lihat Unggah sertifikat ke IAM.

Sertifikat pihak ketiga

Jika ACM tidak tersedia di wilayah Anda, Anda dapat membeli sertifikat terpercaya dari pihak ketiga. Sertifikat pihak ketiga dapat digunakan untuk mendekripsi lalu lintas HTTPS di penyeimbang beban, pada instans backend, atau keduanya.

Sertifikat yang ditandatangani sendiri

Untuk pengembangan dan pengujian, Anda dapat membuat dan menandatangani sertifikat sendiri dengan alat sumber terbuka. Sertifikat yang ditandatangani sendiri gratis dan mudah dibuat, namun tidak dapat digunakan untuk dekripsi front-end di situs publik. Jika Anda mencoba untuk menggunakan sertifikat yang ditandatangani sendiri untuk koneksi HTTPS ke klien, peramban pengguna menampilkan pesan galat yang menunjukkan bahwa situs web Anda tidak aman. Namun, Anda dapat, menggunakan sertifikat yang ditandatangani sendiri untuk mengamankan koneksi backend tanpa masalah.

Buat dan tandatangani sertifikat X509

Anda dapat membuat sertifikat X509 untuk aplikasi Anda dengan `OpenSSL`. OpenSSL adalah perpustakaan sumber terbuka standar yang mendukung berbagai fungsi kriptografi, termasuk pembuatan dan penandatanganan sertifikat x509. Untuk informasi selengkapnya tentang OpenSSL, kunjungi www.openssl.org.

> **Note**
>
> Anda hanya perlu membuat sertifikat secara lokal jika Anda ingin menggunakan HTTPS dalam lingkungan instans tunggal atau mengenkripsi ulang pada backend dengan sertifikat yang ditandatangani sendiri. Jika Anda memiliki nama domain, Anda dapat membuat sertifikat AWS dan menggunakannya dengan lingkungan load-balanced secara gratis dengan menggunakan AWS Certificate Manager (ACM). Lihat Meminta Sertifikat di Panduan Pengguna AWS Certificate Manager untuk petunjuk.

Jalankan `openssl version` pada baris perintah untuk melihat apakah Anda sudah memasang OpenSSL. Jika tidak, Anda dapat membangun dan menginstal kode sumber menggunakan instruksi di GitHub repositori publik, atau menggunakan manajer paket favorit Anda. OpenSSL juga diinstal pada gambar Linux Elastic Beanstalk, jadi alternatif cepat adalah terhubung ke EC2 instance di lingkungan berjalan dengan menggunakan perintah EB CLI: eb ssh

```
~/eb$ eb ssh
[ec2-user@ip-255-55-55-255 ~]$ openssl version
OpenSSL 1.0.1k-fips 8 Jan 2015
```

Anda perlu membuat kunci pribadi RSA untuk membuat permintaan penandatanganan sertifikat (CSR). Untuk membuat kunci pribadi Anda, gunakan perintah openssl genrsa:

```
[ec2-user@ip-255-55-55-255 ~]$ openssl genrsa 2048 > privatekey.pem
Generating RSA private key, 2048 bit long modulus
.....................................................................................
+++
..............+++
e is 65537 (0x10001)
```

privatekey.pem

Nama file tempat Anda ingin menyimpan kunci pribadi. Biasanya, perintah openssl genrsa mencetak konten kunci pribadi ke layar, tetapi perintah ini menyalurkan output ke file. Pilih nama file apa pun, dan simpan file di tempat yang aman sehingga Anda dapat mengambilnya nanti. Jika Anda kehilangan kunci pribadi, Anda tidak akan dapat menggunakan sertifikat Anda.

CSR adalah file yang Anda kirim ke otoritas sertifikat (CA) untuk mengajukan sertifikat server digital. Untuk membuat CSR, gunakan perintah openssl req:

```
$ openssl req -new -key privatekey.pem -out csr.pem
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
```

Masukkan informasi yang diminta dan tekan Masukkan. Tabel berikut menjelaskan dan menunjukkan contoh untuk setiap bidang.

Nama	Deskripsi	Contoh
Nama Negara	Singkatan ISO dua huruf untuk negara Anda.	AS = Amerika Serikat
Negara Bagian atau Provinsi	Nama negara bagian atau provinsi tempat organisasi Anda berada. Anda tidak dapat menyingkat nama ini.	Washington
Nama Lokal	Nama kota tempat organisasi Anda berada.	Seattle
Nama Organisasi	Nama lengkap legal organisasi Anda. Jangan menyingkat nama organisasi Anda.	Contoh Perusahaan
Unit Organisasi	Opsional, untuk informasi organisasi tambahan.	Pemasaran
Nama Umum	Nama domain yang memenuhi syarat untuk situs web Anda. Ini harus sesuai dengan nama domain yang pengguna lihat ketika	www.example.com

Nama	Deskripsi	Contoh
	mereka mengunjungi situs Anda, jika tidak, kesalahan sertifikat akan ditampilkan.	
Alamat email	Alamat email administrator situs.	someone@example.com

Anda dapat mengirimkan permintaan penandatanganan ke pihak ketiga untuk ditandatangani, atau menandatanganinya sendiri untuk pengembangan dan pengujian. Sertifikat yang ditandatangani sendiri juga dapat digunakan untuk HTTPS backend antara penyeimbang beban dan instance. EC2

Untuk menandatangani sertifikat, gunakan perintah openssl x509. Contoh berikut menggunakan kunci pribadi dari langkah sebelumnya (*privatekey.pem*) dan permintaan penandatanganan (*csr.pem*) untuk membuat sertifikat publik bernama *public.crt* yang valid selama *365* berhari-hari.

```
$ openssl x509 -req -days 365 -in csr.pem -signkey privatekey.pem -out public.crt
Signature ok
subject=/C=us/ST=washington/L=seattle/O=example corporation/OU=marketing/
CN=www.example.com/emailAddress=someone@example.com
Getting Private key
```

Simpan kunci pribadi dan sertifikat publik untuk digunakan nanti. Anda dapat membuang permintaan penandatanganan. Selalu simpan kunci pribadi di lokasi yang aman dan hindari menambahkannya ke kode sumber Anda.

Untuk menggunakan sertifikat dengan platform Windows Server, Anda harus mengubahnya ke format PFX. Gunakan perintah berikut untuk membuat sertifikat PFX dari file sertifikat pribadi dan publik:

```
$ openssl pkcs12 -export -out example.com.pfx -inkey privatekey.pem -in public.crt
Enter Export Password: password
Verifying - Enter Export Password: password
```

Setelah Anda memiliki sertifikat, Anda dapat mengunggahnya ke IAM untuk digunakan dengan penyeimbang beban, atau mengonfigurasi instans di lingkungan Anda untuk mengakhiri HTTPS.

Unggah sertifikat ke IAM

Untuk menggunakan sertifikat Anda dengan penyeimbang beban lingkungan Elastic Beanstalk Anda, unggah sertifikat dan kunci pribadi ke (IAM). AWS Identity and Access Management Anda dapat

menggunakan sertifikat yang disimpan di IAM dengan penyeimbang beban Elastic Load Balancing dan distribusi Amazon. CloudFront

> **Note**
>
> AWS Certificate Manager (ACM) adalah alat pilihan untuk menyediakan, mengelola, dan menyebarkan sertifikat server Anda. Untuk informasi selengkapnya tentang permintaan sertifikat ACM, lihat Meminta Sertifikat di AWS Certificate Manager Panduan Pengguna. Untuk informasi tentang mengimpor sertifikat pihak ke tiga ke ACM, lihat Mengimpor Sertifikat di Panduan Pengguna AWS Certificate Manager . Gunakan IAM untuk mengunggah sertifikat hanya jika ACM tidak tersedia di Wilayah Anda AWS.

Anda dapat menggunakan AWS Command Line Interface (AWS CLI) untuk mengunggah sertifikat Anda. Perintah berikut mengunggah sertifikat yang ditandatangani sendiri bernama *`https-cert.crt`* dengan kunci pribadi bernama: *`private-key.pem`*

```
$ aws iam upload-server-certificate --server-certificate-name elastic-beanstalk-x509 --
certificate-body file://https-cert.crt --private-key file://private-key.pem
{
    "ServerCertificateMetadata": {
        "ServerCertificateId": "AS5YBEIONO2Q7CAIHKNGC",
        "ServerCertificateName": "elastic-beanstalk-x509",
        "Expiration": "2017-01-31T23:06:22Z",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:server-certificate/elastic-beanstalk-x509",
        "UploadDate": "2016-02-01T23:10:34.167Z"
    }
}
```

`file://`Awalan memberitahu AWS CLI untuk memuat isi file dalam direktori saat ini. *`elastic-beanstalk-x509`*menentukan nama untuk memanggil sertifikat di IAM.

Jika Anda membeli sertifikat dari otoritas sertifikat dan menerima file rantai sertifikat, unggah juga dengan menyertakan opsi `--certificate-chain`:

```
$ aws iam upload-server-certificate --server-certificate-name elastic-beanstalk-x509 --
certificate-chain file://certificate-chain.pem --certificate-body file://https-cert.crt
 --private-key file://private-key.pem
```

Buat catatan dari Amazon Resource Name (ARN) untuk sertifikat Anda. Anda akan menggunakan catatan itu saat memperbarui pengaturan konfigurasi penyeimbang beban untuk menggunakan HTTPS.

> **Note**
>
> Sertifikat yang diunggah ke IAM akan tetap disimpan meskipun sertifikat itu tidak lagi digunakan di penyeimbang beban lingkungan mana pun. Sertifikat ini berisi data sensitif. Saat Anda tidak lagi membutuhkan sertifikat tersebut untuk lingkungan apapun, pastikan untuk menghapusnya. Untuk detail tentang menghapus sertifikat dari IAM, lihat https://docs.aws.amazon.com/IAM/latest/UserGuide/id_credentials_server-certs.html#delete-server-certificate.

Untuk informasi selengkapnya tentang sertifikat server di IAM, lihat Bekerja dengan Sertifikat Server di Panduan Pengguna IAM.

Menyimpan kunci pribadi dengan aman di Amazon S3

Kunci pribadi yang Anda gunakan untuk menandatangani sertifikat publik pribadi Anda bersifat pribadi dan tidak boleh diserahkan ke kode sumber. Anda dapat menghindari menyimpan kunci pribadi dalam file konfigurasi dengan mengunggahnya ke Amazon S3, dan mengonfigurasi Elastic Beanstalk untuk mengunduh file dari Amazon S3 selama deployment aplikasi.

Contoh berikut menunjukkan bagian Sumber Daya dan file dari file konfigurasi mengunduh file kunci pribadi dari bucket Amazon S3.

Example .ebextensions/privatekey.config

```
Resources:
  AWSEBAutoScalingGroup:
    Metadata:
      AWS::CloudFormation::Authentication:
        S3Auth:
          type: "s3"
          buckets: ["elasticbeanstalk-us-west-2-123456789012"]
          roleName:
            "Fn::GetOptionSetting":
              Namespace: "aws:autoscaling:launchconfiguration"
              OptionName: "IamInstanceProfile"
```

```
              DefaultValue: "aws-elasticbeanstalk-ec2-role"
files:
  # Private key
  "/etc/pki/tls/certs/server.key":
    mode: "000400"
    owner: root
    group: root
    authentication: "S3Auth"
    source: https://elasticbeanstalk-us-west-2-123456789012.s3.us-west-2.amazonaws.com/
server.key
```

Ganti nama bucket dan URL dalam contoh dengan milik Anda. Entri pertama dalam file ini menambahkan metode otentikasi yang dinamai `S3Auth` ke metadata grup Auto Scaling lingkungan. Jika Anda telah mengonfigurasi profil instans khusus untuk lingkungan Anda, itu akan digunakan, jika tidak nilai default `aws-elasticbeanstalk-ec2-role` akan diterapkan. Profil instans default memiliki izin untuk membaca dari bucket penyimpanan Elastic Beanstalk. Jika Anda menggunakan bucket yang berbeda, tambahkan izin ke profil instans.

Entri kedua menggunakan `S3Auth` metode otentikasi untuk mengunduh kunci pribadi dari URL yang ditentukan dan menyimpannya ke `/etc/pki/tls/certs/server.key`. Kemudian server proksi dapat membaca kunci pribadi dari lokasi ini untuk mengakhiri koneksi HTTPS pada instans.

Profil instans yang ditetapkan ke EC2 instance lingkungan Anda harus memiliki izin untuk membaca objek kunci dari bucket yang ditentukan. Verifikasi bahwa profil instans memiliki izin untuk membaca objek di IAM, dan bahwa izin pada bucket dan objek tidak melarang profil instans.

Melihat izin bucket

1. Buka Konsol Manajemen Amazon S3.
2. Pilih bucket.
3. Pilih Properti lalu pilih Izin.
4. Verifikasi bahwa akun Anda adalah penerima pada bucket dengan izin baca.
5. Jika kebijakan bucket dilampirkan, pilih Kebijakan bucket untuk melihat izin yang ditetapkan ke bucket.

Mengkonfigurasi Penghentian HTTPS di penyeimbang beban

Untuk memperbarui AWS Elastic Beanstalk lingkungan agar menggunakan HTTPS, Anda perlu mengonfigurasi pendengar HTTPS untuk penyeimbang beban di lingkungan Anda. Dua jenis

penyeimbang beban mendukung pendengar HTTPS: Classic Load Balancer dan Application Load Balancer.

Anda dapat menggunakan konsol Elastic Beanstalk atau file konfigurasi untuk mengonfigurasi pendengar yang aman dan menetapkan sertifikat.

> **Note**
>
> Lingkungan instans tunggal tidak memiliki penyeimbang beban dan tidak mendukung penghentian HTTPS di penyeimbang beban.

Mengonfigurasi pendengar yang aman menggunakan konsol Elastic Beanstalk

Untuk menetapkan sertifikat untuk penyeimbang beban lingkungan Anda

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Pada kategori konfigurasi Penyeimbang beban, pilih Edit.

   > **Note**
   >
   > Jika kategori konfigurasi Penyeimbang beban tidak memiliki tombol Edit, lingkungan Anda tidak memiliki penyeimbang beban.

5. Pada halaman Ubah penyeimbang beban, prosedurnya bervariasi tergantung pada jenis penyeimbang beban yang terkait dengan lingkungan Anda.
   - Classic Load Balancer
     a. Pilih Tambahkan pendengar.
     b. Di kotak dialog pendengar Classic Load Balancer, konfigurasikan pengaturan berikut:
        - Untuk port Pendengar, ketik port lalu lintas masuk, biasanya `443`.
        - Untuk protokol Pendengar, pilih HTTPS.
        - Untuk port Instans, ketik `80`.

- Untuk Protokol instans, pilih HTTP.
- Untuk Sertifikat SSL, pilih sertifikat Anda.

c. Pilih Tambahkan.

- Penyeimbang Beban Aplikasi

a. Pilih Tambahkan pendengar.

b. Di kotak dialog pendengar Application Load Balancer, konfigurasikan pengaturan berikut:

- Untuk Port, ketik port lalu lintas masuk, biasanya 443.
- Untuk Protokol, pilih HTTPS.
- Untuk Sertifikat SSL, pilih sertifikat Anda.

c. Pilih Tambahkan.

> **Note**
>
> Untuk Classic Load Balancer dan Application Load Balancer, jika menu drop-down tidak menampilkan sertifikat apa pun, Anda harus membuat atau mengunggah sertifikat untuk nama domain khusus Anda di (ACM)AWS Certificate Manager (lebih disukai). Atau, unggah sertifikat ke IAM dengan AWS CLI.

- Penyeimbang Beban Jaringan

a. Pilih Tambahkan pendengar.

b. Di kotak dialog pendengar Network Load Balancer, untuk Port, ketik port lalu lintas masuk, biasanya 443.

c. Pilih Tambahkan.

6. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Mengonfigurasi pendengar yang aman menggunakan file konfigurasi

Anda dapat mengonfigurasi pendengar yang aman pada penyeimbang beban Anda dengan salah satu file konfigurasi berikut.

Example .ebextensions/securelistener-qqclb.config

Gunakan contoh ini bila lingkungan Anda memiliki Classic Load Balancer. Contoh ini menggunakan opsi di namespace `aws:elb:listener` untuk mengonfigurasi pendengar HTTPS pada port 443 dengan sertifikat tertentu, dan untuk meneruskan lalu lintas terdekripsi ke instans di lingkungan Anda pada port 80.

```
option_settings:
  aws:elb:listener:443:
    SSLCertificateId: arn:aws:acm:us-east-2:1234567890123:certificate/
####################################
    ListenerProtocol: HTTPS
    InstancePort: 80
```

Ganti teks yang disorot dengan ARN sertifikat Anda. Sertifikat dapat berupa sertifikat yang Anda buat atau unggah di AWS Certificate Manager (ACM) (disukai), atau yang Anda unggah ke IAM dengan sertifikat. AWS CLI

Untuk informasi selengkapnya tentang opsi konfigurasi Classic Load Balancer, lihat Namespace konfigurasi Classic Load Balancer.

Example .ebextensions/securelistener-alb.config

Gunakan contoh ini bila lingkungan Anda memiliki Application Load Balancer. Contoh ini menggunakan opsi di namespace `aws:elbv2:listener` untuk mengonfigurasi pendengar HTTPS pada port 443 dengan sertifikat tertentu. Pendengar merutekan lalu lintas ke proses default.

```
option_settings:
  aws:elbv2:listener:443:
    ListenerEnabled: 'true'
    Protocol: HTTPS
    SSLCertificateArns: arn:aws:acm:us-east-2:1234567890123:certificate/
####################################
```

Example .ebextensions/securelistener-nlb.config

Gunakan contoh ini bila lingkungan Anda memiliki Network Load Balancer. Contoh ini menggunakan opsi di namespace `aws:elbv2:listener` untuk mengonfigurasi pendengar pada port 443. Pendengar merutekan lalu lintas ke proses default.

```
option_settings:
```

```
  aws:elbv2:listener:443:
    ListenerEnabled: 'true'
```

Mengonfigurasi grup keamanan Anda

Jika Anda mengonfigurasi penyeimbang beban Anda untuk meneruskan lalu lintas ke port instans selain port 80, Anda harus menambahkan aturan untuk grup keamanan Anda yang mengizinkan lalu lintas masuk melalui port instans dari penyeimbang beban Anda. Jika Anda membuat lingkungan Anda di VPC khusus, Elastic Beanstalk menambahkan aturan ini untuk Anda.

Anda menambahkan aturan ini dengan menambahkan kunci `Resources` ke file konfigurasi di direktori `.ebextensions` untuk aplikasi Anda.

Contoh file konfigurasi berikut menambahkan aturan masuk ke `AWSEBSecurityGroup` grup keamanan. Ini mengizinkan lalu lintas pada port 1000 dari grup keamanan penyeimbang beban.

Example .ebextensions/sg-ingressfromlb.config

```
Resources:
  sslSecurityGroupIngress:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: {"Fn::GetAtt" : ["AWSEBSecurityGroup", "GroupId"]}
      IpProtocol: tcp
      ToPort: 1000
      FromPort: 1000
      SourceSecurityGroupId: {"Fn::GetAtt" : ["AWSEBLoadBalancerSecurityGroup",
 "GroupId"]}
```

Mengkonfigurasi Pengakhiran HTTPS pada instance

Anda dapat menggunakan file konfigurasi untuk mengonfigurasi server proksi yang melewati lalu lintas ke aplikasi Anda untuk mengakhiri koneksi HTTPS. Hal ini berguna jika Anda ingin menggunakan HTTPS dengan lingkungan intans tunggal, atau jika Anda mengonfigurasi penyeimbang beban Anda untuk melewati lalu lintas tanpa mendekripsinya.

Untuk mengaktifkan HTTPS, Anda harus mengizinkan lalu lintas masuk di port 443 ke EC2 instance tempat aplikasi Elastic Beanstalk Anda berjalan. Anda melakukan ini dengan menggunakan `Resources` kunci dalam file konfigurasi untuk menambahkan aturan untuk port 443 ke aturan ingress untuk AWSEBSecurity grup keamanan Grup.

Snippet berikut menambahkan aturan masuk ke `AWSEBSecurityGroup` grup keamanan yang membuka port 443 ke semua lalu lintas untuk lingkungan instans tunggal:

`.ebextensions/https-instance-securitygroup.config`

```
Resources:
  sslSecurityGroupIngress:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: {"Fn::GetAtt" : ["AWSEBSecurityGroup", "GroupId"]}
      IpProtocol: tcp
      ToPort: 443
      FromPort: 443
      CidrIp: 0.0.0.0/0
```

Dalam lingkungan yang seimbang dengan beban di Amazon Virtual Private Cloud (Amazon VPC) default, Anda dapat mengubah kebijakan ini hanya untuk menerima lalu lintas dari penyeimbang beban. Lihat Mengkonfigurasi end-to-end enkripsi dalam lingkungan Elastic Beanstalk Beanstalk yang seimbang untuk contoh.

Platform

- Mengakhiri HTTPS pada EC2 instance yang menjalankan Docker
- Mengakhiri HTTPS pada EC2 instance yang menjalankan Go
- Mengakhiri HTTPS pada EC2 instance yang menjalankan Java SE
- Mengakhiri HTTPS pada EC2 instance yang menjalankan Node.js
- Mengakhiri HTTPS pada EC2 instance yang menjalankan PHP
- Mengakhiri HTTPS pada EC2 instance yang menjalankan Python
- Mengakhiri HTTPS pada EC2 instance yang menjalankan Ruby
- Mengakhiri HTTPS pada EC2 instance yang menjalankan Tomcat
- Mengakhiri HTTPS di EC2 instans Amazon yang menjalankan .NET Core di Linux
- Mengakhiri HTTPS di EC2 instans Amazon yang menjalankan .NET

Mengakhiri HTTPS pada EC2 instance yang menjalankan Docker

Untuk kontainer Docker, Anda menggunakan file konfigurasi untuk mengaktifkan HTTPS.

Tambahkan snippet berikut ke file konfigurasi Anda, menggantikan sertifikat dan materi kunci pribadi seperti yang diinstruksikan, dan simpan snippet di direktori `.ebextensions` paket sumber Anda. File konfigurasi melakukan tugas berikut:

- Kunci `files` tersebut membuat file berikut pada instans:

  `/etc/nginx/conf.d/https.conf`

  Konfigurasikan server nginx. File ini dimuat ketika layanan nginx dimulai.

  `/etc/pki/tls/certs/server.crt`

  Membuat file sertifikat pada instans. Ganti *`certificate file contents`* dengan isi sertifikat Anda.

  > **Note**
  >
  > YAML bergantung pada indentasi yang konsisten. Cocokkan tingkat indentasi saat mengganti konten dalam file konfigurasi contoh dan pastikan bahwa editor teks Anda menggunakan spasi, bukan karakter tab, untuk indentasi.

  Jika Anda memiliki sertifikat menengah, sertakan sertifikat tersebut di `server.crt` setelah sertifikat situs Anda.

  ```
      -----BEGIN CERTIFICATE-----
  certificate file contents
  -----END CERTIFICATE-----
  -----BEGIN CERTIFICATE-----
  first intermediate certificate
  -----END CERTIFICATE-----
  -----BEGIN CERTIFICATE-----
  second intermediate certificate
  -----END CERTIFICATE-----
  ```

  `/etc/pki/tls/certs/server.key`

  Membuat file kunci pribadi pada instans. Ganti *`private key contents`* dengan isi kunci pribadi yang digunakan untuk membuat permintaan sertifikat atau sertifikat yang ditandatangani sendiri.

Example .ebextensions/https-instance.config

```
files:
  /etc/nginx/conf.d/https.conf:
    mode: "000644"
    owner: root
    group: root
    content: |
      # HTTPS Server

      server {
        listen 443;
        server_name localhost;

        ssl on;
        ssl_certificate /etc/pki/tls/certs/server.crt;
        ssl_certificate_key /etc/pki/tls/certs/server.key;

        ssl_session_timeout 5m;

        ssl_protocols TLSv1 TLSv1.1 TLSv1.2;
        ssl_prefer_server_ciphers on;

        location / {
          proxy_pass http://docker;
          proxy_http_version 1.1;

          proxy_set_header Connection "";
          proxy_set_header Host $host;
          proxy_set_header X-Real-IP $remote_addr;
          proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
          proxy_set_header X-Forwarded-Proto https;
        }
      }

  /etc/pki/tls/certs/server.crt:
    mode: "000400"
    owner: root
    group: root
    content: |
      -----BEGIN CERTIFICATE-----
      certificate file contents
      -----END CERTIFICATE-----
```

```
  /etc/pki/tls/certs/server.key:
    mode: "000400"
    owner: root
    group: root
    content: |
      -----BEGIN RSA PRIVATE KEY-----
      private key contents # See note below.
      -----END RSA PRIVATE KEY-----
```

> **Note**
>
> Hindari melakukan file konfigurasi yang berisi kunci pribadi Anda ke kontrol sumber. Setelah Anda menguji konfigurasi dan mengonfirmasi bahwa konfigurasi berfungsi, simpan kunci pribadi Anda di Amazon S3 dan ubah konfigurasi untuk mengunduhnya selama deployment. Untuk instruksi, lihat Menyimpan kunci pribadi dengan aman di Amazon S3.

Pada lingkungan instans tunggal, Anda juga harus mengubah grup keamanan instans untuk mengizinkan lalu lintas pada port 443. File konfigurasi berikut mengambil ID grup keamanan menggunakan AWS CloudFormation fungsi dan menambahkan aturan ke dalamnya.

Example .ebextensions/ .config https-instance-single

```
Resources:
  sslSecurityGroupIngress:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: {"Fn::GetAtt" : ["AWSEBSecurityGroup", "GroupId"]}
      IpProtocol: tcp
      ToPort: 443
      FromPort: 443
      CidrIp: 0.0.0.0/0
```

Untuk lingkungan yang seimbang beban, Anda mengonfigurasi penyeimbang beban untuk meneruskan lalu lintas aman melalui yang tidak tersentuh, atau mendekripsi dan mengenkripsi ulang untuk enkripsi. end-to-end

Mengakhiri HTTPS pada EC2 instance yang menjalankan Go

Untuk tipe kontainer Go, Anda mengaktifkan HTTPS dengan file konfigurasi dan file konfigurasi nginx yang mengonfigurasi server nginx untuk menggunakan HTTPS.

Tambahkan snippet berikut ke file konfigurasi Anda, menggantikan sertifikat dan placeholder kunci pribadi seperti yang diperintahkan, dan simpan snippet di direktori `.ebextensions` paket sumber Anda. File konfigurasi melakukan tugas berikut:

- Kunci `Resources` tersebut mengaktifkan port 443 pada grup keamanan yang digunakan oleh instans lingkungan Anda.
- Kunci `files` tersebut membuat file berikut pada instans:

  `/etc/pki/tls/certs/server.crt`

  Membuat file sertifikat pada instans. Ganti *`certificate file contents`* dengan isi sertifikat Anda.

  > **Note**
  >
  > YAML bergantung pada indentasi yang konsisten. Cocokkan tingkat indentasi saat mengganti konten dalam file konfigurasi contoh dan pastikan bahwa editor teks Anda menggunakan spasi, bukan karakter tab, untuk indentasi.

  Jika Anda memiliki sertifikat menengah, sertakan sertifikat tersebut di `server.crt` setelah sertifikat situs Anda.

  ```
      -----BEGIN CERTIFICATE-----
  certificate file contents
  -----END CERTIFICATE-----
  -----BEGIN CERTIFICATE-----
  first intermediate certificate
  -----END CERTIFICATE-----
  -----BEGIN CERTIFICATE-----
  second intermediate certificate
  -----END CERTIFICATE-----
  ```

  `/etc/pki/tls/certs/server.key`

  Membuat file kunci pribadi pada instans. Ganti *`private key contents`* dengan isi kunci pribadi yang digunakan untuk membuat permintaan sertifikat atau sertifikat yang ditandatangani sendiri.
- Kunci `container_commands` tersebut memulai ulang serverr nginx setelah semuanya dikonfigurasi sehingga server memuat file konfigurasi nginx.

Example .ebextensions/https-instance.config

```
files:
  /etc/pki/tls/certs/server.crt:
    content: |
      -----BEGIN CERTIFICATE-----
      certificate file contents
      -----END CERTIFICATE-----

  /etc/pki/tls/certs/server.key:
    content: |
      -----BEGIN RSA PRIVATE KEY-----
      private key contents # See note below.
      -----END RSA PRIVATE KEY-----

container_commands:
  01restart_nginx:
    command: "service nginx restart"
```

> **Note**
>
> Hindari melakukan file konfigurasi yang berisi kunci pribadi Anda ke kontrol sumber. Setelah Anda menguji konfigurasi dan mengonfirmasi bahwa konfigurasi berfungsi, simpan kunci pribadi Anda di Amazon S3 dan ubah konfigurasi untuk mengunduhnya selama deployment. Untuk instruksi, lihat Menyimpan kunci pribadi dengan aman di Amazon S3.

Tempatkan hal berikut dalam sebuah file dengan ekstensi `.conf` di `.ebextensions/nginx/conf.d/` direktori paket sumber Anda (misalnya, `.ebextensions/nginx/conf.d/https.conf`). Ganti *`app_port`* dengan nomor port yang didengarkan aplikasi Anda. Contoh ini mengonfigurasi server nginx untuk mendengarkan pada port 443 menggunakan SSL. Untuk informasi selengkapnya tentang file konfigurasi ini pada platform Go, lihat Mengonfigurasi server proksi.

Example . ebextensions/nginx/conf.d/https.conf

```
# HTTPS server

server {
    listen       443;
    server_name  localhost;
```

```
    ssl                 on;
    ssl_certificate     /etc/pki/tls/certs/server.crt;
    ssl_certificate_key  /etc/pki/tls/certs/server.key;

    ssl_session_timeout  5m;

    ssl_protocols  TLSv1 TLSv1.1 TLSv1.2;
    ssl_prefer_server_ciphers   on;

    location / {
        proxy_pass  http://localhost:app_port;
        proxy_set_header   Connection "";
        proxy_http_version 1.1;
        proxy_set_header        Host            $host;
        proxy_set_header        X-Real-IP       $remote_addr;
        proxy_set_header        X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header        X-Forwarded-Proto https;
    }
}
```

Pada lingkungan instans tunggal, Anda juga harus mengubah grup keamanan instans untuk mengizinkan lalu lintas pada port 443. File konfigurasi berikut mengambil ID grup keamanan menggunakan AWS CloudFormation fungsi dan menambahkan aturan ke dalamnya.

Example .ebextensions/ .config https-instance-single

```
Resources:
  sslSecurityGroupIngress:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: {"Fn::GetAtt" : ["AWSEBSecurityGroup", "GroupId"]}
      IpProtocol: tcp
      ToPort: 443
      FromPort: 443
      CidrIp: 0.0.0.0/0
```

Untuk lingkungan yang seimbang beban, Anda mengonfigurasi penyeimbang beban untuk meneruskan lalu lintas aman melalui yang tidak tersentuh, atau mendekripsi dan mengenkripsi ulang untuk enkripsi. end-to-end

Mengakhiri HTTPS pada EC2 instance yang menjalankan Java SE

Untuk jenis kontainer Java SE, Anda mengaktifkan HTTPS dengan file konfigurasi .ebextensions, dan file konfigurasi nginx yang mengonfigurasi server nginx untuk menggunakan HTTPS.

Semua AL2 AL2 023/platform mendukung fitur konfigurasi proxy yang seragam. Untuk informasi selengkapnya tentang mengonfigurasi server proxy pada versi platform Anda yang menjalankan AL2 023/AL2, lihat. Konfigurasi proksi terbalik

Tambahkan snippet berikut ke file konfigurasi Anda, menggantikan sertifikat dan placeholder kunci pribadi seperti yang diperintahkan, dan simpan snippet dalam direktori `.ebextensions`. File konfigurasi melakukan tugas berikut:

- Kunci `files` tersebut membuat file berikut pada instans:

  `/etc/pki/tls/certs/server.crt`

  Membuat file sertifikat pada instans. Ganti *`certificate file contents`* dengan isi sertifikat Anda.

  > **Note**
  >
  > YAML bergantung pada indentasi yang konsisten. Cocokkan tingkat indentasi saat mengganti konten dalam file konfigurasi contoh dan pastikan bahwa editor teks Anda menggunakan spasi, bukan karakter tab, untuk indentasi.

  Jika Anda memiliki sertifikat menengah, sertakan sertifikat tersebut di `server.crt` setelah sertifikat situs Anda.

```
    -----BEGIN CERTIFICATE-----
certificate file contents
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
first intermediate certificate
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
second intermediate certificate
-----END CERTIFICATE-----
```

`/etc/pki/tls/certs/server.key`

Membuat file kunci pribadi pada instans. Ganti *`private key contents`* dengan isi kunci pribadi yang digunakan untuk membuat permintaan sertifikat atau sertifikat yang ditandatangani sendiri.

- Kunci `container_commands` tersebut memulai ulang serverr nginx setelah semuanya dikonfigurasi sehingga server memuat file konfigurasi nginx.

Example .ebextensions/https-instance.config

```
files:
  /etc/pki/tls/certs/server.crt:
    content: |
      -----BEGIN CERTIFICATE-----
      certificate file contents
      -----END CERTIFICATE-----

  /etc/pki/tls/certs/server.key:
    content: |
      -----BEGIN RSA PRIVATE KEY-----
      private key contents # See note below.
      -----END RSA PRIVATE KEY-----

container_commands:
  01restart_nginx:
    command: "service nginx restart"
```

> **Note**
>
> Hindari melakukan file konfigurasi yang berisi kunci pribadi Anda ke kontrol sumber. Setelah Anda menguji konfigurasi dan mengonfirmasi bahwa konfigurasi berfungsi, simpan kunci pribadi Anda di Amazon S3 dan ubah konfigurasi untuk mengunduhnya selama deployment. Untuk instruksi, lihat Menyimpan kunci pribadi dengan aman di Amazon S3.

Tempatkan hal berikut dalam sebuah file dengan ekstensi `.conf` di `.ebextensions/nginx/conf.d/` direktori paket sumber Anda (misalnya, `.ebextensions/nginx/conf.d/https.conf`). Ganti *`app_port`* dengan nomor port yang didengarkan aplikasi Anda. Contoh ini mengonfigurasi

server nginx untuk mendengarkan pada port 443 menggunakan SSL. Untuk informasi selengkapnya tentang file konfigurasi ini pada platform Java SE, lihat Mengonfigurasi server proksi.

Example . ebextensions/nginx/conf.d/https.conf

```
# HTTPS server

server {
    listen       443;
    server_name  localhost;

    ssl                  on;
    ssl_certificate      /etc/pki/tls/certs/server.crt;
    ssl_certificate_key  /etc/pki/tls/certs/server.key;

    ssl_session_timeout  5m;

    ssl_protocols  TLSv1 TLSv1.1 TLSv1.2;
    ssl_prefer_server_ciphers   on;

    location / {
        proxy_pass  http://localhost:app_port;
        proxy_set_header   Connection "";
        proxy_http_version 1.1;
        proxy_set_header        Host            $host;
        proxy_set_header        X-Real-IP       $remote_addr;
        proxy_set_header        X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header        X-Forwarded-Proto https;
    }
}
```

Pada lingkungan instans tunggal, Anda juga harus mengubah grup keamanan instans untuk mengizinkan lalu lintas pada port 443. File konfigurasi berikut mengambil ID grup keamanan menggunakan AWS CloudFormation fungsi dan menambahkan aturan ke dalamnya.

Example .ebextensions/ .config https-instance-single

```
Resources:
  sslSecurityGroupIngress:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: {"Fn::GetAtt" : ["AWSEBSecurityGroup", "GroupId"]}
      IpProtocol: tcp
```

```
      ToPort: 443
      FromPort: 443
      CidrIp: 0.0.0.0/0
```

Untuk lingkungan yang seimbang beban, Anda mengonfigurasi penyeimbang beban untuk meneruskan lalu lintas aman melalui yang tidak tersentuh, atau mendekripsi dan mengenkripsi ulang untuk enkripsi. end-to-end

Mengakhiri HTTPS pada EC2 instance yang menjalankan Node.js

File konfigurasi contoh berikut memperluas konfigurasi nginx default untuk mendengarkan di port 443 dan menghentikan koneksi SSL/TLS dengan sertifikat publik dan kunci pribadi.

Jika Anda mengonfigurasi lingkungan Anda untuk pelaporan kondisi yang ditingkatkan, Anda perlu mengonfigurasi nginx untuk menghasilkan log akses. Untuk melakukannya, batalkan komentar blok baris di bawah komentar yang berbunyi `# For enhanced health...` dengan menghapus karakter utama#.

Example .ebextensions/https-instance.config

```
files:
  /etc/nginx/conf.d/https.conf:
    mode: "000644"
    owner: root
    group: root
    content: |
      # HTTPS server

      server {
          listen       443;
          server_name  localhost;

          ssl                  on;
          ssl_certificate      /etc/pki/tls/certs/server.crt;
          ssl_certificate_key  /etc/pki/tls/certs/server.key;

          ssl_session_timeout  5m;

          ssl_protocols  TLSv1 TLSv1.1 TLSv1.2;
          ssl_prefer_server_ciphers   on;

          # For enhanced health reporting support, uncomment this block:
```

```
        #if ($time_iso8601 ~ "^(\d{4})-(\d{2})-(\d{2})T(\d{2})") {
        #    set $year $1;
        #    set $month $2;
        #    set $day $3;
        #    set $hour $4;
        #}
        #access_log /var/log/nginx/healthd/application.log.$year-$month-$day-$hour
healthd;
        #access_log  /var/log/nginx/access.log  main;

        location / {
            proxy_pass  http://nodejs;
            proxy_set_header   Connection "";
            proxy_http_version 1.1;
            proxy_set_header        Host            $host;
            proxy_set_header        X-Real-IP       $remote_addr;
            proxy_set_header        X-Forwarded-For $proxy_add_x_forwarded_for;
            proxy_set_header        X-Forwarded-Proto https;
        }
    }

 /etc/pki/tls/certs/server.crt:
   mode: "000400"
   owner: root
   group: root
   content: |
     -----BEGIN CERTIFICATE-----
     certificate file contents
     -----END CERTIFICATE-----

 /etc/pki/tls/certs/server.key:
   mode: "000400"
   owner: root
   group: root
   content: |
     -----BEGIN RSA PRIVATE KEY-----
     private key contents # See note below.
     -----END RSA PRIVATE KEY-----
```

Kunci `files` tersebut membuat file berikut pada instans:

`/etc/nginx/conf.d/https.conf`

Konfigurasikan server nginx. File ini dimuat ketika layanan nginx dimulai.

`/etc/pki/tls/certs/server.crt`

Membuat file sertifikat pada instans. Ganti *`certificate file contents`* dengan isi sertifikat Anda.

> **Note**
>
> YAML bergantung pada indentasi yang konsisten. Cocokkan tingkat indentasi saat mengganti konten dalam file konfigurasi contoh dan pastikan bahwa editor teks Anda menggunakan spasi, bukan karakter tab, untuk indentasi.

Jika Anda memiliki sertifikat menengah, sertakan sertifikat tersebut di `server.crt` setelah sertifikat situs Anda.

```
    -----BEGIN CERTIFICATE-----
certificate file contents
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
first intermediate certificate
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
second intermediate certificate
-----END CERTIFICATE-----
```

`/etc/pki/tls/certs/server.key`

Membuat file kunci pribadi pada instans. Ganti *`private key contents`* dengan isi kunci pribadi yang digunakan untuk membuat permintaan sertifikat atau sertifikat yang ditandatangani sendiri.

> **Note**
>
> Hindari melakukan file konfigurasi yang berisi kunci pribadi Anda ke kontrol sumber. Setelah Anda menguji konfigurasi dan mengonfirmasi bahwa konfigurasi berfungsi, simpan kunci pribadi Anda di Amazon S3 dan ubah konfigurasi untuk mengunduhnya selama deployment. Untuk instruksi, lihat Menyimpan kunci pribadi dengan aman di Amazon S3.

Pada lingkungan instans tunggal, Anda juga harus mengubah grup keamanan instans untuk mengizinkan lalu lintas pada port 443. File konfigurasi berikut mengambil ID grup keamanan menggunakan AWS CloudFormation fungsi dan menambahkan aturan ke dalamnya.

Example .ebextensions/ .config https-instance-single

```
Resources:
  sslSecurityGroupIngress:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: {"Fn::GetAtt" : ["AWSEBSecurityGroup", "GroupId"]}
      IpProtocol: tcp
      ToPort: 443
      FromPort: 443
      CidrIp: 0.0.0.0/0
```

Untuk lingkungan yang seimbang beban, Anda mengonfigurasi penyeimbang beban untuk meneruskan lalu lintas aman melalui yang tidak tersentuh, atau mendekripsi dan mengenkripsi ulang untuk enkripsi. end-to-end

Mengakhiri HTTPS pada EC2 instance yang menjalankan PHP

Untuk jenis kontainer PHP, Anda menggunakan file konfigurasi untuk mengaktifkan Apache HTTP Server untuk menggunakan HTTPS.

Tambahkan snippet berikut ke file konfigurasi Anda, menggantikan sertifikat dan materi kunci pribadi seperti yang diinstruksikan, dan simpan snippet di direktori `.ebextensions` paket sumber Anda.

File konfigurasi melakukan tugas berikut:

- Kunci `packages` tersebut menggunakan yum untuk memasang `mod24_ssl`.
- Kunci `files` tersebut membuat file berikut pada instans:

  `/etc/httpd/conf.d/ssl.conf`

  Mengonfigurasi server Apache. File ini memuat ketika layanan Apache dimulai.

  `/etc/pki/tls/certs/server.crt`

  Membuat file sertifikat pada instans. Ganti *`certificate file contents`* dengan isi sertifikat Anda.

> **Note**
>
> YAML bergantung pada indentasi yang konsisten. Cocokkan tingkat indentasi saat mengganti konten dalam file konfigurasi contoh dan pastikan bahwa editor teks Anda menggunakan spasi, bukan karakter tab, untuk indentasi.

Jika Anda memiliki sertifikat menengah, sertakan sertifikat tersebut di `server.crt` setelah sertifikat situs Anda.

```
    -----BEGIN CERTIFICATE-----
certificate file contents
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
first intermediate certificate
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
second intermediate certificate
-----END CERTIFICATE-----
```

`/etc/pki/tls/certs/server.key`

Membuat file kunci pribadi pada instans. Ganti *`private key contents`* dengan isi kunci pribadi yang digunakan untuk membuat permintaan sertifikat atau sertifikat yang ditandatangani sendiri.

Example .ebextensions/https-instance.config

```
packages:
  yum:
    mod24_ssl : []

files:
  /etc/httpd/conf.d/ssl.conf:
    mode: "000644"
    owner: root
    group: root
    content: |
      LoadModule ssl_module modules/mod_ssl.so
```

```
      Listen 443
      <VirtualHost *:443>
        <Proxy *>
          Order deny,allow
          Allow from all
        </Proxy>

        SSLEngine             on
        SSLCertificateFile    "/etc/pki/tls/certs/server.crt"
        SSLCertificateKeyFile "/etc/pki/tls/certs/server.key"
        SSLCipherSuite        EECDH+AESGCM:EDH+AESGCM:AES256+EECDH:AES256+EDH
        SSLProtocol           All -SSLv2 -SSLv3
        SSLHonorCipherOrder   On
        SSLSessionTickets     Off

        Header always set Strict-Transport-Security "max-age=63072000;
includeSubdomains; preload"
        Header always set X-Frame-Options DENY
        Header always set X-Content-Type-Options nosniff

        ProxyPass / http://localhost:80/ retry=0
        ProxyPassReverse / http://localhost:80/
        ProxyPreserveHost on
        RequestHeader set X-Forwarded-Proto "https" early

      </VirtualHost>

  /etc/pki/tls/certs/server.crt:
    mode: "000400"
    owner: root
    group: root
    content: |
      -----BEGIN CERTIFICATE-----
      certificate file contents
      -----END CERTIFICATE-----

  /etc/pki/tls/certs/server.key:
    mode: "000400"
    owner: root
    group: root
    content: |
      -----BEGIN RSA PRIVATE KEY-----
      private key contents # See note below.
```

```
      -----END RSA PRIVATE KEY-----
```

> **Note**
>
> Hindari melakukan file konfigurasi yang berisi kunci pribadi Anda ke kontrol sumber. Setelah Anda menguji konfigurasi dan mengonfirmasi bahwa konfigurasi berfungsi, simpan kunci pribadi Anda di Amazon S3 dan ubah konfigurasi untuk mengunduhnya selama deployment. Untuk instruksi, lihat Menyimpan kunci pribadi dengan aman di Amazon S3.

Pada lingkungan instans tunggal, Anda juga harus mengubah grup keamanan instans untuk mengizinkan lalu lintas pada port 443. File konfigurasi berikut mengambil ID grup keamanan menggunakan AWS CloudFormation fungsi dan menambahkan aturan ke dalamnya.

Example .ebextensions/ .config https-instance-single

```
Resources:
  sslSecurityGroupIngress:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: {"Fn::GetAtt" : ["AWSEBSecurityGroup", "GroupId"]}
      IpProtocol: tcp
      ToPort: 443
      FromPort: 443
      CidrIp: 0.0.0.0/0
```

Untuk lingkungan yang seimbang beban, Anda mengonfigurasi penyeimbang beban untuk meneruskan lalu lintas aman melalui yang tidak tersentuh, atau mendekripsi dan mengenkripsi ulang untuk enkripsi. end-to-end

Mengakhiri HTTPS pada EC2 instance yang menjalankan Python

Untuk jenis kontainer Python yang menggunakan Apache HTTP Server dengan Web Server Gateway Interface (WSGI), Anda menggunakan file konfigurasi untuk mengaktifkan Apache HTTP Server untuk menggunakan HTTPS.

Tambahkan snippet berikut ke file konfigurasi Anda, menggantikan sertifikat dan materi kunci pribadi seperti yang diinstruksikan, dan simpan snippet di direktori `.ebextensions` paket sumber Anda. File konfigurasi melakukan tugas berikut:

- Kunci `packages` tersebut menggunakan yum untuk memasang `mod24_ssl`.
- Kunci `files` tersebut membuat file berikut pada instans:

  `/etc/httpd/conf.d/ssl.conf`

  Mengonfigurasi server Apache. Jika aplikasi Anda tidak bernama `application.py`, ganti teks yang disorot dalam nilai `WSGIScriptAlias` dengan jalur lokal untuk aplikasi Anda. Sebagai contoh, aplikasi django mungkin berada di `django/wsgi.py`. Lokasi harus sesuai dengan nilai opsi `WSGIPath` yang Anda tetapkan untuk lingkungan Anda.

  Tergantung pada persyaratan aplikasi Anda, Anda mungkin juga perlu menambahkan direktori lain ke parameter python-path.

  `/etc/pki/tls/certs/server.crt`

  Membuat file sertifikat pada instans. Ganti *`certificate file contents`* dengan isi sertifikat Anda.

  > **Note**
  >
  > YAML bergantung pada indentasi yang konsisten. Cocokkan tingkat indentasi saat mengganti konten dalam file konfigurasi contoh dan pastikan bahwa editor teks Anda menggunakan spasi, bukan karakter tab, untuk indentasi.

  Jika Anda memiliki sertifikat menengah, sertakan sertifikat tersebut di `server.crt` setelah sertifikat situs Anda.

```
    -----BEGIN CERTIFICATE-----
certificate file contents
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
first intermediate certificate
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
second intermediate certificate
-----END CERTIFICATE-----
```

`/etc/pki/tls/certs/server.key`

Membuat file kunci pribadi pada instans. Ganti *`private key contents`* dengan isi kunci pribadi yang digunakan untuk membuat permintaan sertifikat atau sertifikat yang ditandatangani sendiri.

- Kunci `container_commands` tersebut menghentikan layanan httpd setelah semuanya telah dikonfigurasi sehingga layanan menggunakan file `https.conf` dan sertifikat baru.

> **Note**
>
> Contoh tersebut bekerja hanya di lingkungan yang menggunakan platform Python.

Example .ebextensions/https-instance.config

```
packages:
  yum:
    mod24_ssl : []

files:
  /etc/httpd/conf.d/ssl.conf:
    mode: "000644"
    owner: root
    group: root
    content: |
      LoadModule wsgi_module modules/mod_wsgi.so
      WSGIPythonHome /opt/python/run/baselinenv
      WSGISocketPrefix run/wsgi
      WSGIRestrictEmbedded On
      Listen 443
      <VirtualHost *:443>
        SSLEngine on
        SSLCertificateFile "/etc/pki/tls/certs/server.crt"
        SSLCertificateKeyFile "/etc/pki/tls/certs/server.key"

        Alias /static/ /opt/python/current/app/static/
        <Directory /opt/python/current/app/static>
        Order allow,deny
        Allow from all
        </Directory>
```

```
        WSGIScriptAlias / /opt/python/current/app/application.py

        <Directory /opt/python/current/app>
        Require all granted
        </Directory>

        WSGIDaemonProcess wsgi-ssl processes=1 threads=15 display-name=%{GROUP} \
          python-path=/opt/python/current/app \
          python-home=/opt/python/run/venv \
          home=/opt/python/current/app \
          user=wsgi \
          group=wsgi
        WSGIProcessGroup wsgi-ssl

      </VirtualHost>

  /etc/pki/tls/certs/server.crt:
    mode: "000400"
    owner: root
    group: root
    content: |
      -----BEGIN CERTIFICATE-----
      certificate file contents
      -----END CERTIFICATE-----

  /etc/pki/tls/certs/server.key:
    mode: "000400"
    owner: root
    group: root
    content: |
      -----BEGIN RSA PRIVATE KEY-----
      private key contents # See note below.
      -----END RSA PRIVATE KEY-----

container_commands:
  01killhttpd:
    command: "killall httpd"
  02waitforhttpddeath:
    command: "sleep 3"
```

> **Note**
>
> Hindari melakukan file konfigurasi yang berisi kunci pribadi Anda ke kontrol sumber. Setelah Anda menguji konfigurasi dan mengonfirmasi bahwa konfigurasi berfungsi, simpan kunci pribadi Anda di Amazon S3 dan ubah konfigurasi untuk mengunduhnya selama deployment. Untuk instruksi, lihat Menyimpan kunci pribadi dengan aman di Amazon S3.

Pada lingkungan instans tunggal, Anda juga harus mengubah grup keamanan instans untuk mengizinkan lalu lintas pada port 443. File konfigurasi berikut mengambil ID grup keamanan menggunakan AWS CloudFormation fungsi dan menambahkan aturan ke dalamnya.

Example .ebextensions/ .config https-instance-single

```
Resources:
  sslSecurityGroupIngress:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: {"Fn::GetAtt" : ["AWSEBSecurityGroup", "GroupId"]}
      IpProtocol: tcp
      ToPort: 443
      FromPort: 443
      CidrIp: 0.0.0.0/0
```

Untuk lingkungan yang seimbang beban, Anda mengonfigurasi penyeimbang beban untuk meneruskan lalu lintas aman melalui yang tidak tersentuh, atau mendekripsi dan mengenkripsi ulang untuk enkripsi. end-to-end

Mengakhiri HTTPS pada EC2 instance yang menjalankan Ruby

Untuk jenis kontainer Ruby, cara Anda mengaktifkan HTTPS tergantung pada jenis server aplikasi yang digunakan.

Topik

- Konfigurasikan HTTPS untuk Ruby dengan Puma
- Konfigurasikan HTTPS untuk Ruby dengan Passenger

Konfigurasikan HTTPS untuk Ruby dengan Puma

Untuk jenis kontainer Ruby yang menggunakan Puma sebagai server aplikasi, Anda menggunakan file konfigurasi untuk mengaktifkan HTTPS.

Tambahkan snippet berikut ke file konfigurasi Anda, menggantikan sertifikat dan materi kunci pribadi seperti yang diinstruksikan, dan simpan snippet di direktori `.ebextensions` paket sumber Anda. File konfigurasi melakukan tugas berikut:

- Kunci `files` tersebut membuat file berikut pada instans:

  `/etc/nginx/conf.d/https.conf`

  Konfigurasikan server nginx. File ini dimuat ketika layanan nginx dimulai.

  `/etc/pki/tls/certs/server.crt`

  Membuat file sertifikat pada instans. Ganti *`certificate file contents`* dengan isi sertifikat Anda.

  > **Note**
  >
  > YAML bergantung pada indentasi yang konsisten. Cocokkan tingkat indentasi saat mengganti konten dalam file konfigurasi contoh dan pastikan bahwa editor teks Anda menggunakan spasi, bukan karakter tab, untuk indentasi.

  Jika Anda memiliki sertifikat menengah, sertakan sertifikat tersebut di `server.crt` setelah sertifikat situs Anda.

```
    -----BEGIN CERTIFICATE-----
certificate file contents
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
first intermediate certificate
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
second intermediate certificate
-----END CERTIFICATE-----
```

`/etc/pki/tls/certs/server.key`

Membuat file kunci pribadi pada instans. Ganti *`private key contents`* dengan isi kunci pribadi yang digunakan untuk membuat permintaan sertifikat atau sertifikat yang ditandatangani sendiri.

- Kunci `container_commands` tersebut memulai ulang server nginx setelah semuanya dikonfigurasi sehingga server menggunakan file `https.conf` baru.

Example .ebextensions/https-instance.config

```
files:
  /etc/nginx/conf.d/https.conf:
    content: |
      # HTTPS server

      server {
          listen       443;
          server_name  localhost;

          ssl                  on;
          ssl_certificate      /etc/pki/tls/certs/server.crt;
          ssl_certificate_key  /etc/pki/tls/certs/server.key;

          ssl_session_timeout  5m;

          ssl_protocols  TLSv1 TLSv1.1 TLSv1.2;
          ssl_prefer_server_ciphers   on;

          location / {
              proxy_pass  http://my_app;
              proxy_set_header        Host            $host;
              proxy_set_header        X-Forwarded-For $proxy_add_x_forwarded_for;
              proxy_set_header        X-Forwarded-Proto https;
          }

          location /assets {
            alias /var/app/current/public/assets;
            gzip_static on;
            gzip on;
            expires max;
            add_header Cache-Control public;
```

```
        }

        location /public {
          alias /var/app/current/public;
          gzip_static on;
          gzip on;
          expires max;
          add_header Cache-Control public;
        }
      }

  /etc/pki/tls/certs/server.crt:
    content: |
      -----BEGIN CERTIFICATE-----
      certificate file contents
      -----END CERTIFICATE-----

  /etc/pki/tls/certs/server.key:
    content: |
      -----BEGIN RSA PRIVATE KEY-----
      private key contents # See note below.
      -----END RSA PRIVATE KEY-----

container_commands:
  01restart_nginx:
    command: "service nginx restart"
```

> **Note**
>
> Hindari melakukan file konfigurasi yang berisi kunci pribadi Anda ke kontrol sumber. Setelah Anda menguji konfigurasi dan mengonfirmasi bahwa konfigurasi berfungsi, simpan kunci pribadi Anda di Amazon S3 dan ubah konfigurasi untuk mengunduhnya selama deployment. Untuk instruksi, lihat Menyimpan kunci pribadi dengan aman di Amazon S3.

Pada lingkungan instans tunggal, Anda juga harus mengubah grup keamanan instans untuk mengizinkan lalu lintas pada port 443. File konfigurasi berikut mengambil ID grup keamanan menggunakan AWS CloudFormation fungsi dan menambahkan aturan ke dalamnya.

Example .ebextensions/ .config https-instance-single

```
Resources:
```

```
  sslSecurityGroupIngress:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: {"Fn::GetAtt" : ["AWSEBSecurityGroup", "GroupId"]}
      IpProtocol: tcp
      ToPort: 443
      FromPort: 443
      CidrIp: 0.0.0.0/0
```

Untuk lingkungan yang seimbang beban, Anda mengonfigurasi penyeimbang beban untuk meneruskan lalu lintas aman melalui yang tidak tersentuh, atau mendekripsi dan mengenkripsi ulang untuk enkripsi. end-to-end

Konfigurasikan HTTPS untuk Ruby dengan Passenger

Untuk jenis kontainer Ruby yang menggunakan Passenger sebagai server aplikasi, Anda menggunakan file konfigurasi dan file JSON untuk mengaktifkan HTTPS.

Untuk mengonfigurasi HTTPS untuk Ruby dengan Passenger

1. Tambahkan snippet berikut ke file konfigurasi Anda, menggantikan sertifikat dan materi kunci pribadi seperti yang diinstruksikan, dan simpan snippet di direktori `.ebextensions` paket sumber Anda. File konfigurasi melakukan tugas berikut:

   - Kunci `files` tersebut membuat file berikut pada instans:

     `/etc/pki/tls/certs/server.crt`

     Membuat file sertifikat pada instans. Ganti *`certificate file contents`* dengan isi sertifikat Anda.

     > **Note**
     >
     > YAML bergantung pada indentasi yang konsisten. Cocokkan tingkat indentasi saat mengganti konten dalam file konfigurasi contoh dan pastikan bahwa editor teks Anda menggunakan spasi, bukan karakter tab, untuk indentasi.

     Jika Anda memiliki sertifikat menengah, sertakan sertifikat tersebut di `server.crt` setelah sertifikat situs Anda.

```
    -----BEGIN CERTIFICATE-----
  certificate file contents
  -----END CERTIFICATE-----
  -----BEGIN CERTIFICATE-----
  first intermediate certificate
  -----END CERTIFICATE-----
  -----BEGIN CERTIFICATE-----
  second intermediate certificate
  -----END CERTIFICATE-----
```

`/etc/pki/tls/certs/server.key`

Membuat file kunci pribadi pada instans. Ganti *`private key contents`* dengan isi kunci pribadi yang digunakan untuk membuat permintaan sertifikat atau sertifikat yang ditandatangani sendiri.

Example Snippet .ebextensions yang mengonfigurasi HTTPS untuk Ruby dengan Passenger

```
files:
  /etc/pki/tls/certs/server.crt:
    content: |
      -----BEGIN CERTIFICATE-----
      certificate file contents
      -----END CERTIFICATE-----

  /etc/pki/tls/certs/server.key:
    content: |
      -----BEGIN RSA PRIVATE KEY-----
      private key contents # See note below.
      -----END RSA PRIVATE KEY-----
```

> **Note**
>
> Hindari melakukan file konfigurasi yang berisi kunci pribadi Anda ke kontrol sumber. Setelah Anda menguji konfigurasi dan mengonfirmasi bahwa konfigurasi berfungsi, simpan kunci pribadi Anda di Amazon S3 dan ubah konfigurasi untuk mengunduhnya selama deployment. Untuk instruksi, lihat Menyimpan kunci pribadi dengan aman di Amazon S3.

2. Buat file teks dan tambahkan JSON berikut ke file tersebut. Simpan file dalam direktori akar paket sumber Anda dengan nama `passenger-standalone.json`. File JSON ini mengonfigurasi Passenger untuk menggunakan HTTPS.

   > **Important**
   >
   > File JSON ini tidak boleh berisi tanda urutan byte (BOM). Jika JSON berisi tanda urutan byte, perpustakaan Passenger JSON tidak akan membaca file dengan benar dan layanan Passenger tidak akan dimulai.

   Example passenger-standalone.json

   ```
   {
     "ssl" : true,
     "ssl_port" : 443,
     "ssl_certificate" : "/etc/pki/tls/certs/server.crt",
     "ssl_certificate_key" : "/etc/pki/tls/certs/server.key"
   }
   ```

Pada lingkungan instans tunggal, Anda juga harus mengubah grup keamanan instans untuk mengizinkan lalu lintas pada port 443. File konfigurasi berikut mengambil ID grup keamanan menggunakan AWS CloudFormation fungsi dan menambahkan aturan ke dalamnya.

Example .ebextensions/ .config https-instance-single

```
Resources:
  sslSecurityGroupIngress:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: {"Fn::GetAtt" : ["AWSEBSecurityGroup", "GroupId"]}
      IpProtocol: tcp
      ToPort: 443
      FromPort: 443
      CidrIp: 0.0.0.0/0
```

Untuk lingkungan yang seimbang beban, Anda mengonfigurasi penyeimbang beban untuk meneruskan lalu lintas aman melalui yang tidak tersentuh, atau mendekripsi dan mengenkripsi ulang untuk enkripsi. end-to-end

Mengakhiri HTTPS pada EC2 instance yang menjalankan Tomcat

Untuk jenis kontainer Tomcat, Anda menggunakan file konfigurasi untuk mengaktifkan Apache HTTP Server agar menggunakan HTTPS ketika bertindak sebagai proksi terbalik untuk Tomcat.

Tambahkan snippet berikut ke file konfigurasi Anda, menggantikan sertifikat dan materi kunci pribadi seperti yang diinstruksikan, dan simpan snippet di direktori `.ebextensions` paket sumber Anda. File konfigurasi melakukan tugas berikut:

- Kunci `files` tersebut membuat file berikut pada instans:

  `/etc/pki/tls/certs/server.crt`

  Membuat file sertifikat pada instans. Ganti *`certificate file contents`* dengan isi sertifikat Anda.

  > **Note**
  >
  > YAML bergantung pada indentasi yang konsisten. Cocokkan tingkat indentasi saat mengganti konten dalam file konfigurasi contoh dan pastikan bahwa editor teks Anda menggunakan spasi, bukan karakter tab, untuk indentasi.

  `/etc/pki/tls/certs/server.key`

  Membuat file kunci pribadi pada instans. Ganti *`private key contents`* dengan isi kunci pribadi yang digunakan untuk membuat permintaan sertifikat atau sertifikat yang ditandatangani sendiri.

  `/opt/elasticbeanstalk/hooks/appdeploy/post/99_start_httpd.sh`

  Membuat skrip kait pasca-deployment untuk memulai kembali layanan httpd.

Example .ebextensions/https-instance.config

```
files:
  /etc/pki/tls/certs/server.crt:
    mode: "000400"
    owner: root
    group: root
    content: |
```

```
    -----BEGIN CERTIFICATE-----
    certificate file contents
    -----END CERTIFICATE-----

  /etc/pki/tls/certs/server.key:
    mode: "000400"
    owner: root
    group: root
    content: |
      -----BEGIN RSA PRIVATE KEY-----
      private key contents # See note below.
      -----END RSA PRIVATE KEY-----

  /opt/elasticbeanstalk/hooks/appdeploy/post/99_start_httpd.sh:
    mode: "000755"
    owner: root
    group: root
    content: |
      #!/usr/bin/env bash
      sudo service httpd restart
```

Anda juga harus mengonfigurasi server proksi lingkungan Anda untuk mendengarkan port 443. Konfigurasi Apache 2.4 berikut menambahkan pendengar pada port 443. Untuk mempelajari informasi lebih lanjut, lihat Mengonfigurasi server proksi.

Example . ebextensions/httpd/conf.d/ssl.conf

```
Listen 443
<VirtualHost *:443>
  ServerName server-name
  SSLEngine on
  SSLCertificateFile "/etc/pki/tls/certs/server.crt"
  SSLCertificateKeyFile "/etc/pki/tls/certs/server.key"

  <Proxy *>
    Require all granted
  </Proxy>
  ProxyPass / http://localhost:8080/ retry=0
  ProxyPassReverse / http://localhost:8080/
  ProxyPreserveHost on

  ErrorLog /var/log/httpd/elasticbeanstalk-ssl-error_log
```

```
</VirtualHost>
```

Vendor sertifikat Anda mungkin menyertakan sertifikat perantara yang dapat Anda instal untuk kompatibilitas yang lebih baik dengan klien seluler. Konfigurasikan Apache dengan paket otoritas sertifikat menengah (CA) dengan menambahkan hal berikut ke file konfigurasi SSL Anda (lihat Memperluas dan mengganti konfigurasi Apache default - Amazon Linux AMI () AL1 untuk lokasi):

- Pada konten file `ssl.conf`, tentukan file rantai:

```
SSLCertificateKeyFile "/etc/pki/tls/certs/server.key"
SSLCertificateChainFile "/etc/pki/tls/certs/gd_bundle.crt"
SSLCipherSuite        EECDH+AESGCM:EDH+AESGCM:AES256+EECDH:AES256+EDH
```

- Tambahkan entri baru ke kunci `files` dengan konten sertifikat menengah:

```
files:
  /etc/pki/tls/certs/gd_bundle.crt:
    mode: "000400"
    owner: root
    group: root
    content: |
      -----BEGIN CERTIFICATE-----
      First intermediate certificate
      -----END CERTIFICATE-----
      -----BEGIN CERTIFICATE-----
      Second intermediate certificate
      -----END CERTIFICATE-----
```

> **Note**
>
> Hindari melakukan file konfigurasi yang berisi kunci pribadi Anda ke kontrol sumber. Setelah Anda menguji konfigurasi dan mengonfirmasi bahwa konfigurasi berfungsi, simpan kunci pribadi Anda di Amazon S3 dan ubah konfigurasi untuk mengunduhnya selama deployment. Untuk instruksi, lihat Menyimpan kunci pribadi dengan aman di Amazon S3.

Pada lingkungan instans tunggal, Anda juga harus mengubah grup keamanan instans untuk mengizinkan lalu lintas pada port 443. File konfigurasi berikut mengambil ID grup keamanan menggunakan AWS CloudFormation fungsi dan menambahkan aturan ke dalamnya.

Example .ebextensions/ .config https-instance-single

```
Resources:
  sslSecurityGroupIngress:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: {"Fn::GetAtt" : ["AWSEBSecurityGroup", "GroupId"]}
      IpProtocol: tcp
      ToPort: 443
      FromPort: 443
      CidrIp: 0.0.0.0/0
```

Untuk lingkungan yang seimbang beban, Anda mengonfigurasi penyeimbang beban untuk meneruskan lalu lintas aman melalui yang tidak tersentuh, atau mendekripsi dan mengenkripsi ulang untuk enkripsi. end-to-end

Mengakhiri HTTPS di EC2 instans Amazon yang menjalankan .NET Core di Linux

Untuk jenis kontainer .NET Core pada Linux, Anda mengaktifkan HTTPS dengan file konfigurasi `.ebextensions`, dan file konfigurasi nginx yang mengonfigurasi server nginx untuk menggunakan HTTPS.

Tambahkan snippet berikut ke file konfigurasi Anda, menggantikan sertifikat dan placeholder kunci pribadi seperti yang diperintahkan, dan simpan snippet dalam direktori `.ebextensions`. File konfigurasi melakukan tugas berikut:

- Kunci `files` tersebut membuat file berikut pada instans:

  `/etc/pki/tls/certs/server.crt`

  Membuat file sertifikat pada instans. Ganti *`certificate file contents`* dengan isi sertifikat Anda.

> **Note**
>
> YAML bergantung pada indentasi yang konsisten. Cocokkan tingkat indentasi saat mengganti konten dalam file konfigurasi contoh dan pastikan bahwa editor teks Anda menggunakan spasi, bukan karakter tab, untuk indentasi.

Jika Anda memiliki sertifikat menengah, sertakan sertifikat tersebut di `server.crt` setelah sertifikat situs Anda.

```
    -----BEGIN CERTIFICATE-----
certificate file contents
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
first intermediate certificate
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
second intermediate certificate
-----END CERTIFICATE-----
```

`/etc/pki/tls/certs/server.key`

Membuat file kunci pribadi pada instans. Ganti *`private key contents`* dengan isi kunci pribadi yang digunakan untuk membuat permintaan sertifikat atau sertifikat yang ditandatangani sendiri.

- Kunci `container_commands` tersebut memulai ulang serverr nginx setelah semuanya dikonfigurasi sehingga server memuat file konfigurasi nginx.

Example .ebextensions/https-instance.config

```
files:
  /etc/pki/tls/certs/server.crt:
    content: |
      -----BEGIN CERTIFICATE-----
      certificate file contents
      -----END CERTIFICATE-----

  /etc/pki/tls/certs/server.key:
    content: |
```

```
      -----BEGIN RSA PRIVATE KEY-----
      private key contents # See note below.
      -----END RSA PRIVATE KEY-----

container_commands:
  01restart_nginx:
    command: "systemctl restart nginx"
```

> **Note**
>
> Hindari melakukan file konfigurasi yang berisi kunci pribadi Anda ke kontrol sumber. Setelah Anda menguji konfigurasi dan mengonfirmasi bahwa konfigurasi berfungsi, simpan kunci pribadi Anda di Amazon S3 dan ubah konfigurasi untuk mengunduhnya selama deployment. Untuk instruksi, lihat Menyimpan kunci pribadi dengan aman di Amazon S3.

Tempatkan hal berikut dalam sebuah file dengan ekstensi `.conf` di `.platform/nginx/conf.d/` direktori paket sumber Anda (misalnya, `.platform/nginx/conf.d/https.conf`). Ganti *`app_port`* dengan nomor port yang didengarkan aplikasi Anda. Contoh ini mengonfigurasi server nginx untuk mendengarkan pada port 443 menggunakan SSL. Untuk informasi selengkapnya tentang file konfigurasi ini pada .NET Core di platform Linux, lihat the section called “Server proksi”.

Example . platform/nginx/conf.d/https.conf

```
# HTTPS server

server {
    listen       443 ssl;
    server_name  localhost;

    ssl_certificate      /etc/pki/tls/certs/server.crt;
    ssl_certificate_key  /etc/pki/tls/certs/server.key;

    ssl_session_timeout  5m;

    ssl_protocols  TLSv1 TLSv1.1 TLSv1.2;
    ssl_prefer_server_ciphers   on;

    location / {
        proxy_pass  http://localhost:app_port;
        proxy_set_header   Connection "";
```

```
        proxy_http_version 1.1;
        proxy_set_header        Host            $host;
        proxy_set_header        X-Real-IP       $remote_addr;
        proxy_set_header        X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header        X-Forwarded-Proto https;
    }
}
```

Pada lingkungan instans tunggal, Anda juga harus mengubah grup keamanan instans untuk mengizinkan lalu lintas pada port 443. File konfigurasi berikut mengambil ID grup keamanan menggunakan AWS CloudFormation fungsi dan menambahkan aturan ke dalamnya.

Example .ebextensions/ .config https-instance-single

```
Resources:
  sslSecurityGroupIngress:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: {"Fn::GetAtt" : ["AWSEBSecurityGroup", "GroupId"]}
      IpProtocol: tcp
      ToPort: 443
      FromPort: 443
      CidrIp: 0.0.0.0/0
```

Untuk lingkungan yang seimbang beban, Anda mengonfigurasi penyeimbang beban untuk meneruskan lalu lintas aman melalui yang tidak tersentuh, atau mendekripsi dan mengenkripsi ulang untuk enkripsi. end-to-end

Mengakhiri HTTPS di EC2 instans Amazon yang menjalankan .NET

File konfigurasi berikut membuat dan menjalankan PowerShell skrip Windows yang melakukan tugas-tugas berikut:

- Memeriksa sertifikat HTTPS yang ada yang mengikat ke port 443.
- Mendapat sertifikat PFX dari bucket Amazon S3.

> **Note**
>
> Tambahkan `AmazonS3ReadOnlyAccess` kebijakan `aws-elasticbeanstalk-ec2-role` untuk mengakses sertifikat SSL di bucket Amazon S3.

- Mendapat kata sandi dari AWS Secrets Manager.

> **Note**
>
> Tambahkan pernyataan `aws-elasticbeanstalk-ec2-role` yang memungkinkan `secretsmanager:GetSecretValue` tindakan untuk rahasia yang berisi kata sandi sertifikat

- Menginstal sertifikat.
- Mengikat sertifikat ke port 443.

> **Note**
>
> Untuk menghapus titik akhir HTTP (port 80), masukkan perintah `Remove-WebBinding` di bawah bagian Hapus pengikatan HTTP pada contoh.

Example .ebextensions/ .config https-instance-dotnet

```
files:
  "C:\\certs\\install-cert.ps1":
    content: |
      import-module webadministration
      ## Settings - replace the following values with your own
      $bucket = "amzn-s3-demo-bucket"  ## S3 bucket name
      $certkey = "example.com.pfx"    ## S3 object key for your PFX certificate
      $secretname = "example_secret"  ## AWS Secrets Manager name for a secret that
 contains the certificate's password
      ##

      # Set variables
      $certfile = "C:\cert.pfx"
      $pwd = Get-SECSecretValue -SecretId $secretname | select -expand SecretString

      # Clean up existing binding
      if ( Get-WebBinding "Default Web Site" -Port 443 ) {
        Echo "Removing WebBinding"
        Remove-WebBinding -Name "Default Web Site" -BindingInformation *:443:
      }
      if ( Get-Item -path IIS:\SslBindings\0.0.0.0!443 ) {
        Echo "Deregistering WebBinding from IIS"
```

```
        Remove-Item -path IIS:\SslBindings\0.0.0.0!443
      }

      # Download certificate from S3
      Read-S3Object -BucketName $bucket -Key $certkey -File $certfile

      # Install certificate
      Echo "Installing cert..."
      $securepwd = ConvertTo-SecureString -String $pwd -Force -AsPlainText
      $cert = Import-PfxCertificate -FilePath $certfile cert:\localMachine\my -Password
 $securepwd

      # Create site binding
      Echo "Creating and registering WebBinding"
      New-WebBinding -Name "Default Web Site" -IP "*" -Port 443 -Protocol https
      New-Item -path IIS:\SslBindings\0.0.0.0!443 -value $cert -Force

      ## Remove the HTTP binding
      ## (optional) Uncomment the following line to unbind port 80
      # Remove-WebBinding -Name "Default Web Site" -BindingInformation *:80:
      ##

      # Update firewall
      netsh advfirewall firewall add rule name="Open port 443" protocol=TCP
 localport=443 action=allow dir=OUT

commands:
  00_install_ssl:
    command: powershell -NoProfile -ExecutionPolicy Bypass -file C:\\certs\\install-
cert.ps1
```

Pada lingkungan instans tunggal, Anda juga harus mengubah grup keamanan instans untuk mengizinkan lalu lintas pada port 443. File konfigurasi berikut mengambil ID grup keamanan menggunakan AWS CloudFormation fungsi dan menambahkan aturan ke dalamnya.

Example .ebextensions/ .config https-instance-single

```
Resources:
  sslSecurityGroupIngress:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: {"Fn::GetAtt" : ["AWSEBSecurityGroup", "GroupId"]}
      IpProtocol: tcp
```

```
    ToPort: 443
    FromPort: 443
    CidrIp: 0.0.0.0/0
```

Untuk lingkungan yang seimbang beban, Anda mengonfigurasi penyeimbang beban untuk meneruskan lalu lintas aman melalui yang tidak tersentuh, atau mendekripsi dan mengenkripsi ulang untuk enkripsi. end-to-end

Mengkonfigurasi end-to-end enkripsi dalam lingkungan Elastic Beanstalk Beanstalk yang seimbang

Mengakhiri koneksi yang aman pada penyeimbang beban dan menggunakan HTTP pada backend mungkin cukup untuk aplikasi Anda. Lalu lintas jaringan antara sumber daya AWS tidak dapat didengarkan oleh instans yang bukan bagian dari koneksi, meskipun mereka berjalan di bawah akun yang sama.

Namun, jika Anda mengembangkan aplikasi yang harus mematuhi peraturan eksternal yang ketat, Anda mungkin diminta untuk mengamankan semua koneksi jaringan. Anda bisa menggunakan konsol Elastic Beanstalk atau file konfigurasi untuk membuat penyeimbang beban lingkungan Elastic Beanstalk Anda terhubung ke instans backend dengan aman untuk memenuhi persyaratan ini. Prosedur berikut berfokus pada file konfigurasi.

Pertama, tambahkan pendengar yang aman ke penyeimbang beban Anda, jika Anda belum melakukannya.

Anda juga harus mengonfigurasi instans di lingkungan Anda untuk mendengarkan port yang aman dan mengakhiri koneksi HTTPS. Konfigurasi bervariasi per platform. Lihat Mengkonfigurasi Pengakhiran HTTPS pada instance untuk instruksi. Anda dapat menggunakan sertifikat yang ditandatangani sendiri untuk EC2 instans tanpa masalah.

Selanjutnya, konfigurasikan pendengar untuk meneruskan lalu lintas menggunakan HTTPS pada port yang aman yang digunakan oleh aplikasi Anda. Gunakan salah satu file konfigurasi berikut, berdasarkan jenis penyeimbang beban yang digunakan oleh lingkungan Anda.

`.ebextensions/https-reencrypt-clb.config`

Gunakan file konfigurasi ini dengan Classic Load Balancer. Selain mengonfigurasi penyeimbang beban, file konfigurasi juga mengubah pemeriksaan kondisi default untuk menggunakan port 443 dan HTTPS, untuk memastikan bahwa penyeimbang beban dapat terhubung dengan aman.

```
option_settings:
  aws:elb:listener:443:
    InstancePort: 443
    InstanceProtocol: HTTPS
  aws:elasticbeanstalk:application:
    Application Healthcheck URL: HTTPS:443/
```

`.ebextensions/https-reencrypt-alb.config`

Gunakan file konfigurasi ini dengan Application Load Balancer.

```
option_settings:
  aws:elbv2:listener:443:
    DefaultProcess: https
    ListenerEnabled: 'true'
    Protocol: HTTPS
  aws:elasticbeanstalk:environment:process:https:
    Port: '443'
    Protocol: HTTPS
```

`.ebextensions/https-reencrypt-nlb.config`

Gunakan file konfigurasi ini dengan Network Load Balancer.

```
option_settings:
  aws:elbv2:listener:443:
    DefaultProcess: https
    ListenerEnabled: 'true'
  aws:elasticbeanstalk:environment:process:https:
    Port: '443'
```

Opsi `DefaultProcess` berikut dinamai demikian karena Application Load Balancers, yang dapat memiliki pendengar nondefault pada port yang sama untuk lalu lintas ke jalur tertentu (lihat Application Load Balancer untuk detail). Untuk Network Load Balancer, opsi menentukan satu-satunya proses target untuk pendengar ini.

Pada contoh ini, kami menamai proses `https` karena proses ini mendengarkan lalu lintas yang aman (HTTPS). Pendengar mengirimkan lalu lintas ke proses pada port yang ditunjuk menggunakan protokol TCP, karena Network Load Balancer hanya bekerja dengan TCP. Tidak apa-apa, karena lalu lintas jaringan untuk HTTP dan HTTPS diimplementasikan di atas TCP.

> **Note**
>
> Konsol EB CLI dan Elastic Beanstalk menerapkan nilai yang direkomendasikan untuk pilihan sebelumnya. Anda harus menghapus pengaturan ini jika Anda ingin menggunakan file konfigurasi untuk mengonfigurasi hal yang sama. Lihat Nilai yang disarankan untuk rincian selengkapnya.

Dalam tugas berikutnya, Anda perlu mengubah grup keamanan penyeimbang beban untuk mengizinkan lalu lintas. Bergantung pada Amazon Virtual Private Cloud (Amazon VPC) tempat Anda meluncurkan lingkungan—VPC default atau VPC khusus—grup keamanan penyeimbang beban akan bervariasi. Pada VPC default, Elastic Load Balancing menyediakan grup keamanan default yang dapat digunakan semua penyeimbang beban. Pada Amazon VPC yang Anda buat, Elastic Beanstalk membuat grup keamanan untuk digunakan penyeimbang beban.

Untuk mendukung kedua skenario, Anda dapat membuat grup keamanan dan memerintahkan Elastic Beanstalk untuk menggunakannya. File konfigurasi berikut membuat grup keamanan dan melampirkannya ke penyeimbang beban.

`.ebextensions/https-lbsecuritygroup.config`

```
option_settings:
  # Use the custom security group for the load balancer
  aws:elb:loadbalancer:
    SecurityGroups: '`{ "Ref" : "loadbalancersg" }`'
    ManagedSecurityGroup: '`{ "Ref" : "loadbalancersg" }`'

Resources:
  loadbalancersg:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: load balancer security group
      VpcId: vpc-########
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort: 443
          ToPort: 443
          CidrIp: 0.0.0.0/0
        - IpProtocol: tcp
          FromPort: 80
          ToPort: 80
```

```
          CidrIp: 0.0.0.0/0
      SecurityGroupEgress:
        - IpProtocol: tcp
          FromPort: 80
          ToPort: 80
          CidrIp: 0.0.0.0/0
```

Ganti teks yang disorot dengan ID VPC default atau khusus. Contoh sebelumnya mencakup masuk dan keluar melalui port 80 untuk mengizinkan koneksi HTTP. Anda dapat menghapus properti tersebut jika Anda hanya ingin mengizinkan koneksi aman.

Terakhir, tambahkan aturan masuk dan keluar yang mengizinkan komunikasi melalui port 443 antara grup keamanan penyeimbang beban dan grup keamanan instans.

`.ebextensions/https-backendsecurity.config`

```
Resources:
  # Add 443-inbound to instance security group (AWSEBSecurityGroup)
  httpsFromLoadBalancerSG:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: {"Fn::GetAtt" : ["AWSEBSecurityGroup", "GroupId"]}
      IpProtocol: tcp
      ToPort: 443
      FromPort: 443
      SourceSecurityGroupId: {"Fn::GetAtt" : ["loadbalancersg", "GroupId"]}
  # Add 443-outbound to load balancer security group (loadbalancersg)
  httpsToBackendInstances:
    Type: AWS::EC2::SecurityGroupEgress
    Properties:
      GroupId: {"Fn::GetAtt" : ["loadbalancersg", "GroupId"]}
      IpProtocol: tcp
      ToPort: 443
      FromPort: 443
      DestinationSecurityGroupId: {"Fn::GetAtt" : ["AWSEBSecurityGroup", "GroupId"]}
```

Melakukan hal ini secara terpisah dari pembuatan grup keamanan memungkinkan Anda untuk membatasi grup keamanan sumber dan tujuan tanpa membuat ketergantungan melingkar.

Setelah Anda menyelesaikan semua tugas sebelumnya, penyeimbang beban terhubung ke instans backend Anda dengan aman menggunakan HTTPS. Penyeimbang beban tidak peduli jika sertifikat

instans Anda ditandatangani sendiri atau dikeluarkan oleh otoritas sertifikat terpercaya, dan akan menerima sertifikat apa pun yang diberikan padanya.

Anda dapat mengubah perilaku ini dengan menambahkan kebijakan untuk penyeimbang beban yang memerintahkannya untuk hanya mempercayai sertifikat tertentu. File konfigurasi berikut membuat dua kebijakan. Salah satu kebijakan menentukan sertifikat publik, dan yang lain memerintahkan penyeimbang beban untuk hanya percaya sertifikat itu untuk koneksi ke instans port 443.

`.ebextensions/https-backendauth.config`

```
option_settings:
  # Backend Encryption Policy
  aws:elb:policies:backendencryption:
    PublicKeyPolicyNames: backendkey
    InstancePorts:  443
  # Public Key Policy
  aws:elb:policies:backendkey:
    PublicKey: |
      -----BEGIN CERTIFICATE-----
      ################################################################
      ################################################################
      ################################################################
      ################################################################
      ################################################
      -----END CERTIFICATE-----
```

Ganti teks yang disorot dengan konten sertifikat publik EC2 instans Anda.

Mengonfigurasi penyeimbang beban lingkungan Anda untuk TCP Passthrough

Jika Anda tidak ingin penyeimbang beban di AWS Elastic Beanstalk lingkungan Anda mendekripsi lalu lintas HTTPS, Anda dapat mengonfigurasi pendengar aman untuk menyampaikan permintaan ke instance backend apa adanya.

> **Important**
>
> Mengkonfigurasi penyeimbang beban untuk menyampaikan lalu lintas HTTPS tanpa mendekripsi itu menghadirkan kerugian. Penyeimbang beban tidak dapat melihat permintaan terenkripsi sehingga tidak dapat mengoptimalkan metrik routing atau respons laporan.

Pertama, konfigurasikan EC2 instance lingkungan Anda untuk menghentikan HTTPS. Uji konfigurasi pada lingkungan instans tunggal untuk memastikan semuanya berfungsi sebelum menambahkan penyeimbang beban ke dalam campuran.

Tambahkan file konfigurasi ke proyek Anda untuk mengonfigurasi pendengar pada port 443 yang melewati paket TCP sebagaimana adanya ke port 443 pada instans backend:

`.ebextensions/https-lb-passthrough.config`

```
option_settings:
  aws:elb:listener:443:
    ListenerProtocol: TCP
    InstancePort: 443
    InstanceProtocol: TCP
```

Pada Amazon Virtual Private Cloud (Amazon VPC) default, Anda juga perlu menambahkan aturan ke grup keamanan instans untuk mengizinkan lalu lintas masuk pada 443 dari penyeimbang beban:

`.ebextensions/https-instance-securitygroup.config`

```
Resources:
  443inboundfromloadbalancer:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: {"Fn::GetAtt" : ["AWSEBSecurityGroup", "GroupId"]}
      IpProtocol: tcp
      ToPort: 443
      FromPort: 443
      SourceSecurityGroupName: { "Fn::GetAtt": ["AWSEBLoadBalancer",
 "SourceSecurityGroup.GroupName"] }
```

Pada VPC khusus, Elastic Beanstalk memperbarui konfigurasi grup keamanan untuk Anda.

Mengonfigurasi HTTP ke pengalihan HTTPS

Topik ini menjelaskan cara menangani lalu lintas HTTP ke aplikasi Anda jika pengguna akhir masih memulainya. Anda melakukan ini dengan mengonfigurasi pengalihan HTTP ke HTTPS, kadang-kadang disebut sebagai memaksa HTTPS.

Untuk mengonfigurasi pengalihan, Anda mengkonfigurasi lingkungan Anda dulu untuk menangani lalu lintas HTTPS. Kemudian Anda mengalihkan lalu lintas HTTP ke HTTPS. Kedua langkah ini dibahas dalam subbagian berikut.

Konfigurasikan lingkungan Anda untuk menangani lalu lintas HTTPS

Bergantung pada konfigurasi penyeimbang beban lingkungan Anda, lakukan salah satu hal berikut:

- Lingkungan yang seimbang dengan beban — Konfigurasikan penyeimbang beban Anda untuk mengakhiri HTTPS.
- Lingkungan instans tunggal — Konfigurasikan aplikasi Anda untuk mengakhiri koneksi HTTPS pada instans. Konfigurasi ini tergantung pada platform lingkungan Anda.

Alihkan lalu lintas HTTP ke HTTPS

Untuk mengarahkan lalu lintas HTTP ke HTTPS untuk aplikasi Anda, Anda dapat mengonfigurasi server web pada instance lingkungan Anda atau Anda dapat mengonfigurasi Application Load Balancer lingkungan.

Konfigurasikan server web instance

Metode ini bekerja pada lingkungan server web apa pun. Konfigurasikan server web di EC2 instans Amazon Anda untuk merespons lalu lintas HTTP dengan status respons pengalihan HTTP.

Konfigurasi ini tergantung pada platform lingkungan Anda. Temukan folder untuk platform Anda di `https-redirect`koleksi aktif GitHub, dan gunakan contoh file konfigurasi di folder itu.

Jika lingkungan Anda menggunakan pemeriksaan kondisi Elastic Load Balancing, penyeimbang beban mengharapkan instans yang sehat untuk merespon pesan pemeriksaan kondisi HTTP dengan respon HTTP 200 (OK). Oleh karena itu, server web Anda tidak boleh mengalihkan pesan ini ke HTTPS. Contoh file konfigurasi di `https-redirect` menangani persyaratan ini dengan benar.

Konfigurasikan penyeimbang beban

Metode ini berfungsi jika Anda memiliki lingkungan load-balanced yang menggunakan Application Load Balancer. Application Load Balancer dapat mengirim respons pengalihan saat lalu lintas HTTP masuk. Dalam hal ini, Anda tidak perlu mengonfigurasi pengalihan instans lingkungan Anda.

Kami memiliki dua contoh file konfigurasi GitHub yang menunjukkan cara mengkonfigurasi Application Load Balancer untuk pengalihan.

- File konfigurasi `alb-http-to-https-redirection-full.config` tersebut membuat pendengar HTTPS pada port 443, dan memodifikasi pendengar port 80 default untuk mengalihkan lalu lintas HTTP masuk ke HTTPS.

- File `alb-http-to-https-redirection.config`konfigurasi mengharapkan 443 listener didefinisikan. Untuk mendefinisikannya, Anda dapat menggunakan ruang nama konfigurasi Elastic Beanstalk standar, atau konsol Elastic Beanstalk. Kemudian file konfigurasi menangani modifikasi port 80 pendengar untuk pengalihan.

Platform Elastic Beanstalk

AWS Elastic Beanstalk menyediakan berbagai platform tempat Anda dapat membangun aplikasi Anda. Anda mendesain aplikasi web untuk salah satu platform ini, dan Elastic Beanstalk men-deploy kode Anda ke versi platform yang Anda pilih untuk membuat lingkungan aplikasi aktif.

Elastic Beanstalk menyediakan platform untuk bahasa pemrograman yang berbeda, server aplikasi, dan kontainer Docker. Beberapa platform memiliki beberapa versi yang didukung secara bersamaan.

Topik

- Glosarium Platform Elastic Beanstalk
- Model tanggung jawab bersama untuk pemeliharaan platform Elastic Beanstalk
- Kebijakan dukungan platform Elastic Beanstalk
- Jadwal rilis platform Elastic Beanstalk
- Platform yang didukung Elastic Beanstalk
- Platform Elastic Beanstalk Linux
- Memperluas platform Linux Elastic Beanstalk

Glosarium Platform Elastic Beanstalk

Berikut ini adalah istilah kunci yang terkait dengan AWS Elastic Beanstalk platform dan siklus hidupnya.

Waktu Aktif

: Perangkat lunak waktu aktif khusus bahasa pemrograman (framework, library, interpreter, vm, dll) diperlukan untuk menjalankan kode aplikasi Anda.

Komponen Elastic Beanstalk

: Komponen perangkat lunak yang Elastic Beanstalk tambahkan ke platform untuk mengaktifkan fungsionalitas Elastic Beanstalk. Misalnya, agen kondisi yang ditingkatkan diperlukan untuk mengumpulkan dan melaporkan informasi kesehatan.

Platform

Kombinasi dari sistem operasi (OS), waktu aktif, server web, server aplikasi, dan komponen Elastic Beanstalk. Platform menyediakan komponen yang tersedia untuk menjalankan aplikasi Anda.

Versi platform

Kombinasi versi khusus dari sistem operasi (OS), waku aktif, server web, server aplikasi, dan komponen Elastic Beanstalk. Anda membuat lingkungan Elastic Beanstalk berdasarkan versi platform dan men-deploy aplikasi Anda untuk itu.

Versi platform memiliki nomor versi semantik dari formulir X.Y.Z, dimana X adalah versi utama, Y adalah versi minor, dan Z adalah versi patch.

Versi platform dapat berada di salah satu status berikut:

- Direkomendasikan — Versi platform terbaru di cabang platform yang didukung. Versi ini berisi sebagian besar up-to-date komponen dan direkomendasikan untuk digunakan di lingkungan produksi.
- Tidak Direkomendasikan — Versi platform apa pun yang bukan versi terbaru di cabang platformnya. Meskipun versi ini mungkin tetap berfungsi, kami sangat menyarankan untuk memperbarui ke versi platform terbaru. Anda dapat menggunakan pembaruan platform terkelola untuk membantu tetap up-to-date secara otomatis.

Anda dapat memverifikasi apakah versi platform direkomendasikan menggunakan perintah AWS CLI describe-platform-version dan memeriksa bidangnya. `PlatformLifecycleState`

Cabang platform

Sederet versi platform yang berbagi versi tertentu (biasanya besar) dari beberapa komponen mereka, seperti sistem operasi (OS), runtime, atau komponen Elastic Beanstalk. Misalnya: Python 3.13 berjalan di 64bit Amazon Linux 2023; IIS 10.0 berjalan pada Windows Server 64bit 2025. Cabang platform menerima pembaruan dalam bentuk versi platform baru. Setiap versi platform berturut-turut di cabang adalah pembaruan dari yang sebelumnya.

Versi yang direkomendasikan di setiap cabang platform yang didukung tersedia untuk Anda tanpa syarat untuk pembuatan lingkungan. Versi platform sebelumnya tersedia untuk Anda jika Anda menggunakan lingkungan dengannya pada saat versi platform digantikan oleh versi platform baru. Versi platform sebelumnya tidak memiliki up-to-date komponen terbanyak dan tidak direkomendasikan untuk digunakan.

Cabang platform dapat berada di salah satu status berikut:

- Didukung – Cabang platform saat ini. Ini seluruhnya terdiri dari komponen yang didukung. Komponen yang didukung belum mencapai End of Life (EOL), sebagaimana ditentukan oleh pemasoknya. Pembaruan platform yang sedang berlangsung, dan disarankan untuk digunakan di lingkungan produksi. Untuk daftar cabang platform yang didukung, lihat Platform yang didukung Elastic Beanstalk di panduan Platform AWS Elastic Beanstalk .
- Beta – Sebuah pratinjau, cabang platform pra-rilis. Ini eksperimental di alam. Ini mungkin menerima pembaruan platform yang sedang berlangsung untuk sementara waktu, tetapi tidak memiliki support jangka panjang. Cabang platform beta tidak direkomendasikan untuk digunakan di lingkungan produksi. Gunakan hanya untuk evaluasi. Untuk daftar cabang platform beta, lihat Versi Platform Elastic Beanstalk dalam Beta Publik di panduan Platform AWS Elastic Beanstalk .
- Deprecated — Cabang platform di mana satu atau lebih komponen (seperti runtime atau sistem operasi) mendekati End of Life (EOL) atau telah mencapai EOL, seperti yang ditunjuk oleh pemasok mereka. Sementara cabang platform yang tidak digunakan lagi terus menerima versi platform baru hingga tanggal pensiunnya, komponen yang telah mencapai EOL tidak menerima pembaruan. Misalnya, jika versi runtime mencapai EOL, cabang platform akan ditandai sebagai usang tetapi akan terus menerima pembaruan sistem operasi hingga tanggal pensiun cabang platform. Cabang platform tidak akan terus menerima pembaruan ke versi runtime EOL. Cabang platform yang tidak digunakan lagi tidak disarankan untuk digunakan.
- Pensiunan — Cabang platform yang tidak lagi menerima pembaruan apa pun. Cabang platform yang sudah pensiun tidak tersedia untuk membuat lingkungan Elastic Beanstalk baru menggunakan konsol Elastic Beanstalk. Jika lingkungan Anda menggunakan cabang platform yang sudah pensiun, Anda harus memperbarui ke cabang platform yang didukung untuk terus menerima pembaruan. Cabang platform yang sudah pensiun tidak disarankan untuk digunakan. Untuk detail selengkapnya tentang cabang platform yang sudah pensiun, lihatthe section called “Kebijakan dukungan platform”. Untuk daftar cabang platform yang dijadwalkan pensiun, lihat Pensiunan jadwal cabang platform. Untuk melihat cabang platform pensiunan sebelumnya, lihat Riwayat cabang platform pensiun.

Jika lingkungan Anda menggunakan cabang platform yang tidak digunakan lagi atau sudah pensiun, kami rekomendasikan Anda memperbaruinya ke versi platform di cabang platform yang mendukung. Lihat perinciannya di the section called “Pembaruan platform”.

Anda dapat memverifikasi status cabang platform menggunakan perintah AWS CLI describe-platform-version dan memeriksa bidangnya. `PlatformBranchLifecycleState`

Pembaruan Platform

Rilis versi platform baru yang berisi pembaruan untuk beberapa komponen Platform—OS, waktu aktif, server web, server aplikasi, dan komponen Elastic Beanstalk. Pembaruan platform mengikuti taksonomi versi semantik, dan dapat memiliki tiga tingkatan:

- Pembaruan utama – Pembaruan yang memiliki perubahan yang tidak kompatibel dengan versi platform yang ada. Anda mungkin perlu memodifikasi aplikasi Anda agar berjalan dengan benar pada versi utama yang baru. Sebuah update besar memiliki nomor versi platform utama yang baru.
- Pembaruan kecil — Pembaruan yang memiliki perubahan yang kompatibel dengan versi platform yang ada dalam banyak kasus. Tergantung pada aplikasi Anda, Anda mungkin perlu memodifikasi aplikasi Anda agar berjalan dengan benar pada versi minor baru. Pembaruan kecil memiliki nomor versi platform kecil yang baru.
- Pembaruan patch – Pembaruan yang terdiri dari rilis pemeliharaan (perbaikan bug, pembaruan keamanan, dan peningkatan kinerja) yang kompatibel dengan versi platform yang ada. Pembaruan patch memiliki nomor versi platform patch baru.

Pembaruan yang Terkelola

Fitur Elastic Beanstalk yang secara otomatis menerapkan patch dan pembaruan kecil untuk sistem operasi (OS), waktu aktif, server web, server aplikasi, dan komponen Elastic Beanstalk untuk versi platform yang didukung Elastic Beanstalk. Pembaruan yang terkelola menerapkan versi platform yang lebih baru di cabang platform yang sama untuk lingkungan Anda. Anda dapat mengonfigurasi pembaruan terkelola untuk menerapkan hanya pembaruan patch, atau minor dan patch update. Anda juga dapat menonaktifkan pembaruan terkelola sepenuhnya.

Untuk informasi selengkapnya, lihat Pembaruan platform yang dikelola.

Model tanggung jawab bersama untuk pemeliharaan platform Elastic Beanstalk

AWS dan pelanggan kami berbagi tanggung jawab untuk mencapai tingkat keamanan dan kepatuhan komponen perangkat lunak yang tinggi. Model bersama ini mengurangi beban operasional Anda.

Untuk detailnya, lihat Model Tanggung Jawab AWS Bersama.

AWS Elastic Beanstalk membantu Anda melakukan sisi Anda dari model tanggung jawab bersama dengan menyediakan fitur pembaruan terkelola. Fitur ini secara otomatis menerapkan patch dan

minor update untuk versi platform yang didukung Elastic Beanstalk. Jika pembaruan yang terkelola gagal, Elastic Beanstalk memberitahu Anda tentang kegagalan untuk memastikan bahwa Anda menyadari hal itu dan dapat mengambil tindakan segera.

Untuk informasi selengkapnya, lihat Pembaruan platform yang dikelola.

Selain itu, Elastic Beanstalk melakukan hal berikut:

- Mempublikasikan kebijakan dukungan platform dan jadwal masa pensiun untuk 12 bulan mendatang.
- Rilis pembaruan patch, kecil, dan utama dari sistem operasi (OS), waktu aktif, server aplikasi, dan komponen server web biasanya dalam 30 hari sejak ketersediaannya. Elastic Beanstalk bertanggung jawab untuk membuat pembaruan komponen Elastic Beanstalk yang ada pada versi platform yang didukung. Semua pembaruan lainnya datang langsung dari pemasok mereka (pemilik atau komunitas).

Kami mengumumkan semua pembaruan pada platform kami yang didukung dalam catatan rilis kami di panduan Catatan AWS Elastic Beanstalk Rilis. Kami juga menyediakan daftar semua platform yang didukung dan komponennya, bersama dengan riwayat platform, dalam panduan AWS Elastic Beanstalk Platform. Untuk informasi selengkapnya, lihat Platform dan riwayat komponen yang didukung.

Anda bertanggung jawab untuk melakukan hal berikut:

- Perbarui semua komponen yang Anda kontrol (diidentifikasi sebagai Pelanggan dalam Model Tanggung Jawab AWS Bersama). Ini termasuk memastikan keamanan aplikasi Anda, data Anda, dan komponen apa pun yang diperlukan aplikasi Anda dan yang Anda unduh.
- Pastikan lingkungan Elastic Beanstalk Anda berjalan pada versi platform yang didukung, dan migrasikan lingkungan yang berjalan pada versi platform yang sudah pensiun ke versi yang didukung.
- Jika Anda menggunakan image mesin Amazon (AMI) khusus untuk lingkungan Elastic Beanstalk Anda, tambal, rawat, dan uji AMI kustom Anda sehingga tetap terkini dan kompatibel dengan versi platform Elastic Beanstalk yang didukung. Untuk informasi selengkapnya tentang mengelola lingkungan dengan AMI kustom, lihatMenggunakan image mesin Amazon kustom (AMI) di lingkungan Elastic Beanstalk Anda.
- Menyelesaikan semua masalah yang muncul dalam upaya pembaruan terkelola yang gagal dan coba lagi pembaruan.

- Pasang patch OS, waktu aktif, server aplikasi, dan server web sendiri jika Anda memilih keluar dari pembaruan yang terkelola Elastic Beanstalk. Anda dapat melakukannya dengan menerapkan pembaruan platform secara manual atau langsung menambal komponen pada semua sumber daya lingkungan yang relevan.
- Kelola keamanan dan kepatuhan AWS layanan apa pun yang Anda gunakan di luar Elastic Beanstalk sesuai AWS dengan Model Tanggung Jawab Bersama.

Kebijakan dukungan platform Elastic Beanstalk

Elastic Beanstalk mendukung cabang platform yang masih menerima pembaruan kecil dan patch yang sedang berlangsung dari pemasok mereka (pemilik atau komunitas). Untuk definisi lengkap dari syarat terkait, lihat Glosarium Platform Elastic Beanstalk.

Cabang platform pensiun

Ketika komponen cabang platform yang didukung ditandai End of Life (EOL) oleh pemasoknya, Elastic Beanstalk menandai cabang platform sebagai pensiun. Komponen cabang platform meliputi: sistem operasi (OS), versi bahasa runtime, server aplikasi, atau server web.

Setelah cabang platform ditandai sebagai pensiun, kebijakan berikut berlaku:

- Elastic Beanstalk berhenti memberikan pembaruan pemeliharaan, termasuk pembaruan keamanan.
- Elastic Beanstalk tidak lagi memberikan dukungan teknis untuk cabang platform pensiunan.
- Elastic Beanstalk tidak lagi membuat cabang platform tersedia bagi pelanggan Elastic Beanstalk baru untuk diterapkan ke lingkungan baru. Ada masa tenggang 90 hari dari tanggal pensiun yang dipublikasikan untuk pelanggan yang sudah ada dengan lingkungan aktif yang berjalan di cabang platform pensiunan.

> **Note**
>
> Cabang platform yang sudah pensiun tidak akan tersedia di konsol Elastic Beanstalk. Namun, itu akan tersedia melalui AWS CLI, EB CLI dan EB API untuk pelanggan yang memiliki lingkungan yang ada berdasarkan cabang platform pensiunan. Pelanggan yang sudah ada juga dapat menggunakan lingkungan Clone dan konsol lingkungan Rebuild.

Untuk daftar cabang platform yang dijadwalkan pensiun, lihat topik jadwal platform Elastic Beanstalk berikut. Jadwal cabang platform yang pensiun

Untuk informasi selengkapnya tentang apa yang diharapkan ketika cabang platform lingkungan Anda pensiun, lihatFAQ Pensiun Platform.

Melampaui masa tenggang 90 hari

Kebijakan kami untuk cabang platform yang sudah pensiun tidak menghapus akses ke lingkungan atau menghapus sumber daya. Namun, pelanggan yang sudah ada yang menjalankan lingkungan Elastic Beanstalk di cabang platform yang sudah pensiun harus menyadari risiko melakukannya. Lingkungan seperti itu dapat berakhir dalam situasi yang tidak terduga, karena Elastic Beanstalk tidak dapat memberikan pembaruan keamanan, dukungan teknis, atau perbaikan terbaru untuk cabang platform yang sudah pensiun karena pemasok menandai komponen EOL mereka.

Misalnya, kerentanan keamanan yang merugikan dan kritis dapat muncul di lingkungan yang berjalan di cabang platform yang sudah pensiun. Atau tindakan EB API dapat berhenti bekerja untuk lingkungan jika menjadi tidak kompatibel dengan layanan Elastic Beanstalk dari waktu ke waktu. Peluang untuk jenis risiko ini meningkat semakin lama lingkungan di cabang platform pensiun tetap aktif. Untuk terus mendapatkan keuntungan dari keamanan yang kritis, kinerja, dan peningkatan fungsionalitas yang ditawarkan oleh pemasok komponen dalam rilis yang lebih baru, kami sangat menyarankan Anda untuk memperbarui semua lingkungan Elastic Beanstalk Anda ke versi platform yang didukung.

Jika aplikasi Anda mengalami masalah saat berjalan di cabang platform yang sudah pensiun dan Anda tidak dapat memigrasikannya ke platform yang didukung, Anda harus mempertimbangkan alternatif lain. Solusinya termasuk merangkum aplikasi ke dalam gambar Docker untuk menjalankannya sebagai wadah Docker. Ini akan memungkinkan pelanggan untuk menggunakan salah satu solusi Docker kami, seperti platform Elastic AL2 Beanstalk 023/Docker kami, atau layanan berbasis Docker lainnya seperti Amazon AL2 ECS atau Amazon EKS. Alternatif non-Docker termasuk AWS CodeDeploy layanan kami, yang memungkinkan penyesuaian lengkap runtime yang Anda inginkan.

Jadwal rilis platform Elastic Beanstalk

Selain rilis irama bulanan versi cabang platform baru, pemeliharaan rilis kami juga mencakup proses berikut:

- Rilis cabang platform baru — Ini biasanya memperkenalkan versi utama baru dari bahasa run-time, sistem operasi atau server aplikasi.
- Pensiun cabang platform — Kita harus pensiun dari cabang platform ketika salah satu komponennya mencapai End of Life (EOL). Untuk informasi lebih lanjut tentang kebijakan kami untuk cabang pensiunan, lihat Kebijakan dukungan platform Elastic Beanstalk

Topik

- Sumber daya perencanaan
- Rilis cabang platform mendatang
- Jadwal cabang platform yang pensiun
- Sejarah cabang platform pensiunan
- Server pensiunan dan riwayat sistem operasi

Sumber daya perencanaan

Sumber daya berikut dapat membantu Anda merencanakan pemeliharaan dan dukungan untuk aplikasi Anda yang berjalan pada platform Elastic Beanstalk.

- AWS Elastic Beanstalk Panduan Platform — Panduan ini menyediakan daftar komponen terperinci untuk setiap cabang platform kami. Ini juga menyediakan riwayat platform berdasarkan tanggal rilis dengan detail yang sama. Panduan ini dapat memberi tahu Anda ketika komponen tertentu dari cabang platform Anda berubah. Jika aplikasi Anda mulai berperilaku berbeda, Anda juga dapat mereferensikan tanggal kemunculannya di panduan platform untuk melihat apakah ada perubahan platform yang mungkin memengaruhi aplikasi Anda.
- AWS Elastic Beanstalk Catatan Rilis — Catatan Rilis kami mengumumkan semua rilis platform kami, baik minor maupun mayor. Ini termasuk pembaruan platform bulanan kami, rilis keamanan, perbaikan terbaru, dan pengumuman pensiun. Anda dapat berlangganan umpan RSS kami dari dokumentasi Catatan Rilis.

Rilis cabang platform mendatang

Tabel berikut mencantumkan cabang platform Elastic Beanstalk yang akan datang dan tanggal rilis targetnya. Tanggal-tanggal ini bersifat tentatif dan dapat berubah sewaktu-waktu.

Versi runtime/cabang platform	Sistem Operasi	Tanggal rilis target
Corretto 25	Amazon Linux 2023	Oktober 2025
Corretto 25 with Tomcat 11	Amazon Linux 2023	Oktober 2025
Python 3.14	Amazon Linux 2023	November 2025
Node.js 24	Amazon Linux 2023	November 2025
.NET 10	Amazon Linux 2023	Desember 2025
PHP 8.5	Amazon Linux 2023	Januari 2026
Ruby 3.5	Amazon Linux 2023	Februari 2026

Jadwal cabang platform yang pensiun

Tabel berikut ini mencantumkan cabang platform Elastic Beanstalk yang dijadwalkan untuk pensiun, karena beberapa komponennya mencapai End of Life (EOL).

Untuk daftar lebih rinci tentang cabang platform pensiun yang menyertakan komponen spesifiknya, lihat versi platform yang pensiun di panduan AWS Elastic Beanstalk Platform.

Versi runtime/cabang platform	Sistem Operasi	Target tanggal pensiun
Node.js 18 AL2023	Amazon Linux 2023	Juli 31, 2025
Node.js 18 AL2	Amazon Linux 2	Juli 31, 2025

Sejarah cabang platform pensiunan

Tabel berikut mencantumkan cabang platform Elastic Beanstalk yang sudah dalam status pensiun. Anda dapat melihat riwayat terperinci dari cabang platform ini dan komponennya dalam riwayat Platform panduan AWS Elastic Beanstalk Platform.

Amazon Linux 2023 (AL2023)

Versi runtime/cabang platform	Tanggal pensiun		
.NET 6 AL2023	April 8, 2025		

Amazon Linux 2 (AL2)

Versi runtime/cabang platform	Tanggal pensiun		
Corretto 11 with Tomcat 8.5 AL2	Oktober 10, 2024		
Corretto 8 with Tomcat 8.5 AL2	Oktober 10, 2024		
Corretto 11 with Tomcat 7 AL2	Juni 29, 2022		
Corretto 8 with Tomcat 7 AL2	Juni 29, 2022		
Node.js 16 AL2	Oktober 10, 2024		
Node.js 14 AL2	Oktober 10, 2024		
Node.js 12 AL2	Desember 23, 2022		
Node.js 10 AL2	Juni 29, 2022		
PHP 8.0 AL2	Oktober 10, 2024		
PHP 7.4 AL2	9 Juni 2023		
PHP 7.3 AL2	Juni 29, 2022		
PHP 7.2 AL2	Juni 29, 2022		

Versi runtime/cabang platform	Tanggal pensiun		
Python 3.8 AL2	April 8, 2025		
Python 3.7 AL2	Oktober 10, 2024		
Ruby 3.0 AL2	Oktober 10, 2024		
Ruby 2.7 AL2	Oktober 10, 2024		
Ruby 2.6 AL2	Desember 23, 2022		
Ruby 2.5 AL2	Juni 29, 2022		

Amazon Linux AMI (AL1)

Versi runtime/cabang platform	Tanggal pensiun		
Single Container Docker	Juli 18, 2022		
Multicontainer Docker	Juli 18, 2022		
Preconfigured Docker - GlassFish 5.0 with Java 8	Juli 18, 2022		
Go 1	Juli 18, 2022		
Java 8	Juli 18, 2022		
Java 7	Juli 18, 2022		
Java 8 with Tomcat 8.5	Juli 18, 2022		

Versi runtime/cabang platform	Tanggal pensiun		
Java 7 with Tomcat 7	Juli 18, 2022		
Node.js	Juli 18, 2022		
PHP 7.2 - 7.3	Juli 18, 2022		
Python 3.6	Juli 18, 2022		
Ruby 2,4, 2.5, 2.6 with Passenger	Juli 18, 2022		
Ruby 2.4, 2.5, 2.6 with Puma	Juli 18, 2022		
Go 1.3–1.10	31 Oktober 2020		
Java 6	31 Oktober 2020		
Node.js 4.x–8.x	31 Oktober 2020		
PHP 5.4–5.6	31 Oktober 2020		
PHP 7.0–7.1	31 Oktober 2020		
Python 2.6, 2.7, 3.4	31 Oktober 2020		
Ruby 1.9.3	31 Oktober 2020		
Ruby 2.0–2.3	31 Oktober 2020		

> **Note**
>
> Pada tanggal 18 Juli 2022, Elastic Beanstalk menetapkan status semua cabang platform berdasarkan Amazon Linux AMI () untuk pensiun. AL1 Untuk informasi selengkapnya, lihat FAQ Pensiun Platform.

Windows Server

Versi runtime/cabang platform	Tanggal pensiun		
IIS 8.5 berjalan pada 64bit Windows Server (& Core) 2012 R2 versi 0.1.0	Juni 29, 2022		
IIS 8.5 berjalan pada 64bit Windows Server (& Core) 2012 R2 versi 1.2.0	Juni 29, 2022		
IIS 10.0 berjalan pada 64bit Windows Server 2016 (& Core) versi 1.2.0	Juni 29, 2022		
IIS 8 berjalan pada 64bit Windows Server 2012 R1 Platform Branch	Juni 22, 2022		
IIS 8 berjalan pada 64bit Windows Server	Juni 22, 2022		

Versi runtime/cabang platform	Tanggal pensiun		
2012 R1 versi 0.1.0			
IIS 8 berjalan pada 64bit Windows Server 2012 R1 versi 1.2.0	Juni 22, 2022		

> **Note**
>
> Untuk informasi lebih lanjut tentang pensiunnya cabang platform Windows 2012 R2, lihat cabang platform Windows Server 2012 R2 pensiun di Catatan Rilis.AWS Elastic Beanstalk

Server pensiunan dan riwayat sistem operasi

Tabel berikut memberikan riwayat sistem operasi, server aplikasi, dan server web yang tidak lagi didukung oleh platform Elastic Beanstalk. Semua cabang platform yang menggunakan komponen ini sekarang sudah pensiun. Tanggal tersebut mencerminkan tanggal pensiun cabang platform Elastic Beanstalk terakhir yang menyertakan komponen.

Sistem Operasi)

Versi OS	Tanggal masa pensiun platform		
Windows Server 2012 R2 running IIS 8.5	Desember 4, 2023		
Windows Server Core 2012 R2 running IIS 8.5	Desember 4, 2023		

Versi OS	Tanggal masa pensiun platform		
Amazon Linux AMI (AL1)	Juli 18, 2022		
Windows Server 2012 R1	Juni 22, 2022		
Windows Server 2008 R2	28 Oktober 2019		

Server aplikasi

Versi server aplikasi	Tanggal masa pensiun platform		
Tomcat 7	29 Juni 2022 untuk platform Amazon Linux 2 (AL2) 18 Juli 2022 untuk platform Amazon Linux AMI (AL1)		
Tomcat 8	31 Oktober 2020		
Tomcat 6	31 Oktober 2020		

Server web

Versi server web	Tanggal masa pensiun platform		
IIS 8 berjalan pada Windows Server 64bit	Juni 22, 2022		
Apache HTTP Server 2.2	31 Oktober 2020		
Nginx 1.12.2	31 Oktober 2020		

Platform yang didukung Elastic Beanstalk

AWS Elastic Beanstalk menyediakan berbagai platform tempat Anda dapat membangun aplikasi Anda. Anda mendesain aplikasi web untuk salah satu platform ini, dan Elastic Beanstalk men-deploy kode Anda ke versi platform yang Anda pilih untuk membuat lingkungan aplikasi aktif.

Elastic Beanstalk menyediakan sumber daya yang diperlukan untuk menjalankan aplikasi Anda, termasuk satu atau beberapa instans Amazon. EC2 Tumpukan perangkat lunak yang berjalan di EC2 instans Amazon bergantung pada versi platform tertentu yang Anda pilih untuk lingkungan Anda.

Nama tumpukan solusi untuk cabang platform

Anda dapat menggunakan nama tumpukan solusi untuk versi cabang platform tertentu untuk meluncurkan lingkungan dengan EB CLI, ElasticBeanstalk API, atau CLI.AWS Panduan AWS Elastic Beanstalk Platform mencantumkan nama tumpukan solusi di bawah versi cabang platform di bagian Elastic Beanstalk Supported Platforms dan Platform history.

Untuk mengambil semua nama tumpukan solusi yang dapat Anda gunakan untuk membuat lingkungan, gunakan ListAvailableSolutionStacksAPI atau `aws elasticbeanstalk list-available-solution-stacks`di AWS CLI.

Anda dapat menyesuaikan dan mengonfigurasi perangkat lunak yang tergantung pada Anda di platform Anda. Pelajari selengkapnya di Menyesuaikan perangkat lunak pada server Linux dan Menyesuaikan perangkat lunak pada server Windows. Catatan rilis terperinci tersedia untuk rilis terbaru di Catatan Rilis AWS Elastic Beanstalk.

Platform dan riwayat komponen yang didukung

Panduan AWS Elastic Beanstalk Platform mencantumkan semua versi cabang platform saat ini di bagian Platform yang Didukung Elastic Beanstalk. Panduan Platform juga mencantumkan riwayat platform untuk setiap platform, yang mencakup daftar versi platform cabang sebelumnya. Untuk melihat riwayat platform untuk setiap platform, pilih salah satu tautan berikut.

- Docker
- Go
- Jawa SE
- Tomcat (menjalankan Java SE)
- .NET Core di Linux

- .NET di Windows Server
- Node.js
- PHP
- Python
- Ruby

Platform Elastic Beanstalk Linux

Platform Elastic Beanstalk Linux menyediakan sejumlah besar fungsionalitas di luar kotak. Anda dapat memperpanjang platform dalam beberapa cara untuk mendukung aplikasi Anda. Lihat perinciannya di the section called “Memperluas platform Linux”.

Sebagian besar platform yang didukung Elastic Beanstalk didasarkan pada sistem operasi Linux. Secara khusus, platform ini didasarkan pada Amazon Linux, distribusi Linux yang disediakan oleh AWS. Platform Elastic Beanstalk Linux menggunakan instans Amazon Elastic Compute Cloud EC2 (Amazon), dan instance ini menjalankan Amazon Linux.

Topik

- Versi Amazon Linux yang didukung
- Daftar platform Linux Elastic Beanstalk
- Alur kerja deployment instans
- Alur kerja penerapan instans untuk ECS yang berjalan di Amazon Linux 2 dan yang lebih baru
- Alat skrip platform untuk lingkungan Elastic Beanstalk Anda

Versi Amazon Linux yang didukung

AWS Elastic Beanstalk mendukung platform berbasis Amazon Linux 2 dan Amazon Linux 2023.

Untuk informasi selengkapnya tentang Amazon Linux 2 dan Amazon Linux 2023, lihat berikut ini:

- Amazon Linux 2 — Amazon Linux di Panduan EC2 Pengguna Amazon.
- Amazon Linux 2023 - Apa itu Amazon Linux 2023? di Panduan Pengguna Amazon Linux 2023

Untuk rincian tentang versi platform yang didukung, lihat Platform yang didukung Elastic Beanstalk.

> **Note**
>
> Anda dapat memigrasikan aplikasi Anda dari Elastic AL2 Beanstalk AL1 atau cabang platform ke cabang platform 023 yang setara. AL2 Untuk informasi selengkapnya, lihat Memigrasi aplikasi Elastic Beanstalk Linux Anda ke Amazon Linux 2023 atau Amazon Linux 2.

Amazon Linux 2023

AWS mengumumkan ketersediaan umum Amazon Linux 2023 pada bulan Maret 2023. Panduan Pengguna Amazon Linux 2023 merangkum perbedaan utama antara Amazon Linux 2 dan Amazon Linux 2023. Untuk informasi selengkapnya, lihat Membandingkan Amazon Linux 2 dan Amazon Linux 2023 di panduan pengguna.

Ada tingkat kompatibilitas yang tinggi antara platform Elastic Beanstalk Amazon Linux 2 dan Amazon Linux 2023. Meskipun ada beberapa perbedaan yang perlu diperhatikan:

- Layanan Metadata Instans Versi 1 (IMDSv1) - Pengaturan IMDSv1 opsi Nonaktifkan default ke platform 023. `true` AL2 Defaultnya ada `false` di AL2 platform.
- alat instans pkg-repo - Alat ini tidak tersedia untuk lingkungan yang berjalan pada platform 023. pkg-repo AL2 Namun, Anda dapat secara manual menerapkan paket dan pembaruan sistem operasi ke instans AL2 023. Untuk informasi selengkapnya, lihat Mengelola paket dan pembaruan sistem operasi di Panduan Pengguna Amazon Linux 2023.
- HTTPd Konfigurasi Apache - `httpd.conf` File Apache untuk platform AL2 023 memiliki beberapa pengaturan konfigurasi yang berbeda dari yang untuk: AL2
  - Tolak akses ke seluruh sistem file server secara default. Pengaturan ini dijelaskan dalam Lindungi File Server secara Default di halaman Tips Keamanan situs web Apache.
  - Hentikan pengguna untuk mengganti fitur keamanan yang telah Anda konfigurasikan. Konfigurasi menolak akses untuk mengatur `.htaccess` di semua direktori, kecuali yang diaktifkan secara khusus. Pengaturan ini dijelaskan dalam Melindungi Pengaturan Sistem di halaman Tips Keamanan situs web Apache. Tutorial Server HTTP Apache: halaman file.htaccess menyatakan pengaturan ini dapat membantu meningkatkan kinerja.
  - Tolak akses ke file dengan pola nama `.ht*`. Pengaturan ini mencegah klien web melihat `.htaccess` dan `.htpasswd` file.

Anda dapat mengubah salah satu pengaturan konfigurasi di atas untuk lingkungan Anda. Untuk informasi selengkapnya, lihat Mengonfigurasi Apache HTTPD.

Daftar platform Linux Elastic Beanstalk

Daftar berikut menyediakan platform Linux yang didukung Elastic Beanstalk untuk berbagai bahasa pemrograman serta untuk wadah Docker. Elastic Beanstalk menawarkan platform berbasis Amazon Linux 2 dan Amazon Linux 2023 untuk semuanya. Untuk mempelajari lebih lanjut tentang platform, pilih tautan yang sesuai.

- Docker (dan ECS Docker)
- Go
- Tomcat (menjalankan Java SE)
- Jawa SE
- .NET Core di Linux
- Node.js
- PHP
- Python
- Ruby

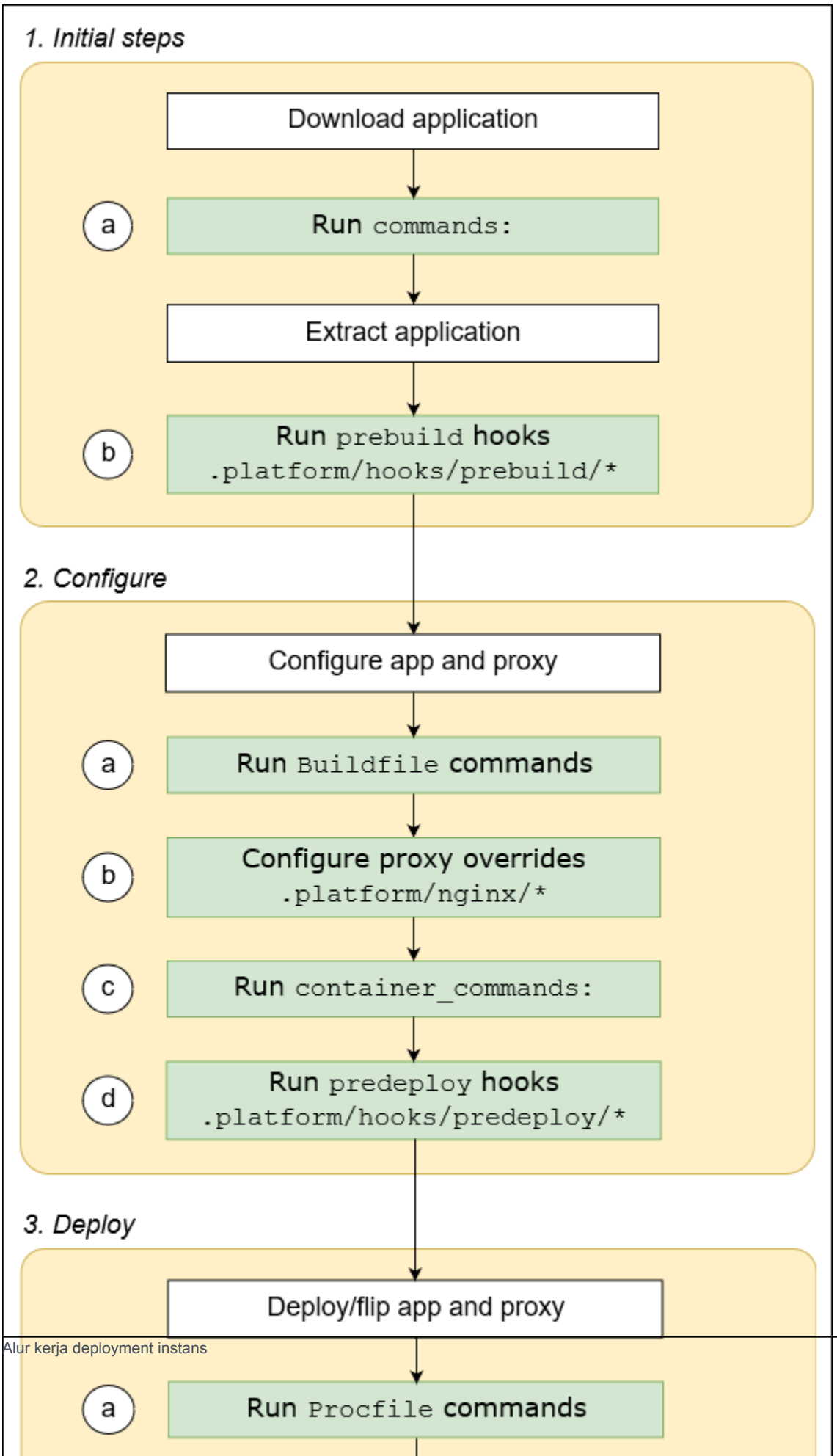
Alur kerja deployment instans

> **Note**
>
> Informasi di bagian ini tidak berlaku untuk ECS yang berjalan di cabang platform Amazon Linux 2 dan Amazon Linux 2023. Untuk informasi lebih lanjut, lihat bagian selanjutnyaAlur kerja penerapan instans untuk ECS yang berjalan di Amazon Linux 2 dan yang lebih baru.

Dengan banyak cara untuk memperluas platform lingkungan Anda, itu berguna untuk mengetahui apa yang terjadi setiap kali Elastic Beanstalk menyediakan instans atau menjalankan deployment ke sebuah instans. Diagram berikut menunjukkan seluruh alur kerja deployment ini. Ini menggambarkan fase yang berbeda dalam penyebaran dan langkah-langkah yang diambil Elastic Beanstalk di setiap fase.

Catatan

- Diagram tidak mewakili set lengkap dari langkah-langkah yang Elastic Beanstalk ambil pada instans lingkungan selama deployment. Kami menyediakan diagram ini untuk ilustrasi, untuk menyediakan Anda dengan urutan dan konteks untuk pelaksanaan penyesuaian Anda.
- Untuk mempermudah, diagram hanya menyebutkan subdirektori hook `.platform/hooks/*` (untuk penerapan aplikasi), dan bukan subdirektori hook `.platform/confighooks/*` (untuk deployment konfigurasi). Hook di subdirektori terakhir berjalan selama langkah yang sama persis seperti hook di subdirektori yang sesuai yang ditunjukkan dalam diagram.


1. Initial steps
Download application
a
Run commands:
Extract application
b
Run prebuild hooks
.platform/hooks/prebuild/*
2. Configure
Configure app and proxy
a
Run Buildfile commands
b
Configure proxy overrides
.platform/nginx/*
c
Run container_commands:
d
Run predeploy hooks
.platform/hooks/predeploy/*
3. Deploy
Deploy/flip app and proxy
a
Run Procfile commands

Daftar berikut detail tahap deployment dan langkah-langkah.

1. Langkah awal

   Elastic Beanstalk mengunduh dan mengekstrak aplikasi Anda. Setelah setiap langkah ini, Elastic Beanstalk menjalankan salah satu langkah yang diperpanjang.

   a. Menjalankan perintah yang ditemukan di bagian perintah: dari file konfigurasi.

   b. Menjalankan file yang dapat dieksekusi yang ditemukan di direktori `.platform/hooks/prebuild` dari paket sumber Anda (`.platform/confighooks/prebuild` untuk deployment konfigurasi).

2. Konfigurasikan

   Elastic Beanstalk mengonfigurasi aplikasi Anda dan server proksi.

   a. Menjalankan perintah yang ditemukan di `Buildfile` di paket sumber Anda.

   b. Salinan file konfigurasi proksi kustom Anda, jika Anda memiliki direktori `.platform/nginx` dari paket sumber Anda, ke lokasi waktu aktif mereka.

   c. Menjalankan perintah yang ditemukan di bagian container_commands: dari file konfigurasi.

   d. Menjalankan file executable yang ditemukan di direktori `.platform/hooks/predeploy` dari paket sumber Anda (`.platform/confighooks/predeploy` untuk deployment konfigurasi).

3. Menyebarkan

   Elastic Beanstalk men-deploy dan menjalankan aplikasi Anda dan server proksi.

   a. Menjalankan perintah yang ditemukan di file `Procfile` dalam paket sumber Anda.

   b. Menjalankan atau menjalankan ulang server proksi dengan file konfigurasi proksi kustom Anda, jika Anda memilikinya.

   c. Menjalankan file executable yang ditemukan di direktori `.platform/hooks/postdeploy` dari paket sumber Anda (`.platform/confighooks/postdeploy` untuk deployment konfigurasi).

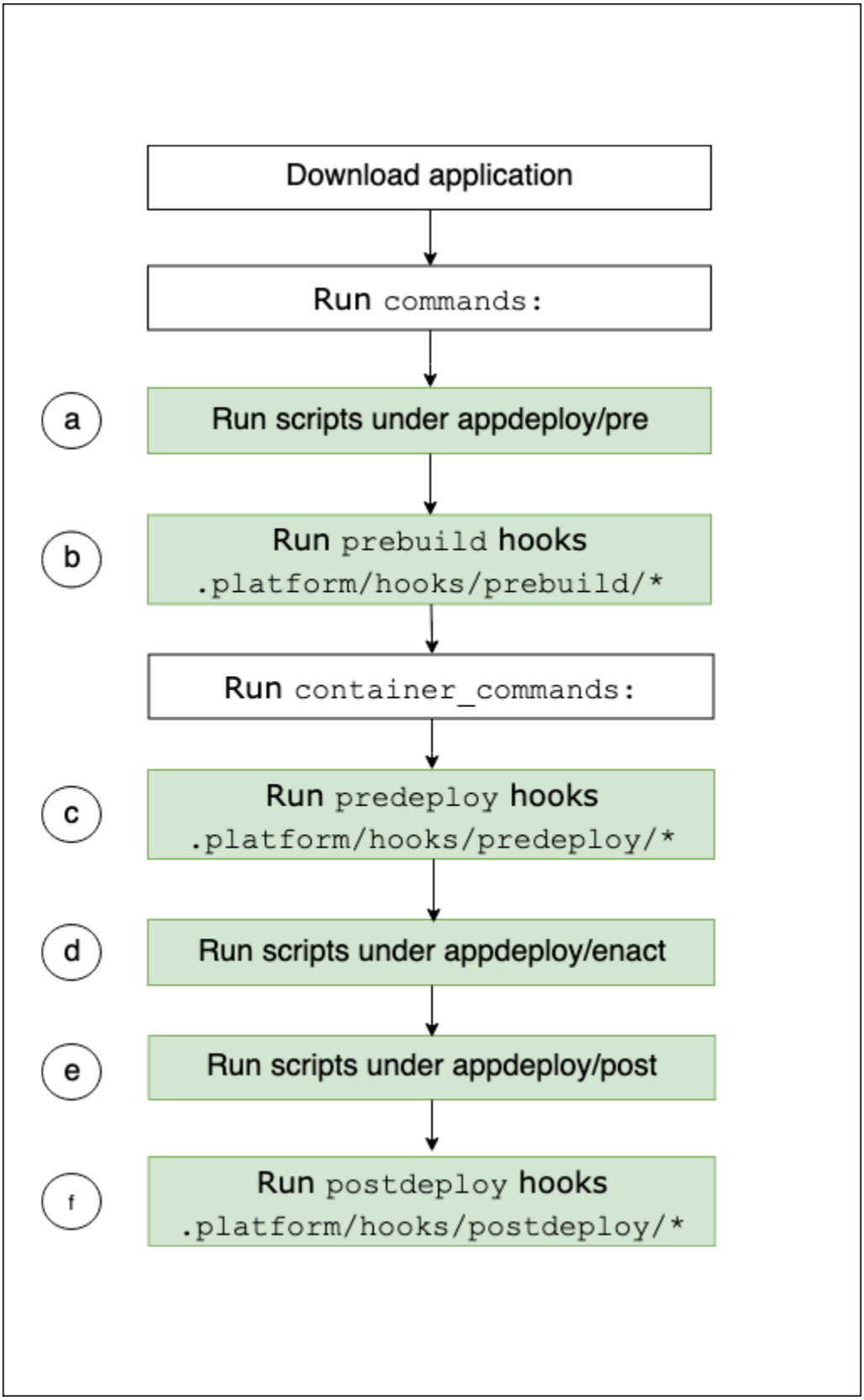
Alur kerja penerapan instans untuk ECS yang berjalan di Amazon Linux 2 dan yang lebih baru

Bagian sebelumnya menjelaskan fitur ekstensibilitas yang didukung di seluruh fase alur kerja penerapan aplikasi. Ada beberapa perbedaan untuk cabang platform Docker ECS yang berjalan di Amazon Linux 2 dan yang lebih baru. Bagian ini menjelaskan bagaimana konsep-konsep tersebut berlaku untuk cabang platform khusus ini.

Dengan banyak cara untuk memperluas platform lingkungan Anda, itu berguna untuk mengetahui apa yang terjadi setiap kali Elastic Beanstalk menyediakan instans atau menjalankan deployment ke sebuah instans. Diagram berikut menunjukkan seluruh alur kerja penerapan ini untuk lingkungan berdasarkan ECS yang berjalan di Amazon Linux 2 dan ECS yang berjalan di cabang platform Amazon Linux 2023. Ini menggambarkan fase yang berbeda dalam penyebaran dan langkah-langkah yang diambil Elastic Beanstalk di setiap fase.

Tidak seperti alur kerja yang dijelaskan di bagian sebelumnya, fase Konfigurasi penerapan tidak mendukung fitur ekstensibilitas berikut: `Buildfile` perintah, `Procfile` perintah, konfigurasi proxy terbalik.

> **Catatan**
>
> - Diagram tidak mewakili set lengkap dari langkah-langkah yang Elastic Beanstalk ambil pada instans lingkungan selama deployment. Kami menyediakan diagram ini untuk ilustrasi, untuk menyediakan Anda dengan urutan dan konteks untuk pelaksanaan penyesuaian Anda.
> - Untuk mempermudah, diagram hanya menyebutkan subdirektori hook `.platform/hooks/*` (untuk penerapan aplikasi), dan bukan subdirektori hook `.platform/confighooks/*` (untuk deployment konfigurasi). Hook di subdirektori terakhir berjalan selama langkah yang sama persis seperti hook di subdirektori yang sesuai yang ditunjukkan dalam diagram.



Daftar berikut merinci langkah-langkah alur kerja penerapan.

a. Menjalankan file yang dapat dieksekusi yang ditemukan di `appdeploy/pre` direktori di bawah. `EBhooksDir`
b. Menjalankan file yang dapat dieksekusi yang ditemukan di direktori `.platform/hooks/prebuild` dari paket sumber Anda (`.platform/confighooks/prebuild` untuk deployment konfigurasi).
c. Menjalankan file yang dapat dieksekusi yang ditemukan di direktori `.platform/hooks/predeploy` dari paket sumber Anda (`.platform/confighooks/predeploy` untuk deployment konfigurasi).
d. Menjalankan file yang dapat dieksekusi yang ditemukan di `appdeploy/enact` direktori di bawah. `EBhooksDir`
e. Menjalankan file yang dapat dieksekusi yang ditemukan di `appdeploy/post` direktori di bawah. `EBhooksDir`
f. Menjalankan file yang dapat dieksekusi yang ditemukan di direktori `.platform/hooks/postdeploy` dari paket sumber Anda (`.platform/confighooks/postdeploy` untuk deployment konfigurasi).

Referensi untuk `EBhooksDir` mewakili jalur direktori kait platform. Untuk mengambil nama jalur direktori gunakan alat skrip get-config pada baris perintah instance lingkungan Anda seperti yang ditunjukkan:

```
$ /opt/elasticbeanstalk/bin/get-config platformconfig -k EBhooksDir
```

Alat skrip platform untuk lingkungan Elastic Beanstalk Anda

Topik ini menjelaskan alat yang AWS Elastic Beanstalk menyediakan lingkungan yang menggunakan platform Amazon Linux. Alat-alat tersebut terletak di EC2 contoh Amazon dari lingkungan Elastic Beanstalk.

get-config

Gunakan `get-config` alat ini untuk mengambil nilai variabel lingkungan teks biasa dan informasi platform dan instance lainnya. Alat ini tersedia di `/opt/elasticbeanstalk/bin/get-config`.

Perintah get-config

Setiap `get-config` perintah alat mengembalikan jenis informasi tertentu. Gunakan sintaks berikut untuk menjalankan perintah dari salah satu alat.

```
$ /opt/elasticbeanstalk/bin/get-config command [ options ]
```

Contoh berikut menjalankan perintah `environment`.

```
$ /opt/elasticbeanstalk/bin/get-config environment -k PORT
```

Bergantung pada perintah dan opsi yang Anda pilih, alat mengembalikan objek (JSON atau YAMAL) dengan pasangan kunci-nilai atau nilai tunggal.

Anda dapat menguji `get-config` dengan menggunakan SSH untuk terhubung ke EC2 instans di lingkungan Elastic Beanstalk Anda.

> **Note**
>
> Ketika Anda menjalankan `get-config` untuk pengujian, beberapa perintah mungkin memerlukan hak pengguna root untuk mengakses informasi dasar. Jika Anda mendapatkan kesalahan izin akses, jalankan perintah lagi di bawah `sudo`.
> Anda tidak perlu menambahkan `sudo` saat menggunakan alat dalam skrip yang Anda deploy ke lingkungan Anda. Elastic Beanstalk menjalankan semua skrip Anda sebagai pengguna root.

Bagian berikut menjelaskan perintah untuk alat.

optionsettings – pilihan konfigurasi

`get-config optionsettings`Perintah mengembalikan objek yang mencantumkan opsi konfigurasi yang disetel di lingkungan dan digunakan oleh platform pada instance lingkungan. Mereka diatur oleh namespace.

```
$ /opt/elasticbeanstalk/bin/get-config optionsettings
{"aws:elasticbeanstalk:application:environment":
{"JDBC_CONNECTION_STRING":""},"aws:elasticbeanstalk:container:tomcat:jvmoptions":{"JVM
 Options":"","Xms":"256m","Xmx":"256m"},"aws:elasticbeanstalk:environment:proxy":
{"ProxyServer":"nginx","StaticFiles":
[""]},"aws:elasticbeanstalk:healthreporting:system":
{"SystemType":"enhanced"},"aws:elasticbeanstalk:hostmanager":
{"LogPublicationControl":"false"}}
```

Untuk mengembalikan nilai opsi konfigurasi tertentu, gunakan pilihan `--namespace` (`-n`) untuk menentukan namespace, dan pilihan `--option-name` (`-o`) untuk menentukan nama pilihan.

```
$ /opt/elasticbeanstalk/bin/get-config optionsettings -
n aws:elasticbeanstalk:container:php:phpini -o memory_limit
256M
```

lingkungan – Sifat lingkungan

`get-config environment`Perintah mengembalikan objek yang berisi daftar properti lingkungan, termasuk baik yang dikonfigurasi pengguna dan disediakan oleh Elastic Beanstalk. Properti yang dikonfigurasi pengguna didefinisikan di konsol sebagai Teks biasa atau dengan namespace opsi konfigurasi. aws:elasticbeanstalk:application:environment

```
$ /opt/elasticbeanstalk/bin/get-config environment
{"JDBC_CONNECTION_STRING":"","RDS_PORT":"3306","RDS_HOSTNAME":"anj9aw1b0tbj6b.cijbpanmxz5u.us-
west-2.rds.amazonaws.com","RDS_USERNAME":"testusername","RDS_DB_NAME":"ebdb","RDS_PASSWORD":"te
```

Misalnya, Elastic Beanstalk menyediakan properti lingkungan untuk menghubungkan ke instans Amazon RDS DB terintegrasi (misalnya,). `RDS_HOSTNAME` Properti koneksi RDS ini muncul dalam output dari. `get-config environment` Namun, mereka tidak muncul di output`get-config optionsettings`. Ini karena mereka tidak diatur dalam opsi konfigurasi.

Untuk mengembalikan properti lingkungan tertentu, gunakan `--key` (`-k`) pilihan untuk menentukan kunci properti.

```
$ /opt/elasticbeanstalk/bin/get-config environment -k TESTPROPERTY
testvalue
```

> **Note**
>
> `get-config`Alat ini tidak dapat mengambil variabel lingkungan yang menyimpan rahasia. Untuk informasi selengkapnya tentang cara mengambil nilai secara terprogram dari penyimpanan rahasia atau parameter, lihat atau. Menggunakan Secrets Manager Menggunakan Systems Manager Parameter Store

kontainer – Nilai-nilai konfigurasi pada instans

`get-config container`Perintah mengembalikan objek yang mencantumkan nilai konfigurasi platform dan lingkungan untuk instance lingkungan.

Contoh berikut menunjukkan output untuk perintah pada lingkungan Amazon Linux 2 Tomcat.

```
$ /opt/elasticbeanstalk/bin/get-config container
{"common_log_list":["/var/log/eb-engine.log","/var/log/eb-
hooks.log"],"default_log_list":["/var/log/nginx/access.log","/var/log/nginx/
error.log"],"environment_name":"myenv-1da84946","instance_port":"80","log_group_name_prefix":"/
aws/elasticbeanstalk","proxy_server":"nginx","static_files":
[""],"xray_enabled":"false"}
```

Untuk mengembalikan nilai kunci tertentu, gunakan pilihan `--key` (`-k`) untuk menentukan kunci.

```
$ /opt/elasticbeanstalk/bin/get-config container -k environment_name
myenv-1da84946
```

addons – Menambahkan nilai konfigurasi

`get-config addons`Perintah mengembalikan objek yang berisi informasi konfigurasi add-on lingkungan. Gunakan untuk mengambil konfigurasi database Amazon RDS yang terkait dengan lingkungan.

```
$ /opt/elasticbeanstalk/bin/get-config addons
{"rds":{"Description":"RDS Environment variables","env":
{"RDS_DB_NAME":"ebdb","RDS_HOSTNAME":"ea13k2wimu1dh8i.c18mnpu5rwvg.us-
east-2.rds.amazonaws.com","RDS_PASSWORD":"password","RDS_PORT":"3306","RDS_USERNAME":"user"}}}
```

Anda bisa membatasi hasilnya dengan dua cara. Untuk mengambil nilai untuk add-on tertentu, gunakan pilihan `--add-on` (`-a`) untuk menentukan nama add-on.

```
$ /opt/elasticbeanstalk/bin/get-config addons -a rds
{"Description":"RDS Environment variables","env":
{"RDS_DB_NAME":"ebdb","RDS_HOSTNAME":"ea13k2wimu1dh8i.c18mnpu5rwvg.us-
east-2.rds.amazonaws.com","RDS_PASSWORD":"password","RDS_PORT":"3306","RDS_USERNAME":"user"}}
```

Untuk mengembalikan nilai kunci tertentu dalam add-on, tambahkan pilihan `--key` (`-k`) untuk menentukan kunci.

```
$ /opt/elasticbeanstalk/bin/get-config addons -a rds -k RDS_DB_NAME
ebdb
```

platformconfig – nilai konfigurasi konstan

`get-config platformconfig`Perintah mengembalikan objek yang berisi informasi konfigurasi platform yang konstan ke versi platform. Outputnya sama di semua lingkungan yang menjalankan versi platform yang sama. Objek output untuk perintah memiliki dua objek tertanam:

- `GeneralConfig`— Berisi informasi yang konstan di seluruh versi terbaru dari semua cabang platform Amazon Linux 2 dan Amazon Linux 2023.
- `PlatformSpecificConfig`— Berisi informasi yang konstan untuk versi platform dan khusus untuk itu.

Contoh berikut menunjukkan output untuk perintah pada lingkungan yang menggunakan Tomcat 8.5 menjalankan cabang platform Corretto 11.

```
$ /opt/elasticbeanstalk/bin/get-config platformconfig
{"GeneralConfig":{"AppUser":"webapp","AppDeployDir":"/var/app/
current/","AppStagingDir":"/var/app/
staging/","ProxyServer":"nginx","DefaultInstancePort":"80"},"PlatformSpecificConfig":
{"ApplicationPort":"8080","JavaVersion":"11","TomcatVersion":"8.5"}}
```

Untuk mengembalikan nilai kunci tertentu, gunakan pilihan `--key` (`-k`) untuk menentukan kunci. Kunci ini unik di dua benda tertanam. Anda tidak perlu menentukan objek yang berisi kunci.

```
$ /opt/elasticbeanstalk/bin/get-config platformconfig -k AppStagingDir
/var/app/staging/
```

pilihan output get-config

Gunakan pilihan `--output` untuk menentukan format objek output. Nilai yang valid adalah `JSON` (default) dan `YAML`. Ini adalah opsi global. Anda harus menentukannya sebelum nama perintah.

Contoh berikut mengembalikan nilai opsi konfigurasi dalam format YAMM.

```
$ /opt/elasticbeanstalk/bin/get-config --output YAML optionsettings
aws:elasticbeanstalk:application:environment:
  JDBC_CONNECTION_STRING: ""
aws:elasticbeanstalk:container:tomcat:jvmoptions:
```

```
  JVM Options: ""
  Xms: 256m
  Xmx: 256m
aws:elasticbeanstalk:environment:proxy:
  ProxyServer: nginx
  StaticFiles:
        - ""
aws:elasticbeanstalk:healthreporting:system:
  SystemType: enhanced
aws:elasticbeanstalk:hostmanager:
  LogPublicationControl: "false"
```

pkg-repo

> **Note**
>
> `pkg-repo`Alat ini tidak tersedia untuk lingkungan berdasarkan platform Amazon Linux 2023. Namun, Anda dapat secara manual menerapkan paket dan pembaruan sistem operasi ke instans AL2 023. Untuk informasi selengkapnya, lihat Mengelola paket dan pembaruan sistem operasi di Panduan Pengguna Amazon Linux 2023

Dalam beberapa keadaan mendesak, Anda mungkin perlu memperbarui EC2 instans Amazon Anda dengan patch keamanan Amazon Linux 2 yang belum dirilis dengan versi platform Elastic Beanstalk yang diperlukan. Anda tidak dapat melakukan pembaruan manual pada lingkungan Elastic Beanstalk Anda secara default. Ini karena versi platform dikunci ke versi tertentu dari repositori Amazon Linux 2. Kunci ini memastikan bahwa instance menjalankan versi perangkat lunak yang didukung dan konsisten. Untuk kasus yang mendesak, `pkg-repo` alat ini memungkinkan solusi untuk memperbarui paket yum secara manual di Amazon Linux 2 jika Anda perlu menginstalnya di lingkungan sebelum dirilis dalam versi platform Elastic Beanstalk yang baru.

`pkg-repo`Alat pada platform Amazon Linux 2 menyediakan kemampuan untuk membuka kunci repositori `yum` paket. Anda kemudian dapat melakukan patch keamanan yum update secara manual. Sebaliknya, Anda dapat mengikuti pembaruan dengan menggunakan alat untuk mengunci repositori paket yum untuk mencegah pembaruan lebih lanjut. `pkg-repo`Alat ini tersedia di `/opt/elasticbeanstalk/bin/pkg-repo` direktori semua EC2 instance di lingkungan Elastic Beanstalk Anda.

Perubahan menggunakan `pkg-repo` alat dibuat hanya pada EC2 contoh di mana alat digunakan. Mereka tidak memengaruhi instance lain atau mencegah pembaruan lingkungan di masa mendatang.

Contoh yang diberikan nanti dalam topik ini menjelaskan cara menerapkan perubahan di semua instance dengan memanggil `pkg-repo` perintah dari skrip dan file konfigurasi.

> **Warning**
>
> Kami tidak merekomendasikan alat ini untuk sebagian besar pengguna. Setiap perubahan manual yang diterapkan pada versi platform yang tidak terkunci dianggap tidak sesuai. Opsi ini hanya layak bagi pengguna dalam keadaan mendesak yang dapat menerima risiko berikut:
>
> - Versi Package tidak dapat dijamin konsisten di semua instance di lingkungan Anda.
> - Lingkungan yang dimodifikasi menggunakan `pkg-repo` alat tidak dijamin berfungsi dengan baik. Mereka belum diuji dan diverifikasi pada platform yang didukung Elastic Beanstalk.
>
> Kami sangat menyarankan untuk menerapkan praktik terbaik yang mencakup rencana pengujian dan backout. Untuk membantu memfasilitasi praktik terbaik, Anda dapat menggunakan konsol Elastic Beanstalk dan EB CLI untuk mengkloning lingkungan dan menukar lingkungan. URLs Untuk informasi selengkapnya tentang penggunaan operasi ini, lihat penerapan Biru/Hijau di bagian Mengelola lingkungan dari panduan ini.

Jika Anda berencana untuk mengedit file konfigurasi repositori yum secara manual, jalankan alat terlebih dahulu. `pkg-repo` `pkg-repo`Alat ini mungkin tidak berfungsi sebagaimana dimaksud di lingkungan Amazon Linux 2 dengan file konfigurasi repositori yum yang diedit secara manual. Ini karena alat mungkin tidak mengenali perubahan konfigurasi.

Untuk informasi selengkapnya tentang repositori paket Amazon Linux, lihat topik repositori Package di Panduan Pengguna Amazon. EC2

perintah pkg-repo

Gunakan sintaks berikut untuk menjalankan perintah `pkg-repo` alat.

```
$ /opt/elasticbeanstalk/bin/pkg-repo command [options]
```

`pkg-repo`Perintahnya adalah sebagai berikut:

- lock— mengunci repositori `yum` paket ke versi tertentu

- unlock— membuka repositori `yum` paket dari versi tertentu
- status— daftar semua repositori `yum` paket dan status kunci mereka saat ini
- help— menunjukkan bantuan umum atau bantuan untuk satu perintah

Opsi berlaku untuk perintah sebagai berikut:

- `lock`, `unlock` dan `status` — opsi:`-h`,`--help`, atau tidak ada (default).
- `help`— opsi:`lock`,`unlock`,`status`, atau tidak ada (default).

Contoh berikut menjalankan perintah unlock.

```
$ sudo /opt/elasticbeanstalk/bin/pkg-repo unlock
Amazon Linux 2 core package repo successfully unlocked
Amazon Linux 2 extras package repo successfully unlocked
```

Contoh berikut menjalankan perintah lock.

```
$ sudo /opt/elasticbeanstalk/bin/pkg-repo lock
Amazon Linux 2 core package repo successfully locked
Amazon Linux 2 extras package repo successfully locked
```

Contoh berikut menjalankan perintah status.

```
$ sudo /opt/elasticbeanstalk/bin/pkg-repo status
Amazon Linux 2 core package repo is currently UNLOCKED
Amazon Linux 2 extras package repo is currently UNLOCKED
```

Contoh berikut menjalankan help perintah untuk lock perintah.

```
$ sudo /opt/elasticbeanstalk/bin/pkg-repo help lock
```

Contoh berikut menjalankan help perintah untuk `pkg-repo` alat ini.

```
$ sudo /opt/elasticbeanstalk/bin/pkg-repo help
```

Anda dapat menguji `pkg-repo` dengan menggunakan SSH untuk terhubung ke sebuah instans di lingkungan Elastic Beanstalk Anda. Salah satu opsi SSH adalah perintah EB CLI. eb ssh

> **Note**
>
> `pkg-repo`Alat ini membutuhkan hak akses pengguna root untuk dijalankan. Jika Anda mendapatkan kesalahan izin akses, jalankan perintah lagi di bawah `sudo`.
> Anda tidak perlu menambahkan `sudo` saat menggunakan alat dalam skrip atau file konfigurasi yang Anda gunakan ke lingkungan Anda. Elastic Beanstalk menjalankan semua skrip Anda sebagai pengguna root.

contoh pkg-repo

Bagian sebelumnya memberikan contoh baris perintah untuk pengujian pada EC2 instance individual dari lingkungan Elastic Beanstalk. Pendekatan ini dapat membantu untuk pengujian. Namun, ini hanya memperbarui satu instance pada satu waktu, jadi tidak praktis untuk menerapkan perubahan ke semua instance di lingkungan.

Pendekatan yang lebih pragmatis adalah dengan menggunakan skrip hook platform atau file `.ebextensions`konfigurasi untuk menerapkan perubahan di semua instance secara konsisten.

Contoh panggilan berikut `pkg-repo` dari file konfigurasi dalam `.ebextensions`folder. Elastic Beanstalk menjalankan perintah `update_package.config` dalam file saat Anda menerapkan bundel sumber aplikasi Anda.

```
.ebextensions
### update_package.config
```

Untuk menerima versi terbaru dari paket docker, konfigurasi ini menentukan paket docker dalam perintah. yum update

```
### update_package.config ###

commands:
  update_package:
    command: |
      /opt/elasticbeanstalk/bin/pkg-repo unlock
      yum update docker -y
      /opt/elasticbeanstalk/bin/pkg-repo lock
      yum clean all -y
      rm -rf /var/cache/yum
```

Konfigurasi ini tidak menentukan paket apa pun dalam yum update perintah. Semua pembaruan yang tersedia diterapkan sebagai hasilnya.

```
### update_package.config ###

commands:
  update_package:
    command: |
      /opt/elasticbeanstalk/bin/pkg-repo unlock
      yum update -y
      /opt/elasticbeanstalk/bin/pkg-repo lock
      yum clean all -y
      rm -rf /var/cache/yum
```

Contoh panggilan berikut `pkg-repo` dari skrip bash sebagai hook platform. Elastic Beanstalk `update_package.sh` menjalankan file skrip yang terletak di subdirektori. `prebuild`

```
.platform
### hooks
    ### prebuild
        ### update_package.sh
```

Untuk menerima versi terbaru dari paket docker, skrip ini menentukan paket docker dalam perintah. yum update Jika nama paket dihilangkan, semua pembaruan yang tersedia diterapkan. Contoh file konfigurasi sebelumnya menunjukkan ini.

```
### update_package.sh ###

#!/bin/bash

/opt/elasticbeanstalk/bin/pkg-repo unlock
yum update docker -y
/opt/elasticbeanstalk/bin/pkg-repo lock
yum clean all -y
rm -rf /var/cache/yum
```

download-source-bundle (Hanya AMI Amazon Linux)

Di cabang platform Amazon Linux AMI (sebelumnya Amazon Linux 2), Elastic Beanstalk menyediakan alat tambahan, yaitu. `download-source-bundle` Gunakan alat ini untuk

mengunduh kode sumber aplikasi Anda saat menerapkan platform Anda. Alat ini tersedia di `/opt/elasticbeanstalk/bin/download-source-bundle`.

Contoh skrip `00-unzip.sh` terletak di folder `appdeploy/pre` pada instans lingkungan. Ini menunjukkan bagaimana menggunakan `download-source-bundle` untuk men-download kode sumber aplikasi ke `/opt/elasticbeanstalk/deploy/appsource` folder selama penyebaran.

Memperluas platform Linux Elastic Beanstalk

Topik ini menjelaskan cara memperluas platform Linux Anda dengan perintah, skrip, perangkat lunak, dan konfigurasi Anda sendiri. Anda mungkin perlu memperluas platform Anda untuk mengubah server dan konfigurasi proxy default. Atau Anda mungkin perlu menyesuaikan cara platform membangun atau memulai aplikasi Anda.

Topik

- Buildfile dan Procfile
- Hook platform
- File konfigurasi
- Konfigurasi proksi terbalik
- Contoh aplikasi dengan ekstensi

Buildfile dan Procfile

Beberapa platform memungkinkan Anda untuk menyesuaikan bagaimana Anda membangun atau mempersiapkan aplikasi Anda, dan untuk menentukan proses yang menjalankan aplikasi Anda. Setiap topik platform individu secara khusus menyebutkan Buildfile dan/atau Procfile jika platform mendukung mereka. Cari platform spesifik Anda di bawah Platform.

Untuk semua platform pendukung, sintaks dan semantik identik, dan seperti yang dijelaskan pada halaman ini. Topik platform individu menyebutkan penggunaan spesifik file-file ini untuk membangun dan menjalankan aplikasi dalam bahasa masing-masing.

Buildfile

Untuk menentukan perintah kustom build dan konfigurasi untuk aplikasi Anda, menempatkan file yang bernama `Buildfile` di direktori root dari sumber aplikasi Anda. Nama file peka huruf besar/kecil. Gunakan sintaks berikut untuk `Buildfile` Anda.

```
<process_name>: <command>
```

Perintah di `Buildfile` Anda harus sesuai dengan ekspresi reguler berikut: `^[A-Za-z0-9_-]+:\s*[^\s].*$`

Elastic Beanstalk tidak memantau aplikasi yang dijalankan dengan `Buildfile`. Gunakan `Buildfile` untuk perintah yang berjalan dalam waktu singkat dan berakhir setelah menyelesaikan tugas mereka. Untuk proses aplikasi yang berjalan lama yang seharusnya tidak keluar, gunakan Procfile.

Semua jalur di `Buildfile` adalah relatif terhadap akar dari paket sumber. Dalam contoh berikut `Buildfile`, `build.sh` adalah script shell yang terletak di akar paket sumber.

Example Buildfile

```
make: ./build.sh
```

Jika Anda ingin memberikan langkah-langkah pembuatan kustom, kami sarankan Anda menggunakan hook platform `predeploy` untuk apa pun kecuali perintah yang paling sederhana, bukan `Buildfile`. Hook platform memungkinkan skrip yang lebih kaya dan penanganan kesalahan yang lebih baik. Hook platform dijelaskan di bagian selanjutnya.

Procfile

Untuk menentukan perintah kustom untuk memulai dan menjalankan aplikasi Anda, menempatkan file yang bernama `Procfile` di direktori root dari sumber aplikasi Anda. Nama file peka huruf besar/kecil. Gunakan sintaks berikut untuk `Procfile` Anda. Anda dapat menentukan satu atau lebih perintah.

```
<process_name1>: <command1>
<process_name2>: <command2>
...
```

Setiap baris di `Procfile` harus sesuai dengan ekspresi reguler berikut: `^[A-Za-z0-9_-]+:\s*[^\s].*$`

Menggunakan `Procfile` untuk proses aplikasi yang berjalan lama yang seharusnya tidak keluar. Elastic Beanstalk mengharapkan proses berjalan dari `Procfile` untuk berjalan terus menerus. Elastic Beanstalk memonitor proses ini dan memulai ulang setiap proses yang berakhir. Untuk proses berjalan pendek, gunakan Buildfile.

Semua jalur di `Procfile` adalah relatif terhadap akar dari paket sumber. Contoh berikut `Procfile` mendefinisikan tiga proses. Yang pertama, disebut `web` dalam contoh, adalah aplikasi web utama.

Example Procfile

```
web: bin/myserver
cache: bin/mycache
foo: bin/fooapp
```

Elastic Beanstalk mengonfigurasi server proksi untuk meneruskan permintaan ke aplikasi web utama Anda pada port 5000, dan Anda dapat mengonfigurasi nomor port ini. Sebuah penggunaan umum untuk `Procfile` adalah untuk lulus nomor port ini untuk aplikasi Anda sebagai argumen perintah. Untuk detail tentang konfigurasi proxy, lihatKonfigurasi proksi terbalik.

Elastic Beanstalk menangkap output standar dan kesalahan aliran dari proses `Procfile` dalam berkas log. Elastic Beanstalk menamakan berkas log setelah proses dan menyimpannya di `/var/log`. Misalnya, proses `web` dalam contoh sebelumnya menghasilkan log bernama `web-1.log` dan `web-1.error.log` untuk `stdout` dan `stderr`, masing-masing.

Hook platform

Hook platform dirancang khusus untuk memperluas platform lingkungan Anda. Ini adalah skrip kustom dan file eksekusi lainnya yang Anda men-deploy sebagai bagian dari kode sumber aplikasi Anda, dan Elastic Beanstalk berjalan selama berbagai tahap penyediaan instans.

> **Note**
>
> Hook platform tidak didukung pada versi platform Amazon Linux AMI (sebelumnya Amazon Linux 2).

Hook platform deployment aplikasi

Deployment aplikasi terjadi ketika Anda menyediakan paket sumber baru untuk deployment, atau ketika Anda membuat perubahan konfigurasi yang memerlukan penghentian dan rekreasi semua instans lingkungan.

Untuk menyediakan platform hook yang berjalan selama deployment aplikasi, menempatkan file di bawah direktori `.platform/hooks` di paket sumber Anda, di salah satu subdirektori berikut.

- `prebuild` – File di sini berjalan setelah platform mesin Elastic Beanstalk mengunggah dan mengekstrak paket sumber aplikasi, dan sebelum menyiapkan dan mengonfigurasi aplikasi dan server web.

  File `prebuild` berjalan setelah menjalankan perintah yang ditemukan di bagian perintah dari setiap file konfigurasi dan sebelum menjalankan perintah `Buildfile`.
- `predeploy` – File di sini berjalan setelah mesin platform Elastic Beanstalk mengatur dan mengonfigurasi aplikasi dan server web, dan sebelum men-deploy mereka ke lokasi waktu aktif akhir mereka.

  File `predeploy` berjalan setelah menjalankan perintah yang ditemukan di bagian container_commands dari setiap file konfigurasi dan sebelum menjalankan perintah `Procfile`.
- `postdeploy` – File di sini berjalan setelah mesin platform Elastic Beanstalk men-deploy aplikasi dan server proksi.

  Ini adalah langkah alur kerja deployment terakhir.

Hook platform deployment konfigurasi

Deployment konfigurasi terjadi ketika Anda membuat perubahan konfigurasi yang hanya memperbarui instans lingkungan tanpa menciptakan mereka. Pembaharuan pilihan berikut ini menyebabkan pembaruan konfigurasi.

- Properti lingkungan dan pengaturan khusus platform
- File statis
- AWS X-Ray daemon
- Penyimpanan log dan streaming
- Port aplikasi (untuk detailnya lihatKonfigurasi proksi terbalik)

Untuk menyediakan hook yang berjalan selama deployment konfigurasi, menempatkan mereka di bawah direktori `.platform/confighooks` dalam paket sumber Anda. Tiga subdirektori yang sama seperti untuk aplikasi deployment hook berlaku.

Selengkapnya tentang hook platform

File hook dapat berupa file biner, atau berkas skrip yang dimulai dengan baris `#!` yang berisi jalur penerjemah mereka, seperti `#!/bin/bash`. Semua file harus memiliki izin eksekusi. Gunakan

`chmod +x` untuk mengatur izin eksekusi pada file hook Anda. Untuk semua versi platform berbasis Amazon Linux 2023 dan Amazon Linux 2 yang dirilis pada atau setelah 29 April 2022, Elastic Beanstalk secara otomatis memberikan izin eksekusi ke semua skrip kait platform. Dalam hal ini Anda tidak perlu memberikan izin eksekusi secara manual. Untuk daftar versi platform ini, lihat catatan rilis Linux 29 April 2022 di Panduan Catatan AWS Elastic Beanstalk Rilis.

Elastic Beanstalk menjalankan file di masing-masing direktori ini dalam urutan leksikografis nama file. Semua file berjalan sebagai pengguna `root`. Direktori kerja saat ini (cwd) untuk hook platform adalah direktori root aplikasi. Untuk file `prebuild` dan `predeploy` itu adalah direktori pementasan aplikasi, dan untuk file `postdeploy` itu adalah direktori aplikasi saat ini. Jika salah satu file gagal (keluar dengan kode keluar non-nol), deployment dibatalkan dan gagal.

Skrip teks kait platform mungkin gagal jika berisi karakter pemisah baris Windows Carriage Return / Line Feed (CRLF). Jika file disimpan di host Windows, kemudian ditransfer ke server Linux, itu mungkin berisi jeda baris Windows CRLF. Untuk platform yang dirilis pada atau setelah 29 Desember 2022, Elastic Beanstalk secara otomatis mengonversi karakter Windows CRLF menjadi karakter pemutus baris Linux Line Feed (LF) di file teks kait platform. Jika aplikasi Anda berjalan pada platform Amazon Linux 2 yang dirilis sebelum tanggal ini, Anda harus mengonversi karakter Windows CRLF ke karakter LF Linux. Salah satu cara untuk mencapai ini adalah dengan membuat dan menyimpan file skrip pada host Linux. Alat yang mengonversi karakter ini juga tersedia di internet.

File hook memiliki akses ke semua properti lingkungan yang telah Anda tetapkan dalam opsi aplikasi, dan variabel lingkungan sistem `HOME`, `PATH`, dan `PORT`.

Untuk mendapatkan nilai variabel lingkungan dan opsi konfigurasi lainnya ke skrip kait platform Anda, Anda dapat menggunakan utilitas `get-config` yang Elastic Beanstalk sediakan dalam instans lingkungan. Untuk rincian selengkapnya, lihat the section called “Alat skrip platform”.

File konfigurasi

Anda dapat menambahkan direktori file konfigurasi ke direktori `.ebextensions` dari kode sumber aplikasi Anda untuk mengonfigurasi berbagai aspek lingkungan Elastic Beanstalk Anda. Antara lain, file konfigurasi memungkinkan Anda menyesuaikan perangkat lunak dan file lain pada instans lingkungan Anda dan menjalankan perintah inisialisasi pada instans. Untuk informasi selengkapnya, lihat the section called “Server Linux”.

Anda juga dapat mengatur opsi konfigurasi menggunakan file konfigurasi. Banyak opsi mengontrol perilaku platform, dan beberapa opsi ini adalah spesifik platform.

Untuk platform berbasis Amazon Linux 2 dan Amazon Linux 2023, sebaiknya gunakan Buildfile, Procfile, dan hook platform untuk mengonfigurasi dan menjalankan kode khusus pada instans lingkungan Anda selama penyediaan instans. Mekanisme ini dijelaskan di bagian sebelumnya di halaman ini. Anda masih dapat menggunakan perintah dan perintah kontainer di file konfigurasi `.ebextensions`, tetapi tidak mudah untuk digunakan. Misalnya, menulis skrip perintah di dalam file YAML dapat menantang dari sudut pandang sintaks. Anda masih perlu menggunakan file `.ebextensions` konfigurasi untuk skrip apa pun yang membutuhkan referensi ke AWS CloudFormation sumber daya.

Konfigurasi proksi terbalik

Semua versi platform Amazon Linux 2 dan Amazon Linux 2023 menggunakan nginx sebagai server proxy terbalik default mereka. Platform Tomcat, Node.js, PHP, dan Python juga mendukung Apache HTTPD sebagai alternatif. Untuk memilih Apache pada platform ini, mengatur pilihan `ProxyServer` di namespace `aws:elasticbeanstalk:environment:proxy` ke `apache`. Semua platform mengaktifkan konfigurasi server proksi dengan cara yang seragam, seperti yang dijelaskan di bagian ini.

> **Note**
>
> Pada Amazon Linux AMI versi platform (sebelumnya Amazon Linux 2) Anda mungkin harus mengonfigurasi server proksi secara berbeda. Anda dapat menemukan rincian warisan ini di bawah topik platform masing-masing dalam panduan ini.

Elastic Beanstalk mengonfigurasi server proksi pada instans lingkungan Anda untuk meneruskan lalu lintas web ke aplikasi web utama pada URL akar lingkungan; misalnya, `http://my-env.elasticbeanstalk.com`.

Secara default, Elastic Beanstalk mengonfigurasi proksi untuk meneruskan permintaan masuk pada port 80 untuk aplikasi web utama Anda pada port 5000. Anda dapat mengonfigurasi nomor port ini dengan menetapkan properti lingkungan `PORT` menggunakan namespace aws:elasticbeanstalk:application:environment dalam file konfigurasi, seperti yang ditunjukkan dalam contoh berikut.

```
option_settings:
  - namespace:  aws:elasticbeanstalk:application:environment
    option_name:  PORT
```

```
    value:  <main_port_number>
```

Untuk informasi selengkapnya tentang pengaturan variabel lingkungan untuk aplikasi Anda, lihat the section called “Pengaturan opsi”.

Aplikasi Anda harus mendengarkan pada port yang dikonfigurasi untuk itu di proksi. Jika Anda mengubah port default menggunakan properti lingkungan `PORT`, kode Anda dapat mengaksesnya dengan membaca nilai variabel lingkungan `PORT`. Sebagai contoh, panggilan `os.Getenv("PORT")` di Go, atau `System.getenv("PORT")` di Java. Jika Anda mengonfigurasi proksi Anda untuk mengirim lalu lintas ke beberapa proses aplikasi, Anda dapat mengonfigurasi beberapa properti lingkungan, dan menggunakan nilai-nilai mereka di kedua konfigurasi proksi dan kode aplikasi Anda. Pilihan lain adalah untuk lulus nilai port untuk proses sebagai argumen perintah dalam `Procfile`. Untuk informasi selengkapnya, lihat Buildfile dan Procfile.

Mengonfigurasi nginx

Elastic Beanstalk menggunakan nginx sebagai proksi terbalik default untuk memetakan aplikasi Anda ke penyeimbangan beban Elastic Load Balancing Anda. Elastic Beanstalk menyediakan konfigurasi nginx default yang dapat Anda perpanjang atau timpa sepenuhnya dengan konfigurasi Anda sendiri.

> **Note**
>
> Saat Anda menambahkan atau mengedit file konfigurasi `.conf` nginx, pastikan untuk menyandikannya sebagai UTF-8.

Untuk memperpanjang konfigurasi nginx default Elastic Beanstalk, tambahkan file konfigurasi `.conf` ke folder yang bernama `.platform/nginx/conf.d/` dalam paket sumber aplikasi Anda. Konfigurasi nginx Elastic Beanstalk mencakup file `.conf` dalam folder ini secara automatik.

```
~/workspace/my-app/
|-- .platform
|   `-- nginx
|       `-- conf.d
|           `-- myconf.conf
`-- other source files
```

Untuk mengesampingkan konfigurasi nginx default Elastic Beanstalk sepenuhnya, sertakan konfigurasi di paket sumber Anda di `.platform/nginx/nginx.conf`:

```
~/workspace/my-app/
|-- .platform
|   `-- nginx
|       `-- nginx.conf
`-- other source files
```

Jika Anda menimpa konfigurasi nginx Elastic Beanstalk, tambahkan baris berikut ke `nginx.conf` untuk menarik dalam konfigurasi Elastic Beanstalk untuk Peningkatan pelaporan dan pemantauan kesehatan di Elastic Beanstalk, pemetaan aplikasi otomatis, dan file statis.

```
include conf.d/elasticbeanstalk/*.conf;
```

Mengonfigurasi Apache HTTPD

Platform Tomcat, Node.js, PHP, dan Python memungkinkan Anda memilih server proksi Apache HTTPD sebagai alternatif untuk nginx. Ini bukan default. Contoh berikut mengonfigurasi Elastic Beanstalk untuk menggunakan Apache HTTPD.

Example .ebextensions/httpd-proxy.config

```
option_settings:
  aws:elasticbeanstalk:environment:proxy:
    ProxyServer: apache
```

Anda dapat memperpanjang Elastic Beanstalk konfigurasi Apache default dengan file konfigurasi tambahan Anda. Atau, Anda dapat sepenuhnya mengganti konfigurasi Apache default Elastic Beanstalk.

Untuk memperpanjang konfigurasi Apache default Elastic Beanstalk, tambahkan file konfigurasi `.conf` ke folder bernama `.platform/httpd/conf.d` di paket sumber aplikasi Anda. Konfigurasi Apache Elastic Beanstalk mencakup file `.conf` di folder ini secara otomatis.

```
~/workspace/my-app/
|-- .ebextensions
|   -- httpd-proxy.config
|-- .platform
|   -- httpd
|      -- conf.d
|         -- port5000.conf
```

```
|         -- ssl.conf
-- index.jsp
```

Sebagai contoh, konfigurasi Apache 2.4 berikut menambahkan listener di port 5000.

Example . platform/httpd/conf.d/port5000.conf

```
listen 5000
<VirtualHost *:5000>
  <Proxy *>
    Require all granted
  </Proxy>
  ProxyPass / http://localhost:8080/ retry=0
  ProxyPassReverse / http://localhost:8080/
  ProxyPreserveHost on

  ErrorLog /var/log/httpd/elasticbeanstalk-error_log
</VirtualHost>
```

Untuk mengesampingkan konfigurasi Apache default Elastic Beanstalk sepenuhnya, sertakan konfigurasi dalam paket sumber Anda di `.platform/httpd/conf/httpd.conf`.

```
~/workspace/my-app/
|-- .ebextensions
|   -- httpd-proxy.config
|-- .platform
|   `-- httpd
|       `-- conf
|           `-- httpd.conf
`-- index.jsp
```

> **Note**
>
> Jika Anda mengganti konfigurasi Apache Elastic Beanstalk, tambahkan baris berikut ke `httpd.conf` Anda untuk menarik dalam konfigurasi Elastic Beanstalk untuk Peningkatan pelaporan dan pemantauan kesehatan di Elastic Beanstalk, pemetaan aplikasi otomatis, dan file statis.
>
> ```
> IncludeOptional conf.d/elasticbeanstalk/*.conf
> ```

Contoh aplikasi dengan ekstensi

Contoh berikut menunjukkan bundel sumber aplikasi dengan beberapa fitur ekstensibilitas yang `Procfile` didukung platform Elastic Beanstalk Amazon Linux 2 dan Amazon Linux 2023.`.ebextensions`: a, file konfigurasi, kait khusus, dan file konfigurasi proxy.

```
~/my-app/
|-- web.jar
|-- Procfile
|-- readme.md
|-- .ebextensions/
|   |-- options.config        # Option settings
|   `-- cloudwatch.config     # Other .ebextensions sections, for example files and
 container commands
`-- .platform/
    |-- nginx/                # Proxy configuration
    |   |-- nginx.conf
    |   `-- conf.d/
    |       `-- custom.conf
    |-- hooks/                # Application deployment hooks
    |   |-- prebuild/
    |   |   |-- 01_set_secrets.sh
    |   |   `-- 12_update_permissions.sh
    |   |-- predeploy/
    |   |   `-- 01_some_service_stop.sh
    |   `-- postdeploy/
    |       |-- 01_set_tmp_file_permissions.sh
    |       |-- 50_run_something_after_app_deployment.sh
    |       `-- 99_some_service_start.sh
    `-- confighooks/          # Configuration deployment hooks
        |-- prebuild/
        |   `-- 01_set_secrets.sh
        |-- predeploy/
        |   `-- 01_some_service_stop.sh
        `-- postdeploy/
            |-- 01_run_something_after_config_deployment.sh
            `-- 99_some_service_start.sh
```

> **Note**
>
> Beberapa ekstensi ini tidak didukung di Amazon Linux AMI platform versi (sebelumnya Amazon Linux 2).

Menyebarkan aplikasi.NET Windows dengan Elastic Beanstalk

> **Lihat .NET di Pusat AWS Pengembang**
>
> Sudahkah Anda mampir ke Pusat Pengembang.Net kami? Ini adalah one stop shop kami untuk semua hal. NET di AWS.
> Untuk informasi lebih lanjut, lihat .NET di Pusat AWS Pengembang.

Bab ini memberikan instruksi untuk mengkonfigurasi dan menyebarkan aplikasi web ASP.NET dan .NET Core Windows Anda ke. AWS Elastic Beanstalk Elastic Beanstalk memudahkan penerapan, pengelolaan, dan skala aplikasi web.NET (Windows) Anda menggunakan Amazon Web Services.

Anda dapat menerapkan aplikasi Anda hanya dalam beberapa menit menggunakan Elastic Beanstalk Command Line Interface (EB CLI) atau dengan menggunakan konsol Elastic Beanstalk. Setelah Anda menerapkan aplikasi Elastic Beanstalk Anda, Anda dapat terus menggunakan EB CLI untuk mengelola aplikasi dan lingkungan Anda, atau Anda dapat menggunakan konsol Elastic Beanstalk,, atau. AWS CLI APIs

Bab ini menyediakan tutorial berikut:

- QuickStart untuk .NET Core di Windows— Step-by-step instruksi untuk membuat dan menyebarkan aplikasi Hello World .NET Core Windows menggunakan EB CLI.
- QuickStart untuk ASP.NET— Step-by-step instruksi untuk membuat dan menyebarkan aplikasi Hello World ASP.NET menggunakan aplikasi. AWS Toolkit for Visual Studio

Jika Anda memerlukan bantuan dengan pengembangan aplikasi Windows .NET Core, ada beberapa tempat yang dapat Anda kunjungi:

- .NET Development Forum - Posting pertanyaan Anda dan dapatkan umpan balik.
- .NET Developer Center — Toko serba ada untuk kode sampel, dokumentasi, alat, dan sumber daya tambahan.
- AWS SDK for .NET Documentation — Baca tentang menyiapkan SDK dan menjalankan contoh kode, fitur SDK, dan informasi terperinci tentang operasi API untuk SDK.

> **Note**
>
> Platform ini tidak mendukung lingkungan pekerja. Lihat perinciannya di Lingkungan pekerja Elastic Beanstalk.

Topik

- QuickStart: Menyebarkan .NET Core pada aplikasi Windows ke Elastic Beanstalk
- QuickStart: Menyebarkan aplikasi ASP.NET ke Elastic Beanstalk
- Menyiapkan lingkungan pengembangan .NET Anda
- Menggunakan platform Elastic Beanstalk .NET Windows
- Menambahkan instans DB Amazon RDS ke lingkungan aplikasi .NET Anda
- The AWS Toolkit for Visual Studio
- Memigrasi aplikasi .NET On-Premise Anda ke Elastic Beanstalk
- Rekomendasi untuk komponen pensiun Windows Server pada Elastic Beanstalk

QuickStart: Menyebarkan .NET Core pada aplikasi Windows ke Elastic Beanstalk

QuickStart Tutorial ini memandu Anda melalui proses pembuatan .NET Core pada aplikasi Windows dan menyebarkannya ke AWS Elastic Beanstalk lingkungan.

> **Note**
>
> Contoh tutorial dimaksudkan untuk demonstrasi. Jangan gunakan aplikasi untuk lalu lintas produksi.

Bagian-bagian

- AWS Akun Anda
- Prasyarat
- Langkah 1: Buat .NET Core pada aplikasi Windows
- Langkah 2: Jalankan aplikasi Anda secara lokal

- Langkah 3: Menyebarkan .NET Core Anda pada aplikasi Windows dengan EB CLI
- Langkah 4: Jalankan aplikasi Anda di Elastic Beanstalk
- Langkah 5: Bersihkan
- AWS sumber daya untuk aplikasi Anda
- Langkah selanjutnya
- Terapkan dengan konsol Elastic Beanstalk

AWS Akun Anda

Jika Anda belum menjadi AWS pelanggan, Anda perlu membuat AWS akun. Mendaftar memungkinkan Anda mengakses Elastic Beanstalk AWS dan layanan lain yang Anda butuhkan.

Jika Anda sudah memiliki AWS akun, Anda dapat melanjutkan kePrasyarat.

Buat AWS akun

Mendaftar untuk Akun AWS

Jika Anda tidak memiliki Akun AWS, selesaikan langkah-langkah berikut untuk membuatnya.

Untuk mendaftar untuk Akun AWS

1. Buka https://portal.aws.amazon.com/billing/pendaftaran.
2. Ikuti petunjuk online.

   Bagian dari prosedur pendaftaran melibatkan menerima panggilan telepon atau pesan teks dan memasukkan kode verifikasi pada keypad telepon.

   Saat Anda mendaftar untuk sebuah Akun AWS, sebuah Pengguna root akun AWSdibuat. Pengguna root memiliki akses ke semua Layanan AWS dan sumber daya di akun. Sebagai praktik keamanan terbaik, tetapkan akses administratif ke pengguna, dan gunakan hanya pengguna root untuk melakukan tugas yang memerlukan akses pengguna root.

AWS mengirimkan email konfirmasi setelah proses pendaftaran selesai. Kapan saja, Anda dapat melihat aktivitas akun Anda saat ini dan mengelola akun Anda dengan masuk https://aws.amazon.comke/ dan memilih Akun Saya.

Buat pengguna dengan akses administratif

Setelah Anda mendaftar Akun AWS, amankan Pengguna root akun AWS, aktifkan AWS IAM Identity Center, dan buat pengguna administratif sehingga Anda tidak menggunakan pengguna root untuk tugas sehari-hari.

Amankan Anda Pengguna root akun AWS

1. Masuk ke AWS Management Consolesebagai pemilik akun dengan memilih pengguna Root dan memasukkan alamat Akun AWS email Anda. Di laman berikutnya, masukkan kata sandi.

   Untuk bantuan masuk dengan menggunakan pengguna root, lihat Masuk sebagai pengguna root di AWS Sign-In Panduan Pengguna.
2. Mengaktifkan autentikasi multi-faktor (MFA) untuk pengguna root Anda.

   Untuk petunjuk, lihat Mengaktifkan perangkat MFA virtual untuk pengguna Akun AWS root (konsol) Anda di Panduan Pengguna IAM.

Buat pengguna dengan akses administratif

1. Aktifkan Pusat Identitas IAM.

   Untuk mendapatkan petunjuk, silakan lihat Mengaktifkan AWS IAM Identity Center di Panduan Pengguna AWS IAM Identity Center .
2. Di Pusat Identitas IAM, berikan akses administratif ke pengguna.

   Untuk tutorial tentang menggunakan Direktori Pusat Identitas IAM sebagai sumber identitas Anda, lihat Mengkonfigurasi akses pengguna dengan default Direktori Pusat Identitas IAM di Panduan AWS IAM Identity Center Pengguna.

Masuk sebagai pengguna dengan akses administratif

- Untuk masuk dengan pengguna Pusat Identitas IAM, gunakan URL masuk yang dikirim ke alamat email saat Anda membuat pengguna Pusat Identitas IAM.

  Untuk bantuan masuk menggunakan pengguna Pusat Identitas IAM, lihat Masuk ke portal AWS akses di Panduan AWS Sign-In Pengguna.

Tetapkan akses ke pengguna tambahan

1. Di Pusat Identitas IAM, buat set izin yang mengikuti praktik terbaik menerapkan izin hak istimewa paling sedikit.

   Untuk petunjuk, lihat Membuat set izin di Panduan AWS IAM Identity Center Pengguna.
2. Tetapkan pengguna ke grup, lalu tetapkan akses masuk tunggal ke grup.

   Untuk petunjuk, lihat Menambahkan grup di Panduan AWS IAM Identity Center Pengguna.

Prasyarat

Untuk mengikuti prosedur di panduan ini, Anda memerlukan terminal atau shell baris perintah untuk menjalankan perintah. Perintah ditampilkan dalam daftar yang didahului oleh simbol prompt (**>**) dan nama direktori saat ini, bila sesuai.

```
C:\eb-project> this is a command
this is output
```

EB CLI

Tutorial ini menggunakan Elastic Beanstalk Command Line Interface (EB CLI). Untuk detail tentang pemasangan dan konfigurasi EB CLI, lihat Instal EB CLI dengan skrip pengaturan (disarankan) dan Mengonfigurasi EB CLI.

.NET Core pada Windows

Jika Anda tidak memiliki .NET SDK diinstal pada mesin lokal Anda, Anda dapat menginstalnya dengan mengikuti tautan Download.NET di situs dokumentasi .NET.

Verifikasi instalasi .NET SDK Anda dengan menjalankan perintah berikut.

```
C:\> dotnet --info
```

Langkah 1: Buat .NET Core pada aplikasi Windows

Buat direktori proyek.

```
C:\> mkdir eb-dotnetcore
```

```
C:\> cd eb-dotnetcore
```

Selanjutnya, buat contoh aplikasi layanan RESTful web Hello World dengan menjalankan perintah berikut.

```
C:\eb-dotnetcore> dotnet new web --name HelloElasticBeanstalk
C:\eb-dotnetcore> cd HelloElasticBeanstalk
```

Langkah 2: Jalankan aplikasi Anda secara lokal

Jalankan perintah berikut untuk menjalankan aplikasi Anda secara lokal.

```
C:\eb-dotnetcore\HelloElasticBeasntalk> dotnet run
```

Outputnya akan terlihat seperti teks berikut.

```
info: Microsoft.Hosting.Lifetime[14]
      Now listening on: https://localhost:7222
info: Microsoft.Hosting.Lifetime[14]
      Now listening on: http://localhost:5228
info: Microsoft.Hosting.Lifetime[0]
      Application started. Press Ctrl+C to shut down.
info: Microsoft.Hosting.Lifetime[0]
      Hosting environment: Development
info: Microsoft.Hosting.Lifetime[0]
      Content root path: C:\Users\Administrator\eb-dotnetcore\HelloElasticBeanstalk
```

> **Note**
>
> `dotnet`Perintah memilih port secara acak saat menjalankan aplikasi secara lokal. Dalam contoh ini port adalah 5228. Saat Anda menerapkan aplikasi ke lingkungan Elastic Beanstalk Anda, aplikasi akan berjalan pada port 5000.

Masukkan alamat URL `http://localhost:`*`port`* di browser web Anda. Untuk contoh spesifik ini, perintahnya adalah`http://localhost:5228`. Browser web harus menampilkan “Hello World!”.

Langkah 3: Menyebarkan .NET Core Anda pada aplikasi Windows dengan EB CLI

Jalankan perintah berikut untuk membuat lingkungan Elastic Beanstalk untuk aplikasi ini.

Untuk membuat lingkungan dan menyebarkan .NET Core Anda pada aplikasi Windows

1. Jalankan perintah berikut di `HelloElasticBeanstalk` direktori untuk mempublikasikan dan zip aplikasi Anda.

```
C:\eb-dotnetcore\HelloElasticBeasntalk> dotnet publish -o site
C:\eb-dotnetcore\HelloElasticBeasntalk> cd site
C:\eb-dotnetcore\HelloElasticBeasntalk\site> Compress-Archive -Path * -
DestinationPath ../site.zip
C:\eb-dotnetcore\HelloElasticBeasntalk\site> cd ..
```

2. Buat file baru dalam `HelloElasticBeanstalk` panggilan `aws-windows-deployment-manifest.json` dengan konten berikut:

```
{
    "manifestVersion": 1,
    "deployments": {
        "aspNetCoreWeb": [
        {
            "name": "test-dotnet-core",
            "parameters": {
                "appBundle": "site.zip",
                "iisPath": "/",
                "iisWebSite": "Default Web Site"
            }
        }
        ]
    }
}
```

3. Inisialisasi repositori EB CLI Anda dengan perintah eb init.

```
C:\eb-dotnetcore\HelloElasticBeasntalk> eb init -p iis dotnet-windows-server-
tutorial --region us-east-2
```

Perintah ini membuat aplikasi bernama `dotnet-windows-server-tutorial` dan mengkonfigurasi repositori lokal Anda untuk membuat lingkungan dengan versi platform server Windows terbaru.

4. Buat lingkungan baru dan deploy aplikasi Anda dengan eb create. Elastic Beanstalk secara otomatis membuat file zip untuk aplikasi Anda dan memulainya pada port 5000.

```
C:\eb-dotnetcore\HelloElasticBeasntalk> eb create dotnet-windows-server-env
```

Dibutuhkan sekitar lima menit untuk Elastic Beanstalk untuk menciptakan lingkungan Anda.

Langkah 4: Jalankan aplikasi Anda di Elastic Beanstalk

Ketika proses untuk membuat lingkungan Anda selesai, buka situs web Anda denganeb open.

```
C:\eb-dotnetcore\HelloElasticBeasntalk> eb open
```

Selamat! Anda telah menerapkan .NET Core pada aplikasi Windows dengan Elastic Beanstalk! Ini akan membuka jendela peramban menggunakan nama domain yang dibuat untuk aplikasi Anda.

Langkah 5: Bersihkan

Anda dapat menghentikan lingkungan Anda ketika Anda selesai bekerja dengan aplikasi Anda. Elastic Beanstalk AWS mengakhiri semua sumber daya yang terkait dengan lingkungan Anda.

Untuk mengakhiri lingkungan Elastic Beanstalk Anda dengan EB CLI jalankan perintah berikut.

```
C:\eb-dotnetcore\HelloElasticBeasntalk> eb terminate
```

AWS sumber daya untuk aplikasi Anda

Anda baru saja membuat aplikasi instance tunggal. Ini berfungsi sebagai aplikasi sampel langsung dengan satu EC2 instance, sehingga tidak memerlukan penyeimbangan beban atau penskalaan otomatis. Untuk aplikasi contoh tunggal Elastic Beanstalk menciptakan sumber daya berikut: AWS

- EC2 instance — Mesin EC2 virtual Amazon yang dikonfigurasi untuk menjalankan aplikasi web pada platform yang Anda pilih.

Setiap platform menjalankan satu set perangkat lunak, file konfigurasi, dan penulisan yang berbeda untuk mendukung versi bahasa, kerangka kerja, kontainer web tertentu, atau kombinasi dari semua ini. Sebagian besar platform menggunakan Apache atau nginx sebagai proksi terbalik yang memproses lalu lintas web di depan aplikasi web Anda, meneruskan permintaan ke aplikasi web, menyajikan aset statis, dan menghasilkan log akses dan kesalahan.

- Grup keamanan instans — Grup EC2 keamanan Amazon yang dikonfigurasi untuk memungkinkan lalu lintas masuk di port 80. Sumber daya ini memungkinkan lalu lintas HTTP dari penyeimbang beban mencapai EC2 instance yang menjalankan aplikasi web Anda. Secara default, lalu lintas tidak diizinkan di port lain.
- Bucket Amazon S3 – Lokasi penyimpanan untuk kode sumber, log, dan artifact lainnya yang dibuat saat Anda menggunakan Elastic Beanstalk.
- CloudWatch Alarm Amazon — Dua CloudWatch alarm yang memantau beban pada instans di lingkungan Anda dan dipicu jika beban terlalu tinggi atau terlalu rendah. Saat alarm terpicu, grup Auto Scaling Anda akan menaikkan atau menurunkan skala sebagai respons.
- AWS CloudFormation stack - Elastic AWS CloudFormation Beanstalk digunakan untuk meluncurkan sumber daya di lingkungan Anda dan menyebarkan perubahan konfigurasi. Sumber daya ditentukan di sebuah templat yang dapat Anda lihat di Konsol AWS CloudFormation.
- Nama domain — Nama domain yang merutekan ke aplikasi web Anda dalam formulir *`subdomain`*.*`region`*.elasticbeanstalk.com.

Elastic Beanstalk mengelola semua sumber daya tersebut. Ketika Anda mengakhiri lingkungan, Elastic Beanstalk mengakhiri semua sumber daya yang dimuatnya.

Langkah selanjutnya

Setelah lingkungan Anda menjalankan aplikasi, Anda dapat men-deploy versi baru aplikasi atau aplikasi yang berbeda kapan saja. Menyebarkan versi aplikasi baru sangat cepat karena tidak memerlukan penyediaan atau memulai ulang instance. EC2 Anda juga dapat menjelajahi lingkungan baru Anda menggunakan konsol Elastic Beanstalk. Untuk langkah-langkah mendetail, lihat Menjelajahi lingkungan Anda di bagian Memulai panduan ini.

> **Coba lebih banyak tutorial**
>
> Jika Anda ingin mencoba tutorial lain dengan aplikasi contoh yang berbeda, lihatQuickStart untuk ASP.NET.

Setelah Anda menerapkan satu atau dua contoh aplikasi dan siap untuk mulai mengembangkan dan menjalankan .NET Core pada aplikasi Windows secara lokal, lihat Menyiapkan lingkungan pengembangan .NET Anda

Terapkan dengan konsol Elastic Beanstalk

Anda juga dapat menggunakan konsol Elastic Beanstalk untuk meluncurkan aplikasi sampel. Untuk langkah-langkah rinci, lihat Membuat aplikasi contoh di Bab Memulai panduan ini.

QuickStart: Menyebarkan aplikasi ASP.NET ke Elastic Beanstalk

QuickStart Tutorial ini memandu Anda melalui proses membuat aplikasi ASP.NET dan menyebarkannya ke lingkungan AWS Elastic Beanstalk .

> **Note**
>
> Contoh tutorial dimaksudkan untuk demonstrasi. Jangan gunakan aplikasi untuk lalu lintas produksi.

Bagian-bagian

- AWS Akun Anda
- Prasyarat
- Langkah 1: Buat aplikasi ASP.NET
- Langkah 2: Jalankan aplikasi Anda secara lokal
- Langkah 3: Menyebarkan aplikasi ASP.NET Anda dengan AWS Toolkit for Visual Studio
- Langkah 4: Jalankan aplikasi Anda di Elastic Beanstalk
- Langkah 5: Bersihkan
- AWS sumber daya untuk aplikasi Anda
- Langkah selanjutnya
- Terapkan dengan konsol Elastic Beanstalk

AWS Akun Anda

Jika Anda belum menjadi AWS pelanggan, Anda perlu membuat AWS akun. Mendaftar memungkinkan Anda mengakses Elastic Beanstalk AWS dan layanan lain yang Anda butuhkan.

Jika Anda sudah memiliki AWS akun, Anda dapat melanjutkan kePrasyarat.

Buat AWS akun

Mendaftar untuk Akun AWS

Jika Anda tidak memiliki Akun AWS, selesaikan langkah-langkah berikut untuk membuatnya.

Untuk mendaftar untuk Akun AWS

1. Buka https://portal.aws.amazon.com/billing/pendaftaran.
2. Ikuti petunjuk online.

   Bagian dari prosedur pendaftaran melibatkan menerima panggilan telepon atau pesan teks dan memasukkan kode verifikasi pada keypad telepon.

   Saat Anda mendaftar untuk sebuah Akun AWS, sebuah Pengguna root akun AWSdibuat. Pengguna root memiliki akses ke semua Layanan AWS dan sumber daya di akun. Sebagai praktik keamanan terbaik, tetapkan akses administratif ke pengguna, dan gunakan hanya pengguna root untuk melakukan tugas yang memerlukan akses pengguna root.

AWS mengirimi Anda email konfirmasi setelah proses pendaftaran selesai. Kapan saja, Anda dapat melihat aktivitas akun Anda saat ini dan mengelola akun Anda dengan masuk https://aws.amazon.comke/ dan memilih Akun Saya.

Buat pengguna dengan akses administratif

Setelah Anda mendaftar Akun AWS, amankan Pengguna root akun AWS, aktifkan AWS IAM Identity Center, dan buat pengguna administratif sehingga Anda tidak menggunakan pengguna root untuk tugas sehari-hari.

Amankan Anda Pengguna root akun AWS

1. Masuk ke AWS Management Consolesebagai pemilik akun dengan memilih pengguna Root dan memasukkan alamat Akun AWS email Anda. Di laman berikutnya, masukkan kata sandi.

   Untuk bantuan masuk dengan menggunakan pengguna root, lihat Masuk sebagai pengguna root di AWS Sign-In Panduan Pengguna.
2. Mengaktifkan autentikasi multi-faktor (MFA) untuk pengguna root Anda.

Untuk petunjuk, lihat Mengaktifkan perangkat MFA virtual untuk pengguna Akun AWS root (konsol) Anda di Panduan Pengguna IAM.

Buat pengguna dengan akses administratif

1. Aktifkan Pusat Identitas IAM.

   Untuk mendapatkan petunjuk, silakan lihat Mengaktifkan AWS IAM Identity Center di Panduan Pengguna AWS IAM Identity Center .

2. Di Pusat Identitas IAM, berikan akses administratif ke pengguna.

   Untuk tutorial tentang menggunakan Direktori Pusat Identitas IAM sebagai sumber identitas Anda, lihat Mengkonfigurasi akses pengguna dengan default Direktori Pusat Identitas IAM di Panduan AWS IAM Identity Center Pengguna.

Masuk sebagai pengguna dengan akses administratif

- Untuk masuk dengan pengguna Pusat Identitas IAM, gunakan URL masuk yang dikirim ke alamat email saat Anda membuat pengguna Pusat Identitas IAM.

  Untuk bantuan masuk menggunakan pengguna Pusat Identitas IAM, lihat Masuk ke portal AWS akses di Panduan AWS Sign-In Pengguna.

Tetapkan akses ke pengguna tambahan

1. Di Pusat Identitas IAM, buat set izin yang mengikuti praktik terbaik menerapkan izin hak istimewa paling sedikit.

   Untuk petunjuknya, lihat Membuat set izin di Panduan AWS IAM Identity Center Pengguna.

2. Tetapkan pengguna ke grup, lalu tetapkan akses masuk tunggal ke grup.

   Untuk petunjuk, lihat Menambahkan grup di Panduan AWS IAM Identity Center Pengguna.

Prasyarat

QuickStart Tutorial ini memandu Anda melalui pembuatan aplikasi “Hello World” dan menyebarkannya ke lingkungan Elastic Beanstalk dengan Visual Studio dan. AWS Toolkit for Visual Studio

Visual Studio

Untuk mengunduh dan menginstal Visual Studio ikuti petunjuk di halaman unduhan Visual Studio. Contoh ini menggunakan Visual Studio 2022. Selama instalasi Visual Studio pilih item spesifik ini:

- Pada tab Workloads — pilih ASP.NET dan pengembangan web.
- Pada tab Komponen individu - pilih alat pengembangan.NET Framework 4.8 dan templat proyek dan item .NET Framework.

AWS Toolkit for Visual Studio

Untuk mengunduh dan mengatur AWS Toolkit for Visual Studio ikuti petunjuk dalam topik Memulai Panduan AWS Toolkit for Visual Studio Pengguna.

Langkah 1: Buat aplikasi ASP.NET

Selanjutnya, buat aplikasi yang akan Anda terapkan ke lingkungan Elastic Beanstalk. Kami akan membuat aplikasi web ASP.NET “Hello World”.

Untuk membuat aplikasi ASP.NET

1. Luncurkan Visual Studio. Di menu File, pilih Baru, lalu Project.
2. Kotak dialog Create a new project akan ditampilkan. Pilih aplikasi web ASP.NET (.NET Framework), lalu pilih Berikutnya.
3. Pada dialog Configure your new project, masukkan `eb-aspnet` nama Project Anda. Dari menu dropdown Framework pilih .NET Framework 4.8, lalu pilih Create.

   Perhatikan direktori proyek. Dalam contoh ini, direktori proyek adalah`C:\Users\Administrator\source\repos\eb-aspnet\eb-aspnet`.
4. Dialog Create a new ASP.NET Web Application ditampilkan. Pilih template Kosong. Selanjutnya pilih Buat.

Pada titik ini, Anda telah membuat proyek aplikasi web ASP.NET kosong menggunakan Visual Studio. Selanjutnya, kita akan membuat formulir web yang akan berfungsi sebagai titik masuk untuk aplikasi web ASP.NET.

5. Dari menu Project, pilih Add New Item. Pada halaman Tambahkan Item Baru, pilih Formulir Web dan beri nama`Default.aspx`. Selanjutnya pilih Tambah.
6. Tambahkan yang berikut ini ke `Default.aspx`:

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Default.aspx.cs"
 Inherits="eb_aspnet.Default" %>

<!DOCTYPE html>

<html xmlns="https://www.w3.org/1999/xhtml">
<head runat="server">
    <title>Hello Elastic Beanstalk!</title>
</head>
<body>
    <form id="body" runat="server">
        <div>
            Hello Elastic Beanstalk! This is an ASP.NET on Windows Server
 application.
        </div>
    </form>
</body>
</html>
```

Langkah 2: Jalankan aplikasi Anda secara lokal

Di Visual Studio, dari menu Debug pilih Start Debugging untuk menjalankan aplikasi Anda secara lokal. Halaman harus menampilkan “Hello Elastic Beanstalk! Ini adalah ASP.NET pada aplikasi Windows Server.”

Langkah 3: Menyebarkan aplikasi ASP.NET Anda dengan AWS Toolkit for Visual Studio

Ikuti langkah-langkah ini untuk membuat lingkungan Elastic Beanstalk dan menyebarkan aplikasi baru Anda ke sana.

Untuk membuat lingkungan dan menyebarkan aplikasi ASP.NET Anda

1. Di Solution Explorer, klik kanan aplikasi Anda, lalu pilih Publish to AWS Elastic Beanstalk.
2. Pilih nama untuk aplikasi dan lingkungan Elastic Beanstalk baru Anda.
3. Di luar titik ini, Anda dapat melanjutkan dengan default yang disediakan oleh Elastic Beanstalk atau memodifikasi salah satu opsi dan pengaturan sesuai keinginan Anda.
4. Pada halaman Review, pilih Deploy. Ini akan mengemas aplikasi web ASP.NET Anda dan menyebarkannya ke Elastic Beanstalk.

   Dibutuhkan sekitar lima menit untuk Elastic Beanstalk untuk menciptakan lingkungan Anda. Fitur deployment Elastic Beanstalk akan memantau lingkungan Anda sampai tersedia dengan kode yang baru di-deploy. Pada tab Env: **< `environment name` >**, Anda akan melihat status untuk lingkungan Anda.

Langkah 4: Jalankan aplikasi Anda di Elastic Beanstalk

Ketika proses untuk membuat lingkungan Anda selesai, tab Env: **< `environment name` >**, menampilkan informasi tentang lingkungan dan aplikasi Anda, termasuk URL domain untuk meluncurkan aplikasi Anda. Pilih URL ini di tab ini atau salin dan tempel ke browser web Anda.

Selamat! Anda telah menerapkan aplikasi ASP.NET dengan Elastic Beanstalk!

Langkah 5: Bersihkan

Ketika Anda selesai bekerja dengan aplikasi Anda, Anda dapat mengakhiri lingkungan Anda di. AWS Toolkit for Visual Studio

Untuk mengakhiri lingkungan Anda

1. Perluas simpul Elastic Beanstalk dan simpul aplikasi di AWS Explorer. Klik kanan lingkungan aplikasi Anda dan pilih Mengakhiri Lingkungan.
2. Saat diminta, pilih Ya untuk mengonfirmasi bahwa Anda ingin mengakhiri lingkungan. Diperlukan beberapa menit bagi Elastic Beanstalk untuk AWS menghentikan sumber daya yang berjalan di lingkungan.

AWS sumber daya untuk aplikasi Anda

Anda baru saja membuat aplikasi instance tunggal. Ini berfungsi sebagai aplikasi sampel langsung dengan satu EC2 instance, sehingga tidak memerlukan penyeimbangan beban atau penskalaan otomatis. Untuk aplikasi contoh tunggal Elastic Beanstalk menciptakan sumber daya berikut: AWS

- EC2 instance — Mesin EC2 virtual Amazon yang dikonfigurasi untuk menjalankan aplikasi web pada platform yang Anda pilih.

  Setiap platform menjalankan satu set perangkat lunak, file konfigurasi, dan penulisan yang berbeda untuk mendukung versi bahasa, kerangka kerja, kontainer web tertentu, atau kombinasi dari semua ini. Sebagian besar platform menggunakan Apache atau nginx sebagai proksi terbalik yang memproses lalu lintas web di depan aplikasi web Anda, meneruskan permintaan ke aplikasi web, menyajikan aset statis, dan menghasilkan log akses dan kesalahan.
- Grup keamanan instans — Grup EC2 keamanan Amazon yang dikonfigurasi untuk memungkinkan lalu lintas masuk di port 80. Sumber daya ini memungkinkan lalu lintas HTTP dari penyeimbang beban mencapai EC2 instance yang menjalankan aplikasi web Anda. Secara default, lalu lintas tidak diizinkan di port lain.
- Bucket Amazon S3 – Lokasi penyimpanan untuk kode sumber, log, dan artifact lainnya yang dibuat saat Anda menggunakan Elastic Beanstalk.
- CloudWatch Alarm Amazon — Dua CloudWatch alarm yang memantau beban pada instans di lingkungan Anda dan dipicu jika beban terlalu tinggi atau terlalu rendah. Saat alarm terpicu, grup Auto Scaling Anda akan menaikkan atau menurunkan skala sebagai respons.
- AWS CloudFormation stack - Elastic AWS CloudFormation Beanstalk digunakan untuk meluncurkan sumber daya di lingkungan Anda dan menyebarkan perubahan konfigurasi. Sumber daya ditentukan di sebuah templat yang dapat Anda lihat di Konsol AWS CloudFormation.
- Nama domain — Nama domain yang merutekan ke aplikasi web Anda dalam formulir *`subdomain`*.*`region`*.elasticbeanstalk.com.

Elastic Beanstalk mengelola semua sumber daya tersebut. Ketika Anda mengakhiri lingkungan, Elastic Beanstalk mengakhiri semua sumber daya yang dimuatnya.

Langkah selanjutnya

Setelah lingkungan Anda menjalankan aplikasi, Anda dapat men-deploy versi baru aplikasi atau aplikasi yang berbeda kapan saja. Menyebarkan versi aplikasi baru sangat cepat karena tidak

memerlukan penyediaan atau memulai ulang instance. EC2 Anda juga dapat menjelajahi lingkungan baru Anda menggunakan konsol Elastic Beanstalk. Untuk langkah-langkah mendetail, lihat Menjelajahi lingkungan Anda di bagian Memulai panduan ini.

> **Coba lebih banyak tutorial**
>
> Jika Anda ingin mencoba tutorial lain dengan aplikasi contoh yang berbeda, lihatQuickStart untuk .NET Core di Windows.

Setelah Anda menerapkan satu atau dua contoh aplikasi dan siap untuk mulai mengembangkan dan menjalankan aplikasi ASP.NET secara lokal, lihat Menyiapkan lingkungan pengembangan .NET Anda

Terapkan dengan konsol Elastic Beanstalk

Anda juga dapat menggunakan konsol Elastic Beanstalk untuk meluncurkan aplikasi sampel. Untuk langkah-langkah rinci, lihat Membuat aplikasi contoh di Bab Memulai panduan ini.

Menyiapkan lingkungan pengembangan .NET Anda

Topik ini memberikan petunjuk untuk menyiapkan lingkungan pengembangan.NET Windows untuk menguji aplikasi Anda secara lokal sebelum menerapkannya. AWS Elastic Beanstalk Ini juga merujuk situs web yang menyediakan instruksi instalasi untuk alat yang berguna.

Bagian-bagian

- Menginstal IDE
- Memasang AWS Toolkit for Visual Studio

Jika Anda perlu mengelola AWS sumber daya dari dalam aplikasi Anda, instal AWS SDK for .NET. Sebagai contoh, Anda dapat menggunakan Amazon S3 untuk menyimpan dan mengambil data.

Dengan AWS SDK for .NET, Anda dapat memulai dalam hitungan menit dengan satu paket yang dapat diunduh lengkap dengan template proyek Visual Studio, pustaka .NET, AWS sampel kode C #, dan dokumentasi. Contoh praktis disediakan di C# tentang bagaimana menggunakan pustaka untuk membangun aplikasi. Tutorial video online dan dokumentasi referensi disediakan untuk membantu Anda mempelajari cara menggunakan pustaka dan sampel kode.

Kunjungi beranda AWS SDK for .NET untuk informasi selengkapnya dan petunjuk instalasi.

Menginstal IDE

Lingkungan pengembangan terintegrasi (IDEs) menyediakan berbagai fitur yang memfasilitasi pengembangan aplikasi. Jika Anda belum menggunakan IDE untuk pengembangan .NET, cobalah Visual Studio Community untuk memulai.

Kunjungi halaman Studio Visual Community untuk mengunduh dan menginstal Visual Studio Community.

Memasang AWS Toolkit for Visual Studio

AWS Toolkit for Visual StudioIni adalah plug-in open source untuk Visual Studio IDE yang memudahkan pengembang untuk mengembangkan, men-debug, dan menyebarkan aplikasi.NET menggunakan. AWS Kunjungi beranda Toolkit for Visual Studio untuk petunjuk instalasi.

Menggunakan platform Elastic Beanstalk .NET Windows

Topik ini menjelaskan cara mengkonfigurasi, membangun, dan menjalankan aplikasi web ASP.NET dan .NET Core Windows Anda di Elastic Beanstalk.

AWS Elastic Beanstalk mendukung sejumlah platform untuk berbagai versi kerangka pemrograman .NET dan Windows Server. Lihat .NET di Windows Server dengan IIS di dokumen Platform AWS Elastic Beanstalk untuk daftar yang lengkap.

Elastic Beanstalk menyediakan opsi konfigurasi yang dapat Anda gunakan untuk menyesuaikan perangkat lunak yang berjalan EC2 pada instance di lingkungan Elastic Beanstalk Anda. Anda dapat mengonfigurasi variabel lingkungan yang dibutuhkan oleh aplikasi Anda, mengaktifkan rotasi log ke Amazon S3, dan mengatur pengaturan kerangka kerja .NET.

Opsi konfigurasi tersedia di konsol Elastic Beanstalk untuk memodifikasi konfigurasi dari lingkungan yang sedang berjalan. Agar Anda tidak kehilangan konfigurasi lingkungan ketika mengakhirinya, Anda dapat menggunakan konfigurasi tersimpan untuk menyimpan pengaturan dan kemudian menerapkannya ke lingkungan lain.

Untuk menyimpan pengaturan di kode sumber, Anda dapat menyertakan file konfigurasi. Pengaturan di file konfigurasi diterapkan setiap kali Anda membuat lingkungan atau men-deploy aplikasi Anda. Anda juga dapat menggunakan file konfigurasi untuk menginstal paket, menjalankan penulisan, dan melakukan operasi penyesuaian instans lainnya selama deployment.

Pengaturan yang diterapkan di konsol Elastic Beanstalk mengganti pengaturan yang sama di file konfigurasi, jika ada. Hal tersebut memungkinkan Anda memiliki pengaturan default di file konfigurasi, dan mengganti keduanya dengan pengaturan khusus lingkungan di konsol tersebut. Untuk informasi selengkapnya tentang yang diutamakan, dan metode lain untuk mengubah pengaturan, lihat Opsi konfigurasi.

Mengonfigurasi lingkungan .NET Anda di konsol Elastic Beanstalk

Anda dapat menggunakan konsol Elastic Beanstalk untuk mengaktifkan rotasi log ke Amazon S3, mengonfigurasi variabel yang dapat dibaca aplikasi dari lingkungan, dan mengubah pengaturan kerangka kerja .NET.

Untuk mengonfigurasi lingkungan .NET Anda di konsol Elastic Beanstalk

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Dalam kategori konfigurasi Pembaruan, pemantauan, dan pencatatan, pilih Edit.

Opsi kontainer

- Target waktu aktif .NET – Atur ke `2.0` untuk menjalankan CLR v2.
- Aktifkan aplikasi 32-bit – Atur ke `True` untuk menjalankan aplikasi 32-bit.

Opsi log

Bagian Opsi Log memiliki dua pengaturan:

- Profil instans – Menentukan profil instans yang memiliki izin untuk mengakses bucket Amazon S3 yang terkait dengan aplikasi Anda.
- Aktifkan rotasi file log ke Amazon S3 — Menentukan apakah file log untuk EC2 instans Amazon aplikasi Anda disalin ke bucket Amazon S3 yang terkait dengan aplikasi Anda.

Properti lingkungan

Bagian Properti Lingkungan memungkinkan Anda menentukan pengaturan konfigurasi lingkungan pada EC2 instance Amazon yang menjalankan aplikasi Anda. Pengaturan ini diteruskan sebagai pasangan nilai kunci ke aplikasi. Gunakan `System.GetEnvironmentVariable` untuk membacanya. Kunci identik dapat ada di kedua `web.config` dan sebagai properti lingkungan. Gunakan namespace `System.Configuration` untuk membaca nilai-nilai dari `web.config`.

```
NameValueCollection appConfig = ConfigurationManager.AppSettings;
string endpoint = appConfig["API_ENDPOINT"];
```

Lihat Variabel lingkungan dan pengaturan perangkat lunak lainnya untuk informasi selengkapnya.

Namespace aws:elasticbeanstalk:container:dotnet:apppool

Anda dapat menggunakan file konfigurasi untuk mengatur opsi konfigurasi dan melakukan tugas-tugas konfigurasi instans lain selama deployment. Opsi konfigurasi dapat berupa platform khusus atau berlaku untuk semua platform di layanan Elastic Beanstalk secara keseluruhan. Opsi konfigurasi diatur ke dalam ruang nama.

Platform .NET menentukan opsi di namespace `aws:elasticbeanstalk:container:dotnet:apppool` yang dapat Anda gunakan untuk mengonfigurasi waktu aktif .NET.

File konfigurasi contoh berikut menunjukkan pengaturan untuk masing-masing opsi yang tersedia di namespace ini:

Example .ebextensions/go-settings.config

```
option_settings:
  aws:elasticbeanstalk:container:dotnet:apppool:
    Target Runtime: 2.0
    Enable 32-bit Applications: True
```

Elastic Beanstalk memberikan banyak opsi konfigurasi untuk menyesuaikan lingkungan Anda. Selain file konfigurasi, Anda juga dapat mengatur opsi konfigurasi menggunakan konsol tersebut, konfigurasi tersimpan, EB CLI, atau AWS CLI. Lihat Opsi konfigurasi untuk informasi selengkapnya.

Migrasi di versi utama dari platform server Elastic Beanstalk Windows

AWS Elastic Beanstalk telah memiliki beberapa versi utama dari platform Windows Server-nya. Halaman ini mencakup perbaikan utama untuk setiap versi utama, dan hal yang harus dipertimbangkan sebelum Anda memigrasi ke versi yang lebih baru.

Platform Windows Server saat ini di versi 2 (v2). Jika aplikasi Anda menggunakan versi platform Windows Server sebelum versi v2, kami sarankan Anda bermigrasi ke v2.

Apa yang baru di versi utama platform server Windows

Platform server Windows V2

Versi 2 (v2) platform Elastic Beanstalk Windows Server dirilis pada bulan Februari 2019. V2 membawa perilaku platform Windows Server lebih dekat dengan platform Elastic Beanstalk berbasis Linux dalam beberapa cara penting. V2 sepenuhnya kompatibel dengan v1, membuat migrasi dari v1 mudah.

Platform Windows Server sekarang mendukung hal berikut:

- Versioning – Setiap rilis mendapat nomor versi baru, dan Anda dapat mengacu ke versi sebelumnya (yang masih tersedia untuk Anda) saat membuat dan mengelola lingkungan.
- Peningkatan kondisi – Untuk detailnya, lihat Peningkatan pelaporan dan pemantauan kesehatan di Elastic Beanstalk.
- Deployment Tetap dan Bergulir dengan Batch Tambahan – Untuk detail tentang kebijakan deployment, lihat Men-deploy aplikasi ke lingkungan Elastic Beanstalk.
- Pembaruan tetap – Untuk detail tentang tipe pembaruan, lihat Perubahan konfigurasi.
- Pembaruan platform terkelola – Untuk detailnya, lihat Pembaruan platform yang dikelola.

> **Note**
>
> Fitur deployment dan pembaruan baru tergantung pada peningkatan kondisi. Aktifkan peningkatan kondisi untuk menggunakannya. Untuk detailnya, lihat Mengaktifkan pelaporan kondisi yang ditingkatkan Elastic Beanstalk.

Platform server Windows V1

Versi 1.0.0 (v1) platform Elastic Beanstalk Windows Server dirilis pada bulan Oktober 2015. Versi ini mengubah urutan tempat Elastic Beanstalk memproses perintah di file konfigurasi selama pembuatan dan pembaruan lingkungan.

Versi platform sebelumnya tidak memiliki nomor versi di nama tumpukan solusi:

- 64bit Windows Server 2012 R2 menjalankan IIS 8.5
- 64bit Windows Server Core 2012 R2 menjalankan IIS 8.5
- 64bit Windows Server 2012 menjalankan IIS 8
- 64bit Windows Server 2008 R2 menjalankan IIS 7.5

Di versi sebelumnya, urutan pemrosesan file konfigurasi tidak konsisten. Selama pembuatan lingkungan, `Container Commands` yang menjalankan sumber aplikasi di-deploy ke IIS. Selama deployment ke lingkungan berjalan, perintah kontainer dijalankan sebelum versi baru di-deploy. Selama menaikkan skala, file konfigurasi tidak diproses sama sekali.

Selain itu, IIS dimulai sebelum perintah kontainer berjalan. Perilaku ini menyebabkan beberapa pelanggan untuk menerapkan solusi di kontainer perintah, menghentikan server IIS sebelum menjalankan perintah, dan memulai lagi setelah mereka selesai.

Versi 1 memperbaiki ketidakkonsistenan dan membawa perilaku platform Windows Server lebih dekat dengan platform Elastic Beanstalk berbasis Linux. Di platform v1, Elastic Beanstalk selalu menjalankan kontainer perintah sebelum memulai server IIS.

Tumpukan solusi platform v1 memiliki v1 setelah versi Windows Server:

- 64bit Windows Server 2012 R2 v1.1.0 menjalankan IIS 8.5
- 64bit Windows Server Core 2012 R2 v1.1.0 menjalankan IIS 8.5
- 64bit Windows Server 2012 v1.1.0 menjalankan IIS 8
- 64bit Windows Server 2008 R2 v1.1.0 menjalankan IIS 7.5

Selain itu, platform v1 mengekstrak isi paket sumber aplikasi Anda ke `C:\staging\` sebelum menjalankan kontainer perintah. Setelah kontainer perintah selesai, isi folder ini dikompresi ke dalam file .zip dan di-deploy ke IIS. Alur kerja ini mengizinkan Anda untuk memodifikasi isi paket sumber aplikasi Anda dengan perintah atau tulisan sebelum deployment.

Migrasi dari versi utama sebelumnya dari platform server Windows

Baca bagian ini untuk pertimbangan migrasi sebelum memperbarui lingkungan. Untuk memperbarui platform lingkungan Anda ke versi yang lebih baru, lihat Memperbarui versi platform lingkungan Elastic Beanstalk Anda.

Dari V1 ke V2

Platform Windows Server v2 tidak mendukung .NET Core 1.x dan 2.0. Jika Anda memigrasi aplikasi Anda dari Windows Server v1 ke v2, dan aplikasi Anda menggunakan salah satu versi .NET Core ini, perbarui aplikasi Anda ke versi .NET Core yang didukung v2. Untuk daftar versi yang didukung, lihat .NET di Windows Server dengan IIS di Platform AWS Elastic Beanstalk .

Jika aplikasi Anda menggunakan Amazon Machine Image (AMI) khusus, buat AMI khusus baru berdasarkan platform Windows Server v2 AMI. Untuk pelajari selengkapnya, lihat Menggunakan image mesin Amazon kustom (AMI) di lingkungan Elastic Beanstalk Anda.

> **Note**
>
> Fitur deployment dan pembaruan yang baru bagi Windows Server v2 bergantung pada peningkatan kondisi. Ketika Anda memigrasi lingkungan ke v2, peningkatan kondisi dinonaktifkan. Aktifkan untuk menggunakan fitur ini. Untuk detailnya, lihat Mengaktifkan pelaporan kondisi yang ditingkatkan Elastic Beanstalk.

Dari pre-V1

Selain pertimbangan untuk migrasi dari v1, jika Anda memigrasi aplikasi Anda dari tumpukan solusi Windows Server sebelum versi v1, dan Anda saat ini menggunakan kontainer perintah, hapus perintah yang ditambahkan untuk mengatasi masalah sekitar ketidakkonsistenan pengolahan ketika Anda bermigrasi ke versi yang lebih baru. Dimulai dengan v1, kontainer perintah dijamin untuk berjalan sepenuhnya sebelum sumber aplikasi yang di-deploy dan sebelum IIS dimulai. Hal ini memungkinkan Anda untuk membuat perubahan ke sumber di `C:\staging` dan memodifikasi file konfigurasi IIS selama langkah ini tanpa masalah.

Misalnya, Anda dapat menggunakan file AWS CLI untuk mengunduh file DLL ke sumber aplikasi Anda dari Amazon S3:

```
.ebextensions\copy-dll.config
```

```
container_commands:
  copy-dll:
    command: aws s3 cp s3://amzn-s3-demo-bucket/dlls/large-dll.dll .\lib\
```

Untuk informasi selengkapnya tentang menggunakan file konfigurasi, lihat Penyesuaian lingkungan lanjutan dengan file konfigurasi (`.ebextensions`).

Menjalankan beberapa aplikasi dan aplikasi ASP.NET Core dengan manifes deployment

Anda dapat menggunakan manifes deployment untuk memberitahu Elastic Beanstalk cara men-deploy aplikasi Anda. Dengan menggunakan metode ini, Anda tidak perlu menggunakan `MSDeploy` untuk menghasilkan paket sumber untuk satu aplikasi ASP.NET yang berjalan di jalur akar situs web Anda. Sebaliknya, Anda dapat menggunakan file manifes untuk menjalankan beberapa aplikasi di jalur yang berbeda. Atau, Anda dapat memberitahu Elastic Beanstalk untuk men-deploy dan menjalankan aplikasi dengan ASP.NET Core. Anda juga dapat menggunakan manifes deployment untuk mengonfigurasi kolam aplikasi tempat aplikasi Anda dijalankan.

Manifes deployment menambahkan dukungan untuk Aplikasi .NET Core ke Elastic Beanstalk. Anda dapat men-deploy aplikasi .NET Framework tanpa manifes deployment. Namun, aplikasi .NET Core memerlukan manifes deployment untuk berjalan di Elastic Beanstalk. Ketika Anda menggunakan manifes deployment, Anda membuat arsip situs untuk setiap aplikasi, dan kemudian memaketkan arsip situs di arsip ZIP kedua yang berisi manifes deployment.

Manifes deployment juga menambahkan kemampuan untuk menjalankan beberapa aplikasi di jalur yang berbeda. Manifes deployment mendefinisikan array target deployment, masing-masing dengan arsip situs dan jalur tempat IIS harus menjalankannya. Sebagai contoh, Anda dapat menjalankan API web di jalur `/api` untuk melayani permintaan asinkron, dan aplikasi web di jalur akar yang mengonsumsi API.

Anda juga dapat menggunakan manifes deployment untuk menjalankan beberapa aplikasi menggunakan kolam aplikasi di IIS atau Kestrel. Anda dapat mengonfigurasi kolam aplikasi untuk memulai ulang aplikasi Anda secara berkala, menjalankan aplikasi 32-bit, atau menggunakan versi tertentu dari waktu aktif .NET Framework.

Untuk kustomisasi penuh, Anda dapat menulis skrip penyebaran Anda sendiri di Windows PowerShell dan memberi tahu Elastic Beanstalk skrip mana yang harus dijalankan untuk menginstal, menghapus, dan memulai ulang aplikasi Anda.

Manifes deployment dan fitur terkait memerlukan platform Windows Server versi 1.2.0 atau lebih baru.

Untuk informasi mendetail tentang semua opsi konfigurasi, properti, dan fitur lanjutan yang tersedia seperti melewatkan penyetelan ulang IIS, lihat referensi skema manifes penerapan.

Bagian-bagian

- Aplikasi .NET core
- Jalankan beberapa aplikasi
- Mengonfigurasi kolam aplikasi
- Menentukan deployment khusus
- Referensi skema manifes penerapan

Aplikasi .NET core

Anda dapat menggunakan manifes deployment untuk menjalankan aplikasi .NET Core di Elastic Beanstalk. .NET Core adalah versi lintas platform dari .NET yang datang dengan alat baris perintah (`dotnet`). Anda dapat menggunakannya untuk menghasilkan aplikasi, menjalankannya secara lokal, dan menyiapkannya untuk penerbitan.

Untuk menjalankan aplikasi .NET Core di Elastic Beanstalk, Anda dapat menjalankan `dotnet publish` dan paket output di arsip ZIP, tidak termasuk direktori yang berisi. Menempatkan arsip situs di paket sumber dengan manifes deployment dengan target deployment tipe `aspNetCoreWeb`.

Manifes deployment berikut menjalankan aplikasi .NET Core dari arsip situs bernama `dotnet-core-app.zip` di jalur akar.

Example aws-windows-deployment-manifest.json - .NET inti

```
{
  "manifestVersion": 1,
  "deployments": {
    "aspNetCoreWeb": [
      {
        "name": "my-dotnet-core-app",
        "parameters": {
          "archive": "dotnet-core-app.zip",
          "iisPath": "/"
        }
```

```
      }
    ]
  }
}
```

Paketkan manifes dan arsip situs di arsip ZIP untuk membuat paket sumber.

Example dotnet-core-bundle.zip

```
.
|-- aws-windows-deployment-manifest.json
`-- dotnet-core-app.zip
```

Arsip situs berisi kode aplikasi yang dikompilasi, dependensi, dan file `web.config`.

Example dotnet-core-app.zip

```
.
|-- Microsoft.AspNetCore.Hosting.Abstractions.dll
|-- Microsoft.AspNetCore.Hosting.Server.Abstractions.dll
|-- Microsoft.AspNetCore.Hosting.dll
|-- Microsoft.AspNetCore.Http.Abstractions.dll
|-- Microsoft.AspNetCore.Http.Extensions.dll
|-- Microsoft.AspNetCore.Http.Features.dll
|-- Microsoft.AspNetCore.Http.dll
|-- Microsoft.AspNetCore.HttpOverrides.dll
|-- Microsoft.AspNetCore.Server.IISIntegration.dll
|-- Microsoft.AspNetCore.Server.Kestrel.dll
|-- Microsoft.AspNetCore.WebUtilities.dll
|-- Microsoft.Extensions.Configuration.Abstractions.dll
|-- Microsoft.Extensions.Configuration.EnvironmentVariables.dll
|-- Microsoft.Extensions.Configuration.dll
|-- Microsoft.Extensions.DependencyInjection.Abstractions.dll
|-- Microsoft.Extensions.DependencyInjection.dll
|-- Microsoft.Extensions.FileProviders.Abstractions.dll
|-- Microsoft.Extensions.FileProviders.Physical.dll
|-- Microsoft.Extensions.FileSystemGlobbing.dll
|-- Microsoft.Extensions.Logging.Abstractions.dll
|-- Microsoft.Extensions.Logging.dll
|-- Microsoft.Extensions.ObjectPool.dll
|-- Microsoft.Extensions.Options.dll
|-- Microsoft.Extensions.PlatformAbstractions.dll
|-- Microsoft.Extensions.Primitives.dll
```

```
|-- Microsoft.Net.Http.Headers.dll
|-- System.Diagnostics.Contracts.dll
|-- System.Net.WebSockets.dll
|-- System.Text.Encodings.Web.dll
|-- dotnet-core-app.deps.json
|-- dotnet-core-app.dll
|-- dotnet-core-app.pdb
|-- dotnet-core-app.runtimeconfig.json
`-- web.config
```

Jalankan beberapa aplikasi

Anda dapat menjalankan beberapa aplikasi menggunakan manifes deployment dengan mendefinisikan beberapa target deployment.

Manifes deployment berikut mengonfigurasi dua aplikasi .NET Core. `WebApiSampleApp`Aplikasi ini mengimplementasikan API web sederhana dan menyajikan permintaan asinkron di jalur. `/api` Aplikasi `DotNetSampleApp` adalah aplikasi web yang melayani permintaan di jalur akar.

Example aws-windows-deployment-manifest.json - beberapa aplikasi

```
{
  "manifestVersion": 1,
  "deployments": {
    "aspNetCoreWeb": [
      {
        "name": "WebAPISample",
        "parameters": {
          "appBundle": "WebApiSampleApp.zip",
          "iisPath": "/api"
        }
      },
      {
        "name": "DotNetSample",
        "parameters": {
          "appBundle": "DotNetSampleApp.zip",
          "iisPath": "/"
        }
      }
    ]
  }
}
```

Aplikasi sampel dengan beberapa aplikasi tersedia di sini:

- Bundel sumber yang dapat digunakan - -v2.zip dotnet-multiapp-sample-bundle
- Kode sumber - dotnet-multiapp-sample-source-v2.zip

Mengonfigurasi kolam aplikasi

Anda dapat mendukung beberapa aplikasi di lingkungan Windows Anda. Dua pendekatan yang tersedia:

- Anda dapat menggunakan model out-of-process hosting dengan server web Kestrel. Dengan model ini, Anda mengonfigurasi beberapa aplikasi untuk berjalan dalam satu kolam aplikasi.
- Anda dapat menggunakan model hosting dalam proses. Dengan model ini, Anda menggunakan beberapa kolam aplikasi untuk menjalankan beberapa aplikasi hanya dengan satu aplikasi di setiap kolam. Jika Anda menggunakan server IIS dan perlu menjalankan beberapa aplikasi, Anda harus menggunakan pendekatan ini.

Untuk mengonfigurasi Kestrel agar menjalankan beberapa aplikasi di satu kolam aplikasi, tambahkan `hostingModel="OutofProcess"` di file `web.config`. Pertimbangkan contoh berikut.

Example web.config - untuk model hosting Kestrel out-of-process

```
<configuration>
<location path="." inheritInChildApplications="false">
<system.webServer>
<handlers>
<add
    name="aspNetCore"
    path="*" verb="*"
    modules="AspNetCoreModuleV2"
    resourceType="Unspecified" />
</handlers>
<aspNetCore
    processPath="dotnet"
    arguments=".\CoreWebApp-5-0.dll"
    stdoutLogEnabled="false"
    stdoutLogFile=".\logs\stdout"
    hostingModel="OutofProcess" />
</system.webServer>
</location>
```

```
</configuration>
```

Example aws-windows-deployment-manifest.json - beberapa aplikasi

```
{
"manifestVersion": 1,
  "deployments": {"msDeploy": [
      {"name": "Web-app1",
        "parameters": {"archive": "site1.zip",
          "iisPath": "/"
        }
      },
      {"name": "Web-app2",
        "parameters": {"archive": "site2.zip",
          "iisPath": "/app2"
        }
      }
    ]
  }
}
```

IIS tidak mendukung beberapa aplikasi dalam di satu kolam aplikasi karena menggunakan model hosting dalam proses. Oleh karena itu, Anda perlu mengonfigurasi beberapa aplikasi dengan menetapkan setiap aplikasi ke satu kolam aplikasi. Dengan kata lain, tetapkan hanya satu aplikasi ke satu kolam aplikasi.

Anda dapat mengonfigurasi IIS untuk menggunakan kolam aplikasi yang berbeda di file `aws-windows-deployment-manifest.json`. Lakukan pembaruan berikut saat Anda merujuk ke file contoh berikutnya:

- Tambahkan bagian `iisConfig` yang mencakup subbagian yang disebut `appPools`.
- Di blok `appPools`, buat daftar kolam aplikasi.
- Di bagian `deployments`, definisikan sebuah bagian `parameters` untuk setiap aplikasi.
- Untuk setiap aplikasi bagian `parameters` menentukan sebuah arsip, sebuah jalur untuk menjalankannya, dan sebuah `appPool` tempat untuk menjalankannya.

Manifes deployment berikut mengonfigurasi dua kolam aplikasi yang memulai ulang aplikasi mereka setiap 10 menit. Mereka juga melampirkan aplikasi mereka ke aplikasi web .NET Framework yang berjalan di jalur yang ditentukan.

Example aws-windows-deployment-manifest.json - satu aplikasi per kumpulan aplikasi

```
{
"manifestVersion": 1,
  "iisConfig": {"appPools": [
      {"name": "MyFirstPool",
       "recycling": {"regularTimeInterval": 10}
      },
      {"name": "MySecondPool",
       "recycling": {"regularTimeInterval": 10}
      }
     ]
    },
  "deployments": {"msDeploy": [
      {"name": "Web-app1",
        "parameters": {
           "archive": "site1.zip",
           "iisPath": "/",
           "appPool": "MyFirstPool"
           }
      },
      {"name": "Web-app2",
        "parameters": {
           "archive": "site2.zip",
           "iisPath": "/app2",
           "appPool": "MySecondPool"
         }
      }
     ]
    }
}
```

Menentukan deployment khusus

Untuk kontrol lebih, Anda dapat sepenuhnya menyesuaikan deployment aplikasi dengan mendefinisikan deployment khusus.

Manifes penerapan ini menginstruksikan Elastic Beanstalk untuk mengeksekusi skrip dalam mode 32-bit. PowerShell Ini menentukan tiga skrip: `install` script (`siteInstall.ps1`) yang berjalan selama peluncuran dan penerapan instance, `uninstall` script (`siteUninstall.ps1`) yang dijalankan sebelum menginstal versi baru selama penerapan, dan `restart` script

(`siteRestart.ps1`) yang berjalan saat Anda memilih Restart App Server di konsol manajemen. AWS

Example aws-windows-deployment-manifest.json - penyebaran kustom

```
{
  "manifestVersion": 1,
  "deployments": {
    "custom": [
      {
        "name": "Custom site",
        "architecture" : 32,
        "scripts": {
          "install": {
            "file": "siteInstall.ps1"
          },
          "restart": {
            "file": "siteRestart.ps1"
          },
          "uninstall": {
            "file": "siteUninstall.ps1"
          }
        }
      }
    ]
  }
}
```

Sertakan Artifact yang diperlukan untuk menjalankan aplikasi dalam paket sumber Anda dengan manifes dan penulisan.

Example C. ustom-site-bundle Zip

```
.
|-- aws-windows-deployment-manifest.json
|-- siteInstall.ps1
|-- siteRestart.ps1
|-- siteUninstall.ps1
`-- site-contents.zip
```

Referensi skema manifes penerapan

Manifes penyebaran adalah file JSON yang mendefinisikan bagaimana Elastic Beanstalk harus menyebarkan dan mengkonfigurasi aplikasi Windows Anda. Bagian ini menyediakan referensi komprehensif untuk semua properti yang didukung dan opsi konfigurasi dalam skema manifes.

Struktur manifes

Manifes penerapan mengikuti skema JSON tertentu dengan struktur tingkat atas berikut:

Example Struktur manifes dasar

```
{
  "manifestVersion": 1,
  "skipIISReset": false,
  "iisConfig": {
    "appPools": [...]
  },
  "deployments": {
    "msDeploy": [...],
    "aspNetCoreWeb": [...],
    "custom": [...]
  }
}
```

Properti tingkat atas

`manifestVersion`(diperlukan)

Jenis: Angka

Default: 1

Nilai yang valid: 1

Menentukan versi skema manifes. Saat ini, hanya versi 1 yang didukung.

`skipIISReset` (opsional)

Jenis: Boolean

Default: false

Mengontrol apakah IIS disetel ulang selama penerapan aplikasi. Bendera ini memengaruhi keduanya `msDeploy` dan jenis `aspNetCoreWeb` penerapan.

Perilaku:

- Tidak ditentukan atau *`false`* (default): Reset IIS dilakukan selama operasi penginstalan, uninstall, dan pembaruan. Ini adalah perilaku tradisional.
- *`true`*: Reset IIS dilewati selama operasi penerapan.

Manfaat:

- Mengurangi waktu henti - Aplikasi mengalami interupsi layanan yang lebih pendek selama penerapan.
- Penerapan yang lebih cepat — Menghilangkan waktu yang diperlukan IIS untuk memulai ulang dan menginisialisasi ulang sepenuhnya.

> **Note**
>
> Saat menggunakan`skipIISReset`, RestartAppServeroperasi melakukan reset IIS terlepas dari pengaturan bendera ini.

Contoh:

```
{
  "manifestVersion": 1,
  "skipIISReset": true,
  "deployments": {
    "aspNetCoreWeb": [
      {
        "name": "my-dotnet-core-app",
        "parameters": {
          "archive": "dotnet-core-app.zip",
          "iisPath": "/"
        }
      }
    ]
  }
}
```

`deployments`(diperlukan)

Jenis: Objek

Berisi konfigurasi penerapan untuk aplikasi Anda. Objek ini dapat mencakup`msDeploy`,`aspNetCoreWeb`, dan jenis `custom` penyebaran.

`iisConfig` (opsional)

Jenis: Objek

Mendefinisikan pengaturan konfigurasi IIS untuk diterapkan sebelum menerapkan aplikasi. Saat ini mendukung konfigurasi kumpulan aplikasi.

Konfigurasi IIS

`iisConfig`Bagian ini memungkinkan Anda untuk mengkonfigurasi pengaturan IIS sebelum menerapkan aplikasi Anda. Ini sangat berguna untuk menyiapkan kumpulan aplikasi dengan konfigurasi tertentu.

Kolam aplikasi

Kumpulan aplikasi menyediakan isolasi antar aplikasi dan memungkinkan Anda mengonfigurasi pengaturan runtime untuk grup aplikasi.

Example Konfigurasi kolam aplikasi

```
{
  "iisConfig": {
    "appPools": [
      {
        "name": "MyAppPool",
        "enable32Bit": false,
        "managedPipelineMode": "Integrated",
        "managedRuntimeVersion": "v4.0",
        "queueLength": 1000,
        "cpu": {
          "limitPercentage": 80,
          "limitAction": "Throttle",
          "limitMonitoringInterval": 5
        },
        "recycling": {
          "regularTimeInterval": 1440,
          "requestLimit": 10000,
          "memory": 1048576,
          "privateMemory": 524288
```

```
        }
      }
    ]
  }
}
```

Properti kolam aplikasi

`name`(diperlukan)

Tipe: String

Nama kolam aplikasi. Nama ini digunakan untuk mereferensikan kumpulan dalam konfigurasi penerapan.

`enable32Bit` (opsional)

Jenis: Boolean

Memungkinkan aplikasi 32-bit untuk berjalan pada versi 64-bit Windows. Setel ke aplikasi `true` lama yang membutuhkan kompatibilitas 32-bit.

`managedPipelineMode` (opsional)

Tipe: String

Nilai yang valid: “Terintegrasi”, “Klasik”

Menentukan modus permintaan-pemrosesan untuk kumpulan aplikasi.

`managedRuntimeVersion` (opsional)

Tipe: String

Nilai yang valid: “Tidak Ada Kode Terkelola”, “v2.0", “v4.0"

Menentukan versi.NET Framework untuk kolam aplikasi.

`queueLength` (opsional)

Tipe: Integer

Jumlah maksimum permintaan yang HTTP.sys antrian untuk kumpulan aplikasi sebelum menolak permintaan tambahan.

Konfigurasi CPU

`cpu`Objek mengonfigurasi batas penggunaan CPU dan pemantauan untuk kumpulan aplikasi.

`limitPercentage` (opsional)

Jenis: Angka

Persentase maksimum waktu CPU yang dapat dikonsumsi oleh proses pekerja di kumpulan aplikasi.

`limitAction` (opsional)

Tipe: String

Nilai yang valid: "NoAction“, “KillW3wp”, “Throttle”, "” ThrottleUnderLoad

Tindakan yang harus diambil ketika batas CPU tercapai.

`limitMonitoringInterval` (opsional)

Jenis: Angka

Atur ulang periode (dalam menit) untuk pemantauan CPU dan batas pelambatan.

Konfigurasi daur ulang

`recycling`Objek mengonfigurasi kapan dan bagaimana proses pekerja kumpulan aplikasi didaur ulang.

`regularTimeInterval` (opsional)

Tipe: Integer

Interval waktu (dalam menit) setelah itu kolam aplikasi didaur ulang. Setel ke 0 untuk menonaktifkan daur ulang berbasis waktu.

`requestLimit` (opsional)

Tipe: Integer

Jumlah maksimum permintaan yang diproses oleh kumpulan aplikasi sebelum didaur ulang.

`memory` (opsional)

Tipe: Integer

Jumlah memori virtual (dalam kilobyte) yang memicu daur ulang proses pekerja.

`privateMemory` (opsional)

Tipe: Integer

Jumlah memori pribadi (dalam kilobyte) yang memicu daur ulang proses pekerja.

Jenis deployment

`deployments`Objek berisi array konfigurasi penerapan untuk berbagai jenis aplikasi. Setiap jenis penerapan memiliki properti dan kasus penggunaan tertentu.

MSDeploy penyebaran

MSDeploy deployment digunakan untuk aplikasi.NET Framework tradisional yang dapat digunakan menggunakan Web Deploy (). MSDeploy

Example MSDeploy konfigurasi penerapan

```
{
  "deployments": {
    "msDeploy": [
      {
        "name": "WebApp",
        "description": "Main web application",
        "parameters": {
          "appBundle": "webapp.zip",
          "iisPath": "/",
          "appPool": "DefaultAppPool"
        }
      }
    ]
  }
}
```

MSDeploy properti penyebaran

`name`(diperlukan)

Tipe: String

Nama unik untuk penyebaran. Nama ini harus unik di semua penerapan dalam manifes.

`description` (opsional)

Tipe: String

Deskripsi penyebaran yang dapat dibaca manusia.

`parameters`(diperlukan)

Jenis: Objek

Parameter konfigurasi untuk MSDeploy operasi.

`scripts` (opsional)

Jenis: Objek

PowerShell skrip untuk dijalankan pada berbagai tahap siklus hidup penerapan.

MSDeploy parameter

`appBundle`(diperlukan)

Tipe: String

Path ke bundel aplikasi (file ZIP) relatif terhadap file manifes. Bundel ini berisi file aplikasi untuk menyebarkan.

`iisPath` (opsional)

Tipe: String

Default: “/”

Jalur direktori virtual di IIS tempat aplikasi akan digunakan. Gunakan “/” untuk jalur root atau “/api” untuk subdirektori.

`appPool` (opsional)

Tipe: String

Nama kumpulan aplikasi untuk menjalankan aplikasi ini.

Penerapan ASP.NET Core

Penerapan ASP.NET Core dirancang khusus untuk aplikasi.NET Core dan .NET 5+.

Example Konfigurasi penerapan ASP.NET Core

```
{
  "deployments": {
    "aspNetCoreWeb": [
      {
        "name": "CoreAPI",
        "description": "ASP.NET Core Web API",
        "parameters": {
          "appBundle": "coreapi.zip",
          "iisPath": "/api",
          "appPool": "CoreAppPool"
        }
      }
    ]
  }
}
```

Penerapan ASP.NET Core menggunakan struktur properti yang sama dengan MSDeploy penerapan, dengan perbedaan utamanya adalah lingkungan runtime dan model hosting yang digunakan untuk aplikasi.

Parameter penyebaran ASP.NET Core

`appBundle`(diperlukan)

Tipe: String

Path ke bundel aplikasi relatif terhadap file manifes. Ini bisa berupa arsip ZIP atau jalur direktori yang berisi aplikasi ASP.NET Core yang diterbitkan.

`iisPath` (opsional)

Tipe: String

Default: “/”

Jalur direktori virtual di IIS untuk aplikasi ASP.NET Core.

`appPool` (opsional)

Tipe: String

Kumpulan aplikasi untuk aplikasi ASP.NET Core. Pool akan dikonfigurasi dengan tepat untuk hosting ASP.NET Core.

Penerapan khusus

Penerapan khusus memberikan kontrol penuh atas proses penyebaran melalui skrip. PowerShell Jenis penerapan ini berguna untuk skenario kompleks yang memerlukan instalasi, konfigurasi, atau logika penerapan khusus.

Example Konfigurasi penerapan khusus

```
{
  "deployments": {
    "custom": [
      {
        "name": "CustomService",
        "description": "Custom Windows service deployment",
        "architecture": 32,
        "scripts": {
          "install": {
            "file": "install-service.ps1"
          },
          "restart": {
            "file": "restart-service.ps1"
          },
          "uninstall": {
            "file": "uninstall-service.ps1",
            "ignoreErrors": true
          }
        }
      }
    ]
  }
}
```

Properti penyebaran kustom

`name`(diperlukan)

Tipe: String

Nama unik untuk penerapan kustom.

`description` (opsional)

Tipe: String

Deskripsi penerapan kustom.

`architecture` (opsional)

Tipe: Integer

Default: 32

Nilai yang valid: 32, 64

Spesifikasi arsitektur untuk mode eksekusi skrip PowerShell

`scripts`(diperlukan)

Jenis: Objek

PowerShell skrip yang menentukan perilaku penerapan. Penerapan khusus mendukung jenis skrip tambahan dibandingkan dengan jenis penerapan lainnya.

Skrip penyebaran

Skrip penerapan adalah PowerShell skrip yang berjalan pada titik-titik tertentu selama siklus hidup penerapan. Jenis penyebaran yang berbeda mendukung kumpulan peristiwa skrip yang berbeda.

Acara skrip

Peristiwa skrip berikut tersedia tergantung pada jenis penerapan:

Skrip penerapan standar (MSDeploy dan Web) aspNetCore

`preInstall`

Berjalan sebelum aplikasi diinstal atau diperbarui.

`postInstall`

Berjalan setelah aplikasi diinstal atau diperbarui.

`preRestart`

Berjalan sebelum aplikasi dimulai ulang.

`postRestart`

Berjalan setelah aplikasi dimulai ulang.

`preUninstall`

Berjalan sebelum aplikasi dihapus.

`postUninstall`

Berjalan setelah aplikasi dihapus.

Skrip penerapan khusus (hanya penerapan khusus)

`install`

Skrip instalasi utama untuk penerapan kustom. Skrip ini bertanggung jawab untuk menginstal aplikasi atau layanan.

`restart`

Script untuk me-restart aplikasi atau layanan. Disebut ketika lingkungan dimulai ulang.

`uninstall`

Script untuk menghapus instalasi aplikasi atau layanan. Dipanggil selama penghentian lingkungan atau penghapusan aplikasi.

Properti skrip

Setiap skrip didefinisikan sebagai objek dengan properti berikut:

`file`(diperlukan)

Tipe: String

Path ke file PowerShell skrip relatif terhadap file manifes. Skrip harus memiliki `.ps1` ekstensi.

`ignoreErrors` (opsional)

Jenis: Boolean

Default: false

Saat disetel ke`true`, penerapan berlanjut meskipun skrip gagal. Gunakan ini untuk skrip non-kritis atau operasi pembersihan.

Example Contoh konfigurasi skrip

```
{
  "scripts": {
    "preInstall": {
      "file": "backup-config.ps1",
      "ignoreErrors": true
    },
    "postInstall": {
      "file": "configure-app.ps1"
    }
  }
}
```

Menggunakan Peluncuran EC2 Cepat dengan cabang platform Windows

Fitur Peluncuran EC2 Cepat mengurangi waktu peluncuran instans Windows di lingkungan Elastic Beanstalk Anda. Tujuan dari topik ini adalah untuk memandu Anda menggunakan fitur ini dengan lingkungan Elastic Beanstalk Anda. Dimulai dengan platform Windows versi 2.16.2, dirilis pada 22 Januari 2025, rilis platform Elastic Beanstalk menyertakan basis dengan Peluncuran Cepat diaktifkan. AMIs EC2

Ketersediaan Peluncuran EC2 Cepat Default

Versi platform Elastic Beanstalk Windows terbaru AMIs menyertakan EC2 basis dengan Peluncuran Cepat diaktifkan secara otomatis, tanpa biaya tambahan. Namun, ketika versi platform yang lebih baru dirilis, Peluncuran EC2 Cepat mungkin tidak tetap diaktifkan secara otomatis berdasarkan AMIs versi platform yang lebih lama.

Kami merekomendasikan untuk meningkatkan ke versi platform Windows terbaru untuk menggunakan basis AMIs dengan Peluncuran EC2 Cepat diaktifkan secara otomatis. Namun, jika Anda perlu terus menggunakan versi platform yang ada, Anda dapat mengaktifkan Peluncuran EC2 Cepat secara manual di AMI dasar lingkungan Anda. Untuk petunjuk, silakan lihat Mengkonfigurasi Peluncuran EC2 Cepat secara manual.

Mengkonfigurasi Peluncuran EC2 Cepat secara manual

> **Note**
>
> Mengaktifkan Peluncuran EC2 Cepat secara manual dapat menimbulkan biaya tambahan dibandingkan dengan menggunakan versi platform dengan Peluncuran EC2 Cepat diaktifkan

secara otomatis. Untuk informasi selengkapnya tentang biaya Peluncuran EC2 Cepat, lihat halaman Mengelola biaya untuk Peluncuran EC2 Cepat sumber daya dasar di Panduan EC2 Pengguna Amazon.

Ikuti langkah-langkah ini untuk mengaktifkan Peluncuran EC2 Cepat pada AMI basis Windows yang digunakan oleh lingkungan Elastic Beanstalk Anda:

Untuk mengaktifkan Peluncuran EC2 Cepat secara manual untuk lingkungan Elastic Beanstalk Anda

1. Identifikasi AMI dasar lingkungan Anda:

   Ikuti langkah-langkah dalam Membuat AMI Kustom untuk mengidentifikasi ID AMI dasar lingkungan Anda. Perhatikan bahwa Anda tidak perlu membuat AMI khusus - Anda hanya perlu mengikuti langkah-langkah untuk menemukan ID AMI dasar Anda saat ini.
2. Aktifkan Peluncuran EC2 Cepat di AMI:

   Gunakan petunjuk di Aktifkan Peluncuran EC2 Cepat di Panduan EC2 Pengguna Amazon untuk mengonfigurasi Peluncuran EC2 Cepat untuk AMI Anda.

Menambahkan instans DB Amazon RDS ke lingkungan aplikasi .NET Anda

Topik ini memberikan instruksi untuk membuat Amazon RDS menggunakan konsol Elastic Beanstalk. Anda dapat menggunakan instans DB Amazon Relational Database Service (Amazon RDS) untuk menyimpan data yang dikumpulkan dan dimodifikasi oleh aplikasi Anda. Basis data dapat digabungkan ke lingkungan Anda dan dikelola oleh Elastic Beanstalk, atau dapat dibuat sebagai dipisahkan dan dikelola secara eksternal oleh layanan lain. Dalam instruksi ini database digabungkan dengan lingkungan Anda dan dikelola oleh Elastic Beanstalk. Untuk informasi lebih lanjut tentang mengintegrasikan Amazon RDS dengan Elastic Beanstalk, lihat. Menambahkan basis data ke lingkungan Elastic Beanstalk Anda

Bagian-bagian

- Menambahkan instans DB ke lingkungan Anda
- Mengunduh driver
- Menghubungkan ke basis data

Menambahkan instans DB ke lingkungan Anda

Untuk menambahkan instans DB ke lingkungan Anda

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Di kategori konfigurasi Basis data, pilih Edit.
5. Pilih mesin DB, dan masukkan nama pengguna dan kata sandi.
6. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Menambahkan instans DB memakan waktu sekitar 10 menit. Ketika pembaruan lingkungan selesai, nama host instans DB dan informasi koneksi lainnya tersedia untuk aplikasi Anda melalui properti lingkungan berikut:

Nama properti	Deskripsi	Nilai properti
`RDS_HOSTNAME`	Nama host instans DB.	Di tab Konektivitas & keamanan di konsol Amazon RDS: Titik akhir.
`RDS_PORT`	Port tempat instans DB menerima koneksi. Nilai default bervariasi di antara mesin DB.	Di tab Konektivitas & keamanan di konsol Amazon RDS: Port.
`RDS_DB_NAME`	Nama basis data, **ebdb**.	Di tab Konfigurasi di konsol Amazon RDS: Nama DB.
`RDS_USERNAME`	Nama pengguna yang Anda konfigurasi untuk basis data Anda.	Di tab Konfigurasi di konsol Amazon RDS: Nama pengguna utama.
`RDS_PASSWORD`	Kata sandi yang Anda konfigurasi untuk basis data Anda.	Tidak tersedia untuk referensi di konsol Amazon RDS.

Untuk informasi selengkapnya tentang mengonfigurasi instance database yang digabungkan dengan lingkungan Elastic Beanstalk, lihat. Menambahkan basis data ke lingkungan Elastic Beanstalk Anda

Mengunduh driver

Unduh dan instal paket `EntityFramework` dan driver basis data untuk lingkungan pengembangan Anda dengan `NuGet`.

Penyedia basis data kerangka kerja entitas umum untuk .NET

- Server SQL – `Microsoft.EntityFrameworkCore.SqlServer`
- MySQL – `Pomelo.EntityFrameworkCore.MySql`
- PostgreSQL – `Npgsql.EntityFrameworkCore.PostgreSQL`

Menghubungkan ke basis data

Elastic Beanstalk memberikan informasi koneksi untuk instans DB terlampir di properti lingkungan. Gunakan `ConfigurationManager.AppSettings` untuk membaca properti dan mengonfigurasi koneksi basis data.

Example Helpers.cs - metode string koneksi

```
using System;
using System.Collections.Generic;
using System.Configuration;
using System.Linq;
using System.Web;

namespace MVC5App.Models
{
  public class Helpers
  {
    public static string GetRDSConnectionString()
    {
      var appConfig = ConfigurationManager.AppSettings;

      string dbname = appConfig["RDS_DB_NAME"];

      if (string.IsNullOrEmpty(dbname)) return null;
```

```
      string username = appConfig["RDS_USERNAME"];
      string password = appConfig["RDS_PASSWORD"];
      string hostname = appConfig["RDS_HOSTNAME"];
      string port = appConfig["RDS_PORT"];

      return "Data Source=" + hostname + ";Initial Catalog=" + dbname + ";User ID=" +
 username + ";Password=" + password + ";";
    }
  }
}
```

Gunakan string koneksi untuk menginisialisasi konteks basis data Anda.

Example DBContext.cs

```
using System.Data.Entity;
using System.Security.Claims;
using System.Threading.Tasks;
using Microsoft.AspNet.Identity;
using Microsoft.AspNet.Identity.EntityFramework;

namespace MVC5App.Models
{
  public class RDSContext : DbContext
  {
    public RDSContext()
      : base(GetRDSConnectionString())
    {
    }

    public static RDSContext Create()
    {
      return new RDSContext();
    }
  }
}
```

The AWS Toolkit for Visual Studio

Visual Studio menyediakan templat untuk bahasa pemrograman yang berbeda dan tipe aplikasi. Anda dapat mulai dengan salah satu templat ini. AWS Toolkit for Visual Studio juga menyediakan tiga

template proyek yang mem-bootstrap pengembangan aplikasi Anda AWS : Console Project AWS , Web Project, AWS dan Empty Project. Untuk contoh ini, Anda akan membuat Aplikasi Web ASP.NET baru.

Untuk membuat proyek aplikasi web ASP.NET baru

1. Di Visual Studio, di menu File, klik Baru dan kemudian klik Proyek.
2. Di kotak dialog Proyek Baru, klik Templat Terinstal, klik Visual C#, dan kemudian klik Web. Klik Aplikasi Web Kosong ASP.NET, ketik nama proyek, lalu klik OK.

Untuk menjalankan proyek

Lakukan salah satu dari berikut ini:

1. Tekan F5.
2. Pilih Mulai Debugging dari menu Debug.

Uji secara lokal

Visual Studio memudahkan Anda untuk menguji aplikasi Anda secara lokal. Untuk menguji atau menjalankan aplikasi web ASP.NET, Anda memerlukan web server. Visual Studio menawarkan beberapa opsi, seperti Internet Information Services (IIS), IIS Express, atau Visual Studio Development Server bawaan. Untuk mempelajari tentang masing-masing opsi ini dan memutuskan mana yang terbaik untuk Anda, lihat Web Server di Visual Studio untuk Proyek Web ASP.NET.

Membuat lingkungan Elastic Beanstalk

Setelah menguji aplikasi Anda, Anda siap untuk men-deploynya ke Elastic Beanstalk.

> **Note**
>
> File konfigurasi perlu menjadi bagian dari proyek agar dimasukkan dalam arsip. Atau, alih-alih menyertakan file konfigurasi di proyek, Anda dapat menggunakan Visual Studio untuk men-deploy semua file di folder proyek. Di Penjelajah Solusi, klik kanan nama proyek, dan kemudian klik Properti. Klik tab Paket/Publikasikan Web. Di bagian Item yang di-deploy, pilih Semua file di Folder Proyek di daftar drop-down.

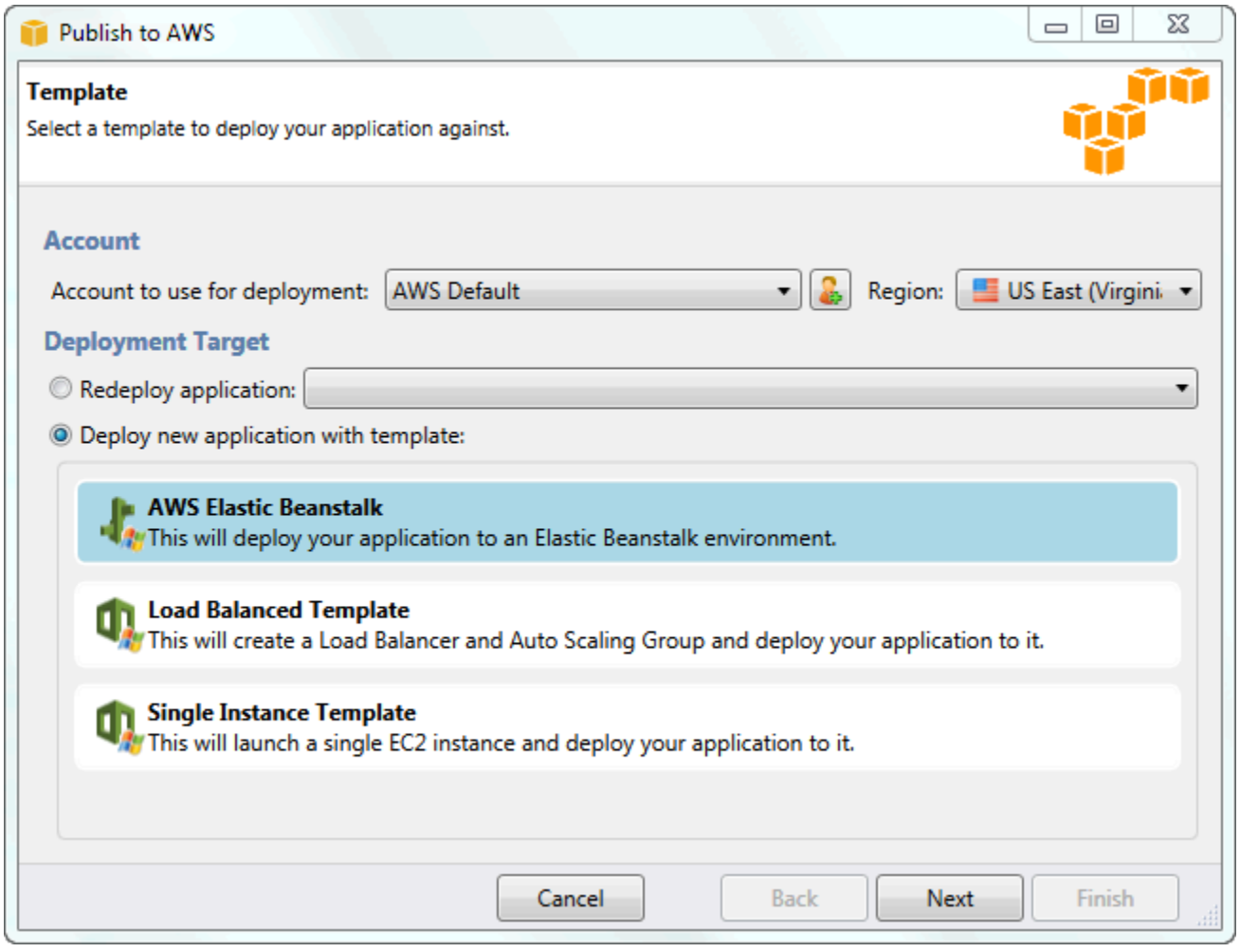
Untuk menyebarkan aplikasi Anda ke Elastic AWS Beanstalk menggunakan toolkit untuk Visual Studio

1. Di Solution Explorer, klik kanan aplikasi Anda dan kemudian pilih Publish to AWS.
2. Di wizard Publikasikan ke AWS, masukkan informasi akun Anda.

   a. Untuk akun AWS yang akan digunakan untuk deployment, pilih akun Anda atau pilih Lainnya untuk memasukkan informasi akun baru.

   b. Untuk Wilayah, pilih Wilayah tempat Anda ingin men-deploy aplikasi Anda. Untuk informasi tentang AWS Wilayah yang tersedia, lihat AWS Elastic Beanstalk Titik Akhir dan Kuota di. Referensi Umum AWS Jika Anda memilih wilayah yang tidak didukung oleh Elastic Beanstalk, maka opsi untuk men-deploy ke Elastic Beanstalk tidak tersedia.

   c. Klik Deploy aplikasi baru dengan templat dan pilih Elastic Beanstalk. Kemudian klik Selanjutnya.



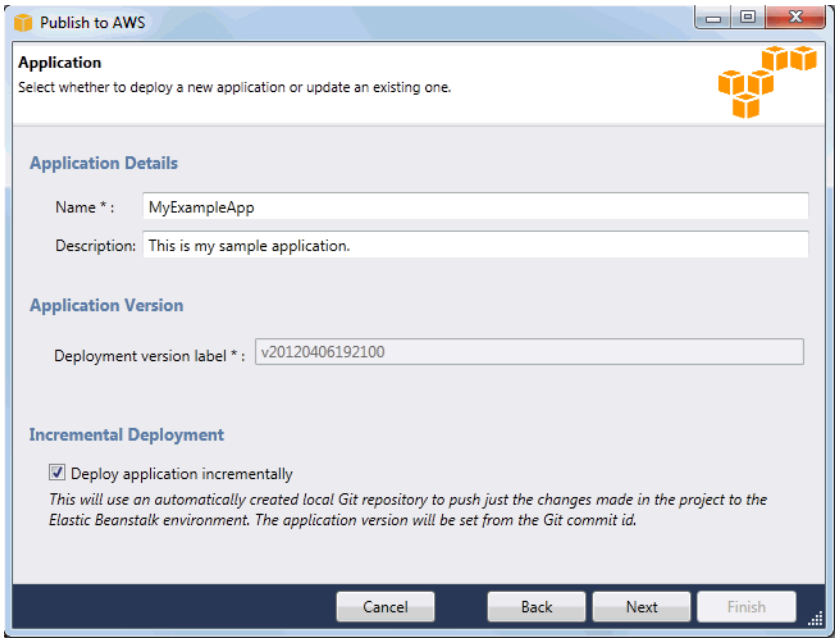
3. Di halaman Aplikasi, masukkan detail aplikasi Anda.

a. Untuk Nama, ketik nama aplikasi.

b. Untuk Deskripsi, ketik deskripsi aplikasi. Langkah ini opsional.

c. Label versi aplikasi secara otomatis muncul di Label versi deployment.

d. Pilih Deploy aplikasi secara bertahap untuk men-deploy file yang diubah saja. Deployment tambahan lebih cepat karena Anda memperbarui hanya file yang berubah, bukan semua file. Jika Anda memilih opsi ini, versi aplikasi akan diatur dari Git commit ID. Jika Anda memilih untuk tidak men-deploy aplikasi Anda secara bertahap, maka Anda dapat memperbarui label versi di kotak Label versi deployment.



e. Klik Berikutnya.

4. Di halaman Lingkungan, jelaskan detail lingkungan Anda.

a. Pilih Buat lingkungan baru untuk aplikasi ini.

b. Untuk Nama, ketik nama untuk lingkungan Anda.

c. Untuk Deskripsi, cirikan lingkungan anda. Langkah ini opsional.

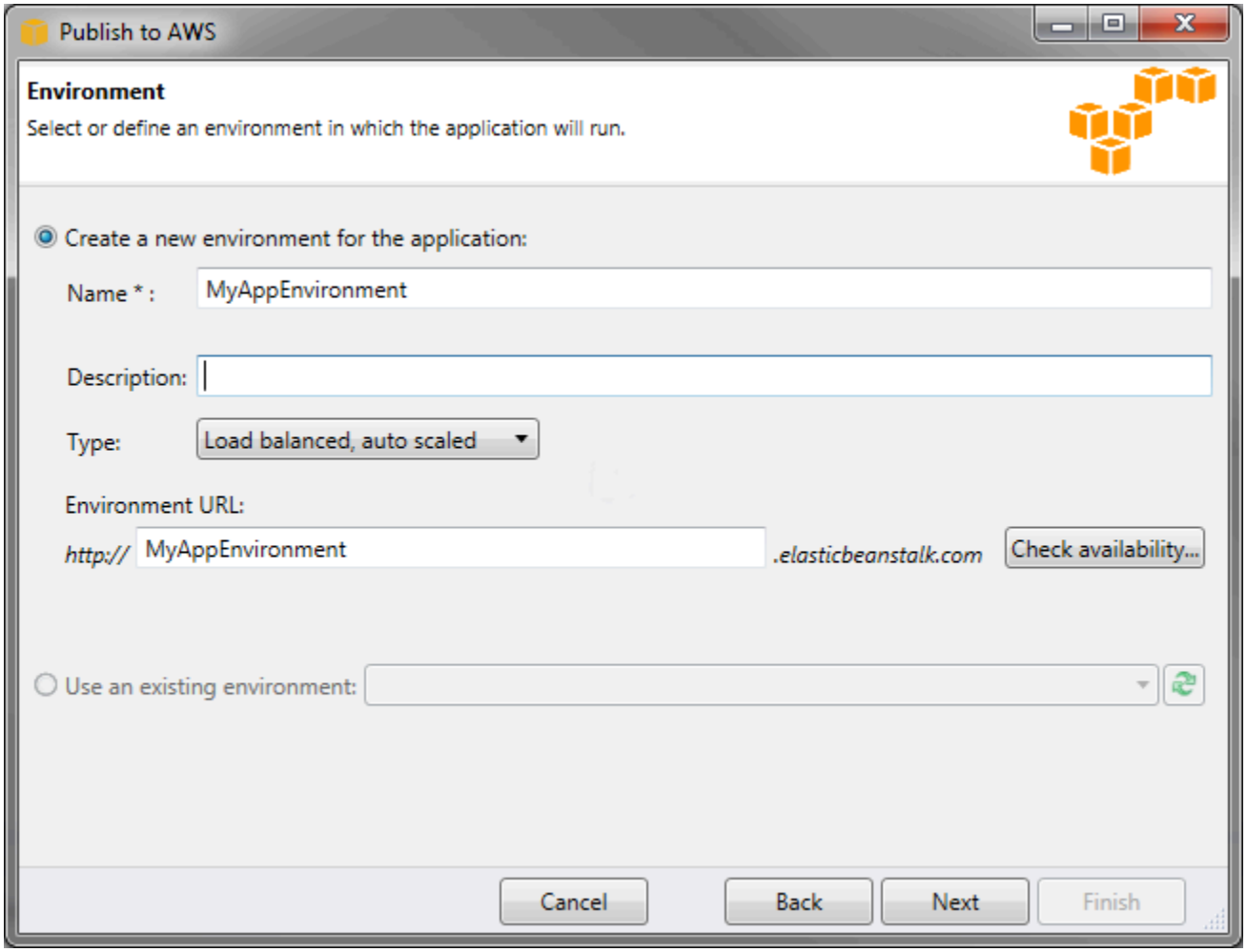
d. Pilih Tipe lingkungan yang Anda inginkan.

Anda dapat memilih Beban seimbang, diskalakan otomatis atau lingkungan Instans tunggal. Untuk informasi selengkapnya, lihat Jenis lingkungan.

> **Note**
>
> Untuk lingkungan instans tunggal, penyeimbangan beban, penskalaan otomatis, dan pemeriksaan kondisi pengaturan URL tidak berlaku.

e. URL lingkungan secara otomatis muncul di Lingkungan URL setelah Anda memindahkan kursor Anda ke kotak tersebut.

f. Klik Periksa ketersediaan untuk memastikan URL lingkungan tersedia.



g. Klik Berikutnya.

5. Di halaman Opsi AWS , konfigurasikan opsi tambahan dan informasi keamanan untuk deployment Anda.

   a. Untuk Tipe Kontainer, pilih 64bit Windows Server 2012 menjalankan IIS 8 atau 64bit Windows Server 2008 menjalankan IIS 7.5.

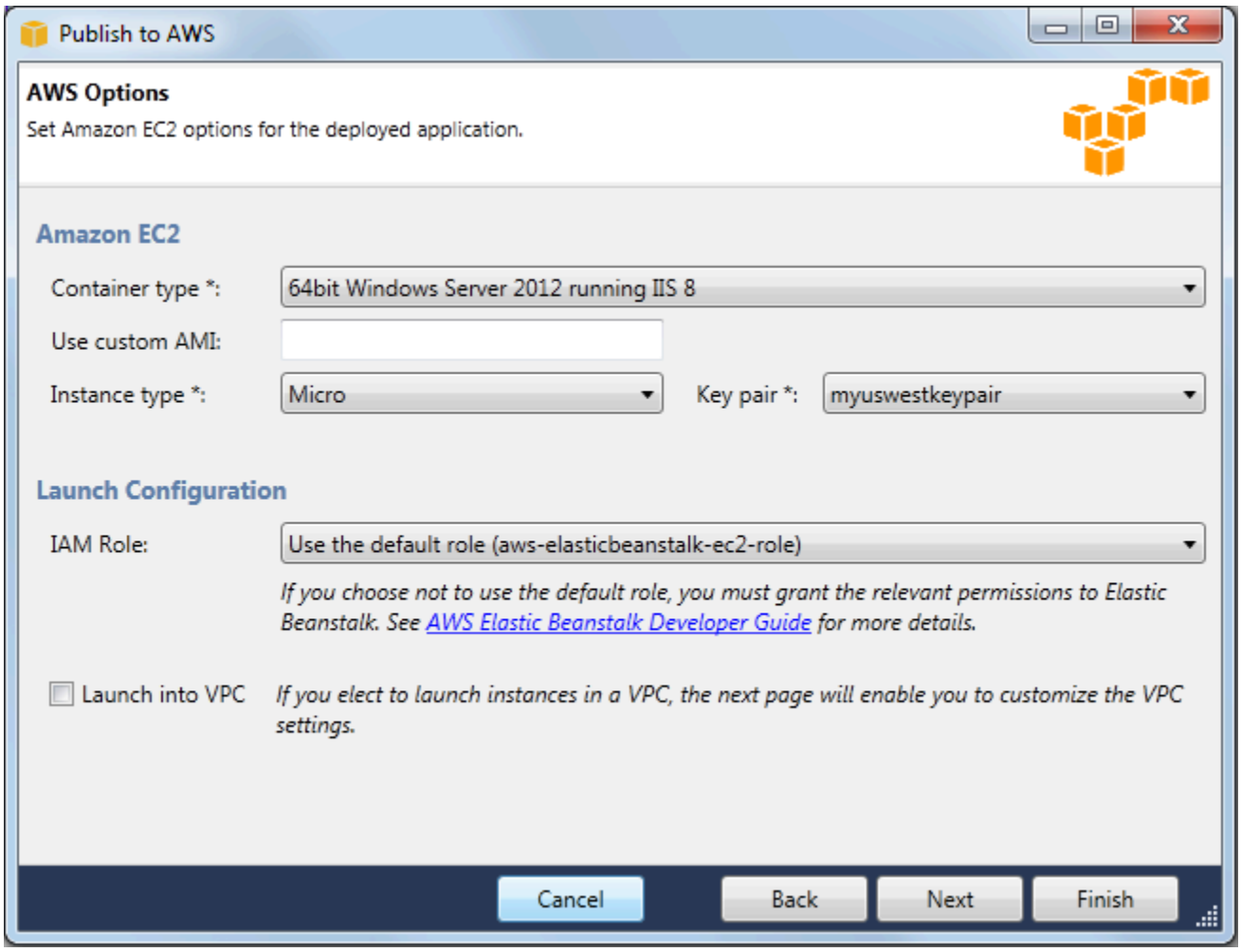
b. Untuk Tipe Instans, pilih Micro.

c. Untuk Pasangan kunci, pilih Buat pasangan kunci baru. Ketik nama untuk pasangan kunci baru—di contoh ini, kita menggunakan **`myuswestkeypair`**—dan kemudian klik OK. Sebuah key pair memungkinkan akses remote desktop ke instans Amazon Anda. EC2 Untuk informasi selengkapnya tentang pasangan EC2 kunci Amazon, lihat Menggunakan Kredensial di Panduan Pengguna Amazon Elastic Compute Cloud.

d. Pilih profil instans.

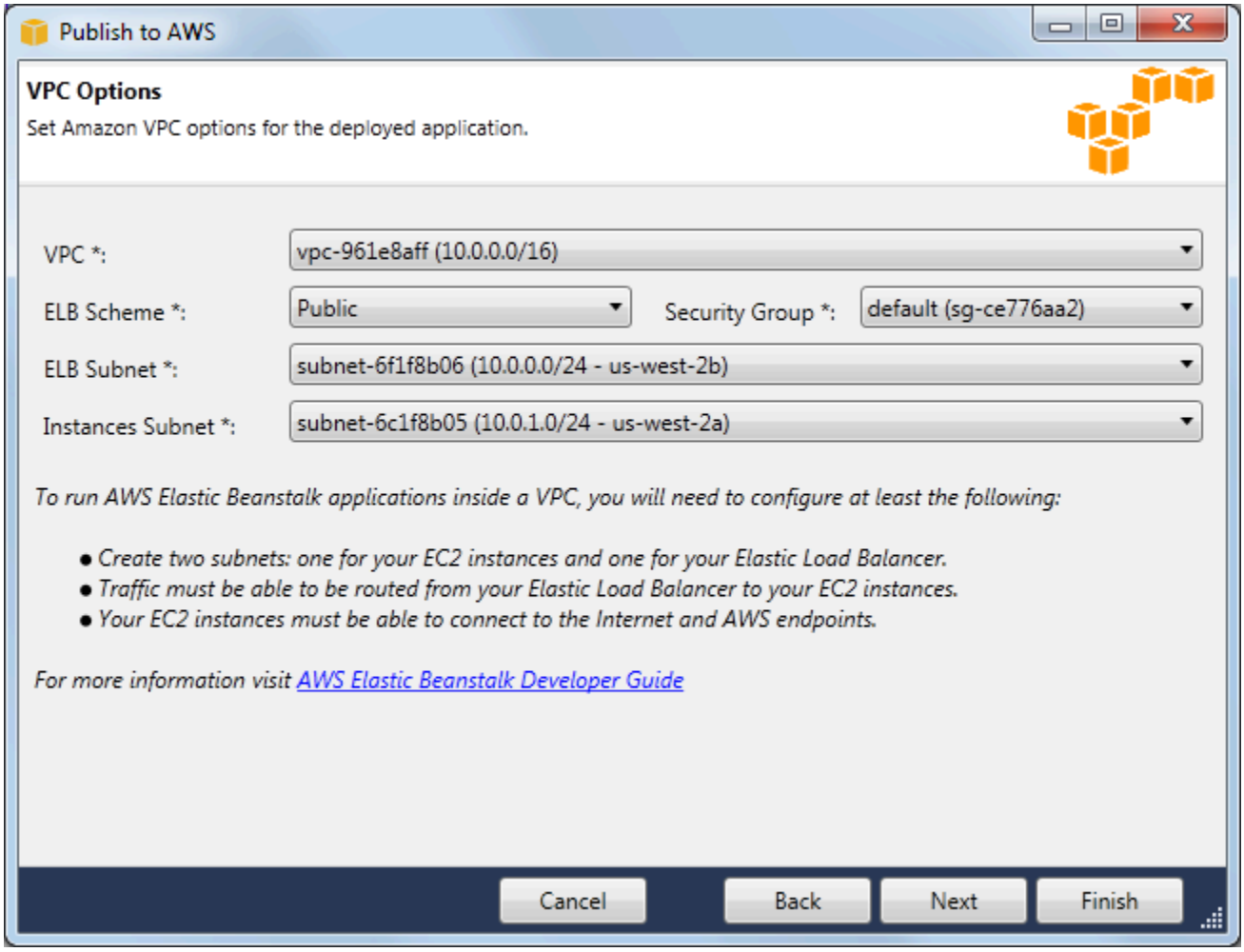
   Jika Anda tidak memiliki profil instans, pilih Buat profil instans default. Untuk informasi tentang menggunakan profil instans dengan Elastic Beanstalk, lihat Mengelola profil instans Elastic Beanstalk.

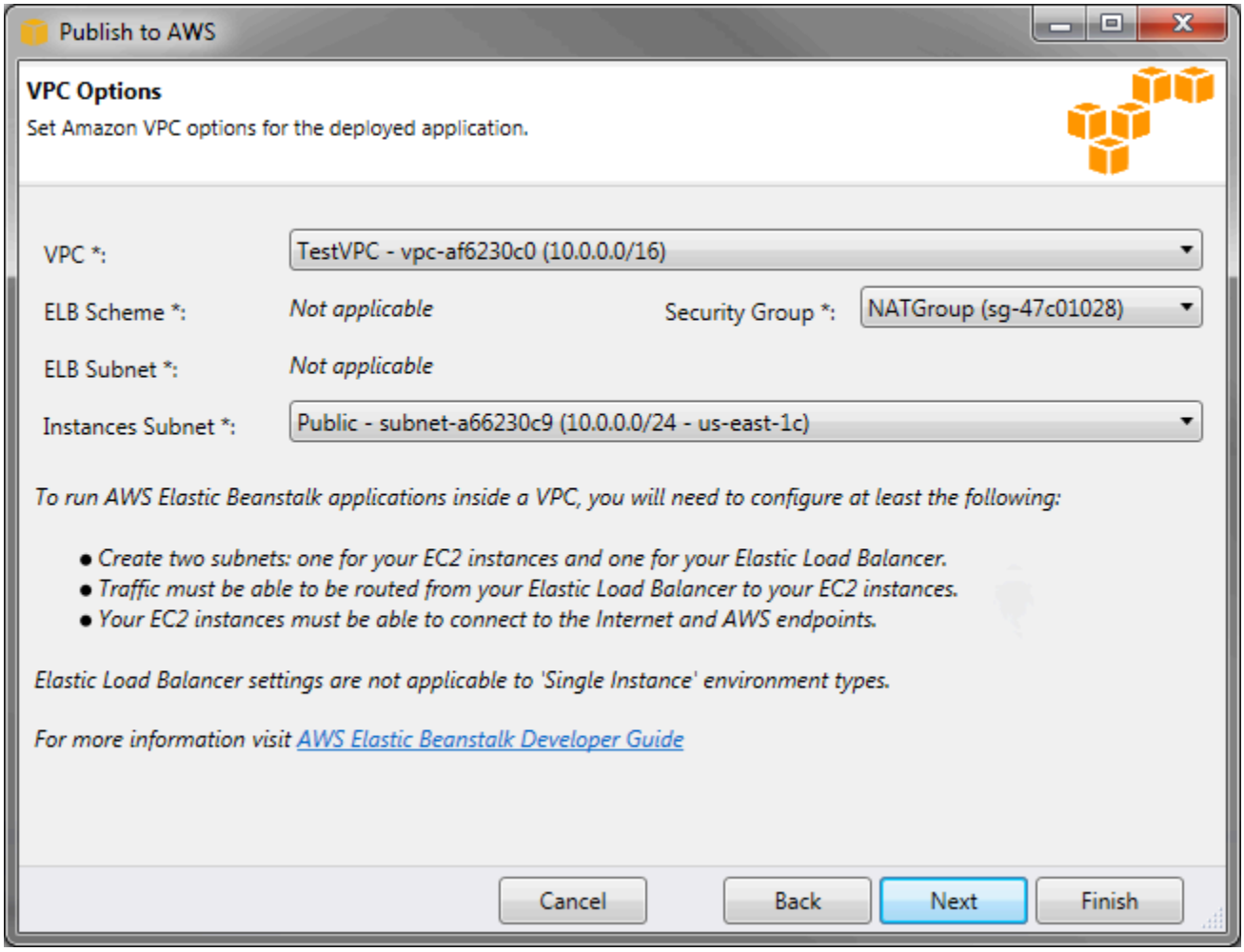
e. Jika Anda memiliki VPC khusus yang ingin Anda gunakan dengan lingkungan Anda, klik Luncurkan ke VPC. Anda dapat mengonfigurasi informasi VPC di halaman berikutnya. Untuk informasi selengkapnya tentang Amazon VPC, lanjutkan ke Amazon Virtual Private Cloud (Amazon VPC). Untuk daftar tipe kontainer bukan warisan yang didukung, lihat the section called “Mengapa beberapa versi platform ditandai sebagai legasi?”



f. Klik Berikutnya.

6. Jika Anda memilih untuk meluncurkan lingkungan Anda di dalam VPC, Opsi VPC muncul; jika tidak, halaman Opsi tambahan akan muncul. Di sini Anda akan mengonfigurasi opsi VPC Anda.


Publish to AWS
VPC Options
Set Amazon VPC options for the deployed application.
VPC *:
vpc-961e8aff (10.0.0.0/16)
ELB Scheme *:
Public
Security Group *:
default (sg-ce776aa2)
ELB Subnet *:
subnet-6f1f8b06 (10.0.0.0/24 - us-west-2b)
Instances Subnet *:
subnet-6c1f8b05 (10.0.1.0/24 - us-west-2a)
To run AWS Elastic Beanstalk applications inside a VPC, you will need to configure at least the following:
Create two subnets: one for your EC2 instances and one for your Elastic Load Balancer.
Traffic must be able to be routed from your Elastic Load Balancer to your EC2 instances.
Your EC2 instances must be able to connect to the Internet and AWS endpoints.
For more information visit AWS Elastic Beanstalk Developer Guide
Cancel
Back
Next
Finish



a. Pilih VPC ID VPC di tempat Anda ingin meluncurkan lingkungan Anda.

b. Untuk lingkungan yang seimbang dengan beban dan dapat diskalakan, pilih privat untuk Skema ELB jika Anda tidak ingin elastic load balancer Anda tersedia di Internet.

   Untuk lingkungan instans tunggal, opsi ini tidak berlaku karena lingkungan tidak memiliki penyeimbang beban. Untuk informasi selengkapnya, lihat Jenis lingkungan.

c. Untuk lingkungan yang seimbang beban dan dapat diskalakan, pilih subnet untuk penyeimbang beban elastis dan instans. EC2 Jika Anda membuat subnet publik dan pribadi, pastikan penyeimbang beban elastis dan EC2 instans dikaitkan dengan subnet yang benar. Secara default, Amazon VPC membuat subnet publik default menggunakan 10.0.0.0/24 dan subnet privat menggunakan 10.0.1.0/24. Anda dapat melihat subnet yang ada di konsol VPC Amazon di. https://console.aws.amazon.com/vpc/

Untuk lingkungan instans tunggal, VPC Anda hanya membutuhkan subnet publik untuk instans. Memilih subnet untuk penyeimbang beban tidak berlaku karena lingkungan tidak memiliki penyeimbang beban. Untuk informasi selengkapnya, lihat Jenis lingkungan.

d. Untuk lingkungan yang seimbang dengan beban dan dapat diskalakan, pilih grup keamanan yang Anda buat untuk instans Anda, jika berlaku.

   Untuk lingkungan instans tunggal, Anda tidak memerlukan perangkat NAT. Pilih grup keamanan default. Elastic Beanstalk menugaskan alamat IP elastis ke instans yang memungkinkan instans mengakses Internet.

e. Klik Berikutnya.

7. Di halaman Opsi aplikasi, konfigurasi opsi aplikasi Anda.

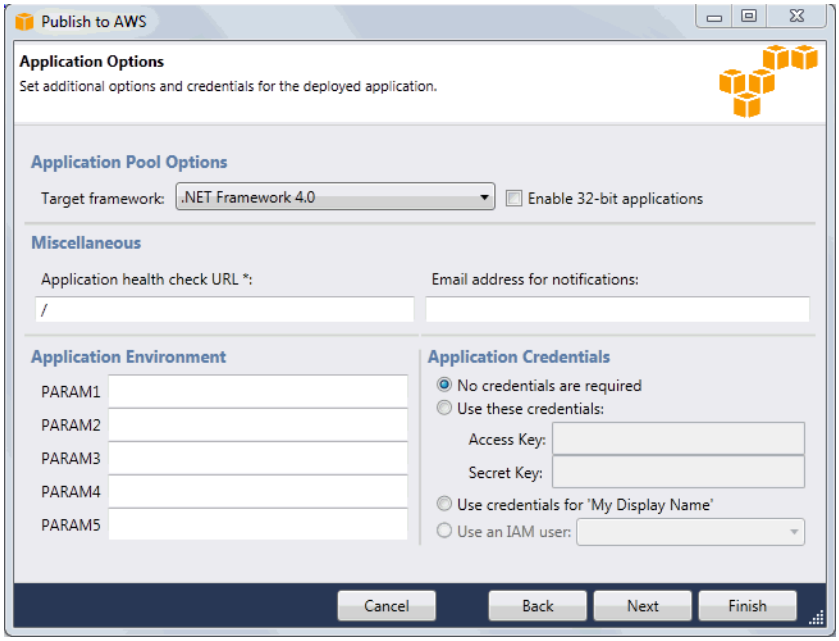
   a. Untuk kerangka kerja Target, pilih .NET Framework 4.0.

   b. Elastic Load Balancing menggunakan pemeriksaan kesehatan untuk menentukan apakah EC2 instans Amazon yang menjalankan aplikasi Anda sehat. Pemeriksaan kondisi menentukan status kondisi instans dengan memeriksa URL tertentu pada set interval. Anda dapat mengganti URL default untuk mencocokkan sumber daya yang ada di aplikasi Anda (misalnya, `/myapp/index.aspx`) dengan memasukkannya ke kotak URL pemeriksaan kondisi aplikasi. Untuk informasi selengkapnya tentang pemeriksaan kondisi aplikasi, lihat Pemeriksaan kondisi.

   c. Ketik alamat email jika Anda ingin menerima notifikasi Amazon Simple Notification Service (Amazon SNS) tentang peristiwa penting yang mempengaruhi aplikasi Anda.

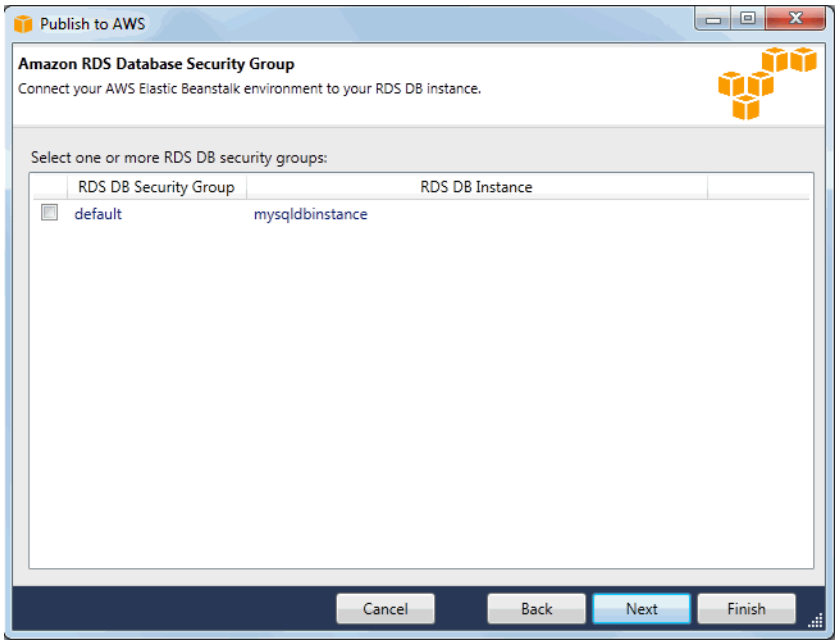
   d. Bagian Application Environment memungkinkan Anda menentukan variabel lingkungan pada EC2 instance Amazon yang menjalankan aplikasi Anda. Pengaturan ini mengaktifkan portabilitas yang lebih besar dengan menghilangkan kebutuhan untuk mengompilasi ulang kode sumber Anda saat Anda bergerak di antara lingkungan.

   e. Pilih opsi kredensial aplikasi yang ingin Anda gunakan untuk men-deploy aplikasi Anda.

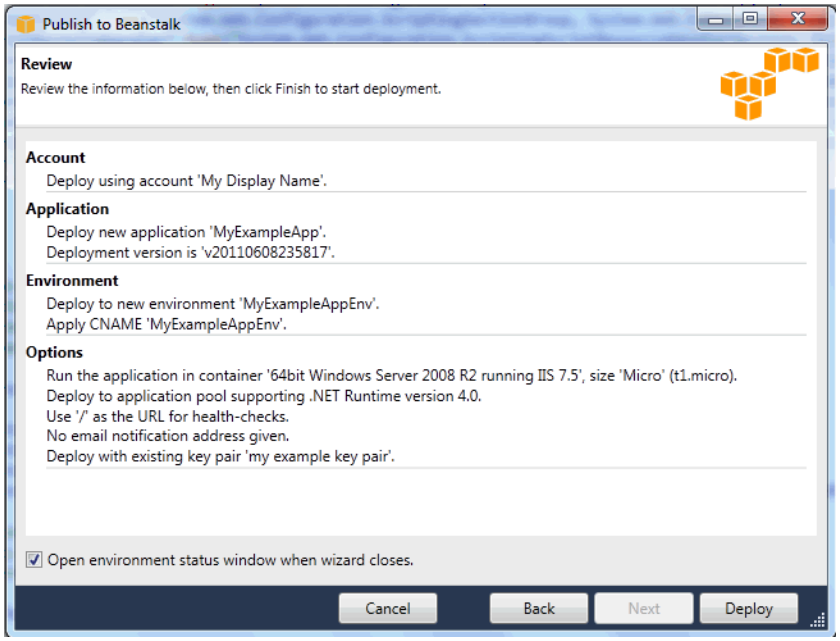


f. Klik Berikutnya.

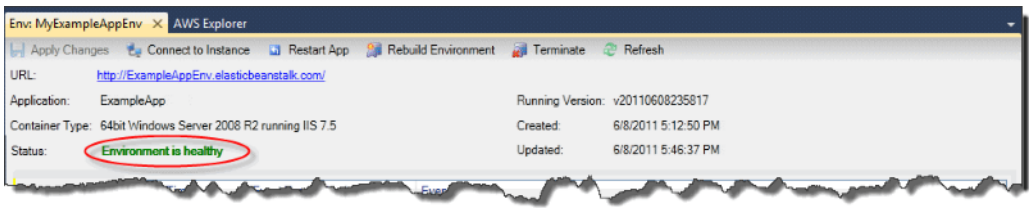
8. Jika Anda sebelumnya telah mengatur basis data Amazon RDS, halaman Grup Keamanan DB Amazon RDS akan muncul. Jika Anda ingin menghubungkan lingkungan Elastic Beanstalk Anda ke Instans DB Amazon RDS Anda, kemudian pilih satu atau lebih grup keamanan. Jika tidak, lanjutkan ke langkah berikutnya. Saat Anda siap, klik Selanjutnya.



9. Tinjau opsi deployment Anda. Jika semuanya seperti yang Anda inginkan, klik Deploy.



Proyek ASP.NET Anda akan diekspor sebagai file web deploy, diunggah ke Amazon S3, dan terdaftar sebagai versi aplikasi baru dengan Elastic Beanstalk. Fitur deployment Elastic Beanstalk akan memantau lingkungan Anda sampai tersedia dengan kode yang baru di-deploy. Di tab env:<nama lingkungan>, Anda akan melihat status untuk lingkungan Anda.



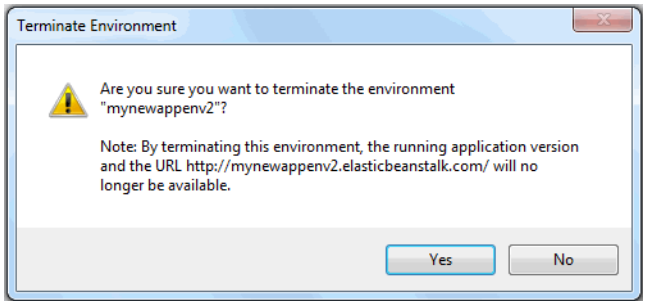
Mengakhiri lingkungan

Untuk menghindari biaya untuk AWS sumber daya yang tidak digunakan, Anda dapat menghentikan lingkungan berjalan menggunakan Toolkit for Visual AWS Studio.

> **Note**
>
> Anda selalu dapat meluncurkan lingkungan baru menggunakan versi yang sama nantinya.

Untuk mengakhiri lingkungan

1. Perluas simpul Elastic Beanstalk dan simpul aplikasi di AWS Explorer. Klik kanan lingkungan aplikasi Anda dan pilih Mengakhiri Lingkungan.
2. Saat diminta, klik Ya untuk mengonfirmasi bahwa Anda ingin mengakhiri lingkungan. Diperlukan beberapa menit bagi Elastic Beanstalk untuk AWS menghentikan sumber daya yang berjalan di lingkungan.



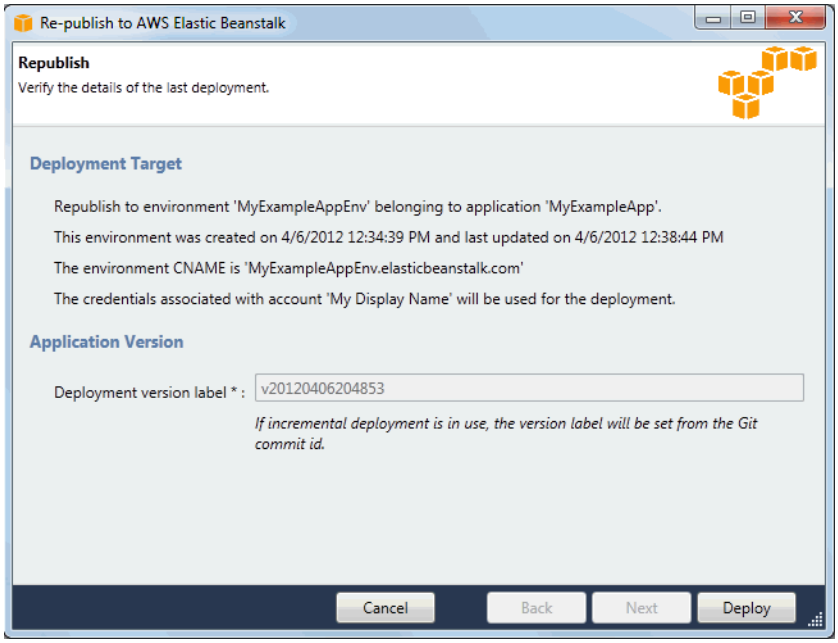
> **Note**
>
> Ketika Anda mengakhiri lingkungan Anda, CNAME yang terkait dengan lingkungan yang diakhiri akan tersedia bagi siapa saja untuk digunakan.

Men-deploy ke lingkungan Anda

Sekarang, setelah Anda menguji aplikasi Anda, mudah untuk mengedit dan men-deploy ulang aplikasi dan melihat hasilnya dalam beberapa saat.

Untuk mengedit dan menyebarkan kembali aplikasi web ASP.NET Anda

1. Di Solution Explorer, klik kanan aplikasi Anda, lalu klik Republish to Environment < > ***your environment name***. Wizard Publikasikan ulang ke AWS Elastic Beanstalk terbuka.



2. Tinjau detail deployment Anda dan klik Deploy.

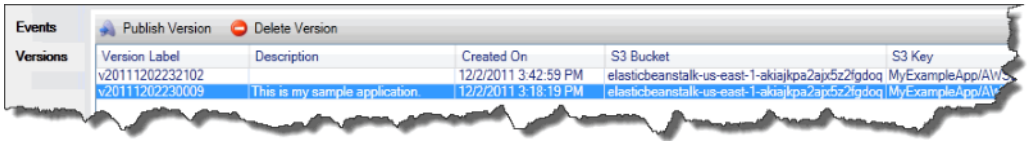
> **Note**
>
> Jika Anda ingin mengubah salah satu pengaturan, Anda dapat mengklik Batalkan dan gunakan wizard Publikasikan ke AWS sebagai gantinya. Untuk petunjuk, lihat Membuat lingkungan Elastic Beanstalk.

Proyek web ASP.NET terbaru Anda akan diekspor sebagai file web deploy dengan label versi baru, diunggah ke Amazon S3, dan terdaftar sebagai versi aplikasi baru dengan Elastic Beanstalk. Fitur deployment Elastic Beanstalk memantau lingkungan Anda yang ada sampai tersedia dengan kode yang baru di-deploy. Pada tab env: < ***environment name*** >, Anda akan melihat status lingkungan Anda.

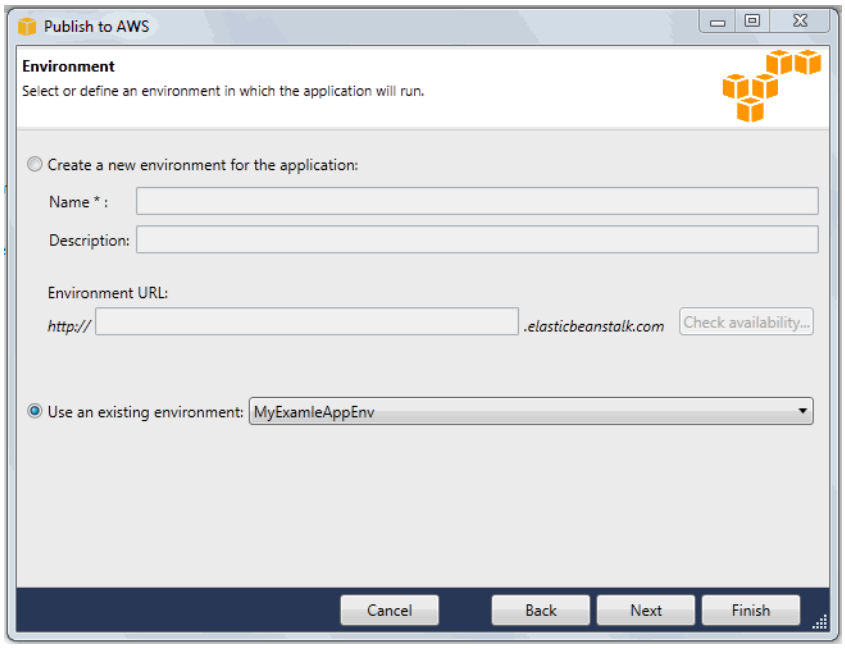
Anda juga dapat men-deploy aplikasi yang ada ke lingkungan yang ada, jika, misalnya, Anda perlu kembali ke versi aplikasi sebelumnya.

Untuk men-deploy versi aplikasi ke lingkungan yang ada

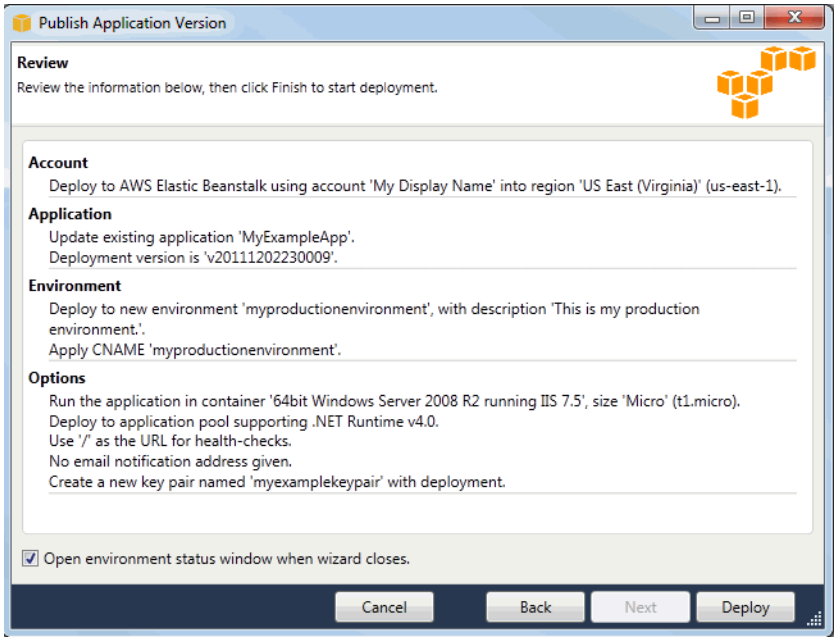
1. Klik kanan aplikasi Elastic Beanstalk Anda dengan memperluas simpul Elastic Beanstalk di AWS Explorer. Pilih Lihat Status.
2. Di tab App: < ***application name*** >, klik Versi.



3. Klik versi aplikasi yang ingin Anda deploy dan klik Publikasikan Versi.
4. Di wizard Publikasikan Versi Aplikasi, klik Selanjutnya.



5. Tinjau opsi deployment Anda, dan klik Deploy.



Proyek ASP.NET Anda akan diekspor sebagai file web deploy dan diunggah ke Amazon S3. Fitur deployment Elastic Beanstalk akan memantau lingkungan Anda sampai tersedia dengan kode yang baru di-deploy. Pada tab env: < ***environment name*** >, Anda akan melihat status untuk lingkungan Anda.

Mengelola lingkungan aplikasi Elastic Beanstalk Anda

Dengan AWS Toolkit for Visual Studio dan AWS Management Console, Anda dapat mengubah penyediaan dan konfigurasi sumber daya yang digunakan oleh AWS lingkungan aplikasi Anda. Untuk informasi tentang cara mengelola lingkungan aplikasi menggunakan AWS Management Console, lihatMenciptakan lingkungan di Elastic Beanstalk. Bagian ini membahas pengaturan layanan tertentu yang dapat Anda edit di Toolkit for Visual Studio AWS sebagai bagian dari konfigurasi lingkungan aplikasi Anda.

Mengubah pengaturan konfigurasi lingkungan

Saat Anda menerapkan aplikasi Anda, Elastic Beanstalk mengonfigurasi sejumlah layanan komputasi awan. AWS Anda dapat mengontrol bagaimana layanan individual ini dikonfigurasi menggunakan AWS Toolkit for Visual Studio.

Untuk mengedit pengaturan lingkungan aplikasi

- Perluas simpul Elastic Beanstalk dan node aplikasi Anda. Kemudian klik kanan lingkungan Elastic Beanstalk Anda di AWS Explorer. Pilih Lihat Status.

  Sekarang, Anda dapat mengonfigurasi pengaturan untuk hal berikut:

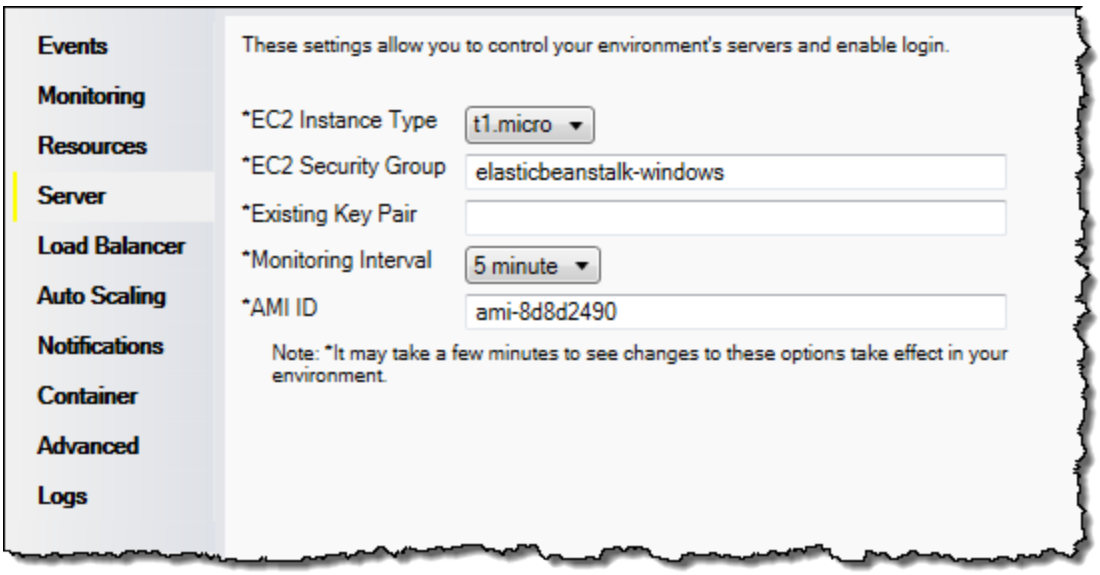
  - Server
  - Penyeimbangan beban
  - Penskalaan otomatis
  - Notifikasi
  - Properti lingkungan

Mengkonfigurasi instance EC2 server menggunakan AWS toolkit untuk Visual Studio

Amazon Elastic Compute Cloud (Amazon EC2) adalah layanan web yang Anda gunakan untuk meluncurkan dan mengelola instance server di pusat data Amazon. Anda dapat menggunakan

instans EC2 server Amazon kapan saja, selama yang Anda butuhkan, dan untuk tujuan hukum apa pun. Instans tersedia dalam berbagai ukuran dan konfigurasi. Untuk informasi lebih lanjut, buka Amazon EC2.

Anda dapat mengedit konfigurasi instans Amazon lingkungan EC2 Elastic Beanstalk dengan tab Server di dalam AWS tab lingkungan aplikasi Anda di Toolkit for Visual Studio.



Jenis EC2 instans Amazon

Tipe instans menampilkan tipe instans yang tersedia untuk aplikasi Elastic Beanstalk Anda. Ubah tipe instans untuk memilih server dengan karakteristik (termasuk ukuran memori dan kekuatan CPU) yang paling sesuai untuk aplikasi Anda. Sebagai contoh, aplikasi dengan operasi intensif dan berjalan lama dapat memerlukan lebih banyak CPU atau memori.

Untuk informasi selengkapnya tentang jenis EC2 instans Amazon yang tersedia untuk aplikasi Elastic Beanstalk, lihat Jenis Instans di Panduan Pengguna Amazon Elastic Compute Cloud.

Grup EC2 keamanan Amazon

Anda dapat mengontrol akses ke aplikasi Elastic Beanstalk Anda menggunakan EC2 Amazon Security Group. Grup keamanan mendefinisikan aturan firewall untuk instans Anda. Aturan ini menentukan lalu lintas jaringan ingress (misalnya, yang masuk) yang harus dikirim ke instans Anda. Semua lalu lintas ingress lainnya akan dibuang. Anda dapat memodifikasi aturan untuk grup kapan saja. Aturan baru secara otomatis diberlakukan untuk semua instans berjalan dan instans yang diluncurkan di masa mendatang.

Anda dapat mengatur grup EC2 keamanan Amazon menggunakan AWS Management Console atau dengan menggunakan AWS Toolkit for Visual Studio. Anda dapat menentukan akses kontrol Grup EC2 Keamanan Amazon ke aplikasi Elastic Beanstalk Anda dengan memasukkan nama satu atau beberapa nama grup keamanan EC2 Amazon (dibatasi oleh koma) ke dalam kotak teks Grup Keamanan. EC2

> **Note**
>
> Pastikan port 80 (HTTP) dapat diakses dari 0.0.0.0/0 sebagai rentang sumber CIDR jika Anda ingin mengaktifkan pemeriksaan kondisi untuk aplikasi Anda. Untuk informasi selengkapnya tentang pemeriksaan kondisi, lihat Pemeriksaan kondisi.

Untuk membuat grup keamanan menggunakan AWS toolkit untuk Visual Studio

1. Di Visual Studio, di AWS Explorer, perluas EC2 node Amazon, lalu klik dua kali Grup Keamanan.
2. Klik Buat Grup Keamanan, dan masukkan nama dan deskripsi untuk grup keamanan Anda.
3. Klik OK.

Untuk informasi selengkapnya tentang Grup EC2 Keamanan Amazon, lihat Menggunakan Grup Keamanan di Panduan Pengguna Amazon Elastic Compute Cloud.

Pasangan EC2 kunci Amazon

Anda dapat masuk dengan aman ke EC2 instans Amazon yang disediakan untuk aplikasi Elastic Beanstalk Anda dengan key pair Amazon. EC2

> **Important**
>
> Anda harus membuat EC2 key pair Amazon dan mengonfigurasi Elastic Beanstalk—instans Amazon yang disediakan untuk EC2 menggunakan key EC2 pair Amazon sebelum Anda dapat mengakses instans Amazon yang disediakan Elastic Beanstalk—Instans Amazon yang disediakan. EC2 Anda dapat membuat key pair menggunakan AWS wizard Publish to di dalam AWS Toolkit for Visual Studio saat Anda menerapkan aplikasi ke Elastic Beanstalk. Jika Anda ingin membuat pasangan kunci tambahan menggunakan Toolkit, ikuti langkah-langkah di bawah ini. Atau, Anda dapat mengatur pasangan EC2 kunci Amazon

menggunakan AWS Management Console. Untuk petunjuk cara membuat key pair untuk Amazon EC2, lihat Panduan Memulai Amazon Elastic Compute Cloud.

Kotak teks Existing Key Pair memungkinkan Anda menentukan nama EC2 key pair Amazon yang dapat Anda gunakan untuk masuk dengan aman ke EC2 instans Amazon yang menjalankan aplikasi Elastic Beanstalk Anda.

Untuk menentukan nama EC2 key pair Amazon

1. Perluas EC2 node Amazon dan klik dua kali Pasangan Kunci.
2. Klik Buat Pasangan Kunci dan masukkan nama pasangan kunci.
3. Klik OK.

Untuk informasi selengkapnya tentang pasangan EC2 kunci Amazon, buka Menggunakan EC2 Kredensial Amazon di Panduan Pengguna Amazon Elastic Compute Cloud. Untuk informasi selengkapnya tentang menghubungkan ke EC2 instans Amazon, lihatMembuat daftar dan menghubungkan ke server instans.

Interval pemantauan

Secara default, hanya CloudWatch metrik dasar Amazon yang diaktifkan. Mereka mengembalikan data dengan periode lima menit. Anda dapat mengaktifkan CloudWatch metrik satu menit yang lebih terperinci dengan memilih 1 menit untuk Interval Pemantauan di bagian Server pada tab Konfigurasi untuk lingkungan Anda di. AWS Toolkit for Eclipse

> **Note**
>
> Biaya CloudWatch layanan Amazon dapat berlaku untuk metrik interval satu menit. Lihat Amazon CloudWatch untuk informasi lebih lanjut.

ID AMI khusus

Anda dapat mengganti AMI default yang digunakan untuk EC2 instans Amazon Anda dengan AMI kustom Anda sendiri dengan memasukkan pengenal AMI kustom Anda ke dalam kotak ID AMI Kustom di bagian Server pada tab Konfigurasi untuk lingkungan Anda di. AWS Toolkit for Eclipse

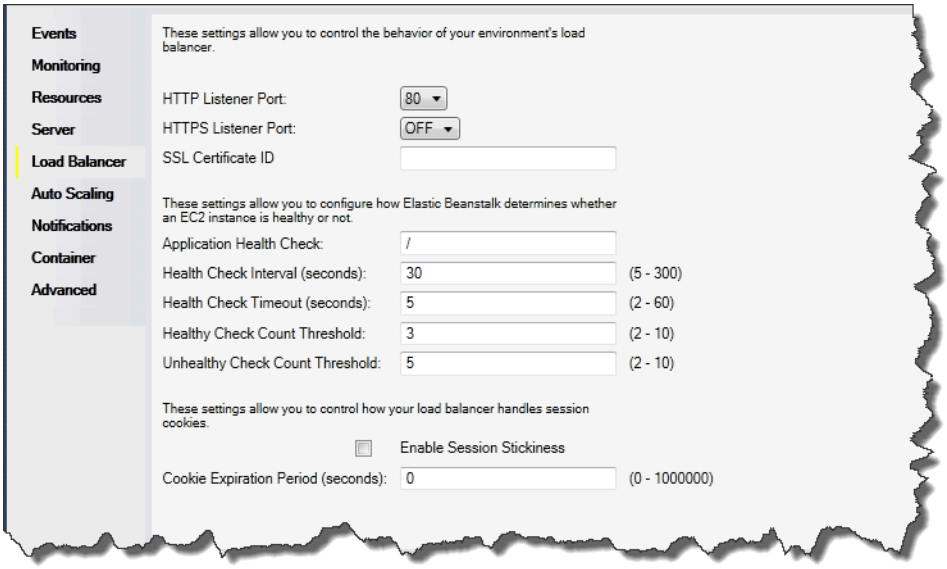
> **Important**
>
> Menggunakan AMI Anda sendiri adalah tugas lanjutan yang harus Anda lakukan dengan hati-hati. Jika Anda membutuhkan AMI khusus, kami sarankan Anda mulai dengan Elastic Beanstalk AMI default dan kemudian memodifikasinya. Agar dianggap sehat, Elastic Beanstalk EC2 mengharapkan instans Amazon memenuhi serangkaian persyaratan, termasuk memiliki manajer host yang berjalan. Jika persyaratan ini tidak terpenuhi, lingkungan Anda mungkin tidak bekerja dengan baik.

Mengkonfigurasi Elastic Load Balancing menggunakan toolkit AWS untuk Visual Studio

Elastic Load Balancing adalah layanan web Amazon yang membantu Anda meningkatkan ketersediaan dan skalabilitas aplikasi Anda. Layanan ini memudahkan Anda untuk mendistribusikan beban aplikasi antara dua atau lebih EC2 instance Amazon. Elastic Load Balancing memungkinkan ketersediaan melalui redundansi dan mendukung pertumbuhan lalu lintas untuk aplikasi Anda.

Elastic Load Balancing memungkinkan Anda secara otomatis mendistribusikan dan menyeimbangkan lalu lintas aplikasi masuk di antara semua instans yang Anda jalankan. Layanan ini juga memudahkan untuk menambahkan instans baru ketika Anda perlu meningkatkan kapasitas aplikasi Anda.

Elastic Beanstalk secara otomatis menyediakan Elastic Load Balancing saat Anda men-deploy aplikasi. Anda dapat mengedit konfigurasi instans Amazon lingkungan EC2 Elastic Beanstalk dengan tab Load Balancer di dalam tab lingkungan aplikasi Anda di Toolkit AWS for Visual Studio.



Bagian berikut menjelaskan parameter Elastic Load Balancing yang dapat Anda konfigurasi untuk aplikasi Anda.

Port

Load balancer yang disediakan untuk menangani permintaan aplikasi Elastic Beanstalk Anda mengirimkan permintaan ke instans Amazon yang menjalankan aplikasi Anda. EC2 Penyeimbang beban yang disediakan dapat mendengarkan permintaan pada port HTTP dan HTTPS serta merutekan permintaan ke EC2 instans Amazon di aplikasi Anda. AWS Elastic Beanstalk Secara default, penyeimbang beban menangani permintaan di port HTTP. Setidaknya salah satu port (HTTP atau HTTPS) harus diaktifkan.



> **Important**
>
> Pastikan bahwa port yang Anda tentukan tidak terkunci; jika tidak, pengguna tidak akan dapat terhubung ke aplikasi Elastic Beanstalk Anda.

Mengontrol port HTTP

Untuk mematikan port HTTP, pilih OFF untuk Port Listener HTTP. Untuk mengaktifkan port HTTP, Anda pilih port HTTP (misalnya, 80) dari daftar.

> **Note**
>
> Untuk mengakses lingkungan Anda menggunakan port selain port default 80, seperti port 8080, tambahkan listener ke penyeimbang beban yang ada dan konfigurasikan listener baru untuk mendengarkan di port tersebut.
> Misalnya, menggunakan penyeimbang beban AWS CLI untuk Klasik, ketik perintah berikut, ganti *LOAD_BALANCER_NAME* dengan nama penyeimbang beban Anda untuk Elastic Beanstalk.
>
> ```
> aws elb create-load-balancer-listeners --load-balancer-name LOAD_BALANCER_NAME
>  --listeners "Protocol=HTTP, LoadBalancerPort=8080, InstanceProtocol=HTTP,
>  InstancePort=80"
> ```
>
> Misalnya, menggunakan AWS CLI untuk Application Load Balancers, ketik perintah berikut, ganti *LOAD_BALANCER_ARN* dengan ARN penyeimbang beban Anda untuk Elastic Beanstalk.
>
> ```
> aws elbv2 create-listener --load-balancer-arn LOAD_BALANCER_ARN --protocol HTTP
>  --port 8080
> ```
>
> Jika Anda ingin Elastic Beanstalk memantau lingkungan Anda, jangan hapus listener di port 80.

Mengontrol port HTTPS

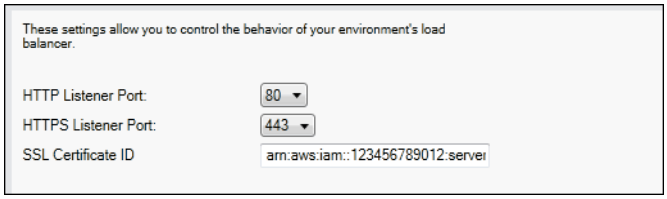
Elastic Load Balancing mendukung protokol HTTPS/TLS untuk mengaktifkan enkripsi lalu lintas untuk koneksi klien ke penyeimbang beban. Koneksi dari penyeimbang beban ke EC2 instance menggunakan enkripsi plaintext. Secara default, port HTTPS dimatikan.

Untuk mengaktifkan port HTTPS

1. Buat sertifikat baru menggunakan AWS Certificate Manager (ACM) atau unggah sertifikat dan kunci ke AWS Identity and Access Management (IAM). Untuk informasi selengkapnya tentang permintaan sertifikat ACM, lihat Meminta Sertifikat di AWS Certificate Manager Panduan

Pengguna. Untuk informasi tentang mengimpor sertifikat pihak ke tiga ke ACM, lihat Mengimpor Sertifikat di Panduan Pengguna AWS Certificate Manager . Jika ACM tidak tersedia di wilayah Anda, gunakan AWS Identity and Access Management (IAM) untuk mengunggah sertifikat pihak ketiga. Layanan ACM dan IAM menyimpan sertifikat dan menyediakan Amazon Resource Name (ARN) untuk sertifikat SSL. Untuk informasi selengkapnya tentang pembuatan dan pengunggahan sertifikat ke IAM, lihat Bekerja dengan Sertifikat Server di Panduan Pengguna IAM.

2. Tentukan port HTTPS dengan memilih port untuk Port Listener HTTPS.



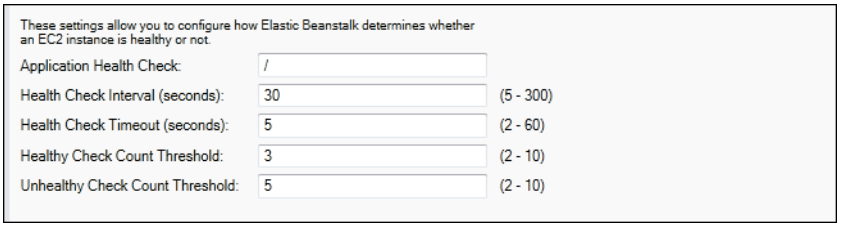
3. Untuk ID Sertifikat SSL, masukkan Amazon Resources Name (ARN) dari sertifikat SSL Anda. Misalnya, **`arn:aws:iam::123456789012:server-certificate/abc/certs/build`** atau **`arn:aws:acm:us-east-2:123456789012:certificate/12345678-12ab-34cd-56ef-12345678`**. Gunakan sertifikat SSL yang Anda buat atau unggah di langkah 1.

Untuk mematikan port HTTPS, pilih OFF untuk HTTPS Listener Port.

Pemeriksaan kondisi

Definisi pemeriksaan kondisi mencakup URL untuk di-kueri untuk instans kondisi. Secara default, Elastic Beanstalk menggunakan TCP:80 untuk kontainer bukan warisan dan HTTP:80 untuk kontainer warisan. Anda dapat mengganti URL default untuk mencocokkan sumber daya yang ada di aplikasi Anda (misalnya, `/myapp/default.aspx`) dengan memasukkannya ke kotak URL Pemeriksaan Kondisi Aplikasi. Jika Anda mengganti URL default, maka Elastic Beanstalk menggunakan HTTP untuk meng-kueri sumber daya. Untuk memeriksa apakah Anda menggunakan tipe kontainer warisan, lihat the section called “Mengapa beberapa versi platform ditandai sebagai legasi?”

Anda dapat mengontrol pengaturan untuk pemeriksaan kesehatan menggunakan bagian Pemeriksaan Kesehatan EC2 Instans pada panel Load Balancing.



Definisi pemeriksaan kondisi mencakup URL untuk di-kueri untuk instans kondisi. Ganti URL default untuk mencocokkan sumber daya yang ada di aplikasi Anda (misalnya, `/myapp/index.jsp`) dengan memasukkannya ke kotak URL Pemeriksaan Kondisi Aplikasi.

Daftar berikut menjelaskan parameter pemeriksaan kondisi yang dapat Anda atur untuk aplikasi Anda.

- Untuk Interval Pemeriksaan Kesehatan (detik), masukkan jumlah detik Elastic Load Balancing menunggu di antara pemeriksaan kesehatan untuk instans Amazon aplikasi Anda. EC2
- Untuk Timeout Pemeriksaan Kondisi (detik), tentukan jumlah detik tunggu Elastic Load Balancing untuk sebuah respons sebelum menganggap instans tidak responsif.
- Untuk Ambang Batas Jumlah Pemeriksaan Sehat dan Ambang Batas Jumlah Pemeriksaan Tidak Sehat, tentukan jumlah probe URL yang berhasil atau tidak berhasil berturut-turut sebelum Elastic Load Balancing mengubah status kondisi instans. Sebagai contoh, menentukan **5** untuk Ambang Batas Jumlah Pemeriksaan Tidak Sehat berarti URL harus mengembalikan pesan kesalahan atau timeout lima kali berturut-turut sebelum Elastic Load Balancing menganggap pemeriksaan kondisi gagal.

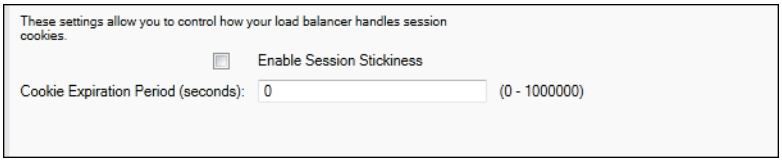
Sesi

Secara default, penyeimbang beban merutekan setiap permintaan secara independen ke instans server dengan beban terkecil. Sebagai perbandingan, sesi lekat mengikat sesi pengguna ke instans server tertentu, sehingga semua permintaan yang datang dari pengguna selama sesi dikirim ke server instans yang sama.

Elastic Beanstalk menggunakan cookie HTTP yang dihasilkan penyeimbang beban saat sesi lekat diaktifkan untuk aplikasi. Penyeimbang beban menggunakan cookie yang dihasilkan penyeimbang beban khusus untuk melacak instans aplikasi untuk setiap permintaan. Ketika penyeimbang beban menerima permintaan, pertama-tama penyeimbang beban memeriksa apakah cookie ini ada di permintaan. Jika ada, permintaan tersebut dikirim ke instans aplikasi yang ditentukan di cookie. Jika tidak ada cookie, penyeimbang beban memilih instans aplikasi berdasarkan algoritme penyeimbangan beban yang ada. Cookie dimasukkan ke dalam respons untuk mengikat permintaan

berikutnya dari pengguna yang sama ke instans aplikasi. Konfigurasi kebijakan mendefinisikan kedaluwarsa cookie, yang menetapkan durasi validitas untuk setiap cookie.

Anda dapat menggunakan bagian Sesi di tab Penyeimbang Beban untuk menentukan apakah penyeimbang beban untuk aplikasi Anda memungkinkan kelengketan sesi.



Untuk informasi selengkapnya di Elastic Load Balancing, lanjutkan ke Panduan Developer Elastic Load Balancing.

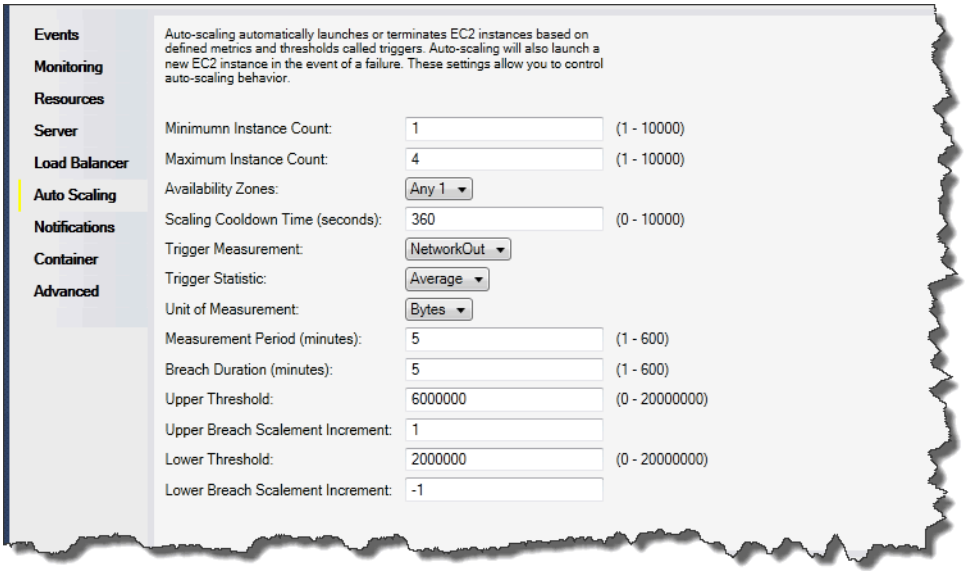
Mengkonfigurasi Auto Scaling menggunakan toolkit AWS untuk Visual Studio

Amazon EC2 Auto Scaling adalah layanan web Amazon yang dirancang untuk secara otomatis meluncurkan atau menghentikan instans EC2 Amazon berdasarkan pemicu yang ditentukan pengguna. Pengguna dapat menyiapkan Grup Auto Scaling dan kaitkan pemicu dengan grup-grup ini untuk secara otomatis mengukur sumber daya komputasi berdasarkan metrik, seperti penggunaan bandwidth atau utilisasi CPU. EC2 Auto Scaling Amazon berfungsi dengan Amazon CloudWatch untuk mengambil metrik instance server yang menjalankan aplikasi Anda.

EC2 Auto Scaling Amazon memungkinkan Anda mengambil sekelompok EC2 instans Amazon dan mengatur berbagai parameter agar grup ini bertambah atau berkurang jumlahnya secara otomatis. EC2 Auto Scaling Amazon dapat menambah atau menghapus EC2 instans Amazon dari grup tersebut untuk membantu Anda menangani perubahan lalu lintas ke aplikasi dengan mulus.

Amazon EC2 Auto Scaling juga memantau kesehatan setiap EC2 instans Amazon yang diluncurkan. Jika ada instance yang dihentikan secara tidak terduga, Amazon EC2 Auto Scaling mendeteksi penghentian dan meluncurkan instance pengganti. Kemampuan ini memungkinkan Anda mempertahankan jumlah EC2 instans Amazon yang tetap dan diinginkan secara otomatis.

Elastic Beanstalk menyediakan EC2 Amazon Auto Scaling untuk aplikasi Anda. Anda dapat mengedit konfigurasi instans Amazon lingkungan EC2 Elastic Beanstalk dengan tab Auto Scaling di dalam tab lingkungan aplikasi Anda di Toolkit AWS for Visual Studio.

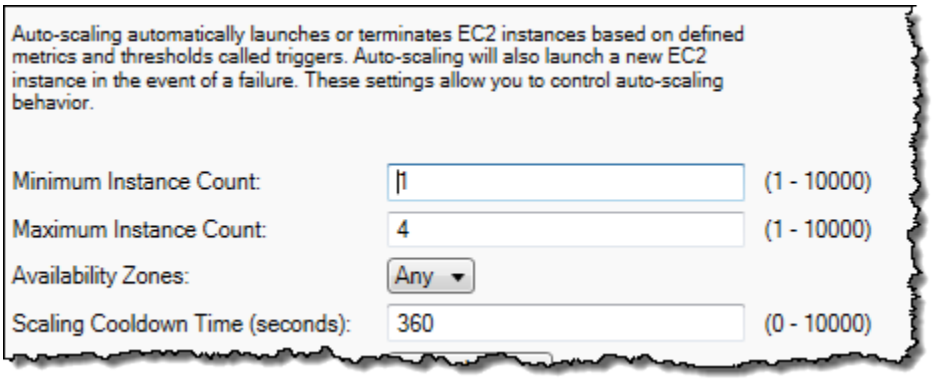


Bagian berikut membahas cara mengonfigurasi parameter Auto Scaling untuk aplikasi Anda.

Meluncurkan konfigurasi

Anda dapat mengedit konfigurasi peluncuran untuk mengontrol bagaimana aplikasi Elastic Beanstalk Anda menyediakan sumber daya EC2 Amazon Auto Scaling.

Kotak Jumlah Instans Minimum dan Jumlah Instans Maksimum memungkinkan Anda menentukan ukuran minimum dan maksimum grup Auto Scaling yang digunakan aplikasi Elastic Beanstalk Anda.



> **Note**
>
> Untuk mempertahankan jumlah EC2 instans Amazon yang tetap, setel Jumlah Instans Minimum dan Hitungan Instance Maksimum ke nilai yang sama.

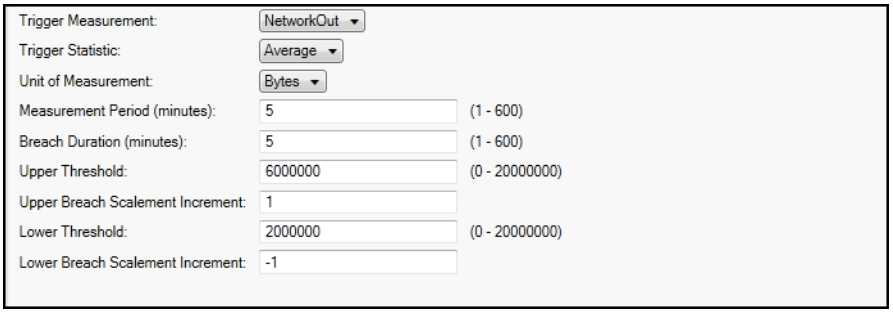
Kotak Availability Zones memungkinkan Anda menentukan jumlah Availability Zone yang Anda inginkan untuk EC2 instans Amazon. Hal ini penting untuk mengatur jumlah ini jika Anda ingin membangun aplikasi yang toleran terhadap kesalahan. Jika satu Availability Zone mengalami masalah, instans Anda masih akan berjalan di Availability Zone lainnya.

> **Note**
>
> Saat ini, tidak mungkin untuk menentukan Availability Zone mana instans Anda akan berada.

Pemicu

Pemicu adalah mekanisme EC2 Auto Scaling Amazon yang Anda atur untuk memberi tahu sistem saat Anda ingin menambah (memperkecil) jumlah instance, dan saat Anda ingin mengurangi (menskalakan) jumlah instans. Anda dapat mengonfigurasi pemicu untuk diaktifkan pada metrik apa pun yang dipublikasikan ke Amazon CloudWatch, seperti pemanfaatan CPU, dan menentukan apakah kondisi yang Anda tentukan telah terpenuhi. Bila ambang batas atas atau bawah dari kondisi yang telah Anda tentukan untuk metrik telah dilanggar selama jangka waktu tertentu, pemicu akan meluncurkan proses yang berjalan lama yang disebut Aktivitas Penskalaan.

Anda dapat menentukan pemicu penskalaan untuk aplikasi Elastic Beanstalk Anda menggunakan Toolkit AWS for Visual Studio.



Pemicu EC2 Auto Scaling Amazon berfungsi dengan menonton CloudWatch metrik Amazon tertentu untuk sebuah instans. Pemicu mencakup penggunaan CPU, lalu lintas jaringan, dan aktivitas disk. Gunakan pengaturan Pengukuran Pemicu untuk memilih metrik pemicu Anda.

Daftar berikut menjelaskan parameter pemicu yang dapat Anda konfigurasikan menggunakan AWS Management Console.

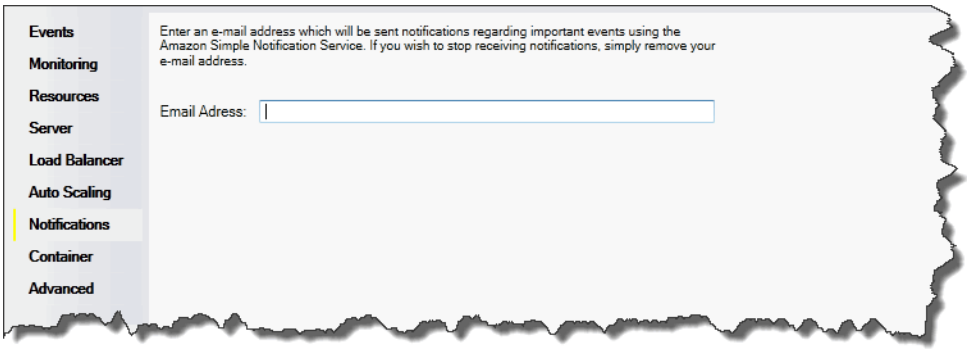
- Anda dapat menentukan statistik pemicu mana yang harus digunakan. Anda dapat memilih Minimum, Maksimum, Jumlah, atau Rata-rata untuk Statistik Pemicu.

- Untuk Unit Pengukuran, tentukan unit untuk pengukuran pemicu.
- Nilai dalam kotak Periode Pengukuran menentukan seberapa sering Amazon CloudWatch mengukur metrik untuk pemicu Anda. Durasi Pelanggaran adalah jumlah waktu metrik dapat melampaui batas yang ditetapkan (sebagaimana ditentukan untuk Ambang Batas Atas dan Ambang Batas Bawah) sebelum pemicu diaktifkan.
- Untuk Peningkatan Skala Pelanggaran Atas dan Peningkatan Skala Pelanggaran Bawah, tentukan jumlah EC2 instans Amazon yang akan ditambahkan atau dihapus saat melakukan aktivitas penskalaan.

Untuk informasi selengkapnya tentang EC2 Auto Scaling Amazon, lihat bagian Auto EC2 Scaling Amazon di Amazon Elastic Compute Cloud Documentation.

Mengkonfigurasi notifikasi menggunakan AWS toolkit untuk Visual Studio

Elastic Beanstalk menggunakan Amazon Simple Notification Service (Amazon SNS) untuk memberitahu Anda tentang peristiwa penting yang mempengaruhi aplikasi Anda. Untuk mengaktifkan notifikasi Amazon SNS, masukkan alamat email Anda di kotak Alamat Email. Untuk menonaktifkan notifikasi ini, hapus alamat email Anda dari kotak.



Mengkonfigurasi kontainer.NET menggunakan AWS toolkit untuk Visual Studio

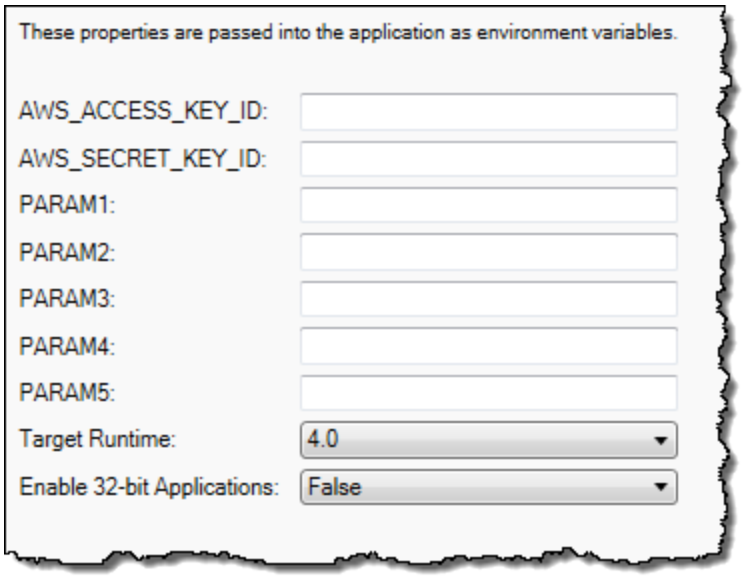
Panel Container/.net Options memungkinkan Anda menyempurnakan perilaku EC2 instans Amazon Anda dan mengaktifkan atau menonaktifkan rotasi log Amazon S3. Anda dapat menggunakan AWS Toolkit for Visual Studio untuk mengonfigurasi informasi kontainer Anda.

> **Note**
>
> Anda dapat memodifikasi pengaturan konfigurasi Anda dengan nol downtime dengan menukar CNAME untuk lingkungan Anda. Untuk informasi selengkapnya, lihat Deployment Biru/Hijau dengan Elastic Beanstalk.

Jika mau, Anda bisa menambah jumlah parameter. Untuk informasi memperluas parameter, lihat Pengaturan opsi.

Untuk mengakses panel opsi Container/.NET untuk aplikasi Elastic Beanstalk Anda

1. Di AWS Toolkit for Visual Studio, perluas node Elastic Beanstalk dan node aplikasi Anda.
2. Di AWS Explorer, klik dua kali lingkungan Elastic Beanstalk Anda.
3. Di bagian bawah panel Gambaran Umum, klik tab Konfigurasi.
4. Di bawah Kontainer, Anda dapat mengonfigurasi opsi kontainer.

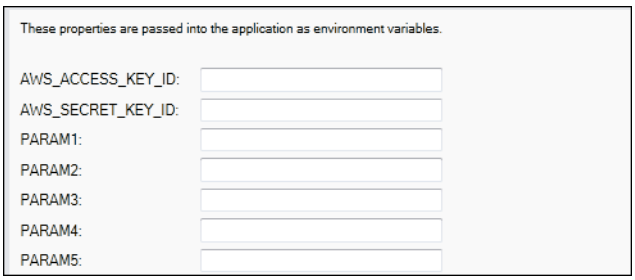


Opsi kontainer .NET

Anda dapat memilih versi .NET Framework untuk aplikasi Anda. Pilih 2.0 atau 4.0 untuk Waktu aktif target. Pilih Aktifkan Aplikasi 32-bit jika Anda ingin mengaktifkan aplikasi 32-bit.

Pengaturan aplikasi

Bagian Pengaturan Aplikasi memungkinkan Anda menentukan variabel lingkungan yang dapat Anda baca dari kode aplikasi Anda.



Mengelola akun

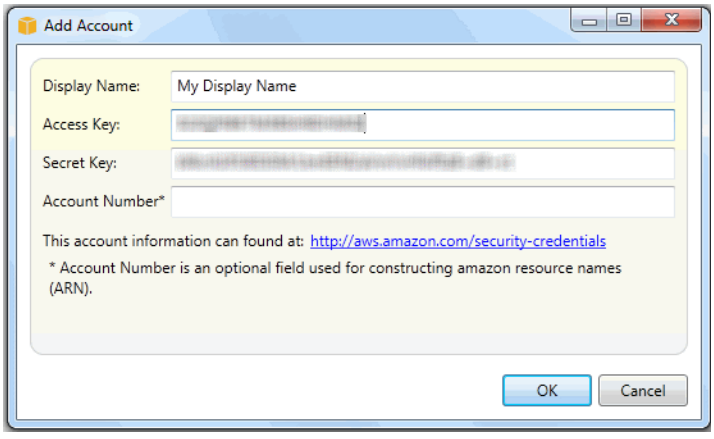
Jika Anda ingin menyiapkan AWS akun yang berbeda untuk melakukan tugas yang berbeda, seperti pengujian, pementasan, dan produksi, Anda dapat menambahkan, mengedit, dan menghapus akun menggunakan AWS Toolkit for Visual Studio.

Untuk mengelola beberapa akun

1. Di Visual Studio, di menu Lihat, klik AWS Explorer.
2. Di samping daftar Akun, klik tombol Tambahkan Akun.



Kotak dialog Tambahkan Akun muncul.



3. Isi informasi yang diminta.

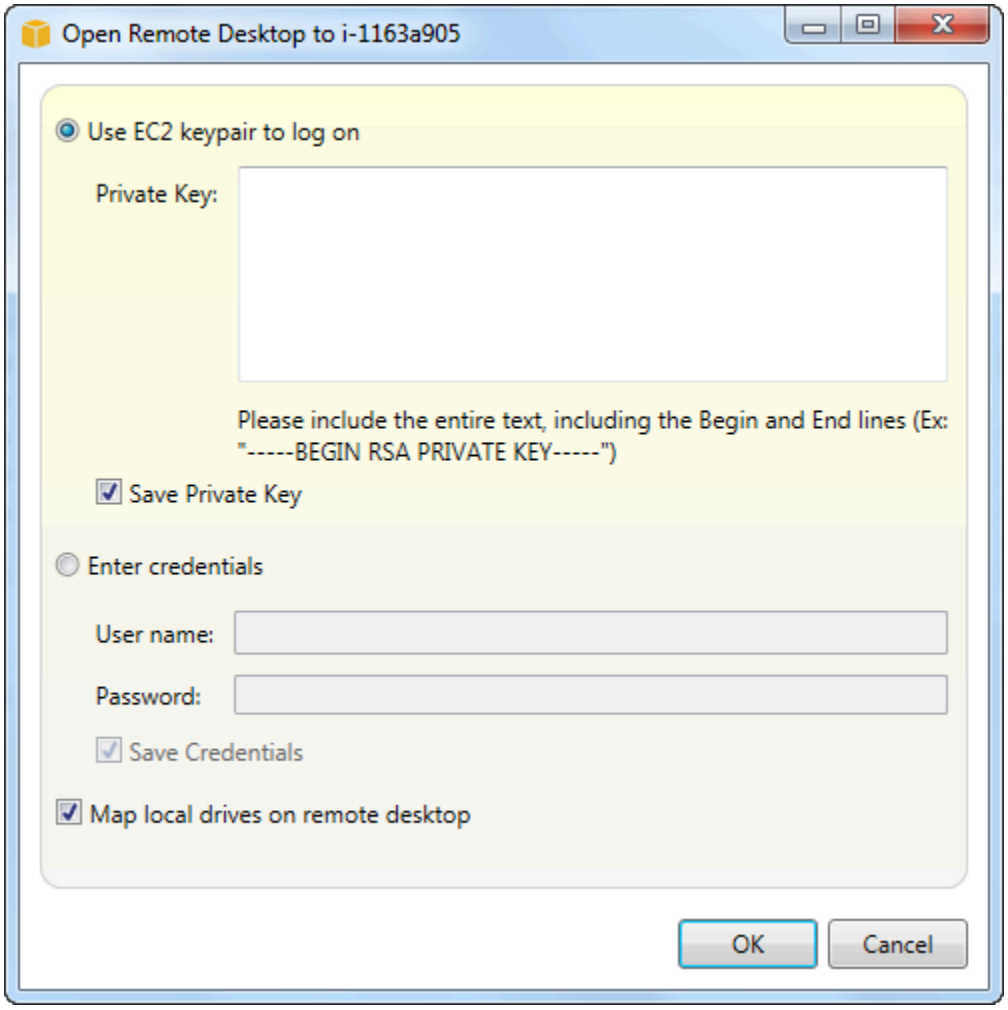
4. Informasi akun Anda sekarang muncul di tab AWS Explorer. Saat Anda mempublikasikan ke Elastic Beanstalk, Anda dapat memilih akun mana yang ingin Anda gunakan.

Membuat daftar dan menghubungkan ke server instans

Anda dapat melihat daftar EC2 instans Amazon yang menjalankan lingkungan aplikasi Elastic Beanstalk melalui Toolkit AWS for Visual Studio atau dari Management Console. AWS Anda dapat terhubung ke instans-instans ini menggunakan Remote Desktop Connection. Untuk informasi tentang mencantumkan dan menghubungkan ke instance server menggunakan AWS Management Console, lihatMembuat daftar dan menghubungkan ke server instans. Bagian berikut akan membantu Anda melihat dan menghubungkan Anda ke instance server menggunakan AWS Toolkit for Visual Studio.

Untuk melihat dan terhubung ke EC2 instans Amazon untuk suatu lingkungan

1. Di Visual Studio, di AWS Explorer, perluas EC2 node Amazon dan klik dua kali Instans.
2. Klik kanan ID EC2 instance Amazon yang berjalan di load balancer aplikasi Anda di kolom Instance dan pilih Open Remote Desktop dari menu konteks.



3. Pilih Gunakan EC2 keypair untuk masuk dan menempelkan konten file kunci pribadi yang Anda gunakan untuk menyebarkan aplikasi Anda di kotak Kunci pribadi. Atau, masukkan nama pengguna dan kata sandi Anda di kotak teks Nama pengguna dan Kata sandi.

> **Note**
>
> Jika pasangan kunci disimpan di dalam Toolkit, kotak teks tidak muncul.

4. Klik OK.

Pemantauan kondisi aplikasi

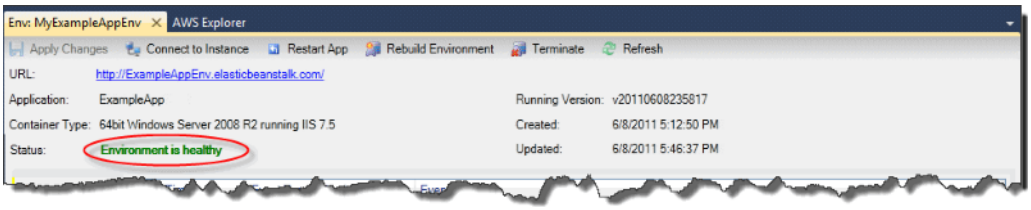
Ketika Anda menjalankan situs web produksi, penting untuk mengetahui bahwa aplikasi Anda tersedia dan menanggapi permintaan. Untuk membantu pemantauan responsivitas aplikasi Anda,

Elastic Beanstalk memberikan fitur agar Anda dapat memantau statistik tentang aplikasi Anda dan membuat pemberitahuan yang memicu ketika ambang batas terlampaui.

Untuk informasi tentang pemantauan kondisi yang disediakan oleh Elastic Beanstalk, lihat Pelaporan kondisi dasar.

Anda dapat mengakses informasi operasional tentang aplikasi Anda dengan menggunakan AWS Toolkit for Visual Studio atau Management AWS Console.

Kit alat menampilkan status lingkungan dan kondisi aplikasi Anda di bidang Status.



Untuk memantau kondisi aplikasi

1. Di AWS Toolkit for Visual Studio, AWS di Explorer, perluas node Elastic Beanstalk, dan kemudian perluas node aplikasi Anda.
2. Klik kanan lingkungan Elastic Beanstalk Anda, dan kemudian klik Lihat Status.
3. Di tab lingkungan aplikasi Anda, klik Pemantauan.

   Panel Pemantauan mencakup serangkaian grafik yang menunjukkan penggunaan sumber daya untuk lingkungan aplikasi tertentu Anda.

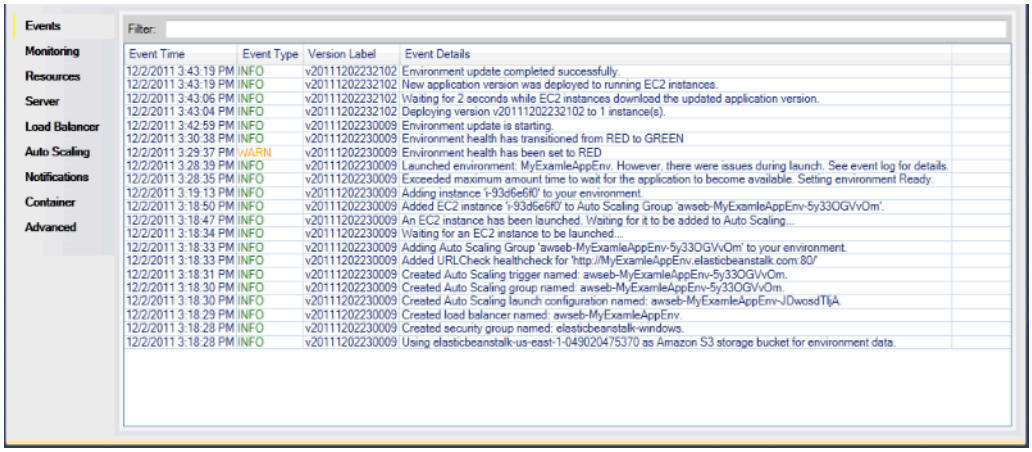


> **Note**
>
> Secara default, rentang waktu diatur ke jam terakhir. Untuk memodifikasi pengaturan ini, di daftar Rentang Waktu, klik rentang waktu yang berbeda.

Anda dapat menggunakan AWS Toolkit for Visual Studio atau AWS Management Console untuk melihat peristiwa yang terkait dengan aplikasi Anda.

Untuk melihat peristiwa aplikasi

1. Di AWS Toolkit for Visual Studio, AWS di Explorer, perluas node Elastic Beanstalk dan node aplikasi Anda.
2. Klik kanan lingkungan Elastic Beanstalk Anda di AWS Explorer dan kemudian klik Lihat Status.
3. Di tab lingkungan aplikasi Anda, klik Peristiwa.



Men-deploy aplikasi Elastic Beanstalk di .NET menggunakan alat deployment

AWS Toolkit for Visual Studio mencakup alat penyebaran, alat baris perintah yang menyediakan fungsionalitas yang sama dengan wizard penyebaran di Toolkit. AWS Anda dapat menggunakan alat deployment di alur pembangunan atau penulisan lain untuk mengotomatisasi deployment ke Elastic Beanstalk.

Alat deployment mendukung deployment awal dan deployment ulang. Jika Anda sebelumnya men-deploy aplikasi Anda menggunakan alat deployment, Anda dapat men-deploy ulang menggunakan

wizard deployment dalam Visual Studio. Demikian juga, jika Anda telah men-deploy menggunakan wizard, Anda dapat men-deploy ulang menggunakan alat deployment.

> **Note**
>
> Alat deployment tidak menerapkan nilai yang disarankan untuk opsi konfigurasi seperti konsol atau EB CLI. Gunakan file konfigurasi untuk memastikan bahwa setiap pengaturan yang Anda butuhkan dikonfigurasi ketika Anda meluncurkan lingkungan Anda.

Bab ini memandu Anda melalui deploy sampel aplikasi .NET untuk Elastic Beanstalk menggunakan alat deployment, dan kemudian men-deploy ulang aplikasi menggunakan deployment tambahan. Untuk diskusi yang lebih dalam tentang alat deployment, termasuk opsi parameter, lihat Alat Deployment.

Prasyarat

Untuk menggunakan alat penyebaran, Anda perlu menginstal AWS Toolkit for Visual Studio. Untuk informasi tentang prasyarat dan petunjuk instalasi, lihat Toolkit for Visual Studio AWS.

Alat deployment biasanya diinstal di salah satu direktori berikut di Windows:

32-bit	64-bit
`C:\Program Files\AWS Tools\Deployment Tool\awsdeploy.exe`	`C:\Program Files (x86)\AWS Tools\Deployment Tool\awsdeploy.exe`

Men-deploy ke Elastic Beanstalk

Untuk men-deploy aplikasi sampel untuk Elastic Beanstalk menggunakan alat deployment, Anda harus terlebih dahulu memodifikasi file konfigurasi `ElasticBeanstalkDeploymentSample.txt`, yang disediakan di direktori `Samples`. File konfigurasi ini berisi informasi yang diperlukan untuk menyebarkan aplikasi Anda, termasuk nama aplikasi, versi aplikasi, nama lingkungan, dan kredenal AWS akses Anda. Setelah memodifikasi file konfigurasi, Anda kemudian menggunakan baris perintah untuk men-deploy aplikasi sampel. File web deploy Anda diunggah ke Amazon S3, dan terdaftar sebagai versi aplikasi baru dengan Elastic Beanstalk. Dibutuhkan beberapa menit untuk men-deploy

aplikasi Anda. Setelah lingkungan sehat, alat deployment mengeluarkan URL untuk aplikasi yang berjalan.

Untuk men-deploy aplikasi .NET ke Elastic Beanstalk

1. Dari `Samples` subdirektori tempat alat penyebaran diinstal, buka `ElasticBeanstalkDeploymentSample.txt` dan masukkan kunci AWS akses dan kunci AWS rahasia Anda seperti pada contoh berikut.

   ```
   ### AWS Access Key and Secret Key used to create and deploy the application
    instance
   AWSAccessKey = AKIAIOSFODNN7EXAMPLE
   AWSSecretKey = wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
   ```

   > **Note**
   >
   > Untuk akses API, Anda memerlukan access key ID dan secret access key. Gunakan kunci akses pengguna IAM alih-alih kunci Pengguna root akun AWS akses. Untuk informasi selengkapnya tentang membuat kunci akses, lihat Mengelola kunci akses untuk pengguna IAM di Panduan Pengguna IAM.

2. Di prompt baris perintah, ketik hal berikut:

   ```
   C:\Program Files (x86)\AWS Tools\Deployment Tool>awsdeploy.exe /w Samples
   \ElasticBeanstalkDeploymentSample.txt
   ```

   Dibutuhkan beberapa menit untuk men-deploy aplikasi Anda. Jika deployment berhasil, Anda akan melihat pesan, `Application deployment completed; environment health is Green`.

   > **Note**
   >
   > Jika Anda menerima kesalahan berikut, CNAME sudah ada.
   >
   > ```
   > [Error]: Deployment to AWS Elastic Beanstalk failed with exception: DNS name
   >  (MyAppEnv.elasticbeanstalk.com) is not available.
   > ```

Karena CNAME harus unik, Anda perlu mengubah `Environment.CNAME` di `ElasticBeanstalkDeploymentSample.txt`.

3. Di peramban web Anda, arahkan ke URL aplikasi Anda yang berjalan. URL akan dalam bentuk <CNAME.elasticbeanstalk.com> (misalnya, **`MyAppEnv.elasticbeanstalk.com`**).

Memigrasi aplikasi .NET On-Premise Anda ke Elastic Beanstalk

AWS Elastic Beanstalk menyediakan jalur migrasi yang efisien untuk aplikasi Windows Anda yang berjalan di Internet Information Services (IIS) melalui Elastic Beanstalk Command Line Interface (EB CLI). eb migratePerintah secara otomatis menemukan situs IIS, aplikasi, dan direktori virtual Anda, mempertahankan konfigurasinya, dan menyebarkannya ke Cloud. AWS

Kemampuan migrasi bawaan ini menawarkan pendekatan yang lebih sederhana yang mengurangi kompleksitas dan waktu yang biasanya terkait dengan migrasi cloud. Proses migrasi membantu menjaga fungsionalitas aplikasi dan integritas konfigurasi selama transisi ke AWS.

Untuk petunjuk lengkap dan terperinci tentang memigrasi aplikasi IIS Anda AWS Elastic Beanstalk, lihat bagian dalam Migrasi aplikasi IIS panduan ini.

Rekomendasi untuk komponen pensiun Windows Server pada Elastic Beanstalk

Topik ini memberikan rekomendasi jika aplikasi Anda saat ini berjalan di cabang platform Windows Server 2012 R2 yang sudah pensiun. Ini juga membahas dukungan usang untuk versi protokol TLS 1.0 dan 1.1 pada titik akhir API AWS layanan kami dan cabang platform yang terpengaruh.

Cabang platform Windows Server 2012 R2 pensiun

Elastic Beanstalk mempensiunkan cabang platform Windows Server 2012 R2 pada 4 Desember 2023, dan AMIs menjadikan platform yang terkait dengan platform tersebut menjadi pribadi pada 10 April 2024. Tindakan ini mencegah peluncuran instance di lingkungan Windows Server 2012 Anda yang menggunakan AMI Beanstalk default.

Jika Anda memiliki lingkungan yang berjalan di cabang platform Windows yang sudah pensiun, kami sarankan Anda memigrasikannya ke salah satu platform Windows Server berikut, yang saat ini dan didukung penuh:

- Windows Server 2022 dengan IIS 10.0 versi 2.x
- Windows Server 2019 dengan IIS 10.0 versi 2.x

Untuk pertimbangan migrasi lengkap, lihatMigrasi dari versi utama sebelumnya dari platform server Windows.

Untuk informasi selengkapnya tentang penghentian platform, lihat. Kebijakan dukungan platform Elastic Beanstalk

> **Note**
>
> Jika Anda tidak dapat bermigrasi ke platform yang didukung penuh ini, sebaiknya gunakan kustom yang AMIs dibuat dengan Windows Server 2012 R2 atau Windows Server 2012 R2 Core AMIs sebagai gambar dasar, jika Anda belum melakukannya. Untuk petunjuk mendetail, lihat Mempertahankan akses ke Amazon Machine Image (AMI) untuk platform yang sudah pensiun. Hubungi Pusat AWS Dukungan jika Anda memerlukan akses sementara ke AMI saat Anda melakukan salah satu langkah migrasi ini.

TLS 1.2 Kompatibilitas

Per 31 Desember 2023, AWS mulai sepenuhnya menerapkan TLS 1.2 di semua AWS titik akhir API. Tindakan ini menghapus kemampuan untuk menggunakan TLS versi 1.0 dan 1.1 dengan semua AWS APIs. Informasi ini awalnya dikomunikasikan pada 28 Juni 2022. Untuk menghindari risiko dampak ketersediaan, tingkatkan lingkungan apa pun yang menjalankan versi platform yang diidentifikasi di sini ke versi yang lebih baru sesegera mungkin, jika Anda belum melakukannya.

Potensi dampak

Versi platform Elastic Beanstalk yang menjalankan TLS v1.1 atau sebelumnya terpengaruh. Perubahan ini memengaruhi tindakan lingkungan yang mencakup tetapi tidak terbatas pada hal-hal berikut: penerapan konfigurasi, penerapan aplikasi, penskalaan otomatis, peluncuran lingkungan baru, rotasi log, laporan kesehatan yang disempurnakan, dan menerbitkan log aplikasi ke bucket Amazon S3 yang terkait dengan aplikasi Anda.

Versi Platform Windows yang Terkena Dampak

Pelanggan dengan lingkungan Elastic Beanstalk pada versi platform berikut disarankan untuk memutakhirkan setiap lingkungan yang sesuai ke platform Windows versi 2.8.3 atau yang lebih baru, dirilis pada 18 Feb 2022.

- Windows Server 2019 - platform versi 2.8.2 atau versi sebelumnya

Pelanggan dengan lingkungan Elastic Beanstalk pada versi platform berikut disarankan untuk memutakhirkan setiap lingkungan yang sesuai ke platform Windows versi 2.10.7 atau yang lebih baru, dirilis pada 28 Des 2022.

- Windows Server 2016 - platform versi 2.10.6 atau versi sebelumnya
- Windows Server 2012 — semua versi platform; platform ini dihentikan pada 4 Desember 2023
- Windows Server 2008 — semua versi platform; platform ini dihentikan pada 28 Oktober 2019

Untuk daftar versi platform Windows Server terbaru dan didukung, lihat Platform yang Didukung dalam panduan AWS Elastic Beanstalk Platform.

Untuk detail dan praktik terbaik tentang memperbarui lingkungan Anda, lihatMemperbarui versi platform lingkungan Elastic Beanstalk Anda.

Menyebarkan aplikasi.NET core (Linux) dengan Elastic Beanstalk

> **Lihat .NET di Pusat AWS Pengembang**
>
> Sudahkah Anda mampir ke Pusat Pengembang.Net kami? Ini adalah one stop shop kami untuk semua hal. NET di AWS.
> Untuk informasi lebih lanjut, lihat .NET di Pusat AWS Pengembang.

Bab ini memberikan instruksi untuk mengkonfigurasi dan menyebarkan aplikasi web .NET core (Linux) Anda ke. AWS Elastic Beanstalk Elastic Beanstalk memudahkan penerapan, pengelolaan, dan skala aplikasi web.NET core (Linux) Anda menggunakan Amazon Web Services.

Anda dapat menerapkan aplikasi Anda hanya dalam beberapa menit menggunakan Elastic Beanstalk Command Line Interface (EB CLI) atau dengan menggunakan konsol Elastic Beanstalk. Setelah Anda menerapkan aplikasi Elastic Beanstalk, Anda dapat terus menggunakan EB CLI untuk mengelola aplikasi dan lingkungan Anda, atau Anda dapat menggunakan konsol Elastic Beanstalk, atau. AWS CLI APIs

Ikuti langkah-langkah dalam QuickStart untuk .NET Core di Linux untuk step-by-step petunjuk untuk membuat dan menyebarkan aplikasi web ASP.NET Core Hello World dengan EB CLI.

Topik

- QuickStart: Menyebarkan aplikasi .NET Core pada Linux ke Elastic Beanstalk
- Menyiapkan inti .NET Anda di lingkungan pengembangan Linux untuk Elastic Beanstalk
- Menggunakan inti Elastic Beanstalk .NET pada platform Linux
- The AWS Toolkit for Visual Studio - Bekerja dengan.Net Core pada Elastic Beanstalk
- Migrasi dari.NET pada platform Windows Server ke .NET Core pada platform Linux di Elastic Beanstalk

QuickStart: Menyebarkan aplikasi .NET Core pada Linux ke Elastic Beanstalk

QuickStart Tutorial ini memandu Anda melalui proses pembuatan .NET Core pada aplikasi Linux dan menyebarkannya ke AWS Elastic Beanstalk lingkungan.

> **Note**
>
> Contoh tutorial dimaksudkan untuk demonstrasi. Jangan gunakan aplikasi untuk lalu lintas produksi.

Bagian-bagian

- AWS Akun Anda
- Prasyarat
- Langkah 1: Buat .NET Core pada aplikasi Linux
- Langkah 2: Jalankan aplikasi Anda secara lokal
- Langkah 3: Menyebarkan .NET Core Anda pada aplikasi Linux dengan EB CLI
- Langkah 4: Jalankan aplikasi Anda di Elastic Beanstalk
- Langkah 5: Bersihkan
- AWS sumber daya untuk aplikasi Anda
- Langkah selanjutnya
- Terapkan dengan konsol Elastic Beanstalk

AWS Akun Anda

Jika Anda belum menjadi AWS pelanggan, Anda perlu membuat AWS akun. Mendaftar memungkinkan Anda mengakses Elastic Beanstalk AWS dan layanan lain yang Anda butuhkan.

Jika Anda sudah memiliki AWS akun, Anda dapat melanjutkan kePrasyarat.

Buat AWS akun

Mendaftar untuk Akun AWS

Jika Anda tidak memiliki Akun AWS, selesaikan langkah-langkah berikut untuk membuatnya.

Untuk mendaftar untuk Akun AWS

1. Buka https://portal.aws.amazon.com/billing/pendaftaran.
2. Ikuti petunjuk online.

   Bagian dari prosedur pendaftaran melibatkan menerima panggilan telepon atau pesan teks dan memasukkan kode verifikasi pada keypad telepon.

   Saat Anda mendaftar untuk sebuah Akun AWS, sebuah Pengguna root akun AWSdibuat. Pengguna root memiliki akses ke semua Layanan AWS dan sumber daya di akun. Sebagai praktik keamanan terbaik, tetapkan akses administratif ke pengguna, dan gunakan hanya pengguna root untuk melakukan tugas yang memerlukan akses pengguna root.

AWS mengirimi Anda email konfirmasi setelah proses pendaftaran selesai. Kapan saja, Anda dapat melihat aktivitas akun Anda saat ini dan mengelola akun Anda dengan masuk https://aws.amazon.comke/ dan memilih Akun Saya.

Buat pengguna dengan akses administratif

Setelah Anda mendaftar Akun AWS, amankan Pengguna root akun AWS, aktifkan AWS IAM Identity Center, dan buat pengguna administratif sehingga Anda tidak menggunakan pengguna root untuk tugas sehari-hari.

Amankan Anda Pengguna root akun AWS

1. Masuk ke AWS Management Consolesebagai pemilik akun dengan memilih pengguna Root dan memasukkan alamat Akun AWS email Anda. Di laman berikutnya, masukkan kata sandi.

   Untuk bantuan masuk dengan menggunakan pengguna root, lihat Masuk sebagai pengguna root di AWS Sign-In Panduan Pengguna.
2. Mengaktifkan autentikasi multi-faktor (MFA) untuk pengguna root Anda.

   Untuk petunjuk, lihat Mengaktifkan perangkat MFA virtual untuk pengguna Akun AWS root (konsol) Anda di Panduan Pengguna IAM.

Buat pengguna dengan akses administratif

1. Aktifkan Pusat Identitas IAM.

Untuk mendapatkan petunjuk, silakan lihat Mengaktifkan AWS IAM Identity Center di Panduan Pengguna AWS IAM Identity Center .

2. Di Pusat Identitas IAM, berikan akses administratif ke pengguna.

   Untuk tutorial tentang menggunakan Direktori Pusat Identitas IAM sebagai sumber identitas Anda, lihat Mengkonfigurasi akses pengguna dengan default Direktori Pusat Identitas IAM di Panduan AWS IAM Identity Center Pengguna.

Masuk sebagai pengguna dengan akses administratif

- Untuk masuk dengan pengguna Pusat Identitas IAM, gunakan URL masuk yang dikirim ke alamat email saat Anda membuat pengguna Pusat Identitas IAM.

  Untuk bantuan masuk menggunakan pengguna Pusat Identitas IAM, lihat Masuk ke portal AWS akses di Panduan AWS Sign-In Pengguna.

Tetapkan akses ke pengguna tambahan

1. Di Pusat Identitas IAM, buat set izin yang mengikuti praktik terbaik menerapkan izin hak istimewa paling sedikit.

   Untuk petunjuknya, lihat Membuat set izin di Panduan AWS IAM Identity Center Pengguna.

2. Tetapkan pengguna ke grup, lalu tetapkan akses masuk tunggal ke grup.

   Untuk petunjuk, lihat Menambahkan grup di Panduan AWS IAM Identity Center Pengguna.

Prasyarat

Untuk mengikuti prosedur di panduan ini, Anda memerlukan terminal atau shell baris perintah untuk menjalankan perintah. Perintah ditampilkan dalam daftar yang diawali dengan simbol prompt ($) dan nama direktori saat ini, bila perlu.

```
~/eb-project$ this is a command
this is output
```

Di Linux dan macOS, Anda dapat menggunakan shell dan manajer paket pilihan Anda. Pada Windows Anda dapat menginstal Windows Subsystem untuk Linux untuk mendapatkan versi Windows terintegrasi dari Ubuntu dan Bash.

EB CLI

Tutorial ini menggunakan Elastic Beanstalk Command Line Interface (EB CLI). Untuk detail tentang pemasangan dan konfigurasi EB CLI, lihat Instal EB CLI dengan skrip pengaturan (disarankan) dan Mengonfigurasi EB CLI.

.NET Core pada Linux

Jika Anda tidak memiliki .NET SDK diinstal pada mesin lokal Anda, Anda dapat menginstalnya dengan mengikuti tautan Download.NET di situs web dokumentasi.NET.

Verifikasi instalasi .NET SDK Anda dengan menjalankan perintah berikut.

```
~$ dotnet --info
```

Langkah 1: Buat .NET Core pada aplikasi Linux

Buat direktori proyek.

```
~$ mkdir eb-dotnetcore
~$ cd eb-dotnetcore
```

Selanjutnya, buat contoh aplikasi Hello World dengan menjalankan perintah berikut.

```
~/eb-dotnetcore$ dotnet new web --name HelloElasticBeanstalk
~/eb-dotnetcore$ cd HelloElasticBeanstalk
```

Langkah 2: Jalankan aplikasi Anda secara lokal

Jalankan perintah berikut untuk menjalankan aplikasi Anda secara lokal.

```
~/eb-dotnetcore/HelloElasticBeasntalk$ dotnet run
```

Outputnya akan terlihat seperti teks berikut.

```
Building...
info: Microsoft.Hosting.Lifetime[14]
      Now listening on: https://localhost:7294
info: Microsoft.Hosting.Lifetime[14]
      Now listening on: http://localhost:5052
info: Microsoft.Hosting.Lifetime[0]
      Application started. Press Ctrl+C to shut down.
info: Microsoft.Hosting.Lifetime[0]
      Hosting environment: Development
info: Microsoft.Hosting.Lifetime[0]
```

> **Note**
>
> `dotnet`Perintah memilih port secara acak saat menjalankan aplikasi secara lokal. Dalam contoh ini port adalah 5052. Saat Anda menerapkan aplikasi ke lingkungan Elastic Beanstalk Anda, aplikasi akan berjalan pada port 5000.

Masukkan alamat URL `http://localhost:`*`port`* di browser web Anda. Untuk contoh spesifik ini, perintahnya adalah`http://localhost:5052`. Browser web harus menampilkan “Hello World!”.

Langkah 3: Menyebarkan .NET Core Anda pada aplikasi Linux dengan EB CLI

Jalankan perintah berikut untuk membuat lingkungan Elastic Beanstalk untuk aplikasi ini.

Untuk membuat lingkungan dan menyebarkan .NET Core Anda di aplikasi Linux

1. Kompilasi dan publikasikan aplikasi Anda ke folder untuk penyebaran ke lingkungan Elastic Beanstalk yang akan Anda buat.

   ```
   ~$ cd eb-dotnetcore/HelloElasticBeanstalk
   ~/eb-dotnetcore/HelloElasticBeanstalk$ dotnet publish -o site
   ```

2. Arahkan ke direktori situs tempat Anda baru saja menerbitkan aplikasi.

   ```
   ~/eb-dotnetcore/HelloElasticBeanstalk$ cd site
   ```

3. Inisialisasi repositori EB CLI Anda dengan perintah eb init.

Perhatikan detail berikut mengenai versi cabang platform yang Anda tentukan dalam perintah:

- Ganti *x.y.z* dalam perintah berikut dengan versi terbaru dari cabang platform .NET 6 pada AL2 023.
- Untuk menemukan versi cabang platform terbaru, lihat .NET Core pada platform yang Didukung Linux dalam panduan AWS Elastic Beanstalk Platform.
- Contoh nama tumpukan solusi yang menyertakan nomor versi adalah`64bit-amazon-linux-2023-v`**`3.1.1`**`-running-.net-6`. Dalam contoh ini versi cabang adalah 3.1.1.

```
~eb-dotnetcore/HelloElasticBeanstalk/site$ eb init -p 64bit-amazon-linux-2023-
vx.y.z-running-.net-6 dotnetcore-tutorial --region us-east-2
Application dotnetcore-tutorial has been created.
```

Perintah ini membuat aplikasi bernama `dotnetcore-tutorial` dan mengkonfigurasi repositori lokal Anda untuk membuat lingkungan dengan.NET Core pada versi platform Linux yang ditentukan dalam perintah.

4. (Opsional) Jalankan eb init lagi untuk mengonfigurasi key pair default sehingga Anda dapat menggunakan SSH untuk terhubung ke EC2 instance yang menjalankan aplikasi Anda.

```
~eb-dotnetcore/HelloElasticBeanstalk/site$ eb init
Do you want to set up SSH for your instances?
(y/n): y
Select a keypair.
1) my-keypair
2) [ Create new KeyPair ]
```

Pilih pasangan kunci jika Anda sudah memilikinya, atau ikuti prompt untuk membuatnya. Jika Anda tidak melihat prompt atau perlu mengubah pengaturan Anda nanti, jalankan eb init -i.

5. Buat lingkungan baru dan deploy aplikasi Anda dengan eb create. Elastic Beanstalk secara otomatis membuat file zip untuk aplikasi Anda dan memulainya pada port 5000.

kepada

```
~eb-dotnetcore/HelloElasticBeanstalk/site$ eb create dotnet-tutorial
```

Dibutuhkan sekitar lima menit untuk Elastic Beanstalk untuk menciptakan lingkungan Anda.

Langkah 4: Jalankan aplikasi Anda di Elastic Beanstalk

Ketika proses untuk membuat lingkungan Anda selesai, buka situs web Anda denganeb open.

```
~eb-dotnetcore/HelloElasticBeanstalk/site$ eb open
```

Selamat! Anda telah menerapkan aplikasi.NET Core di Linux dengan Elastic Beanstalk! Ini akan membuka jendela peramban menggunakan nama domain yang dibuat untuk aplikasi Anda.

Langkah 5: Bersihkan

Anda dapat menghentikan lingkungan Anda ketika Anda selesai bekerja dengan aplikasi Anda. Elastic Beanstalk AWS mengakhiri semua sumber daya yang terkait dengan lingkungan Anda.

Untuk mengakhiri lingkungan Elastic Beanstalk Anda dengan EB CLI jalankan perintah berikut.

```
~eb-dotnetcore/HelloElasticBeanstalk/site$ eb terminate
```

AWS sumber daya untuk aplikasi Anda

Anda baru saja membuat aplikasi instance tunggal. Ini berfungsi sebagai aplikasi sampel langsung dengan satu EC2 instance, sehingga tidak memerlukan penyeimbangan beban atau penskalaan otomatis. Untuk aplikasi contoh tunggal Elastic Beanstalk menciptakan sumber daya berikut: AWS

- EC2 instance — Mesin EC2 virtual Amazon yang dikonfigurasi untuk menjalankan aplikasi web pada platform yang Anda pilih.

  Setiap platform menjalankan satu set perangkat lunak, file konfigurasi, dan penulisan yang berbeda untuk mendukung versi bahasa, kerangka kerja, kontainer web tertentu, atau kombinasi dari semua ini. Sebagian besar platform menggunakan Apache atau nginx sebagai proksi terbalik yang memproses lalu lintas web di depan aplikasi web Anda, meneruskan permintaan ke aplikasi web, menyajikan aset statis, dan menghasilkan log akses dan kesalahan.
- Grup keamanan instans — Grup EC2 keamanan Amazon yang dikonfigurasi untuk memungkinkan lalu lintas masuk di port 80. Sumber daya ini memungkinkan lalu lintas HTTP dari penyeimbang beban mencapai EC2 instance yang menjalankan aplikasi web Anda. Secara default, lalu lintas tidak diizinkan di port lain.
- Bucket Amazon S3 – Lokasi penyimpanan untuk kode sumber, log, dan artifact lainnya yang dibuat saat Anda menggunakan Elastic Beanstalk.

- CloudWatch Alarm Amazon — Dua CloudWatch alarm yang memantau beban pada instans di lingkungan Anda dan dipicu jika beban terlalu tinggi atau terlalu rendah. Saat alarm terpicu, grup Auto Scaling Anda akan menaikkan atau menurunkan skala sebagai respons.
- AWS CloudFormation stack - Elastic AWS CloudFormation Beanstalk digunakan untuk meluncurkan sumber daya di lingkungan Anda dan menyebarkan perubahan konfigurasi. Sumber daya ditentukan di sebuah templat yang dapat Anda lihat di Konsol AWS CloudFormation.
- Nama domain — Nama domain yang merutekan ke aplikasi web Anda dalam formulir *`subdomain`*.*`region`*.elasticbeanstalk.com.

Elastic Beanstalk mengelola semua sumber daya tersebut. Ketika Anda mengakhiri lingkungan, Elastic Beanstalk mengakhiri semua sumber daya yang dimuatnya.

Langkah selanjutnya

Setelah lingkungan Anda menjalankan aplikasi, Anda dapat men-deploy versi baru aplikasi atau aplikasi yang berbeda kapan saja. Menyebarkan versi aplikasi baru sangat cepat karena tidak memerlukan penyediaan atau memulai ulang instance. EC2 Anda juga dapat menjelajahi lingkungan baru Anda menggunakan konsol Elastic Beanstalk. Untuk langkah-langkah mendetail, lihat Menjelajahi lingkungan Anda di bagian Memulai panduan ini.

Setelah Anda menerapkan satu atau dua contoh aplikasi dan siap untuk mulai mengembangkan dan menjalankan .NET Core pada aplikasi Linux secara lokal, lihat. Menyiapkan inti .NET Anda di lingkungan pengembangan Linux untuk Elastic Beanstalk

Terapkan dengan konsol Elastic Beanstalk

Anda juga dapat menggunakan konsol Elastic Beanstalk untuk meluncurkan aplikasi sampel. Untuk langkah-langkah rinci, lihat Membuat aplikasi contoh di Bab Memulai panduan ini.

Menyiapkan inti .NET Anda di lingkungan pengembangan Linux untuk Elastic Beanstalk

Topik ini memberikan instruksi untuk menyiapkan inti .NET di lingkungan pengembangan Linux untuk menguji aplikasi Anda secara lokal sebelum menerapkannya. AWS Elastic Beanstalk Ini juga merujuk situs web yang menyediakan instruksi instalasi untuk alat yang berguna.

Bagian-bagian

- Menginstal .NET Core SDK
- Menginstal IDE
- Instalasi AWS Toolkit for Visual Studio

Menginstal .NET Core SDK

Anda dapat menggunakan .NET Core SDK untuk mengembangkan aplikasi yang berjalan di Linux.

Lihat halaman unduhan .NET untuk mengunduh dan menginstal .NET Core SDK.

Menginstal IDE

Lingkungan pengembangan terintegrasi (IDEs) menyediakan berbagai fitur yang memfasilitasi pengembangan aplikasi. Jika Anda belum menggunakan IDE untuk pengembangan .NET, cobalah Visual Studio Community untuk memulai.

Lihat halaman Studio Visual Community untuk mengunduh dan menginstal Visual Studio Community.

Instalasi AWS Toolkit for Visual Studio

AWS Toolkit for Visual StudioIni adalah plugin open source untuk Visual Studio IDE yang memudahkan pengembang untuk mengembangkan, men-debug, dan menyebarkan aplikasi.NET menggunakan. AWS Lihat beranda Toolkit for Visual Studio untuk petunjuk instalasi.

Menggunakan inti Elastic Beanstalk .NET pada platform Linux

Topik ini menjelaskan cara mengonfigurasi, membangun, dan menjalankan inti .NET Anda di aplikasi Linux di Elastic Beanstalk.

AWS Elastic Beanstalk mendukung sejumlah cabang platform untuk berbagai versi framework .NET Core yang berjalan pada sistem operasi Linux. Lihat inti .NET di Linux di AWS Elastic Beanstalk Platform untuk daftar lengkapnya.

Untuk detail tentang berbagai cara memperluas platform berbasis Linux Elastic Beanstalk, lihat the section called “Memperluas platform Linux”.

.NET Core pada pertimbangan platform Linux

Server proksi

Elastic Beanstalk .NET Core di platform Linux mencakup proksi terbalik yang meneruskan permintaan untuk aplikasi Anda. Secara default, Elastic Beanstalk menggunakan NGINX sebagai server proxy. Anda dapat memilih untuk tidak menggunakan server proksi, dan mengonfigurasi Kestrel sebagai server web Anda. Kestrel disertakan secara default di templat proyek ASP.NET Core.

Struktur aplikasi

Anda dapat mempublikasikan aplikasi yang bergantung pada waktu aktif yang menggunakan waktu aktif .NET Core yang disediakan oleh Elastic Beanstalk. Anda juga dapat mempublikasikan aplikasi mandiri yang mencakup waktu aktif .NET Core dan dependensi aplikasi Anda di paket sumber. Untuk pelajari selengkapnya, lihat the section called “Aplikasi paketan”.

Konfigurasi platform

Untuk mengonfigurasi proses yang berjalan di instans server di lingkungan Anda, sertakan Procfile opsional di paket sumber Anda. `Procfile` diperlukan jika Anda memiliki lebih dari satu aplikasi di paket sumber Anda.

Kami merekomendasikan Anda agar selalu menyediakan `Procfile` di paket sumber dengan aplikasi Anda. Dengan cara ini, Anda secara tepat mengontrol proses mana yang dijalankan Elastic Beanstalk untuk aplikasi Anda.

Opsi konfigurasi tersedia di konsol Elastic Beanstalk untuk memodifikasi konfigurasi dari lingkungan yang sedang berjalan. Agar Anda tidak kehilangan konfigurasi lingkungan ketika mengakhirinya, Anda dapat menggunakan konfigurasi tersimpan untuk menyimpan pengaturan dan kemudian menerapkannya ke lingkungan lain.

Untuk menyimpan pengaturan di kode sumber, Anda dapat menyertakan file konfigurasi. Pengaturan di file konfigurasi diterapkan setiap kali Anda membuat lingkungan atau men-deploy aplikasi Anda. Anda juga dapat menggunakan file konfigurasi untuk menginstal paket, menjalankan penulisan, dan melakukan operasi penyesuaian instans lainnya selama deployment.

Pengaturan yang diterapkan di konsol Elastic Beanstalk mengganti pengaturan yang sama di file konfigurasi, jika ada. Hal tersebut memungkinkan Anda memiliki pengaturan default di file konfigurasi, dan mengganti keduanya dengan pengaturan khusus lingkungan di konsol tersebut. Untuk informasi

selengkapnya tentang yang diutamakan, dan metode lain untuk mengubah pengaturan, lihat Opsi konfigurasi.

Mengonfigurasi .NET Core Anda di lingkungan Linux

Setelan platform.NET Core pada Linux memungkinkan Anda untuk menyempurnakan perilaku instans Amazon EC2 Anda. Anda dapat mengedit konfigurasi instans Amazon lingkungan Elastic Beanstalk menggunakan konsol EC2 Elastic Beanstalk.

Gunakan konsol Elastic Beanstalk untuk mengaktifkan rotasi log ke Amazon S3 dan mengonfigurasi variabel yang dapat dibaca aplikasi Anda dari lingkungan.

Untuk mengonfigurasi .NET Core di lingkungan Linux Anda menggunakan konsol Elastic Beanstalk

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Dalam kategori konfigurasi Pembaruan, pemantauan, dan pencatatan, pilih Edit.

Opsi log

Bagian Opsi Log memiliki dua pengaturan:

- Profil instans – Menentukan profil instans yang memiliki izin untuk mengakses bucket Amazon S3 yang terkait dengan aplikasi Anda.
- Aktifkan rotasi file log ke Amazon S3 — Menentukan apakah file log untuk EC2 instans Amazon aplikasi Anda disalin ke bucket Amazon S3 yang terkait dengan aplikasi Anda.

Properti lingkungan

Bagian Properti Lingkungan memungkinkan Anda menentukan pengaturan konfigurasi lingkungan pada EC2 instance Amazon yang menjalankan aplikasi Anda. Properti lingkungan diberikan sebagai pasangan nilai kunci ke aplikasi.

Di dalam .NET Core di lingkungan Linux yang berjalan di Elastic Beanstalk, variabel lingkungan dapat diakses menggunakan `Environment.GetEnvironmentVariable("variable-name")`.

Sebagai contoh, Anda dapat membaca properti bernama `API_ENDPOINT` ke variabel dengan kode berikut.

```
string endpoint = Environment.GetEnvironmentVariable("API_ENDPOINT");
```

Lihat Variabel lingkungan dan pengaturan perangkat lunak lainnya untuk informasi selengkapnya.

.NET Core di namespace konfigurasi Linux

Anda dapat menggunakan file konfigurasi untuk mengatur opsi konfigurasi dan melakukan tugas-tugas konfigurasi instans lain selama deployment. Opsi konfigurasi dapat berupa platform khusus atau berlaku untuk semua platform di layanan Elastic Beanstalk secara keseluruhan. Opsi konfigurasi diatur ke dalam ruang nama.

.NET Core di platform Linux mendukung opsi di namespace berikut, selain opsi yang didukung untuk semua lingkungan Elastic Beanstalk:

- `aws:elasticbeanstalk:environment:proxy`— Pilih untuk menggunakan NGINX atau tidak ada server proxy. Nilai yang benar adalah `nginx` atau `none`.

File konfigurasi contoh berikut menunjukkan penggunaan .NET Core di opsi konfigurasi khusus Linux.

Example .ebextensions/proxy-settings.config

```
option_settings:
  aws:elasticbeanstalk:environment:proxy:
    ProxyServer: none
```

Elastic Beanstalk memberikan banyak opsi konfigurasi untuk menyesuaikan lingkungan Anda. Selain file konfigurasi, Anda juga dapat mengatur opsi konfigurasi menggunakan konsol tersebut, konfigurasi tersimpan, EB CLI, atau AWS CLI. Untuk informasi selengkapnya, lihat Opsi konfigurasi.

Aplikasi bundling untuk .NET Core pada platform Linux Elastic Beanstalk

Anda dapat menjalankan kedua aplikasi .NET Core yang bergantung pada waktu aktif dan mandiri di AWS Elastic Beanstalk.

Sebuah aplikasi yang bergantung pada waktu aktif menggunakan waktu aktif .NET Core yang Elastic Beanstalk berikan untuk menjalankan aplikasi Anda. Elastic Beanstalk menggunakan file

`runtimeconfig.json` di paket sumber Anda untuk menentukan waktu aktif yang digunakan untuk aplikasi Anda. Elastic Beanstalk memilih waktu aktif terbaru yang kompatibel dan tersedia untuk aplikasi Anda.

Sebuah aplikasi mandiri mencakup waktu aktif .NET Core, aplikasi Anda, dan dependensinya. Untuk menggunakan versi waktu aktif .NET Core yang tidak disertakan Elastic Beanstalk di platformnya, sediakan aplikasi mandiri.

Contoh

Anda dapat mengompilasi kedua aplikasi mandiri dan yang bergantung pada waktu aktif dengan perintah `dotnet publish`. Untuk pelajari selengkapnya tentang penerbitan aplikasi .NET Core, lihat Gambaran umum penerbitan aplikasi .NET Core di dokumentasi .NET Core.

Struktur file contoh berikut menentukan satu aplikasi yang menggunakan waktu aktif .NET Core yang disediakan Elastic Beanstalk.

```
### appsettings.Development.json
### appsettings.json
### dotnetcoreapp.deps.json
### dotnetcoreapp.dll
### dotnetcoreapp.pdb
### dotnetcoreapp.runtimeconfig.json
### web.config
### Procfile
### .ebextensions
### .platform
```

Anda dapat menyertakan beberapa aplikasi di paket sumber Anda. Contoh berikut menentukan dua aplikasi untuk dijalankan di web server yang sama. Untuk menjalankan beberapa aplikasi, Anda harus menyertakan Procfile di paket sumber Anda. Untuk contoh aplikasi lengkap, lihat dotnet-core-linux-multiple-apps.zip.

```
### DotnetMultipleApp1
#   ### Amazon.Extensions.Configuration.SystemsManager.dll
#   ### appsettings.Development.json
#   ### appsettings.json
#   ### AWSSDK.Core.dll
#   ### AWSSDK.Extensions.NETCore.Setup.dll
#   ### AWSSDK.SimpleSystemsManagement.dll
#   ### DotnetMultipleApp1.deps.json
```

```
#   ### DotnetMultipleApp1.dll
#   ### DotnetMultipleApp1.pdb
#   ### DotnetMultipleApp1.runtimeconfig.json
#   ### Microsoft.Extensions.PlatformAbstractions.dll
#   ### Newtonsoft.Json.dll
#   ### web.config
### DotnetMultipleApp2
#   ### Amazon.Extensions.Configuration.SystemsManager.dll
#   ### appsettings.Development.json
#   ### appsettings.json
#   ### AWSSDK.Core.dll
#   ### AWSSDK.Extensions.NETCore.Setup.dll
#   ### AWSSDK.SimpleSystemsManagement.dll
#   ### DotnetMultipleApp2.deps.json
#   ### DotnetMultipleApp2.dll
#   ### DotnetMultipleApp2.pdb
#   ### DotnetMultipleApp2.runtimeconfig.json
#   ### Microsoft.Extensions.PlatformAbstractions.dll
#   ### Newtonsoft.Json.dll
#   ### web.config
### Procfile
### .ebextensions
### .platform
```

Menggunakan Procfile untuk mengkonfigurasi .NET Core Anda di lingkungan Linux Elastic Beanstalk

Untuk menjalankan beberapa aplikasi di server web yang sama, Anda harus menyertakan `Procfile` di paket sumber Anda yang memberitahukan Elastic Beanstalk aplikasi mana yang akan dijalankan.

Kami merekomendasikan Anda agar selalu menyediakan `Procfile` di paket sumber dengan aplikasi Anda. Dengan cara ini, Anda secara tepat mengontrol proses mana yang dijalankan Elastic Beanstalk untuk aplikasi Anda dan argumen yang diterima proses ini.

Contoh berikut menggunakan `Procfile` untuk menentukan dua aplikasi untuk Elastic Beanstalk agar berjalan di server web yang sama.

Example Procfile

```
web: dotnet ./dotnet-core-app1/dotnetcoreapp1.dll
web2: dotnet ./dotnet-core-app2/dotnetcoreapp2.dll
```

Untuk detail tentang menulis dan menggunakan a`Procfile`, lihatBuildfile dan Procfile.

Mengonfigurasi server proksi

AWS Elastic Beanstalk menggunakan NGINX sebagai proxy terbalik untuk menyampaikan permintaan ke aplikasi Anda. Elastic Beanstalk menyediakan konfigurasi NGINX default yang dapat Anda perpanjang atau ganti sepenuhnya dengan konfigurasi Anda sendiri.

Secara default, Elastic Beanstalk mengonfigurasi proxy NGINX untuk meneruskan permintaan ke aplikasi Anda pada port 5000. Anda dapat mengganti port default dengan mengatur `PORT` properti lingkungan ke port yang didengarkan aplikasi utama Anda.

> **Note**
>
> Port yang didengarkan aplikasi Anda tidak memengaruhi port yang didengarkan server NGINX untuk menerima permintaan dari penyeimbang beban.

Mengkonfigurasi server proxy pada versi platform Anda

Semua AL2 AL2 023/platform mendukung fitur konfigurasi proxy yang seragam. Untuk informasi selengkapnya tentang mengonfigurasi server proxy pada versi platform Anda yang menjalankan AL2 023/AL2, lihat. Konfigurasi proksi terbalik

Contoh file konfigurasi berikut memperluas konfigurasi NGINX lingkungan Anda. Konfigurasi mengarahkan permintaan ke `/api` ke aplikasi web kedua yang mendengarkan port 5200 di web server. Secara default, Elastic Beanstalk meneruskan permintaan ke satu aplikasi yang mendengarkan port 5000.

Example **`01_custom.conf`**

```
location /api {
     proxy_pass          http://127.0.0.1:5200;
     proxy_http_version  1.1;

     proxy_set_header   Upgrade $http_upgrade;
     proxy_set_header   Connection $http_connection;
     proxy_set_header   Host $host;
     proxy_cache_bypass $http_upgrade;
     proxy_set_header   X-Forwarded-For $proxy_add_x_forwarded_for;
     proxy_set_header   X-Forwarded-Proto $scheme;
```

```
}
```

The AWS Toolkit for Visual Studio - Bekerja dengan.Net Core pada Elastic Beanstalk

Topik ini menunjukkan bagaimana Anda dapat melakukan tugas berikut menggunakan AWS Toolkit for Visual Studio:

- Membuat aplikasi web ASP.NET Core menggunakan templat Visual Studio.
- Membuat lingkungan Elastic Beanstalk Amazon Linux.
- Men-deploy aplikasi web ASP.NET Core ke lingkungan Amazon Linux baru.

Topik ini juga mengeksplorasi bagaimana Anda dapat menggunakan AWS Toolkit for Visual Studio untuk mengelola lingkungan aplikasi Elastic Beanstalk Anda dan memantau kesehatan aplikasi Anda.

AWS Toolkit for Visual Studio Ini adalah plugin untuk Visual Studio IDE. Dengan kit alat, Anda dapat men-deploy dan mengelola aplikasi di Elastic Beanstalk ketika Anda bekerja di lingkungan Visual Studio Anda.

Bagian-bagian

- Prasyarat
- Membuat proyek aplikasi baru
- Membuat lingkungan Elastic Beanstalk dan men-deploy aplikasi Anda
- Mengakhiri lingkungan
- Mengelola lingkungan aplikasi Elastic Beanstalk Anda
- Pemantauan kondisi aplikasi

Prasyarat

Sebelum memulai tutorial ini, Anda perlu menginstal AWS Toolkit for Visual Studio. Untuk petunjuk, lihat Menyiapkan AWS Toolkit for Visual Studio.

Jika Anda belum pernah menggunakan toolkit sebelumnya, hal pertama yang harus Anda lakukan setelah menginstal toolkit adalah mendaftarkan AWS kredensialnya dengan toolkit. Untuk informasi selengkapnya tentang ini, lihat Menyediakan kredensial AWS.

Membuat proyek aplikasi baru

Jika Anda tidak memiliki proyek aplikasi .NET Core di Visual Studio, Anda dapat dengan mudah membuatnya menggunakan salah satu templat proyek Visual Studio.

Untuk membuat proyek aplikasi web ASP.NET Core baru

1. Di Visual Studio, di menu File, pilih Baru dan kemudian pilih Proyek.
2. Di kotak dialog Buat proyek baru, pilih C#, pilih Linux, dan kemudian pilih Cloud.
3. Dari daftar templat proyek yang ditampilkan, pilih Aplikasi Web ASP.NET Core, dan kemudian pilih Selanjutnya.

   > **Note**
   >
   > Jika Anda tidak melihat Aplikasi Web ASP.NET Core tercantum di templat proyek, Anda dapat menginstalnya di Visual Studio.
   >
   > 1. Gulir ke bagian bawah daftar templat dan pilih tautan Instal lebih banyak alat dan fitur yang terletak di daftar templat.
   > 2. Jika Anda diminta untuk mengizinkan aplikasi Visual Studio untuk membuat perubahan pada perangkat Anda, pilih Ya.
   > 3. Pilih tab Beban Kerja, lalu pilih ASP.NET dan pengembangan web.
   > 4. Pilih tombol Modifikasi. Penginstal Studio Visual menginstal templat proyek.
   > 5. Setelah penginstal selesai, keluar dari panel untuk kembali ke tempat yang Anda tinggalkan di Visual Studio.

4. Di kotak dialog Konfigurasikan proyek baru, masukkan Nama proyek. Nama solusi default ke nama proyek Anda. Selanjutnya, pilih Buat.
5. Di kotak dialog Buat aplikasi web ASP.NET Core baru, pilih .NET Core, dan kemudian pilih ASP.NET Core 3.1. Dari daftar tipe aplikasi yang ditampilkan, pilih Aplikasi Web, lalu pilih tombol Buat.

Visual Studio menampilkan kotak dialog Buat Proyek ketika membuat aplikasi Anda. Setelah Visual Studio selesai menghasilkan aplikasi Anda, panel dengan nama aplikasi Anda akan ditampilkan.

Membuat lingkungan Elastic Beanstalk dan men-deploy aplikasi Anda

Bagian ini menjelaskan cara membuat lingkungan Elastic Beanstalk untuk aplikasi Anda dan men-deploy aplikasi Anda ke lingkungan tersebut.

Untuk membuat lingkungan baru dan men-deploy aplikasi Anda

1. Di Visual Studio, pilih Lihat, kemudian Penjelajah Solusi.
2. Di Penjelajah Solusi, buka menu konteks (klik kanan) untuk aplikasi Anda, kemudian pilih Publikasikan ke AWS Elastic Beanstalk.
3. Di wizard Publikasikan ke AWS Elastic Beanstalk, masukkan informasi akun Anda.

    a. Untuk Profil akun yang digunakan, pilih akun default Anda atau pilih ikon Tambah akun lain untuk memasukkan informasi akun baru.

    b. Untuk Wilayah, pilih Wilayah tempat Anda ingin men-deploy aplikasi Anda. Untuk informasi tentang AWS Wilayah yang tersedia, lihat AWS Elastic Beanstalk Titik Akhir dan Kuota di. Referensi Umum AWS Jika Anda memilih Wilayah yang tidak didukung oleh Elastic Beanstalk, maka opsi untuk men-deploy ke Elastic Beanstalk tidak tersedia.

    c. Pilih Buat lingkungan aplikasi baru, lalu pilih Selanjutnya.
4. Di kotak dialog Lingkungan Aplikasi, masukkan detail lingkungan aplikasi baru Anda.
5. Pada kotak dialog AWSopsi berikutnya, setel EC2 opsi Amazon dan opsi AWS terkait lainnya untuk aplikasi yang Anda gunakan.

    a. Untuk jenis Container pilih 64bit Amazon Linux 2 v yang ***<n.n.n>*** menjalankan .NET Core.

    > **Note**
    >
    > Kami sarankan Anda memilih versi platform Linux saat ini. Versi ini berisi keamanan terbaru dan perbaikan bug yang disertakan di Amazon Machine Image (AMI) terbaru kami.

    b. Untuk Tipe Instans, pilih t2.micro. (Memilih tipe instans micro meminimalkan biaya yang terkait dengan menjalankan instans.)

    c. Untuk Pasangan kunci, pilih Buat pasangan kunci baru. Masukkan nama untuk pasangan kunci baru, lalu pilih OK. (Di contoh ini, kami menggunakan **`myuseastkeypair`**.) Sebuah key pair memungkinkan akses remote desktop ke instans Amazon Anda. EC2 Untuk

informasi selengkapnya tentang pasangan EC2 kunci Amazon, lihat Menggunakan Kredensial di Panduan Pengguna Amazon Elastic Compute Cloud.

d. Untuk aplikasi lalu lintas yang sederhana dan rendah, pilih Lingkungan instans tunggal. Untuk informasi selengkapnya, lihat Jenis lingkungan

e. Pilih Selanjutnya.

Untuk informasi lebih lanjut tentang AWS opsi yang tidak digunakan dalam contoh ini, pertimbangkan halaman-halaman berikut:

- Untuk Gunakan AMI khusus, lihat Menggunakan image mesin Amazon kustom (AMI) di lingkungan Elastic Beanstalk Anda.
- Jika Anda tidak memilih Lingkungan instans tunggal, Anda harus memilih Tipe keseimbangan beban. Lihat Penyeimbang beban untuk lingkungan Elastic Beanstalk Anda untuk informasi selengkapnya.
- Elastic Beanstalk menggunakan konfigurasi Amazon VPC (Amazon Virtual Private Cloud) default jika Anda tidak memilih Gunakan VPC non-default. Untuk informasi selengkapnya, lihat Menggunakan Elastic Beanstalk dengan Amazon VPC.
- Memilih opsi Mengaktifkan Deployment Bergulir membagi deployment ke dalam batch untuk menghindari potensi downtime selama deployment. Untuk informasi selengkapnya, lihat Men-deploy aplikasi ke lingkungan Elastic Beanstalk.
- Memilih opsi Akses Basis Data Relasional menghubungkan lingkungan Elastic Beanstalk Anda ke basis data Amazon RDS yang dibuat sebelumnya dengan Grup Keamanan DB Amazon RDS. Untuk informasi selengkapnya, lihat Mengontrol Akses dengan Grup Keamanan di Panduan Pengguna Amazon RDS.

6. Pilih Selanjutnya di kotak dialog Izin.
7. Pilih Selanjutnya di kotak dialog Opsi Aplikasi.
8. Tinjau opsi deployment Anda. Setelah memverifikasi bahwa pengaturan Anda sudah benar, pilih Deploy.

Aplikasi web ASP.NET Core Anda diekspor sebagai file web deploy. File web deploy Anda diunggah ke Amazon S3, dan terdaftar sebagai versi aplikasi baru dengan Elastic Beanstalk. Fitur deployment Elastic Beanstalk memantau lingkungan Anda sampai tersedia dengan kode yang baru di-deploy. Status untuk lingkungan Anda ditampilkan di tab Env:<nama lingkungan>. Setelah pembaruan status Lingkungan sehat, Anda dapat memilih alamat URL untuk meluncurkan aplikasi web.

Mengakhiri lingkungan

Untuk menghindari timbulnya biaya untuk AWS sumber daya yang tidak digunakan, Anda dapat menggunakan AWS Toolkit for Visual Studio untuk menghentikan lingkungan yang sedang berjalan.

> **Note**
>
> Anda selalu dapat meluncurkan lingkungan baru menggunakan versi yang sama nantinya.

Untuk mengakhiri lingkungan

1. Perluas simpul Elastic Beanstalk dan simpul aplikasi Anda. Di AWS Explorer, buka menu konteks (klik kanan) untuk lingkungan aplikasi Anda dan pilih Mengakhiri Lingkungan.
2. Saat diminta, pilih Ya untuk mengonfirmasi bahwa Anda ingin mengakhiri lingkungan. Dibutuhkan beberapa menit bagi Elastic Beanstalk untuk AWS menghentikan sumber daya yang berjalan di lingkungan.

Status untuk lingkungan Anda di tab Env:<nama lingkungan> berubah menjadi Mengakhiri dan akhirnya Diakhiri.

> **Note**
>
> Ketika Anda mengakhiri lingkungan Anda, CNAME yang terkait dengan lingkungan yang diakhiri akan tersedia bagi siapa saja untuk digunakan.

Mengelola lingkungan aplikasi Elastic Beanstalk Anda

Bagian ini menjelaskan pengaturan layanan tertentu yang dapat Anda edit di Toolkit for Visual Studio AWS sebagai bagian dari konfigurasi lingkungan aplikasi Anda. Dengan AWS Toolkit for Visual Studio dan AWS Management Console, Anda dapat mengubah penyediaan dan konfigurasi sumber daya yang digunakan oleh AWS lingkungan aplikasi Anda. Untuk informasi tentang cara mengelola lingkungan aplikasi menggunakan AWS Management Console, lihatMenciptakan lingkungan di Elastic Beanstalk.

Mengubah pengaturan konfigurasi lingkungan

Saat Anda menerapkan aplikasi Anda, Elastic Beanstalk mengonfigurasi beberapa layanan komputasi awan yang terhubung. AWS Anda dapat mengontrol bagaimana layanan individual ini dikonfigurasi dengan menggunakan AWS Toolkit for Visual Studio.

Untuk mengedit pengaturan lingkungan aplikasi

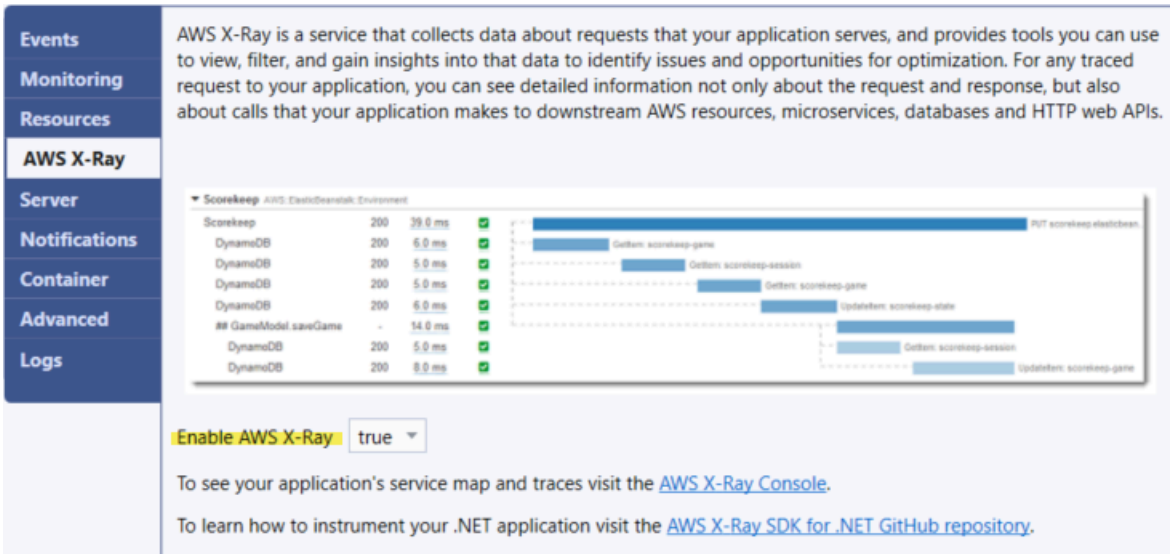
1. Di Visual Studio, di menu File, pilih AWS Explorer.
2. Perluas simpul Elastic Beanstalk dan simpul aplikasi Anda. Buka menu konteks (klik kanan) untuk lingkungan aplikasi Anda dan pilih Lihat Status.

   Sekarang, Anda dapat mengonfigurasi pengaturan untuk hal berikut:

   - AWS X-Ray
   - Server
   - Penyeimbang beban (hanya berlaku untuk lingkungan banyak instans)
   - Auto Scaling (hanya berlaku untuk lingkungan dengan banyak instans)
   - Notifikasi
   - Kontainer
   - Opsi Konfigurasi Lanjutan

Mengkonfigurasi AWS X-Ray menggunakan AWS toolkit untuk Visual Studio

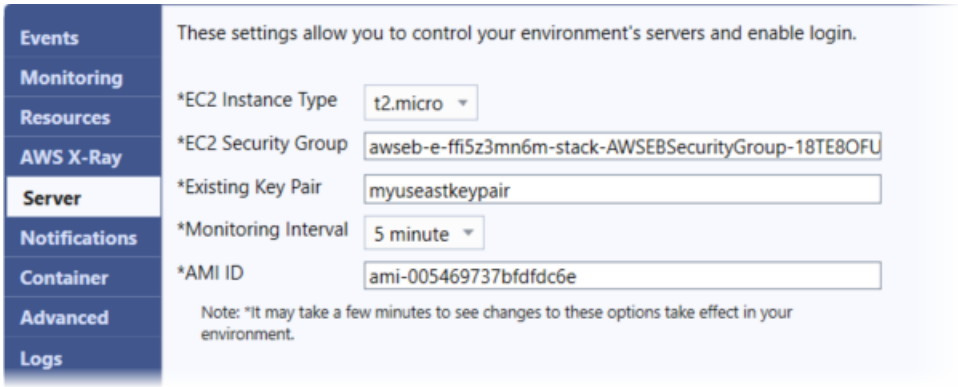
AWS X-Ray menyediakan kemampuan penelusuran permintaan, pengumpulan pengecualian, dan pembuatan profil. Dengan panel AWS X-Ray, Anda dapat mengaktifkan atau menonaktifkan X-Ray untuk aplikasi Anda. Untuk informasi selengkapnya tentang X-Ray, lihat Panduan Developer AWS X-Ray.



Mengkonfigurasi EC2 instance menggunakan AWS toolkit untuk Visual Studio

Anda dapat menggunakan Amazon Elastic Compute Cloud (Amazon EC2) untuk meluncurkan dan mengelola instans server di pusat data Amazon. Anda dapat menggunakan instans EC2 server Amazon kapan saja, selama yang Anda butuhkan, dan untuk tujuan hukum apa pun. Instans tersedia dalam berbagai ukuran dan konfigurasi. Untuk informasi selengkapnya, lihat Amazon EC2.

Anda dapat mengedit konfigurasi EC2 instans Amazon dengan tab Server di dalam tab lingkungan aplikasi Anda di AWS Toolkit for Visual Studio.



Jenis EC2 instans Amazon

Tipe instans menampilkan tipe instans yang tersedia untuk aplikasi Elastic Beanstalk Anda. Ubah tipe instans untuk memilih server dengan karakteristik (termasuk ukuran memori dan kekuatan CPU) yang paling sesuai untuk aplikasi Anda. Sebagai contoh, aplikasi dengan operasi intensif dan berjalan lama dapat memerlukan lebih banyak CPU atau memori.

Untuk informasi selengkapnya tentang jenis EC2 instans Amazon yang tersedia untuk aplikasi Elastic Beanstalk, lihat Jenis Instans di Panduan Pengguna Amazon Elastic Compute Cloud.

Grup EC2 keamanan Amazon

Anda dapat mengontrol akses ke aplikasi Elastic Beanstalk Anda menggunakan EC2 Amazon Security Group. Grup keamanan menentukan aturan firewall untuk instans Anda. Aturan ini menentukan lalu lintas jaringan masuk mana yang harus dikirim ke instans Anda. Semua lalu lintas masuk lainnya dibuang. Anda dapat memodifikasi aturan untuk grup kapan saja. Aturan baru secara otomatis diberlakukan untuk semua instans berjalan dan instans yang diluncurkan di masa mendatang.

Anda dapat menentukan Grup EC2 Keamanan Amazon mana yang mengontrol akses ke aplikasi Elastic Beanstalk Anda. Untuk melakukan ini, masukkan nama grup EC2 keamanan Amazon tertentu (memisahkan beberapa grup secruity dengan koma) ke dalam kotak teks Grup EC2 Keamanan. Anda dapat melakukannya dengan menggunakan AWS Management Console atau AWS Toolkit for Visual Studio.

Untuk membuat grup keamanan menggunakan AWS toolkit untuk Visual Studio

1. Di Visual Studio, di AWS Explorer, perluas EC2 node Amazon, lalu pilih Grup Keamanan.
2. Pilih Buat Grup Keamanan, lalu masukkan nama dan deskripsi untuk grup keamanan Anda.
3. Pilih OK.

Untuk informasi selengkapnya tentang Grup EC2 Keamanan Amazon, lihat Menggunakan Grup Keamanan di Panduan Pengguna Amazon Elastic Compute Cloud.

Pasangan EC2 kunci Amazon

Anda dapat masuk dengan aman ke EC2 instans Amazon yang disediakan untuk aplikasi Elastic Beanstalk Anda dengan key pair Amazon. EC2

> **Important**
>
> Anda harus membuat EC2 key pair Amazon dan mengonfigurasi EC2 instans Amazon yang disediakan oleh Elastic Beanstalk agar dapat mengakses instans ini. Anda dapat membuat key pair menggunakan AWS wizard Publish to di dalam AWS Toolkit for Visual Studio saat Anda menerapkan aplikasi ke Elastic Beanstalk. Jika Anda ingin membuat pasangan kunci

tambahan menggunakan Toolkit, ikuti langkah-langkah yang dijelaskan di sini. Atau, Anda dapat mengatur pasangan EC2 kunci Amazon menggunakan AWS Management Console. Untuk petunjuk cara membuat key pair untuk Amazon EC2, lihat Panduan Memulai Amazon Elastic Compute Cloud.

Kotak teks Existing Key Pair memungkinkan Anda menentukan nama EC2 key pair Amazon yang dapat Anda gunakan untuk masuk dengan aman ke EC2 instans Amazon yang menjalankan aplikasi Elastic Beanstalk Anda.

Untuk menentukan nama EC2 key pair Amazon

1. Perluas EC2 node Amazon dan pilih Pasangan Kunci.
2. Pilih Buat Pasangan Kunci dan masukkan nama pasangan kunci.
3. Pilih OK.

Untuk informasi selengkapnya tentang pasangan EC2 kunci Amazon, buka Menggunakan EC2 Kredensial Amazon di Panduan Pengguna Amazon Elastic Compute Cloud. Untuk informasi selengkapnya tentang menghubungkan ke EC2 instans Amazon, lihat

Interval pemantauan

Secara default, hanya CloudWatch metrik dasar Amazon yang diaktifkan. Mereka mengembalikan data dengan periode lima menit. Anda dapat mengaktifkan CloudWatch metrik satu menit yang lebih terperinci dengan memilih 1 menit untuk Interval Pemantauan di bagian Server pada tab Konfigurasi untuk lingkungan Anda di. AWS Toolkit for Eclipse

Note

Biaya CloudWatch layanan Amazon dapat berlaku untuk metrik interval satu menit. Lihat Amazon CloudWatch untuk informasi lebih lanjut.

ID AMI khusus

Anda dapat mengganti AMI default yang digunakan untuk EC2 instans Amazon Anda dengan AMI kustom Anda sendiri dengan memasukkan pengenal AMI kustom Anda ke dalam kotak ID AMI Kustom di bagian Server pada tab Konfigurasi untuk lingkungan Anda di. AWS Toolkit for Eclipse

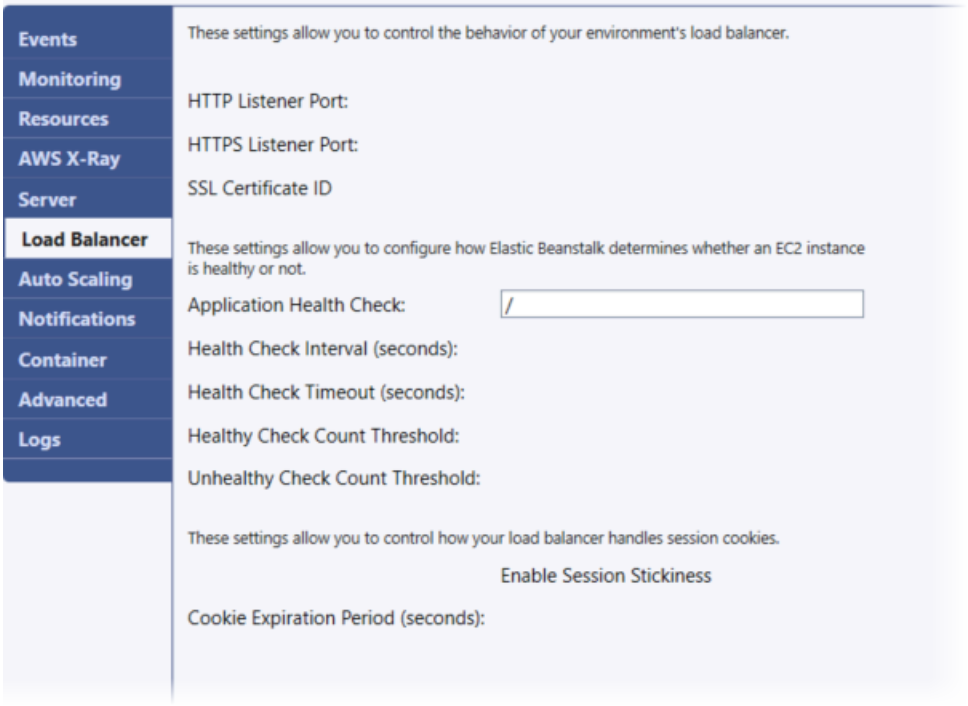
> **Important**
>
> Menggunakan AMI Anda sendiri adalah tugas lanjutan yang harus Anda lakukan dengan hati-hati. Jika Anda membutuhkan AMI khusus, kami sarankan Anda mulai dengan Elastic Beanstalk AMI default dan kemudian memodifikasinya. Agar dianggap sehat, Elastic Beanstalk EC2 mengharapkan instans Amazon memenuhi serangkaian persyaratan, termasuk memiliki manajer host yang berjalan. Jika persyaratan ini tidak terpenuhi, lingkungan Anda mungkin tidak bekerja dengan baik.

Mengkonfigurasi Elastic Load Balancing menggunakan toolkit AWS untuk Visual Studio

Elastic Load Balancing adalah layanan web Amazon yang membantu Anda meningkatkan ketersediaan dan skalabilitas aplikasi Anda. Layanan ini memudahkan Anda untuk mendistribusikan beban aplikasi antara dua atau lebih EC2 instance Amazon. Elastic Load Balancing meningkatkan ketersediaan melalui menyediakan redundansi tambahan dan mendukung pertumbuhan lalu lintas untuk aplikasi Anda.

Dengan Elastic Load Balancing, Anda dapat secara otomatis mendistribusikan dan menyeimbangkan lalu lintas aplikasi masuk di antara semua instans berjalan Anda. Anda juga dapat dengan mudah menambahkan instans baru ketika diperlukan untuk meningkatkan kapasitas aplikasi Anda.

Elastic Beanstalk secara otomatis menyediakan Elastic Load Balancing saat Anda men-deploy aplikasi. Anda dapat mengedit konfigurasi instans Amazon lingkungan EC2 Elastic Beanstalk dengan tab Load Balancer di dalam tab lingkungan aplikasi Anda di Toolkit AWS for Visual Studio.



Bagian berikut menjelaskan parameter Elastic Load Balancing yang dapat Anda konfigurasi untuk aplikasi Anda.

Port

Load balancer yang disediakan untuk menangani permintaan aplikasi Elastic Beanstalk Anda mengirimkan permintaan ke instans Amazon yang menjalankan aplikasi Anda. EC2 Penyeimbang beban yang disediakan dapat mendengarkan permintaan pada port HTTP dan HTTPS serta merutekan permintaan ke EC2 instans Amazon di aplikasi Anda. AWS Elastic Beanstalk Secara default, penyeimbang beban menangani permintaan di port HTTP. Agar dapat bekerja, setidaknya salah satu port (HTTP atau HTTPS) harus diaktifkan.



> **Important**
>
> Pastikan bahwa port yang Anda tentukan tidak terkunci; jika tidak, Anda tidak akan dapat terhubung ke aplikasi Elastic Beanstalk Anda.

Mengontrol port HTTP

Untuk mematikan port HTTP, pilih OFF untuk Port Listener HTTP. Untuk mengaktifkan port HTTP, Anda pilih port HTTP (misalnya, 80) dari daftar.

> **Note**
>
> Untuk mengakses lingkungan Anda menggunakan port selain port default 80, seperti port 8080, tambahkan listener ke penyeimbang beban yang ada dan konfigurasikan listener baru untuk mendengarkan di port tersebut.
> Misalnya, menggunakan penyeimbang beban AWS CLI untuk Klasik, ketik perintah berikut, ganti *LOAD_BALANCER_NAME* dengan nama penyeimbang beban Anda untuk Elastic Beanstalk.
>
> ```
> aws elb create-load-balancer-listeners --load-balancer-name LOAD_BALANCER_NAME
>  --listeners "Protocol=HTTP, LoadBalancerPort=8080, InstanceProtocol=HTTP,
>  InstancePort=80"
> ```
>
> Misalnya, menggunakan AWS CLI untuk Application Load Balancers, ketik perintah berikut, ganti *LOAD_BALANCER_ARN* dengan ARN penyeimbang beban Anda untuk Elastic Beanstalk.
>
> ```
> aws elbv2 create-listener --load-balancer-arn LOAD_BALANCER_ARN --protocol HTTP
>  --port 8080
> ```
>
> Jika Anda ingin Elastic Beanstalk memantau lingkungan Anda, jangan hapus listener di port 80.

Mengontrol port HTTPS

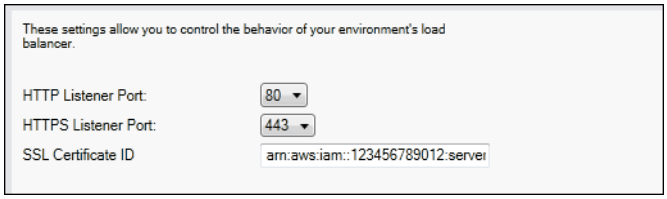
Elastic Load Balancing mendukung protokol HTTPS/TLS untuk mengaktifkan enkripsi lalu lintas untuk koneksi klien ke penyeimbang beban. Koneksi dari penyeimbang beban ke EC2 instance menggunakan enkripsi plaintext. Secara default, port HTTPS dimatikan.

Untuk mengaktifkan port HTTPS

1. Buat sertifikat baru menggunakan AWS Certificate Manager (ACM) atau unggah sertifikat dan kunci ke AWS Identity and Access Management (IAM). Untuk informasi selengkapnya tentang permintaan sertifikat ACM, lihat Meminta Sertifikat di AWS Certificate Manager Panduan

Pengguna. Untuk informasi tentang mengimpor sertifikat pihak ke tiga ke ACM, lihat Mengimpor Sertifikat di Panduan Pengguna AWS Certificate Manager . Jika ACM tidak tersedia di wilayah Anda, gunakan AWS Identity and Access Management (IAM) untuk mengunggah sertifikat pihak ketiga. Layanan ACM dan IAM menyimpan sertifikat dan menyediakan Amazon Resource Name (ARN) untuk sertifikat SSL. Untuk informasi selengkapnya tentang pembuatan dan pengunggahan sertifikat ke IAM, lihat Bekerja dengan Sertifikat Server di Panduan Pengguna IAM.

2. Tentukan port HTTPS dengan memilih port untuk Port Listener HTTPS.



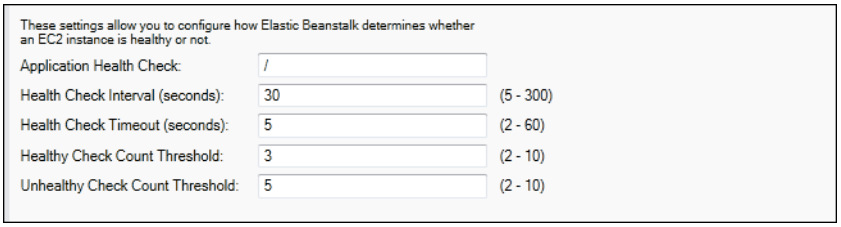
3. Untuk ID Sertifikat SSL, masukkan Amazon Resources Name (ARN) dari sertifikat SSL Anda. Misalnya, **`arn:aws:iam::123456789012:server-certificate/abc/certs/build`** atau **`arn:aws:acm:us-east-2:123456789012:certificate/12345678-12ab-34cd-56ef-12345678`**. Gunakan sertifikat SSL yang Anda buat atau unggah di langkah 1.

Untuk mematikan port HTTPS, pilih OFF untuk HTTPS Listener Port.

Pemeriksaan kondisi

Definisi pemeriksaan kondisi mencakup URL untuk di-kueri untuk instans kondisi. Secara default, Elastic Beanstalk menggunakan TCP:80 untuk kontainer bukan warisan dan HTTP:80 untuk kontainer warisan. Anda dapat mengganti URL default untuk mencocokkan sumber daya yang ada di aplikasi Anda (misalnya, `/myapp/default.aspx`) dengan memasukkannya ke kotak URL Pemeriksaan Kondisi Aplikasi. Jika Anda mengganti URL default, maka Elastic Beanstalk menggunakan HTTP untuk meng-kueri sumber daya. Untuk memeriksa apakah Anda menggunakan tipe kontainer warisan, lihat the section called “Mengapa beberapa versi platform ditandai sebagai legasi?”

Anda dapat mengontrol pengaturan untuk pemeriksaan kesehatan menggunakan bagian Pemeriksaan Kesehatan EC2 Instans pada panel Load Balancing.



Definisi pemeriksaan kondisi mencakup URL untuk di-kueri untuk instans kondisi. Ganti URL default untuk mencocokkan sumber daya yang ada di aplikasi Anda (misalnya, `/myapp/index.jsp`) dengan memasukkannya ke kotak URL Pemeriksaan Kondisi Aplikasi.

Daftar berikut menjelaskan parameter pemeriksaan kondisi yang dapat Anda atur untuk aplikasi Anda.

- Untuk Interval Pemeriksaan Kesehatan (detik), masukkan jumlah detik Elastic Load Balancing menunggu di antara pemeriksaan kesehatan untuk instans Amazon aplikasi Anda. EC2
- Untuk Timeout Pemeriksaan Kondisi (detik), tentukan jumlah detik tunggu Elastic Load Balancing untuk sebuah respons sebelum menganggap instans tidak responsif.
- Untuk Ambang Batas Jumlah Pemeriksaan Sehat dan Ambang Batas Jumlah Pemeriksaan Tidak Sehat, tentukan jumlah probe URL yang berhasil atau tidak berhasil berturut-turut sebelum Elastic Load Balancing mengubah status kondisi instans. Sebagai contoh, menentukan **5** untuk Ambang Batas Jumlah Pemeriksaan Tidak Sehat berarti URL harus mengembalikan pesan kesalahan atau timeout lima kali berturut-turut sebelum Elastic Load Balancing menganggap pemeriksaan kondisi gagal.

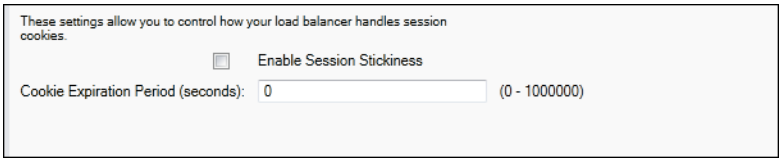
Sesi

Secara default, penyeimbang beban merutekan setiap permintaan secara independen ke instans server dengan beban terkecil. Sebagai perbandingan, sesi lekat mengikat sesi pengguna ke instans server tertentu, sehingga semua permintaan yang datang dari pengguna selama sesi dikirim ke server instans yang sama.

Elastic Beanstalk menggunakan cookie HTTP yang dihasilkan penyeimbang beban saat sesi lekat diaktifkan untuk aplikasi. Penyeimbang beban menggunakan cookie yang dihasilkan penyeimbang beban khusus untuk melacak instans aplikasi untuk setiap permintaan. Ketika penyeimbang beban menerima permintaan, pertama-tama penyeimbang beban memeriksa apakah cookie ini ada di permintaan. Jika ada, permintaan tersebut dikirim ke instans aplikasi yang ditentukan di cookie. Jika tidak ada cookie, penyeimbang beban memilih instans aplikasi berdasarkan algoritme penyeimbangan beban yang ada. Cookie dimasukkan ke dalam respons untuk mengikat permintaan

berikutnya dari pengguna yang sama ke instans aplikasi. Konfigurasi kebijakan menentukan kedaluwarsa cookie, yang menetapkan durasi validitas untuk setiap cookie.

Anda dapat menggunakan bagian Sesi di tab Penyeimbang Beban untuk menentukan apakah penyeimbang beban untuk aplikasi Anda mengizinkan kelengketan sesi.



Untuk informasi selengkapnya di Elastic Load Balancing, lihat Panduan Developer Elastic Load Balancing.

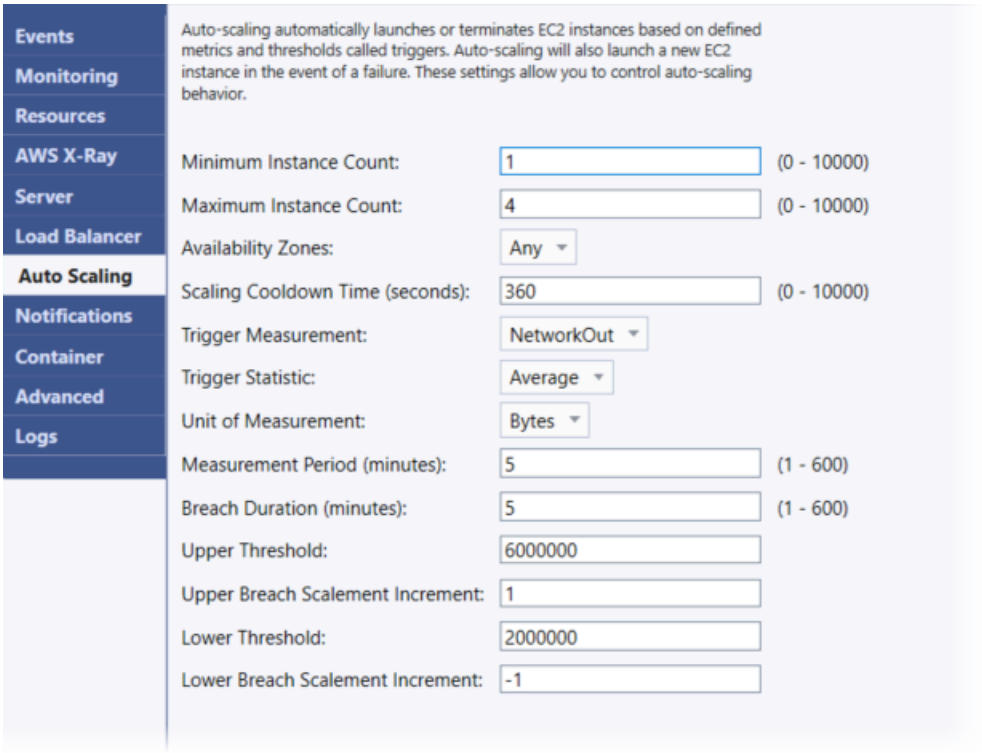
Mengkonfigurasi Auto Scaling menggunakan toolkit AWS untuk Visual Studio

Amazon EC2 Auto Scaling adalah layanan web Amazon yang dirancang untuk secara otomatis meluncurkan atau menghentikan instans EC2 Amazon berdasarkan pemicu yang ditentukan pengguna. Anda dapat menyiapkan Grup Auto Scaling dan kaitkan pemicu dengan grup-grup ini untuk secara otomatis mengukur sumber daya komputasi berdasarkan metrik, seperti penggunaan bandwidth atau utilisasi CPU. EC2 Auto Scaling Amazon berfungsi dengan Amazon CloudWatch untuk mengambil metrik instance server yang menjalankan aplikasi Anda.

EC2 Auto Scaling Amazon memungkinkan Anda mengambil grup EC2 instans Amazon dan mengatur berbagai parameter agar grup ini bertambah atau berkurang jumlahnya secara otomatis. EC2 Auto Scaling Amazon dapat menambah atau menghapus EC2 instans Amazon dari grup tersebut untuk membantu Anda menangani perubahan lalu lintas ke aplikasi dengan mulus.

Amazon EC2 Auto Scaling juga memantau kesehatan setiap EC2 instans Amazon yang diluncurkan. Jika ada instance yang dihentikan secara tidak terduga, Amazon EC2 Auto Scaling mendeteksi penghentian dan meluncurkan instance pengganti. Kemampuan ini memungkinkan Anda mempertahankan jumlah EC2 instans Amazon yang tetap dan diinginkan secara otomatis.

Elastic Beanstalk menyediakan EC2 Amazon Auto Scaling untuk aplikasi Anda. Anda dapat mengedit konfigurasi instans Amazon lingkungan EC2 Elastic Beanstalk dengan tab Auto Scaling di dalam tab lingkungan aplikasi Anda di Toolkit AWS for Visual Studio.

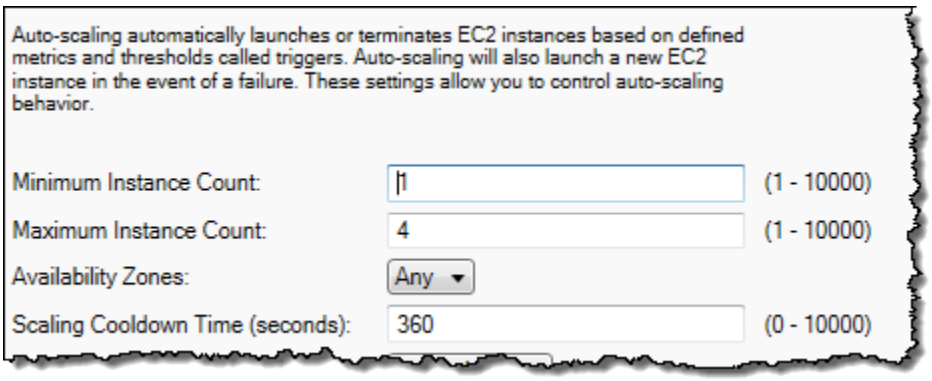


Bagian berikut membahas cara mengonfigurasi parameter Auto Scaling untuk aplikasi Anda.

Meluncurkan konfigurasi

Anda dapat mengedit konfigurasi peluncuran untuk mengontrol bagaimana aplikasi Elastic Beanstalk Anda menyediakan sumber daya EC2 Amazon Auto Scaling.

Kotak Jumlah Instans Minimum dan Jumlah Instans Maksimum memungkinkan Anda menentukan ukuran minimum dan maksimum grup Auto Scaling yang digunakan aplikasi Elastic Beanstalk Anda.



> **Note**
>
> Untuk mempertahankan jumlah EC2 instans Amazon yang tetap, setel Jumlah Instans Minimum dan Hitungan Instance Maksimum ke nilai yang sama.

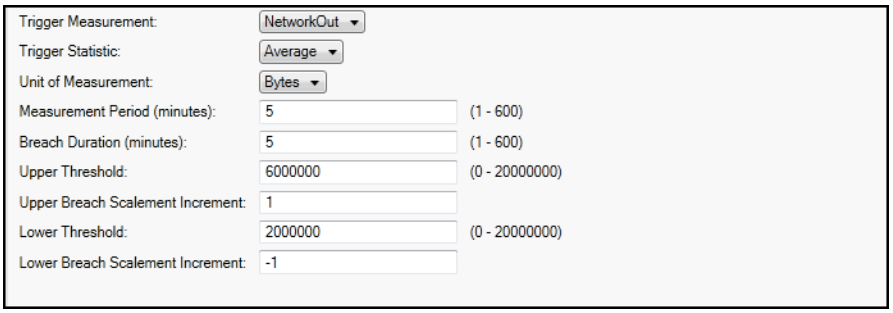
Kotak Availability Zones memungkinkan Anda menentukan jumlah Availability Zone yang Anda inginkan untuk EC2 instans Amazon. Hal ini penting untuk mengatur jumlah ini jika Anda ingin membangun aplikasi yang toleran terhadap kesalahan. Jika satu Availability Zone bermasalah, instans Anda masih akan berjalan di Availability Zone lainnya.

> **Note**
>
> Saat ini, tidak mungkin untuk menentukan Availability Zone mana instans Anda akan berada.

Pemicu

Pemicu adalah mekanisme EC2 Auto Scaling Amazon yang Anda atur untuk memberi tahu sistem saat Anda ingin menambah (meningkatkan skala) atau mengurangi (menskalakan) jumlah instance. Anda dapat mengonfigurasi pemicu untuk diaktifkan pada metrik apa pun yang dipublikasikan ke Amazon CloudWatch (misalnya, pemanfaatan CPU) dan menentukan apakah kondisi yang Anda tentukan telah terpenuhi. Bila ambang batas atas atau bawah dari kondisi yang telah Anda tentukan untuk metrik telah dilanggar selama jangka waktu tertentu, pemicu akan meluncurkan proses yang berjalan lama yang disebut Aktivitas Penskalaan.

Anda dapat menentukan pemicu penskalaan untuk aplikasi Elastic Beanstalk Anda menggunakan Toolkit AWS for Visual Studio.



Pemicu EC2 Auto Scaling Amazon berfungsi dengan memantau CloudWatch metrik Amazon tertentu dari instance tertentu. Metrik mencakup penggunaan CPU, lalu lintas jaringan, dan aktivitas disk. Gunakan pengaturan Pengukuran Pemicu untuk memilih metrik pemicu Anda.

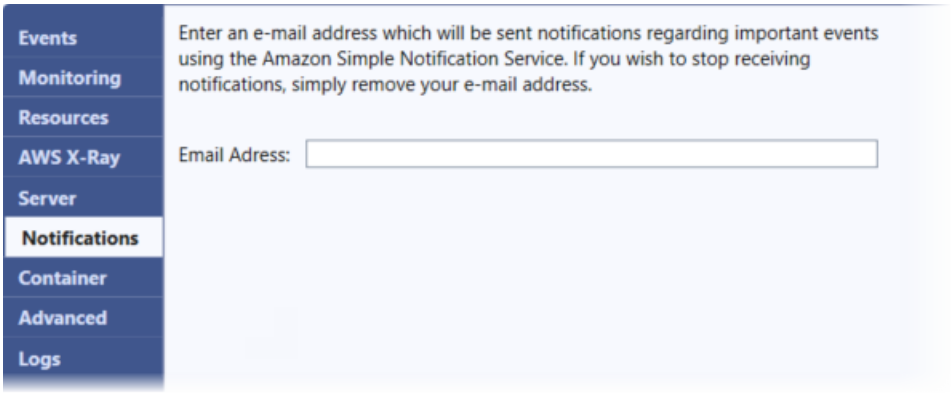
Daftar berikut menjelaskan parameter pemicu yang dapat Anda konfigurasikan menggunakan AWS Management Console.

- Anda dapat menentukan statistik pemicu mana yang harus digunakan. Anda dapat memilih Minimum, Maksimum, Jumlah, atau Rata-rata untuk Statistik Pemicu.
- Untuk Unit Pengukuran, tentukan unit untuk pengukuran pemicu.
- Nilai dalam kotak Periode Pengukuran menentukan seberapa sering Amazon CloudWatch mengukur metrik untuk pemicu Anda. Durasi Pelanggaran adalah jumlah waktu metrik dapat melampaui batas yang ditetapkan (sebagaimana ditentukan untuk Ambang Batas Atas dan Ambang Batas Bawah) sebelum pemicu diaktifkan.
- Untuk Peningkatan Skala Pelanggaran Atas dan Peningkatan Skala Pelanggaran Bawah, tentukan jumlah EC2 instans Amazon yang akan ditambahkan atau dihapus saat melakukan aktivitas penskalaan.

Untuk informasi selengkapnya tentang EC2 Auto Scaling Amazon, lihat bagian Auto EC2 Scaling Amazon di Amazon Elastic Compute Cloud Documentation.

Mengkonfigurasi notifikasi menggunakan AWS toolkit untuk Visual Studio

Elastic Beanstalk menggunakan Amazon Simple Notification Service (Amazon SNS) untuk memberitahu Anda tentang peristiwa penting yang mempengaruhi aplikasi Anda. Untuk mengaktifkan notifikasi Amazon SNS, masukkan alamat email Anda di kotak Alamat Email. Untuk menonaktifkan notifikasi ini, hapus alamat email Anda dari kotak.

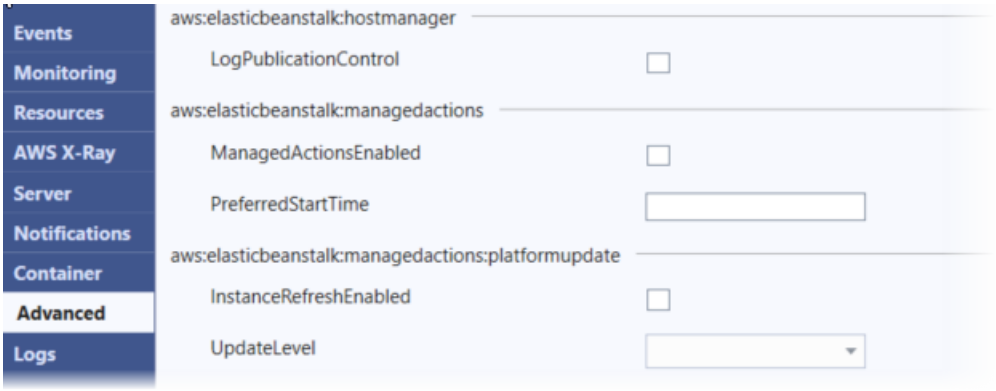


Mengkonfigurasi opsi lingkungan tambahan menggunakan AWS toolkit untuk Visual Studio

Elastic Beanstalk menentukan sejumlah besar opsi konfigurasi yang dapat Anda gunakan untuk mengonfigurasi perilaku lingkungan Anda dan sumber daya yang ada di dalamnya. Opsi konfigurasi

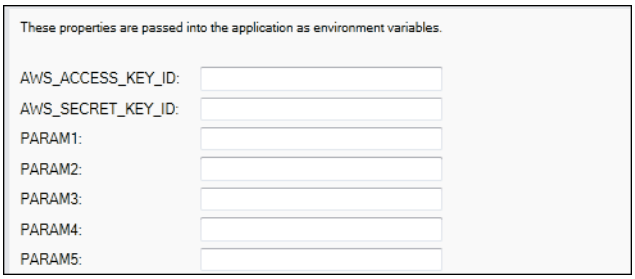
diatur ke dalam namespace, seperti `aws:autoscaling:asg`. Setiap namespace menentukan opsi untuk grup Auto Scaling lingkungan. Panel Lanjutan mencantumkan namespace opsi konfigurasi dalam urutan abjad yang dapat Anda perbarui setelah pembuatan lingkungan.

Untuk daftar lengkap namespace dan opsi termasuk nilai default dan yang didukung untuk masing-masing, lihat Opsi umum untuk semua lingkungan dan .NET Core pada opsi platform Linux.



Mengkonfigurasi kontainer.NET Core menggunakan AWS toolkit untuk Visual Studio

Panel Kontainer memungkinkan Anda menentukan variabel lingkungan yang dapat Anda baca dari kode aplikasi Anda.



Pemantauan kondisi aplikasi

Topik ini menjelaskan cara memantau kesehatan situs web aplikasi Anda. Penting untuk mengetahui bahwa situs web produksi Anda tersedia dan menanggapi permintaan. Elastic Beanstalk menyediakan fitur untuk membantu Anda memantau responsivitas aplikasi Anda. Ini memantau statistik tentang aplikasi Anda dan memberitahu Anda ketika ambang batas terlampaui.

Untuk informasi tentang pemantauan kondisi yang disediakan oleh Elastic Beanstalk, lihat Pelaporan kondisi dasar.

Anda dapat mengakses informasi operasional tentang aplikasi Anda dengan menggunakan AWS Toolkit for Visual Studio atau Management AWS Console.

Kit alat menampilkan status lingkungan dan kondisi aplikasi Anda di bidang Status.

Untuk memantau kondisi aplikasi

1. Di AWS Toolkit for Visual Studio, AWS di Explorer, perluas node Elastic Beanstalk, dan kemudian perluas node aplikasi Anda.
2. Buka menu konteks (klik kanan) untuk lingkungan aplikasi Anda dan pilih Lihat Status.
3. Di tab lingkungan aplikasi Anda, pilih Pemantauan.

   Panel Pemantauan mencakup serangkaian grafik yang menunjukkan penggunaan sumber daya untuk lingkungan aplikasi tertentu Anda.

   > **Note**
   >
   > Secara default, rentang waktu diatur ke jam terakhir. Untuk mengubah pengaturan ini, di daftar Rentang Waktu, pilih rentang waktu yang berbeda.

Anda dapat menggunakan AWS Toolkit for Visual Studio atau AWS Management Console untuk melihat peristiwa yang terkait dengan aplikasi Anda.

Untuk melihat peristiwa aplikasi

1. Di AWS Toolkit for Visual Studio, AWS di Explorer, perluas node Elastic Beanstalk dan node aplikasi Anda.
2. Buka menu konteks (klik kanan) untuk lingkungan aplikasi Anda dan pilih Lihat Status.
3. Di tab lingkungan aplikasi Anda, pilih Peristiwa.

Migrasi dari.NET pada platform Windows Server ke .NET Core pada platform Linux di Elastic Beanstalk

Anda dapat memigrasi aplikasi yang berjalan di platform .NET di Windows Server ke .NET Core di platform Linux. Berikut ini adalah beberapa pertimbangan saat bermigrasi dari Windows ke platform Linux.

Pertimbangan untuk migrasi ke .NET Core di platform Linux

Luas	Perubahan dan informasi
Konfigurasi aplikasi	Di platform Windows, Anda menggunakan manifes deployment untuk menentukan aplikasi yang berjalan di lingkungan Anda. .NET Core di platform Linux menggunak an Procfile untuk menentukan aplikasi yang berjalan di instans lingkungan Anda. Untuk detail tentang paketan aplikasi, lihat the section called “Aplikasi paketan”.
Server proksi	Di platform Windows, Anda menggunakan IIS sebagai server proksi aplikasi Anda. .NET Core di platform Linux mencakup nginx sebagai proksi terbalik secara default. Anda dapat memilih untuk tidak menggunakan server proksi dan menggunakan Kestrel sebagai server web Anda. Untuk pelajari selengkapnya, lihat the section called “Server proksi”.
Perutean	Di platform Windows, Anda menggunakan IIS di kode aplikasi Anda dan sertakan manifes deployment untuk mengonfigurasi jalur IIS. Untuk .NET Core di platform Linux, Anda menggunakan Perutean ASP.NET Core di kode aplikasi Anda, dan perbarui konfigurasi nginx lingkungan Anda. Untuk pelajari selengkapnya, lihat the section called “Server proksi”.
Log	Platform Linux dan Windows mengalirkan log yang berbeda. Untuk detailnya, lihat the section called “Bagaimana Elastic Beanstalk mengatur Log CloudWatch ”.

Menyebarkan aplikasi Go dengan Elastic Beanstalk

Bab ini memberikan instruksi untuk mengonfigurasi dan menerapkan aplikasi web Go Anda ke. AWS Elastic Beanstalk Elastic Beanstalk memudahkan penerapan, pengelolaan, dan skala aplikasi web Go Anda menggunakan Amazon Web Services.

Anda dapat menerapkan aplikasi Anda hanya dalam beberapa menit menggunakan Elastic Beanstalk Command Line Interface (EB CLI) atau dengan menggunakan konsol Elastic Beanstalk. Setelah Anda menerapkan aplikasi Elastic Beanstalk Anda, Anda dapat terus menggunakan EB CLI untuk mengelola aplikasi dan lingkungan Anda, atau Anda dapat menggunakan konsol Elastic Beanstalk,, atau. AWS CLI APIs

Ikuti step-by-step petunjuk QuickStart untuk Go untuk membuat dan menyebarkan aplikasi web Hello World Go dengan EB CLI.

Topik

- QuickStart: Menyebarkan aplikasi Go ke Elastic Beanstalk
- Menyiapkan lingkungan pengembangan Go Anda untuk Elastic Beanstalk
- Menggunakan platform Go Elastic Beanstalk

QuickStart: Menyebarkan aplikasi Go ke Elastic Beanstalk

QuickStart Tutorial ini memandu Anda melalui proses pembuatan aplikasi Go dan menyebarkannya ke AWS Elastic Beanstalk lingkungan.

> **Note**
>
> Contoh tutorial dimaksudkan untuk demonstrasi. Jangan gunakan aplikasi untuk lalu lintas produksi.

Bagian-bagian

- AWS Akun Anda
- Prasyarat
- Langkah 1: Buat aplikasi Go

- Langkah 2: Terapkan aplikasi Go Anda dengan EB CLI
- Langkah 3: Jalankan aplikasi Anda di Elastic Beanstalk
- Langkah 4: Membersihkan
- AWS sumber daya untuk aplikasi Anda
- Langkah selanjutnya
- Terapkan dengan konsol Elastic Beanstalk

AWS Akun Anda

Jika Anda belum menjadi AWS pelanggan, Anda perlu membuat AWS akun. Mendaftar memungkinkan Anda mengakses Elastic Beanstalk AWS dan layanan lain yang Anda butuhkan.

Jika Anda sudah memiliki AWS akun, Anda dapat melanjutkan kePrasyarat.

Buat AWS akun

Mendaftar untuk Akun AWS

Jika Anda tidak memiliki Akun AWS, selesaikan langkah-langkah berikut untuk membuatnya.

Untuk mendaftar untuk Akun AWS

1. Buka https://portal.aws.amazon.com/billing/pendaftaran.
2. Ikuti petunjuk online.

   Bagian dari prosedur pendaftaran melibatkan menerima panggilan telepon atau pesan teks dan memasukkan kode verifikasi pada keypad telepon.

   Saat Anda mendaftar untuk sebuah Akun AWS, sebuah Pengguna root akun AWSdibuat. Pengguna root memiliki akses ke semua Layanan AWS dan sumber daya di akun. Sebagai praktik keamanan terbaik, tetapkan akses administratif ke pengguna, dan gunakan hanya pengguna root untuk melakukan tugas yang memerlukan akses pengguna root.

AWS mengirimi Anda email konfirmasi setelah proses pendaftaran selesai. Kapan saja, Anda dapat melihat aktivitas akun Anda saat ini dan mengelola akun Anda dengan masuk https://aws.amazon.comke/ dan memilih Akun Saya.

Buat pengguna dengan akses administratif

Setelah Anda mendaftar Akun AWS, amankan Pengguna root akun AWS, aktifkan AWS IAM Identity Center, dan buat pengguna administratif sehingga Anda tidak menggunakan pengguna root untuk tugas sehari-hari.

Amankan Anda Pengguna root akun AWS

1. Masuk ke AWS Management Consolesebagai pemilik akun dengan memilih pengguna Root dan memasukkan alamat Akun AWS email Anda. Di laman berikutnya, masukkan kata sandi.

   Untuk bantuan masuk dengan menggunakan pengguna root, lihat Masuk sebagai pengguna root di AWS Sign-In Panduan Pengguna.
2. Mengaktifkan autentikasi multi-faktor (MFA) untuk pengguna root Anda.

   Untuk petunjuk, lihat Mengaktifkan perangkat MFA virtual untuk pengguna Akun AWS root (konsol) Anda di Panduan Pengguna IAM.

Buat pengguna dengan akses administratif

1. Aktifkan Pusat Identitas IAM.

   Untuk mendapatkan petunjuk, silakan lihat Mengaktifkan AWS IAM Identity Center di Panduan Pengguna AWS IAM Identity Center .
2. Di Pusat Identitas IAM, berikan akses administratif ke pengguna.

   Untuk tutorial tentang menggunakan Direktori Pusat Identitas IAM sebagai sumber identitas Anda, lihat Mengkonfigurasi akses pengguna dengan default Direktori Pusat Identitas IAM di Panduan AWS IAM Identity Center Pengguna.

Masuk sebagai pengguna dengan akses administratif

- Untuk masuk dengan pengguna Pusat Identitas IAM, gunakan URL masuk yang dikirim ke alamat email saat Anda membuat pengguna Pusat Identitas IAM.

  Untuk bantuan masuk menggunakan pengguna Pusat Identitas IAM, lihat Masuk ke portal AWS akses di Panduan AWS Sign-In Pengguna.

Tetapkan akses ke pengguna tambahan

1. Di Pusat Identitas IAM, buat set izin yang mengikuti praktik terbaik menerapkan izin hak istimewa paling sedikit.

   Untuk petunjuknya, lihat Membuat set izin di Panduan AWS IAM Identity Center Pengguna.
2. Tetapkan pengguna ke grup, lalu tetapkan akses masuk tunggal ke grup.

   Untuk petunjuk, lihat Menambahkan grup di Panduan AWS IAM Identity Center Pengguna.

Prasyarat

Untuk mengikuti prosedur di panduan ini, Anda memerlukan terminal atau shell baris perintah untuk menjalankan perintah. Perintah ditampilkan dalam daftar yang diawali dengan simbol prompt ($) dan nama direktori saat ini, bila perlu.

```
~/eb-project$ this is a command
this is output
```

Di Linux dan macOS, Anda dapat menggunakan shell dan manajer paket pilihan Anda. Pada Windows Anda dapat menginstal Windows Subsystem untuk Linux untuk mendapatkan versi Windows terintegrasi dari Ubuntu dan Bash.

EB CLI

Tutorial ini menggunakan Elastic Beanstalk Command Line Interface (EB CLI). Untuk detail tentang pemasangan dan konfigurasi EB CLI, lihat Instal EB CLI dengan skrip pengaturan (disarankan) dan Mengonfigurasi EB CLI.

Langkah 1: Buat aplikasi Go

Buat direktori proyek.

```
~$ mkdir eb-go
~$ cd eb-go
```

Selanjutnya, buat aplikasi yang akan Anda deploy menggunakan Elastic Beanstalk. Kami akan membuat layanan RESTful web “Hello World”.

Contoh ini membuat penyesuaian salam pembuka yang bervariasi berdasarkan jalur yang digunakan untuk mengakses layanan.

Buat file teks di direktori ini bernama `application.go` dengan konten berikut.

Example **`~/eb-go/application.go`**

```
package main

import (
 "fmt"
 "net/http"
)

func handler(w http.ResponseWriter, r *http.Request) {
 if r.URL.Path == "/" {
  fmt.Fprintf(w, "Hello World! Append a name to the URL to say hello. For example, use
 %s/Mary to say hello to Mary.", r.Host)
 } else {
  fmt.Fprintf(w, "Hello, %s!", r.URL.Path[1:])
 }
}

func main() {
 http.HandleFunc("/", handler)
 http.ListenAndServe(":5000", nil)
}
```

Langkah 2: Terapkan aplikasi Go Anda dengan EB CLI

Selanjutnya, Anda membuat lingkungan aplikasi Anda dan men-deploy aplikasi yang dikonfigurasi dengan Elastic Beanstalk.

Untuk membuat lingkungan baru dan men-deploy aplikasi Go Anda

1. Inisialisasi repositori EB CLI Anda dengan perintah eb init.

   ```
   ~/eb-go$ eb init -p go go-tutorial --region us-east-2
   Application go-tutorial has been created.
   ```

Perintah ini membuat aplikasi bernama `go-tutorial` dan mengonfigurasi repositori lokal Anda untuk membuat lingkungan dengan versi platform Go terbaru.

2. (Opsional) Jalankan eb init lagi untuk mengonfigurasi key pair default sehingga Anda dapat menggunakan SSH untuk terhubung ke EC2 instance yang menjalankan aplikasi Anda.

```
~/eb-go$ eb init
Do you want to set up SSH for your instances?
(y/n): y
Select a keypair.
1) my-keypair
2) [ Create new KeyPair ]
```

Pilih pasangan kunci jika Anda sudah memilikinya, atau ikuti prompt untuk membuatnya. Jika Anda tidak melihat prompt atau perlu mengubah pengaturan Anda nanti, jalankan eb init -i.

3. Buat lingkungan baru dan deploy aplikasi Anda dengan eb create. Elastic Beanstalk secara otomatis membuat file zip untuk aplikasi Anda dan memulainya pada port 5000.

```
~/eb-go$ eb create go-env
```

Dibutuhkan sekitar lima menit untuk Elastic Beanstalk untuk menciptakan lingkungan Anda.

Langkah 3: Jalankan aplikasi Anda di Elastic Beanstalk

Ketika proses untuk membuat lingkungan Anda selesai, buka situs web Anda denganeb open.

```
~/eb-go$ eb open
```

Selamat! Anda telah menerapkan aplikasi Go dengan Elastic Beanstalk! Ini akan membuka jendela peramban menggunakan nama domain yang dibuat untuk aplikasi Anda.

Langkah 4: Membersihkan

Anda dapat menghentikan lingkungan Anda ketika Anda selesai bekerja dengan aplikasi Anda. Elastic Beanstalk AWS mengakhiri semua sumber daya yang terkait dengan lingkungan Anda.

Untuk mengakhiri lingkungan Elastic Beanstalk Anda dengan EB CLI jalankan perintah berikut.

```
~/eb-go$ eb terminate
```

AWS sumber daya untuk aplikasi Anda

Anda baru saja membuat aplikasi instance tunggal. Ini berfungsi sebagai aplikasi sampel langsung dengan satu EC2 instance, sehingga tidak memerlukan penyeimbangan beban atau penskalaan otomatis. Untuk aplikasi contoh tunggal Elastic Beanstalk menciptakan sumber daya berikut: AWS

- EC2 instance — Mesin EC2 virtual Amazon yang dikonfigurasi untuk menjalankan aplikasi web pada platform yang Anda pilih.

  Setiap platform menjalankan satu set perangkat lunak, file konfigurasi, dan penulisan yang berbeda untuk mendukung versi bahasa, kerangka kerja, kontainer web tertentu, atau kombinasi dari semua ini. Sebagian besar platform menggunakan Apache atau nginx sebagai proksi terbalik yang memproses lalu lintas web di depan aplikasi web Anda, meneruskan permintaan ke aplikasi web, menyajikan aset statis, dan menghasilkan log akses dan kesalahan.
- Grup keamanan instans — Grup EC2 keamanan Amazon yang dikonfigurasi untuk memungkinkan lalu lintas masuk di port 80. Sumber daya ini memungkinkan lalu lintas HTTP dari penyeimbang beban mencapai EC2 instance yang menjalankan aplikasi web Anda. Secara default, lalu lintas tidak diizinkan di port lain.
- Bucket Amazon S3 – Lokasi penyimpanan untuk kode sumber, log, dan artifact lainnya yang dibuat saat Anda menggunakan Elastic Beanstalk.
- CloudWatch Alarm Amazon — Dua CloudWatch alarm yang memantau beban pada instans di lingkungan Anda dan dipicu jika beban terlalu tinggi atau terlalu rendah. Saat alarm terpicu, grup Auto Scaling Anda akan menaikkan atau menurunkan skala sebagai respons.
- AWS CloudFormation stack - Elastic AWS CloudFormation Beanstalk digunakan untuk meluncurkan sumber daya di lingkungan Anda dan menyebarkan perubahan konfigurasi. Sumber daya ditentukan di sebuah templat yang dapat Anda lihat di Konsol AWS CloudFormation.
- Nama domain — Nama domain yang merutekan ke aplikasi web Anda dalam formulir *subdomain*.*region*.elasticbeanstalk.com.

Elastic Beanstalk mengelola semua sumber daya tersebut. Ketika Anda mengakhiri lingkungan, Elastic Beanstalk mengakhiri semua sumber daya yang dimuatnya.

Langkah selanjutnya

Setelah lingkungan Anda menjalankan aplikasi, Anda dapat men-deploy versi baru aplikasi atau aplikasi yang berbeda kapan saja. Menyebarkan versi aplikasi baru sangat cepat karena tidak memerlukan penyediaan atau memulai ulang instance. EC2 Anda juga dapat menjelajahi lingkungan baru Anda menggunakan konsol Elastic Beanstalk. Untuk langkah-langkah mendetail, lihat Menjelajahi lingkungan Anda di bagian Memulai panduan ini.

Setelah Anda men-deploy satu atau dua aplikasi sampel dan siap untuk mulai mengembangkan dan menjalankan aplikasi Go secara lokal, lihat Menyiapkan lingkungan pengembangan Go Anda untuk Elastic Beanstalk.

Terapkan dengan konsol Elastic Beanstalk

Anda juga dapat menggunakan konsol Elastic Beanstalk untuk meluncurkan aplikasi sampel. Untuk langkah-langkah rinci, lihat Membuat aplikasi contoh di Bab Memulai panduan ini.

Menyiapkan lingkungan pengembangan Go Anda untuk Elastic Beanstalk

Topik ini memberikan instruksi untuk menyiapkan lingkungan pengembangan Go untuk menguji aplikasi Anda secara lokal sebelum menerapkannya. AWS Elastic Beanstalk Ini juga merujuk situs web yang menyediakan instruksi instalasi untuk alat yang berguna.

Menginstal Go

Untuk menjalankan aplikasi Go secara lokal, pasang Go. Jika Anda tidak memerlukan versi tertentu, dapatkan versi terbaru yang didukung Elastic Beanstalk. Untuk daftar versi yang didukung, lihat Go di dokumen Platform AWS Elastic Beanstalk .

Unduh Go di https://golang. org/doc/install.

Menginstal AWS SDK for Go

Jika Anda perlu mengelola AWS sumber daya dari dalam aplikasi Anda, instal AWS SDK for Go dengan menggunakan perintah berikut.

```
$ go get github.com/aws/aws-sdk-go
```

Untuk informasi selengkapnya, lihat AWS SDK for Go.

Menggunakan platform Go Elastic Beanstalk

Topik ini menjelaskan cara mengonfigurasi, membangun, dan menjalankan aplikasi Go Anda di Elastic Beanstalk.

AWS Elastic Beanstalk mendukung sejumlah cabang platform untuk berbagai versi bahasa pemrograman Go. Lihat Buka di dokumen AWS Elastic Beanstalk Platform untuk daftar lengkapnya.

Untuk aplikasi Go sederhana, ada dua cara untuk men-deploy aplikasi Anda:

- Sediakan paket sumber dengan file sumber di akar bernama `application.go` yang berisi paket utama untuk aplikasi Anda. Elastic Beanstalk membangun biner menggunakan perintah berikut:

  ```
  go build -o bin/application application.go
  ```

  Setelah aplikasi dibangun, Elastic Beanstalk memulainya di port 5000.
- Sediakan paket sumber dengan file biner bernama `application`. File biner dapat ditemukan baik di akar paket sumber atau di direktori `bin/` dari paket sumber. Jika Anda menempatkan file biner `application` di kedua lokasi, Elastic Beanstalk menggunakan file di direktori `bin/`.

  Elastic Beanstalk meluncurkan aplikasi ini di port 5000.

Dalam kedua kasus tersebut, dengan cabang platform Go kami yang didukung, Anda juga dapat memberikan persyaratan modul dalam file yang disebut`go.mod`. Untuk informasi selengkapnya, lihat Migrasi ke Modul Go di blog Go.

Untuk aplikasi Go yang lebih kompleks, ada dua cara untuk men-deploy aplikasi Anda:

- Sediakan paket sumber yang mencakup file sumber aplikasi Anda, bersama dengan Buildfile dan Procfile. Buildfile mencakup perintah untuk membangun aplikasi, dan Procfile mencakup petunjuk untuk menjalankan aplikasi.
- Sediakan paket sumber yang mencakup file biner aplikasi Anda, bersama dengan Procfile. Procfile mencakup petunjuk untuk menjalankan aplikasi.

Platform Go mencakup server proksi untuk melayani aset statis dan meneruskan lalu lintas ke aplikasi Anda. Anda dapat memperpanjang atau mengganti konfigurasi proksi default untuk skenario lanjutan.

Untuk detail tentang berbagai cara untuk memperluas platform berbasis Linux Elastic Beanstalk, lihat the section called “Memperluas platform Linux”.

Mengonfigurasi lingkungan Go Anda

Pengaturan platform Go memungkinkan Anda menyempurnakan perilaku instans Amazon EC2 Anda. Anda dapat mengedit konfigurasi instans Amazon lingkungan Elastic Beanstalk menggunakan konsol EC2 Elastic Beanstalk.

Gunakan konsol Elastic Beanstalk untuk mengaktifkan rotasi log ke Amazon S3 dan mengonfigurasi variabel yang dapat dibaca aplikasi Anda dari lingkungan.

Untuk mengonfigurasi lingkungan Go Anda di konsol Elastic Beanstalk

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Dalam kategori konfigurasi Pembaruan, pemantauan, dan pencatatan, pilih Edit.

Opsi log

Bagian Opsi Log memiliki dua pengaturan:

- Profil instans – Menentukan profil instans yang memiliki izin untuk mengakses bucket Amazon S3 yang terkait dengan aplikasi Anda.
- Aktifkan rotasi file log ke Amazon S3 — Menentukan apakah file log untuk EC2 instans Amazon aplikasi Anda disalin ke bucket Amazon S3 yang terkait dengan aplikasi Anda.

File statis

Untuk meningkatkan kinerja, Anda dapat menggunakan bagian File statis untuk mengkonfigurasi server proxy untuk melayani file statis (misalnya, HTML atau gambar) dari satu set direktori di dalam aplikasi web Anda. Untuk setiap direktori, Anda mengatur jalur virtual ke pemetaan direktori. Saat

server proksi menerima permintaan untuk file di jalur yang ditentukan, server langsung menyajikan file daripada merutekan permintaan ke aplikasi Anda.

Untuk detail tentang mengonfigurasi file statis menggunakan file konfigurasi atau konsol Elastic Beanstalk, lihat. the section called “File statis”

Properti lingkungan

Bagian Properti Lingkungan memungkinkan Anda menentukan pengaturan konfigurasi lingkungan pada EC2 instance Amazon yang menjalankan aplikasi Anda. Properti lingkungan diberikan sebagai pasangan nilai kunci ke aplikasi.

Di dalam lingkungan Go yang berjalan di Elastic Beanstalk, variabel lingkungan dapat diakses menggunakan fungsi `os.Getenv`. Sebagai contoh, Anda dapat membaca properti bernama `API_ENDPOINT` ke variabel dengan kode berikut:

```
endpoint := os.Getenv("API_ENDPOINT")
```

Lihat Variabel lingkungan dan pengaturan perangkat lunak lainnya untuk informasi selengkapnya.

Namespace konfigurasi Go

Anda dapat menggunakan file konfigurasi untuk mengatur opsi konfigurasi dan melakukan tugas-tugas konfigurasi instans lain selama deployment. Opsi konfigurasi dapat berupa platform khusus atau berlaku untuk semua platform di layanan Elastic Beanstalk secara keseluruhan. Opsi konfigurasi diatur ke dalam ruang nama.

Platform Go tidak menentukan namespace khusus platform. Anda dapat mengonfigurasi proksi untuk menyajikan file statis dengan menggunakan namespace `aws:elasticbeanstalk:environment:proxy:staticfiles`. Untuk detail dan contoh, lihat the section called “File statis”.

Elastic Beanstalk memberikan banyak opsi konfigurasi untuk menyesuaikan lingkungan Anda. Selain file konfigurasi, Anda juga dapat mengatur opsi konfigurasi menggunakan konsol tersebut, konfigurasi tersimpan, EB CLI, atau AWS CLI. Lihat Opsi konfigurasi untuk informasi selengkapnya.

Platform Go Amazon Linux AMI (Amazon Linux 2 terdahulu)

Jika lingkungan Go Elastic Beanstalk Anda menggunakan versi platform Amazon Linux AMI (Amazon Linux 2 terdahulu), baca informasi tambahan di bagian ini.

> **Catatan**
>
> - Informasi dalam topik ini hanya berlaku untuk cabang platform berbasis Amazon Linux AMI (AL1). AL2023/ cabang AL2 platform tidak kompatibel dengan versi platform Amazon Linux AMI (AL1) sebelumnya dan memerlukan pengaturan konfigurasi yang berbeda.
> - Pada tanggal 18 Juli 2022, Elastic Beanstalk menetapkan status semua cabang platform berdasarkan Amazon Linux AMI () untuk pensiun. AL1 Untuk informasi selengkapnya tentang migrasi ke cabang platform Amazon Linux 2023 saat ini dan didukung penuh, lihat. Memigrasi aplikasi Elastic Beanstalk Linux Anda ke Amazon Linux 2023 atau Amazon Linux 2

Ruang nama konfigurasi Go - Amazon Linux AMI () AL1

Anda dapat menggunakan file konfigurasi untuk mengatur opsi konfigurasi dan melakukan tugas-tugas konfigurasi instans lain selama deployment. Opsi konfigurasi dapat berupa platform khusus atau berlaku untuk semua platform di layanan Elastic Beanstalk secara keseluruhan. Opsi konfigurasi diatur ke dalam ruang nama.

> **Note**
>
> Informasi dalam topik ini hanya berlaku untuk cabang platform berbasis Amazon Linux AMI (AL1). AL2023/ cabang AL2 platform tidak kompatibel dengan versi platform Amazon Linux AMI (AL1) sebelumnya dan memerlukan pengaturan konfigurasi yang berbeda.

Platform Go Amazon Linux AMI mendukung satu namespace konfigurasi khusus platform selain namespace yang didukung oleh semua platform. Namespace `aws:elasticbeanstalk:container:golang:staticfiles` memungkinkan Anda menentukan opsi yang memetakan jalur di aplikasi web Anda ke folder di paket sumber aplikasi Anda yang berisi konten statis.

Sebagai contoh, file konfigurasi ini memberitahu server proksi untuk menyajikan file di folder `staticimages` di jalur `/images`:

Example .ebextensions/go-settings.config

```
option_settings:
```

```
  aws:elasticbeanstalk:container:golang:staticfiles:
    /html: statichtml
    /images: staticimages
```

Elastic Beanstalk memberikan banyak opsi konfigurasi untuk menyesuaikan lingkungan Anda. Selain file konfigurasi, Anda juga dapat mengatur opsi konfigurasi menggunakan konsol tersebut, konfigurasi tersimpan, EB CLI, atau AWS CLI. Untuk informasi selengkapnya, lihat Opsi konfigurasi.

Mengkonfigurasi perintah start khusus dengan Procfile pada Elastic Beanstalk

Untuk menentukan perintah kustom untuk memulai aplikasi Go, sertakan file bernama `Procfile` di akar paket sumber Anda.

Untuk detail tentang menulis dan menggunakan a`Procfile`, lihatBuildfile dan Procfile.

Example Procfile

```
web: bin/server
queue_process: bin/queue_processor
foo: bin/fooapp
```

Anda harus memanggil `web` aplikasi utama, dan daftarkan sebagai perintah pertama di `Procfile`. Elastic Beanstalk mengekspos aplikasi `web` utama di akar URL lingkungan; misalnya, `http://my-go-env.elasticbeanstalk.com`.

Elastic Beanstalk juga menjalankan aplikasi yang namanya tidak memiliki prefiks `web_`, tetapi aplikasi ini tidak tersedia dari luar instans Anda.

Elastic Beanstalk mengharapkan proses berjalan dari `Procfile` agar berjalan secara terus menerus. Elastic Beanstalk memantau aplikasi ini dan memulai ulang setiap proses yang berakhir. Untuk proses yang berjalan singkat, gunakan perintah Buildfile.

Menggunakan Procfile di Amazon Linux AMI (Amazon Linux 2 terdahulu)

Jika lingkungan Go Elastic Beanstalk Anda menggunakan versi platform Amazon Linux AMI (Amazon Linux 2 terdahulu), baca informasi tambahan di bagian ini.

> **Catatan**
>
> - Informasi dalam topik ini hanya berlaku untuk cabang platform berbasis Amazon Linux AMI (AL1). AL2023/ cabang AL2 platform tidak kompatibel dengan versi platform Amazon Linux AMI (AL1) sebelumnya dan memerlukan pengaturan konfigurasi yang berbeda.
> - Pada tanggal 18 Juli 2022, Elastic Beanstalk menetapkan status semua cabang platform berdasarkan Amazon Linux AMI () untuk pensiun. AL1 Untuk informasi selengkapnya tentang migrasi ke cabang platform Amazon Linux 2023 saat ini dan didukung penuh, lihat. Memigrasi aplikasi Elastic Beanstalk Linux Anda ke Amazon Linux 2023 atau Amazon Linux 2

Port lewat - Amazon Linux AMI (AL1)

> **Note**
>
> Informasi dalam topik ini hanya berlaku untuk cabang platform berbasis Amazon Linux AMI (AL1). AL2023/ cabang AL2 platform tidak kompatibel dengan versi platform Amazon Linux AMI (AL1) sebelumnya dan memerlukan pengaturan konfigurasi yang berbeda.

Elastic Beanstalk mengonfigurasi proksi nginx untuk meneruskan permintaan ke aplikasi Anda di nomor port yang ditentukan di `PORT` properti lingkungan untuk aplikasi Anda. Aplikasi Anda harus selalu mendengarkan di port tersebut. Anda dapat mengakses variabel ini dalam aplikasi Anda dengan memanggil metode `os.Getenv("PORT")`.

Elastic Beanstalk menggunakan nomor port yang ditentukan di properti lingkungan `PORT` untuk port untuk aplikasi pertama di `Procfile`, dan kemudian menambahkan nomor port untuk setiap aplikasi berikutnya di `Procfile` sebanyak 100. Jika properti lingkungan `PORT` tidak diatur, Elastic Beanstalk menggunakan 5000 untuk port awal.

Di contoh sebelumnya, properti lingkungan `PORT` untuk aplikasi `web` adalah 5000, aplikasi `queue_process` adalah 5100, dan aplikasi `foo` adalah 5200.

Anda dapat menentukan port awal dengan mengatur opsi `PORT` dengan namespace aws:elasticbeanstalk:aplication:environment, seperti yang ditunjukkan di contoh berikut.

```
option_settings:
```

```
  - namespace:  aws:elasticbeanstalk:application:environment
    option_name:  PORT
    value:  <first_port_number>
```

Untuk informasi selengkapnya tentang pengaturan properti lingkungan untuk aplikasi Anda, lihat Pengaturan opsi.

Kustom build dan konfigurasi dengan Buildfile pada Elastic Beanstalk

Untuk menentukan pembangunan kustom dan perintah konfigurasi untuk aplikasi Go Anda, sertakan file bernama `Buildfile` di akar paket sumber Anda. Nama file peka terhadap huruf besar-kecil. Gunakan format berikut untuk `Buildfile`:

```
<process_name>: <command>
```

Perintah di `Buildfile` Anda harus sesuai dengan ekspresi reguler berikut: `^[A-Za-z0-9_]+:\s*.+$`.

Elastic Beanstalk tidak memantau aplikasi yang dijalankan dengan `Buildfile`. Gunakan `Buildfile` untuk perintah yang berjalan dalam waktu singkat dan berakhir setelah menyelesaikan tugas mereka. Untuk proses aplikasi yang berjalan lama yang seharusnya tidak keluar, gunakan Procfile sebagai gantinya.

Di contoh berikut dari sebuah `Buildfile`, `build.sh` adalah penulisan shell yang terletak di akar paket sumber:

```
make: ./build.sh
```

Semua jalur di `Buildfile` bersifat relatif terhadap akar paket sumber. Jika Anda tahu sebelumnya tempat file berada di instans, Anda dapat menyertakan jalur absolut di `Buildfile`.

Mengonfigurasi server proksi

Elastic Beanstalk menggunakan nginx sebagai proksi terbalik untuk memetakan aplikasi Anda ke penyeimbang beban Elastic Load Balancing di port 80. Elastic Beanstalk menyediakan konfigurasi nginx default yang dapat Anda perluas atau sepenuhnya diganti dengan konfigurasi Anda sendiri.

Secara default, Elastic Beanstalk mengonfigurasi proksi nginx untuk meneruskan permintaan ke aplikasi Anda di port 5000. Anda dapat mengganti port default dengan mengatur `PORT` properti lingkungan ke port yang didengarkan aplikasi utama Anda.

> **Note**
>
> Port yang didengar aplikasi Anda tidak mempengaruhi port yang didengar server nginx untuk menerima permintaan dari penyeimbang beban.

Mengkonfigurasi server proxy pada versi platform Anda

Semua AL2 AL2 023/platform mendukung fitur konfigurasi proxy yang seragam. Untuk informasi selengkapnya tentang mengonfigurasi server proxy pada versi platform Anda yang menjalankan AL2 023/AL2, lihat. Konfigurasi proksi terbalik

Mengonfigurasi proksi di Amazon Linux AMI (Amazon Linux 2 terdahulu)

> **Catatan**
>
> - Informasi dalam topik ini hanya berlaku untuk cabang platform berbasis Amazon Linux AMI (AL1). AL2023/ cabang AL2 platform tidak kompatibel dengan versi platform Amazon Linux AMI (AL1) sebelumnya dan memerlukan pengaturan konfigurasi yang berbeda.
> - Pada tanggal 18 Juli 2022, Elastic Beanstalk menetapkan status semua cabang platform berdasarkan Amazon Linux AMI () untuk pensiun. AL1 Untuk informasi selengkapnya tentang migrasi ke cabang platform Amazon Linux 2023 saat ini dan didukung penuh, lihat. Memigrasi aplikasi Elastic Beanstalk Linux Anda ke Amazon Linux 2023 atau Amazon Linux 2

Jika lingkungan Go Elastic Beanstalk Anda menggunakan versi platform Amazon Linux AMI (Amazon Linux 2 terdahulu), baca informasi di bagian ini.

Memperluas dan mengganti konfigurasi proxy default — Amazon Linux AMI () AL1

Elastic Beanstalk menggunakan nginx sebagai proksi terbalik untuk memetakan aplikasi Anda ke penyeimbang beban di port 80. Jika Anda ingin menyediakan konfigurasi nginx Anda sendiri, Anda dapat mengganti konfigurasi default yang disediakan oleh Elastic Beanstalk dengan menyertakan file `.ebextensions/nginx/nginx.conf` di paket sumber Anda. Jika file tersedia, Elastic Beanstalk menggunakannya sebagai ganti file konfigurasi nginx default.

Jika Anda ingin menyertakan arahan selain yang ada di blok `nginx.conf` `http`, Anda juga dapat menyediakan file konfigurasi tambahan di direktori `.ebextensions/nginx/conf.d/` paket sumber Anda. Semua file di direktori ini harus memiliki ekstensi `.conf`.

Untuk mengambil keuntungan dari fungsionalitas yang disediakan oleh Elastic Beanstalk, seperti Peningkatan pelaporan dan pemantauan kesehatan di Elastic Beanstalk, pemetaan aplikasi otomatis, dan file statis, Anda harus menyertakan baris berikut di blok `server` file konfigurasi nginx Anda:

```
include conf.d/elasticbeanstalk/*.conf;
```

Menyebarkan aplikasi Java dengan Elastic Beanstalk

Bab ini memberikan instruksi untuk mengkonfigurasi dan menyebarkan aplikasi Java Anda ke. AWS Elastic Beanstalk Elastic Beanstalk memudahkan penerapan, pengelolaan, dan skala aplikasi web Java Anda menggunakan Amazon Web Services.

Anda dapat menerapkan aplikasi Anda hanya dalam beberapa menit menggunakan Elastic Beanstalk Command Line Interface (EB CLI) atau dengan menggunakan konsol Elastic Beanstalk. Setelah Anda menerapkan aplikasi Elastic Beanstalk Anda, Anda dapat terus menggunakan EB CLI untuk mengelola aplikasi dan lingkungan Anda, atau Anda dapat menggunakan konsol Elastic Beanstalk,, atau. AWS CLI APIs

Ikuti step-by-step petunjuk QuickStart untuk Java untuk membuat dan menyebarkan aplikasi web Hello World Java dengan EB CLI. Jika Anda tertarik dengan step-by-step instruksi untuk membuat aplikasi Hello World Java JSP sederhana untuk digunakan dengan EB CLI ke platform berbasis Tomcat kami, coba. QuickStart untuk Java di Tomcat

Cabang platform Java

AWS Elastic Beanstalk mendukung dua platform untuk aplikasi Java.

- Tomcat — Platform berbasis Apache Tomcat, wadah web open source untuk aplikasi yang menggunakan servlet Java dan JavaServer Pages (JSPs) untuk melayani permintaan HTTP. Tomcat memfasilitasi pengembangan aplikasi web dengan menyediakan multithreading, konfigurasi keamanan deklaratif, dan penyesuaian yang luas. Elastic Beanstalk memiliki cabang platform untuk masing-masing versi utama Tomcat saat ini. Untuk informasi selengkapnya, lihat Platform Tomcat.
- Java SE — Platform untuk aplikasi yang tidak menggunakan wadah web, atau menggunakan yang lain selain Tomcat, seperti Jetty atau. GlassFish Anda dapat menyertakan pustaka Java Archives (JARs) yang digunakan oleh aplikasi Anda dalam bundel sumber yang Anda gunakan ke Elastic Beanstalk. Untuk informasi selengkapnya, lihat Platform Java SE.

Cabang terbaru dari platform Tomcat dan Java SE didasarkan pada Amazon Linux 2 dan yang lebih baru, dan menggunakan Corretto AWS —distribusi Java SE. Nama-nama cabang platform ini termasuk kata Corretto, bukan Java.

Untuk daftar versi platform saat ini, lihat Tomcat dan Java SE di panduan Platform AWS Elastic Beanstalk .

AWS alat

AWS menyediakan beberapa alat untuk bekerja dengan Java dan Elastic Beanstalk. Terlepas dari cabang platform yang Anda pilih, Anda dapat menggunakan AWS SDK for Java untuk menggunakan layanan AWS lain dari dalam aplikasi Java Anda. AWS SDK for Java adalah sekumpulan pustaka yang memungkinkan Anda untuk AWS APIs menggunakan dari kode aplikasi Anda tanpa menulis panggilan HTTP mentah dari awal.

Jika Anda lebih suka mengelola aplikasi Anda dari baris perintah, instal Elastic Beanstalk Command Line Interface (EB CLI) dan gunakan untuk membuat, memantau, dan mengelola lingkungan Elastic Beanstalk Anda. Jika Anda menjalankan beberapa lingkungan untuk aplikasi Anda, EB CLI terintegrasi dengan Git untuk membiarkan Anda mengaitkan masing-masing lingkungan Anda dengan cabang Git yang berbeda.

Topik

- QuickStart: Menyebarkan aplikasi Java ke Elastic Beanstalk
- QuickStart: Menyebarkan aplikasi web Java JSP untuk Tomcat ke Elastic Beanstalk
- Menyiapkan lingkungan pengembangan Java Anda
- Lebih banyak contoh aplikasi dan tutorial Elastic Beanstalk untuk Java
- Menggunakan platform Elastic Beanstalk Tomcat
- Menggunakan platform Java SE Elastic Beanstalk
- Menambahkan instans Amazon RDS DB ke lingkungan Java Elastic Beanstalk Anda
- Alat dan sumber daya Java

QuickStart: Menyebarkan aplikasi Java ke Elastic Beanstalk

QuickStart Tutorial ini memandu Anda melalui proses pembuatan aplikasi Java dan menyebarkannya ke AWS Elastic Beanstalk lingkungan.

> **Note**
>
> Contoh tutorial dimaksudkan untuk demonstrasi. Jangan gunakan aplikasi untuk lalu lintas produksi.

Bagian-bagian

- AWS Akun Anda
- Prasyarat
- Langkah 1: Buat aplikasi Java
- Langkah 2: Jalankan aplikasi Anda secara lokal
- Langkah 3: Menyebarkan aplikasi Java Anda dengan EB CLI
- Langkah 4: Jalankan aplikasi Anda di Elastic Beanstalk
- Langkah 5: Bersihkan
- AWS sumber daya untuk aplikasi Anda
- Langkah selanjutnya
- Terapkan dengan konsol Elastic Beanstalk

AWS Akun Anda

Jika Anda belum menjadi AWS pelanggan, Anda perlu membuat AWS akun. Mendaftar memungkinkan Anda mengakses Elastic Beanstalk AWS dan layanan lain yang Anda butuhkan.

Jika Anda sudah memiliki AWS akun, Anda dapat melanjutkan kePrasyarat.

Buat AWS akun

Mendaftar untuk Akun AWS

Jika Anda tidak memiliki Akun AWS, selesaikan langkah-langkah berikut untuk membuatnya.

Untuk mendaftar untuk Akun AWS

1. Buka https://portal.aws.amazon.com/billing/pendaftaran.
2. Ikuti petunjuk online.

   Bagian dari prosedur pendaftaran melibatkan menerima panggilan telepon atau pesan teks dan memasukkan kode verifikasi pada keypad telepon.

   Saat Anda mendaftar untuk sebuah Akun AWS, sebuah Pengguna root akun AWSdibuat. Pengguna root memiliki akses ke semua Layanan AWS dan sumber daya di akun. Sebagai praktik keamanan terbaik, tetapkan akses administratif ke pengguna, dan gunakan hanya pengguna root untuk melakukan tugas yang memerlukan akses pengguna root.

AWS mengirimi Anda email konfirmasi setelah proses pendaftaran selesai. Kapan saja, Anda dapat melihat aktivitas akun Anda saat ini dan mengelola akun Anda dengan masuk https://aws.amazon.comke/ dan memilih Akun Saya.

Buat pengguna dengan akses administratif

Setelah Anda mendaftar Akun AWS, amankan Pengguna root akun AWS, aktifkan AWS IAM Identity Center, dan buat pengguna administratif sehingga Anda tidak menggunakan pengguna root untuk tugas sehari-hari.

Amankan Anda Pengguna root akun AWS

1. Masuk ke AWS Management Consolesebagai pemilik akun dengan memilih pengguna Root dan memasukkan alamat Akun AWS email Anda. Di laman berikutnya, masukkan kata sandi.

   Untuk bantuan masuk dengan menggunakan pengguna root, lihat Masuk sebagai pengguna root di AWS Sign-In Panduan Pengguna.
2. Mengaktifkan autentikasi multi-faktor (MFA) untuk pengguna root Anda.

   Untuk petunjuk, lihat Mengaktifkan perangkat MFA virtual untuk pengguna Akun AWS root (konsol) Anda di Panduan Pengguna IAM.

Buat pengguna dengan akses administratif

1. Aktifkan Pusat Identitas IAM.

   Untuk mendapatkan petunjuk, silakan lihat Mengaktifkan AWS IAM Identity Center di Panduan Pengguna AWS IAM Identity Center .
2. Di Pusat Identitas IAM, berikan akses administratif ke pengguna.

   Untuk tutorial tentang menggunakan Direktori Pusat Identitas IAM sebagai sumber identitas Anda, lihat Mengkonfigurasi akses pengguna dengan default Direktori Pusat Identitas IAM di Panduan AWS IAM Identity Center Pengguna.

Masuk sebagai pengguna dengan akses administratif

- Untuk masuk dengan pengguna Pusat Identitas IAM, gunakan URL masuk yang dikirim ke alamat email saat Anda membuat pengguna Pusat Identitas IAM.

Untuk bantuan masuk menggunakan pengguna Pusat Identitas IAM, lihat Masuk ke portal AWS akses di Panduan AWS Sign-In Pengguna.

Tetapkan akses ke pengguna tambahan

1. Di Pusat Identitas IAM, buat set izin yang mengikuti praktik terbaik menerapkan izin hak istimewa paling sedikit.

   Untuk petunjuknya, lihat Membuat set izin di Panduan AWS IAM Identity Center Pengguna.
2. Tetapkan pengguna ke grup, lalu tetapkan akses masuk tunggal ke grup.

   Untuk petunjuk, lihat Menambahkan grup di Panduan AWS IAM Identity Center Pengguna.

Prasyarat

Untuk mengikuti prosedur di panduan ini, Anda memerlukan terminal atau shell baris perintah untuk menjalankan perintah. Perintah ditampilkan dalam daftar yang diawali dengan simbol prompt ($) dan nama direktori saat ini, bila perlu.

```
~/eb-project$ this is a command
this is output
```

Di Linux dan macOS, Anda dapat menggunakan shell dan manajer paket pilihan Anda. Pada Windows Anda dapat menginstal Windows Subsystem untuk Linux untuk mendapatkan versi Windows terintegrasi dari Ubuntu dan Bash.

EB CLI

Tutorial ini menggunakan Elastic Beanstalk Command Line Interface (EB CLI). Untuk detail tentang pemasangan dan konfigurasi EB CLI, lihat Instal EB CLI dengan skrip pengaturan (disarankan) dan Mengonfigurasi EB CLI.

Java dan Maven

Jika Anda tidak menginstal Amazon Corretto di mesin lokal Anda, Anda dapat menginstalnya dengan mengikuti petunjuk penginstalan di Panduan Pengguna Amazon Corretto.

Verifikasi instalasi Java Anda dengan menjalankan perintah berikut.

```
~$ java -version
```

Tutorial ini menggunakan Maven. Ikuti petunjuk pengunduhan dan penginstalan di situs web Apache Maven Project. Untuk informasi lebih lanjut tentang Maven, lihat Pusat Pengguna Maven di situs web Apache Maven Project.

Verifikasi instalasi Maven Anda dengan menjalankan perintah berikut.

```
~$ mvn -v
```

Langkah 1: Buat aplikasi Java

Buat direktori proyek.

```
~$ mkdir eb-java
~$ cd eb-java
```

Selanjutnya, buat aplikasi yang akan Anda deploy menggunakan Elastic Beanstalk. Kami akan membuat layanan RESTful web “Hello World”.

Contoh ini menggunakan framework Spring Boot. Aplikasi ini membuka listener pada port 5000. Elastic Beanstalk meneruskan permintaan ke aplikasi Anda pada port 5000 secara default.

Buat file-file berikut:

File ini membuat aplikasi Spring Boot sederhana.

Example **~/eb-java/src/main/java/com/example/Application.java**

```
package com.example;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class Application {

    public static void main(String[] args) {
        SpringApplication.run(Application.class, args);
```

```
    }
}
```

File ini membuat pemetaan yang mengembalikan String yang kita definisikan di sini.

Example **~/eb-java/src/main/java/com/example/Controller.java**

```
package com.example;

import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RestController;

@RestController
public class Controller {

    @GetMapping("/")
    public String index() {
       return "Hello Elastic Beanstalk!";
    }
}
```

File ini mendefinisikan konfigurasi proyek Maven.

Example **~/eb-java/pom.xml**

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/
maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>

  <parent>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-parent</artifactId>
    <version>3.2.3</version>
  </parent>

  <groupId>com.example</groupId>
  <artifactId>BeanstalkJavaExample</artifactId>
  <version>1.0-SNAPSHOT</version>

  <properties>
```

```
    <java.version>21</java.version>
  </properties>

  <dependencies>
    <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-data-rest</artifactId>
    </dependency>
  </dependencies>

  <build>
    <plugins>
      <plugin>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-maven-plugin</artifactId>
      </plugin>
    </plugins>
  </build>

</project>
```

File properti ini mengganti port default menjadi 5000. Ini adalah port default yang mengirimkan lalu lintas Elastic Beanstalk untuk aplikasi Java.

Example **`~/eb-java/application.properties`**

```
server.port=5000
```

Langkah 2: Jalankan aplikasi Anda secara lokal

Package aplikasi Anda dengan perintah berikut:

```
~/eb-java$ mvn clean package
```

Jalankan aplikasi Anda secara lokal dengan perintah berikut:

```
~/eb-java$ java -jar target/BeanstalkJavaExample-1.0-SNAPSHOT.jar
```

Saat aplikasi sedang berjalan, `http://127.0.0.1:5000/` navigasikan ke browser Anda. Anda akan melihat teks “Halo Elastic Beanstalk!”.

Langkah 3: Menyebarkan aplikasi Java Anda dengan EB CLI

Sebelum menerapkan aplikasi Java Anda ke Elastic Beanstalk, mari bersihkan aplikasi build dari direktori Anda dan buat Buildfile dan Procfile untuk mengontrol bagaimana aplikasi dibangun dan dijalankan di lingkungan Elastic Beanstalk Anda.

Untuk mempersiapkan dan mengkonfigurasi penerapan aplikasi

1. Bersihkan aplikasi yang dibangun.

   ```
   ~/eb-java$ mvn clean
   ```

2. Buat `Buildfile` Anda.

   Example **~/eb-java/Buildfile**

   ```
   build: mvn clean package
   ```

   Ini `Buildfile` menentukan perintah yang digunakan untuk membangun aplikasi Anda. Jika Anda tidak menyertakan aplikasi `Buildfile` untuk Java, Elastic Beanstalk tidak mencoba membangun aplikasi Anda.

3. Buat `Procfile` Anda.

   Example **~/eb-java/Procfile**

   ```
   web: java -jar target/BeanstalkJavaExample-1.0-SNAPSHOT.jar
   ```

   Ini `Procfile` menentukan perintah yang digunakan untuk menjalankan aplikasi Anda. Jika Anda tidak menyertakan aplikasi `Procfile` untuk Java, Elastic Beanstalk mengasumsikan ada satu file JAR di root bundel sumber Anda dan mencoba menjalankannya dengan perintah. `java -jar`

Sekarang setelah Anda menyiapkan file konfigurasi untuk membangun dan memulai aplikasi Anda, Anda siap untuk menerapkannya.

Untuk membuat lingkungan dan menyebarkan aplikasi Java Anda

1. Inisialisasi repositori EB CLI Anda dengan perintah eb init.

```
~/eb-java eb init -p corretto java-tutorial --region us-east-2

Application java-tutorial has been created.
```

Perintah ini membuat aplikasi bernama `java-tutorial` dan mengkonfigurasi repositori lokal Anda untuk membuat lingkungan dengan versi platform Java terbaru.

2. (Opsional) Jalankan eb init lagi untuk mengonfigurasi key pair default sehingga Anda dapat menggunakan SSH untuk terhubung ke EC2 instance yang menjalankan aplikasi Anda.

```
~/eb-java$ eb init
Do you want to set up SSH for your instances?
(y/n): y
Select a keypair.
1) my-keypair
2) [ Create new KeyPair ]
```

Pilih pasangan kunci jika Anda sudah memilikinya, atau ikuti prompt untuk membuatnya. Jika Anda tidak melihat prompt atau perlu mengubah pengaturan Anda nanti, jalankan eb init -i.

3. Buat lingkungan baru dan deploy aplikasi Anda dengan eb create. Elastic Beanstalk secara otomatis membuat file zip untuk aplikasi Anda dan memulainya pada port 5000.

```
~/eb-java$ eb create java-env
```

Dibutuhkan sekitar lima menit untuk Elastic Beanstalk untuk menciptakan lingkungan Anda.

Langkah 4: Jalankan aplikasi Anda di Elastic Beanstalk

Ketika proses untuk membuat lingkungan Anda selesai, buka situs web Anda denganeb open.

```
~/eb-java eb open
```

Selamat! Anda telah menerapkan aplikasi Java dengan Elastic Beanstalk! Ini akan membuka jendela peramban menggunakan nama domain yang dibuat untuk aplikasi Anda.

Langkah 5: Bersihkan

Anda dapat menghentikan lingkungan Anda ketika Anda selesai bekerja dengan aplikasi Anda. Elastic Beanstalk AWS mengakhiri semua sumber daya yang terkait dengan lingkungan Anda.

Untuk mengakhiri lingkungan Elastic Beanstalk Anda dengan EB CLI jalankan perintah berikut.

```
~/eb-java$ eb terminate
```

AWS sumber daya untuk aplikasi Anda

Anda baru saja membuat aplikasi instance tunggal. Ini berfungsi sebagai aplikasi sampel langsung dengan satu EC2 instance, sehingga tidak memerlukan penyeimbangan beban atau penskalaan otomatis. Untuk aplikasi contoh tunggal Elastic Beanstalk menciptakan sumber daya berikut: AWS

- EC2 instance — Mesin EC2 virtual Amazon yang dikonfigurasi untuk menjalankan aplikasi web pada platform yang Anda pilih.

  Setiap platform menjalankan satu set perangkat lunak, file konfigurasi, dan penulisan yang berbeda untuk mendukung versi bahasa, kerangka kerja, kontainer web tertentu, atau kombinasi dari semua ini. Sebagian besar platform menggunakan Apache atau nginx sebagai proksi terbalik yang memproses lalu lintas web di depan aplikasi web Anda, meneruskan permintaan ke aplikasi web, menyajikan aset statis, dan menghasilkan log akses dan kesalahan.
- Grup keamanan instans — Grup EC2 keamanan Amazon yang dikonfigurasi untuk memungkinkan lalu lintas masuk di port 80. Sumber daya ini memungkinkan lalu lintas HTTP dari penyeimbang beban mencapai EC2 instance yang menjalankan aplikasi web Anda. Secara default, lalu lintas tidak diizinkan di port lain.
- Bucket Amazon S3 – Lokasi penyimpanan untuk kode sumber, log, dan artifact lainnya yang dibuat saat Anda menggunakan Elastic Beanstalk.
- CloudWatch Alarm Amazon — Dua CloudWatch alarm yang memantau beban pada instans di lingkungan Anda dan dipicu jika beban terlalu tinggi atau terlalu rendah. Saat alarm terpicu, grup Auto Scaling Anda akan menaikkan atau menurunkan skala sebagai respons.
- AWS CloudFormation stack - Elastic AWS CloudFormation Beanstalk digunakan untuk meluncurkan sumber daya di lingkungan Anda dan menyebarkan perubahan konfigurasi. Sumber daya ditentukan di sebuah templat yang dapat Anda lihat di Konsol AWS CloudFormation.
- Nama domain — Nama domain yang merutekan ke aplikasi web Anda dalam formulir *subdomain*.*region*.elasticbeanstalk.com.

Elastic Beanstalk mengelola semua sumber daya tersebut. Ketika Anda mengakhiri lingkungan, Elastic Beanstalk mengakhiri semua sumber daya yang dimuatnya.

Langkah selanjutnya

Setelah lingkungan Anda menjalankan aplikasi, Anda dapat men-deploy versi baru aplikasi atau aplikasi yang berbeda kapan saja. Menyebarkan versi aplikasi baru sangat cepat karena tidak memerlukan penyediaan atau memulai ulang instance. EC2 Anda juga dapat menjelajahi lingkungan baru Anda menggunakan konsol Elastic Beanstalk. Untuk langkah-langkah mendetail, lihat Menjelajahi lingkungan Anda di bagian Memulai panduan ini.

> **Coba lebih banyak tutorial**
>
> Jika Anda ingin mencoba tutorial lain dengan aplikasi contoh yang berbeda, lihatContoh aplikasi dan tutorial.

Setelah Anda menerapkan satu atau dua contoh aplikasi dan siap untuk mulai mengembangkan dan menjalankan aplikasi Java secara lokal, lihat. Menyiapkan lingkungan pengembangan Java Anda

Terapkan dengan konsol Elastic Beanstalk

Anda juga dapat menggunakan konsol Elastic Beanstalk untuk meluncurkan aplikasi sampel. Untuk langkah-langkah rinci, lihat Membuat aplikasi contoh di Bab Memulai panduan ini.

QuickStart: Menyebarkan aplikasi web Java JSP untuk Tomcat ke Elastic Beanstalk

Tutorial ini memandu Anda melalui proses pembuatan aplikasi web Java sederhana menggunakan JavaServer Pages (JSPs). Jika Anda ingin menggabungkan beberapa aplikasi web dalam bentuk file WAR dalam satu lingkungan Elastic Beanstalk, lihat. Paketan beberapa file WAR untuk lingkungan Tomcat

> **Note**
>
> Contoh tutorial dimaksudkan untuk demonstrasi. Jangan gunakan aplikasi untuk lalu lintas produksi.

Bagian-bagian

- AWS Akun Anda
- Prasyarat
- Langkah 1: Buat aplikasi Java JSP
- Langkah 2: Menyebarkan aplikasi Java JSP Anda dengan EB CLI
- Langkah 3: Jalankan aplikasi Anda di Elastic Beanstalk
- Langkah 4: Membersihkan
- AWS sumber daya untuk aplikasi Anda
- Langkah selanjutnya
- Terapkan dengan konsol Elastic Beanstalk

AWS Akun Anda

Jika Anda belum menjadi AWS pelanggan, Anda perlu membuat AWS akun. Mendaftar memungkinkan Anda mengakses Elastic Beanstalk AWS dan layanan lain yang Anda butuhkan.

Jika Anda sudah memiliki AWS akun, Anda dapat melanjutkan kePrasyarat.

Buat AWS akun

Mendaftar untuk Akun AWS

Jika Anda tidak memiliki Akun AWS, selesaikan langkah-langkah berikut untuk membuatnya.

Untuk mendaftar untuk Akun AWS

1. Buka https://portal.aws.amazon.com/billing/pendaftaran.
2. Ikuti petunjuk online.

   Bagian dari prosedur pendaftaran melibatkan menerima panggilan telepon atau pesan teks dan memasukkan kode verifikasi pada keypad telepon.

   Saat Anda mendaftar untuk sebuah Akun AWS, sebuah Pengguna root akun AWSdibuat. Pengguna root memiliki akses ke semua Layanan AWS dan sumber daya di akun. Sebagai praktik keamanan terbaik, tetapkan akses administratif ke pengguna, dan gunakan hanya pengguna root untuk melakukan tugas yang memerlukan akses pengguna root.

AWS mengirimi Anda email konfirmasi setelah proses pendaftaran selesai. Kapan saja, Anda dapat melihat aktivitas akun Anda saat ini dan mengelola akun Anda dengan masuk https://aws.amazon.comke/ dan memilih Akun Saya.

Buat pengguna dengan akses administratif

Setelah Anda mendaftar Akun AWS, amankan Pengguna root akun AWS, aktifkan AWS IAM Identity Center, dan buat pengguna administratif sehingga Anda tidak menggunakan pengguna root untuk tugas sehari-hari.

Amankan Anda Pengguna root akun AWS

1. Masuk ke AWS Management Consolesebagai pemilik akun dengan memilih pengguna Root dan memasukkan alamat Akun AWS email Anda. Di laman berikutnya, masukkan kata sandi.

   Untuk bantuan masuk dengan menggunakan pengguna root, lihat Masuk sebagai pengguna root di AWS Sign-In Panduan Pengguna.
2. Mengaktifkan autentikasi multi-faktor (MFA) untuk pengguna root Anda.

   Untuk petunjuk, lihat Mengaktifkan perangkat MFA virtual untuk pengguna Akun AWS root (konsol) Anda di Panduan Pengguna IAM.

Buat pengguna dengan akses administratif

1. Aktifkan Pusat Identitas IAM.

   Untuk mendapatkan petunjuk, silakan lihat Mengaktifkan AWS IAM Identity Center di Panduan Pengguna AWS IAM Identity Center .
2. Di Pusat Identitas IAM, berikan akses administratif ke pengguna.

   Untuk tutorial tentang menggunakan Direktori Pusat Identitas IAM sebagai sumber identitas Anda, lihat Mengkonfigurasi akses pengguna dengan default Direktori Pusat Identitas IAM di Panduan AWS IAM Identity Center Pengguna.

Masuk sebagai pengguna dengan akses administratif

- Untuk masuk dengan pengguna Pusat Identitas IAM, gunakan URL masuk yang dikirim ke alamat email saat Anda membuat pengguna Pusat Identitas IAM.

Untuk bantuan masuk menggunakan pengguna Pusat Identitas IAM, lihat Masuk ke portal AWS akses di Panduan AWS Sign-In Pengguna.

Tetapkan akses ke pengguna tambahan

1. Di Pusat Identitas IAM, buat set izin yang mengikuti praktik terbaik menerapkan izin hak istimewa paling sedikit.

   Untuk petunjuknya, lihat Membuat set izin di Panduan AWS IAM Identity Center Pengguna.
2. Tetapkan pengguna ke grup, lalu tetapkan akses masuk tunggal ke grup.

   Untuk petunjuk, lihat Menambahkan grup di Panduan AWS IAM Identity Center Pengguna.

Prasyarat

Untuk mengikuti prosedur di panduan ini, Anda memerlukan terminal atau shell baris perintah untuk menjalankan perintah. Perintah ditampilkan dalam daftar yang diawali dengan simbol prompt ($) dan nama direktori saat ini, bila perlu.

```
~/eb-project$ this is a command
this is output
```

Di Linux dan macOS, Anda dapat menggunakan shell dan manajer paket pilihan Anda. Pada Windows Anda dapat menginstal Windows Subsystem untuk Linux untuk mendapatkan versi Windows terintegrasi dari Ubuntu dan Bash.

EB CLI

Tutorial ini menggunakan Elastic Beanstalk Command Line Interface (EB CLI). Untuk detail tentang pemasangan dan konfigurasi EB CLI, lihat Instal EB CLI dengan skrip pengaturan (disarankan) dan Mengonfigurasi EB CLI.

Langkah 1: Buat aplikasi Java JSP

Buat direktori proyek.

```
~$ mkdir eb-tomcat
```

```
~$ cd eb-tomcat
```

Selanjutnya, buat aplikasi yang akan Anda deploy menggunakan Elastic Beanstalk. Kami akan membuat aplikasi web “Hello World”.

Buat file JSP sederhana bernama`index.jsp`.

Example **`~/eb-tomcat/index.jsp`**

```
<html>
  <body>
    <%out.println("Hello Elastic Beanstalk!");%>
  </body>
</html>
```

Langkah 2: Menyebarkan aplikasi Java JSP Anda dengan EB CLI

Jalankan perintah berikut untuk membuat lingkungan Elastic Beanstalk untuk aplikasi ini.

Untuk membuat lingkungan dan menyebarkan aplikasi Java JSP Anda

1. Inisialisasi repositori EB CLI Anda dengan perintah eb init.

   ```
   ~/eb-tomcat$ eb init -p tomcat tomcat-tutorial --region us-east-2
   ```

   Perintah ini membuat aplikasi bernama `tomcat-tutorial` dan mengkonfigurasi repositori lokal Anda untuk membuat lingkungan dengan versi platform Tomcat terbaru.
2. (Opsional) Jalankan eb init lagi untuk mengonfigurasi key pair default sehingga Anda dapat menggunakan SSH untuk terhubung ke EC2 instance yang menjalankan aplikasi Anda.

   ```
   ~/eb-go$ eb init
   Do you want to set up SSH for your instances?
   (y/n): y
   Select a keypair.
   1) my-keypair
   2) [ Create new KeyPair ]
   ```

   Pilih pasangan kunci jika Anda sudah memilikinya, atau ikuti prompt untuk membuatnya. Jika Anda tidak melihat prompt atau perlu mengubah pengaturan Anda nanti, jalankan eb init -i.

3. Buat lingkungan baru dan deploy aplikasi Anda dengan eb create. Elastic Beanstalk secara otomatis membuat file zip untuk aplikasi Anda dan memulainya pada port 5000.

   ```
   ~/eb-tomcat$ eb create tomcat-env
   ```

   Dibutuhkan sekitar lima menit untuk Elastic Beanstalk untuk menciptakan lingkungan Anda.

Langkah 3: Jalankan aplikasi Anda di Elastic Beanstalk

Ketika proses untuk membuat lingkungan Anda selesai, buka situs web Anda denganeb open.

```
~/eb-tomcat$ eb open
```

Selamat! Anda telah menerapkan aplikasi Java JSP dengan Elastic Beanstalk! Ini akan membuka jendela peramban menggunakan nama domain yang dibuat untuk aplikasi Anda.

Langkah 4: Membersihkan

Anda dapat menghentikan lingkungan Anda ketika Anda selesai bekerja dengan aplikasi Anda. Elastic Beanstalk AWS mengakhiri semua sumber daya yang terkait dengan lingkungan Anda.

Untuk mengakhiri lingkungan Elastic Beanstalk Anda dengan EB CLI jalankan perintah berikut.

```
~/eb-tomcat$ eb terminate
```

AWS sumber daya untuk aplikasi Anda

Anda baru saja membuat aplikasi instance tunggal. Ini berfungsi sebagai aplikasi sampel langsung dengan satu EC2 instance, sehingga tidak memerlukan penyeimbangan beban atau penskalaan otomatis. Untuk aplikasi contoh tunggal Elastic Beanstalk menciptakan sumber daya berikut: AWS

- EC2 instance — Mesin EC2 virtual Amazon yang dikonfigurasi untuk menjalankan aplikasi web pada platform yang Anda pilih.

  Setiap platform menjalankan satu set perangkat lunak, file konfigurasi, dan penulisan yang berbeda untuk mendukung versi bahasa, kerangka kerja, kontainer web tertentu, atau kombinasi dari semua ini. Sebagian besar platform menggunakan Apache atau nginx sebagai proksi terbalik yang memproses lalu lintas web di depan aplikasi web Anda, meneruskan permintaan ke aplikasi web, menyajikan aset statis, dan menghasilkan log akses dan kesalahan.

- Grup keamanan instans — Grup EC2 keamanan Amazon yang dikonfigurasi untuk memungkinkan lalu lintas masuk di port 80. Sumber daya ini memungkinkan lalu lintas HTTP dari penyeimbang beban mencapai EC2 instance yang menjalankan aplikasi web Anda. Secara default, lalu lintas tidak diizinkan di port lain.
- Bucket Amazon S3 – Lokasi penyimpanan untuk kode sumber, log, dan artifact lainnya yang dibuat saat Anda menggunakan Elastic Beanstalk.
- CloudWatch Alarm Amazon — Dua CloudWatch alarm yang memantau beban pada instans di lingkungan Anda dan dipicu jika beban terlalu tinggi atau terlalu rendah. Saat alarm terpicu, grup Auto Scaling Anda akan menaikkan atau menurunkan skala sebagai respons.
- AWS CloudFormation stack - Elastic AWS CloudFormation Beanstalk digunakan untuk meluncurkan sumber daya di lingkungan Anda dan menyebarkan perubahan konfigurasi. Sumber daya ditentukan di sebuah templat yang dapat Anda lihat di Konsol AWS CloudFormation.
- Nama domain — Nama domain yang merutekan ke aplikasi web Anda dalam formulir *`subdomain`*.*`region`*.elasticbeanstalk.com.

Elastic Beanstalk mengelola semua sumber daya tersebut. Ketika Anda mengakhiri lingkungan, Elastic Beanstalk mengakhiri semua sumber daya yang dimuatnya.

Langkah selanjutnya

Setelah lingkungan Anda menjalankan aplikasi, Anda dapat men-deploy versi baru aplikasi atau aplikasi yang berbeda kapan saja. Menyebarkan versi aplikasi baru sangat cepat karena tidak memerlukan penyediaan atau memulai ulang instance. EC2 Anda juga dapat menjelajahi lingkungan baru Anda menggunakan konsol Elastic Beanstalk. Untuk langkah-langkah mendetail, lihat Menjelajahi lingkungan Anda di bagian Memulai panduan ini.

> **Coba lebih banyak tutorial**
>
> Jika Anda ingin mencoba tutorial lain dengan aplikasi contoh yang berbeda, lihatContoh aplikasi dan tutorial.

Setelah Anda menerapkan satu atau dua contoh aplikasi dan siap untuk mulai mengembangkan dan menjalankan aplikasi Java dalam wadah web Tomcat lokal, lihat. Menyiapkan lingkungan pengembangan Java Anda

Terapkan dengan konsol Elastic Beanstalk

Anda juga dapat menggunakan konsol Elastic Beanstalk untuk meluncurkan aplikasi sampel. Untuk langkah-langkah rinci, lihat Membuat aplikasi contoh di Bab Memulai panduan ini.

Menyiapkan lingkungan pengembangan Java Anda

Topik ini memberikan petunjuk untuk menyiapkan lingkungan pengembangan Java untuk menguji aplikasi Anda secara lokal sebelum menerapkannya. AWS Elastic Beanstalk Ini juga merujuk situs web yang menyediakan instruksi instalasi untuk alat yang berguna.

Bagian-bagian

- Menginstal kit pengembangan Java
- Menginstal kontainer web
- Mengunduh pustaka
- Menginstal AWS SDK for Java
- Menginstal IDE atau editor teks

Menginstal kit pengembangan Java

Instal Java Development Kit (JDK). Jika Anda tidak memiliki preferensi, dapatkan versi terbarunya. Unduh JDK di oracle.com

JDK mencakup kompilator Java, dapat Anda gunakan untuk membangun file sumber Anda ke dalam file kelas yang dapat dijalankan di server web Elastic Beanstalk.

Menginstal kontainer web

Jika Anda belum memiliki wadah web atau kerangka kerja lain, instal versi Tomcat yang didukung Elastic Beanstalk untuk sistem operasi Amazon Linux Anda. Untuk daftar versi Apache Tomcat saat ini yang didukung Elastic Beanstalk, lihat Tomcat di dokumen Platform.AWS Elastic Beanstalk Unduh versi Tomcat yang berlaku untuk lingkungan Anda dari situs web Apache Tomcat.

Mengunduh pustaka

Platform Elastic Beanstalk mencakup beberapa pustaka secara default. Unduh pustaka yang akan digunakan aplikasi Anda dan simpan di folder proyek Anda untuk men-deploy di paket sumber aplikasi Anda.

Jika Anda telah menginstal Tomcat secara lokal, Anda dapat menyalin pustaka API servlet API dan JavaServer Pages (JSP) dari folder instalasi. Jika Anda men-deploy ke versi platform Tomcat, Anda tidak perlu menyertakan file-file ini ke paket sumber Anda, tetapi Anda perlu memilikinya di `classpath` untuk mengompilasi setiap kelas yang menggunakannya.

JUnit, Google Guava, dan Apache Commons menyediakan beberapa perpustakaan yang berguna. Kunjungi halaman beranda mereka untuk pelajari selengkapnya:

- Unduh JUnit
- Unduh Google Guava
- Unduh Apache Commons

Menginstal AWS SDK for Java

Jika Anda perlu mengelola AWS sumber daya dari dalam aplikasi Anda, instal AWS SDK for Java. Misalnya, dengan AWS SDK untuk Java, Anda dapat menggunakan Amazon DynamoDB (DynamoDB) untuk berbagi status sesi aplikasi Apache Tomcat di beberapa server web. Untuk informasi selengkapnya, lihat Mengelola Status Sesi Tomcat dengan Amazon DynamoDB di dokumentasi AWS SDK for Java.

Kunjungi beranda AWS SDK for Java untuk informasi selengkapnya dan petunjuk instalasi.

Menginstal IDE atau editor teks

Lingkungan pengembangan terintegrasi (IDEs) menyediakan berbagai fitur yang memfasilitasi pengembangan aplikasi. Jika Anda belum menggunakan IDE untuk pengembangan Java, coba Eclipse dan IntelliJ dan lihat mana yang paling sesuai untuk Anda.

- Instal Eclipse IDE untuk Pengembang Java EE
- Instal IntelliJ

IDE mungkin saja menambahkan file ke folder proyek yang mungkin tidak ingin Anda masukkan ke kontrol sumber. Untuk mencegah memasukkan file-file ini ke kontrol sumber, gunakan `.gitignore` atau padanan alat kontrol sumber Anda.

Jika Anda baru ingin memulai coding dan tidak memerlukan semua fitur IDE, pertimbangkan untuk menginstal Sublime Text.

> **Note**
>
> Pada tanggal 31 Mei 2023, AWS Toolkit for Eclipsemencapai akhir hayat dan tidak lagi didukung oleh AWS. Untuk detail tambahan mengenai akhir siklus hidup AWS Toolkit for Eclipse, lihat file README.md di repositori. AWS Toolkit for Eclipse GitHub

Lebih banyak contoh aplikasi dan tutorial Elastic Beanstalk untuk Java

Bagian ini menyediakan aplikasi dan tutorial tambahan. QuickStart untuk Java di TomcatTopik QuickStart untuk Java dan topik yang terletak sebelumnya dalam topik ini memandu Anda melalui peluncuran aplikasi Java sampel dengan EB CLI.

Untuk memulai aplikasi Java aktif AWS Elastic Beanstalk, yang Anda butuhkan hanyalah bundel sumber aplikasi untuk diunggah sebagai versi aplikasi pertama Anda dan untuk menyebarkan ke lingkungan. Ketika Anda membuat lingkungan, Elastic Beanstalk mengalokasikan semua sumber daya AWS yang diperlukan untuk menjalankan aplikasi web yang dapat diskalakan.

Meluncurkan lingkungan dengan sampel aplikasi Java

Elastic Beanstalk menyediakan contoh aplikasi satu halaman untuk setiap platform serta contoh yang lebih kompleks yang menunjukkan penggunaan sumber daya AWS tambahan seperti Amazon RDS dan fitur khusus bahasa atau platform dan. APIs

Sampel satu halaman adalah kode yang sama yang Anda dapatkan ketika Anda membuat lingkungan tanpa menyediakan kode sumber Anda sendiri. Contoh yang lebih kompleks di-host GitHub dan mungkin perlu dikompilasi atau dibangun sebelum diterapkan ke lingkungan Elastic Beanstalk.

Sampel

Nama	Versi yang didukung	Tipe lingkun	Sumber	Deskripsi
Tomc (satu halan	Semua cabang platform Tomcat dengan Corretto	Serve Web Peker	tomcat.zip	Aplikasi web Tomcat dengan satu halaman (`index.jsp`) dikonfigurasi untuk ditampilkan di akar situs web. Untuk lingkungan pekerja, sampel ini mencakup file `cron.yaml` yang mengonfigurasi tugas terjadwal yang memanggil `scheduled.jsp` sekali per menit. Saat `scheduled.jsp` dipanggil , berkas log ditulis di `/tmp/sample-app.log` . Terakhir, file konfigurasi disertakan di `.ebextensions` yang menyalin log dari `/tmp/` ke lokasi yang dibaca oleh Elastic Beanstalk saat Anda meminta log lingkungan. Jika Anda mengaktifkan integrasi X-Ray di lingkungan yang menjalankan sampel ini, aplikasi akan menampilkan konten tambahan terkait X-Ray dan menyediakan opsi untuk menghasilkan informasi debug yang dapat Anda lihat di konsol X-Ray.
Corre (satu halan	Corretto 11 Corretto 8	Serve Web	corretto.zip	Aplikasi Corretto dengan file konfigurasi `Buildfile` dan `Procfile`. Jika Anda mengaktifkan integrasi X-Ray di lingkungan yang menjalankan sampel ini, aplikasi akan menampilkan konten tambahan terkait X-Ray dan menyediakan opsi untuk menghasilkan informasi debug yang dapat Anda lihat di konsol X-Ray.

Nama	Versi yang didukung	Tipe lingkun	Sumber	Deskripsi
Score	Java 8	Serve Web	Kloning repo di .com GitHub	Scorekeep adalah API RESTful web yang menggunakan framework Spring untuk menyediakan antarmuka untuk membuat dan mengelola pengguna, sesi, dan game. API adalah paket dengan aplikasi web Angular 1.5 yang menggunakan API melalui HTTP. Aplikasi ini menggunakan fitur platform Java SE untuk mengunduh dependensi dan membangun pada instans, meminimal kan ukuran paket sumber. Aplikasi ini juga mencakup file konfigurasi nginx yang mengganti konfigurasi default untuk melayani aplikasi web frontend secara statis di port 80 melalui proksi, dan merutekan permintaan ke jalur di bawah `/api` ke API yang berjalan di `localhost:5000` . Scorekeep juga mencakup cabang `xray` yang menunjukkan cara menginstrumentasi aplikasi Java untuk digunakan dengan AWS X-Ray. Ini menunjukkan instrumen tasi permintaan HTTP masuk dengan filter servlet, instrumentasi klien AWS SDK otomatis dan manual, konfigurasi perekam, dan instrumentasi permintaan HTTP keluar dan klien SQL. Lihat readme untuk petunjuk atau gunakan tutorial memulai AWS X-Ray untuk mencoba aplikasi dengan X-Ray.

Nama	Versi yang didukung	Tipe lingkun	Sumber	Deskripsi
Does it Have Snak	Tomcat 8 dengan Java 8	Serve Web	Kloning repo di .com GitHub	Apakah ia memiliki ular? adalah aplikasi web Tomcat yang menunjukkan penggunaan file konfigurasi Elastic Beanstalk, Amazon RDS, JDBC, PostgreSQL, JSPs Servlets,, Dukungan Tag Sederhana, File Tag, Log4J, Bootstrap, dan Jackson. Kode sumber untuk proyek ini termasuk membangun penulisan minimal yang mengompilasi servlet dan model ke dalam file kelas dan paket file yang diperlukan ke Arsip Web yang dapat Anda deploy ke lingkungan Elastic Beanstalk. Lihat file readme di repositori proyek untuk petunjuk lengkap.
Locus Load Gene	Java 8	Serve Web	Kloning repo di .com GitHub	Aplikasi web yang dapat Anda gunakan untuk menguji beban aplikasi web lain yang berjalan di lingkungan Elastic Beanstalk yang berbeda. Menunjukkan penggunaan file `Buildfile` dan `Procfile`, DynamoDB, dan Locust, alat pengujian beban sumber terbuka.

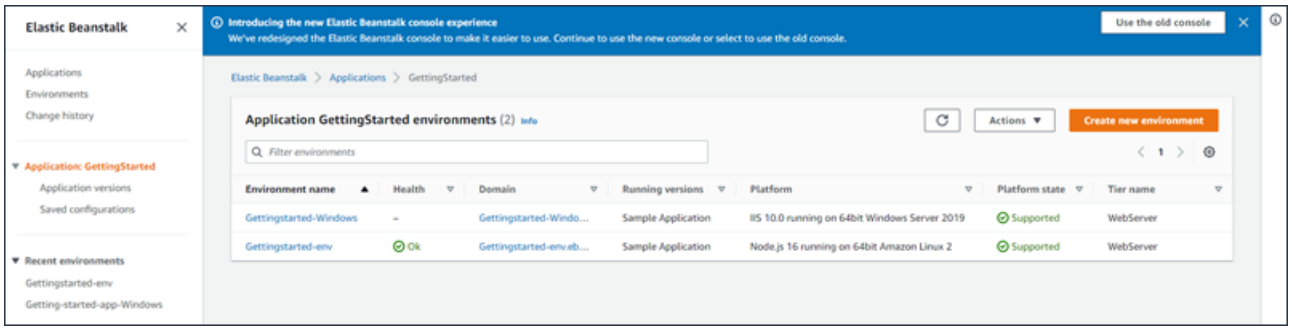
Unduh aplikasi sampel dan deploy ke Elastic Beanstalk dengan mengikuti langkah-langkah berikut:

Untuk meluncurkan lingkungan dengan aplikasi sampel (konsol)

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Pada panel navigasi, pilih Aplikasi. Pilih aplikasi yang ada dalam daftar. Anda juga dapat memilih untuk membuatnya, mengikuti instruksi diMengelola aplikasi.

3. Pada halaman ikhtisar aplikasi, pilih Buat lingkungan baru.

   Gambar berikut menampilkan halaman ikhtisar aplikasi.



   Ini meluncurkan wizard Create environment. Wizard menyediakan serangkaian langkah bagi Anda untuk membuat lingkungan baru.

4. Untuk tingkat Lingkungan, pilih lingkungan server Web atau tingkat lingkungan Worker. Anda tidak dapat mengubah tingkat lingkungan setelah pembuatan.

   > **Note**
   >
   > .NET di platform Windows Server tidak mendukung tingkat lingkungan pekerja.

   Bidang informasi Aplikasi default, berdasarkan aplikasi yang Anda pilih sebelumnya.

   Dalam informasi Lingkungan mengelompokkan nama Environment default, berdasarkan nama aplikasi. Jika Anda lebih suka nama lingkungan yang berbeda, Anda dapat memasukkan nilai lain di bidang. Anda dapat secara opsional memasukkan nama Domain; jika tidak, Elastic Beanstalk menghasilkan nilai secara otomatis. Anda juga dapat secara opsional memasukkan deskripsi Lingkungan.

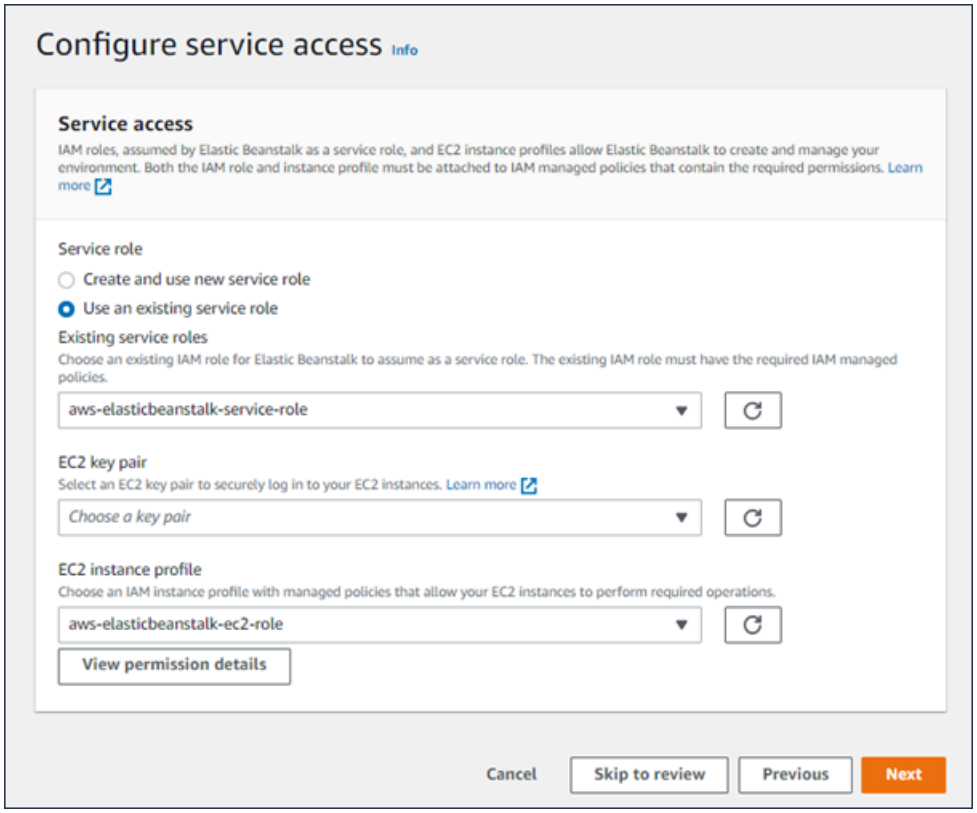
5. Untuk Platform, pilih platform dan cabang platform yang sesuai dengan bahasa yang digunakan aplikasi Anda.

   > **Note**
   >
   > Elastic Beanstalk mendukung beberapa versi untuk sebagian besar platform yang tercantum. Secara default, konsol tersebut memilih versi yang direkomendasikan untuk platform dan cabang platform yang Anda pilih. Jika aplikasi Anda memerlukan versi yang

berbeda, Anda dapat memilihnya di sini. Untuk informasi tentang versi platform yang didukung, lihat the section called “Platform yang didukung”.

6. Untuk kode Aplikasi, Anda memiliki beberapa pilihan untuk meluncurkan aplikasi sampel.
   - Untuk meluncurkan aplikasi sampel default tanpa memasok kode sumber, pilih Contoh aplikasi. Tindakan ini memilih aplikasi satu halaman yang disediakan Elastic Beanstalk untuk platform yang Anda pilih sebelumnya.
   - Jika Anda mengunduh contoh aplikasi dari panduan ini atau sumber lain, lakukan langkah-langkah berikut.
     a. Pilih Unggah kode Anda.
     b. Selanjutnya pilih File lokal, lalu di bawah Unggah aplikasi, pilih Pilih file.
     c. Sistem operasi komputer Anda akan memberi Anda antarmuka untuk memilih file lokal yang Anda unduh. Pilih file bundel sumber dan lanjutkan.
7. Untuk Preset, pilih Single instance.
8. Pilih Berikutnya.
9. Halaman akses layanan Konfigurasi ditampilkan.

Gambar berikut mengilustrasikan halaman akses layanan Konfigurasi.



10. Pilih nilai dari dropdown Peran Layanan yang Ada.
11. (Opsional) Jika sebelumnya Anda membuat EC2 key pair, Anda dapat memilihnya dari dropdown bidang EC2 key pair. Anda akan menggunakannya untuk masuk dengan aman ke EC2 instans Amazon yang disediakan Elastic Beanstalk untuk aplikasi Anda. Jika Anda melewati langkah ini, Anda selalu dapat membuat dan menetapkan EC2 key pair setelah lingkungan dibuat. Untuk informasi selengkapnya, lihat EC2 key pair.
12. Selanjutnya, kita akan fokus pada daftar dropdown profil EC2 instance. Nilai yang ditampilkan dalam daftar dropdown ini dapat bervariasi, tergantung pada apakah akun Anda sebelumnya telah membuat lingkungan baru.

    Pilih salah satu item berikut, berdasarkan nilai yang ditampilkan dalam daftar Anda.

    - Jika `aws-elasticbeanstalk-ec2-role` ditampilkan dalam daftar dropdown, pilih dari daftar dropdown.
    - Jika nilai lain ditampilkan dalam daftar, dan itu adalah profil EC2 instance default yang ditujukan untuk lingkungan Anda, pilih dari daftar dropdown.
    - Jika daftar dropdown profil EC2 instance tidak mencantumkan nilai apa pun, Anda harus membuat profil instance.

      > **Buat profil instance**
      >
      > Untuk membuat profil instance, kita akan mengambil jalan memutar ke prosedur lain di halaman yang sama ini. Pergi ke akhir prosedur ini dan memperluas prosedur yang berikut, Buat IAM Role EC2 misalnya profil.
      > Selesaikan langkah-langkah di Buat Peran IAM EC2 misalnya profil untuk membuat Peran IAM yang selanjutnya dapat Anda pilih untuk profil EC2 instans. Kemudian kembali ke langkah ini.

    Sekarang Anda telah membuat Peran IAM, dan menyegarkan daftar, itu ditampilkan sebagai pilihan dalam daftar dropdown. Pilih Peran IAM yang baru saja Anda buat dari daftar dropdown profil EC2 instance.
13. Pilih Lewati untuk Meninjau di halaman Konfigurasi akses layanan.

    Ini akan memilih nilai default untuk langkah ini dan melewati langkah-langkah opsional.
14. Halaman Review menampilkan ringkasan dari semua pilihan Anda.

Untuk lebih menyesuaikan lingkungan Anda, pilih Edit di samping langkah yang menyertakan item apa pun yang ingin Anda konfigurasikan. Anda dapat mengatur opsi berikut hanya selama pembuatan lingkungan:

- Nama lingkungan
- Nama domain
- Versi platform
- Prosesor
- VPC
- Tingkat

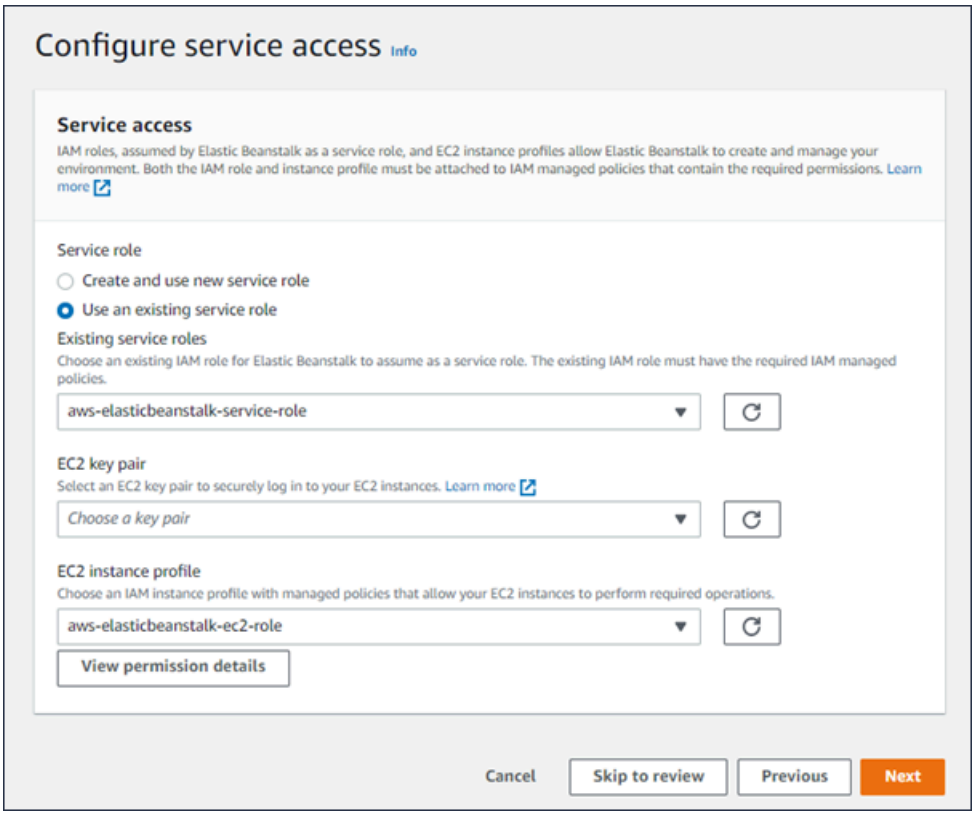
Anda dapat mengubah pengaturan berikut setelah pembuatan lingkungan, tetapi mereka memerlukan instans baru atau sumber daya lain untuk disediakan dan dapat memakan waktu lama untuk menerapkan:

- Jenis instans, volume root, key pair, dan peran AWS Identity and Access Management (IAM)
- Basis data Amazon RDS internal
- Penyeimbang beban

Untuk detail di semua pengaturan yang tersedia, lihat Wizard pembuatan lingkungan baru.

15. Pilih Kirim di bagian bawah halaman untuk menginisialisasi pembuatan lingkungan baru Anda.

Buat Peran IAM EC2 misalnya profil



Untuk membuat profil EC2 instance

1. Pilih Buat peran.
2. Untuk jenis entitas Tepercaya, pilih AWS layanan.
3. Untuk kasus Penggunaan, pilih Elastic Beanstalk — Compute.
4. Pilih Berikutnya.
5. Verifikasi bahwa kebijakan Izin menyertakan yang berikut ini, lalu pilih Berikutnya:
   - `AWSElasticBeanstalkWebTier`
   - `AWSElasticBeanstalkWorkerTier`
   - `AWSElasticBeanstalkMulticontainerDocker`
6. Pilih Buat peran.
7. Kembali ke tab Konfigurasi akses layanan, segarkan daftar, lalu pilih profil EC2 instance yang baru dibuat.

Langkah selanjutnya

Setelah Anda memiliki lingkungan yang menjalankan aplikasi, Anda dapat men-deploy versi baru aplikasi atau aplikasi yang sama sekali berbeda setiap saat. Menyebarkan versi aplikasi baru sangat cepat karena tidak memerlukan penyediaan atau memulai ulang instance. EC2

Setelah men-deploy satu atau dua aplikasi sampel, dan Anda siap mulai mengembangkan dan menjalankan aplikasi Java secara lokal, lihat bagian selanjutnya untuk menyiapkan lingkungan pengembangan Java dengan semua alat yang akan diperlukan.

Menggunakan platform Elastic Beanstalk Tomcat

Topik ini menjelaskan cara mengonfigurasi, membangun, dan menjalankan aplikasi Java Anda yang berjalan di platform Elastic Beanstalk Tomcat.

Platform AWS Elastic Beanstalk Tomcat adalah seperangkat versi platform untuk aplikasi web Java yang dapat berjalan dalam wadah web Tomcat. Tomcat berjalan di belakang server proksi nginx. Setiap cabang platform sesuai dengan versi utama Tomcat.

Opsi konfigurasi tersedia di konsol Elastic Beanstalk untuk memodifikasi konfigurasi dari lingkungan yang sedang berjalan. Agar Anda tidak kehilangan konfigurasi lingkungan ketika mengakhirinya, Anda dapat menggunakan konfigurasi tersimpan untuk menyimpan pengaturan dan kemudian menerapkannya ke lingkungan lain.

Untuk menyimpan pengaturan di kode sumber, Anda dapat menyertakan file konfigurasi. Pengaturan di file konfigurasi diterapkan setiap kali Anda membuat lingkungan atau men-deploy aplikasi Anda. Anda juga dapat menggunakan file konfigurasi untuk menginstal paket, menjalankan penulisan, dan melakukan operasi penyesuaian instans lainnya selama deployment.

Platform Tomcat Elastic Beanstalk mencakup proksi terbalik yang meneruskan permintaan ke aplikasi Anda. Anda dapat menggunakan opsi konfigurasi untuk mengonfigurasi server proksi untuk melayani aset statis dari folder di kode sumber Anda untuk mengurangi beban di aplikasi Anda. Untuk skenario lanjutan, Anda dapat menyertakan file `.conf` milik Anda sendiri di paket sumber Anda untuk memperluas konfigurasi proksi Elastic Beanstalk atau menimpa sepenuhnya.

> **Note**
>
> Elastic Beanstalk mendukung nginx (default) dan Server HTTP Apache sebagai server proksi di platform Tomcat. Jika lingkungan Elastic Beanstalk Tomcat Anda menggunakan cabang

platform Amazon Linux AMI (Amazon Linux 2 terdahulu), Anda juga memiliki opsi untuk menggunakan Apache HTTP Server Versi 2.2. Apache (terbaru) adalah default di cabang-cabang platform sebelumnya.
Pada tanggal 18 Juli 2022, Elastic Beanstalk menetapkan status semua cabang platform berdasarkan Amazon Linux AMI () untuk pensiun. AL1 Untuk informasi selengkapnya tentang migrasi ke cabang platform Amazon Linux 2023 saat ini dan didukung penuh, lihat. Memigrasi aplikasi Elastic Beanstalk Linux Anda ke Amazon Linux 2023 atau Amazon Linux 2

Anda harus mengemas aplikasi Java di file arsip aplikasi web (WAR) dengan struktur tertentu. Untuk informasi tentang struktur yang diperlukan dan bagaimana kaitannya dengan struktur direktori proyek Anda, lihat Menata folder proyek Anda.

Untuk menjalankan beberapa aplikasi di server web yang sama, Anda dapat memaket beberapa file WAR menjadi paket sumber tunggal. Setiap aplikasi di beberapa paket sumber WAR berjalan di jalur akar (`ROOT.war` berjalan di *`myapp`*`.elasticbeanstalk.com/`) atau di jalur yang berada langsung di bawahnya (`app2.war` berjalan di *`myapp`*`.elasticbeanstalk.com/`*`app2`*`/`), seperti yang ditentukan oleh nama WAR. Di satu paket sumber WAR, aplikasi selalu berjalan di jalur akar.

Pengaturan yang diterapkan di konsol Elastic Beanstalk mengganti pengaturan yang sama di file konfigurasi, jika ada. Hal tersebut memungkinkan Anda memiliki pengaturan default di file konfigurasi, dan mengganti keduanya dengan pengaturan khusus lingkungan di konsol tersebut. Untuk informasi selengkapnya tentang yang diutamakan, dan metode lain untuk mengubah pengaturan, lihat Opsi konfigurasi.

Untuk detail tentang berbagai cara memperluas platform berbasis Linux Elastic Beanstalk, lihat the section called “Memperluas platform Linux”.

Topik

- Mengonfigurasi lingkungan Tomcat Anda
- Namespace konfigurasi Tomcat
- Paketan beberapa file WAR untuk lingkungan Tomcat
- Menata folder proyek Anda
- Mengonfigurasi server proksi

Mengonfigurasi lingkungan Tomcat Anda

Platform Elastic Beanstalk Tomcat menyediakan beberapa opsi khusus platform selain opsi standar yang dimiliki semua platform. Opsi ini memungkinkan Anda untuk mengonfigurasi mesin virtual Java (JVM) yang berjalan di server web lingkungan Anda, dan menentukan properti sistem yang menyediakan string konfigurasi informasi untuk aplikasi Anda.

Anda dapat menggunakan konsol Elastic Beanstalk untuk mengaktifkan rotasi log ke Amazon S3 dan mengonfigurasi variabel yang dapat dibaca aplikasi Anda dari lingkungan.

Untuk mengonfigurasi lingkungan Tomcat Anda di konsol Elastic Beanstalk

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Dalam kategori konfigurasi Pembaruan, pemantauan, dan pencatatan, pilih Edit.

Opsi kontainer

Anda dapat menentukan opsi khusus platform berikut ini:

- Server proksi – Server proksi untuk digunakan di instans lingkungan Anda. Secara default, nginx digunakan.

Opsi kontainer JVM

Ukuran tumpukan di mesin virtual Java (JVM) menentukan berapa banyak objek yang dapat dibuat aplikasi Anda di memori sebelum pengumpulan sampah terjadi. Anda dapat memodifikasi Ukuran Tumpukan JVM Awal (**`-Xms option`**) dan Ukuran Tumpukan JVM Maksimum (opsi). `-Xmx` Ukuran tumpukan awal yang lebih besar mengizinkan lebih banyak objek yang akan dibuat sebelum pengumpulan sampah terjadi, tetapi ini juga berarti bahwa pengumpul sampah akan memakan waktu lebih lama untuk memadatkan tumpukan. Ukuran tumpukan maksimum menentukan jumlah maksimum memori yang JVM dapat alokasikan ketika memperluas tumpukan selama aktivitas berat.

> **Note**
>
> Memori yang tersedia tergantung pada jenis EC2 instans Amazon. Untuk informasi selengkapnya tentang jenis EC2 instans yang tersedia untuk lingkungan Elastic Beanstalk, lihat Jenis Instans di Panduan Pengguna Amazon Elastic Compute Cloud untuk Instans Linux.

Generasi permanen adalah bagian dari tumpukan JVM yang menyimpan definisi kelas dan metadata terkait. Untuk memodifikasi ukuran generasi permanen, ketik ukuran baru di opsi Maximum JVM PermGen Size (`-XX:MaxPermSize`). Pengaturan ini hanya berlaku untuk Java 7 dan sebelumnya. Opsi ini tidak digunakan lagi di JDK 8 dan digantikan oleh opsi Size (). MaxMetaspace `-XX:MaxMetaspaceSize`

> **Important**
>
> JDK 17 menghapus dukungan opsi Java`-XX:MaxPermSize`. Penggunaan opsi ini dengan lingkungan yang berjalan pada cabang platform Elastic Beanstalk dengan Corretto 17 akan menghasilkan kesalahan. Elastic Beanstalk merilis cabang platform pertamanya yang menjalankan Tomcat dengan Corretto 17 pada 13 Juli 2023.
> Untuk informasi lebih lanjut, lihat sumber daya berikut.
>
> - Situs dokumentasi Oracle Java: Dihapus Java Options
> - Situs dokumentasi Oracle Java: Bagian Metadata Kelas dalam Pertimbangan Lain

Untuk informasi selengkapnya tentang platform Elastic Beanstalk dan komponennya, lihat Platform yang Didukung dalam panduan Platform.AWS Elastic Beanstalk

Opsi log

Bagian Opsi Log memiliki dua pengaturan:

- Profil instans – Menentukan profil instans yang memiliki izin untuk mengakses bucket Amazon S3 yang terkait dengan aplikasi Anda.
- Aktifkan rotasi file log ke Amazon S3 — Menentukan apakah file log untuk EC2 instans Amazon aplikasi Anda disalin ke bucket Amazon S3 yang terkait dengan aplikasi Anda.

File statis

Untuk meningkatkan kinerja, Anda dapat menggunakan bagian File statis untuk mengkonfigurasi server proxy untuk melayani file statis (misalnya, HTML atau gambar) dari satu set direktori di dalam aplikasi web Anda. Untuk setiap direktori, Anda mengatur jalur virtual ke pemetaan direktori. Saat server proksi menerima permintaan untuk file di jalur yang ditentukan, server langsung menyajikan file daripada merutekan permintaan ke aplikasi Anda.

Untuk detail tentang mengonfigurasi file statis menggunakan file konfigurasi atau konsol Elastic Beanstalk, lihat. the section called “File statis”

Properti lingkungan

Di bagian Properti Lingkungan, Anda dapat menentukan pengaturan konfigurasi lingkungan pada EC2 instans Amazon yang menjalankan aplikasi Anda. Properti lingkungan diberikan sebagai pasangan nilai kunci ke aplikasi.

Platform Tomcat menentukan properti placeholder bernama `JDBC_CONNECTION_STRING` untuk lingkungan Tomcat untuk melewati string koneksi ke basis data eksternal.

> **Note**
>
> Jika Anda melampirkan instans DB RDS ke lingkungan Anda, bangun koneksi string JDBC secara dinamis dari properti lingkungan Amazon Relational Database Service (Amazon RDS) yang disediakan oleh Elastic Beanstalk. Gunakan JDBC_CONNECTION_STRING hanya untuk instans basis data yang tidak disediakan menggunakan Elastic Beanstalk.
> Untuk informasi selengkapnya tentang menggunakan Amazon RDS dengan aplikasi Java Anda, lihat Menambahkan instans Amazon RDS DB ke lingkungan Java Elastic Beanstalk Anda.

Untuk versi platform Tomcat yang dirilis sebelum 26 Maret 2025, variabel lingkungan dapat diakses menggunakan. `System.getProperty()` Misalnya, Anda bisa membaca properti bernama `API_ENDPOINT` dari variabel dengan kode berikut.

```
String endpoint = System.getProperty("API_ENDPOINT");
```

Versi platform Tomcat yang dirilis pada atau setelah 26 Maret 2025, juga dapat digunakan `System.getenv` untuk mengakses variabel lingkungan teks biasa. Anda dapat terus menggunakan

`System.getProperty` untuk mengakses variabel lingkungan plaintext. Namun, variabel lingkungan yang disimpan sebagai rahasia hanya tersedia menggunakan`System.getenv`. Misalnya, Anda bisa membaca variabel lingkungan bernama `API_KEY` dengan kode berikut.

```
String apiKey = System.getenv("API_KEY");
```

> **Important**
>
> Penambahan `System.getenv()` akses untuk variabel lingkungan dalam versi platform Tomcat yang dirilis pada atau setelah 26 Maret 2025 dapat menyebabkan perilaku tak terduga dalam aplikasi yang memberikan prioritas variabel lingkungan daripada properti sistem Java atau ketika secara eksplisit beralih dari ke. `System.getProperty()` `System.getenv()`
> Karena properti sistem (diteruskan melalui baris perintah) memerlukan pelarian shell untuk karakter khusus sementara variabel lingkungan tidak, nilai dapat diselesaikan secara berbeda saat menggunakan variabel lingkungan alih-alih properti sistem Java.
> Jika aplikasi Anda terpengaruh, pertimbangkan:
>
> - Menghapus karakter escape dari nilai properti lingkungan Anda saat menggunakan `System.getenv()`
> - Mengkonfigurasi aplikasi Anda untuk digunakan secara eksplisit `System.getProperty()`
> - Menguji aplikasi Anda secara menyeluruh saat memutakhirkan untuk memastikan perilaku yang konsisten

Lihat Variabel lingkungan dan pengaturan perangkat lunak lainnya untuk informasi selengkapnya.

Namespace konfigurasi Tomcat

Anda dapat menggunakan file konfigurasi untuk mengatur opsi konfigurasi dan melakukan tugas-tugas konfigurasi instans lain selama deployment. Opsi konfigurasi dapat berupa platform khusus atau berlaku untuk semua platform di layanan Elastic Beanstalk secara keseluruhan. Opsi konfigurasi diatur ke dalam ruang nama.

Platform Tomcat mendukung opsi di namespace berikut, selain opsi yang didukung untuk semua lingkungan Elastic Beanstalk:

- `aws:elasticbeanstalk:container:tomcat:jvmoptions` – Modifikasi pengaturan JVM. Opsi di namespace ini sesuai dengan opsi di konsol manajemen, sebagai berikut:
  - `Xms` – Opsi baris perintah JVM
  - `JVM Options` – Opsi baris perintah JVM
- `aws:elasticbeanstalk:environment:proxy` – Pilih server proksi lingkungan.

File konfigurasi contoh berikut menunjukkan penggunaan opsi konfigurasi tertentu Tomcat.

Example .ebextensions/tomcat-settings.config

```
option_settings:
  aws:elasticbeanstalk:container:tomcat:jvmoptions:
    Xms: 512m
    JVM Options: '-Xmn128m'
  aws:elasticbeanstalk:application:environment:
    API_ENDPOINT: mywebapi.zkpexsjtmd.us-west-2.elasticbeanstalk.com
  aws:elasticbeanstalk:environment:proxy:
    ProxyServer: apache
```

Elastic Beanstalk memberikan banyak opsi konfigurasi untuk menyesuaikan lingkungan Anda. Selain file konfigurasi, Anda juga dapat mengatur opsi konfigurasi menggunakan konsol tersebut, konfigurasi tersimpan, EB CLI, atau AWS CLI. Lihat Opsi konfigurasi untuk informasi selengkapnya.

Platform Tomcat Amazon Linux AMI (Amazon Linux 2 terdahulu)

Jika lingkungan Tomcat Elastic Beanstalk Anda menggunakan versi platform Amazon Linux AMI (Amazon Linux 2 terdahulu), baca informasi tambahan di bagian ini.

> **Catatan**
>
> - Informasi dalam topik ini hanya berlaku untuk cabang platform berbasis Amazon Linux AMI (AL1). AL2023/ cabang AL2 platform tidak kompatibel dengan versi platform Amazon Linux AMI (AL1) sebelumnya dan memerlukan pengaturan konfigurasi yang berbeda.
> - Pada tanggal 18 Juli 2022, Elastic Beanstalk menetapkan status semua cabang platform berdasarkan Amazon Linux AMI () untuk pensiun. AL1 Untuk informasi selengkapnya tentang migrasi ke cabang platform Amazon Linux 2023 saat ini dan didukung penuh, lihat.

> [Memigrasi aplikasi Elastic Beanstalk Linux Anda ke Amazon Linux 2023 atau Amazon Linux 2]

Ruang nama konfigurasi Tomcat - Amazon Linux AMI () AL1

Platform Tomcat Amazon Linux AMI mendukung opsi tambahan di namespace berikut:

- `aws:elasticbeanstalk:container:tomcat:jvmoptions` – Selain opsi yang disebutkan sebelumnya di halaman ini untuk namespace ini, versi platform Amazon Linux AMI sebelumnya juga mendukung:
  - `XX:MaxPermSize` – Ukuran generasi permanen JVM maksimum
- `aws:elasticbeanstalk:environment:proxy` – Selain memilih server proksi, juga mengonfigurasi kompresi respons.

File konfigurasi contoh berikut menunjukkan penggunaan opsi konfigurasi namespace proksi.

Example .ebextensions/tomcat-settings.config

```
option_settings:
  aws:elasticbeanstalk:environment:proxy:
    GzipCompression: 'true'
    ProxyServer: nginx
```

Sertakan file konfigurasi Elastic Beanstalk - Amazon Linux AMI () AL1

Untuk men-deploy file konfigurasi `.ebextensions`, sertakan mereka di sumber aplikasi Anda. Untuk satu aplikasi, tambahkan `.ebextensions` ke file WAR terkompresi dengan menjalankan perintah berikut:

Example

```
zip -ur your_application.war .ebextensions
```

Untuk aplikasi yang membutuhkan beberapa file WAR, lihat [Paketan beberapa file WAR untuk lingkungan Tomcat] untuk petunjuk lebih lanjut.

Paketan beberapa file WAR untuk lingkungan Tomcat

Jika aplikasi web Anda terdiri dari beberapa komponen aplikasi web, Anda dapat menyederhanakan deployment dan mengurangi biaya operasi dengan menjalankan komponen di satu lingkungan, bukan menjalankan lingkungan yang terpisah untuk setiap komponen. Strategi ini efektif untuk aplikasi ringan yang tidak memerlukan banyak sumber daya, dan untuk pengembangan dan pengujian lingkungan.

Untuk men-deploy beberapa aplikasi web ke lingkungan Anda, gabungkan setiap file arsip aplikasi web (WAR) komponen ke dalam satu paket sumber.

Untuk membuat paket sumber aplikasi yang berisi beberapa file WAR, atur file WAR menggunakan struktur berikut.

```
MyApplication.zip
### .ebextensions
### .platform
### foo.war
### bar.war
### ROOT.war
```

Saat Anda menerapkan bundel sumber yang berisi beberapa file WAR ke AWS Elastic Beanstalk lingkungan, setiap aplikasi dapat diakses dari jalur berbeda dari nama domain root. Contoh sebelumnya mencakup tiga aplikasi: `foo`, `bar`, dan `ROOT`. `ROOT.war` adalah nama file khusus yang memberitahu Elastic Beanstalk untuk menjalankan aplikasi di akar domain, sehingga tiga aplikasi tersedia di `http://MyApplication.elasticbeanstalk.com/foo`, `http://MyApplication.elasticbeanstalk.com/bar`, dan `http://MyApplication.elasticbeanstalk.com`.

Paket sumber dapat mencakup file WAR, folder `.ebextensions` opsional, dan folder `.platform` opsional. Untuk detail tentang folder konfigurasi opsional ini, lihat the section called “Memperluas platform Linux”.

Untuk meluncurkan lingkungan (konsol)

1. Buka konsol Elastic Beanstalk dengan tautan yang telah dikonfigurasi sebelumnya: console.aws.amazon. com/elasticbeanstalk/home#/newApplication?applicationName=Tutorial&EnvironmentType= LoadBalanced
2. Untuk Platform, pilih platform dan cabang platform yang sesuai dengan bahasa yang digunakan oleh aplikasi Anda, atau platform Docker untuk aplikasi berbasis kontainer.

3. Untuk Kode aplikasi, pilih Unggah kode Anda.
4. Pilih File lokal, pilih Pilih file, dan kemudian buka paket sumber.
5. Pilih Tinjau dan Luncurkan.
6. Tinjau pengaturan yang tersedia, dan kemudian pilih Buat aplikasi.

Untuk informasi selengkapnya tentang membuat paket sumber, lihat Buat bundel sumber aplikasi Elastic Beanstalk.

Menata folder proyek Anda

Untuk bekerja saat men-deploy ke server Tomcat, arsip aplikasi web (file WAR) Java Platform Enterprise Edition (Java EE) yang dikompilasi harus terstruktur sesuai dengan pedoman. Direktori proyek Anda tidak harus memenuhi standar yang sama, tetapi sebaiknya Anda menyusunnya dengan cara yang sama untuk menyederhanakan kompilasi dan pengemasan. Penataan folder proyek Anda seperti konten file WAR juga membantu Anda memahami bagaimana file terkait dan bagaimana mereka berperilaku di web server.

Di hirarki yang direkomendasikan berikut ini, kode sumber untuk aplikasi web ditempatkan di direktori `src`, untuk mengisolasinya dari penulisan membangun dan file WAR yang dihasilkannya.

```
~/workspace/my-app/
|-- build.sh            - Build script that compiles classes and creates a WAR
|-- README.MD           - Readme file with information about your project, notes
|-- ROOT.war            - Source bundle artifact created by build.sh
`-- src                 - Source code folder
    |-- WEB-INF         - Folder for private supporting files
    |   |-- classes     - Compiled classes
    |   |-- lib         - JAR libraries
    |   |-- tags        - Tag files
    |   |-- tlds        - Tag Library Descriptor files
    |   `-- web.xml     - Deployment Descriptor
    |-- com             - Uncompiled classes
    |-- css             - Style sheets
    |-- images          - Image files
    |-- js              - JavaScript files
    `-- default.jsp     - JSP (JavaServer Pages) webpage
```

Konten folder `src` sesuai dengan apa yang akan Anda kemas dan deploy ke server, dengan pengecualian folder `com`. Folder `com` berisi kelas-kelas Anda yang tidak dikompilasi (file `.java`). Ini

perlu dikompilasi dan ditempatkan di direktori `WEB-INF/classes` untuk dapat diakses dari kode aplikasi Anda.

Direktori `WEB-INF` berisi kode dan konfigurasi yang tidak dilayani secara publik di web server. Folder lain di akar direktori sumber (`css`, `images`, dan `js`) tersedia untuk umum di jalur yang sesuai di web server.

Contoh berikut identik dengan direktori proyek sebelumnya, kecuali direktori berisi lebih banyak file dan subdirektori. Proyek contoh ini mencakup tanda sederhana, model dan dukungan kelas, dan file Java Server Page (JSP) untuk sumber daya `record`. Ini juga mencakup style sheet dan JavaScript untuk Bootstrap, file JSP default, dan halaman kesalahan untuk kesalahan 404.

`WEB-INF/lib` termasuk file Java Archive (JAR) yang berisi driver Java Database Connectivity (JDBC) untuk PostgreSQL. `WEB-INF/classes` kosong karena file kelas belum dikompilasi.

```
~/workspace/my-app/
|-- build.sh
|-- README.MD
|-- ROOT.war
`-- src
    |-- WEB-INF
    |   |-- classes
    |   |-- lib
    |   |   `-- postgresql-9.4-1201.jdbc4.jar
    |   |-- tags
    |   |   `-- header.tag
    |   |-- tlds
    |   |   `-- records.tld
    |   `-- web.xml
    |-- com
    |   `-- myapp
    |       |-- model
    |       |   `-- Record.java
    |       `-- web
    |           `-- ListRecords.java
    |-- css
    |   |-- bootstrap.min.css
    |   `-- myapp.css
    |-- images
    |   `-- myapp.png
    |-- js
    |   `-- bootstrap.min.js
```

```
    |-- 404.jsp
    |-- default.jsp
    `-- records.jsp
```

Membangun file WAR dengan penulisan shell

`build.sh` adalah penulisan shell yang sangat sederhana yang mengompilasi kelas Java, membangun sebuah file WAR, dan menyalin ke direktori `webapps` Tomcat untuk pengujian lokal.

```
cd src
javac -d WEB-INF/classes com/myapp/model/Record.java
javac -classpath WEB-INF/lib/*:WEB-INF/classes -d WEB-INF/classes com/myapp/model/
Record.java
javac -classpath WEB-INF/lib/*:WEB-INF/classes -d WEB-INF/classes com/myapp/web/
ListRecords.java

jar -cvf ROOT.war *.jsp images css js WEB-INF
cp ROOT.war /Library/Tomcat/webapps
mv ROOT.war ../
```

Di dalam file WAR, Anda menemukan struktur yang sama yang ada di direktori `src` di contoh sebelumnya, tidak termasuk `src/com`. Perintah `jar` secara otomatis membuat file `META-INF/MANIFEST.MF`.

```
~/workspace/my-app/ROOT.war
|-- META-INF
|   `-- MANIFEST.MF
|-- WEB-INF
|   |-- classes
|   |   `-- com
|   |       `-- myapp
|   |           |-- model
|   |           |   `-- Records.class
|   |           `-- web
|   |               `-- ListRecords.class
|   |-- lib
|   |   `-- postgresql-9.4-1201.jdbc4.jar
|   |-- tags
|   |   `-- header.tag
|   |-- tlds
|   |   `-- records.tld
|   `-- web.xml
```

```
|-- css
|   |-- bootstrap.min.css
|   `-- myapp.css
|-- images
|   `-- myapp.png
|-- js
|   `-- bootstrap.min.js
|-- 404.jsp
|-- default.jsp
`-- records.jsp
```

Menggunakan `.gitignore`

Untuk menghindari melakukan kompilasi file kelas dan file WAR ke repositori Git Anda, atau melihat pesan tentang mereka muncul ketika Anda menjalankan perintah Git, tambahkan tipe file yang relevan ke file bernama `.gitignore` di folder proyek Anda.

~/ruang kerja/myapp/.gitignore

```
*.zip
*.class
```

Mengonfigurasi server proksi

Platform Tomcat menggunakan nginx (default) atau Server HTTP Apache sebagai proksi terbalik untuk meneruskan permintaan dari port 80 di instans ke kontainer web Tomcat Anda yang mendengarkan port 8080. Elastic Beanstalk menyediakan konfigurasi proksi default yang dapat Anda perluas atau sepenuhnya diganti dengan konfigurasi Anda sendiri.

Mengkonfigurasi server proxy pada versi platform Anda

Semua AL2 AL2 023/platform mendukung fitur konfigurasi proxy yang seragam. Untuk informasi selengkapnya tentang mengonfigurasi server proxy pada versi platform Anda yang menjalankan AL2 023/AL2, lihat. Konfigurasi proksi terbalik

Mengonfigurasi proksi di platform Tomcat Amazon Linux AMI (Amazon Linux 2 terdahulu)

Jika lingkungan Tomcat Elastic Beanstalk Anda menggunakan versi platform Amazon Linux AMI (Amazon Linux 2 terdahulu), baca informasi tambahan di bagian ini.

> **Catatan**
>
> - Informasi dalam topik ini hanya berlaku untuk cabang platform berbasis Amazon Linux AMI (AL1). AL2023/ cabang AL2 platform tidak kompatibel dengan versi platform Amazon Linux AMI (AL1) sebelumnya dan memerlukan pengaturan konfigurasi yang berbeda.
> - Pada tanggal 18 Juli 2022, Elastic Beanstalk menetapkan status semua cabang platform berdasarkan Amazon Linux AMI () untuk pensiun. AL1 Untuk informasi selengkapnya tentang migrasi ke cabang platform Amazon Linux 2023 saat ini dan didukung penuh, lihat. Memigrasi aplikasi Elastic Beanstalk Linux Anda ke Amazon Linux 2023 atau Amazon Linux 2

Memilih server proxy untuk lingkungan Tomcat Anda - Amazon Linux AMI () AL1

Versi platform Tomcat berdasarkan Amazon Linux AMI (Amazon Linux 2 terdahulu) menggunakan Apache 2.4 untuk proksi secara default. Anda dapat memilih untuk menggunakan Apache 2.2 atau nginx dengan menyertakan file konfigurasi di kode sumber Anda. Contoh berikut mengonfigurasi Elastic Beanstalk untuk menggunakan nginx.

Example .ebextensions/nginx-proxy.config

```
option_settings:
  aws:elasticbeanstalk:environment:proxy:
    ProxyServer: nginx
```

Migrasi dari Apache 2.2 ke Apache 2.4 - Amazon Linux AMI () AL1

Jika aplikasi Anda dikembangkan untuk Apache 2.2, baca bagian ini untuk mempelajari tentang migrasi ke Apache 2.4.

Dimulai dengan konfigurasi platform versi 3.0.0 Tomcat, yang dirilis dengan Java dengan pembaruan platform Tomcat pada 24 Mei 2018, Apache 2.4 adalah proksi default dari platform Tomcat. File `.conf` Apache 2.4 sebagian besar, tetapi tidak sepenuhnya, kompatibel dengan Apache 2.2. Elastic Beanstalk mencakup file `.conf` default yang bekerja dengan benar dengan setiap versi Apache. Jika aplikasi Anda tidak menyesuaikan konfigurasi Apache, seperti yang dijelaskan di Memperluas dan mengganti konfigurasi Apache default - Amazon Linux AMI () AL1, aplikasi harus bermigrasi ke Apache 2.4 tanpa masalah apa pun.

Jika aplikasi Anda memperluas atau mengganti konfigurasi Apache, Anda mungkin harus membuat beberapa perubahan untuk bermigrasi ke Apache 2.4. Untuk informasi selengkapnya, lihat Meningkatkan ke 2.4 dari 2.2 di situs Apache Software Foundation. Sebagai tindakan sementara, sampai Anda berhasil bermigrasi ke Apache 2.4, Anda dapat memilih untuk menggunakan Apache 2.2 dengan aplikasi Anda dengan menyertakan file konfigurasi di kode sumber Anda.

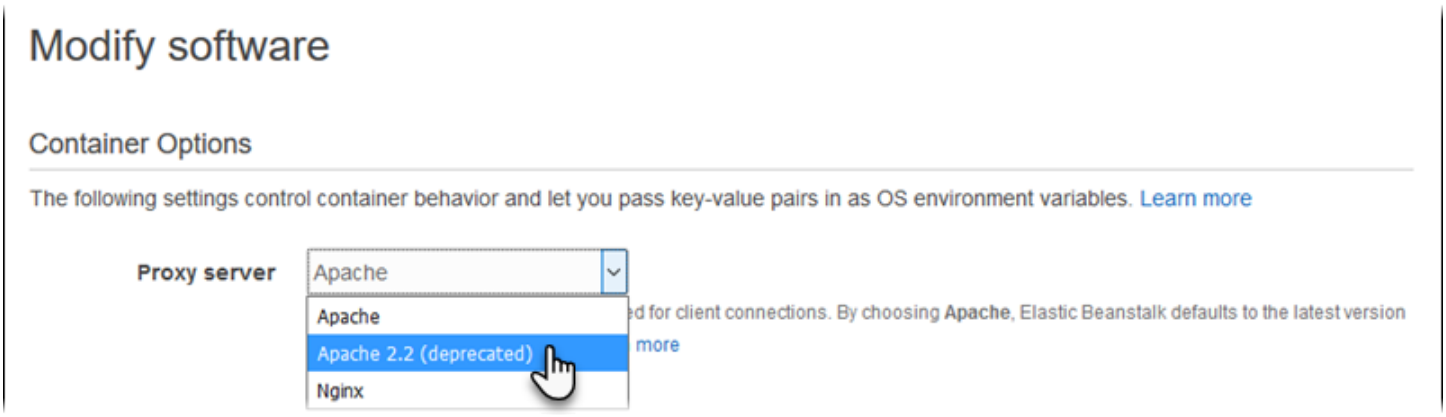
Example .ebextensions/ .config apache-legacy-proxy

```
option_settings:
  aws:elasticbeanstalk:environment:proxy:
    ProxyServer: apache/2.2
```

Untuk perbaikan cepat, Anda juga dapat memilih server proksi di konsol Elastic Beanstalk.

Untuk memilih proksi di lingkungan Tomcat Anda di konsol Elastic Beanstalk

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Dalam kategori konfigurasi Pembaruan, pemantauan, dan pencatatan, pilih Edit.
5. Untuk server proksi, pilih `Apache 2.2 (deprecated)`.
6. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.



Memperluas dan mengganti konfigurasi Apache default - Amazon Linux AMI () AL1

Anda dapat memperpanjang konfigurasi Apache default Elastic Beanstalk dengan file konfigurasi tambahan Anda. Atau, Anda dapat sepenuhnya mengganti konfigurasi Apache default Elastic Beanstalk.

> **Note**
>
> - Semua platform Amazon Linux 2 mendukung fitur konfigurasi proksi yang seragam. Untuk detail tentang mengonfigurasi server proxy pada versi platform Tomcat yang menjalankan Amazon Linux 2, lihat. Konfigurasi proksi terbalik
> - Jika Anda memigrasi aplikasi Elastic Beanstalk Anda ke platform Amazon Linux 2, pastikan untuk juga membaca informasi di the section called “Migrasi ke 023/ AL2 AL2”.

Untuk memperpanjang konfigurasi Apache default Elastic Beanstalk, tambahkan file konfigurasi `.conf` ke folder bernama `.ebextensions/httpd/conf.d` di paket sumber aplikasi Anda. Konfigurasi Apache Elastic Beanstalk mencakup file `.conf` di folder ini secara otomatis.

```
~/workspace/my-app/
|-- .ebextensions
|   -- httpd
|      -- conf.d
|         -- myconf.conf
|         -- ssl.conf
-- index.jsp
```

Sebagai contoh, konfigurasi Apache 2.4 berikut menambahkan listener di port 5000.

Example . ebextensions/httpd/conf.d/port5000.conf

```
listen 5000
<VirtualHost *:5000>
  <Proxy *>
    Require all granted
  </Proxy>
  ProxyPass / http://localhost:8080/ retry=0
  ProxyPassReverse / http://localhost:8080/
  ProxyPreserveHost on
```

```
  ErrorLog /var/log/httpd/elasticbeanstalk-error_log
</VirtualHost>
```

Untuk mengambil alih konfigurasi Apache default Elastic Beanstalk sepenuhnya, sertakan konfigurasi di paket sumber Anda di `.ebextensions/httpd/conf/httpd.conf`.

```
~/workspace/my-app/
|-- .ebextensions
|   `-- httpd
|       `-- conf
|           `-- httpd.conf
`-- index.jsp
```

Jika Anda mengganti konfigurasi Apache Elastic Beanstalk, tambahkan baris berikut ke `httpd.conf` untuk menarik konfigurasi Elastic Beanstalk untuk Peningkatan pelaporan dan pemantauan kesehatan di Elastic Beanstalk, kompresi respons, dan file statis.

```
IncludeOptional conf.d/*.conf
IncludeOptional conf.d/elasticbeanstalk/*.conf
```

Jika lingkungan Anda menggunakan Apache 2.2 sebagai proksi, ganti arahan `IncludeOptional` dengan `Include`. Untuk detail tentang perilaku kedua arahan ini dalam dua versi Apache, lihat Sertakan di Apache 2.4, di Apache 2.4, dan Sertakan IncludeOptional di Apache 2.2.

> **Note**
>
> Untuk mengganti listener default di port 80, termasuk sebuah file bernama `00_application.conf` di `.ebextensions/httpd/conf.d/elasticbeanstalk/` untuk menimpa konfigurasi Elastic Beanstalk.

Untuk contoh kerja, lihatlah file konfigurasi default Elastic Beanstalk di `/etc/httpd/conf/httpd.conf` pada instans di lingkungan Anda. Semua file di folder `.ebextensions/httpd` di paket sumber Anda akan disalin ke `/etc/httpd` selama deployment.

Memperluas konfigurasi nginx default - Amazon Linux AMI () AL1

Untuk memperpanjang konfigurasi nginx default Elastic beanstalk, tambahkan file konfigurasi `.conf` ke folder bernama `.ebextensions/nginx/conf.d/` di paket sumber aplikasi Anda. Konfigurasi nginx Elastic Beanstalk mencakup file `.conf` di folder ini secara otomatis.

```
~/workspace/my-app/
|-- .ebextensions
|   `-- nginx
|       `-- conf.d
|           |-- elasticbeanstalk
|           |   `-- my-server-conf.conf
|           `-- my-http-conf.conf
`-- index.jsp
```

File dengan ekstensi .conf di folder `conf.d` yang disertakan di blok `http` dari konfigurasi default. File di folder `conf.d/elasticbeanstalk` yang disertakan di blok `server` di dalam blok `http`.

Untuk mengganti konfigurasi nginx default Elastic Beanstalk sepenuhnya, sertakan konfigurasi di paket sumber Anda di `.ebextensions/nginx/nginx.conf`.

```
~/workspace/my-app/
|-- .ebextensions
|   `-- nginx
|       `-- nginx.conf
`-- index.jsp
```

Catatan

- Jika Anda mengganti konfigurasi nginx Elastic Beanstalk, tambahkan baris berikut ke blok `server` konfigurasi Anda untuk menarik konfigurasi Elastic Beanstalk untuk listener port 80, kompresi respons, dan file statis.

  ```
  include conf.d/elasticbeanstalk/*.conf;
  ```

- Untuk mengganti listener default di port 80, termasuk sebuah file bernama `00_application.conf` di `.ebextensions/nginx/conf.d/elasticbeanstalk/` untuk menimpa konfigurasi Elastic Beanstalk.
- Juga sertakan baris berikut di blok `http` konfigurasi Anda untuk menarik dalam konfigurasi Elastic Beanstalk untuk Peningkatan pelaporan dan pemantauan kesehatan di Elastic Beanstalk dan logging.

  ```
      include       conf.d/*.conf;
  ```

Untuk contoh kerja, lihatlah file konfigurasi default Elastic Beanstalk di `/etc/nginx/nginx.conf` pada instans di lingkungan Anda. Semua file di folder `.ebextensions/nginx` di paket sumber Anda akan disalin ke `/etc/nginx` selama deployment.

Menggunakan platform Java SE Elastic Beanstalk

Topik ini menjelaskan cara mengkonfigurasi, membangun, dan menjalankan aplikasi Java Anda yang berjalan di platform AWS Elastic Beanstalk Java SE.

Platform Elastic Beanstalk Java SE adalah seperangkat versi platform untuk aplikasi web Java yang dapat berjalan sendiri dari file JAR yang dikompilasi. Anda dapat mengompilasi aplikasi Anda secara lokal atau mengunggah kode sumber dengan penulisan membangun untuk mengompilasinya pada instans. Versi platform Java SE dikelompokkan ke dalam cabang platform, yang masing-masing sesuai dengan versi utama Java.

> **Note**
>
> Elastic Beanstalk tidak mengurai file JAR aplikasi Anda. Jauhkan file yang dibutuhkan Elastic Beanstalk di luar file JAR. Sebagai contoh, sertakan file `cron.yaml` lingkungan pekerja di akar paket sumber aplikasi Anda, di sebelah file JAR.

Opsi konfigurasi tersedia di konsol Elastic Beanstalk untuk memodifikasi konfigurasi dari lingkungan yang sedang berjalan. Agar Anda tidak kehilangan konfigurasi lingkungan ketika mengakhirinya, Anda dapat menggunakan konfigurasi tersimpan untuk menyimpan pengaturan dan kemudian menerapkannya ke lingkungan lain.

Untuk menyimpan pengaturan di kode sumber, Anda dapat menyertakan file konfigurasi. Pengaturan di file konfigurasi diterapkan setiap kali Anda membuat lingkungan atau men-deploy aplikasi Anda. Anda juga dapat menggunakan file konfigurasi untuk menginstal paket, menjalankan penulisan, dan melakukan operasi penyesuaian instans lainnya selama deployment.

Platform Java SE Elastic Beanstalk mencakup server nginx yang bertindak sebagai proksi terbalik, melayani cache konten statis dan meneruskan permintaan ke aplikasi Anda. Platform menyediakan opsi konfigurasi untuk mengonfigurasi server proksi untuk melayani aset statis dari folder di kode sumber Anda untuk mengurangi beban di aplikasi Anda. Untuk skenario lanjutan, Anda dapat menyertakan file .conf milik Anda sendiri di paket sumber Anda untuk memperluas konfigurasi proksi Elastic Beanstalk atau menimpa sepenuhnya.

Jika Anda hanya menyediakan satu file JAR untuk sumber aplikasi Anda (sendiri, tidak di paket sumber), Elastic Beanstalk mengganti nama file JAR Anda menjadi `application.jar`, dan kemudian menjalankannya menggunakan `java -jar application.jar`. Untuk mengonfigurasi proses yang berjalan di instans server di lingkungan Anda, sertakan Procfile opsional di paket sumber Anda. `Procfile` diperlukan jika Anda memiliki lebih dari satu JAR di akar paket sumber Anda, atau jika Anda ingin menyesuaikan perintah java untuk mengatur opsi JVM.

Kami merekomendasikan agar Anda selalu menyediakan `Procfile` di paket sumber di samping aplikasi Anda. Dengan cara ini, Anda secara tepat mengontrol proses mana yang dijalankan Elastic Beanstalk untuk aplikasi Anda dan argumen yang diterima proses ini.

Untuk mengompilasi kelas Java dan menjalankan perintah build lainnya pada EC2 instance di lingkungan Anda pada waktu penerapan, sertakan Buildfile dalam bundel sumber aplikasi Anda. A `Buildfile` memungkinkan Anda menerapkan kode sumber apa adanya dan membangun di server alih-alih mengkompilasi JARs secara lokal. Platform Java SE termasuk alat membangun umum untuk membiarkan Anda membangun pada server.

Untuk detail tentang berbagai cara memperluas platform berbasis Linux Elastic Beanstalk, lihat the section called “Memperluas platform Linux”.

Mengonfigurasi lingkungan Java SE Anda

Pengaturan platform Java SE memungkinkan Anda menyempurnakan perilaku instans Amazon EC2 Anda. Anda dapat mengedit konfigurasi instans Amazon lingkungan Elastic Beanstalk menggunakan konsol EC2 Elastic Beanstalk.

Gunakan konsol Elastic Beanstalk untuk mengaktifkan rotasi log ke Amazon S3 dan mengonfigurasi variabel yang dapat dibaca aplikasi Anda dari lingkungan.

Untuk mengonfigurasi lingkungan Java SE Anda di konsol Elastic Beanstalk

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Dalam kategori konfigurasi Pembaruan, pemantauan, dan pencatatan, pilih Edit.

Opsi log

Bagian Opsi Log memiliki dua pengaturan:

- Profil instans – Menentukan profil instans yang memiliki izin untuk mengakses bucket Amazon S3 yang terkait dengan aplikasi Anda.
- Aktifkan rotasi file log ke Amazon S3 — Menentukan apakah file log untuk EC2 instans Amazon aplikasi Anda disalin ke bucket Amazon S3 yang terkait dengan aplikasi Anda.

File statis

Untuk meningkatkan kinerja, Anda dapat menggunakan bagian File statis untuk mengkonfigurasi server proxy untuk melayani file statis (misalnya, HTML atau gambar) dari satu set direktori di dalam aplikasi web Anda. Untuk setiap direktori, Anda mengatur jalur virtual ke pemetaan direktori. Saat server proksi menerima permintaan untuk file di jalur yang ditentukan, server langsung menyajikan file daripada merutekan permintaan ke aplikasi Anda.

Untuk detail tentang mengonfigurasi file statis menggunakan file konfigurasi atau konsol Elastic Beanstalk, lihat. the section called “File statis”

Properti lingkungan

Bagian Properti Lingkungan memungkinkan Anda menentukan pengaturan konfigurasi lingkungan pada EC2 instance Amazon yang menjalankan aplikasi Anda. Properti lingkungan diberikan sebagai pasangan nilai kunci ke aplikasi.

Di dalam lingkungan Java SE yang berjalan di Elastic Beanstalk, variabel lingkungan dapat diakses menggunakan `System.getenv()`. Sebagai contoh, Anda dapat membaca properti bernama `API_ENDPOINT` ke variabel dengan kode berikut:

```
String endpoint = System.getenv("API_ENDPOINT");
```

Lihat Variabel lingkungan dan pengaturan perangkat lunak lainnya untuk informasi selengkapnya.

Namespace konfigurasi Java SE

Anda dapat menggunakan file konfigurasi untuk mengatur opsi konfigurasi dan melakukan tugas-tugas konfigurasi instans lain selama deployment. Opsi konfigurasi dapat berupa platform khusus

atau berlaku untuk semua platform di layanan Elastic Beanstalk secara keseluruhan. Opsi konfigurasi diatur ke dalam ruang nama.

Platform Java SE tidak menentukan namespace tertentu platform manapun. Anda dapat mengonfigurasi proksi untuk menyajikan file statis dengan menggunakan namespace `aws:elasticbeanstalk:environment:proxy:staticfiles`. Untuk detail dan contoh, lihat the section called “File statis”.

Elastic Beanstalk memberikan banyak opsi konfigurasi untuk menyesuaikan lingkungan Anda. Selain file konfigurasi, Anda juga dapat mengatur opsi konfigurasi menggunakan konsol tersebut, konfigurasi tersimpan, EB CLI, atau AWS CLI. Lihat Opsi konfigurasi untuk informasi selengkapnya.

Platform Java SE Amazon Linux AMI (Amazon Linux 2 terdahulu)

Jika lingkungan Java SE Elastic Beanstalk Anda menggunakan versi platform Amazon Linux AMI (Amazon Linux 2 terdahulu), baca informasi tambahan di bagian ini.

> **Catatan**
>
> - Informasi dalam topik ini hanya berlaku untuk cabang platform berbasis Amazon Linux AMI (AL1). AL2023/ cabang AL2 platform tidak kompatibel dengan versi platform Amazon Linux AMI (AL1) sebelumnya dan memerlukan pengaturan konfigurasi yang berbeda.
> - Pada tanggal 18 Juli 2022, Elastic Beanstalk menetapkan status semua cabang platform berdasarkan Amazon Linux AMI () untuk pensiun. AL1 Untuk informasi selengkapnya tentang migrasi ke cabang platform Amazon Linux 2023 saat ini dan didukung penuh, lihat. Memigrasi aplikasi Elastic Beanstalk Linux Anda ke Amazon Linux 2023 atau Amazon Linux 2

Ruang nama konfigurasi Java SE - Amazon Linux AMI () AL1

Anda dapat menggunakan file konfigurasi untuk mengatur opsi konfigurasi dan melakukan tugas-tugas konfigurasi instans lain selama deployment. Opsi konfigurasi dapat berupa platform khusus atau berlaku untuk semua platform di layanan Elastic Beanstalk secara keseluruhan. Opsi konfigurasi diatur ke dalam ruang nama.

Platform Java SE mendukung satu namespace konfigurasi tertentu platform selain namespace yang didukung oleh semua platform. Namespace

`aws:elasticbeanstalk:container:java:staticfiles` memungkinkan Anda menentukan opsi yang memetakan jalur di aplikasi web Anda ke folder di paket sumber aplikasi Anda yang berisi konten statis.

Sebagai contoh, snippet option_settings menentukan dua opsi di namespace file statis. Yang pertama memetakan jalur `/public` ke folder bernama `public`, dan yang kedua memetakan jalur `/images` ke folder bernama `img`:

```
option_settings:
  aws:elasticbeanstalk:container:java:staticfiles:
    /html: statichtml
    /images: staticimages
```

Folder yang Anda petakan menggunakan namespace ini harus folder yang sebenarnya di akar paket sumber Anda. Anda tidak dapat memetakan jalur ke folder di file JAR.

Elastic Beanstalk memberikan banyak opsi konfigurasi untuk menyesuaikan lingkungan Anda. Selain file konfigurasi, Anda juga dapat mengatur opsi konfigurasi menggunakan konsol tersebut, konfigurasi tersimpan, EB CLI, atau AWS CLI. Lihat Opsi konfigurasi untuk informasi selengkapnya.

Membangun JARs di server dengan Buildfile

Anda dapat membangun file kelas aplikasi dan JAR (s) pada EC2 instance di lingkungan Anda dengan menjalankan perintah build dari `Buildfile` file dalam bundel sumber Anda.

Perintah di `Buildfile` hanya berjalan sekali dan harus diakhiri setelah selesai, sedangkan perintah di Procfile diharapkan untuk menjalankan demi kehidupan aplikasi dan akan dimulai ulang jika mereka mengakhirinya. Untuk menjalankan aplikasi Anda, gunakan file`Procfile`. JARs

Untuk detail tentang penempatan dan sintaks a`Buildfile`, lihatBuildfile dan Procfile.

Contoh `Buildfile` berikut ini menjalankan Apache Maven untuk membangun aplikasi web dari kode sumber. Untuk aplikasi sampel yang menggunakan fitur ini, lihat sampel aplikasi web Java.

Example Buildfile

```
build: mvn assembly:assembly -DdescriptorId=jar-with-dependencies
```

Platform Java SE mencakup alat membangun berikut, yang dapat Anda panggil dari penulisan membangun Anda:

- `javac` – Java compiler
- `ant` – Apache Ant
- `mvn` – Apache Maven
- `gradle` – Gradle

Mengonfigurasi proses aplikasi dengan Procfile

Jika Anda memiliki lebih dari satu file JAR di akar paket sumber aplikasi Anda, Anda harus menyertakan file `Procfile` yang memberitahu Elastic Beanstalk JAR mana yang dijalankan. Anda juga dapat menyertakan file `Procfile` untuk satu aplikasi JAR untuk mengonfigurasi mesin virtual Java (JVM) yang menjalankan aplikasi Anda.

Kami merekomendasikan agar Anda selalu menyediakan `Procfile` di paket sumber di samping aplikasi Anda. Dengan cara ini, Anda secara tepat mengontrol proses mana yang dijalankan Elastic Beanstalk untuk aplikasi Anda dan argumen yang diterima proses ini.

Untuk detail tentang menulis dan menggunakan `Procfile` lihatBuildfile dan Procfile.

Example Procfile

```
web: java -Xms256m -jar server.jar
cache: java -jar mycache.jar
web_foo: java -jar other.jar
```

Perintah yang menjalankan JAR utama di aplikasi Anda harus disebut `web`, dan menjadi perintah pertama yang tercantum di `Procfile`. Server nginx meneruskan semua permintaan HTTP yang diterima dari penyeimbang beban lingkungan Anda ke aplikasi ini.

Elastic Beanstalk mengasumsikan bahwa semua entri di Procfile harus berjalan setiap saat dan secara otomatis memulai ulang aplikasi yang ditentukan di Procfile yang berakhir. Untuk menjalankan perintah yang akan mengakhiri dan tidak harus dimulai ulang, gunakan `Buildfile`.

Menggunakan Procfile di Amazon Linux AMI (Amazon Linux 2 terdahulu)

Jika lingkungan Java SE Elastic Beanstalk Anda menggunakan versi platform Amazon Linux AMI (Amazon Linux 2 terdahulu), baca informasi tambahan di bagian ini.

> **Catatan**
>
> - Informasi dalam topik ini hanya berlaku untuk cabang platform berbasis Amazon Linux AMI (AL1). AL2023/ cabang AL2 platform tidak kompatibel dengan versi platform Amazon Linux AMI (AL1) sebelumnya dan memerlukan pengaturan konfigurasi yang berbeda.
> - Pada tanggal 18 Juli 2022, Elastic Beanstalk menetapkan status semua cabang platform berdasarkan Amazon Linux AMI () untuk pensiun. AL1 Untuk informasi selengkapnya tentang migrasi ke cabang platform Amazon Linux 2023 saat ini dan didukung penuh, lihat. Memigrasi aplikasi Elastic Beanstalk Linux Anda ke Amazon Linux 2023 atau Amazon Linux 2

Port lewat - Amazon Linux AMI (AL1)

Secara default, Elastic Beanstalk mengonfigurasi proksi nginx untuk meneruskan permintaan ke aplikasi Anda di port 5000. Anda dapat mengganti port default dengan mengatur `PORT` properti lingkungan ke port yang didengarkan aplikasi utama Anda.

Jika Anda menggunakan `Procfile` untuk menjalankan beberapa aplikasi, Elastic Beanstalk di versi platform Amazon Linux AMI mengharapkan setiap aplikasi tambahan untuk mendengarkan di port 100 yang lebih tinggi dari sebelumnya. Elastic Beanstalk mengatur variabel PORT dapat diakses dari dalam setiap aplikasi ke port yang mengharapkan untuk menjalankan aplikasi. Anda dapat mengakses variabel ini dalam kode aplikasi Anda dengan memanggil `System.getenv("PORT")`.

Pada contoh `Procfile` sebelumnya, aplikasi `web` mendengarkan di port 5000, `cache` mendengarkan di port 5100, dan `web_foo` mendengarkan di port 5200. `web` mengonfigurasi port mendengarkannya dengan membaca variabel `PORT`, dan menambahkan 100 ke jumlah tersebut untuk menentukan port `cache` mana yang mendengarkan sehingga dapat mengirimkan permintaan.

Mengonfigurasi server proksi

Elastic Beanstalk menggunakan nginx sebagai proksi terbalik untuk memetakan aplikasi Anda ke penyeimbang beban Elastic Load Balancing di port 80. Elastic Beanstalk menyediakan konfigurasi nginx default yang dapat Anda perluas atau sepenuhnya diganti dengan konfigurasi Anda sendiri.

Secara default, Elastic Beanstalk mengonfigurasi proksi nginx untuk meneruskan permintaan ke aplikasi Anda di port 5000. Anda dapat mengganti port default dengan mengatur `PORT` properti lingkungan ke port yang didengarkan aplikasi utama Anda.

> **Note**
>
> Port yang didengar aplikasi Anda tidak mempengaruhi port yang didengar server nginx untuk menerima permintaan dari penyeimbang beban.

Mengkonfigurasi server proxy pada versi platform Anda

Semua AL2 AL2 023/platform mendukung fitur konfigurasi proxy yang seragam. Untuk informasi selengkapnya tentang mengonfigurasi server proxy pada versi platform Anda yang menjalankan AL2 023/AL2, lihat. Konfigurasi proksi terbalik

Mengonfigurasi proksi di Amazon Linux AMI (Amazon Linux 2 terdahulu)

Jika lingkungan Java SE Elastic Beanstalk Anda menggunakan versi platform Amazon Linux AMI (Amazon Linux 2 terdahulu), baca informasi tambahan di bagian ini.

> **Catatan**
>
> - Informasi dalam topik ini hanya berlaku untuk cabang platform berbasis Amazon Linux AMI (AL1). AL2023/ cabang AL2 platform tidak kompatibel dengan versi platform Amazon Linux AMI (AL1) sebelumnya dan memerlukan pengaturan konfigurasi yang berbeda.
> - Pada tanggal 18 Juli 2022, Elastic Beanstalk menetapkan status semua cabang platform berdasarkan Amazon Linux AMI () untuk pensiun. AL1 Untuk informasi selengkapnya tentang migrasi ke cabang platform Amazon Linux 2023 saat ini dan didukung penuh, lihat. Memigrasi aplikasi Elastic Beanstalk Linux Anda ke Amazon Linux 2023 atau Amazon Linux 2

Memperluas dan mengganti konfigurasi proxy default — Amazon Linux AMI () AL1

Untuk memperpanjang konfigurasi nginx default Elastic beanstalk, tambahkan file konfigurasi `.conf` ke folder bernama `.ebextensions/nginx/conf.d/` di paket sumber aplikasi Anda. Konfigurasi nginx Elastic Beanstalk mencakup file `.conf` di folder ini secara otomatis.

```
~/workspace/my-app/
|-- .ebextensions
|   `-- nginx
|       `-- conf.d
```

```
|           `-- myconf.conf
`-- web.jar
```

Untuk mengganti konfigurasi nginx default Elastic Beanstalk sepenuhnya, sertakan konfigurasi di paket sumber Anda di `.ebextensions/nginx/nginx.conf`:

```
~/workspace/my-app/
|-- .ebextensions
|   `-- nginx
|       `-- nginx.conf
`-- web.jar
```

Jika Anda mengganti konfigurasi nginx Elastic Beanstalk, tambahkan baris berikut ke `nginx.conf` untuk menarik konfigurasi Elastic Beanstalk untuk Peningkatan pelaporan dan pemantauan kesehatan di Elastic Beanstalk, pemetaan aplikasi otomatis, dan file statis.

```
 include conf.d/elasticbeanstalk/*.conf;
```

Contoh konfigurasi berikut dari Aplikasi sampel Scorekeep mengganti konfigurasi default Elastic Beanstalk untuk melayani aplikasi web statis dari subdirektori `public` dari `/var/app/current`, saat platform Java SE menyalin kode sumber aplikasi. lokasi `/api` meneruskan lalu lintas ke rute di bawah `/api/` ke aplikasi Spring yang mendengarkan di port 5000. Semua lalu lintas lainnya dilayani oleh aplikasi web di jalur akar.

Example

```
user                    nginx;
error_log               /var/log/nginx/error.log warn;
pid                     /var/run/nginx.pid;
worker_processes        auto;
worker_rlimit_nofile    33282;

events {
    worker_connections  1024;
}

http {
  include       /etc/nginx/mime.types;
  default_type  application/octet-stream;

  log_format  main  '$remote_addr - $remote_user [$time_local] "$request" '
```

```
                    '$status $body_bytes_sent "$http_referer" '
                    '"$http_user_agent" "$http_x_forwarded_for"';

    include       conf.d/*.conf;

    map $http_upgrade $connection_upgrade {
        default     "upgrade";
    }

    server {
        listen        80 default_server;
        root /var/app/current/public;

        location / {
        }git pull


        location /api {
            proxy_pass          http://127.0.0.1:5000;
            proxy_http_version  1.1;

            proxy_set_header    Connection          $connection_upgrade;
            proxy_set_header    Upgrade             $http_upgrade;
            proxy_set_header    Host                $host;
            proxy_set_header    X-Real-IP           $remote_addr;
            proxy_set_header    X-Forwarded-For     $proxy_add_x_forwarded_for;
        }

        access_log    /var/log/nginx/access.log main;

        client_header_timeout 60;
        client_body_timeout   60;
        keepalive_timeout     60;
        gzip                  off;
        gzip_comp_level       4;

        # Include the Elastic Beanstalk generated locations
        include conf.d/elasticbeanstalk/01_static.conf;
        include conf.d/elasticbeanstalk/healthd.conf;
    }
}
```

Menambahkan instans Amazon RDS DB ke lingkungan Java Elastic Beanstalk Anda

Topik ini memberikan instruksi untuk membuat Amazon RDS menggunakan konsol Elastic Beanstalk. Anda dapat menggunakan instans DB Amazon Relational Database Service (Amazon RDS) untuk menyimpan data yang dikumpulkan dan dimodifikasi oleh aplikasi Anda. Basis data dapat dilampirkan ke lingkungan Anda dan dikelola oleh Elastic Beanstalk, atau dibuat dan dikelola secara eksternal.

Jika Anda menggunakan Amazon RDS untuk pertama kalinya, tambahkan instans DB ke lingkungan uji dengan menggunakan konsol Elastic Beanstalk dan verifikasi apakah aplikasi Anda dapat terhubung ke sana.

Untuk menambahkan instans DB ke lingkungan Anda

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Di kategori konfigurasi Basis data, pilih Edit.
5. Pilih mesin DB, dan masukkan nama pengguna dan kata sandi.
6. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Menambahkan instans DB memakan waktu sekitar 10 menit. Ketika pembaruan lingkungan selesai, nama host instans DB dan informasi koneksi lainnya tersedia untuk aplikasi Anda melalui properti lingkungan berikut:

Nama properti	Deskripsi	Nilai properti
`RDS_HOSTNAME`	Nama host instans DB.	Di tab Konektivitas & keamanan di konsol Amazon RDS: Titik akhir.
`RDS_PORT`	Port tempat instans DB menerima koneksi. Nilai default bervariasi di antara mesin DB.	Di tab Konektivitas & keamanan di konsol Amazon RDS: Port.

Nama properti	Deskripsi	Nilai properti
`RDS_DB_NAME`	Nama basis data, **ebdb**.	Di tab Konfigurasi di konsol Amazon RDS: Nama DB.
`RDS_USERNAME`	Nama pengguna yang Anda konfigurasi untuk basis data Anda.	Di tab Konfigurasi di konsol Amazon RDS: Nama pengguna utama.
`RDS_PASSWORD`	Kata sandi yang Anda konfigurasi untuk basis data Anda.	Tidak tersedia untuk referensi di konsol Amazon RDS.

Untuk informasi selengkapnya tentang mengonfigurasi instans DB internal, lihat Menambahkan basis data ke lingkungan Elastic Beanstalk Anda. Untuk petunjuk tentang konfigurasi basis data eksternal untuk digunakan dengan Elastic Beanstalk, lihat Menggunakan Elastic Beanstalk dengan Amazon RDS.

Untuk terhubung ke basis data, tambahkan driver file JAR yang sesuai ke aplikasi Anda, muat kelas driver di kode Anda, dan buat objek koneksi dengan properti lingkungan yang disediakan oleh Elastic Beanstalk.

Bagian

- Mengunduh driver JDBC
- Menghubungkan ke basis data (platform Java SE)
- Menghubungkan ke basis data (platform Tomcat)
- Pemecahan masalah koneksi basis data

Mengunduh driver JDBC

Anda akan membutuhkan file JAR driver JDBC untuk mesin DB yang Anda pilih. Simpan file JAR di kode sumber Anda dan sertakan ke dalam classpath Anda ketika Anda mengompilasi kelas yang membuat koneksi ke basis data.

Anda dapat menemukan driver terbaru untuk mesin DB Anda di lokasi-lokasi berikut:

- MySQL – MySQL Connector/J

- Oracle SE-1 – Oracle JDBC Driver
- Postgres – PostgreSQL JDBC Driver
- SQL Server – Microsoft JDBC Driver

Untuk menggunakan driver JDBC, panggil `Class.forName()` untuk memuatnya sebelum membuat koneksi dengan `DriverManager.getConnection()` di kode Anda.

JDBC menggunakan string koneksi dengan format berikut:

```
jdbc:driver://hostname:port/dbName?user=userName&password=password
```

Anda dapat mengambil hostname, port, nama basis data, nama pengguna, dan kata sandi dari variabel lingkungan yang disediakan Elastic Beanstalk untuk aplikasi Anda. Nama driver khusus untuk tipe basis data dan versi driver Anda. Berikut ini adalah contoh nama driver:

- `mysql` untuk MySQL
- `postgresql` untuk PostgreSQL
- `oracle:thin` untuk Oracle Thin
- `oracle:oci` untuk Oracle OCI
- `oracle:oci8` untuk Oracle OCI 8
- `oracle:kprb` untuk Oracle KPRB
- `sqlserver` untuk SQL Server

Menghubungkan ke basis data (platform Java SE)

Di lingkungan Java SE, gunakan `System.getenv()` untuk membaca variabel koneksi dari lingkungan. Contoh kode berikut menunjukkan kelas yang membuat koneksi ke basis data PostgreSQL.

```
private static Connection getRemoteConnection() {
    if (System.getenv("RDS_HOSTNAME") != null) {
      try {
      Class.forName("org.postgresql.Driver");
      String dbName = System.getenv("RDS_DB_NAME");
      String userName = System.getenv("RDS_USERNAME");
      String password = System.getenv("RDS_PASSWORD");
      String hostname = System.getenv("RDS_HOSTNAME");
```

```
      String port = System.getenv("RDS_PORT");
      String jdbcUrl = "jdbc:postgresql://" + hostname + ":" + port + "/" + dbName + "?
user=" + userName + "&password=" + password;
      logger.trace("Getting remote connection with connection string from environment
 variables.");
      Connection con = DriverManager.getConnection(jdbcUrl);
      logger.info("Remote connection successful.");
      return con;
    }
    catch (ClassNotFoundException e) { logger.warn(e.toString());}
    catch (SQLException e) { logger.warn(e.toString());}
    }
    return null;
  }
```

Menghubungkan ke basis data (platform Tomcat)

Di lingkungan Tomcat, properti lingkungan disediakan sebagai properti sistem yang dapat diakses dengan `System.getProperty()`.

Contoh kode berikut menunjukkan kelas yang membuat koneksi ke basis data PostgreSQL.

```
private static Connection getRemoteConnection() {
    if (System.getProperty("RDS_HOSTNAME") != null) {
      try {
      Class.forName("org.postgresql.Driver");
      String dbName = System.getProperty("RDS_DB_NAME");
      String userName = System.getProperty("RDS_USERNAME");
      String password = System.getProperty("RDS_PASSWORD");
      String hostname = System.getProperty("RDS_HOSTNAME");
      String port = System.getProperty("RDS_PORT");
      String jdbcUrl = "jdbc:postgresql://" + hostname + ":" + port + "/" + dbName + "?
user=" + userName + "&password=" + password;
      logger.trace("Getting remote connection with connection string from environment
 variables.");
      Connection con = DriverManager.getConnection(jdbcUrl);
      logger.info("Remote connection successful.");
      return con;
    }
    catch (ClassNotFoundException e) { logger.warn(e.toString());}
    catch (SQLException e) { logger.warn(e.toString());}
    }
    return null;
```

```
  }
```

Jika Anda mengalami kesulitan mendapatkan koneksi atau menjalankan pernyataan SQL, cobalah menempatkan kode berikut di file JSP. Kode ini menghubungkan ke instans DB, membuat tabel, dan menuliskannya.

```
<%@ page import="java.sql.*" %>
<%
  // Read RDS connection information from the environment
  String dbName = System.getProperty("RDS_DB_NAME");
  String userName = System.getProperty("RDS_USERNAME");
  String password = System.getProperty("RDS_PASSWORD");
  String hostname = System.getProperty("RDS_HOSTNAME");
  String port = System.getProperty("RDS_PORT");
  String jdbcUrl = "jdbc:mysql://" + hostname + ":" +
    port + "/" + dbName + "?user=" + userName + "&password=" + password;

  // Load the JDBC driver
  try {
    System.out.println("Loading driver...");
    Class.forName("com.mysql.jdbc.Driver");
    System.out.println("Driver loaded!");
  } catch (ClassNotFoundException e) {
    throw new RuntimeException("Cannot find the driver in the classpath!", e);
  }

  Connection conn = null;
  Statement setupStatement = null;
  Statement readStatement = null;
  ResultSet resultSet = null;
  String results = "";
  int numresults = 0;
  String statement = null;

  try {
    // Create connection to RDS DB instance
    conn = DriverManager.getConnection(jdbcUrl);

    // Create a table and write two rows
    setupStatement = conn.createStatement();
    String createTable = "CREATE TABLE Beanstalk (Resource char(50));";
    String insertRow1 = "INSERT INTO Beanstalk (Resource) VALUES ('EC2 Instance');";
    String insertRow2 = "INSERT INTO Beanstalk (Resource) VALUES ('RDS Instance');";
```

```
    setupStatement.addBatch(createTable);
    setupStatement.addBatch(insertRow1);
    setupStatement.addBatch(insertRow2);
    setupStatement.executeBatch();
    setupStatement.close();

  } catch (SQLException ex) {
    // Handle any errors
    System.out.println("SQLException: " + ex.getMessage());
    System.out.println("SQLState: " + ex.getSQLState());
    System.out.println("VendorError: " + ex.getErrorCode());
  } finally {
    System.out.println("Closing the connection.");
    if (conn != null) try { conn.close(); } catch (SQLException ignore) {}
  }

  try {
    conn = DriverManager.getConnection(jdbcUrl);

    readStatement = conn.createStatement();
    resultSet = readStatement.executeQuery("SELECT Resource FROM Beanstalk;");

    resultSet.first();
    results = resultSet.getString("Resource");
    resultSet.next();
    results += ", " + resultSet.getString("Resource");

    resultSet.close();
    readStatement.close();
    conn.close();

  } catch (SQLException ex) {
    // Handle any errors
    System.out.println("SQLException: " + ex.getMessage());
    System.out.println("SQLState: " + ex.getSQLState());
    System.out.println("VendorError: " + ex.getErrorCode());
  } finally {
       System.out.println("Closing the connection.");
      if (conn != null) try { conn.close(); } catch (SQLException ignore) {}
  }
%>
```

Untuk menampilkan hasil, tempatkan kode berikut di bodi bagian HTML dari file JSP.

```
<p>Established connection to RDS. Read first two rows: <%= results %></p>
```

Pemecahan masalah koneksi basis data

Jika Anda mengalami masalah menghubungkan ke basis data dari dalam aplikasi Anda, tinjau log kontainer web dan basis data.

Meninjau log

Anda dapat melihat semua log dari lingkungan Elastic Beanstalk Anda dari dalam Eclipse. Jika Anda tidak membuka tampilan AWS Explorer, pilih panah di sebelah AWS ikon oranye di bilah alat, lalu pilih Tampilkan Tampilan AWS Explorer. Perluas AWS Elastic Beanstalk dan nama lingkungan Anda, dan kemudian buka menu konteks (klik kanan) untuk server. Pilih Buka di WTP Editor Server.

Pilih tab Log tampilan Server untuk melihat log agregat dari lingkungan Anda. Untuk membuka log terbaru, pilih tombol Segarkan di sudut kanan atas halaman.

Gulir ke bawah untuk menemukan log Tomcat di `/var/log/tomcat7/catalina.out`. Jika Anda memuat halaman web dari contoh kami sebelumnya beberapa kali, Anda mungkin melihat berikut ini.

```
-------------------------------------
/var/log/tomcat7/catalina.out
-------------------------------------
INFO: Server startup in 9285 ms
Loading driver...
Driver loaded!
SQLException: Table 'Beanstalk' already exists
SQLState: 42S01
VendorError: 1050
Closing the connection.
Closing the connection.
```

Semua informasi yang aplikasi web kirim ke output standar muncul di log kontainer web. Di contoh sebelumnya, aplikasi mencoba untuk membuat tabel setiap kali halaman dimuat. Hal ini menyebabkan penangkapan pengecualian SQL di setiap memuat halaman setelah yang pertama.

Sebagai contoh, sebelumnya dapat diterima. Tetapi, di aplikasi yang sebenarnya, jaga definisi basis data Anda di objek skema, lakukan transaksi dari dalam kelas model, dan koordinasikan permintaan dengan pengendali servlet.

Menghubungkan ke instans DB RDS

Anda dapat terhubung langsung ke instans DB RDS di lingkungan Elastic Beanstalk Anda dengan menggunakan aplikasi klien MySQL.

Pertama, buka grup keamanan untuk instans DB RDS Anda untuk mengizinkan lalu lintas dari komputer Anda.

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Di kategori konfigurasi Basis data, pilih Edit.
5. Di samping Titik akhir, pilih tautan konsol Amazon RDS.
6. Di halaman detail instans Dasbor RDS, di bawah Keamanan dan Jaringan, pilih grup keamanan yang dimulai dengan RDS di samping Grup Keamanan.

   > **Note**
   >
   > Basis data mungkin memiliki beberapa entri berlabel Grup Keamanan. Gunakan yang pertama, yang dimulai dengan awseb, hanya jika Anda memiliki akun lama yang tidak memiliki Amazon Virtual Private Cloud (Amazon VPC) default.

7. Di Detail grup keamanan, pilih tab Masuk, dan kemudian pilih Edit.
8. Tambahkan aturan untuk MySQL (port 3306) yang mengizinkan lalu lintas dari alamat IP Anda, ditentukan dalam format CIDR.
9. Pilih Simpan. Perubahan segera berlaku.

Kembali ke detail konfigurasi Elastic Beanstalk untuk lingkungan Anda dan perhatikan titik akhir. Anda akan menggunakan nama domain untuk terhubung ke instans DB RDS.

Instal klien MySQL dan mulai koneksi ke basis data di port 3306. Di Windows, instal MySQL Workbench dari beranda MySQL dan ikuti petunjuknya.

Di Linux, instal klien MySQL menggunakan manajer paket untuk distribusi Anda. Contoh berikut bekerja di Ubuntu dan turunan Debian lainnya.

```
// Install MySQL client
$ sudo apt-get install mysql-client-5.5
...
// Connect to database
$ mysql -h aas839jo2vwhwb.cnubrrfwfka8.us-west-2.rds.amazonaws.com -u username -
ppassword ebdb
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 117
Server version: 5.5.40-log Source distribution
...
```

Setelah Anda terhubung, Anda dapat menjalankan perintah SQL untuk melihat status basis data, apakah tabel dan baris Anda sudah dibuat, dan informasi lainnya.

```
mysql> SELECT Resource from Beanstalk;
+--------------+
| Resource     |
+--------------+
| EC2 Instance |
| RDS Instance |
+--------------+
2 rows in set (0.01 sec)
```

Alat dan sumber daya Java

Ada beberapa tempat yang dapat dikunjungi untuk mendapatkan bantuan tambahan ketika mengembangkan aplikasi Java Anda.

Sumber Daya	Deskripsi
Forum Pengembangan AWS Java	Sampaikan pertanyaan Anda dan dapatkan umpan balik.
Pusat Pengembang Java	Tempat yang lengkap untuk kode sampel, dokumentasi, alat, dan sumber daya tambahan.

Menyebarkan aplikasi Node.js dengan Elastic Beanstalk

Bab ini memberikan instruksi untuk mengkonfigurasi dan menyebarkan aplikasi web Node.js Anda ke. AWS Elastic Beanstalk Ini juga menyediakan panduan untuk tugas-tugas umum seperti integrasi database dan bekerja dengan kerangka kerja Express. Elastic Beanstalk memudahkan penerapan, pengelolaan, dan skala aplikasi web Node.js Anda menggunakan Amazon Web Services.

Anda dapat menerapkan aplikasi Anda hanya dalam beberapa menit menggunakan Elastic Beanstalk Command Line Interface (EB CLI) atau dengan menggunakan konsol Elastic Beanstalk. Setelah Anda menerapkan aplikasi Elastic Beanstalk, Anda dapat terus menggunakan EB CLI untuk mengelola aplikasi dan lingkungan Anda, atau Anda dapat menggunakan konsol Elastic Beanstalk, atau. AWS CLI APIs

Ikuti step-by-step petunjuk QuickStart untuk Node.js untuk membuat dan menyebarkan aplikasi web Hello World Node.js dengan EB CLI.

Topik

- QuickStart: Menyebarkan aplikasi Node.js ke Elastic Beanstalk
- Menyiapkan lingkungan pengembangan Node.js Anda untuk Elastic Beanstalk
- Menggunakan platform Node.js Elastic Beanstalk
- Lebih banyak contoh aplikasi dan tutorial Elastic Beanstalk untuk Node.js
- Menyebarkan aplikasi Node.js Express ke Elastic Beanstalk
- Menyebarkan aplikasi Node.js Express dengan pengelompokan ke Elastic Beanstalk
- Men-deploy aplikasi Node.js dengan DynamoDB ke Elastic Beanstalk
- Menambahkan instans Amazon RDS DB ke lingkungan Node.js Elastic Beanstalk Anda
- Alat dan sumber daya Node.js

QuickStart: Menyebarkan aplikasi Node.js ke Elastic Beanstalk

QuickStart Tutorial ini memandu Anda melalui proses membuat aplikasi Node.js dan menyebarkannya ke AWS Elastic Beanstalk lingkungan.

> **Note**
>
> Contoh tutorial dimaksudkan untuk demonstrasi. Jangan gunakan aplikasi untuk lalu lintas produksi.

Bagian-bagian

- AWS Akun Anda
- Prasyarat
- Langkah 1: Buat aplikasi Node.js
- Langkah 2: Jalankan aplikasi Anda secara lokal
- Langkah 3: Terapkan aplikasi Node.js Anda dengan EB CLI
- Langkah 4: Jalankan aplikasi Anda di Elastic Beanstalk
- Langkah 5: Bersihkan
- AWS sumber daya untuk aplikasi Anda
- Langkah selanjutnya
- Terapkan dengan konsol Elastic Beanstalk

AWS Akun Anda

Jika Anda belum menjadi AWS pelanggan, Anda perlu membuat AWS akun. Mendaftar memungkinkan Anda mengakses Elastic Beanstalk AWS dan layanan lain yang Anda butuhkan.

Jika Anda sudah memiliki AWS akun, Anda dapat melanjutkan kePrasyarat.

Buat AWS akun

Mendaftar untuk Akun AWS

Jika Anda tidak memiliki Akun AWS, selesaikan langkah-langkah berikut untuk membuatnya.

Untuk mendaftar untuk Akun AWS

1. Buka https://portal.aws.amazon.com/billing/pendaftaran.
2. Ikuti petunjuk online.

Bagian dari prosedur pendaftaran melibatkan menerima panggilan telepon atau pesan teks dan memasukkan kode verifikasi pada keypad telepon.

Saat Anda mendaftar untuk sebuah Akun AWS, sebuah Pengguna root akun AWSdibuat. Pengguna root memiliki akses ke semua Layanan AWS dan sumber daya di akun. Sebagai praktik keamanan terbaik, tetapkan akses administratif ke pengguna, dan gunakan hanya pengguna root untuk melakukan tugas yang memerlukan akses pengguna root.

AWS mengirimi Anda email konfirmasi setelah proses pendaftaran selesai. Kapan saja, Anda dapat melihat aktivitas akun Anda saat ini dan mengelola akun Anda dengan masuk https://aws.amazon.comke/ dan memilih Akun Saya.

Buat pengguna dengan akses administratif

Setelah Anda mendaftar Akun AWS, amankan Pengguna root akun AWS, aktifkan AWS IAM Identity Center, dan buat pengguna administratif sehingga Anda tidak menggunakan pengguna root untuk tugas sehari-hari.

Amankan Anda Pengguna root akun AWS

1. Masuk ke AWS Management Consolesebagai pemilik akun dengan memilih pengguna Root dan memasukkan alamat Akun AWS email Anda. Di laman berikutnya, masukkan kata sandi.

   Untuk bantuan masuk dengan menggunakan pengguna root, lihat Masuk sebagai pengguna root di AWS Sign-In Panduan Pengguna.
2. Mengaktifkan autentikasi multi-faktor (MFA) untuk pengguna root Anda.

   Untuk petunjuk, lihat Mengaktifkan perangkat MFA virtual untuk pengguna Akun AWS root (konsol) Anda di Panduan Pengguna IAM.

Buat pengguna dengan akses administratif

1. Aktifkan Pusat Identitas IAM.

   Untuk mendapatkan petunjuk, silakan lihat Mengaktifkan AWS IAM Identity Center di Panduan Pengguna AWS IAM Identity Center .
2. Di Pusat Identitas IAM, berikan akses administratif ke pengguna.

Untuk tutorial tentang menggunakan Direktori Pusat Identitas IAM sebagai sumber identitas Anda, lihat Mengkonfigurasi akses pengguna dengan default Direktori Pusat Identitas IAM di Panduan AWS IAM Identity Center Pengguna.

Masuk sebagai pengguna dengan akses administratif

- Untuk masuk dengan pengguna Pusat Identitas IAM, gunakan URL masuk yang dikirim ke alamat email saat Anda membuat pengguna Pusat Identitas IAM.

  Untuk bantuan masuk menggunakan pengguna Pusat Identitas IAM, lihat Masuk ke portal AWS akses di Panduan AWS Sign-In Pengguna.

Tetapkan akses ke pengguna tambahan

1. Di Pusat Identitas IAM, buat set izin yang mengikuti praktik terbaik menerapkan izin hak istimewa paling sedikit.

   Untuk petunjuknya, lihat Membuat set izin di Panduan AWS IAM Identity Center Pengguna.
2. Tetapkan pengguna ke grup, lalu tetapkan akses masuk tunggal ke grup.

   Untuk petunjuk, lihat Menambahkan grup di Panduan AWS IAM Identity Center Pengguna.

Prasyarat

Untuk mengikuti prosedur di panduan ini, Anda memerlukan terminal atau shell baris perintah untuk menjalankan perintah. Perintah ditampilkan dalam daftar yang diawali dengan simbol prompt ($) dan nama direktori saat ini, bila perlu.

```
~/eb-project$ this is a command
this is output
```

Di Linux dan macOS, Anda dapat menggunakan shell dan manajer paket pilihan Anda. Pada Windows Anda dapat menginstal Windows Subsystem untuk Linux untuk mendapatkan versi Windows terintegrasi dari Ubuntu dan Bash.

EB CLI

Tutorial ini menggunakan Elastic Beanstalk Command Line Interface (EB CLI). Untuk detail tentang pemasangan dan konfigurasi EB CLI, lihat Instal EB CLI dengan skrip pengaturan (disarankan) dan Mengonfigurasi EB CLI.

Node.js

Instal Node.js di mesin lokal Anda dengan mengikuti Cara menginstal Node.js di situs web Node.js.

Verifikasi instalasi Node.js Anda dengan menjalankan perintah berikut.

```
~$ node -v
```

Langkah 1: Buat aplikasi Node.js

Buat direktori proyek.

```
~$ mkdir eb-nodejs
~$ cd eb-nodejs
```

Selanjutnya, buat aplikasi yang akan Anda deploy menggunakan Elastic Beanstalk. Kami akan membuat layanan RESTful web “Hello World”.

Example **~/eb-nodejs/server.js**

```
const http = require('node:http');

const hostname = '127.0.0.1';
const port = 8080;

const server = http.createServer((req, res) => {
  res.statusCode = 200;
  res.setHeader('Content-Type', 'text/plain');
  res.end('Hello Elastic Beanstalk!\n');
});

server.listen(port, hostname, () => {
  console.log(`Server running at http://${hostname}:${port}/`);
});
```

Aplikasi ini membuka pendengar pada port 8080. Elastic Beanstalk meneruskan permintaan ke aplikasi Anda pada port 8080 secara default untuk Node.js.

Langkah 2: Jalankan aplikasi Anda secara lokal

Jalankan perintah berikut untuk menjalankan aplikasi Anda secara lokal.

```
~/eb-nodejs$ node server.js
```

Anda akan melihat teks berikut.

```
Server running at http://127.0.0.1:8080/
```

Masukkan alamat URL `http://127.0.0.1:8080/` di browser web Anda. Browser harus menampilkan “Hello Elastic Beanstalk!”.

Langkah 3: Terapkan aplikasi Node.js Anda dengan EB CLI

Jalankan perintah berikut untuk membuat lingkungan Elastic Beanstalk untuk aplikasi ini.

Untuk membuat lingkungan dan menyebarkan aplikasi Node.js Anda

1. Inisialisasi repositori EB CLI Anda dengan perintah eb init.

   ```
   ~/eb-nodejs$ eb init -p node.js nodejs-tutorial --region us-east-2
   ```

   Perintah ini membuat aplikasi bernama `nodejs-tutorial` dan mengkonfigurasi repositori lokal Anda untuk membuat lingkungan dengan versi platform Node.js terbaru.
2. (Opsional) Jalankan eb init lagi untuk mengonfigurasi key pair default sehingga Anda dapat menggunakan SSH untuk terhubung ke EC2 instance yang menjalankan aplikasi Anda.

   ```
   ~/eb-nodejs$ eb init
   Do you want to set up SSH for your instances?
   (y/n): y
   Select a keypair.
   1) my-keypair
   2) [ Create new KeyPair ]
   ```

   Pilih pasangan kunci jika Anda sudah memilikinya, atau ikuti prompt untuk membuatnya. Jika Anda tidak melihat prompt atau perlu mengubah pengaturan Anda nanti, jalankan eb init -i.

3. Buat lingkungan baru dan deploy aplikasi Anda dengan eb create. Elastic Beanstalk secara otomatis membuat file zip untuk aplikasi Anda dan menerapkannya ke instance di lingkungan. EC2 Setelah menerapkan aplikasi Anda, Elastic Beanstalk memulainya di port 8080.

   ```
   ~/eb-nodejs$ eb create nodejs-env
   ```

   Dibutuhkan sekitar lima menit untuk Elastic Beanstalk untuk menciptakan lingkungan Anda.

Langkah 4: Jalankan aplikasi Anda di Elastic Beanstalk

Ketika proses untuk membuat lingkungan Anda selesai, buka situs web Anda denganeb open.

```
~/eb-nodejs$ eb open
```

Selamat! Anda telah menerapkan aplikasi Node.js dengan Elastic Beanstalk! Ini akan membuka jendela peramban menggunakan nama domain yang dibuat untuk aplikasi Anda.

Langkah 5: Bersihkan

Anda dapat menghentikan lingkungan Anda ketika Anda selesai bekerja dengan aplikasi Anda. Elastic Beanstalk AWS mengakhiri semua sumber daya yang terkait dengan lingkungan Anda.

Untuk mengakhiri lingkungan Elastic Beanstalk Anda dengan EB CLI jalankan perintah berikut.

```
~/eb-nodejs$ eb terminate
```

AWS sumber daya untuk aplikasi Anda

Anda baru saja membuat aplikasi instance tunggal. Ini berfungsi sebagai aplikasi sampel langsung dengan satu EC2 instance, sehingga tidak memerlukan penyeimbangan beban atau penskalaan otomatis. Untuk aplikasi contoh tunggal Elastic Beanstalk menciptakan sumber daya berikut: AWS

- EC2 instance — Mesin EC2 virtual Amazon yang dikonfigurasi untuk menjalankan aplikasi web pada platform yang Anda pilih.

  Setiap platform menjalankan satu set perangkat lunak, file konfigurasi, dan penulisan yang berbeda untuk mendukung versi bahasa, kerangka kerja, kontainer web tertentu, atau kombinasi dari semua ini. Sebagian besar platform menggunakan Apache atau nginx sebagai proksi terbalik yang

memproses lalu lintas web di depan aplikasi web Anda, meneruskan permintaan ke aplikasi web, menyajikan aset statis, dan menghasilkan log akses dan kesalahan.

- Grup keamanan instans — Grup EC2 keamanan Amazon yang dikonfigurasi untuk memungkinkan lalu lintas masuk di port 80. Sumber daya ini memungkinkan lalu lintas HTTP dari penyeimbang beban mencapai EC2 instance yang menjalankan aplikasi web Anda. Secara default, lalu lintas tidak diizinkan di port lain.
- Bucket Amazon S3 – Lokasi penyimpanan untuk kode sumber, log, dan artifact lainnya yang dibuat saat Anda menggunakan Elastic Beanstalk.
- CloudWatch Alarm Amazon — Dua CloudWatch alarm yang memantau beban pada instans di lingkungan Anda dan dipicu jika beban terlalu tinggi atau terlalu rendah. Saat alarm terpicu, grup Auto Scaling Anda akan menaikkan atau menurunkan skala sebagai respons.
- AWS CloudFormation stack - Elastic AWS CloudFormation Beanstalk digunakan untuk meluncurkan sumber daya di lingkungan Anda dan menyebarkan perubahan konfigurasi. Sumber daya ditentukan di sebuah templat yang dapat Anda lihat di Konsol AWS CloudFormation.
- Nama domain — Nama domain yang merutekan ke aplikasi web Anda dalam formulir *`subdomain`*.*`region`*.elasticbeanstalk.com.

Elastic Beanstalk mengelola semua sumber daya tersebut. Ketika Anda mengakhiri lingkungan, Elastic Beanstalk mengakhiri semua sumber daya yang dimuatnya.

Langkah selanjutnya

Setelah lingkungan Anda menjalankan aplikasi, Anda dapat men-deploy versi baru aplikasi atau aplikasi yang berbeda kapan saja. Menyebarkan versi aplikasi baru sangat cepat karena tidak memerlukan penyediaan atau memulai ulang instance. EC2 Anda juga dapat menjelajahi lingkungan baru Anda menggunakan konsol Elastic Beanstalk. Untuk langkah-langkah mendetail, lihat Menjelajahi lingkungan Anda di bagian Memulai panduan ini.

> **Coba lebih banyak tutorial**
>
> Jika Anda ingin mencoba tutorial lain dengan aplikasi contoh yang berbeda, lihatLebih banyak contoh aplikasi dan tutorial Elastic Beanstalk untuk Node.js.

Setelah Anda menerapkan satu atau dua contoh aplikasi dan siap untuk mulai mengembangkan dan menjalankan aplikasi Node.js secara lokal, lihat. Menyiapkan lingkungan pengembangan Node.js Anda untuk Elastic Beanstalk

Terapkan dengan konsol Elastic Beanstalk

Anda juga dapat menggunakan konsol Elastic Beanstalk untuk meluncurkan aplikasi sampel. Untuk langkah-langkah rinci, lihat Membuat aplikasi contoh di Bab Memulai panduan ini.

Menyiapkan lingkungan pengembangan Node.js Anda untuk Elastic Beanstalk

Topik ini memberikan petunjuk untuk menyiapkan lingkungan pengembangan Node.js untuk menguji aplikasi Anda secara lokal sebelum menerapkannya. AWS Elastic Beanstalk Ini juga merujuk situs web yang menyediakan instruksi instalasi untuk alat yang berguna.

Topik

- Instal Node.js
- Konfirmasi instalasi npm
- Instal AWS SDK untuk Node.js
- Pasang generator Express
- Siapkan kerangka kerja dan server Express

Instal Node.js

Instal Node.js untuk menjalankan aplikasi Node.js secara lokal. Jika Anda tidak memiliki preferensi, dapatkan versi terbaru yang didukung oleh Elastic Beanstalk. Lihat Node.js dalam dokumen Platform AWS Elastic Beanstalk untuk daftar versi yang didukung.

Unduh Node.js di nodejs.org.

Konfirmasi instalasi npm

Node.js menggunakan pengelola paket npm untuk membantu Anda menginstal alat dan kerangka kerja untuk digunakan dalam aplikasi Anda. Karena npm didistribusikan dengan Node.js, Anda

akan secara otomatis menginstalnya ketika Anda mengunduh dan menginstal Node.js. Untuk mengonfirmasi npm telah diinstal Anda dapat menjalankan perintah berikut:

```
$ npm -v
```

Untuk informasi lebih lanjut tentang npm, kunjungi situs web npmjs.

Instal AWS SDK untuk Node.js

Jika Anda perlu mengelola AWS sumber daya dari dalam aplikasi Anda, instal AWS SDK untuk JavaScript di Node.js. Instal SDK dengan npm:

```
$ npm install aws-sdk
```

Kunjungi AWS SDK for JavaScript di beranda Node.js untuk informasi selengkapnya.

Pasang generator Express

Express adalah kerangka aplikasi web yang berjalan pada Node.js. Untuk menggunakannya, pertama-tama instal aplikasi baris perintah generator Express. Setelah generator Express diinstal, Anda dapat menjalankan express perintah untuk menghasilkan struktur proyek dasar untuk aplikasi web Anda. Setelah proyek dasar, file, dan dependensi diinstal, Anda dapat memulai server Express lokal di mesin pengembangan Anda.

> **Note**
>
> - Langkah-langkah ini memandu Anda menginstal generator Express pada sistem operasi Linux.
> - Untuk Linux, tergantung pada tingkat izin Anda ke direktori sistem, Anda mungkin perlu mengawali beberapa perintah ini. `sudo`

Untuk menginstal generator Express di lingkungan pengembangan Anda

1. Buat direktori kerja untuk kerangka kerja dan server Express Anda.

   ```
   ~$ mkdir node-express
   ~$ cd node-express
   ```

2. Menginstal Express secara global sehingga Anda memiliki akses ke perintah `express`.

```
~/node-express$ npm install -g express-generator
```

3. Bergantung pada sistem pengoperasian, Anda mungkin perlu mengatur jalur untuk menjalankan perintah `express`. Output dari langkah sebelumnya memberikan informasi jika Anda perlu mengatur variabel jalur Anda. Berikut ini adalah contoh untuk Linux.

```
~/node-express$ export PATH=$PATH:/usr/local/share/npm/bin/express
```

   Ketika Anda mengikuti tutorial di chapter ini, Anda harus menjalankan express perintah dari direktori yang berbeda. Setiap tutorial menyiapkan struktur proyek Express dasar di direktorinya sendiri.

Anda sekarang telah menginstal generator baris perintah Express. Anda dapat menggunakannya untuk membuat direktori framework untuk aplikasi web Anda, menyiapkan dependensi, dan memulai server aplikasi web. Selanjutnya, kita akan melalui langkah-langkah untuk mencapai ini di `node-express` direktori yang kita buat.

Siapkan kerangka kerja dan server Express

Ikuti langkah-langkah ini untuk membuat direktori dan konten kerangka kerja Express dasar. Tutorial dalam Bab ini juga mencakup langkah-langkah ini untuk mengatur kerangka dasar Express di setiap direktori aplikasi tutorial.

Untuk mengatur kerangka kerja dan server Express

1. Jalankan perintah `express`. Perintah menghasilkan `package.json`, `app.js`, dan beberapa direktori.

```
~/node-express$ express
```

   Saat diminta, ketik **y** jika Anda ingin melanjutkan.

2. Persiapkan dependensi lokal.

```
~/node-express$ npm install
```

3. Verifikasi server aplikasi web dimulai.

```
~/node-express$ npm start
```

Anda akan melihat output yang serupa dengan yang berikut:

```
> nodejs@0.0.0 start /home/local/user/node-express
> node ./bin/www
```

Server berjalan pada port 3000 secara default. Untuk mengujinya, jalankan `curl http://localhost:3000` di terminal lain, atau buka browser di komputer lokal dan masukkan alamat URL`http://localhost:3000`.

Tekan Ctrl+C untuk menghentikan server.

Menggunakan platform Node.js Elastic Beanstalk

Topik ini menjelaskan cara mengonfigurasi, membangun, dan menjalankan aplikasi Node.js Anda di Elastic Beanstalk.

AWS Elastic Beanstalk mendukung sejumlah cabang platform untuk berbagai versi bahasa pemrograman Node.js. Lihat Node.js di dokumen AWS Elastic Beanstalk Platform untuk daftar lengkap.

Elastic Beanstalk menyediakan opsi konfigurasi yang dapat Anda gunakan untuk menyesuaikan perangkat lunak yang berjalan EC2 pada instance di lingkungan Elastic Beanstalk Anda. Anda dapat mengonfigurasi variabel lingkungan yang diperlukan oleh aplikasi Anda, mengaktifkan rotasi log ke Amazon S3, dan memetakan folder di sumber aplikasi Anda yang berisi file statis ke jalur yang disajikan oleh server proxy.

Opsi konfigurasi tersedia di konsol Elastic Beanstalk untuk memodifikasi konfigurasi dari lingkungan yang sedang berjalan. Agar Anda tidak kehilangan konfigurasi lingkungan ketika mengakhirinya, Anda dapat menggunakan konfigurasi tersimpan untuk menyimpan pengaturan dan kemudian menerapkannya ke lingkungan lain.

Untuk menyimpan pengaturan di kode sumber, Anda dapat menyertakan file konfigurasi. Pengaturan di file konfigurasi diterapkan setiap kali Anda membuat lingkungan atau men-deploy aplikasi Anda. Anda juga dapat menggunakan file konfigurasi untuk menginstal paket, menjalankan penulisan, dan melakukan operasi penyesuaian instans lainnya selama deployment.

Anda dapat menyertakan `Package.json` file dalam bundel sumber Anda untuk menginstal paket selama penerapan, untuk menyediakan perintah mulai, dan untuk menentukan Node.js versi yang Anda ingin aplikasi Anda gunakan. Anda dapat menyertakan file `npm-shrinkwrap.json` untuk mengunci versi dependensi.

Node.jsPlatform ini mencakup server proxy untuk melayani aset statis, meneruskan lalu lintas ke aplikasi Anda, dan mengompres respons. Anda dapat memperpanjang atau menimpa konfigurasi proksi default untuk skenario lanjutan.

Ada beberapa opsi untuk memulai aplikasi Anda. Anda dapat menambahkan a Procfileke bundel sumber Anda untuk menentukan perintah yang memulai aplikasi Anda. Jika Anda tidak memberikan `Procfile` tetapi menyediakan `package.json` file, Elastic Beanstalk berjalan. `npm start` Jika Anda juga tidak menyediakannya, Elastic Beanstalk `app.js` mencari `server.js` file atau, dalam urutan ini, dan menjalankan skrip.

Pengaturan yang diterapkan di konsol Elastic Beanstalk mengganti pengaturan yang sama di file konfigurasi, jika ada. Hal tersebut memungkinkan Anda memiliki pengaturan default di file konfigurasi, dan mengganti keduanya dengan pengaturan khusus lingkungan di konsol tersebut. Untuk informasi selengkapnya tentang yang diutamakan, dan metode lain untuk mengubah pengaturan, lihat Opsi konfigurasi.

Untuk detail tentang berbagai cara memperluas platform berbasis Linux Elastic Beanstalk, lihat the section called “Memperluas platform Linux”.

Mengkonfigurasi lingkungan Anda Node.js

Anda dapat menggunakan pengaturan Node.js platform untuk menyempurnakan perilaku instans Amazon EC2 Anda. Anda dapat mengedit konfigurasi EC2 instans Amazon untuk lingkungan Elastic Beanstalk menggunakan konsol Elastic Beanstalk.

Gunakan konsol Elastic Beanstalk untuk mengaktifkan rotasi log ke Amazon S3 dan mengonfigurasi variabel yang dapat dibaca aplikasi Anda dari lingkungan.

Untuk mengonfigurasi Node.js lingkungan Anda di konsol Elastic Beanstalk

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.

4. Dalam kategori konfigurasi Pembaruan, pemantauan, dan pencatatan, pilih Edit.

Opsi kontainer

Anda dapat menentukan opsi khusus platform berikut ini:

- Server proksi – Server proksi untuk digunakan di instans lingkungan Anda. Secara default, NGINX digunakan.

Opsi log

Bagian Opsi Log memiliki dua pengaturan:

- Profil instans — Menentukan profil instans yang memiliki izin untuk mengakses bucket Amazon S3 yang terkait dengan aplikasi Anda.
- Aktifkan rotasi file log ke Amazon S3 — Menentukan apakah file log untuk EC2 instans Amazon aplikasi Anda disalin ke bucket Amazon S3 yang terkait dengan aplikasi Anda.

File statis

Untuk meningkatkan kinerja, Anda dapat menggunakan bagian File statis untuk mengkonfigurasi server proxy untuk melayani file statis (misalnya, HTML atau gambar) dari satu set direktori di dalam aplikasi web Anda. Untuk setiap direktori, Anda mengatur jalur virtual ke pemetaan direktori. Saat server proksi menerima permintaan untuk file di jalur yang ditentukan, server langsung menyajikan file daripada merutekan permintaan ke aplikasi Anda.

Untuk detail tentang mengonfigurasi file statis menggunakan file konfigurasi atau konsol Elastic Beanstalk, lihat. the section called “File statis”

Properti lingkungan

Gunakan bagian Properti Lingkungan untuk menentukan pengaturan konfigurasi lingkungan pada EC2 instance Amazon yang menjalankan aplikasi Anda. Pengaturan ini diteruskan sebagai pasangan nilai kunci ke aplikasi.

Di dalam Node.js lingkungan yang berjalan di AWS Elastic Beanstalk, Anda dapat mengakses variabel lingkungan dengan menjalankan`process.env.ENV_VARIABLE`.

```
var endpoint = process.env.API_ENDPOINT
```

Node.jsPlatform menetapkan variabel PORT lingkungan ke port tempat server proxy melewati lalu lintas. Untuk informasi selengkapnya, lihat Mengonfigurasi server proksi.

Lihat Variabel lingkungan dan pengaturan perangkat lunak lainnya untuk informasi selengkapnya.

Mengkonfigurasi lingkungan Amazon Linux AMI (sebelumnya Amazon Linux 2) Node.js

Kategori konfigurasi perangkat lunak konsol berikut hanya didukung pada lingkungan Elastic Beanstalk Node.js yang menggunakan versi platform Amazon Linux AMI (sebelum Amazon Linux 2).

> **Catatan**
>
> - Informasi dalam topik ini hanya berlaku untuk cabang platform berbasis Amazon Linux AMI (AL1). AL2023/ cabang AL2 platform tidak kompatibel dengan versi platform Amazon Linux AMI (AL1) sebelumnya dan memerlukan pengaturan konfigurasi yang berbeda.
> - Pada tanggal 18 Juli 2022, Elastic Beanstalk menetapkan status semua cabang platform berdasarkan Amazon Linux AMI () untuk pensiun. AL1 Untuk informasi selengkapnya tentang migrasi ke cabang platform Amazon Linux 2023 saat ini dan didukung penuh, lihat. Memigrasi aplikasi Elastic Beanstalk Linux Anda ke Amazon Linux 2023 atau Amazon Linux 2

Opsi wadah - Amazon Linux AMI (AL1)

Di halaman konfigurasi, tentukan hal berikut:

- Proxy server - Menentukan server web mana yang akan digunakan untuk Node.js koneksi proxy. Secara default, NGINX digunakan. Jika Anda memilih tidak ada, pemetaan file statis tidak berlaku, dan GZIP kompresi dinonaktifkan.
- Node.jsversi - Menentukan versi. Node.js Untuk daftar Node.js versi yang didukung, lihat Node.jsdi panduan AWS Elastic Beanstalk Platform.
- GZIPkompresi - Menentukan apakah GZIP kompresi diaktifkan. Secara default, GZIP kompresi diaktifkan.
- Perintah node - Memungkinkan Anda memasukkan perintah yang digunakan untuk memulai Node.js aplikasi. String kosong (default) berarti Elastic `app.js` Beanstalk menggunakan`server.js`, lalu, dan kemudian. `npm start`

Node.jskonfigurasi namespace

Anda dapat menggunakan file konfigurasi untuk mengatur opsi konfigurasi dan melakukan tugas-tugas konfigurasi instans lain selama deployment. Opsi konfigurasi dapat berupa platform khusus atau berlaku untuk semua platform di layanan Elastic Beanstalk secara keseluruhan. Opsi konfigurasi diatur ke dalam ruang nama.

Anda dapat memilih proxy yang akan digunakan pada instance untuk lingkungan Anda dengan menggunakan `aws:elasticbeanstalk:environment:proxy` namespace. Contoh berikut mengkonfigurasi lingkungan Anda untuk menggunakan server Apache HTTPD proxy.

Example .ebextensions/nodejs-settings.config

```
option_settings:
  aws:elasticbeanstalk:environment:proxy:
    ProxyServer: apache
```

Anda dapat mengonfigurasi proksi untuk menyajikan file statis dengan menggunakan namespace `aws:elasticbeanstalk:environment:proxy:staticfiles`. Untuk informasi lebih lanjut dan contoh, lihat the section called “File statis”.

Elastic Beanstalk memberikan banyak opsi konfigurasi untuk menyesuaikan lingkungan Anda. Selain file konfigurasi, Anda juga dapat mengatur opsi konfigurasi menggunakan konsol tersebut, konfigurasi tersimpan, EB CLI, atau AWS CLI. Untuk informasi selengkapnya, lihat Opsi konfigurasi.

Platform Amazon Linux AMI (sebelumnya Amazon Linux 2) Node.js

Jika lingkungan Elastic Node.js Beanstalk Anda menggunakan versi platform AMI Amazon Linux (sebelumnya Amazon Linux 2), pertimbangkan konfigurasi dan rekomendasi spesifik di bagian ini.

> **Catatan**
>
> - Informasi dalam topik ini hanya berlaku untuk cabang platform berbasis Amazon Linux AMI (AL1). AL2023/ cabang AL2 platform tidak kompatibel dengan versi platform Amazon Linux AMI (AL1) sebelumnya dan memerlukan pengaturan konfigurasi yang berbeda.
> - Pada tanggal 18 Juli 2022, Elastic Beanstalk menetapkan status semua cabang platform berdasarkan Amazon Linux AMI () untuk pensiun. AL1 Untuk informasi selengkapnya tentang migrasi ke cabang platform Amazon Linux 2023 saat ini dan didukung penuh, lihat.

Memigrasi aplikasi Elastic Beanstalk Linux Anda ke Amazon Linux 2023 atau Amazon Linux 2

Node.jsopsi konfigurasi khusus platform - Amazon Linux AMI () AL1

Elastic Beanstalk mendukung beberapa opsi konfigurasi khusus platform untuk versi platform Amazon Linux AMI. Node.js Anda dapat memilih server proxy mana yang akan dijalankan di depan aplikasi Anda, memilih versi tertentu Node.js untuk dijalankan, dan memilih perintah yang digunakan untuk menjalankan aplikasi Anda.

Untuk server proxy, Anda dapat menggunakan server Apache proxy NGINX atau proxy. Anda dapat mengatur none nilai ke `ProxyServer` opsi. Dengan pengaturan ini, Elastic Beanstalk menjalankan aplikasi Anda secara mandiri, tidak di belakang server proksi. Jika lingkungan Anda menjalankan aplikasi mandiri, perbarui kode Anda untuk mendengarkan port yang NGINX meneruskan lalu lintas ke.

```
var port = process.env.PORT || 8080;

app.listen(port, function() {
  console.log('Server running at http://127.0.0.1:%s', port);
});
```

Node.jsversi bahasa - Amazon Linux AMI (AL1)

Dalam hal versi bahasa yang didukung, platform Node.js Amazon Linux AMI berbeda dengan platform yang dikelola Elastic Beanstalk lainnya. Ini karena setiap versi Node.js platform hanya mendukung beberapa versi Node.js bahasa. Untuk daftar Node.js versi yang didukung, lihat Node.jsdi panduan AWS Elastic Beanstalk Platform.

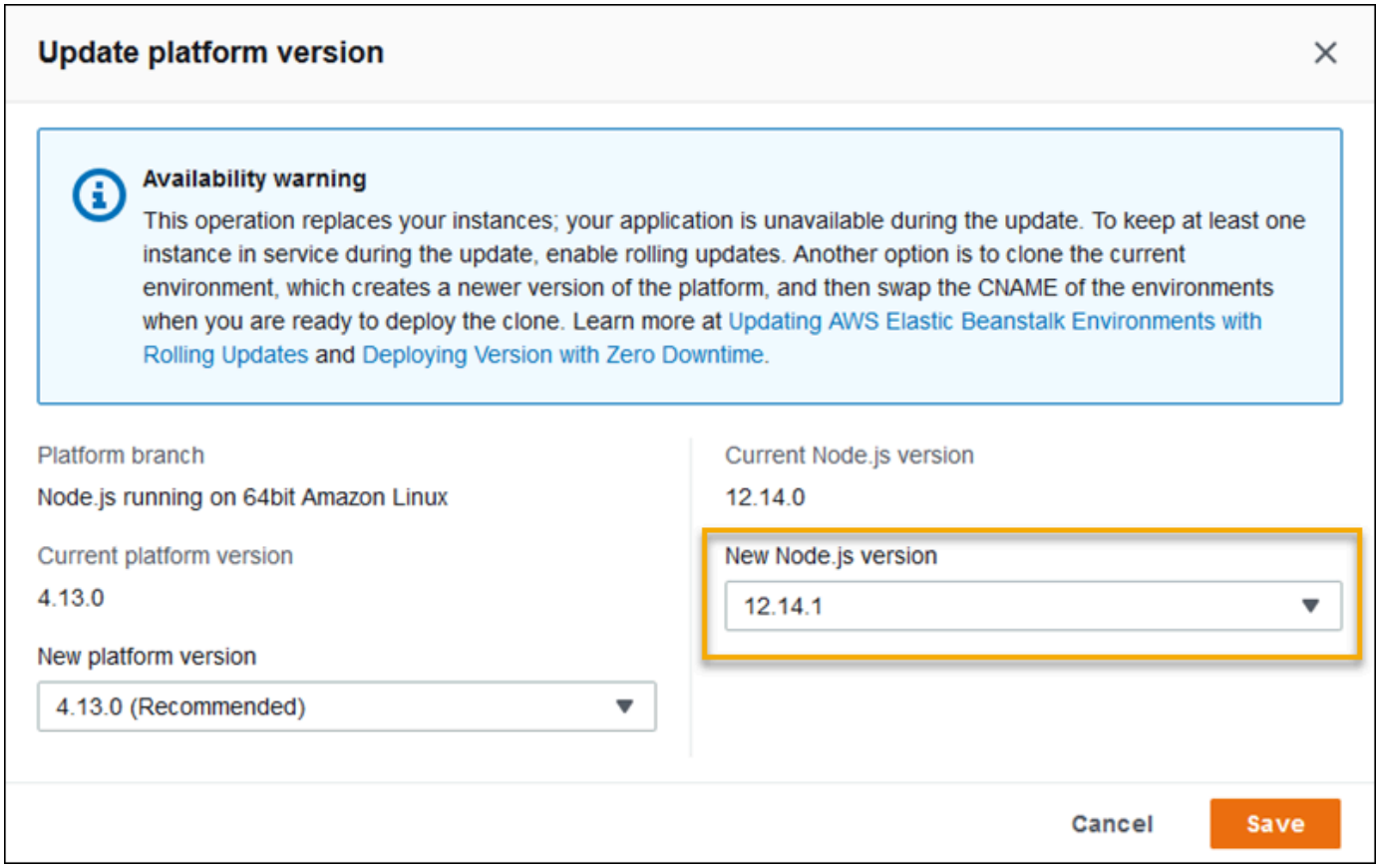
Anda dapat menggunakan opsi konfigurasi spesifik platform untuk mengatur versi bahasa. Untuk petunjuk, lihat the section called “Mengkonfigurasi lingkungan Anda Node.js”. Atau, gunakan konsol Elastic Beanstalk untuk Node.js memperbarui versi yang digunakan lingkungan Anda sebagai bagian dari memperbarui versi platform Anda.

> **Note**
>
> Ketika dukungan untuk versi Node.js yang Anda gunakan dihapus dari platform, Anda harus mengubah atau menghapus pengaturan versi sebelum melakukan pembaruan platform. Hal

ini mungkin terjadi ketika kelemahan keamanan diidentifikasi untuk satu atau beberapa versi Node.js.
Ketika ini terjadi, mencoba memperbarui ke versi baru platform yang tidak mendukung konfigurasi NodeVersiongagal. Untuk menghindari perlunya membuat lingkungan baru, ubah opsi NodeVersionkonfigurasi ke versi Node.js yang didukung oleh versi platform lama dan yang baru, atau hapus pengaturan opsi, lalu lakukan pembaruan platform.

Untuk mengonfigurasi Node.js versi lingkungan Anda di konsol Elastic Beanstalk

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada halaman gambaran umum lingkungan, di bawah Platform, pilih Ubah.
4. Pada kotak dialog Perbarui versi platform, pilih Node.js versi.



5. Pilih Simpan.

Node.jsruang nama konfigurasi - Amazon Linux AMI () AL1

Platform AMI Node.js Amazon Linux mendefinisikan opsi tambahan di ruang `aws:elasticbeanstalk:container:nodejs` nama `aws:elasticbeanstalk:container:nodejs:staticfiles` dan ruang nama.

File konfigurasi berikut memberitahu Elastic Beanstalk `npm start` untuk digunakan untuk menjalankan aplikasi. Ini juga mengatur jenis proxy ke Apache dan memungkinkan kompresi. Terakhir, file tersebut mengonfigurasi proksi untuk menyediakan file statis dari dua direktori sumber. Salah satu sumber adalah HTML file di `html` jalur di bawah root situs web dari direktori `statichtml` sumber. Sumber lainnya adalah file gambar di `images` jalur di bawah root situs web dari direktori `staticimages` sumber.

Example .ebextensions/node-settings.config

```
option_settings:
  aws:elasticbeanstalk:container:nodejs:
    NodeCommand: "npm start"
    ProxyServer: apache
    GzipCompression: true
  aws:elasticbeanstalk:container:nodejs:staticfiles:
    /html: statichtml
    /images: staticimages
```

Elastic Beanstalk memberikan banyak opsi konfigurasi untuk menyesuaikan lingkungan Anda. Selain file konfigurasi, Anda juga dapat mengatur opsi konfigurasi menggunakan konsol tersebut, konfigurasi tersimpan, EB CLI, atau AWS CLI. Untuk informasi selengkapnya, lihat Opsi konfigurasi.

Mengkonfigurasi perintah start khusus dengan Procfile on Elastic Beanstalk

Anda dapat menyertakan file yang dipanggil `Procfile` di root bundel sumber Anda untuk menentukan perintah yang memulai aplikasi Anda.

Example Procfile

```
web: node index.js
```

Untuk informasi tentang `Procfile` penggunaan, lihatBuildfile dan Procfile.

> **Note**
>
> Fitur ini menggantikan `NodeCommand` opsi lama di namespace. `aws:elasticbeanstalk:container:nodejs`

Mengonfigurasi dependensi aplikasi Anda pada Elastic Beanstalk

Aplikasi Anda mungkin memiliki dependensi pada beberapa Node.js modul, seperti yang Anda tentukan dalam `require()` pernyataan. Modul-modul ini disimpan dalam `node_modules` direktori. Ketika aplikasi Anda berjalan, Node.js memuat modul dari direktori ini. Untuk informasi selengkapnya, lihat Memuat dari folder node_modules dalam dokumentasi. Node.js

Anda dapat menentukan dependensi modul ini menggunakan file. `package.json` Jika Elastic Beanstalk mendeteksi file ini *`node_modules`* dan direktori tidak ada, Elastic Beanstalk berjalan sebagai pengguna webapp. *`npm install`* `npm install`Perintah menginstal dependensi di `node_modules` direktori, yang dibuat Elastic Beanstalk sebelumnya. `npm install`Perintah mengakses paket yang tercantum dalam `package.json` file dari registri npm publik atau lokasi lain. Untuk informasi selengkapnya, lihat situs web npm Docs.

Jika Elastic Beanstalk `node_modules` mendeteksi direktori, Elastic Beanstalk tidak berjalan, bahkan jika ada file. `npm install` `package.json` Elastic Beanstalk mengasumsikan bahwa paket dependensi tersedia di direktori untuk diakses dan dimuat`node_modules`. Node.js

Bagian berikut memberikan informasi selengkapnya tentang membuat dependensi Node.js modul untuk aplikasi Anda.

> **Note**
>
> Jika Anda mengalami masalah penerapan saat Elastic Beanstalk `npm install` berjalan, pertimbangkan pendekatan alternatif. Sertakan `node_modules` direktori dengan modul dependensi dalam bundel sumber aplikasi Anda. Melakukannya dapat menghindari masalah dengan menginstal dependensi dari registri npm publik saat Anda menyelidiki masalah tersebut. Karena modul ketergantungan bersumber dari direktori lokal, dong ini mungkin juga membantu mengurangi waktu penerapan. Untuk informasi selengkapnya, lihat Menyertakan Node.js dependensi dalam direktori node_modules

Menentukan Node.js dependensi dengan file package.json

Sertakan file `package.json` di akar sumber proyek Anda untuk menentukan paket dependensi dan memberikan perintah mulai. Saat `package.json` file ada, dan `node_modules` direktori tidak ada di root sumber proyek Anda, Elastic `npm install` Beanstalk berjalan sebagai pengguna webapp untuk menginstal dependensi dari registri npm publik. Elastic Beanstalk juga `start` menggunakan perintah untuk memulai aplikasi Anda. Untuk informasi selengkapnya tentang `package.json` file, lihat Menentukan dependensi dalam `package.json` file di situs web npm Docs.

Gunakan kata kunci `scripts` untuk memberikan perintah mulai. Saat ini, `scripts` kata kunci digunakan sebagai pengganti `NodeCommand` opsi lama di `aws:elasticbeanstalk:container:nodejs` namespace.

Example package.json – Express

```
{
    "name": "my-app",
    "version": "0.0.1",
    "private": true,
    "dependencies": {
      "ejs": "latest",
      "aws-sdk": "latest",
      "express": "latest",
      "body-parser": "latest"
    },
    "scripts": {
      "start": "node app.js"
    }
  }
```

Mode produksi dan dependensi pengembang

Untuk menentukan dependensi Anda dalam *`package.json`* file, gunakan atribut dependensi dan devDependencies. Atribut dependencies menunjuk paket yang dibutuhkan oleh aplikasi Anda dalam produksi. Atribut devDependencies menunjuk paket yang hanya diperlukan untuk pengembangan dan pengujian lokal.

Elastic `npm install` Beanstalk berjalan sebagai pengguna webapp dengan perintah berikut. Opsi perintah bervariasi tergantung pada versi npm yang disertakan pada cabang platform tempat aplikasi Anda berjalan.

- npm v6 - Elastic Beanstalk menginstal dependensi dalam mode produksi secara default. Ia menggunakan perintah`npm install --production`.
- npm v7 atau lebih besar — Elastic Beanstalk menghilangkan devDependencies. Ia menggunakan perintah`npm install --omit=dev`.

Kedua perintah yang tercantum di atas tidak menginstal paket yang devDependencies.

Jika Anda perlu menginstal paket devDependencies, setel properti lingkungan NPM_USE_PRODUCTION ke. `false` Dengan pengaturan ini kami tidak akan menggunakan opsi di atas saat menjalankan npm install. Ini akan mengakibatkan paket devDependencies diinstal.

SSH dan HTTPS

Dimulai dengan rilis platform Amazon Linux 2 7 Maret 2023, Anda juga dapat menggunakan protokol SSH dan HTTPS untuk mengambil paket dari repositori Git. Platform branch Node.js 16 mendukung protokol SSH dan HTTPS. Node.js14 hanya mendukung protokol HTTPS.

Example package.json - Node.js 16 mendukung HTTPS dan SSH

```
...
"dependencies": {
  "aws-sdk": "https://github.com/aws/aws-sdk-js.git",
  "aws-chime": "git+ssh://git@github.com:aws/amazon-chime-sdk-js.git"
}
```

Versi dan rentang versi

> **Important**
>
> Fitur untuk menentukan rentang versi tidak tersedia untuk cabang platform Node.js yang berjalan pada AL2 023. Kami hanya mendukung satu versi Node.js dalam cabang Node.js tertentu di AL2 023. Jika `package.json` file Anda menentukan rentang versi, kami akan mengabaikannya dan default ke versi cabang platform Node.js.

Gunakan `engines` kata kunci dalam `package.json` file untuk menentukan Node.js versi yang Anda ingin aplikasi Anda gunakan. Anda juga dapat menentukan rentang versi

menggunakan notasi npm. Untuk informasi selengkapnya tentang sintaks untuk rentang versi, lihat Pembuatan Versi Semantik menggunakan npm di situs web. Node.js `engines`Kata kunci dalam Node.js `package.json` file menggantikan `NodeVersion` opsi lama di namespace. `aws:elasticbeanstalk:container:nodejs`

Example **`package.json`**— Node.js Versi tunggal

```
{
    ...
    "engines": { "node" : "14.16.0" }
  }
```

Example **`package.json`**— rentang Node.js versi

```
{
    ...
    "engines": { "node" : ">=10 <11" }
  }
```

Ketika rentang versi ditunjukkan, Elastic Beanstalk menginstal versi terbaru yang tersedia Node.js platform dalam kisaran tersebut. Dalam contoh ini, rentang menunjukkan bahwa versi harus lebih besar dari atau sama dengan versi 10, tetapi kurang dari versi 11. Hasilnya, Elastic Beanstalk menginstal versi 10.x.y terbaru, Node.js yang tersedia di platform yang didukung.

Ketahuilah bahwa Anda hanya dapat menentukan Node.js versi yang sesuai dengan cabang platform Anda. Misalnya, jika Anda menggunakan cabang platform Node.js 16, Anda hanya dapat menentukan 16.x.y Node.js versi. Anda dapat menggunakan opsi rentang versi yang didukung oleh npm untuk mengizinkan lebih banyak fleksibilitas. Untuk Node.js versi yang valid untuk setiap cabang platform, lihat Node.jsdi panduan AWS Elastic Beanstalk Platform.

> **Note**
>
> Ketika support versi Node.js yang digunakan dihapus dari platform, Anda harus mengubah atau menghapus pengaturan versi Node.js sebelum melakukan pembaruan platform. Hal ini mungkin terjadi ketika kelemahan keamanan diidentifikasi pada satu atau beberapa versi Node.js.
>
> Ketika ini terjadi, upaya untuk memperbarui ke versi platform baru yang tidak support versi Node.js yang dikonfigurasi akan gagal. Agar tidak perlu membuat lingkungan baru, ubah pengaturan versi Node.js di `package.json` ke versi Node.js yang didukung versi platform

lama dan yang baru. Anda memiliki opsi untuk menentukan rentang versi Node.js yang mencakup versi yang didukung, seperti yang dijelaskan sebelumnya dalam topik ini. Anda juga memiliki opsi untuk menghapus pengaturan, dan kemudian menyebarkan bundel sumber baru.

Menyertakan Node.js dependensi dalam direktori node_modules

Untuk menyebarkan paket dependensi ke instance lingkungan bersama dengan kode aplikasi Anda, sertakan mereka dalam direktori yang diberi nama `node_modules` di root sumber proyek Anda. Untuk informasi selengkapnya, lihat Mengunduh dan menginstal paket secara lokal di situs web npm Docs.

Saat Anda menerapkan `node_modules` direktori ke versi AL2 Node.js platform AL2 023/, Elastic Beanstalk mengasumsikan bahwa Anda menyediakan paket dependensi Anda sendiri, dan menghindari penginstalan dependensi yang ditentukan dalam file package.json. Node.jsmencari dependensi di direktori. `node_modules` Untuk informasi selengkapnya, lihat Memuat dari node_modules Folder dalam dokumentasi. Node.js

> **Note**
>
> Jika Anda mengalami masalah penerapan saat Elastic Beanstalk `npm install` berjalan, pertimbangkan untuk menggunakan pendekatan yang dijelaskan dalam topik ini sebagai solusi saat Anda menyelidiki masalah tersebut.

Mengunci dependensi dengan Elastic npm shrinkwrap Beanstalk

Node.jsPlatform berjalan `npm install` sebagai pengguna webapp setiap kali Anda menerapkan. Ketika versi baru dependensi Anda tersedia, mereka diinstal saat Anda menerapkan aplikasi Anda, berpotensi menyebabkan penerapan memakan waktu lama.

Agar tidak perlu meningkatkan dependensi dengan membuat file `npm-shrinkwrap.json` yang mengunci dependensi aplikasi Anda ke versi saat ini.

```
$ npm install
$ npm shrinkwrap
wrote npm-shrinkwrap.json
```

Sertakan file ini dalam paket sumber Anda untuk memastikan bahwa dependensi hanya diinstal sekali.

Mengonfigurasi server proksi

Elastic Beanstalk NGINX dapat Apache HTTPD menggunakan atau sebagai proxy terbalik untuk memetakan aplikasi Anda ke penyeimbang beban Elastic Load Balancing di port 80. Nilai default-nya NGINX. Elastic Beanstalk menyediakan konfigurasi proksi default yang dapat Anda perluas atau sepenuhnya diganti dengan konfigurasi Anda sendiri.

Secara default, Elastic Beanstalk mengonfigurasi proxy untuk meneruskan permintaan ke aplikasi Anda pada port 5000. Anda dapat mengganti port default dengan mengatur properti lingkungan `PORT` ke port yang didengarkan aplikasi utama Anda.

> **Note**
>
> Port yang didengarkan aplikasi Anda tidak memengaruhi port yang didengarkan NGINX server untuk menerima permintaan dari penyeimbang beban.

Mengkonfigurasi server proxy pada versi platform Anda

Semua AL2 AL2 023/platform mendukung fitur konfigurasi proxy yang seragam. Untuk informasi selengkapnya tentang mengonfigurasi server proxy pada versi platform Anda yang menjalankan AL2 023/AL2, lihat. Konfigurasi proksi terbalik

Mengonfigurasi proksi di Amazon Linux AMI (Amazon Linux 2 terdahulu)

Jika lingkungan Elastic Node.js Beanstalk Anda menggunakan versi platform AMI Amazon Linux (sebelumnya Amazon Linux 2), baca informasi di bagian ini.

> **Catatan**
>
> - Informasi dalam topik ini hanya berlaku untuk cabang platform berbasis Amazon Linux AMI (AL1). AL2023/ cabang AL2 platform tidak kompatibel dengan versi platform Amazon Linux AMI (AL1) sebelumnya dan memerlukan pengaturan konfigurasi yang berbeda.
> - Pada tanggal 18 Juli 2022, Elastic Beanstalk menetapkan status semua cabang platform berdasarkan Amazon Linux AMI () untuk pensiun. AL1 Untuk informasi selengkapnya tentang migrasi ke cabang platform Amazon Linux 2023 saat ini dan didukung penuh, lihat.

> [Memigrasi aplikasi Elastic Beanstalk Linux Anda ke Amazon Linux 2023 atau Amazon Linux 2]

Memperluas dan mengganti konfigurasi proxy default — Amazon Linux AMI () AL1

Node.jsPlatform ini menggunakan proxy terbalik untuk menyampaikan permintaan dari port 80 pada instance ke aplikasi Anda yang mendengarkan pada port 8081. Elastic Beanstalk menyediakan konfigurasi proksi default yang dapat Anda perluas atau sepenuhnya diganti dengan konfigurasi Anda sendiri.

Untuk memperluas konfigurasi default, tambahkan file `.conf` ke `/etc/nginx/conf.d` dengan file konfigurasi. Untuk contoh spesifik, lihat[Mengakhiri HTTPS pada EC2 instance yang menjalankan Node.js].

Node.jsPlatform menetapkan variabel PORT lingkungan ke port tempat server proxy melewati lalu lintas. Baca variabel ini dalam kode Anda untuk mengonfigurasi port untuk aplikasi Anda.

```
    var port = process.env.PORT || 3000;

    var server = app.listen(port, function () {
        console.log('Server running at http://127.0.0.1:' + port + '/');
    });
```

NGINXKonfigurasi default meneruskan lalu lintas ke server hulu yang diberi `nodejs` nama di. `127.0.0.1:8081` Dimungkinkan untuk menghapus konfigurasi default dan menyediakan konfigurasi Anda sendiri dalam [file konfigurasi].

Example .ebextensions/proxy.config

Contoh berikut menghapus konfigurasi default dan menambahkan konfigurasi kustom yang meneruskan lalu lintas ke port 5000, bukan 8081.

```
files:
  /etc/nginx/conf.d/proxy.conf:
    mode: "000644"
    owner: root
    group: root
    content: |
      upstream nodejs {
        server 127.0.0.1:5000;
```

```
        keepalive 256;
      }

      server {
        listen 8080;

        if ($time_iso8601 ~ "^(\d{4})-(\d{2})-(\d{2})T(\d{2})") {
            set $year $1;
            set $month $2;
            set $day $3;
            set $hour $4;
        }
        access_log /var/log/nginx/healthd/application.log.$year-$month-$day-$hour
 healthd;
        access_log  /var/log/nginx/access.log  main;

        location / {
            proxy_pass  http://nodejs;
            proxy_set_header   Connection "";
            proxy_http_version 1.1;
            proxy_set_header        Host            $host;
            proxy_set_header        X-Real-IP       $remote_addr;
            proxy_set_header        X-Forwarded-For $proxy_add_x_forwarded_for;
        }

        gzip on;
        gzip_comp_level 4;
        gzip_types text/html text/plain text/css application/json application/x-
javascript text/xml application/xml application/xml+rss text/javascript;

        location /static {
            alias /var/app/current/static;
        }

      }

  /opt/elasticbeanstalk/hooks/configdeploy/post/99_kill_default_nginx.sh:
    mode: "000755"
    owner: root
    group: root
    content: |
      #!/bin/bash -xe
      rm -f /etc/nginx/conf.d/00_elastic_beanstalk_proxy.conf
      service nginx stop
```

```
      service nginx start

container_commands:
  removeconfig:
    command: "rm -f /tmp/deployment/config/
#etc#nginx#conf.d#00_elastic_beanstalk_proxy.conf /etc/nginx/
conf.d/00_elastic_beanstalk_proxy.conf"
```

Contoh konfigurasi (`/etc/nginx/conf.d/proxy.conf`) menggunakan konfigurasi default di `/etc/nginx/conf.d/00_elastic_beanstalk_proxy.conf` sebagai basis untuk menyertakan blok server default dengan pengaturan kompresi dan log, dan pemetaan file statis.

`removeconfig`Perintah menghapus konfigurasi default untuk wadah sehingga server proxy menggunakan konfigurasi kustom. Elastic Beanstalk menciptakan kembali konfigurasi default ketika setiap konfigurasi di-deploy. Untuk menjelaskan hal ini, dalam contoh berikut, post-configuration-deployment hook (`/opt/elasticbeanstalk/hooks/configdeploy/post/99_kill_default_nginx.sh`) ditambahkan. Ini menghapus konfigurasi default dan memulai ulang server proksi.

> **Note**
>
> Konfigurasi default mungkin berubah di versi Node.js platform yang akan datang. Gunakan versi konfigurasi terbaru sebagai dasar penyesuaian Anda untuk memastikan kompatibilitas.

Jika Anda mengganti konfigurasi default, Anda harus menentukan pemetaan dan kompresi file statis apa pun. GZIP Ini karena platform tidak dapat menerapkan pengaturan standar.

Lebih banyak contoh aplikasi dan tutorial Elastic Beanstalk untuk Node.js

Bagian ini menyediakan aplikasi dan tutorial tambahan. QuickStart untuk Node.jsTopik yang terletak sebelumnya dalam topik ini memandu Anda melalui peluncuran contoh aplikasi Node.js dengan EB CLI.

Untuk memulai aplikasi Node.js aktif AWS Elastic Beanstalk, yang Anda butuhkan hanyalah bundel sumber aplikasi untuk diunggah sebagai versi aplikasi pertama Anda dan untuk menyebarkan ke lingkungan.

Meluncurkan lingkungan dengan sampel aplikasi Node.js

Elastic Beanstalk menyediakan contoh aplikasi satu halaman untuk setiap platform serta contoh yang lebih kompleks yang menunjukkan penggunaan sumber daya AWS tambahan seperti Amazon RDS dan fitur khusus bahasa atau platform dan. APIs

> **Note**
>
> Ikuti langkah-langkah dalam `README.md` file bundel sumber untuk menerapkannya.

Sampel

Jenis lingkungan	Paket sumber	Deskripsi
Server Web	nodejs.zip	Aplikasi satu halaman. Untuk meluncurkan aplikasi sampel dengan EB CLI, lihat. QuickStart untuk Node.js Anda juga dapat menggunakan konsol Elastic Beanstalk untuk meluncurkan aplikasi sampel. Untuk langkah-langkah rinci, lihat Membuat aplikasi contoh di Bab Memulai panduan ini.
Server Web dengan Amazon RDS	nodejs-example-express-rds.zip	Aplikasi log hiking yang menggunakan framework Express dan Amazon Relational Database Service (RDS). Tutorial
Server Web dengan Amazon ElastiCache	nodejs-example-express-elasticache.zip	Aplikasi web ekspres yang menggunakan Amazon ElastiCache untuk pengelompokan. Pengklasteran meningkatkan ketersediaan, performa, dan keamanan aplikasi web Anda yang tinggi. Tutorial

Jenis lingkungan	Paket sumber	Deskripsi
Server Web dengan DynamoDB, Amazon SNS dan Amazon SQS	nodejs-example-dynamo.zip	situs web Express yang mengumpulkan informasi kontak pengguna untuk kampanye pemasaran perusahaan baru. Menggunakan AWS SDK for JavaScript di Node.js untuk menulis entri ke tabel DynamoDB, dan file konfigurasi Elastic Beanstalk untuk membuat resource di DynamoDB, Amazon SNS, dan Amazon SQS. Tutorial

Langkah selanjutnya

Setelah Anda memiliki lingkungan yang menjalankan aplikasi, Anda dapat men-deploy versi baru aplikasi atau aplikasi yang sama sekali berbeda setiap saat. Menyebarkan versi aplikasi baru sangat cepat karena tidak memerlukan penyediaan atau memulai ulang instance. EC2 Untuk detail deployment aplikasi, lihat Men-deploy Versi Baru Aplikasi Anda.

Setelah Anda menerapkan satu atau dua contoh aplikasi dan siap untuk mulai mengembangkan dan menjalankan aplikasi Node.js secara lokal, lihat Menyiapkan lingkungan pengembangan Node.js Anda untuk Elastic Beanstalk untuk menyiapkan lingkungan pengembangan Node.js dengan semua alat yang Anda perlukan.

Menyebarkan aplikasi Node.js Express ke Elastic Beanstalk

Bagian ini memandu Anda melalui penerapan aplikasi sampel ke Elastic Beanstalk menggunakan Elastic Beanstalk Command Line Interface (EB CLI) dan kemudian memperbarui aplikasi untuk menggunakan framework Express.

Prasyarat

Tutorial ini membutuhkan prasyarat berikut:

- Runtime Node.js
- Perangkat lunak pengelola paket Node.js default, npm
- Generator baris perintah Express

- Antarmuka Baris Perintah Elastic Beanstalk (EB CLI)

Untuk detail tentang menginstal tiga komponen pertama yang terdaftar dan menyiapkan lingkungan pengembangan lokal Anda, lihatMenyiapkan lingkungan pengembangan Node.js Anda untuk Elastic Beanstalk. Untuk tutorial ini, Anda tidak perlu menginstal AWS SDK untuk Node.js, yang juga disebutkan dalam topik yang direferensikan.

Untuk detail tentang menginstal dan mengonfigurasi CLI EB, lihat dan. Instal EB CLI dengan skrip pengaturan (disarankan) Mengonfigurasi EB CLI

Membuat lingkungan Elastic Beanstalk

Direktori aplikasi Anda

Tutorial ini menggunakan direktori yang disebut `nodejs-example-express-rds` untuk bundel sumber aplikasi. Buat `nodejs-example-express-rds` direktori untuk tutorial ini.

```
~$ mkdir nodejs-example-express-rds
```

> **Note**
>
> Setiap tutorial dalam chapter ini menggunakan direktorinya sendiri untuk bundel sumber aplikasi. Nama direktori cocok dengan nama aplikasi sampel yang digunakan oleh tutorial.

Ubah direktori kerja Anda saat ini menjadi`nodejs-example-express-rds`.

```
~$ cd nodejs-example-express-rds
```

Sekarang, mari kita mengatur lingkungan Elastic Beanstalk yang menjalankan platform Node.js dan aplikasi sampel. Kita akan menggunakan antarmuka baris perintah Elastic Beanstalk (EB CLI).

Untuk mengonfigurasi repositori EB CLI untuk aplikasi Anda dan membuat lingkungan Elastic Beanstalk yang menjalankan platform Node.js

1. Buat repositori dengan perintah eb init.

   ```
   ~/nodejs-example-express-rds$ eb init --platform node.js --region <region>
   ```

Perintah ini membuat file konfigurasi dalam folder bernama `.elasticbeanstalk` yang menentukan pengaturan dalam membuat lingkungan untuk aplikasi Anda, dan membuat sebuah aplikasi Elastic Beanstalk yang diambil dari nama folder saat ini.

2. Buat lingkungan yang menjalankan aplikasi sampel dengan perintah eb create.

```
~/nodejs-example-express-rds$ eb create --sample nodejs-example-express-rds
```

Perintah ini membuat lingkungan yang seimbang beban dengan pengaturan default untuk platform Node.js dan sumber daya berikut:

- EC2 instance — Mesin virtual Amazon Elastic Compute Cloud (Amazon EC2) yang dikonfigurasi untuk menjalankan aplikasi web pada platform yang Anda pilih.

  Setiap platform menjalankan satu set perangkat lunak, file konfigurasi dan penulisan tertentu untuk mendukung versi bahasa, kerangka kerja, kontainer web tertentu, atau kombinasi dari semua ini. Sebagian besar platform menggunakan Apache atau NGINX sebagai proksi terbalik yang ada di depan aplikasi web Anda, meneruskan permintaan ke aplikasi web, menyajikan aset statis, dan menghasilkan log akses dan kesalahan.
- Grup keamanan instans — Grup EC2 keamanan Amazon yang dikonfigurasi untuk memungkinkan lalu lintas masuk pada port 80. Sumber daya ini memungkinkan lalu lintas HTTP dari penyeimbang beban mencapai EC2 instance yang menjalankan aplikasi web Anda. Secara default, lalu lintas tidak diizinkan di port lain.
- Penyeimbang beban – Penyeimbang beban Elastic Load Balancing yang dikonfigurasi untuk mendistribusikan permintaan ke instans yang menjalankan aplikasi Anda. Penyeimbang beban juga menghilangkan kebutuhan untuk mengekspos instans Anda langsung ke internet.
- Grup keamanan penyeimbang beban - Grup EC2 keamanan Amazon yang dikonfigurasi untuk memungkinkan lalu lintas masuk di port 80. Sumber daya ini memungkinkan lalu lintas HTTP dari internet mencapai penyeimbang beban. Secara default, lalu lintas tidak diizinkan di port lain.
- Grup Auto Scaling – Grup Auto Scaling yang dikonfigurasi untuk menggantikan instans jika diakhiri atau menjadi tidak tersedia.
- Bucket Amazon S3 – Lokasi penyimpanan untuk kode sumber, log, dan artifact lainnya yang dibuat saat Anda menggunakan Elastic Beanstalk.

- CloudWatch Alarm Amazon — Dua CloudWatch alarm yang memantau beban pada instans di lingkungan Anda dan yang dipicu jika beban terlalu tinggi atau terlalu rendah. Saat alarm terpicu, grup Auto Scaling Anda akan menaikkan atau menurunkan skala sebagai respons.
- AWS CloudFormation stack - Elastic AWS CloudFormation Beanstalk digunakan untuk meluncurkan sumber daya di lingkungan Anda dan menyebarkan perubahan konfigurasi. Sumber daya ditentukan di sebuah templat yang dapat Anda lihat di Konsol AWS CloudFormation.
- Nama domain — Nama domain yang merutekan ke aplikasi web Anda dalam formulir *subdomain*.*region*.elasticbeanstalk.com.

> **Keamanan domain**
>
> Untuk meningkatkan keamanan aplikasi Elastic Beanstalk Anda, domain elasticbeanstalk.com terdaftar di Daftar Akhiran Publik (PSL).
> Jika Anda perlu mengatur cookie sensitif di nama domain default untuk aplikasi Elastic Beanstalk Anda, kami sarankan Anda menggunakan cookie `__Host-` dengan awalan untuk meningkatkan keamanan. Praktik ini melindungi domain Anda dari upaya pemalsuan permintaan lintas situs (CSRF). Untuk informasi selengkapnya, lihat halaman Set-Cookie di Jaringan Pengembang Mozilla.

3. Ketika pembuatan lingkungan selesai, gunakan perintah eb open untuk membuka URL lingkungan di browser default.

```
~/nodejs-example-express-rds$ eb open
```

Anda sekarang telah membuat lingkungan Elastic Beanstalk Node.js dengan aplikasi sampel. Anda dapat memperbaruinya dengan aplikasi Anda sendiri. Selanjutnya, kami memperbarui aplikasi sampel untuk menggunakan kerangka kerja Express.

Perbarui aplikasi untuk menggunakan Express

Setelah Anda membuat lingkungan dengan aplikasi sampel, Anda dapat memperbaruinya dengan aplikasi Anda sendiri. Dalam prosedur ini, pertama-tama kita menjalankan npm install perintah express dan untuk mengatur kerangka kerja Express di direktori aplikasi Anda. Kemudian kami menggunakan EB CLI untuk memperbarui lingkungan Elastic Beanstalk Anda dengan aplikasi yang diperbarui.

Memperbarui aplikasi Anda untuk menggunakan Express

1. Jalankan perintah `express`. Perintah menghasilkan `package.json`, `app.js`, dan beberapa direktori.

   ```
   ~/nodejs-example-express-rds$ express
   ```

   Saat diminta, ketik **y** jika Anda ingin melanjutkan.

   > **Note**
   >
   > Jika express perintah tidak berfungsi, Anda mungkin belum menginstal generator baris perintah Express seperti yang dijelaskan di bagian Prasyarat sebelumnya. Atau pengaturan jalur direktori untuk mesin lokal Anda mungkin perlu diatur untuk menjalankan express perintah. Lihat bagian Prasyarat untuk langkah-langkah rinci tentang pengaturan lingkungan pengembangan Anda, sehingga Anda dapat melanjutkan dengan tutorial ini.

2. Persiapkan dependensi lokal.

   ```
   ~/nodejs-example-express-rds$ npm install
   ```

3. (Opsional) Verifikasi server aplikasi web dimulai.

   ```
   ~/nodejs-example-express-rds$ npm start
   ```

   Anda akan melihat output yang serupa dengan yang berikut:

   ```
   > nodejs@0.0.0 start /home/local/user/node-express
   > node ./bin/www
   ```

   Server berjalan pada port 3000 secara default. Untuk mengujinya, jalankan `curl http://localhost:3000` di terminal lain, atau buka browser di komputer lokal dan masukkan alamat URL`http://localhost:3000`.

   Tekan Ctrl+C untuk menghentikan server.

4. Terapkan perubahan ke lingkungan Elastic Beanstalk Anda dengan perintah. eb deploy

```
~/nodejs-example-express-rds$ eb deploy
```

5. Setelah lingkungan berwarna hijau dan siap, refresh URL untuk memverifikasi bahwa itu berfungsi. Anda akan melihat halaman web yang bertuliskan Selamat Datang di Express.

Selanjutnya, mari kita perbarui aplikasi Express untuk menyajikan file statis dan menambahkan halaman baru.

Untuk mengonfigurasi file statis dan menambahkan halaman baru ke aplikasi Express

1. Tambahkan file konfigurasi kedua di `.ebextensions`folder dengan konten berikut:

   `nodejs-example-express-rds/.ebextensions/staticfiles.config`

   ```
   option_settings:
       aws:elasticbeanstalk:environment:proxy:staticfiles:
           /stylesheets: public/stylesheets
   ```

   Pengaturan ini mengonfigurasi server proksi untuk menyajikan file di folder `public` pada jalur `/public` aplikasi. Melayani file secara statis dari server proxy mengurangi beban pada aplikasi Anda. Untuk informasi selengkapnya, lihat File statis sebelumnya di Bab ini.

2. (Opsional) Untuk mengonfirmasi bahwa pemetaan statis dikonfigurasi dengan benar, komentari konfigurasi pemetaan statis di. `nodejs-example-express-rds/app.js` Ini menghapus pemetaan dari aplikasi node.

   ```
   //  app.use(express.static(path.join(__dirname, 'public')));
   ```

   Pemetaan file statis dalam `staticfiles.config` file dari langkah sebelumnya harus tetap memuat stylesheet dengan sukses, bahkan setelah Anda mengomentari baris ini. Untuk memverifikasi bahwa pemetaan file statis dimuat melalui konfigurasi file statis proxy, bukan aplikasi ekspres, hapus nilai berikut. `option_settings:` Setelah dihapus dari konfigurasi file statis dan aplikasi node, stylesheet akan gagal dimuat.

   Ingatlah untuk mengatur ulang isi dari kedua `nodejs-example-express-rds/app.js` dan `staticfiles.config` ketika Anda selesai pengujian.

3. Menambahkan `nodejs-example-express-rds/routes/hike.js`. Ketik berikut ini:

```
exports.index = function(req, res) {
 res.render('hike', {title: 'My Hiking Log'});
};

exports.add_hike = function(req, res) {
};
```

4. Perbarui `nodejs-example-express-rds/app.js` untuk menyertakan tiga baris baru.

   Pertama, tambahkan baris berikut untuk menambahkan `require` pada rute ini:

   ```
   var hike = require('./routes/hike');
   ```

   File Anda akan terlihat mirip dengan snippet berikut:

   ```
   var express = require('express');
   var path = require('path');
   var hike = require('./routes/hike');
   ```

   Kemudian, tambahkan dua baris berikut ke `nodejs-example-express-rds/app.js` setelah `var app = express();`

   ```
   app.get('/hikes', hike.index);
   app.post('/add_hike', hike.add_hike);
   ```

   File Anda akan terlihat mirip dengan snippet berikut:

   ```
   var app = express();
   app.get('/hikes', hike.index);
   app.post('/add_hike', hike.add_hike);
   ```

5. Salin `nodejs-example-express-rds/views/index.jade` ke `nodejs-example-express-rds/views/hike.jade`.

   ```
   ~/nodejs-example-express-rds$ cp views/index.jade views/hike.jade
   ```

6. Terapkan perubahan dengan eb deployperintah.

   ```
   ~/nodejs-example-express-rds$ eb deploy
   ```

7. Lingkungan Anda akan diperbarui setelah beberapa menit. Setelah lingkungan Anda hijau dan siap, verifikasi bahwa itu bekerja dengan me-refresh peramban Anda dan menambahkan **`hikes`** di akhir URL (misalnya, `http://node-express-env-syypntcz2q.elasticbeanstalk.com/hikes`).

   Anda akan melihat halaman web berjudul Log Hiking Saya.

Anda sekarang telah membuat aplikasi web yang menggunakan kerangka kerja Express. Di bagian berikutnya, kita akan memodifikasi aplikasi untuk menggunakan Amazon Relational Database Service (RDS) untuk menyimpan log hiking.

Perbarui aplikasi untuk menggunakan Amazon RDS

Pada langkah selanjutnya ini kami memperbarui aplikasi untuk menggunakan Amazon RDS for MySQL.

Untuk memperbarui aplikasi Anda untuk menggunakan RDS untuk MySQL

1. Untuk membuat database RDS untuk MySQL yang digabungkan ke lingkungan Elastic Beanstalk Anda, ikuti petunjuk dalam topik Menambahkan database yang disertakan nanti dalam chapter ini. Menambahkan instance database membutuhkan waktu sekitar 10 menit.
2. Perbarui bagian dependensi di `package.json` dengan konten berikut:

```
"dependencies": {
    "async": "^3.2.4",
    "express": "4.18.2",
    "jade": "1.11.0",
    "mysql": "2.18.1",
    "node-uuid": "^1.4.8",
    "body-parser": "^1.20.1",
    "method-override": "^3.0.0",
    "morgan": "^1.10.0",
    "errorhandler": "^1.5.1"
  }
```

3. Jalankan npm install.

```
~/nodejs-example-express-rds$ npm install
```

4. Perbarui `app.js` untuk terhubung ke database, membuat tabel, dan menyisipkan log hiking default tunggal. Setiap kali aplikasi ini digunakan, aplikasi ini akan menjatuhkan tabel kenaikan sebelumnya dan membuatnya kembali.

```
/**
 * Module dependencies.
 */

 const express = require('express')
 , routes = require('./routes')
 , hike = require('./routes/hike')
 , http = require('http')
 , path = require('path')
 , mysql = require('mysql')
 , async = require('async')
 , bodyParser = require('body-parser')
 , methodOverride = require('method-override')
 , morgan = require('morgan')
 , errorhandler = require('errorhandler');

const { connect } = require('http2');

const app = express()

app.set('views', __dirname + '/views')
app.set('view engine', 'jade')
app.use(methodOverride())
app.use(bodyParser.json())
app.use(bodyParser.urlencoded({ extended: true }))
app.use(express.static(path.join(__dirname, 'public')))


app.set('connection', mysql.createConnection({
host: process.env.RDS_HOSTNAME,
user: process.env.RDS_USERNAME,
password: process.env.RDS_PASSWORD,
port: process.env.RDS_PORT}));

function init() {
 app.get('/', routes.index);
 app.get('/hikes', hike.index);
 app.post('/add_hike', hike.add_hike);
}
```

```
const client = app.get('connection');
async.series([
 function connect(callback) {
   client.connect(callback);
   console.log('Connected!');
 },
 function clear(callback) {
   client.query('DROP DATABASE IF EXISTS mynode_db', callback);
 },
 function create_db(callback) {
   client.query('CREATE DATABASE mynode_db', callback);
 },
 function use_db(callback) {
   client.query('USE mynode_db', callback);
 },
 function create_table(callback) {
    client.query('CREATE TABLE HIKES (' +
                        'ID VARCHAR(40), ' +
                        'HIKE_DATE DATE, ' +
                        'NAME VARCHAR(40), ' +
                        'DISTANCE VARCHAR(40), ' +
                        'LOCATION VARCHAR(40), ' +
                        'WEATHER VARCHAR(40), ' +
                        'PRIMARY KEY(ID))', callback);
 },
 function insert_default(callback) {
   const hike = {HIKE_DATE: new Date(), NAME: 'Hazard Stevens',
         LOCATION: 'Mt Rainier', DISTANCE: '4,027m vertical', WEATHER:'Bad', ID:
 '12345'};
   client.query('INSERT INTO HIKES set ?', hike, callback);
 }
], function (err, results) {
 if (err) {
   console.log('Exception initializing database.');
   throw err;
 } else {
   console.log('Database initialization complete.');
   init();
 }
});

module.exports = app
```

5. Tambahkan konten berikut ke`routes/hike.js`. Ini akan memungkinkan rute untuk memasukkan log hiking baru ke database HIKES.

```
const uuid = require('node-uuid');
exports.index = function(req, res) {
  res.app.get('connection').query( 'SELECT * FROM HIKES', function(err,
rows) {
    if (err) {
      res.send(err);
    } else {
      console.log(JSON.stringify(rows));
      res.render('hike', {title: 'My Hiking Log', hikes: rows});
  }});
};
exports.add_hike = function(req, res){
  const input = req.body.hike;
  const hike = { HIKE_DATE: new Date(), ID: uuid.v4(), NAME: input.NAME,
  LOCATION: input.LOCATION, DISTANCE: input.DISTANCE, WEATHER: input.WEATHER};
  console.log('Request to log hike:' + JSON.stringify(hike));
  req.app.get('connection').query('INSERT INTO HIKES set ?', hike, function(err) {
      if (err) {
        res.send(err);
      } else {
        res.redirect('/hikes');
      }
   });
};
```

6. Ganti konten `routes/index.js` dengan yang berikut ini:

```
/*
 * GET home page.
 */

exports.index = function(req, res){
  res.render('index', { title: 'Express' });
};
```

7. Tambahkan template giok berikut `views/hike.jade` untuk menyediakan antarmuka pengguna untuk menambahkan log hiking.

```
extends layout
```

```
block content
  h1= title
  p Welcome to #{title}

  form(action="/add_hike", method="post")
    table(border="1")
      tr
        td Your Name
        td
          input(name="hike[NAME]", type="textbox")
      tr
        td Location
        td
          input(name="hike[LOCATION]", type="textbox")
      tr
        td Distance
        td
          input(name="hike[DISTANCE]", type="textbox")
      tr
        td Weather
        td
          input(name="hike[WEATHER]", type="radio", value="Good")
          | Good
          input(name="hike[WEATHER]", type="radio", value="Bad")
          | Bad
          input(name="hike[WEATHER]", type="radio", value="Seattle", checked)
          | Seattle
      tr
        td(colspan="2")
          input(type="submit", value="Record Hike")

  div
    h3 Hikes
    table(border="1")
      tr
        td Date
        td Name
        td Location
        td Distance
        td Weather
      each hike in hikes
        tr
          td #{hike.HIKE_DATE.toDateString()}
          td #{hike.NAME}
```

```
            td #{hike.LOCATION}
            td #{hike.DISTANCE}
            td #{hike.WEATHER}
```

8. Terapkan perubahan dengan eb deployperintah.

```
~/nodejs-example-express-rds$ eb deploy
```

Bersihkan

Jika Anda selesai bekerja dengan Elastic Beanstalk, Anda dapat menghentikan lingkungan Anda.

Gunakan perintah eb terminate untuk mengakhiri lingkungan Anda dan semua sumber daya yang dimuatnya.

```
~/nodejs-example-express-rds$ eb terminate
The environment "nodejs-example-express-rds-env" and all associated instances will be
 terminated.
To confirm, type the environment name: nodejs-example-express-rds-env
INFO: terminateEnvironment is starting.
...
```

Menyebarkan aplikasi Node.js Express dengan pengelompokan ke Elastic Beanstalk

Tutorial ini memandu Anda melalui penerapan aplikasi sampel ke Elastic Beanstalk menggunakan Elastic Beanstalk Command Line Interface (EB CLI), dan kemudian memperbarui aplikasi untuk menggunakan framework Express, Amazon, dan clustering. ElastiCache Pengklasteran meningkatkan ketersediaan, performa, dan keamanan aplikasi web Anda yang tinggi. Untuk mempelajari lebih lanjut tentang Amazon ElastiCache, buka Apa itu Amazon ElastiCache (Memcached)? di Panduan Pengguna Amazon ElastiCache (Memcached).

> **Note**
>
> Contoh ini menciptakan AWS sumber daya, yang mungkin dikenakan biaya untuk Anda. Untuk informasi lebih lanjut tentang AWS harga, lihathttps://aws.amazon.com/pricing/. Beberapa layanan adalah bagian dari Tingkat Penggunaan AWS Gratis. Jika Anda adalah

pelanggan baru, Anda dapat menguji layanan ini secara gratis. Lihat https://aws.amazon.com/free/ untuk informasi selengkapnya.

Prasyarat

Tutorial ini membutuhkan prasyarat berikut:

- Runtime Node.js
- Perangkat lunak pengelola paket Node.js default, npm
- Generator baris perintah Express
- Antarmuka Baris Perintah Elastic Beanstalk (EB CLI)

Untuk detail tentang menginstal tiga komponen pertama yang terdaftar dan menyiapkan lingkungan pengembangan lokal Anda, lihatMenyiapkan lingkungan pengembangan Node.js Anda untuk Elastic Beanstalk. Untuk tutorial ini, Anda tidak perlu menginstal AWS SDK untuk Node.js, yang juga disebutkan dalam topik yang direferensikan.

Untuk detail tentang menginstal dan mengonfigurasi CLI EB, lihat dan. Instal EB CLI dengan skrip pengaturan (disarankan) Mengonfigurasi EB CLI

Membuat lingkungan Elastic Beanstalk

Direktori aplikasi Anda

Tutorial ini menggunakan direktori yang disebut `nodejs-example-express-elasticache` untuk bundel sumber aplikasi. Buat `nodejs-example-express-elasticache` direktori untuk tutorial ini.

```
~$ mkdir nodejs-example-express-elasticache
```

> **Note**
>
> Setiap tutorial dalam chapter ini menggunakan direktorinya sendiri untuk bundel sumber aplikasi. Nama direktori cocok dengan nama aplikasi sampel yang digunakan oleh tutorial.

Ubah direktori kerja Anda saat ini menjadi`nodejs-example-express-elasticache`.

```
~$ cd nodejs-example-express-elasticache
```

Sekarang, mari kita mengatur lingkungan Elastic Beanstalk yang menjalankan platform Node.js dan aplikasi sampel. Kita akan menggunakan antarmuka baris perintah Elastic Beanstalk (EB CLI).

Untuk mengonfigurasi repositori EB CLI untuk aplikasi Anda dan membuat lingkungan Elastic Beanstalk yang menjalankan platform Node.js

1. Buat repositori dengan perintah eb init.

   ```
   ~/nodejs-example-express-elasticache$ eb init --platform node.js --region <region>
   ```

   Perintah ini membuat file konfigurasi dalam folder bernama `.elasticbeanstalk` yang menentukan pengaturan dalam membuat lingkungan untuk aplikasi Anda, dan membuat sebuah aplikasi Elastic Beanstalk yang diambil dari nama folder saat ini.

2. Buat lingkungan yang menjalankan aplikasi sampel dengan perintah eb create.

   ```
   ~/nodejs-example-express-elasticache$ eb create --sample nodejs-example-express-elasticache
   ```

   Perintah ini membuat lingkungan yang seimbang beban dengan pengaturan default untuk platform Node.js dan sumber daya berikut:

   - EC2 instance — Mesin virtual Amazon Elastic Compute Cloud (Amazon EC2) yang dikonfigurasi untuk menjalankan aplikasi web pada platform yang Anda pilih.

     Setiap platform menjalankan satu set perangkat lunak, file konfigurasi dan penulisan tertentu untuk mendukung versi bahasa, kerangka kerja, kontainer web tertentu, atau kombinasi dari semua ini. Sebagian besar platform menggunakan Apache atau NGINX sebagai proksi terbalik yang ada di depan aplikasi web Anda, meneruskan permintaan ke aplikasi web, menyajikan aset statis, dan menghasilkan log akses dan kesalahan.
   - Grup keamanan instans — Grup EC2 keamanan Amazon yang dikonfigurasi untuk memungkinkan lalu lintas masuk pada port 80. Sumber daya ini memungkinkan lalu lintas HTTP dari penyeimbang beban mencapai EC2 instance yang menjalankan aplikasi web Anda. Secara default, lalu lintas tidak diizinkan di port lain.

- Penyeimbang beban – Penyeimbang beban Elastic Load Balancing yang dikonfigurasi untuk mendistribusikan permintaan ke instans yang menjalankan aplikasi Anda. Penyeimbang beban juga menghilangkan kebutuhan untuk mengekspos instans Anda langsung ke internet.
- Grup keamanan penyeimbang beban - Grup EC2 keamanan Amazon yang dikonfigurasi untuk memungkinkan lalu lintas masuk di port 80. Sumber daya ini memungkinkan lalu lintas HTTP dari internet mencapai penyeimbang beban. Secara default, lalu lintas tidak diizinkan di port lain.
- Grup Auto Scaling – Grup Auto Scaling yang dikonfigurasi untuk menggantikan instans jika diakhiri atau menjadi tidak tersedia.
- Bucket Amazon S3 – Lokasi penyimpanan untuk kode sumber, log, dan artifact lainnya yang dibuat saat Anda menggunakan Elastic Beanstalk.
- CloudWatch Alarm Amazon — Dua CloudWatch alarm yang memantau beban pada instans di lingkungan Anda dan yang dipicu jika beban terlalu tinggi atau terlalu rendah. Saat alarm terpicu, grup Auto Scaling Anda akan menaikkan atau menurunkan skala sebagai respons.
- AWS CloudFormation stack - Elastic AWS CloudFormation Beanstalk digunakan untuk meluncurkan sumber daya di lingkungan Anda dan menyebarkan perubahan konfigurasi. Sumber daya ditentukan di sebuah templat yang dapat Anda lihat di Konsol AWS CloudFormation.
- Nama domain — Nama domain yang merutekan ke aplikasi web Anda dalam formulir *subdomain*.*region*.elasticbeanstalk.com.

> **Keamanan domain**
>
> Untuk meningkatkan keamanan aplikasi Elastic Beanstalk Anda, domain elasticbeanstalk.com terdaftar di Daftar Akhiran Publik (PSL).
> Jika Anda perlu mengatur cookie sensitif di nama domain default untuk aplikasi Elastic Beanstalk Anda, kami sarankan Anda menggunakan cookie `__Host-` dengan awalan untuk meningkatkan keamanan. Praktik ini melindungi domain Anda dari upaya pemalsuan permintaan lintas situs (CSRF). Untuk informasi selengkapnya, lihat halaman Set-Cookie di Jaringan Pengembang Mozilla.

3. Ketika pembuatan lingkungan selesai, gunakan perintah eb open untuk membuka URL lingkungan di browser default.

```
~/nodejs-example-express-elasticache$ eb open
```

Anda sekarang telah membuat lingkungan Elastic Beanstalk Node.js dengan aplikasi sampel. Anda dapat memperbaruinya dengan aplikasi Anda sendiri. Selanjutnya, kami memperbarui aplikasi sampel untuk menggunakan kerangka kerja Express.

Perbarui aplikasi untuk menggunakan Express

Memperbarui aplikasi sampel di lingkungan Elastic Beanstalk untuk menggunakan kerangka kerja Express.

Anda dapat mengunduh kode sumber akhir nodejs-example-express-elasticachedari.zip.

Memperbarui aplikasi Anda untuk menggunakan Express

Setelah Anda membuat lingkungan dengan aplikasi sampel, Anda dapat memperbaruinya dengan aplikasi Anda sendiri. Dalam prosedur ini, pertama-tama kita menjalankan npm install perintah express dan untuk mengatur kerangka kerja Express di direktori aplikasi Anda.

1. Jalankan perintah `express`. Perintah menghasilkan `package.json`, `app.js`, dan beberapa direktori.

   ```
   ~/nodejs-example-express-elasticache$ express
   ```

   Saat diminta, ketik **y** jika Anda ingin melanjutkan.

   > **Note**
   >
   > Jika express perintah tidak berfungsi, Anda mungkin belum menginstal generator baris perintah Express seperti yang dijelaskan di bagian Prasyarat sebelumnya. Atau pengaturan jalur direktori untuk mesin lokal Anda mungkin perlu diatur untuk menjalankan express perintah. Lihat bagian Prasyarat untuk langkah-langkah rinci tentang pengaturan lingkungan pengembangan Anda, sehingga Anda dapat melanjutkan dengan tutorial ini.

2. Persiapkan dependensi lokal.

   ```
   ~/nodejs-example-express-elasticache$ npm install
   ```

3. (Opsional) Verifikasi server aplikasi web dimulai.

```
~/nodejs-example-express-elasticache$ npm start
```

Anda akan melihat output yang serupa dengan yang berikut:

```
> nodejs@0.0.0 start /home/local/user/node-express
> node ./bin/www
```

Server berjalan pada port 3000 secara default. Untuk mengujinya, jalankan `curl http://localhost:3000` di terminal lain, atau buka browser di komputer lokal dan masukkan alamat URL`http://localhost:3000`.

Tekan Ctrl+C untuk menghentikan server.

4. Ganti nama `nodejs-example-express-elasticache/app.js` ke `nodejs-example-express-elasticache/express-app.js`.

```
~/nodejs-example-express-elasticache$ mv app.js express-app.js
```

5. Perbarui baris `var app = express();` dalam `nodejs-example-express-elasticache/express-app.js` ke berikut ini:

```
var app = module.exports = express();
```

6. Di komputer lokal Anda, buat file bernama `nodejs-example-express-elasticache/app.js` dengan kode berikut.

```
/**
 * Module dependencies.
 */

 const express = require('express'),
 session = require('express-session'),
 bodyParser = require('body-parser'),
 methodOverride = require('method-override'),
 cookieParser = require('cookie-parser'),
 fs = require('fs'),
 filename = '/var/nodelist',
 app = express();

let MemcachedStore = require('connect-memcached')(session);
```

```
function setup(cacheNodes) {
 app.use(bodyParser.raw());
 app.use(methodOverride());
 if (cacheNodes.length > 0) {
   app.use(cookieParser());

   console.log('Using memcached store nodes:');
   console.log(cacheNodes);

   app.use(session({
     secret: 'your secret here',
     resave: false,
     saveUninitialized: false,
     store: new MemcachedStore({ 'hosts': cacheNodes })
   }));
 } else {
   console.log('Not using memcached store.');
   app.use(session({
     resave: false,
     saveUninitialized: false, secret: 'your secret here'
   }));
 }

 app.get('/', function (req, resp) {
   if (req.session.views) {
     req.session.views++
     resp.setHeader('Content-Type', 'text/html')
     resp.send(`You are session: ${req.session.id}. Views: ${req.session.views}`)
   } else {
     req.session.views = 1
     resp.send(`You are session: ${req.session.id}. No views yet, refresh the page!
`)
   }
 });

 if (!module.parent) {
   console.log('Running express without cluster. Listening on port %d',
 process.env.PORT || 5000)
   app.listen(process.env.PORT || 5000)
 }
}

console.log("Reading elastic cache configuration")
```

```
// Load elasticache configuration.
fs.readFile(filename, 'UTF8', function (err, data) {
 if (err) throw err;

 let cacheNodes = []
 if (data) {
   let lines = data.split('\n');
   for (let i = 0; i < lines.length; i++) {
     if (lines[i].length > 0) {
       cacheNodes.push(lines[i])
     }
   }
 }

 setup(cacheNodes)
});

module.exports = app;
```

7. Ganti isi `nodejs-example-express-elasticache/bin/www` file dengan yang berikut ini:

```
#!/usr/bin/env node

/**
 * Module dependencies.
 */

const app = require('../app');
const cluster = require('cluster');
const debug = require('debug')('nodejs-example-express-elasticache:server');
const http = require('http');
const workers = {},
  count = require('os').cpus().length;

function spawn() {
  const worker = cluster.fork();
  workers[worker.pid] = worker;
  return worker;
}


/**
 * Get port from environment and store in Express.
 */
```

```
const port = normalizePort(process.env.PORT || '3000');
app.set('port', port);

if (cluster.isMaster) {
  for (let i = 0; i < count; i++) {
    spawn();
  }

  // If a worker dies, log it to the console and start another worker.
  cluster.on('exit', function (worker, code, signal) {
    console.log('Worker ' + worker.process.pid + ' died.');
    cluster.fork();
  });

  // Log when a worker starts listening
  cluster.on('listening', function (worker, address) {
    console.log('Worker started with PID ' + worker.process.pid + '.');
  });

} else {
  /**
   * Create HTTP server.
   */

  let server = http.createServer(app);

  /**
   * Event listener for HTTP server "error" event.
   */

  function onError(error) {
    if (error.syscall !== 'listen') {
      throw error;
    }

    const bind = typeof port === 'string'
      ? 'Pipe ' + port
      : 'Port ' + port;

    // handle specific listen errors with friendly messages
    switch (error.code) {
      case 'EACCES':
        console.error(bind + ' requires elevated privileges');
```

```
        process.exit(1);
        break;
      case 'EADDRINUSE':
        console.error(bind + ' is already in use');
        process.exit(1);
        break;
      default:
        throw error;
    }
  }

  /**
   * Event listener for HTTP server "listening" event.
   */

  function onListening() {
    const addr = server.address();
    const bind = typeof addr === 'string'
      ? 'pipe ' + addr
      : 'port ' + addr.port;
    debug('Listening on ' + bind);
  }

  /**
   * Listen on provided port, on all network interfaces.
   */

  server.listen(port);
  server.on('error', onError);
  server.on('listening', onListening);
}

/**
 * Normalize a port into a number, string, or false.
 */

function normalizePort(val) {
  const port = parseInt(val, 10);

  if (isNaN(port)) {
    // named pipe
    return val;
  }
```

```
  if (port >= 0) {
    // port number
    return port;
  }

  return false;
}
```

8. Terapkan perubahan ke lingkungan Elastic Beanstalk Anda dengan perintah. eb deploy

```
~/nodejs-example-express-elasticache$ eb deploy
```

9. Lingkungan Anda akan diperbarui setelah beberapa menit. Setelah lingkungan berwarna hijau dan siap, refresh URL untuk memverifikasi bahwa itu berfungsi. Anda akan melihat halaman web yang bertuliskan "Selamat Datang di Express".

Anda dapat mengakses log untuk EC2 instance yang menjalankan aplikasi Anda. Untuk petunjuk tentang mengakses log Anda, lihat Melihat log dari EC2 instans Amazon di lingkungan Elastic Beanstalk Anda.

Selanjutnya, mari kita perbarui aplikasi Express untuk menggunakan Amazon ElastiCache.

Untuk memperbarui aplikasi Express Anda untuk menggunakan Amazon ElastiCache

1. Di komputer lokal Anda, buat direktori `.ebextensions` di direktori tingkat atas paket sumber Anda. Dalam contoh ini, kami menggunakan `nodejs-example-express-elasticache/.ebextensions`.
2. Buat file konfigurasi `nodejs-example-express-elasticache/.ebextensions/elasticache-iam-with-script.config` dengan snippet berikut. Untuk informasi lebih lanjut tentang file konfigurasi, lihat Node.jskonfigurasi namespace. Hal ini membuat pengguna IAM dengan izin yang diperlukan untuk menemukan simpul elasticache dan menulis ke file kapan pun cache berubah. Anda juga dapat menyalin file nodejs-example-express-elasticachedari.zip. Untuk informasi lebih lanjut tentang ElastiCache properti, lihatContoh: ElastiCache.

> **Note**
>
> YAML bergantung pada indentasi yang konsisten. Cocokkan tingkat indentasi saat mengganti konten dalam file konfigurasi contoh dan pastikan bahwa editor teks Anda menggunakan spasi, bukan karakter tab, untuk indentasi.

```
Resources:
  MyCacheSecurityGroup:
    Type: 'AWS::EC2::SecurityGroup'
    Properties:
      GroupDescription: "Lock cache down to webserver access only"
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort:
            Fn::GetOptionSetting:
              OptionName: CachePort
              DefaultValue: 11211
          ToPort:
            Fn::GetOptionSetting:
              OptionName: CachePort
              DefaultValue: 11211
          SourceSecurityGroupName:
            Ref: AWSEBSecurityGroup
  MyElastiCache:
    Type: 'AWS::ElastiCache::CacheCluster'
    Properties:
      CacheNodeType:
        Fn::GetOptionSetting:
          OptionName: CacheNodeType
          DefaultValue: cache.t2.micro
      NumCacheNodes:
        Fn::GetOptionSetting:
          OptionName: NumCacheNodes
          DefaultValue: 1
      Engine:
        Fn::GetOptionSetting:
          OptionName: Engine
          DefaultValue: redis
      VpcSecurityGroupIds:
        -
```

```
          Fn::GetAtt:
            - MyCacheSecurityGroup
            - GroupId
  AWSEBAutoScalingGroup :
    Metadata :
      ElastiCacheConfig :
        CacheName :
          Ref : MyElastiCache
        CacheSize :
           Fn::GetOptionSetting:
             OptionName : NumCacheNodes
             DefaultValue: 1
  WebServerUser :
    Type : AWS::IAM::User
    Properties :
      Path : "/"
      Policies:
        -
          PolicyName: root
          PolicyDocument :
            Statement :
              -
                Effect : Allow
                Action :
                  - cloudformation:DescribeStackResource
                  - cloudformation:ListStackResources
                  - elasticache:DescribeCacheClusters
                Resource : "*"
  WebServerKeys :
    Type : AWS::IAM::AccessKey
    Properties :
      UserName :
        Ref: WebServerUser

Outputs:
  WebsiteURL:
    Description: sample output only here to show inline string function parsing
    Value: |
      http://`{ "Fn::GetAtt" : [ "AWSEBLoadBalancer", "DNSName" ] }`
  MyElastiCacheName:
    Description: Name of the elasticache
    Value:
      Ref : MyElastiCache
  NumCacheNodes:
```

```
    Description: Number of cache nodes in MyElastiCache
    Value:
      Fn::GetOptionSetting:
        OptionName : NumCacheNodes
        DefaultValue: 1

files:
  "/etc/cfn/cfn-credentials" :
    content : |
      AWSAccessKeyId=`{ "Ref" : "WebServerKeys" }`
      AWSSecretKey=`{ "Fn::GetAtt" : ["WebServerKeys", "SecretAccessKey"] }`
    mode : "000400"
    owner : root
    group : root

  "/etc/cfn/get-cache-nodes" :
    content : |
      # Define environment variables for command line tools
      export AWS_ELASTICACHE_HOME="/home/ec2-user/elasticache/$(ls /home/ec2-user/
elasticache/)"
      export AWS_CLOUDFORMATION_HOME=/opt/aws/apitools/cfn
      export PATH=$AWS_CLOUDFORMATION_HOME/bin:$AWS_ELASTICACHE_HOME/bin:$PATH
      export AWS_CREDENTIAL_FILE=/etc/cfn/cfn-credentials
      export JAVA_HOME=/usr/lib/jvm/jre

      # Grab the Cache node names and configure the PHP page
      aws cloudformation list-stack-resources --stack `{ "Ref" :
 "AWS::StackName" }` --region `{ "Ref" : "AWS::Region" }` --output text | grep
 MyElastiCache | awk '{print $4}' | xargs -I {} aws elasticache describe-cache-
clusters --cache-cluster-id {} --region `{ "Ref" : "AWS::Region" }` --show-
cache-node-info --output text | grep '^ENDPOINT' | awk '{print $2 ":" $3}' >
 `{ "Fn::GetOptionSetting" : { "OptionName" : "NodeListPath", "DefaultValue" : "/
var/www/html/nodelist" } }`
    mode : "000500"
    owner : root
    group : root

  "/etc/cfn/hooks.d/cfn-cache-change.conf" :
    "content": |
      [cfn-cache-size-change]
      triggers=post.update
      path=Resources.AWSEBAutoScalingGroup.Metadata.ElastiCacheConfig
      action=/etc/cfn/get-cache-nodes
      runas=root
```

```
sources :
  "/home/ec2-user/elasticache" : "https://elasticache-downloads.s3.amazonaws.com/
AmazonElastiCacheCli-latest.zip"

commands:
  make-elasticache-executable:
    command: chmod -R ugo+x /home/ec2-user/elasticache/*/bin/*

packages :
  "yum" :
    "aws-apitools-cfn"  : []

container_commands:
  initial_cache_nodes:
    command: /etc/cfn/get-cache-nodes
```

3. Di komputer lokal Anda, buat file konfigurasi `nodejs-example-express-elasticache/.ebextensions/elasticache_settings.config` dengan cuplikan berikut untuk dikonfigurasi. ElastiCache

```
option_settings:
  "aws:elasticbeanstalk:customoption":
     CacheNodeType: cache.t2.micro
     NumCacheNodes: 1
     Engine: memcached
     NodeListPath: /var/nodelist
```

4. Di komputer lokal Anda, ganti `nodejs-example-express-elasticache/express-app.js` dengan snippet berikut. File ini membaca daftar simpul dari disk (`/var/nodelist`) dan mengonfigurasi express untuk menggunakan `memcached` sebagai penyimpanan sesi jika ada simpul. File Anda akan terlihat seperti berikut.

```
/**
 * Module dependencies.
 */

var express = require('express'),
    session = require('express-session'),
    bodyParser = require('body-parser'),
    methodOverride = require('method-override'),
    cookieParser = require('cookie-parser'),
```

```
    fs = require('fs'),
    filename = '/var/nodelist',
    app = module.exports = express();

var MemcachedStore = require('connect-memcached')(session);

function setup(cacheNodes) {
  app.use(bodyParser.raw());
  app.use(methodOverride());
  if (cacheNodes) {
      app.use(cookieParser());

      console.log('Using memcached store nodes:');
      console.log(cacheNodes);

      app.use(session({
          secret: 'your secret here',
          resave: false,
          saveUninitialized: false,
          store: new MemcachedStore({'hosts': cacheNodes})
      }));
  } else {
    console.log('Not using memcached store.');
    app.use(cookieParser('your secret here'));
    app.use(session());
  }

  app.get('/', function(req, resp){
  if (req.session.views) {
      req.session.views++
      resp.setHeader('Content-Type', 'text/html')
      resp.write('Views: ' + req.session.views)
      resp.end()
   } else {
      req.session.views = 1
      resp.end('Refresh the page!')
    }
  });

  if (!module.parent) {
      console.log('Running express without cluster.');
      app.listen(process.env.PORT || 5000);
  }
}
```

```
// Load elasticache configuration.
fs.readFile(filename, 'UTF8', function(err, data) {
    if (err) throw err;

    var cacheNodes = [];
    if (data) {
        var lines = data.split('\n');
        for (var i = 0 ; i < lines.length ; i++) {
            if (lines[i].length > 0) {
                cacheNodes.push(lines[i]);
            }
        }
    }
    setup(cacheNodes);
});
```

5. Di komputer lokal Anda, perbarui `package.json` dengan konten berikut:

```
  "dependencies": {
    "cookie-parser": "~1.4.4",
    "debug": "~2.6.9",
    "express": "~4.16.1",
    "http-errors": "~1.6.3",
    "jade": "~1.11.0",
    "morgan": "~1.9.1",
    "connect-memcached": "*",
    "express-session": "*",
    "body-parser": "*",
    "method-override": "*"
  }
```

6. Jalankan npm install.

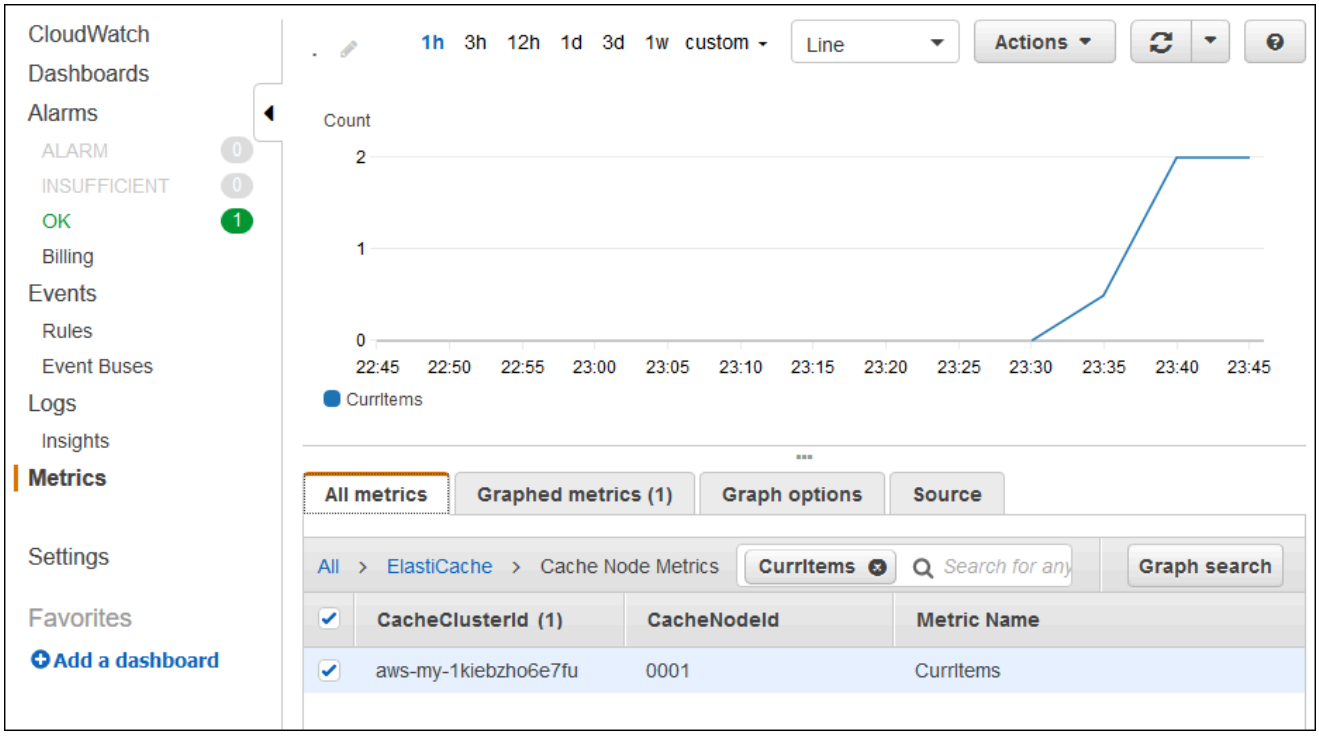
```
~/nodejs-example-express-elasticache$ npm install
```

7. Men-deploy aplikasi yang diperbarui.

```
~/nodejs-example-express-elasticache$ eb deploy
```

8. Lingkungan Anda akan diperbarui setelah beberapa menit. Setelah lingkungan Anda berwarna hijau dan siap, verifikasi apakah kode berfungsi.

a. Periksa CloudWatch konsol Amazon untuk melihat ElastiCache metrik Anda. Untuk melihat ElastiCache metrik, pilih Metrik di panel kiri, lalu cari. CurrItems Pilih ElastiCache > Metrik Node Cache, lalu pilih node cache Anda untuk melihat jumlah item dalam cache.



> **Note**
>
> Pastikan Anda melihat wilayahnya sama dengan tempat Anda men-deploy aplikasi Anda.

Jika Anda menyalin dan menempelkan URL aplikasi Anda ke browser web lain dan menyegarkan halaman, Anda akan melihat CurrItem jumlah Anda naik setelah 5 menit.

b. Ambil snapshot log Anda. Untuk informasi selengkapnya tentang mendapatkan log kembali, lihat Melihat log dari EC2 instans Amazon di lingkungan Elastic Beanstalk Anda.

c. Memeriksa file `/var/log/nodejs/nodejs.log` dalam paket log. Anda akan melihat sesuatu yang serupa dengan yang berikut:

```
Using memcached store nodes:
[ 'aws-my-1oys9co8zt1uo.1iwtrn.0001.use1.cache.amazonaws.com:11211' ]
```

Pembersihan

Jika aplikasi tidak ingin dijalankan, Anda dapat membersihkan dengan mengakhiri lingkungan dan menghapus aplikasi Anda.

Gunakan perintah `eb terminate` untuk mengakhiri lingkungan Anda dan perintah `eb delete` untuk menghapus aplikasi Anda.

Untuk mengakhiri lingkungan Anda

Dari direktori tempat Anda membuat repositori lokal, jalankan `eb terminate`.

```
$ eb terminate
```

Proses ini dapat menghabiskan waktu beberapa menit. Elastic Beanstalk menampilkan pesan setelah lingkungan berhasil diakhiri.

Men-deploy aplikasi Node.js dengan DynamoDB ke Elastic Beanstalk

Tutorial ini dan contoh nodejs-example-dynamoaplikasi.zip memandu Anda melalui proses penerapan aplikasi Node.js yang menggunakan AWS SDK untuk JavaScript di Node.js untuk berinteraksi dengan layanan Amazon DynamoDB. Anda akan membuat tabel DynamoDB yang ada di database yang dipisahkan, atau eksternal, dari lingkungan. AWS Elastic Beanstalk Anda juga akan mengonfigurasi aplikasi untuk menggunakan database terpisah. Dalam lingkungan produksi, praktik terbaik adalah menggunakan database yang dipisahkan dari lingkungan Elastic Beanstalk sehingga independen dari siklus hidup lingkungan. Praktek ini juga memungkinkan Anda untuk melakukan penyebaran biru/hijau.

Contoh aplikasi menggambarkan hal berikut:

- Tabel DynamoDB yang menyimpan data teks yang disediakan pengguna.
- File konfigurasi untuk membuat tabel.
- Topik Layanan Pemberitahuan Sederhana Amazon.
- Penggunaan file package.json untuk menginstal paket selama penerapan.

Bagian-bagian

- Prasyarat
- Membuat lingkungan Elastic Beanstalk
- Menambahkan izin ke instans lingkungan Anda
- Menyebarkan contoh aplikasi
- Buat tabel DynamoDB
- Perbarui file konfigurasi aplikasi
- Konfigurasikan lingkungan Anda untuk ketersediaan yang tinggi
- Pembersihan
- Langkah selanjutnya

Prasyarat

Tutorial ini membutuhkan prasyarat berikut:

- Runtime Node.js
- Perangkat lunak pengelola paket Node.js default, npm
- Generator baris perintah Express
- Antarmuka Baris Perintah Elastic Beanstalk (EB CLI)

Untuk detail tentang menginstal tiga komponen pertama yang terdaftar dan menyiapkan lingkungan pengembangan lokal Anda, lihatMenyiapkan lingkungan pengembangan Node.js Anda untuk Elastic Beanstalk. Untuk tutorial ini, Anda tidak perlu menginstal AWS SDK untuk Node.js, yang juga disebutkan dalam topik yang direferensikan.

Untuk detail tentang menginstal dan mengonfigurasi CLI EB, lihat dan. Instal EB CLI dengan skrip pengaturan (disarankan) Mengonfigurasi EB CLI

Membuat lingkungan Elastic Beanstalk

Direktori aplikasi Anda

Tutorial ini menggunakan direktori yang disebut `nodejs-example-dynamo` untuk bundel sumber aplikasi. Buat `nodejs-example-dynamo` direktori untuk tutorial ini.

```
~$ mkdir nodejs-example-dynamo
```

> **Note**
>
> Setiap tutorial dalam chapter ini menggunakan direktorinya sendiri untuk bundel sumber aplikasi. Nama direktori cocok dengan nama aplikasi sampel yang digunakan oleh tutorial.

Ubah direktori kerja Anda saat ini menjadi`nodejs-example-dynamo`.

```
~$ cd nodejs-example-dynamo
```

Sekarang, mari kita mengatur lingkungan Elastic Beanstalk yang menjalankan platform Node.js dan aplikasi sampel. Kita akan menggunakan antarmuka baris perintah Elastic Beanstalk (EB CLI).

Untuk mengonfigurasi repositori EB CLI untuk aplikasi Anda dan membuat lingkungan Elastic Beanstalk yang menjalankan platform Node.js

1. Buat repositori dengan perintah eb init.

   ```
   ~/nodejs-example-dynamo$ eb init --platform node.js --region <region>
   ```

   Perintah ini membuat file konfigurasi dalam folder bernama `.elasticbeanstalk` yang menentukan pengaturan dalam membuat lingkungan untuk aplikasi Anda, dan membuat sebuah aplikasi Elastic Beanstalk yang diambil dari nama folder saat ini.
2. Buat lingkungan yang menjalankan aplikasi sampel dengan perintah eb create.

   ```
   ~/nodejs-example-dynamo$ eb create --sample nodejs-example-dynamo
   ```

   Perintah ini membuat lingkungan yang seimbang beban dengan pengaturan default untuk platform Node.js dan sumber daya berikut:

   - EC2 instance — Mesin virtual Amazon Elastic Compute Cloud (Amazon EC2) yang dikonfigurasi untuk menjalankan aplikasi web pada platform yang Anda pilih.

     Setiap platform menjalankan satu set perangkat lunak, file konfigurasi dan penulisan tertentu untuk mendukung versi bahasa, kerangka kerja, kontainer web tertentu, atau kombinasi dari semua ini. Sebagian besar platform menggunakan Apache atau NGINX sebagai proksi terbalik yang ada di depan aplikasi web Anda, meneruskan permintaan ke aplikasi web, menyajikan aset statis, dan menghasilkan log akses dan kesalahan.

- Grup keamanan instans — Grup EC2 keamanan Amazon yang dikonfigurasi untuk memungkinkan lalu lintas masuk pada port 80. Sumber daya ini memungkinkan lalu lintas HTTP dari penyeimbang beban mencapai EC2 instance yang menjalankan aplikasi web Anda. Secara default, lalu lintas tidak diizinkan di port lain.
- Penyeimbang beban – Penyeimbang beban Elastic Load Balancing yang dikonfigurasi untuk mendistribusikan permintaan ke instans yang menjalankan aplikasi Anda. Penyeimbang beban juga menghilangkan kebutuhan untuk mengekspos instans Anda langsung ke internet.
- Grup keamanan penyeimbang beban - Grup EC2 keamanan Amazon yang dikonfigurasi untuk memungkinkan lalu lintas masuk di port 80. Sumber daya ini memungkinkan lalu lintas HTTP dari internet mencapai penyeimbang beban. Secara default, lalu lintas tidak diizinkan di port lain.
- Grup Auto Scaling – Grup Auto Scaling yang dikonfigurasi untuk menggantikan instans jika diakhiri atau menjadi tidak tersedia.
- Bucket Amazon S3 – Lokasi penyimpanan untuk kode sumber, log, dan artifact lainnya yang dibuat saat Anda menggunakan Elastic Beanstalk.
- CloudWatch Alarm Amazon — Dua CloudWatch alarm yang memantau beban pada instans di lingkungan Anda dan yang dipicu jika beban terlalu tinggi atau terlalu rendah. Saat alarm terpicu, grup Auto Scaling Anda akan menaikkan atau menurunkan skala sebagai respons.
- AWS CloudFormation stack - Elastic AWS CloudFormation Beanstalk digunakan untuk meluncurkan sumber daya di lingkungan Anda dan menyebarkan perubahan konfigurasi. Sumber daya ditentukan di sebuah templat yang dapat Anda lihat di Konsol AWS CloudFormation.
- Nama domain — Nama domain yang merutekan ke aplikasi web Anda dalam formulir *subdomain*.*region*.elasticbeanstalk.com.

> **Keamanan domain**
>
> Untuk meningkatkan keamanan aplikasi Elastic Beanstalk Anda, domain elasticbeanstalk.com terdaftar di Daftar Akhiran Publik (PSL).
> Jika Anda perlu mengatur cookie sensitif di nama domain default untuk aplikasi Elastic Beanstalk Anda, kami sarankan Anda menggunakan cookie `__Host-` dengan awalan untuk meningkatkan keamanan. Praktik ini melindungi domain Anda dari upaya pemalsuan permintaan lintas situs (CSRF). Untuk informasi selengkapnya, lihat halaman Set-Cookie di Jaringan Pengembang Mozilla.

3. Ketika pembuatan lingkungan selesai, gunakan perintah eb open untuk membuka URL lingkungan di browser default.

```
~/nodejs-example-dynamo$ eb open
```

Anda sekarang telah membuat lingkungan Elastic Beanstalk Node.js dengan aplikasi sampel. Anda dapat memperbaruinya dengan aplikasi Anda sendiri. Selanjutnya, kami memperbarui aplikasi sampel untuk menggunakan kerangka kerja Express.

Menambahkan izin ke instans lingkungan Anda

Aplikasi Anda berjalan pada satu atau beberapa EC2 instance di belakang penyeimbang beban, yang melayani permintaan HTTP dari Internet. Ketika menerima permintaan yang mengharuskannya untuk menggunakan AWS layanan, aplikasi menggunakan izin dari instance yang dijalankannya untuk mengakses layanan tersebut.

Aplikasi contoh menggunakan izin instance untuk menulis data ke tabel DynamoDB, dan mengirim notifikasi ke topik Amazon SNS dengan SDK untuk di Node.js. JavaScript Tambahkan kebijakan terkelola berikut ke profil instans default untuk memberikan izin EC2 instans di lingkungan Anda untuk mengakses DynamoDB dan Amazon SNS:

- AmazonDynamoDBFullAkses
- SNSFullAkses Amazon

Untuk menambahkan kebijakan ke profil instans default

1. Buka Halaman peran di konsol IAM.
2. Pilih aws-elasticbeanstalk-ec2-peran.
3. Di tab Izin, pilih Lampirkan kebijakan.
4. Pilih kebijakan terkelola untuk layanan tambahan yang digunakan aplikasi Anda. Untuk tutorial ini, pilih `AmazonSNSFullAccess` dan`AmazonDynamoDBFullAccess`.
5. Memilih Lampirkan kebijakan.

Lihat Mengelola profil instans Elastic Beanstalk untuk selengkapnya tentang mengelola profil instans.

Menyebarkan contoh aplikasi

Sekarang lingkungan Anda siap untuk Anda gunakan dan jalankan contoh aplikasi untuk tutorial ini: nodejs-example-dynamo.zip.

Untuk menyebarkan dan menjalankan aplikasi contoh tutorial

1. Ubah direktori kerja Anda saat ini ke direktori aplikasi`nodejs-example-dynamo`.

   ```
   ~$ cd nodejs-example-dynamo
   ```

2. Unduh dan ekstrak isi contoh bundel sumber aplikasi nodejs-example-dynamo.zip ke direktori `nodejs-example-dynamo` aplikasi.
3. Terapkan aplikasi contoh ke lingkungan Elastic Beanstalk Anda dengan perintah. eb deploy

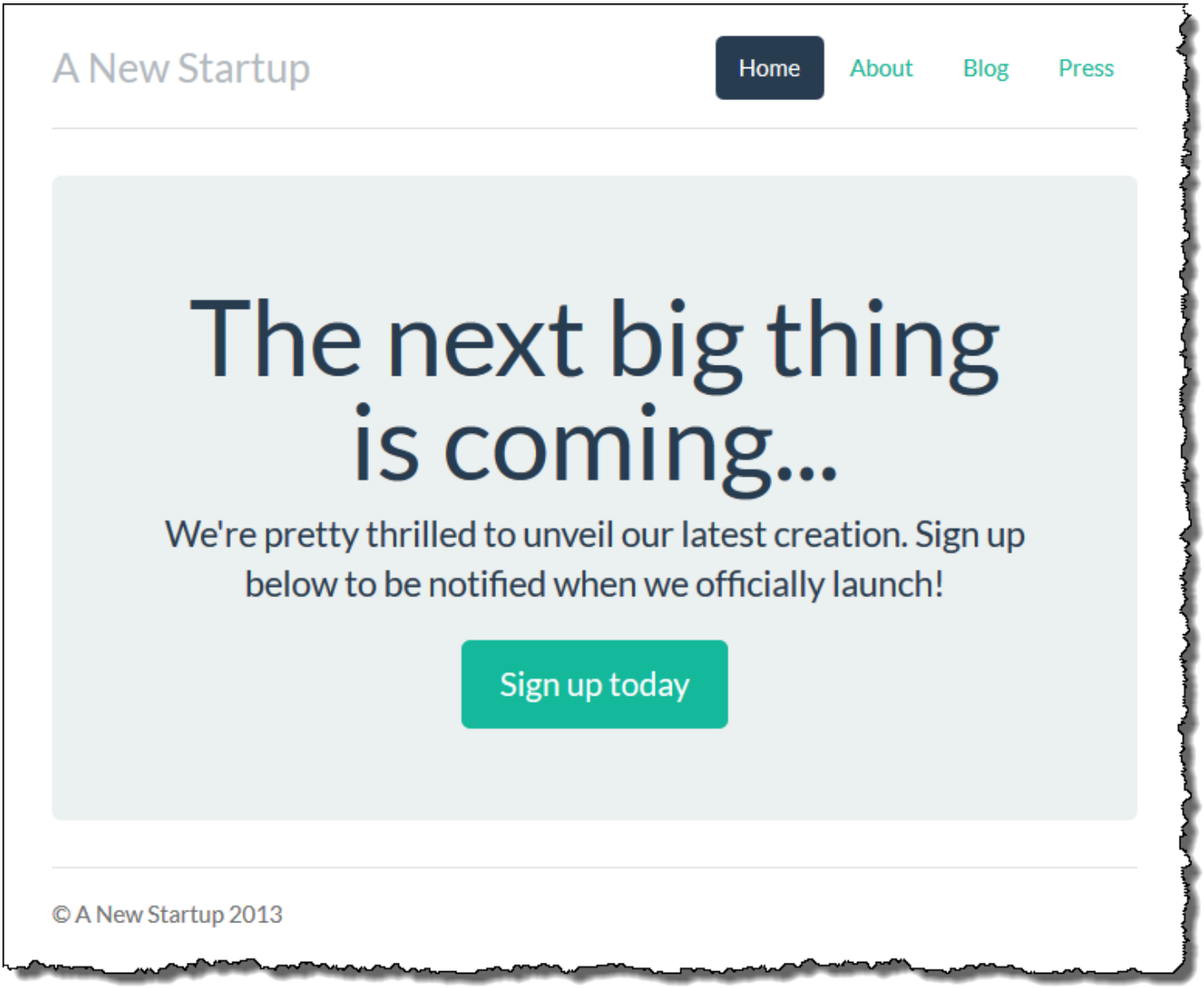
   ```
   ~/nodejs-example-dynamo$ eb deploy
   ```

   > **Note**
   >
   > Secara default, `eb deploy` perintah membuat file ZIP dari folder proyek Anda. Anda dapat mengonfigurasi EB CLI untuk men-deploy Artifact dari proses membangun Anda bukan membuat file ZIP folder proyek Anda. Untuk informasi selengkapnya, lihat Men-deploy artifact bukan folder proyek.

4. Ketika pembuatan lingkungan selesai, gunakan perintah eb open untuk membuka URL lingkungan di browser default.

   ```
   ~/nodejs-example-dynamo$ eb open
   ```

Situs mengumpulkan informasi kontak pengguna dan menggunakan tabel DynamoDB untuk menyimpan data. Untuk menambahkan entri, pilih Daftar hari ini, masukkan nama dan alamat email, lalu pilih Daftar!. Aplikasi web menulis isi formulir ke tabel dan memicu notifikasi email Amazon SNS.



Sekarang, topik Amazon SNS dikonfigurasi dengan email placeholder untuk pemberitahuan. Anda akan memperbarui konfigurasi segera, tetapi sementara itu Anda dapat memverifikasi tabel DynamoDB dan topik Amazon SNS di AWS Management Console.

Untuk melihat tabel

1. Buka halaman Tabel di konsol DynamoDB.
2. Temukan tabel yang dibuat aplikasi. Nama dimulai dengan awseb dan berisi. StartupSignupsTable
3. Pilih tabel, memilih Item, lalu pilih Mulai pencarian untuk menampilkan semua item dalam tabel.

Tabel berisi entri untuk setiap alamat email yang dikirimkan pada situs pendaftaran. Selain menulis ke tabel, aplikasi mengirim pesan ke topik Amazon SNS yang memiliki dua langganan, satu untuk pemberitahuan email kepada Anda, dan satu lagi untuk antrean Amazon Simple Queue Service yang dapat dibaca aplikasi pekerja untuk memproses permintaan dan mengirim email ke pelanggan yang tertarik.

Untuk melihat topik

1. Buka halaman Topik di konsol Amazon SNS.
2. Temukan topik yang dibuat oleh aplikasi. Nama dimulai dengan awseb dan berisi. NewSignupTopic
3. Pilih topik untuk melihat langganannya.

Aplikasi (`app.js`) mendefinisikan dua rute. Root path (`/`) mengembalikan halaman web yang dirender dari template Embedded JavaScript (EJS) dengan formulir yang diisi pengguna untuk mendaftarkan nama dan alamat email mereka. Dengan mengirim formulir maka akan mengirimkan permintaan POST dengan data formulir ke rute `/signup`, yang menulis entri ke tabel DynamoDB dan menerbitkan pesan ke topik Amazon SNS untuk memberitahu pemilik pendaftaran.

Aplikasi sampel menyertakan file konfigurasi yang membuat tabel DynamoDB, topik Amazon SNS, dan antrean Amazon SQS yang digunakan oleh aplikasi. Hal ini memungkinkan Anda membuat lingkungan baru dan menguji fungsionalitasnya dengan segera, tetapi memiliki kelemahan mengikat tabel DynamoDB ke lingkungan. Untuk lingkungan produksi, Anda harus membuat tabel DynamoDB di luar lingkungan untuk menghindari kehilangan itu ketika Anda mengakhiri lingkungan atau memperbarui konfigurasinya.

Buat tabel DynamoDB

Untuk menggunakan tabel DynamoDB eksternal dengan aplikasi yang berjalan di Elastic Beanstalk, pertama buat tabel di DynamoDB. Ketika Anda membuat tabel di luar Elastic Beanstalk, tabel tersebut sepenuhnya independen dari Elastic Beanstalk dan lingkungan Elastic Beanstalk Anda, dan tidak akan diakhiri oleh Elastic Beanstalk.

Buat tabel dengan pengaturan berikut:

- Nama tabel – **`nodejs-tutorial`**
- Kunci utama – **`email`**
- Tipe kunci primer – String

Untuk membuat tabel DynamoDB

1. Buka Halaman tabel di konsol manajemen DynamoDB.
2. Pilih Buat tabel.
3. Ketik Nama tabel dan Kunci utama.
4. Pilih tipe kunci primer.
5. Pilih Buat.

Perbarui file konfigurasi aplikasi

Perbarui file konfigurasi di sumber aplikasi untuk menggunakan tabel nodejs-tutorial daripada membuat yang baru.

Untuk memperbarui aplikasi contoh untuk penggunaan produksi

1. Ubah direktori kerja Anda saat ini ke direktori aplikasi`nodejs-example-dynamo`.

   ```
   ~$ cd nodejs-example-dynamo
   ```

2. Buka `.ebextensions/options.config` dan ubah nilai pengaturan berikut:

   - NewSignupEmail— Alamat email Anda.
   - STARTUP_SIGNUP_TABLE – nodejs-tutorial

   Example .ebextensions/options.config

   ```
   option_settings:
     aws:elasticbeanstalk:customoption:
       NewSignupEmail: you@example.com
     aws:elasticbeanstalk:application:environment:
       THEME: "flatly"
       AWS_REGION: '`{"Ref" : "AWS::Region"}`'
       STARTUP_SIGNUP_TABLE: nodejs-tutorial
       NEW_SIGNUP_TOPIC: '`{"Ref" : "NewSignupTopic"}`'
     aws:elasticbeanstalk:container:nodejs:
       ProxyServer: nginx
     aws:elasticbeanstalk:container:nodejs:staticfiles:
       /static: /static
   ```

```
  aws:autoscaling:asg:
    Cooldown: "120"
  aws:autoscaling:trigger:
    Unit: "Percent"
    Period: "1"
    BreachDuration: "2"
    UpperThreshold: "75"
    LowerThreshold: "30"
    MeasureName: "CPUUtilization"
```

Ini berlaku konfigurasi berikut untuk aplikasi:

- Alamat email yang digunakan topik Amazon SNS untuk notifikasi diatur ke alamat Anda, atau alamat yang Anda masukkan dalam file. `options.config`
- Tabel nodejs-tutorial akan digunakan sebagai pengganti yang dibuat oleh. `.ebextensions/create-dynamodb-table.config`

3. Hapus `.ebextensions/create-dynamodb-table.config`.

```
~/nodejs-tutorial$ rm .ebextensions/create-dynamodb-table.config
```

Lain kali Anda men-deploy aplikasi, tabel yang dibuat oleh file konfigurasi ini akan dihapus.

4. Terapkan aplikasi yang diperbarui ke lingkungan Elastic Beanstalk Anda dengan perintah. eb deploy

```
~/nodejs-example-dynamo$ eb deploy
```

5. Ketika pembuatan lingkungan selesai, gunakan perintah eb open untuk membuka URL lingkungan di browser default.

```
~/nodejs-example-dynamo$ eb open
```

Ketika Anda men-deploy, Elastic Beanstalk memperbarui konfigurasi topik Amazon SNS dan menghapus tabel DynamoDB yang dibuat ketika Anda men-deploy versi pertama aplikasi.

Sekarang, ketika Anda mengakhiri lingkungan, tabel nodejs-tutorial tidak akan dihapus. Hal ini memungkinkan Anda melakukan deployment biru/hijau, memodifikasi file konfigurasi, atau menghapus situs web Anda tanpa risiko kehilangan data.

Buka situs Anda di peramban dan verifikasi apakah formulir tersebut bekerja seperti yang diharapkan. Membuat beberapa entri, dan kemudian periksa konsol DynamoDB untuk memverifikasi tabel.

Untuk melihat tabel

1. Buka Halaman tabel di konsol DynamoDB.
2. Temukan tabel nodejs-tutorial.
3. Pilih tabel, memilih Item, lalu pilih Melai pencarian untuk menampilkan semua item dalam tabel.

Anda juga dapat melihat bahwa Elastic Beanstalk menghapus tabel yang dibuat sebelumnya.

Konfigurasikan lingkungan Anda untuk ketersediaan yang tinggi

Terakhir, konfigurasikan grup Auto Scaling lingkungan Anda dengan jumlah instans minimum yang lebih tinggi. Jalankan setidaknya dua instans setiap saat untuk mencegah terjadinya kegagalan di satu titik server web di lingkungan Anda, dan mengizinkan Anda untuk men-deploy perubahan tanpa membuat situs Anda keluar dari layanan.

Mengonfigurasi grup Auto Scaling lingkungan Anda untuk ketersediaan yang tinggi

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Pada kategori konfigurasi Kapasitas, pilih Edit.
5. Pada bagian Grup Auto Scaling, set Instans minimum ke **2**.
6. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Pembersihan

Setelah Anda selesai bekerja dengan kode demo, Anda dapat mengakhiri lingkungan Anda. Elastic Beanstalk menghapus semua sumber daya AWS terkait, seperti instans EC2 Amazon, instans database, penyeimbang beban, grup keamanan,dan alarm.

Menghapus sumber daya tidak menghapus aplikasi Elastic Beanstalk, sehingga Anda dapat membuat lingkungan baru untuk aplikasi Anda kapan saja.

Untuk mengakhiri lingkungan Elastic Beanstalk Anda dari konsol

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pilih Tindakan, lalu pilih Terminate environment.
4. Gunakan kotak dialog di layar untuk mengonfirmasi pengakhiran lingkungan.

Anda juga dapat menghapus tabel DynamoDB eksternal yang Anda buat.

Untuk menghapus tabel DynamoDB

1. Buka Halaman tabel di konsol DynamoDB.
2. Pilih tabel.
3. Pilih Tindakan, dan lalu Hapus tabel.
4. Pilih Hapus.

Langkah selanjutnya

Contoh aplikasi menggunakan file konfigurasi untuk mengonfigurasi pengaturan perangkat lunak dan membuat AWS sumber daya sebagai bagian dari lingkungan Anda. Lihat Penyesuaian lingkungan lanjutan dengan file konfigurasi (`.ebextensions`) untuk informasi selengkapnya tentang file konfigurasi dan penggunaannya.

Contoh aplikasi untuk tutorial ini menggunakan kerangka web Express untuk Node.js. Untuk informasi selengkapnya tentang Express, lihat dokumentasi resmi di expressjs.com.

Terakhir, jika Anda berencana menggunakan aplikasi dalam lingkungan produksi, konfigurasikan nama domain khusus untuk lingkungan Anda dan aktifkan HTTPS untuk koneksi yang aman.

Menambahkan instans Amazon RDS DB ke lingkungan Node.js Elastic Beanstalk Anda

Topik ini memberikan instruksi untuk membuat Amazon RDS menggunakan konsol Elastic Beanstalk. Anda dapat menggunakan instans DB Amazon Relational Database Service (Amazon RDS) untuk menyimpan data yang dikumpulkan dan dimodifikasi oleh aplikasi Anda. Basis data dapat

digabungkan ke lingkungan Anda dan dikelola oleh Elastic Beanstalk, atau dapat dibuat sebagai dipisahkan dan dikelola secara eksternal oleh layanan lain. Dalam instruksi ini database digabungkan dengan lingkungan Anda dan dikelola oleh Elastic Beanstalk. Untuk informasi lebih lanjut tentang mengintegrasikan Amazon RDS dengan Elastic Beanstalk, lihat. Menambahkan basis data ke lingkungan Elastic Beanstalk Anda

Bagian-bagian

- Menambahkan instans DB ke lingkungan Anda
- Mengunduh driver
- Menyambungkan ke basis data

Menambahkan instans DB ke lingkungan Anda

Untuk menambahkan instans DB ke lingkungan Anda

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Di kategori konfigurasi Basis data, pilih Edit.
5. Pilih mesin DB, dan masukkan nama pengguna dan kata sandi.
6. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Menambahkan instans DB memakan waktu sekitar 10 menit. Ketika pembaruan lingkungan selesai, nama host instans DB dan informasi koneksi lainnya tersedia untuk aplikasi Anda melalui properti lingkungan berikut:

Nama properti	Deskripsi	Nilai properti
`RDS_HOSTNAME`	Nama host instans DB.	Di tab Konektivitas & keamanan di konsol Amazon RDS: Titik akhir.

Nama properti	Deskripsi	Nilai properti
`RDS_PORT`	Port tempat instans DB menerima koneksi. Nilai default bervariasi di antara mesin DB.	Di tab Konektivitas & keamanan di konsol Amazon RDS: Port.
`RDS_DB_NAME`	Nama basis data, **ebdb**.	Di tab Konfigurasi di konsol Amazon RDS: Nama DB.
`RDS_USERNAME`	Nama pengguna yang Anda konfigurasi untuk basis data Anda.	Di tab Konfigurasi di konsol Amazon RDS: Nama pengguna utama.
`RDS_PASSWORD`	Kata sandi yang Anda konfigurasi untuk basis data Anda.	Tidak tersedia untuk referensi di konsol Amazon RDS.

Untuk informasi selengkapnya tentang mengonfigurasi instance database yang digabungkan dengan lingkungan Elastic Beanstalk, lihat. Menambahkan basis data ke lingkungan Elastic Beanstalk Anda

Mengunduh driver

Tambahkan driver basis data ke file `package.json` proyek Anda di bawah `dependencies`.

Example **`package.json`** – Express dengan MySQL

```
{
  "name": "my-app",
  "version": "0.0.1",
  "private": true,
  "dependencies": {
    "ejs": "latest",
    "aws-sdk": "latest",
    "express": "latest",
    "body-parser": "latest",
    "mysql": "latest"
  },
  "scripts": {
    "start": "node app.js"
```

```
  }
}
```

Paket driver umum untuk Node.js

- MySQL – mysql
- PostgreSQL – node-postgres
- SQL Server – node-mssql
- Oracle – node-oracledb

Menyambungkan ke basis data

Elastic Beanstalk memberikan informasi koneksi untuk instans DB terlampir di properti lingkungan. Gunakan `process.env.`*`VARIABLE`* untuk membaca properti dan mengonfigurasi koneksi basis data.

Example app.js – Koneksi basis data MySQL

```
var mysql = require('mysql');

var connection = mysql.createConnection({
  host     : process.env.RDS_HOSTNAME,
  user     : process.env.RDS_USERNAME,
  password : process.env.RDS_PASSWORD,
  port     : process.env.RDS_PORT
});

connection.connect(function(err) {
  if (err) {
    console.error('Database connection failed: ' + err.stack);
    return;
  }

  console.log('Connected to database.');
});

connection.end();
```

Untuk informasi selengkapnya tentang membangun string koneksi menggunakan node-mysql, lihat npmjs. org/package/mysql.

Alat dan sumber daya Node.js

Ada beberapa tempat yang dapat dikunjungi untuk mendapatkan bantuan tambahan ketika mengembangkan aplikasi Node.js Anda:

Sumber Daya	Deskripsi
GitHub	Instal AWS SDK untuk Node.js menggunakan GitHub.
AWS SDK untuk Node.js (Pratinjau Pengembang)	Tempat yang lengkap mengakomodasikan segala kebutuhan Anda mulai dari kode sampel, dokumentasi, alat, dan sumber daya tambahan.

Menyebarkan aplikasi PHP dengan Elastic Beanstalk

Anda dapat menerapkan aplikasi PHP dalam beberapa menit menggunakan Elastic Beanstalk Command Line Interface (EB CLI) atau konsol Elastic Beanstalk. Untuk mempelajari caranya, mulailah dengan QuickStart tutorial, lalu tinjau contoh-contoh lanjutan.

Sumber daya untuk menyebarkan aplikasi PHP ke Elastic Beanstalk

- QuickStart untuk PHP— Step-by-step instruksi untuk menyebarkan aplikasi Hello World PHP menggunakan EB CLI.
- Menggunakan platform PHP Elastic Beanstalk— Cara menggunakan fitur platform khusus untuk PHP.
- Contoh lanjutan— Cara tutorial untuk kerangka kerja dan aplikasi PHP umum, ditambah cara menambahkan database Amazon RDS ke lingkungan Anda.

Untuk info lebih lanjut tentang pengembangan dengan PHP di AWS, lihat sumber daya berikut:

- GitHub— Instal AWS SDK for PHP GitHub menggunakan.
- PHP Developer Center — Alat, dokumen, dan kode sampel untuk mengembangkan aplikasi PHP. AWS
- AWS SDK for FAQs PHP — Dapatkan jawaban atas pertanyaan umum.

QuickStart: Menyebarkan aplikasi PHP ke Elastic Beanstalk

Dalam tutorial berikut, Anda akan belajar cara membuat dan menyebarkan contoh aplikasi PHP ke AWS Elastic Beanstalk lingkungan menggunakan EB CLI.

> ⚠ Peringatan - Bukan untuk penggunaan produksi!
>
> Contoh dimaksudkan untuk demonstrasi saja. Jangan gunakan contoh aplikasi dalam produksi.

AWS Akun Anda

Jika Anda belum menjadi AWS pelanggan, Anda perlu membuat AWS akun untuk menggunakan Elastic Beanstalk.

Buat AWS akun

Mendaftar untuk Akun AWS

Jika Anda tidak memiliki Akun AWS, selesaikan langkah-langkah berikut untuk membuatnya.

Untuk mendaftar untuk Akun AWS

1. Buka https://portal.aws.amazon.com/billing/pendaftaran.
2. Ikuti petunjuk online.

   Bagian dari prosedur pendaftaran melibatkan menerima panggilan telepon atau pesan teks dan memasukkan kode verifikasi pada keypad telepon.

   Saat Anda mendaftar untuk sebuah Akun AWS, sebuah Pengguna root akun AWSdibuat. Pengguna root memiliki akses ke semua Layanan AWS dan sumber daya di akun. Sebagai praktik keamanan terbaik, tetapkan akses administratif ke pengguna, dan gunakan hanya pengguna root untuk melakukan tugas yang memerlukan akses pengguna root.

AWS mengirimkan email konfirmasi setelah proses pendaftaran selesai. Kapan saja, Anda dapat melihat aktivitas akun Anda saat ini dan mengelola akun Anda dengan masuk https://aws.amazon.comke/ dan memilih Akun Saya.

Buat pengguna dengan akses administratif

Setelah Anda mendaftar Akun AWS, amankan Pengguna root akun AWS, aktifkan AWS IAM Identity Center, dan buat pengguna administratif sehingga Anda tidak menggunakan pengguna root untuk tugas sehari-hari.

Amankan Anda Pengguna root akun AWS

1. Masuk ke AWS Management Consolesebagai pemilik akun dengan memilih pengguna Root dan memasukkan alamat Akun AWS email Anda. Di laman berikutnya, masukkan kata sandi.

   Untuk bantuan masuk dengan menggunakan pengguna root, lihat Masuk sebagai pengguna root di AWS Sign-In Panduan Pengguna.

2. Mengaktifkan autentikasi multi-faktor (MFA) untuk pengguna root Anda.

   Untuk petunjuk, lihat Mengaktifkan perangkat MFA virtual untuk pengguna Akun AWS root (konsol) Anda di Panduan Pengguna IAM.

Buat pengguna dengan akses administratif

1. Aktifkan Pusat Identitas IAM.

   Untuk mendapatkan petunjuk, silakan lihat Mengaktifkan AWS IAM Identity Center di Panduan Pengguna AWS IAM Identity Center .

2. Di Pusat Identitas IAM, berikan akses administratif ke pengguna.

   Untuk tutorial tentang menggunakan Direktori Pusat Identitas IAM sebagai sumber identitas Anda, lihat Mengkonfigurasi akses pengguna dengan default Direktori Pusat Identitas IAM di Panduan AWS IAM Identity Center Pengguna.

Masuk sebagai pengguna dengan akses administratif

- Untuk masuk dengan pengguna Pusat Identitas IAM, gunakan URL masuk yang dikirim ke alamat email saat Anda membuat pengguna Pusat Identitas IAM.

  Untuk bantuan masuk menggunakan pengguna Pusat Identitas IAM, lihat Masuk ke portal AWS akses di Panduan AWS Sign-In Pengguna.

Tetapkan akses ke pengguna tambahan

1. Di Pusat Identitas IAM, buat set izin yang mengikuti praktik terbaik menerapkan izin hak istimewa paling sedikit.

   Untuk petunjuk, lihat Membuat set izin di Panduan AWS IAM Identity Center Pengguna.

2. Tetapkan pengguna ke grup, lalu tetapkan akses masuk tunggal ke grup.

   Untuk petunjuk, lihat Menambahkan grup di Panduan AWS IAM Identity Center Pengguna.

Prasyarat

- Antarmuka Baris Perintah Elastic Beanstalk - Untuk instalasi, lihat. Instal EB CLI dengan skrip pengaturan (disarankan)
- PHP - Instal PHP di mesin lokal Anda dengan mengikuti instruksi Instalasi dan Konfigurasi di situs web PHP.

Langkah 1: Buat aplikasi PHP

Untuk memulai cepat ini, Anda akan membuat aplikasi Hello World PHP.

Buat direktori proyek.

```
~$ mkdir eb-php
~$ cd eb-php
```

Selanjutnya, buat `index.php` file di direktori proyek dan tambahkan kode berikut.

Example **`index.php`**

```
<?php
  echo "Hello from a PHP application running in Elastic Beanstalk!";
?>
```

Langkah 2: Jalankan aplikasi Anda secara lokal

Gunakan perintah berikut untuk menjalankan aplikasi Anda secara lokal.

```
php -S localhost:5000
```

Buka browser untuk http://localhost:5000.

Anda akan melihat pesan halo Anda di browser dan mencatat pesan di terminal Anda.

Hentikan server lokal dengan masuk`Control+c`, sehingga Anda dapat menggunakan Elastic Beanstalk.

Langkah 3: Inisialisasi dan gunakan aplikasi PHP Anda

Selanjutnya, Anda akan menyebarkan aplikasi Anda ke lingkungan menggunakan konsol Elastic Beanstalk atau EB CLI. Untuk tutorial ini, Anda akan menggunakan EB CLI dengan opsi interaktif untuk menginisialisasi lingkungan.

Untuk menginisialisasi lingkungan Anda dan menciptakan lingkungan

1. Jalankan perintah init berikut.

   ```
   eb init -i
   ```

   Perintah init membuat aplikasi secara interaktif. Nama aplikasi akan default ke folder lokal yaitu`eb-php`.

   Untuk semua prompt, kecuali akses SSH, terima default untuk membuat lingkungan dengan versi platform PHP terbaru. Untuk mengatasi masalah, Anda dapat mengatur akses SSH dengan menjalankan kembali `eb init -i` perintah di lain waktu, atau menyambung menggunakan Amazon Instance EC2 Connect atau Session Manager.
2. Buat lingkungan dan terapkan aplikasi Anda

   Jalankan perintah berikut untuk membuat lingkungan bernama`blue-env`.

   ```
   eb create blue-env
   ```

   Ketika Anda menjalankan eb create perintah untuk pertama kalinya, Elastic Beanstalk secara otomatis membangun file zip aplikasi Anda, yang disebut bundel sumber. Selanjutnya, Elastic Beanstalk menciptakan lingkungan dengan satu atau lebih instance EC2 Amazon, dan kemudian menyebarkan aplikasi ke lingkungan.

   Menyebarkan aplikasi Anda ke Elastic Beanstalk mungkin memakan waktu hingga lima menit.

Langkah 4: Jelajahi aplikasi cloud Anda

Ketika proses untuk membuat lingkungan Anda selesai, aplikasi Anda harus berjalan dan mendengarkan permintaan pada port 5000. Connect ke aplikasi Anda dengan perintah berikut:

```
eb open
```

`eb open`Perintah membuka tab browser ke subdomain khusus yang dibuat untuk aplikasi Anda.

Langkah 5: Perbarui dan terapkan ulang aplikasi Anda

Setelah Anda membuat aplikasi dan disebarkan ke lingkungan, Anda dapat menerapkan versi baru aplikasi atau aplikasi lain kapan saja. Menerapkan versi aplikasi baru lebih cepat karena tidak memerlukan penyediaan atau memulai ulang instans Amazon. EC2

Perbarui kode PHP Anda untuk menyertakan nilai REQUEST_TIME dari lingkungan server:

```
<?php
  echo "Hello from a PHP application running in Elastic Beanstalk!";

  $timestamp = $_SERVER['REQUEST_TIME'];
  echo '<br/>Request time: ' . date('Y/m/d H:i:s', $timestamp);
?>
```

Redeploy kode PHP Anda ke Elastic Beanstalk dengan perintah berikut:

```
eb deploy
```

Ketika Anda menjalankan eb deploy, EB CLI memaketkan konten direktori proyek Anda dan men-deploy ke lingkungan Anda.

Setelah penerapan selesai, segarkan halaman atau sambungkan kembali ke aplikasi Anda. `eb open` Anda harus melihat pembaruan Anda. Jika tidak, pecahkan masalah dengan menjalankan server lokal Anda lagi untuk memverifikasi perubahan Anda.

> **Selamat!**
>
> Anda telah membuat, menerapkan, dan memperbarui aplikasi PHP dengan Elastic Beanstalk!

Bersihkan

Setelah Anda selesai bekerja dengan kode demo, Anda dapat mengakhiri lingkungan Anda. Elastic Beanstalk menghapus semua sumber daya AWS terkait, seperti instans EC2 Amazon, instans database, penyeimbang beban, grup keamanan,dan alarm.

Menghapus sumber daya tidak menghapus aplikasi Elastic Beanstalk, sehingga Anda dapat membuat lingkungan baru untuk aplikasi Anda kapan saja.

Untuk mengakhiri lingkungan Elastic Beanstalk Anda dari konsol

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pilih Tindakan, lalu pilih Terminate environment.
4. Gunakan kotak dialog di layar untuk mengonfirmasi pengakhiran lingkungan.

Atau, Anda dapat mengakhiri lingkungan Anda dengan EB CLI dengan perintah berikut:

```
eb terminate
```

Langkah selanjutnya

Anda dapat menjelajahi lingkungan aplikasi Anda menggunakan konsol Elastic Beanstalk. Untuk info selengkapnya, lihat Menjelajahi lingkungan Anda.

Untuk contoh lanjutan menggunakan PHP, lihatContoh lanjutan untuk PHP di Elastic Beanstalk.

Menggunakan platform PHP Elastic Beanstalk

AWS Elastic Beanstalk menyediakan dan mendukung berbagai cabang platform untuk berbagai versi PHP. Platform mendukung aplikasi web PHP yang berjalan berdiri sendiri atau di bawah Composer. Lihat PHP di dokumen AWS Elastic Beanstalk Platform untuk daftar lengkap cabang platform yang didukung.

Elastic Beanstalk menyediakan opsi konfigurasi yang dapat Anda gunakan untuk menyesuaikan perangkat lunak yang berjalan pada instans EC2 Amazon di lingkungan Elastic Beanstalk Anda. Anda dapat mengonfigurasi variabel lingkungan yang diperlukan oleh aplikasi Anda, mengaktifkan rotasi log ke Amazon S3, memetakan folder di sumber aplikasi Anda yang berisi file statis ke jalur yang disajikan oleh server proxy, dan mengatur pengaturan inisialisasi PHP umum.

Pilihan konfigurasi tersedia di konsol Elastic Beanstalk untuk memodifikasi konfigurasi dari lingkungan yang sedang berjalan. Agar Anda tidak kehilangan konfigurasi lingkungan ketika mengakhirinya, Anda dapat menggunakan konfigurasi tersimpan untuk menyimpan pengaturan dan kemudian menerapkannya ke lingkungan lain.

Untuk menyimpan pengaturan di kode sumber, Anda dapat menyertakan file konfigurasi. Pengaturan di file konfigurasi diterapkan setiap kali Anda membuat lingkungan atau men-deploy aplikasi Anda. Anda juga dapat menggunakan file konfigurasi untuk menginstal paket, menjalankan skrip, dan melakukan operasi penyesuaian instans lainnya selama deployment.

Jika Anda menggunakan Composer, Anda dapat menyertakan file `composer.json` di paket sumber untuk menginstal paket selama deployment.

Untuk konfigurasi PHP lanjutan dan pengaturan PHP yang tidak disediakan sebagai opsi konfigurasi, Anda dapat menggunakan file konfigurasi untuk menyediakan file `INI` yang dapat memperluas dan mengganti pengaturan default yang diterapkan oleh Elastic Beanstalk, atau menginstal ekstensi tambahan.

Pengaturan yang diterapkan di konsol Elastic Beanstalk menimpa pengaturan yang sama dalam file konfigurasi, jika ada. Hal tersebut memungkinkan Anda memiliki pengaturan default di file konfigurasi, dan mengganti keduanya dengan pengaturan khusus lingkungan di konsol tersebut. Untuk informasi selengkapnya tentang yang diutamakan, dan metode lain untuk mengubah pengaturan, lihat Opsi konfigurasi.

Untuk detail tentang berbagai cara memperluas platform berbasis Linux Elastic Beanstalk, lihat the section called “Memperluas platform Linux”.

Topik platform PHP

- Menginstal AWS SDK for PHP
- Pertimbangan untuk PHP 8.1 di Amazon Linux 2
- Mengonfigurasi lingkungan PHP Anda
- Ruang nama untuk konfigurasi
- Menginstal dependensi aplikasi PHP Elastic Beanstalk Anda
- Memperbarui Komposer pada Elastic Beanstalk
- Memperluas php.ini dalam konfigurasi Elastic Beanstalk Anda

Menginstal AWS SDK for PHP

Jika Anda perlu mengelola AWS sumber daya dari dalam aplikasi Anda, instal file AWS SDK untuk PHP. Misalnya, dengan SDK for PHP, Anda dapat menggunakan Amazon DynamoDB (DynamoDB) untuk menyimpan informasi pengguna dan sesi tanpa membuat basis data relasional.

Untuk menginstal SDK for PHP dengan Composer

```
$ composer require aws/aws-sdk-php
```

Untuk informasi lebih lanjut, lihat AWS SDK for PHPberanda. Untuk petunjuk, lihat Menginstal AWS SDK for PHP.

Pertimbangan untuk PHP 8.1 di Amazon Linux 2

Baca bagian ini jika Anda menggunakan PHP 8.1 di cabang platform Amazon Linux 2.

Pertimbangan untuk PHP 8.1 di Amazon Linux 2

> **Note**
>
> Informasi dalam topik ini hanya berlaku untuk PHP 8.1 di cabang platform Amazon Linux 2. Ini tidak berlaku untuk cabang platform PHP berdasarkan AL2 023. Ini juga tidak berlaku untuk cabang platform PHP 8.0 Amazon Linux 2.

Elastic Beanstalk menyimpan paket RPM terkait PHP 8.1 untuk PHP 8.1 di cabang platform Amazon Linux 2 EC2 pada instance di direktori lokal, bukan repositori Amazon Linux. Anda dapat menggunakan rpm -i untuk menginstal paket. Dimulai dengan PHP 8.1 Platform Versi 3.5.0, Elastic Beanstalk menyimpan paket RPM terkait PHP 8.1 di direktori lokal berikut. EC2

`/opt/elasticbeanstalk/RPMS`

Contoh berikut menginstal php-debuginfo paket.

```
$rpm -i /opt/elasticbeanstalk/RPMS/php-debuginfo-8.1.8-1.amzn2.x86_64.rpm
```

Versi dalam nama paket akan bervariasi sesuai dengan versi aktual yang tercantum di direktori EC2 lokal`/opt/elasticbeanstalk/RPMS`. Gunakan sintaks yang sama untuk menginstal paket PHP 8.1 RPM lainnya.

Perluas bagian berikut untuk menampilkan daftar paket RPM yang kami sediakan.

Paket RPM

Daftar berikut menyediakan paket RMP yang disediakan platform Elastic Beanstalk PHP 8.1 di Amazon Linux 2. Ini terletak di direktori lokal`/opt/elasticbeanstalk/RPMS`.

Nomor versi 8.1.8-1 dan 3.7.0-1 dalam nama paket yang tercantum hanyalah sebuah contoh.

- `php-8.1.8-1.amzn2.x86_64.rpm`
- `php-bcmath-8.1.8-1.amzn2.x86_64.rpm`
- `php-cli-8.1.8-1.amzn2.x86_64.rpm`
- `php-common-8.1.8-1.amzn2.x86_64.rpm`
- `php-dba-8.1.8-1.amzn2.x86_64.rpm`
- `php-dbg-8.1.8-1.amzn2.x86_64.rpm`
- `php-debuginfo-8.1.8-1.amzn2.x86_64.rpm`
- `php-devel-8.1.8-1.amzn2.x86_64.rpm`
- `php-embedded-8.1.8-1.amzn2.x86_64.rpm`
- `php-enchant-8.1.8-1.amzn2.x86_64.rpm`
- `php-fpm-8.1.8-1.amzn2.x86_64.rpm`
- `php-gd-8.1.8-1.amzn2.x86_64.rpm`
- `php-gmp-8.1.8-1.amzn2.x86_64.rpm`
- `php-intl-8.1.8-1.amzn2.x86_64.rpm`
- `php-ldap-8.1.8-1.amzn2.x86_64.rpm`
- `php-mbstring-8.1.8-1.amzn2.x86_64.rpm`
- `php-mysqlnd-8.1.8-1.amzn2.x86_64.rpm`
- `php-odbc-8.1.8-1.amzn2.x86_64.rpm`
- `php-opcache-8.1.8-1.amzn2.x86_64.rpm`
- `php-pdo-8.1.8-1.amzn2.x86_64.rpm`
- `php-pear-1.10.13-1.amzn2.noarch.rpm`
- `php-pgsql-8.1.8-1.amzn2.x86_64.rpm`
- `php-process-8.1.8-1.amzn2.x86_64.rpm`
- `php-pspell-8.1.8-1.amzn2.x86_64.rpm`
- `php-snmp-8.1.8-1.amzn2.x86_64.rpm`
- `php-soap-8.1.8-1.amzn2.x86_64.rpm`
- `php-sodium-8.1.8-1.amzn2.x86_64.rpm`
- `php-xml-8.1.8-1.amzn2.x86_64.rpm`

- `php-pecl-imagick-3.7.0-1.amzn2.x86_64.rpm`
- `php-pecl-imagick-debuginfo-3.7.0-1.amzn2.x86_64.rpm`
- `php-pecl-imagick-devel-3.7.0-1.amzn2.noarch.rpm`

Anda dapat menggunakan paket PEAR dan PECL untuk menginstal ekstensi umum. Untuk informasi lebih lanjut tentang PEAR, lihat situs web PEAR PHP Extension and Application Repository. Untuk informasi lebih lanjut tentang PECL, lihat situs web ekstensi PECL.

Contoh perintah berikut menginstal ekstensi Memcached.

```
$pecl install memcache
```

Atau Anda juga bisa menggunakan yang berikut ini:

```
$pear install pecl/memcache
```

Contoh perintah berikut menginstal ekstensi Redis.

```
$pecl install redis
```

Atau Anda juga bisa menggunakan yang berikut ini:

```
$pear install pecl/redis
```

Mengonfigurasi lingkungan PHP Anda

Anda dapat menggunakan konsol Elastic Beanstalk untuk mengaktifkan rotasi log ke Amazon S3, mengonfigurasi variabel yang dapat dibaca aplikasi Anda dari lingkungan, dan mengubah pengaturan PHP.

Untuk mengonfigurasi lingkungan PHP Anda di konsol Elastic Beanstalk

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.

4. Dalam kategori konfigurasi Pembaruan, pemantauan, dan pencatatan, pilih Edit.

Pengaturan PHP

- Server proksi – Server proksi yang akan digunakan pada instans lingkungan Anda. Secara default, nginx digunakan.
- Akar dokumen – Folder yang berisi halaman default situs Anda. Jika halaman selamat datang Anda tidak berada di akar paket sumber Anda, tentukan folder yang memuatnya itu relatif terhadap jalur akar. Misalnya, `/public` jika halaman selamat datang berada dalam folder bernama `public`.
- Batas memori – Jumlah maksimum memori yang dapat dialokasikan oleh skrip. Misalnya, `512M`.
- Kompresi output Zlib – Atur ke `On` untuk memampatkan respons.
- Ijinkan URL fopen – Atur ke `Off` untuk mencegah skrip mengunduh file dari lokasi jarak jauh.
- Kesalahan tampilan – Atur ke `On` untuk menampilkan pesan kesalahan internal untuk debugging.
- Waktu eksekusi maks – Waktu maksimum dalam detik yang boleh dijalankan skrip sebelum lingkungan mengakhirinya.

Opsi log

Bagian Opsi Log memiliki dua pengaturan:

- Profil instans– Menentukan profil instans yang memiliki izin untuk mengakses bucket Amazon S3 yang terkait dengan aplikasi Anda.
- Aktifkan rotasi file log ke Amazon S3 — Menentukan apakah file log untuk EC2 instans Amazon aplikasi Anda disalin ke bucket Amazon S3 yang terkait dengan aplikasi Anda.

File statis

Untuk meningkatkan kinerja, Anda dapat menggunakan bagian File statis untuk mengkonfigurasi server proxy untuk melayani file statis (misalnya, HTML atau gambar) dari satu set direktori di dalam aplikasi web Anda. Untuk setiap direktori, Anda mengatur jalur virtual ke pemetaan direktori. Saat server proksi menerima permintaan untuk file di jalur yang ditentukan, server langsung menyajikan file daripada merutekan permintaan ke aplikasi Anda.

Untuk detail tentang mengonfigurasi file statis menggunakan file konfigurasi atau konsol Elastic Beanstalk, lihat. the section called “File statis”

Properti lingkungan

Bagian Properti Lingkungan memungkinkan Anda menentukan pengaturan konfigurasi lingkungan pada EC2 instance Amazon yang menjalankan aplikasi Anda. Pengaturan ini diteruskan sebagai pasangan nilai kunci ke aplikasi.

Kode aplikasi Anda dapat mengakses properti lingkungan dengan menggunakan fungsi `$_SERVER` atau `get_cfg_var`.

```
$endpoint = $_SERVER['API_ENDPOINT'];
```

Lihat Variabel lingkungan dan pengaturan perangkat lunak lainnya untuk informasi selengkapnya.

Ruang nama untuk konfigurasi

Anda dapat menggunakan file konfigurasi untuk mengatur opsi konfigurasi dan melakukan tugas-tugas konfigurasi instans lain selama deployment. Opsi konfigurasi dapat berupa platform khusus atau berlaku untuk semua platform di layanan Elastic Beanstalk secara keseluruhan. Opsi konfigurasi diatur ke dalam ruang nama.

Ruang nama berikut mengonfigurasi layanan proxy dan opsi khusus PHP Anda:

- `aws:elasticbeanstalk:environment:proxy:staticfiles`— mengkonfigurasi proxy lingkungan untuk melayani file statis. Anda menentukan pemetaan jalur virtual ke direktori aplikasi.
- `aws:elasticbeanstalk:environment:proxy`— tentukan server proxy lingkungan.
- `aws:elasticbeanstalk:container:php:phpini`— konfigurasikan opsi khusus PHP. Namespace ini termasuk`composer_options`, yang tidak tersedia di konsol Elastic Beanstalk. Opsi ini menetapkan opsi kustom untuk digunakan saat menginstal dependensi menggunakan Komposer melalui perintah. `composer.phar install` Untuk informasi selengkapnya tentang perintah ini, termasuk opsi yang tersedia, lihat menginstal di situs getcomposer.org.

Contoh berikut file konfigurasi menentukan opsi file statis yang memetakan sebuah direktori bernama `staticimages` ke jalur `/images`, dan menampilkan pengaturan untuk masing-masing opsi yang tersedia di namespace `aws:elasticbeanstalk:container:php:phpini`:

Example .ebextensions/php-settings.config

```
option_settings:
```

```
  aws:elasticbeanstalk:environment:proxy:
    ProxyServer: apache
  aws:elasticbeanstalk:environment:proxy:staticfiles:
    /images: staticimages
  aws:elasticbeanstalk:container:php:phpini:
    document_root: /public
    memory_limit: 128M
    zlib.output_compression: "Off"
    allow_url_fopen: "On"
    display_errors: "Off"
    max_execution_time: 60
    composer_options: vendor/package
```

> **Note**
>
> Namespace `aws:elasticbeanstalk:environment:proxy:staticfiles` tidak ditegaskan pada cabang platform Amazon Linux AMI PHP (Amazon Linux 2 yang terdahulu).

Elastic Beanstalk memberikan banyak opsi konfigurasi untuk menyesuaikan lingkungan Anda. Selain file konfigurasi, Anda juga dapat mengatur opsi konfigurasi menggunakan konsol tersebut, konfigurasi tersimpan, EB CLI, atau AWS CLI. Untuk informasi selengkapnya, lihat Opsi konfigurasi.

Menginstal dependensi aplikasi PHP Elastic Beanstalk Anda

Topik ini menjelaskan cara mengkonfigurasi aplikasi Anda untuk menginstal paket PHP lain yang diperlukan. Aplikasi Anda mungkin memiliki dependensi pada paket PHP lainnya. Anda dapat mengonfigurasi aplikasi Anda untuk menginstal dependensi ini pada instans Amazon Elastic Compute Cloud (Amazon EC2) lingkungan. Atau, Anda dapat menyertakan dependensi aplikasi Anda dalam paket sumber dan men-deploy mereka dengan aplikasi. Bagian berikut membahas kedua cara tersebut.

Gunakan file Composer untuk menginstal dependensi pada instans

Gunakan `composer.json` file di root sumber proyek Anda untuk menggunakan komposer untuk menginstal paket yang diperlukan aplikasi Anda pada EC2 instance Amazon lingkungan Anda.

Example composer.json

```
{
```

```
    "require": {
        "monolog/monolog": "1.0.*"
    }
}
```

Saat file `composer.json` ada, Elastic Beanstalk menjalankan `composer.phar install` untuk menginstal dependensi. Anda dapat menambahkan opsi untuk ditambahkan ke perintah dengan mengatur opsi `composer_options` dalam namespace `aws:elasticbeanstalk:container:php:phpini`.

Sertakan dependensi dalam paket sumber

Jika aplikasi Anda memiliki dependensi dalam jumlah yang banyak, menginstalnya mungkin perlu waktu lama. Hal ini dapat meningkatkan operasi penskalaan dan deployment, karena dependensi diinstal pada setiap instans baru.

Untuk menghindari dampak negatif pada waktu deployment, gunakan Composer di lingkungan pengembangan Anda untuk menetapkan dependensi dan menginstalnya ke folder `vendor`.

Untuk menyertakan dependensi dalam paket sumber aplikasi Anda

1. Jalankan perintah berikut:

   ```
   % composer install
   ```

2. Sertakan folder `vendor` yang dihasilkan di dalam akar paket sumber aplikasi Anda.

Ketika Elastic Beanstalk menemukan folder `vendor` pada instans, file `composer.json` akan diabaikan (meskipun ada). Aplikasi Anda kemudian menggunakan dependensi dari folder `vendor`.

Memperbarui Komposer pada Elastic Beanstalk

Topik ini menjelaskan cara mengkonfigurasi Elastic Beanstalk agar Composer tetap up to date. Anda mungkin harus memperbarui Composer jika Anda melihat kesalahan ketika Anda mencoba untuk menginstal paket dengan file Composer, atau jika Anda tidak dapat menggunakan versi platform terbaru. Di antara pembaruan platform, Anda dapat memperbarui Composer di instance lingkungan Anda melalui penggunaan file konfigurasi di folder Anda `.ebextensions`.

Anda dapat memperbarui sendiri Komposer dengan konfigurasi berikut.

```
commands:
  01updateComposer:
    command: /usr/bin/composer.phar self-update 2.7.0
```

Pengaturan opsi berikut menetapkan variabel COMPOSER_HOME lingkungan, yang mengkonfigurasi lokasi cache Komposer.

```
option_settings:
  - namespace: aws:elasticbeanstalk:application:environment
    option_name: COMPOSER_HOME
    value: /home/webapp/composer-home
```

Anda dapat menggabungkan keduanya dalam file konfigurasi yang sama di `.ebextensions` folder Anda.

Example .ebextensions/composer.config

```
commands:
  01updateComposer:
    command: /usr/bin/composer.phar self-update 2.7.0

option_settings:
  - namespace: aws:elasticbeanstalk:application:environment
    option_name: COMPOSER_HOME
    value: /home/webapp/composer-home
```

> **Note**
>
> Karena pembaruan instalasi Komposer pada rilis platform 22 Februari 2024, AL2 023 dan rilis platform 28 Februari 2024, AL2 pembaruan mandiri Komposer mungkin gagal jika disetel saat pembaruan mandiri COMPOSER_HOME dijalankan.
> Perintah gabungan berikut akan gagal dijalankan: `export COMPOSER_HOME=/home/webapp/composer-home && /usr/bin/composer.phar self-update 2.7.0`
> Namun, contoh sebelumnya akan berhasil. Pada contoh sebelumnya, pengaturan opsi untuk tidak COMPOSER_HOME akan diteruskan ke `01updateComposer` eksekusi, dan itu tidak akan diatur ketika perintah pembaruan mandiri dijalankan.

> **Important**
>
> Jika Anda menghilangkan nomor versi dari perintah `composer.phar self-update`, Composer akan memperbarui ke versi terbaru yang tersedia setiap kali Anda men-deploy kode sumber, dan ketika instans baru disediakan oleh Auto Scaling. Hal ini dapat menyebabkan operasi penskalaan dan deployment gagal jika versi Composer yang dirilis tidak kompatibel dengan aplikasi Anda.

Untuk informasi lebih lanjut tentang Platform PHP Elastic Beanstalk, termasuk versi Composer, lihat versi platform PHP dalam dokumen Platform AWS Elastic Beanstalk .

Memperluas php.ini dalam konfigurasi Elastic Beanstalk Anda

Gunakan file konfigurasi dengan blok `files` untuk menambahkan file `.ini` ke `/etc/php.d/` pada instans di lingkungan Anda. File konfigurasi utama, `php.ini`, menarik pengaturan dari file pada folder ini dalam urutan abjad. Banyak ekstensi diaktifkan secara default oleh file dalam folder ini.

Example .ebextensions/mongo.config

```
files:
  "/etc/php.d/99mongo.ini":
    mode: "000755"
    owner: root
    group: root
    content: |
      extension=mongo.so
```

Contoh lanjutan untuk PHP di Elastic Beanstalk

Untuk memulai aplikasi PHP aktif AWS Elastic Beanstalk, Anda memerlukan bundel sumber aplikasi untuk diunggah sebagai versi aplikasi pertama Anda untuk menyebarkan ke lingkungan.

Kami merekomendasikan QuickStart untuk PHP untuk memulai dengan aplikasi PHP sederhana yang digunakan dengan EB CLI.

Contoh PHP tingkat lanjut

- Menambahkan instans Amazon RDS DB ke lingkungan PHP Elastic Beanstalk Anda
- Men-deploy aplikasi Laravel ke Elastic Beanstalk

- Men-deploy aplikasi CakePHP ke Elastic Beanstalk
- Men-deploy aplikasi Symfony ke Elastic Beanstalk
- Men-deploy aplikasi PHP ketersediaan tinggi dengan basis data Amazon RDS eksternal ke Elastic Beanstalk
- Menyebarkan WordPress situs web dengan ketersediaan tinggi dengan database Amazon RDS eksternal ke Elastic Beanstalk
- Men-deploy situs web Drupal ketersediaan tinggi dengan basis data Amazon RDS eksternal ke Elastic Beanstalk

Menambahkan instans Amazon RDS DB ke lingkungan PHP Elastic Beanstalk Anda

Topik ini memberikan instruksi untuk membuat Amazon RDS menggunakan konsol Elastic Beanstalk. Anda dapat menggunakan instans DB Amazon Relational Database Service (Amazon RDS) untuk menyimpan data yang dikumpulkan dan dimodifikasi oleh aplikasi Anda. Basis data dapat digabungkan ke lingkungan Anda dan dikelola oleh Elastic Beanstalk, atau dapat dibuat sebagai dipisahkan dan dikelola secara eksternal oleh layanan lain. Dalam instruksi ini database digabungkan dengan lingkungan Anda dan dikelola oleh Elastic Beanstalk. Untuk informasi lebih lanjut tentang mengintegrasikan Amazon RDS dengan Elastic Beanstalk, lihat. Menambahkan basis data ke lingkungan Elastic Beanstalk Anda

Bagian-bagian

- Menambahkan instans DB ke lingkungan Anda
- Mengunduh driver
- Menghubungkan ke database dengan PDO atau My SQLi
- Menghubungkan ke basis data dengan Symfony

Menambahkan instans DB ke lingkungan Anda

Untuk menambahkan instans DB ke lingkungan Anda

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.

3. Pada panel navigasi, pilih Konfigurasi.
4. Di kategori konfigurasi Basis data, pilih Edit.
5. Pilih mesin DB, dan masukkan nama pengguna dan kata sandi.
6. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Menambahkan instans DB memakan waktu sekitar 10 menit. Ketika pembaruan lingkungan selesai, nama host instans DB dan informasi koneksi lainnya tersedia untuk aplikasi Anda melalui properti lingkungan berikut:

Nama properti	Deskripsi	Nilai properti
`RDS_HOSTNAME`	Nama host instans DB.	Di tab Konektivitas & keamanan di konsol Amazon RDS: Titik akhir.
`RDS_PORT`	Port tempat instans DB menerima koneksi. Nilai default bervariasi di antara mesin DB.	Di tab Konektivitas & keamanan di konsol Amazon RDS: Port.
`RDS_DB_NAME`	Nama basis data, **ebdb**.	Di tab Konfigurasi di konsol Amazon RDS: Nama DB.
`RDS_USERNAME`	Nama pengguna yang Anda konfigurasi untuk basis data Anda.	Di tab Konfigurasi di konsol Amazon RDS: Nama pengguna utama.
`RDS_PASSWORD`	Kata sandi yang Anda konfigurasi untuk basis data Anda.	Tidak tersedia untuk referensi di konsol Amazon RDS.

Untuk informasi selengkapnya tentang mengonfigurasi instance database yang digabungkan dengan lingkungan Elastic Beanstalk, lihat. Menambahkan basis data ke lingkungan Elastic Beanstalk Anda

Mengunduh driver

Untuk menggunakan PHP Data objek (PDO) agar terhubung ke basis data, instal driver yang cocok dengan mesin basis data yang Anda pilih.

- MySQL – PDO_MYSQL
- PostgreSQL – PDO_PGSQL
- Oracle – PDO_OCI
- Server SQL – PDO_SQLSRV

Untuk informasi selengkapnya, lihat http://php.net/manual/en/pdo.installation.php.

Menghubungkan ke database dengan PDO atau My SQLi

Anda dapat menggunakan `$_SERVER[`VARIABLE`]` untuk membaca informasi koneksi dari lingkungan.

Untuk PDO, buat Data Source Name (DSN) dari host, port, dan nama. Berikan DSN ke konstruktor untuk PDO dengan nama pengguna dan kata sandi basis data.

Example Connect ke basis data RDS dengan PDO - MySQL

```
<?php
$dbhost = $_SERVER['RDS_HOSTNAME'];
$dbport = $_SERVER['RDS_PORT'];
$dbname = $_SERVER['RDS_DB_NAME'];
$charset = 'utf8' ;

$dsn = "mysql:host={$dbhost};port={$dbport};dbname={$dbname};charset={$charset}";
$username = $_SERVER['RDS_USERNAME'];
$password = $_SERVER['RDS_PASSWORD'];

$pdo = new PDO($dsn, $username, $password);
?>
```

Untuk driver lain, ganti `mysql` dengan nama driver Anda – `pgsql`, `oci`, atau `sqlsrv`.

Untuk MySQLi, berikan nama host, nama pengguna, kata sandi, nama database, dan port ke `mysqli` konstruktor.

Example Hubungkan ke basis data RDS dengan mysqli_connect()

```
$link = new mysqli($_SERVER['RDS_HOSTNAME'], $_SERVER['RDS_USERNAME'],
 $_SERVER['RDS_PASSWORD'], $_SERVER['RDS_DB_NAME'], $_SERVER['RDS_PORT']);
```

Menghubungkan ke basis data dengan Symfony

Untuk Symfony versi 3.2 dan yang lebih baru, Anda dapat menggunakan `%env(`*`PROPERTY_NAME`*`)%` untuk mengatur parameter basis data dalam file konfigurasi berdasarkan properti lingkungan yang ditetapkan oleh Elastic Beanstalk.

Example app/config/parameters.yml

```
parameters:
    database_driver:   pdo_mysql
    database_host:     '%env(RDS_HOSTNAME)%'
    database_port:     '%env(RDS_PORT)%'
    database_name:     '%env(RDS_DB_NAME)%'
    database_user:     '%env(RDS_USERNAME)%'
    database_password: '%env(RDS_PASSWORD)%'
```

Lihat Parameter Eksternal (Symfony 3.4) untuk informasi selengkapnya.

Untuk versi sebelumnya dari Symfony, variabel lingkungan hanya dapat diakses jika dimulai dengan `SYMFONY__`. Artinya, properti lingkungan yang ditetapkan Elastic Beanstalk tidak dapat diakses, dan Anda harus menentukan properti lingkungan Anda sendiri untuk memberikan informasi koneksi ke Symfony.

Untuk terhubung ke basis data dengan Symfony 2, buat properti lingkungan untuk setiap parameter. Kemudian, gunakan `%`*`property.name`*`%` untuk mengakses variabel transformasi Symfony dalam file konfigurasi. Sebagai contoh, properti lingkungan yang bernama `SYMFONY__DATABASE__USER` dapat diakses sebagai `database.user`.

```
    database_user:     "%database.user%"
```

Lihat Parameter Eksternal (Symfony 2.8) untuk informasi selengkapnya.

Men-deploy aplikasi Laravel ke Elastic Beanstalk

Laravel adalah kerangka kerja open source, model-view-controller (MVC) untuk PHP. Tutorial ini memandu Anda melalui proses menghasilkan aplikasi Laravel, menerapkannya ke AWS Elastic

Beanstalk lingkungan, dan mengonfigurasinya untuk terhubung ke instance database Amazon Relational Database Service (Amazon RDS).

Bagian-bagian

- Prasyarat
- Meluncurkan lingkungan Elastic Beanstalk
- Menginstal Laravel dan menghasilkan sebuah situs web
- Men-deploy aplikasi Anda
- Mengonfigurasi pengaturan Composer
- Menambahkan basis data ke lingkungan Anda
- Pembersihan
- Langkah selanjutnya

Prasyarat

Tutorial ini mengasumsikan Anda memiliki pengetahuan tentang operasi Elastic Beanstalk dasar dan konsol Elastic Beanstalk. Jika belum, ikuti petunjuk di Pelajari cara memulai dengan Elastic Beanstalk untuk meluncurkan lingkungan Elastic Beanstalk pertama Anda.

Untuk mengikuti prosedur di panduan ini, Anda memerlukan terminal atau shell baris perintah untuk menjalankan perintah. Perintah ditampilkan dalam daftar yang diawali dengan simbol prompt ($) dan nama direktori saat ini, bila perlu.

```
~/eb-project$ this is a command
this is output
```

Di Linux dan macOS, Anda dapat menggunakan shell dan manajer paket pilihan Anda. Pada Windows Anda dapat menginstal Windows Subsystem untuk Linux untuk mendapatkan versi Windows terintegrasi dari Ubuntu dan Bash.

Laravel 6 membutuhkan PHP 7.2 atau yang lebih baru. Laravel 6 juga membutuhkan ekstensi PHP yang tercantum dalam topik persyaratan server dalam dokumentasi Laravel resmi. 4. Ikuti instruksi untuk menginstal PHP and Composer.

Untuk informasi dukungan dan pemeliharaan Laravel, lihat topik kebijakan dukungan pada dokumentasi Laravel resmi.

Meluncurkan lingkungan Elastic Beanstalk

Gunakan konsol Elastic Beanstalk untuk membuat lingkungan Elastic Beanstalk. Pilih platform PHP lalu terima pengaturan default dan kode sampel.

Untuk meluncurkan lingkungan (konsol)

1. Buka konsol Elastic Beanstalk menggunakan tautan yang telah dikonfigurasi sebelumnya: console.aws.amazon. com/elasticbeanstalk/home#/newApplication?applicationName=Tutorial&EnvironmentType= LoadBalanced
2. Untuk Platform, pilih platform dan cabang platform yang sesuai dengan bahasa yang digunakan oleh aplikasi Anda.
3. Untuk Kode aplikasi, pilih Aplikasi sampel.
4. Pilih Tinjau dan Luncurkan.
5. Tinjau opsi yang tersedia. Pilih opsi tersedia yang ingin Anda gunakan, dan saat Anda siap, pilih Buat aplikasi.

Pembuatan lingkungan membutuhkan waktu sekitar 5 menit dan membuat sumber daya berikut:

- EC2 instance — Mesin virtual Amazon Elastic Compute Cloud (Amazon EC2) yang dikonfigurasi untuk menjalankan aplikasi web pada platform yang Anda pilih.

  Setiap platform menjalankan satu set perangkat lunak, file konfigurasi dan penulisan tertentu untuk mendukung versi bahasa, kerangka kerja, kontainer web tertentu, atau kombinasi dari semua ini. Sebagian besar platform menggunakan Apache atau NGINX sebagai proksi terbalik yang ada di depan aplikasi web Anda, meneruskan permintaan ke aplikasi web, menyajikan aset statis, dan menghasilkan log akses dan kesalahan.
- Grup keamanan instans — Grup EC2 keamanan Amazon yang dikonfigurasi untuk memungkinkan lalu lintas masuk pada port 80. Sumber daya ini memungkinkan lalu lintas HTTP dari penyeimbang beban mencapai EC2 instance yang menjalankan aplikasi web Anda. Secara default, lalu lintas tidak diizinkan di port lain.
- Penyeimbang beban – Penyeimbang beban Elastic Load Balancing yang dikonfigurasi untuk mendistribusikan permintaan ke instans yang menjalankan aplikasi Anda. Penyeimbang beban juga menghilangkan kebutuhan untuk mengekspos instans Anda langsung ke internet.

- Grup keamanan penyeimbang beban - Grup EC2 keamanan Amazon yang dikonfigurasi untuk memungkinkan lalu lintas masuk di port 80. Sumber daya ini memungkinkan lalu lintas HTTP dari internet mencapai penyeimbang beban. Secara default, lalu lintas tidak diizinkan di port lain.
- Grup Auto Scaling – Grup Auto Scaling yang dikonfigurasi untuk menggantikan instans jika diakhiri atau menjadi tidak tersedia.
- Bucket Amazon S3 – Lokasi penyimpanan untuk kode sumber, log, dan artifact lainnya yang dibuat saat Anda menggunakan Elastic Beanstalk.
- CloudWatch Alarm Amazon — Dua CloudWatch alarm yang memantau beban pada instans di lingkungan Anda dan yang dipicu jika beban terlalu tinggi atau terlalu rendah. Saat alarm terpicu, grup Auto Scaling Anda akan menaikkan atau menurunkan skala sebagai respons.
- AWS CloudFormation stack - Elastic AWS CloudFormation Beanstalk digunakan untuk meluncurkan sumber daya di lingkungan Anda dan menyebarkan perubahan konfigurasi. Sumber daya ditentukan di sebuah templat yang dapat Anda lihat di Konsol AWS CloudFormation.
- Nama domain — Nama domain yang merutekan ke aplikasi web Anda dalam formulir *`subdomain`*.*`region`*.elasticbeanstalk.com.

> **Keamanan domain**
>
> Untuk meningkatkan keamanan aplikasi Elastic Beanstalk Anda, domain elasticbeanstalk.com terdaftar di Daftar Akhiran Publik (PSL).
> Jika Anda perlu mengatur cookie sensitif di nama domain default untuk aplikasi Elastic Beanstalk Anda, kami sarankan Anda menggunakan cookie `__Host-` dengan awalan untuk meningkatkan keamanan. Praktik ini melindungi domain Anda dari upaya pemalsuan permintaan lintas situs (CSRF). Untuk informasi selengkapnya, lihat halaman Set-Cookie di Jaringan Pengembang Mozilla.

Semua sumber daya ini dikelola oleh Elastic Beanstalk. Ketika Anda mengakhiri lingkungan, Elastic Beanstalk mengakhiri semua sumber daya yang dimuatnya.

> **Note**
>
> Bucket Amazon S3 yang dibuat Elastic Beanstalk dibagi antar lingkungan dan tidak dihapus selama pengakhiran lingkungan. Untuk informasi selengkapnya, lihat Menggunakan Elastic Beanstalk dengan Amazon S3.

Menginstal Laravel dan menghasilkan sebuah situs web

Composer dapat menginstal Laravel dan membuat proyek kerja dengan satu perintah:

```
~$ composer create-project --prefer-dist laravel/laravel eb-laravel
```

Komposer yang diinstal Laravel dan dependensinya, dan menghasilkan proyek default.

Jika Anda mengalami masalah saat menginstal Laravel, buka topik instalasi di dokumentasi resmi: https://laravel.com/docs/ 6.x.

Men-deploy aplikasi Anda

Buat paket sumber berisi file yang dibuat oleh Composer. Perintah berikut membuat paket sumber yang bernama `laravel-default.zip`. Paket sumber tidak termasuk file dalam folder `vendor`, yang memerlukan banyak ruang dan tidak perlu men-deploy aplikasi Anda ke Elastic Beanstalk.

```
~/eb-laravel$ zip ../laravel-default.zip -r * .[^.]* -x "vendor/*"
```

Mengunggah paket sumber ke Elastic Beanstalk untuk men-deploy Laravel ke lingkungan Anda.

Untuk men-deploy paket sumber

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Di halaman gambaran umum lingkungan, pilih Unggah dan deploy.
4. Gunakan kotak dialog di layar untuk mengunggah paket sumber.
5. Pilih Deploy.
6. Ketika deployment selesai, Anda dapat memilih URL situs untuk membuka situs web Anda di tab baru.

> **Note**
>
> Untuk mengoptimalkan paket sumber selanjutnya, inisialisasi repositori Git dan gunakan perintah `git archive` untuk membuat paket sumber. Proyek Laravel default berisi file

`.gitignore` yang meminta Git untuk mengecualikan folder `vendor` dan file lain yang tidak diperlukan untuk deployment.

Mengonfigurasi pengaturan Composer

Ketika deployment selesai, klik URL untuk membuka aplikasi Laravel Anda di peramban:



Apa ini? Secara default, Elastic Beanstalk menyajikan akar proyek Anda di jalur akar situs web. Dalam hal ini, meskipun, halaman default (`index.php`) tersebut berada di satu tingkat di bawah dalam folder `public`. Anda dapat memverifikasinya dengan menambahkan `/public` ke URL. Misalnya, `http://laravel.us-east-2.elasticbeanstalk.com/public`.

Untuk menyajikan aplikasi Laravel di jalur akar, gunakan konsol Elastic Beanstalk untuk mengonfigurasikan akar dokumen untuk situs web.

Untuk mengonfigurasi akar dokumen situs web Anda

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Dalam kategori konfigurasi Pembaruan, pemantauan, dan pencatatan, pilih Edit.
5. Untuk Akar Dokumen, masukkan **`/public`**.
6. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.
7. Setelah pembaruan selesai, klik URL untuk membuka kembali situs Anda di peramban.



Sejauh ini, baik-baik saja. Selanjutnya Anda akan menambahkan basis data ke lingkungan Anda dan mengonfigurasi Laravel agar terhubung ke sana.

Menambahkan basis data ke lingkungan Anda

Meluncurkan instans DB RDS di lingkungan Elastic Beanstalk Anda. Anda dapat menggunakan database MySQL SQLServer,, atau PostgreSQL dengan Laravel pada Elastic Beanstalk. Untuk contoh ini, kita akan menggunakan MySQL.

Untuk menambahkan instans DB RDS ke lingkungan Elastic Beanstalk Anda

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Pada kategori konfigurasi Basis data, pilih Edit.
5. Untuk Mesin, pilih mysql.
6. Ketik nama pengguna utama dan kata sandi. Elastic Beanstalk akan memberikan nilai-nilai ini ke aplikasi Anda menggunakan properti lingkungan.
7. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Membuat instans basis data membutuhkan waktu sekitar 10 menit. Untuk informasi lebih lanjut tentang database yang digabungkan ke lingkungan Elastic Beanstalk, lihat. Menambahkan basis data ke lingkungan Elastic Beanstalk Anda

Sementara itu, Anda dapat memperbarui kode sumber untuk membaca informasi koneksi dari lingkungan. Elastic Beanstalk menyediakan detail koneksi menggunakan variabel lingkungan, seperti `RDS_HOSTNAME`, yang dapat Anda akses dari aplikasi Anda.

Konfigurasi basis data Laravel disimpan dalam sebuah file bernama `database.php` dalam folder `config` pada kode proyek Anda. Temukan entri `mysql` dan modifikasi variabel `host`, `database`, `username`, `and password` untuk membaca nilai-nilai yang sesuai dari Elastic Beanstalk:

Example ~/Eb- .php laravel/config/database

```
...
    'connections' => [

        'sqlite' => [
            'driver' => 'sqlite',
            'database' => env('DB_DATABASE', database_path('database.sqlite')),
            'prefix' => '',
        ],

        'mysql' => [
            'driver' => 'mysql',
            'host' => env('RDS_HOSTNAME', '127.0.0.1'),
            'port' => env('RDS_PORT', '3306'),
            'database' => env('RDS_DB_NAME', 'forge'),
            'username' => env('RDS_USERNAME', 'forge'),
            'password' => env('RDS_PASSWORD', ''),
            'unix_socket' => env('DB_SOCKET', ''),
            'charset' => 'utf8mb4',
            'collation' => 'utf8mb4_unicode_ci',
            'prefix' => '',
            'strict' => true,
            'engine' => null,
        ],
...
```

Untuk memverifikasi bahwa koneksi basis data dikonfigurasi dengan benar, tambahkan kode ke `index.php` untuk menghubungkan ke basis data dan menambahkan beberapa kode ke respons default:

Example ~/Eb- .php laravel/public/index

```
...
```

```
if(DB::connection()->getDatabaseName())
{
   echo "Connected to database ".DB::connection()->getDatabaseName();
}
$response->send();
...
```

Ketika instans DB telah menyelesaikan peluncuran, paketkan dan deploy aplikasi yang telah diperbarui ke lingkungan Anda.

Untuk memperbarui lingkungan Elastic Beanstalk

1. Buat paket sumber baru:

   ```
   ~/eb-laravel$ zip ../laravel-v2-rds.zip -r * .[^.]* -x "vendor/*"
   ```

2. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
3. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
4. Pilih Unggah dan Deploy.
5. Pilih Jelajahi, dan unggah `laravel-v2-rds.zip`.
6. Pilih Deploy.

Men-deploy versi baru dari aplikasi Anda membutuhkan waktu kurang dari satu menit. Ketika deployment selesai, refresh halaman web lagi untuk memverifikasi bahwa koneksi basis data berhasil:



Pembersihan

Setelah Anda selesai bekerja dengan kode demo, Anda dapat mengakhiri lingkungan Anda. Elastic Beanstalk menghapus semua sumber daya AWS terkait, seperti instans EC2 Amazon, instans database, penyeimbang beban, grup keamanan,dan alarm.

Menghapus sumber daya tidak menghapus aplikasi Elastic Beanstalk, sehingga Anda dapat membuat lingkungan baru untuk aplikasi Anda kapan saja.

Untuk mengakhiri lingkungan Elastic Beanstalk Anda dari konsol

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pilih Tindakan, lalu pilih Terminate environment.
4. Gunakan kotak dialog di layar untuk mengonfirmasi pengakhiran lingkungan.

Selain itu, Anda dapat mengakhiri sumber daya basis data yang Anda buat di luar lingkungan Elastic Beanstalk Anda. Ketika Anda mengakhiri instans DB Amazon RDS, Anda dapat mengambil snapshot dan memulihkan data ke instans lain kelak.

Untuk mengakhiri instans DB RDS Anda

1. Buka konsol Amazon RDS.
2. Pilih Basis data.
3. Pilih instans DB Anda.
4. Pilih Tindakan, dan lalu pilih Hapus.
5. Pilih apakah akan membuat snapshot, dan kemudian memilih Hapus.

Langkah selanjutnya

Untuk informasi selengkapnya tentang Laravel, kunjungi situs web resmi Laravel di laravel.com.

Ketika aplikasi terus dikembangkan, Anda mungkin akan menginginkan sebuah cara untuk mengelola lingkungan dan men-deploy aplikasi Anda tanpa membuat file .zip secara manual dan mengunggahnya ke konsol Elastic Beanstalk. Elastic Beanstalk Command Line Interface (EB CLI

easy-to-use) menyediakan perintah untuk membuat, mengkonfigurasi, dan menyebarkan aplikasi ke lingkungan Elastic Beanstalk dari baris perintah.

Dalam tutorial ini, Anda menggunakan konsol Elastic Beanstalk untuk mengonfigurasi opsi komposer. Untuk membuat bagian konfigurasi ini dari sumber aplikasi Anda, Anda dapat menggunakan file konfigurasi seperti berikut.

Example .ebextensions/composer.config

```
option_settings:
  aws:elasticbeanstalk:container:php:phpini:
    document_root: /public
```

Untuk informasi selengkapnya, lihat Penyesuaian lingkungan lanjutan dengan file konfigurasi (`.ebextensions`).

Menjalankan instans Amazon RDS DB dalam lingkungan Elastic Beanstalk Anda sangat bagus untuk pengembangan dan pengujian, tetapi mengikat siklus hidup basis data Anda ke lingkungan Anda. Lihat Menambahkan instans Amazon RDS DB ke lingkungan PHP Elastic Beanstalk Anda untuk petunjuk dalam menghubungkan ke basis data yang berjalan di luar lingkungan Anda.

Terakhir, jika Anda berencana menggunakan aplikasi dalam lingkungan produksi, Anda akan ingin mengonfigurasikan nama domain khusus untuk lingkungan Anda dan mengaktifkan HTTPS untuk koneksi yang aman.

Men-deploy aplikasi CakePHP ke Elastic Beanstalk

CakePHP merupakan sumber terbuka, kerangka kerja MVC untuk PHP. Tutorial ini memandu Anda dalam proses menghasilkan proyek CakePHP, men-deploy ke suatu lingkungan Elastic Beanstalk, dan mengonfigurasikannya agar terhubung ke instans basis data Amazon RDS.

Bagian-bagian

- Prasyarat
- Meluncurkan lingkungan Elastic Beanstalk
- Menginstal CakePHP dan menghasilkan situs web
- Deploy aplikasi Anda
- Menambahkan basis data ke lingkungan Anda

- Pembersihan
- Langkah selanjutnya

Prasyarat

Tutorial ini mengasumsikan Anda memiliki pengetahuan tentang operasi Elastic Beanstalk dasar dan konsol Elastic Beanstalk. Jika belum, ikuti petunjuk di Pelajari cara memulai dengan Elastic Beanstalk untuk meluncurkan lingkungan Elastic Beanstalk pertama Anda.

Untuk mengikuti prosedur di panduan ini, Anda memerlukan terminal atau shell baris perintah untuk menjalankan perintah. Perintah ditampilkan dalam daftar yang diawali dengan simbol prompt ($) dan nama direktori saat ini, bila perlu.

```
~/eb-project$ this is a command
this is output
```

Di Linux dan macOS, Anda dapat menggunakan shell dan manajer paket pilihan Anda. Pada Windows Anda dapat menginstal Windows Subsystem untuk Linux untuk mendapatkan versi Windows terintegrasi dari Ubuntu dan Bash.

CakePHP 4 membutuhkan PHP 7.4 atau yang lebih baru. CakePHP 4 juga membutuhkan ekstensi PHP yang tercantum dalam dokumentasi instalasi CakePHP resmi. Anda harus menginstal PHP dan Composer.

Meluncurkan lingkungan Elastic Beanstalk

Gunakan konsol Elastic Beanstalk untuk membuat lingkungan Elastic Beanstalk. Pilih platform PHP lalu terima pengaturan default dan kode sampel.

Untuk meluncurkan lingkungan (konsol)

1. Buka konsol Elastic Beanstalk menggunakan tautan yang telah dikonfigurasi sebelumnya: console.aws.amazon. com/elasticbeanstalk/home#/newApplication?applicationName=Tutorial&EnvironmentType= LoadBalanced
2. Untuk Platform, pilih platform dan cabang platform yang sesuai dengan bahasa yang digunakan oleh aplikasi Anda.
3. Untuk Kode aplikasi, pilih Aplikasi sampel.
4. Pilih Tinjau dan Luncurkan.

5. Tinjau opsi yang tersedia. Pilih opsi tersedia yang ingin Anda gunakan, dan saat Anda siap, pilih Buat aplikasi.

Pembuatan lingkungan membutuhkan waktu sekitar 5 menit dan membuat sumber daya berikut:

- EC2 instance — Mesin virtual Amazon Elastic Compute Cloud (Amazon EC2) yang dikonfigurasi untuk menjalankan aplikasi web pada platform yang Anda pilih.

  Setiap platform menjalankan satu set perangkat lunak, file konfigurasi dan penulisan tertentu untuk mendukung versi bahasa, kerangka kerja, kontainer web tertentu, atau kombinasi dari semua ini. Sebagian besar platform menggunakan Apache atau NGINX sebagai proksi terbalik yang ada di depan aplikasi web Anda, meneruskan permintaan ke aplikasi web, menyajikan aset statis, dan menghasilkan log akses dan kesalahan.
- Grup keamanan instans — Grup EC2 keamanan Amazon yang dikonfigurasi untuk memungkinkan lalu lintas masuk pada port 80. Sumber daya ini memungkinkan lalu lintas HTTP dari penyeimbang beban mencapai EC2 instance yang menjalankan aplikasi web Anda. Secara default, lalu lintas tidak diizinkan di port lain.
- Penyeimbang beban – Penyeimbang beban Elastic Load Balancing yang dikonfigurasi untuk mendistribusikan permintaan ke instans yang menjalankan aplikasi Anda. Penyeimbang beban juga menghilangkan kebutuhan untuk mengekspos instans Anda langsung ke internet.
- Grup keamanan penyeimbang beban - Grup EC2 keamanan Amazon yang dikonfigurasi untuk memungkinkan lalu lintas masuk di port 80. Sumber daya ini memungkinkan lalu lintas HTTP dari internet mencapai penyeimbang beban. Secara default, lalu lintas tidak diizinkan di port lain.
- Grup Auto Scaling – Grup Auto Scaling yang dikonfigurasi untuk menggantikan instans jika diakhiri atau menjadi tidak tersedia.
- Bucket Amazon S3 – Lokasi penyimpanan untuk kode sumber, log, dan artifact lainnya yang dibuat saat Anda menggunakan Elastic Beanstalk.
- CloudWatch Alarm Amazon — Dua CloudWatch alarm yang memantau beban pada instans di lingkungan Anda dan yang dipicu jika beban terlalu tinggi atau terlalu rendah. Saat alarm terpicu, grup Auto Scaling Anda akan menaikkan atau menurunkan skala sebagai respons.
- AWS CloudFormation stack - Elastic AWS CloudFormation Beanstalk digunakan untuk meluncurkan sumber daya di lingkungan Anda dan menyebarkan perubahan konfigurasi. Sumber daya ditentukan di sebuah templat yang dapat Anda lihat di Konsol AWS CloudFormation.
- Nama domain — Nama domain yang merutekan ke aplikasi web Anda dalam formulir *subdomain*.*region*.elasticbeanstalk.com.

> **Keamanan domain**
>
> Untuk meningkatkan keamanan aplikasi Elastic Beanstalk Anda, domain elasticbeanstalk.com terdaftar di Daftar Akhiran Publik (PSL).
> Jika Anda perlu mengatur cookie sensitif di nama domain default untuk aplikasi Elastic Beanstalk Anda, kami sarankan Anda menggunakan cookie `__Host-` dengan awalan untuk meningkatkan keamanan. Praktik ini melindungi domain Anda dari upaya pemalsuan permintaan lintas situs (CSRF). Untuk informasi selengkapnya, lihat halaman Set-Cookie di Jaringan Pengembang Mozilla.

Semua sumber daya ini dikelola oleh Elastic Beanstalk. Ketika Anda mengakhiri lingkungan, Elastic Beanstalk mengakhiri semua sumber daya yang dimuatnya.

> **Note**
>
> Bucket Amazon S3 yang dibuat Elastic Beanstalk dibagi antar lingkungan dan tidak dihapus selama pengakhiran lingkungan. Untuk informasi selengkapnya, lihat Menggunakan Elastic Beanstalk dengan Amazon S3.

Menginstal CakePHP dan menghasilkan situs web

Composer dapat menginstal CakePHP dan membuat proyek kerja dengan satu perintah:

```
~$ composer create-project --prefer-dist cakephp/app eb-cake
```

Composer menginstal CakePHP dan sekitar 20 dependensi, dan menghasilkan proyek default.

Jika Anda mengalami masalah apa pun saat menginstal CakePHP, kunjungi topik instalasi dalam dokumentasi resmi: http://book.cakephp.org/4.0/en/installation.html

Deploy aplikasi Anda

Buat paket sumber berisi file yang dibuat oleh Composer. Perintah berikut membuat paket sumber yang bernama `cake-default.zip`. Paket sumber tidak termasuk file dalam folder `vendor`, yang memerlukan banyak ruang dan tidak perlu men-deploy aplikasi Anda ke Elastic Beanstalk.

```
eb-cake zip ../cake-default.zip -r * .[^.]* -x "vendor/*"
```

Unggah paket sumber ke Elastic Beanstalk untuk men-deploy CakePHP ke lingkungan Anda.

Untuk men-deploy paket sumber

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Di halaman gambaran umum lingkungan, pilih Unggah dan deploy.
4. Gunakan kotak dialog di layar untuk mengunggah paket sumber.
5. Pilih Deploy.
6. Ketika deployment selesai, Anda dapat memilih URL situs untuk membuka situs web Anda di tab baru.

> **Note**
>
> Untuk mengoptimalkan paket sumber selanjutnya, inisialisasi repositori Git dan gunakan perintah `git archive` untuk membuat paket sumber. Proyek Symfony default berisi file `.gitignore` yang meminta Git untuk mengecualikan folder `vendor` dan file lain yang tidak diperlukan untuk deployment.

Ketika proses selesai, klik URL untuk membuka aplikasi CakePHP Anda di peramban.

Sejauh ini, baik-baik saja. Selanjutnya Anda akan menambahkan basis data ke lingkungan Anda dan mengonfigurasi CakePHP agar terhubung ke sana.

Menambahkan basis data ke lingkungan Anda

Luncurkan instans basis data Amazon RDS dalam lingkungan Elastic Beanstalk Anda. Anda dapat menggunakan database MySQL, SQLServer, atau PostgreSQL dengan CakePHP pada Elastic Beanstalk. Untuk contoh ini, kita akan menggunakan PostgreSQL.

Untuk menambahkan instans DB Amazon RDS ke lingkungan Elastic Beanstalk Anda

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS

2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Dalam Basis Data, Pilih Edit.
5. Untuk Mesin DB, pilih postgres.
6. Ketik nama pengguna utama dan kata sandi. Elastic Beanstalk akan memberikan nilai-nilai ini ke aplikasi Anda menggunakan properti lingkungan.
7. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Membuat instans basis data membutuhkan waktu sekitar 10 menit. Sementara itu, Anda dapat memperbarui kode sumber untuk membaca informasi koneksi dari lingkungan. Elastic Beanstalk menyediakan detail koneksi menggunakan variabel lingkungan seperti `RDS_HOSTNAME` yang dapat Anda akses dari aplikasi Anda.

Konfigurasi basis data CakePHP berada dalam sebuah file bernama `app.php` dalam folder `config` pada kode proyek Anda. Buka file ini dan tambahkan beberapa kode yang membaca variabel lingkungan dari `$_SERVER` dan menetapkannya ke variabel lokal. Masukkan baris yang disorot dalam contoh di bawah ini setelah baris pertama (`<?php`):

Example ~/Eb- .php cake/config/app

```
<?php
if (!defined('RDS_HOSTNAME')) {
  define('RDS_HOSTNAME', $_SERVER['RDS_HOSTNAME']);
  define('RDS_USERNAME', $_SERVER['RDS_USERNAME']);
  define('RDS_PASSWORD', $_SERVER['RDS_PASSWORD']);
  define('RDS_DB_NAME', $_SERVER['RDS_DB_NAME']);
}
return [
...
```

Koneksi basis data dikonfigurasi lebih lanjut di `app.php`. Temukan bagian berikut ini dan ubah konfigurasi sumber data default dengan nama driver yang cocok dengan mesin basis data Anda (`Mysql`, `Sqlserver`, atau `Postgres`), dan atur variabel `host`, `username`, `password` dan `database` untuk membaca nilai-nilai yang sesuai dari Elastic Beanstalk:

Example ~/Eb- .php cake/config/app

```
...
```

```
     /**
     * Connection information used by the ORM to connect
     * to your application's datastores.
     * Drivers include Mysql Postgres Sqlite Sqlserver
     * See vendor\cakephp\cakephp\src\Database\Driver for complete list
     */
    'Datasources' => [
        'default' => [
            'className' => 'Cake\Database\Connection',
            'driver' => 'Cake\Database\Driver\Postgres',
            'persistent' => false,
            'host' => RDS_HOSTNAME,
            /*
             * CakePHP will use the default DB port based on the driver selected
             * MySQL on MAMP uses port 8889, MAMP users will want to uncomment
             * the following line and set the port accordingly
             */
            //'port' => 'non_standard_port_number',
            'username' => RDS_USERNAME,
            'password' => RDS_PASSWORD,
            'database' => RDS_DB_NAME,
            /*
             * You do not need to set this flag to use full utf-8 encoding (internal
 default since CakePHP 3.6).
             */
            //'encoding' => 'utf8mb4',
            'timezone' => 'UTC',
            'flags' => [],
            'cacheMetadata' => true,
            'log' => false,
...
```

Ketika instans DB telah menyelesaikan peluncuran, paketkan dan deploy aplikasi yang telah diperbarui ke lingkungan Anda:

Untuk memperbarui lingkungan Elastic Beanstalk

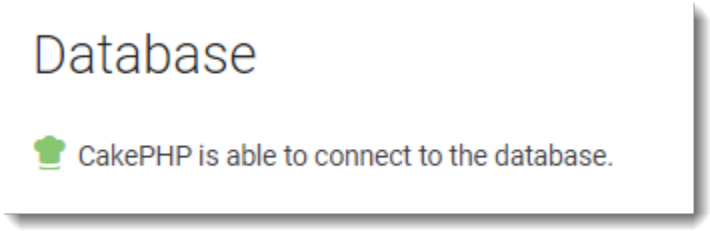
1. Buat paket sumber baru:

   ```
   ~/eb-cake$ zip ../cake-v2-rds.zip -r * .[^.]* -x "vendor/*"
   ```

2. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS

3. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
4. Pilih Unggah dan Deploy.
5. Pilih Jelajahi dan unggah `cake-v2-rds.zip`.
6. Pilih Deploy.

Men-deploy versi baru dari aplikasi Anda membutuhkan waktu kurang dari satu menit. Ketika deployment selesai, refresh halaman web lagi untuk memverifikasi bahwa koneksi basis data berhasil:



Pembersihan

Setelah Anda selesai bekerja dengan kode demo, Anda dapat mengakhiri lingkungan Anda. Elastic Beanstalk menghapus semua sumber daya AWS terkait, seperti instans EC2 Amazon, instans database, penyeimbang beban, grup keamanan,dan alarm.

Menghapus sumber daya tidak menghapus aplikasi Elastic Beanstalk, sehingga Anda dapat membuat lingkungan baru untuk aplikasi Anda kapan saja.

Untuk mengakhiri lingkungan Elastic Beanstalk Anda dari konsol

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pilih Tindakan, lalu pilih Terminate environment.
4. Gunakan kotak dialog di layar untuk mengonfirmasi pengakhiran lingkungan.

Selain itu, Anda dapat mengakhiri sumber daya basis data yang Anda buat di luar lingkungan Elastic Beanstalk Anda. Ketika Anda mengakhiri instans DB Amazon RDS, Anda dapat mengambil snapshot dan memulihkan data ke instans lain kelak.

Untuk mengakhiri instans DB RDS Anda

1. Buka konsol Amazon RDS.
2. Pilih Basis data.
3. Pilih instans DB Anda.
4. Pilih Tindakan, dan lalu pilih Hapus.
5. Pilih apakah akan membuat snapshot, dan kemudian memilih Hapus.

Langkah selanjutnya

Untuk informasi lebih lanjut tentang CakePHP, baca bukunya di book.cakephp.org.

Ketika aplikasi terus dikembangkan, Anda mungkin akan menginginkan sebuah cara untuk mengelola lingkungan dan men-deploy aplikasi Anda tanpa membuat file .zip secara manual dan mengunggahnya ke konsol Elastic Beanstalk. Elastic Beanstalk Command Line Interface (EB CLI easy-to-use) menyediakan perintah untuk membuat, mengkonfigurasi, dan menyebarkan aplikasi ke lingkungan Elastic Beanstalk dari baris perintah.

Menjalankan instans Amazon RDS DB dalam lingkungan Elastic Beanstalk Anda sangat bagus untuk pengembangan dan pengujian, tetapi mengikat siklus hidup basis data Anda ke lingkungan Anda. Lihat Menambahkan instans Amazon RDS DB ke lingkungan PHP Elastic Beanstalk Anda untuk petunjuk dalam menghubungkan ke basis data yang berjalan di luar lingkungan Anda.

Terakhir, jika Anda berencana menggunakan aplikasi dalam lingkungan produksi, Anda akan ingin mengonfigurasikan nama domain khusus untuk lingkungan Anda dan mengaktifkan HTTPS untuk koneksi yang aman.

Men-deploy aplikasi Symfony ke Elastic Beanstalk

Symfony adalah kerangka kerja sumber terbuka dalam mengembangkan aplikasi web PHP dinamis. Tutorial ini memandu Anda melalui proses menghasilkan aplikasi Symfony dan menyebarkannya ke lingkungan. AWS Elastic Beanstalk

Bagian-bagian

- Prasyarat
- Meluncurkan lingkungan Elastic Beanstalk

- Menginstal Symfony dan menghasilkan sebuah situs web
- Deploy aplikasi Anda
- Mengonfigurasi pengaturan Composer
- Pembersihan
- Langkah selanjutnya

Prasyarat

Tutorial ini mengasumsikan Anda memiliki pengetahuan tentang operasi Elastic Beanstalk dasar dan konsol Elastic Beanstalk. Jika belum, ikuti petunjuk di Pelajari cara memulai dengan Elastic Beanstalk untuk meluncurkan lingkungan Elastic Beanstalk pertama Anda.

Untuk mengikuti prosedur di panduan ini, Anda memerlukan terminal atau shell baris perintah untuk menjalankan perintah. Perintah ditampilkan dalam daftar yang diawali dengan simbol prompt ($) dan nama direktori saat ini, bila perlu.

```
~/eb-project$ this is a command
this is output
```

Di Linux dan macOS, Anda dapat menggunakan shell dan manajer paket pilihan Anda. Pada Windows Anda dapat menginstal Windows Subsystem untuk Linux untuk mendapatkan versi Windows terintegrasi dari Ubuntu dan Bash.

Symfony 4.4.9 membutuhkan PHP 7.1.3 atau yang lebih baru. Symfony 4.4.9 juga membutuhkan ekstensi PHP yang tercantum dalam topik persyaratan teknis dalam dokumentasi instalasi Symfony resmi. Dalam tutorial ini, kita menggunakan PHP 7.2 dan versi platform Elastic Beanstalk yang sesuai. Sebelum Anda melanjutkan, Anda harus menginstal PHP dan Composer.

Untuk informasi dukungan dan pemeliharaan Symfony, lihat topik rilis symfony dalam situs web Symfony. Untuk informasi selengkapnya tentang pembaruan yang berkaitan dengan dukungan versi PHP untuk Symfony 4.4.9, lihat topik catatan rilis Symfony 4.4.9 dalam situs web Symfony.

Meluncurkan lingkungan Elastic Beanstalk

Gunakan konsol Elastic Beanstalk untuk membuat lingkungan Elastic Beanstalk. Pilih platform PHP lalu terima pengaturan default dan kode sampel.

Untuk meluncurkan lingkungan (konsol)

1. Buka konsol Elastic Beanstalk menggunakan tautan yang telah dikonfigurasi sebelumnya: console.aws.amazon. com/elasticbeanstalk/home#/newApplication?applicationName=Tutorial&EnvironmentType= LoadBalanced
2. Untuk Platform, pilih platform dan cabang platform yang sesuai dengan bahasa yang digunakan oleh aplikasi Anda.
3. Untuk Kode aplikasi, pilih Aplikasi sampel.
4. Pilih Tinjau dan Luncurkan.
5. Tinjau opsi yang tersedia. Pilih opsi tersedia yang ingin Anda gunakan, dan saat Anda siap, pilih Buat aplikasi.

Pembuatan lingkungan membutuhkan waktu sekitar 5 menit dan membuat sumber daya berikut:

- EC2 instance — Mesin virtual Amazon Elastic Compute Cloud (Amazon EC2) yang dikonfigurasi untuk menjalankan aplikasi web pada platform yang Anda pilih.

  Setiap platform menjalankan satu set perangkat lunak, file konfigurasi dan penulisan tertentu untuk mendukung versi bahasa, kerangka kerja, kontainer web tertentu, atau kombinasi dari semua ini. Sebagian besar platform menggunakan Apache atau NGINX sebagai proksi terbalik yang ada di depan aplikasi web Anda, meneruskan permintaan ke aplikasi web, menyajikan aset statis, dan menghasilkan log akses dan kesalahan.
- Grup keamanan instans — Grup EC2 keamanan Amazon yang dikonfigurasi untuk memungkinkan lalu lintas masuk pada port 80. Sumber daya ini memungkinkan lalu lintas HTTP dari penyeimbang beban mencapai EC2 instance yang menjalankan aplikasi web Anda. Secara default, lalu lintas tidak diizinkan di port lain.
- Penyeimbang beban – Penyeimbang beban Elastic Load Balancing yang dikonfigurasi untuk mendistribusikan permintaan ke instans yang menjalankan aplikasi Anda. Penyeimbang beban juga menghilangkan kebutuhan untuk mengekspos instans Anda langsung ke internet.
- Grup keamanan penyeimbang beban - Grup EC2 keamanan Amazon yang dikonfigurasi untuk memungkinkan lalu lintas masuk di port 80. Sumber daya ini memungkinkan lalu lintas HTTP dari internet mencapai penyeimbang beban. Secara default, lalu lintas tidak diizinkan di port lain.
- Grup Auto Scaling – Grup Auto Scaling yang dikonfigurasi untuk menggantikan instans jika diakhiri atau menjadi tidak tersedia.

- Bucket Amazon S3 – Lokasi penyimpanan untuk kode sumber, log, dan artifact lainnya yang dibuat saat Anda menggunakan Elastic Beanstalk.
- CloudWatch Alarm Amazon — Dua CloudWatch alarm yang memantau beban pada instans di lingkungan Anda dan yang dipicu jika beban terlalu tinggi atau terlalu rendah. Saat alarm terpicu, grup Auto Scaling Anda akan menaikkan atau menurunkan skala sebagai respons.
- AWS CloudFormation stack - Elastic AWS CloudFormation Beanstalk digunakan untuk meluncurkan sumber daya di lingkungan Anda dan menyebarkan perubahan konfigurasi. Sumber daya ditentukan di sebuah templat yang dapat Anda lihat di Konsol AWS CloudFormation.
- Nama domain — Nama domain yang merutekan ke aplikasi web Anda dalam formulir *subdomain*.*region*.elasticbeanstalk.com.

> **Keamanan domain**
>
> Untuk meningkatkan keamanan aplikasi Elastic Beanstalk Anda, domain elasticbeanstalk.com terdaftar di Daftar Akhiran Publik (PSL).
> Jika Anda perlu mengatur cookie sensitif di nama domain default untuk aplikasi Elastic Beanstalk Anda, kami sarankan Anda menggunakan cookie `__Host-` dengan awalan untuk meningkatkan keamanan. Praktik ini melindungi domain Anda dari upaya pemalsuan permintaan lintas situs (CSRF). Untuk informasi selengkapnya, lihat halaman Set-Cookie di Jaringan Pengembang Mozilla.

Semua sumber daya ini dikelola oleh Elastic Beanstalk. Ketika Anda mengakhiri lingkungan, Elastic Beanstalk mengakhiri semua sumber daya yang dimuatnya.

> **Note**
>
> Bucket Amazon S3 yang dibuat Elastic Beanstalk dibagi antar lingkungan dan tidak dihapus selama pengakhiran lingkungan. Untuk informasi selengkapnya, lihat Menggunakan Elastic Beanstalk dengan Amazon S3.

Menginstal Symfony dan menghasilkan sebuah situs web

Composer dapat menginstal Symfony dan membuat proyek kerja dengan satu perintah:

```
~$ composer create-project symfony/website-skeleton eb-symfony
```

Composer menginstal Symfony dan dependensinya, dan menghasilkan proyek default.

Jika Anda mengalami masalah saat menginstal Symfony, buka topik instalasi dalam dokumentasi Symfony resmi.

Deploy aplikasi Anda

Pergi ke direktori proyek.

```
~$ cd eb-symfony
```

Buat paket sumber berisi file yang dibuat oleh Composer. Perintah berikut membuat paket sumber yang bernama `symfony-default.zip`. Paket sumber tidak termasuk file dalam folder `vendor`, yang memerlukan banyak ruang dan tidak perlu men-deploy aplikasi Anda ke Elastic Beanstalk.

```
eb-symfony$ zip ../symfony-default.zip -r * .[^.]* -x "vendor/*"
```

Unggah paket sumber ke Elastic Beanstalk untuk men-deploy Symfony ke lingkungan Anda.

Untuk men-deploy paket sumber

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Di halaman gambaran umum lingkungan, pilih Unggah dan deploy.
4. Gunakan kotak dialog di layar untuk mengunggah paket sumber.
5. Pilih Deploy.
6. Ketika deployment selesai, Anda dapat memilih URL situs untuk membuka situs web Anda di tab baru.

> **Note**
>
> Untuk mengoptimalkan paket sumber selanjutnya, inisialisasi repositori Git dan gunakan perintah `git archive` untuk membuat paket sumber. Proyek Symfony default berisi file `.gitignore` yang meminta Git untuk mengecualikan folder `vendor` dan file lain yang tidak diperlukan untuk deployment.

Mengonfigurasi pengaturan Composer

Setelah deployment selesai, klik URL untuk membuka aplikasi Symfony Anda di peramban.

Apa ini? Secara default, Elastic Beanstalk menyajikan akar proyek Anda di jalur akar situs web. Dalam hal ini, meskipun, halaman default (`app.php`) tersebut berada di satu tingkat di bawah dalam folder `web`. Anda dapat memverifikasinya dengan menambahkan `/public` ke URL. Misalnya, `http://symfony.us-east-2.elasticbeanstalk.com/public`.

Untuk menyajikan aplikasi Symfony di jalur akar, gunakan konsol Elastic Beanstalk untuk mengonfigurasikan akar dokumen bagi situs web.

Untuk mengonfigurasi akar dokumen situs web Anda

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Dalam kategori konfigurasi Pembaruan, pemantauan, dan pencatatan, pilih Edit.
5. Untuk Akar dokumen, masukkan **`/public`**.
6. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.
7. Setelah pembaruan selesai, klik URL untuk membuka kembali situs Anda di peramban.

Pembersihan

Setelah Anda selesai bekerja dengan kode demo, Anda dapat mengakhiri lingkungan Anda. Elastic Beanstalk menghapus semua sumber daya AWS terkait, seperti instans EC2 Amazon, instans database, penyeimbang beban, grup keamanan,dan alarm.

Menghapus sumber daya tidak menghapus aplikasi Elastic Beanstalk, sehingga Anda dapat membuat lingkungan baru untuk aplikasi Anda kapan saja.

Untuk mengakhiri lingkungan Elastic Beanstalk Anda dari konsol

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pilih Tindakan, lalu pilih Terminate environment.

4. Gunakan kotak dialog di layar untuk mengonfirmasi pengakhiran lingkungan.

Langkah selanjutnya

Untuk informasi lebih lanjut tentang Symfony, lihat Apa itu Symfony? di symfony.com.

Ketika aplikasi terus dikembangkan, Anda mungkin akan menginginkan sebuah cara untuk mengelola lingkungan dan men-deploy aplikasi Anda tanpa membuat file .zip secara manual dan mengunggahnya ke konsol Elastic Beanstalk. Elastic Beanstalk Command Line Interface (EB CLI easy-to-use) menyediakan perintah untuk membuat, mengkonfigurasi, dan menyebarkan aplikasi ke lingkungan Elastic Beanstalk dari baris perintah.

Dalam tutorial ini, Anda menggunakan konsol Elastic Beanstalk untuk mengonfigurasi opsi komposer. Untuk membuat bagian konfigurasi ini dari sumber aplikasi Anda, Anda dapat menggunakan file konfigurasi seperti berikut.

Example .ebextensions/composer.config

```
option_settings:
  aws:elasticbeanstalk:container:php:phpini:
    document_root: /public
```

Untuk informasi selengkapnya, lihat Penyesuaian lingkungan lanjutan dengan file konfigurasi (`.ebextensions`).

Symfony menggunakan file konfigurasinya sendiri dalam mengonfigurasi koneksi basis data. Untuk petunjuk tentang melakukan koneksi ke basis data dengan Symfony, lihat Menghubungkan ke basis data dengan Symfony.

Terakhir, jika Anda berencana menggunakan aplikasi dalam lingkungan produksi, Anda akan ingin mengonfigurasikan nama domain khusus untuk lingkungan Anda dan mengaktifkan HTTPS untuk koneksi yang aman.

Men-deploy aplikasi PHP ketersediaan tinggi dengan basis data Amazon RDS eksternal ke Elastic Beanstalk

Tutorial ini memandu Anda melalui proses peluncuran instans RDS DB eksternal ke AWS Elastic Beanstalk, dan mengkonfigurasi lingkungan ketersediaan tinggi menjalankan aplikasi PHP untuk terhubung ke sana. Menjalankan instans DB eksternal Elastic Beanstalk akan memisahkan basis

data dari siklus hidup lingkungan Anda. Hal ini memungkinkan Anda terhubung ke basis data yang sama dari beberapa lingkungan, menukarkan satu basis data dengan yang lain, atau melakukan deployment biru/hijau tanpa memengaruhi basis data Anda.

Tutorial menggunakan aplikasi PHP sampel yang menggunakan basis data MySQL untuk menyimpan data teks yang disediakan pengguna. Aplikasi sampel menggunakan file konfigurasi untuk mengonfigurasikan pengaturan PHP dan untuk membuat sebuah tabel dalam basis data untuk aplikasi yang akan digunakan. Aplikasi sampel juga menampilkan cara menggunakan file Composer untuk menginstal paket selama deployment.

Bagian-bagian

- Prasyarat
- Meluncurkan instans DB pada Amazon RDS
- Buat lingkungan Elastic Beanstalk
- Mengonfigurasi grup keamanan, properti lingkungan, dan penskalaan
- Men-deploy aplikasi sampel
- Pembersihan
- Langkah selanjutnya

Prasyarat

Sebelum memulai, unduh contoh bundel sumber aplikasi dari GitHub: eb-demo-php-simple-app-1.3.zip

Prosedur dalam tutorial ini untuk tugas Amazon Relational Database Service (Amazon RDS) yang menganggap Anda sedang meluncurkan sumber daya di Amazon Virtual Private Cloud (Amazon VPC) default. Semua akun baru menyertakan VPC default di setiap wilayah. Jika Anda tidak memiliki VPC default, prosedurnya akan bervariasi. Lihat petunjuk Menggunakan Elastic Beanstalk dengan Amazon RDS untuk platform EC2 -Classic dan VPC kustom.

Meluncurkan instans DB pada Amazon RDS

Untuk menggunakan basis data eksternal dengan aplikasi yang berjalan di Elastic Beanstalk, hal pertama yang dilakukan adalah meluncurkan instans DB dengan Amazon RDS. Ketika Anda meluncurkan instans dengan Amazon RDS, instans sepenuhnya tidak akan bergantung kepada Elastic Beanstalk dan lingkungan Elastic Beanstalk Anda, dan tidak akan diakhiri atau dipantau oleh Elastic Beanstalk.

Gunakan konsol Amazon RDS untuk meluncurkan instans DB MySQL Multi-AZ. Memilih deployment Multi-AZ dapat memastikan apakah basis data Anda akan gagal dan terus tersedia jika instans DB sumber keluar dari layanan.

Untuk meluncurkan instans DB RDS pada VPC default

1. Buka konsol RDS.
2. Di panel navigasi, pilih Basis Data.
3. Pilih Buat basis data.
4. Pilih Pembuatan Standar.

> **Important**
>
> Jangan pilih Pembuatan Mudah. Jika Anda memilihnya, Anda tidak dapat mengonfigurasi pengaturan yang diperlukan untuk meluncurkan RDS DB ini.

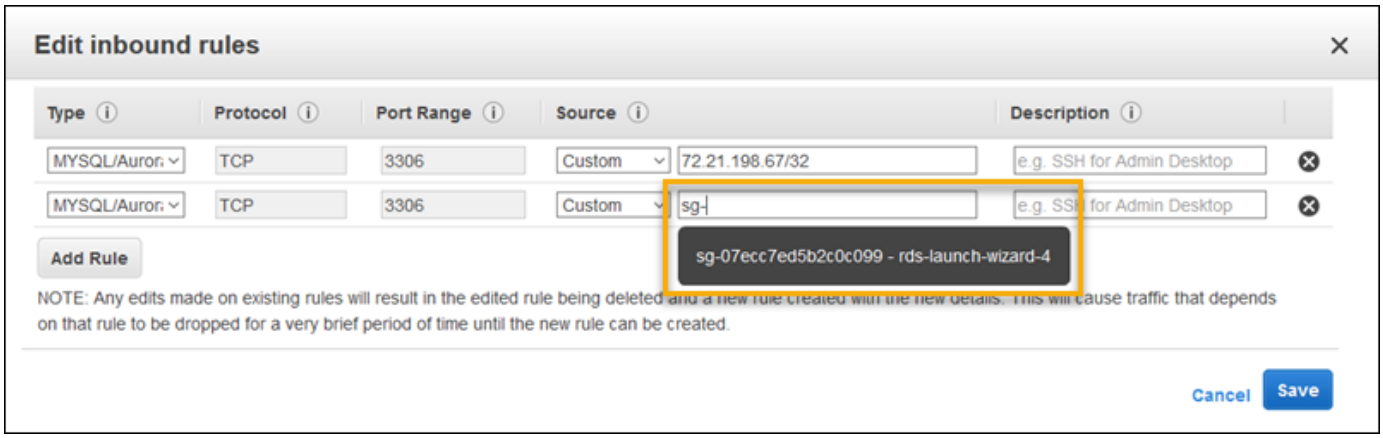
5. Dalam Konfigurasi tambahan, untuk Nama basis data awal, ketik **ebdb**.
6. Tinjau pengaturan default dan sesuaikan pengaturan ini sesuai dengan kebutuhan spesifik Anda. Perhatikan opsi berikut:
   - Kelas instans DB – Memilih ukuran instans yang memiliki jumlah memori dan daya CPU yang sesuai dengan beban kerja Anda.
   - Penerapan multi-AZ — Untuk ketersediaan tinggi, atur ini ke Buat Replica/Reader simpul Aurora di AZ yang berbeda.
   - Nama pengguna utama dan Kata sandi utama – Nama pengguna dan kata sandi basis data. Catat pengaturan ini karena Anda akan menggunakannya nanti.
7. Verifikasi pengaturan default untuk opsi lainnya, dan kemudian pilih Buat basis data.

Berikutnya, modifikasi grup keamanan yang dilampirkan ke instans DB Anda untuk memperbolehkan lintas masuk pada port yang sesuai. Modifikasi grup keamanan adalah grup keamanan yang sama yang akan Anda lampirkan ke lingkungan Elastic Beanstalk Anda kelak, sehingga aturan yang ditambahkan akan memberikan izin masuk ke sumber daya lain dalam grup keamanan yang sama.

Untuk mengubah aturan masuk pada grup keamanan yang dilampirkan ke instans RDS Anda

1. Buka konsol Amazon RDS.
2. Pilih Basis data.

3. Pilih nama instans DB Anda untuk menampilkan detailnya.
4. Di bagian Konektivitas, catat Subnet, grup Keamanan, dan Titik Akhir yang ditampilkan di halaman ini. Ini agar Anda dapat menggunakan informasi ini nanti.
5. Di bawah Keamanan, Anda dapat melihat grup keamanan yang terkait dengan instans DB. Buka tautan untuk melihat grup keamanan di EC2 konsol Amazon.
6. Pada detail grup keamanan, pilih Masuk.
7. Pilih Edit.
8. Pilih Tambahkan Aturan.
9. Untuk Jenis, pilih mesin DB yang digunakan aplikasi Anda.
10. Untuk Sumber, ketik **`sg-`** untuk melihat daftar grup keamanan yang tersedia. Pilih grup keamanan yang terkait dengan grup Auto Scaling yang digunakan dengan lingkungan Elastic Beanstalk Anda. Ini agar EC2 instans Amazon di lingkungan dapat memiliki akses ke database.



11. Pilih Simpan.

Membuat instans DB memakan waktu sekitar 10 menit. Sementara itu, buat lingkungan Elastic Beanstalk Anda.

Buat lingkungan Elastic Beanstalk

Gunakan konsol Elastic Beanstalk untuk membuat lingkungan Elastic Beanstalk. Pilih platform PHP lalu terima pengaturan default dan kode sampel. Setelah Anda meluncurkan lingkungan, Anda dapat mengonfigurasi lingkungan untuk terhubung ke database, lalu menyebarkan aplikasi sampel yang Anda unduh. GitHub

Untuk meluncurkan lingkungan (konsol)

1. Buka konsol Elastic Beanstalk menggunakan tautan yang telah dikonfigurasi sebelumnya: console.aws.amazon. com/elasticbeanstalk/home#/newApplication?applicationName=Tutorial&EnvironmentType= LoadBalanced
2. Untuk Platform, pilih platform dan cabang platform yang sesuai dengan bahasa yang digunakan oleh aplikasi Anda.
3. Untuk Kode aplikasi, pilih Aplikasi sampel.
4. Pilih Tinjau dan Luncurkan.
5. Tinjau opsi yang tersedia. Pilih opsi tersedia yang ingin Anda gunakan, dan saat Anda siap, pilih Buat aplikasi.

Pembuatan lingkungan membutuhkan waktu sekitar 5 menit dan membuat sumber daya berikut:

- EC2 instance — Mesin virtual Amazon Elastic Compute Cloud (Amazon EC2) yang dikonfigurasi untuk menjalankan aplikasi web pada platform yang Anda pilih.

  Setiap platform menjalankan satu set perangkat lunak, file konfigurasi dan penulisan tertentu untuk mendukung versi bahasa, kerangka kerja, kontainer web tertentu, atau kombinasi dari semua ini. Sebagian besar platform menggunakan Apache atau NGINX sebagai proksi terbalik yang ada di depan aplikasi web Anda, meneruskan permintaan ke aplikasi web, menyajikan aset statis, dan menghasilkan log akses dan kesalahan.
- Grup keamanan instans — Grup EC2 keamanan Amazon yang dikonfigurasi untuk memungkinkan lalu lintas masuk pada port 80. Sumber daya ini memungkinkan lalu lintas HTTP dari penyeimbang beban mencapai EC2 instance yang menjalankan aplikasi web Anda. Secara default, lalu lintas tidak diizinkan di port lain.
- Penyeimbang beban – Penyeimbang beban Elastic Load Balancing yang dikonfigurasi untuk mendistribusikan permintaan ke instans yang menjalankan aplikasi Anda. Penyeimbang beban juga menghilangkan kebutuhan untuk mengekspos instans Anda langsung ke internet.
- Grup keamanan penyeimbang beban - Grup EC2 keamanan Amazon yang dikonfigurasi untuk memungkinkan lalu lintas masuk di port 80. Sumber daya ini memungkinkan lalu lintas HTTP dari internet mencapai penyeimbang beban. Secara default, lalu lintas tidak diizinkan di port lain.
- Grup Auto Scaling – Grup Auto Scaling yang dikonfigurasi untuk menggantikan instans jika diakhiri atau menjadi tidak tersedia.

- Bucket Amazon S3 – Lokasi penyimpanan untuk kode sumber, log, dan artifact lainnya yang dibuat saat Anda menggunakan Elastic Beanstalk.
- CloudWatch Alarm Amazon — Dua CloudWatch alarm yang memantau beban pada instans di lingkungan Anda dan yang dipicu jika beban terlalu tinggi atau terlalu rendah. Saat alarm terpicu, grup Auto Scaling Anda akan menaikkan atau menurunkan skala sebagai respons.
- AWS CloudFormation stack - Elastic AWS CloudFormation Beanstalk digunakan untuk meluncurkan sumber daya di lingkungan Anda dan menyebarkan perubahan konfigurasi. Sumber daya ditentukan di sebuah templat yang dapat Anda lihat di Konsol AWS CloudFormation.
- Nama domain — Nama domain yang merutekan ke aplikasi web Anda dalam formulir *subdomain*.*region*.elasticbeanstalk.com.

> **Keamanan domain**
>
> Untuk meningkatkan keamanan aplikasi Elastic Beanstalk Anda, domain elasticbeanstalk.com terdaftar di Daftar Akhiran Publik (PSL).
> Jika Anda perlu mengatur cookie sensitif di nama domain default untuk aplikasi Elastic Beanstalk Anda, kami sarankan Anda menggunakan cookie `__Host-` dengan awalan untuk meningkatkan keamanan. Praktik ini melindungi domain Anda dari upaya pemalsuan permintaan lintas situs (CSRF). Untuk informasi selengkapnya, lihat halaman Set-Cookie di Jaringan Pengembang Mozilla.

Semua sumber daya ini dikelola oleh Elastic Beanstalk. Ketika Anda mengakhiri lingkungan, Elastic Beanstalk mengakhiri semua sumber daya yang dimuatnya. Instans DB RDS yang diluncurkan berada di luar lingkungan Anda, sehingga Anda bertanggung jawab untuk mengelola siklus hidupnya.

> **Note**
>
> Bucket Amazon S3 yang dibuat Elastic Beanstalk dibagi antar lingkungan dan tidak dihapus selama pengakhiran lingkungan. Untuk informasi selengkapnya, lihat Menggunakan Elastic Beanstalk dengan Amazon S3.

Mengonfigurasi grup keamanan, properti lingkungan, dan penskalaan

Menambahkan grup keamanan instans DB Anda ke lingkungan Anda yang sedang berjalan. Prosedur ini menyebabkan Elastic Beanstalk menyediakan kembali instans di lingkungan Anda dengan grup keamanan tambahan terlampir.

Untuk menambahkan grup keamanan ke lingkungan Anda

- Lakukan salah satu dari berikut ini:
  - Untuk menambahkan grup keamanan menggunakan konsol Elastic Beanstalk
    a. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
    b. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
    c. Pada panel navigasi, pilih Konfigurasi.
    d. Pada kategori konfigurasi Instans, pilih Edit.
    e. Di bawah grup EC2 keamanan, pilih grup keamanan untuk dilampirkan ke instance, selain grup keamanan instans yang dibuat Elastic Beanstalk.
    f. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.
    g. Baca peringatan, kemudian pilih Konfirmasi.
  - Untuk menambahkan grup keamanan menggunakan file konfigurasi, gunakan file `securitygroup-addexisting.config`contoh.

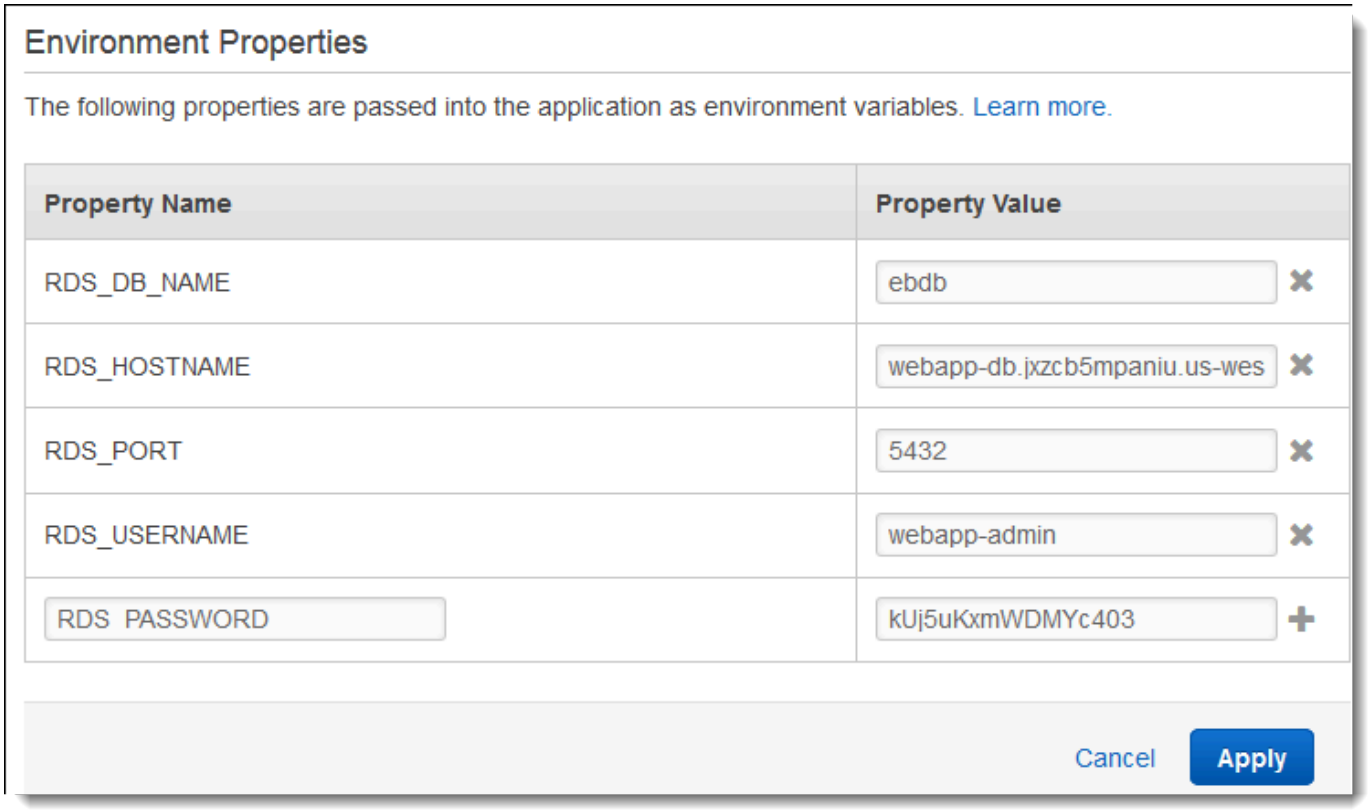
Selanjutnya, gunakan properti lingkungan untuk meneruskan informasi koneksi ke lingkungan Anda. Aplikasi sampel menggunakan set properti default yang cocok dengan properti yang dikonfigurasi Elastic Beanstalk saat Anda menyediakan basis data di lingkungan Anda.

Untuk mengonfigurasi properti lingkungan bagi instans DB Amazon RDS

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Dalam kategori konfigurasi Pembaruan, pemantauan, dan pencatatan, pilih Edit.

5. Pada bagian Properti lingkungan, tentukan variabel yang dibaca aplikasi Anda untuk membangun string koneksi. Untuk kompatibilitas dengan lingkungan yang memiliki instans DB RDS terintegrasi, gunakan nama dan nilai-nilai berikut. Anda dapat menemukan semua nilai, kecuali untuk kata sandi Anda, di Konsol RDS.

Nama properti	Deskripsi	Nilai properti
`RDS_HOSTNAME`	Nama host instans DB.	Di tab Konektivitas & keamanan di konsol Amazon RDS: Titik akhir.
`RDS_PORT`	Port tempat instans DB menerima koneksi. Nilai default bervariasi di antara mesin DB.	Di tab Konektivitas & keamanan di konsol Amazon RDS: Port.
`RDS_DB_NAME`	Nama basis data, **ebdb**.	Di tab Konfigurasi di konsol Amazon RDS: Nama DB.
`RDS_USERNAME`	Nama pengguna yang Anda konfigurasi untuk basis data Anda.	Di tab Konfigurasi di konsol Amazon RDS: Nama pengguna utama.
`RDS_PASSWORD`	Kata sandi yang Anda konfigurasi untuk basis data Anda.	Tidak tersedia untuk referensi di konsol Amazon RDS.



6. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Terakhir, konfigurasikan grup Auto Scaling lingkungan Anda dengan jumlah instans minimum yang lebih tinggi. Jalankan setidaknya dua instans setiap saat untuk mencegah terjadinya kegagalan di satu titik server web di lingkungan Anda, dan mengizinkan Anda untuk men-deploy perubahan tanpa membuat situs Anda keluar dari layanan.

Mengonfigurasi grup Auto Scaling lingkungan Anda untuk ketersediaan yang tinggi

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Pada kategori konfigurasi Kapasitas, pilih Edit.
5. Pada bagian Grup Auto Scaling, set Instans minimum ke **2**.
6. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Men-deploy aplikasi sampel

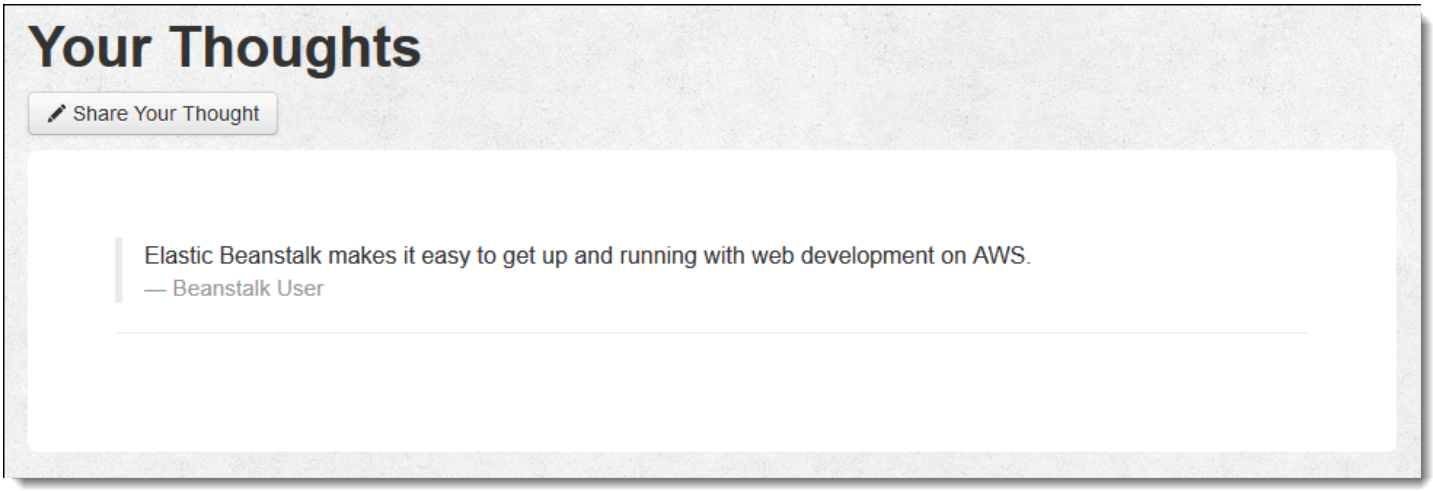
Sekarang lingkungan Anda siap menjalankan aplikasi sampel dan terhubung ke Amazon RDS. Men-deploy aplikasi sampel ke lingkungan Anda.

> **Note**
>
> Unduh bundel sumber dari GitHub, jika Anda belum melakukannya: eb-demo-php-simple-app-1.3.zip

Untuk men-deploy paket sumber

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Di halaman gambaran umum lingkungan, pilih Unggah dan deploy.
4. Gunakan kotak dialog di layar untuk mengunggah paket sumber.
5. Pilih Deploy.
6. Ketika deployment selesai, Anda dapat memilih URL situs untuk membuka situs web Anda di tab baru.

Situs ini mengumpulkan komentar pengguna dan menggunakan basis data MySQL untuk menyimpan data. Untuk menambahkan komentar, pilih Bagikan Pemikiran Anda, masukkan komentar, dan kemudian pilih Kirim Pemikiran Anda. Aplikasi web menuliskan komentar ke basis data sehingga setiap instans di lingkungan dapat membacanya, dan tidak akan hilang jika instans keluar dari layanan.



Pembersihan

Setelah Anda selesai bekerja dengan kode demo, Anda dapat mengakhiri lingkungan Anda. Elastic Beanstalk menghapus semua sumber daya AWS terkait, seperti instans EC2 Amazon, instans database, penyeimbang beban, grup keamanan,dan alarm.

Menghapus sumber daya tidak menghapus aplikasi Elastic Beanstalk, sehingga Anda dapat membuat lingkungan baru untuk aplikasi Anda kapan saja.

Untuk mengakhiri lingkungan Elastic Beanstalk Anda dari konsol

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pilih Tindakan, lalu pilih Terminate environment.
4. Gunakan kotak dialog di layar untuk mengonfirmasi pengakhiran lingkungan.

Selain itu, Anda dapat mengakhiri sumber daya basis data yang Anda buat di luar lingkungan Elastic Beanstalk Anda. Ketika Anda mengakhiri instans DB Amazon RDS, Anda dapat mengambil snapshot dan memulihkan data ke instans lain kelak.

Untuk mengakhiri instans DB RDS Anda

1. Buka konsol Amazon RDS.
2. Pilih Basis data.
3. Pilih instans DB Anda.

4. Pilih Tindakan, dan lalu pilih Hapus.
5. Pilih apakah akan membuat snapshot, dan kemudian memilih Hapus.

Langkah selanjutnya

Ketika aplikasi terus dikembangkan, Anda mungkin akan menginginkan sebuah cara untuk mengelola lingkungan dan men-deploy aplikasi Anda tanpa membuat file .zip secara manual dan mengunggahnya ke konsol Elastic Beanstalk. Elastic Beanstalk Command Line Interface (EB CLI easy-to-use) menyediakan perintah untuk membuat, mengkonfigurasi, dan menyebarkan aplikasi ke lingkungan Elastic Beanstalk dari baris perintah.

Aplikasi sampel menggunakan file konfigurasi untuk mengonfigurasi pengaturan PHP dan membuat tabel dalam basis data jika tidak ada. Anda juga dapat menggunakan file konfigurasi untuk mengonfigurasi pengaturan grup keamanan instans Anda selama pembuatan lingkungan untuk menghindari pembaruan konfigurasi yang cukup memakan waktu. Lihat Penyesuaian lingkungan lanjutan dengan file konfigurasi (`.ebextensions`) untuk informasi selengkapnya.

Untuk pengembangan dan pengujian, Anda mungkin ingin menggunakan fungsionalitas Elastic Beanstalk dalam menambahkan instans DB terkelola langsung ke lingkungan Anda. Terkait petunjuk tentang pengaturan basis data di dalam lingkungan Anda, lihat Menambahkan basis data ke lingkungan Elastic Beanstalk Anda.

Jika Anda membutuhkan basis data berperforma tinggi, pertimbangkan untuk menggunakan Amazon Aurora. Amazon Aurora merupakan mesin basis data kompatibel dengan MySQL yang menawarkan fitur basis data komersial dengan biaya rendah. Untuk menghubungkan aplikasi Anda ke basis data yang berbeda, ulangi langkah konfigurasi grup keamanan dan perbarui properti lingkungan terkait RDS.

Terakhir, jika Anda berencana menggunakan aplikasi dalam lingkungan produksi, Anda akan ingin mengonfigurasikan nama domain khusus untuk lingkungan Anda dan mengaktifkan HTTPS untuk koneksi yang aman.

Menyebarkan WordPress situs web dengan ketersediaan tinggi dengan database Amazon RDS eksternal ke Elastic Beanstalk

Tutorial ini menjelaskan cara meluncurkan instans Amazon RDS DB yang eksternal AWS Elastic Beanstalk, lalu cara mengonfigurasi lingkungan ketersediaan tinggi yang menjalankan WordPress

situs web untuk terhubung dengannya. Situs web menggunakan Amazon Elastic File System (Amazon EFS) sebagai penyimpanan bersama untuk file yang diunggah.

Menjalankan instans DB eksternal Elastic Beanstalk akan memisahkan basis data dari siklus hidup lingkungan Anda. Hal ini memungkinkan Anda terhubung ke basis data yang sama dari beberapa lingkungan, menukarkan satu basis data dengan yang lain, atau melakukan deployment biru/hijau tanpa memengaruhi basis data Anda.

> **Note**
>
> Untuk informasi terkini tentang kompatibilitas rilis PHP dengan WordPress versi, lihat Kompatibilitas PHP dan WordPress Versi di WordPress situs web. Anda harus merujuk ke informasi ini sebelum Anda meng-upgrade ke rilis baru PHP untuk WordPress implementasi Anda.

Topik

- Prasyarat
- Meluncurkan instans DB pada Amazon RDS
- Unduh WordPress
- Meluncurkan lingkungan Elastic Beanstalk
- Mengonfigurasi grup keamanan dan properti lingkungan
- Mengonfigurasi dan men-deploy aplikasi Anda
- Instal WordPress
- Memperbarui kunci dan salt
- Menghapus batasan akses
- Mengonfigurasi grup Auto Scaling
- Tingkatkan WordPress
- Bersihkan
- Langkah selanjutnya

Prasyarat

Tutorial ini mengasumsikan Anda memiliki pengetahuan tentang operasi Elastic Beanstalk dasar dan konsol Elastic Beanstalk. Jika belum, ikuti petunjuk di Pelajari cara memulai dengan Elastic Beanstalk untuk meluncurkan lingkungan Elastic Beanstalk pertama Anda.

Untuk mengikuti prosedur di panduan ini, Anda memerlukan terminal atau shell baris perintah untuk menjalankan perintah. Perintah ditampilkan dalam daftar yang diawali dengan simbol prompt ($) dan nama direktori saat ini, bila perlu.

```
~/eb-project$ this is a command
this is output
```

Di Linux dan macOS, Anda dapat menggunakan shell dan manajer paket pilihan Anda. Pada Windows Anda dapat menginstal Windows Subsystem untuk Linux untuk mendapatkan versi Windows terintegrasi dari Ubuntu dan Bash.

VPC default

Prosedur Amazon Relational Database Service (Amazon RDS) di tutorial ini mengasumsikan bahwa Anda meluncurkan sumber daya pada Amazon Virtual Private Cloud (Amazon VPC) default. Semua akun baru menyertakan VPC default di setiap AWS Wilayah. Jika Anda tidak memiliki VPC default, prosedurnya akan bervariasi. Lihat petunjuk Menggunakan Elastic Beanstalk dengan Amazon RDS untuk platform EC2 -Classic dan VPC kustom.

AWS Daerah

Aplikasi sampel menggunakan Amazon EFS, yang hanya berfungsi di AWS Wilayah yang mendukung Amazon EFS. Untuk mempelajari tentang AWS Wilayah yang didukung, lihat Titik Akhir dan Kuota Amazon Elastic File System di bagian. Referensi Umum AWS

Meluncurkan instans DB pada Amazon RDS

Ketika Anda meluncurkan sebuah instans dengan Amazon RDS, instans tersebut sepenuhnya independen dari Elastic Beanstalk dan lingkungan Elastic Beanstalk Anda, dan tidak akan diakhiri atau dipantau oleh Elastic Beanstalk.

Pada langkah-langkah berikut, Anda akan menggunakan konsol Amazon RDS untuk:

- Peluncuran basis data dengan mesin MySQL.

- Mengaktifkan Deployment Multi-AZ. Ini menciptakan siaga di Availability Zone (AZ) yang berbeda untuk menyediakan redundansi data, menghilangkan I/O pembekuan, dan meminimalkan lonjakan latensi selama pencadangan sistem.

Untuk meluncurkan instans DB RDS pada VPC default

1. Buka konsol RDS.
2. Di panel navigasi, pilih Basis Data.
3. Pilih Buat basis data.
4. Pilih Pembuatan Standar.

> **Important**
>
> Jangan pilih Pembuatan Mudah. Jika Anda memilihnya, Anda tidak dapat mengonfigurasi pengaturan yang diperlukan untuk meluncurkan RDS DB ini.

5. Dalam Konfigurasi tambahan, untuk Nama basis data awal, ketik **ebdb**.
6. Tinjau pengaturan default dan sesuaikan pengaturan ini sesuai dengan kebutuhan spesifik Anda. Perhatikan opsi berikut:
   - Kelas instans DB – Memilih ukuran instans yang memiliki jumlah memori dan daya CPU yang sesuai dengan beban kerja Anda.
   - Penerapan multi-AZ — Untuk ketersediaan tinggi, atur ini ke Buat Replica/Reader simpul Aurora di AZ yang berbeda.
   - Nama pengguna utama dan Kata sandi utama – Nama pengguna dan kata sandi basis data. Catat pengaturan ini karena Anda akan menggunakannya nanti.
7. Verifikasi pengaturan default untuk opsi lainnya, dan kemudian pilih Buat basis data.

Setelah instans DB Anda dibuat, modifikasi grup keamanan yang terlampir untuk memungkinkan lalu lintas masuk pada port yang sesuai.

> **Note**
>
> Ini adalah grup keamanan yang sama yang akan Anda lampirkan ke lingkungan Elastic Beanstalk Anda nanti, sehingga aturan yang Anda tambahkan sekarang akan memberikan izin masuk ke sumber daya lain dalam grup keamanan yang sama.

Untuk mengubah aturan masuk pada grup keamanan yang dilampirkan ke instans RDS Anda

1. Buka konsol Amazon RDS.
2. Pilih Basis data.
3. Pilih nama instans DB Anda untuk menampilkan detailnya.
4. Di bagian Konektivitas, catat Subnet, grup Keamanan, dan Titik Akhir yang ditampilkan di halaman ini. Ini agar Anda dapat menggunakan informasi ini nanti.
5. Di bawah Keamanan, Anda dapat melihat grup keamanan yang terkait dengan instans DB. Buka tautan untuk melihat grup keamanan di EC2 konsol Amazon.
6. Pada detail grup keamanan, pilih Masuk.
7. Pilih Edit.
8. Pilih Tambahkan Aturan.
9. Untuk Jenis, pilih mesin DB yang digunakan aplikasi Anda.
10. Untuk Sumber, ketik **`sg-`** untuk melihat daftar grup keamanan yang tersedia. Pilih grup keamanan yang terkait dengan grup Auto Scaling yang digunakan dengan lingkungan Elastic Beanstalk Anda. Ini agar EC2 instans Amazon di lingkungan dapat memiliki akses ke database.



11. Pilih Simpan.

Membuat instans DB memakan waktu sekitar 10 menit. Sementara itu, unduh WordPress dan buat lingkungan Elastic Beanstalk Anda.

Unduh WordPress

Untuk mempersiapkan penerapan WordPress menggunakan AWS Elastic Beanstalk, Anda harus menyalin WordPress file ke komputer Anda dan memberikan informasi konfigurasi yang benar.

Untuk membuat WordPress proyek

1. Unduh WordPress dari wordpress.org.

```
~$curl https://wordpress.org/wordpress-6.2.tar.gz -o wordpress.tar.gz
```

2. Mengunduh file konfigurasi dari repositori sampel.

```
~$ wget https://github.com/aws-samples/eb-php-wordpress/releases/download/v1.1/eb-
php-wordpress-v1.zip
```

3. Ekstrak WordPress dan ubah nama folder.

```
~$ tar -xvf wordpress.tar.gz
~$ mv wordpress wordpress-beanstalk
~$ cd wordpress-beanstalk
```

4. Ekstrak file konfigurasi melalui WordPress instalasi.

```
~/wordpress-beanstalk$ unzip ../eb-php-wordpress-v1.zip
 creating: .ebextensions/
inflating: .ebextensions/dev.config
inflating: .ebextensions/efs-create.config
inflating: .ebextensions/efs-mount.config
inflating: .ebextensions/loadbalancer-sg.config
inflating: .ebextensions/wordpress.config
inflating: LICENSE
inflating: README.md
inflating: wp-config.php
```

Meluncurkan lingkungan Elastic Beanstalk

Gunakan konsol Elastic Beanstalk untuk membuat lingkungan Elastic Beanstalk. Setelah Anda meluncurkan lingkungan, Anda dapat mengonfigurasinya untuk terhubung ke database, lalu menyebarkan WordPress kode ke lingkungan.

Pada langkah-langkah berikut, Anda akan menggunakan konsol Elastic Beanstalk untuk:

- Membuat aplikasi Elastic Beanstalk menggunakan platform PHP terkelola.
- Menerima pengaturan default dan kode sampel.

Untuk meluncurkan lingkungan (konsol)

1. Buka konsol Elastic Beanstalk menggunakan tautan yang telah dikonfigurasi sebelumnya: console.aws.amazon. com/elasticbeanstalk/home#/newApplication?applicationName=Tutorial&EnvironmentType= LoadBalanced
2. Untuk Platform, pilih platform dan cabang platform yang sesuai dengan bahasa yang digunakan oleh aplikasi Anda.
3. Untuk Kode aplikasi, pilih Aplikasi sampel.
4. Pilih Tinjau dan Luncurkan.
5. Tinjau opsi yang tersedia. Pilih opsi tersedia yang ingin Anda gunakan, dan saat Anda siap, pilih Buat aplikasi.

Pembuatan lingkungan membutuhkan waktu sekitar lima menit hingga sumber daya berikut dibuat.

Sumber daya yang dibuat Elastic Beanstalk

- EC2 instance — Mesin virtual Amazon Elastic Compute Cloud (Amazon EC2) yang dikonfigurasi untuk menjalankan aplikasi web pada platform yang Anda pilih.

  Setiap platform menjalankan satu set perangkat lunak, file konfigurasi dan penulisan tertentu untuk mendukung versi bahasa, kerangka kerja, kontainer web tertentu, atau kombinasi dari semua ini. Sebagian besar platform menggunakan Apache atau NGINX sebagai proksi terbalik yang ada di depan aplikasi web Anda, meneruskan permintaan ke aplikasi web, menyajikan aset statis, dan menghasilkan log akses dan kesalahan.
- Grup keamanan instans — Grup EC2 keamanan Amazon yang dikonfigurasi untuk memungkinkan lalu lintas masuk pada port 80. Sumber daya ini memungkinkan lalu lintas HTTP dari penyeimbang

beban mencapai EC2 instance yang menjalankan aplikasi web Anda. Secara default, lalu lintas tidak diizinkan di port lain.

- Penyeimbang beban – Penyeimbang beban Elastic Load Balancing yang dikonfigurasi untuk mendistribusikan permintaan ke instans yang menjalankan aplikasi Anda. Penyeimbang beban juga menghilangkan kebutuhan untuk mengekspos instans Anda langsung ke internet.
- Grup keamanan penyeimbang beban - Grup EC2 keamanan Amazon yang dikonfigurasi untuk memungkinkan lalu lintas masuk di port 80. Sumber daya ini memungkinkan lalu lintas HTTP dari internet mencapai penyeimbang beban. Secara default, lalu lintas tidak diizinkan di port lain.
- Grup Auto Scaling – Grup Auto Scaling yang dikonfigurasi untuk menggantikan instans jika diakhiri atau menjadi tidak tersedia.
- Bucket Amazon S3 – Lokasi penyimpanan untuk kode sumber, log, dan artifact lainnya yang dibuat saat Anda menggunakan Elastic Beanstalk.
- CloudWatch Alarm Amazon — Dua CloudWatch alarm yang memantau beban pada instans di lingkungan Anda dan yang dipicu jika beban terlalu tinggi atau terlalu rendah. Saat alarm terpicu, grup Auto Scaling Anda akan menaikkan atau menurunkan skala sebagai respons.
- AWS CloudFormation stack - Elastic AWS CloudFormation Beanstalk digunakan untuk meluncurkan sumber daya di lingkungan Anda dan menyebarkan perubahan konfigurasi. Sumber daya ditentukan di sebuah templat yang dapat Anda lihat di Konsol AWS CloudFormation.
- Nama domain — Nama domain yang merutekan ke aplikasi web Anda dalam formulir *`subdomain`*.*`region`*.elasticbeanstalk.com.

> **Keamanan domain**
>
> Untuk meningkatkan keamanan aplikasi Elastic Beanstalk Anda, domain elasticbeanstalk.com terdaftar di Daftar Akhiran Publik (PSL).
> Jika Anda perlu mengatur cookie sensitif di nama domain default untuk aplikasi Elastic Beanstalk Anda, kami sarankan Anda menggunakan cookie `__Host-` dengan awalan untuk meningkatkan keamanan. Praktik ini melindungi domain Anda dari upaya pemalsuan permintaan lintas situs (CSRF). Untuk informasi selengkapnya, lihat halaman Set-Cookie di Jaringan Pengembang Mozilla.

Semua sumber daya ini dikelola oleh Elastic Beanstalk. Ketika Anda mengakhiri lingkungan, Elastic Beanstalk mengakhiri semua sumber daya yang dimuatnya.

Instans Amazon RDS yang diluncurkan berada di luar lingkungan Anda, Anda bertanggung jawab untuk mengelola siklus hidupnya.

> **Note**
>
> Bucket Amazon S3 yang dibuat Elastic Beanstalk dibagi antar lingkungan dan tidak dihapus selama pengakhiran lingkungan. Untuk informasi selengkapnya, lihat Menggunakan Elastic Beanstalk dengan Amazon S3.

Mengonfigurasi grup keamanan dan properti lingkungan

Menambahkan grup keamanan instans DB Anda ke lingkungan Anda yang sedang berjalan. Prosedur ini menyebabkan Elastic Beanstalk menyediakan kembali instans di lingkungan Anda dengan grup keamanan tambahan terlampir.

Untuk menambahkan grup keamanan ke lingkungan Anda

- Lakukan salah satu dari berikut ini:
  - Untuk menambahkan grup keamanan menggunakan konsol Elastic Beanstalk
    a. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
    b. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
    c. Pada panel navigasi, pilih Konfigurasi.
    d. Pada kategori konfigurasi Instans, pilih Edit.
    e. Di bawah grup EC2 keamanan, pilih grup keamanan untuk dilampirkan ke instance, selain grup keamanan instans yang dibuat Elastic Beanstalk.
    f. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.
    g. Baca peringatan, kemudian pilih Konfirmasi.
  - Untuk menambahkan grup keamanan menggunakan file konfigurasi, gunakan file `securitygroup-addexisting.config`contoh.

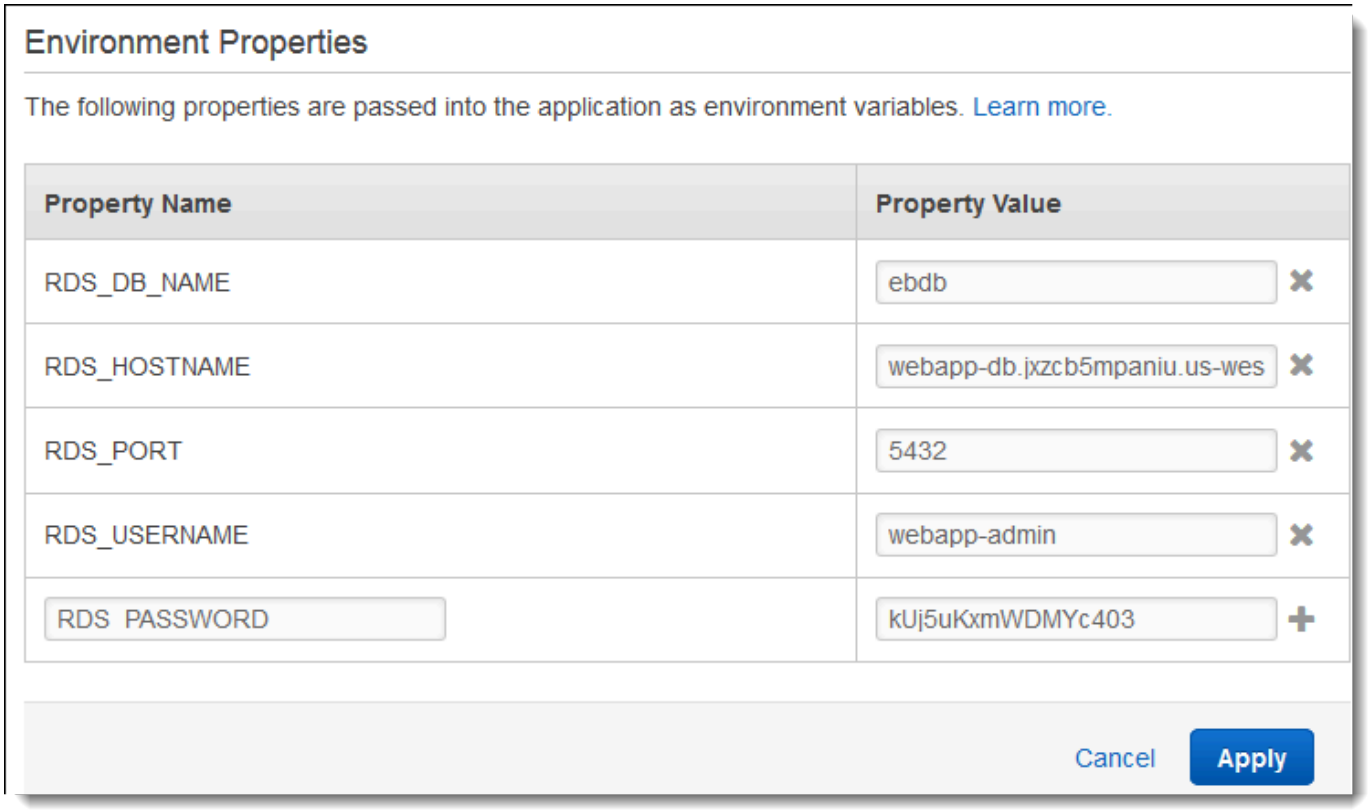
Selanjutnya, gunakan properti lingkungan untuk meneruskan informasi koneksi ke lingkungan Anda.

WordPress Aplikasi ini menggunakan seperangkat properti default yang cocok dengan properti yang dikonfigurasi Elastic Beanstalk saat Anda menyediakan database dalam lingkungan Anda.

Untuk mengonfigurasi properti lingkungan bagi instans DB Amazon RDS

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Dalam kategori konfigurasi Pembaruan, pemantauan, dan pencatatan, pilih Edit.
5. Pada bagian Properti lingkungan, tentukan variabel yang dibaca aplikasi Anda untuk membangun string koneksi. Untuk kompatibilitas dengan lingkungan yang memiliki instans DB RDS terintegrasi, gunakan nama dan nilai-nilai berikut. Anda dapat menemukan semua nilai, kecuali untuk kata sandi Anda, di Konsol RDS.

Nama properti	Deskripsi	Nilai properti
`RDS_HOSTNAME`	Nama host instans DB.	Di tab Konektivitas & keamanan di konsol Amazon RDS: Titik akhir.
`RDS_PORT`	Port tempat instans DB menerima koneksi. Nilai default bervariasi di antara mesin DB.	Di tab Konektivitas & keamanan di konsol Amazon RDS: Port.
`RDS_DB_NAME`	Nama basis data, **ebdb**.	Di tab Konfigurasi di konsol Amazon RDS: Nama DB.
`RDS_USERNAME`	Nama pengguna yang Anda konfigurasi untuk basis data Anda.	Di tab Konfigurasi di konsol Amazon RDS: Nama pengguna utama.
`RDS_PASSWORD`	Kata sandi yang Anda konfigurasi untuk basis data Anda.	Tidak tersedia untuk referensi di konsol Amazon RDS.



6. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Mengonfigurasi dan men-deploy aplikasi Anda

Verifikasi bahwa struktur folder `wordpress-beanstalk` Anda benar, seperti yang ditunjukkan.

```
wordpress-beanstalk$ tree -aL 1
.
### .ebextensions
### index.php
### LICENSE
### license.txt
### readme.html
### README.md
### wp-activate.php
### wp-admin
### wp-blog-header.php
### wp-comments-post.php
### wp-config.php
### wp-config-sample.php
```

```
### wp-content
### wp-cron.php
### wp-includes
### wp-links-opml.php
### wp-load.php
### wp-login.php
### wp-mail.php
### wp-settings.php
### wp-signup.php
### wp-trackback.php
### xmlrpc.php
```

File `wp-config.php` yang dikustomisasi dari repo proyek menggunakan variabel lingkungan yang Anda tetapkan di langkah sebelumnya untuk mengonfigurasi koneksi basis data. Folder `.ebextensions` berisi file konfigurasi yang membuat sumber daya tambahan dalam lingkungan Elastic Beanstalk Anda.

File konfigurasi memerlukan modifikasi untuk dapat bekerja dengan akun Anda. Ganti nilai placeholder dalam file dengan yang sesuai IDs dan buat bundel sumber.

Untuk memperbarui file konfigurasi dan membuat paket sumber

1. Memodifikasi file konfigurasi sebagai berikut.

   - `.ebextensions/dev.config`— Membatasi akses ke lingkungan Anda untuk melindunginya selama proses WordPress instalasi. Ganti alamat IP placeholder di dekat bagian atas file dengan alamat IP publik komputer yang akan Anda gunakan untuk mengakses situs web lingkungan Anda untuk menyelesaikan instalasi Anda WordPress .

     > **Note**
     >
     > Tergantung pada jaringan, Anda mungkin harus menggunakan blok alamat IP.

   - `.ebextensions/efs-create.config`— Membuat sistem file EFS dan memasang titik di setiap Ketersediaan Zone/subnet di VPC Anda. Identifikasi VPC dan subnet default Anda IDs di konsol VPC Amazon.

2. Membuat paket sumber berisi file dalam folder proyek Anda. Perintah berikut membuat paket sumber yang bernama `wordpress-beanstalk.zip`.

   ```
   ~/eb-wordpress$ zip ../wordpress-beanstalk.zip -r * .[^.]*
   ```

Unggah bundel sumber ke Elastic Beanstalk WordPress untuk diterapkan ke lingkungan Anda.

Untuk men-deploy paket sumber

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Di halaman gambaran umum lingkungan, pilih Unggah dan deploy.
4. Gunakan kotak dialog di layar untuk mengunggah paket sumber.
5. Pilih Deploy.
6. Ketika deployment selesai, Anda dapat memilih URL situs untuk membuka situs web Anda di tab baru.

Instal WordPress

Untuk menyelesaikan WordPress instalasi Anda

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pilih URL lingkungan untuk membuka situs Anda di peramban. Anda diarahkan ke wizard WordPress instalasi karena Anda belum mengkonfigurasi situs.
4. Lakukan instalasi standar. File `wp-config.php` sudah ada dalam kode sumber dan dikonfigurasi untuk membaca informasi koneksi basis data dari lingkungan. Anda seharusnya tidak diminta untuk mengonfigurasi sambungan.

Instalasi memakan waktu sekitar satu menit.

Memperbarui kunci dan salt

File WordPress konfigurasi `wp-config.php` juga membaca nilai untuk kunci dan garam dari properti lingkungan. Saat ini, properti ini sudah diatur ke `test` oleh file `wordpress.config` dalam folder `.ebextensions`.

Salt hash dapat berupa nilai apa pun yang memenuhi persyaratan properti lingkungan, tetapi Anda tidak boleh menyimpannya dalam kontrol sumber. Gunakan konsol Elastic Beanstalk untuk mengatur properti ini secara langsung pada lingkungan.

Untuk memperbarui properti lingkungan

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Di bawah Perangkat lunak, pilih Edit.
5. Untuk `Environment properties`, modifikasi properti berikut:
   - `AUTH_KEY` – Nilai yang dipilih untuk `AUTH_KEY`.
   - `SECURE_AUTH_KEY` – Nilai yang dipilih untuk `SECURE_AUTH_KEY`.
   - `LOGGED_IN_KEY` – Nilai yang dipilih untuk `LOGGED_IN_KEY`.
   - `NONCE_KEY` – Nilai yang dipilih untuk `NONCE_KEY`.
   - `AUTH_SALT` – Nilai yang dipilih untuk `AUTH_SALT`.
   - `SECURE_AUTH_SALT` – Nilai yang dipilih untuk `SECURE_AUTH_SALT`.
   - `LOGGED_IN_SALT` – Nilai yang dipilih untuk `LOGGED_IN_SALT`.
   - `NONCE_SALT` — Nilai yang dipilih untuk `NONCE_SALT`.
6. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

> **Note**
>
> Mengatur properti di lingkungan secara langsung menggantikan nilai pada `wordpress.config`.

Menghapus batasan akses

Proyek sampel berisi file konfigurasi `loadbalancer-sg.config`. File konfigurasi tersebut membuat grup keamanan dan memberikannya ke penyeimbang beban lingkungan, menggunakan alamat IP yang dikonfigurasi di `dev.config`. Hal tersebut membatasi akses HTTP pada port 80 ke sambungan dari jaringan Anda. Jika tidak, pihak luar berpotensi terhubung ke situs Anda sebelum Anda menginstal WordPress dan mengonfigurasi akun admin Anda.

Sekarang setelah Anda menginstal WordPress, hapus file konfigurasi untuk membuka situs ke dunia.

Untuk menghapus pembatasan dan memperbarui lingkungan Anda

1. Hapus file `.ebextensions/loadbalancer-sg.config` dari direktori proyek Anda.

```
~/wordpress-beanstalk$ rm .ebextensions/loadbalancer-sg.config
```

2. Buat paket sumber.

```
~/eb-wordpress$ zip ../wordpress-beanstalk-v2.zip -r * .[^.]*
```

Unggah bundel sumber ke Elastic Beanstalk WordPress untuk diterapkan ke lingkungan Anda.

Untuk men-deploy paket sumber

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Di halaman gambaran umum lingkungan, pilih Unggah dan deploy.
4. Gunakan kotak dialog di layar untuk mengunggah paket sumber.
5. Pilih Deploy.
6. Ketika deployment selesai, Anda dapat memilih URL situs untuk membuka situs web Anda di tab baru.

Mengonfigurasi grup Auto Scaling

Terakhir, konfigurasikan grup Auto Scaling lingkungan Anda dengan jumlah instans minimum yang lebih tinggi. Jalankan setidaknya dua instans setiap saat untuk mencegah terjadinya kegagalan di satu titik server web di lingkungan Anda. Hal ini juga memungkinkan Anda untuk men-deploy perubahan tanpa membuat situs Anda keluar dari layanan.

Mengonfigurasi grup Auto Scaling lingkungan Anda untuk ketersediaan yang tinggi

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.

4. Pada kategori konfigurasi Kapasitas, pilih Edit.
5. Pada bagian Grup Auto Scaling, set Instans minimum ke **2**.
6. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Untuk mendukung unggahan konten di beberapa instans, proyek sampel menggunakan Amazon EFS untuk membuat sistem file bersama. Buat kiriman di situs dan unggah konten untuk menyimpannya di sistem file bersama. Lihat kiriman dan refresh halaman beberapa kali untuk temuan kedua instans dan verifikasi apakah sistem file bersama tersebut bekerja.

Tingkatkan WordPress

Untuk meningkatkan ke versi baru WordPress, buat cadangan situs Anda dan terapkan ke lingkungan baru.

> **Important**
>
> Jangan gunakan fungsionalitas pembaruan di dalam WordPress atau perbarui file sumber Anda untuk menggunakan versi baru. Kedua tindakan ini dapat mengakibatkan posting Anda URLs mengembalikan kesalahan 404 meskipun mereka masih dalam database dan sistem file.

Untuk meng-upgrade WordPress

1. Di konsol WordPress admin, gunakan alat ekspor untuk mengekspor posting Anda ke file XHTML.
2. Terapkan dan instal versi baru WordPress ke Elastic Beanstalk dengan langkah yang sama yang Anda gunakan untuk menginstal versi sebelumnya. Untuk menghindari waktu henti, Anda dapat menyiasatinya dengan membuat lingkungan dengan versi baru.
3. Pada versi baru, instal alat WordPress Importir di konsol admin dan gunakan untuk mengimpor file XHTML yang berisi posting Anda. Jika kiriman dibuat oleh pengguna admin pada versi lama, berikanlah kiriman tersebut ke pengguna admin di situs baru daripada mencoba mengimpor pengguna admin.
4. Jika Anda men-deploy versi baru ke lingkungan terpisah, lakukan pertukaran CNAME untuk mengalihkan pengguna dari situs lama ke yang baru.

Bersihkan

Setelah Anda selesai bekerja dengan kode demo, Anda dapat mengakhiri lingkungan Anda. Elastic Beanstalk menghapus semua sumber daya AWS terkait, seperti instans EC2 Amazon, instans database, penyeimbang beban, grup keamanan,dan alarm.

Menghapus sumber daya tidak menghapus aplikasi Elastic Beanstalk, sehingga Anda dapat membuat lingkungan baru untuk aplikasi Anda kapan saja.

Untuk mengakhiri lingkungan Elastic Beanstalk Anda dari konsol

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pilih Tindakan, lalu pilih Terminate environment.
4. Gunakan kotak dialog di layar untuk mengonfirmasi pengakhiran lingkungan.

Selain itu, Anda dapat mengakhiri sumber daya basis data yang Anda buat di luar lingkungan Elastic Beanstalk Anda. Ketika Anda mengakhiri instans DB Amazon RDS, Anda dapat mengambil snapshot dan memulihkan data ke instans lain kelak.

Untuk mengakhiri instans DB RDS Anda

1. Buka konsol Amazon RDS.
2. Pilih Basis data.
3. Pilih instans DB Anda.
4. Pilih Tindakan, dan lalu pilih Hapus.
5. Pilih apakah akan membuat snapshot, dan kemudian memilih Hapus.

Langkah selanjutnya

Ketika aplikasi terus dikembangkan, Anda mungkin akan menginginkan sebuah cara untuk mengelola lingkungan dan men-deploy aplikasi Anda tanpa membuat file .zip secara manual dan mengunggahnya ke konsol Elastic Beanstalk. Elastic Beanstalk Command Line Interface (EB CLI easy-to-use) menyediakan perintah untuk membuat, mengkonfigurasi, dan menyebarkan aplikasi ke lingkungan Elastic Beanstalk dari baris perintah.

Aplikasi sampel menggunakan file konfigurasi untuk mengonfigurasi pengaturan PHP dan membuat tabel dalam basis data, jika belum ada. Anda juga dapat menggunakan file konfigurasi untuk mengonfigurasi pengaturan grup keamanan instans Anda selama pembuatan lingkungan untuk menghindari pembaruan konfigurasi yang cukup memakan waktu. Lihat Penyesuaian lingkungan lanjutan dengan file konfigurasi (`.ebextensions`) untuk informasi selengkapnya.

Untuk pengembangan dan pengujian, Anda mungkin ingin menggunakan fungsionalitas Elastic Beanstalk dalam menambahkan instans DB terkelola langsung ke lingkungan Anda. Terkait petunjuk tentang pengaturan basis data di dalam lingkungan Anda, lihat Menambahkan basis data ke lingkungan Elastic Beanstalk Anda.

Jika Anda membutuhkan basis data berperforma tinggi, pertimbangkan untuk menggunakan Amazon Aurora. Amazon Aurora merupakan mesin basis data kompatibel dengan MySQL yang menawarkan fitur basis data komersial dengan biaya rendah. Untuk menghubungkan aplikasi Anda ke basis data yang berbeda, ulangi langkah konfigurasi grup keamanan dan perbarui properti lingkungan terkait RDS.

Terakhir, jika Anda berencana menggunakan aplikasi dalam lingkungan produksi, Anda akan ingin mengonfigurasikan nama domain khusus untuk lingkungan Anda dan mengaktifkan HTTPS untuk koneksi yang aman.

Men-deploy situs web Drupal ketersediaan tinggi dengan basis data Amazon RDS eksternal ke Elastic Beanstalk

Tutorial ini memandu Anda melalui proses peluncuran instans RDS DB eksternal ke AWS Elastic Beanstalk. Kemudian dijelaskan tentang konfigurasi lingkungan ketersediaan tinggi yang menjalankan situs web Drupal untuk terhubung dengannya. Situs web menggunakan Amazon Elastic File System (Amazon EFS) sebagai penyimpanan bersama untuk file yang diunggah. Menjalankan instans DB eksternal ke Elastic Beanstalk memisahkan database dari siklus hidup lingkungan Anda, dan memungkinkan Anda terhubung ke database yang sama dari beberapa lingkungan, menukar satu database dengan yang lain, atau melakukan penerapan tanpa memengaruhi database Anda. blue/green

Bagian-bagian

- Prasyarat
- Meluncurkan instans DB pada Amazon RDS
- Meluncurkan lingkungan Elastic Beanstalk
- Mengonfigurasi pengaturan keamanan dan properti lingkungan

- Mengonfigurasi dan men-deploy aplikasi Anda
- Menginstal Drupal
- Perbarui konfigurasi Drupal dan hapus pembatasan akses
- Mengonfigurasi grup Auto Scaling
- Pembersihan
- Langkah selanjutnya

Prasyarat

Tutorial ini mengasumsikan Anda memiliki pengetahuan tentang operasi Elastic Beanstalk dasar dan konsol Elastic Beanstalk. Jika belum, ikuti petunjuk di Pelajari cara memulai dengan Elastic Beanstalk untuk meluncurkan lingkungan Elastic Beanstalk pertama Anda.

Untuk mengikuti prosedur di panduan ini, Anda memerlukan terminal atau shell baris perintah untuk menjalankan perintah. Perintah ditampilkan dalam daftar yang diawali dengan simbol prompt ($) dan nama direktori saat ini, bila perlu.

```
~/eb-project$ this is a command
this is output
```

Di Linux dan macOS, Anda dapat menggunakan shell dan manajer paket pilihan Anda. Pada Windows Anda dapat menginstal Windows Subsystem untuk Linux untuk mendapatkan versi Windows terintegrasi dari Ubuntu dan Bash.

Prosedur dalam tutorial ini untuk tugas Amazon Relational Database Service (Amazon RDS) yang menganggap Anda sedang meluncurkan sumber daya di Amazon Virtual Private Cloud (Amazon VPC) default. Semua akun baru menyertakan VPC default di setiap wilayah. Jika Anda tidak memiliki VPC default, prosedurnya akan bervariasi. Lihat petunjuk Menggunakan Elastic Beanstalk dengan Amazon RDS untuk platform EC2 -Classic dan VPC kustom.

Aplikasi sampel menggunakan Amazon EFS. Ini hanya berfungsi di AWS Wilayah yang mendukung Amazon EFS. Untuk mempelajari tentang AWS Wilayah pendukung, lihat Titik Akhir dan Kuota Amazon Elastic File System di bagian. Referensi Umum AWS

Jika platform lingkungan Elastic Beanstalk Anda menggunakan PHP 7.4 atau sebelumnya, kami sarankan Anda menggunakan Drupal versi 8.9.13 untuk tutorial ini. Untuk platform yang diinstal dengan PHP 8.0 atau yang lebih baru, kami sarankan Anda menggunakan Drupal 9.1.5.

Untuk informasi lebih lanjut tentang rilis Drupal dan versi PHP yang didukung, lihat Persyaratan PHP di situs Drupal. Versi inti yang direkomendasikan Drupal tercantum di situs web https://www.drupal.org/project/drupal.

Meluncurkan instans DB pada Amazon RDS

Untuk menggunakan basis data eksternal dengan aplikasi yang berjalan di Elastic Beanstalk, hal pertama yang dilakukan adalah meluncurkan instans DB dengan Amazon RDS. Ketika Anda meluncurkan instans dengan Amazon RDS, instans sepenuhnya tidak akan bergantung kepada Elastic Beanstalk dan lingkungan Elastic Beanstalk Anda, dan tidak akan diakhiri atau dipantau oleh Elastic Beanstalk.

Gunakan konsol Amazon RDS untuk meluncurkan instans DB MySQL Multi-AZ. Memilih deployment Multi-AZ memastikan bahwa basis data Anda akan failover dan terus tersedia jika instans DB sumber keluar dari layanan.

Untuk meluncurkan instans DB RDS pada VPC default

1. Buka konsol RDS.
2. Di panel navigasi, pilih Basis Data.
3. Pilih Buat basis data.
4. Pilih Pembuatan Standar.

> **Important**
>
> Jangan pilih Pembuatan Mudah. Jika Anda memilihnya, Anda tidak dapat mengonfigurasi pengaturan yang diperlukan untuk meluncurkan RDS DB ini.

5. Dalam Konfigurasi tambahan, untuk Nama basis data awal, ketik **ebdb**.
6. Tinjau pengaturan default dan sesuaikan pengaturan ini sesuai dengan kebutuhan spesifik Anda. Perhatikan opsi berikut:

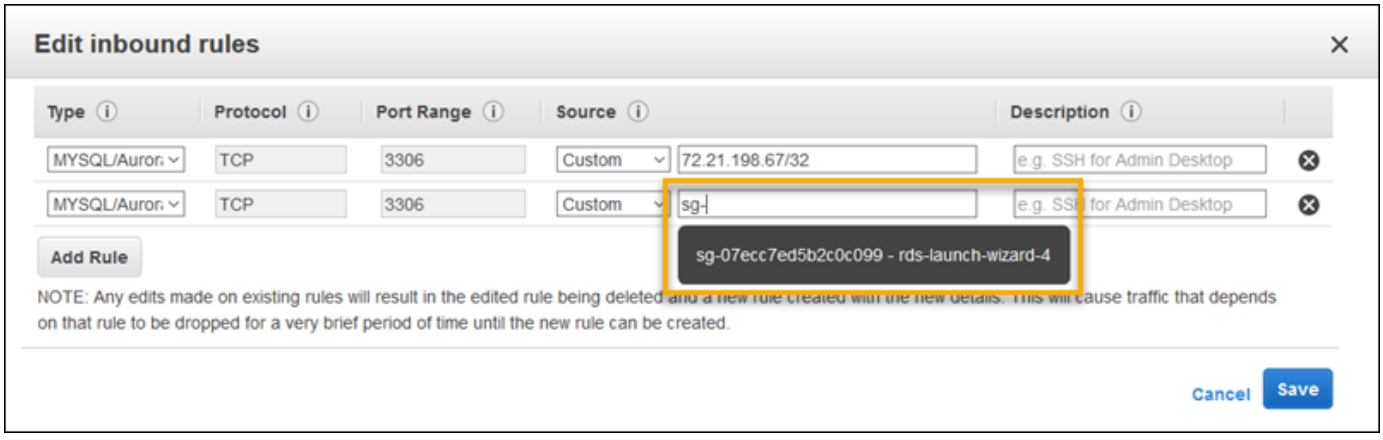
   - Kelas instans DB – Memilih ukuran instans yang memiliki jumlah memori dan daya CPU yang sesuai dengan beban kerja Anda.
   - Penerapan multi-AZ — Untuk ketersediaan tinggi, atur ini ke Buat Replica/Reader simpul Aurora di AZ yang berbeda.
   - Nama pengguna utama dan Kata sandi utama – Nama pengguna dan kata sandi basis data. Catat pengaturan ini karena Anda akan menggunakannya nanti.

7. Verifikasi pengaturan default untuk opsi lainnya, dan kemudian pilih Buat basis data.

Berikutnya, modifikasi grup keamanan yang dilampirkan ke instans DB Anda untuk memperbolehkan lintas masuk pada port yang sesuai. Modifikasi grup keamanan adalah grup keamanan yang sama yang akan Anda lampirkan ke lingkungan Elastic Beanstalk Anda kelak, sehingga aturan yang ditambahkan akan memberikan izin masuk ke sumber daya lain dalam grup keamanan yang sama.

Untuk mengubah aturan masuk pada grup keamanan yang dilampirkan ke instans RDS Anda

1. Buka konsol Amazon RDS.
2. Pilih Basis data.
3. Pilih nama instans DB Anda untuk menampilkan detailnya.
4. Di bagian Konektivitas, catat Subnet, grup Keamanan, dan Titik Akhir yang ditampilkan di halaman ini. Ini agar Anda dapat menggunakan informasi ini nanti.
5. Di bawah Keamanan, Anda dapat melihat grup keamanan yang terkait dengan instans DB. Buka tautan untuk melihat grup keamanan di EC2 konsol Amazon.
6. Pada detail grup keamanan, pilih Masuk.
7. Pilih Edit.
8. Pilih Tambahkan Aturan.
9. Untuk Jenis, pilih mesin DB yang digunakan aplikasi Anda.
10. Untuk Sumber, ketik **`sg-`** untuk melihat daftar grup keamanan yang tersedia. Pilih grup keamanan yang terkait dengan grup Auto Scaling yang digunakan dengan lingkungan Elastic Beanstalk Anda. Ini agar EC2 instans Amazon di lingkungan dapat memiliki akses ke database.



11. Pilih Simpan.

Membuat instans DB memakan waktu sekitar 10 menit. Sementara itu, luncurkan lingkungan Elastic Beanstalk Anda.

Meluncurkan lingkungan Elastic Beanstalk

Gunakan konsol Elastic Beanstalk untuk membuat lingkungan Elastic Beanstalk. Pilih platform PHP lalu terima pengaturan default dan kode sampel. Setelah Anda meluncurkan lingkungan, Anda dapat mengonfigurasi lingkungan untuk terhubung ke basis data, kemudian men-deploy kode Drupal ke lingkungan.

Untuk meluncurkan lingkungan (konsol)

1. Buka konsol Elastic Beanstalk menggunakan tautan yang telah dikonfigurasi sebelumnya: console.aws.amazon. com/elasticbeanstalk/home#/newApplication?applicationName=Tutorial&EnvironmentType= LoadBalanced
2. Untuk Platform, pilih platform dan cabang platform yang sesuai dengan bahasa yang digunakan oleh aplikasi Anda.
3. Untuk Kode aplikasi, pilih Aplikasi sampel.
4. Pilih Tinjau dan Luncurkan.
5. Tinjau opsi yang tersedia. Pilih opsi tersedia yang ingin Anda gunakan, dan saat Anda siap, pilih Buat aplikasi.

Pembuatan lingkungan membutuhkan waktu sekitar 5 menit dan membuat sumber daya berikut:

- EC2 instance — Mesin virtual Amazon Elastic Compute Cloud (Amazon EC2) yang dikonfigurasi untuk menjalankan aplikasi web pada platform yang Anda pilih.

  Setiap platform menjalankan satu set perangkat lunak, file konfigurasi dan penulisan tertentu untuk mendukung versi bahasa, kerangka kerja, kontainer web tertentu, atau kombinasi dari semua ini. Sebagian besar platform menggunakan Apache atau NGINX sebagai proksi terbalik yang ada di depan aplikasi web Anda, meneruskan permintaan ke aplikasi web, menyajikan aset statis, dan menghasilkan log akses dan kesalahan.
- Grup keamanan instans — Grup EC2 keamanan Amazon yang dikonfigurasi untuk memungkinkan lalu lintas masuk pada port 80. Sumber daya ini memungkinkan lalu lintas HTTP dari penyeimbang beban mencapai EC2 instance yang menjalankan aplikasi web Anda. Secara default, lalu lintas tidak diizinkan di port lain.

- Penyeimbang beban – Penyeimbang beban Elastic Load Balancing yang dikonfigurasi untuk mendistribusikan permintaan ke instans yang menjalankan aplikasi Anda. Penyeimbang beban juga menghilangkan kebutuhan untuk mengekspos instans Anda langsung ke internet.
- Grup keamanan penyeimbang beban - Grup EC2 keamanan Amazon yang dikonfigurasi untuk memungkinkan lalu lintas masuk di port 80. Sumber daya ini memungkinkan lalu lintas HTTP dari internet mencapai penyeimbang beban. Secara default, lalu lintas tidak diizinkan di port lain.
- Grup Auto Scaling – Grup Auto Scaling yang dikonfigurasi untuk menggantikan instans jika diakhiri atau menjadi tidak tersedia.
- Bucket Amazon S3 – Lokasi penyimpanan untuk kode sumber, log, dan artifact lainnya yang dibuat saat Anda menggunakan Elastic Beanstalk.
- CloudWatch Alarm Amazon — Dua CloudWatch alarm yang memantau beban pada instans di lingkungan Anda dan yang dipicu jika beban terlalu tinggi atau terlalu rendah. Saat alarm terpicu, grup Auto Scaling Anda akan menaikkan atau menurunkan skala sebagai respons.
- AWS CloudFormation stack - Elastic AWS CloudFormation Beanstalk digunakan untuk meluncurkan sumber daya di lingkungan Anda dan menyebarkan perubahan konfigurasi. Sumber daya ditentukan di sebuah templat yang dapat Anda lihat di Konsol AWS CloudFormation.
- Nama domain — Nama domain yang merutekan ke aplikasi web Anda dalam formulir *`subdomain`*.*`region`*.elasticbeanstalk.com.

> **Keamanan domain**
>
> Untuk meningkatkan keamanan aplikasi Elastic Beanstalk Anda, domain elasticbeanstalk.com terdaftar di Daftar Akhiran Publik (PSL).
> Jika Anda perlu mengatur cookie sensitif di nama domain default untuk aplikasi Elastic Beanstalk Anda, kami sarankan Anda menggunakan cookie `__Host-` dengan awalan untuk meningkatkan keamanan. Praktik ini melindungi domain Anda dari upaya pemalsuan permintaan lintas situs (CSRF). Untuk informasi selengkapnya, lihat halaman Set-Cookie di Jaringan Pengembang Mozilla.

Semua sumber daya ini dikelola oleh Elastic Beanstalk. Ketika Anda mengakhiri lingkungan, Elastic Beanstalk mengakhiri semua sumber daya yang dimuatnya. Instans DB RDS yang diluncurkan berada di luar lingkungan Anda, sehingga Anda bertanggung jawab untuk mengelola siklus hidupnya.

> **Note**
>
> Bucket Amazon S3 yang dibuat Elastic Beanstalk dibagi antar lingkungan dan tidak dihapus selama pengakhiran lingkungan. Untuk informasi selengkapnya, lihat Menggunakan Elastic Beanstalk dengan Amazon S3.

Mengonfigurasi pengaturan keamanan dan properti lingkungan

Menambahkan grup keamanan instans DB Anda ke lingkungan Anda yang sedang berjalan. Prosedur ini menyebabkan Elastic Beanstalk menyediakan kembali instans di lingkungan Anda dengan grup keamanan tambahan terlampir.

Untuk menambahkan grup keamanan ke lingkungan Anda

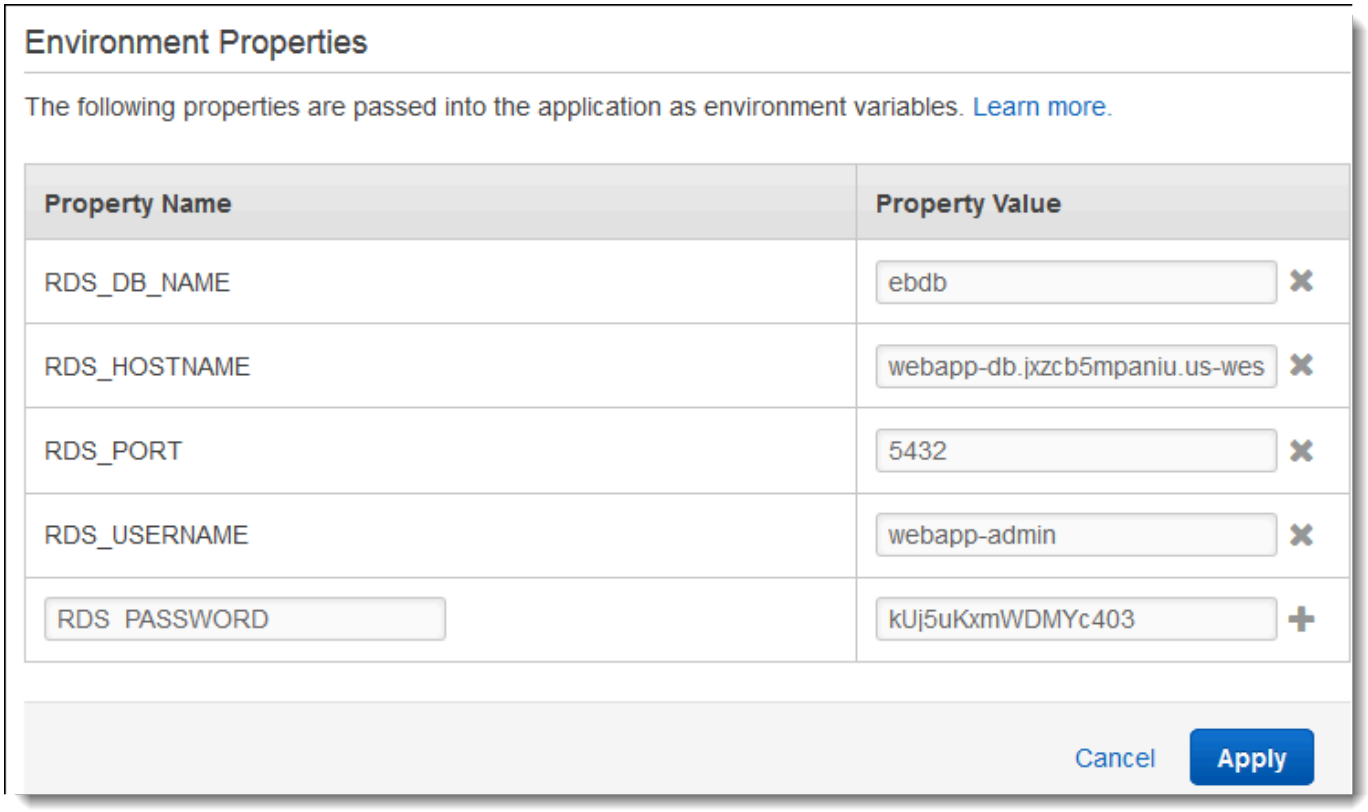
- Lakukan salah satu dari berikut ini:
  - Untuk menambahkan grup keamanan menggunakan konsol Elastic Beanstalk
    a. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
    b. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
    c. Pada panel navigasi, pilih Konfigurasi.
    d. Pada kategori konfigurasi Instans, pilih Edit.
    e. Di bawah grup EC2 keamanan, pilih grup keamanan untuk dilampirkan ke instance, selain grup keamanan instans yang dibuat Elastic Beanstalk.
    f. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.
    g. Baca peringatan, kemudian pilih Konfirmasi.
  - Untuk menambahkan grup keamanan menggunakan file konfigurasi, gunakan file `securitygroup-addexisting.config`contoh.

Selanjutnya, gunakan properti lingkungan untuk meneruskan informasi koneksi ke lingkungan Anda. Aplikasi sampel menggunakan set properti default yang cocok dengan properti yang dikonfigurasi Elastic Beanstalk saat Anda menyediakan basis data di lingkungan Anda.

Untuk mengonfigurasi properti lingkungan bagi instans DB Amazon RDS

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Dalam kategori konfigurasi Pembaruan, pemantauan, dan pencatatan, pilih Edit.
5. Pada bagian Properti lingkungan, tentukan variabel yang dibaca aplikasi Anda untuk membangun string koneksi. Untuk kompatibilitas dengan lingkungan yang memiliki instans DB RDS terintegrasi, gunakan nama dan nilai-nilai berikut. Anda dapat menemukan semua nilai, kecuali untuk kata sandi Anda, di Konsol RDS.

Nama properti	Deskripsi	Nilai properti
`RDS_HOSTNAME`	Nama host instans DB.	Di tab Konektivitas & keamanan di konsol Amazon RDS: Titik akhir.
`RDS_PORT`	Port tempat instans DB menerima koneksi. Nilai default bervariasi di antara mesin DB.	Di tab Konektivitas & keamanan di konsol Amazon RDS: Port.
`RDS_DB_NAME`	Nama basis data, **ebdb**.	Di tab Konfigurasi di konsol Amazon RDS: Nama DB.
`RDS_USERNAME`	Nama pengguna yang Anda konfigurasi untuk basis data Anda.	Di tab Konfigurasi di konsol Amazon RDS: Nama pengguna utama.
`RDS_PASSWORD`	Kata sandi yang Anda konfigurasi untuk basis data Anda.	Tidak tersedia untuk referensi di konsol Amazon RDS.



6. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Setelah menginstal Drupal, Anda perlu terhubung ke instans dengan menggunakan SSH untuk menerima kembali beberapa detail konfigurasi. Menetapkan kunci SSH ke instans lingkungan Anda.

Untuk mengonfigurasi SSH

1. Jika sebelumnya Anda belum membuat key pair, buka halaman pasangan kunci EC2 konsol Amazon dan ikuti petunjuk untuk membuatnya.
2. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
3. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
4. Pada panel navigasi, pilih Konfigurasi.
5. Dalam Keamanan, pilih Edit.
6. Untuk EC2 key pair, pilih key pair Anda.
7. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Mengonfigurasi dan men-deploy aplikasi Anda

Untuk membuat proyek Drupal untuk Elastic Beanstalk, unduh kode sumber Drupal dan gabungkan dengan file di aws-samples/repositori aktif. eb-php-drupal GitHub

Untuk membuat proyek Drupal

1. Jalankan perintah berikut untuk mengunduh Drupal dari www.drupal.org/download. Untuk mempelajari selengkapnya tentang unduhan, lihat situs web Drupal.

   Jika platform lingkungan Elastic Beanstalk Anda menggunakan PHP 7.4 atau sebelumnya, kami sarankan Anda mengunduh Drupal versi 8.9.13 untuk tutorial ini. Anda dapat menjalankan perintah berikut untuk mengunduhnya.

   ```
   ~$ curl https://ftp.drupal.org/files/projects/drupal-8.9.13.tar.gz -o drupal.tar.gz
   ```

   Jika platform Anda menggunakan PHP 8.0 atau yang lebih baru, kami sarankan Anda mengunduh Drupal 9.1.5. Anda bisa menggunakan perintah ini untuk mengunduhnya.

   ```
   ~$ curl https://ftp.drupal.org/files/projects/drupal-9.1.5.tar.gz -o drupal.tar.gz
   ```

   Untuk informasi selengkapnya tentang rilis Drupal dan versi PHP yang mereka dukung, lihat Persyaratan PHP dalam dokumentasi resmi Drupal. Versi inti yang disarankan Drupal tercantum dalam situs web Drupal.
2. Gunakan perintah berikut untuk mengunduh file konfigurasi dari repositori sampel:

   ```
   ~$ wget https://github.com/aws-samples/eb-php-drupal/releases/download/v1.1/eb-php-drupal-v1.zip
   ```
3. Mengekstraksi Drupal dan ubah nama folder.

   Jika Anda mengunduh Drupal 8.9.13:

   ```
   ~$ tar -xvf drupal.tar.gz
   ~$ mv drupal-8.9.13 drupal-beanstalk
   ~$ cd drupal-beanstalk
   ```

   Jika Anda mengunduh Drupal 9.1.5:

```
~$ tar -xvf drupal.tar.gz
~$ mv drupal-9.1.5 drupal-beanstalk
~$ cd drupal-beanstalk
```

4. Mengekstraksi file konfigurasi melalui instalasi Drupal.

```
~/drupal-beanstalk$ unzip ../eb-php-drupal-v1.zip
 creating: .ebextensions/
 inflating: .ebextensions/dev.config
 inflating: .ebextensions/drupal.config
 inflating: .ebextensions/efs-create.config
 inflating: .ebextensions/efs-filesystem.template
 inflating: .ebextensions/efs-mount.config
 inflating: .ebextensions/loadbalancer-sg.config
 inflating: LICENSE
 inflating: README.md
 inflating: beanstalk-settings.php
```

Verifikasi bahwa struktur folder `drupal-beanstalk` Anda benar, seperti yang ditunjukkan.

```
drupal-beanstalk$ tree -aL 1
.
### autoload.php
### beanstalk-settings.php
### composer.json
### composer.lock
### core
### .csslintrc
### .ebextensions
### .ebextensions
### .editorconfig
### .eslintignore
### .eslintrc.json
### example.gitignore
### .gitattributes
### .htaccess
### .ht.router.php
### index.php
### LICENSE
### LICENSE.txt
### modules
```

```
### profiles
### README.md
### README.txt
### robots.txt
### sites
### themes
### update.php
### vendor
### web.config
```

File `beanstalk-settings.php` dari repo proyek menggunakan variabel lingkungan yang Anda tetapkan di langkah sebelumnya untuk mengonfigurasi koneksi basis data. Folder `.ebextensions` berisi file konfigurasi yang membuat sumber daya tambahan dalam lingkungan Elastic Beanstalk Anda.

File konfigurasi memerlukan modifikasi untuk dapat bekerja dengan akun Anda. Ganti nilai placeholder dalam file dengan yang sesuai IDs dan buat bundel sumber.

Untuk memperbarui file konfigurasi dan membuat paket sumber.

1. Memodifikasi file konfigurasi sebagai berikut.

   - `.ebextensions/dev.config` – membatasi akses ke lingkungan Anda ke alamat IP Anda untuk melindunginya selama proses instalasi Drupal. Ganti alamat IP placeholder dekat bagian atas file dengan alamat IP publik Anda.
   - `.ebextensions/efs-create.config` – membuat sistem file EFS dan titik pemasangan di setiap Availability Zone / subnet di VPC Anda. Identifikasi VPC dan subnet default Anda IDs di konsol VPC Amazon.

2. Membuat paket sumber berisi file dalam folder proyek Anda. Perintah berikut membuat paket sumber yang bernama `drupal-beanstalk.zip`. Paket sumber tidak termasuk file dalam folder `vendor`, yang memerlukan banyak ruang dan tidak perlu men-deploy aplikasi Anda ke Elastic Beanstalk.

   ```
   ~/eb-drupal$ zip ../drupal-beanstalk.zip -r * .[^.]* -x "vendor/*"
   ```

Unggah paket sumber ke Elastic Beanstalk untuk men-deploy Drupal ke lingkungan Anda.

Untuk men-deploy paket sumber

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Di halaman gambaran umum lingkungan, pilih Unggah dan deploy.
4. Gunakan kotak dialog di layar untuk mengunggah paket sumber.
5. Pilih Deploy.
6. Ketika deployment selesai, Anda dapat memilih URL situs untuk membuka situs web Anda di tab baru.

Menginstal Drupal

Untuk menyelesaikan instalasi Drupal Anda

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pilih URL lingkungan untuk membuka situs Anda di peramban. Anda dialihkan ke wizard instalasi Drupal karena situs belum dikonfigurasikan.
4. Lakukan instalasi standar dengan pengaturan berikut untuk basis data:

   - Nama basis data – Nama DB yang ditunjukkan di konsol Amazon RDS.
   - Nama pengguna dan kata sandi basis data – Nilai Nama Pengguna Utama dan Kata Sandi Utama yang Anda masukkan saat membuat basis data Anda.
   - Opsi Lanjutan > Host – Titik akhir instans DB yang ditunjukkan di konsol Amazon RDS.

Instalasi memakan waktu sekitar satu menit.

Perbarui konfigurasi Drupal dan hapus pembatasan akses

Proses instalasi Drupal yang membuat sebuah file bernama `settings.php` dalam folder `sites/default` pada instans. Anda memerlukan file ini dalam kode sumber Anda untuk menghindari pengaturan ulang situs pada deployment berikutnya, kecuali file saat ini berisi rahasia yang tidak ingin Anda serahkan ke sumber. Connect ke instans aplikasi untuk mendapatkan kembali informasi dari file pengaturan.

Untuk terhubung ke instans aplikasi Anda dengan SSH

1. Buka halaman instans EC2 konsol Amazon.
2. Memilih instans aplikasi. Namanya diambil dari nama lingkungan Elastic Beanstalk Anda.
3. Pilih Connect.
4. Ikuti petunjuk untuk menghubungkan instans dengan SSH. Perintahnya terlihat serupa dengan yang berikut.

```
$ ssh -i ~/.ssh/mykey ec2-user@ec2-00-55-33-222.us-west-2.compute.amazonaws.com
```

Mendapatkan id direktori sinkronisasi dari baris terakhir file pengaturan.

```
[ec2-user ~]$ tail -n 1 /var/app/current/sites/default/settings.php
$config_directories['sync'] = 'sites/default/files/
config_4ccfX2sPQm79p1mk5IbUq9S_FokcENO4mxyC-L18-4g_xKj_7j9ydn31kDOYOgnzMu071Tvc4Q/
sync';
```

File ini juga berisi kunci hash situs saat ini, tetapi Anda dapat mengabaikan nilainya saat ini dan menggunakan milik Anda sendiri.

Tetapkan jalur direktori sinkronisasi dan kunci hash ke properti lingkungan. File pengaturan yang dikustomisasi dari repo proyek membaca properti ini untuk mengonfigurasi situs selama deployment, selain properti koneksi basis data yang Anda tetapkan sebelumnya.

Properti konfigurasi Drupal

- `SYNC_DIR` – Jalur ke direktori sinkronisasi.
- `HASH_SALT` – Setiap nilai string yang memenuhi persyaratan properti lingkungan.

Untuk mengonfigurasi variabel lingkungan di konsol Elastic Beanstalk

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Dalam kategori konfigurasi Pembaruan, pemantauan, dan pencatatan, pilih Edit.

5. Gulir ke bawah ke variabel lingkungan Runtime.
6. Pilih Tambahkan variabel lingkungan.
7. Untuk Sumber pilih Teks biasa.

> **Note**
>
> Nilai Secrets Manager dan SSM Parameter Store di drop-down adalah untuk mengonfigurasi variabel lingkungan sebagai rahasia untuk menyimpan data sensitif, seperti kredensyal dan kunci API. Untuk informasi selengkapnya, lihat Menggunakan Elastic Beanstalk dengan dan AWS Secrets Manager Parameter Store AWS Systems Manager.

8. Masukkan nama variabel Lingkungan dan pasangan nilai variabel Lingkungan.
9. Jika Anda perlu menambahkan lebih banyak variabel ulangi Langkah 6 hingga Langkah 8.
10. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Akhirnya, proyek sampel menyertakan file konfigurasi (`loadbalancer-sg.config`) yang membuat sebuah grup keamanan dan menetapkan ke penyeimbang beban lingkungan, menggunakan alamat IP yang dikonfigurasi di `dev.config` untuk membatasi akses HTTP pada port 80 ke sambungan dari jaringan Anda. Jika tidak, pihak luar bisa berpotensi terhubung ke situs Anda sebelum Anda telah menginstal Drupal dan mengonfigurasikan akun admin Anda.

Untuk memperbarui konfigurasi Drupal dan menghapus pembatasan akses

1. Hapus file `.ebextensions/loadbalancer-sg.config` dari direktori proyek Anda.

```
~/drupal-beanstalk$ rm .ebextensions/loadbalancer-sg.config
```

2. Salin file `settings.php` yang dikustomisasi ke dalam folder situs.

```
~/drupal-beanstalk$ cp beanstalk-settings.php sites/default/settings.php
```

3. Buat paket sumber.

```
~/eb-drupal$ zip ../drupal-beanstalk-v2.zip -r * .[^.]* -x "vendor/*"
```

Unggah paket sumber ke Elastic Beanstalk untuk men-deploy Drupal ke lingkungan Anda.

Untuk men-deploy paket sumber

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Di halaman gambaran umum lingkungan, pilih Unggah dan deploy.
4. Gunakan kotak dialog di layar untuk mengunggah paket sumber.
5. Pilih Deploy.
6. Ketika deployment selesai, Anda dapat memilih URL situs untuk membuka situs web Anda di tab baru.

Mengonfigurasi grup Auto Scaling

Terakhir, konfigurasikan grup Auto Scaling lingkungan Anda dengan jumlah instans minimum yang lebih tinggi. Jalankan setidaknya dua instans setiap saat untuk mencegah terjadinya kegagalan di satu titik server web di lingkungan Anda, dan mengizinkan Anda untuk men-deploy perubahan tanpa membuat situs Anda keluar dari layanan.

Mengonfigurasi grup Auto Scaling lingkungan Anda untuk ketersediaan yang tinggi

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Pada kategori konfigurasi Kapasitas, pilih Edit.
5. Pada bagian Grup Auto Scaling, set Instans minimum ke **2**.
6. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Untuk mendukung unggahan konten di beberapa instans, proyek sampel menggunakan Amazon Elastic File System untuk membuat sistem file bersama. Buat kiriman di situs dan unggah konten untuk menyimpannya di sistem file bersama. Lihat kiriman dan refresh halaman beberapa kali untuk temuan kedua instans dan verifikasi apakah sistem file bersama tersebut bekerja.

Pembersihan

Setelah Anda selesai bekerja dengan kode demo, Anda dapat mengakhiri lingkungan Anda. Elastic Beanstalk menghapus semua sumber daya AWS terkait, seperti instans EC2 Amazon, instans database, penyeimbang beban, grup keamanan,dan alarm.

Menghapus sumber daya tidak menghapus aplikasi Elastic Beanstalk, sehingga Anda dapat membuat lingkungan baru untuk aplikasi Anda kapan saja.

Untuk mengakhiri lingkungan Elastic Beanstalk Anda dari konsol

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pilih Tindakan, lalu pilih Terminate environment.
4. Gunakan kotak dialog di layar untuk mengonfirmasi pengakhiran lingkungan.

Selain itu, Anda dapat mengakhiri sumber daya basis data yang Anda buat di luar lingkungan Elastic Beanstalk Anda. Ketika Anda mengakhiri instans DB Amazon RDS, Anda dapat mengambil snapshot dan memulihkan data ke instans lain kelak.

Untuk mengakhiri instans DB RDS Anda

1. Buka konsol Amazon RDS.
2. Pilih Basis data.
3. Pilih instans DB Anda.
4. Pilih Tindakan, dan lalu pilih Hapus.
5. Pilih apakah akan membuat snapshot, dan kemudian memilih Hapus.

Langkah selanjutnya

Ketika aplikasi terus dikembangkan, Anda mungkin akan menginginkan sebuah cara untuk mengelola lingkungan dan men-deploy aplikasi Anda tanpa membuat file .zip secara manual dan mengunggahnya ke konsol Elastic Beanstalk. Elastic Beanstalk Command Line Interface (EB CLI easy-to-use) menyediakan perintah untuk membuat, mengkonfigurasi, dan menyebarkan aplikasi ke lingkungan Elastic Beanstalk dari baris perintah.

Aplikasi sampel menggunakan file konfigurasi untuk mengonfigurasi pengaturan PHP dan membuat tabel dalam basis data jika tidak ada. Anda juga dapat menggunakan file konfigurasi untuk mengonfigurasi pengaturan grup keamanan instans Anda selama pembuatan lingkungan untuk menghindari pembaruan konfigurasi yang cukup memakan waktu. Lihat Penyesuaian lingkungan lanjutan dengan file konfigurasi (`.ebextensions`) untuk informasi selengkapnya.

Untuk pengembangan dan pengujian, Anda mungkin ingin menggunakan fungsionalitas Elastic Beanstalk dalam menambahkan instans DB terkelola langsung ke lingkungan Anda. Terkait petunjuk tentang pengaturan basis data di dalam lingkungan Anda, lihat Menambahkan basis data ke lingkungan Elastic Beanstalk Anda.

Jika Anda membutuhkan basis data berperforma tinggi, pertimbangkan untuk menggunakan Amazon Aurora. Amazon Aurora merupakan mesin basis data kompatibel dengan MySQL yang menawarkan fitur basis data komersial dengan biaya rendah. Untuk menghubungkan aplikasi Anda ke basis data yang berbeda, ulangi langkah konfigurasi grup keamanan dan perbarui properti lingkungan terkait RDS.

Terakhir, jika Anda berencana menggunakan aplikasi dalam lingkungan produksi, Anda akan ingin mengonfigurasikan nama domain khusus untuk lingkungan Anda dan mengaktifkan HTTPS untuk koneksi yang aman.

Menyebarkan aplikasi Python dengan Elastic Beanstalk

Bab ini memberikan instruksi untuk mengkonfigurasi dan menyebarkan aplikasi web Python Anda ke. AWS Elastic Beanstalk Elastic Beanstalk memudahkan penerapan, pengelolaan, dan skala aplikasi web Python Anda menggunakan Amazon Web Services.

Anda dapat menerapkan aplikasi Anda hanya dalam beberapa menit menggunakan Elastic Beanstalk Command Line Interface (EB CLI) atau dengan menggunakan konsol Elastic Beanstalk. Setelah Anda menerapkan aplikasi Elastic Beanstalk Anda, Anda dapat terus menggunakan EB CLI untuk mengelola aplikasi dan lingkungan Anda, atau Anda dapat menggunakan konsol Elastic Beanstalk,, atau. AWS CLI APIs

Ikuti langkah-langkah dalam QuickStart untuk Python untuk step-by-step instruksi untuk membuat dan menyebarkan aplikasi web Python Hello World dengan EB CLI.

Topik

- QuickStart: Menyebarkan aplikasi Python ke Elastic Beanstalk
- Menyiapkan lingkungan pengembangan Python Anda untuk Elastic Beanstalk
- Menggunakan platform Python Elastic Beanstalk
- Men-deploy aplikasi Flask ke Elastic Beanstalk
- Men-deploy aplikasi Django ke Elastic Beanstalk
- Menambahkan instans Amazon RDS DB ke lingkungan Python Elastic Beanstalk Anda
- Alat Python dan sumber daya

QuickStart: Menyebarkan aplikasi Python ke Elastic Beanstalk

QuickStart Tutorial ini memandu Anda melalui proses pembuatan aplikasi Python dan menyebarkannya ke lingkungan. AWS Elastic Beanstalk

> **Note**
>
> Contoh tutorial dimaksudkan untuk demonstrasi. Jangan gunakan aplikasi untuk lalu lintas produksi.

Bagian-bagian

- AWS Akun Anda
- Prasyarat
- Langkah 1: Buat aplikasi Python
- Langkah 2: Jalankan aplikasi Anda secara lokal
- Langkah 3: Menyebarkan aplikasi Python Anda dengan EB CLI
- Langkah 4: Jalankan aplikasi Anda di Elastic Beanstalk
- Langkah 5: Bersihkan
- AWS sumber daya untuk aplikasi Anda
- Langkah selanjutnya
- Terapkan dengan konsol Elastic Beanstalk

AWS Akun Anda

Jika Anda belum menjadi AWS pelanggan, Anda perlu membuat AWS akun. Mendaftar memungkinkan Anda mengakses Elastic Beanstalk AWS dan layanan lain yang Anda butuhkan.

Jika Anda sudah memiliki AWS akun, Anda dapat melanjutkan kePrasyarat.

Buat AWS akun

Mendaftar untuk Akun AWS

Jika Anda tidak memiliki Akun AWS, selesaikan langkah-langkah berikut untuk membuatnya.

Untuk mendaftar untuk Akun AWS

1. Buka https://portal.aws.amazon.com/billing/pendaftaran.
2. Ikuti petunjuk online.

   Bagian dari prosedur pendaftaran melibatkan menerima panggilan telepon atau pesan teks dan memasukkan kode verifikasi pada keypad telepon.

   Saat Anda mendaftar untuk sebuah Akun AWS, sebuah Pengguna root akun AWSdibuat. Pengguna root memiliki akses ke semua Layanan AWS dan sumber daya di akun. Sebagai praktik keamanan terbaik, tetapkan akses administratif ke pengguna, dan gunakan hanya pengguna root untuk melakukan tugas yang memerlukan akses pengguna root.

AWS mengirimi Anda email konfirmasi setelah proses pendaftaran selesai. Kapan saja, Anda dapat melihat aktivitas akun Anda saat ini dan mengelola akun Anda dengan masuk https://aws.amazon.comke/ dan memilih Akun Saya.

Buat pengguna dengan akses administratif

Setelah Anda mendaftar Akun AWS, amankan Pengguna root akun AWS, aktifkan AWS IAM Identity Center, dan buat pengguna administratif sehingga Anda tidak menggunakan pengguna root untuk tugas sehari-hari.

Amankan Anda Pengguna root akun AWS

1. Masuk ke AWS Management Consolesebagai pemilik akun dengan memilih pengguna Root dan memasukkan alamat Akun AWS email Anda. Di laman berikutnya, masukkan kata sandi.

   Untuk bantuan masuk dengan menggunakan pengguna root, lihat Masuk sebagai pengguna root di AWS Sign-In Panduan Pengguna.
2. Mengaktifkan autentikasi multi-faktor (MFA) untuk pengguna root Anda.

   Untuk petunjuk, lihat Mengaktifkan perangkat MFA virtual untuk pengguna Akun AWS root (konsol) Anda di Panduan Pengguna IAM.

Buat pengguna dengan akses administratif

1. Aktifkan Pusat Identitas IAM.

   Untuk mendapatkan petunjuk, silakan lihat Mengaktifkan AWS IAM Identity Center di Panduan Pengguna AWS IAM Identity Center .
2. Di Pusat Identitas IAM, berikan akses administratif ke pengguna.

   Untuk tutorial tentang menggunakan Direktori Pusat Identitas IAM sebagai sumber identitas Anda, lihat Mengkonfigurasi akses pengguna dengan default Direktori Pusat Identitas IAM di Panduan AWS IAM Identity Center Pengguna.

Masuk sebagai pengguna dengan akses administratif

- Untuk masuk dengan pengguna Pusat Identitas IAM, gunakan URL masuk yang dikirim ke alamat email saat Anda membuat pengguna Pusat Identitas IAM.

Untuk bantuan masuk menggunakan pengguna Pusat Identitas IAM, lihat Masuk ke portal AWS akses di Panduan AWS Sign-In Pengguna.

Tetapkan akses ke pengguna tambahan

1. Di Pusat Identitas IAM, buat set izin yang mengikuti praktik terbaik menerapkan izin hak istimewa paling sedikit.

   Untuk petunjuknya, lihat Membuat set izin di Panduan AWS IAM Identity Center Pengguna.
2. Tetapkan pengguna ke grup, lalu tetapkan akses masuk tunggal ke grup.

   Untuk petunjuk, lihat Menambahkan grup di Panduan AWS IAM Identity Center Pengguna.

Prasyarat

Untuk mengikuti prosedur di panduan ini, Anda memerlukan terminal atau shell baris perintah untuk menjalankan perintah. Perintah ditampilkan dalam daftar yang diawali dengan simbol prompt ($) dan nama direktori saat ini, bila perlu.

```
~/eb-project$ this is a command
this is output
```

Di Linux dan macOS, Anda dapat menggunakan shell dan manajer paket pilihan Anda. Pada Windows Anda dapat menginstal Windows Subsystem untuk Linux untuk mendapatkan versi Windows terintegrasi dari Ubuntu dan Bash.

EB CLI

Tutorial ini menggunakan Elastic Beanstalk Command Line Interface (EB CLI). Untuk detail tentang pemasangan dan konfigurasi EB CLI, lihat Instal EB CLI dengan skrip pengaturan (disarankan) dan Mengonfigurasi EB CLI.

Kerangka kerja Python dan Flask

Konfirmasikan bahwa Anda memiliki versi Python yang berfungsi dengan `pip` diinstal dengan menjalankan perintah berikut.

```
~$ python3 --version
```

```
Python 3.N.N
>~$ python3 -m pip --version
pip X.Y.Z from ... (python 3.N.N)
```

Jika salah satu perintah sebelumnya mengembalikan “Python tidak ditemukan “, jalankan perintah berikut yang menggunakan `python` alih-alih. `python3` Pengaturan alias dan tautan simbolik dapat bervariasi menurut sistem operasi dan penyesuaian individual, sehingga `python3` perintah mungkin tidak berfungsi pada mesin Anda.

```
~$ python --version
Python 3.N.N
>~$ python -m pip --version
pip X.Y.Z from ... (python 3.N.N)
```

Jika Anda tidak menginstal Python di komputer lokal Anda, Anda dapat mengunduhnya dari halaman unduhan Python di situs web Python. Untuk daftar versi bahasa Python yang didukung oleh Elastic Beanstalk, lihat Platform Python yang didukung dalam panduan Platform.AWS Elastic Beanstalk Situs web unduhan Python menyediakan tautan ke Panduan Pengembang Python, tempat Anda akan menemukan instruksi instalasi dan penyiapan.

> **Note**
>
> `pip`Paket Python disertakan secara default dengan Python 3.4 atau yang lebih baru.

Jika output Anda menunjukkan bahwa Anda memiliki versi Python yang didukung, tetapi tidak`pip`, lihat halaman Instalasi di situs web pip.pypa.io. Ini memberikan panduan untuk menginstal pip dalam lingkungan Python yang tidak memilikinya.

Konfirmasikan apakah Flask diinstal dengan menjalankan perintah berikut:

```
~$ pip list | grep Flask
```

Jika Flask tidak diinstal, Anda dapat menginstalnya dengan perintah berikut:

```
~$ pip install Flask
```

Langkah 1: Buat aplikasi Python

Buat direktori proyek.

```
~$ mkdir eb-python
~$ cd eb-python
```

Buat sampel “Halo Elastic Beanstalk!” Aplikasi Python yang akan Anda gunakan menggunakan Elastic Beanstalk.

Buat file teks bernama `application.py` di direktori yang baru saja Anda buat dengan konten berikut.

Example **`~/eb-python/application.py`**

```
from flask import Flask
application = Flask(__name__)

@application.route('/')
def hello_elastic_beanstalk():
        return 'Hello Elastic Beanstalk!'
```

Buat file teks bernama `requirements.txt` dengan baris berikut. File ini berisi `pip` paket yang diperlukan untuk menjalankan aplikasi.

Example **`~/eb-python/requirements.txt`**

```
Flask
```

Langkah 2: Jalankan aplikasi Anda secara lokal

Jalankan perintah berikut untuk menjalankan aplikasi Anda secara lokal.

```
~/eb-python$ export FLASK_APP=application.py && flask run --port 5000
```

Anda akan melihat output yang mirip dengan berikut ini

```
Serving Flask app 'application.py'
Debug mode: off
```

```
WARNING: This is a development server. Do not use it in a production deployment. Use a
 production WSGI server instead.
Running on http://127.0.0.1:5000
Press CTRL+C to quit
127.0.0.1 - - [01/Jan/1970 00:00:00] "GET / HTTP/1.1" 200 -
```

Arahkan ke `http://localhost:5000` di browser web Anda. Browser web harus menampilkan "Hello Elastic Beanstalk!".

Langkah 3: Menyebarkan aplikasi Python Anda dengan EB CLI

Jalankan perintah berikut untuk membuat lingkungan Elastic Beanstalk untuk aplikasi ini.

Untuk membuat lingkungan dan menyebarkan aplikasi Python Anda

1. Inisialisasi repositori EB CLI Anda dengan perintah eb init.

   ```
   ~/eb-python$ eb init -p python-3.9 python-tutorial --region us-east-2
   ```

   Perintah ini membuat aplikasi bernama `python-tutorial` dan mengkonfigurasi repositori lokal Anda untuk membuat lingkungan dengan versi platform Python yang disediakan.
2. (Opsional) Jalankan eb init lagi untuk mengonfigurasi key pair default sehingga Anda dapat menggunakan SSH untuk terhubung ke EC2 instance yang menjalankan aplikasi Anda.

   ```
   ~/eb-python$ eb init
   Do you want to set up SSH for your instances?
   (y/n): y
   Select a keypair.
   1) my-keypair
   2) [ Create new KeyPair ]
   ```

   Pilih pasangan kunci jika Anda sudah memilikinya, atau ikuti prompt untuk membuatnya. Jika Anda tidak melihat prompt atau perlu mengubah pengaturan Anda nanti, jalankan eb init -i.
3. Buat lingkungan baru dan deploy aplikasi Anda dengan eb create. Elastic Beanstalk secara otomatis membuat file zip untuk aplikasi Anda dan memulainya pada port 5000.

   ```
   ~/eb-python$ eb create python-env
   ```

   Dibutuhkan sekitar lima menit untuk Elastic Beanstalk untuk menciptakan lingkungan Anda.

Langkah 4: Jalankan aplikasi Anda di Elastic Beanstalk

Ketika proses untuk membuat lingkungan Anda selesai, buka situs web Anda denganeb open.

```
~/eb-python$ eb open
```

Selamat! Anda telah menerapkan aplikasi Python dengan Elastic Beanstalk! Ini akan membuka jendela peramban menggunakan nama domain yang dibuat untuk aplikasi Anda.

Langkah 5: Bersihkan

Anda dapat menghentikan lingkungan Anda ketika Anda selesai bekerja dengan aplikasi Anda. Elastic Beanstalk AWS mengakhiri semua sumber daya yang terkait dengan lingkungan Anda.

Untuk mengakhiri lingkungan Elastic Beanstalk Anda dengan EB CLI jalankan perintah berikut.

```
~/eb-python$ eb terminate
```

AWS sumber daya untuk aplikasi Anda

Anda baru saja membuat aplikasi instance tunggal. Ini berfungsi sebagai aplikasi sampel langsung dengan satu EC2 instance, sehingga tidak memerlukan penyeimbangan beban atau penskalaan otomatis. Untuk aplikasi contoh tunggal Elastic Beanstalk menciptakan sumber daya berikut: AWS

- EC2 instance — Mesin EC2 virtual Amazon yang dikonfigurasi untuk menjalankan aplikasi web pada platform yang Anda pilih.

  Setiap platform menjalankan satu set perangkat lunak, file konfigurasi, dan penulisan yang berbeda untuk mendukung versi bahasa, kerangka kerja, kontainer web tertentu, atau kombinasi dari semua ini. Sebagian besar platform menggunakan Apache atau nginx sebagai proksi terbalik yang memproses lalu lintas web di depan aplikasi web Anda, meneruskan permintaan ke aplikasi web, menyajikan aset statis, dan menghasilkan log akses dan kesalahan.
- Grup keamanan instans — Grup EC2 keamanan Amazon yang dikonfigurasi untuk memungkinkan lalu lintas masuk di port 80. Sumber daya ini memungkinkan lalu lintas HTTP dari penyeimbang beban mencapai EC2 instance yang menjalankan aplikasi web Anda. Secara default, lalu lintas tidak diizinkan di port lain.
- Bucket Amazon S3 – Lokasi penyimpanan untuk kode sumber, log, dan artifact lainnya yang dibuat saat Anda menggunakan Elastic Beanstalk.

- CloudWatch Alarm Amazon — Dua CloudWatch alarm yang memantau beban pada instans di lingkungan Anda dan dipicu jika beban terlalu tinggi atau terlalu rendah. Saat alarm terpicu, grup Auto Scaling Anda akan menaikkan atau menurunkan skala sebagai respons.
- AWS CloudFormation stack - Elastic AWS CloudFormation Beanstalk digunakan untuk meluncurkan sumber daya di lingkungan Anda dan menyebarkan perubahan konfigurasi. Sumber daya ditentukan di sebuah templat yang dapat Anda lihat di Konsol AWS CloudFormation.
- Nama domain — Nama domain yang merutekan ke aplikasi web Anda dalam formulir *subdomain*.*region*.elasticbeanstalk.com.

Elastic Beanstalk mengelola semua sumber daya tersebut. Ketika Anda mengakhiri lingkungan, Elastic Beanstalk mengakhiri semua sumber daya yang dimuatnya.

Langkah selanjutnya

Setelah lingkungan Anda menjalankan aplikasi, Anda dapat men-deploy versi baru aplikasi atau aplikasi yang berbeda kapan saja. Menyebarkan versi aplikasi baru sangat cepat karena tidak memerlukan penyediaan atau memulai ulang instance. EC2 Anda juga dapat menjelajahi lingkungan baru Anda menggunakan konsol Elastic Beanstalk. Untuk langkah-langkah mendetail, lihat Menjelajahi lingkungan Anda di bagian Memulai panduan ini.

> **Coba lebih banyak tutorial**
>
> Jika Anda ingin mencoba tutorial lain dengan aplikasi contoh yang berbeda, lihat tutorial berikut:
>
> - Men-deploy aplikasi Flask ke Elastic Beanstalk
> - Men-deploy aplikasi Django ke Elastic Beanstalk

Setelah Anda menerapkan satu atau dua contoh aplikasi dan siap untuk mulai mengembangkan dan menjalankan aplikasi Python secara lokal, lihat. Menyiapkan lingkungan pengembangan Python Anda untuk Elastic Beanstalk

Terapkan dengan konsol Elastic Beanstalk

Anda juga dapat menggunakan konsol Elastic Beanstalk untuk meluncurkan aplikasi sampel. Untuk langkah-langkah rinci, lihat Membuat aplikasi contoh di Bab Memulai panduan ini.

Menyiapkan lingkungan pengembangan Python Anda untuk Elastic Beanstalk

Topik ini memberikan instruksi untuk menyiapkan lingkungan pengembangan Python untuk menguji aplikasi Anda secara lokal sebelum menerapkannya. AWS Elastic Beanstalk Ini juga merujuk situs web yang menyediakan instruksi instalasi untuk alat yang berguna.

Untuk mengikuti prosedur di panduan ini, Anda memerlukan terminal atau shell baris perintah untuk menjalankan perintah. Perintah ditampilkan dalam daftar yang diawali dengan simbol prompt ($) dan nama direktori saat ini, bila perlu.

```
~/eb-project$ this is a command
this is output
```

Di Linux dan macOS, Anda dapat menggunakan shell dan manajer paket pilihan Anda. Pada Windows Anda dapat menginstal Windows Subsystem untuk Linux untuk mendapatkan versi Windows terintegrasi dari Ubuntu dan Bash.

Bagian-bagian

- Prasyarat
- Menggunakan lingkungan virtual
- Mengonfigurasi proyek Python pada Elastic Beanstalk

Prasyarat

Daftar berikut memberikan prasyarat umum untuk bekerja dengan Elastic Beanstalk dan aplikasi Python Anda:

- Bahasa Python - Instal versi bahasa Python yang disertakan pada versi platform Elastic Beanstalk Python pilihan Anda. Untuk daftar versi bahasa Python yang didukung, lihat Platform Python yang didukung dalam panduan Platform.AWS Elastic Beanstalk Jika Anda belum menyiapkan Python di mesin pengembangan Anda, lihat halaman unduhan Python di situs web Python.
- **`pip`**utilitas — `pip` Utilitas adalah penginstal paket Python. Ini menginstal dan mencantumkan dependensi untuk proyek Anda, sehingga Elastic Beanstalk tahu cara mengatur lingkungan aplikasi Anda. Untuk informasi selengkapnya`pip`, lihat halaman pip di situs web pip.pypa.io.

- (Opsional) Antarmuka Baris Perintah Elastic Beanstalk (EB CLI) - CLI EB dapat mengemas aplikasi Anda dengan file penerapan yang diperlukan. Itu juga dapat membuat lingkungan Elastic Beanstalk dan menyebarkan aplikasi Anda ke sana. Anda juga dapat melakukan penerapan melalui konsol Elastic Beanstalk, sehingga EB CLI tidak sepenuhnya diperlukan.
- **SSH**Instalasi yang berfungsi — Anda dapat terhubung ke instance yang sedang berjalan dengan protokol SSH untuk memeriksa atau men-debug penerapan.
- **`virtualenv`**paket — `virtualenv` Alat ini menciptakan lingkungan pengembangan dan pengujian untuk aplikasi Anda. Elastic Beanstalk dapat mereplikasi lingkungan ini tanpa menginstal paket tambahan yang tidak diperlukan oleh aplikasi Anda. Untuk informasi lebih lanjut, lihat situs web virtualenv. Setelah menginstal Python, Anda dapat menginstal `virtualenv` paket dengan perintah berikut:

```
$ pip install virtualenv
```

Menggunakan lingkungan virtual

Setelah prasyarat telah diinstal, siapkan lingkungan virtual dengan `virtualenv` untuk menginstal dependensi aplikasi Anda. Dengan menggunakan lingkungan virtual, Anda dapat membedakan dengan tepat paket mana yang dibutuhkan oleh aplikasi Anda sehingga paket yang diperlukan diinstal pada EC2 instance yang menjalankan aplikasi Anda.

Untuk menyiapkan lingkungan virtual

1. Buka jendela baris-perintah dan ketik:

   ```
   $ virtualenv /tmp/eb_python_app
   ```

   Ganti *`eb_python_app`* dengan nama yang masuk akal untuk aplikasi Anda (menggunakan nama aplikasi Anda adalah ide yang bagus). Perintah `virtualenv` membuat lingkungan virtual untuk Anda dalam direktori tertentu dan mencetak hasil dari tindakannya:

   ```
   Running virtualenv with interpreter /usr/bin/python
   New python executable in /tmp/eb_python_app/bin/python3.12
   Also creating executable in /tmp/eb_python_app/bin/python
   Installing setuptools, pip...done.
   ```

2. Setelah lingkungan virtual Anda siap, mulai dengan menjalankan skrip `activate` yang terletak di direktori `bin` lingkungan tersebut. Misalnya, untuk memulai lingkungan eb_python_app yang dibuat di langkah sebelumnya, Anda akan mengetik:

   ```
   $ source /tmp/eb_python_app/bin/activate
   ```

   Lingkungan virtual mencetak namanya (misalnya: `(eb_python_app)`) di awal setiap prompt perintah, mengingatkan bahwa Anda berada di lingkungan Python virtual.

3. Untuk berhenti menggunakan lingkungan virtual Anda dan kembali ke juru bahasa Python default sistem dengan semua perpustakaan terinstal, jalankan perintah `deactivate`.

   ```
   (eb_python_app) $ deactivate
   ```

> **Note**
>
> Setelah dibuat, Anda dapat menghidupkan ulang lingkungan virtual kapan pun itu dengan menjalankan skrip `activate` kembali.

Mengonfigurasi proyek Python pada Elastic Beanstalk

Anda dapat menggunakan CLI Elastic Beanstalk untuk mempersiapkan aplikasi Python Anda untuk deployment dengan menggunakan Elastic Beanstalk.

Untuk mengonfigurasi aplikasi Python untuk deployment dengan menggunakan Elastic Beanstalk

1. Dari dalam lingkungan virtual, kembali ke bagian atas direktori berstruktur pohon milik proyek (`python_eb_app`) Anda, dan ketik:

   ```
   pip freeze >requirements.txt
   ```

   Perintah ini menyalin nama dan versi paket yang diinstal di lingkungan virtual Anda `kerequirements.txt`, Misalnya, jika paket PyYaml, versi 3.11 diinstal di lingkungan virtual Anda, file akan berisi baris:

   ```
   PyYAML==3.11
   ```

Hal ini mengizinkan Elastic Beanstalk meniru lingkungan Python aplikasi Anda menggunakan paket dan versi sama yang biasa mengembangkan dan menguji aplikasi Anda.

2. Konfigurasi repositori EB CLI dengan perintah eb init. Ikuti prompt untuk memilih wilayah, platform, dan opsi lainnya.

Secara default, Elastic Beanstalk mencari file bernama `application.py` untuk memulai aplikasi Anda. Jika ini tidak ada dalam proyek Python yang telah Anda buat, diperlukan beberapa penyesuaian lingkungan aplikasi Anda. Anda juga perlu mengatur variabel lingkungan sehingga modul aplikasi Anda dapat dimuat. Lihat Menggunakan platform Python Elastic Beanstalk untuk informasi selengkapnya.

Menggunakan platform Python Elastic Beanstalk

Topik ini menjelaskan cara mengonfigurasi, membangun, dan menjalankan aplikasi Python Anda di Elastic Beanstalk.

AWS Elastic Beanstalk mendukung sejumlah cabang platform untuk berbagai versi bahasa pemrograman Python. Lihat Python di dokumen AWS Elastic Beanstalk Platform untuk daftar lengkapnya.

Aplikasi web Python dapat berjalan di belakang server proxy dengan WSGI. Elastic Beanstalk menyediakan Gunicorn sebagai server WSGI default.

Anda dapat menambahkan `Procfile` ke paket sumber untuk menentukan dan mengonfigurasi server WSGI bagi aplikasi Anda. Untuk detail selengkapnya, lihat the section called “Procfile”.

Anda dapat menggunakan file `Pipfile` dan `Pipfile.lock` yang dibuat oleh Pipenv untuk menentukan dependensi paket Python dan persyaratan lainnya. Untuk detail tentang menentukan dependensi, lihat the section called “Menentukan dependensi”.

Elastic Beanstalk menyediakan opsi konfigurasi yang dapat Anda gunakan untuk menyesuaikan perangkat lunak yang berjalan EC2 pada instance di lingkungan Elastic Beanstalk Anda. Anda dapat mengonfigurasi variabel lingkungan yang diperlukan aplikasi Anda, mengaktifkan rotasi log ke Amazon S3, dan memetakan folder dalam sumber aplikasi Anda yang berisi file statis ke jalur yang disajikan server proksi.

Opsi konfigurasi tersedia di konsol Elastic Beanstalk untuk memodifikasi konfigurasi dari lingkungan yang sedang berjalan. Agar Anda tidak kehilangan konfigurasi lingkungan ketika mengakhirinya,

Anda dapat menggunakan konfigurasi tersimpan untuk menyimpan pengaturan dan kemudian menerapkannya ke lingkungan lain.

Untuk menyimpan pengaturan di kode sumber, Anda dapat menyertakan file konfigurasi. Pengaturan di file konfigurasi diterapkan setiap kali Anda membuat lingkungan atau men-deploy aplikasi Anda. Anda juga dapat menggunakan file konfigurasi untuk menginstal paket, menjalankan penulisan, dan melakukan operasi penyesuaian instans lainnya selama deployment.

Pengaturan yang diterapkan di konsol Elastic Beanstalk mengganti pengaturan yang sama di file konfigurasi, jika ada. Hal tersebut memungkinkan Anda memiliki pengaturan default di file konfigurasi, dan mengganti keduanya dengan pengaturan khusus lingkungan di konsol tersebut. Untuk informasi lebih lanjut tentang yang diutamakan, dan metode lain untuk mengubah pengaturan, lihat Opsi konfigurasi.

Untuk paket Python tersedia dari `pip`, Anda dapat menyertakan file persyaratan di akar kode sumber aplikasi Anda. Elastic Beanstalk menginstal paket dependensi yang ditentukan dalam file persyaratan selama deployment. Untuk detail selengkapnya, lihat the section called “Menentukan dependensi”.

Untuk detail tentang berbagai cara memperluas platform berbasis Linux Elastic Beanstalk, lihat the section called “Memperluas platform Linux”.

Mengonfigurasi lingkungan Python Anda

Pengaturan platform Python memungkinkan Anda menyempurnakan perilaku instans Amazon Anda. EC2 Anda dapat mengedit konfigurasi instans Amazon lingkungan Elastic Beanstalk menggunakan konsol EC2 Elastic Beanstalk.

Gunakan konsol Elastic Beanstalk untuk mengonfigurasi AWS X-Ray pengaturan proses Python, mengaktifkan, mengaktifkan rotasi log ke Amazon S3, dan mengonfigurasi variabel yang dapat dibaca aplikasi Anda dari lingkungan.

Untuk mengonfigurasi lingkungan Python Anda di konsol Elastic Beanstalk

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Dalam kategori konfigurasi Pembaruan, pemantauan, dan pencatatan, pilih Edit.

Pengaturan Python

- Server proksi – Server proksi yang akan digunakan pada instans lingkungan Anda. Secara default, nginx digunakan.
- Jalur WSGI – Nama atau jalur ke file aplikasi utama Anda. Misalnya, `application.py`, atau `django/wsgi.py`.
- NumProcesses— Jumlah proses yang dijalankan pada setiap instance aplikasi.
- NumThreads— Jumlah thread untuk dijalankan di setiap proses.

AWS X-Ray pengaturan

- Daemon X-Ray - Jalankan AWS X-Ray daemon untuk memproses data jejak dari. AWS X-Ray SDK for Python

Opsi log

Bagian Opsi Log memiliki dua pengaturan:

- Profil instans– Menentukan profil instans yang memiliki izin untuk mengakses bucket Amazon S3 yang terkait dengan aplikasi Anda.
- Aktifkan rotasi file log ke Amazon S3 — Menentukan apakah file log untuk EC2 instans Amazon aplikasi Anda disalin ke bucket Amazon S3 yang terkait dengan aplikasi Anda.

File statis

Untuk meningkatkan kinerja, Anda dapat menggunakan bagian File statis untuk mengkonfigurasi server proxy untuk melayani file statis (misalnya, HTML atau gambar) dari satu set direktori di dalam aplikasi web Anda. Untuk setiap direktori, Anda mengatur jalur virtual ke pemetaan direktori. Saat server proksi menerima permintaan untuk file di jalur yang ditentukan, server langsung menyajikan file daripada merutekan permintaan ke aplikasi Anda.

Untuk detail tentang mengonfigurasi file statis menggunakan file konfigurasi atau konsol Elastic Beanstalk, lihat. the section called “File statis”

Secara default, server proksi di lingkungan Python menyajikan file apa pun dalam folder bernama `static` di jalur `/static`. Misalnya, jika sumber aplikasi Anda berisi file bernama `logo.png` dalam folder bernama `static`, server proksi menyajikannya untuk pengguna di

subdomain.elasticbeanstalk.com/static/logo.png. Anda dapat mengonfigurasi pemetaan tambahan seperti yang dijelaskan di bagian ini.

Properti lingkungan

Anda dapat menggunakan properti lingkungan untuk memberikan informasi ke aplikasi Anda dan mengonfigurasi variabel lingkungan. Misalnya, Anda dapat membuat properti lingkungan bernama CONNECTION_STRING yang menentukan string koneksi yang dapat digunakan aplikasi Anda agar terhubung ke basis data.

Di dalam lingkungan Python yang berjalan di Elastic Beanstalk, nilai-nilai ini dapat diakses menggunakan kamus os.environ Python. Untuk informasi lebih lanjut, lihat http://docs.python. org/library/os.html.

Anda dapat menggunakan kode yang terlihat mirip dengan yang berikut untuk mengakses kunci dan parameter:

```
import os
endpoint = os.environ['API_ENDPOINT']
```

Properti lingkungan juga dapat memberikan informasi ke kerangka kerja. Misalnya, Anda dapat membuat properti bernama DJANGO_SETTINGS_MODULE untuk mengonfigurasi Django agar menggunakan modul pengaturan khusus. Bergantung pada lingkungan, nilai bisa saja development.settings, production.settings, dll.

Lihat Variabel lingkungan dan pengaturan perangkat lunak lainnya untuk informasi selengkapnya.

Namespace konfigurasi Python

Anda dapat menggunakan file konfigurasi untuk mengatur opsi konfigurasi dan melakukan tugas-tugas konfigurasi instans lain selama deployment. Opsi konfigurasi dapat berupa platform khusus atau berlaku untuk semua platform di layanan Elastic Beanstalk secara keseluruhan. Opsi konfigurasi diatur ke dalam ruang nama.

Platform Python mendefinisikan opsi dalam namespace aws:elasticbeanstalk:environment:proxy, aws:elasticbeanstalk:environment:proxy:staticfiles, dan aws:elasticbeanstalk:container:python.

File konfigurasi contoh berikut menentukan pengaturan opsi konfigurasi untuk membuat properti lingkungan bernama DJANGO_SETTINGS_MODULE, memilih server proksi Apache, menentukan dua

opsi file statis yang memetakan direktori bernama `statichtml` ke jalur `/html` dan sebuah direktori bernama `staticimages` ke jalur `/images`, dan menentukan pengaturan tambahan di namespace `aws:elasticbeanstalk:container:python`. Namespace ini berisi opsi yang memungkinkan Anda menentukan lokasi skrip WSGI dalam kode sumber Anda, dan jumlah utas serta proses untuk berjalan di WSGI.

```
option_settings:
  aws:elasticbeanstalk:application:environment:
    DJANGO_SETTINGS_MODULE: production.settings
  aws:elasticbeanstalk:environment:proxy:
    ProxyServer: apache
  aws:elasticbeanstalk:environment:proxy:staticfiles:
    /html: statichtml
    /images: staticimages
  aws:elasticbeanstalk:container:python:
    WSGIPath: ebdjango.wsgi:application
    NumProcesses: 3
    NumThreads: 20
```

> **Catatan**
>
> - Jika Anda menggunakan versi platform Python Amazon Linux AMI (Amazon Linux 2 yang terdahulu), ganti nilai `WSGIPath` dengan `ebdjango/wsgi.py`. Nilai dalam contoh berfungsi dengan menggunakan server Gunicorn WSGI, yang tidak didukung pada versi platform Amazon Linux AMI.
> - Selain itu, versi platform yang lebih lama ini menggunakan namespace yang berbeda dalam mengonfigurasi file statis —`aws:elasticbeanstalk:container:python:staticfiles`. Platform tersebut memiliki nama opsi dan makna yang sama dengan namespace file statis standar.

File konfigurasi juga mendukung beberapa kunci untuk memodifikasi perangkat lunak lebih lanjut pada instans lingkungan Anda. Contoh ini menggunakan kunci paket untuk menginstal Memcached dengan `yum` dan perintah kontainer untuk menjalankan perintah yang mengonfigurasi server selama deployment:

```
packages:
  yum:
```

```
    libmemcached-devel: '0.31'

container_commands:
  collectstatic:
    command: "django-admin.py collectstatic --noinput"
  01syncdb:
    command: "django-admin.py syncdb --noinput"
    leader_only: true
  02migrate:
    command: "django-admin.py migrate"
    leader_only: true
  03wsgipass:
    command: 'echo "WSGIPassAuthorization On" >> ../wsgi.conf'
  99customize:
    command: "scripts/customize.sh"
```

Elastic Beanstalk memberikan banyak opsi konfigurasi untuk menyesuaikan lingkungan Anda. Selain file konfigurasi, Anda juga dapat mengatur opsi konfigurasi menggunakan konsol tersebut, konfigurasi tersimpan, EB CLI, atau AWS CLI. Untuk informasi selengkapnya, lihat Opsi konfigurasi.

Yang dapat **`python3`** dieksekusi

Versi `python3` executable yang tersedia pada EC2 instance di lingkungan Elastic Beanstalk Python tidak akan selalu sesuai dengan versi Python yang sama yang digunakan oleh platform. Misalnya, pada platform Python 3.12 AL2 023, menunjuk `/usr/bin/python3` ke Python 3.9. Ini karena Python 3.9 adalah sistem Python pada 023. AL2 Untuk informasi selengkapnya, lihat Python di AL2 023 di Panduan Pengguna Amazon Linux 2023. Anda dapat mengakses executable yang sesuai dengan versi Python yang digunakan oleh platform di lokasi berversi (misalnya`/usr/bin/python3.12`) atau di direktori lingkungan virtual aplikasi (mis.). `bin /var/app/venv/staging-LQM1lest/bin/python3` Platform menggunakan executable Python yang benar yang sesuai dengan cabang platform.

Mengkonfigurasi server WSGI dengan Procfile pada Elastic Beanstalk

Anda dapat menambahkan `Procfile` ke paket sumber untuk menentukan dan mengonfigurasi server WSGI bagi aplikasi Anda. Anda dapat menentukan perintah mulai dan menjalankan kustom di`Procfile`.

Anda menggunakan `Procfile`, maka akan mengganti opsi namespace `aws:elasticbeanstalk:container:python` yang ditetapkan menggunakan file konfigurasi.

Contoh berikut menggunakan `Procfile` untuk menentukan uWSGI sebagai server dan mengonfigurasinya.

Example Procfile

```
web: uwsgi --http :8000 --wsgi-file application.py --master --processes 4 --threads 2
```

Contoh berikut menggunakan `Procfile` untuk mengonfigurasi Gunicorn, server WSGI default.

Example Procfile

```
web: gunicorn --bind :8000 --workers 3 --threads 2 project.wsgi:application
```

> **Catatan**
>
> - Jika Anda mengonfigurasi server WSGI selain Gunicorn, pastikan untuk menentukannya juga sebagai dependensi aplikasi Anda, sehingga terinstal pada instans lingkungan Anda. Untuk detail tentang spesifikasi dependensi, lihat the section called “Menentukan dependensi”.
> - Port default untuk server WSGI adalah 8000. Jika Anda menentukan nomor port yang berbeda di perintah `Procfile` Anda, mengatur properti lingkungan `PORT` ke nomor port ini juga.

Menentukan dependensi menggunakan file persyaratan pada Elastic Beanstalk

Topik ini menjelaskan cara mengkonfigurasi aplikasi Anda untuk menginstal paket Python lain yang diperlukan. Aplikasi Python tipikal memiliki dependensi pada paket Python pihak ketiga lainnya. Dengan platform Elastic Beanstalk Python, Anda memiliki beberapa cara untuk menentukan paket Python yang bergantung pada aplikasi Anda.

Gunakan **`pip`** dan **`requirements.txt`**

Alat standar untuk memasang paket Python adalah `pip`. Alat ini memiliki fitur yang memungkinkan Anda untuk menentukan semua paket yang Anda butuhkan (dan juga versinya) dalam satu file

persyaratan. Untuk informasi selengkapnya tentang file persyaratan, lihat Format File Persyaratan di situs web dokumentasi pip.

Buat file bernama `requirements.txt` dan letakkan di direktori tingkat atas dari paket sumber Anda. Berikut ini adalah contoh file `requirements.txt` untuk Django.

```
Django==2.2
mysqlclient==2.0.3
```

Di lingkungan pengembangan, Anda dapat menggunakan perintah `pip freeze` untuk menghasilkan file persyaratan Anda.

```
~/my-app$ pip freeze > requirements.txt
```

Untuk memastikan bahwa file persyaratan Anda hanya berisi paket yang benar-benar digunakan oleh aplikasi Anda, gunakan lingkungan virtual yang hanya menginstal paket tersebut. Di luar lingkungan virtual, output dari `pip freeze` akan menyertakan semua paket `pip` yang diinstal pada mesin pengembangan Anda, termasuk yang disertakan pada sistem operasi Anda.

> **Note**
>
> Pada versi platform Python Amazon Linux AMI, Elastic Beanstalk aslinya tidak mendukung Pipenv atau Pipfiles. Jika Anda menggunakan Pipenv untuk mengelola dependensi aplikasi Anda, jalankan perintah berikut untuk menghasilkan file `requirements.txt`.
>
> ```
> ~/my-app$ pipenv lock -r > requirements.txt
> ```
>
> Untuk mempelajari selengkapnya, lihat Menghasilkan requirements.txt dalam dokumentasi Pipenv.

Gunakan Pipenv dan `Pipfile`

Pipenv adalah alat pengemasan Python modern. Alat tersebut mengombinasikan instalasi paket dengan pembuatan dan pengelolaan file dependensi dan virtualenv untuk aplikasi Anda. Untuk informasi lebih lanjut, lihat Pipenv: Alur Kerja Python Dev untuk Manusia.

Pipenv memelihara dua file:

- `Pipfile`— File ini berisi berbagai jenis dependensi dan persyaratan.
- `Pipfile.lock`— File ini berisi snapshot versi yang memungkinkan build deterministik.

Anda dapat membuat file-file ini di lingkungan pengembangan Anda dan memasukkannya ke dalam direktori tingkat atas bundel sumber yang Anda gunakan ke Elastic Beanstalk. Untuk informasi selengkapnya tentang dua file ini, lihat Contoh Pipfile dan Pipfile.lock.

Contoh berikut menggunakan Pipenv untuk menginstal Django dan kerangka kerja REST Django. Perintah ini membuat file `Pipfile` dan `Pipfile.lock`.

```
~/my-app$ pipenv install django
~/my-app$ pipenv install djangorestframework
```

Precedence

Jika Anda menyertakan lebih dari satu file persyaratan yang dijelaskan dalam topik ini, Elastic Beanstalk hanya menggunakan salah satunya. Daftar berikut menunjukkan yang diutamakan, dalam urutan menurun.

1. `requirements.txt`
2. `Pipfile.lock`
3. `Pipfile`

> **Note**
>
> Dimulai dengan rilis platform Amazon Linux 2 7 Maret 2023, jika Anda menyediakan lebih dari satu file ini, Elastic Beanstalk akan mengeluarkan pesan konsol yang menyatakan salah satu file dependensi mana yang digunakan selama penerapan.

Langkah-langkah berikut menjelaskan logika yang diikuti Elastic Beanstalk untuk menginstal dependensi saat menerapkan instance.

- Jika ada `requirements.txt` file, kami menggunakan perintah`pip install -r requirements.txt`.

- Dimulai dengan rilis platform Amazon Linux 2 7 Maret 2023, jika tidak ada `requirements.txt` file, tetapi ada`Pipfile.lock`, kami menggunakan perintah`pipenv sync`. Sebelum rilis itu, kami menggunakan`pipenv install --ignore-pipfile`.
- Jika tidak ada `requirements.txt` file atau a`Pipfile.lock`, tetapi ada`Pipfile`, kami menggunakan perintah`pipenv install --skip-lock`.
- Jika tidak satu pun dari tiga file persyaratan ditemukan, kami tidak menginstal dependensi aplikasi apa pun.

Men-deploy aplikasi Flask ke Elastic Beanstalk

Tutorial ini memandu Anda melalui proses menghasilkan aplikasi Flask dan menyebarkannya ke lingkungan AWS Elastic Beanstalk . Flask adalah kerangka kerja aplikasi web sumber terbuka untuk Python.

Dalam tutorial ini, Anda akan melakukan hal berikut:

- Menyiapkan lingkungan virtual Python dengan Flask
- Buat aplikasi Flask
- Men-deploy situs Anda dengan EB CLI
- Pembersihan

Prasyarat

Tutorial ini mengasumsikan Anda memiliki pengetahuan tentang operasi Elastic Beanstalk dasar dan konsol Elastic Beanstalk. Jika belum, ikuti petunjuk di Pelajari cara memulai dengan Elastic Beanstalk untuk meluncurkan lingkungan Elastic Beanstalk pertama Anda.

Untuk mengikuti prosedur di panduan ini, Anda memerlukan terminal atau shell baris perintah untuk menjalankan perintah. Perintah ditampilkan dalam daftar yang diawali dengan simbol prompt ($) dan nama direktori saat ini, bila perlu.

```
~/eb-project$ this is a command
this is output
```

Di Linux dan macOS, Anda dapat menggunakan shell dan manajer paket pilihan Anda. Pada Windows Anda dapat menginstal Windows Subsystem untuk Linux untuk mendapatkan versi Windows terintegrasi dari Ubuntu dan Bash.

Flask membutuhkan Python 3.7 atau yang lebih baru. Dalam tutorial ini kita menggunakan Python 3.7 dan versi platform Elastic Beanstalk yang sesuai. Pasang Python dengan mengikuti petunjuk di Menyiapkan lingkungan pengembangan Python Anda untuk Elastic Beanstalk.

Kerangka kerja Flask akan diinstal sebagai bagian dari tutorial.

Tutorial ini juga menggunakan Elastic Beanstalk Command Line Interface (EB CLI). Untuk detail dalam menginstal dan mengonfigurasi EB CLI, lihat Instal EB CLI dengan skrip pengaturan (disarankan) dan Mengonfigurasi EB CLI.

Menyiapkan lingkungan virtual Python dengan Flask

Membuat direktori proyek dan lingkungan virtual untuk aplikasi Anda, dan menginstal Flask.

Untuk mempersiapkan lingkungan proyek Anda

1. Buat direktori proyek.

   ```
   ~$ mkdir eb-flask
   ~$ cd eb-flask
   ```

2. Buat dan aktifkan lingkungan virtual bernama `virt`:

   ```
   ~/eb-flask$ virtualenv virt
   ~$ source virt/bin/activate
   (virt) ~/eb-flask$
   ```

   Anda akan melihat `(virt)` ditambahkan ke prompt perintah Anda, yang menunjukkan bahwa Anda berada di lingkungan virtual. Gunakan lingkungan virtual hingga akhir tutorial ini.

3. Pasang Flask dengan `pip install`:

   ```
   (virt)~/eb-flask$ pip install flask==2.0.3
   ```

4. Lihat perpustakaan yang diinstal dengan `pip freeze`:

   ```
   (virt)~/eb-flask$ pip freeze
   click==8.1.1
   Flask==2.0.3
   itsdangerous==2.1.2
   Jinja2==3.1.1
   ```

```
MarkupSafe==2.1.1
Werkzeug==2.1.0
```

Perintah ini mencantumkan semua paket yang diinstal di lingkungan virtual Anda. Berhubung Anda berada di lingkungan virtual, paket yang diinstal secara global seperti EB CLI tidak akan ditampilkan.

5. Simpan output dari `pip freeze` ke file bernama `requirements.txt`.

```
(virt)~/eb-flask$ pip freeze > requirements.txt
```

File ini meminta Elastic Beanstalk untuk menginstal perpustakaan selama deployment. Untuk informasi selengkapnya, lihat Menentukan dependensi menggunakan file persyaratan pada Elastic Beanstalk.

Buat aplikasi Flask

Selanjutnya, buat aplikasi yang akan Anda deploy menggunakan Elastic Beanstalk. Kami akan membuat layanan RESTful web “Hello World”.

Buat file teks dalam direktori ini bernama `application.py` dengan isi berikut:

Example **`~/eb-flask/application.py`**

```
from flask import Flask

# print a nice greeting.
def say_hello(username = "World"):
    return '<p>Hello %s!</p>\n' % username

# some bits of text for the page.
header_text = '''
    <html>\n<head> <title>EB Flask Test</title> </head>\n<body>'''
instructions = '''
    <p><em>Hint</em>: This is a RESTful web service! Append a username
    to the URL (for example: <code>/Thelonious</code>) to say hello to
    someone specific.</p>\n'''
home_link = '<p><a href="/">Back</a></p>\n'
footer_text = '</body>\n</html>'

# EB looks for an 'application' callable by default.
```

```
application = Flask(__name__)

# add a rule for the index page.
application.add_url_rule('/', 'index', (lambda: header_text +
    say_hello() + instructions + footer_text))

# add a rule when the page is accessed with a name appended to the site
# URL.
application.add_url_rule('/<username>', 'hello', (lambda username:
    header_text + say_hello(username) + home_link + footer_text))

# run the app.
if __name__ == "__main__":
    # Setting debug to True enables debug output. This line should be
    # removed before deploying a production app.
    application.debug = True
    application.run()
```

Contoh ini membuat penyesuaian salam pembuka yang bervariasi berdasarkan jalur yang digunakan untuk mengakses layanan.

> **Note**
>
> Dengan menambahkan `application.debug = True` sebelum menjalankan aplikasi, output debug diaktifkan jika seandainya ada sesuatu yang tidak beres. Hal ini merupakan penerapan yang baik untuk pengembangan, tetapi Anda harus menghapus pernyataan debug dalam kode produksi, karena output debug dapat mengungkapkan aspek internal aplikasi Anda.

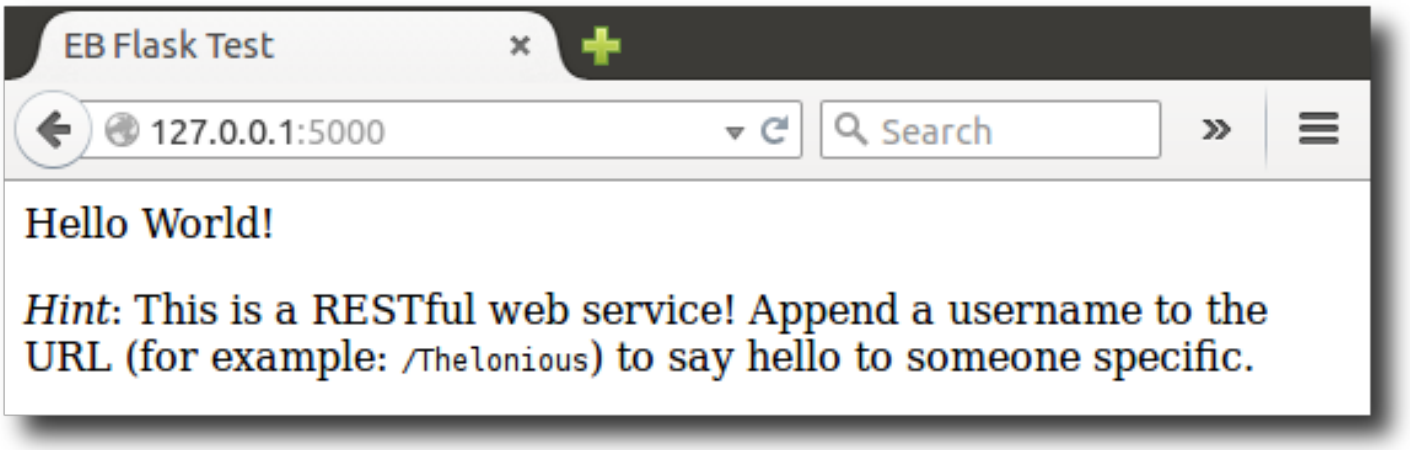
Menggunakan `application.py` sebagai nama file dan memberikan objek `application` yang dapat dipanggil (objek Flask, dalam hal ini) mengizinkan Elastic Beanstalk dengan mudah untuk menemukan kode aplikasi Anda.

Jalankan `application.py` dengan Python:

```
(virt) ~/eb-flask$ python application.py
 * Serving Flask app "application" (lazy loading)
 * Environment: production
   WARNING: Do not use the development server in a production environment.
   Use a production WSGI server instead.
```

```
 * Debug mode: on
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 313-155-123
```

Buka `http://127.0.0.1:5000/` di peramban web Anda. Anda harus melihat aplikasi yang berjalan, menunjukkan halaman indeks:



Periksa log server untuk melihat output dari permintaan Anda. Anda dapat menghentikan server web dan mengembalikan ke lingkungan virtual Anda dengan mengetik Ctrl+C.

Jika Anda justru mendapati output debug, perbaiki kesalahan dan pastikan aplikasi berjalan secara lokal sebelum mengonfigurasinya untuk Elastic Beanstalk.

Men-deploy situs Anda dengan EB CLI

Anda telah menambahkan semua yang dibutuhkan untuk men-deploy aplikasi Anda pada Elastic Beanstalk. Direktori proyek Anda seharusnya sekarang terlihat seperti ini:

```
~/eb-flask/
|-- virt
|-- application.py
`-- requirements.txt
```

Folder `virt`, bagaimanapun, tidak diperlukan aplikasi untuk berjalan di Elastic Beanstalk. Ketika Anda men-deploy, Elastic Beanstalk akan membuat lingkungan virtual baru pada instans server dan menginstal perpustakaan yang tercantum dalam `requirements.txt`. Untuk meminimalkan ukuran

paket sumber yang Anda unggah selama deployment, tambahkan file .ebignore yang meminta EB CLI untuk meninggalkan folder `virt`.

Example ~/eb-flask/.ebignore

```
virt
```

Selanjutnya, Anda akan membuat lingkungan aplikasi dan men-deploy aplikasi Anda yang dikonfigurasi dengan Elastic Beanstalk.

Untuk membuat lingkungan dan men-deploy aplikasi Flask Anda

1. Inisialisasi repositori EB CLI Anda dengan perintah eb init:

   ```
   ~/eb-flask$ eb init -p python-3.7 flask-tutorial --region us-east-2
   Application flask-tutorial has been created.
   ```

   Perintah ini membuat aplikasi baru bernama `flask-tutorial` dan mengkonfigurasi repositori lokal Anda untuk membuat lingkungan dengan versi platform Python 3.7 terbaru.

2. (opsional) Jalankan eb init lagi untuk mengonfigurasi keypair default sehingga Anda dapat terhubung ke EC2 instance yang menjalankan aplikasi Anda dengan SSH:

   ```
   ~/eb-flask$ eb init
   Do you want to set up SSH for your instances?
   (y/n): y
   Select a keypair.
   1) my-keypair
   2) [ Create new KeyPair ]
   ```

   Pilih pasangan kunci jika Anda sudah memilikinya, atau ikuti prompt untuk membuat yang baru. Jika Anda tidak melihat prompt atau perlu mengubah pengaturan Anda nanti, jalankan eb init -i.

3. Buat lingkungan dan deploy aplikasi Anda ke dalamnya dengan eb create:

   ```
   ~/eb-flask$ eb create flask-env
   ```

Pembuatan lingkungan membutuhkan waktu sekitar 5 menit dan membuat sumber daya berikut:

- EC2 instance — Mesin virtual Amazon Elastic Compute Cloud (Amazon EC2) yang dikonfigurasi untuk menjalankan aplikasi web pada platform yang Anda pilih.

  Setiap platform menjalankan satu set perangkat lunak, file konfigurasi dan penulisan tertentu untuk mendukung versi bahasa, kerangka kerja, kontainer web tertentu, atau kombinasi dari semua ini. Sebagian besar platform menggunakan Apache atau NGINX sebagai proksi terbalik yang ada di depan aplikasi web Anda, meneruskan permintaan ke aplikasi web, menyajikan aset statis, dan menghasilkan log akses dan kesalahan.
- Grup keamanan instans — Grup EC2 keamanan Amazon yang dikonfigurasi untuk memungkinkan lalu lintas masuk pada port 80. Sumber daya ini memungkinkan lalu lintas HTTP dari penyeimbang beban mencapai EC2 instance yang menjalankan aplikasi web Anda. Secara default, lalu lintas tidak diizinkan di port lain.
- Penyeimbang beban – Penyeimbang beban Elastic Load Balancing yang dikonfigurasi untuk mendistribusikan permintaan ke instans yang menjalankan aplikasi Anda. Penyeimbang beban juga menghilangkan kebutuhan untuk mengekspos instans Anda langsung ke internet.
- Grup keamanan penyeimbang beban - Grup EC2 keamanan Amazon yang dikonfigurasi untuk memungkinkan lalu lintas masuk di port 80. Sumber daya ini memungkinkan lalu lintas HTTP dari internet mencapai penyeimbang beban. Secara default, lalu lintas tidak diizinkan di port lain.
- Grup Auto Scaling – Grup Auto Scaling yang dikonfigurasi untuk menggantikan instans jika diakhiri atau menjadi tidak tersedia.
- Bucket Amazon S3 – Lokasi penyimpanan untuk kode sumber, log, dan artifact lainnya yang dibuat saat Anda menggunakan Elastic Beanstalk.
- CloudWatch Alarm Amazon — Dua CloudWatch alarm yang memantau beban pada instans di lingkungan Anda dan yang dipicu jika beban terlalu tinggi atau terlalu rendah. Saat alarm terpicu, grup Auto Scaling Anda akan menaikkan atau menurunkan skala sebagai respons.
- AWS CloudFormation stack - Elastic AWS CloudFormation Beanstalk digunakan untuk meluncurkan sumber daya di lingkungan Anda dan menyebarkan perubahan konfigurasi. Sumber daya ditentukan di sebuah templat yang dapat Anda lihat di Konsol AWS CloudFormation.
- Nama domain — Nama domain yang merutekan ke aplikasi web Anda dalam formulir *subdomain*.*region*.elasticbeanstalk.com.

> **Keamanan domain**
>
> Untuk meningkatkan keamanan aplikasi Elastic Beanstalk Anda, domain elasticbeanstalk.com terdaftar di Daftar Akhiran Publik (PSL).

Jika Anda perlu mengatur cookie sensitif di nama domain default untuk aplikasi Elastic Beanstalk Anda, kami sarankan Anda menggunakan cookie `__Host-` dengan awalan untuk meningkatkan keamanan. Praktik ini melindungi domain Anda dari upaya pemalsuan permintaan lintas situs (CSRF). Untuk informasi selengkapnya, lihat halaman Set-Cookie di Jaringan Pengembang Mozilla.

Semua sumber daya ini dikelola oleh Elastic Beanstalk. Ketika Anda mengakhiri lingkungan, Elastic Beanstalk mengakhiri semua sumber daya yang dimuatnya.

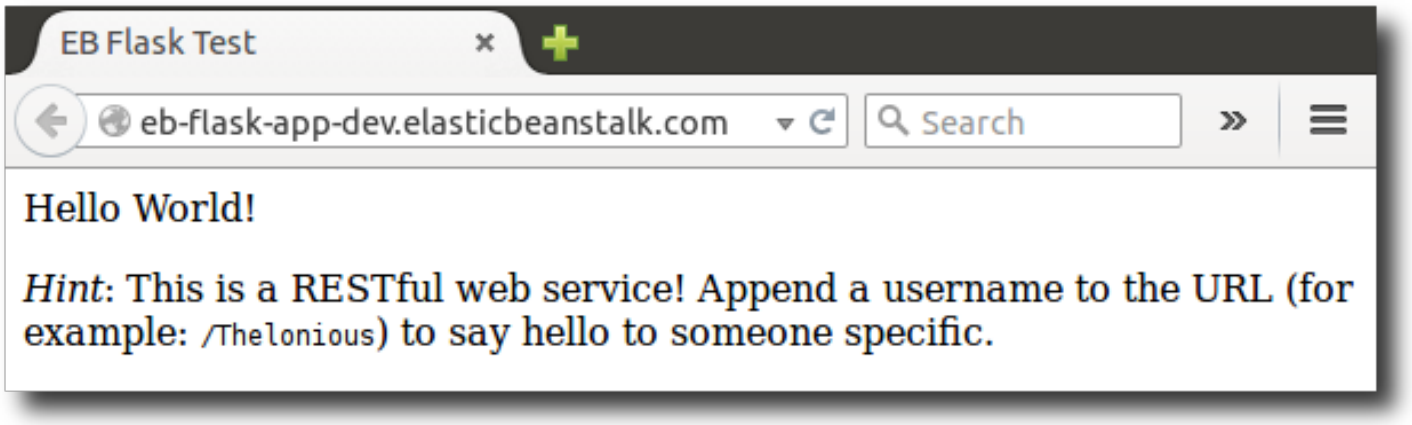
Note

Bucket Amazon S3 yang dibuat Elastic Beanstalk dibagi antar lingkungan dan tidak dihapus selama pengakhiran lingkungan. Untuk informasi selengkapnya, lihat Menggunakan Elastic Beanstalk dengan Amazon S3.

Ketika proses pembuatan lingkungan selesai, buka situs web Anda dengan eb open:

```
~/eb-flask$ eb open
```

Ini akan membuka jendela peramban menggunakan nama domain yang dibuat untuk aplikasi Anda. Anda akan melihat situs web Flask yang sama seperti yang Anda telah buat dan uji secara lokal.



Jika aplikasi tidak berjalan, atau menerima pesan kesalahan, lihat Pemecahan Masalah Deployment untuk bantuan tentang cara menentukan penyebab kesalahan.

Jika Anda benar-benar melihat aplikasi Anda berjalan, maka selamat, Anda telah men-deploy aplikasi Flask pertama Anda dengan Elastic Beanstalk!

Pembersihan

Setelah Anda selesai bekerja dengan kode demo, Anda dapat mengakhiri lingkungan Anda. Elastic Beanstalk menghapus semua sumber daya AWS terkait, seperti instans EC2 Amazon, instans database, penyeimbang beban, grup keamanan,dan alarm.

Menghapus sumber daya tidak menghapus aplikasi Elastic Beanstalk, sehingga Anda dapat membuat lingkungan baru untuk aplikasi Anda kapan saja.

Untuk mengakhiri lingkungan Elastic Beanstalk Anda dari konsol

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pilih Tindakan, lalu pilih Terminate environment.
4. Gunakan kotak dialog di layar untuk mengonfirmasi pengakhiran lingkungan.

Atau, dengan EB CLI:

```
~/eb-flask$ eb terminate flask-env
```

Langkah selanjutnya

Untuk informasi lebih selengkapnya tentang Flask, kunjungi flask.pocoo.org.

Jika anda ingin mencoba kerangka kerja web Python lain, lihat Men-deploy aplikasi Django ke Elastic Beanstalk.

Men-deploy aplikasi Django ke Elastic Beanstalk

Tutorial ini memandu dalam deployment situs web Django default yang dibuat otomatis ke lingkungan AWS Elastic Beanstalk yang menjalankan Python. Tutorial ini menunjukkan Anda cara meng-host sebuah aplikasi web Python di cloud dengan menggunakan lingkungan Elastic Beanstalk.

Dalam tutorial ini, Anda akan melakukan hal berikut:

- Menyiapkan lingkungan virtual Python dan menginstal Django
- Buat proyek Django
- Konfigurasikan aplikasi Django Anda untuk Elastic Beanstalk
- Men-deploy situs Anda dengan EB CLI
- Perbarui aplikasi Anda
- Bersihkan

Prasyarat

Untuk mengikuti tutorial ini, Anda harus memiliki semua Prasyarat Umum untuk Python yang diinstal, termasuk paket-paket berikut:

- Python 3.7 atau yang lebih baru
- `pip`
- `virtualenv`
- `awsebcli`

Kerangka kerja Django akan diinstal sebagai bagian dari tutorial.

> **Note**
>
> Membuat lingkungan dengan EB CLI memerlukan peran layanan. Peran layanan dapat dibuat dengan membuat lingkungan di konsol Elastic Beanstalk. Jika Anda tidak memiliki peran layanan, EB CLI mencoba membuatnya saat Anda menjalankan `eb create`.

Menyiapkan lingkungan virtual Python dan menginstal Django

Buat lingkungan virtual dengan `virtualenv` dan menggunakannya untuk menginstal Django dan dependensinya. Dengan menggunakan lingkungan virtual, Anda dapat mengetahui dengan tepat paket mana yang dibutuhkan aplikasi Anda, sehingga paket yang diperlukan diinstal pada EC2 instance Amazon yang menjalankan aplikasi Anda.

Langkah-langkah berikut menunjukkan perintah yang harus Anda masukkan untuk Windows dan sistem berbasis Unix, yang ditampilkan pada tab terpisah.

Untuk mempersiapkan lingkungan virtual Anda

1. Buat dan aktifkan lingkungan virtual bernama `eb-virt`.

   Unix-based systems

   ```
   ~$ virtualenv ~/eb-virt
   ```

   Windows

   ```
   C:\> virtualenv %HOMEPATH%\eb-virt
   ```

2. Aktifkan lingkungan virtual.

   Unix-based systems

   ```
   ~$ source ~/eb-virt/bin/activate
   (eb-virt) ~$
   ```

   Windows

   ```
   C:\>%HOMEPATH%\eb-virt\Scripts\activate
   (eb-virt) C:\>
   ```

   Anda akan melihat `(eb-virt)` ditambahkan ke prompt perintah Anda, yang menunjukkan bahwa Anda berada di lingkungan virtual.

   > **Note**
   >
   > Prompt perintah Linux ~$ di direktori home Anda ditunjukkan hingga akhir dari instruksi ini. Di Windows ini`C:\Users\`*USERNAME*>, di *USERNAME* mana nama login Windows Anda.

3. Gunakan `pip` untuk menginstal Django.

   ```
   (eb-virt)~$ pip install django==2.2
   ```

> **Note**
>
> Versi Django yang Anda pasang harus kompatibel dengan versi Python pada konfigurasi Phyton Elastic Beanstalk yang Anda pilih untuk men-deploy aplikasi Anda. Untuk informasi tentang deployment, lihat ??? dalam topik ini.
> Untuk informasi lebih lanjut tentang versi platform Python saat ini, lihat Python di dokumen Platform AWS Elastic Beanstalk .
> Untuk kesesuaian versi Django dengan Python, lihat Versi Python apa yang dapat saya gunakan dengan Django?

4. Untuk memverifikasi bahwa Django terpasang, masukkan berikut ini.

```
(eb-virt)~$ pip freeze
Django==2.2
...
```

Perintah ini mencantumkan semua paket yang diinstal di lingkungan virtual Anda. Kemudian, Anda menggunakan output dari perintah ini untuk mengonfigurasi proyek Anda untuk digunakan dengan Elastic Beanstalk.

Buat proyek Django

Sekarang Anda siap untuk membuat proyek Django dan menjalankannya pada mesin Anda, menggunakan lingkungan virtual.

> **Note**
>
> Tutorial ini menggunakan SQLite, yang merupakan mesin database yang disertakan dalam Python. Basis data di-deploy dengan file proyek Anda. Untuk lingkungan produksi, kami sarankan Anda menggunakan Amazon Relational Database Service (Amazon RDS), dan memisahkannya dari lingkungan Anda. Untuk informasi selengkapnya, lihat Menambahkan instans Amazon RDS DB ke lingkungan Python Elastic Beanstalk Anda.

Untuk menghasilkan aplikasi Django

1. Aktifkan lingkungan virtual Anda.

Unix-based systems

```
~$ source ~/eb-virt/bin/activate
(eb-virt) ~$
```

Windows

```
C:\>%HOMEPATH%\eb-virt\Scripts\activate
(eb-virt) C:\>
```

Anda akan melihat prefiks (`eb-virt`) ditambahkan ke prompt perintah Anda, yang menunjukkan bahwa Anda berada di lingkungan virtual.

> **Note**
>
> Prompt perintah Linux ~$ di direktori home Anda dan direktori home Linux ~/ ditunjukkan hingga akhir instruksi ini. Di Windows iniC:\Users*USERNAME*>, di *USERNAME* mana nama login Windows Anda.

2. Gunakan perintah `django-admin startproject` untuk membuat proyek Django bernama `ebdjango`.

```
(eb-virt)~$ django-admin startproject ebdjango
```

Perintah ini membuat sebuah situs Django standar bernama ebdjango dengan struktur direktori berikut.

```
~/ebdjango
  |-- ebdjango
  |   |-- __init__.py
  |   |-- settings.py
  |   |-- urls.py
  |   `-- wsgi.py
  `-- manage.py
```

3. Jalankan situs Django Anda secara lokal dengan `manage.py runserver`.

```
(eb-virt) ~$ cd ebdjango
```

```
(eb-virt) ~/ebdjango$ python manage.py runserver
```

4. Di peramban web, buka `http://127.0.0.1:8000/` untuk melihat situs.
5. Periksa log server untuk melihat output dari permintaan Anda. Untuk menghentikan server web dan kembali ke lingkungan virtual Anda, tekan Ctrl+C.

```
Django version 2.2, using settings 'ebdjango.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C.
[07/Sep/2018 20:14:09] "GET / HTTP/1.1" 200 16348
Ctrl+C
```

Konfigurasikan aplikasi Django Anda untuk Elastic Beanstalk

Sekarang setelah Anda memiliki situs bertenaga Django pada mesin lokal Anda, Anda dapat mengonfigurasi untuk deployment dengan Elastic Beanstalk.

Secara default, Elastic Beanstalk mencari file bernama `application.py` untuk memulai aplikasi Anda. Karena ini tidak ada di proyek Django yang Anda buat, Anda perlu membuat beberapa penyesuaian pada lingkungan aplikasi Anda. Anda juga harus mengatur variabel lingkungan sehingga modul aplikasi Anda dapat dimuat.

Mengonfigurasi situs Anda untuk Elastic Beanstalk

1. Aktifkan lingkungan virtual Anda.

   Unix-based systems

   ```
   ~/ebdjango$ source ~/eb-virt/bin/activate
   ```

   Windows

   ```
   C:\Users\USERNAME\ebdjango>%HOMEPATH%\eb-virt\Scripts\activate
   ```

2. Jalankan `pip freeze`, kemudian simpan output ke file bernama `requirements.txt`.

```
(eb-virt) ~/ebdjango$ pip freeze > requirements.txt
```

Elastic `requirements.txt` Beanstalk digunakan untuk menentukan paket mana yang akan diinstal EC2 pada instance yang menjalankan aplikasi Anda.

3. Membuat sebuah direktori bernama `.ebextensions`.

```
(eb-virt) ~/ebdjango$ mkdir .ebextensions
```

4. Di direktori `.ebextensions`, tambahkan file konfigurasi bernama `django.config` dengan teks berikut.

Example ~/ebdjango/.ebextensions/django.config

```
option_settings:
  aws:elasticbeanstalk:container:python:
    WSGIPath: ebdjango.wsgi:application
```

Pengaturan ini, `WSGIPath`, menentukan lokasi script WSGI yang digunakan Elastic Beanstalk untuk memulai aplikasi Anda.

> **Note**
>
> Jika Anda menggunakan versi platform Python Amazon Linux AMI (Amazon Linux 2 yang terdahulu), ganti nilai `WSGIPath` dengan `ebdjango/wsgi.py`. Nilai dalam contoh berfungsi dengan menggunakan server Gunicorn WSGI, yang tidak didukung pada versi platform Amazon Linux AMI.

5. Nonaktifkan lingkungan virtual Anda dengan perintah `deactivate`.

```
(eb-virt) ~/ebdjango$ deactivate
```

Aktifkan kembali lingkungan virtual setiap kali Anda perlu menambahkan paket ke aplikasi Anda atau menjalankan aplikasi secara lokal.

Men-deploy situs Anda dengan EB CLI

Anda telah menambahkan semua yang dibutuhkan untuk men-deploy aplikasi Anda pada Elastic Beanstalk. Direktori proyek Anda seharusnya terlihat seperti ini.

```
~/ebdjango/
|-- .ebextensions
|   `-- django.config
|-- ebdjango
|   |-- __init__.py
|   |-- settings.py
|   |-- urls.py
|   `-- wsgi.py
|-- db.sqlite3
|-- manage.py
`-- requirements.txt
```

Selanjutnya, Anda akan membuat lingkungan aplikasi dan men-deploy aplikasi Anda yang dikonfigurasi dengan Elastic Beanstalk.

Segera setelah deployment, Anda akan mengedit konfigurasi Django untuk menambahkan nama domain yang ditetapkan Elastic Beanstalk untuk aplikasi Anda ke `ALLOWED_HOSTS` Django. Kemudian Anda akan men-deploy kembali aplikasi Anda. Ini adalah persyaratan keamanan Django, dirancang untuk mencegah serangan header `Host` HTTP. Untuk informasi lebih lanjut, lihat Validasi header host.

Untuk membuat lingkungan dan men-deploy aplikasi Django Anda

> **Note**
>
> Tutorial ini menggunakan EB CLI sebagai mekanisme deployment, tetapi Anda juga dapat menggunakan konsol Elastic Beanstalk untuk men-deploy file .zip yang berisi konten proyek Anda.

1. Inisialisasi repositori EB CLI Anda dengan perintah eb init.

   ```
   ~/ebdjango$ eb init -p python-3.7 django-tutorial
   Application django-tutorial has been created.
   ```

Perintah ini membuat aplikasi bernama `django-tutorial`. Ini juga mengkonfigurasi repositori lokal Anda untuk membuat lingkungan dengan versi platform Python 3.7 terbaru.

2. (Opsional) Jalankan eb init lagi untuk mengonfigurasi key pair default sehingga Anda dapat menggunakan SSH untuk terhubung ke EC2 instance yang menjalankan aplikasi Anda.

```
~/ebdjango$ eb init
Do you want to set up SSH for your instances?
(y/n): y
Select a keypair.
1) my-keypair
2) [ Create new KeyPair ]
```

Pilih pasangan kunci jika Anda sudah memilikinya, atau ikuti prompt untuk membuatnya. Jika Anda tidak melihat prompt atau perlu mengubah pengaturan Anda nanti, jalankan eb init -i.

3. Membuat lingkungan dan men-deploy aplikasi Anda ke sana dengan eb create.

```
~/ebdjango$ eb create django-env
```

> **Note**
>
> Jika Anda melihat pesan kesalahan "peran layanan diperlukan", jalankan `eb create` secara interaktif (tanpa menentukan nama lingkungan) dan EB CLI membuat peran untuk Anda.

Perintah ini membuat lingkungan Elastic Beanstalk dengan beban seimbang bernama `django-env`. Membuat lingkungan membutuhkan waktu sekitar 5 menit. Saat Elastic Beanstalk membuat sumber daya yang diperlukan untuk menjalankan aplikasi Anda, Elastic Beanstalk mengeluarkan pesan informasi yang disampaikan EB CLI ke terminal Anda.

4. Ketika proses pembuatan lingkungan selesai, temukan nama domain lingkungan baru Anda dengan menjalankan eb status.

```
~/ebdjango$ eb status
Environment details for: django-env
  Application name: django-tutorial
  ...
  CNAME: eb-django-app-dev.elasticbeanstalk.com
```

```
  ...
```

Nama domain lingkungan Anda adalah nilai dari properti `CNAME`.

5. Buka file `settings.py` dalam direktori `ebdjango`. Temukan pengaturan `ALLOWED_HOSTS`, dan kemudian tambahkan nama domain aplikasi yang Anda temukan di langkah sebelumnya ke nilai pengaturan. Jika Anda tidak dapat menemukan pengaturan ini dalam file, tambahkan itu ke baris baru.

```
...
ALLOWED_HOSTS = ['eb-django-app-dev.elasticbeanstalk.com']
```

6. Simpan file, dan kemudian deploy aplikasi Anda dengan menjalankan eb deploy. Ketika Anda menjalankan eb deploy, EB CLI memaketkan konten direktori proyek Anda dan men-deploy ke lingkungan Anda.

```
~/ebdjango$ eb deploy
```

> **Note**
>
> Jika Anda menggunakan Git dengan proyek Anda, lihat Menggunakan EB CLI dengan Git.

7. Ketika proses pembaruan lingkungan selesai, buka situs web Anda dengan eb open.

```
~/ebdjango$ eb open
```

Ini akan membuka jendela peramban menggunakan nama domain yang dibuat untuk aplikasi Anda. Anda seharusnya melihat situs web Django yang sama seperti yang Anda telah buat dan uji secara lokal.

Jika aplikasi tidak berjalan, atau menerima pesan kesalahan, lihat Pemecahan masalah deployment untuk bantuan tentang cara menentukan penyebab kesalahan.

Jika Anda benar-benar melihat aplikasi Anda berjalan, maka selamat, Anda telah men-deploy aplikasi Django pertama Anda dengan Elastic Beanstalk!

Perbarui aplikasi Anda

Sekarang setelah menjalankan aplikasi Elastic Beanstalk, Anda dapat memperbarui dan men-deploy kembali aplikasi Anda atau konfigurasinya, dan Elastic Beanstalk melakukan pekerjaan memperbarui instans dan memulai versi aplikasi baru Anda.

Untuk contoh ini, kita akan mengaktifkan konsol admin Django dan mengonfigurasi beberapa pengaturan lain.

Mengubah pengaturan situs Anda

Secara default, situs web Django Anda menggunakan zona waktu UTC untuk menampilkan waktu. Anda dapat mengubah ini dengan menentukan zona waktu di `settings.py`.

Untuk mengubah zona waktu situs Anda

1. Modifikasi pengaturan `TIME_ZONE` dalam `settings.py`.

   Example ~/ .py ebdjango/ebdjango/settings

   ```
   ...
   # Internationalization
   LANGUAGE_CODE = 'en-us'
   TIME_ZONE = 'US/Pacific'
   USE_I18N = True
   USE_L10N = True
   USE_TZ = True
   ```

   Untuk daftar zona waktu, kunjungi halaman ini.

2. Men-deploy aplikasi ke lingkungan Elastic Beanstalk Anda.

   ```
   ~/ebdjango/$ eb deploy
   ```

Membuat administrator situs

Anda dapat membuat administrator situs untuk aplikasi Django Anda untuk mengakses konsol admin langsung dari situs web. Detail login administrator disimpan dengan aman di citra basis data lokal yang disertakan dalam proyek default yang dihasilkan Django.

Untuk membuat administrator situs

1. Inisialisasi basis data lokal aplikasi Django Anda.

```
(eb-virt) ~/ebdjango$ python manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, sessions
Running migrations:
  Applying contenttypes.0001_initial... OK
  Applying auth.0001_initial... OK
  Applying admin.0001_initial... OK
  Applying admin.0002_logentry_remove_auto_add... OK
  Applying admin.0003_logentry_add_action_flag_choices... OK
  Applying contenttypes.0002_remove_content_type_name... OK
  Applying auth.0002_alter_permission_name_max_length... OK
  Applying auth.0003_alter_user_email_max_length... OK
  Applying auth.0004_alter_user_username_opts... OK
  Applying auth.0005_alter_user_last_login_null... OK
  Applying auth.0006_require_contenttypes_0002... OK
  Applying auth.0007_alter_validators_add_error_messages... OK
  Applying auth.0008_alter_user_username_max_length... OK
  Applying auth.0009_alter_user_last_name_max_length... OK
  Applying sessions.0001_initial... OK
```

2. Jalankan `manage.py createsuperuser` untuk membuat administrator.

```
(eb-virt) ~/ebdjango$ python manage.py createsuperuser
Username: admin
Email address: me@mydomain.com
Password: ********
Password (again): ********
Superuser created successfully.
```

3. Untuk memberitahu Django tempat menyimpan file statis, tentukan `STATIC_ROOT` dalam `settings.py`.

   Example ~/ .py ebdjango/ebdjango/settings

```
# Static files (CSS, JavaScript, Images)
# https://docs.djangoproject.com/en/2.2/howto/static-files/
STATIC_URL = '/static/'
STATIC_ROOT = 'static'
```

4. Jalankan `manage.py collectstatic` untuk mengisi `static` direktori dengan aset statis (JavaScript, CSS, dan gambar) untuk situs admin.

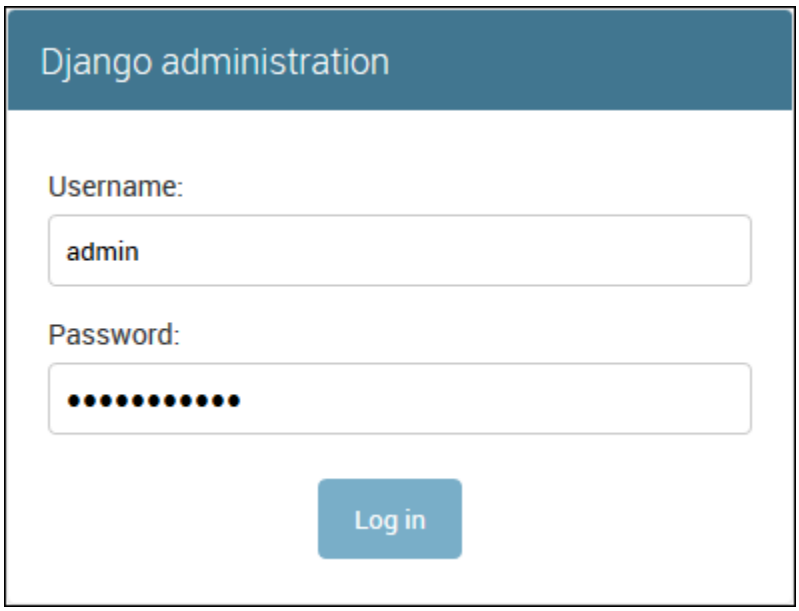
```
(eb-virt) ~/ebdjango$ python manage.py collectstatic
119 static files copied to ~/ebdjango/static
```

5. Men-deploy aplikasi Anda.

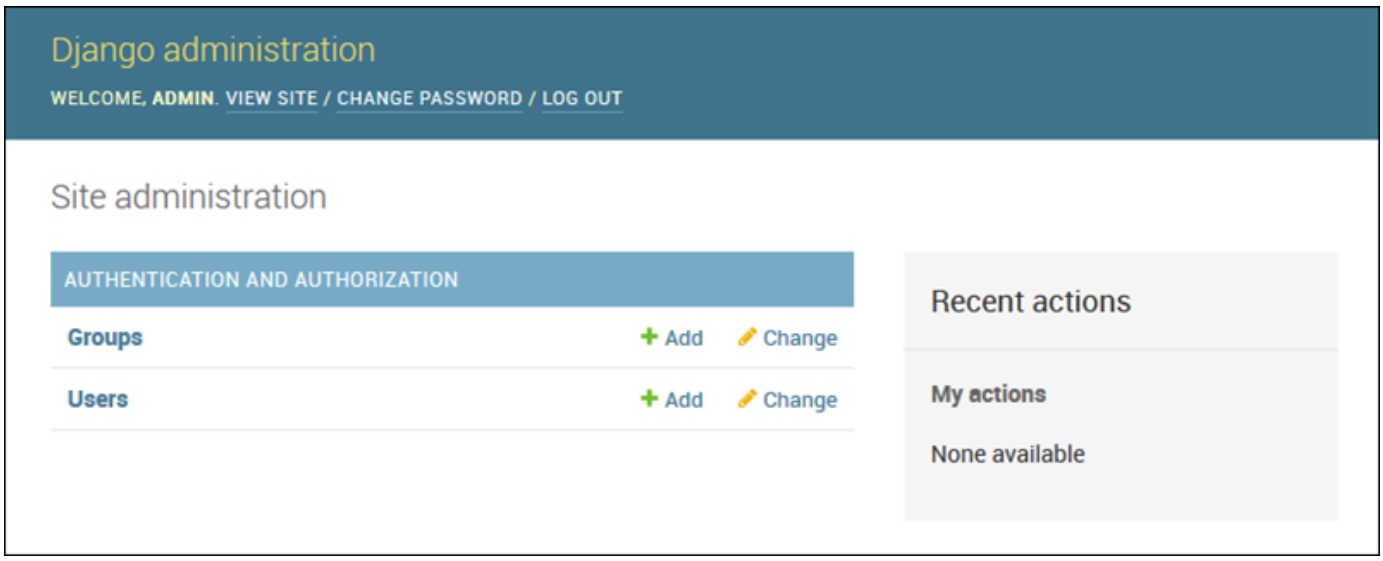
```
~/ebdjango$ eb deploy
```

6. Lihat konsol admin dengan membuka situs di peramban Anda, tambahkan `/admin/` ke URL situs, seperti berikut ini.

```
http://djang-env.p33kq46sfh.us-west-2.elasticbeanstalk.com/admin/
```



7. Masuk dengan nama pengguna dan kata sandi yang Anda konfigurasikan pada langkah 2.



Anda dapat menggunakan prosedur lokal yang serupa updating/testing diikuti oleheb deploy. Elastic Beanstalk melakukan pekerjaan pembaruan server langsung Anda, sehingga Anda dapat fokus pada pengembangan aplikasi bukan administrasi server!

Menambahkan file konfigurasi migrasi basis data

Anda dapat menambahkan perintah ke skrip `.ebextensions` yang dijalankan saat situs Anda diperbarui. Hal ini memungkinkan Anda untuk menghasilkan migrasi basis data secara otomatis.

Untuk menambahkan langkah migrasi saat aplikasi Anda di-deploy

1. Buat file lonfigurasi bernama `db-migrate.config` dengan konten berikut.

   Example ~/ebdjango/.ebextensions/db-migrate.config

   ```
   container_commands:
     01_migrate:
       command: "source /var/app/venv/*/bin/activate && python3 manage.py migrate"
       leader_only: true
   option_settings:
     aws:elasticbeanstalk:application:environment:
       DJANGO_SETTINGS_MODULE: ebdjango.settings
   ```

   File konfigurasi ini mengaktifkan lingkungan virtual server dan menjalankan `manage.py migrate` perintah selama proses penyebaran, sebelum memulai aplikasi Anda. Karena berjalan sebelum aplikasi dimulai, Anda juga harus mengonfigurasi variabel lingkungan

`DJANGO_SETTINGS_MODULE` secara eksplisit (biasanya `wsgi.py` akan mengurus hal ini untuk Anda selama startup). Menentukan `leader_only: true` dalam perintah akan memastikan bahwa itu hanya akan dijalankan sekali ketika Anda men-deploy ke beberapa instans.

2. Men-deploy aplikasi Anda.

```
~/ebdjango$ eb deploy
```

Bersihkan

Untuk menghemat jam instans dan AWS sumber daya lainnya di antara sesi pengembangan, hentikan lingkungan Elastic Beanstalk Anda dengan. eb terminate

```
~/ebdjango$ eb terminate django-env
```

Perintah ini mengakhiri lingkungan dan semua sumber AWS daya yang berjalan di dalamnya. Perintah tersebut bagaimana pun juga tidak akan menghapus aplikasi, sehingga Anda selalu dapat membuat lebih banyak lingkungan dengan konfigurasi yang sama dengan menjalankan eb create lagi.

Jika Anda telah selesai dengan aplikasi sampel, Anda juga dapat menghapus folder proyek dan lingkungan virtual.

```
~$ rm -rf ~/eb-virt
~$ rm -rf ~/ebdjango
```

Langkah selanjutnya

Untuk informasi selengkapnya tentang Django, termasuk tutorial secara mendalam, lihat Dokumentasi resmi.

Jika Anda ingin mencoba kerangka kerja web Python lain, lihat Men-deploy aplikasi Flask ke Elastic Beanstalk.

Menambahkan instans Amazon RDS DB ke lingkungan Python Elastic Beanstalk Anda

Topik ini memberikan instruksi untuk membuat Amazon RDS menggunakan konsol Elastic Beanstalk. Anda dapat menggunakan instans DB Amazon Relational Database Service (Amazon RDS) untuk menyimpan data yang dikumpulkan dan dimodifikasi oleh aplikasi Anda. Basis data dapat digabungkan ke lingkungan Anda dan dikelola oleh Elastic Beanstalk, atau dapat dibuat sebagai dipisahkan dan dikelola secara eksternal oleh layanan lain. Dalam instruksi ini database digabungkan dengan lingkungan Anda dan dikelola oleh Elastic Beanstalk. Untuk informasi lebih lanjut tentang mengintegrasikan Amazon RDS dengan Elastic Beanstalk, lihat. Menambahkan basis data ke lingkungan Elastic Beanstalk Anda

Bagian-bagian

- Menambahkan instans DB ke lingkungan Anda
- Mengunduh driver
- Menghubungkan ke basis data

Menambahkan instans DB ke lingkungan Anda

Untuk menambahkan instans DB ke lingkungan Anda

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Di kategori konfigurasi Basis data, pilih Edit.
5. Pilih mesin DB, dan masukkan nama pengguna dan kata sandi.
6. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Menambahkan instans DB memakan waktu sekitar 10 menit. Ketika pembaruan lingkungan selesai, nama host instans DB dan informasi koneksi lainnya tersedia untuk aplikasi Anda melalui properti lingkungan berikut:

Nama properti	Deskripsi	Nilai properti
`RDS_HOSTNAME`	Nama host instans DB.	Di tab Konektivitas & keamanan di konsol Amazon RDS: Titik akhir.
`RDS_PORT`	Port tempat instans DB menerima koneksi. Nilai default bervariasi di antara mesin DB.	Di tab Konektivitas & keamanan di konsol Amazon RDS: Port.
`RDS_DB_NAME`	Nama basis data, **ebdb**.	Di tab Konfigurasi di konsol Amazon RDS: Nama DB.
`RDS_USERNAME`	Nama pengguna yang Anda konfigurasi untuk basis data Anda.	Di tab Konfigurasi di konsol Amazon RDS: Nama pengguna utama.
`RDS_PASSWORD`	Kata sandi yang Anda konfigurasi untuk basis data Anda.	Tidak tersedia untuk referensi di konsol Amazon RDS.

Untuk informasi selengkapnya tentang mengonfigurasi instance database yang digabungkan dengan lingkungan Elastic Beanstalk, lihat. Menambahkan basis data ke lingkungan Elastic Beanstalk Anda

Mengunduh driver

Menambahkan driver basis data ke file persyaratan proyek Anda.

Example requirements.txt – Django dengan MySQL

```
Django==2.2
mysqlclient==2.0.3
```

Paket driver umum untuk Python

- MySQL – `mysqlclient`
- PostgreSQL – `psycopg2`

- Oracle – `cx_Oracle`
- SQL Server – `adodbapi`

Untuk informasi selengkapnya lihat Python DatabaseInterfaces dan Django 2.2 - basis data yang didukung.

Menghubungkan ke basis data

Elastic Beanstalk memberikan informasi koneksi untuk instans DB terlampir di properti lingkungan. Gunakan `os.environ['VARIABLE']` untuk membaca properti dan mengonfigurasi koneksi basis data.

Example File pengaturan Django – kamus BASIS DATA

```
import os

if 'RDS_HOSTNAME' in os.environ:
    DATABASES = {
        'default': {
            'ENGINE': 'django.db.backends.mysql',
            'NAME': os.environ['RDS_DB_NAME'],
            'USER': os.environ['RDS_USERNAME'],
            'PASSWORD': os.environ['RDS_PASSWORD'],
            'HOST': os.environ['RDS_HOSTNAME'],
            'PORT': os.environ['RDS_PORT'],
        }
    }
```

Alat Python dan sumber daya

Ada beberapa tempat yang dapat dikunjungi untuk mendapatkan bantuan tambahan ketika mengembangkan aplikasi Python Anda:

Sumber Daya	Deskripsi
AWS SDK untuk Python (Boto3) pada GitHub	Instal Boto3 bersumber dari. GitHub

Sumber Daya	Deskripsi
AWS SDK untuk Python (Boto3) beranda	AWS SDK untuk Python (Boto3) Beranda.
Pusat Pengembang Python	Tempat yang lengkap mengakomodasikan segala kebutuhan Anda mulai dari kode sampel, dokumentasi, alat, dan sumber daya tambahan.

Menyebarkan aplikasi Ruby dengan Elastic Beanstalk

Bab ini memberikan instruksi untuk mengkonfigurasi dan menyebarkan aplikasi web Ruby Anda ke. AWS Elastic Beanstalk Elastic Beanstalk memudahkan penerapan, pengelolaan, dan skala aplikasi web Ruby Anda menggunakan Amazon Web Services.

Anda dapat menerapkan aplikasi Anda hanya dalam beberapa menit menggunakan Elastic Beanstalk Command Line Interface (EB CLI) atau dengan menggunakan konsol Elastic Beanstalk. Setelah Anda menerapkan aplikasi Elastic Beanstalk Anda, Anda dapat terus menggunakan EB CLI untuk mengelola aplikasi dan lingkungan Anda, atau Anda dapat menggunakan konsol Elastic Beanstalk, atau. AWS CLI APIs

Bab ini juga memberikan step-by-step instruksi untuk menyebarkan aplikasi sampel ke Elastic Beanstalk menggunakan EB CLI, dan kemudian memperbarui aplikasi untuk menggunakan kerangka kerja aplikasi web Rails dan Sinatra.

Topik dalam Bab ini mengasumsikan bahwa Anda memiliki pengetahuan tentang lingkungan Elastic Beanstalk. Jika Anda belum pernah menggunakan Elastic Beanstalk sebelumnya, cobalah tutorial memulai untuk mempelajari dasarnya.

Topik

- Menyiapkan lingkungan pengembangan Ruby Anda untuk Elastic Beanstalk
- Menggunakan platform Ruby Elastic Beanstalk
- Men-deploy aplikasi rails ke Elastic Beanstalk
- Men-deploy aplikasi sinatra ke Elastic Beanstalk
- Menambahkan instans Amazon RDS DB ke lingkungan Ruby Elastic Beanstalk Anda

Menyiapkan lingkungan pengembangan Ruby Anda untuk Elastic Beanstalk

Bab ini memberikan instruksi untuk menyiapkan lingkungan pengembangan Ruby untuk menguji aplikasi Anda secara lokal sebelum menerapkannya. AWS Elastic Beanstalk Ini juga merujuk situs web yang menyediakan instruksi instalasi untuk alat yang berguna.

Untuk mengikuti prosedur di panduan ini, Anda memerlukan terminal atau shell baris perintah untuk menjalankan perintah. Perintah ditampilkan dalam daftar yang diawali dengan simbol prompt ($) dan nama direktori saat ini, bila perlu.

```
~/eb-project$ this is a command
this is output
```

Di Linux dan macOS, Anda dapat menggunakan shell dan manajer paket pilihan Anda. Pada Windows Anda dapat menginstal Windows Subsystem untuk Linux untuk mendapatkan versi Windows terintegrasi dari Ubuntu dan Bash.

Bagian-bagian

- Menginstal Ruby
- Memasang AWS SDK for Ruby
- Menginstal IDE atau editor teks

Menginstal Ruby

Instal GCC jika Anda tidak memiliki C compiler. Di Ubuntu, gunakan `apt`.

```
~$ sudo apt install gcc
```

Di Amazon Linux, gunakan `yum`.

```
~$ sudo yum install gcc
```

Instal RVM untuk mengelola instalasi bahasa Ruby pada mesin Anda. Gunakan perintah di rvm.io untuk mendapatkan kunci proyek dan menjalankan skrip instalasi.

```
~$ gpg2 --recv-keys key1 key2
~$ curl -sSL https://get.rvm.io | bash -s stable
```

Skrip ini menginstal RVM dalam folder bernama `.rvm` di direktori pengguna Anda, dan memodifikasi profil shell Anda untuk memuat skrip setup setiap kali Anda membuka terminal baru. Muat skrip secara manual untuk memulai.

```
~$ source ~/.rvm/scripts/rvm
```

Gunakan `rvm get head` untuk mendapatkan versi terbaru.

```
~$ rvm get head
```

Melihat versi Ruby yang tersedia.

```
~$ rvm list known
```

Periksa Ruby di dokumen Platform AWS Elastic Beanstalk untuk menemukan versi terbaru Ruby yang tersedia pada platform Elastic Beanstalk. Instal versi tersebut.

```
~$ rvm install 3.2
```

Uji instalasi Ruby Anda.

```
~$ ruby --version
```

Memasang AWS SDK for Ruby

Jika Anda perlu mengelola AWS sumber daya dari dalam aplikasi Anda, instal file AWS SDK untuk Ruby. Misalnya, dengan SDK for Ruby, Anda dapat menggunakan Amazon DynamoDB (DynamoDB) untuk menyimpan informasi pengguna dan sesi tanpa membuat basis data relasional.

Pasang SDK for Ruby dan dependensinya dengan perintah `gem`.

```
$ gem install aws-sdk
```

Kunjungi beranda AWS SDK untuk Ruby untuk informasi lebih lanjut dan petunjuk instalasi.

Menginstal IDE atau editor teks

Lingkungan pengembangan terintegrasi (IDEs) menyediakan berbagai fitur yang memfasilitasi pengembangan aplikasi. Jika Anda belum menggunakan IDE untuk pengembangan Ruby, coba Aptana RubyMine dan lihat mana yang paling cocok untuk Anda.

- Instal Aptana
- RubyMine

> **Note**
>
> IDE mungkin saja menambahkan file ke folder proyek yang mungkin tidak ingin Anda masukkan ke kontrol sumber. Untuk mencegah memasukkan file-file ini ke kontrol sumber, gunakan `.gitignore` atau padanan alat kontrol sumber Anda.

Jika Anda baru ingin memulai coding dan tidak memerlukan semua fitur IDE, pertimbangkan untuk menginstal Sublime Text.

Menggunakan platform Ruby Elastic Beanstalk

Topik ini menjelaskan cara mengonfigurasi, membangun, dan menjalankan aplikasi Ruby Anda di Elastic Beanstalk.

AWS Elastic Beanstalk mendukung sejumlah cabang platform untuk berbagai versi bahasa pemrograman Ruby. Lihat Ruby di dokumen AWS Elastic Beanstalk Platform untuk daftar lengkapnya.

Aplikasi web Ruby dapat berjalan di belakang server proxy NGINX di bawah server aplikasi Puma. Jika Anda menggunakan RubyGems, Anda dapat menyertakan `Gemfile`dalam bundel sumber Anda untuk menginstal paket selama penerapan.

Konfigurasi server aplikasi

Elastic Beanstalk menginstal server aplikasi Puma berdasarkan cabang platform Ruby yang Anda pilih saat Anda membuat lingkungan Anda. Untuk informasi selengkapnya tentang komponen yang disediakan bersama versi platform Ruby, lihat Platform yang Didukung dalam panduan AWS Elastic Beanstalk Platform.

Anda dapat mengonfigurasi aplikasi Anda dengan server Puma yang Anda sediakan sendiri. Ini memberikan opsi untuk menggunakan versi Puma selain yang sudah diinstal sebelumnya dengan cabang platform Ruby. Anda juga dapat mengonfigurasi aplikasi Anda untuk menggunakan server aplikasi yang berbeda, seperti Passenger. Untuk melakukannya, Anda harus menyertakan dan menyesuaikan a `Gemfile` dalam penerapan Anda. Anda juga diharuskan mengonfigurasi a `Procfile` untuk memulai server aplikasi. Untuk informasi selengkapnya lihat Mengonfigurasi proses aplikasi dengan Procfile.

Opsi konfigurasi lainnya

Elastic Beanstalk menyediakan opsi konfigurasi yang dapat Anda gunakan untuk menyesuaikan perangkat lunak yang berjalan pada instans Amazon Elastic Compute Cloud (EC2Amazon) di lingkungan Elastic Beanstalk Anda. Anda dapat mengonfigurasi variabel lingkungan yang diperlukan aplikasi Anda, mengaktifkan rotasi log ke Amazon S3, dan memetakan folder dalam sumber aplikasi Anda yang berisi file statis ke jalur yang disajikan server proksi. Platform ini juga telah menentukan beberapa variabel lingkungan umum sebelumnya yang terkait Rails dan Rack untuk kemudahan dalam penemuan dan penggunaannya.

Pilihan konfigurasi tersedia di konsol Elastic Beanstalk untuk memodifikasi konfigurasi dari lingkungan yang sedang berjalan. Agar Anda tidak kehilangan konfigurasi lingkungan ketika mengakhirinya, Anda dapat menggunakan konfigurasi tersimpan untuk menyimpan pengaturan dan kemudian menerapkannya ke lingkungan lain.

Untuk menyimpan pengaturan di kode sumber, Anda dapat menyertakan file konfigurasi. Pengaturan di file konfigurasi diterapkan setiap kali Anda membuat lingkungan atau men-deploy aplikasi Anda. Anda juga dapat menggunakan file konfigurasi untuk menginstal paket, menjalankan penulisan, dan melakukan operasi penyesuaian instans lainnya selama deployment.

Pengaturan yang diterapkan di konsol Elastic Beanstalk mengganti pengaturan yang sama di file konfigurasi, jika ada. Hal tersebut memungkinkan Anda memiliki pengaturan default di file konfigurasi, dan mengganti keduanya dengan pengaturan khusus lingkungan di konsol tersebut. Untuk informasi selengkapnya tentang yang diutamakan, dan metode lain untuk mengubah pengaturan, lihat Opsi konfigurasi.

Untuk detail tentang berbagai cara memperluas platform berbasis Linux Elastic Beanstalk, lihat the section called “Memperluas platform Linux”.

Mengonfigurasi lingkungan Ruby Anda

Anda dapat menggunakan konsol Elastic Beanstalk untuk mengaktifkan rotasi log ke Amazon S3, mengonfigurasi variabel yang dapat dibaca aplikasi Anda dari lingkungan.

Untuk mengakses pengaturan konfigurasi perangkat lunak untuk lingkungan Anda

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Dalam kategori konfigurasi Pembaruan, pemantauan, dan pencatatan, pilih Edit.

Opsi log

Bagian opsi Log memiliki dua pengaturan:

- Profil instans– Menentukan profil instans yang memiliki izin untuk mengakses bucket Amazon S3 yang terkait dengan aplikasi Anda.
- Aktifkan rotasi file log ke Amazon S3 — Menentukan apakah file log untuk EC2 instans Amazon aplikasi Anda disalin ke bucket Amazon S3 yang terkait dengan aplikasi Anda.

File statis

Untuk meningkatkan kinerja, Anda dapat menggunakan bagian File statis untuk mengkonfigurasi server proxy untuk melayani file statis (misalnya, HTML atau gambar) dari satu set direktori di dalam aplikasi web Anda. Untuk setiap direktori, Anda mengatur jalur virtual ke pemetaan direktori. Saat server proksi menerima permintaan untuk file di jalur yang ditentukan, server langsung menyajikan file daripada merutekan permintaan ke aplikasi Anda.

Untuk detail tentang mengonfigurasi file statis menggunakan file konfigurasi atau konsol Elastic Beanstalk, lihat. the section called “File statis”

Secara default, server proxy di lingkungan Ruby dikonfigurasi untuk melayani file statis sebagai berikut:

- File dalam `public` folder disajikan dari `/public` jalur dan root domain (/path).
- File di `public/assets` subfolder disajikan dari `/assets` jalur.

Contoh berikut menggambarkan cara kerja konfigurasi default:

- Jika sumber aplikasi Anda berisi file bernama `logo.png` dalam folder bernama`public`, server proxy menyajikannya kepada pengguna dari *`subdomain`*`.elasticbeanstalk.com/public/logo.png` dan*`subdomain`*`.elasticbeanstalk.com/logo.png`.
- Jika sumber aplikasi Anda berisi file bernama `logo.png` dalam folder bernama `assets` di dalam `public` folder, server proxy menyajikannya*`subdomain`*`.elasticbeanstalk.com/assets/logo.png`.

Anda dapat mengonfigurasi pemetaan tambahan untuk file statis. Untuk informasi selengkapnya, lihat Namespace konfigurasi Ruby dalam topik ini.

> **Note**
>
> Untuk versi platform sebelum Ruby 2.7 AL2 versi 3.3.7, konfigurasi server proxy Elastic Beanstalk nginx default tidak mendukung penyajian file statis dari root domain (). *subdomain*`.elasticbeanstalk.com/` Versi platform ini dirilis pada 21 Oktober 2021. Untuk informasi selengkapnya lihat Versi platform baru - Ruby di Catatan AWS Elastic Beanstalk Rilis.

Properti lingkungan

Bagian Properti Lingkungan memungkinkan Anda menentukan pengaturan konfigurasi lingkungan pada EC2 instance Amazon yang menjalankan aplikasi Anda. Properti lingkungan diberikan sebagai pasangan nilai kunci ke aplikasi.

Platform Ruby menetapkan properti berikut untuk konfigurasi lingkungan:

- BUNDLE_WITHOUT – Daftar grup yang dipisahkan oleh titik dua untuk diabaikan ketika menginstal dependensi dari Gemfile.
- BUNDLER_DEPLOYMENT_MODE – Atur ke `true` (default) untuk menginstal dependensi dalam mode deployment menggunakan Bundler. Atur ke `false` untuk menjalankan `bundle install` dalam mode pengembangan.

  > **Note**
  >
  > Properti lingkungan ini tidak ditentukan pada cabang platform Ruby Amazon Linux AMI (Amazon Linux 2 yang terdahulu).

- RAILS_SKIP_ASSET_COMPILATION – Atur ke `true` agar tidak menjalankan `rake assets:precompile` selama deployment.
- RAILS_SKIP_MIGRATIONS – Atur ke `true` agar tidak menjalankan `rake db:migrate` selama deployment.
- RACK_ENV – Menentukan tahap lingkungan untuk Rack. Misalnya, `development`, `production`, atau `test`.

Di dalam lingkungan Ruby yang berjalan di Elastic Beanstalk, variabel lingkungan dapat diakses menggunakan objek `ENV`. Sebagai contoh, Anda dapat membaca properti bernama `API_ENDPOINT` ke variabel dengan kode berikut:

```
endpoint = ENV['API_ENDPOINT']
```

Lihat Variabel lingkungan dan pengaturan perangkat lunak lainnya untuk informasi selengkapnya.

Namespace konfigurasi Ruby

Anda dapat menggunakan file konfigurasi untuk mengatur opsi konfigurasi dan melakukan tugas-tugas konfigurasi instans lain selama deployment. Opsi konfigurasi dapat berupa platform khusus atau berlaku untuk semua platform di layanan Elastic Beanstalk secara keseluruhan. Opsi konfigurasi diatur ke dalam ruang nama.

Anda dapat menggunakan namespace `aws:elasticbeanstalk:environment:proxy:staticfiles` untuk mengonfigurasi proksi lingkungan untuk menyajikan file statis. Anda menentukan pemetaan jalur virtual ke direktori aplikasi.

Platform Ruby tidak menentukan namespace tertentu platform. Sebagai gantinya, platform Ruby menentukan properti lingkungan untuk opsi umum Rails dan Rack.

File konfigurasi berikut menentukan opsi file statis yang memetakan direktori bernama `staticimages` ke jalur `/images`, menetapkan setiap properti lingkungan yang ditentukan platform, dan menetapkan properti lingkungan tambahan bernama `LOGGING`.

Example .ebextensions/ruby-settings.config

```
option_settings:
  aws:elasticbeanstalk:environment:proxy:staticfiles:
    /images: staticimages
  aws:elasticbeanstalk:application:environment:
    BUNDLE_WITHOUT: test
    BUNDLER_DEPLOYMENT_MODE: true
    RACK_ENV: development
    RAILS_SKIP_ASSET_COMPILATION: true
    RAILS_SKIP_MIGRATIONS: true
    LOGGING: debug
```

> **Note**
>
> Properti lingkungan `BUNDLER_DEPLOYMENT_MODE` dan namespace `aws:elasticbeanstalk:environment:proxy:staticfiles` tidak tetapditentukan pada cabang platform Ruby Amazon Linux AMI (Amazon Linux 2 yang terdahulu).

Elastic Beanstalk memberikan banyak opsi konfigurasi untuk menyesuaikan lingkungan Anda. Selain file konfigurasi, Anda juga dapat mengatur opsi konfigurasi menggunakan konsol tersebut, konfigurasi tersimpan, EB CLI, atau AWS CLI. Untuk informasi selengkapnya, lihat Opsi konfigurasi.

Menginstal paket dengan Gemfile di Elastic Beanstalk

Untuk menggunakan RubyGems untuk menginstal paket yang dibutuhkan aplikasi Anda, sertakan `Gemfile` file di root sumber proyek Anda.

Example Gemfile

```
source "https://rubygems.org"
gem 'sinatra'
gem 'json'
gem 'rack-parser'
```

Saat file `Gemfile` ada, Elastic Beanstalk menjalankan `bundle install` untuk menginstal dependensi. Untuk informasi selengkapnya, lihat halaman Gemfiles dan Bundle di situs web Bundler.io.

> **Note**
>
> Anda dapat menggunakan versi Puma yang berbeda selain default yang sudah diinstal sebelumnya dengan platform Ruby. Untuk melakukannya, sertakan entri dalam `Gemfile` yang menentukan versi. Anda juga dapat menentukan server aplikasi yang berbeda, seperti Penumpang, dengan menggunakan yang disesuaikan`Gemfile`.
> Untuk kedua kasus ini, Anda harus mengonfigurasi a `Procfile` untuk memulai server aplikasi.
> Untuk informasi selengkapnya lihat Mengonfigurasi proses aplikasi dengan Procfile.

Mengkonfigurasi proses aplikasi dengan Procfile pada Elastic Beanstalk.

Untuk menentukan perintah yang memulai aplikasi Ruby Anda, sertakan file bernama `Procfile` pada akar paket sumber Anda.

> **Note**
>
> Elastic Beanstalk tidak mendukung fitur ini pada cabang platform Ruby Amazon Linux AMI (Amazon Linux 2 yang terdahulu). Cabang platform dengan nama yang mengandung dengan Puma atau dengan Passenger, terlepas dari versi Ruby-nya, mendahului Amazon Linux 2 dan tidak mendukung fitur `Procfile`.

Untuk detail tentang menulis dan menggunakan a`Procfile`, lihatBuildfile dan Procfile.

Bila Anda tidak menyediakan`Procfile`, Elastic Beanstalk menghasilkan default. `Procfile` Jika Anda `Gemfile` menyertakan Puma, Elastic Beanstalk mengasumsikan Anda ingin menggunakan versi Puma yang Anda sediakan dan menghasilkan default berikut. `Procfile`

```
web: bundle exec puma -C /opt/elasticbeanstalk/config/private/pumaconf.rb
```

Jika Anda `Gemfile` tidak menyertakan Puma, Elastic Beanstalk mengasumsikan Anda menggunakan server aplikasi Puma yang sudah diinstal sebelumnya dan menghasilkan default berikut. `Procfile` Di cabang platform Amazon Linux 2 Ruby, Elastic Beanstalk selalu menghasilkan default `Procfile` berikut jika Anda tidak menyediakan file. `Procfile`

```
web: puma -C /opt/elasticbeanstalk/config/private/pumaconf.rb
```

> **Note**
>
> Pada 10 Oktober 2024, cabang platform Ruby Amazon Linux 2 terakhir dihentikan. Semua cabang platform Ruby yang didukung saat ini didasarkan pada Amazon Linux 2023. Untuk informasi tentang migrasi, lihatMigrasi dari Amazon Linux 2 ke Amazon Linux 2023.

Jika Anda ingin menggunakan server aplikasi Passenger, gunakan file contoh berikut untuk mengonfigurasi lingkungan Ruby Anda untuk menginstal dan menggunakan Passenger.

1. Gunakan file contoh ini untuk menginstal Passenger.

   Example Gemfile

   ```
   source 'https://rubygems.org'
   gem 'passenger'
   ```

2. Gunakan file contoh ini untuk menginstruksikan Elastic Beanstalk untuk memulai Passenger.

   Example Procfile

   ```
   web: bundle exec passenger start /var/app/current --socket /var/run/puma/my_app.sock
   ```

> **Note**
>
> Anda tidak perlu mengubah apa pun dalam konfigurasi server proksi nginx untuk menggunakan Passenger. Untuk menggunakan server aplikasi lain, Anda mungkin perlu menyesuaikan konfigurasi nginx untuk meneruskan permintaan ke aplikasi Anda dengan benar.

Men-deploy aplikasi rails ke Elastic Beanstalk

Rails adalah open source, model-view-controller (MVC) framework untuk Ruby. Tutorial ini memandu Anda melalui proses menghasilkan aplikasi Rails dan menyebarkannya ke lingkungan AWS Elastic Beanstalk .

Bagian-bagian

- Prasyarat
- Pengetahuan Elastic Beanstalk dasar
- Meluncurkan lingkungan Elastic Beanstalk
- Menginstal rails dan menghasilkan sebuah situs web
- Mengonfigurasi pengaturan rails
- Deploy aplikasi Anda
- Pembersihan
- Langkah selanjutnya

Prasyarat

Pengetahuan Elastic Beanstalk dasar

Tutorial ini mengasumsikan Anda memiliki pengetahuan tentang operasi Elastic Beanstalk dasar dan konsol Elastic Beanstalk. Jika belum, ikuti petunjuk di Pelajari cara memulai dengan Elastic Beanstalk untuk meluncurkan lingkungan Elastic Beanstalk pertama Anda.

Baris perintah

Untuk mengikuti prosedur dalam panduan ini, Anda memerlukan terminal atau shell baris perintah untuk menjalankan perintah. Perintah ditampilkan dalam daftar yang diawali dengan simbol prompt ($) dan nama direktori saat ini, bila perlu.

```
~/eb-project$ this is a command
this is output
```

Di Linux dan macOS, Anda dapat menggunakan shell dan manajer paket pilihan Anda. Pada Windows Anda dapat menginstal Windows Subsystem untuk Linux untuk mendapatkan versi Windows terintegrasi dari Ubuntu dan Bash.

Dependensi Rails

Kerangka Rails 6.1.4.1 memiliki dependensi berikut. Pastikan Anda telah menginstal semuanya.

- Ruby 2.5.0 atau yang lebih baru — Untuk petunjuk pemasangan, lihat. Menyiapkan lingkungan pengembangan Ruby Anda untuk Elastic Beanstalk

  Dalam tutorial ini kita menggunakan Ruby 3.0.2 dan versi platform Elastic Beanstalk yang sesuai.
- Node.js – Untuk instruksi instalasi, lihat Menginstal Node.js melalui manajer paket.
- Yarn – Untuk instruksi instalasi, lihat Instalasi pada situs web Yarn.

Meluncurkan lingkungan Elastic Beanstalk

Gunakan konsol Elastic Beanstalk untuk membuat lingkungan Elastic Beanstalk. Pilih platform Ruby lalu terima pengaturan default dan kode sampel.

Untuk meluncurkan lingkungan (konsol)

1. Buka konsol Elastic Beanstalk menggunakan tautan yang telah dikonfigurasi sebelumnya: console.aws.amazon. com/elasticbeanstalk/home#/newApplication?applicationName=Tutorial&EnvironmentType= LoadBalanced
2. Untuk Platform, pilih platform dan cabang platform yang sesuai dengan bahasa yang digunakan oleh aplikasi Anda.
3. Untuk Kode aplikasi, pilih Aplikasi sampel.
4. Pilih Tinjau dan Luncurkan.
5. Tinjau opsi yang tersedia. Pilih opsi tersedia yang ingin Anda gunakan, dan saat Anda siap, pilih Buat aplikasi.

Pembuatan lingkungan membutuhkan waktu sekitar 5 menit dan membuat sumber daya berikut:

- EC2 instance — Mesin virtual Amazon Elastic Compute Cloud (Amazon EC2) yang dikonfigurasi untuk menjalankan aplikasi web pada platform yang Anda pilih.

  Setiap platform menjalankan satu set perangkat lunak, file konfigurasi dan penulisan tertentu untuk mendukung versi bahasa, kerangka kerja, kontainer web tertentu, atau kombinasi dari semua ini. Sebagian besar platform menggunakan Apache atau NGINX sebagai proksi terbalik yang ada di depan aplikasi web Anda, meneruskan permintaan ke aplikasi web, menyajikan aset statis, dan menghasilkan log akses dan kesalahan.
- Grup keamanan instans — Grup EC2 keamanan Amazon yang dikonfigurasi untuk memungkinkan lalu lintas masuk pada port 80. Sumber daya ini memungkinkan lalu lintas HTTP dari penyeimbang beban mencapai EC2 instance yang menjalankan aplikasi web Anda. Secara default, lalu lintas tidak diizinkan di port lain.
- Penyeimbang beban – Penyeimbang beban Elastic Load Balancing yang dikonfigurasi untuk mendistribusikan permintaan ke instans yang menjalankan aplikasi Anda. Penyeimbang beban juga menghilangkan kebutuhan untuk mengekspos instans Anda langsung ke internet.
- Grup keamanan penyeimbang beban - Grup EC2 keamanan Amazon yang dikonfigurasi untuk memungkinkan lalu lintas masuk di port 80. Sumber daya ini memungkinkan lalu lintas HTTP dari internet mencapai penyeimbang beban. Secara default, lalu lintas tidak diizinkan di port lain.
- Grup Auto Scaling – Grup Auto Scaling yang dikonfigurasi untuk menggantikan instans jika diakhiri atau menjadi tidak tersedia.

- Bucket Amazon S3 – Lokasi penyimpanan untuk kode sumber, log, dan artifact lainnya yang dibuat saat Anda menggunakan Elastic Beanstalk.
- CloudWatch Alarm Amazon — Dua CloudWatch alarm yang memantau beban pada instans di lingkungan Anda dan yang dipicu jika beban terlalu tinggi atau terlalu rendah. Saat alarm terpicu, grup Auto Scaling Anda akan menaikkan atau menurunkan skala sebagai respons.
- AWS CloudFormation stack - Elastic AWS CloudFormation Beanstalk digunakan untuk meluncurkan sumber daya di lingkungan Anda dan menyebarkan perubahan konfigurasi. Sumber daya ditentukan di sebuah templat yang dapat Anda lihat di Konsol AWS CloudFormation.
- Nama domain — Nama domain yang merutekan ke aplikasi web Anda dalam formulir *subdomain*.*region*.elasticbeanstalk.com.

> **Keamanan domain**
>
> Untuk meningkatkan keamanan aplikasi Elastic Beanstalk Anda, domain elasticbeanstalk.com terdaftar di Daftar Akhiran Publik (PSL).
> Jika Anda perlu mengatur cookie sensitif di nama domain default untuk aplikasi Elastic Beanstalk Anda, kami sarankan Anda menggunakan cookie `__Host-` dengan awalan untuk meningkatkan keamanan. Praktik ini melindungi domain Anda dari upaya pemalsuan permintaan lintas situs (CSRF). Untuk informasi selengkapnya, lihat halaman Set-Cookie di Jaringan Pengembang Mozilla.

Semua sumber daya ini dikelola oleh Elastic Beanstalk. Ketika Anda mengakhiri lingkungan, Elastic Beanstalk mengakhiri semua sumber daya yang dimuatnya.

> **Note**
>
> Bucket Amazon S3 yang dibuat Elastic Beanstalk dibagi antar lingkungan dan tidak dihapus selama pengakhiran lingkungan. Untuk informasi selengkapnya, lihat Menggunakan Elastic Beanstalk dengan Amazon S3.

Menginstal rails dan menghasilkan sebuah situs web

Menginstal Rails dan dependensinya dengan perintah `gem`.

```
~$ gem install rails
```

```
Fetching: concurrent-ruby-1.1.9.gem
Successfully installed concurrent-ruby-1.1.9
Fetching: rack-2.2.3.gem
Successfully installed rack-2.2.3
...
```

Uji instalasi Rails Anda.

```
~$ rails --version
Rails 6.1.4.1
```

Gunakan `rails new` dengan nama aplikasi untuk membuat proyek Rails baru.

```
~$ rails new ~/eb-rails
```

Rails membuat direktori dengan nama yang ditentukan, menghasilkan semua file yang dibutuhkan untuk menjalankan proyek sampel secara lokal, dan kemudian menjalankan bundler untuk menginstal semua dependensi (Gems) yang ditentukan dalam Gemfile proyek.

> **Note**
>
> Proses ini menginstal versi Puma terbaru untuk proyek tersebut. Versi ini mungkin berbeda dari versi yang disediakan Elastic Beanstalk pada versi platform Ruby di lingkungan Anda. Untuk melihat versi Puma yang disediakan oleh Elastic Beanstalk, lihat Riwayat Platform Ruby di panduan Platform.AWS Elastic Beanstalk Untuk informasi lebih lanjut tentang versi Puma terbaru, lihat situs web Puma.io. Jika ada ketidakcocokan antara dua versi Puma, gunakan salah satu opsi berikut:
>
> - Gunakan versi Puma yang diinstal oleh `rails new` perintah sebelumnya. Dalam hal ini Anda harus menambahkan untuk platform `Procfile` untuk menggunakan versi server Puma yang Anda sediakan sendiri. Untuk informasi selengkapnya, lihat Mengkonfigurasi proses aplikasi dengan Procfile pada Elastic Beanstalk..
> - Perbarui versi Puma agar konsisten dengan versi yang sudah diinstal sebelumnya pada versi platform Ruby lingkungan Anda. Untuk melakukannya, ubah versi Puma di Gemfile yang terletak di root direktori sumber proyek Anda. Lalu laribundle `update`. Untuk informasi lebih lanjut, lihat halaman pembaruan bundel di situs web Bundler.io.

Uji instalasi Rails Anda dengan menjalankan proyek default secara lokal.

```
~$ cd eb-rails
~/eb-rails$ rails server
=> Booting Puma
=> Rails 6.1.4.1 application starting in development
=> Run `bin/rails server --help` for more startup options
Puma starting in single mode...
* Puma version: 5.5.2 (ruby 3.0.2-p107) ("Zawgyi")
*  Min threads: 5
*  Max threads: 5
*  Environment: development
*          PID: 77857
* Listening on http://127.0.0.1:3000
* Listening on http://[::1]:3000
Use Ctrl-C to stop
...
```

Buka `http://localhost:3000` di peramban web untuk melihat proyek default yang sedang bertindak.



Halaman ini hanya terlihat dalam mode pengembangan. Tambahkan beberapa konten ke halaman depan aplikasi untuk mendukung deployment produksi ke Elastic Beanstalk. Gunakan `rails generate` untuk membuat pengendali, rute, dan tampilan untuk halaman selamat datang Anda.

```
~/eb-rails$ rails generate controller WelcomePage welcome
      create  app/controllers/welcome_page_controller.rb
       route  get 'welcome_page/welcome'
      invoke  erb
      create    app/views/welcome_page
      create    app/views/welcome_page/welcome.html.erb
      invoke  test_unit
      create    test/controllers/welcome_page_controller_test.rb
      invoke  helper
      create    app/helpers/welcome_page_helper.rb
      invoke    test_unit
      invoke  assets
      invoke    coffee
      create      app/assets/javascripts/welcome_page.coffee
      invoke    scss
      create      app/assets/stylesheets/welcome_page.scss.
```

Hal ini memberikan semua yang Anda butuhkan untuk mengakses halaman di `/welcome_page/welcome`. Namun, sebelum Anda mempublikasikan perubahan, ubah konten pada tampilan dan tambahkan rute untuk membuat halaman ini muncul di tingkat atas situs.

Gunakan editor teks untuk mengedit konten di `app/views/welcome_page/welcome.html.erb`. Untuk contoh ini, Anda akan menggunakan `cat` hanya untuk menimpa isi dari file yang ada.

Example app/views/welcome_page/welcome.html.erb

```
<h1>Welcome!</h1>
<p>This is the front page of my first Rails application on Elastic Beanstalk.</p>
```

Terakhir, tambahkan rute berikut ke `config/routes.rb`:

Example config/routes.rb

```
Rails.application.routes.draw do
  get 'welcome_page/welcome'
  root 'welcome_page#welcome'
```

Ini memberi tahu Rails untuk merutekan permintaan ke akar situs web menuju metode sambutan pengendali halaman sambutan, yang membuat konten dalam tampilan selamat datang (`welcome.html.erb`).

Agar Elastic Beanstalk berhasil menyebarkan aplikasi pada platform Ruby, kita perlu memperbarui. `Gemfile.lock` Beberapa dependensi `Gemfile.lock` mungkin spesifik platform. Oleh karena itu, kita **`platform ruby`** perlu menambahkan `Gemfile.lock` sehingga semua dependensi yang diperlukan diinstal dengan penerapan.

Example

```
~/eb-rails$ bundle lock --add-platform ruby
Fetching gem metadata from https://rubygems.org/.....
Resolving dependencies...
Writing lockfile to /Users/janedoe/EBDPT/RubyApps/eb-rails-doc-app/Gemfile.lock
```

Mengonfigurasi pengaturan rails

Gunakan konsol Elastic Beanstalk untuk mengonfigurasi Rails dengan properti lingkungan. Tetapkan properti lingkungan `SECRET_KEY_BASE` menjadi string hingga 256 karakter alfanumerik.

Rails menggunakan properti ini untuk membuat kunci. Oleh karena itu Anda harus tetap merahasiakannya dan tidak menyimpannya dalam kontrol sumber dalam teks biasa. Sebagai gantinya, Anda memberikannya ke kode Rails pada lingkungan Anda melalui properti lingkungan.

Untuk mengonfigurasi variabel lingkungan di konsol Elastic Beanstalk

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Dalam kategori konfigurasi Pembaruan, pemantauan, dan pencatatan, pilih Edit.
5. Gulir ke bawah ke variabel lingkungan Runtime.
6. Pilih Tambahkan variabel lingkungan.
7. Untuk Sumber pilih Teks biasa.

> **Note**
>
> Nilai Secrets Manager dan SSM Parameter Store di drop-down adalah untuk mengonfigurasi variabel lingkungan sebagai rahasia untuk menyimpan data sensitif, seperti kredensyal dan kunci API. Untuk informasi selengkapnya, lihat Menggunakan Elastic Beanstalk dengan dan AWS Secrets Manager Parameter Store AWS Systems Manager.

8. Masukkan nama variabel Lingkungan dan pasangan nilai variabel Lingkungan.
9. Jika Anda perlu menambahkan lebih banyak variabel ulangi Langkah 6 hingga Langkah 8.
10. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Sekarang Anda siap untuk men-deploy situs ke lingkungan Anda.

Deploy aplikasi Anda

Buat paket sumber yang berisi file yang dibuat oleh Rails. Perintah berikut membuat paket sumber yang bernama `rails-default.zip`.

```
~/eb-rails$ zip ../rails-default.zip -r * .[^.]*
```

Unggah paket sumber ke Elastic Beanstalk untuk men-deploy Rails ke lingkungan Anda.

Untuk men-deploy paket sumber

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Di halaman gambaran umum lingkungan, pilih Unggah dan deploy.
4. Gunakan kotak dialog di layar untuk mengunggah paket sumber.
5. Pilih Deploy.
6. Ketika deployment selesai, Anda dapat memilih URL situs untuk membuka situs web Anda di tab baru.

Pembersihan

Setelah Anda selesai bekerja dengan kode demo, Anda dapat mengakhiri lingkungan Anda. Elastic Beanstalk menghapus semua sumber daya AWS terkait, seperti instans EC2 Amazon, instans database, penyeimbang beban, grup keamanan,dan alarm.

Menghapus sumber daya tidak menghapus aplikasi Elastic Beanstalk, sehingga Anda dapat membuat lingkungan baru untuk aplikasi Anda kapan saja.

Untuk mengakhiri lingkungan Elastic Beanstalk Anda dari konsol

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pilih Tindakan, lalu pilih Terminate environment.
4. Gunakan kotak dialog di layar untuk mengonfirmasi pengakhiran lingkungan.

Langkah selanjutnya

Untuk informasi lebih lanjut tentang Rails, kunjungi rubyonrails.org.

Ketika aplikasi terus dikembangkan, Anda mungkin akan menginginkan sebuah cara untuk mengelola lingkungan dan men-deploy aplikasi Anda tanpa membuat file .zip secara manual dan mengunggahnya ke konsol Elastic Beanstalk. Elastic Beanstalk Command Line Interface (EB CLI easy-to-use) menyediakan perintah untuk membuat, mengkonfigurasi, dan menyebarkan aplikasi ke lingkungan Elastic Beanstalk dari baris perintah.

Terakhir, jika Anda berencana menggunakan aplikasi dalam lingkungan produksi, Anda akan ingin mengonfigurasikan nama domain khusus untuk lingkungan Anda dan mengaktifkan HTTPS untuk koneksi yang aman.

Men-deploy aplikasi sinatra ke Elastic Beanstalk

Panduan ini menunjukkan cara men-deploy aplikasi web Sinatra ke AWS Elastic Beanstalk.

Prasyarat

Tutorial ini mengasumsikan Anda memiliki pengetahuan tentang operasi Elastic Beanstalk dasar dan konsol Elastic Beanstalk. Jika belum, ikuti petunjuk di Pelajari cara memulai dengan Elastic Beanstalk untuk meluncurkan lingkungan Elastic Beanstalk pertama Anda.

Untuk mengikuti prosedur di panduan ini, Anda memerlukan terminal atau shell baris perintah untuk menjalankan perintah. Perintah ditampilkan dalam daftar yang diawali dengan simbol prompt ($) dan nama direktori saat ini, bila perlu.

```
~/eb-project$ this is a command
this is output
```

Di Linux dan macOS, Anda dapat menggunakan shell dan manajer paket pilihan Anda. Pada Windows Anda dapat menginstal Windows Subsystem untuk Linux untuk mendapatkan versi Windows terintegrasi dari Ubuntu dan Bash.

Sinatra 2.1.0 membutuhkan Ruby 2.3.0 atau yang lebih baru. Dalam tutorial ini kita menggunakan Ruby 3.0.2 dan versi platform Elastic Beanstalk yang sesuai. Instal Ruby dengan mengikuti petunjuk di Menyiapkan lingkungan pengembangan Ruby Anda untuk Elastic Beanstalk.

Meluncurkan lingkungan Elastic Beanstalk

Gunakan konsol Elastic Beanstalk untuk membuat lingkungan Elastic Beanstalk. Pilih platform Ruby lalu terima pengaturan default dan kode sampel.

Untuk meluncurkan lingkungan (konsol)

1. Buka konsol Elastic Beanstalk menggunakan tautan yang telah dikonfigurasi sebelumnya: console.aws.amazon. com/elasticbeanstalk/home#/newApplication?applicationName=Tutorial&EnvironmentType= LoadBalanced
2. Untuk Platform, pilih platform dan cabang platform yang sesuai dengan bahasa yang digunakan oleh aplikasi Anda.
3. Untuk Kode aplikasi, pilih Aplikasi sampel.
4. Pilih Tinjau dan Luncurkan.
5. Tinjau opsi yang tersedia. Pilih opsi tersedia yang ingin Anda gunakan, dan saat Anda siap, pilih Buat aplikasi.

Pembuatan lingkungan membutuhkan waktu sekitar 5 menit dan membuat sumber daya berikut:

- EC2 instance — Mesin virtual Amazon Elastic Compute Cloud (Amazon EC2) yang dikonfigurasi untuk menjalankan aplikasi web pada platform yang Anda pilih.

  Setiap platform menjalankan satu set perangkat lunak, file konfigurasi dan penulisan tertentu untuk mendukung versi bahasa, kerangka kerja, kontainer web tertentu, atau kombinasi dari semua ini. Sebagian besar platform menggunakan Apache atau NGINX sebagai proksi terbalik yang ada di depan aplikasi web Anda, meneruskan permintaan ke aplikasi web, menyajikan aset statis, dan menghasilkan log akses dan kesalahan.
- Grup keamanan instans — Grup EC2 keamanan Amazon yang dikonfigurasi untuk memungkinkan lalu lintas masuk pada port 80. Sumber daya ini memungkinkan lalu lintas HTTP dari penyeimbang beban mencapai EC2 instance yang menjalankan aplikasi web Anda. Secara default, lalu lintas tidak diizinkan di port lain.
- Penyeimbang beban – Penyeimbang beban Elastic Load Balancing yang dikonfigurasi untuk mendistribusikan permintaan ke instans yang menjalankan aplikasi Anda. Penyeimbang beban juga menghilangkan kebutuhan untuk mengekspos instans Anda langsung ke internet.
- Grup keamanan penyeimbang beban - Grup EC2 keamanan Amazon yang dikonfigurasi untuk memungkinkan lalu lintas masuk di port 80. Sumber daya ini memungkinkan lalu lintas HTTP dari internet mencapai penyeimbang beban. Secara default, lalu lintas tidak diizinkan di port lain.
- Grup Auto Scaling – Grup Auto Scaling yang dikonfigurasi untuk menggantikan instans jika diakhiri atau menjadi tidak tersedia.
- Bucket Amazon S3 – Lokasi penyimpanan untuk kode sumber, log, dan artifact lainnya yang dibuat saat Anda menggunakan Elastic Beanstalk.
- CloudWatch Alarm Amazon — Dua CloudWatch alarm yang memantau beban pada instans di lingkungan Anda dan yang dipicu jika beban terlalu tinggi atau terlalu rendah. Saat alarm terpicu, grup Auto Scaling Anda akan menaikkan atau menurunkan skala sebagai respons.
- AWS CloudFormation stack - Elastic AWS CloudFormation Beanstalk digunakan untuk meluncurkan sumber daya di lingkungan Anda dan menyebarkan perubahan konfigurasi. Sumber daya ditentukan di sebuah templat yang dapat Anda lihat di Konsol AWS CloudFormation.
- Nama domain — Nama domain yang merutekan ke aplikasi web Anda dalam formulir *`subdomain`*.*`region`*.elasticbeanstalk.com.

> **Keamanan domain**
>
> Untuk meningkatkan keamanan aplikasi Elastic Beanstalk Anda, domain elasticbeanstalk.com terdaftar di Daftar Akhiran Publik (PSL).
> Jika Anda perlu mengatur cookie sensitif di nama domain default untuk aplikasi Elastic Beanstalk Anda, kami sarankan Anda menggunakan cookie `__Host-` dengan awalan untuk meningkatkan keamanan. Praktik ini melindungi domain Anda dari upaya pemalsuan permintaan lintas situs (CSRF). Untuk informasi selengkapnya, lihat halaman Set-Cookie di Jaringan Pengembang Mozilla.

Semua sumber daya ini dikelola oleh Elastic Beanstalk. Ketika Anda mengakhiri lingkungan, Elastic Beanstalk mengakhiri semua sumber daya yang dimuatnya.

> **Note**
>
> Bucket Amazon S3 yang dibuat Elastic Beanstalk dibagi antar lingkungan dan tidak dihapus selama pengakhiran lingkungan. Untuk informasi selengkapnya, lihat Menggunakan Elastic Beanstalk dengan Amazon S3.

Tulis situs web dasar sinatra

Untuk membuat dan men-deploy aplikasi sinatra

1. Membuat file konfigurasi bernama config.ru dengan konten berikut.

   Example config.ru

   ```
   require './helloworld'
   run Sinatra::Application
   ```

2. Buat file kode Ruby bernama helloworld.rb dengan konten berikut.

   Example helloworld.rb

   ```
   require 'sinatra'
   get '/' do
     "Hello World!"
   ```

```
end
```

3. Buat Gemfile dengan konten berikut.

   Example Gemfile

   ```
   source 'https://rubygems.org'
   gem 'sinatra'
   gem 'puma'
   ```

4. Jalankan pemasangan bundel untuk menghasilkan `Gemfile.lock`

   Example

   ```
   ~/eb-sinatra$ bundle install
   Fetching gem metadata from https://rubygems.org/....
   Resolving dependencies...
   Using bundler 2.2.22
   Using rack 2.2.3
   ...
   ```

5. Agar Elastic Beanstalk berhasil menyebarkan aplikasi pada platform Ruby, kita perlu memperbarui. `Gemfile.lock` Beberapa dependensi `Gemfile.lock` mungkin spesifik platform. Oleh karena itu, kita **`platform ruby`** perlu menambahkan `Gemfile.lock` sehingga semua dependensi yang diperlukan diinstal dengan penerapan.

   Example

   ```
   ~/eb-sinatra$ bundle lock --add-platform ruby
   Fetching gem metadata from https://rubygems.org/....
   Resolving dependencies...
   Writing lockfile to /Users/janedoe/EBDPT/RubyApps/eb-sinatra/Gemfile.lock
   ```

6. Buat Procfile dengan konten berikut.

   Example Procfile

   ```
   web: bundle exec puma -C /opt/elasticbeanstalk/config/private/pumaconf.rb
   ```

Men-deploy aplikasi Anda

Membuat paket sumber yang berisi file sumber Anda. Perintah berikut membuat paket sumber yang bernama `sinatra-default.zip`.

```
~/eb-sinatra$ zip ../sinatra-default.zip -r * .[^.]*
```

Unggah paket sumber ke Elastic Beanstalk untuk men-deploy Sinatra ke lingkungan Anda.

Untuk men-deploy paket sumber

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Di halaman gambaran umum lingkungan, pilih Unggah dan deploy.
4. Gunakan kotak dialog di layar untuk mengunggah paket sumber.
5. Pilih Deploy.
6. Ketika deployment selesai, Anda dapat memilih URL situs untuk membuka situs web Anda di tab baru.

Pembersihan

Setelah Anda selesai bekerja dengan kode demo, Anda dapat mengakhiri lingkungan Anda. Elastic Beanstalk menghapus semua sumber daya AWS terkait, seperti instans EC2 Amazon, instans database, penyeimbang beban, grup keamanan,dan alarm.

Menghapus sumber daya tidak menghapus aplikasi Elastic Beanstalk, sehingga Anda dapat membuat lingkungan baru untuk aplikasi Anda kapan saja.

Untuk mengakhiri lingkungan Elastic Beanstalk Anda dari konsol

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pilih Tindakan, lalu pilih Terminate environment.
4. Gunakan kotak dialog di layar untuk mengonfirmasi pengakhiran lingkungan.

Langkah selanjutnya

Untuk informasi lebih lanjut tentang Sinatra, kunjungi sinatrarb.com.

Ketika aplikasi terus dikembangkan, Anda mungkin akan menginginkan sebuah cara untuk mengelola lingkungan dan men-deploy aplikasi Anda tanpa membuat file .zip secara manual dan mengunggahnya ke konsol Elastic Beanstalk. Elastic Beanstalk Command Line Interface (EB CLI easy-to-use) menyediakan perintah untuk membuat, mengkonfigurasi, dan menyebarkan aplikasi ke lingkungan Elastic Beanstalk dari baris perintah.

Terakhir, jika Anda berencana menggunakan aplikasi dalam lingkungan produksi, Anda akan ingin mengonfigurasikan nama domain khusus untuk lingkungan Anda dan mengaktifkan HTTPS untuk koneksi yang aman.

Menambahkan instans Amazon RDS DB ke lingkungan Ruby Elastic Beanstalk Anda

Topik ini memberikan instruksi untuk membuat Amazon RDS menggunakan konsol Elastic Beanstalk. Anda dapat menggunakan instans DB Amazon Relational Database Service (Amazon RDS) untuk menyimpan data yang dikumpulkan dan dimodifikasi oleh aplikasi Anda. Basis data dapat digabungkan ke lingkungan Anda dan dikelola oleh Elastic Beanstalk, atau dapat dibuat sebagai dipisahkan dan dikelola secara eksternal oleh layanan lain. Dalam instruksi ini database digabungkan dengan lingkungan Anda dan dikelola oleh Elastic Beanstalk. Untuk informasi lebih lanjut tentang mengintegrasikan Amazon RDS dengan Elastic Beanstalk, lihat. Menambahkan basis data ke lingkungan Elastic Beanstalk Anda

Bagian-bagian

- Menambahkan instans DB ke lingkungan Anda
- Mengunduh adaptor
- Menyambungkan ke basis data

Menambahkan instans DB ke lingkungan Anda

Untuk menambahkan instans DB ke lingkungan Anda

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS

2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Di kategori konfigurasi Basis data, pilih Edit.
5. Pilih mesin DB, dan masukkan nama pengguna dan kata sandi.
6. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Menambahkan instans DB memakan waktu sekitar 10 menit. Ketika pembaruan lingkungan selesai, nama host instans DB dan informasi koneksi lainnya tersedia untuk aplikasi Anda melalui properti lingkungan berikut:

Nama properti	Deskripsi	Nilai properti
`RDS_HOSTNAME`	Nama host instans DB.	Di tab Konektivitas & keamanan di konsol Amazon RDS: Titik akhir.
`RDS_PORT`	Port tempat instans DB menerima koneksi. Nilai default bervariasi di antara mesin DB.	Di tab Konektivitas & keamanan di konsol Amazon RDS: Port.
`RDS_DB_NAME`	Nama basis data, **ebdb**.	Di tab Konfigurasi di konsol Amazon RDS: Nama DB.
`RDS_USERNAME`	Nama pengguna yang Anda konfigurasi untuk basis data Anda.	Di tab Konfigurasi di konsol Amazon RDS: Nama pengguna utama.
`RDS_PASSWORD`	Kata sandi yang Anda konfigurasi untuk basis data Anda.	Tidak tersedia untuk referensi di konsol Amazon RDS.

Untuk informasi selengkapnya tentang mengonfigurasi instance database yang digabungkan dengan lingkungan Elastic Beanstalk, lihat. Menambahkan basis data ke lingkungan Elastic Beanstalk Anda

Mengunduh adaptor

Menambahkan adaptor basis data ke file gem proyek Anda.

Example Gemfile – Rails dengan MySQL

```
source 'https://rubygems.org'
gem 'puma'
gem 'rails', '~> 6.1.4', '>= 6.1.4.1'
gem 'mysql2'
```

Gems adaptor umum untuk Ruby

- MySQL – mysql2
- PostgreSQL – pg
- Oracle – activerecord-oracle_enhanced-adapter
- SQL Server – activerecord-sqlserver-adapter

Menyambungkan ke basis data

Elastic Beanstalk memberikan informasi koneksi untuk instans DB terlampir di properti lingkungan. Gunakan `ENV['VARIABLE']` untuk membaca properti dan mengonfigurasi koneksi basis data.

Example config/database.yml— Konfigurasi basis data Ruby on rails (MySQL)

```
production:
  adapter: mysql2
  encoding: utf8
  database: <%= ENV['RDS_DB_NAME'] %>
  username: <%= ENV['RDS_USERNAME'] %>
  password: <%= ENV['RDS_PASSWORD'] %>
  host: <%= ENV['RDS_HOSTNAME'] %>
  port: <%= ENV['RDS_PORT'] %>
```

Menyebarkan dengan wadah Docker ke Elastic Beanstalk

Bab ini menjelaskan bagaimana Anda dapat menggunakan Elastic Beanstalk untuk menyebarkan aplikasi web dari kontainer Docker. Kontainer Docker terisi sendiri dan mencakup semua informasi konfigurasi dan perangkat lunak yang diperlukan aplikasi web Anda untuk menjalankan. Dengan kontainer Docker Anda dapat menentukan lingkungan runtime Anda sendiri. Anda juga dapat memilih bahasa pemrograman dan dependensi aplikasi Anda sendiri, seperti manajer paket atau alat, yang biasanya tidak didukung oleh platform Elastic Beanstalk lainnya.

Ikuti langkah-langkah QuickStart untuk Docker untuk membuat aplikasi “Hello World” Docker dan menerapkannya ke lingkungan Elastic Beanstalk menggunakan EB CLI.

Topik

- Cabang platform Elastic Beanstalk Docker
- Menggunakan cabang platform Elastic Beanstalk Docker
- Menggunakan cabang platform Docker yang dikelola ECS di Elastic Beanstalk
- Menggunakan gambar dari repositori pribadi di Elastic Beanstalk
- Mengkonfigurasi lingkungan Elastic Beanstalk Docker
- Platform warisan

Cabang platform Elastic Beanstalk Docker

Platform Elastic Beanstalk Docker mendukung cabang platform berikut:

Docker menjalankan Amazon Linux 2 dan Docker menjalankan 023 AL2

Elastic Beanstalk menyebarkan wadah Docker dan kode sumber ke instance dan mengelolanya. EC2 Cabang platform ini menawarkan dukungan multi-kontainer. Anda dapat menggunakan alat Docker Compose untuk menyederhanakan konfigurasi, pengujian, dan penerapan aplikasi Anda. Untuk informasi lebih lanjut tentang cabang platform ini, lihatthe section called “Cabang platform Docker”.

ECS berjalan di Amazon Linux 2 dan ECS berjalan pada 023 AL2

Kami menyediakan cabang ini untuk pelanggan yang membutuhkan jalur migrasi ke AL2 023/ AL2 dari cabang platform pensiunan Multi-container Docker yang berjalan di (Amazon Linux AMI). Cabang

platform terbaru mendukung semua fitur dari cabang platform pensiunan. Tidak ada perubahan pada kode sumber yang diperlukan. Untuk informasi selengkapnya, lihat Migrasi aplikasi Elastic Beanstalk Anda dari Docker Multi-container yang dikelola ECS ke ECS di Amazon Linux 2023 AL1 . Jika Anda tidak memiliki lingkungan Elastic Beanstalk yang berjalan di cabang platform berbasis ECS, kami sarankan Anda menggunakan cabang platform, Docker Running on 64bit 023. AL2 Ini menawarkan pendekatan yang lebih sederhana dan membutuhkan lebih sedikit sumber daya.

Untuk daftar versi komponen perangkat lunak yang terkait dengan masing-masing cabang platform ini, lihat Docker di dokumen AWS Elastic Beanstalk Platform.

Cabang platform pensiunan berjalan di Amazon Linux AMI () AL1

Pada tanggal 18 Juli 2022, Elastic Beanstalk menetapkan status semua cabang platform berdasarkan Amazon Linux AMI () untuk pensiun. AL1 Perluas setiap bagian berikut untuk membaca lebih lanjut tentang setiap cabang platform yang sudah pensiun dan jalur migrasinya ke cabang platform terbaru yang berjalan di Amazon Linux 2 atau Amazon Linux 2023 (disarankan).

Docker (Amazon Linux AMI)

Cabang platform ini dapat menyebarkan gambar Docker, dijelaskan dalam definisi Dockerfile atau v1. `Dockerrun.aws.json` Cabang platform ini hanya menjalankan satu kontainer untuk setiap instance. Cabang platformnya yang sukses, Docker berjalan pada 64bit AL2 023 dan Docker yang berjalan di 64bit Amazon Linux 2 mendukung beberapa kontainer Docker per instance.

Kami menyarankan Anda membuat lingkungan Anda dengan cabang platform Docker yang lebih baru dan didukung yang berjalan pada 64bit AL2 023. Anda kemudian dapat memigrasikan aplikasi Anda ke lingkungan yang baru dibuat. Untuk informasi selengkapnya tentang membuat lingkungan ini, lihatthe section called “Cabang platform Docker”. Untuk informasi selengkapnya tentang migrasi, lihatMemigrasi aplikasi Elastic Beanstalk Linux Anda ke Amazon Linux 2023 atau Amazon Linux 2.

Docker multi-wadah (Amazon Linux AMI)

Cabang platform ini menggunakan Amazon ECS untuk mengoordinasikan penyebaran beberapa kontainer Docker ke cluster Amazon ECS di lingkungan Elastic Beanstalk. Jika saat ini Anda menggunakan cabang platform yang sudah pensiun ini, kami sarankan Anda bermigrasi ke cabang platform ECS Running on Amazon Linux 2023 terbaru. Cabang platform terbaru mendukung semua fitur dari cabang platform yang dihentikan ini. Tidak ada perubahan pada kode sumber yang diperlukan. Untuk informasi selengkapnya, lihat Migrasi aplikasi Elastic Beanstalk Anda dari Docker Multi-container yang dikelola ECS ke ECS di Amazon Linux 2023 AL1 .

kontainer Docker yang dikonfigurasi sebelumnya

Selain platform Docker yang disebutkan sebelumnya, ada juga cabang GlassFish platform Docker yang telah dikonfigurasi sebelumnya yang berjalan pada sistem operasi Amazon Linux AMI (). AL1

Cabang platform ini telah digantikan oleh cabang platform Docker yang berjalan pada 64bit AL2 023 dan Docker yang berjalan di 64bit Amazon Linux 2. Untuk informasi selengkapnya, lihat Menerapkan GlassFish aplikasi ke platform Docker.

Menggunakan cabang platform Elastic Beanstalk Docker

Bagian ini menjelaskan cara menyiapkan image Docker Anda untuk diluncurkan dengan salah satu cabang platform Elastic Beanstalk Docker running atau 023. AL2 AL2

Ikuti langkah-langkah QuickStart untuk Docker untuk membuat aplikasi “Hello World” Docker dan menerapkannya ke lingkungan Elastic Beanstalk menggunakan EB CLI.

Topik

- QuickStart: Menyebarkan aplikasi Docker ke Elastic Beanstalk
- Mempersiapkan image Docker Anda untuk penyebaran ke Elastic Beanstalk

QuickStart: Menyebarkan aplikasi Docker ke Elastic Beanstalk

QuickStart Tutorial ini memandu Anda melalui proses pembuatan aplikasi Docker dan menyebarkannya ke lingkungan AWS Elastic Beanstalk .

> **Note**
>
> Contoh tutorial dimaksudkan untuk demonstrasi. Jangan gunakan aplikasi untuk lalu lintas produksi.

Bagian-bagian

- AWS Akun Anda
- Prasyarat
- Langkah 1: Buat aplikasi dan wadah Docker

- Langkah 2: Jalankan aplikasi Anda secara lokal
- Langkah 3: Terapkan aplikasi Docker Anda dengan EB CLI
- Langkah 4: Jalankan aplikasi Anda di Elastic Beanstalk
- Langkah 5: Bersihkan
- AWS sumber daya untuk aplikasi Anda
- Langkah selanjutnya
- Terapkan dengan konsol Elastic Beanstalk

AWS Akun Anda

Jika Anda belum menjadi AWS pelanggan, Anda perlu membuat AWS akun. Mendaftar memungkinkan Anda mengakses Elastic Beanstalk AWS dan layanan lain yang Anda butuhkan.

Jika Anda sudah memiliki AWS akun, Anda dapat melanjutkan kePrasyarat.

Buat AWS akun

Mendaftar untuk Akun AWS

Jika Anda tidak memiliki Akun AWS, selesaikan langkah-langkah berikut untuk membuatnya.

Untuk mendaftar untuk Akun AWS

1. Buka https://portal.aws.amazon.com/billing/pendaftaran.
2. Ikuti petunjuk online.

   Bagian dari prosedur pendaftaran melibatkan menerima panggilan telepon atau pesan teks dan memasukkan kode verifikasi pada keypad telepon.

   Saat Anda mendaftar untuk sebuah Akun AWS, sebuah Pengguna root akun AWSdibuat. Pengguna root memiliki akses ke semua Layanan AWS dan sumber daya di akun. Sebagai praktik keamanan terbaik, tetapkan akses administratif ke pengguna, dan gunakan hanya pengguna root untuk melakukan tugas yang memerlukan akses pengguna root.

AWS mengirimi Anda email konfirmasi setelah proses pendaftaran selesai. Kapan saja, Anda dapat melihat aktivitas akun Anda saat ini dan mengelola akun Anda dengan masuk https://aws.amazon.comke/ dan memilih Akun Saya.

Buat pengguna dengan akses administratif

Setelah Anda mendaftar Akun AWS, amankan Pengguna root akun AWS, aktifkan AWS IAM Identity Center, dan buat pengguna administratif sehingga Anda tidak menggunakan pengguna root untuk tugas sehari-hari.

Amankan Anda Pengguna root akun AWS

1. Masuk ke AWS Management Consolesebagai pemilik akun dengan memilih pengguna Root dan memasukkan alamat Akun AWS email Anda. Di laman berikutnya, masukkan kata sandi.

   Untuk bantuan masuk dengan menggunakan pengguna root, lihat Masuk sebagai pengguna root di AWS Sign-In Panduan Pengguna.
2. Mengaktifkan autentikasi multi-faktor (MFA) untuk pengguna root Anda.

   Untuk petunjuk, lihat Mengaktifkan perangkat MFA virtual untuk pengguna Akun AWS root (konsol) Anda di Panduan Pengguna IAM.

Buat pengguna dengan akses administratif

1. Aktifkan Pusat Identitas IAM.

   Untuk mendapatkan petunjuk, silakan lihat Mengaktifkan AWS IAM Identity Center di Panduan Pengguna AWS IAM Identity Center .
2. Di Pusat Identitas IAM, berikan akses administratif ke pengguna.

   Untuk tutorial tentang menggunakan Direktori Pusat Identitas IAM sebagai sumber identitas Anda, lihat Mengkonfigurasi akses pengguna dengan default Direktori Pusat Identitas IAM di Panduan AWS IAM Identity Center Pengguna.

Masuk sebagai pengguna dengan akses administratif

- Untuk masuk dengan pengguna Pusat Identitas IAM, gunakan URL masuk yang dikirim ke alamat email saat Anda membuat pengguna Pusat Identitas IAM.

  Untuk bantuan masuk menggunakan pengguna Pusat Identitas IAM, lihat Masuk ke portal AWS akses di Panduan AWS Sign-In Pengguna.

Tetapkan akses ke pengguna tambahan

1. Di Pusat Identitas IAM, buat set izin yang mengikuti praktik terbaik menerapkan izin hak istimewa paling sedikit.

   Untuk petunjuknya, lihat Membuat set izin di Panduan AWS IAM Identity Center Pengguna.
2. Tetapkan pengguna ke grup, lalu tetapkan akses masuk tunggal ke grup.

   Untuk petunjuk, lihat Menambahkan grup di Panduan AWS IAM Identity Center Pengguna.

Prasyarat

Untuk mengikuti prosedur di panduan ini, Anda memerlukan terminal atau shell baris perintah untuk menjalankan perintah. Perintah ditampilkan dalam daftar yang diawali dengan simbol prompt ($) dan nama direktori saat ini, bila perlu.

```
~/eb-project$ this is a command
this is output
```

Di Linux dan macOS, Anda dapat menggunakan shell dan manajer paket pilihan Anda. Pada Windows Anda dapat menginstal Windows Subsystem untuk Linux untuk mendapatkan versi Windows terintegrasi dari Ubuntu dan Bash.

EB CLI

Tutorial ini menggunakan Elastic Beanstalk Command Line Interface (EB CLI). Untuk detail tentang pemasangan dan konfigurasi EB CLI, lihat Instal EB CLI dengan skrip pengaturan (disarankan) dan Mengonfigurasi EB CLI.

Docker

Untuk mengikuti tutorial ini, Anda memerlukan instalasi lokal Docker yang berfungsi. Untuk informasi lebih lanjut, lihat Dapatkan Docker di situs web dokumentasi Docker.

Verifikasi daemon Docker sedang berjalan dengan menjalankan perintah berikut.

```
~$ docker info
```

Langkah 1: Buat aplikasi dan wadah Docker

Untuk contoh ini, kita membuat image Docker dari contoh aplikasi Flask yang juga direferensikan. Men-deploy aplikasi Flask ke Elastic Beanstalk

Aplikasi ini terdiri dari dua file:

- `app.py`— file Python yang berisi kode yang akan dijalankan dalam wadah.
- `Dockerfile`— Dockerfile untuk membangun gambar Anda.

Tempatkan kedua file di root direktori.

```
~/eb-docker-flask/
|-- Dockerfile
|-- app.py
```

Tambahkan konten berikut ke Anda`Dockerfile`.

Example **`~/eb-docker-flask/Dockerfile`**

```
FROM python:3.12
COPY . /app
WORKDIR /app
RUN pip install Flask==3.0.2
EXPOSE 5000
CMD [ "python3", "-m" , "flask", "run", "--host=0.0.0.0"]
```

Tambahkan konten berikut ke `app.py` file Anda.

Example **`~/eb-docker-flask/app.py`**

```
from flask import Flask
app = Flask(__name__)
@app.route('/')
def hello_world():
    return 'Hello Elastic Beanstalk! This is a Docker application'
```

Bangun wadah Docker Anda, beri tag gambar dengan. `eb-docker-flask`

```
~/eb-docker-flask$ docker build -t eb-docker-flask
```

Langkah 2: Jalankan aplikasi Anda secara lokal

Gunakan perintah docker build untuk membuat image container Anda secara lokal, menandai gambar dengan. `eb-docker-flask` Periode (`.`) di akhir perintah mengkhususkan bahwa path adalah direktori lokal.

```
~/eb-docker-flask$ docker run -dp 127.0.0.1:5000:5000 eb-docker-flask .
```

Jalankan container Anda dengan perintah docker run. Perintah akan mencetak ID dari wadah yang sedang berjalan. -dOpsi ini menjalankan docker dalam mode latar belakang. -pOpsi ini mengekspos aplikasi Anda di port 5000. Elastic Beanstalk melayani lalu lintas ke port 5000 pada platform Docker secara default.

```
~/eb-docker-flask$ docker run -dp 127.0.0.1:5000:5000 eb-docker-flask container-id
```

Arahkan ke `http://127.0.0.1:5000/` di browser Anda. Anda akan melihat teks “Halo Elastic Beanstalk! Ini adalah aplikasi Docker”.

Jalankan perintah docker kill untuk mengakhiri container.

```
~/eb-docker-flask$ docker kill container-id
```

Langkah 3: Terapkan aplikasi Docker Anda dengan EB CLI

Jalankan perintah berikut untuk membuat lingkungan Elastic Beanstalk untuk aplikasi ini.

Untuk membuat lingkungan dan menerapkan aplikasi Docker Anda

1. Inisialisasi repositori EB CLI Anda dengan perintah eb init.

   ```
   ~/eb-docker-flask$ eb init -p docker docker-tutorial us-east-2
   Application docker-tutorial has been created.
   ```

   Perintah ini membuat aplikasi bernama `docker-tutorial` dan mengkonfigurasi repositori lokal Anda untuk membuat lingkungan dengan versi platform Docker terbaru.
2. (Opsional) Jalankan eb init lagi untuk mengonfigurasi key pair default sehingga Anda dapat menggunakan SSH untuk terhubung ke EC2 instance yang menjalankan aplikasi Anda.

   ```
   ~/eb-docker-flask$ eb init
   ```

```
Do you want to set up SSH for your instances?
(y/n): y
Select a keypair.
1) my-keypair
2) [ Create new KeyPair ]
```

Pilih pasangan kunci jika Anda sudah memilikinya, atau ikuti prompt untuk membuatnya. Jika Anda tidak melihat prompt atau perlu mengubah pengaturan Anda nanti, jalankan eb init -i.

3. Buat lingkungan baru dan deploy aplikasi Anda dengan eb create. Elastic Beanstalk secara otomatis membuat file zip untuk aplikasi Anda dan memulainya pada port 5000.

```
~/eb-docker-flask$ eb create docker-tutorial
```

Dibutuhkan sekitar lima menit untuk Elastic Beanstalk untuk menciptakan lingkungan Anda.

Langkah 4: Jalankan aplikasi Anda di Elastic Beanstalk

Ketika proses untuk membuat lingkungan Anda selesai, buka situs web Anda denganeb open.

```
~/eb-docker-flask$ eb open
```

Selamat! Anda telah menerapkan aplikasi Docker dengan Elastic Beanstalk! Ini akan membuka jendela peramban menggunakan nama domain yang dibuat untuk aplikasi Anda.

Langkah 5: Bersihkan

Anda dapat menghentikan lingkungan Anda ketika Anda selesai bekerja dengan aplikasi Anda. Elastic Beanstalk AWS mengakhiri semua sumber daya yang terkait dengan lingkungan Anda.

Untuk mengakhiri lingkungan Elastic Beanstalk Anda dengan EB CLI jalankan perintah berikut.

```
~/eb-docker-flask$ eb terminate
```

AWS sumber daya untuk aplikasi Anda

Anda baru saja membuat aplikasi instance tunggal. Ini berfungsi sebagai aplikasi sampel langsung dengan satu EC2 instance, sehingga tidak memerlukan penyeimbangan beban atau penskalaan otomatis. Untuk aplikasi contoh tunggal Elastic Beanstalk menciptakan sumber daya berikut: AWS

- EC2 instance — Mesin EC2 virtual Amazon yang dikonfigurasi untuk menjalankan aplikasi web pada platform yang Anda pilih.

  Setiap platform menjalankan satu set perangkat lunak, file konfigurasi, dan penulisan yang berbeda untuk mendukung versi bahasa, kerangka kerja, kontainer web tertentu, atau kombinasi dari semua ini. Sebagian besar platform menggunakan Apache atau nginx sebagai proksi terbalik yang memproses lalu lintas web di depan aplikasi web Anda, meneruskan permintaan ke aplikasi web, menyajikan aset statis, dan menghasilkan log akses dan kesalahan.
- Grup keamanan instans — Grup EC2 keamanan Amazon yang dikonfigurasi untuk memungkinkan lalu lintas masuk di port 80. Sumber daya ini memungkinkan lalu lintas HTTP dari penyeimbang beban mencapai EC2 instance yang menjalankan aplikasi web Anda. Secara default, lalu lintas tidak diizinkan di port lain.
- Bucket Amazon S3 – Lokasi penyimpanan untuk kode sumber, log, dan artifact lainnya yang dibuat saat Anda menggunakan Elastic Beanstalk.
- CloudWatch Alarm Amazon — Dua CloudWatch alarm yang memantau beban pada instans di lingkungan Anda dan dipicu jika beban terlalu tinggi atau terlalu rendah. Saat alarm terpicu, grup Auto Scaling Anda akan menaikkan atau menurunkan skala sebagai respons.
- AWS CloudFormation stack - Elastic AWS CloudFormation Beanstalk digunakan untuk meluncurkan sumber daya di lingkungan Anda dan menyebarkan perubahan konfigurasi. Sumber daya ditentukan di sebuah templat yang dapat Anda lihat di Konsol AWS CloudFormation.
- Nama domain — Nama domain yang merutekan ke aplikasi web Anda dalam formulir *`subdomain`*.*`region`*.elasticbeanstalk.com.

Elastic Beanstalk mengelola semua sumber daya tersebut. Ketika Anda mengakhiri lingkungan, Elastic Beanstalk mengakhiri semua sumber daya yang dimuatnya.

Langkah selanjutnya

Setelah lingkungan Anda menjalankan aplikasi, Anda dapat men-deploy versi baru aplikasi atau aplikasi yang berbeda kapan saja. Menyebarkan versi aplikasi baru sangat cepat karena tidak memerlukan penyediaan atau memulai ulang instance. EC2 Anda juga dapat menjelajahi lingkungan baru Anda menggunakan konsol Elastic Beanstalk. Untuk langkah-langkah mendetail, lihat Menjelajahi lingkungan Anda di bagian Memulai panduan ini.

Setelah Anda menerapkan satu atau dua contoh aplikasi dan siap untuk mulai mengembangkan dan menjalankan aplikasi Docker secara lokal, lihat. Mempersiapkan image Docker Anda untuk penyebaran ke Elastic Beanstalk

Terapkan dengan konsol Elastic Beanstalk

Anda juga dapat menggunakan konsol Elastic Beanstalk untuk meluncurkan aplikasi sampel. Untuk langkah-langkah rinci, lihat Membuat aplikasi contoh di Bab Memulai panduan ini.

Mempersiapkan image Docker Anda untuk penyebaran ke Elastic Beanstalk

Bagian ini menjelaskan cara menyiapkan image Docker Anda untuk penerapan ke Elastic Beanstalk dengan salah satu cabang platform Docker running atau 023. AL2 AL2 File konfigurasi yang Anda perlukan bergantung pada apakah gambar Anda lokal, jarak jauh, dan apakah Anda menggunakan Docker Compose.

> **Note**
>
> Untuk contoh prosedur yang meluncurkan lingkungan Docker, lihat topiknya. QuickStart untuk Docker

Topik

- Mengelola gambar Anda dengan Docker Compose di Elastic Beanstalk
- Mengelola gambar tanpa Docker Compose di Elastic Beanstalk
- Membangun gambar kustom dengan Dockerfile

Mengelola gambar Anda dengan Docker Compose di Elastic Beanstalk

Anda dapat memilih untuk menggunakan Docker Compose untuk mengelola berbagai layanan dalam satu file YAMB. Untuk mempelajari lebih lanjut tentang Docker Compose, lihat Mengapa menggunakan Compose? di situs web Docker.

- Buat a`docker-compose.yml`. File ini diperlukan jika Anda menggunakan Docker Compose untuk mengelola aplikasi Anda dengan Elastic Beanstalk. Jika semua penerapan Anda bersumber dari gambar di repositori publik, maka tidak ada file konfigurasi lain yang diperlukan. Jika gambar sumber penyebaran Anda berada di repositori pribadi, Anda harus melakukan beberapa konfigurasi tambahan. Untuk informasi selengkapnya, lihat Menggunakan gambar dari repositori pribadi. Untuk informasi lebih lanjut tentang file `docker-compose.yml`, lihat Buat referensi file di situs web Docker.

- `Dockerfile`Itu opsional. Buat satu jika Anda membutuhkan Elastic Beanstalk untuk membangun dan menjalankan gambar kustom lokal. Untuk informasi lebih lanjut tentang `Dockerfile`, lihat Referensi Dockerfile di situs web Docker.
- Anda mungkin perlu membuat `.zip` file. Jika Anda hanya menggunakan `Dockerfile` file untuk menyebarkan aplikasi Anda, Anda tidak perlu membuatnya. Jika Anda menggunakan file konfigurasi tambahan, file.zip harus menyertakan file`Dockerfile`, `docker-compose.yml` file aplikasi Anda, dan dependensi file aplikasi apa pun. The `Dockerfile` dan `docker-compose.yml` harus berada di root, atau tingkat atas, dari arsip.zip. Jika Anda menggunakan EB CLI untuk menyebarkan aplikasi Anda, itu membuat file.zip untuk Anda.

Untuk mempelajari lebih lanjut tentang Docker Compose dan cara menginstalnya, lihat situs DockerSekilas tentang Docker ComposedanPasang Docker Buat.

Mengelola gambar tanpa Docker Compose di Elastic Beanstalk

Jika Anda tidak menggunakan Docker Compose untuk mengelola image Docker, Anda harus mengonfigurasi `Dockerrun.aws.json` file`Dockerfile`, atau keduanya.

- Buat `Dockerfile` untuk membangun Elastic Beanstalk dan jalankan gambar khusus secara lokal.
- Buat file `Dockerrun.aws.json v1` untuk menyebarkan gambar Docker dari repositori host ke Elastic Beanstalk.
- Anda mungkin perlu membuat `.zip` file. Jika Anda hanya menggunakan salah satu dari salah satu file`Dockerrun.aws.json`, `Dockerfile` atau file, maka Anda tidak perlu membuat file.zip. Jika Anda menggunakan kedua file, maka Anda memerlukan file.zip. File.zip harus menyertakan file `Dockerfile` dan file`Dockerrun.aws.json`, bersama dengan file yang berisi file aplikasi Anda ditambah dependensi file aplikasi apa pun. Jika Anda menggunakan EB CLI untuk menyebarkan aplikasi Anda, langkah itu akan membuat file `.zip` untuk Anda.

`Dockerrun.aws.json`file konfigurasi v1

File `Dockerrun.aws.json` menjelaskan cara menggunakan gambar Docker jauh sebagai aplikasi Elastic Beanstalk. File JSON ini khusus untuk Elastic Beanstalk. Jika aplikasi Anda berjalan pada gambar yang tersedia di repositori host, Anda dapat menentukan gambar dalam file `Dockerrun.aws.json v1` dan menghapus `Dockerfile`.

Versi `Dockerrun.aws.json`

Parameter `AWSEBDockerrunVersion` menunjukkan versi file `Dockerrun.aws.json`.

- Platform Docker AL2 dan AL2 023 menggunakan versi file berikut.
  - `Dockerrun.aws.json v3`— lingkungan yang menggunakan Docker Compose.
  - `Dockerrun.aws.json v1`— lingkungan yang tidak menggunakan Docker Compose.
- ECS yang berjalan di Amazon Linux 2 dan ECS yang berjalan di AL2 023 menggunakan file tersebut. `Dockerrun.aws.json v2` Platform pensiunan ECS-Multicontainer Docker Amazon Linux AMI AL1 () juga menggunakan versi yang sama ini.

DockerRun.aws.json v1

Tombol dan nilai yang valid untuk file `Dockerrun.aws.json v1` tersebut meliputi operasi berikut ini:

AWSEBDockerrunVersi

(Wajib) Tentukan nomor versi 1 jika Anda tidak menggunakan Docker Compose untuk mengelola gambar Anda.

Autentikasi

(Diperlukan hanya untuk repositori pribadi) Menentukan objek Amazon S3 menyimpan file `.dockercfg`.

Lihat Menggunakan gambar dari repositori pribadi di Elastic Beanstalk di Menggunakan gambar dari repositori pribadi nanti di bagian ini.

image

Menentukan gambar dasar Docker pada repositori Docker yang ada dari mana Anda sedang membangun sebuah kontainer Docker. Tentukan nilai kunci Nama dalam format *`<organization>/<image name>`* untuk gambar di Docker Hub, atau *`<site>/<organization name>/<image name>`* untuk situs lain.

Bila Anda menentukan gambar pada file `Dockerrun.aws.json`, setiap contoh di lingkungan Elastic Beanstalk Anda menjalankan `docker pull` untuk menjalankan gambar. Opsional,

termasuk kunci Update. Nilai defaultnya adalah`true`dan menginstruksikan Elastic Beanstalk untuk memeriksa repositori, menarik setiap update ke gambar, dan menimpa gambar cache.

Saat menggunakan `Dockerfile`, tidak boleh menentukan kunci imagekunci di file `Dockerrun.aws.json`. Elastic Beanstalk selalu membuat dan menggunakan gambar yang dijelaskan dalam`Dockerfile`ketika salah satu hadir.

Port

(Diperlukan saat Anda menentukan kunci image) Cantumkan port untuk mengekspos pada kontainer Docker. Elastic Beanstalk ContainerPortmenggunakan nilai untuk menghubungkan wadah Docker ke proxy terbalik yang berjalan di host.

Anda dapat menentukan beberapa port kontainer, tapi Elastic Beanstalk hanya menggunakan port pertama. Menggunakan port ini untuk menghubungkan kontainer Anda ke reverse proxy host dan rute permintaan dari internet publik. Jika Anda menggunakan a`Dockerfile`, ContainerPortnilai pertama harus cocok dengan entri pertama dalam daftar EXPOSE. `Dockerfile`

Secara opsional, Anda dapat menentukan daftar port di HostPort. HostPortentri menentukan port host yang ContainerPortnilainya dipetakan. Jika Anda tidak menentukan HostPortnilai, nilai defaultnya. ContainerPort

```
{
  "Image": {
    "Name": "image-name"
  },
  "Ports": [
    {
      "ContainerPort": 8080,
      "HostPort": 8000
    }
  ]
}
```

Volume

Memetakan volume dari EC2 instance ke container Docker Anda. Menentukan satu atau lebih array volume untuk dipetakan.

```
{
```

```
  "Volumes": [
    {
      "HostDirectory": "/path/inside/host",
      "ContainerDirectory": "/path/inside/container"
    }
  ]
...
```

Pencatatan log

Menentukan direktori di dalam kontainer tempat aplikasi Anda menulis log. Elastic Beanstalk mengunggah log apapun dalam direktori ini ke Amazon S3 ketika Anda meminta log tail atau bundel log. Jika Anda memutar log ke folder bernama`rotated`dalam direktori ini, Anda juga dapat mengkonfigurasi Elastic Beanstalk untuk mengunggah log diputar ke Amazon S3 untuk penyimpanan permanen. Untuk informasi selengkapnya, lihat Melihat log dari EC2 instans Amazon di lingkungan Elastic Beanstalk Anda.

, , perintah

Menentukan perintah untuk dijalankan dalam kontainer. Jika Anda menentukanEntrypoint, kemudianPerintahditambahkan sebagai argumen untukEntrypoint. Untuk informasi lebih lanjut, lihatCMDdalam dokumentasi Docker.

Entrypoint

Tentukan perintah default untuk menjalankan ketika kontainer dimulai. Untuk informasi lebih lanjut, lihatENTRYPOINTdalam dokumentasi Docker.

Potongan gambar berikut adalah contoh yang menunjukkan sintaksfile `Dockerrun.aws.json` untuk kontainer tunggal.

```
{
  "AWSEBDockerrunVersion": "1",
  "Image": {
    "Name": "janedoe/image",
    "Update": "true"
  },
  "Ports": [
    {
      "ContainerPort": "1234"
    }
  ],
```

```
  "Volumes": [
    {
      "HostDirectory": "/var/app/mydb",
      "ContainerDirectory": "/etc/mysql"
    }
  ],
  "Logging": "/var/log/nginx",
  "Entrypoint": "/app/bin/myapp",
  "Command": "--argument"
}>
```

Anda dapat menyediakan Elastic Beanstalk dengan hanya file `Dockerrun.aws.json`, atau dengan arsip `.zip` yang berisi baik file `Dockerrun.aws.json` maupun `Dockerfile`. Ketika Anda memberikan kedua file, file `Dockerfile` menjelaskan gambar Docker dan file `Dockerrun.aws.json` menyediakan informasi tambahan untuk deployment, seperti yang akan dijelaskan dalam bagian ini.

> **Note**
>
> Dua file harus berada di root, atau tingkat atas, dari arsip `.zip`. Jangan membuat arsip dari direktori yang berisi berkas. Sebaliknya, navigasikan ke direktori itu dan buat arsip di sana. Ketika Anda menyediakan kedua file, tidak menentukan gambar di file `Dockerrun.aws.json`. Elastic Beanstalk membuat dan menggunakan gambar yang dijelaskan dalam `Dockerfile` dan mengabaikan gambar yang ditentukan dalam file `Dockerrun.aws.json`.

Membangun gambar kustom dengan Dockerfile

Anda harus membuat `Dockerfile` Jika Anda belum memiliki image yang ada di repositori.

potongan berikut ini adalah contoh file `Dockerfile`. Jika Anda mengikuti petunjuk di QuickStart untuk Docker, Anda dapat mengunggah `Dockerfile`seperti tertulis. Elastic Beanstalk menjalankan permainan 2048 ketika Anda menggunakan ini `Dockerfile`.

Untuk informasi lebih lanjut tentang petunjuk yang dapat Anda sertakan dalam `Dockerfile`, lihatReferensi filedi situs web Docker.

```
FROM ubuntu:12.04
```

```
RUN apt-get update
RUN apt-get install -y nginx zip curl

RUN echo "daemon off;" >> /etc/nginx/nginx.conf
RUN curl -o /usr/share/nginx/www/master.zip -L https://codeload.github.com/
gabrielecirulli/2048/zip/master
RUN cd /usr/share/nginx/www/ && unzip master.zip && mv 2048-master/* . && rm -rf 2048-
master master.zip

EXPOSE 80

CMD ["/usr/sbin/nginx", "-c", "/etc/nginx/nginx.conf"]
```

> **Note**
>
> Anda dapat menjalankan build multi-tahap dari satu Dockerfile untuk menghasilkan gambar berukuran lebih kecil dengan pengurangan kompleksitas yang signifikan. Untuk informasi selengkapnya, lihat Menggunakan build multi-tahap di situs web dokumentasi Docker.

Menggunakan cabang platform Docker yang dikelola ECS di Elastic Beanstalk

Topik ini memberikan gambaran umum tentang cabang platform Docker yang dikelola Elastic Beanstalk ECS untuk Amazon Linux 2 dan Amazon Linux 2023. Ini juga menyediakan informasi konfigurasi yang khusus untuk platform yang dikelola Docker ECS.

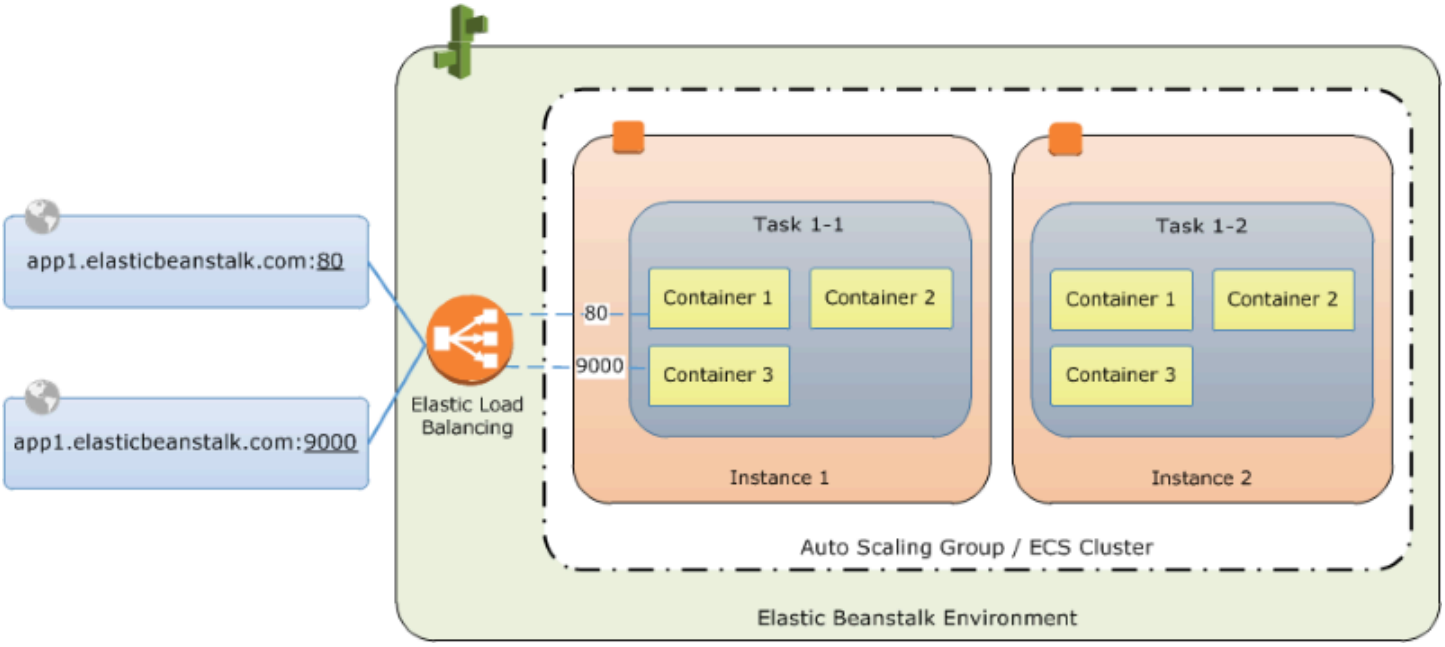
Migrasi dari Multi-container Docker di AL1

Pada tanggal 18 Juli 2022, Elastic Beanstalk menetapkan status semua cabang platform berdasarkan Amazon Linux AMI () untuk pensiun. AL1 Meskipun bagian ini menyediakan informasi konfigurasi untuk platform yang sudah pensiun ini, kami sangat menyarankan Anda untuk bermigrasi ke cabang platform terbaru yang didukung. Jika saat ini Anda menggunakan Docker Multi-kontainer yang sudah pensiun yang berjalan di cabang AL1 platform, Anda dapat bermigrasi ke ECS Running terbaru di cabang platform 023. AL2 Cabang platform terbaru mendukung semua fitur dari cabang platform yang dihentikan. Tidak ada perubahan pada kode sumber yang diperlukan. Untuk informasi selengkapnya, lihat Migrasi aplikasi Elastic Beanstalk Anda dari Docker Multi-container yang dikelola ECS ke ECS di Amazon Linux 2023 AL1 .

Ikhtisar platform Docker yang dikelola ECS

Elastic Beanstalk menggunakan Amazon Elastic Container Service (Amazon ECS) Container Service (Amazon ECS) untuk mengoordinasikan penerapan kontainer ke lingkungan Docker yang dikelola ECS. Amazon ECS menyediakan alat untuk mengelola sekelompok contoh yang menjalankan kontainer Docker. Elastic Beanstalk mengurus tugas-tugas Amazon ECS termasuk penciptaan cluster, definisi tugas dan eksekusi. Setiap instance di lingkungan menjalankan kumpulan kontainer yang sama, yang didefinisikan dalam file `Dockerrun.aws.json` v2. Untuk mendapatkan hasil maksimal dari Docker, Elastic Beanstalk memungkinkan Anda membuat lingkungan tempat instance EC2 Amazon Anda menjalankan beberapa kontainer Docker secara berdampingan.

Diagram berikut menunjukkan contoh lingkungan Elastic Beanstalk yang dikonfigurasi dengan tiga kontainer Docker yang berjalan di EC2 setiap instans Amazon dalam grup Auto Scaling:



Sumber daya Amazon ECS dibuat oleh Elastic Beanstalk

Saat Anda membuat lingkungan menggunakan platform Docker terkelola ECS, Elastic Beanstalk secara otomatis membuat dan mengonfigurasi beberapa sumber daya Amazon Elastic Container Service saat membangun lingkungan. Dengan melakukan itu, ia membuat wadah yang diperlukan pada setiap EC2 instance Amazon.

- Kluster Amazon ECS- Contoh kontainer di Amazon ECS diatur ke dalam kluster. Saat digunakan dengan Elastic Beanstalk, satu cluster selalu dibuat untuk setiap lingkungan Docker yang dikelola ECS. Cluster ECS juga berisi penyedia kapasitas grup Auto Scaling dan sumber daya lainnya.

- Definisi Tugas Amazon ECS - Elastic Beanstalk `Dockerrun.aws.json` menggunakan v2 dalam project Anda untuk menghasilkan definisi tugas Amazon ECS yang digunakan untuk mengonfigurasi instance container di lingkungan.
- Tugas Amazon ECS- Elastic Beanstalk berkomunikasi dengan Amazon ECS untuk menjalankan tugas pada setiap contoh di lingkungan untuk mengkoordinasikan deployment kontainer. Dalam lingkungan yang terukur, Elastic Beanstalk memulai tugas baru setiap kali contoh ditambahkan ke kluster.
- Agen Kontainer Amazon ECS— Agen berjalan dalam kontainer Docker pada contoh di lingkungan Anda. Agen melakukan jajak pendapat layanan Amazon ECS dan menunggu tugas untuk dijalankan.
- Volume Data Amazon ECS — Selain volume yang Anda tentukan di `Dockerrun.aws.json` v2, Elastic Beanstalk menyisipkan definisi volume ke dalam definisi tugas untuk memfasilitasi pengumpulan log.

  Elastic Beanstalk membuat volume log pada contoh kontainer, satu untuk setiap kontainer, di `/var/log/containers/`*`containername`*. Volume ini diberi nama `awseb-logs-`*`containername`* dan disediakan untuk kontainer untuk dipasang. Lihat Format definisi kontainer untuk rincian bagaimana cara memasangnya.

Untuk informasi selengkapnya tentang sumber daya Amazon ECS, lihat Panduan Pengembang Layanan Kontainer Elastis Amazon.

`Dockerrun.aws.json`berkas v2

Instance kontainer memerlukan file konfigurasi bernama`Dockerrun.aws.json`. Instans kontainer mengacu pada instans Amazon yang menjalankan Docker EC2 terkelola ECS di lingkungan Elastic Beanstalk. File ini khusus untuk Elastic Beanstalk dan dapat digunakan sendiri atau dikombinasikan dengan kode sumber dan konten dalam bundel sumber untuk menciptakan lingkungan pada platform Docker.

> **Note**
>
> Format Versi 2 `Dockerrun.aws.json` menambahkan dukungan untuk beberapa kontainer per EC2 instans Amazon dan hanya dapat digunakan dengan platform Docker yang dikelola ECS. Formatnya berbeda secara signifikan dari versi file konfigurasi lain yang mendukung cabang platform Docker yang tidak dikelola oleh ECS.

Lihat detail tentang format yang diperbarui dan file contoh. `Dockerrun.aws.json v2`

Gambar Docker

Platform Docker yang dikelola ECS untuk Elastic Beanstalk membutuhkan gambar untuk dibuat sebelumnya dan disimpan dalam repositori gambar online publik atau pribadi sebelum membuat lingkungan Elastic Beanstalk.

> **Note**
>
> Membangun gambar khusus selama penerapan dengan a tidak `Dockerfile` didukung oleh platform Docker yang dikelola ECS di Elastic Beanstalk. Membuat gambar Anda dan menerapkannya ke repositori online sebelum membuat lingkungan Elastic Beanstalk.

Tentukan gambar dengan nama di `Dockerrun.aws.json` v2.

Untuk mengonfigurasi Elastic Beanstalk untuk mengautentikasi ke repositori pribadi, sertakan parameter dalam file v2 Anda. `authentication` `Dockerrun.aws.json`

Deployment kontainer gagal

Jika tugas Amazon ECS gagal, satu atau lebih kontainer di lingkungan Elastic Beanstalk Anda tidak akan mulai. Elastic Beanstalk tidak memutar kembali lingkungan multi-kontainer karena tugas Amazon ECS yang gagal. Jika kontainer gagal dimulai di lingkungan Anda, terapkan ulang versi saat ini atau versi sebelumnya yang bekerja dari konsol Elastic Beanstalk.

Untuk menggunakan versi yang sudah ada

1. Buka konsol Elastic Beanstalk di wilayah lingkungan Anda.
2. KlikTindakandi sebelah kanan nama aplikasi Anda dan kemudian klikLihat versi aplikasi.
3. Pilih versi aplikasi Anda dan klikDeploy.

Memperluas platform Docker berbasis ECS untuk Elastic Beanstalk

Elastic Beanstalk menawarkan fitur ekstensibilitas yang memungkinkan Anda menerapkan perintah, skrip, perangkat lunak, dan konfigurasi Anda sendiri ke penerapan aplikasi Anda. Alur kerja penerapan untuk cabang platform ECS AL2 dan AL2 023 sedikit berbeda dari platform berbasis Linux

lainnya. Untuk informasi selengkapnya, lihat Alur kerja penerapan instans untuk ECS yang berjalan di Amazon Linux 2 dan yang lebih baru.

Konfigurasi Docker terkelola ECS untuk Elastic Beanstalk

Bab ini menjelaskan cara mengonfigurasi lingkungan Docker terkelola ECS Anda. Daftar berikut merangkum item konfigurasi yang dijelaskan oleh Bab ini.

- `Dockerrun.aws.json`v2 - File konfigurasi ini menentukan repositori gambar Anda dan nama gambar Docker Anda, di antara komponen lainnya.
- EC2 Peran profil instans — Jika Anda memiliki profil instans khusus, kami menjelaskan cara mengonfigurasinya sehingga izin yang diperlukan ECS untuk mengelola kontainer Anda tetap terkini.
- Pendengar Elastic Load Balancing — Anda harus mengonfigurasi beberapa pendengar Elastic Load Balancing jika Anda membutuhkan lingkungan untuk mendukung lalu lintas masuk untuk proxy atau layanan lain yang tidak berjalan pada port HTTP default.

Topik

- Mengkonfigurasi file DockerRun.aws.json v2
- Kebijakan dan peran EC2 instans yang dikelola kontainer
- Menggunakan beberapa pendengar Elastic Load Balancing

Mengkonfigurasi file DockerRun.aws.json v2

`Dockerrun.aws.json v2`adalah file konfigurasi Elastic Beanstalk yang menjelaskan cara menerapkan satu set wadah Docker yang dihosting di cluster ECS di lingkungan Elastic Beanstalk. Platform Elastic Beanstalk menciptakan definisi tugas ECS, yang mencakup definisi wadah ECS. Definisi ini dijelaskan dalam file `Dockerrun.aws.json` konfigurasi.

Definisi kontainer dalam `Dockerrun.aws.json` file menjelaskan kontainer yang akan diterapkan ke setiap EC2 instans Amazon di cluster ECS. Dalam hal ini EC2 instance Amazon juga disebut sebagai instance container host, karena instance ini meng-host container Docker. File konfigurasi juga menjelaskan volume data yang akan dibuat pada instance wadah host agar kontainer Docker dipasang. Untuk informasi lebih lanjut dan diagram komponen dalam lingkungan Docker yang dikelola ECS di Elastic Beanstalk, lihat sebelumnya di chapter ini. Ikhtisar platform Docker yang dikelola ECS

File `Dockerrun.aws.json` dapat digunakan sendiri atau di-zip dengan kode sumber tambahan dalam satu arsip. Kode sumber yang diarsipkan dengan a `Dockerrun.aws.json` diterapkan ke instance EC2 penampung Amazon dan dapat diakses di direktori. `/var/app/current/`

Topik

- Dockerrun.aws.json v2
- Format volume
- Format ARN Peran Eksekusi
- Format definisi kontainer
- Format otentikasi — menggunakan gambar dari repositori pribadi
- Contoh DockerRun.aws.json v2

`Dockerrun.aws.json` v2

`Dockerrun.aws.json`File ini mencakup bagian-bagian berikut:

AWSEBDockerrunVersi

Menentukan nomor versi sebagai nilai 2 untuk lingkungan Docker yang dikelola ECS.

executionRoleArn

Menentukan peran IAM pelaksanaan tugas untuk berbagai tujuan dan layanan yang terkait dengan akun Anda. Agar aplikasi Anda menggunakan variabel lingkungan Elastic Beanstalk yang disimpan sebagai rahasia, Anda harus menentukan ARN dari peran eksekusi tugas yang memberikan izin yang diperlukan. Kasus penggunaan umum lainnya mungkin juga memerlukan parameter ini. Untuk informasi selengkapnya, lihat Format ARN Peran Eksekusi.

volume

Membuat volume dari folder di instans EC2 penampung Amazon, atau dari bundel sumber Anda (digunakan ke`/var/app/current`). Pasang volume ini ke jalur dalam wadah Docker Anda menggunakan `mountPoints` di `containerDefinitions` bagian.

containerDefinitions

Array definisi kontainer.

otentikasi (opsional)

Lokasi di Amazon S3 dari `.dockercfg` file yang berisi data otentikasi untuk repositori pribadi.

Bagian ContainerDefinitions dan volume menggunakan pemformatan yang sama dengan bagian yang sesuai dari file definisi tugas Amazon ECS. `Dockerrun.aws.json` Untuk informasi selengkapnya tentang format definisi tugas dan daftar lengkap parameter definisi tugas, lihat definisi tugas Amazon ECS di Panduan Pengembang Layanan Kontainer Elastis Amazon.

Format volume

Parameter volume membuat volume dari salah satu folder dalam instance EC2 penampung Amazon, atau dari bundel sumber Anda (digunakan ke`/var/app/current`).

Volume ditentukan dalam format berikut:

```
"volumes": [
    {
      "name": "volumename",
      "host": {
        "sourcePath": "/path/on/host/instance"
      }
    }
  ],
```

Pasang volume ini ke jalur dalam kontainer Docker Anda menggunakan `mountPoints` definisi kontainer.

Elastic Beanstalk mengkonfigurasi volume tambahan untuk log, satu untuk setiap kontainer. Ini harus dipasang oleh kontainer Docker Anda untuk menulis log ke contoh host.

Untuk detail selengkapnya, lihat `mountPoints` bidang di bagian Format definisi kontainer berikut.

Format ARN Peran Eksekusi

Agar aplikasi Anda menggunakan variabel lingkungan Elastic Beanstalk yang disimpan sebagai rahasia, Anda harus menentukan peran IAM eksekusi tugas. Peran harus memberikan izin penampung Amazon ECS untuk melakukan panggilan AWS API atas nama Anda menggunakan AWS Secrets Manager rahasia atau AWS Systems Manager parameter Parameter Store untuk mereferensikan data sensitif. Untuk petunjuk untuk membuat peran IAM eksekusi tugas dengan izin yang diperlukan untuk akun Anda, lihat peran IAM eksekusi tugas Amazon ECS di Panduan Pengembang Layanan Amazon Elastic Container.

```
{
"AWSEBDockerrunVersion": 2,
```

```
  "executionRoleArn": "arn:aws:iam::111122223333:role/ecsTaskExecutionRole",
```

Izin tambahan diperlukan untuk platform Docker yang dikelola Amazon ECS

EC2 hibah profil instance **`iam:PassRole`** ke ECS

Agar profil EC2 instans Anda dapat memberikan peran ini ke wadah ECS, Anda harus menyertakan `iam:PassRole` izin yang ditunjukkan dalam contoh berikut. `iam:PassRole`Ini memungkinkan izin EC2 instance untuk meneruskan peran eksekusi tugas ke wadah ECS.

Dalam contoh ini, kami membatasi EC2 instance untuk hanya meneruskan peran ke layanan ECS. Meskipun kondisi ini tidak diperlukan, kami menambahkannya untuk mengikuti praktik terbaik untuk mengurangi ruang lingkup izin yang dibagikan. Kami mencapai ini dengan `Condition` elemen.

> **Note**
>
> Setiap penggunaan peran eksekusi tugas ECS IAM memerlukan izin. `iam:PassRole`
> Ada kasus penggunaan umum lainnya yang memerlukan peran layanan terkelola eksekusi tugas ECS. Untuk informasi selengkapnya, lihat peran IAM eksekusi tugas Amazon ECS di Panduan Pengembang Layanan Kontainer Elastis Amazon.

Example kebijakan dengan **`iam:PassRole`** izin

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": [
                "arn:aws:iam::123456789012:role/ecs-task-execution-role"
            ],
            "Condition": {
                "StringLike": {
                    "iam:PassedToService": "ecs-tasks.amazonaws.com"
                }
            }
        }
    ]
```

```
}
```

Memberikan rahasia dan parameter akses ke agen kontainer Amazon ECS

Peran IAM eksekusi tugas Amazon ECS juga memerlukan izin untuk mengakses rahasia dan penyimpanan parameter. Serupa dengan persyaratan peran profil EC2 instans, agen kontainer ECS memerlukan izin untuk menarik sumber daya Secrets Manager atau Systems Manager yang diperlukan. Untuk informasi selengkapnya, lihat izin Secrets Manager atau Systems Manager di Panduan Pengembang Layanan Amazon Elastic Container

Memberikan rahasia dan parameter akses ke instance Elastic Beanstalk EC2

Untuk mendukung rahasia yang dikonfigurasi sebagai variabel lingkungan, Anda juga perlu menambahkan izin ke profil EC2 instans Anda. Untuk informasi selengkapnya, lihat Mengambil rahasia dan parameter ke variabel lingkungan Elastic Beanstalk dan Izin IAM yang diperlukan untuk Secrets Manager.

Contoh berikut menggabungkan contoh sebelumnya dengan `iam:PassRole` contoh yang diberikan dalam referensiIzin IAM yang diperlukan untuk Secrets Manager. Mereka menambahkan izin yang diperlukan EC2 instance untuk mengakses AWS Secrets Manager dan AWS Systems Manager menyimpan untuk mengambil rahasia dan data parameter untuk menginisialisasi variabel lingkungan Elastic Beanstalk yang telah dikonfigurasi untuk rahasia.

Example Kebijakan Secrets Manager dikombinasikan dengan **`iam:PassRole`** izin

```
{
    "Version": "2012-10-17",
    "Statement": [
       {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": [
                "arn:aws:iam::123456789012:role/ecs-task-execution-role"
            ],
            "Condition": {
                "StringLike": {
                    "iam:PassedToService": "ecs-tasks.amazonaws.com"
                }
            }
        },
        {
```

```
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetSecretValue",
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:secretsmanager:us-east-1:111122223333:secret:my-secret",
                "arn:aws:kms:us-east-1:111122223333:key/my-key"
            ]
        }
    ]
}
```

Example Kebijakan Systems Manager dikombinasikan dengan **`iam:PassRole`** izin

```
{
    "Version": "2012-10-17",
    "Statement": [
       {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": [
                "arn:aws:iam::123456789012:role/ecs-task-execution-role"
            ],
            "Condition": {
                "StringLike": {
                    "iam:PassedToService": "ecs-tasks.amazonaws.com"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetParameter",
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:ssm:us-east-1:111122223333:parameter/my-parameter",
                "arn:aws:kms:us-east-1:111122223333:key/my-key"
            ]
        }
    ]
}
```

Format definisi kontainer

Contoh berikut menunjukkan subset parameter yang umum digunakan di bagian ContainerDefinitions. Parameter opsional juga tersedia.

Platform Beanstalk menciptakan definisi tugas ECS, yang mencakup definisi wadah ECS. Beanstalk mendukung sub-set parameter untuk definisi wadah ECS. Untuk informasi selengkapnya, lihat Definisi kontainer di Panduan Pengembang Layanan Kontainer Elastis Amazon.

File `Dockerrun.aws.json` berisi satu baris atau object yang lain dari definisi kontainer dengan bidang berikut:

nama

Nama kontainer. Lihat Parameter Definisi Kontainer Standar untuk informasi tentang panjang maksimum dan karakter yang diizinkan.

gambar

Nama gambar Docker dalam repositori Docker online tempat Anda membuat container Docker. Perhatikan konvensi ini:

- Gambar di repositori resmi di Docker Hub menggunakan satu nama (misalnya, `ubuntu` atau `mongo`).
- Gambar di repositori lain di Docker Hub memenuhi syarat dengan nama organisasi (misalnya, `amazon/amazon-ecs-agent`.
- Gambar di repositori online lainnya memenuhi syarat lebih lanjut dengan nama domain (misalnya, `quay.io/assemblyline/ubuntu`).

Lingkungan

Sebuah susunan variabel lingkungan untuk melewati container

Sebagai contoh, entri berikut mendefinisikan variabel lingkungan dengan nama **`Container`** dan nilai **`PHP`**:

```
"environment": [
  {
    "name": "Container",
    "value": "PHP"
  }
```

```
],
```

penting

Benar jika tugas harus berhenti jika kontainer gagal. Kontainer yang tidak penting dapat selesai atau mogok tanpa memengaruhi container lainnya pada contoh.

memori

Jumlah memori pada contoh kontainer untuk cadangan untuk kontainer. Tentukan bilangan bulat bukan nol untuk salah satu atau kedua parameter `memory` atau `memoryReservation` dalam definisi kontainer.

memoryReservation

Batas lunak (di MiB) memori untuk cadangan untuk kontainer. Tentukan bilangan bulat bukan nol untuk salah satu atau kedua parameter `memory` atau `memoryReservation` dalam definisi kontainer.

mountPoints

Volume dari instance EC2 penampung Amazon untuk dipasang, dan lokasi pada sistem file kontainer Docker untuk memasangnya. Saat Anda memasang volume yang berisi konten aplikasi, kontainer Anda dapat membaca data yang Anda unggah di bundel sumber Anda. Ketika Anda memasang log volume untuk menulis log data, Elastic Beanstalk dapat mengumpulkan data log dari volume ini.

Elastic Beanstalk membuat volume log pada contoh kontainer, satu untuk setiap kontainer Docker, di `/var/log/containers/`*`containername`*. Volume ini diberi nama `awseb-logs-`*`containername`* dan harus dipasang ke lokasi dalam struktur file kontainer di mana log ditulis.

Misalnya, titik pemasangan berikut memetakan lokasi log nginx dalam kontainer ke volume yang dihasilkan Elastic Beanstalk untuk kontainer `nginx-proxy`.

```
{
  "sourceVolume": "awseb-logs-nginx-proxy",
  "containerPath": "/var/log/nginx"
}
```

portMappings

Port jaringan peta pada kontainer untuk port pada host.

Tautan

Daftar kontainer untuk ditautkan. kontainer yang ditautkan dapat saling menemukan dan berkomunikasi dengan aman.

volumesFrom

Pasang semua volume dari kontainer yang berbeda. Misalnya, untuk memasang volume dari kontainer bernama `web`:

```
"volumesFrom": [
  {
    "sourceContainer": "web"
  }
],
```

Format otentikasi — menggunakan gambar dari repositori pribadi

`authentication`Bagian ini berisi data otentikasi untuk repositori pribadi. Entri ini opsional.

Tambahkan informasi tentang bucket Amazon S3 yang berisi berkas otentikasi di parameter `authentication` dari file `Dockerrun.aws.json`. Pastikan bahwa parameter `authentication` berisi bucket Amazon S3 yang valid dan kunci. bucket Amazon S3 harus di-hosting di wilayah yang sama dengan lingkungan yang menggunakannya. Elastic Beanstalk tidak akan mengunduh file dari Amazon S3 bucket yang dihosting di wilayah lain.

Menggunakan format berikut:

```
"authentication": {
    "bucket": "amzn-s3-demo-bucket",
    "key": "mydockercfg"
  },
```

Untuk informasi tentang menghasilkan dan mengunggah file otentikasi, lihat Menggunakan gambar dari repositori pribadi di Elastic Beanstalk.

Contoh DockerRun.aws.json v2

Potongan berikut adalah contoh yang menggambarkan sintaks file `Dockerrun.aws.json` untuk sebuah contoh dengan dua kontainer.

```
{
```

```
  "AWSEBDockerrunVersion": 2,
  "volumes": [
    {
      "name": "php-app",
      "host": {
        "sourcePath": "/var/app/current/php-app"
      }
    },
    {
      "name": "nginx-proxy-conf",
      "host": {
        "sourcePath": "/var/app/current/proxy/conf.d"
      }
    }
  ],
  "containerDefinitions": [
    {
      "name": "php-app",
      "image": "php:fpm",
      "environment": [
        {
          "name": "Container",
          "value": "PHP"
        }
      ],
      "essential": true,
      "memory": 128,
      "mountPoints": [
        {
          "sourceVolume": "php-app",
          "containerPath": "/var/www/html",
          "readOnly": true
        }
      ]
    },
    {
      "name": "nginx-proxy",
      "image": "nginx",
      "essential": true,
      "memory": 128,
      "portMappings": [
        {
          "hostPort": 80,
          "containerPort": 80
```

```
        }
      ],
      "links": [
        "php-app"
      ],
      "mountPoints": [
        {
          "sourceVolume": "php-app",
          "containerPath": "/var/www/html",
          "readOnly": true
        },
        {
          "sourceVolume": "nginx-proxy-conf",
          "containerPath": "/etc/nginx/conf.d",
          "readOnly": true
        },
        {
          "sourceVolume": "awseb-logs-nginx-proxy",
          "containerPath": "/var/log/nginx"
        }
      ]
    }
  ]
}
```

Kebijakan dan peran EC2 instans yang dikelola kontainer

Saat Anda membuat lingkungan di konsol Elastic Beanstalk, ini akan meminta Anda untuk membuat profil instance default yang menyertakan kebijakan terkelola. `AWSElasticBeanstalkMulticontainerDocker` Jadi awalnya, profil EC2 instance default Anda, harus menyertakan kebijakan terkelola ini. Jika lingkungan Anda menggunakan peran profil EC2 instans kustom alih-alih default, pastikan kebijakan `AWSElasticBeanstalkMulticontainerDocker` terkelola dilampirkan sehingga izin yang diperlukan untuk pengelolaan kontainer tetap up-to-date ada.

Elastic Beanstalk menggunakan Amazon ECS-dioptimalkan AMI dengan agen kontainer Amazon ECS yang berjalan dalam Docker container. Agen berkomunikasi dengan Amazon ECS untuk mengkoordinasikan deployment kontainer. Untuk berkomunikasi dengan Amazon ECS, setiap EC2 instans Amazon harus memiliki izin IAM yang sesuai, yang ditentukan dalam kebijakan terkelola ini. Lihat AWSElasticBeanstalkMulticontainerDockerdi Panduan Referensi Kebijakan AWS Terkelola untuk melihat izin ini.

Jika Anda menggunakan variabel lingkungan Elastic Beanstalk yang dikonfigurasi untuk mengakses rahasia atau parameter yang AWS Secrets Manager disimpan AWS Systems Manager di atau Parameter Store, Anda harus EC2 menyesuaikan profil instans Anda dengan izin tambahan. Untuk informasi selengkapnya, lihat Format ARN Peran Eksekusi.

Menggunakan beberapa pendengar Elastic Load Balancing

Anda dapat mengonfigurasi beberapa pendengar Elastic Load Balancing di lingkungan Docker yang dikelola ECS untuk mendukung lalu lintas masuk untuk proxy atau layanan lain yang tidak berjalan di port HTTP default.

Buat folder `.ebextensions` di bundel sumber Anda dan tambahkan file dengan ekstensi file `.config`. Contoh berikut menunjukkan file konfigurasi yang menciptakan pendengar Elastic Load Balancing pada port 8080.

`.ebextensions/elb-listener.config`

```
option_settings:
  aws:elb:listener:8080:
    ListenerProtocol: HTTP
    InstanceProtocol: HTTP
    InstancePort: 8080
```

Jika lingkungan Anda berjalan di kustomAmazon Virtual Private Cloud(Amazon VPC) yang Anda buat, Elastic Beanstalk mengurus sisanya. Dalam VPC default, Anda perlu mengkonfigurasi grup keamanan instans Anda untuk memungkinkan masuknya dari load balancer. Menambahkan file konfigurasi kedua yang menambahkan aturan masuknya ke grup keamanan:

`.ebextensions/elb-ingress.config`

```
Resources:
  port8080SecurityGroupIngress:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: {"Fn::GetAtt" : ["AWSEBSecurityGroup", "GroupId"]}
      IpProtocol: tcp
      ToPort: 8080
      FromPort: 8080
      SourceSecurityGroupName: { "Fn::GetAtt": ["AWSEBLoadBalancer",
 "SourceSecurityGroup.GroupName"] }
```

Untuk informasi selengkapnya tentang format file konfigurasi, lihat Menambahkan dan menyesuaikan sumber daya lingkungan Elastic Beanstalk dan Pengaturan opsi.

Selain menambahkan listener ke konfigurasi Elastic Load Balancing dan membuka port di grup keamanan, Anda perlu memetakan port pada instance host ke port pada container Docker di bagian `containerDefinitions` `Dockerrun.aws.json` file v2. Kebijakan berikut menunjukkan sebuah contoh.

```
"portMappings": [
  {
    "hostPort": 8080,
    "containerPort": 8080
  }
]
```

Lihat `Dockerrun.aws.json` v2 untuk detail tentang format file `Dockerrun.aws.json` v2.

Membuat lingkungan Docker yang dikelola ECS dengan konsol Elastic Beanstalk

Tutorial ini merinci konfigurasi kontainer dan persiapan kode sumber untuk lingkungan Docker yang dikelola ECS yang menggunakan dua kontainer.

Container, aplikasi PHP dan proxy nginx, berjalan berdampingan di setiap instance Amazon Elastic Compute Cloud (EC2Amazon) di lingkungan Elastic Beanstalk. Setelah membuat lingkungan dan memverifikasi bahwa aplikasi berjalan, Anda akan terhubung ke contoh kontainer untuk melihat bagaimana semuanya cocok bersama-sama.

Bagian-bagian

- Tentukan wadah Docker yang dikelola ECS
- Tambahkan konten
- Menerapkan ke Elastic Beanstalk
- Hubungkan ke instans kontainer
- Periksa agen kontainer Amazon ECS

Tentukan wadah Docker yang dikelola ECS

Langkah pertama dalam menciptakan lingkungan Docker baru adalah untuk membuat sebuah direktori untuk data aplikasi Anda. Folder ini dapat ditemukan di mana saja pada mesin lokal Anda dan memiliki nama apapun yang Anda pilih. Selain file konfigurasi kontainer, folder ini akan berisi konten yang akan Anda upload ke Elastic Beanstalk dan menerapkannya ke lingkungan Anda.

> **Note**
>
> Semua kode untuk tutorial ini tersedia di repositori awslabs di. GitHub https://github.com/awslabs/eb-docker-nginx-proxy

File yang digunakan Elastic Beanstalk untuk mengonfigurasi wadah pada instance EC2 Amazon adalah file teks berformat JSON bernama v2. `Dockerrun.aws.json` Versi platform Docker yang dikelola ECS menggunakan format Versi 2 dari file ini. Format ini hanya dapat digunakan dengan platform Docker yang dikelola ECS, karena berbeda secara signifikan dari versi file konfigurasi lain yang mendukung cabang platform Docker yang tidak dikelola oleh ECS.

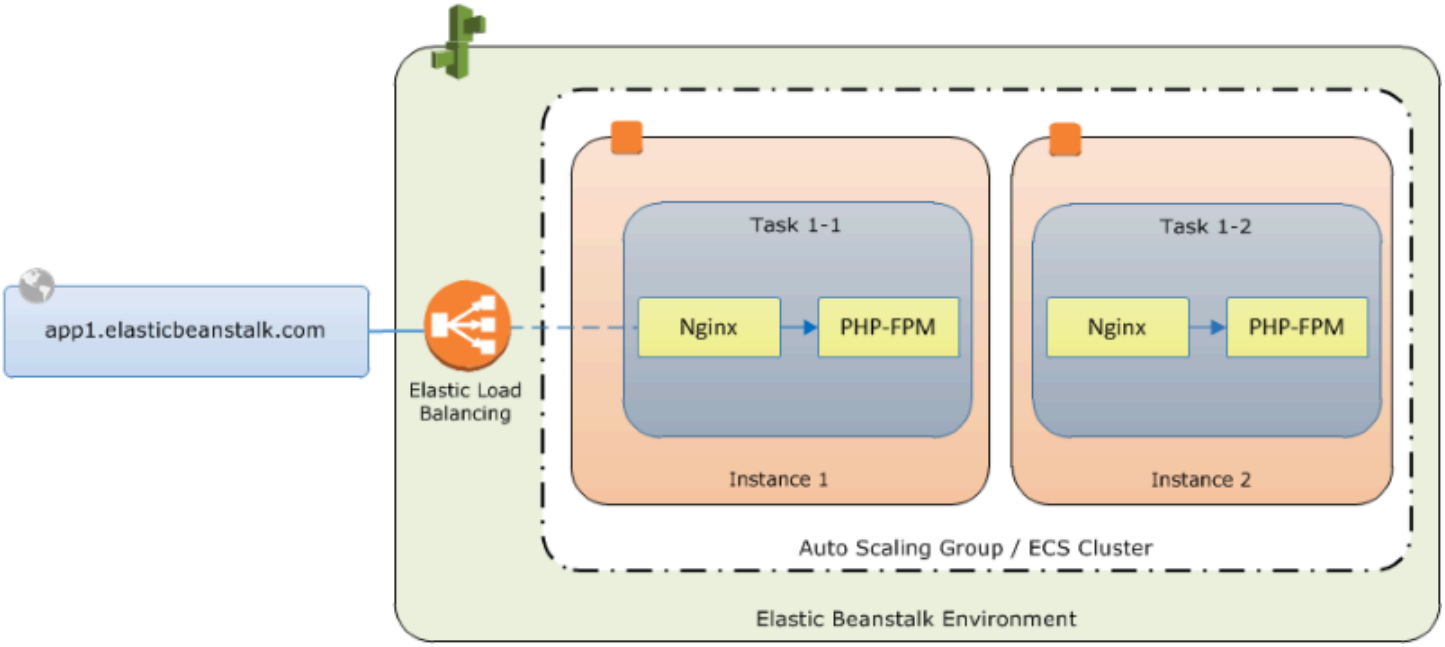
Buat file teks `Dockerrun.aws.json` v2 dengan nama ini di root aplikasi Anda dan tambahkan teks berikut:

```
{
  "AWSEBDockerrunVersion": 2,
  "volumes": [
    {
      "name": "php-app",
      "host": {
        "sourcePath": "/var/app/current/php-app"
      }
    },
    {
      "name": "nginx-proxy-conf",
      "host": {
        "sourcePath": "/var/app/current/proxy/conf.d"
      }
    }
  ],
  "containerDefinitions": [
    {
      "name": "php-app",
```

```
      "image": "php:fpm",
      "essential": true,
      "memory": 128,
      "mountPoints": [
        {
          "sourceVolume": "php-app",
          "containerPath": "/var/www/html",
          "readOnly": true
        }
      ]
    },
    {
      "name": "nginx-proxy",
      "image": "nginx",
      "essential": true,
      "memory": 128,
      "portMappings": [
        {
          "hostPort": 80,
          "containerPort": 80
        }
      ],
      "links": [
        "php-app"
      ],
      "mountPoints": [
        {
          "sourceVolume": "php-app",
          "containerPath": "/var/www/html",
          "readOnly": true
        },
        {
          "sourceVolume": "nginx-proxy-conf",
          "containerPath": "/etc/nginx/conf.d",
          "readOnly": true
        },
        {
          "sourceVolume": "awseb-logs-nginx-proxy",
          "containerPath": "/var/log/nginx"
        }
      ]
    }
  ]
```

```
}
```

Contoh konfigurasi ini mendefinisikan dua kontainer, sebuah situs web PHP dengan proxy nginx di depannya. Kedua kontainer akan berjalan berdampingan dalam kontainer Docker pada setiap contoh di lingkungan Elastic Beanstalk Anda, mengakses konten berbagi (isi situs web) dari volume pada contoh host, yang juga didefinisikan dalam file ini. Kontainer-kontainer itu sendiri dibuat dari gambar yang dihosting di repositori resmi di Docker Hub. Lingkungan yang dihasilkan terlihat seperti berikut ini:



Volume yang didefinisikan dalam konfigurasi sesuai dengan konten yang akan Anda buat berikutnya dan upload sebagai bagian dari bundel sumber aplikasi Anda. Kontainer mengakses konten pada host dengan memasang volume di bagian `mountPoints` dari definisi kontainer.

Untuk informasi lebih lanjut tentang format `Dockerrun.aws.json` v2 dan parameternya, lihatFormat definisi kontainer.

Tambahkan konten

Selanjutnya Anda akan menambahkan beberapa konten untuk situs PHP Anda untuk ditampilkan kepada pengunjung, dan file konfigurasi untuk proxy nginx.

php-app/index.php

```
<h1>Hello World!!!</h1>
```

```
<h3>PHP Version <pre><?= phpversion()?></pre></h3>
```

php-app/static.html

```
<h1>Hello World!</h1>
<h3>This is a static HTML page.</h3>
```

proxy/conf.d/default.conf

```
server {
  listen 80;
  server_name localhost;
  root /var/www/html;

  index index.php;

  location ~ [^/]\.php(/|$) {
    fastcgi_split_path_info ^(.+?\.php)(/.*)$;
    if (!-f $document_root$fastcgi_script_name) {
      return 404;
    }

    include fastcgi_params;
    fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
    fastcgi_param PATH_INFO $fastcgi_path_info;
    fastcgi_param PATH_TRANSLATED $document_root$fastcgi_path_info;

    fastcgi_pass php-app:9000;
    fastcgi_index index.php;
  }
}
```

Menerapkan ke Elastic Beanstalk

Folder aplikasi Anda sekarang berisi file-file berikut ini:

```
### Dockerrun.aws.json
### php-app
#   ### index.php
#   ### static.html
### proxy
    ### conf.d
```

```
        ### default.conf
```

Ini semua yang Anda butuhkan untuk menciptakan lingkungan Elastic Beanstalk. Buat arsip `.zip` dari file dan folder di atas (tidak termasuk folder proyek tingkat atas). Untuk membuat arsip di Windows explorer, pilih isi folder proyek, klik kanan, pilih Kirim Ke, dan kemudian klik Folder terkompresi (zip)

> **Note**
>
> Untuk informasi tentang struktur berkas yang diperlukan dan petunjuk untuk membuat arsip di lingkungan lain, lihat Buat bundel sumber aplikasi Elastic Beanstalk

Selanjutnya, upload bundel sumber ke Elastic Beanstalk dan buat lingkungan Anda. Untuk Platform, pilih Docker. Untuk cabang Platform, pilih ECS yang berjalan di 64bit Amazon Linux 2023.

Untuk meluncurkan lingkungan (konsol)

1. Buka konsol Elastic Beanstalk dengan tautan yang telah dikonfigurasi sebelumnya: console.aws.amazon. com/elasticbeanstalk/home#/newApplication?applicationName=Tutorial&EnvironmentType= LoadBalanced
2. Untuk Platform, pilih platform dan cabang platform yang sesuai dengan bahasa yang digunakan oleh aplikasi Anda, atau platform Docker untuk aplikasi berbasis kontainer.
3. Untuk Kode aplikasi, pilih Unggah kode Anda.
4. Pilih File lokal, pilih Pilih file, dan kemudian buka paket sumber.
5. Pilih Tinjau dan Luncurkan.
6. Tinjau pengaturan dan kemudian pilih Buat applikasi.

Konsol Elastic Beanstalk mengarahkan Anda kembali ke dasbor manajemen untuk lingkungan baru Anda. Layar ini menunjukkan status kesehatan lingkungan dan acara output oleh layanan Elastic Beanstalk. Ketika statusnya Hijau, klik URL disebelah nama lingkungan untuk melihat situs web baru Anda.

Hubungkan ke instans kontainer

Selanjutnya Anda akan terhubung ke EC2 instans Amazon di lingkungan Elastic Beanstalk Anda untuk melihat beberapa bagian yang bergerak beraksi.

Cara termudah untuk terhubung ke sebuah instance di lingkungan Anda adalah dengan menggunakan EB CLI. Untuk menggunakannya,instal EB CLI, jika Anda belum melakukannya. Anda juga harus mengonfigurasi lingkungan Anda dengan key pair Amazon EC2 SSH. Gunakan salah satu konsol halaman konfigurasi keamananatau EB CLI perintah eb init untuk melakukan itu. Untuk menyambung ke contoh lingkungan, gunakan EB CLI perintah eb ssh.

Sekarang setelah Anda terhubung ke EC2 instans Amazon yang menghosting container docker Anda, Anda dapat melihat bagaimana segala sesuatunya diatur. Jalankan `ls` pada`/var/app/current`:

```
[ec2-user@ip-10-0-0-117 ~]$ ls /var/app/current
Dockerrun.aws.json  php-app  proxy
```

Direktori ini berisi berkas dari bundel sumber yang Anda upload ke Elastic Beanstalk selama pembuatan lingkungan.

```
[ec2-user@ip-10-0-0-117 ~]$ ls /var/log/containers
nginx-proxy    nginx-proxy-4ba868dbb7f3-stdouterr.log
php-app        php-app-dcc3b3c8522c-stdouterr.log       rotated
```

Di sinilah log dibuat pada contoh kontainer dan dikumpulkan oleh Elastic Beanstalk. Elastic Beanstalk membuat volume di direktori ini untuk setiap kontainer, yang Anda pasang ke lokasi kontainer tempat log ditulis.

Anda juga dapat melihat Docker untuk melihat kontainer berjalan dengan `docker ps`.

```
[ec2-user@ip-10-0-0-117 ~]$ sudo docker ps
CONTAINER ID   IMAGE                              COMMAND                  CREATED
  STATUS                  PORTS                                  NAMES

4ba868dbb7f3   nginx                              "/docker-entrypoint.…"   4 minutes ago
   Up 4 minutes            0.0.0.0:80->80/tcp, :::80->80/tcp   ecs-awseb-Tutorials-env-
dc2aywfjwg-1-nginx-proxy-acca84ef87c4aca15400
dcc3b3c8522c   php:fpm                            "docker-php-entrypoi…"   4 minutes ago
   Up 4 minutes            9000/tcp                               ecs-awseb-Tutorials-env-
dc2aywfjwg-1-php-app-b8d38ae288b7b09e8101
d9367c0baad6   amazon/amazon-ecs-agent:latest   "/agent"                 5 minutes ago
  Up 5 minutes (healthy)                                    ecs-agent
```

Ini menunjukkan dua kontainer berjalan yang Anda gunakan, serta agen kontainer Amazon ECS yang mengkoordinasikan deployment.

Periksa agen kontainer Amazon ECS

EC2 Instans Amazon di lingkungan Docker yang dikelola ECS di Elastic Beanstalk menjalankan proses agen dalam wadah Docker. Agen ini menghubungkan ke layanan Amazon ECS untuk mengkoordinasikan penyebaran kontainer. Penyebaran ini berjalan sebagai tugas di Amazon ECS, yang dikonfigurasi dalam file definisi tugas. Elastic Beanstalk menciptakan file definisi tugas ini berdasarkan `Dockerrun.aws.json`yang Anda upload dalam bundel sumber.

Periksa status agen kontainer dengan HTTP get request to `http://localhost:51678/v1/metadata`:

```
[ec2-user@ip-10-0-0-117 ~]$ curl http://localhost:51678/v1/metadata
{
  "Cluster":"awseb-Tutorials-env-dc2aywfjwg",
  "ContainerInstanceArn":"arn:aws:ecs:us-west-2:123456789012:container-instance/awseb-
Tutorials-env-dc2aywfjwg/db7be5215cd74658aacfcb292a6b944f",
  "Version":"Amazon ECS Agent - v1.57.1 (089b7b64)"
}
```

Struktur ini menunjukkan nama cluster Amazon ECS, dan ARN (Amazon Resource Name) dari instance cluster (instans EC2 Amazon yang Anda sambungkan).

Untuk informasi lebih lanjut, buat permintaan HTTP get kehttp://localhost:51678/v1/tasks:

```
[ec2-user@ip-10-0-0-117 ~]$ curl http://localhost:51678/v1/tasks
{
   "Tasks":[
      {
         "Arn":"arn:aws:ecs:us-west-2:123456789012:task/awseb-Tutorials-env-dc2aywfjwg/
bbde7ebe1d4e4537ab1336340150a6d6",
         "DesiredStatus":"RUNNING",
         "KnownStatus":"RUNNING",
         "Family":"awseb-Tutorials-env-dc2aywfjwg",
         "Version":"1",
         "Containers":[
            {

 "DockerId":"dcc3b3c8522cb9510b7359689163814c0f1453b36b237204a3fd7a0b445d2ea6",
               "DockerName":"ecs-awseb-Tutorials-env-dc2aywfjwg-1-php-app-
b8d38ae288b7b09e8101",
               "Name":"php-app",
               "Volumes":[
```

```
                  {
                     "Source":"/var/app/current/php-app",
                     "Destination":"/var/www/html"
                  }
               ]
            },
            {

 "DockerId":"4ba868dbb7f3fb3328b8afeb2cb6cf03e3cb1cdd5b109e470f767d50b2c3e303",
               "DockerName":"ecs-awseb-Tutorials-env-dc2aywfjwg-1-nginx-proxy-
acca84ef87c4aca15400",
               "Name":"nginx-proxy",
               "Ports":[
                  {
                     "ContainerPort":80,
                     "Protocol":"tcp",
                     "HostPort":80
                  },
                  {
                     "ContainerPort":80,
                     "Protocol":"tcp",
                     "HostPort":80
                  }
               ],
               "Volumes":[
                  {
                     "Source":"/var/app/current/php-app",
                     "Destination":"/var/www/html"
                  },
                  {
                     "Source":"/var/log/containers/nginx-proxy",
                     "Destination":"/var/log/nginx"
                  },
                  {
                     "Source":"/var/app/current/proxy/conf.d",
                     "Destination":"/etc/nginx/conf.d"
                  }
               ]
            }
         ]
      }
   ]
}
```

Struktur ini menjelaskan tugas yang dijalankan untuk menerapkan dua docker kontainer dari proyek contoh tutorial ini. Informasi berikut ini ditampilkan secara default.

- KnownStatus— `RUNNING` Status menunjukkan bahwa kontainer masih aktif.
- Keluarga— Nama definisi tugas yang Elastic Beanstalk dibuat dari`Dockerrun.aws.json`.
- Versi— Versi definisi tugas. Ini bertambah setiap kali file definisi tugas diperbarui.
- Kontainer— Informasi tentang kontainer yang berjalan pada contoh.

Informasi lebih lanjut tersedia dari layanan Amazon ECS itu sendiri, yang dapat Anda hubungi menggunakan AWS Command Line Interface. Untuk petunjuk tentang penggunaan AWS CLI dengan Amazon ECS, dan informasi tentang Amazon ECS secara umum, lihat Panduan Pengguna Amazon ECS.

Migrasi aplikasi Elastic Beanstalk Anda dari Docker Multi-container yang dikelola ECS ke ECS di Amazon Linux 2023 AL1

> **Note**
>
> Pada tanggal 18 Juli 2022, Elastic Beanstalk menetapkan status semua cabang platform berdasarkan Amazon Linux AMI () untuk pensiun. AL1 .

Topik ini memandu Anda dalam migrasi aplikasi Anda dari cabang platform pensiunan Multi-container Docker yang berjalan di 64bit Amazon Linux ke ECS Running pada 64bit 023. AL2 Cabang platform target ini saat ini dan didukung. Seperti cabang Multi-container Docker sebelumnya, AL1 cabang AL2platform ECS 023 yang lebih baru menggunakan Amazon ECS untuk mengoordinasikan penyebaran beberapa kontainer Docker ke cluster Amazon ECS di lingkungan Elastic Beanstalk. Cabang platform ECS AL2 023 yang baru mendukung semua fitur di cabang platform Multi-container AL1 Docker sebelumnya. Juga, file `Dockerrun.aws.json` v2 yang sama didukung.

Bagian-bagian

- Bermigrasi dengan konsol Elastic Beanstalk
- Migrasi dengan AWS CLI

Bermigrasi dengan konsol Elastic Beanstalk

Untuk bermigrasi menggunakan konsol Elastic Beanstalk, gunakan kode sumber yang sama ke lingkungan baru yang didasarkan pada cabang platform ECS Running on 023. AL2 Tidak ada perubahan pada kode sumber yang diperlukan.

Untuk bermigrasi ke cabang platform ECS Running di Amazon Linux 2023

1. Menggunakan sumber aplikasi yang sudah diterapkan ke lingkungan lama, buat bundel sumber aplikasi. Anda dapat menggunakan bundel sumber aplikasi yang sama dan file `Dockerrun.aws.json` v2 yang sama.
2. Buat lingkungan baru menggunakan cabang platform ECS Running di Amazon Linux 2023. Gunakan bundel sumber dari langkah sebelumnya untuk kode Aplikasi. Untuk langkah-langkah yang lebih rinci, lihat Menerapkan ke Elastic Beanstalk di tutorial ECS managed Docker sebelumnya di chapter ini.

Migrasi dengan AWS CLI

Anda juga memiliki opsi untuk menggunakan AWS Command Line Interface (AWS CLI) untuk memigrasikan lingkungan Docker Amazon Linux Docker Multi-container yang ada ke cabang platform ECS 023 yang lebih baru AL2. Dalam hal ini Anda tidak perlu membuat lingkungan baru atau menerapkan kembali kode sumber Anda. Anda hanya perlu menjalankan perintah AWS CLI update-environment. Ini akan melakukan pembaruan platform untuk memigrasikan lingkungan Anda yang ada ke cabang platform ECS Amazon Linux 2023.

Gunakan sintaks berikut untuk memigrasikan lingkungan Anda ke cabang platform baru.

```
aws elasticbeanstalk update-environment \
--environment-name my-env \
--solution-stack-name "64bit Amazon Linux 2023 version running ECS" \
--region my-region
```

Berikut ini adalah contoh perintah untuk memigrasikan lingkungan beta-101 ke versi 3.0.0 dari cabang platform ECS Amazon Linux 2023 di wilayah us-east-1.

```
aws elasticbeanstalk update-environment \
--environment-name beta-101 \
--solution-stack-name "64bit Amazon Linux 2023 v4.0.0 running ECS" \
```

```
--region us-east-1
```

`solution-stack-name`Parameter menyediakan cabang platform dan versinya. Gunakan versi cabang platform terbaru dengan menentukan nama tumpukan solusi yang tepat. Versi setiap cabang platform disertakan dalam nama stack solusi, seperti yang ditunjukkan pada contoh di atas. Untuk daftar tumpukan solusi terbaru untuk platform Docker, lihat Platform yang didukung di panduan AWS Elastic Beanstalk Platform.

> **Note**
>
> list-available-solution-stacksPerintah ini menyediakan daftar versi platform yang tersedia untuk akun Anda di AWS Wilayah.
>
> ```
> aws elasticbeanstalk list-available-solution-stacks --region us-east-1 --query
>  SolutionStacks
> ```

Untuk mempelajari selengkapnya AWS CLI, lihat Panduan AWS Command Line Interface Pengguna. Untuk informasi lebih lanjut tentang AWS CLI perintah untuk Elastic Beanstalk, AWS CLI lihat Referensi Perintah untuk Elastic Beanstalk.

Menggunakan gambar dari repositori pribadi di Elastic Beanstalk

Topik ini menjelaskan cara mengautentikasi ke repositori gambar online pribadi dengan Elastic Beanstalk. Elastic Beanstalk harus mengautentikasi dengan registri online sebelum dapat menarik dan menyebarkan gambar Anda. Ada beberapa opsi konfigurasi.

Menggunakan gambar dari repositori Amazon ECR

Anda dapat menyimpan gambar Docker kustom Anda AWS dengan Amazon Elastic Container Registry (Amazon ECR).

Saat Anda menyimpan gambar Docker di Amazon ECR, Elastic Beanstalk secara otomatis mengautentikasi registri Amazon ECR dengan profil instans lingkungan Anda. Oleh karena itu, Anda harus memberikan izin kepada instans Anda untuk mengakses gambar di repositori Amazon ECR Anda. Untuk melakukannya, tambahkan izin ke profil instans lingkungan Anda dengan melampirkan kebijakan EC2 ContainerRegistryReadOnly terkelola Amazon ke profil instance. Ini menyediakan akses hanya-baca ke semua repositori Amazon ECR di akun Anda. Anda juga memiliki opsi untuk

hanya mengakses repositori tunggal dengan menggunakan templat berikut untuk membuat kebijakan khusus:

```
{
    "Version": "2012-10-17",
    "Statement": [
      {
        "Sid": "AllowEbAuth",
        "Effect": "Allow",
        "Action": [
          "ecr:GetAuthorizationToken"
        ],
        "Resource": [
          "*"
        ]
      },
      {
        "Sid": "AllowPull",
        "Effect": "Allow",
        "Resource": [
          "arn:aws:ecr:us-east-2:account-id:repository/repository-name"
        ],
        "Action": [
          "ecr:GetAuthorizationToken",
          "ecr:BatchCheckLayerAvailability",
          "ecr:GetDownloadUrlForLayer",
          "ecr:GetRepositoryPolicy",
          "ecr:DescribeRepositories",
          "ecr:ListImages",
          "ecr:BatchGetImage"
        ]
      }
    ]
  }
```

Ganti Amazon Resource Name (ARN) di kebijakan di atas dengan ARN repositori Anda.

Anda harus menentukan informasi gambar dalam `Dockerrun.aws.json` file Anda. Konfigurasi akan berbeda tergantung pada platform mana yang Anda gunakan.

Untuk platform Docker yang dikelola ECS, gunakan `image` kunci dalam objek definisi kontainer:

```
"containerDefinitions": [
```

```
        {
        "name": "my-image",
        "image": "account-id.dkr.ecr.us-east-2.amazonaws.com/repository-name:latest",
```

Untuk platform Docker lihat gambar dengan URL. URL masuk dalam `Image` definisi `Dockerrun.aws.json` file Anda:

```
"Image": {
    "Name": "account-id.dkr.ecr.us-east-2.amazonaws.com/repository-name:latest",
    "Update": "true"
  },
```

Menggunakan Parameter Store AWS Systems Manager (SSM)

Anda dapat mengkonfigurasi Elastic Beanstalk untuk log in ke repositori pribadi Anda sebelum memulai proses deployment. Hal ini memungkinkan Elastic Beanstalk untuk mengakses gambar dari repositori dan menyebarkan gambar-gambar ini ke lingkungan Elastic Beanstalk Anda.

Konfigurasi ini memulai peristiwa di fase prebuild dari proses deployment Elastic Beanstalk. Anda mengatur ini di.ebextentionsDirektori Konfigurasi. Konfigurasi ini menggunakan skrip hook platform yang memanggil docker login untuk mengotentikasi ke registri online yang host repositori privat. Skrip hook platform dengan aman membaca data kredensyal dari variabel lingkungan Elastic Beanstalk yang diinisialisasi oleh konfigurasi yang sumber nilai dari Parameter Store. AWS Systems Manager Platform Elastic Beanstalk Docker dan ECS Docker yang dikelola ECS Anda harus merupakan versi yang dirilis pada atau setelah 26 Maret 2025 untuk mendukung konfigurasi variabel lingkungan ini. laporan rinci dari langkah-langkah konfigurasi berikut ini.

Untuk mengonfigurasi Elastic Beanstalk untuk mengautentikasi ke repositori pribadi Anda dengan Parameter Store AWS Systems Manager

> **Note**
>
> Anda perlu mengatur kredensyal Anda di AWS Systems Manager Parameter Store dan juga menyiapkan izin IAM yang diperlukan untuk menyelesaikan langkah-langkah ini. Untuk informasi selengkapnya, lihat Prasyarat untuk mengonfigurasi rahasia sebagai variabel lingkungan.

1. Membuat `.ebextensions`struktur direktori Anda sebagai berikut.

```
### .ebextensions
#   ### env.config
### .platform
#   ### confighooks
#   #   ### prebuild
#   #       ### 01login.sh
#   ### hooks
#       ### prebuild
#           ### 01login.sh
### docker-compose.yml
```

2. Gunakan AWS Systems ManagerParameter Store untuk menyimpan kredensyal repositori pribadi Anda. Jalankan put-parameter perintah AWS CLI Systems Manager berikut untuk membuat ini di Parameter Store.

```
aws ssm put-parameter --name USER --type String --value "username"
aws ssm put-parameter --name PASSWD --type String --value "passwd"
```

3. Membuat file `env.config` berikut dan menempatkannya di direktori `.ebextensions` seperti yang ditunjukkan dalam struktur direktori sebelumnya. Konfigurasi ini menggunakan aws:elasticbeanstalk:aplikasi:environmentsecrets namespace untuk menginisialisasi variabel lingkungan `PASSWD` Elastic `USER` Beanstalk dan ke nilai-nilai yang disimpan di Systems Manager Parameter Store.

> **Note**
>
> USERdanPASSWDdalam script harus sesuai dengan string yang sama yang digunakan dalam sebelumnyassm put-parameterPerintah.

```
option_settings:
  aws:elasticbeanstalk:application:environmentsecrets:
    USER: arn:aws:ssm:us-east-1:111122223333:parameter/user
    PASSWD: arn:aws:secretsmanager:us-east-1:111122223333:passwd
```

4. Membuat file script `01login.sh` berikut dan menempatkan di direktori berikut (juga ditampilkan dalam struktur direktori sebelumnya):

- `.platform/confighooks/prebuild`
- `.platform/hooks/prebuild`

```
### example 01login.sh
#!/bin/bash
echo $PASSWD | docker login -u $USER --password-stdin
```

`01login.sh`Referensi variabel lingkungan Elastic Beanstalk yang Anda konfigurasikan di Langkah 3 untuk menyimpan kredensyal repositori, dan menyalurkan kata sandi langsung ke perintah di aliran input. docker login `stdin` `--password-stdin`Opsi ini menggunakan aliran input, mencegah kata sandi bertahan dalam riwayat shell atau log audit. Untuk informasi selengkapnya tentang otentikasi dengan antarmuka baris perintah Docker, lihat docker login di situs web dokumentasi Docker.

> **Catatan**
>
> - Semua file script harus memiliki izin eksekusi. Gunakanchmod +xuntuk mengatur izin eksekusi pada file hook Anda. Untuk semua versi platform berbasis Amazon Linux 2 yang dirilis pada atau setelah 29 April 2022, Elastic Beanstalk secara otomatis memberikan izin eksekusi ke semua skrip kait platform. Dalam hal ini Anda tidak perlu memberikan izin eksekusi secara manual. Untuk daftar versi platform ini, lihat catatan rilis platform Linux 29 April 2022 di Panduan Catatan AWS Elastic Beanstalk Rilis.
> - File Hook dapat berupa file biner atau file skrip yang dimulai dengan garis #! yang berisi jalur penerjemah mereka, seperti #!/bin/bash.
> - Untuk informasi lebih lanjut, lihat the section called “Hook platform” pada Memperluas platform Elastic BeanstalkLinux .

Setelah Elastic Beanstalk dapat mengotentikasi dengan registri online yang host repositori pribadi, gambar Anda dapat digunakan dan ditarik.

Menggunakan file **`Dockerrun.aws.json`**

Bagian ini menjelaskan pendekatan lain untuk mengotentikasi Elastic Beanstalk ke repositori pribadi. Dengan pendekatan ini, Anda menghasilkan file otentikasi dengan perintah Docker, dan kemudian

mengunggah file otentikasi ke ember Amazon S3. Anda juga harus menyertakan informasi bucket di file `Dockerrun.aws.json` Anda.

Untuk menghasilkan dan menyediakan file otentikasi ke Elastic Beanstalk

1. Menghasilkan file autentikasi dengan perintah docker login. Untuk repositori di Docker Hub, jalankandocker login:

   ```
   $ docker login
   ```

   Untuk pendaftar lainnya, sertakan URL server registri:

   ```
   $ docker login registry-server-url
   ```

   > **Note**
   >
   > Jika lingkungan Elastic Beanstalk Anda menggunakan versi platform Amazon Linux AMI Docker (sebelum Amazon Linux 2), baca informasi yang sesuai di the section called "Konfigurasi docker di Amazon Linux AMI (sebelumnya Amazon Linux 2)".

   Untuk informasi selengkapnya tentang file autentikasi, lihat Simpan gambar di Docker Hub dan docker login di situs web Docker.

2. Unggah salinan file autentikasi yang bernama `.dockercfg` ke bucket aman Amazon S3 .

   - Bucket Amazon S3 harus di-host Wilayah AWS sama dengan lingkungan yang menggunakannya. Elastic Beanstalk tidak dapat men-download file dari bucket Amazon S3 yang di-host di Kawasan lain.
   - Memberikan izin untuk operasi `s3:GetObject` untuk IAM role di profil instans. Untuk informasi selengkapnya, lihat Mengelola profil instans Elastic Beanstalk.

3. Sertakan informasi bucket Amazon S3 di parameter `Authentication` pada file `Dockerrun.aws.json`.

   Contoh berikut menunjukkan penggunaan file otentikasi bernama `mydockercfg` dalam sebuah bucket bernama `amzn-s3-demo-bucket` untuk menggunakan gambar privat di registri pihak ketiga. Untuk nomor versi yang benar`AWSEBDockerrunVersion`, lihat catatan yang mengikuti contoh.

```
{
  "AWSEBDockerrunVersion": "version-no",
  "Authentication": {
    "Bucket": "amzn-s3-demo-bucket",
    "Key": "mydockercfg"
  },
  "Image": {
    "Name": "quay.io/johndoe/private-image",
    "Update": "true"
  },
  "Ports": [
    {
      "ContainerPort": "1234"
    }
  ],
  "Volumes": [
    {
      "HostDirectory": "/var/app/mydb",
      "ContainerDirectory": "/etc/mysql"
    }
  ],
  "Logging": "/var/log/nginx"
}
```

Versi **`Dockerrun.aws.json`**

Parameter `AWSEBDockerrunVersion` menunjukkan versi file `Dockerrun.aws.json`.

- Platform Docker AL2 dan AL2 023 menggunakan versi file berikut.
  - `Dockerrun.aws.json v3`— lingkungan yang menggunakan Docker Compose.
  - `Dockerrun.aws.json v1`— lingkungan yang tidak menggunakan Docker Compose.
- ECS yang berjalan di Amazon Linux 2 dan ECS yang berjalan di AL2 023 menggunakan file tersebut. `Dockerrun.aws.json v2` Platform pensiunan ECS-Multicontainer Docker Amazon Linux AMI AL1 () juga menggunakan versi yang sama ini.

Setelah Elastic Beanstalk dapat mengotentikasi dengan registri online yang host repositori pribadi, gambar Anda dapat digunakan dan ditarik.

Mengkonfigurasi lingkungan Elastic Beanstalk Docker

Bab ini menjelaskan informasi konfigurasi tambahan untuk semua cabang platform Docker yang didukung, termasuk cabang platform Docker yang dikelola ECS. Kecuali cabang platform tertentu atau komponen cabang platform diidentifikasi dalam suatu bagian, itu berlaku untuk semua lingkungan yang menjalankan platform Docker dan ECS yang didukung Docker dan ECS.

> **Note**
>
> Jika lingkungan Node.js Elastic Beanstalk Anda menggunakan versi platform Amazon Linux AMI (Amazon Linux 2 yang terdahulu), baca informasi di the section called “Konfigurasi docker di Amazon Linux AMI (sebelumnya Amazon Linux 2)”.

Bagian

- Mengkonfigurasi perangkat lunak di lingkungan Docker
- Referensi variabel lingkungan dalam kontainer
- Menggunakan fitur interpolasi untuk variabel lingkungan dengan Docker Compose
- Membuat log untuk pelaporan kesehatan yang ditingkatkan dengan Docker Compose
- Pencatatan khusus wadah Docker dengan Docker Compose
- Gambar Docker
- Mengkonfigurasi pembaruan terkelola untuk lingkungan Docker
- Ruang nama konfigurasi Docker
- Konfigurasi docker di Amazon Linux AMI (sebelumnya Amazon Linux 2)

Mengkonfigurasi perangkat lunak di lingkungan Docker

Anda dapat menggunakan konsol Elastic Beanstalk untuk mengkonfigurasi perangkat lunak yang berjalan pada contoh lingkungan Anda.

Untuk mengonfigurasi lingkungan Docker Anda di konsol Elastic Beanstalk

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Dalam kategori konfigurasi Pembaruan, pemantauan, dan pencatatan, pilih Edit.
5. Membuat perubahan konfigurasi yang diperlukan.
6. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Untuk informasi tentang mengkonfigurasi pengaturan perangkat lunak di lingkungan apa pun, lihatthe section called “Variabel lingkungan dan pengaturan perangkat lunak”. Bagian berikut mencakup informasi spesifik Docker.

Opsi kontainer

Opsi kontainerbagian memiliki pilihan platform-spesifik. Untuk lingkungan Docker, ini memungkinkan Anda memilih apakah lingkungan Anda menyertakan server proxy NGINX atau tidak.

Lingkungan dengan Docker Compose

Jika Anda mengelola lingkungan Docker dengan Docker Compose, Elastic Beanstalk mengasumsikan bahwa Anda menjalankan server proxy sebagai kontainer. Oleh karena itu defaultnya adalahTidak adauntukServer proksipengaturan, dan Elastic Beanstalk tidak menyediakan konfigurasi NGINX.

> **Note**
>
> Bahkan jika Anda memilihNGINXsebagai server proxy, pengaturan ini diabaikan dalam lingkungan dengan Docker Compose. Pengaturan Server proksi masih default pada Tidak ada.

Karena proxy server web NGINX dinonaktifkan untuk platform Docker di Amazon Linux 2 dengan Docker Compose, Anda harus mengikuti petunjuk untuk menghasilkan log untuk pelaporan kondisi yang ditingkatkan. Untuk informasi selengkapnya, lihat Membuat log untuk pelaporan kesehatan yang ditingkatkan dengan Docker Compose.

Properti lingkungan (variabel lingkungan)

Anda dapat menggunakan properti lingkungan, (juga dikenal sebagai variabel lingkungan), untuk meneruskan nilai, titik akhir seperti, pengaturan debug, dan informasi lainnya ke aplikasi Anda. Bagian Environment variable pada konsol memungkinkan Anda menentukan variabel lingkungan pada EC2 instance yang menjalankan aplikasi Anda. Variabel lingkungan diteruskan sebagai pasangan kunci-nilai ke aplikasi.

Kode aplikasi Anda yang berjalan dalam kontainer dapat merujuk ke variabel lingkungan dengan nama dan membaca nilainya. Kode sumber yang membaca variabel lingkungan ini akan bervariasi menurut bahasa pemrograman. Anda dapat menemukan instruksi untuk membaca nilai variabel lingkungan dalam bahasa pemrograman yang didukung platform terkelola Elastic Beanstalk di topik platform masing-masing. Untuk daftar tautan ke topik ini, lihatthe section called “Variabel lingkungan dan pengaturan perangkat lunak”.

Rahasia dan parameter dalam variabel lingkungan Elastic Beanstalk

Elastic Beanstalk menawarkan kemampuan AWS Secrets Manager untuk AWS Systems Manager referensi dan Parameter Menyimpan data dalam variabel lingkungan. Ini adalah opsi aman untuk aplikasi Anda untuk mengakses rahasia dan parameter asli yang disimpan oleh layanan ini tanpa harus mengelola panggilan API ke mereka. Platform Elastic Beanstalk Docker dan ECS Docker yang dikelola ECS Anda harus merupakan versi yang dirilis pada atau setelah 26 Maret 2025 untuk mendukung fitur ini. Untuk informasi selengkapnya tentang penggunaan variabel lingkungan untuk mereferensikan rahasia, lihatMengambil rahasia dan parameter ke variabel lingkungan Elastic Beanstalk.

Lingkungan dengan Docker Compose

Jika Anda mengelola lingkungan Docker Anda dengan Docker Compose, Anda harus membuat beberapa konfigurasi tambahan untuk mengambil variabel lingkungan dalam kontainer. Agar executable yang berjalan di container Anda untuk mengakses variabel lingkungan ini, Anda harus mereferensikannya di file. `docker-compose.yml` Untuk mengetahui informasi selengkapnya, lihat Referensi variabel lingkungan dalam kontainer.

Referensi variabel lingkungan dalam kontainer

Jika Anda menggunakan alat Docker Compose pada platform Docker Amazon Linux 2, Elastic Beanstalk menghasilkan file lingkungan Docker Compose bernama `.env` di direktori root proyek aplikasi Anda. File ini menyimpan variabel lingkungan Anda dikonfigurasi untuk Elastic Beanstalk.

> **Note**
>
> Jika Anda menyertakan file `.env` dalam bundel aplikasi Anda, Elastic Beanstalk tidak akan menghasilkan file `.env`.

Agar kontainer untuk referensi variabel lingkungan yang Anda tentukan di Elastic Beanstalk, Anda harus mengikuti salah satu atau kedua pendekatan konfigurasi ini.

- Tambahkan file `.env` yang dihasilkan oleh Elastic Beanstalk ke opsi konfigurasi `env_file` di file `docker-compose.yml`.
- Langsung menentukan variabel lingkungan di file `docker-compose.yml`.

Berikut ini adalah contoh file . Sampel file `docker-compose.yml` menunjukkan kedua pendekatan.

- Jika anda menentukan properti lingkungan `DEBUG_LEVEL=1` dan `LOG_LEVEL=error`, Elastic Beanstalk menghasilkan file `.env` berikut untuk Anda:

```
DEBUG_LEVEL=1
LOG_LEVEL=error
```

- Dalam file `docker-compose.yml`, opsi konfigurasi `env_file` menunjuk ke file `.env`, dan itu juga mendefinisikan variabel lingkungan `DEBUG=1` secara langsung di file `docker-compose.yml`.

```
services:
  web:
    build: .
    environment:
      - DEBUG=1
    env_file:
      - .env
```

> **Catatan**
>
> - Jika Anda menetapkan variabel lingkungan yang sama di kedua file, variabel didefinisikan dalam file `docker-compose.yml` memiliki keutamaan lebih tinggi dari variabel yang didefinisikan dalam file `.env`.

- Hati-hati untuk tidak meninggalkan spasi antara tanda sama (=) dan nilai yang diberikan ke variabel Anda untuk mencegah spasi ditambahkan ke string.

Untuk mempelajari lebih lanjut tentang variabel lingkungan di Docker Compose, lihatVariabel lingkungan dalam Compose

Menggunakan fitur interpolasi untuk variabel lingkungan dengan Docker Compose

Dimulai dengan rilis platform 28 Juli 2023, cabang platform Docker Amazon Linux 2 menawarkan fitur interpolasi Docker Compose. Dengan fitur ini, nilai dalam file Compose dapat disetel oleh variabel dan diinterpolasi saat runtime. Untuk informasi selengkapnya tentang fitur ini, lihat Interpolasi di situs web dokumentasi Docker.

Important

Jika Anda ingin menggunakan fitur ini dengan aplikasi Anda, ketahuilah bahwa Anda harus menerapkan pendekatan yang menggunakan kait platform.
Ini diperlukan karena mitigasi yang kami terapkan di mesin platform. Mitigasi ini memastikan kompatibilitas mundur untuk pelanggan yang tidak mengetahui fitur interpolasi baru dan memiliki aplikasi yang ada yang menggunakan variabel lingkungan dengan karakter tersebut. $ Mesin platform yang diperbarui lolos dari interpolasi secara default dengan mengganti karakter dengan $ karakter. $$

Berikut ini adalah contoh skrip hook platform yang dapat Anda atur untuk memungkinkan penggunaan fitur interpolasi.

```
#!/bin/bash

: '
example data format in .env file
key1=value1
key2=value2
'
envfile="/var/app/staging/.env"
tempfile=$(mktemp)
```

```
while IFS= read -r line; do
  # split each env var string at '='
  split_str=(${line//=/ })
  if [ ${#split_str[@]} -eq 2 ]; then
    # replace '$$' with '$'
    replaced_str=${split_str[1]//\$\$/\$}
    # update the value of env var using ${replaced_str}
    line="${split_str[0]}=${replaced_str}"
  fi
  # append the updated env var to the tempfile
  echo "${line}" #"${tempfile}"
done < "${envfile}"
# replace the original .env file with the tempfile
mv "${tempfile}" "${envfile}"
```

Tempatkan kait platform di bawah kedua direktori ini:

- `.platform/confighooks/predeploy/`
- `.platform/hooks/predeploy/`

Untuk informasi selengkapnya, lihat Hook platform di topik Memperluas platform Linux dari panduan ini.

Membuat log untuk pelaporan kesehatan yang ditingkatkan dengan Docker Compose

Agen health Elastic Beanstalk menyediakan sistem operasi dan metrik kesehatan aplikasi untuk lingkungan Elastic Beanstalk. Hal ini bergantung pada web server format log yang relay informasi dalam format tertentu.

Elastic Beanstalk mengasumsikan bahwa Anda menjalankan proxy server web sebagai kontainer. Akibatnya, proxy server web NGINX dinonaktifkan untuk lingkungan Docker menjalankan Docker Compose. Anda harus mengkonfigurasi server Anda untuk menulis log di lokasi dan format yang menggunakan agen kesehatan Elastic Beanstalk. Dengan demikian, Anda dapat memanfaatkan sepenuhnya pelaporan kesehatan yang disempurnakan, meskipun proxy server web dinonaktifkan.

Untuk petunjuk tentang cara melakukannya, lihat Konfigurasi log server web

Pencatatan khusus wadah Docker dengan Docker Compose

Agar dapat memecahkan masalah dan memantau layanan kontainer secara efisien, Anda dapat meminta instans log dari Elastic Beanstalk melalui konsol manajemen lingkungan atau EB CLI. Log instans terdiri dari log bundel dan log ekor, digabungkan dan dikemas untuk memungkinkan Anda melihat log dan peristiwa terbaru dengan cara yang efisien dan mudah.

Elastic Beanstalk menciptakan direktori log pada contoh kontainer, satu untuk setiap layanan didefinisikan dalam file `docker-compose.yml`, di `/var/log/eb-docker/containers/<service name>`. Jika Anda menggunakan fitur Docker Compose pada platform Docker Amazon Linux 2, Anda dapat memasang direktori ini ke lokasi dalam struktur file penampung tempat log ditulis. Ketika Anda memasang direktori log untuk menulis data log, Elastic Beanstalk dapat mengumpulkan data log dari direktori ini.

Jika aplikasi Anda berada di platform Docker yang tidak menggunakan Docker Compose, Anda dapat mengikuti prosedur standar yang dijelaskan di Pencatatan khusus wadah Docker dengan Docker Compose.

Untuk mengonfigurasi file log layanan Anda menjadi file ekor dan log bundel yang dapat diambil

1. Mengedit file `docker-compose.yml`.
2. Di bawah kunci `volumes` untuk layanan Anda, tambahkan bind mount menjadi seperti ini:

   `"${EB_LOG_BASE_DIR}/<service name>:<log directory inside container>`

   Dalam sampel file `docker-compose.yml` di bawah ini:

   - `nginx-proxy`adalah *`<service name>`*
   - `/var/log/nginx`adalah *`<log directory inside container>`*

```
services:
  nginx-proxy:
    image: "nginx"
    volumes:
      - "${EB_LOG_BASE_DIR}/nginx-proxy:/var/log/nginx"
```

- Direktori `var/log/nginx` berisi log untuk layanan proksi-nginx dalam kontainer, dan akan dipetakan ke direktori `/var/log/eb-docker/containers/nginx-proxy` pada host.
- Semua log di direktori ini sekarang dapat diambil sebagai log bundel dan log ekor melalui Elastic Beanstalk'sPermintaan log instancefungsional.

> **Catatan**
>
> - $ {EB_LOG_BASE_DIR}adalah variabel lingkungan yang ditetapkan oleh Elastic Beanstalk dengan nilai`/var/log/eb-docker/containers`.
> - Elastic Beanstalk secara otomatis menciptakan direktori `/var/log/eb-docker/containers/<service name>` untuk setiap layanan di file `docker-compose.yml`.

Gambar Docker

Cabang platform Docker yang dikelola Docker dan ECS untuk Elastic Beanstalk mendukung penggunaan gambar Docker yang disimpan dalam repositori gambar online publik atau pribadi.

Tentukan gambar berdasarkan nama di`Dockerrun.aws.json`. Perhatikan konvensi ini:

- Gambar di repositori resmi di Docker Hub menggunakan satu nama (misalnya,`ubuntu`ataumongo).
- Gambar di repositori lain di Docker Hub memenuhi syarat dengan nama organisasi (misalnya,`amazon/amazon-ecs-agent`).
- Gambar di repositori online lainnya memenuhi syarat lebih lanjut dengan nama domain (misalnya,`quay.io/assemblyline/ubuntu`atau*`account-id`*`.dkr.ecr.us-east-2.amazonaws.com/ubuntu:trusty`).

Untuk lingkungan yang menggunakan platform Docker saja, Anda juga dapat membuat gambar Anda sendiri selama pembuatan lingkungan dengan Dockerfile. Lihat Membangun gambar kustom dengan Dockerfile. Untuk rincian selengkapnya, Platform Docker yang dikelola ECS tidak mendukung fungsi ini.

Mengkonfigurasi pembaruan terkelola untuk lingkungan Docker

DenganPembaruan platform terkelola, Anda dapat mengkonfigurasi lingkungan Anda untuk secara otomatis memperbarui ke versi terbaru dari platform pada jadwal.

Dalam kasus lingkungan Docker, Anda mungkin ingin memutuskan apakah pembaruan platform otomatis harus terjadi di seluruh versi Docker — ketika versi platform baru menyertakan versi Docker baru. Elastic Beanstalk mendukung pembaruan platform terkelola di seluruh versi Docker saat memperbarui dari lingkungan yang menjalankan versi platform Docker yang lebih baru dari 2.9.0. Saat versi platform baru menyertakan versi baru Docker, Elastic Beanstalk menambah nomor versi pembaruan minor. Oleh karena itu, untuk mengizinkan pembaruan platform terkelola di seluruh versi Docker, aktifkan pembaruan platform terkelola untuk pembaruan versi minor dan patch. Untuk mencegah pembaruan platform terkelola di seluruh versi Docker, aktifkan pembaruan platform terkelola untuk menerapkan pembaruan versi patch saja.

Misalnya, hal berikut file konfigurasi memungkinkan pembaruan platform terkelola pada 9:00 AM UTC setiap hari Selasa untuk pembaruan versi minor dan patch, sehingga memungkinkan untuk pembaruan terkelola di seluruh versi Docker:

Example .ebextensions/ .config managed-platform-update

```
option_settings:
  aws:elasticbeanstalk:managedactions:
    ManagedActionsEnabled: true
    PreferredStartTime: "Tue:09:00"
  aws:elasticbeanstalk:managedactions:platformupdate:
    UpdateLevel: minor
```

Untuk lingkungan yang menjalankan Docker platform versi 2.9.0 atau sebelumnya, Elastic Beanstalk tidak pernah melakukan pembaruan platform terkelola jika versi platform baru mencakup versi Docker baru.

Ruang nama konfigurasi Docker

Anda dapat menggunakan file konfigurasi untuk mengatur pilihan konfigurasi dan melakukan tugas-tugas konfigurasi contoh lain selama penyebaran. Opsi konfigurasi dapat berupa platform khusus atau berlaku untuk semua platform di layanan Elastic Beanstalk secara keseluruhan. Opsi konfigurasi diatur ke dalam ruang nama.

> **Note**
>
> Informasi ini hanya berlaku untuk lingkungan Docker yang tidak menjalankan Docker Compose. Opsi ini memiliki perilaku yang berbeda dengan lingkungan Docker yang

menjalankan Docker Compose. Untuk informasi lebih lanjut tentang layanan proxy dengan Docker Compose lihat Opsi kontainer.

Platform Docker mendukung opsi di ruang nama berikut, selain opsi yang didukung untuk semua lingkungan Elastic Beanstalk:

- `aws:elasticbeanstalk:environment:proxy`— Pilih server proxy untuk lingkungan Anda. Docker mendukung baik menjalankan Nginx atau tidak ada server proxy.

File konfigurasi contoh berikut mengkonfigurasi lingkungan Docker untuk menjalankan tidak ada server proxy.

Example .ebextensions/docker-settings.config

```
option_settings:
  aws:elasticbeanstalk:environment:proxy:
    ProxyServer: none
```

Konfigurasi docker di Amazon Linux AMI (sebelumnya Amazon Linux 2)

Jika lingkungan Elastic Beanstalk Docker Anda menggunakan versi platform Amazon Linux AMI (sebelumnya Amazon Linux 2), baca informasi tambahan di bagian ini.

Menggunakan file otentikasi untuk repositori pribadi

Informasi ini relevan bagi Anda jika Anda menggunakan gambar dari repositori pribadi. Dimulai dengan Docker versi 1.7, perintah docker login mengubah nama file autentikasi, dan format file. Versi platform Amazon Linux AMI Docker (sebelumnya Amazon Linux 2) memerlukan file konfiigurasi dengan format `~/.dockercfg` lama.

Dengan Docker versi 1.7 dan setelahnya, perintah docker login untuk membuat file otentikasi di `~/.docker/config.json` dengan format berikut.

```
{
    "auths":{
      "server":{
        "auth":"key"
      }
```

```
    }
  }
```

Dengan Docker versi 1.6.2 dan sebelumnya, perintah docker login untuk membuat file autentikasi di `~/.dockercfg` dalam format berikut.

```
{
    "server" :
    {
      "auth" : "auth_token",
      "email" : "email"
    }
  }
```

Untuk mengonversi file `config.json`, hapus bagian luar kunci `auths`, tambahkan kunci `email`, dan ratakan dokumen JSON untuk mencocokkan dengan format lama.

Pada versi platform Amazon Linux 2 Docker, Elastic Beanstalk menggunakan nama dan format file otentikasi yang lebih baru. Jika Anda menggunakan versi platform Amazon Linux 2 Docker, Anda dapat menggunakan file otentikasi yang dibuat perintah docker login tanpa konversi apapun.

Mengkonfigurasi volume penyimpanan tambahan

Untuk meningkatkan kinerja di Amazon Linux AMI, Elastic Beanstalk mengonfigurasi dua volume penyimpanan Amazon EBS untuk instans Amazon lingkungan Docker Anda. EC2 Selain volume akar yang disiapkan untuk semua lingkungan Elastic Beanstalk, volume 12GB kedua bernama `xvdcz` disiapkan untuk penyimpanan gambar pada lingkungan Docker.

Jika Anda membutuhkan lebih banyak ruang penyimpanan atau peningkatan IOPS untuk Docker gambar, Anda dapat menyesuaikan volume penyimpanan gambar dengan menggunakan pilihan konfigurasi `BlockDeviceMapping` di namespace aws:autoscaling:launchconfiguration.

Misalnya, file konfigurasi berikut meningkatkan volume penyimpanan ukuran 100 GB dengan 500 provisioned IOPS:

Example .ebextensions/blockdevice-xvdcz.config

```
option_settings:
  aws:autoscaling:launchconfiguration:
    BlockDeviceMappings: /dev/xvdcz=:100::io1:500
```

Jika Anda menggunakan pilihan `BlockDeviceMappings` untuk mengkonfigurasi volume tambahan untuk aplikasi Anda, Anda harus menyertakan pemetaan untuk `xvdcz` guna memastikan bahwa aplikasi itu dibuat. Contoh berikut mengkonfigurasi dua volume, volume penyimpanan gambar `xvdcz` dengan pengaturan default dan volume aplikasi 24 GB tambahan bernama `sdh`:

Example .ebextensions/blockdevice-sdh.config

```
option_settings:
  aws:autoscaling:launchconfiguration:
    BlockDeviceMappings: /dev/xvdcz=:12:true:gp2,/dev/sdh=:24
```

> **Note**
>
> Ketika Anda mengubah pengaturan dalam namespace ini, Elastic Beanstalk menggantikan semua contoh di lingkungan Anda dengan contoh menjalankan konfigurasi baru. Lihat Perubahan konfigurasi untuk rincian selengkapnya.

Platform warisan

Bab ini mencantumkan konten yang terkait dengan platform Docker sebelumnya yang tidak lagi didukung oleh AWS Elastic Beanstalk. Topik yang tercantum di sini tetap dalam dokumen ini sebagai referensi untuk setiap pelanggan yang menggunakan fitur atau komponen ini sebelum pensiun mereka.

Topik

- Bermigrasi ke Elastic Beanstalk Docker yang berjalan di Amazon Linux 2 dari Multi-container Docker yang berjalan di Amazon Linux
- GlassFish Wadah Docker yang telah dikonfigurasi sebelumnya pada Elastic Beanstalk

Bermigrasi ke Elastic Beanstalk Docker yang berjalan di Amazon Linux 2 dari Multi-container Docker yang berjalan di Amazon Linux

Sebelum rilis ECS Running pada cabang platform Amazon Linux 2 64bit, Elastic Beanstalk menawarkan jalur migrasi alternatif ke Amazon Linux 2 untuk pelanggan dengan lingkungan berdasarkan Multi-container Docker yang berjalan pada cabang platform Amazon Linux 64bit. Topik

ini menjelaskan jalur migrasi tersebut, dan tetap ada dalam dokumen ini sebagai referensi untuk setiap pelanggan yang menyelesaikan jalur migrasi tersebut.

Kami sekarang merekomendasikan agar pelanggan dengan lingkungan berdasarkan Multi-container Docker yang berjalan pada cabang platform Amazon Linux 64bit bermigrasi ke ECS Running pada cabang platform Amazon Linux 2 64bit. Tidak seperti jalur migrasi alternatif, pendekatan ini terus menggunakan Amazon ECS untuk mengoordinasikan penerapan kontainer ke lingkungan Docker yang dikelola ECS. Aspek ini memungkinkan pendekatan yang lebih mudah. Tidak ada perubahan pada kode sumber yang diperlukan, dan `Dockerrun.aws.json` v2 yang sama didukung. Untuk informasi selengkapnya, lihat Migrasi aplikasi Elastic Beanstalk Anda dari Docker Multi-container yang dikelola ECS ke ECS di Amazon Linux 2023 AL1 .

Migrasi Legacy dari Multi-container Docker di Amazon Linux ke cabang platform Docker Amazon Linux 2

Anda dapat migrasi aplikasi yang berjalan di Platform Docker multi-kontainer di Amazon Linux AMI ke platform Amazon Linux 2 Docker. Platform Docker Multi-kontainer di Amazon Linux AMI meminta Anda menentukan gambar aplikasi bawaan untuk dijalankan sebagai kontainer. Setelah bermigrasi, Anda tidak akan lagi memiliki batasan ini, karena platform Amazon Linux 2 Docker juga memungkinkan Elastic Beanstalk untuk membuat gambar kontainer Anda selama deployment. Aplikasi Anda akan terus berjalan di lingkungan multi-kontainer dengan tambahan manfaat dari alat Docker Compose.

Docker Compose adalah alat untuk mendefinisikan dan menjalankan aplikasi Docker multi-container. Untuk mempelajari lebih lanjut tentang Docker Compose dan cara menginstalnya, lihat situs DockerSekilas tentang Docker ComposedanPasang Docker Buat.

File **`docker-compose.yml`**

Alat Docker Compose menggunakan file `docker-compose.yml` untuk konfigurasi layanan aplikasi Anda. File ini menggantikan file `Dockerrun.aws.json v2` dalam direktori proyek aplikasi Anda dan bundel sumber aplikasi. Anda membuat file `docker-compose.yml` secara manual, dan akan merasakan manfaatnya untuk mereferensikan file `Dockerrun.aws.json v2` untuk sebagian besar nilai parameter.

Di bawah ini adalah contoh file `docker-compose.yml` dan file `Dockerrun.aws.json v2` yang terkait untuk aplikasi yang sama. Untuk informasi lebih lanjut tentang file `docker-compose.yml`, lihatBuat referensi file. Untuk informasi lebih lanjut tentang file `Dockerrun.aws.json v2`, lihat `Dockerrun.aws.json v2`.

docker-compose.yml

```
version: '2.4'
services:
  php-app:
    image: "php:fpm"
    volumes:
      - "./php-app:/var/www/html:ro
"
      - "${EB_LOG_BASE_DIR}/php-app
:/var/log/sample-app"
    mem_limit: 128m
    environment:
      Container: PHP
  nginx-proxy:
    image: "nginx"
    ports:
      - "80:80"
    volumes:
      - "./php-app:/var/www/html:ro
"
      - "./proxy/conf.d:/etc/nginx/
conf.d:ro"
      - "${EB_LOG_BASE_DIR}/nginx-p
roxy:/var/log/nginx"
    mem_limit: 128m
    links:
      - php-app
```

Dockerrun.aws.json v2

```
{
  "AWSEBDockerrunVersion": 2,
  "volumes": [
    {
      "name": "php-app",
      "host": {
        "sourcePath": "/var/app/
current/php-app"
      }
    },
    {
      "name": "nginx-proxy-conf",
      "host": {
        "sourcePath": "/var/app/
current/proxy/conf.d"
      }
    }
  ],
  "containerDefinitions": [
    {
      "name": "php-app",
      "image": "php:fpm",
      "environment": [
        {
          "name": "Container",
          "value": "PHP"
        }
      ],
      "essential": true,
      "memory": 128,
      "mountPoints": [
        {
          "sourceVolume": "php-app"
,
          "containerPath": "/var/www
/html",
          "readOnly": true
        }
      ]
    },
```

docker-compose.yml	Dockerrun.aws.json v2

```
    {
      "name": "nginx-proxy",
      "image": "nginx",
      "essential": true,
      "memory": 128,
      "portMappings": [
        {
          "hostPort": 80,
          "containerPort": 80
        }
      ],
      "links": [
        "php-app"
      ],
      "mountPoints": [
        {
          "sourceVolume": "php-app"
,
          "containerPath": "/var/www
/html",
          "readOnly": true
        },
        {
          "sourceVolume": "nginx-pr
oxy-conf",
          "containerPath": "/etc/ngi
nx/conf.d",
          "readOnly": true
        },
        {
          "sourceVolume": "awseb-lo
gs-nginx-proxy",
          "containerPath": "/var/log
/nginx"
        }
      ]
    }
  ]
}
```

Pertimbangan Migrasi Tambahan

Platform Docker Amazon Linux 2 dan Multi-kontainer Docker Amazon Linux AMI platform menerapkan properti lingkungan secara berbeda. Kedua platform ini juga memiliki direktori log yang berbeda yang mana Elastic Beanstalk membuat untuk setiap kontainer mereka. Setelah Anda bermigrasi dari platform Amazon Linux AMI Multi-container Docker, Anda perlu mengetahui implementasi yang berbeda ini untuk lingkungan platform Amazon Linux 2 Docker baru Anda.

Luas	Platform Docker di Amazon Linux 2 dengan Docker Compose	Platform Docker multi-kontainer di Amazon Linux AMI
Properti lingkungan	Agar kontainer Anda dapat mengakses properti lingkungan Anda harus menambahkan referensi ke file `.env` di file `docker-compose.yml` . Elastic Beanstalk menghasilkan file `.env`, yang mencantumkan masing-masing properti sebagai variabel lingkungan. Untuk informasi selengkapnya, lihat Referensi variabel lingkungan dalam kontainer.	Elastic Beanstalk dapat secara langsung memberikan properti lingkungan ke kontainer. Kode Anda berjalan dalam kontainer dapat mengakses properti ini sebagai variabel lingkungan tanpa konfigurasi tambahan.
Direktori log	Untuk setiap kontainer Elastic Beanstalk menciptakan direktori log yang disebut `/var/log/eb-docker/containers/` *`<service name>`* (atau `${EB_LOG_BASE_DIR}/<service name>`). Untuk informasi selengkapnya, lihat Pencatatan khusus wadah Docker dengan Docker Compose.	Untuk setiap kontainer, Elastic Beanstalk menciptakan direktori log yang disebut `/var/log/containers/` *`<containername>`* . Untuk informasi lebih lanjut, lihat `mountPoints` dalam Format definisi kontainer.

Langkah migrasi

Migrasi ke Platform Docker Amazon Linux 2

1. Buat file `docker-compose.yml` untuk aplikasi Anda, berdasarkan file `Dockerrun.aws.json` v2 yang ada. Untuk informasi selengkapnya, lihat bagian File `docker-compose.yml`.

2. Dalam direktori root folder proyek aplikasi Anda, ganti file `Dockerrun.aws.json v2` dengan `docker-compose.yml` yang baru Anda buat.

   Struktur direktori Anda harus sebagai berikut.

   ```
   ~/myApplication
   |-- docker-compose.yml
   |-- .ebextensions
   |-- php-app
   |-- proxy
   ```

3. Menggunakaneb initperintah untuk mengkonfigurasi direktori lokal Anda untuk penyebaran ke Elastic Beanstalk.

   ```
   ~/myApplication$ eb init -p docker application-name
   ```

4. Menggunakaneb createperintah untuk membuat lingkungan dan menyebarkan gambar Docker Anda.

   ```
   ~/myApplication$ eb create environment-name
   ```

5. Jika aplikasi Anda adalah aplikasi web, setelah lingkungan Anda diluncurkan, gunakaneb openPerintah untuk melihatnya di peramban web.

   ```
   ~/myApplication$ eb open environment-name
   ```

6. Anda dapat menampilkan status lingkungan yang baru dibuat menggunakaneb statusPerintah.

   ```
   ~/myApplication$ eb status environment-name
   ```

GlassFish Wadah Docker yang telah dikonfigurasi sebelumnya pada Elastic Beanstalk

> **Note**
>
> Pada tanggal 18 Juli 2022, Elastic Beanstalk menetapkan status semua cabang platform berdasarkan Amazon Linux AMI () untuk pensiun. AL1 Untuk informasi selengkapnya tentang

migrasi ke cabang platform Amazon Linux 2023 saat ini dan didukung penuh, lihat. Memigrasi aplikasi Elastic Beanstalk Linux Anda ke Amazon Linux 2023 atau Amazon Linux 2

Cabang GlassFish platform Docker yang telah dikonfigurasi sebelumnya yang berjalan di Amazon Linux AMI (AL1) tidak lagi didukung. Untuk memigrasikan GlassFish aplikasi Anda ke platform Amazon Linux 2023 yang didukung, terapkan GlassFish dan kode aplikasi Anda ke image Amazon Linux 2023 Docker. Untuk informasi selengkapnya, lihat topik berikut, the section called “Tutorial - GlassFish di Docker: jalur ke Amazon Linux 2023”.

Memulai dengan wadah Docker yang telah dikonfigurasi sebelumnya - di Amazon Linux AMI (sebelumnya Amazon Linux 2)

Bagian ini menunjukkan kepada Anda cara mengembangkan aplikasi contoh secara lokal dan kemudian menerapkan aplikasi Anda ke Elastic Beanstalk dengan kontainer Docker yang telah dikonfigurasikan sebelumnya.

Siapkan lingkungan pengembangan lokal Anda

Untuk walk-through ini kita menggunakan aplikasi GlassFish contoh.

Menyiapkan lingkungan Anda

1. Buat folder baru untuk aplikasi contoh.

```
~$ mkdir eb-preconf-example
~$ cd eb-preconf-example
```

2. undung contoh kode aplikasi ke dalam folder baru.

```
~$ wget https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/samples/docker-
glassfish-v1.zip
~$ unzip docker-glassfish-v1.zip
~$ rm docker-glassfish-v1.zip
```

Mengembangkan dan menguji secara lokal

Untuk mengembangkan GlassFish aplikasi contoh

1. TambahkanDockerfileke folder akar aplikasi Anda. Dalam file, tentukan image dasar AWS Elastic Beanstalk Docker yang akan digunakan untuk menjalankan container Docker lokal Anda yang telah dikonfigurasi sebelumnya. Anda nantinya akan menerapkan aplikasi Anda ke versi platform Docker Elastic GlassFish Beanstalk Preconfigured. Pilih gambar dasar Docker yang menggunakan versi platform ini. Untuk mengetahui gambar Docker saat ini dari versi platform, lihatDocker yang dikonfigurasi sebelumnya Bagian dari AWS Elastic Beanstalk Platform yang Didukung halaman di AWS Elastic Beanstalk Platform Panduan.

   Example b-preconf-example~/E/Dockerfile

   ```
   # For Glassfish 5.0 Java 8
   FROM amazon/aws-eb-glassfish:5.0-al-onbuild-2.11.1
   ```

   Untuk informasi lebih lanjut tentang cara menggunakan `Dockerfile` dan , lihat Mempersiapkan image Docker Anda untuk penyebaran ke Elastic Beanstalk.

2. membuat gambar Docker.

   ```
   ~/eb-preconf-example$ docker build -t my-app-image .
   ```

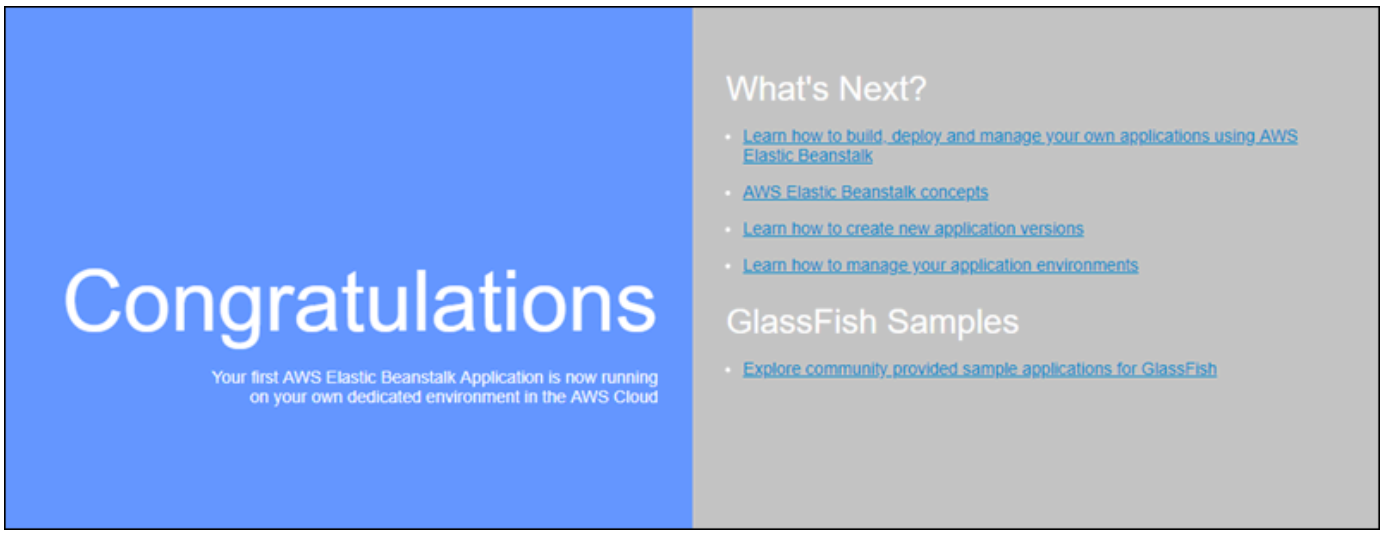
3. Jalankan kontainer Docker dari gambar.

   > **Note**
   >
   > Anda harus menyertakan bendera `-p` untuk memetakan port 8080 pada kontainer ke port localhost 3000. Elastic Beanstalk Docker kontainer selalu mengekspos aplikasi pada port 8080 pada kontainer. Bendera `-it` menjalankan gambar sebagai proses interaktif. Bendera `--rm` membersihkan sistem file kontainer ketika kontainer itu keluar. Anda dapat secara opsional menyertakan `-d` untuk menjalankan gambar sebagai daemon.

   ```
   $ docker run -it --rm -p 3000:8080 my-app-image
   ```

4. Untuk melihat contoh aplikasi, ketik URL berikut ke browser web Anda.

```
http://localhost:3000
```



Men-deploy ke Elastic Beanstalk

Setelah menguji aplikasi Anda, Anda siap untuk menerapkan ke Elastic Beanstalk.

Untuk menerapkan aplikasi Anda ke Elastic Beanstalk

1. Dalam folder akar aplikasi Anda, mengubah nama `Dockerfile` ke `Dockerfile.local`. Langkah ini diperlukan agar Elastic Beanstalk dapat `Dockerfile` menggunakan yang berisi instruksi yang benar untuk Elastic Beanstalk untuk membuat gambar Docker yang disesuaikan pada setiap instans Amazon di lingkungan Elastic Beanstalk Anda. EC2

   > **Note**
   >
   > Anda tidak perlu melakukan langkah ini jika `Dockerfile` milik anda menyertakan instruksi yang memodifikasi image Docker dasar versi platform. Anda tidak perlu menggunakan `Dockerfile` sama sekali jika `Dockerfile` Anda hanya mencakup baris `FROM` untuk menentukan gambar dasar dari yang untuk membangun kontainer. Dalam situasi itu, `Dockerfile` adalah berlebihan.

2. Buat bundel sumber aplikasi.

```
~/eb-preconf-example$ zip myapp.zip -r *
```

3. Buka konsol Elastic Beanstalk dengan tautan yang telah dikonfigurasi sebelumnya: console.aws.amazon. com/elasticbeanstalk/home#/newApplication?applicationName=Tutorial&EnvironmentType= LoadBalanced
4. Untuk Platform, di bawah Dikonfigurasi sebelumnya — Docker, pilih Glassfish.
5. Untuk Kode aplikasi, pilihUnggah kode Anda, lalu pilihUpload.
6. Pilih File lokal, pilih Browse, dan kemudian buka bundel sumber aplikasi yang baru saja Anda buat.
7. Pilih Unggah.
8. Pilih Tinjau dan Luncurkan.
9. Tinjau pengaturan dan kemudian pilih Buat applikasi.
10. Ketika lingkungan dibuat, Anda dapat melihat aplikasi yang digunakan. Pilih URL lingkungan yang ditampilkan di bagian atas dasbor konsol.

Menerapkan GlassFish aplikasi ke platform Docker: jalur migrasi ke Amazon Linux 2023

Tujuan dari tutorial ini adalah untuk menyediakan pelanggan yang menggunakan GlassFish platform Docker yang telah dikonfigurasi sebelumnya (berdasarkan Amazon Linux AMI) dengan jalur migrasi ke Amazon Linux 2023. Anda dapat memigrasikan GlassFish aplikasi Anda ke Amazon Linux 2023 dengan menerapkan GlassFish dan kode aplikasi Anda ke image Amazon Linux 2023 Docker.

Tutorial memandu Anda menggunakan platform AWS Elastic Beanstalk Docker untuk menyebarkan aplikasi berdasarkan server GlassFish aplikasi Java EE ke lingkungan Elastic Beanstalk.

Kami menunjukkan dua pendekatan untuk membangun gambar Docker:

- Sederhana - Berikan kode sumber GlassFish aplikasi Anda dan biarkan Elastic Beanstalk membangun dan menjalankan image Docker sebagai bagian dari penyediaan lingkungan Anda. Ini mudah disiapkan, dengan biaya waktu penyediaan instans yang meningkat..
- Lanjutan— Buat gambar Docker kustom yang berisi kode aplikasi dan dependensi Anda, dan berikan ke Elastic Beanstalk untuk digunakan di lingkungan Anda. Pendekatan ini sedikit lebih terlibat, dan mengurangi waktu penyediaan contoh di lingkungan Anda.

Prasyarat

Tutorial ini mengasumsikan bahwa Anda memiliki pengetahuan tentang operasi Elastic Beanstalk, Elastic antarmuka baris perintah (EB CLI), dan Docker. Jika belum, ikuti petunjuk diPelajari cara memulai dengan Elastic Beanstalkuntuk meluncurkan lingkungan Elastic Beanstalk pertama Anda. Tutorial ini menggunakan EB CLI, tetapi Anda juga dapat membuat lingkungan dan meng-upload aplikasi dengan menggunakan konsol Elastic Beanstalk.

Untuk mengikuti tutorial ini, Anda juga akan membutuhkan komponen Docker berikut:

- Instalasi lokal Docker yang berfungsi. Untuk informasi lebih lanjut, lihat Dapatkan Docker di situs web dokumentasi Docker.
- Akses ke Docker Hub. Anda harus membuat Docker ID untuk mengakses Docker Hub. Untuk informasi lebih lanjut, lihat Bagikan aplikasi di situs web dokumentasi Docker.

Untuk mempelajari lebih lanjut tentang mengkonfigurasi lingkungan Docker pada platform Elastic Beanstalk, lihat Mempersiapkan image Docker Anda untuk penyebaran ke Elastic Beanstalk dalam bab yang sama ini.

Contoh sederhana: berikan kode aplikasi Anda

Ini adalah cara mudah untuk menyebarkan GlassFish aplikasi Anda. Anda memberikan kode sumber aplikasi Anda bersama dengan `Dockerfile`Termasuk dalam tutorial ini. Elastic Beanstalk membangun gambar Docker yang mencakup aplikasi Anda dan tumpukan perangkat lunak. GlassFish Kemudian Elastic Beanstalk menjalankan gambar pada contoh lingkungan Anda.

Masalah dengan pendekatan ini adalah bahwa Elastic Beanstalk menggambar Docker lokal setiap kali menciptakan sebuah contoh untuk lingkungan Anda. Pembuatan gambar meningkatkan waktu penyediaan instans. Dampak ini tidak terbatas pada pembuatan lingkungan awal—hal ini juga terjadi selama tindakan skale-out.

Untuk meluncurkan lingkungan dengan contoh GlassFish aplikasi

1. Unduh contoh `docker-glassfish-al2-v1.zip`, dan kemudian perluas file `.zip` ke dalam direktori di lingkungan pengembangan Anda.

```
~$ curl https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/samples/docker-
glassfish-al2-v1.zip --output docker-glassfish-al2-v1.zip
~$ mkdir glassfish-example
```

```
~$ cd glassfish-example
~/glassfish-example$ unzip ../docker-glassfish-al2-v1.zip
```

Struktur direktori Anda harus sebagai berikut.

```
~/glassfish-example
|-- Dockerfile
|-- Dockerrun.aws.json
|-- glassfish-start.sh
|-- index.jsp
|-- META-INF
|   |-- LICENSE.txt
|   |-- MANIFEST.MF
|   `-- NOTICE.txt
|-- robots.txt
`-- WEB-INF
    `-- web.xml
```

File - file berikut adalah kunci untuk membuat dan menjalankan kontainer Docker di lingkungan Anda:

- `Dockerfile`— Menyediakan instruksi yang Docker gunakan untuk membuat gambar dengan aplikasi Anda dan dependensi yang diperlukan.
- `glassfish-start.sh`— Skrip shell yang dijalankan oleh image Docker untuk memulai aplikasi Anda.
- `Dockerrun.aws.json`— Menyediakan kunci logging, untuk memasukkan log server GlassFish aplikasi dalam permintaan file log. Jika Anda tidak tertarik dengan GlassFish log, Anda dapat menghilangkan file ini.

2. Mengkonfigurasi direktori lokal Anda untuk diterapkan ke Elastic Beanstalk.

```
~/glassfish-example$ eb init -p docker glassfish-example
```

3. (Opsional) Gunakan perintah eb local run untuk membuat dan menjalankan kontainer Anda secara lokal.

```
~/glassfish-example$ eb local run --port 8080
```

> **Note**
>
> Untuk mempelajari lebih lanjut tentang eb local, perintah lihat the section called "eb local". Perintah tidak didukung pada Windows. Atau, Anda dapat membuat dan menjalankan kontainer Anda dengan perintah docker build dan docker run. Untuk informasi lebih lanjut, lihat Dokumentasi Docker .

4. (Opsional) Saat kontainer Anda berjalan, gunakan perintah eb local open untuk melihat aplikasi Anda di peramban web. Alternatif lainnya, buka http://localhost:8080/ di peramban web.

```
~/glassfish-example$ eb local open
```

5. Gunakan perintah eb create untuk menciptakan lingkungan dan menyebarkan aplikasi Anda.

```
~/glassfish-example$ eb create glassfish-example-env
```

6. Setelah lingkungan Anda diluncurkan, gunakan perintah eb open untuk melihatnya di peramban web.

```
~/glassfish-example$ eb open
```

Ketika Anda selesai bekerja dengan contoh, akhiri lingkungan dan hapus sumber daya terkait.

```
~/glassfish-example$ eb terminate --all
```

Contoh lanjutan: berikan gambar docker yang sudah dibangun sebelumnya

Ini adalah cara yang lebih canggih untuk menyebarkan GlassFish aplikasi Anda. Berdasarkan contoh pertama, Anda membuat image Docker yang berisi kode aplikasi dan tumpukan GlassFish perangkat lunak, dan mendorongnya ke Docker Hub. Setelah Anda melakukan langkah satu kali ini, Anda dapat meluncurkan lingkungan Elastic Beanstalk berdasarkan gambar kustom Anda.

Ketika Anda meluncurkan lingkungan dan memberikan gambar Docker Anda, misalnya di lingkungan Anda mengunduh dan gunakan gambar ini secara langsung dan tidak perlu untuk membangun gambar Docker. Oleh karena itu, waktu penyediaan instance berkurang.

> **Catatan**
>
> - Langkah-langkah berikut membuat gambar Docker yang tersedia untuk umum.
> - Anda akan menggunakan perintah Docker dari instalasi Docker lokal Anda, bersamaan dengan mandat Docker Hub Anda. Untuk informasi selengkapnya, lihat bagian Prasyarat sebelumnya dalam topik ini.

Untuk meluncurkan lingkungan dengan image Docker GlassFish aplikasi bawaan

1. Unduh dan perluas contoh `docker-glassfish-al2-v1.zip` seperti pada contoh sederhanasebelumnya. Jika Anda telah menyelesaikan contoh tersebut, Anda dapat menggunakan direktori yang sudah Anda miliki.
2. Membuat gambar Docker dan Mendorong ke Docker Hub. Masukkan ID Docker Anda untuk masuk *`docker-id`* ke Docker Hub.

   ```
   ~/glassfish-example$ docker build -t docker-id/beanstalk-glassfish-example:latest .
   ~/glassfish-example$ docker push docker-id/beanstalk-glassfish-example:latest
   ```

   > **Note**
   >
   > Sebelum mendorong gambar, Anda mungkin harus menjalankan docker login. Anda akan diminta untuk kredensial Docker Hub Anda jika Anda menjalankan perintah tanpa parameter.

3. Membuat direktori tambahan.

   ```
   ~$ mkdir glassfish-prebuilt
   ~$ cd glassfish-prebuilt
   ```

4. Menyalin contoh berikut ke dalam sebuah file bernama `Dockerrun.aws.json`.

   Example **`~/glassfish-prebuilt/Dockerrun.aws.json`**

   ```
   {
     "AWSEBDockerrunVersion": "1",
     "Image": {
       "Name": "docker-username/beanstalk-glassfish-example"
   ```

```
  },
  "Ports": [
    {
      "ContainerPort": 8080,
      "HostPort": 8080
    }
  ],
  "Logging": "/usr/local/glassfish5/glassfish/domains/domain1/logs"
}
```

5. Mengkonfigurasi direktori lokal Anda untuk deployment ke Elastic Beanstalk.

```
~/glassfish-prebuilt$ eb init -p docker glassfish-prebuilt$
```

6. (Opsional) Gunakan perintah eb local run untuk menjalankan kontainer Anda secara lokal.

```
~/glassfish-prebuilt$ eb local run --port 8080
```

7. (Opsional) Saat kontainer Anda berjalan, gunakan perintah eb local open untuk melihat aplikasi Anda di peramban web. Alternatif lainnya, buka http://localhost:8080/ di peramban web.

```
~/glassfish-prebuilt$ eb local open
```

8. Gunakan perintah eb create untuk membuat lingkungan dan menyebarkan gambar Docker Anda.

```
~/glassfish-prebuilt$ eb create glassfish-prebuilt-env
```

9. Setelah lingkungan Anda diluncurkan, gunakan perintah eb open untuk melihatnya di peramban web.

```
~/glassfish-prebuilt$ eb open
```

Ketika Anda selesai bekerja dengan contoh, akhiri lingkungan dan hapus sumber daya terkait.

```
~/glassfish-prebuilt$ eb terminate --all
```

Memantau lingkungan di Elastic Beanstalk

Dengan pemantauan kesehatan Elastic Beanstalk, Anda dapat memverifikasi ketersediaan aplikasi dan membuat peringatan yang aktif ketika metrik melebihi ambang batas Anda. Anda dapat menggunakan pemantauan kesehatan Elastic Beanstalk di konsol dan baris perintah untuk melacak status lingkungan Anda.

Topik

- Memantau kesehatan lingkungan di konsol AWS manajemen
- Menggunakan EB CLI untuk memantau kondisi lingkungan
- Pelaporan kondisi dasar
- Peningkatan pelaporan dan pemantauan kesehatan di Elastic Beanstalk
- Mengelola alarm
- Melihat riwayat perubahan lingkungan Elastic Beanstalk
- Melihat alur kejadian lingkungan Elastic Beanstalk
- Membuat daftar dan menghubungkan ke server instans
- Melihat log dari EC2 instans Amazon di lingkungan Elastic Beanstalk Anda

Memantau kesehatan lingkungan di konsol AWS manajemen

Anda dapat mengakses informasi operasional tentang aplikasi Anda dari konsol Elastic Beanstalk. Konsol tersebut menampilkan status lingkungan dan kondisi aplikasi Anda secara sekilas. Di halaman Lingkungan konsol dan di setiap halaman aplikasi, lingkungan pada daftar diberi kode warna untuk menunjukkan status.

Untuk memantau lingkungan di konsol Elastic Beanstalk

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Di panel navigasi, pilih Pemantauan.

Halaman Pemantauan menunjukkan statistik keseluruhan tentang lingkungan Anda, seperti penggunaan CPU dan latensi rata-rata. Selain statistik keseluruhan, Anda dapat melihat grafik

pemantauan yang menunjukkan penggunaan sumber daya dari waktu ke waktu. Anda dapat mengklik salah satu grafik untuk melihat informasi lebih detail.

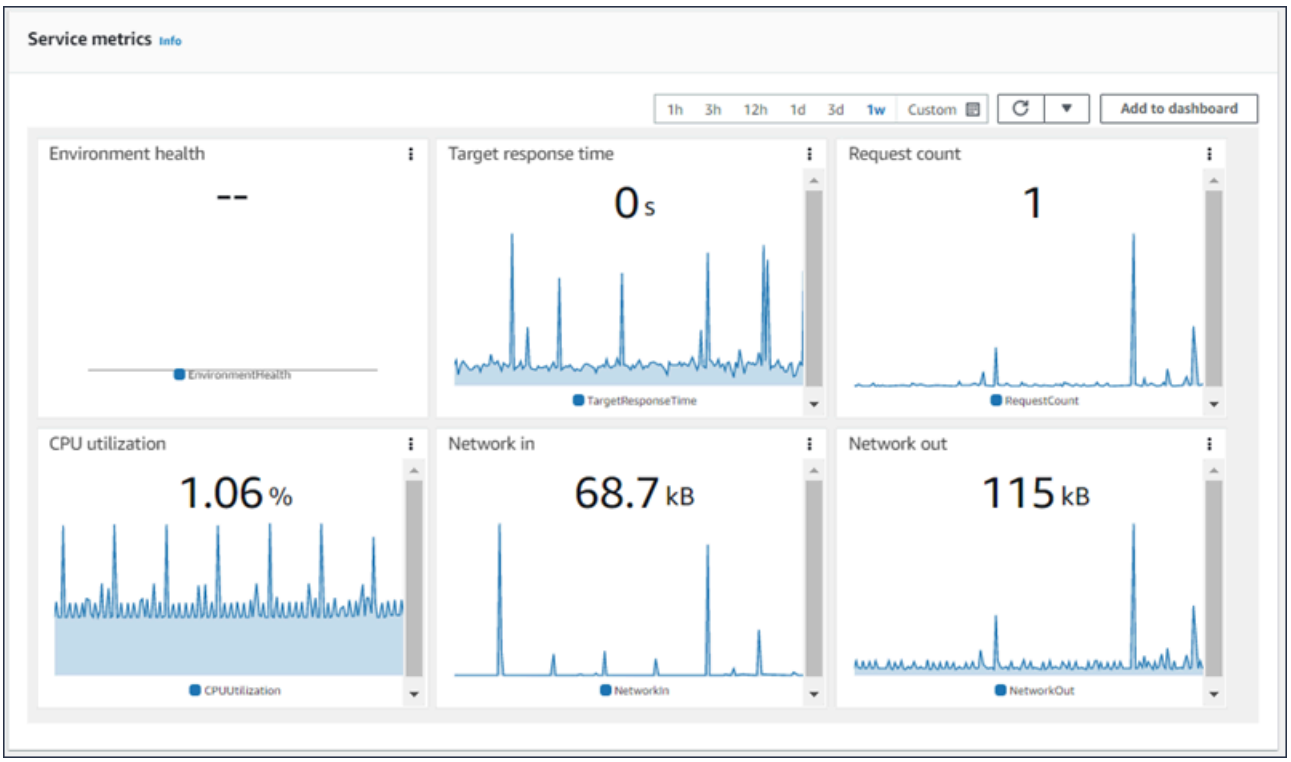
> **Note**
>
> Secara default, hanya CloudWatch metrik dasar yang diaktifkan, yang mengembalikan data dalam periode lima menit. Anda dapat mengaktifkan CloudWatch metrik satu menit yang lebih terperinci dengan mengedit pengaturan konfigurasi lingkungan Anda.

Grafik pemantauan

Halaman Pemantauan menampilkan ikhtisar metrik terkait kesehatan untuk lingkungan Anda. Ini termasuk set metrik default yang disediakan oleh Elastic Load Balancing dan EC2 Amazon, dan grafik yang menunjukkan bagaimana kesehatan lingkungan telah berubah dari waktu ke waktu.

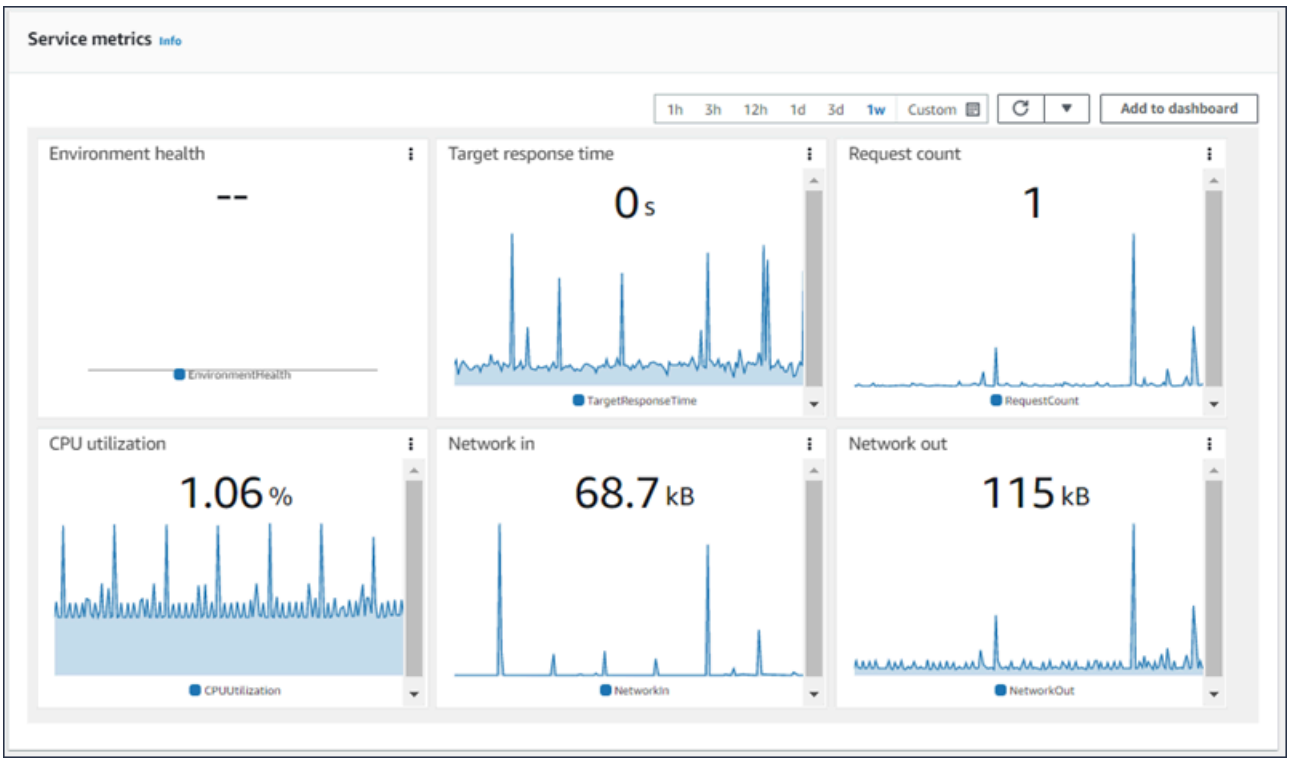
Bilah di atas grafik menyediakan berbagai interval waktu untuk Anda pilih. Misalnya, pilih 1w untuk menampilkan informasi yang membentang selama seminggu terakhir. Atau pilih 3h untuk menampilkan informasi yang mencakup selama tiga jam terakhir.

Untuk variasi pilihan interval waktu yang lebih banyak, pilih Kustom. Dari sini Anda memiliki dua opsi rentang: Absolute atau Relatif. Opsi Absolute memungkinkan Anda menentukan rentang tanggal tertentu, seperti 1 Januari 2023 hingga 30 Juni 2023. Opsi Relatif memungkinkan untuk memilih bilangan bulat dengan unit waktu tertentu: Menit, Jam, Hari, Minggu, atau Bulan. Contohnya termasuk 10 Jam, 10 Hari, dan 10 Bulan.



Menyesuaikan konsol pemantauan

Untuk membuat dan melihat metrik khusus, Anda harus menggunakan Amazon CloudWatch. Dengan CloudWatch Anda dapat membuat dasbor khusus untuk memantau sumber daya Anda dalam satu tampilan. Pilih Tambahkan ke dasbor untuk menavigasi ke CloudWatch konsol Amazon dari halaman Pemantauan. Amazon CloudWatch memberi Anda opsi untuk membuat dasbor baru atau memilih yang sudah ada. Untuk informasi selengkapnya, lihat Menggunakan CloudWatch dasbor Amazon di Panduan CloudWatch Pengguna Amazon.



Elastic Load Balancing dan EC2 metrik Amazon diaktifkan untuk semua lingkungan.

Dengan kesehatan yang ditingkatkan, EnvironmentHealth metrik diaktifkan, dan grafik ditambahkan ke konsol pemantauan secara otomatis. Kondisi yang ditingkatkan juga menambahkan Halaman kondisi ke konsol manajemen. Untuk daftar metrik kesehatan yang disempurnakan yang tersedia, lihatMenerbitkan metrik CloudWatch kustom Amazon untuk suatu lingkungan.

Menggunakan EB CLI untuk memantau kondisi lingkungan

Elastic Beanstalk Command Line Interface (EB CLI) adalah alat baris perintah untuk mengelola lingkungan. AWS Elastic Beanstalk Anda juga dapat menggunakan EB CLI untuk memantau kondisi lingkungan Anda secara langsung dan dengan lebih terperinci daripada yang saat ini tersedia pada konsol Elastic Beanstalk

Setelah menginstal dan mengkonfigurasi CLI EB, Anda dapat meluncurkan lingkungan baru dan menyebarkan kode Anda ke sana dengan perintah. eb create Jika Anda sudah memiliki lingkungan yang Anda buat di konsol Elastic Beanstalk, Anda dapat melampirkan EB CLI dengan menjalankan eb init di folder proyek dan mengikuti petunjuknya (folder proyek bisa kosong).

> **Important**
>
> Pastikan bahwa Anda menggunakan versi terbaru dari EB CLI dengan menjalankan `pip install` dengan pilihan `--upgrade`:
>
> ```
> $ sudo pip install --upgrade awsebcli
> ```
>
> Untuk instruksi instalasi EB CLI yang lengkap, lihat Instal EB CLI dengan skrip pengaturan (disarankan).

Untuk menggunakan EB CLI untuk memantau kondisi lingkungan Anda, Anda harus terlebih dahulu mengonfigurasi folder proyek lokal dengan menjalankan eb init dan mengikuti petunjuknya. Untuk instruksi yang lebih lengkap, lihat Mengonfigurasi EB CLI.

Jika Anda sudah memiliki lingkungan yang berjalan di Elastic Beanstalk dan ingin menggunakan EB CLI untuk memantau kondisinya, ikuti langkah-langkah ini untuk melampirkannya ke lingkungan yang ada.

Untuk melampirkan EB CLI ke lingkungan yang ada

1. Buka terminal baris perintah dan arahkan ke folder pengguna Anda.
2. Buat dan buka folder baru untuk lingkungan Anda.
3. Jalankan perintah eb init, dan kemudian pilih aplikasi dan lingkungan yang kondisinya ingin Anda pantau. Jika Anda hanya memiliki satu lingkungan yang menjalankan aplikasi yang Anda pilih, EB CLI akan memilihnya secara otomatis dan Anda tidak perlu memilih lingkungan, seperti yang ditunjukkan pada contoh berikut.

   ```
   ~/project$ eb init
   Select an application to use
   1) elastic-beanstalk-example
   2) [ Create new Application ]
   (default is 2): 1
   Select the default environment.
   You can change this later by typing "eb use [environment_name]".
   1) elasticBeanstalkEx2-env
   2) elasticBeanstalkExa-env
   (default is 1): 1
   ```

Untuk memantau kondisi dengan menggunakan EB CLI

1. Buka baris perintah dan arahkan ke folder proyek Anda.
2. Jalankan perintah eb health untuk menampilkan status kondisi instans di lingkungan Anda. Dalam contoh ini, ada lima instans yang berjalan pada lingkungan Linux.

```
~/project $ eb health
 elasticBeanstalkExa-env                                Ok
 2015-07-08 23:13:20
WebServer
    Ruby 2.1 (Puma)
  total      ok    warning  degraded  severe    info   pending  unknown
    5        5        0        0        0        0        0        0

  instance-id   status     cause
                                          health
    Overall     Ok
  i-d581497d    Ok
  i-d481497c    Ok
  i-136e00c0    Ok
  i-126e00c1    Ok
  i-8b2cf575    Ok

  instance-id   r/sec    %2xx   %3xx   %4xx   %5xx      p99      p90      p75
 p50     p10                                requests
    Overall     671.8   100.0    0.0    0.0    0.0    0.003    0.002    0.001
 0.001   0.000
  i-d581497d    143.0    1430      0      0      0    0.003    0.002    0.001
 0.001   0.000
  i-d481497c    128.8    1288      0      0      0    0.003    0.002    0.001
 0.001   0.000
  i-136e00c0    125.4    1254      0      0      0    0.004    0.002    0.001
 0.001   0.000
  i-126e00c1    133.4    1334      0      0      0    0.003    0.002    0.001
 0.001   0.000
  i-8b2cf575    141.2    1412      0      0      0    0.003    0.002    0.001
 0.001   0.000

  instance-id   type       az   running     load 1  load 5      user%  nice%
 system%  idle%   iowait%                            cpu
  i-d581497d    t2.micro   1a   12 mins        0.0    0.04        6.2    0.0
 1.0   92.5       0.1
```

```
  i-d481497c    t2.micro   1a   12 mins       0.01    0.09        5.9    0.0
1.6   92.4       0.1
  i-136e00c0    t2.micro   1b   12 mins       0.15    0.07        5.5    0.0
0.9   93.2       0.0
  i-126e00c1    t2.micro   1b   12 mins       0.17    0.14        5.7    0.0
1.4   92.7       0.1
  i-8b2cf575    t2.micro   1c   1 hour        0.19    0.08        6.5    0.0
1.2   92.1       0.1

  instance-id   status     id   version              ago
                                      deployments
  i-d581497d    Deployed   1    Sample Application   12 mins
  i-d481497c    Deployed   1    Sample Application   12 mins
  i-136e00c0    Deployed   1    Sample Application   12 mins
  i-126e00c1    Deployed   1    Sample Application   12 mins
  i-8b2cf575    Deployed   1    Sample Application   1 hour
```

Dalam contoh ini, ada satu instans yang berjalan pada lingkungan Windows.

```
~/project $ eb health
 WindowsSampleApp-env                                  Ok
      2018-05-22 17:33:19
WebServer                                              IIS 10.0 running on 64bit
 Windows Server 2016/2.2.0
  total      ok    warning  degraded  severe    info   pending  unknown
    1        1        0        0        0        0        0        0

  instance-id           status     cause
                                          health
    Overall             Ok
  i-065716fba0e08a351   Ok

  instance-id           r/sec    %2xx   %3xx   %4xx   %5xx      p99      p90
 p75     p50     p10                         requests
    Overall              13.7   100.0    0.0    0.0    0.0    1.403    0.970
 0.710   0.413   0.079
  i-065716fba0e08a351     2.4   100.0    0.0    0.0    0.0    1.102*   0.865
 0.601   0.413   0.091

  instance-id           type       az   running     % user time    % privileged
 time  % idle time                                  cpu
  i-065716fba0e08a351   t2.large   1b   4 hours             0.2
 0.1         99.7
```

```
  instance-id           status     id   version              ago
                                      deployments
  i-065716fba0e08a351   Deployed   2    Sample Application   4 hours
```

Membaca output

Output menampilkan nama lingkungan, kondisi lingkungan secara keseluruhan, dan tanggal saat ini di bagian atas layar.

```
elasticBeanstalkExa-env                                Ok
 2015-07-08 23:13:20
```

Tiga baris berikutnya menampilkan jenis lingkungan (" WebServer "dalam hal ini), konfigurasi (Ruby 2.1 dengan Puma), dan rincian berapa banyak contoh di masing-masing dari tujuh status.

```
WebServer
 Ruby 2.1 (Puma)
  total      ok    warning  degraded  severe    info   pending  unknown
    5        5        0        0        0        0        0        0
```

Sisa output dibagi menjadi empat bagian. Yang pertama menampilkan status dan penyebab status untuk lingkungan secara keseluruhan, dan kemudian untuk setiap instans. Contoh berikut menunjukkan dua instans di lingkungan dengan status `Info` dan penyebab yang menunjukkan bahwa deployment telah dimulai.

```
  instance-id    status     cause
                                    health
    Overall      Ok
  i-d581497d     Info       Performing application deployment (running for 3 seconds)
  i-d481497c     Info       Performing application deployment (running for 3 seconds)
  i-136e00c0     Ok
  i-126e00c1     Ok
  i-8b2cf575     Ok
```

Untuk informasi tentang status kondisi dan warna, lihat Warna dan status kondisi.

Bagian permintaan menampilkan informasi dari log server web pada setiap instans. Dalam contoh ini, setiap instans mengambil permintaan secara normal dan tidak ada kesalahan.

```
  instance-id    r/sec    %2xx   %3xx   %4xx   %5xx      p99      p90      p75      p50
    p10                              requests
    Overall      13.7    100.0    0.0    0.0    0.0    1.403    0.970    0.710   0.413
0.079
i-d581497d       2.4     100.0    0.0    0.0    0.0    1.102*   0.865    0.601   0.413
0.091
i-d481497c       2.7     100.0    0.0    0.0    0.0    0.842*   0.788    0.480   0.305
0.062
i-136e00c0       4.1     100.0    0.0    0.0    0.0    1.520*   1.088    0.883   0.524
0.104
i-126e00c1       2.2     100.0    0.0    0.0    0.0    1.334*   0.791    0.760   0.344
0.197
i-8b2cf575       2.3     100.0    0.0    0.0    0.0    1.162*   0.867    0.698   0.477
0.076
```

Bagian cpu menunjukkan metrik sistem operasi untuk setiap instans. Output berbeda dengan sistem operasi. Berikut adalah output untuk lingkungan Linux.

```
 instance-id   type       az   running     load 1  load 5      user%  nice%  system%
idle%   iowait%                            cpu
 i-d581497d    t2.micro   1a   12 mins       0.0    0.03        0.2    0.0      0.0
99.7       0.1
 i-d481497c    t2.micro   1a   12 mins       0.0    0.03        0.3    0.0      0.0
99.7       0.0
 i-136e00c0    t2.micro   1b   12 mins       0.0    0.04        0.1    0.0      0.0
99.9       0.0
 i-126e00c1    t2.micro   1b   12 mins      0.01    0.04        0.2    0.0      0.0
99.7       0.1
 i-8b2cf575    t2.micro   1c   1 hour        0.0    0.01        0.2    0.0      0.1
99.6       0.1
```

Berikut adalah output untuk lingkungan Windows.

```
 instance-id           type       az   running     % user time    % privileged time  %
idle time
 i-065716fba0e08a351   t2.large   1b   4 hours            0.2                  0.0
    99.8
```

Untuk informasi tentang metrik server dan sistem operasi yang ditampilkan, lihat Metrik instans.

Bagian akhir, deployments, menunjukkan status deployment setiap instans. Jika penerapan bergulir gagal, Anda dapat menggunakan ID deployment, status, dan label versi yang ditampilkan untuk mengidentifikasi instans di lingkungan Anda yang menjalankan versi yang salah.

```
  instance-id   status     id   version              ago
                                  deployments
  i-d581497d    Deployed   1    Sample Application   12 mins
  i-d481497c    Deployed   1    Sample Application   12 mins
  i-136e00c0    Deployed   1    Sample Application   12 mins
  i-126e00c1    Deployed   1    Sample Application   12 mins
  i-8b2cf575    Deployed   1    Sample Application   1 hour
```

Tampilan kondisi interaktif

Perintah eb health menampilkan snapshot dari kondisi lingkungan Anda. Untuk memperbarui informasi yang ditampilkan setiap 10 detik, gunakan pilihan `--refresh`.

```
$ eb health --refresh
 elasticBeanstalkExa-env                          Ok
 2015-07-09 22:10:04 (1 secs)
WebServer
          Ruby 2.1 (Puma)
  total      ok    warning  degraded  severe    info   pending  unknown
    5        5        0        0        0        0        0        0

  instance-id   status     cause
                                    health
    Overall     Ok
  i-bb65c145    Ok         Application deployment completed 35 seconds ago and took 26
 seconds
  i-ba65c144    Ok         Application deployment completed 17 seconds ago and took 25
 seconds
  i-f6a2d525    Ok         Application deployment completed 53 seconds ago and took 26
 seconds
  i-e8a2d53b    Ok         Application deployment completed 32 seconds ago and took 31
 seconds
  i-e81cca40    Ok

  instance-id   r/sec    %2xx   %3xx   %4xx   %5xx      p99      p90      p75     p50
   p10                              requests
    Overall     671.8   100.0    0.0    0.0    0.0    0.003    0.002    0.001   0.001
 0.000
```

```
 i-bb65c145    143.0    1430      0      0      0    0.003    0.002    0.001   0.001
0.000
 i-ba65c144    128.8    1288      0      0      0    0.003    0.002    0.001   0.001
0.000
 i-f6a2d525    125.4    1254      0      0      0    0.004    0.002    0.001   0.001
0.000
 i-e8a2d53b    133.4    1334      0      0      0    0.003    0.002    0.001   0.001
0.000
 i-e81cca40    141.2    1412      0      0      0    0.003    0.002    0.001   0.001
0.000

 instance-id   type       az   running     load 1  load 5      user%  nice%  system%
idle%   iowait%                            cpu
 i-bb65c145    t2.micro   1a   12 mins        0.0    0.03        0.2    0.0      0.0
99.7       0.1
 i-ba65c144    t2.micro   1a   12 mins        0.0    0.03        0.3    0.0      0.0
99.7       0.0
 i-f6a2d525    t2.micro   1b   12 mins        0.0    0.04        0.1    0.0      0.0
99.9       0.0
 i-e8a2d53b    t2.micro   1b   12 mins       0.01    0.04        0.2    0.0      0.0
99.7       0.1
 i-e81cca40    t2.micro   1c   1 hour         0.0    0.01        0.2    0.0      0.1
99.6       0.1

 instance-id   status     id   version              ago
                                 deployments
 i-bb65c145    Deployed   1    Sample Application   12 mins
 i-ba65c144    Deployed   1    Sample Application   12 mins
 i-f6a2d525    Deployed   1    Sample Application   12 mins
 i-e8a2d53b    Deployed   1    Sample Application   12 mins
 i-e81cca40    Deployed   1    Sample Application   1 hour

(Commands: Help,Quit, # # # #)
```

Contoh ini menunjukkan lingkungan yang baru-baru ini telah dinaikkan skala dari satu sampai lima instans. Operasi penskalaan berhasil, dan semua instans sekarang melewati pemeriksaan kondisi dan siap untuk mengambil permintaan. Dalam mode interaktif, status kondisi diperbarui setiap 10 detik. Di sudut kanan atas, timer berhenti ke pembaruan berikutnya.

Di sudut kiri bawah, laporan menampilkan daftar pilihan. Untuk keluar dari mode interaktif, tekan Q. Untuk menggulir, tekan tombol panah. Untuk melihat daftar perintah tambahan, tekan H.

Opsi tampilan kondisi interaktif

Ketika melihat kondisi lingkungan secara interaktif, Anda dapat menggunakan tombol keyboard untuk menyesuaikan tampilan dan memberitahu Elastic Beanstalk untuk mengganti atau memulai ulang masing-masing instans. Untuk melihat daftar perintah yang tersedia saat melihat laporan kondisi dalam mode interaktif, tekan H .

```
  up,down,home,end   Scroll vertically
  left,right         Scroll horizontally
  F                  Freeze/unfreeze data
  X                  Replace instance
  B                  Reboot instance
  <,>                Move sort column left/right
  -,+                Sort order descending/ascending
  P                  Save health snapshot data file
  Z                  Toggle color/mono mode
  Q                  Quit this program

  Views
  1                  All tables/split view
  2                  Status Table
  3                  Request Summary Table
  4                  CPU%/Load Table
  H                  This help menu


(press Q or ESC to return)
```

Pelaporan kondisi dasar

Topik ini menjelaskan fungsionalitas yang ditawarkan oleh kesehatan dasar Elastic Beanstalk.

AWS Elastic Beanstalk menggunakan informasi dari berbagai sumber untuk menentukan apakah lingkungan Anda tersedia dan memproses permintaan dari Internet. Kondisi lingkungan diwakili oleh salah satu dari empat warna, dan ditampilkan pada halaman gambaran umum lingkungan dari konsol Elastic Beanstalk. Ini juga tersedia dari DescribeEnvironmentsAPI dan eb status dengan menelepon dengan CLI EB.

Sistem pelaporan kondisi dasar memberikan informasi tentang kondisi instans di lingkungan Elastic Beanstalk berdasarkan pemeriksaan kondisi yang dilakukan oleh Elastic Load Balancing untuk

lingkungan dengan beban yang seimbang, atau Amazon Elastic Compute Cloud untuk lingkungan instans tunggal.

Selain memeriksa kesehatan EC2 instans Anda, Elastic Beanstalk juga memantau sumber daya lain di lingkungan Anda dan melaporkan sumber daya yang hilang atau salah dikonfigurasi yang dapat menyebabkan lingkungan Anda menjadi tidak tersedia bagi pengguna.

Metrik yang dikumpulkan oleh sumber daya di lingkungan Anda dipublikasikan ke Amazon CloudWatch dalam interval lima menit. Ini termasuk metrik sistem operasi dari EC2, metrik permintaan dari Elastic Load Balancing. Anda dapat melihat grafik berdasarkan CloudWatch metrik ini di halaman Pemantauan konsol lingkungan. Untuk kondisi dasar, metrik ini tidak digunakan untuk menentukan kondisi lingkungan.

Topik

- Warna kondisi
- Pemeriksaan kondisi Elastic Load Balancing
- Instans tunggal dan pemeriksaan kondisi lingkungan tingkat pekerja
- Pemeriksaan tambahan
- CloudWatch Metrik Amazon

Warna kondisi

Elastic Beanstalk melaporkan kondisi lingkungan web server tergantung pada bagaimana aplikasi yang berjalan di dalamnya merespons pemeriksaan kondisi. Elastic Beanstalk menggunakan salah satu dari empat warna untuk menggambarkan status, sebagaimana ditunjukkan dalam tabel berikut:

Warna	Deskripsi
Abu-abu	Lingkungan Anda sedang diperbarui.
Hijau	Lingkungan Anda telah melewati pemeriksaan kondisi terbaru. Setidakny a satu instans di lingkungan Anda tersedia dan menerima permintaan.
Kuning	Lingkungan Anda telah gagal dalam satu atau lebih pemeriksaan kondisi. Beberapa permintaan untuk lingkungan Anda gagal.

Warna	Deskripsi
Merah	Lingkungan Anda telah gagal dalam tiga atau lebih pemeriksaan kondisi, atau sumber daya lingkungan telah menjadi tidak tersedia. Permintaan secara konsisten gagal.

Deskripsi ini hanya berlaku untuk lingkungan yang menggunakan pelaporan kondisi dasar. Lihat Warna dan status kondisi untuk rincian terkait dengan kondisi yang ditingkatkan.

Pemeriksaan kondisi Elastic Load Balancing

Dalam lingkungan dengan beban yang seimbang, Elastic Load Balancing mengirimkan permintaan ke setiap instans di lingkungan setiap 10 detik untuk mengonfirmasi bahwa instans tersebut sehat. Secara default, penyeimbang beban dikonfigurasi untuk membuka koneksi TCP pada port 80. Jika instans mengakui koneksi tersebut, itu dianggap sehat.

Anda dapat memilih untuk mengganti pengaturan ini dengan menentukan sumber daya yang ada dalam aplikasi Anda. Jika Anda menentukan jalur, seperti `/health`, URL pemeriksaan kondisi diatur ke `HTTP:80/health`. URL pemeriksaan kondisi harus diatur ke jalur yang selalu dilayani oleh aplikasi Anda. Jika diatur ke halaman statis yang dilayani atau di-cache oleh server web di depan aplikasi Anda, pemeriksaan kondisi tidak akan mengungkapkan masalah dengan server aplikasi atau kontainer web. Untuk petunjuk cara mengubah URL pemeriksaan kondisi Anda, lihat Pemeriksaan kondisi.

Jika URL pemeriksaan kondisi dikonfigurasi, Elastic Load Balancing mengharapkan permintaan GET yang dikirimkan untuk mengembalikan respons dari `200 OK`. Aplikasi gagal dalam pemeriksaan kondisi jika aplikasi tersebut gagal untuk merespon dalam waktu 5 detik atau jika aplikasi tersebut merespon dengan kode status HTTP lainnya. Setelah 5 kegagalan pemeriksaan kondisi secara berturut-turut, Elastic Load Balancing mengeluarkan instans dari layanan.

Untuk informasi lebih lanjut mengenai pemeriksaan kondisi Elastic Load Balancing, lihat Pemeriksaan Kondisi di Panduan Pengguna Elastic Load Balancing.

> **Note**
>
> Mengonfigurasi URL pemeriksaan kondisi tidak mengubah perilaku pemeriksaan kondisi grup Auto Scaling lingkungan. Instans yang tidak sehat dihapus dari penyeimbang beban, tetapi tidak secara otomatis diganti oleh Amazon EC2 Auto Scaling kecuali Anda mengonfigurasi

Amazon Auto EC2 Scaling untuk menggunakan pemeriksaan kesehatan Elastic Load Balancing sebagai dasar untuk mengganti instans. Untuk mengonfigurasi Amazon EC2 Auto Scaling untuk mengganti instans yang gagal dalam pemeriksaan kesehatan Elastic Load Balancing, lihat. Pengaturan pemeriksaan kesehatan Auto Scaling untuk lingkungan Elastic Beanstalk Anda

Instans tunggal dan pemeriksaan kondisi lingkungan tingkat pekerja

Dalam satu instance atau lingkungan tingkat pekerja, Elastic Beanstalk menentukan kesehatan instans dengan memantau status instans Amazon-nya. EC2 Pengaturan kesehatan Elastic Load Balancing, termasuk pemeriksaan kesehatan HTTP URLs, tidak dapat digunakan dalam jenis lingkungan ini.

Untuk informasi selengkapnya tentang pemeriksaan status EC2 instans Amazon, lihat Memantau Instans dengan Pemeriksaan Status di Panduan EC2 Pengguna Amazon.

Pemeriksaan tambahan

Selain pemeriksaan kondisi Elastic Load Balancing, Elastic Beanstalk memantau sumber daya di lingkungan Anda dan mengubah status kondisi menjadi merah jika sumber daya tersebut gagal untuk men-deploy, tidak dikonfigurasi dengan benar, atau menjadi tidak tersedia. Pemeriksaan ini mengonfirmasi bahwa:

- Grup Auto Scaling lingkungan tersedia dan memiliki minimal satu instans.
- Grup keamanan lingkungan tersedia dan dikonfigurasi untuk mengizinkan lalu lintas masuk pada port 80.
- Lingkungan CNAME ada dan menunjuk ke penyeimbang beban yang tepat.
- Dalam lingkungan pekerja, antrean Amazon Simple Queue Service (Amazon SQS) sedang disurvei setidaknya sekali setiap tiga menit.

CloudWatch Metrik Amazon

Dengan pelaporan kesehatan dasar, layanan Elastic Beanstalk tidak mempublikasikan metrik apa pun ke Amazon. CloudWatch CloudWatch Metrik yang digunakan untuk menghasilkan grafik pada halaman Pemantauan konsol lingkungan diterbitkan oleh sumber daya di lingkungan Anda.

Misalnya, EC2 menerbitkan metrik berikut untuk instans di grup Auto Scaling lingkungan Anda:

`CPUUtilization`

Persentase unit komputasi yang saat ini digunakan.

`DiskReadBytes`, `DiskReadOps`, `DiskWriteBytes`, `DiskWriteOps`

Jumlah byte yang dibaca dan ditulis, dan operasi jumlah baca dan tulis.

`NetworkIn`, `NetworkOut`

Jumlah byte yang dikirim dan diterima.

Elastic Load Balancing menerbitkan metrik berikut untuk penyeimbang beban lingkungan Anda:

`BackendConnectionErrors`

Jumlah kegagalan koneksi antara penyeimbang beban dan instans lingkungan.

`HTTPCode_Backend_2XX`, `HTTPCode_Backend_4XX`

Jumlah kode respons (2XX) yang berhasil dan kesalahan klien (4XX) yang dihasilkan oleh instans di lingkungan Anda.

`Latency`

Jumlah detik antara ketika penyeimbang beban menyampaikan permintaan ke sebuah instans dan ketika respons diterima.

`RequestCount`

Jumlah permintaan yang telah selesai.

Daftar ini bukan daftar yang komprehensif. Untuk daftar lengkap metrik yang dapat dilaporkan untuk sumber daya ini, lihat topik berikut di Panduan CloudWatch Pengembang Amazon:

Metrik

Namespace	Topik
AWS::ElasticLoadBalancing::LoadBalancer	Metrik dan Sumber Daya Elastic Load Balancing

Namespace	Topik
AWS::AutoScaling::AutoScalingKelompok	Metrik dan Sumber Daya Amazon Elastic Compute Cloud
AWS::SQS::Queue	Metrik dan Sumber Daya Amazon SQS
AWS: :RDS:: DBInstance	Dimensi dan Metrik Amazon RDS

Metrik kondisi lingkungan pekerja

Hanya untuk lingkungan pekerja, daemon SQS menerbitkan metrik khusus untuk kesehatan lingkungan CloudWatch, di mana nilai 1 adalah Hijau. Anda dapat meninjau data metrik CloudWatch kesehatan di akun menggunakan `ElasticBeanstalk/SQSD` namespace. Dimensi metrik adalah `EnvironmentName`, dan nama metrik adalah `Health`. Semua instans menerbitkan metrik mereka ke namespace yang sama.

Untuk mengaktifkan daemon untuk menerbitkan metrik, profil instans lingkungan harus memiliki izin untuk memanggil `cloudwatch:PutMetricData`. Izin ini disertakan dalam profil instans default. Untuk informasi selengkapnya, lihat Mengelola profil instans Elastic Beanstalk.

Peningkatan pelaporan dan pemantauan kesehatan di Elastic Beanstalk

Bagian ini menjelaskan fungsionalitas fitur Elastic Beanstalk Enhanced Health.

Pelaporan kesehatan yang disempurnakan adalah fitur yang dapat Anda aktifkan di lingkungan Anda AWS Elastic Beanstalk untuk memungkinkan mengumpulkan informasi tambahan tentang sumber daya di lingkungan Anda. Elastic Beanstalk menganalisis informasi yang dikumpulkan untuk memberikan gambaran yang lebih baik tentang kondisi lingkungan secara keseluruhan dan bantuan dalam identifikasi masalah yang dapat menyebabkan aplikasi Anda menjadi tidak tersedia.

Selain perubahan dalam cara kerja warna kondisi, kondisi yang ditingkatkan menambahkan deskriptor status yang menyediakan indikator kepelikan masalah yang diamati ketika lingkungan berwarna kuning atau merah. Ketika informasi lebih lanjut tersedia tentang status saat ini, Anda dapat memilih tombol Penyebab untuk melihat informasi kondisi yang detail pada halaman kondisi.

Untuk memberikan informasi kesehatan terperinci tentang EC2 instans Amazon yang berjalan di lingkungan Anda, Elastic Beanstalk menyertakan agen kesehatan di Amazon Machine Image (AMI) untuk setiap versi platform yang mendukung peningkatan kesehatan. Agen kondisi memantau log server web dan metrik sistem dan menyampaikannya ke layanan Elastic Beanstalk. Elastic Beanstalk menganalisis metrik dan data ini dari Elastic Load Balancing dan EC2 Amazon Auto Scaling untuk memberikan gambaran keseluruhan tentang kesehatan lingkungan.

Selain mengumpulkan dan menyajikan informasi tentang sumber daya lingkungan Anda, Elastic Beanstalk memantau sumber daya di lingkungan Anda untuk beberapa kondisi kesalahan dan memberikan notifikasi untuk membantu Anda menghindari kegagalan dan menyelesaikan masalah konfigurasi. Faktor-faktor yang mempengaruhi kondisi lingkungan Anda mencakup hasil dari setiap permintaan yang disajikan oleh aplikasi Anda, metrik dari sistem operasi instans Anda, dan status deployment yang terbaru.

Anda dapat melihat status kondisi secara waktu nyata dengan menggunakan halaman gambaran umum lingkungan dari konsol Elastic Beanstalk atau perintah kondisi eb di antarmuka baris Elastic Beanstalk (EB CLI). Untuk merekam dan melacak kesehatan lingkungan dan instans dari waktu ke waktu, Anda dapat mengonfigurasi lingkungan Anda untuk mempublikasikan informasi yang dikumpulkan oleh Elastic Beanstalk untuk pelaporan kesehatan yang ditingkatkan ke CloudWatch Amazon sebagai metrik khusus. CloudWatch biaya untuk metrik khusus berlaku untuk semua metrik selain`EnvironmentHealth`, yang gratis.

> **Catatan platform Windows**
>
> Ketika Anda mengaktifkan pelaporan kondisi yang ditingkatkan pada lingkungan Windows Server, jangan mengubah konfigurasi log IIS. Agar pemantauan kondisi yang ditingkatkan bekerja dengan benar, logging IIS harus dikonfigurasi dengan format W3C dan hanya peristiwa ETW atau kedua tujuan peristiwa berkas log dan peristiwa ETW.
> Selain itu, jangan nonaktifkan atau hentikan layanan Windows agen kondisi Elastic Beanstalk pada salah satu instans lingkungan Anda. Untuk mengumpulkan dan melaporkan informasi kondisi yang ditingkatkan pada instans, layanan ini harus diaktifkan dan berjalan.

Pertama kali Anda membuat lingkungan Elastic Beanstalk meminta Anda untuk membuat peran yang diperlukan dan memungkinkan pelaporan kesehatan yang ditingkatkan secara default. Lanjutkan membaca untuk detail tentang cara kerja pelaporan kondisi yang ditingkatkan, atau lihat Mengaktifkan pelaporan kondisi yang ditingkatkan Elastic Beanstalk untuk mulai menggunakannya segera.

Topik

- Agen kondisi Elastic Beanstalk
- Faktor dalam menentukan kondisi instans dan lingkungan
- Penyesuaian aturan pemeriksaan kondisi
- Peran kondisi yang ditingkatkan
- Otorisasi kondisi yang ditingkatkan
- Peristiwa kondisi yang ditingkatkan
- Perilaku pelaporan kondisi yang ditingkatkan selama pembaruan, deployment, dan penskalaan
- Mengaktifkan pelaporan kondisi yang ditingkatkan Elastic Beanstalk
- Pemantauan kondisi yang ditingkatkan dengan konsol manajemen lingkungan
- Warna dan status kondisi
- Metrik instans
- Mengonfigurasi aturan kondisi yang ditingkatkan untuk lingkungan
- Menerbitkan metrik CloudWatch kustom Amazon untuk suatu lingkungan
- Menggunakan pelaporan kondisi yang ditingkatkan dengan API Elastic Beanstalk
- Format log kondisi yang ditingkatkan
- Pemberitahuan dan pemecahan masalah

Agen kondisi Elastic Beanstalk

Agen kesehatan Elastic Beanstalk adalah proses daemon (atau layanan, di lingkungan Windows) yang berjalan di setiap instans EC2 Amazon di lingkungan Anda, memantau sistem operasi dan metrik kesehatan tingkat aplikasi, serta melaporkan masalah ke Elastic Beanstalk. Agen kondisi termasuk dalam semua versi platform yang dimulai dengan versi 2.0 dari setiap platform.

Agen kesehatan melaporkan metrik serupa dengan yang diterbitkan CloudWatch oleh Amazon EC2 Auto Scaling dan Elastic Load Balancing sebagai bagian dari pelaporan kesehatan dasar, termasuk beban CPU, kode HTTP, dan latensi. Agen kondisi, bagaimanapun, melaporkan langsung ke Elastic Beanstalk, dengan granularitas yang lebih besar dan frekuensi dari pelaporan kondisi dasar.

Untuk kondisi dasar, metrik ini diterbitkan setiap lima menit dan dapat dipantau dengan grafik di konsol manajemen lingkungan. Dengan kondisi yang ditingkatkan, agen kondisi Elastic Beanstalk melaporkan metrik ke Elastic Beanstalk setiap 10 detik. Elastic Beanstalk menggunakan metrik yang

disediakan oleh agen kondisi untuk menentukan status kondisi setiap instans di lingkungan dan, dikombinasikan dengan faktor, untuk menentukan kondisi lingkungan secara keseluruhan.

Kesehatan lingkungan secara keseluruhan dapat dilihat secara real time di halaman ikhtisar lingkungan konsol Elastic Beanstalk, dan diterbitkan CloudWatch oleh Elastic Beanstalk setiap 60 detik. Anda dapat melihat metrik yang terperinci yang dilaporkan oleh agen kondisi secara waktu nyata dengan perintah eb health di EB CLI.

Dengan biaya tambahan, Anda dapat memilih untuk mempublikasikan instans individual dan metrik tingkat lingkungan setiap 60 detik. CloudWatch Metrik yang diterbitkan untuk kemudian CloudWatch dapat digunakan untuk membuat grafik pemantauan di konsol manajemen lingkungan.

Pelaporan kesehatan yang ditingkatkan hanya dikenakan biaya jika Anda memilih untuk mempublikasikan metrik kesehatan yang disempurnakan. CloudWatch Bila Anda menggunakan kondisi yang ditingkatkan, Anda tetap mendapatkan metrik kondisi dasar yang diterbitkan secara gratis, meskipun Anda tidak memilih untuk menerbitkan metrik kondisi yang ditingkatkan.

Lihat Metrik instans untuk detail metrik yang dilaporkan oleh agen kondisi. Untuk detail tentang penerbitan metrik kesehatan yang disempurnakan CloudWatch, lihatMenerbitkan metrik CloudWatch kustom Amazon untuk suatu lingkungan.

Faktor dalam menentukan kondisi instans dan lingkungan

Selain pemeriksaan sistem pelaporan kondisi dasar, termasuk Pemeriksaan kondisi Elastic Load Balancing dan pemantauan sumber daya, pelaporan kondisi yang ditingkatkan Elastic Beanstalk mengumpulkan data tambahan tentang keadaan instans di lingkungan Anda. Ini termasuk metrik sistem operasi, log server, dan keadaan operasi lingkungan yang sedang berlangsung seperti deployment dan pembaruan. Layanan pelaporan kondisi Elastic Beanstalk menggabungkan informasi dari semua sumber yang tersedia dan menganalisanya untuk menentukan kondisi lingkungan secara keseluruhan.

Operasi dan perintah

Ketika Anda melakukan operasi pada lingkungan Anda, seperti men-deploy versi baru dari aplikasi, Elastic Beanstalk membuat beberapa perubahan yang mempengaruhi status kondisi lingkungan.

Misalnya, ketika Anda men-deploy versi baru aplikasi untuk lingkungan yang menjalankan beberapa instans, Anda mungkin melihat pesan yang mirip dengan berikut ini saat Anda memantau kondisi lingkungan dengan EB CLI.

```
  id              status     cause
    Overall       Info       Command is executing on 3 out of 5 instances
  i-bb65c145      Pending    91 % of CPU is in use. 24 % in I/O wait
                             Performing application deployment (running for 31 seconds)
  i-ba65c144      Pending    Performing initialization (running for 12 seconds)
  i-f6a2d525      Ok         Application deployment completed 23 seconds ago and took 26
 seconds
  i-e8a2d53b      Pending    94 % of CPU is in use. 52 % in I/O wait
                             Performing application deployment (running for 33 seconds)
  i-e81cca40      Ok
```

Dalam contoh ini, status keseluruhan lingkungan adalah Ok dan penyebab status ini adalah bahwa Perintah mengeksekusi 3 dari 5 instans. Tiga instans di lingkungan memiliki status Tertunda, menunjukkan bahwa operasi sedang berlangsung.

Saat operasi selesai, Elastic Beanstalk melaporkan informasi tambahan tentang operasi tersebut. Sebagai contoh, Elastic Beanstalk menampilkan informasi berikut tentang instans yang telah diperbarui dengan versi baru dari aplikasi:

```
i-f6a2d525      Ok         Application deployment completed 23 seconds ago and took 26
 seconds
```

Informasi kondisi instans juga mencakup detail tentang deployment terbaru untuk setiap instans di lingkungan Anda. Setiap instans melaporkan ID deployment dan status. ID deployment adalah integer yang meningkat satu setiap kali Anda men-deploy versi baru dari aplikasi Anda atau mengubah pengaturan untuk opsi konfigurasi pada instans, seperti variabel lingkungan. Anda dapat menggunakan informasi deployment untuk mengidentifikasi instans yang menjalankan versi yang salah dari aplikasi Anda setelah deployment bergulir yang gagal.

Di kolom penyebab, Elastic Beanstalk mencakup pesan informasi tentang keberhasilan operasi dan kondisi sehat lainnya di beberapa pemeriksaan kondisi, tetapi mereka tidak bertahan tanpa batas waktu. Penyebab status lingkungan yang tidak sehat bertahan sampai lingkungan kembali ke status yang sehat.

Waktu perintah habis

Elastic Beanstalk memberlakukan waktu perintah habis dari waktu operasi dimulai untuk mengizinkan sebuah instans melakukan transisi ke keadaan yang sehat. Waktu perintah habis ini diatur dalam pembaruan lingkungan Anda dan konfigurasi deployment (di namespace aws:elasticbeanstalk:command) dan secara default ke 10 menit.

Selama pembaruan bergulir, Elastic Beanstalk menerapkan batas waktu terpisah untuk setiap batch dalam operasi tersebut. Waktu habis ini ditetapkan sebagai bagian dari konfigurasi pembaruan bergulir lingkungan (di namespace aws:autoscaling:updatepolicy:rollingupdate). Jika semua instance dalam batch sehat dalam batas waktu pembaruan bergulir, operasi berlanjut ke batch berikutnya. Jika tidak, operasi gagal.

> **Note**
>
> Jika aplikasi Anda tidak lolos pemeriksaan kondisi dengan status OK tetapi stabil pada tingkat yang berbeda, Anda dapat mengatur pilihan `HealthCheckSuccessThreshold` pada pilihan `aws:elasticbeanstalk:command namespace` untuk mengubah tingkat di mana Elastic Beanstalk menganggap sebuah instans sehat.

Agar lingkungan web server dianggap sehat, setiap instans di lingkungan atau batch harus lolos 12 pemeriksaan kondisi berturut-turut selama dua menit. Untuk lingkungan tingkat pekerja, setiap instans harus lolos 18 pemeriksaan kondisi. Sebelum perintah berakhir, Elastic Beanstalk tidak menurunkan status kondisi lingkungan ketika pemeriksaan kondisi gagal. Jika instans di lingkungan menjadi sehat selama waktu perintah habis, operasi berhasil.

Permintaan HTTP

Ketika tidak ada operasi yang berlangsung pada lingkungan, sumber utama informasi tentang instans dan kondisi lingkungan adalah log web server untuk setiap instans. Untuk menentukan kondisi suatu instans dan kondisi lingkungan secara keseluruhan, Elastic Beanstalk mempertimbangkan jumlah permintaan, hasil dari setiap permintaan, dan kecepatan di mana setiap permintaan diselesaikan.

Pada platform berbasis Linux, Elastic Beanstalk membaca dan mengurai log server web untuk mendapatkan informasi tentang permintaan HTTP. Pada platform Windows Server, Elastic Beanstalk menerima informasi ini langsung dari server web IIS.

Lingkungan Anda mungkin tidak memiliki server web yang aktif. Sebagai contoh, platform Multicontainer Docker tidak memasukkan server web. Platform lain memasukkan server web, dan aplikasi Anda mungkin menonaktifkannya. Dalam kasus ini, lingkungan Anda memerlukan konfigurasi tambahan untuk menyediakan agen kondisi Elastic Beanstalk dengan log dalam format yang dibutuhkan untuk menyampaikan informasi kondisi ke layanan Elastic Beanstalk. Lihat Format log kondisi yang ditingkatkan untuk detail selengkapnya.

Metrik sistem operasi

Elastic Beanstalk memantau metrik sistem operasi yang dilaporkan oleh agen kondisi untuk mengidentifikasi instans yang secara konsisten rendah pada sumber daya sistem.

Lihat Metrik instans untuk detail metrik yang dilaporkan oleh agen kondisi.

Penyesuaian aturan pemeriksaan kondisi

Pelaporan kondisi yang ditingkatkan Elastic Beanstalk bergantung pada seperangkat aturan untuk menentukan kondisi lingkungan Anda. Beberapa aturan ini mungkin tidak sesuai untuk aplikasi tertentu Anda. Kasus umum adalah aplikasi yang mengembalikan kesalahan HTTP 4xx oleh desain yang sering terjadi. Elastic Beanstalk, menggunakan salah satu aturan default, menyimpulkan bahwa ada sesuatu yang salah, dan mengubah status kondisi lingkungan Anda dari OK ke Peringatan, Terdegradasi, atau Pelik, tergantung pada tingkat kesalahan. Untuk menangani kasus ini dengan benar, Elastic Beanstalk mengizinkan Anda untuk mengonfigurasi aturan ini dan mengabaikan kesalahan HTTP 4xx aplikasi. Untuk detail selengkapnya, lihat Mengonfigurasi aturan kondisi yang ditingkatkan untuk lingkungan.

Peran kondisi yang ditingkatkan

Pelaporan kondisi yang ditingkatkan memerlukan dua peran—peran layanan untuk Elastic Beanstalk dan profil instans untuk lingkungan. Peran layanan memungkinkan Elastic Beanstalk untuk berinteraksi AWS dengan layanan lain atas nama Anda untuk mengumpulkan informasi tentang sumber daya di lingkungan Anda. Profil instans mengizinkan instans di lingkungan Anda untuk menuliskan log ke Amazon S3 dan untuk mengomunikasikan informasi kondisi yang ditingkatkan ke layanan Elastic Beanstalk.

Ketika Anda membuat lingkungan Elastic Beanstalk menggunakan konsol Elastic Beanstalk atau EB CLI, Elastic Beanstalk membuat peran layanan default dan melampirkan kebijakan terkelola yang diperlukan untuk profil instans default untuk lingkungan Anda.

Jika Anda menggunakan API, SDK, atau AWS CLI untuk membuat lingkungan, Anda harus membuat peran ini terlebih dahulu, dan menentukannya selama pembuatan lingkungan untuk menggunakan peningkatan kesehatan. Untuk petunjuk tentang cara membuat peran yang sesuai untuk lingkungan Anda, lihat Peran Elastic Beanstalk Service, profil instans, dan kebijakan pengguna.

Kami menyarankan agar Anda menggunakan kebijakan terkelola untuk profil instans dan peran layanan Anda. Kebijakan terkelola adalah kebijakan AWS Identity and Access Management

(IAM) yang dipertahankan Elastic Beanstalk. Menggunakan kebijakan terkelola menjamin bahwa lingkungan Anda memiliki semua izin yang diperlukan untuk berfungsi dengan baik.

Untuk profil instans, Anda dapat menggunakan `AWSElasticBeanstalkWebTier` atau kebijakan terkelola `AWSElasticBeanstalkWorkerTier`, untuk lingkungan tingkat server web atau tingkat pekerja, masing-masing. Untuk detail tentang dua kebijakan profil instans terkelola ini, lihat the section called “Profil instans”.

Otorisasi kondisi yang ditingkatkan

Kebijakan terkelola profil instans Elastic Beanstalk berisi izin untuk tindakan tersebut. `elasticbeanstalk:PutInstanceStatistics` Tindakan ini bukan bagian dari API Elastic Beanstalk. Ini adalah bagian dari API berbeda yang digunakan instans lingkungan secara internal untuk mengkomunikasikan informasi kesehatan yang disempurnakan ke layanan Elastic Beanstalk. Anda tidak menyebut API ini secara langsung.

Saat Anda membuat lingkungan baru, otorisasi untuk `elasticbeanstalk:PutInstanceStatistics` tindakan diaktifkan secara default. Untuk meningkatkan keamanan lingkungan Anda dan membantu mencegah spoofing data kesehatan atas nama Anda, sebaiknya Anda tetap mengaktifkan otorisasi untuk tindakan ini. Jika Anda menggunakan kebijakan terkelola untuk profil instans, fitur ini tersedia untuk lingkungan baru Anda tanpa konfigurasi lebih lanjut. Namun, Jika Anda menggunakan profil instans kustom alih-alih kebijakan terkelola, lingkungan Anda mungkin menampilkan status kesehatan Tanpa Data. Hal ini terjadi karena instans tidak diizinkan untuk tindakan yang mengomunikasikan data kondisi yang ditingkatkan ke layanan.

Untuk mengotorisasi tindakan, sertakan pernyataan berikut dalam profil instans Anda.

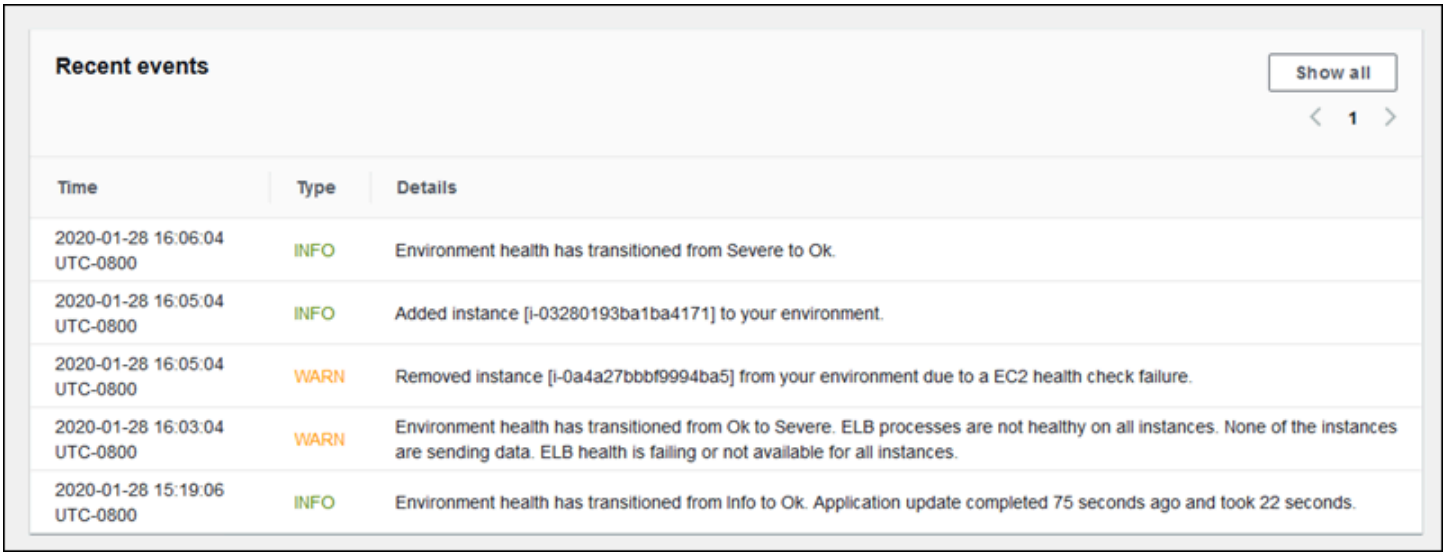
```
{
  "Sid": "ElasticBeanstalkHealthAccess",
  "Action": [
    "elasticbeanstalk:PutInstanceStatistics"
  ],
  "Effect": "Allow",
  "Resource": [
    "arn:aws:elasticbeanstalk:*:*:application/*",
    "arn:aws:elasticbeanstalk:*:*:environment/*"
  ]
}
```

Jika Anda tidak ingin menggunakan otorisasi kesehatan yang ditingkatkan saat ini, nonaktifkan dengan menyetel setel `EnhancedHealthAuthEnabled` opsi di the section called "aws:elasticbeanstalk:healthreporting:system" namespace ke. `false` Jika opsi ini dinonaktifkan, izin yang dijelaskan sebelumnya tidak diperlukan. Anda dapat menghapusnya dari profil instans untuk akses hak istimewa paling sedikit ke aplikasi dan lingkungan Anda.

> **Note**
>
> Sebelumnya pengaturan default untuk `EnhancedHealthAuthEnabled` adalah`false`, yang mengakibatkan otorisasi untuk `elasticbeanstalk:PutInstanceStatistics` tindakan juga dinonaktifkan secara default. Untuk mengaktifkan tindakan ini untuk lingkungan yang ada, setel `EnhancedHealthAuthEnabled` opsi di the section called "aws:elasticbeanstalk:healthreporting:system" namespace ke. `true` Anda dapat mengonfigurasi pilihan ini dengan menggunakan pengaturan pilihan dalam file konfigurasi.

Peristiwa kondisi yang ditingkatkan

Sistem kondisi yang ditingkatkan menghasilkan peristiwa ketika lingkungan melakukan transisi antara statusnya. Contoh berikut menunjukkan peristiwa output oleh transisi lingkungan antara status Info, OK, dan Pelik.



Ketika terjadi transisi ke keadaan yang lebih buruk, peristiwa kondisi yang ditingkatkan mencakup pesan yang menunjukkan penyebab transisi.

Tidak semua perubahan status pada tingkat instans menyebabkan Elastic Beanstalk memancarkan sebuah peristiwa. Untuk mencegah alarm palsu, Elastic Beanstalk menghasilkan peristiwa yang berhubungan dengan kondisi hanya jika masalah berlanjut di beberapa pemeriksaan.

Informasi kondisi tingkat lingkungan secara waktu nyata, termasuk status, warna, dan sebab, tersedia di gambaran umum lingkungan dari konsol Elastic Beanstalk dan EB CLI. Dengan melampirkan CLI EB untuk lingkungan Anda dan menjalankan perintah eb health, Anda juga dapat melihat status secara waktu nyata dari masing-masing instans di lingkungan Anda.

Perilaku pelaporan kondisi yang ditingkatkan selama pembaruan, deployment, dan penskalaan

Mengaktifkan pelaporan kondisi yang ditingkatkan dapat mempengaruhi perilaku lingkungan Anda selama pembaruan konfigurasi dan deployment. Elastic Beanstalk tidak akan menyelesaikan satu batch pembaruan sampai semua instans lolos pemeriksaan kondisi secara konsisten. Selain itu, karena pelaporan kondisi yang ditingkatkan menerapkan standar yang lebih tinggi untuk kondisi dan memantau lebih banyak faktor, instans yang melewati Pemeriksaan kondisi ELB tidak akan selalu lolos dengan pelaporan kondisi yang ditingkatkan. Lihat topik di pembaruan konfigurasi bergulir dan penerapan bergulir untuk detail tentang bagaimana pemeriksaan kondisi mempengaruhi proses pembaruan.

Pelaporan kondisi yang ditinkatkan juga dapat menyoroti kebutuhan untuk menetapkan URL pemeriksaan kondisi untuk Elastic Load Balancing. Ketika lingkungan Anda menaikkan skala untuk memenuhi permintaan, instans baru akan mulai mengambil permintaan segera setelah mereka lulus cukup pemeriksaan kondisi ELB. Jika URL pemeriksaan kondisi tidak dikonfigurasi, ini dapat sesedikit 20 detik setelah instans baru dapat menerima sambungan TCP.

Jika aplikasi Anda belum selesai memulai pada saat penyeimbang beban menyatakannya cukup sehat untuk menerima lalu lintas, Anda akan melihat banyaknya permintaan yang gagal, dan lingkungan Anda akan mulai gagal pemeriksaan kondisi. URL pemeriksaan kondisi yang menyentuh jalur yang dilayani oleh aplikasi Anda dapat mencegah masalah ini. Pemeriksaan kondisi ELB tidak akan lolos sampai permintaan GET ke URL pemeriksaan kondisi mengembalikan kode status 200.

Mengaktifkan pelaporan kondisi yang ditingkatkan Elastic Beanstalk

Topik ini menjelaskan bagaimana peningkatan pelaporan kesehatan diaktifkan. Ini menyediakan prosedur bagi Anda untuk mengaktifkan fitur kesehatan yang disempurnakan untuk lingkungan Anda dengan konsol Elastic Beanstalk, EB CLI, dan dengan konfigurasi.ebextensions.

Lingkungan baru yang dibuat dengan versi platform terbaru termasuk agen AWS Elastic Beanstalk kesehatan, yang mendukung peningkatan pelaporan kesehatan. Jika Anda membuat lingkungan Anda di konsol Elastic Beanstalk atau dengan EB CLI, kondisi yang ditingkatkan diaktifkan secara default. Anda juga dapat mengatur preferensi pelaporan kondisi di kode sumber aplikasi Anda menggunakan file konfigurasi.

Pelaporan kondisi yang ditingkatkan memerlukan profil instans dan peran layanan dengan serangkaian standar izin. Ketika Anda membuat lingkungan di konsol Elastic Beanstalk, Elastic Beanstalk membuat peran yang diperlukan secara otomatis. Lihat Pelajari cara memulai dengan Elastic Beanstalk untuk mendapatkan petunjuk tentang cara membuat lingkungan pertama Anda.

Topik

- Mengaktifkan pelaporan kondisi yang ditingkatkan menggunakan konsol Elastic Beanstalk
- Mengaktifkan pelaporan kondisi yang ditingkatkan menggunakan EB CLI
- Mengaktifkan pelaporan kondisi yang ditingkatkan menggunakan file konfigurasi

Mengaktifkan pelaporan kondisi yang ditingkatkan menggunakan konsol Elastic Beanstalk

Untuk mengaktifkan pelaporan kondisi yang ditingkatkan di lingkungan yang berjalan menggunakan konsol Elastic Beanstalk

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Di kategori konfigurasi Pemantauan, pilih Edit.
5. Di bawah Pelaporan kondisi, untuk Sistem, pilih Peningkatan.

> **Note**
>
> Pilihan untuk pelaporan kondisi yang ditingkatkan tidak muncul jika Anda menggunakan platform atau versi yang tidak didukung.

6. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Konsol Elastic Beanstalk secara default menjadi pelaporan kondisi yang ditingkatkan ketika Anda membuat lingkungan baru dengan versi platform versi 2 (v2). Anda dapat menonaktifkan pelaporan kondisi yang ditingkatkan dengan mengubah pilihan pelaporan kondisi selama pembuatan lingkungan.

Untuk menonaktifkan pelaporan kondisi yang ditingkatkan saat membuat lingkungan menggunakan konsol Elastic Beanstalk

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Buat aplikasi atau pilih yang sudah ada.
3. Buat lingkungan. Pada halaman Buat lingkungan baru, sebelum memilih Buat lingkungan, pilih Konfigurasikan opsi lainnya.
4. Di kategori konfigurasi Pemantauan, pilih Edit.
5. Di bawah Pelaporan kondisi, untuk Sistem, pilih Basic.
6. Pilih Simpan.

Mengaktifkan pelaporan kondisi yang ditingkatkan menggunakan EB CLI

Ketika Anda membuat lingkungan baru dengan perintah eb create, EB CLI mengaktifkan pelaporan kondisi yang ditingkatkan secara default dan memberlakukan profil instans dan peran layanan default.

Anda dapat menentukan peran layanan yang berbeda berdasarkan nama dengan menggunakan pilihan `--service-role`.

Jika Anda memiliki lingkungan yang berjalan dengan pelaporan kondisi dasar pada versi platform v2 dan Anda ingin beralih ke kondisi yang ditingkatkan, ikuti langkah-langkah berikut.

Untuk mengaktifkan kondisi yang ditingkatkan pada lingkungan yang berjalan menggunakan EB CLI

1. Gunakan perintah eb config untuk membuka file konfigurasi di editor teks default.

   ```
   ~/project$ eb config
   ```

2. Temukan namespace `aws:elasticbeanstalk:environment` di bagian pengaturan. Pastikan bahwa nilai `ServiceRole` tidak null dan cocok dengan nama peran layanan Anda.

```
  aws:elasticbeanstalk:environment:
    EnvironmentType: LoadBalanced
    ServiceRole: aws-elasticbeanstalk-service-role
```

3. Di bawah namespace `aws:elasticbeanstalk:healthreporting:system:`, ubah nilai `SystemType` ke **enhanced**.

```
  aws:elasticbeanstalk:healthreporting:system:
    SystemType: enhanced
```

4. Simpan file konfigurasi dan tutup teks editor.
5. EB CLI mulai pembaruan lingkungan untuk menerapkan perubahan konfigurasi Anda. Tunggu operasi selesai atau tekan Ctrl+C untuk keluar dengan aman.

```
~/project$ eb config
Printing Status:
INFO: Environment update is starting.
INFO: Health reporting type changed to ENHANCED.
INFO: Updating environment no-role-test's configuration settings.
```

Mengaktifkan pelaporan kondisi yang ditingkatkan menggunakan file konfigurasi

Anda dapat mengaktifkan pelaporan kondisi yang ditingkatkan dengan menyertakan file konfigurasi di paket sumber Anda. Contoh berikut menunjukkan file konfigurasi yang mengaktifkan pelaporan kondisi yang ditingkatkan dan memberikan layanan default dan profil instans untuk lingkungan:

Example .ebextensions/enhanced-health.config

```
option_settings:
  aws:elasticbeanstalk:healthreporting:system:
    SystemType: enhanced
  aws:autoscaling:launchconfiguration:
    IamInstanceProfile: aws-elasticbeanstalk-ec2-role
  aws:elasticbeanstalk:environment:
    ServiceRole: aws-elasticbeanstalk-service-role
```

Jika Anda membuat profil instans atau peran layanan Anda sendiri, ganti teks yang disorot dengan nama peran tersebut.

Pemantauan kondisi yang ditingkatkan dengan konsol manajemen lingkungan

Saat mengaktifkan pelaporan kesehatan yang disempurnakan AWS Elastic Beanstalk, Anda dapat memantau kesehatan lingkungan di konsol manajemen lingkungan.

Topik

- Gambaran umum lingkungan
- Halaman kondisi lingkungan
- Halaman pemantauan

Gambaran umum lingkungan

Gambaran umum lingkungan menampilkan status kondisi lingkungan dan daftar peristiwa yang memberikan informasi tentang perubahan terbaru dalam status kondisi.

Untuk melihat gambaran umum lingkungan

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.

Untuk informasi detail tentang kondisi lingkungan saat ini, buka halaman Kondisi dengan memilih Penyebab. Sebagai alternatif, di panel navigasi, pilih Kondisi.

Halaman kondisi lingkungan

Halaman Kesehatan menampilkan status kesehatan, metrik, dan penyebab lingkungan dan untuk setiap EC2 instans Amazon di lingkungan.

> **Note**
>
> Elastic Beanstalk menampilkan halaman Kondisi hanya jika Anda memiliki pemantauan kondisi yang ditingkatkan yang aktif untuk lingkungan.

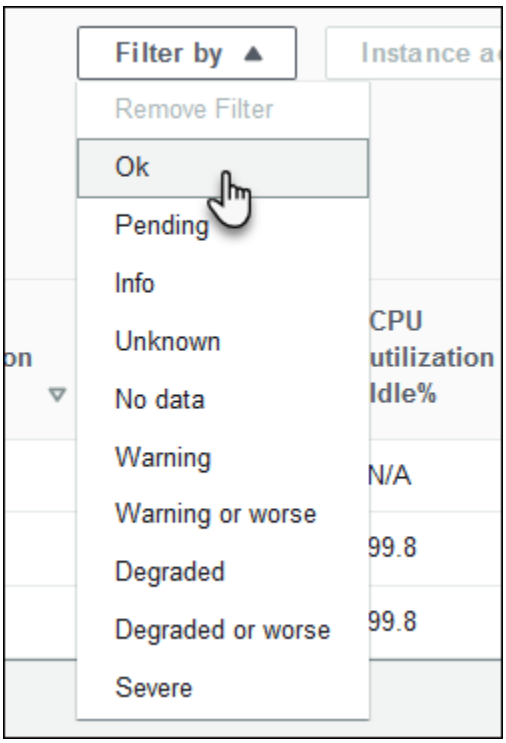
Citra berikut menunjukkan halaman Kondisi untuk lingkungan Linux.



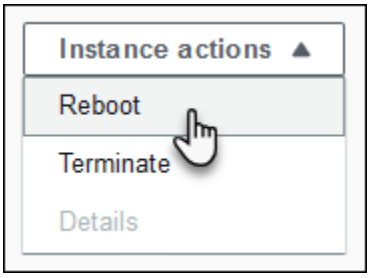
Citra berikut menunjukkan halaman Kondisi untuk lingkungan Windows. Perhatikan bahwa metrik CPU berbeda dari yang ada di lingkungan Linux.



Di bagian atas halaman Anda dapat melihat jumlah total instans lingkungan, serta jumlah instans per status. Untuk menampilkan hanya instans yang memiliki status tertentu, pilih Filter Berdasarkan, dan kemudian pilih status.



Untuk memulai ulang atau menghentikan instans yang tidak sehat, pilih Tindakan Instans, lalu pilih Mulai ulang atau Akhiri.



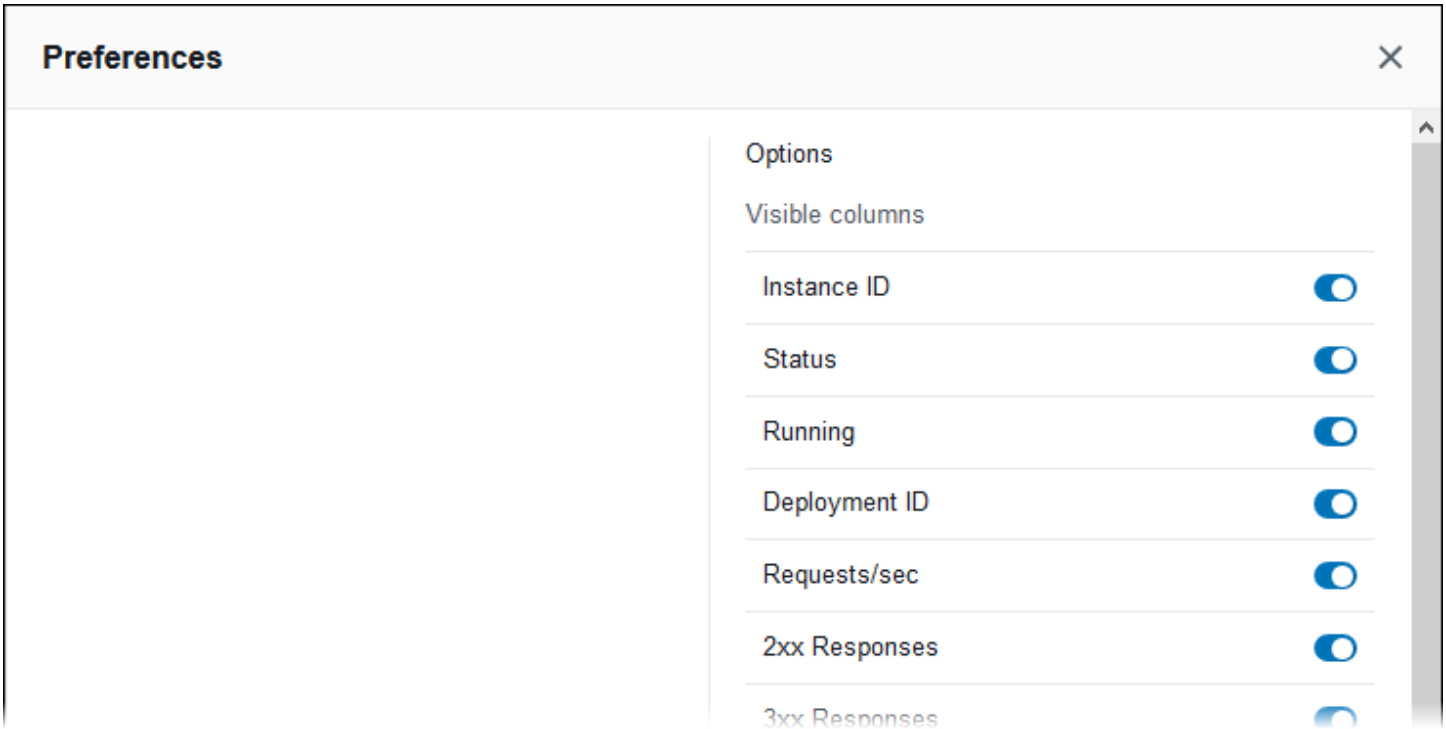
Elastic Beanstalk memperbarui halaman Kondisi setiap 10 detik. Ini melaporkan informasi tentang kondisi lingkungan dan instans.

Untuk setiap EC2 instans Amazon di lingkungan, halaman menampilkan ID dan status instans, jumlah waktu sejak instance diluncurkan, ID penerapan terbaru yang dijalankan pada instance, respons dan latensi permintaan yang disajikan instans, serta informasi pemuatan dan pemanfaatan CPU. Baris Secara keseluruhan menampilkan respon rata-rata dan informasi latensi untuk seluruh lingkungan.

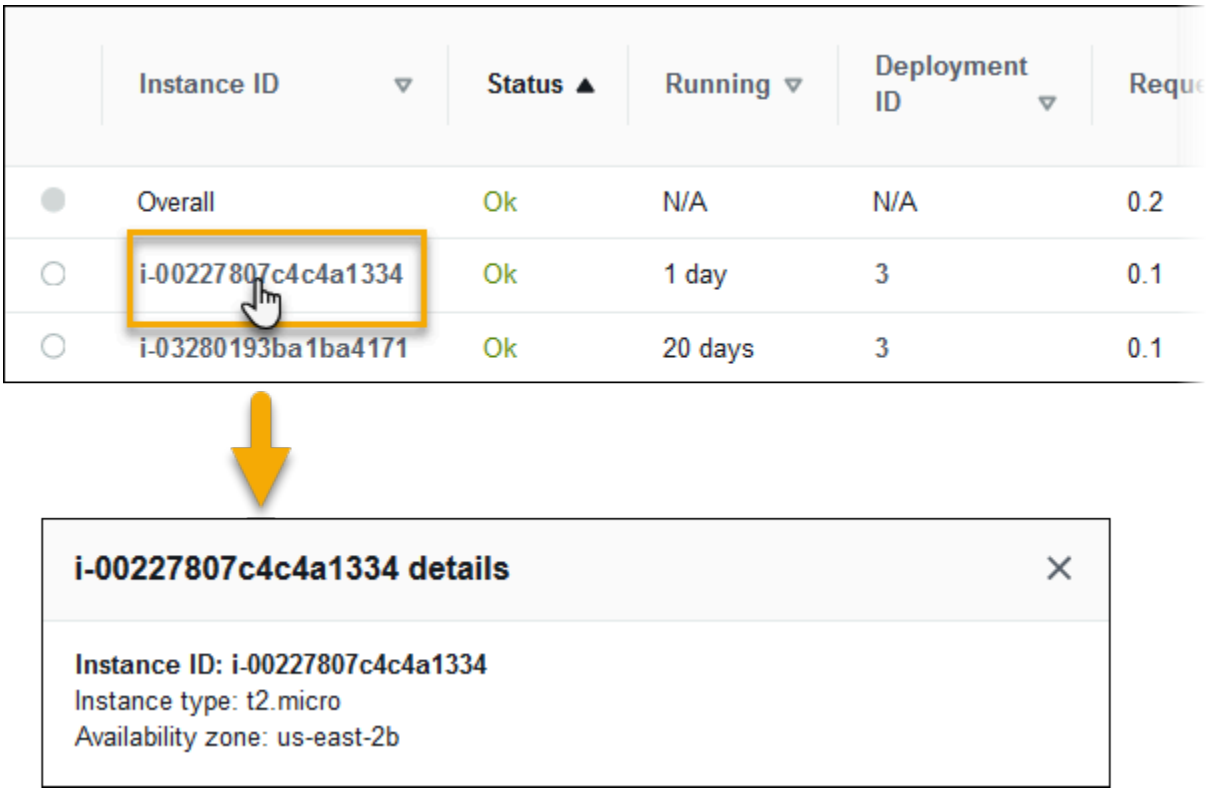
Halaman ini menampilkan banyak detail dalam tabel yang sangat lebar. Untuk menyembunyikan beberapa kolom, pilih



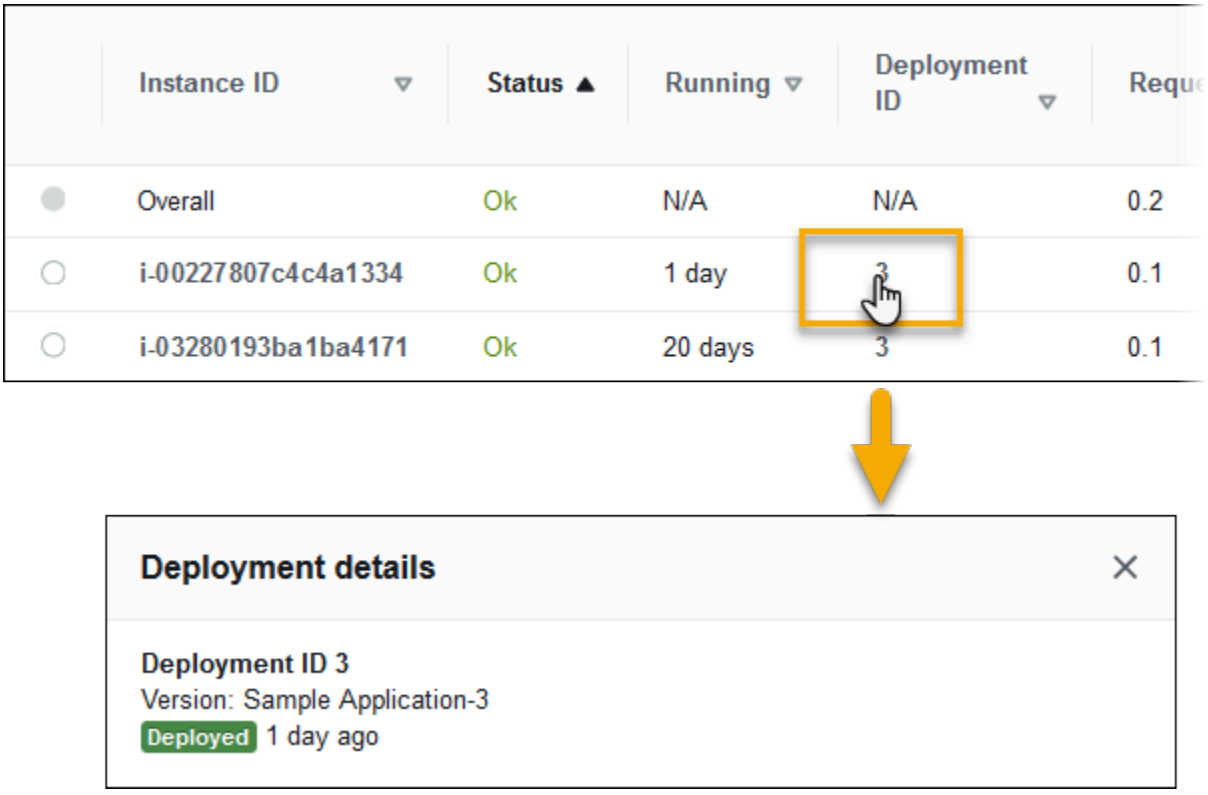
(Preferensi). Pilih atau hapus nama kolom, dan kemudian pilih Mengonfirmasi.



Pilih ID Instans dari setiap instans untuk melihat informasi selengkapnya tentang instans, termasuk Availability Zone dan tipe instans.



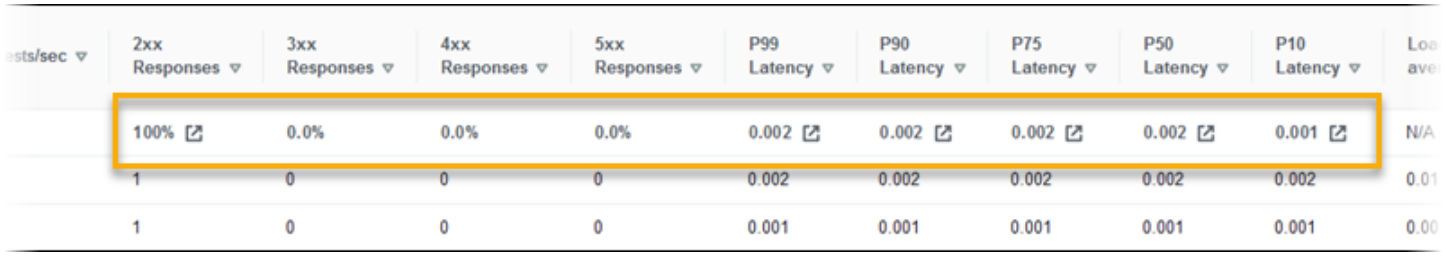
Pilih ID Deployment dari setiap instans untuk melihat informasi tentang deployment yang terakir ke instans.



Informasi deployment mencakup hal berikut:

- ID Deployment—Pengidentifikasi unik untuk deployment. Penerapan IDs dimulai dari 1 dan meningkat satu kali setiap kali Anda menerapkan versi aplikasi baru atau mengubah pengaturan konfigurasi yang memengaruhi perangkat lunak atau sistem operasi yang berjalan pada instance di lingkungan Anda.
- Versi—Label versi dari kode sumber aplikasi yang digunakan dalam deployment.
- Status—Status deployment, bisa `In Progress`, `Deployed`, atau `Failed`.
- Waktu— Untuk deployment dalam proses, waktu deployment dimulai. Untuk deployment yang telah selesai, waktu deployment berakhir.

Jika Anda mengaktifkan integrasi X-Ray di lingkungan Anda dan instrumen aplikasi Anda dengan AWS X-Ray SDK, halaman Health menambahkan link ke AWS X-Ray konsol di baris ikhtisar.



Pilih tautan untuk melihat jejak yang terkait dengan statistik yang disorot di AWS X-Ray konsol.

Halaman pemantauan

Halaman Pemantauan menampilkan statistik ringkasan dan grafik untuk CloudWatch metrik Amazon khusus yang dihasilkan oleh sistem pelaporan kesehatan yang disempurnakan. Lihat Memantau kesehatan lingkungan di konsol AWS manajemen untuk petunjuk tentang cara menambahkan grafik dan statistik ke halaman ini.

Warna dan status kondisi

Pelaporan kondisi yang ditingkatkan mewakili instans dan kondisi lingkungan secara keseluruhan dengan menggunakan empat warna, mirip dengan pelaporan kondisi dasar. Pelaporan kondisi yang ditingkatkan juga menyediakan tujuh status kondisi, yang merupakan deskriptor satu kata yang memberikan indikasi yang lebih baik tentang keadaan lingkungan Anda.

Status instans dan status lingkungan

Setiap kali Elastic Beanstalk menjalankan pemeriksaan kondisi di lingkungan Anda, pelaporan kondisi yang ditingkatkan memeriksa kondisi setiap instans di lingkungan Anda dengan menganalisis semua data tersedia. Jika pemeriksaan tingkat yang lebih rendah gagal, Elastic Beanstalk menurunkan kondisi instans.

Elastic Beanstalk menampilkan informasi kondisi untuk keseluruhan lingkungan (warna, status, dan sebab) di konsol manajemen lingkungan. Informasi ini juga tersedia di EB CLI. Status kondisi dan pesan penyebab untuk masing-masing instans diperbarui setiap 10 detik dan tersedia dari EB CLI saat Anda melihat status kondisi dengan eb health.

Elastic Beanstalk menggunakan perubahan pada kondisi instans untuk mengevaluasi kondisi lingkungan, tetapi tidak segera mengubah status kondisi lingkungan. Ketika sebuah instans gagal dalam pemeriksaan kondisi setidaknya tiga kali dalam periode satu menit, Elastic Beanstalk dapat menurunkan kondisi lingkungan. Tergantung pada jumlah instans di lingkungan dan masalah yang diidentifikasi, satu instans yang tidak sehat dapat menyebabkan Elastic Beanstalk menampilkan pesan informasi atau mengubah status kondisi lingkungan dari hijau (OK) ke kuning (Peringatan) atau merah (Berdegradasi atau Sangat parah).

OK (hijau)

Status ini ditampilkan ketika:

- Sebuah instans melewati pemeriksaan kondisi dan agen kondisi tidak melaporkan adanya masalah.
- Sebagian besar instans di lingkungan melewati pemeriksaan kondisi dan agen kondisi tidak melaporkan isu-isu utama.
- Sebuah instans melewati pemeriksaan kondisi dan menyelesaikan permintaan secara normal.

Contoh: Lingkungan Anda baru-baru ini di-deploy dan menerima permintaan secara normal. Lima persen dari permintaan mengembalikan 400 seri kesalahan. Deployment diselesaikan secara normal pada setiap instans.

Pesan (instans): Deployment aplikasi telah selesai 23 detik yang lalu dan menghabiskan 26 detik.

Peringatan (kuning)

Status ini ditampilkan ketika:

- Agen kondisi melaporkan jumlah kegagalan permintaan yang moderat atau masalah lain untuk instans atau lingkungan.
- Sebuah operasi sedang berlangsung pada sebuah instans dan menghabiskan waktu yang sangat lama.

Contoh: Satu instans di lingkungan memiliki status Pelik.

Pesan (lingkungan): Gangguan layanan 1 dari 5 instans.

Terdegradasi (merah)

Status ini ditampilkan ketika agen kondisi melaporkan tingginya jumlah kegagalan permintaan atau masalah lain untuk instans atau lingkungan.

Contoh: lingkungan sedang dalam proses menskalakan ke atas hingga 5 instans.

Pesan (lingkungan): 4 instans yang aktif berada di bawah ukuran minimum 5 grup Auto Scaling

Pelik (merah)

Status ini ditampilkan ketika agen kondisi melaporkan jumlah kegagalan permintaan yang sangat tinggi atau masalah lain untuk instans atau lingkungan.

Contoh: Elastic Beanstalk tidak dapat menghubungi penyeimbang beban untuk mendapatkan kondisi instans.

Pesan (lingkungan): kondisi ELB gagal atau tidak tersedia untuk semua instans. Tidak satu pun dari instans mengirim data. Tidak dapat mengambil peran “arn:aws:iam: :123456789012:role/”. aws-elasticbeanstalk-service-role Verifikasi bahwa peran ada dan dikonfigurasi dengan benar.

Pesan (Instans): Kondisi instans ELB belum tersedia selama 37 menit. Tidak ada data. Terakhir terlihat 37 menit yang lalu.

Info (Hijau)

Status ini ditampilkan ketika:

- Operasi sedang berlangsung pada sebuah instans.
- Operasi sedang berlangsung pada beberapa instans di lingkungan.

Contoh: Versi aplikasi baru sedang di-deploy untuk instans berjalan.

Pesan (lingkungan): Perintah mengeksekusi 3 dari 5 instans.

Pesan (instans): Melakukan deployment aplikasi (berjalan selama 3 detik).

Tertunda (abu-abu)

Status ini ditampilkan ketika operasi sedang berlangsung pada sebuah instans dalam waktu perintah habis.

Contoh: Anda baru-baru ini membuat lingkungan dan instans sedang bootstrap.

Pesan: Melakukan inisialisasi (berjalan selama 12 detik).

Tidak diketahui (abu-abu)

Status ini ditampilkan ketika Elastic Beanstalk dan agen kondisi melaporkan jumlah data yang tidak mencukupi pada instans.

Contoh: Tidak ada data yang diterima.

Ditangguhkan (abu-abu)

Status ini ditampilkan saat Elastic Beanstalk berhenti memantau kondisi lingkungan. Lingkungan mungkin tidak bekerja dengan benar. Beberapa keadaan kondisi yang pelik, jika bertahan lama, menyebabkan Elastic Beanstalk untuk mentransisikan lingkungan ke status Ditangguhkan.

Contoh: Elastic Beanstalk tidak dapat mengakses peran layanan lingkungan.

Contoh: Grup Auto Scaling yang dibuat Elastic Beanstalk untuk lingkungan telah dihapus.

Pesan: Kondisi lingkungan telah beralih dari OK menjadi Pelik. Tidak ada instans. Kapasitas yang diinginkan grup Auto Scaling diatur ke 1.

Metrik instans

Metrik instans memberikan informasi tentang kondisi instans di lingkungan Anda. Agen kondisi Elastic Beanstalk berjalan pada setiap instans. Ini mengumpulkan dan menyampaikan metrik tentang instans Elastic Beanstalk, yang menganalisis metrik untuk menentukan kondisi instans di lingkungan Anda.

Agen kondisi Elastic Beanstalk pada instans mengumpulkan metrik tentang instans dari server web dan sistem operasi. Untuk mendapatkan informasi server web pada platform berbasis Linux, Elastic Beanstalk membaca dan mengurai log server web. Pada platform Windows Server, Elastic Beanstalk menerima informasi ini langsung dari server web IIS. Server web menyediakan informasi tentang permintaan HTTP yang masuk: berapa banyak permintaan yang masuk, berapa banyak yang mengakibatkan kesalahan, dan berapa lama waktu yang dibutuhkan untuk menyelesaikannya. Sistem operasi menyediakan informasi snapshot tentang keadaan sumber daya instans: beban CPU dan distribusi waktu yang dihabiskan untuk setiap jenis proses.

Agen kondisi mengumpulkan metrik web server dan sistem operasi dan menyampaikannya ke Elastic Beanstalk setiap 10 detik. Elastic Beanstalk menganalisis data dan menggunakan hasil tersebut untuk memperbarui status kondisi untuk setiap instans dan lingkungan.

Topik

- Metrik server web
- Metrik sistem operasi
- Metrik server web menangkap IIS pada Windows server

Metrik server web

Pada platform berbasis Linux, agen kondisi Elastic Beanstalk membaca metrik server web dari log yang dihasilkan oleh kontainer web atau server yang memproses permintaan pada setiap instans di lingkungan Anda. Platform Elastic Beanstalk dikonfigurasi untuk menghasilkan dua log: satu dalam format yang dapat dibaca manusia dan satu dalam format yang dapat dibaca mesin. Agen kondisi menyampaikan log yang dapat dibaca mesin ke Elastic Beanstalk setiap 10 detik.

Untuk informasi lebih lanjut tentang format log yang digunakan oleh Elastic Beanstalk, lihat Format log kondisi yang ditingkatkan.

Pada platform Windows Server, Elastic Beanstalk menambahkan modul ke alur permintaan web server IIS dan menangkap metrik tentang waktu permintaan dan kode respon HTTP. Modul tersebut mengirimkan metrik ini ke agen kondisi pada instans menggunakan saluran komunikasi antar proses performa tinggi (IPC). Untuk detail implementasi, lihat Metrik server web menangkap IIS pada Windows server.

Metrik Web Server yang Dilaporkan

`RequestCount`

Jumlah permintaan yang ditangani oleh web server per detik selama 10 detik terakhir. Ditampilkan sebagai rata-rata `r/sec` (permintaan per detik) di EB CLI dan Halaman kondisi lingkungan.

`Status2xx`, `Status3xx`, `Status4xx`, `Status5xx`

Jumlah permintaan yang mengakibatkan setiap jenis kode status selama 10 detik terakhir. Misalnya, permintaan berhasil mengembalikan 200 OK, pengalihan mengembalikan 301, dan 404 dikembalikan jika URL yang dimasukkan tidak cocok dengan sumber daya dalam aplikasi.

EB CLI dan Halaman kondisi lingkungan menampilkan metrik ini sebagai jumlah permintaan mentah untuk instans, dan sebagai persentase permintaan keseluruhan untuk lingkungan.

`p99.9`, `p99`, `p95`, `p90`, `p85`, `p75`, `p50`, `p10`

Rata-rata latensi untuk yang paling lambat X persen permintaan selama 10 detik terakhir, di mana X adalah perbedaan antara nomor dan 100. Misalnya, `p99 1.403` menunjukkan 1% permintaan paling lambat selama 10 detik terakhir memiliki latensi rata-rata 1,403 detik.

Metrik sistem operasi

Agen kondisi Elastic Beanstalk melaporkan metrik sistem operasi berikut. Elastic Beanstalk menggunakan metrik ini untuk mengidentifikasi instans yang berada di bawah beban berat yang berkelanjutan. Metrik berbeda dengan sistem operasi.

Metrik sistem operasi yang dilaporkan—Linux

`Running`

Jumlah waktu yang telah berlalu sejak instans diluncurkan.

`Load 1`, `Load 5`

Rata-rata muatan dalam periode satu menit dan lima menit terakhir. Ditampilkan sebagai nilai desimal yang menunjukkan jumlah rata-rata proses yang berjalan selama waktu itu. Jika angka yang ditampilkan lebih tinggi dari jumlah v CPUs (utas) yang tersedia, maka sisanya adalah jumlah rata-rata proses yang menunggu.

Misalnya, jika tipe instans Anda memiliki empat vCPUs, dan bebannya 4,5, ada rata-rata 0,5 proses menunggu selama periode waktu tersebut, setara dengan satu proses yang menunggu 50 persen dari waktu.

`User %`, `Nice %`, `System %`, `Idle %`, `I/O Wait %`

Persentase waktu yang telah CPU habiskan di setiap status selama 10 detik terakhir.

Metrik sistem operasi yang dilaporkan—Windows

`Running`

Jumlah waktu yang telah berlalu sejak instans diluncurkan.

`% User Time`, `% Privileged Time`, `% Idle Time`

Persentase waktu yang telah CPU habiskan di setiap status selama 10 detik terakhir.

Metrik server web menangkap IIS pada Windows server

Pada platform Windows Server, Elastic Beanstalk menambahkan modul ke alur permintaan web server IIS dan menangkap metrik tentang waktu permintaan dan kode respon HTTP. Modul tersebut

mengirimkan metrik ini ke agen kondisi pada instans menggunakan saluran komunikasi antar proses performa tinggi (IPC). Agen kondisi mengumpulkan metrik ini, menggabungkannya dengan metrik sistem operasi, dan mengirimkannya ke layanan Elastic Beanstalk.

Detail implementasi

Untuk menangkap metrik dari IIS, Elastic Beanstalk mengimplementasikan `IHttpModule` yang terkelola, dan berlangganan ke peristiwa `BeginRequest` dan `EndRequest`. Hal ini memungkinkan modul untuk melaporkan latensi permintaan HTTP dan kode respon untuk semua permintaan web yang ditangani oleh IIS. Untuk menambahkan modul ke alur permintaan IIS, Elastic Beanstalk mendaftarkan modul di bagian `<modules>` dari file konfigurasi IIS, `%windir%\System32\inetsrv\config\applicationHost.config`.

Modul Elastic Beanstalk di IIS mengirimkan metrik permintaan web yang telah ditangkap kepada agen kondisi pada instans, yang merupakan layanan Windows bernama `HealthD`. Untuk mengirim data ini, modul menggunakan `NetNamedPipeBinding`, yang menyediakan pengikatan yang aman dan andal yang dioptimalkan untuk komunikasi pada mesin.

Mengonfigurasi aturan kondisi yang ditingkatkan untuk lingkungan

AWS Elastic Beanstalk Pelaporan kesehatan yang ditingkatkan bergantung pada seperangkat aturan untuk menentukan kesehatan lingkungan Anda. Beberapa aturan ini mungkin tidak sesuai untuk aplikasi tertentu Anda. Berikut adalah beberapa contoh umum:

- Anda menggunakan alat uji sisi klien. Dalam kasus ini, kesalahan klien (4xx) HTTP yang sering terjadi diharapkan.
- Anda menggunakan AWS WAF bersama dengan Application Load Balancer lingkungan Anda untuk memblokir lalu lintas masuk yang tidak diinginkan. Dalam kasus ini, Application Load Balancer mengembalikan HTTP 403 untuk setiap pesan masuk yang ditolak.

Secara default, Elastic Beanstalk mencakup semua kesalahan HTTP 4xx aplikasi saat menentukan kondisi lingkungan. Ini mengubah status kondisi lingkungan Anda dari OK menjadi Peringatan, Berdegradasi, atau Pelik, tergantung pada tingkat kesalahan. Untuk menangani kasus dengan benar seperti contoh yang kami sebutkan, Elastic Beanstalk memungkinkan Anda untuk mengonfigurasi beberapa aturan kondisi yang ditingkatkan. Anda dapat memilih untuk mengabaikan kesalahan HTTP 4xx aplikasi pada instans lingkungan, atau mengabaikan kesalahan HTTP 4xx yang dikembalikan oleh penyeimbang beban lingkungan. Topik ini menjelaskan cara membuat perubahan konfigurasi ini.

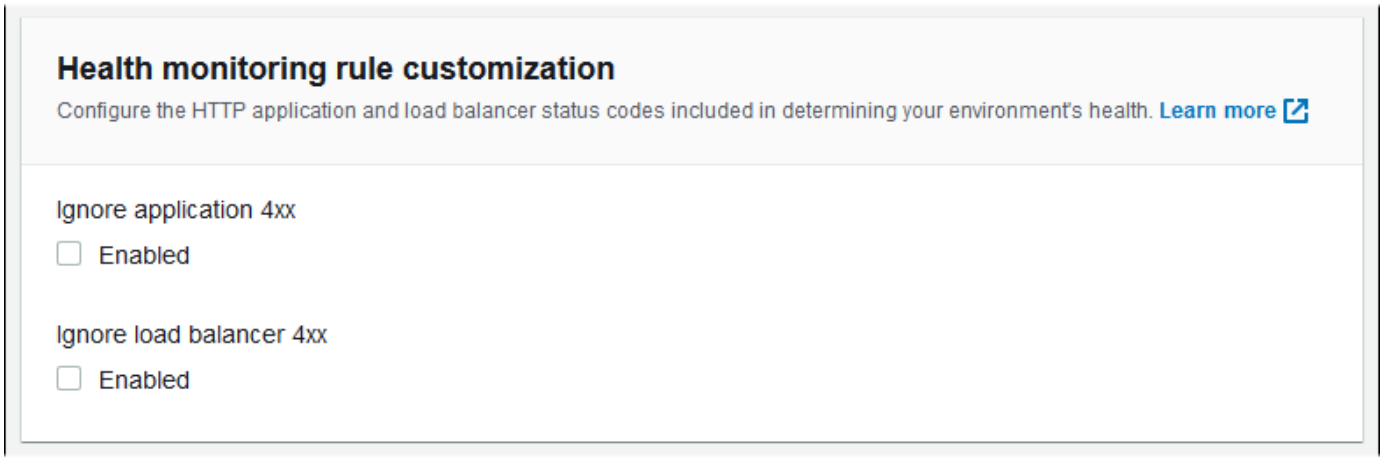
> **Note**
>
> Saat ini, ini adalah satu-satunya penyesuaian aturan kondisi yang ditingkatkan. Anda tidak dapat mengonfigurasi kondisi yang ditingkatkan untuk mengabaikan kesalahan HTTP lain selain 4xx.

Mengonfigurasi aturan kondisi yang ditingkatkan menggunakan konsol Elastic Beanstalk

Anda dapat menggunakan konsol Elastic Beanstalk untuk mengonfigurasi aturan kondisi yang ditingkatkan di lingkungan Anda.

Untuk mengonfigurasi pemeriksaan kode status HTTP 4xx menggunakan konsol Elastic Beanstalk

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Di kategori konfigurasi Pemantauan, pilih Edit.
5. Di bawah Penyesuaian aturan pemantauan kondisi, aktifkan atau nonaktifkan pilihan Abaikan yang diinginkan.



6. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Mengonfigurasi aturan kondisi yang ditingkatkan menggunakan EB CLI

Anda dapat menggunakan EB CLI untuk mengonfigurasi aturan kondisi yang ditingkatkan menyimpan konfigurasi lingkungan Anda secara lokal, menambahkan entri yang mengonfigurasi aturan kondisi yang ditingkatkan, dan kemudian mengunggah konfigurasi ke Elastic Beanstalk. Anda dapat menerapkan konfigurasi yang disimpan untuk lingkungan selama atau setelah pembuatan.

Untuk mengonfigurasi pemeriksaan kode status HTTP 4xx menggunakan EB CLI dan konfigurasi yang disimpan

1. Inisialisasi folder proyek Anda dengan eb init.
2. Buat lingkungan dengan menjalankan perintah eb create.
3. Simpan templat konfigurasi secara lokal dengan menjalankan perintah eb config save. Contoh berikut menggunakan pilihan `--cfg` untuk menentukan nama konfigurasi.

   ```
   $ eb config save --cfg 01-base-state
   Configuration saved at: ~/project/.elasticbeanstalk/saved_configs/01-base-
   state.cfg.yml
   ```

4. Buka file konfigurasi yang disimpan di editor teks.
5. Di bawah `OptionSettings` > `aws:elasticbeanstalk:healthreporting:system:`, tambahkan kunci `ConfigDocument` untuk mencantumkan setiap aturan kondisi yang ditingkatkan untuk mengonfigurasi. `ConfigDocument` berikut menonaktifkan pengecekan kode status HTTP 4xx aplikasi, sambil menjaga pemeriksaan kode HTTP 4xx penyeimbang beban tetap aktif.

   ```
   OptionSettings:
     ...
     aws:elasticbeanstalk:healthreporting:system:
       ConfigDocument:
         Rules:
           Environment:
             Application:
               ApplicationRequests4xx:
                 Enabled: false
             ELB:
               ELBRequests4xx:
                 Enabled: true
         Version: 1
       SystemType: enhanced
   ```

```
...
```

> **Note**
>
> Anda bisa menggabungkan `Rules` dan `CloudWatchMetrics` dalam pengaturan pilihan `ConfigDocument` yang sama. `CloudWatchMetrics` dijelaskan di Menerbitkan metrik CloudWatch kustom Amazon untuk suatu lingkungan.
> Jika sebelumnya Anda mengaktifkan `CloudWatchMetrics`, file konfigurasi yang Anda ambil menggunakan perintah eb config save telah memiliki kunci `ConfigDocument` dengan bagian `CloudWatchMetrics`. Jangan menghapusnya—tambahkan bagian `Rules` ke nilai pilihan `ConfigDocument` yang sama.

6. Simpan file konfigurasi dan tutup teks editor. Dalam contoh ini, berkas konfigurasi yang diperbarui disimpan dengan nama (`02-cloudwatch-enabled.cfg.yml`) yang berbeda dari file konfigurasi yang diunduh. Hal ini membuat konfigurasi yang tersimpan terpisah ketika file diunggah. Anda dapat menggunakan nama yang sama dengan file yang diunduh untuk menimpa konfigurasi yang ada tanpa membuat yang baru.
7. Gunakan perintah eb config put untuk mengunggah file konfigurasi yang diperbarui ke Elastic Beanstalk.

   ```
   $ eb config put 02-cloudwatch-enabled
   ```

   Ketika menggunakan perintah eb config `get` dan `put` dengan konfigurasi yang tersimpan, jangan sertakan ekstensi nama file.
8. Terapkan konfigurasi yang disimpan ke lingkungan Anda yang sedang berjalan.

   ```
   $ eb config --cfg 02-cloudwatch-enabled
   ```

   Pilihan `--cfg` menentukan file konfigurasi bernama yang diterapkan ke lingkungan. Anda dapat menyimpan file konfigurasi secara lokal atau di Elastic Beanstalk. Jika file konfigurasi dengan nama tertentu ada di kedua lokasi tersebut, EB CLI akan menggunakan file lokal.

Mengonfigurasi aturan kondisi yang ditingkatkan menggunakan dokumen config

Dokument konfigurasi (config) untuk aturan kondisi yang ditingkatkan adalah dokumen JSON yang berisi daftar aturan untuk mengonfigurasi.

Contoh berikut menunjukkan dokumen config yang menonaktifkan pemeriksaan kode status aplikasi HTTP 4xx dan mengaktifkan pengecekan kode status penyeimbang beban HTTP 4xx.

```
{
  "Rules": {
    "Environment": {
      "Application": {
        "ApplicationRequests4xx": {
          "Enabled": false
        }
      },
      "ELB": {
        "ELBRequests4xx": {
          "Enabled": true
        }
      }
    }
  },
  "Version": 1
}
```

Untuk AWS CLI, Anda meneruskan dokumen sebagai nilai untuk `Value` kunci dalam argumen pengaturan opsi, yang itu sendiri adalah objek JSON. Dalam hal ini, Anda harus keluar dari tanda petik di dokumen yang disematkan. Perintah berikut memeriksa apakah pengaturan konfigurasi valid.

```
$ aws elasticbeanstalk validate-configuration-settings --application-name my-app --
environment-name my-env --option-settings '[
    {
        "Namespace": "aws:elasticbeanstalk:healthreporting:system",
        "OptionName": "ConfigDocument",
        "Value": "{\"Rules\": { \"Environment\": { \"Application\":
 { \"ApplicationRequests4xx\": { \"Enabled\": false } }, \"ELB\": { \"ELBRequests4xx\":
 {\"Enabled\": true } } } }, \"Version\": 1 }"
    }
]'
```

Untuk file konfigurasi `.ebextensions` di YAML, Anda dapat memberikan dokumen JSON apa adanya.

```
  option_settings:
    - namespace: aws:elasticbeanstalk:healthreporting:system
```

```
      option_name: ConfigDocument
      value: {
  "Rules": {
    "Environment": {
      "Application": {
        "ApplicationRequests4xx": {
          "Enabled": false
        }
      },
      "ELB": {
        "ELBRequests4xx": {
          "Enabled": true
        }
      }
    }
  },
  "Version": 1
}
```

Menerbitkan metrik CloudWatch kustom Amazon untuk suatu lingkungan

Anda dapat mempublikasikan data yang dikumpulkan oleh pelaporan kesehatan yang AWS Elastic Beanstalk disempurnakan ke Amazon CloudWatch sebagai metrik khusus. Menerbitkan metrik untuk CloudWatch memungkinkan Anda memantau perubahan kinerja aplikasi dari waktu ke waktu dan mengidentifikasi potensi masalah dengan melacak cara penggunaan sumber daya dan meminta skala latensi dengan beban.

Dengan menerbitkan metrik ke CloudWatch, Anda juga membuatnya tersedia untuk digunakan dengan grafik pemantauan dan alarm. Satu metrik gratis EnvironmentHealth, diaktifkan secara otomatis saat Anda menggunakan pelaporan kesehatan yang disempurnakan. Metrik khusus selain EnvironmentHealthdikenakan biaya standarCloudWatch .

Untuk mempublikasikan metrik CloudWatch khusus untuk suatu lingkungan, Anda harus terlebih dahulu mengaktifkan pelaporan kesehatan yang ditingkatkan tentang lingkungan. Lihat Mengaktifkan pelaporan kondisi yang ditingkatkan Elastic Beanstalk untuk instruksi.

Topik

- Metrik pelaporan kondisi yang ditingkatkan
- Mengkonfigurasi CloudWatch metrik menggunakan konsol Elastic Beanstalk
- Mengkonfigurasi metrik CloudWatch kustom menggunakan EB CLI

- Menyediakan dokumen konfigurasi metrik khusus

Metrik pelaporan kondisi yang ditingkatkan

Saat Anda mengaktifkan pelaporan kesehatan yang disempurnakan di lingkungan Anda, sistem pelaporan kesehatan yang disempurnakan secara otomatis menerbitkan satu metrik CloudWatch khusus. EnvironmentHealth Untuk mempublikasikan metrik tambahan CloudWatch, konfigurasikan lingkungan Anda dengan metrik tersebut dengan menggunakan konsol Elastic Beanstalk, EB CLI, atau .ebextensions.

Anda dapat mempublikasikan metrik kesehatan yang ditingkatkan berikut dari lingkungan Anda ke CloudWatch.

Metrik yang tersedia—semua platform

`EnvironmentHealth`

Lingkungan saja. Ini adalah satu-satunya CloudWatch metrik yang diterbitkan oleh sistem pelaporan kesehatan yang disempurnakan, kecuali jika Anda mengonfigurasi metrik tambahan. Kondisi lingkungan diwakili oleh salah satu dari tujuh status. Di CloudWatch konsol, status ini dipetakan ke nilai berikut:

- 0 – OK
- 1 – Info
- 5 – Tidak diketahui
- 10 – Tidak ada data
- 15 – Peringatan
- 20 – Berdegradasi
- 25 – Sangat Parah

`InstancesSevere`, `InstancesDegraded`, `InstancesWarning`, `InstancesInfo`, `InstancesOk`, `InstancesPending`, `InstancesUnknown`, `InstancesNoData`

Lingkungan saja. Metrik ini menunjukkan jumlah instans di lingkungan dengan setiap status kondisi. `InstancesNoData` menunjukkan jumlah instans yang tidak ada data yang diterima.

`ApplicationRequestsTotal`, `ApplicationRequests5xx`, `ApplicationRequests4xx`, `ApplicationRequests3xx`, `ApplicationRequests2xx`

Instance dan lingkungan. Menunjukkan jumlah total permintaan yang diselesaikan oleh instans atau lingkungan, dan jumlah permintaan yang dilengkapi dengan setiap kategori kode status.

`ApplicationLatencyP10`, `ApplicationLatencyP50`, `ApplicationLatencyP75`, `ApplicationLatencyP85`, `ApplicationLatencyP90`, `ApplicationLatencyP95`, `ApplicationLatencyP99`, `ApplicationLatencyP99.9`

Instance dan lingkungan. Menunjukkan jumlah waktu rata-rata, dalam detik, yang diperlukan untuk melengkapi x persen yang tercepat dari permintaan.

`InstanceHealth`

Instance saja. Menunjukkan status kondisi instans saat ini. Kondisi instans diwakili oleh salah satu dari tujuh status. Di CloudWatch konsol, status ini dipetakan ke nilai berikut:

- 0 – OK
- 1 – Info
- 5 – Tidak diketahui
- 10 – Tidak ada data
- 15 – Peringatan
- 20 – Berdegradasi
- 25 – Sangat Parah

Metrik yang tersedia—Linux

`CPUIrq`, `CPUIdle`, `CPUUser`, `CPUSystem`, `CPUSoftirq`, `CPUIowait`, `CPUNice`

Instance saja. Menunjukkan persentase waktu yang CPU telah habiskan di setiap status selama satu menit terakhir.

`LoadAverage1min`

Instance saja. Rata-rata beban CPU instans selama satu menit terakhir.

`RootFilesystemUtil`

Instance saja. Menunjukkan persentase ruang disk yang digunakan.

Metrik yang tersedia—Windows

`CPUIdle`, `CPUUser`, `CPUPrivileged`

Hanya instans. Menunjukkan persentase waktu yang CPU telah habiskan di setiap status selama satu menit terakhir.

Mengkonfigurasi CloudWatch metrik menggunakan konsol Elastic Beanstalk

Anda dapat menggunakan konsol Elastic Beanstalk untuk mengonfigurasi lingkungan Anda guna mempublikasikan CloudWatch metrik pelaporan kesehatan yang disempurnakan dan membuatnya tersedia untuk digunakan dengan grafik pemantauan dan alarm.

Untuk mengonfigurasi metrik CloudWatch khusus di konsol Elastic Beanstalk

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Pada kategori konfigurasi Pemantauan, pilih Edit.
5. Di bawah Health reporting, pilih metrik instans dan lingkungan untuk dipublikasikan. CloudWatch Untuk memilih beberapa metrik, tekan tombol Ctrl saat memilih.
6. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Mengaktifkan metrik CloudWatch kustom menambahkannya ke daftar metrik yang tersedia di halaman Pemantauan.

Mengkonfigurasi metrik CloudWatch kustom menggunakan EB CLI

Anda dapat menggunakan EB CLI untuk mengonfigurasi metrik khusus dengan menyimpan konfigurasi lingkungan Anda secara lokal, menambahkan entri yang menentukan metrik untuk diterbitkan, dan kemudian mengunggah konfigurasi ke Elastic Beanstalk. Anda dapat menerapkan konfigurasi yang disimpan untuk lingkungan selama atau setelah pembuatan.

Untuk mengonfigurasi metrik CloudWatch khusus dengan EB CLI dan konfigurasi yang disimpan

1. Inisialisasi folder proyek Anda dengan eb init.

2. Buat lingkungan dengan menjalankan perintah eb create.
3. Simpan templat konfigurasi secara lokal dengan menjalankan perintah eb config save. Contoh berikut menggunakan pilihan `--cfg` untuk menentukan nama konfigurasi.

```
$ eb config save --cfg 01-base-state
Configuration saved at: ~/project/.elasticbeanstalk/saved_configs/01-base-
state.cfg.yml
```

4. Buka file konfigurasi yang disimpan di editor teks.
5. Di `OptionSettings` bawah`>aws:elasticbeanstalk:healthreporting:system:`, tambahkan `ConfigDocument` kunci untuk mengaktifkan setiap CloudWatch metrik yang Anda inginkan. Misalnya, `ConfigDocument` berikut menerbitkan metrik `ApplicationRequests5xx` dan `ApplicationRequests4xx` di tingkat lingkungan, dan metrik `ApplicationRequestsTotal` di tingkat instans.

```
OptionSettings:
  ...
  aws:elasticbeanstalk:healthreporting:system:
    ConfigDocument:
      CloudWatchMetrics:
        Environment:
          ApplicationRequests5xx: 60
          ApplicationRequests4xx: 60
        Instance:
          ApplicationRequestsTotal: 60
      Version: 1
    SystemType: enhanced
...
```

Dalam contoh, 60 menunjukkan jumlah detik antara pengukuran. Saat ini, ini adalah satu-satunya nilai yang didukung.

> **Note**
>
> Anda bisa menggabungkan `CloudWatchMetrics` dan `Rules` dalam pengaturan pilihan `ConfigDocument` yang sama.`Rules` dijelaskan di Mengonfigurasi aturan kondisi yang ditingkatkan untuk lingkungan.
> Jika sebelumnya Anda menggunakan `Rules` untuk mengonfigurasi aturan kondisi yang ditingkatkan, kemudian file konfigurasi yang Anda ambil menggunakan perintah

eb config save telah memiliki kunci `ConfigDocument` dengan bagian `Rules`. Jangan menghapusnya—tambahkan bagian `CloudWatchMetrics` ke nilai pilihan `ConfigDocument` yang sama.

6. Simpan file konfigurasi dan tutup teks editor. Dalam contoh ini, file konfigurasi yang diperbarui disimpan dengan nama (`02-cloudwatch-enabled.cfg.yml`) yang berbeda dari file konfigurasi yang diunduh. Hal ini membuat konfigurasi yang tersimpan terpisah ketika file diunggah. Anda dapat menggunakan nama yang sama dengan file yang diunduh untuk menimpa konfigurasi yang ada tanpa membuat yang baru.
7. Gunakan perintah eb config put untuk mengunggah file konfigurasi yang diperbarui ke Elastic Beanstalk.

   ```
   $ eb config put 02-cloudwatch-enabled
   ```

   Ketika menggunakan perintah eb config `get` dan `put` dengan konfigurasi tersimpan, jangan sertakan ekstensi file.
8. Terapkan konfigurasi yang disimpan ke lingkungan Anda yang sedang berjalan.

   ```
   $ eb config --cfg 02-cloudwatch-enabled
   ```

   Pilihan `--cfg` menentukan file konfigurasi bernama yang diterapkan ke lingkungan. Anda dapat menyimpan file konfigurasi secara lokal atau di Elastic Beanstalk. Jika file konfigurasi dengan nama tertentu ada di kedua lokasi tersebut, EB CLI akan menggunakan file lokal.

Menyediakan dokumen konfigurasi metrik khusus

Dokumen konfigurasi (konfigurasi) untuk metrik CloudWatch kustom Amazon adalah dokumen JSON yang mencantumkan metrik untuk dipublikasikan di tingkat lingkungan dan instans. Contoh berikut menunjukkan dokumen config yang memungkinkan semua metrik khusus tersedia di Linux.

```
{
  "CloudWatchMetrics": {
    "Environment": {
      "ApplicationLatencyP99.9": 60,
      "InstancesSevere": 60,
      "ApplicationLatencyP90": 60,
      "ApplicationLatencyP99": 60,
```

```
      "ApplicationLatencyP95": 60,
      "InstancesUnknown": 60,
      "ApplicationLatencyP85": 60,
      "InstancesInfo": 60,
      "ApplicationRequests2xx": 60,
      "InstancesDegraded": 60,
      "InstancesWarning": 60,
      "ApplicationLatencyP50": 60,
      "ApplicationRequestsTotal": 60,
      "InstancesNoData": 60,
      "InstancesPending": 60,
      "ApplicationLatencyP10": 60,
      "ApplicationRequests5xx": 60,
      "ApplicationLatencyP75": 60,
      "InstancesOk": 60,
      "ApplicationRequests3xx": 60,
      "ApplicationRequests4xx": 60
    },
    "Instance": {
      "ApplicationLatencyP99.9": 60,
      "ApplicationLatencyP90": 60,
      "ApplicationLatencyP99": 60,
      "ApplicationLatencyP95": 60,
      "ApplicationLatencyP85": 60,
      "CPUUser": 60,
      "ApplicationRequests2xx": 60,
      "CPUIdle": 60,
      "ApplicationLatencyP50": 60,
      "ApplicationRequestsTotal": 60,
      "RootFilesystemUtil": 60,
      "LoadAverage1min": 60,
      "CPUIrq": 60,
      "CPUNice": 60,
      "CPUIowait": 60,
      "ApplicationLatencyP10": 60,
      "LoadAverage5min": 60,
      "ApplicationRequests5xx": 60,
      "ApplicationLatencyP75": 60,
      "CPUSystem": 60,
      "ApplicationRequests3xx": 60,
      "ApplicationRequests4xx": 60,
      "InstanceHealth": 60,
      "CPUSoftirq": 60
    }
```

```
  },
  "Version": 1
}
```

Untuk AWS CLI, Anda meneruskan dokumen sebagai nilai untuk `Value` kunci dalam argumen pengaturan opsi, yang itu sendiri adalah objek JSON. Dalam hal ini, Anda harus keluar dari tanda petik di dokumen yang disematkan.

```
$ aws elasticbeanstalk validate-configuration-settings --application-name my-app --
environment-name my-env --option-settings '[
    {
        "Namespace": "aws:elasticbeanstalk:healthreporting:system",
        "OptionName": "ConfigDocument",
        "Value": "{\"CloudWatchMetrics\": {\"Environment\":
 {\"ApplicationLatencyP99.9\": 60,\"InstancesSevere\": 60,\"ApplicationLatencyP90\":
 60,\"ApplicationLatencyP99\": 60,\"ApplicationLatencyP95\": 60,\"InstancesUnknown
\": 60,\"ApplicationLatencyP85\": 60,\"InstancesInfo\": 60,\"ApplicationRequests2xx
\": 60,\"InstancesDegraded\": 60,\"InstancesWarning\": 60,\"ApplicationLatencyP50\":
 60,\"ApplicationRequestsTotal\": 60,\"InstancesNoData\": 60,\"InstancesPending
\": 60,\"ApplicationLatencyP10\": 60,\"ApplicationRequests5xx\": 60,
\"ApplicationLatencyP75\": 60,\"InstancesOk\": 60,\"ApplicationRequests3xx\": 60,
\"ApplicationRequests4xx\": 60},\"Instance\": {\"ApplicationLatencyP99.9\": 60,
\"ApplicationLatencyP90\": 60,\"ApplicationLatencyP99\": 60,\"ApplicationLatencyP95\":
 60,\"ApplicationLatencyP85\": 60,\"CPUUser\": 60,\"ApplicationRequests2xx\":
 60,\"CPUIdle\": 60,\"ApplicationLatencyP50\": 60,\"ApplicationRequestsTotal\":
 60,\"RootFilesystemUtil\": 60,\"LoadAverage1min\": 60,\"CPUIrq\": 60,\"CPUNice
\": 60,\"CPUIowait\": 60,\"ApplicationLatencyP10\": 60,\"LoadAverage5min\": 60,
\"ApplicationRequests5xx\": 60,\"ApplicationLatencyP75\": 60,\"CPUSystem\": 60,
\"ApplicationRequests3xx\": 60,\"ApplicationRequests4xx\": 60,\"InstanceHealth\": 60,
\"CPUSoftirq\": 60}},\"Version\": 1}"
    }
]'
```

Untuk file konfigurasi `.ebextensions` di YAML, Anda dapat memberikan dokumen JSON apa adanya.

```
  option_settings:
    - namespace: aws:elasticbeanstalk:healthreporting:system
      option_name: ConfigDocument
      value: {
  "CloudWatchMetrics": {
    "Environment": {
```

```
      "ApplicationLatencyP99.9": 60,
      "InstancesSevere": 60,
      "ApplicationLatencyP90": 60,
      "ApplicationLatencyP99": 60,
      "ApplicationLatencyP95": 60,
      "InstancesUnknown": 60,
      "ApplicationLatencyP85": 60,
      "InstancesInfo": 60,
      "ApplicationRequests2xx": 60,
      "InstancesDegraded": 60,
      "InstancesWarning": 60,
      "ApplicationLatencyP50": 60,
      "ApplicationRequestsTotal": 60,
      "InstancesNoData": 60,
      "InstancesPending": 60,
      "ApplicationLatencyP10": 60,
      "ApplicationRequests5xx": 60,
      "ApplicationLatencyP75": 60,
      "InstancesOk": 60,
      "ApplicationRequests3xx": 60,
      "ApplicationRequests4xx": 60
    },
    "Instance": {
      "ApplicationLatencyP99.9": 60,
      "ApplicationLatencyP90": 60,
      "ApplicationLatencyP99": 60,
      "ApplicationLatencyP95": 60,
      "ApplicationLatencyP85": 60,
      "CPUUser": 60,
      "ApplicationRequests2xx": 60,
      "CPUIdle": 60,
      "ApplicationLatencyP50": 60,
      "ApplicationRequestsTotal": 60,
      "RootFilesystemUtil": 60,
      "LoadAverage1min": 60,
      "CPUIrq": 60,
      "CPUNice": 60,
      "CPUIowait": 60,
      "ApplicationLatencyP10": 60,
      "LoadAverage5min": 60,
      "ApplicationRequests5xx": 60,
      "ApplicationLatencyP75": 60,
      "CPUSystem": 60,
      "ApplicationRequests3xx": 60,
```

```
      "ApplicationRequests4xx": 60,
      "InstanceHealth": 60,
      "CPUSoftirq": 60
    }
  },
  "Version": 1
}
```

Menggunakan pelaporan kondisi yang ditingkatkan dengan API Elastic Beanstalk

Karena pelaporan kesehatan yang AWS Elastic Beanstalk disempurnakan memiliki persyaratan tumpukan peran dan solusi, Anda harus memperbarui skrip dan kode yang Anda gunakan sebelum rilis pelaporan kesehatan yang disempurnakan sebelum Anda dapat menggunakannya. Untuk menjaga kompatibilitas mundur, pelaporan kondisi yang ditingkatkan tidak diaktifkan secara default ketika Anda membuat lingkungan menggunakan API Elastic Beanstalk.

Anda mengonfigurasi pelaporan kesehatan yang disempurnakan dengan menyetel peran layanan, profil instans, dan opsi CloudWatch konfigurasi Amazon untuk lingkungan Anda. Anda dapat melakukan ini dengan tiga cara: dengan pengaturan pilihan konfigurasi di folder `.ebextensions`, dengan konfigurasi yang tersimpan, atau dengan mengonfigurasinya secara langsung di parameter panggilan `create-environment` `option-settings`.

Untuk menggunakan API, SDKs, atau antarmuka baris AWS perintah (CLI) untuk menciptakan lingkungan yang mendukung peningkatan kesehatan, Anda harus:

- Buat peran layanan dan profil instans dengan izin yang tepat
- Buat lingkungan baru dengan versi platform terbaru
- Atur jenis sistem kondisi, profil instans, dan peran layanan pilihan konfigurasi

Gunakan pilihan konfigurasi berikut di namespace `aws:elasticbeanstalk:healthreporting:system`, `aws:autoscaling:launchconfiguration`, dan `aws:elasticbeanstalk:environment` untuk mengonfigurasi lingkungan Anda untuk pelaporan kondisi yang ditingkatkan.

Pilihan konfigurasi kondisi yang ditingkatkan

SystemType

Namespace: `aws:elasticbeanstalk:healthreporting:system`

Untuk mengaktifkan pelaporan kondisi yang ditingkatkan, atur ke **enhanced**.

IamInstanceProfile

Namespace: `aws:autoscaling:launchconfiguration`

Atur ke nama profil instans yang dikonfigurasi untuk digunakan dengan Elastic Beanstalk.

ServiceRole

Namespace: `aws:elasticbeanstalk:environment`

Tetapkan ke nama peran layanan yang dikonfigurasi untuk digunakan dengan Elastic Beanstalk.

ConfigDocument(opsional)

Namespace: `aws:elasticbeanstalk:healthreporting:system`

Dokumen JSON yang mendefinisikan metrik dan instance dan lingkungan untuk dipublikasikan. CloudWatch Sebagai contoh:

```
{
  "CloudWatchMetrics":
    {
    "Environment":
      {
      "ApplicationLatencyP99.9":60,
      "InstancesSevere":60
      }
    "Instance":
      {
      "ApplicationLatencyP85":60,
      "CPUUser": 60
      }
    }
  "Version":1
}
```

> **Note**
>
> Dokumen Config mungkin memerlukan pemformatan khusus, seperti keluar dari kutipan, tergantung pada bagaimana Anda menyediakannya ke Elastic Beanstalk. Lihat Menyediakan dokumen konfigurasi metrik khusus sebagai contoh.

Format log kondisi yang ditingkatkan

AWS Elastic Beanstalk Platform menggunakan format log server web khusus untuk menyampaikan informasi tentang permintaan HTTP secara efisien ke sistem pelaporan kesehatan yang ditingkatkan. Sistem menganalisis log, mengidentifikasi masalah, dan menetapkan kondisi instans dan lingkungan dengan sesuai. Jika Anda menonaktifkan proksi server web di lingkungan Anda dan melayani permintaan langsung dari kontainer web, Anda masih dapat menggunakan pelaporan kondisi yang ditingkatkani sepenuhnya dengan mengonfigurasi server Anda ke output log di lokasi dan format yang agen kondisi Elastic Beanstalk gunakan.

> **Note**
>
> Informasi di halaman ini hanya relevan untuk platform berbasis Linux. Pada platform Windows Server, Elastic Beanstalk menerima informasi tentang permintaan HTTP langsung dari server web IIS. Untuk detail selengkapnya, lihat Metrik server web menangkap IIS pada Windows server.

Konfigurasi log server web

Platform Elastic Beanstalk dikonfigurasi untuk mengeluarkan dua log dengan informasi tentang permintaan HTTP. Yang pertama adalah dalam format verbose dan memberikan informasi detail tentang permintaan, termasuk informasi agen pengguna yang mengirimkan permintaan dan stempel waktu yang dapat dibaca manusia.

/var/log/nginx/access.log

Contoh berikut adalah dari proksi nginx yang sedang berjalan pada lingkungan web server Ruby, tetapi format ini mirip untuk Apache.

```
172.31.24.3 - - [23/Jul/2015:00:21:20 +0000] "GET / HTTP/1.1" 200 11 "-" "curl/7.22.0
 (x86_64-pc-linux-gnu) libcurl/7.22.0 OpenSSL/1.0.1 zlib/1.2.3.4 libidn/1.23
 librtmp/2.3" "177.72.242.17"
172.31.24.3 - - [23/Jul/2015:00:21:21 +0000] "GET / HTTP/1.1" 200 11 "-" "curl/7.22.0
 (x86_64-pc-linux-gnu) libcurl/7.22.0 OpenSSL/1.0.1 zlib/1.2.3.4 libidn/1.23
 librtmp/2.3" "177.72.242.17"
172.31.24.3 - - [23/Jul/2015:00:21:22 +0000] "GET / HTTP/1.1" 200 11 "-" "curl/7.22.0
 (x86_64-pc-linux-gnu) libcurl/7.22.0 OpenSSL/1.0.1 zlib/1.2.3.4 libidn/1.23
 librtmp/2.3" "177.72.242.17"
172.31.24.3 - - [23/Jul/2015:00:21:22 +0000] "GET / HTTP/1.1" 200 11 "-" "curl/7.22.0
 (x86_64-pc-linux-gnu) libcurl/7.22.0 OpenSSL/1.0.1 zlib/1.2.3.4 libidn/1.23
 librtmp/2.3" "177.72.242.17"
172.31.24.3 - - [23/Jul/2015:00:21:22 +0000] "GET / HTTP/1.1" 200 11 "-" "curl/7.22.0
 (x86_64-pc-linux-gnu) libcurl/7.22.0 OpenSSL/1.0.1 zlib/1.2.3.4 libidn/1.23
 librtmp/2.3" "177.72.242.17"
```

Log kedua dalam format singkat. Ini mencakup informasi yang relevan hanya untuk pelaporan kondisi yang ditingkatkan. Log ini adalah output ke subfolder bernama `healthd` dan berputar setiap jam. Log lama dihapus segera setelah diputar.

/var/log/nginx/healthd/application.log.2015-07-23-00

Contoh berikut menunjukkan log dalam format yang dapat dibaca mesin.

```
1437609879.311"/"200"0.083"0.083"177.72.242.17
1437609879.874"/"200"0.347"0.347"177.72.242.17
1437609880.006"/bad/path"404"0.001"0.001"177.72.242.17
1437609880.058"/"200"0.530"0.530"177.72.242.17
1437609880.928"/bad/path"404"0.001"0.001"177.72.242.17
```

Format log kondisi yang ditingkatkan mencakup informasi berikut:

- Waktu permintaan, dalam waktu Unix
- Alur dari permintaan
- Kode status HTTP untuk hasil
- Waktu permintaan
- Waktu hulu
- Header HTTP `X-Forwarded-For`

Untuk proksi nginx, waktu dicetak dalam detik floating-point, dengan tiga tempat desimal. Untuk Apache, seluruh mikrodetik digunakan.

> **Note**
>
> Jika Anda melihat peringatan yang mirip dengan berikut ini dalam berkas log, di mana `DATE-TIME` adalah tanggal dan waktu, dan Anda menggunakan proxy khusus, seperti di lingkungan Docker multi-kontainer, Anda harus menggunakan .ebextension untuk mengonfigurasi lingkungan Anda sehingga `healthd` dapat membaca berkas log Anda:
>
> ```
> W, [DATE-TIME #1922] WARN -- : log file "/var/log/nginx/healthd/
> application.log.DATE-TIME" does not exist
> ```
>
> Anda dapat memulai dengan .ebextension di Contoh Docker multi-kontainer.

/etc/nginx/conf.d/webapp_healthd.conf

Contoh berikut menunjukkan konfigurasi log untuk nginx dengan format log `healthd` yang tersorot.

```
upstream my_app {
  server unix:///var/run/puma/my_app.sock;
}

log_format healthd '$msec"$uri"'
                '$status"$request_time"$upstream_response_time"'
                '$http_x_forwarded_for';

server {
  listen 80;
  server_name _ localhost; # need to listen to localhost for worker tier

  if ($time_iso8601 ~ "^(\d{4})-(\d{2})-(\d{2})T(\d{2})") {
    set $year $1;
    set $month $2;
    set $day $3;
    set $hour $4;
  }

  access_log  /var/log/nginx/access.log  main;
  access_log /var/log/nginx/healthd/application.log.$year-$month-$day-$hour healthd;
```

```
  location / {
    proxy_pass http://my_app; # match the name of upstream directive which is defined
 above
    proxy_set_header Host $host;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
  }

  location /assets {
    alias /var/app/current/public/assets;
    gzip_static on;
    gzip on;
    expires max;
    add_header Cache-Control public;
  }

  location /public {
    alias /var/app/current/public;
    gzip_static on;
    gzip on;
    expires max;
    add_header Cache-Control public;
  }
}
```

/etc/httpd/conf.d/healthd.conf

Contoh berikut menunjukkan pengaturan untuk log konfigurasi untuk Apache.

```
LogFormat "%{%s}t\"%U\"%s\"%D\"%D\"%{X-Forwarded-For}i" healthd
CustomLog "|/usr/sbin/rotatelogs /var/log/httpd/healthd/application.log.%Y-%m-%d-%H
 3600" healthd
```

Menghasilkan log untuk pelaporan kondisi yang ditingkatkan

Untuk menyediakan log ke agen kondisi, Anda harus melakukan hal berikut:

- Log output dalam format yang benar, seperti yang ditunjukkan pada bagian sebelumnya
- Keluaran log ke `/var/log/nginx/healthd/`
- Log nama menggunakan format berikut: `application.log.$year-$month-$day-$hour`
- Putar log sekali per jam
- Jangan memotong log

Pemberitahuan dan pemecahan masalah

Halaman ini mencantumkan contoh pesan penyebab untuk masalah umum dan tautan ke informasi lebih lanjut. Pesan penyebab muncul di halaman gambaran umum lingkungan dari konsol Elastic Beanstalk dan dicatat dalam peristiwa ketika masalah kondisi tetap ada setelah beberapa pemeriksaan.

Deployment

Elastic Beanstalk memantau lingkungan Anda untuk konsistensi setelah deployment. Jika deployment bergulir gagal, versi aplikasi Anda yang berjalan pada instans di lingkungan Anda dapat bervariasi. Hal ini dapat terjadi jika deployment berhasil pada satu atau lebih batch tetapi gagal sebelum semua batch selesai.

Versi aplikasi yang salah ditemukan pada 2 dari 5 instans. Versi "v1" (deployment 1) yang diharapkan.

Versi aplikasi yang salah pada instans lingkungan. Versi "v1" (deployment 1) yang diharapkan.

Versi aplikasi yang diharapkan tidak berjalan pada beberapa atau semua instans dalam lingkungan.

Versi aplikasi yang salah "v2" (deployment 2). Versi "v1" (deployment 1) yang diharapkan.

Aplikasi yang di-deploy pada instans berbeda dari versi yang diharapkan. Jika deployment gagal, versi yang diharapkan disetel ulang ke versi dari deployment terbaru yang telah sukses. Dalam contoh di atas, deployment pertama (versi "v1") berhasil, tetapi deployment kedua (versi "v2") gagal. Setiap instans yang menjalankan "v2" dianggap tidak sehat.

Untuk mengatasi masalah ini, mulai deployment lain. Anda dapat men-deploy ulang versi sebelumnya yang Anda tahu bekerja, atau mengonfigurasi lingkungan Anda untuk mengabaikan pemeriksaan kondisi selama deployment dan men-deploy ulang versi baru untuk memaksa deployment agar terselesaikan.

Anda juga dapat mengidentifikasi dan mengakhiri instans yang menjalankan versi aplikasi yang salah. Elastic Beanstalk akan meluncurkan instans dengan versi yang benar untuk menggantikan setiap instans yang Anda akhiri. Gunakan perintah kondisi EB CLI untuk mengidentifikasi instans yang menjalankan versi aplikasi yang salah.

Server aplikasi

15% permintaan salah dengan HTTP 4xx

20% dari permintaan ke ELB salah dengan HTTP 4xx.

Persentase permintaan HTTP yang tinggi untuk instans atau lingkungan mengalami kegagalan dengan kesalahan 4xx.

Kode status seri 400 menunjukkan bahwa pengguna membuat permintaan yang buruk, seperti meminta halaman yang tidak ada (404 File Tidak Ditemukan) atau bahwa pengguna tidak memiliki akses (403 Terlarang). Jumlah 404s yang rendah bukannya tidak biasa tetapi jumlah yang besar bisa berarti bahwa ada tautan internal atau eksternal ke halaman yang tidak tersedia. Masalah ini dapat diatasi dengan memperbaiki tautan internal yang buruk dan menambahkan pengalihan untuk tautan eksternal yang buruk.

5% permintaan gagal dengan HTTP 5xx

3% dari permintaan ke ELB gagal dengan HTTP 5xx.

Persentase tinggi permintaan HTTP untuk instans atau lingkungan mengalami kegagalan dengan kode status seri 500.

Kode status seri 500 menunjukkan bahwa server aplikasi mengalami kesalahan internal. Masalah ini menunjukkan bahwa ada kesalahan dalam kode aplikasi Anda dan harus diidentifikasi dan diperbaiki dengan cepat.

95% CPU sedang digunakan

Pada instans, agen kondisi melaporkan persentase yang sangat tinggi dari penggunaan CPU dan menetapkan kondisi instans menjadi Peringatan atau Berdegradasi.

Skalakan lingkungan Anda untuk mengambil beban dari instans.

Instans pekerja

20 pesan menunggu dalam antrian (25 detik yang lalu)

Permintaan sedang ditambahkan ke antrian lingkungan pekerja Anda lebih cepat daripada yang dapat diproses. Skalakan lingkungan Anda untuk meningkatkan kapasitas.

5 pesan dalam Antrian Surat Mati (15 detik yang lalu)

Permintaan pekerja mengalami kegagalan berulang kali dan ditambahkan ke the section called “Antrean surat mati”. Periksa permintaan dalam antrean surat mati untuk mengetahui mengapa mereka mengalami kegagalan.

Sumber daya lainnya

4 instance aktif di bawah ukuran minimum grup Auto Scaling 5

Jumlah instans yang berjalan di lingkungan Anda kurang dari minimum yang dikonfigurasi untuk grup Auto Scaling.

Pemberitahuan grup Auto Scaling (nama grup) telah dihapus atau dimodifikasi

Notifikasi yang dikonfigurasi untuk grup Auto Scaling Anda telah diubah di luar Elastic Beanstalk.

Mengelola alarm

Topik ini memandu Anda melalui langkah-langkah untuk membuat alarm untuk metrik yang Anda pantau. Ini juga memberikan instruksi untuk melihat alarm Anda yang ada dan untuk memeriksa statusnya.

Anda dapat membuat alarm untuk metrik yang Anda pantau dengan menggunakan konsol Elastic Beanstalk. Alarm membantu Anda memantau perubahan pada AWS Elastic Beanstalk lingkungan Anda sehingga Anda dapat dengan mudah mengidentifikasi dan mengurangi masalah sebelum terjadi. Misalnya, Anda dapat mengatur alarm yang memberi tahu Anda ketika penggunaan CPU di suatu lingkungan melebihi ambang batas tertentu sehingga memastikan bahwa Anda diberitahu sebelum potensi masalah terjadi. Untuk informasi selengkapnya, lihat Menggunakan Elastic Beanstalk dengan Amazon CloudWatch.

> **Note**
>
> Elastic CloudWatch Beanstalk digunakan untuk pemantauan dan alarm CloudWatch , artinya biaya diterapkan AWS ke akun Anda untuk alarm apa pun yang Anda gunakan.

Untuk informasi selengkapnya tentang cara pemantauan metrik tertentu, lihat Pelaporan kondisi dasar.

Untuk memeriksa status alarm Anda

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Di panel navigasi, pilih Alarm.

Halaman menampilkan daftar alarm yang ada. Jika ada alarm dalam status alarm, alarm ditandai dengan ikon peringatan
() 
.

4. Untuk memfilter alarm, pilih menu drop-down, lalu pilih filter.
5. Untuk mengedit atau menghapus alarm, pilih ikon edit
(
)
atau ikon hapus
(
),
masing-masing.

Untuk membuat alarm

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Di panel navigasi, pilih Pemantauan.
4. Temukan metrik yang ingin Anda buat alarm, lalu pilih ikon alarm
(
).
Halaman Tambahkan alarm ditampilkan.
5. Masukkan detail mengenai alarm:
   - Nama: Nama untuk alarm ini.
   - Deskripsi (opsional): Deskripsi singkat mengenai alarm ini.
   - Periode: Interval waktu antara pembacaan.
   - Ambang Batas: Menjelaskan perilaku dan nilai yang harus dilampaui metrik untuk memicu alarm.
   - Ubah status setelah: Jumlah waktu setelah ambang batas telah terlampaui yang memicu perubahan status alarm.
   - Beri tahu: Topik Amazon SNS yang diberitahu ketika alarm mengubah status.
   - Beri tahu saat keadaan berubah ke:
     - OK: Metrik berada dalam ambang batas yang ditetapkan.
     - Alarm: Metrik melampaui ambang batas yang ditentukan.

- Data tidak cukup: Alarm baru saja dimulai, metrik tidak tersedia, atau tidak cukup data yang tersedia bagi metrik untuk menentukan status alarm.

6. Pilih Menambahkan. Status lingkungan berubah menjadi abu-abu ketika pembaruan lingkungan. Anda dapat melihat alarm yang Anda buat dengan memilih Alarm di panel navigasi.

Melihat riwayat perubahan lingkungan Elastic Beanstalk

Topik ini menjelaskan bagaimana Anda dapat menggunakan Konsol Elastic Beanstalk untuk melihat riwayat perubahan konfigurasi yang telah dilakukan pada lingkungan Elastic Beanstalk Anda.

Elastic Beanstalk mengambil riwayat perubahan Anda dari peristiwa yang tercatat di AWS CloudTrail dan menampilkannya dalam daftar yang dapat Anda navigasi dan filter dengan mudah.

Panel Riwayat Perubahan menampilkan informasi berikut untuk perubahan yang dibuat pada lingkungan Anda:

- Tanggal dan waktu ketika perubahan dilakukan
- Pengguna IAM yang bertanggung jawab atas perubahan yang dilakukan
- Alat sumber (baik antarmuka baris perintah Elastic Beanstalk (EB CLI) atau konsol) yang digunakan untuk membuat perubahan
- Parameter konfigurasi dan nilai-nilai baru yang ditetapkan

Setiap data sensitif yang merupakan bagian dari perubahan, seperti nama pengguna basis data yang terpengaruh oleh perubahan tersebut, tidak ditampilkan di panel.

Untuk melihat riwayat perubahan

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Ubah riwayat.

   Halaman Ubah Riwayat menunjukkan daftar perubahan konfigurasi yang dibuat pada lingkungan Elastic Beanstalk Anda.

Perhatikan poin-poin berikut tentang menavigasi informasi di halaman ini:

- Anda dapat menelusuri halaman pada daftar dengan memilih **<** (sebelumnya) atau **>** (berikutnya), atau dengan memilih nomor halaman tertentu.
- Di bawah kolom Perubahan konfigurasi, pilih ikon panah untuk berpindah antara memperluas dan memperkecil daftar perubahan di bawah judul Perubahan yang dibuat.
- Gunakan bilah pencarian untuk memfilter hasil dari daftar riwayat perubahan. Anda dapat memasukkan string apa pun untuk mempersempit daftar perubahan yang ditampilkan.

Perhatikan hal berikut tentang memfilter hasil yang ditampilkan:

- Filter pencarian tidak peka terhadap huruf besar/kecil.
- Anda dapat memfilter perubahan yang ditampilkan berdasarkan informasi di bawah kolom Perubahan konfigurasi, bahkan ketika tidak terlihat karena sedang diperkecil di dalam Perubahan yang dibuat.
- Anda hanya dapat memfilter hasil yang ditampilkan. Namun, filter tetap diterapkan meskipun Anda memilih untuk membuka halaman lain untuk menampilkan lebih banyak hasil. Hasil yang difilter juga ditambahkan ke rangkaian hasil halaman berikutnya.

Contoh berikut menunjukkan bagaimana data yang ditampilkan pada layar sebelumnya dapat difilter:

- Masukkan **`GettingStartedApp-env`** di kotak pencarian untuk mempersempit hasil agar hanya menyertakan perubahan yang dibuat pada lingkungan bernama GettingStartedApp-env.
- Masukkan **`example3`** di kotak pencarian untuk mempersempit hasil menjadi hanya menyertakan perubahan yang dibuat oleh pengguna IAM dengan nama pengguna yang berisi string example3.
- Masukkan **`2020-10`** di kotak pencarian untuk mempersempit hasil menjadi hanya menyertakan perubahan yang dilakukan selama bulan Oktober 2020. Ubah nilai pencarian ke **`2020-10-16`** untuk memfilter lebih lanjut hasil yang ditampilkan menjadi hanya menyertakan perubahan yang dibuat pada 16 Oktober 2020.
- Masukkan **`proxy:staticfiles`** di kotak pencarian untuk mempersempit hasil menjadi hanya menyertakan perubahan yang dibuat untuk namespace bernama aws:elasticbeanstalk:environment:proxy:staticfiles. Baris yang ditampilkan adalah hasil dari filter. Hal ini berlaku bahkan untuk hasil yang diperkecil di bawah Perubahan yang dibuat.

Melihat alur kejadian lingkungan Elastic Beanstalk

Topik ini menjelaskan cara mengakses acara dan notifikasi yang terkait dengan aplikasi Anda.

Melihat acara dengan konsol Elastic Beanstalk

Untuk melihat acara dengan konsol Elastic Beanstalk

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Di panel navigasi, pilih Peristiwa.

   Halaman peristiwa menampilkan daftar semua peristiwa yang telah dicatat untuk lingkungan. Anda dapat menelusuri halaman melalui daftar pilihan < (sebelumnya), > (berikutnya), atau nomor halaman. Anda dapat memfilter jenis peristiwa yang ditampilkan dengan menggunakan daftar drop-down Tingkat Kepelikan.

Melihat acara dengan alat baris perintah

EB CLI dan AWS CLI memberikan perintah untuk mengambil peristiwa. Jika Anda mengelola lingkungan Anda menggunakan EB CLI, gunakan eb events untuk mencetak daftar peristiwa. Perintah ini juga memiliki pilihan `--follow` yang akan terus menampilkan peristiwa baru sampai Anda menekan Ctrl+C untuk menghentikan output.

Untuk menarik peristiwa menggunakan AWS CLI, gunakan `describe-events` perintah dan tentukan lingkungan dengan nama atau ID:

```
$ aws elasticbeanstalk describe-events --environment-id e-gbjzqccra3
{
    "Events": [
        {
            "ApplicationName": "elastic-beanstalk-example",
            "EnvironmentName": "elasticBeanstalkExa-env",
            "Severity": "INFO",
            "RequestId": "a4c7bfd6-2043-11e5-91e2-9114455c358a",
            "Message": "Environment update completed successfully.",
            "EventDate": "2015-07-01T22:52:12.639Z"
        },
...
```

Untuk informasi selengkapnya tentang alat baris perintah, lihat???.

Membuat daftar dan menghubungkan ke server instans

Topik ini menjelaskan cara melihat daftar EC2 instans Amazon yang menjalankan lingkungan aplikasi Elastic Beanstalk Anda dan cara menghubungkannya.

Anda dapat melihat daftar EC2 instans Amazon yang menjalankan lingkungan AWS Elastic Beanstalk aplikasi Anda melalui konsol Elastic Beanstalk. Anda dapat terhubung ke instans menggunakan klien SSH apa pun. Anda dapat terhubung ke instans yang menjalankan Windows menggunakan Remote Desktop.

> **Important**
>
> Sebelum dapat mengakses Elastic Beanstalk—instans Amazon yang disediakan, Anda harus membuat key pair EC2 Amazon dan mengonfigurasi Amazon yang disediakan Elastic Beanstalk—Amazon yang disediakan untuk menggunakan EC2 key pair Amazon. EC2instances EC2 Anda dapat mengatur pasangan EC2 kunci Amazon menggunakan AWS Management Console. Untuk petunjuk cara membuat key pair untuk Amazon EC2, lihat Panduan EC2 Memulai Amazon. Untuk informasi selengkapnya tentang cara mengonfigurasi EC2 instans Amazon agar menggunakan EC2 key pair Amazon, lihatEC2 key pair.
> Secara default, Elastic Beanstalk tidak mengaktifkan EC2 koneksi jarak jauh ke instance dalam wadah Windows kecuali yang ada di wadah Windows lama. (Elastic EC2 Beanstalk mengonfigurasi instance dalam wadah Windows lama untuk menggunakan port 3389 untuk koneksi RDP.) Anda dapat mengaktifkan koneksi jarak jauh ke EC2 instans yang menjalankan Windows dengan menambahkan aturan ke grup keamanan yang mengotorisasi lalu lintas masuk ke instans. Kami sangat merekomendasikan agar Anda menghapus aturan saat Anda mengakhiri sambungan jarak jauh. Anda dapat menambahkan aturan lagi pada saat Anda perlu masuk dari jarak jauh. Untuk informasi selengkapnya, lihat Menambahkan Aturan untuk RDP Lalu Lintas Masuk ke Instans Windows dan Connect ke Instans Windows Anda di Panduan Pengguna Amazon Elastic Compute Cloud untuk Microsoft Windows.

Untuk melihat dan terhubung ke EC2 instans Amazon untuk suatu lingkungan

1. Buka EC2 konsol Amazon di https://console.aws.amazon.com/ec2/.
2. Di panel navigasi konsol tersebut, pilih Penyeimbang Beban.
3. Load balancers yang dibuat oleh Elastic Beanstalk memiliki awseb dalam namanya. Temukan penyeimbang beban untuk lingkungan Anda dan klik.

4. Pilih tab Instans di panel bawah konsol tersebut.

   Daftar instans yang digunakan penyeimbang beban untuk lingkungan Elastic Beanstalk Anda ditampilkan. Membuat catatan ID instans yang ingin Anda hubungkan.
5. Di panel navigasi EC2 konsol Amazon, pilih Instans, dan temukan ID instans Anda dalam daftar.
6. Klik kanan ID instans untuk EC2 instans Amazon yang berjalan di penyeimbang beban lingkungan Anda, lalu pilih Connect dari menu konteks.
7. Catat alamat DNS publik instans di tab Deskripsi.
8. Connect ke instance yang menjalankan Linux dengan menggunakan klien SSH pilihan Anda, lalu ketik ssh -i .ec2/mykeypair.pem EC2-user@<public-DNS->. of-the-instance

Untuk informasi selengkapnya tentang menghubungkan ke instans Amazon EC2 Linux, lihat Memulai Instans Amazon EC2 Linux di Panduan EC2 Pengguna Amazon.

Jika lingkungan Elastic Beanstalk Anda menggunakan .NET pada platform Windows Server, lihat Memulai Instans EC2 Amazon Windows di Panduan Pengguna Amazon. EC2

Melihat log dari EC2 instans Amazon di lingkungan Elastic Beanstalk Anda

Topik ini menjelaskan jenis log instance yang disediakan Elastic Beanstalk. Ini juga memberikan instruksi terperinci untuk retreiveing dan mengelolanya.

EC2 Instans Amazon di lingkungan Elastic Beanstalk Anda menghasilkan log yang dapat Anda lihat untuk memecahkan masalah dengan aplikasi atau file konfigurasi Anda. Log yang dibuat oleh server web, server aplikasi, skrip platform Elastic Beanstalk AWS CloudFormation , dan disimpan secara lokal pada instance individual. Anda dapat dengan mudah mengambilnya dengan menggunakan konsol manajemen lingkungan atau EB CLI. Anda juga dapat mengonfigurasi lingkungan Anda untuk mengalirkan log ke Amazon CloudWatch Logs secara real time.

Log ekor adalah 100 baris terakhir dari berkas log yang paling umum digunakan—log operasional Elastic Beanstalk dan log dari server web atau server aplikasi. Ketika Anda meminta log ekor di konsol manajemen lingkungan atau dengan eb logs, instans di lingkungan Anda menggabungkan entri log terbaru ke dalam satu file teks dan mengunduhnya ke Amazon S3.

Log paket adalah log lengkap untuk berbagai berkas log yang lebih luas, termasuk log dari yum dan cron dan beberapa log dari AWS CloudFormation. Ketika Anda meminta log paket, instans

di lingkungan Anda memaketkan berkas log lengkap ke dalam arsip ZIP dan mengunggahnya ke Amazon S3.

Untuk mengunggah log yang dirotasi ke Amazon S3, instans di lingkungan Anda harus memiliki profil instans dengan izin untuk menulis ke bucket Elastic Beanstalk Amazon S3 Anda. Izin ini termasuk dalam profil instans default di mana Elastic Beanstalk meminta Anda untuk membuatnya ketika Anda meluncurkan lingkungan di konsol Elastic Beans untuk pertama kalinya.

Untuk mengambil log instans

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pilih Log di panel navigasi.
4. Pilih Log Permintaan, lalu pilih jenis log yang akan diambil. Untuk mendapatkan log ekor, pilih 100 Baris terakhir. Untuk mendapatkan log paket, pilih Log Lengkap.
5. Ketika Elastic Beanstalk selesai mengambil log Anda, pilih Unduh.

Elastic Beanstalk menyimpan log ekor dan paket dalam bucket Amazon S3, dan menghasilkan URL Amazon S3 yang telah ditandatangani sebelumnya yang dapat Anda gunakan untuk mengakses log Anda. Elastic Beanstalk menghapus file dari Amazon S3 setelah durasi 15 menit.

> **Warning**
>
> Siapa pun yang memiliki URL Amazon S3 yang telah ditandatangani dapat mengakses file sebelum dihapus. Jadikan URL hanya tersedia untuk pihak tepercaya.

> **Note**
>
> Kebijakan pengguna Anda harus memiliki izin `s3:DeleteObject`. Elastic Beanstalk menggunakan izin pengguna Anda untuk menghapus log dari Amazon S3.

Untuk mempertahankan log, Anda dapat mengonfigurasi lingkungan Anda untuk menerbitkan log ke Amazon S3 secara otomatis setelah dirotasi. Untuk mengaktifkan rotasi log ke Amazon S3,

ikuti prosedur di Mengonfigurasi tampilan log instans. Instans di lingkungan Anda akan mencoba mengunggah log yang telah dirotasi satu kali per jam.

Jika aplikasi Anda menghasilkan log di lokasi yang bukan merupakan bagian dari konfigurasi default untuk platform lingkungan Anda, Anda dapat memperpanjang konfigurasi default dengan menggunakan file konfigurasi (`.ebextensions`). Anda dapat menambahkan berkas log aplikasi Anda untuk log ekor, log paket, atau log rotasi.

Untuk streaming log real-time dan penyimpanan jangka panjang, konfigurasikan lingkungan Anda untuk mengalirkan log ke Amazon CloudWatch Logs.

Bagian-bagian

- Log lokasi di EC2 instans Amazon
- Lokasi log di Amazon S3
- Pengaturan rotasi log pada Linux
- Memperluas konfigurasi tugas log default
- Streaming file log ke Amazon CloudWatch Logs

Log lokasi di EC2 instans Amazon

Log disimpan di lokasi standar di EC2 instans Amazon di lingkungan Anda. Elastic Beanstalk menghasilkan log berikut.

Amazon Linux 2

- `/var/log/eb-engine.log`

Amazon Linux AMI (AL1)

> **Note**
>
> Pada tanggal 18 Juli 2022, Elastic Beanstalk menetapkan status semua cabang platform berdasarkan Amazon Linux AMI () untuk pensiun. AL1 Untuk informasi selengkapnya tentang migrasi ke cabang platform Amazon Linux 2023 saat ini dan didukung penuh, lihat. Memigrasi aplikasi Elastic Beanstalk Linux Anda ke Amazon Linux 2023 atau Amazon Linux 2

- `/var/log/eb-activity.log`

- `/var/log/eb-commandprocessor.log`

Windows Server

- `C:\Program Files\Amazon\ElasticBeanstalk\logs\`
- `C:\cfn\log\cfn-init.log`

Log ini berisi pesan tentang aktivitas deployment, termasuk pesan yang terkait dengan file konfigurasi (`.ebextensions`).

Setiap aplikasi dan server web menyimpan log dalam foldernya sendiri:

- Apache – `/var/log/httpd/`
- IIS – `C:\inetpub\wwwroot\`
- Node.js – `/var/log/nodejs/`
- nginx – `/var/log/nginx/`
- Penumpang – `/var/app/support/logs/`
- Puma – `/var/log/puma/`
- Python – `/opt/python/log/`
- Tomcat – `/var/log/tomcat/`

Lokasi log di Amazon S3

Ketika Anda meminta log ekor atau paket dari lingkungan Anda, atau ketika instans mengunggah log dirotasi, mereka disimpan dalam bucket Elastic Beanstalk Anda di Amazon S3. Elastic Beanstalk menciptakan `elasticbeanstalk-`*`region`*`-`*`account-id`* ember yang diberi nama AWS untuk setiap Wilayah tempat Anda membuat lingkungan. Dalam bucket ini, log disimpan di bawah jalur `resources/environments/logs/`*`logtype`*`/`*`environment-id`*`/`*`instance-id`*.

Misalnya, log dari instance`i-0a1fd158`, di AWS lingkungan Elastic `e-mpcwnwheky` Beanstalk `us-west-2` di Region dalam akun, disimpan `123456789012` di lokasi berikut:

- Log Ekor –

  ```
  s3://elasticbeanstalk-us-west-2-123456789012/resources/environments/logs/tail/e-mpcwnwheky/i-0a1fd158
  ```

- Log Paket –

  `s3://elasticbeanstalk-us-west-2-123456789012/resources/environments/logs/bundle/e-mpcwnwheky/i-0a1fd158`

- Log Dirotasi –

  `s3://elasticbeanstalk-us-west-2-123456789012/resources/environments/logs/publish/e-mpcwnwheky/i-0a1fd158`

> **Note**
>
> Anda dapat menemukan ID lingkungan Anda di konsol manajemen lingkungan.

Elastic Beanstalk menghapus log ekor dan paket dari Amazon S3 secara otomatis 15 menit setelah dibuat. Log yang dirotasi bertahan hingga Anda menghapusnya atau memindahkannya ke S3 Glacier.

Pengaturan rotasi log pada Linux

Pada platform Linux, Elastic Beanstalk menggunakan `logrotate` untuk merotasi log secara berkala. Jika dikonfigurasi, setelah log dirotasi secara lokal, tugas rotasi log mengambilnya dan mengunggahnya ke Amazon S3. Log yang dirotasi secara lokal tidak muncul di log ekor atau paket secara default.

Anda dapat menemukan file konfigurasi Elastic Beanstalk untuk `logrotate` di `/etc/logrotate.elasticbeanstalk.hourly/`. Pengaturan rotasi ini khusus untuk platform, dan mungkin berubah dalam versi platform yang akan datang. Untuk informasi selengkapnya tentang pengaturan yang tersedia dan konfigurasi contoh, jalankan `man logrotate`.

File-file konfigurasi dipanggil oleh tugas cron di `/etc/cron.hourly/`. Untuk informasi lebih lanjut tentang `cron`, lihat `man cron`.

Memperluas konfigurasi tugas log default

Elastic Beanstalk menggunakan file dalam `/opt/elasticbeanstalk/tasks` subfolder (Linux`C:\Program Files\Amazon\ElasticBeanstalk\config`) atau (Windows Server) pada instans EC2 Amazon untuk mengonfigurasi tugas untuk log ekor, log bundel, dan rotasi log.

Di Amazon Linux:

- Log Ekor –

  `/opt/elasticbeanstalk/tasks/taillogs.d/`
- Log Paket –

  `/opt/elasticbeanstalk/tasks/bundlelogs.d/`
- Log Dirotasi –

  `/opt/elasticbeanstalk/tasks/publishlogs.d/`

Di Windows Server:

- Log Ekor –

  `c:\Program Files\Amazon\ElasticBeanstalk\config\taillogs.d\`
- Log Paket –

  `c:\Program Files\Amazon\ElasticBeanstalk\config\bundlelogs.d\`
- Log Dirotasi –

  `c:\Program Files\Amazon\ElasticBeanstalk\config\publogs.d\`

Misalnya, file `eb-activity.conf` pada Linux menambahkan dua berkas log untuk tugas log ekor.

`/opt/elasticbeanstalk/tasks/taillogs.d/eb-activity.conf`

```
/var/log/eb-commandprocessor.log
/var/log/eb-activity.log
```

Anda dapat menggunakan file konfigurasi lingkungan (`.ebextensions`) untuk menambahkan file `.conf` Anda sendiri ke folder ini. File `.conf` membuat daftar file log khusus untuk aplikasi Anda, di mana Elastic Beanstalk menambah tugas berkas log.

Gunakan bagian `files` untuk menambahkan file konfigurasi ke tugas yang ingin Anda ubah. Sebagai contoh, teks konfigurasi berikut menambahkan file konfigurasi log untuk setiap instans di lingkungan Anda. File konfigurasi log ini, `cloud-init.conf`, menambahkan `/var/log/cloud-init.log` untuk log ekor.

```
files:
  "/opt/elasticbeanstalk/tasks/taillogs.d/cloud-init.conf" :
    mode: "000755"
    owner: root
    group: root
    content: |
      /var/log/cloud-init.log
```

Tambahkan teks ini ke file dengan ekstensi nama file `.config` ke paket sumber Anda di bawah folder bernama `.ebextensions`.

```
~/workspace/my-app
|-- .ebextensions
|   `-- tail-logs.config
|-- index.php
`-- styles.css
```

Pada platform Linux, Anda juga dapat menggunakan karakter wildcard dalam konfigurasi tugas log. File konfigurasi ini menambahkan semua file dengan ekstensi nama file `.log` folder `log` di akar aplikasi ke log paket.

```
files:
  "/opt/elasticbeanstalk/tasks/bundlelogs.d/applogs.conf" :
    mode: "000755"
    owner: root
    group: root
    content: |
      /var/app/current/log/*.log
```

Konfigurasi tugas log tidak mendukung karakter wildcard pada platform Windows.

> **Note**
>
> Untuk membantu membiasakan diri Anda dengan prosedur penyesuaian log, Anda dapat men-deploy aplikasi sampel menggunakan EB CLI. Untuk ini, EB CLI membuat direktori aplikasi lokal yang berisi subdirektori `.ebextentions` dengan konfigurasi sampel. Anda juga dapat menggunakan berkas log aplikasi sampel untuk menjelajahi fitur pengambilan log yang dijelaskan dalam topik ini.

Untuk informasi selengkapnya tentang penggunaan file konfigurasi, lihat Penyesuaian lingkungan lanjutan dengan file konfigurasi (`.ebextensions`).

Sama seperti memperluas log ekor dan log paket, Anda dapat memperpanjang rotasi log menggunakan file konfigurasi. Setiap kali Elastic Beanstalk merotasi log-nya sendiri dan mengunggahnya ke Amazon S3, itu juga akan berotasi dan mengunggah log tambahan Anda. Ekstensi rotasi log berperilaku berbeda tergantung pada sistem operasi platform. Bagian berikut menjelaskan dua kasus.

Memperluas rotasi log di Linux

Seperti yang dijelaskan di Pengaturan rotasi log pada Linux, Elastic Beanstalk menggunakan `logrotate` untuk merotasi log pada platform Linux. Ketika Anda mengonfigurasi berkas log aplikasi Anda untuk rotasi log, aplikasi tidak perlu membuat salinan berkas log. Elastic Beanstalk mengonfigurasi `logrotate` untuk membuat salinan berkas log aplikasi Anda untuk setiap rotasi. Oleh karena itu, aplikasi harus tetap membuka berkas log ketika aplikasi tidak secara aktif menulis kepada mereka.

Memperluas rotasi log pada server Windows

Pada Server Windows, ketika Anda mengonfigurasi berkas log aplikasi Anda untuk rotasi log, aplikasi harus merotasi berkas log secara berkala. Elastic Beanstalk mencari file dengan nama dimulai dengan pola yang Anda dikonfigurasi, dan mengambilnya untuk mengunduh ke Amazon S3. Selain itu, tanda baca titik dalam nama file diabaikan, dan Elastic Beanstalk menganggap nama hingga tanda baca titik sebagai nama berkas log dasar.

Elastic Beanstalk mengunggah semua versi berkas log dasar kecuali yang terbaru, karena menganggap bahwa salah satu berkas log aplikasi aktif, yang berpotensi dapat terkunci. Aplikasi Anda, oleh karena itu, dapat menjaga berkas log aktif tetap terkunci selama rotasi.

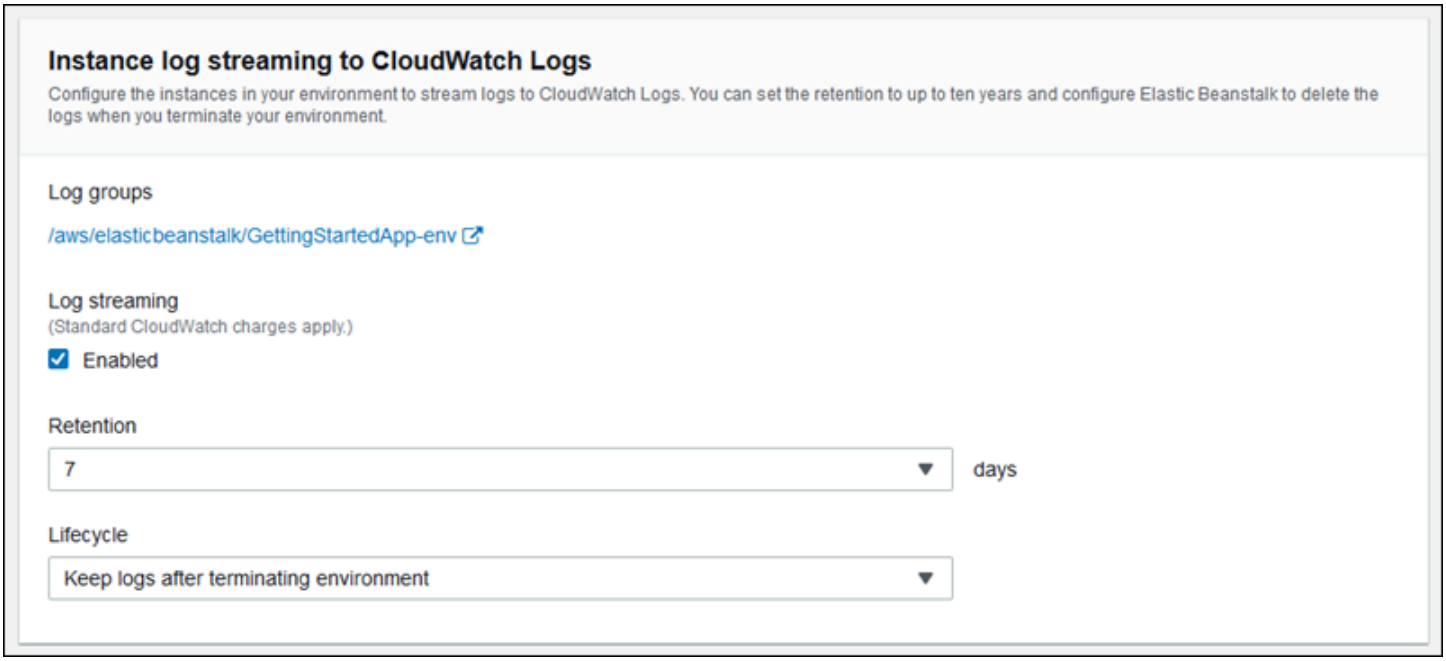
Misalnya, aplikasi Anda menulis ke berkas log bernama `my_log.log`, dan Anda menentukan nama ini di file `.conf` Anda. Aplikasi secara berkala merotasi file. Selama siklus rotasi Elastic Beanstalk,file-file berikut ditemukan dalam folder file log: `my_log.log`, `my_log.0800.log`, `my_log.0830.log`. Elastic Beanstalk menganggap semuanya sebagai versi dari nama dasar `my_log`. File `my_log.log` memiliki waktu modifikasi terbaru, sehingga Elastic Beanstalk hanya mengunggah dua file lainnya, `my_log.0800.log` dan `my_log.0830.log`.

Streaming file log ke Amazon CloudWatch Logs

Anda dapat mengonfigurasi lingkungan Anda untuk mengalirkan log ke Amazon CloudWatch Logs di konsol Elastic Beanstalk atau dengan menggunakan opsi konfigurasi. Dengan CloudWatch Log, setiap instance di lingkungan Anda mengalirkan log ke grup log yang dapat Anda konfigurasikan untuk disimpan selama berminggu-minggu atau bertahun-tahun, bahkan setelah lingkungan Anda dihentikan.

Set pengaliran log bervariasi per lingkungan, tetapi selalu memasukkan `eb-engine.log` dan akses log dari nginx atau server proksi Apache yang berjalan di depan aplikasi Anda.

Anda dapat mengonfigurasi streaming log di konsol Elastic Beanstalk selama pembuatan lingkungan atau untuk lingkungan yang ada. Anda dapat mengatur opsi berikut dari konsol: aktifkan /nonaktifkan streaming CloudWatch log ke Log, atur jumlah hari retensi, dan pilih dari opsi Lifecyle. Dalam contoh berikut, log disimpan hingga tujuh hari, bahkan ketika lingkungan diakhiri.



file konfigurasi berikut mengaktifkan streaming log dengan retensi 180 hari, meskipun lingkungan diakhiri.

Example .ebextensions/log-streaming.config

```
option_settings:
  aws:elasticbeanstalk:cloudwatch:logs:
    StreamLogs: true
    DeleteOnTerminate: false
```

```
    RetentionInDays: 180
```

Menggunakan Elastic Beanstalk dengan layanan lainnya AWS

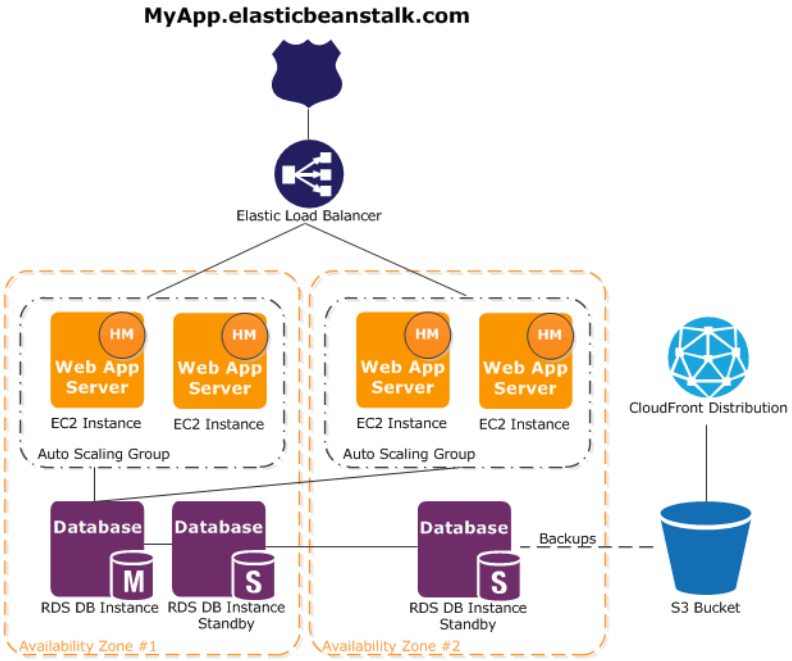
Topik di bagian ini menjelaskan banyak cara Anda dapat menggunakan AWS layanan tambahan dengan aplikasi Elastic Beanstalk Anda. Untuk mengimplementasikan lingkungan aplikasi Anda, Elastic Beanstalk mengelola sumber daya layanan lain atau menggunakan AWS fungsinya. Selain itu, Elastic Beanstalk AWS terintegrasi dengan layanan yang tidak digunakan secara langsung sebagai bagian dari lingkungan Anda.

Topik

- Gambaran umum arsitektur
- Menggunakan Elastic Beanstalk dengan Amazon CloudFront
- Logging panggilan Elastic Beanstalk API dengan AWS CloudTrail
- Menggunakan Elastic Beanstalk dengan Amazon CloudWatch
- Menggunakan Elastic Beanstalk dengan Amazon Logs CloudWatch
- Menggunakan Elastic Beanstalk dengan Amazon EventBridge
- Menemukan dan melacak sumber daya Elastic Beanstalk dengan AWS Config
- Menggunakan Elastic Beanstalk dengan Amazon DynamoDB
- Menggunakan Elastic Beanstalk dengan Amazon ElastiCache
- Menggunakan Elastic Beanstalk dengan Amazon Elastic File System
- Menggunakan Elastic Beanstalk dengan AWS Identity and Access Management
- Menggunakan Elastic Beanstalk dengan Amazon RDS
- Menggunakan Elastic Beanstalk dengan Amazon S3
- Menggunakan Elastic Beanstalk dengan dan AWS Secrets Manager Parameter Store AWS Systems Manager
- Menggunakan Elastic Beanstalk dengan Amazon VPC

Gambaran umum arsitektur

Diagram berikut mengilustrasikan contoh arsitektur Elastic Beanstalk di beberapa Availability Zone yang bekerja dengan produk AWS lain seperti Amazon CloudFront, Amazon Simple Storage Service (Amazon S3), dan Amazon Relational Database Service (Amazon RDS).



Untuk merencanakan toleransi kesalahan, disarankan untuk memiliki EC2 instans Amazon N+1 dan menyebarkan instans Anda di beberapa Availability Zone. Dalam kasus yang tidak mungkin bahwa satu Availability Zone turun, Anda masih akan menjalankan EC2 instans Amazon lainnya di Availability Zone lainnya. Anda dapat menyesuaikan EC2 Auto Scaling Amazon untuk memungkinkan jumlah instans minimum serta beberapa Availability Zone. Untuk petunjuk tentang cara melakukannya, lihat Auto Scaling instance lingkungan Elastic Beanstalk Anda. Untuk informasi selengkapnya tentang membangun aplikasi yang toleran terhadap kesalahan, lanjutkan ke Membangun Aplikasi yang Toleran Terhadap Kesalahan di AWS.

Bagian berikut membahas integrasi lebih detail dengan Amazon CloudFront, Amazon, Amazon DynamoDB Amazon CloudWatch, Amazon RDS, ElastiCache Amazon Route 53, Amazon Simple Storage Service, Amazon VPC, dan IAM.

Menggunakan Elastic Beanstalk dengan Amazon CloudFront

Amazon CloudFront adalah layanan web yang mempercepat distribusi konten web statis dan dinamis Anda, misalnya, .html, .css, .php, gambar, dan file media, ke pengguna akhir. CloudFront mengirimkan konten Anda melalui jaringan lokasi tepi di seluruh dunia. Saat pengguna akhir meminta konten yang Anda sajikan CloudFront, pengguna akan diarahkan ke lokasi tepi yang memberikan latensi terendah, sehingga konten dikirimkan dengan performa terbaik. Jika konten sudah ada di lokasi tepi itu, segera CloudFront kirimkan. Jika konten saat ini tidak berada di lokasi tepi tersebut

CloudFront , ambil dari bucket Amazon S3 atau server HTTP (misalnya, server web) yang telah Anda identifikasi sebagai sumber untuk versi definitif konten Anda.

Setelah Anda membuat dan menerapkan aplikasi Elastic Beanstalk Anda, Anda dapat CloudFront mendaftar dan CloudFront mulai menggunakan untuk mendistribusikan konten Anda. Pelajari selengkapnya CloudFront dari Panduan CloudFront Pengembang Amazon.

Logging panggilan Elastic Beanstalk API dengan AWS CloudTrail

Elastic Beanstalk AWS CloudTrail terintegrasi dengan, layanan yang menyediakan catatan tindakan yang diambil oleh pengguna, peran, atau AWS layanan di Elastic Beanstalk. CloudTrailmenangkap semua panggilan API untuk Elastic Beanstalk sebagai peristiwa, termasuk panggilan dari konsol Elastic Beanstalk, dari EB CLI, dan dari kode Anda ke Elastic Beanstalk. APIs Jika Anda membuat jejak, Anda dapat mengaktifkan pengiriman acara secara terus menerus ke bucket Amazon S3, termasuk CloudTrail acara untuk Elastic Beanstalk. Jika Anda tidak mengonfigurasi jejak, Anda masih dapat melihat peristiwa terbaru di CloudTrail konsol dalam Riwayat acara. Dengan menggunakan informasi yang dikumpulkan oleh CloudTrail, Anda dapat menentukan permintaan yang dibuat untuk Elastic Beanstalk, alamat IP dari mana permintaan dibuat, siapa yang membuat permintaan, kapan dibuat, dan detail tambahan.

Untuk mempelajari selengkapnya CloudTrail, lihat Panduan AWS CloudTrail Pengguna.

Informasi Elastic Beanstalk di CloudTrail

CloudTrail diaktifkan di AWS akun Anda saat Anda membuat akun. Ketika aktivitas terjadi di Elastic Beanstalk, aktivitas tersebut dicatat CloudTrail dalam suatu peristiwa bersama AWS dengan peristiwa layanan lainnya dalam riwayat Peristiwa. Anda dapat melihat, mencari, dan mengunduh acara terbaru di AWS akun Anda. Untuk informasi selengkapnya, lihat Melihat Acara dengan Riwayat CloudTrail Acara.

Untuk catatan peristiwa yang sedang berlangsung di AWS akun Anda, termasuk acara untuk Elastic Beanstalk, buat jejak. Jejak memungkinkan CloudTrail untuk mengirimkan file log ke bucket Amazon S3. Secara default, ketika Anda membuat jejak di konsol, jejak ini diterapkan ke semua Wilayah. Jejak mencatat log peristiwa dari semua Wilayah di partisi AWS dan mengirimkan file log ke bucket Amazon S3 yang Anda tentukan. Selain itu, Anda dapat mengonfigurasi AWS layanan lain untuk menganalisis lebih lanjut dan menindaklanjuti data peristiwa yang dikumpulkan dalam CloudTrail log. Untuk informasi selengkapnya, lihat:

- Gambaran Umum untuk Membuat Jejak

- CloudTrail Layanan dan Integrasi yang Didukung
- Mengonfigurasi Notifikasi Amazon SNS untuk CloudTrail
- Menerima File CloudTrail Log dari Beberapa Wilayah dan Menerima File CloudTrail Log dari Beberapa Akun

Semua tindakan Elastic Beanstalk CloudTrail dicatat oleh dan didokumentasikan dalam Referensi API.AWS Elastic Beanstalk Misalnya, panggilan ke`DescribeApplications`,`UpdateEnvironment`, dan `ListTagsForResource` tindakan menghasilkan entri dalam file CloudTrail log.

Setiap entri peristiwa atau log berisi informasi tentang siapa yang membuat permintaan tersebut. Informasi identitas membantu Anda menentukan hal berikut ini:

- Apakah permintaan tersebut dibuat dengan kredensial root atau pengguna IAM.
- Apakah permintaan tersebut dibuat dengan kredensial keamanan sementara untuk satu peran atau pengguna terfederasi.
- Apakah permintaan itu dibuat oleh AWS layanan lain.

Untuk informasi lain, lihat Elemen userIdentity CloudTrail .

Memahami entri file log Elastic Beanstalk

Trail adalah konfigurasi yang memungkinkan pengiriman peristiwa sebagai file log ke bucket Amazon S3 yang Anda tentukan. CloudTrail file log berisi satu atau lebih entri log. Peristiwa mewakili permintaan tunggal dari sumber manapun dan mencakup informasi tentang tindakan yang diminta, tanggal dan waktu tindakan, parameter permintaan, dan sebagainya. CloudTrail file log bukanlah jejak tumpukan yang diurutkan dari panggilan API publik, sehingga file tersebut tidak muncul dalam urutan tertentu.

Contoh berikut menunjukkan entri CloudTrail log yang menunjukkan `UpdateEnvironment` tindakan yang dipanggil oleh pengguna IAM bernama`intern`, untuk `sample-env` lingkungan dalam aplikasi. `sample-app`

```
{
  "Records": [{
    "eventVersion": "1.05",
    "userIdentity": {
      "type": "IAMUser",
```

```
      "principalId": "AIXDAYQEXAMPLEUMLYNGL",
      "arn": "arn:aws:iam::123456789012:user/intern",
      "accountId": "123456789012",
      "accessKeyId": "ASXIAGXEXAMPLEQULKNXV",
      "userName": "intern",
      "sessionContext": {
        "attributes": {
          "mfaAuthenticated": "false",
          "creationDate": "2016-04-22T00:23:24Z"
        }
      },
      "invokedBy": "signin.amazonaws.com"
    },
    "eventTime": "2016-04-22T00:24:14Z",
    "eventSource": "elasticbeanstalk.amazonaws.com",
    "eventName": "UpdateEnvironment",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "255.255.255.54",
    "userAgent": "signin.amazonaws.com",
    "requestParameters": {
      "applicationName": "sample-app",
      "environmentName": "sample-env",
      "optionSettings": []
    },
    "responseElements": null,
    "requestID": "84ae9ecf-0280-17ce-8612-705c7b132321",
    "eventID": "e48b6a08-c6be-4a22-99e1-c53139cbfb18",
    "eventType": "AwsApiCall",
    "recipientAccountId": "123456789012"
  }]
}
```

Menggunakan Elastic Beanstalk dengan Amazon CloudWatch

Amazon CloudWatch memungkinkan Anda memantau, mengelola, dan mempublikasikan berbagai metrik, serta mengonfigurasi tindakan alarm berdasarkan data dari metrik. CloudWatch Pemantauan Amazon memungkinkan Anda mengumpulkan, menganalisis, dan melihat metrik sistem dan aplikasi sehingga Anda dapat membuat keputusan operasional dan bisnis dengan lebih cepat dan dengan keyakinan yang lebih besar.

Anda dapat menggunakan Amazon CloudWatch untuk mengumpulkan metrik tentang sumber daya Amazon Web Services (AWS) Anda—seperti performa instans Amazon Anda. EC2 Anda juga

dapat mempublikasikan metrik Anda sendiri langsung ke Amazon CloudWatch. CloudWatch Alarm Amazon membantu Anda menerapkan keputusan dengan lebih mudah dengan memungkinkan Anda mengirim pemberitahuan atau secara otomatis membuat perubahan pada sumber daya yang Anda pantau, berdasarkan aturan yang Anda tetapkan. Misalnya, Anda dapat membuat alarm yang memulai tindakan Amazon Auto EC2 Scaling dan Amazon Simple Notification Service (Amazon SNS) Simple Notification Service (Amazon SNS) atas nama Anda.

Elastic Beanstalk secara otomatis CloudWatch menggunakan Amazon untuk membantu Anda memantau status aplikasi dan lingkungan Anda. Anda dapat menavigasi ke CloudWatch konsol Amazon untuk melihat dasbor Anda dan mendapatkan ikhtisar semua sumber daya serta alarm Anda. Anda juga dapat memilih untuk melihat lebih banyak metrik atau menambahkan metrik khusus.

Untuk informasi lebih lanjut tentang Amazon CloudWatch, buka Panduan CloudWatch Pengembang Amazon. Untuk contoh cara menggunakan Amazon CloudWatch dengan Elastic Beanstalk, lihat. the section called “Contoh: Menggunakan CloudWatch metrik Amazon khusus”

Menggunakan Elastic Beanstalk dengan Amazon Logs CloudWatch

Topik ini menjelaskan fitur pemantauan yang dapat disediakan oleh layanan Amazon CloudWatch Logs untuk Elastic Beanstalk. Ini juga memandu Anda melalui pengaturan konfigurasi dan mencantumkan lokasi log untuk setiap platform Elastic Beanstalk.

Menerapkan CloudWatch Log dapat memungkinkan Anda melakukan aktivitas pemantauan berikut:

- Pantau dan arsipkan aplikasi Elastic Beanstalk, sistem, dan file log kustom Anda dari instance EC2 Amazon di lingkungan Anda.
- Konfigurasikan alarm yang memudahkan Anda bereaksi terhadap peristiwa aliran log tertentu yang diekstrak oleh filter metrik Anda.

Agen CloudWatch Log yang diinstal pada setiap EC2 instans Amazon di lingkungan Anda menerbitkan titik data metrik ke CloudWatch layanan untuk setiap grup log yang Anda konfigurasikan. Setiap grup log menerapkan pola filternya sendiri untuk menentukan peristiwa aliran log apa yang akan dikirim CloudWatch sebagai titik data. Log stream yang dimiliki oleh grup log yang sama berbagi pengaturan kontrol penyimpanan, pemantauan, dan akses yang sama. Anda dapat mengonfigurasi Elastic Beanstalk untuk secara otomatis mengalirkan log CloudWatch ke layanan, seperti yang dijelaskan dalam. Streaming log instance ke CloudWatch Log Untuk informasi selengkapnya tentang CloudWatch Log, termasuk terminologi dan konsep, lihat Panduan Pengguna CloudWatch Log Amazon.

Selain log instance, jika Anda mengaktifkan peningkatan kesehatan untuk lingkungan Anda, Anda dapat mengonfigurasi lingkungan untuk mengalirkan informasi kesehatan ke CloudWatch Log. Lihat Streaming informasi kesehatan lingkungan Elastic Beanstalk ke Amazon Logs CloudWatch .

Topik

- Prasyarat untuk melakukan streaming log ke Log CloudWatch
- Bagaimana Elastic Beanstalk mengatur Log CloudWatch
- Streaming log instance ke CloudWatch Log
- Pemecahan Masalah Integrasi Log CloudWatch
- Streaming informasi kesehatan lingkungan Elastic Beanstalk ke Amazon Logs CloudWatch

Prasyarat untuk melakukan streaming log ke Log CloudWatch

Untuk mengaktifkan streaming log dari EC2 instans Amazon lingkungan Anda ke CloudWatch Log, Anda harus memenuhi ketentuan berikut.

- Platform – Karena fitur ini hanya tersedia dalam versi platform yang dirilis pada atau setelah rilis ini, jika Anda menggunakan versi platform sebelumnya, perbarui lingkungan Anda ke yang sekarang.
- Jika Anda tidak memiliki kebijakan yang dikelola AWSElasticBeanstalkWorkerTierElastic Beanstalk AWSElasticBeanstalkWebTieratau Elastic Beanstalk di profil instans Elastic Beanstalk, Anda harus menambahkan yang berikut ini ke profil Anda untuk mengaktifkan fitur ini.

```
{
  "Version": "2012-10-17",
  "Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "logs:PutLogEvents",
      "logs:CreateLogStream"
    ],
    "Resource": [
    "*"
    ]
  }
  ]
}
```

Bagaimana Elastic Beanstalk mengatur Log CloudWatch

Elastic Beanstalk CloudWatch menginstal agen log dengan pengaturan konfigurasi default pada setiap instance yang dibuatnya. Pelajari lebih lanjut di Referensi Agen CloudWatch Log.

Saat Anda mengaktifkan streaming log instance ke CloudWatch Log, Elastic Beanstalk mengirimkan file log dari instance lingkungan Anda ke Log. CloudWatch Platform yang berbeda mengalirkan log yang berbeda. Tabel berikut mencantumkan log, berdasarkan platform.

Platform/Platform Cabang	Log
Docker/ Cabang Platform: Docker Berjalan di Amazon Linux 2 64bit	• /var/log/eb-engine.log • /var/log/eb-hooks.log • /var/log/docker • /var/log/docker-events.log • /var/log/eb-docker/containers/eb-current-app/stdouterr.log • /var/log/nginx/access.log • /var/log/nginx/error.log
Docker/ Cabang Platform: ECS Berjalan di 64bit Amazon Linux 2	• /var/log/docker-events.log • /var/log/eb-ecs-mgr.log • /var/log/eb-engine.log • /var/log/eb-hooks.log • /var/log/ecs/ecs-agent.log • /var/log/ecs/ecs-init.log
Go .NET Core pada Linux Java /Platform Branch: Corretto berjalan di 64bit Amazon Linux 2	• /var/log/eb-engine.log • /var/log/eb-hooks.log • /var/log/web.stdout.log • /var/log/nginx/access.log • /var/log/nginx/error.log
Node.js Python	• /var/log/eb-engine.log • /var/log/eb-hooks.log

Platform/Platform Cabang	Log
	• /var/log/web.stdout.log • /var/log/httpd/access_log • /var/log/httpd/error_log • /var/log/nginx/access.log • /var/log/nginx/error.log
Tomcat PHP	• /var/log/eb-engine.log • /var/log/eb-hooks.log • /var/log/httpd/access_log • /var/log/httpd/error_log • /var/log/nginx/access.log • /var/log/nginx/error.log
.NET pada Windows Server	• C:\inetpub\logs\\ W3 LogFilesSVC1\ u_ex*.log • C:\Program File\ Amazon\ElasticBeanstalk\ log\ AWSDeploy ment .log • C:\Program File\ Amazon\ElasticBeanstalk\ log\ Hooks.log
Ruby	• /var/log/eb-engine.log • /var/log/eb-hooks.log • /var/log/puma/puma.log • /var/log/web.stdout.log • /var/log/nginx/access.log • /var/log/nginx/error.log

Berkas log pada platform Amazon Linux AMI

> **Note**
>
> Pada tanggal 18 Juli 2022, Elastic Beanstalk menetapkan status semua cabang platform berdasarkan Amazon Linux AMI () untuk pensiun. AL1 Untuk informasi selengkapnya tentang

migrasi ke cabang platform Amazon Linux 2023 saat ini dan didukung penuh, lihat. Memigrasi aplikasi Elastic Beanstalk Linux Anda ke Amazon Linux 2023 atau Amazon Linux 2

Tabel berikut mencantumkan berkas log streaming dari instans pada cabang platform yang didasarkan pada Amazon Linux AMI (sebelumnya Amazon Linux 2), oleh platform.

Platform/Platform Cabang	Log
Docker/ Cabang Platform: Docker Berjalan di Amazon Linux 64bit	• /var/log/eb-activity.log • /var/log/nginx/error.log • /var/log/docker-events.log • /var/log/docker • /var/log/nginx/access.log • /var/log/eb-docker/containers/eb-current-app/stdouterr.log
Docker/ Cabang Platform: Multicontainer Docker Berjalan di Amazon Linux 64bit	• /var/log/eb-activity.log • /var/log/ecs/ecs-init.log • /var/log/eb-ecs-mgr.log • /var/log/ecs/ecs-agent.log • /var/log/docker-events.log
Glassfish (Docker yang telah dikonfigurasi)	• /var/log/eb-activity.log • /var/log/nginx/error.log • /var/log/docker-events.log • /var/log/docker • /var/log/nginx/access.log
Go	• /var/log/eb-activity.log • /var/log/nginx/error.log • /var/log/nginx/access.log
Jawa/	• /var/log/eb-activity.log • /var/log/nginx/access.log • /var/log/nginx/error.log

Platform/Platform Cabang	Log
Cabang Platform: Java 8 berjalan di Amazon Linux 64bit Cabang Platform: Java 7 berjalan di Amazon Linux 64bit	• /var/log/web-1.error.log • /var/log/web-1.log
Tomcat	• /var/log/eb-activity.log • /var/log/httpd/error_log • /var/log/httpd/access_log • /var/log/nginx/error_log • /var/log/nginx/access_log
Node.js	• /var/log/eb-activity.log • /var/log/nodejs/nodejs.log • /var/log/nginx/error.log • /var/log/nginx/access.log • /var/log/httpd/error.log • /var/log/httpd/access.log
PHP	• /var/log/eb-activity.log • /var/log/httpd/error_log • /var/log/httpd/access_log
Python	• /var/log/eb-activity.log • /var/log/httpd/error_log • /var/log/httpd/access_log • /opt/python/log/supervisord.log
Ruby/ Cabang Platform: Puma dengan Ruby berjalan di 64bit Amazon Linux	• /var/log/eb-activity.log • /var/log/nginx/error.log • /var/log/puma/puma.log • /var/log/nginx/access.log

Platform/Platform Cabang	Log
Ruby/ Cabang Platform: Penumpang dengan Ruby berjalan di Amazon Linux 64bit	• /var/log/eb-activity.log • /var/app/support/logs/passenger.log • /var/app/support/logs/access.log • /var/app/support/logs/error.log

Elastic Beanstalk mengonfigurasi CloudWatch grup log di Log untuk berbagai file log yang dialirkannya. Untuk mengambil file log tertentu dari CloudWatch Log, Anda harus mengetahui nama grup log yang sesuai. Skema penamaan grup log tergantung pada sistem operasi platform.

Untuk platform Linux, prefiks lokasi berkas log pada instans dengan `/aws/elasticbeanstalk/`*`environment_name`* untuk mendapatkan nama grup log. Misalnya, untuk mengambil file `/var/log/nginx/error.log`, tentukan grup log `/aws/elasticbeanstalk/`*`environment_name`*`/var/log/nginx/error.log`.

Untuk Windows platform, lihat tabel berikut untuk grup log yang sesuai untuk setiap berkas log.

Berkas log pada instans	Grup log
`C:\Program Files\Amazon\ElasticBeanstalk\logs\AWSDeployment.log`	`/aws/elasticbeanstalk/<environment-name>/EBDeploy-Log`
`C:\Program Files\Amazon\ElasticBeanstalk\logs\Hooks.log`	`/aws/elasticbeanstalk/<environment-name>/EBHooks-Log`
`C:\inetpub\logs\LogFiles` (seluruh direktori)	`/aws/elasticbeanstalk/<environment-name>/IIS-Log`

Streaming log instance ke CloudWatch Log

Anda dapat mengaktifkan streaming log instance ke CloudWatch Log menggunakan konsol Elastic Beanstalk, EB CLI, atau opsi konfigurasi.

Sebelum Anda mengaktifkannya, siapkan izin IAM untuk digunakan dengan agen CloudWatch Log. Anda dapat melampirkan kebijakan kustom berikut untuk profil instans yang Anda tetapkan ke lingkungan Anda.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "logs:CreateLogStream",
        "logs:PutLogEvents",
        "logs:DescribeLogGroups",
        "logs:DescribeLogStreams"
      ],
      "Resource": [
        "*"
      ]
    }
  ]
}
```

Streaming log instans menggunakan konsol Elastic Beanstalk

Untuk mengalirkan log instance ke CloudWatch Log

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Dalam kategori konfigurasi Pembaruan, pemantauan, dan pencatatan, pilih Edit.
5. Di bawah Streaming log Instance ke CloudWatch Log:
   - Mengaktifkan Streaming log.
   - Mengatur Retensi untuk jumlah hari untuk menyimpan log.
   - Pilih pengaturan Siklus Hidup yang menentukan apakah log disimpan setelah lingkungan dihentikan.
6. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Setelah Anda mengaktifkan streaming log, Anda dapat kembali ke kategori atau halaman konfigurasi Perangkat Lunak dan menemukan tautan Grup Log. Klik tautan ini untuk melihat log Anda di CloudWatch konsol.

Streaming log instans menggunakan EB CLI

Untuk mengaktifkan streaming log instance ke CloudWatch Log menggunakan EB CLI, gunakan eb logs perintah.

```
$ eb logs --cloudwatch-logs enable
```

Anda juga dapat menggunakan eb logs untuk mengambil log dari CloudWatch Log. Anda dapat mengambil semua log instans lingkungan, atau menggunakan banyak pilihan perintah untuk menentukan himpunan bagian dari log untuk mengambil. Sebagai contoh, perintah berikut mengambil set lengkap dari log instans untuk lingkungan Anda, dan menyimpannya ke direktori di bawah `.elasticbeanstalk/logs`.

```
$ eb logs --all
```

Khususnya, opsi `--log-group` memungkinkan Anda untuk mengambil log instans dari grup log tertentu, sesuai dengan berkas log instans tertentu. Untuk melakukannya, Anda perlu tahu nama grup log yang sesuai dengan berkas log yang ingin Anda ambil. Anda dapat menemukan informasi ini di Bagaimana Elastic Beanstalk mengatur Log CloudWatch .

Streaming log instans menggunakan file konfigurasi

Saat membuat atau memperbarui lingkungan, Anda dapat menggunakan file konfigurasi untuk mengatur dan mengonfigurasi streaming log instance ke CloudWatch Log. File konfigurasi contoh berikut memungkinkan streaming log instans default. Elastic Beanstalk mengalirkan set default berkas log untuk platform lingkungan Anda. Untuk menggunakan contoh, salin teks ke dalam file dengan ekstensi `.config` di direktori `.ebextensions` di tingkat atas paket sumber aplikasi Anda.

```
option_settings:
  - namespace: aws:elasticbeanstalk:cloudwatch:logs
    option_name: StreamLogs
    value: true
```

Streaming berkas log kustom

CloudWatch Integrasi Elastic Beanstalk dengan Log tidak secara langsung mendukung streaming file log khusus yang dihasilkan aplikasi Anda. Untuk melakukan streaming log kustom, gunakan file konfigurasi untuk langsung menginstal CloudWatch agen dan untuk mengkonfigurasi file yang akan didorong. Untuk file konfigurasi contoh, lihat `logs-streamtocloudwatch-linux.config`.

> **Note**
>
> Contoh tidak bekerja pada platform Windows.

Untuk informasi selengkapnya tentang mengonfigurasi CloudWatch Log, lihat referensi file konfigurasi CloudWatch agen di Panduan CloudWatch Pengguna Amazon.

Pemecahan Masalah Integrasi Log CloudWatch

Tidak dapat menemukan log contoh lingkungan

Jika Anda tidak dapat menemukan beberapa log instance lingkungan yang Anda harapkan di CloudWatch Log, selidiki masalah umum berikut:

- IAM role Anda tidak memiliki izin IAM yang diperlukan.
- Anda meluncurkan lingkungan Anda di sebuah Wilayah AWS yang tidak mendukung CloudWatch Log.
- Salah satu berkas log kustom Anda tidak ada di jalur yang Anda tentukan.

Log aplikasi hilang atau terputus-putus

Jika log aplikasi Elastic Beanstalk Anda`/var/log/web.stdout.log`, (), tampak hilang atau terputus-putus, ini mungkin karena pengaturan pembatas laju default di rsyslog dan journald. Meskipun menonaktifkan pembatasan laju sepenuhnya dapat menyelesaikan masalah ini, itu tidak disarankan karena dapat menyebabkan penggunaan disk yang berlebihan, potensi penolakan layanan, atau penurunan kinerja sistem selama ledakan log yang tidak terduga. Sebagai gantinya, Anda dapat menyesuaikan batas tarif menggunakan yang berikut ini `.ebextensions configuration`. Konfigurasi ini meningkatkan interval batas laju menjadi 600 detik dengan batas burst yang lebih tinggi, memberikan keseimbangan antara logging yang tepat dan perlindungan sistem.

Masalah pelambatan

Jika operasi Elastic Beanstalk yang secara bersamaan meluncurkan sejumlah besar instance mengembalikan `Error: fail to create log stream: ThrottlingException: Rate exceeded` pesan seperti, itu melambat dari terlalu banyak panggilan ke API. CloudWatch

Untuk mengatasi masalah pelambatan, lakukan salah satu tindakan berikut:

- Gunakan ukuran batch yang lebih kecil dengan penerapan bergulir untuk mengurangi pembaruan bersamaan.
- Minta kenaikan kuota layanan limit Transaksi Per Detik (TPS) AWS akun Anda. CreateLogStream Untuk informasi selengkapnya, lihat Kuota CloudWatch log dan Mengelola kuota layanan CloudWatch Log Anda di Panduan Pengguna Amazon CloudWatch Logs.

Streaming informasi kesehatan lingkungan Elastic Beanstalk ke Amazon Logs CloudWatch

Jika Anda mengaktifkan pelaporan kesehatan yang disempurnakan untuk lingkungan Anda, Anda dapat mengonfigurasi lingkungan untuk mengalirkan informasi kesehatan ke CloudWatch Log. Streaming ini independen dari streaming log EC2 instans Amazon. Topik ini menjelaskan streaming informasi kondisi lingkungan. Untuk informasi tentang streaming log instans, lihat Menggunakan Elastic Beanstalk dengan Amazon Logs CloudWatch .

Saat Anda mengonfigurasi streaming kesehatan lingkungan, Elastic Beanstalk CloudWatch membuat grup log Log untuk kesehatan lingkungan. Nama grup log adalah `/aws/elasticbeanstalk/`*`environment-name`*`/environment-health.log`. Dalam kelompok log ini, Elastic Beanstalk menciptakan aliran log bernama *`YYYY-MM-DD`*`#`*`<hash-suffix>`* (mungkin ada lebih dari satu aliran log per tanggal).

Ketika status kondisi lingkungan berubah, Elastic Beanstalk menambahkan catatan untuk aliran log kondisi. Catatan mewakili transisi status kondisi—status baru dan deskripsi penyebab perubahan. Sebagai contoh, status lingkungan mungkin berubah menjadi parah karena penyeimbang beban gagal. Untuk deskripsi status kondisi yang disempurnakan, lihat Warna dan status kondisi.

Prasyarat untuk streaming kesehatan lingkungan ke Log CloudWatch

Untuk mengaktifkan streaming kesehatan lingkungan ke CloudWatch Log, Anda harus memenuhi ketentuan berikut:

- Platform – Anda harus menggunakan versi platform yang mendukung peningkatan pelaporan kondisi.
- Izin – Anda harus memberikan izin terkait logging tertentu ke Elastic Beanstalk sehingga dapat bertindak atas nama Anda untuk mengalirkan informasi kondis untuk lingkungan Anda. Jika lingkungan Anda tidak menggunakan peran layanan yang dibuat Elastic Beanstalk untuk itu, `aws-elasticbeanstalk-service-role`, atau peran tertaut layanan akun Anda, `AWSServiceRoleForElasticBeanstalk`, pastikan untuk menambahkan izin berikut ke peran layanan kustom Anda.

```
{
      "Effect": "Allow",
      "Action": [
        "logs:DescribeLogStreams",
        "logs:CreateLogStream",
        "logs:PutLogEvents"
      ],
      "Resource": "arn:aws:logs:*:*:log-group:/aws/elasticbeanstalk/*:log-stream:*"
}
```

Streaming log kesehatan lingkungan ke CloudWatch Log

Anda dapat mengaktifkan streaming kesehatan lingkungan ke CloudWatch Log menggunakan konsol Elastic Beanstalk, EB CLI, atau opsi konfigurasi.

Streaming log kondisi lingkungan menggunakan konsol Elastic Beanstalk

Untuk mengalirkan log kesehatan lingkungan ke CloudWatch Log

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Di kategori konfigurasi Pemantauan, pilih Edit.
5. Di bawah Pelaporan kondisi, pastikan bahwa pelaporan Sistem diatur ke Ditingkatkan.
6. Di bawah streaming acara Kesehatan ke CloudWatch Log
   - Mengaktifkan Streaming log.
   - Mengatur Retensi untuk jumlah hari untuk menyimpan log.

- Pilih pengaturan Siklus Hidup yang menentukan apakah log disimpan setelah lingkungan dihentikan.

7. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Setelah Anda mengaktifkan streaming log, Anda dapat kembali ke kategori atau halaman konfigurasi Pemantauan dan menemukan tautan Grup Log. Klik tautan ini untuk melihat log kesehatan lingkungan Anda di CloudWatch konsol.

Streaming log kondisi lingkungan menggunakan EB CLI

Untuk mengaktifkan streaming log kesehatan lingkungan ke CloudWatch Log menggunakan EB CLI, gunakan eb logs perintah.

```
$ eb logs --cloudwatch-logs enable --cloudwatch-log-source environment-health
```

Anda juga dapat menggunakan eb logs untuk mengambil log dari CloudWatch Log. Sebagai contoh, perintah berikut mengambil semua log kondisi untuk lingkungan Anda, dan menyimpannya ke direktori di bawah `.elasticbeanstalk/logs`.

```
$ eb logs --all --cloudwatch-log-source environment-health
```

Streaming log kondisi lingkungan menggunakan file konfigurasi

Saat membuat atau memperbarui lingkungan, Anda dapat menggunakan file konfigurasi untuk mengatur dan mengonfigurasi streaming log kesehatan lingkungan ke CloudWatch Log. Untuk menggunakan contoh di bawah ini, salin teks ke dalam file dengan ekstensi `.config` di direktori `.ebextensions` di tingkat atas paket sumber aplikasi Anda. Contoh tersebut mengonfigurasi Elastic Beanstalk untuk mengaktifkan streaming log kondisi lingkungan, menjaga log setelah mengakhiri lingkungan, dan menyimpannya selama 30 hari.

Example File konfigurasi streaming Kesehatan

```
#############################################################################
##  Sets up Elastic Beanstalk to stream environment health information
##  to Amazon CloudWatch Logs.
##  Works only for environments that have enhanced health reporting enabled.
#############################################################################

option_settings:
```

```
  aws:elasticbeanstalk:cloudwatch:logs:health:
    HealthStreamingEnabled: true
    ### Settings below this line are optional.
    # DeleteOnTerminate: Delete the log group when the environment is
    # terminated. Default is false. If false, the health data is kept
    # RetentionInDays days.
    DeleteOnTerminate: false
    # RetentionInDays: The number of days to keep the archived health data
    # before it expires, if DeleteOnTerminate isn't set. Default is 7 days.
    RetentionInDays: 30
```

Untuk pilihan default dan nilai yang valid, lihat `aws:elasticbeanstalk:cloudwatch:logs:health`.

Menggunakan Elastic Beanstalk dengan Amazon EventBridge

Menggunakan Amazon EventBridge, Anda dapat mengatur aturan berbasis peristiwa yang memantau sumber daya Elastic Beanstalk Anda dan memulai tindakan target yang menggunakan layanan lain. AWS Misalnya, Anda dapat menetapkan aturan untuk mengirimkan pemberitahuan email dengan menandakan topik Amazon SNS setiap kali kondisi dari lingkungan produksi berubah menjadi status Peringatan. Atau, Anda dapat mengatur fungsi Lambda untuk menyampaikan pemberitahuan ke Slack setiap kali kondisi lingkungan Anda berubah menjadi status Berdegradasi atau Parah.

Anda dapat membuat aturan di Amazon EventBridge untuk bertindak atas salah satu acara Elastic Beanstalk berikut:

- Perubahan status untuk operasi lingkungan (termasuk membuat, memperbarui, dan menghentikan operasi). Peristiwa menentukan apakah perubahan keadaan telah dimulai, berhasil, atau gagal.
- Perubahan negara untuk sumber daya lain.Selain lingkungan, sumber daya lain yang dipantau meliputi penyeimbang beban, grup auto scaling, dan instans.
- Transisi Kesehatan untuk Lingkungan Peristiwa ini menyatakan di mana kondisi lingkungan telah beralih dari satu status kondisi ke status kondisi lainnya.
- Perubahan status untuk pembaruan terkelola. Peristiwa menentukan apakah perubahan keadaan telah dimulai, berhasil, atau gagal.

Untuk menangkap peristiwa Elastic Beanstalk tertentu yang Anda minati, tentukan EventBridge pola spesifik peristiwa yang dapat digunakan untuk mendeteksi peristiwa. Pola peristiwa memiliki struktur

yang sama dengan peristiwa mereka cocokkan. Pola mengutip bidang yang ingin Anda cocokkan dan memberikan nilai yang Anda cari. Peristiwa dipancarkan atas dasar upaya terbaik. Mereka dikirim dari Elastic EventBridge Beanstalk ke hampir real-time dalam keadaan operasional normal. Namun, situasi dapat timbul yang dapat menunda atau mencegah pengiriman suatu peristiwa.

Untuk daftar bidang yang terkandung dalam peristiwa Elastic Beanstalk dan mungkin nilai-nilai string mereka, lihat Pemetaan bidang peristiwa Elastic Beanstalk. Untuk informasi tentang cara kerja EventBridge aturan dengan pola peristiwa, lihat Peristiwa dan Pola Peristiwa di EventBridge.

Pantau sumber daya Elastic Beanstalk dengan EventBridge

Dengan EventBridge, Anda dapat membuat aturan yang menentukan tindakan yang harus diambil saat Elastic Beanstalk memancarkan peristiwa untuk sumber dayanya. Misalnya, Anda dapat membuat aturan yang mengirimkan pesan email kapan pun status lingkungan berubah.

EventBridge Konsol memiliki opsi pola yang telah ditentukan sebelumnya untuk membangun pola acara Elastic Beanstalk. Jika Anda memilih opsi ini di EventBridge konsol saat membuat aturan, Anda dapat membuat pola acara Elastic Beanstalk dengan cepat. Anda hanya perlu memilih bidang peristiwa dan nilai. Ketika Anda membuat pilihan, konsol membangun dan menampilkan pola peristiwa. Atau, Anda dapat secara manual mengedit pola peristiwa yang Anda bangun dan dapat menyimpannya sebagai pola kustom. Konsol juga memberi Anda opsi untuk menampilkan Peristiwa Contoh terperinci yang dapat Anda salin dan tempel ke pola acara yang sedang Anda bangun.

Jika Anda lebih suka mengetik atau menyalin dan menempelkan pola acara ke EventBridge konsol, Anda dapat memilih untuk menggunakan opsi Pola kustom di konsol. Dengan melakukan ini, Anda tidak perlu pergi melalui langkah-langkah memilih bidang dan nilai-nilai yang dijelaskan sebelumnya. Topik ini menawarkan contoh kedua pola pencocokan peristiwa dan peristiwa Elastic Beanstalk yang dapat Anda gunakan.

Untuk membuat aturan untuk peristiwa sumber daya

1. Masuk untuk AWS menggunakan akun yang memiliki izin untuk menggunakan EventBridge dan Elastic Beanstalk.
2. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
3. Di panel navigasi, pilih Aturan.
4. Pilih Buat aturan.
5. Masukkan Nama untuk aturan tersebut, dan, secara opsional, deskripsi.

6. Untuk Bus peristiwa, pilih default. Ketika AWS layanan di akun Anda memancarkan suatu acara, itu selalu masuk ke bus acara default akun Anda.
7. Untuk Tipe aturan, pilih Aturan dengan pola peristiwa.
8. Pilih Selanjutnya.
9. Untuk sumber Acara, pilih AWS acara atau acara EventBridge mitra.
10. (Opsional) Untuk acara Contoh, pilih AWS acara. Masukkan Elastic Beanstalk di kolom pencarian. Ini akan memberikan daftar contoh acara Elastic Beanstalk yang dapat Anda pilih untuk ditampilkan. Langkah ini hanya menampilkan contoh peristiwa yang dapat Anda referensikan. Itu tidak mempengaruhi hasil pembuatan aturan. Contoh peristiwa Elastic BeanstalkBagian nanti dalam topik ini memberikan contoh jenis peristiwa yang sama.
11. Di bagian Pola acara, pilih Formulir pola acara.

> **Note**
>
> Jika Anda sudah memiliki teks untuk pola acara dan tidak memerlukan EventBridge konsol untuk membangunnya untuk Anda, pilih Pola kustom (editor JSON). Anda kemudian dapat memasukkan atau menyalin dan menempelkan teks secara manual ke dalam kotak Pola acara. Pilih Berikutnya, dan lanjutkan ke langkah tentang memasukkan target.

12. Untuk Sumber peristiwa, pilih Layanan AWS .
13. Untuk AWS servis, pilih Elastic Beanstalk.
14. Untuk jenis Acara, pilih Perubahan Status.
15. Langkah ini mencakup bagaimana Anda dapat bekerja dengan jenis detail, status, dan tingkat kepelikan bidang peristiwa untuk Elastic Beanstalk. Ketika Anda memilih bidang ini dan nilai-nilai yang ingin Anda cocokkan, konsol membangun dan menampilkan pola peristiwa.

    - Jika Anda memilih hanya satu nilai untuk Jenis detail spesifik, Anda dapat memilih satu atau beberapa nilai untuk bidang berikutnya dalam hierarki.
    - Jika Anda memilih lebih dari satu nilai untuk Jenis detail spesifik, jangan memilih nilai spesifik untuk bidang berikutnya dalam hierarki. Ini mencegah logika pencocokan ambigu di seluruh bidang dalam pola acara Anda.

    Bidang peristiwa lingkungan tidak terpengaruh oleh hirarki ini, sehingga menampilkan seperti yang dijelaskan di langkah berikutnya.

16. Untuk lingkungan, pilih Lingkungan apa pun atau Lingkungan tertentu.

    - Jika Anda memilih Lingkungan tertentu, Anda dapat memilih satu atau beberapa lingkungan dari daftar dropdown. EventBridge menambahkan semua lingkungan yang Anda pilih di dalam daftar EnvironmentName[] di bagian detail dari pola acara. Kemudian, aturan Anda memfilter semua peristiwa untuk mencakup hanya lingkungan tertentu yang Anda pilih.
    - Jika Anda memilih Lingkungan apa pun, maka tidak ada lingkungan yang ditambahkan ke pola peristiwa Anda. Karena itu, aturan Anda tidak memfilter salah satu peristiwa Elastic Beanstalk berdasarkan lingkungan.

17. Pilih Berikutnya.
18. Untuk Jenis target, pilih Layanan AWS .
19. Untuk Pilih target, pilih tindakan target yang akan diambil ketika peristiwa perubahan status sumber daya diterima dari Elastic Beanstalk.

    Misalnya, Anda dapat menggunakan topik Amazon Simple Notification Service (SNS) untuk mengirim email atau pesan teks ketika peristiwa terjadi. Untuk melakukannya, Anda harus terlebih dahulu membuat topik Amazon SNS menggunakan konsol Amazon SNS. Untuk mempelajari lebih lanjut, lihat Menggunakan Amazon SNS untuk pemberitahuan pengguna.

    > **Important**
    >
    > Beberapa tindakan target mungkin memerlukan penggunaan layanan lain dan dikenakan biaya tambahan, seperti layanan Amazon SNS atau Lambda. Untuk informasi lebih lanjut tentang AWS harga, lihathttps://aws.amazon.com/pricing/. Beberapa layanan adalah bagian dari Tingkat Penggunaan AWS Gratis. Jika Anda adalah pelanggan baru, Anda dapat menguji layanan ini secara gratis. Lihat https://aws.amazon.com/free/ untuk informasi selengkapnya.

20. (Opsional) Pilih Tambahkan target lain untuk menentukan tindakan target tambahan untuk aturan acara.
21. Pilih Berikutnya.
22. (Opsional) Masukkan satu atau lebih tanda untuk aturan. Untuk informasi selengkapnya, lihat EventBridge tag Amazon di Panduan EventBridge Pengguna Amazon.
23. Pilih Berikutnya.
24. Tinjau detail aturan dan pilih Buat aturan.

Contoh pola peristiwa Elastic Beanstalk

Pola peristiwa memiliki struktur yang sama dengan peristiwa mereka cocokkan. Pola mengutip bidang yang ingin Anda cocokkan dan memberikan nilai yang Anda cari.

- Perubahan status kondisi untuk semua lingkungan

```
{
   "source": [
    "aws.elasticbeanstalk"
  ],
  "detail-type": [
    "Health status change"
    ]
}
```

- Perubahan status kondisi untuk lingkungan berikut: `myEnvironment1` dan `myEnvironment2`. Pola peristiwa ini memfilter untuk dua lingkungan tertentu, sedangkan sebelumnya contoh Perubahan status kondisi yang tidak memfilter mengirimkan peristiwa untuk semua lingkungan.

```
{"source": [
    "aws.elasticbeanstalk"
    ],
    "detail-type": [
        "Health status change"
    ],
    "detail": {
        "EnvironmentName": [
            "myEnvironment1",
            "myEnvironment2"
        ]
    }
}
```

- Perubahan status sumber daya Elastic Beanstalk untuk semua lingkungan

```
{
  "source": [
    "aws.elasticbeanstalk"
  ],
  "detail-type": [
    "Elastic Beanstalk resource status change"
```

```
    ]
}
```

- Perubahan status sumber daya Elastic Beanstalk dengan Pembaruan lingkungan *`Status`* gagal dan KESALAHAN `Severity` untuk lingkungan berikut: `myEnvironment1` dan `myEnvironment2`

```
{"source": [
    "aws.elasticbeanstalk"
    ],
    "detail-type": [
        "Elastic Beanstalk resource status change"
    ],
    "detail": {
        "Status": [
            "Environment update failed"
            ],
        "Severity": [
            "ERROR"
            ],
        "EnvironmentName": [
            "myEnvironment1",
            "myEnvironment2"
        ]
    }
}
```

- Perubahan status sumber daya lainnya untuk penyeimbang beban, grup auto scaling, dan instans

```
{
   "source": [
    "aws.elasticbeanstalk"
  ],
  "detail-type": [
    "Other resource status change"
    ]
}
```

- Perubahan status pembaruan yang terkelola untuk semua lingkungan

```
{
   "source": [
    "aws.elasticbeanstalk"
  ],
```

```
  "detail-type": [
    "Managed update status change"
    ]
}
```

- Untuk menangkap Semua peristiwa dari Elastic Beanstalk (tidak termasuk bagian `detail-type`)

```
{
  "source": [
    "aws.elasticbeanstalk"
  ]
}
```

Contoh peristiwa Elastic Beanstalk

Berikut ini adalah kejadian Elastic Beanstalk untuk perubahan status sumber daya:

```
{
   "version":"0",
   "id":"1234a678-1b23-c123-12fd3f456e78",
   "detail-type":"Elastic Beanstalk resource status change",
   "source":"aws.elasticbeanstalk",
   "account":"111122223333",
   "time":"2020-11-03T00:31:54Z",
   "region":"us-east-1",
   "resources":[
      "arn:was:elasticbeanstalk:us-east-1:111122223333:environment/myApplication/
myEnvironment"
   ],
   "detail":{
      "Status":"Environment creation started",
      "EventDate":1604363513951,
      "ApplicationName":"myApplication",
      "Message":"createEnvironment is starting.",
      "EnvironmentName":"myEnvironment",
      "Severity":"INFO"
   }
}
```

Berikut ini adalah kejadian Elastic Beanstalk untuk perubahan status kondisi:

```
{
   "version":"0",
   "id":"1234a678-1b23-c123-12fd3f456e78",
   "detail-type":"Health status change",
   "source":"aws.elasticbeanstalk",
   "account":"111122223333",
   "time":"2020-11-03T00:34:48Z",
   "region":"us-east-1",
   "resources":[
      "arn:was:elasticbeanstalk:us-east-1:111122223333:environment/myApplication/
myEnvironment"
   ],
   "detail":{
      "Status":"Environment health changed",
      "EventDate":1604363687870,
      "ApplicationName":"myApplication",
      "Message":"Environment health has transitioned from Pending to Ok. Initialization
 completed 1 second ago and took 2 minutes.",
      "EnvironmentName":"myEnvironment",
      "Severity":"INFO"
   }
}
```

Pemetaan bidang peristiwa Elastic Beanstalk

Tabel berikut memetakan bidang acara Elastic Beanstalk dan nilai string yang mungkin ke lapangan. EventBridge `detail-type` Untuk informasi selengkapnya tentang cara EventBridge bekerja dengan pola acara untuk suatu layanan, lihat Peristiwa dan Pola Peristiwa di EventBridge.

EventBridge tipe detail bidang	Status Bidang Elastic Beanstalk	Tingkat Keparahan Bidang Elastic Beanstalk	Pesan bidang Elastic Beanstalk
Perubahan status sumber daya	Pembuatan lingkungan dimulai	INFO	createEnvironment dimulai.

EventBridge tipe detail bidang	Status Bidang Elastic Beanstalk	Tingkat Keparahan Bidang Elastic Beanstalk	Pesan bidang Elastic Beanstalk
Elastic Beanstalk	Pembuatan lingkungan berhasil	INFO	createEnvironment berhasil diselesaikan.
	Pembuatan lingkungan berhasil	INFO	Lingkungan yang diluncurkan: <Environment Name>. Namun, ada masalah saat peluncuran. Lihat log peristiwa untuk rincian.
	Pembuatan lingkungan gagal	KESALAHAN	Gagal meluncurkan lingkungan.
	Pembaruan lingkungan dimulai	INFO	Pemutakhiran lingkungan dimulai.
	Memperbarui lingkungan berhasil	INFO	Pemutakhiran lingkungan berhasil diselesaikan.
	Pembaruan lingkungan gagal	KESALAHAN	Gagal men-deploy konfigurasi.
	Penghentian lingkungan dimulai	INFO	terminateEnvironment dimulai.

EventBridge tipe detail bidang	Status Bidang Elastic Beanstalk	Tingkat Keparahan Bidang Elastic Beanstalk	Pesan bidang Elastic Beanstalk
	Penghentian lingkungan berhasil	INFO	terminateEnvironment berhasil diselesaikan.
	Penghentian lingkungan gagal	INFO	Langkah penghentian lingkungan gagal karena setidaknya salah satu alur kerja penghentian lingkungan gagal.
Perubahan status sumber daya lainnya	grup Auto Scaling dibuat	INFO	createEnvironment dimulai.
	Grup Auto Scaling dihapus	INFO	createEnvironment dimulai.
	Instans ditambahkan	INFO	Menambahkan instans [i-123456789a12b1234] ke lingkungan Anda.
	Instans dihapus	INFO	Instans dihapus [i-123456789a12b1234] dari lingkungan Anda.
	Penyeimbang beban dibuat	INFO	Penyeimbang beban yang dibuat bernama: <LB Name>
	Penyeimbang beban dihapus	INFO	Penyeimbang beban yang dihapus bernama: <LB Name>

EventBridge tipe detail bidang	Status Bidang Elastic Beanstalk	Tingkat Keparahan Bidang Elastic Beanstalk	Pesan bidang Elastic Beanstalk
Perubahan status kondisi	Lingkungan kondisi diubah	INFO/ WARN	Kondisi lingkungan telah beralih ke <healthStatus>.
	Lingkungan kondisi diubah	INFO/ WARN	Kondisi lingkungan telah beralih dari <healthStatus> ke <healthStatus>.
Perubahan status pembaruan yang terkelola	Dikelola diperbarui dimulai	INFO	Pembaruan platform yang terkelola sedang berlangsung.
	Pembaruan yang terkelola gagal	INFO	Pembaruan yang terkelola gagal, mencoba kembali dalam % menit.

Menemukan dan melacak sumber daya Elastic Beanstalk dengan AWS Config

AWS Configmemberikan tampilan terperinci tentang konfigurasi AWS sumber daya di AWS akun Anda. Anda dapat melihat bagaimana sumber daya terkait, mendapatkan riwayat perubahan konfigurasi, dan melihat bagaimana hubungan dan konfigurasi berubah seiring waktu. Anda dapat menggunakan AWS Config untuk menentukan aturan yang mengevaluasi konfigurasi sumber daya untuk kepatuhan data.

Beberapa jenis sumber daya Elastic Beanstalk terintegrasi dengan: AWS Config

- Aplikasi
- Versi Aplikasi
- Lingkungan

Bagian berikut menunjukkan cara mengkonfigurasi AWS Config untuk merekam sumber daya dari jenis ini.

Untuk informasi selengkapnya AWS Config, lihat Panduan AWS Config Pengembang. Untuk informasi harga, lihat halaman informasi harga AWS Config.

Menyiapkan AWS Config

Untuk mengatur awalnya AWS Config, lihat topik berikut di Panduan AWS Config Pengembang.

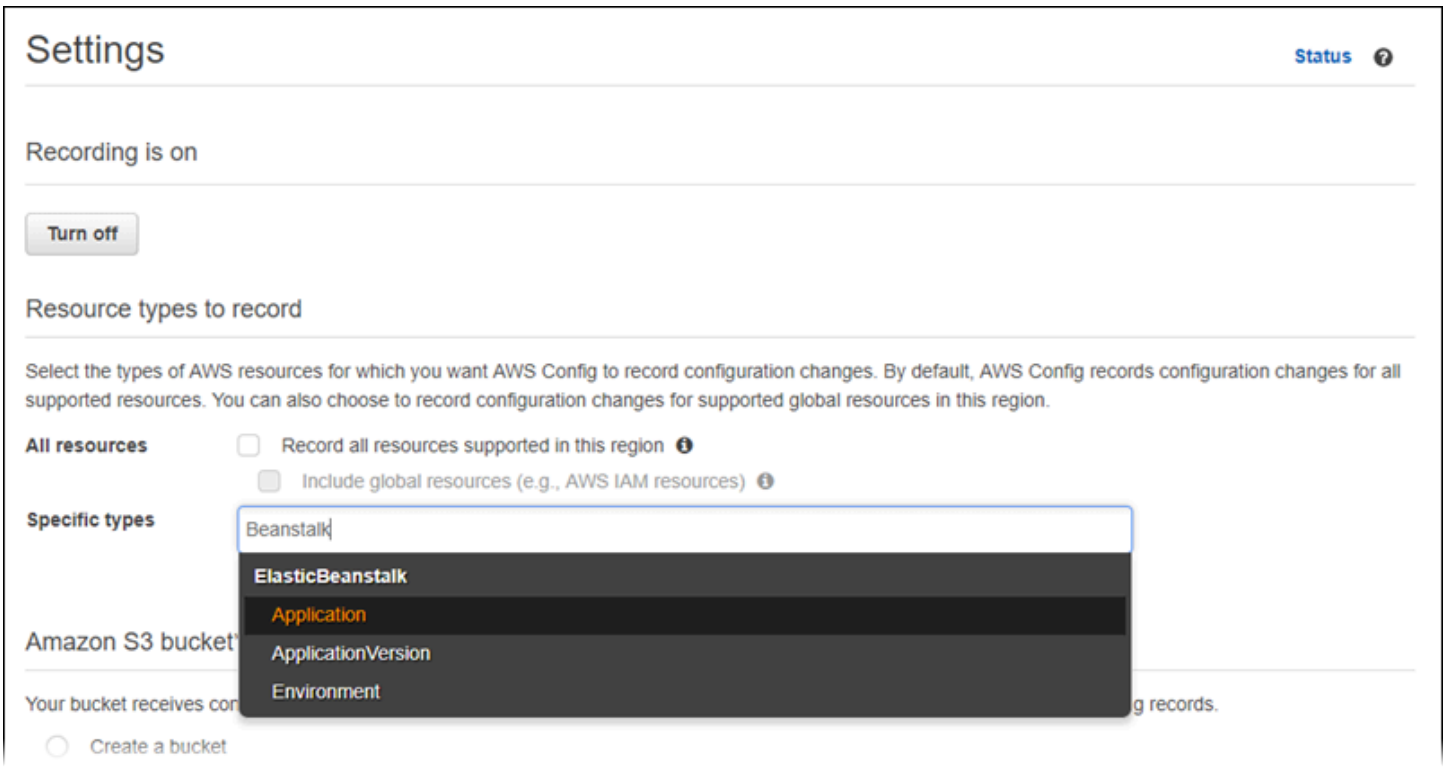
- Menyiapkan AWS Config dengan Konsol
- Menyiapkan AWS Config dengan AWS CLI

Mengkonfigurasi AWS Config untuk merekam sumber daya Elastic Beanstalk

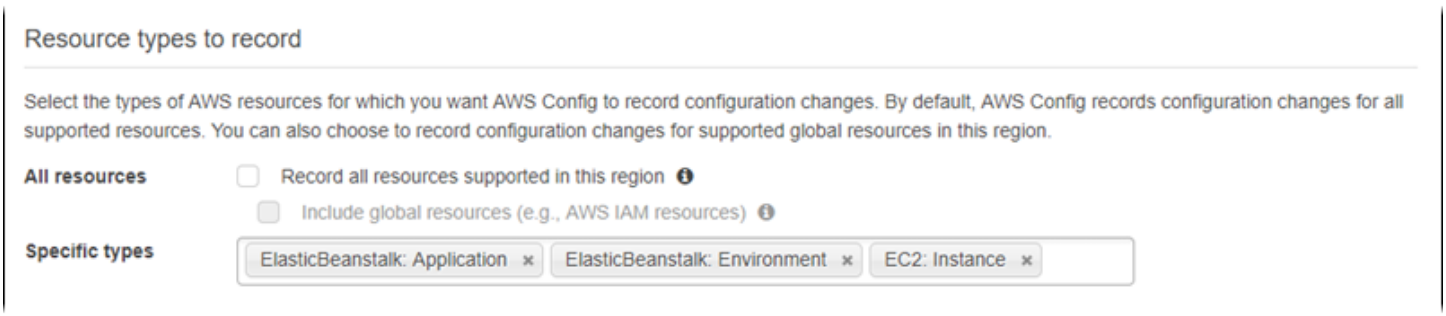
Secara default, AWS Config mencatat perubahan konfigurasi untuk semua jenis sumber daya regional yang didukung yang ditemukan di wilayah tempat lingkungan Anda berjalan. Anda dapat menyesuaikan AWS Config untuk merekam perubahan hanya untuk jenis sumber daya tertentu, atau perubahan pada sumber daya global.

Misalnya, Anda dapat mengonfigurasi AWS Config untuk merekam perubahan sumber daya Elastic Beanstalk dan subset sumber AWS daya lain yang dimulai Elastic Beanstalk untuk Anda. Menggunakan AWS Config Konsol, Anda dapat memilih Elastic Beanstalk sebagai sumber daya AWS Config di halaman Pengaturan dari bidang Jenis Spesifik. Dari sana Anda dapat memilih untuk merekam salah satu jenis sumber daya Elastic Beanstalk: ApplicationVersionAplikasi,, dan Lingkungan.

Gambar berikut menunjukkan halaman AWS Config Pengaturan, dengan jenis sumber daya Elastic Beanstalk yang dapat Anda pilih untuk direkam: ApplicationVersionAplikasi,, dan Lingkungan.



Setelah Anda memilih beberapa jenis sumber daya, ini adalah bagaimana daftar Jenis spesifik muncul.



Untuk mempelajari tentang sumber daya regional vs global, dan untuk prosedur penyesuaian lengkap, lihat Memilih AWS Config Rekaman Sumber Daya yang mana.

Melihat detail konfigurasi Elastic Beanstalk di konsol AWS Config

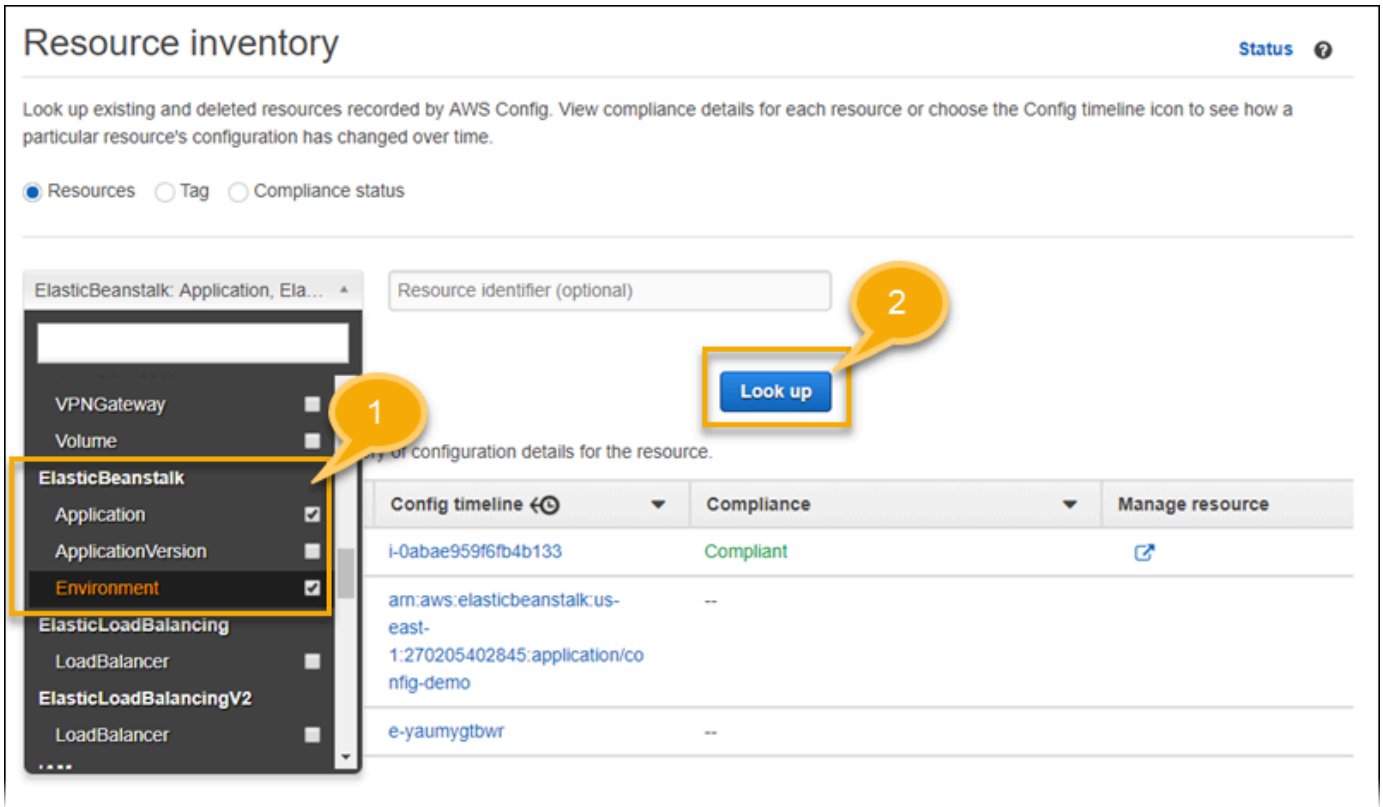
Anda dapat menggunakan AWS Config konsol untuk mencari sumber daya Elastic Beanstalk, dan mendapatkan detail terkini dan historis tentang konfigurasinya. Contoh berikut menunjukkan cara menemukan informasi tentang lingkungan Elastic Beanstalk.

Untuk menemukan lingkungan Elastic Beanstalk di konsol AWS Config

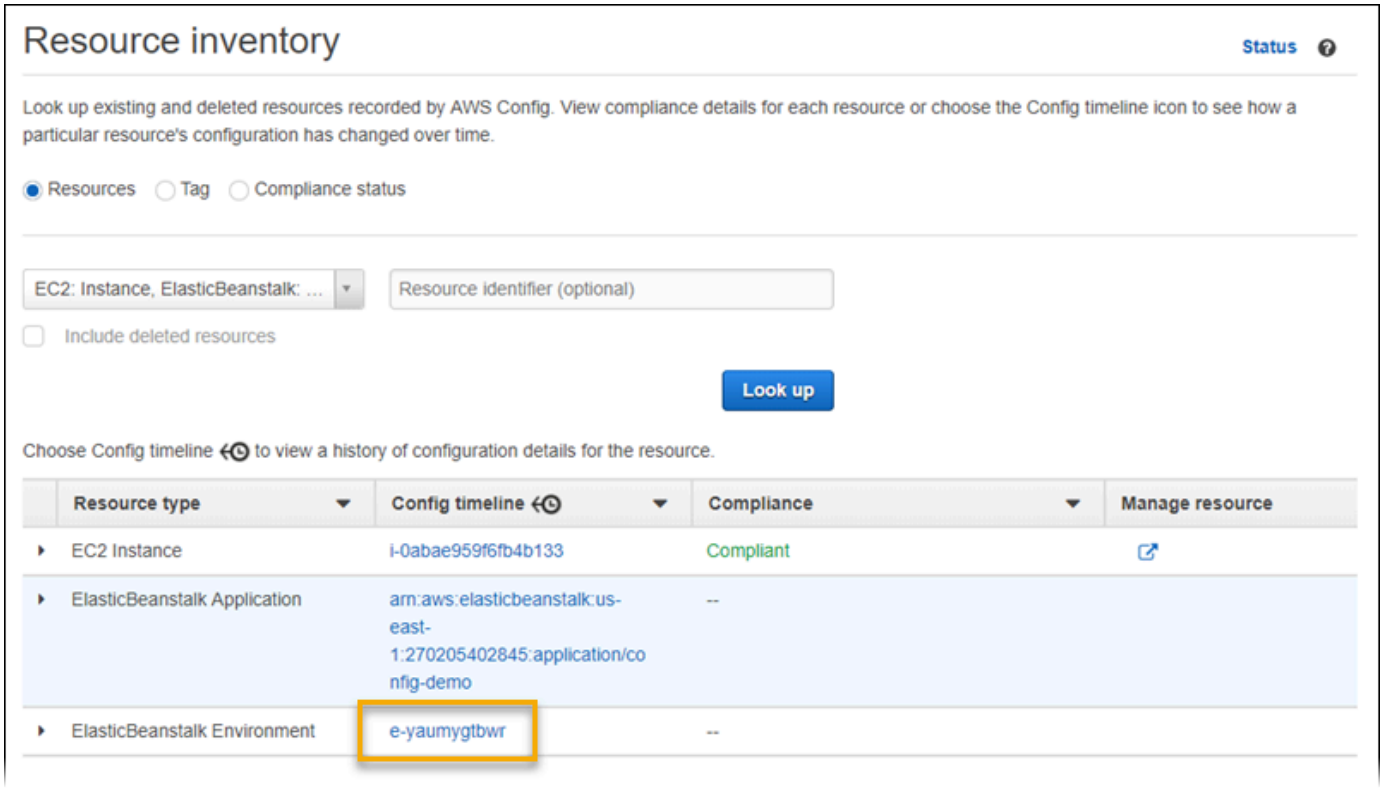
1. Buka konsol AWS Config.
2. Pilih Sumber Daya.
3. Pada halamanInventarisasi sumber daya, pilih Sumber Daya.
4. Buka menu Jenis sumber daya, gulir ke ElasticBeanstalk, lalu pilih satu atau beberapa jenis sumber daya Elastic Beanstalk.

> **Note**
>
> Untuk melihat rincian konfigurasi untuk sumber daya lain yang Elastic Beanstalk buat untuk aplikasi Anda, pilih jenis sumber daya tambahan. Misalnya, Anda dapat memilih Instance di bawah EC2.

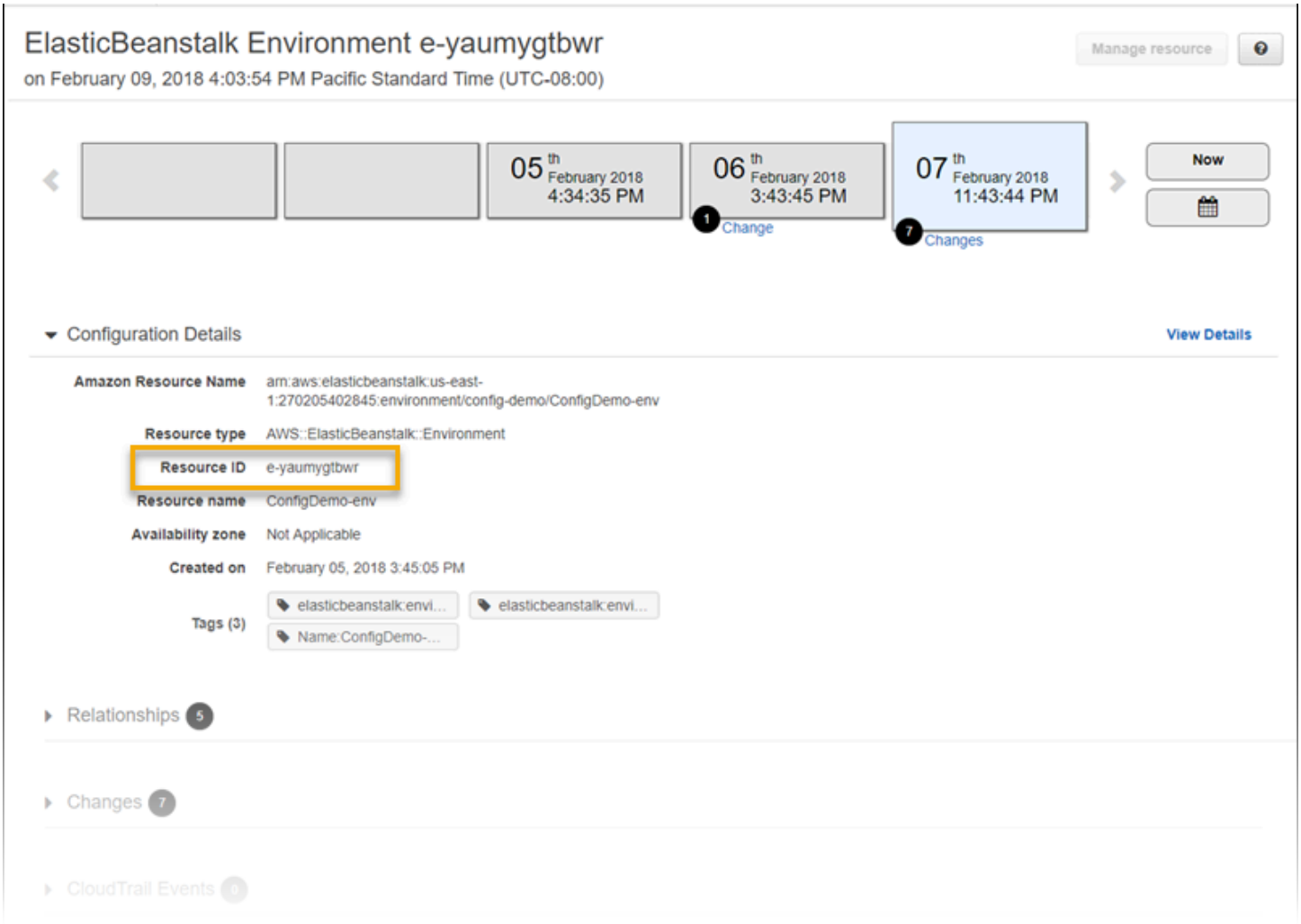
5. Pilih Lihat. Lihat 2 pada gambar berikut.



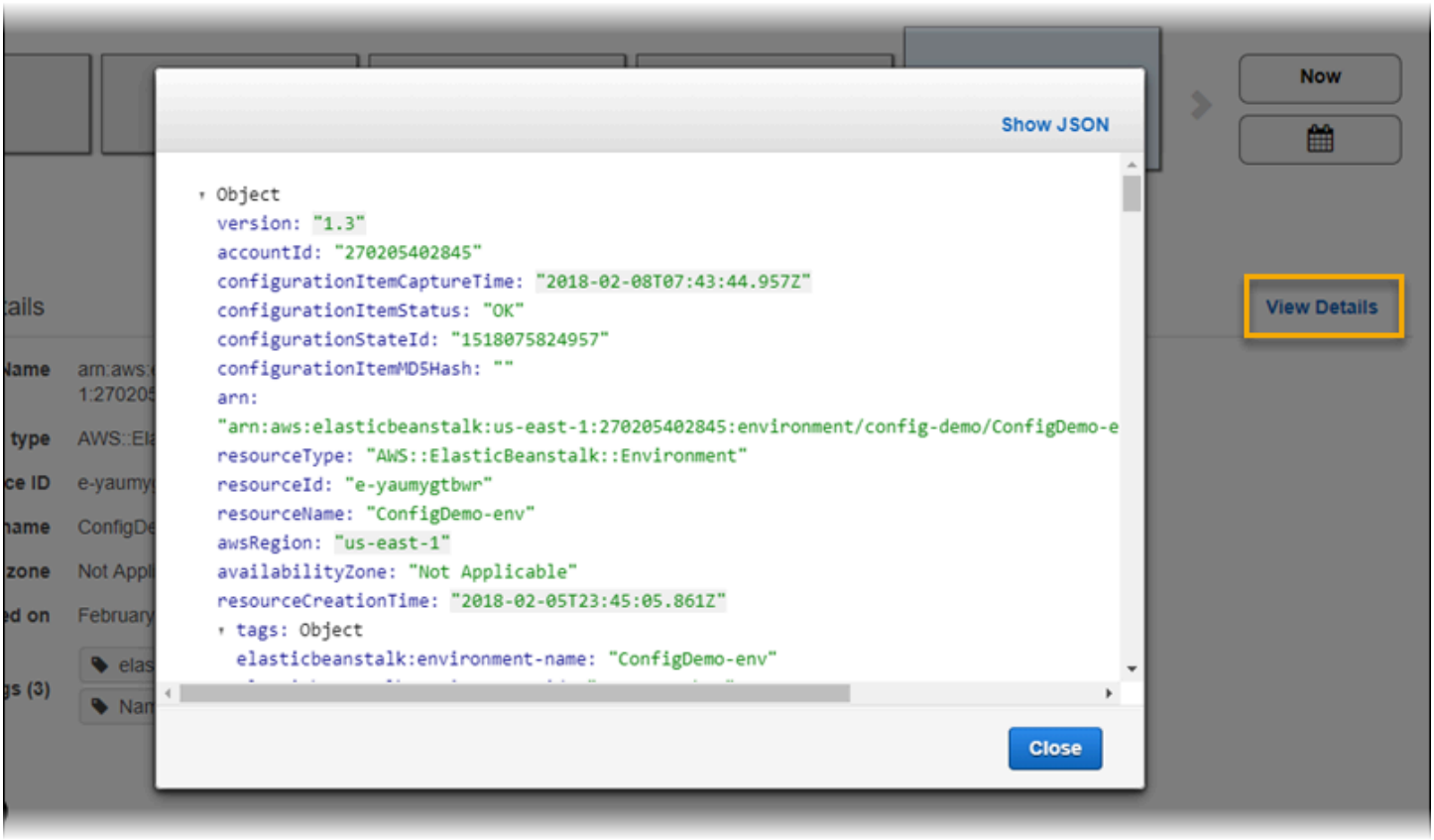
6. Pilih ID sumber daya dalam daftar sumber daya yang AWS Config ditampilkan.



AWS Config menampilkan detail konfigurasi dan informasi lain tentang sumber daya yang Anda pilih.



Untuk melihat detail lengkap konfigurasi yang direkam, pilih Lihat Detail.



Untuk mempelajari lebih lanjut cara menemukan sumber daya dan melihat informasi di halaman ini, lihat Melihat Konfigurasi dan Riwayat AWS Sumber Daya di Panduan AWS Config Pengembang.

Mengevaluasi sumber daya Elastic Beanstalk menggunakan aturan AWS Config

Anda dapat membuat AWS Config aturan, yang mewakili pengaturan konfigurasi ideal untuk sumber daya Elastic Beanstalk Anda. Anda dapat menggunakan Aturan Konfigurasi AWS Terkelola yang telah ditentukan sebelumnya, atau menentukan aturan khusus. AWS Config terus melacak perubahan pada konfigurasi sumber daya Anda untuk menentukan apakah perubahan tersebut melanggar salah satu kondisi dalam aturan Anda. AWS Config Konsol menunjukkan status kepatuhan aturan dan sumber daya Anda.

Jika sumber daya melanggar aturan dan ditandai sebagai tidak sesuai, AWS Config Anda dapat memperingatkan Anda menggunakan topik Amazon Simple Notification Service (Amazon SNS). Untuk menggunakan data dalam AWS Config peringatan ini secara terprogram, gunakan antrean Amazon Simple Queue Service (Amazon SQS) sebagai titik akhir notifikasi untuk topik Amazon SNS. Misalnya, Anda mungkin ingin menulis kode yang memulai alur kerja ketika seseorang mengubah konfigurasi grup Auto Scaling lingkungan Anda.

Untuk mempelajari lebih lanjut tentang menyiapkan dan menggunakan aturan, lihat Mengevaluasi Sumber Daya dengan AWS Config Aturan di Panduan AWS Config Pengembang.

Menggunakan Elastic Beanstalk dengan Amazon DynamoDB

Amazon DynamoDB adalah layanan database NoSQL terkelola sepenuhnya yang menyediakan kinerja cepat dan dapat diprediksi dengan skalabilitas tanpa hambatan. Jika Anda seorang developer, Anda dapat menggunakan DynamoDB untuk membuat tabel database yang dapat menyimpan dan mengambil sejumlah data, dan melayani setiap tingkat lalu lintas permintaan. DynamoDB secara otomatis menyebar data dan lalu lintas untuk tabel diatas jumlah yang cukup server untuk menangani kapasitas permintaan yang ditentukan oleh pelanggan dan jumlah data yang disimpan, sambil mempertahankan kinerja yang konsisten dan cepat. Semua item data disimpan pada solid state drive (SSDs) dan secara otomatis direplikasi di beberapa Availability Zone di AWS Region untuk menyediakan ketersediaan tinggi bawaan dan daya tahan data.

Jika Anda menggunakan tugas periodik di lingkungan pekerja, Elastic Beanstalk menciptakan tabel DynamoDB dan menggunakannya untuk melakukan pemilihan pemimpin dan menyimpan informasi tentang tugas tersebut. Setiap contoh di lingkungan mencoba untuk menulis ke meja setiap beberapa detik untuk menjadi pemimpin dan melakukan tugas ketika dijadwalkan.

Anda dapat menggunakan file konfigurasi untuk membuat tabel DynamoDB untuk aplikasi Anda. Lihat eb-node-express-samplecontoh aplikasi Node.js yang membuat tabel dengan file konfigurasi dan menghubungkannya dengan AWS SDK untuk JavaScript di Node.js. GitHub Untuk panduan contoh menggunakan DynamoDB dengan PHP, lihat Contoh: DynamoDB,, dan CloudWatch SNS. Untuk contoh yang menggunakan AWS SDK untuk Java, lihat Mengelola Status Sesi Tomcat dengan DynamoDB dalam dokumentasi. AWS SDK untuk Java

Ketika Anda membuat tabel DynamoDB menggunakan file konfigurasi, tabel tidak terikat dengan siklus hidup lingkungan Anda, dan tidak dihapus ketika Anda mengakhiri lingkungan Anda. Untuk memastikan bahwa informasi pribadi tidak disimpan secara tidak perlu, hapus catatan yang tidak Anda butuhkan lagi, atau hapus tabel.

Untuk informasi selengkapnya tentang DynamoDB, lihat Panduan Developer DynamoDB.

Menggunakan Elastic Beanstalk dengan Amazon ElastiCache

Amazon ElastiCache adalah layanan web yang memungkinkan pengaturan, pengelolaan, dan penskalaan lingkungan cache dalam memori terdistribusi di cloud. Ini menyediakan cache dalam memori berkinerja tinggi, skalabel, dan hemat biaya, sambil menghilangkan kompleksitas yang terkait

dengan penerapan dan pengelolaan lingkungan cache terdistribusi. ElastiCache sesuai dengan protokol dengan Redis dan Memcached, sehingga kode, aplikasi, dan alat paling populer yang Anda gunakan saat ini dengan lingkungan Redis dan Memcached yang ada akan bekerja dengan mulus dengan layanan ini. Untuk informasi lebih lanjut tentang ElastiCache, buka halaman ElastiCache produk Amazon.

Untuk menggunakan Elastic Beanstalk dengan Amazon ElastiCache

1. Buat ElastiCache cluster.
   - Untuk petunjuk tentang cara membuat ElastiCache klaster dengan Redis, buka Memulai Amazon ElastiCache untuk Redis di Panduan Pengguna ElastiCacheuntuk Redis.
   - Untuk petunjuk tentang cara membuat ElastiCache klaster dengan Memcached, buka Memulai Amazon untuk Memcached di Panduan Pengguna ElastiCache untuk Memcached. ElastiCache
2. Konfigurasikan Grup ElastiCache Keamanan Anda untuk mengizinkan akses dari grup EC2 keamanan Amazon yang digunakan oleh aplikasi Elastic Beanstalk Anda. Untuk petunjuk tentang cara menemukan nama grup EC2 keamanan menggunakan AWS Management Console, lihat EC2 kelompok keamanan di halaman dokumen EC2 Instans.
   - Untuk informasi lebih lanjut tentang Redis, buka Akses Otorisasi di Panduan Pengguna ElastiCache untuk Redis.
   - Untuk informasi lebih lanjut tentang Memcached, buka Otorisasi Akses di Panduan Pengguna ElastiCache untuk Memcached.

Anda dapat menggunakan file konfigurasi untuk menyesuaikan lingkungan Elastic Beanstalk Anda untuk digunakan. ElastiCache Untuk contoh file konfigurasi yang terintegrasi ElastiCache dengan Elastic Beanstalk, lihat. Contoh: ElastiCache

Menggunakan Elastic Beanstalk dengan Amazon Elastic File System

Dengan Amazon Elastic File System (Amazon EFS), Anda dapat membuat sistem file jaringan yang dapat dipasang oleh instans di beberapa Availability Zone. Sistem file Amazon EFS adalah AWS sumber daya yang menggunakan grup keamanan untuk mengontrol akses melalui jaringan yang ada di VPC default atau kustom Anda.

Di lingkungan Elastic Beanstalk, Anda dapat menggunakan Amazon EFS untuk membuat direktori bersama yang menyimpan file untuk aplikasi yang diunggah dan dimodifikasi pengguna. Aplikasi Anda dapat menangani volume Amazon EFS yang terpasang seperti penyimpanan lokal. Dengan begitu, Anda tidak perlu mengubah kode aplikasi Anda untuk meningkatkan skala hingga beberapa instance.

Untuk informasi tentang Amazon EFS, lihat Panduan Pengguna Amazon Elastic File System.

> **Note**
>
> Elastic Beanstalk menciptakan pengguna webapp yang dapat Anda atur sebagai pemilik direktori aplikasi di instans Amazon. EC2 Untuk informasi selengkapnya, lihat Penyimpanan Persisten dalam topik pertimbangan Desain panduan ini.

Bagian-bagian

- File konfigurasi
- Sistem file terenkripsi
- Aplikasi sampel
- Membersihkan sistem file

File konfigurasi

Elastic Beanstalk menyediakan file konfigurasi yang dapat Anda gunakan untuk membuat dan memasang sistem file Amazon EFS. Anda dapat membuat volume Amazon EFS sebagai bagian dari lingkungan Anda, atau memasang volume Amazon EFS yang Anda buat secara independen dari Elastic Beanstalk.

- storage-efs-createfilesystem.config — Menggunakan `Resources` kunci untuk membuat sistem file baru dan memasang titik di Amazon EFS. Semua instans di lingkungan Anda dapat terhubung ke sistem file yang sama untuk penyimpanan yang terukur dan dapat diskalakan. Gunakan `storage-efs-mountfilesystem.config` untuk memasang sistem file pada setiap instans.

> **Sumber daya internal**
>
> Sumber daya apa pun yang Anda buat dengan file konfigurasi terkait dengan siklus hidup lingkungan Anda. Jika Anda mengakhiri lingkungan Anda atau menghapus file konfigurasi, sumber daya ini hilang.

- storage-efs-mountfilesystem.config — Pasang sistem file Amazon EFS ke jalur lokal pada instance di lingkungan Anda. Anda dapat membuat volume sebagai bagian dari lingkungan dengan`storage-efs-createfilesystem.config`. Atau, Anda dapat memasangnya ke lingkungan menggunakan konsol Amazon EFS, AWS CLI, atau AWS SDK.

Untuk menggunakan file konfigurasi, mulai dengan membuat sistem file Amazon EFS Anda dengan `storage-efs-createfilesystem.config`. Ikuti instruksi di file konfigurasi dan menambahkannya ke direktori .ebextensions dalam kode sumber Anda untuk membuat sistem file di VPC Anda.

Terapkan kode sumber Anda yang diperbarui ke lingkungan Elastic Beanstalk Anda. Ini untuk mengonfirmasi bahwa sistem file berhasil dibuat. Kemudian, tambahkan `storage-efs-mountfilesystem.config` untuk memasang sistem file ke instans di lingkungan Anda. Melakukan ini dalam dua penerapan terpisah memastikan bahwa, jika operasi pemasangan gagal, sistem file tetap utuh. Jika Anda melakukan keduanya dalam deployment yang sama, masalah dengan salah satu langkah akan menyebabkan sistem file untuk mengakhiri ketika deployment gagal.

Sistem file terenkripsi

Amazon EFS mendukung sistem file terenkripsi. File `storage-efs-createfilesystem.config`konfigurasi yang dibahas dalam topik ini mendefinisikan dua opsi kustom. Anda dapat menggunakan opsi ini untuk membuat sistem file terenkripsi Amazon EFS. Untuk informasi selengkapnya, lihat instruksi dalam file konfigurasi.

Aplikasi sampel

Elastic Beanstalk juga menyediakan aplikasi sampel yang menggunakan Amazon EFS untuk penyimpanan bersama. Kedua proyek memiliki file konfigurasi yang dapat Anda gunakan dengan installer standar WordPress atau Drupal untuk menjalankan blog atau sistem manajemen konten lainnya di lingkungan yang seimbang beban. Saat pengguna mengunggah foto atau media lain, file

tersebut disimpan di sistem file Amazon EFS. Ini menghindari keharusan menggunakan alternatif, yaitu menggunakan plugin untuk menyimpan file yang diunggah di Amazon S3.

- Load-balanced WordPress - Ini termasuk file konfigurasi untuk menginstal dengan WordPress aman dan menjalankannya di lingkungan Beanstalk Elastic Beanstalk yang seimbang.
- Load-balanced Drupal - Ini termasuk file konfigurasi dan instruksi untuk menginstal Drupal dengan aman dan menjalankannya di lingkungan Elastic Beanstalk yang seimbang.

Membersihkan sistem file

Jika Anda membuat sistem file Amazon EFS yang menggunakan file konfigurasi sebagai bagian dari lingkungan Elastic Beanstalk Anda, Elastic Beanstalk menghapus sistem file saat Anda mengakhiri lingkungan. Untuk meminimalkan biaya penyimpanan aplikasi yang sedang berjalan, hapus file yang tidak dibutuhkan aplikasi Anda secara rutin. Atau, pastikan bahwa kode aplikasi mempertahankan siklus hidup file dengan benar.

> **Important**
>
> Jika Anda membuat sistem file Amazon EFS yang berada di luar lingkungan Elastic Beanstalk dan memasangnya ke instance lingkungan, Elastic Beanstalk tidak menghapus sistem file saat Anda menghentikan lingkungan. Untuk memastikan bahwa informasi pribadi Anda tidak disimpan dan menghindari biaya penyimpanan, hapus file yang disimpan aplikasi Anda jika Anda tidak membutuhkannya lagi. Atau, Anda dapat menghapus seluruh sistem file.

Menggunakan Elastic Beanstalk dengan AWS Identity and Access Management

AWS Identity and Access Management (IAM) membantu Anda mengontrol akses ke sumber daya Anda AWS dengan aman. Bagian ini mencakup bahan referensi untuk bekerja dengan kebijakan IAM, profil instans, dan peran layanan.

Untuk gambaran umum izin, lihat Peran Elastic Beanstalk Service, profil instans, dan kebijakan pengguna. Untuk sebagian besar lingkungan, peran layanan dan instans profil yang diminta oleh konsol Elastic Beanstalk untuk Anda buat saat Anda meluncurkan lingkungan pertama Anda memiliki semua izin yang Anda butuhkan. Demikian juga, kebijakan terkelola yang disediakan oleh Elastic

Beanstalk untuk akses penuh dan akses hanya-baca berisi semua izin pengguna yang diperlukan untuk penggunaan sehari-hari.

Panduan Pengguna IAM menyediakan cakupan izin yang mendalam. AWS

Topik

- Mengelola profil instans Elastic Beanstalk
- Mengelola peran layanan Elastic Beanstalk
- Menggunakan peran yang terhubung dengan layanan untuk Elastic Beanstalk
- Mengelola kebijakan pengguna Elastic Beanstalk
- Format nama sumber Amazon untuk Elastic Beanstalk
- Sumber daya dan kondisi untuk tindakan Elastic Beanstalk
- Menggunakan tag untuk mengontrol akses ke sumber Elastic Beanstalk
- Contoh kebijakan berdasarkan kebijakan terkelola
- Contoh kebijakan berdasarkan izin sumber daya
- Mencegah akses bucket Amazon S3 lintas lingkungan

Mengelola profil instans Elastic Beanstalk

Profil instance adalah wadah untuk peran AWS Identity and Access Management (IAM) yang dapat Anda gunakan untuk meneruskan informasi peran ke EC2 instans Amazon saat instance dimulai.

Jika AWS akun Anda tidak memiliki profil EC2 instans, Anda harus membuatnya menggunakan layanan IAM. Anda kemudian dapat menetapkan profil EC2 instance ke lingkungan baru yang Anda buat. Langkah-langkah Create environment di konsol Elastic Beanstalk memberi Anda akses ke konsol IAM, sehingga Anda dapat EC2 membuat profil instance dengan izin yang diperlukan.

> **Note**
>
> Sebelumnya Elastic Beanstalk membuat `aws-elasticbeanstalk-ec2-role` profil instance default EC2 bernama pertama AWS kali akun membuat lingkungan. Profil instance ini menyertakan kebijakan terkelola default. Jika akun Anda sudah memiliki profil instans ini, itu akan tetap tersedia bagi Anda untuk menetapkan ke lingkungan Anda.
> Namun, pedoman AWS keamanan terbaru tidak mengizinkan AWS layanan untuk secara otomatis membuat peran dengan kebijakan kepercayaan ke AWS layanan lain, EC2 dalam

hal ini. Karena pedoman keamanan ini, Elastic Beanstalk tidak lagi membuat profil instance default. `aws-elasticbeanstalk-ec2-role`

Kebijakan terkelola

Elastic Beanstalk menyediakan beberapa kebijakan terkelola untuk memungkinkan lingkungan Anda memenuhi berbagai kasus penggunaan. Untuk memenuhi kasus penggunaan default untuk lingkungan, kebijakan ini harus dilampirkan ke peran untuk profil EC2 instans.

- AWSElasticBeanstalkWebTier— Memberikan izin bagi aplikasi untuk mengunggah log ke Amazon S3 dan men-debug informasi ke. AWS X-Ray Untuk melihat konten kebijakan terkelola, lihat AWSElasticBeanstalkWebTierdi Panduan Referensi Kebijakan AWS Terkelola.
- AWSElasticBeanstalkWorkerTier— Memberikan izin untuk unggahan log, debugging, publikasi metrik, dan tugas instance pekerja, termasuk manajemen antrian, pemilihan pemimpin, dan tugas berkala. Untuk melihat konten kebijakan terkelola, lihat AWSElasticBeanstalkWorkerTierdi Panduan Referensi Kebijakan AWS Terkelola.
- AWSElasticBeanstalkMulticontainerDocker— Memberikan izin untuk Amazon Elastic Container Service untuk mengoordinasikan tugas klaster untuk lingkungan Docker. Untuk melihat konten kebijakan terkelola, lihat AWSElasticBeanstalkMulticontainerDockerdi Panduan Referensi Kebijakan AWS Terkelola.

Important

Kebijakan terkelola Elastic Beanstalk tidak memberikan izin terperinci—mereka memberikan semua izin yang berpotensi diperlukan untuk bekerja dengan aplikasi Elastic Beanstalk. Dalam beberapa kasus, Anda mungkin ingin membatasi izin kebijakan terkelola kami lebih lanjut. Untuk contoh satu kasus penggunaan, lihatMencegah akses bucket Amazon S3 lintas lingkungan.

Kebijakan terkelola kami juga tidak mencakup izin ke sumber daya khusus yang mungkin Anda tambahkan ke solusi Anda, dan yang tidak dikelola oleh Elastic Beanstalk. Untuk menerapkan izin yang lebih rinci, izin minimum yang diperlukan, atau izin sumber daya kustom, gunakan kebijakan khusus.

Kebijakan hubungan kepercayaan untuk EC2

Untuk mengizinkan EC2 instans di lingkungan Anda mengambil peran yang diperlukan, profil instans harus menentukan Amazon EC2 sebagai entitas tepercaya dalam kebijakan hubungan kepercayaan, sebagai berikut.

```
{
  "Version": "2008-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Untuk menyesuaikan izin, Anda dapat menambahkan kebijakan ke peran yang dilampirkan ke profil instans default atau membuat profil instans Anda sendiri dengan serangkaian izin terbatas.

Bagian-bagian

- Membuat profil instans
- Menambahkan izin ke profil instans default
- Memverifikasi izin yang ditetapkan profil instans Anda
- Memperbarui profil instance out-of-date default

Membuat profil instans

Profil instance adalah pembungkus di sekitar peran IAM standar yang memungkinkan EC2 instance untuk mengambil peran tersebut. Anda dapat membuat profil instance dengan kebijakan terkelola Elastic Beanstalk default. Anda juga dapat membuat profil instans tambahan untuk menyesuaikan izin untuk aplikasi yang berbeda. Atau Anda dapat membuat profil instance yang tidak menyertakan dua kebijakan terkelola yang memberikan izin untuk tingkat pekerja atau lingkungan Docker yang dikelola ECS, jika Anda tidak menggunakan fitur tersebut.

Untuk membuat profil instance dengan kebijakan terkelola default

1. Buka halaman Peran di konsol IAM.
2. Pilih Buat peran.

3. Untuk jenis entitas Tepercaya, pilih AWS layanan.
4. Untuk Service atau use case, pilih Elastic Beanstalk.
5. Untuk kasus Penggunaan, pilih Elastic Beanstalk — Compute.
6. Pilih Berikutnya.
7. Masukkan Nama peran.

   Anda dapat memasukkan nama peran default yang disarankan oleh konsol Elastic Beanstalk:. `aws-elasticbeanstalk-ec2-role`
8. Verifikasi bahwa kebijakan Izin menyertakan yang berikut ini, lalu pilih Berikutnya:

   - `AWSElasticBeanstalkWebTier`
   - `AWSElasticBeanstalkWorkerTier`
   - `AWSElasticBeanstalkMulticontainerDocker`
9. Pilih Buat peran.

Untuk membuat profil instans dengan pilihan spesifik kebijakan terkelola

1. Buka halaman Peran di konsol IAM.
2. Pilih Buat peran.
3. Di bawah Jenis entitas tepercaya, pilih AWS layanan.
4. Di bawah Kasus penggunaan, pilih EC2.
5. Pilih Berikutnya.
6. Lampirkan kebijakan terkelola yang sesuai yang disediakan oleh Elastic Beanstalk dan kebijakan tambahan apa pun yang memberikan izin yang diperlukan aplikasi Anda.
7. Pilih Berikutnya.
8. Masukkan nama untuk peran.
9. (Opsional) Tambahkan tag ke peran.
10. Pilih Buat peran.

Menambahkan izin ke profil instans default

Jika aplikasi Anda mengakses AWS APIs atau sumber daya yang izinnya tidak diberikan di profil instans default, tambahkan kebijakan yang memberikan izin di konsol IAM.

Untuk menambahkan kebijakan ke peran yang dilampirkan ke profil instans default

1. Buka Halaman peran di konsol IAM.
2. Pilih peran yang ditetapkan sebagai profil EC2 instans Anda.
3. Di tab Izin, pilih Lampirkan kebijakan.
4. Pilih kebijakan terkelola untuk layanan tambahan yang digunakan aplikasi Anda. Misalnya, `AmazonS3FullAccess` atau `AmazonDynamoDBFullAccess`.
5. Memilih Lampirkan kebijakan.

Memverifikasi izin yang ditetapkan profil instans Anda

Izin yang ditetapkan ke profil instans default dapat bervariasi tergantung pada kapan dibuat, terakhir kali Anda meluncurkan lingkungan, dan klien mana yang Anda gunakan. Anda dapat memverifikasi izin pada profil instans default di konsol IAM.

Untuk memverifikasi izin profil instans default

1. Buka Halaman Peran di konsol IAM.
2. Pilih peran yang ditetapkan sebagai profil EC2 instans Anda.
3. Pada tab Izin, tinjau daftar kebijakan yang dilampirkan pada peran.
4. Untuk melihat izin yang diberikan kebijakan, pilih kebijakan.

Memperbarui profil instance out-of-date default

Jika profil instans default tidak memiliki izin yang diperlukan, Anda dapat menambahkan kebijakan terkelola ke peran yang ditetapkan sebagai profil EC2 instance secara manual.

Untuk menambahkan kebijakan terkelola ke peran yang dilampirkan ke profil instans default

1. Buka halaman Peran di konsol IAM.
2. Pilih peran yang ditetapkan sebagai profil EC2 instans Anda.
3. Di tab Izin, klik Lampirkan kebijakan.
4. Jenis **`AWSElasticBeanstalk`** untuk memfilter kebijakan.
5. Pilih kebijakan berikut, dan kemudian pilih Lampirkan kebijakan:

   - `AWSElasticBeanstalkWebTier`

- `AWSElasticBeanstalkWorkerTier`
- `AWSElasticBeanstalkMulticontainerDocker`

Mengelola peran layanan Elastic Beanstalk

Untuk mengelola dan memantau lingkungan Anda, AWS Elastic Beanstalk lakukan tindakan terhadap sumber daya lingkungan atas nama Anda. Elastic Beanstalk memerlukan izin tertentu untuk melakukan tindakan ini, dan mengasumsikan peran layanan (IAM) AWS Identity and Access Management untuk mendapatkan izin ini.

Elastic Beanstalk perlu menggunakan kredensial keamanan sementara setiap kali mengasumsikan peran layanan. Untuk mendapatkan kredensial ini, Elastic Beanstalk mengirimkan permintaan ke AWS Security Token Service (AWS STS) pada titik akhir tertentu Wilayah. Untuk informasi lebih lanjut, lihat Kredensial Keamanan Sementara dalam Panduan Pengguna IAM.

> **Note**
>
> Jika AWS STS titik akhir untuk Wilayah tempat lingkungan Anda berada dinonaktifkan, Elastic Beanstalk mengirimkan permintaan pada titik akhir alternatif yang tidak dapat dinonaktifkan. Titik akhir ini dikaitkan dengan Wilayah yang berbeda. Oleh karena itu, permintaan adalah permintaan lintas-wilayah. Untuk informasi selengkapnya, lihat Mengaktifkan dan Menonaktifkan AWS STS di AWS Wilayah di Panduan Pengguna IAM.

Mengelola peran layanan menggunakan konsol Elastic Beanstalk dan EB CLI

Anda dapat menggunakan konsol Elastic Beanstalk dan EB CLI untuk mengatur peran layanan untuk lingkungan Anda dengan seperangkat izin yang memadai. Mereka membuat peran layanan default dan menggunakan kebijakan terkelola di dalamnya.

Kebijakan peran layanan yang terkelola

Elastic Beanstalk menyediakan satu kebijakan yang dikelola untuk pemantauan kondisi yang ditingkatkan, dan satu lagi dengan izin tambahan yang diperlukan untuk Pembaruan platform yang terkelola. Konsol dan EB CLI menetapkan kedua kebijakan ini untuk peran layanan default yang mereka buat untuk Anda. Kebijakan ini hanya boleh digunakan untuk peran layanan default ini. Mereka tidak boleh digunakan dengan pengguna lain atau peran dalam akun Anda.

`AWSElasticBeanstalkEnhancedHealth`

Kebijakan ini memberikan izin untuk Elastic Beanstalk untuk memantau instans dan kondisi lingkungan. Ini juga mencakup tindakan Amazon SQS untuk memungkinkan Elastic Beanstalk memantau aktivitas antrian untuk lingkungan pekerja. Untuk melihat konten kebijakan terkelola ini, lihat AWSElasticBeanstalkEnhancedHealthhalaman di Panduan Referensi Kebijakan AWS Terkelola.

`AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy`

Kebijakan ini memberikan izin untuk Elastic Beanstalk untuk memperbarui lingkungan atas nama Anda untuk melakukan pembaruan platform yang terkelola. Untuk melihat konten kebijakan terkelola ini, lihat AWSElasticBeanstalkManagedUpdatesCustomerRolePolicyhalaman di Panduan Referensi Kebijakan AWS Terkelola.

Pengelompokan izin tingkat layanan

Kebijakan ini dikelompokkan ke dalam pernyataan berdasarkan kumpulan izin yang diberikan.

- *`ElasticBeanstalkPermissions`*— Kelompok izin ini untuk memanggil tindakan layanan Elastic Beanstalk (Elastic Beanstalk). APIs
- *`AllowPassRoleToElasticBeanstalkAndDownstreamServices`* – Kelompok izin ini mengizinkan peran apa pun untuk diteruskan ke Elastic Beanstalk dan ke layanan hilir lainnya seperti AWS CloudFormation.
- *`ReadOnlyPermissions`* – Kelompok izin ini adalah untuk mengumpulkan informasi tentang lingkungan berjalan.
- *`*OperationPermissions`* – Grup dengan pola penamaan ini adalah untuk memanggil operasi yang diperlukan untuk melakukan pembaruan platform.
- *`*BroadOperationPermissions`* – Grup dengan pola penamaan ini adalah untuk memanggil operasi yang diperlukan untuk melakukan pembaruan platform. Mereka juga menyertakan izin yang luas untuk mendukung lingkungan lama.
- *`*TagResource`*— Grup dengan pola penamaan ini adalah untuk panggilan yang menggunakan tag-on-create APIs untuk melampirkan tag pada sumber daya yang sedang dibuat di lingkungan Elastic Beanstalk.

Untuk melihat konten kebijakan terkelola, Anda juga dapat menggunakan halaman Kebijakan di konsol IAM.

> **Important**
>
> Kebijakan terkelola Elastic Beanstalk tidak memberikan izin terperinci—mereka memberikan semua izin yang berpotensi diperlukan untuk bekerja dengan aplikasi Elastic Beanstalk. Dalam beberapa kasus, Anda mungkin ingin membatasi izin kebijakan terkelola kami lebih lanjut. Untuk contoh satu kasus penggunaan, lihatMencegah akses bucket Amazon S3 lintas lingkungan.
>
> Kebijakan terkelola kami juga tidak mencakup izin ke sumber daya khusus yang mungkin Anda tambahkan ke solusi Anda, dan yang tidak dikelola oleh Elastic Beanstalk. Untuk menerapkan izin yang lebih rinci, izin minimum yang diperlukan, atau izin sumber daya kustom, gunakan kebijakan khusus.

Kebijakan terkelola tidak lagi digunakan

Di masa lalu, Elastic Beanstalk AWSElasticBeanstalkServicemendukung kebijakan peran layanan terkelola. Kebijakan ini telah diganti dengan AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy. Anda mungkin masih dapat melihat dan menggunakan kebijakan sebelumnya di konsol IAM.

Untuk melihat konten kebijakan terkelola, lihat AWSElasticBeanstalkServicedi Panduan Referensi Kebijakan AWS Terkelola.

Namun, kami menyarankan agar Anda beralih menggunakan kebijakan terkelola (AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy) yang baru. Tambahkan kebijakan kustom untuk memberikan izin ke sumber daya kustom, jika ada.

Menggunakan konsol Elastic Beanstalk

Ketika Anda meluncurkan lingkungan di konsol Elastic Beanstalk, konsol menciptakan peran layanan default yang bernama `aws-elasticbeanstalk-service-role`, dan melampirkan kebijakan terkelola dengan izin default untuk peran layanan ini.

Mengizinkan Elastic Beanstalk untuk mengasumsikan peran `aws-elasticbeanstalk-service-role`, peran layanan menentukan Elastic Beanstalk sebagai entitas terpercaya dalam kebijakan hubungan kepercayaan.

```
{
    "Version": "2012-10-17",
    "Statement": [
```

```
      {
        "Sid": "",
        "Effect": "Allow",
        "Principal": {
          "Service": "elasticbeanstalk.amazonaws.com"
        },
        "Action": "sts:AssumeRole",
        "Condition": {
          "StringEquals": {
            "sts:ExternalId": "elasticbeanstalk"
          }
        }
      }
    ]
}
```

Saat Anda mengaktifkan Pembaruan platform terkelola untuk lingkungan Anda, Elastic Beanstalk mengasumsikan peran layanan pembaruan terkelola terpisah untuk melakukan pembaruan terkelola. Secara default, konsol Elastic Beanstalk menggunakan peran layanan yang dihasilkan sama, `aws-elasticbeanstalk-service-role`, untuk peran layanan pembaruan terkelola. Jika Anda mengubah peran layanan default, konsol menetapkan peran layanan pembaruan terkelola untuk menggunakan peran yang terhubung dengan layanan pembaruan yang dikelola, `AWSServiceRoleForElasticBeanstalkManagedUpdates`. Untuk informasi selengkapnya tentang peran yang terhubung dengan layanan, lihat the section called “Menggunakan peran yang terhubung dengan layanan”.

> **Note**
>
> Karena masalah izin, layanan Elastic Beanstalk tidak selalu berhasil membuat peran yang terhubung dengan layanan ini untuk Anda. Oleh karena itu, konsol mencoba untuk secara eksplisit membuat itu. Untuk memastikan akun Anda memiliki peran yang terhubung dengan layanan ini, buat lingkungan setidaknya sekali menggunakan konsol, dan mengonfigurasi pembaruan terkelola agar diaktifkan sebelum Anda membuat lingkungan.

Penggunaan EB CLI

Jika Anda meluncurkan lingkungan menggunakan perintah the section called “eb create” dari Elastic Beanstalk Command Line Interface (EB CLI) dan tidak menentukan peran layanan melalui pilihan `--service-role`, Elastic Beanstalk menciptakan peran layanan default `aws-elasticbeanstalk-`

`service-role`. Jika peran layanan default sudah ada, Elastic Beanstalk menggunakannya untuk lingkungan baru. Konsol Elastic Beanstalk juga melakukan tindakan serupa dalam situasi ini.

Tidak seperti di konsol, Anda tidak dapat menentukan peran layanan pembaruan terkelola saat menggunakan opsi perintah EB CLI. Jika Anda mengaktifkan pembaruan terkelola untuk lingkungan Anda, Anda harus menetapkan peran layanan pembaruan dikelola meskipun opsi konfigurasi. Contoh berikut memungkinkan pembaruan terkelola dan menggunakan peran layanan default sebagai peran layanan pembaruan terkelola.

Example .ebextensions/ .config managed-platform-update

```
option_settings:
  aws:elasticbeanstalk:managedactions:
    ManagedActionsEnabled: true
    PreferredStartTime: "Tue:09:00"
    ServiceRoleForManagedUpdates: "aws-elasticbeanstalk-service-role"
  aws:elasticbeanstalk:managedactions:platformupdate:
    UpdateLevel: patch
    InstanceRefreshEnabled: true
```

Mengelola peran layanan menggunakan API Elastic Beanstalk

Ketika Anda menggunakan tindakan `CreateEnvironment` dari API Elastic Beanstalk untuk menciptakan lingkungan, menentukan peran layanan menggunakan pilihan konfigurasi `ServiceRole` di namespace `aws:elasticbeanstalk:environment`. Untuk informasi lebih lanjut tentang penggunaan pemantauan kondis yang ditingkatkan dengan API Elastic Beanstalk, lihat Menggunakan pelaporan kondisi yang ditingkatkan dengan API Elastic Beanstalk.

Selain itu, jika Anda mengaktifkan pembaruan platform terkelola untuk lingkungan Anda, Anda dapat menentukan peran layanan pembaruan terkelola menggunakan pilihan `ServiceRoleForManagedUpdates` dari namespace `aws:elasticbeanstalk:managedactions`.

Menggunakan peran yang terhubung dengan layanan

Peran terkait layanan adalah jenis peran layanan unik yang telah ditentukan sebelumnya oleh Elastic Beanstalk untuk menyertakan semua izin yang diperlukan layanan untuk memanggil layanan lain atas nama Anda. AWS Peran terkait layanan berkaitan dengan akun Anda. Elastic Beanstalk menciptakannya sekali, kemudian menggunakannya kembali saat menciptakan lingkungan tambahan. Untuk informasi selengkapnya tentang penggunaan peran yang terhubung dengan

layanan dengan lingkungan Elastic Beanstalk, lihat Menggunakan peran yang terhubung dengan layanan untuk Elastic Beanstalk.

Jika Anda membuat lingkungan dengan menggunakan API Elastic Beanstalk dan tidak menentukan peran layanan, Elastic Beanstalk membuat peran yang terhubung dengan layanan pemantauan untuk akun Anda, jika salah satu belum ada. Elastic Beanstalk menggunakan peran ini untuk lingkungan baru. Anda juga dapat menggunakan IAM untuk membuat peran yang terhubung dengan layanan pemantauan untuk akun Anda terlebih dahulu. Setelah akun Anda memiliki peran ini, Anda dapat menggunakannya untuk membuat lingkungan menggunakan API Elastic Beanstalk, konsol Elastic Beanstalk, atau EB CLI.

Jika Anda mengaktifkan pembaruan platform terkelola untuk lingkungan dan tentukan `AWSServiceRoleForElasticBeanstalkManagedUpdates` sebagai nilai untuk opsi `ServiceRoleForManagedUpdates` dari namespace `aws:elasticbeanstalk:managedactions`, Elastic Beanstalk membuat peran yang terhubung dengan layanan pembaruan terkelola untuk akun Anda, jika salah satu belum ada. Elastic Beanstalk menggunakan peran untuk melakukan pembaruan terkelola untuk lingkungan baru.

> **Note**
>
> Ketika Elastic Beanstalk mencoba membuat peran yang terhubung dengan layanan pemantauan dan pembaruan terkelola untuk akun Anda saat membuat lingkungan, Anda harus memiliki izin `iam:CreateServiceLinkedRole`. Jika Anda tidak memiliki izin ini, penciptaan lingkungan gagal, dan pesan yang menjelaskan masalah akan ditampilkan. Sebagai alternatif, pengguna lain dengan izin untuk membuat peran yang terhubung dengan layanan dapat menggunakan IAM untuk membuat peran yang terhubung dengan layanan terlebih dahulu. Menggunakan metode ini, Anda tidak perlu izin `iam:CreateServiceLinkedRole` untuk menciptakan lingkungan Anda.

Memverifikasi izin peran layanan default

Izin yang diberikan oleh peran layanan default dapat bervariasi berdasarkan kapan dibuat, terakhir kali Anda meluncurkan lingkungan, dan klien mana yang Anda gunakan. Di konsol IAM, Anda dapat memverifikasi izin yang diberikan oleh peran layanan default.

Untuk memverifikasi izin peran layanan default

1. Di konsol IAM, buka halaman Peran.

2. Pilih aws-elasticbeanstalk-service-role.
3. Pada tab Izin, tinjau daftar kebijakan yang dilampirkan pada peran.
4. Untuk melihat izin yang diberikan kebijakan, pilih kebijakan.

Memperbarui peran layanan out-of-date default

Jika peran layanan default tidak memiliki izin yang diperlukan, Anda dapat memperbaruinya dengan Membuat lingkungan baru di konsol manajemen lingkungan Elastic Beanstalk.

Atau, Anda dapat secara manual menambahkan kebijakan terkelola ke peran layanan default.

Menambahkan kebijakan terkelola ke peran layanan default

1. Di konsol IAM, buka halaman Peran.
2. Pilih aws-elasticbeanstalk-service-role.
3. Di tab Izin, klik Lampirkan kebijakan.
4. Masukkan **`AWSElasticBeanstalk`** untuk memfilter kebijakan.
5. Pilih kebijakan berikut, dan kemudian pilih Lampirkan kebijakan:
   - `AWSElasticBeanstalkEnhancedHealth`
   - `AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy`

Menambahkan izin ke peran layanan default

Jika aplikasi Anda menyertakan file konfigurasi yang merujuk ke AWS sumber daya yang tidak disertakan izin dalam peran layanan default, Elastic Beanstalk mungkin memerlukan izin tambahan. Izin tambahan ini diperlukan untuk menyelesaikan referensi ini ketika proses file konfigurasi selama pembaruan yang terkelola. Jika izin hilang, pembaruan gagal, dan Elastic Beanstalk mengembalikan pesan yang menunjukkan izin yang dibutuhkan. Ikuti langkah-langkah untuk menambahkan izin untuk layanan tambahan untuk peran layanan default di konsol IAM.

Untuk menambahkan kebijakan tambahan ke peran layanan default

1. Di konsol IAM, buka halaman Peran.
2. Pilih aws-elasticbeanstalk-service-role.
3. Di tab Izin, pilih Lampirkan kebijakan.

4. Pilih kebijakan terkelola untuk layanan tambahan yang digunakan aplikasi Anda. Misalnya, `AmazonAPIGatewayAdministrator` atau `AmazonElasticFileSystemFullAccess`.
5. Pilih Pasang kebijakan.

Membuat peran layanan

Jika Anda tidak dapat menggunakan peran layanan default, buat peran layanan.

Membuat peran layanan

1. Di konsol IAM, buka halaman Peran.
2. Pilih Buat peran.
3. Di bawah layanan AWS , pilihAWS Elastic Beanstalk, lalu pilih kasus penggunaan Anda.
4. Pilih Berikutnya: Izin.
5. Pasang kebijakan terkelola `AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy` dan `AWSElasticBeanstalkEnhancedHealth` dan kebijakan tambahan apa pun yang memberikan izin yang diperlukan aplikasi Anda.
6. Pilih Berikutnya: Tag.
7. (Opsional) Tambahkan tag ke peran.
8. Pilih Berikutnya: Tinjauan.
9. Masukkan nama untuk peran.
10. Pilih Buat peran.

Terapkan peran layanan kustom Anda ketika Anda membuat lingkungan baik menggunakan wizard pembuatan lingkungan atau dengan pilihan `--service-role` untuk perintah `eb create`.

Menggunakan peran yang terhubung dengan layanan untuk Elastic Beanstalk

AWS Elastic Beanstalk menggunakan AWS Identity and Access Management peran terkait layanan (IAM). Peran yang terhubung dengan layanan adalah jenis IAM role unik yang terhubung langsung ke Elastic Beanstalk. Peran terkait layanan telah ditentukan sebelumnya oleh Elastic Beanstalk dan mencakup semua izin yang diperlukan layanan untuk memanggil layanan lain atas nama Anda. AWS

Elastic Beanstalk mendefinisikan beberapa jenis peran yang terhubung dengan layanan:

- Peran yang terhubung dengan layanan pemantauan – Mengizinkan Elastic Beanstalk untuk memantau kondisi lingkungan berjalan dan mempublikasikan pemberitahuan acara kondisi.
- Peran yang terhubung dengan layanan pemeliharaan – Mengizinkan Elastic Beanstalk untuk melakukan aktivitas pemeliharaan rutin untuk lingkungan berjalan Anda.
- Peran terkait layanan pembaruan terkelola — Memungkinkan Elastic Beanstalk melakukan pembaruan platform terjadwal di lingkungan berjalan Anda.

Topik

- Peran yang terhubung dengan layanan pemantauan
- Peran yang terhubung dengan layanan pemeliharaan
- Peran yang terhubung dengan layanan permbaharuan yang terkelola

Peran yang terhubung dengan layanan pemantauan

AWS Elastic Beanstalk menggunakan AWS Identity and Access Management peran terkait layanan (IAM). Peran yang terhubung dengan layanan adalah jenis IAM role unik yang terhubung langsung ke Elastic Beanstalk. Peran terkait layanan telah ditentukan sebelumnya oleh Elastic Beanstalk dan mencakup semua izin yang diperlukan layanan untuk memanggil layanan lain atas nama Anda. AWS

Peran yang terhubung dengan layanan memudahkan pengaturan Elastic Beanstalk karena Anda tidak perlu menambahkan izin yang diperlukan secara manual. Elastic Beanstalk menentukan izin peran yang terhubung dengan layanan, kecuali jika ditentukan berbeda, hanya Elastic Beanstalk yang dapat mengasumsikan perannya. Izin yang ditentukan mencakup kebijakan kepercayaan dan kebijakan izin, serta bahwa kebijakan izin tidak dapat dilampirkan ke entitas IAM lainnya.

Anda dapat menghapus peran yang terhubung dengan layanan hanya setelah pertama kali menghapus sumber daya terkait. Ini melindungi sumber daya Elastic Beanstalk Anda karena Anda tidak dapat secara tidak sengaja menghapus izin untuk mengakses sumber daya.

Untuk informasi tentang layanan lain yang mendukung peran yang terhubung dengan layanan, lihat Layanan AWS yang Berfungsi dengan IAM dan cari layanan yang memiliki Ya di kolom Peran yang Terhubung dengan Layanan. Pilih Ya dengan tautan untuk melihat dokumentasi peran yang terhubung dengan layanan untuk layanan tersebut.

Izin peran yang terhubung dengan layanan untuk Elastic Beanstalk

Elastic Beanstalk menggunakan peran terkait layanan AWSServiceRoleForElasticBeanstalkbernama - Memungkinkan Elastic Beanstalk memantau kesehatan lingkungan lari dan mempublikasikan pemberitahuan acara kesehatan.

Peran AWSService RoleForElasticBeanstalk terkait layanan mempercayai layanan berikut untuk mengambil peran:

- `elasticbeanstalk.amazonaws.com`

Kebijakan izin peran AWSService RoleForElasticBeanstalk terkait layanan berisi semua izin yang diperlukan Elastic Beanstalk untuk menyelesaikan tindakan atas nama Anda:

AllowCloudformationReadOperationsOnElasticBeanstalkStacks

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowCloudformationReadOperationsOnElasticBeanstalkStacks",
            "Effect": "Allow",
            "Action": [
                "cloudformation:DescribeStackResource",
                "cloudformation:DescribeStackResources",
                "cloudformation:DescribeStacks"
            ],
            "Resource": [
                "arn:aws:cloudformation:*:*:stack/awseb-*",
                "arn:aws:cloudformation:*:*:stack/eb-*"
            ]
        },
        {
            "Sid": "AllowOperations",
            "Effect": "Allow",
            "Action": [
                "autoscaling:DescribeAutoScalingGroups",
                "autoscaling:DescribeAutoScalingInstances",
                "autoscaling:DescribeNotificationConfigurations",
                "autoscaling:DescribeScalingActivities",
                "autoscaling:PutNotificationConfiguration",
                "ec2:DescribeInstanceStatus",
```

```
                "ec2:AssociateAddress",
                "ec2:DescribeAddresses",
                "ec2:DescribeInstances",
                "ec2:DescribeSecurityGroups",
                "elasticloadbalancing:DescribeInstanceHealth",
                "elasticloadbalancing:DescribeLoadBalancers",
                "elasticloadbalancing:DescribeTargetHealth",
                "elasticloadbalancing:DescribeTargetGroups",
                "sqs:GetQueueAttributes",
                "sqs:GetQueueUrl",
                "sns:Publish"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
```

Anda harus mengonfigurasi izin untuk mengizinkan entitas IAM (seperti pengguna, grup, atau peran) untuk membuat, mengedit, atau menghapus peran terkait layanan. Untuk informasi selengkapnya, silakan lihat Izin Peran Tertaut Layanan di Panduan Pengguna IAM.

Atau, Anda dapat menggunakan kebijakan AWS terkelola untuk menyediakan akses penuh ke Elastic Beanstalk.

Membuat peran yang terhubung dengan layanan untuk Elastic Beanstalk

Anda tidak perlu membuat peran yang terhubung dengan layanan secara manual. Bila Anda membuat lingkungan Elastic Beanstalk menggunakan API Elastic Beanstalk dan tidak menentukan peran layanan, Elastic Beanstalk membuat peran yang terhubung dengan layanan untuk Anda.

> **Important**
>
> Jika Anda menggunakan layanan Elastic Beanstalk sebelum 27 September 2017, ketika layanan tersebut mulai AWSService RoleForElasticBeanstalk mendukung peran terkait layanan, dan akun Anda membutuhkannya, maka Elastic Beanstalk menciptakan peran tersebut di akun Anda. AWSService RoleForElasticBeanstalk Untuk mempelajari lebih lanjut, lihat Peran Baru yang Muncul di Akun IAM Saya.

Saat Elastic Beanstalk mencoba AWSService RoleForElasticBeanstalk membuat peran terkait layanan untuk akun Anda saat membuat lingkungan, Anda harus memiliki izin. `iam:CreateServiceLinkedRole` Jika Anda tidak memiliki izin ini, pembuatan lingkungan gagal, dan Anda melihat pesan yang menjelaskan masalah.

Sebagai alternatif, pengguna lain dengan izin untuk membuat peran yang terhubung dengan layanan dapat menggunakan IAM untuk membuat peran terkait layanan terlebih dahulu. Anda kemudian dapat menciptakan lingkungan Anda bahkan tanpa izin `iam:CreateServiceLinkedRole`.

Anda (atau pengguna lain) dapat menggunakan konsol IAM untuk membuat peran yang terhubung dengan layanan dengan kasus penggunaan Elastic Beanstalk. Di IAM CLI atau IAM API, buat peran yang terhubung dengan layanan dengan nama layanan `elasticbeanstalk.amazonaws.com`. Untuk informasi lebih lanjut, lihat Membuat Peran yang Terhubung dengan Layanan di Panduan Pengguna IAM. Jika Anda menghapus peran yang terhubung dengan layanan ini, Anda dapat menggunakan proses yang sama untuk membuat ulang peran tersebut.

Jika Anda menghapus peran yang terhubung dengan layanan ini, lalu ingin membuatnya lagi, Anda dapat menggunakan proses yang sama untuk membuat ulang peran tersebut di akun Anda. Bila Anda membuat lingkungan Elastic Beanstalk menggunakan API Elastic Beanstalk dan tidak menentukan peran layanan, Elastic Beanstalk membuat peran yang terhubung dengan layanan bagi Anda kembali.

Mengedit peran yang terhubung dengan layanan untuk Elastic Beanstalk

Elastic Beanstalk tidak memungkinkan Anda untuk mengedit peran terkait layanan. AWSService RoleForElasticBeanstalk Setelah membuat peran terkait layanan, Anda tidak dapat mengubah nama peran karena berbagai entitas mungkin merujuk peran tersebut. Namun, Anda dapat mengedit penjelasan peran menggunakan IAM. Untuk informasi lebih lanjut, lihat Mengedit Peran yang Terhubung dengan Layanan di Panduan Pengguna IAM.

Menghapus peran yang terhubung dengan layanan untuk Elastic Beanstalk

Jika Anda tidak perlu lagi menggunakan fitur atau layanan yang memerlukan peran yang terhubung dengan layanan, kami menyarankan Anda menghapus peran tersebut. Dengan begitu, Anda tidak memiliki entitas yang tidak digunakan yang tidak dipantau atau dipelihara secara aktif. Tetapi, Anda harus membersihkan sumber daya peran yang terhubung dengan layanan sebelum menghapusnya secara manual.

Membersihkan peran yang terhubung dengan layanan

Sebelum Anda dapat menggunakan IAM untuk menghapus peran yang terhubung dengan layanan, Anda harus memastikan terlebih dahulu bahwa seluruh lingkungan Elastic Beanstalk menggunakan peran layanan yang berbeda atau dihentikan.

> **Note**
>
> Jika layanan Elastic Beanstalk menggunakan peran yang terhubung dengan layanan saat Anda mencoba untuk menghentikan lingkungan, maka penghentian mungkin gagal. Jika hal itu terjadi, tunggu beberapa menit dan coba mengoperasikannya lagi.

Untuk mengakhiri lingkungan Elastic Beanstalk yang menggunakan (konsol) AWSService RoleForElasticBeanstalk

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pilih Tindakan, lalu pilih Terminate environment.
4. Gunakan kotak dialog di layar untuk mengonfirmasi penghentian lingkungan.

Lihat eb terminate untuk rincian tentang mengakhiri lingkungan Elastic Beanstalk menggunakan EB CLI.

Lihat TerminateEnvironmentdetail tentang mengakhiri lingkungan Elastic Beanstalk menggunakan API.

Menghapus peran tertaut layanan secara manual

Gunakan konsol IAM, IAM CLI, atau IAM API untuk menghapus peran terkait layanan. AWSService RoleForElasticBeanstalk Untuk informasi lebih lanjut, lihat Menghapus Peran yang Terhubung Dengan Layanan dalam Panduan Pengguna IAM.

Wilayah yang didukung untuk peran yang terhubung dengan layanan Elastic Beanstalk

Support Elastic Beanstalk menggunakan peran yang terhubung dengan layanan di semua wilayah tempat layanan tersedia. Untuk informasi lebih lanjut, lihat Kuota dan Titik Akhir AWS Elastic Beanstalk.

Peran yang terhubung dengan layanan pemeliharaan

AWS Elastic Beanstalk menggunakan AWS Identity and Access Management peran terkait layanan (IAM). Peran yang terhubung dengan layanan adalah jenis IAM role unik yang terhubung langsung ke Elastic Beanstalk. Peran terkait layanan telah ditentukan sebelumnya oleh Elastic Beanstalk dan mencakup semua izin yang diperlukan layanan untuk memanggil layanan lain atas nama Anda. AWS

Peran yang terhubung dengan layanan memudahkan pengaturan Elastic Beanstalk karena Anda tidak perlu menambahkan izin yang diperlukan secara manual. Elastic Beanstalk menentukan izin peran yang terhubung dengan layanan, kecuali jika ditentukan berbeda, hanya Elastic Beanstalk yang dapat mengasumsikan perannya. Izin yang ditentukan mencakup kebijakan kepercayaan dan kebijakan izin, serta bahwa kebijakan izin tidak dapat dilampirkan ke entitas IAM lainnya.

Anda dapat menghapus peran yang terhubung dengan layanan hanya setelah pertama kali menghapus sumber daya terkait. Ini melindungi sumber daya Elastic Beanstalk Anda karena Anda tidak dapat secara tidak sengaja menghapus izin untuk mengakses sumber daya.

Untuk informasi tentang layanan lain yang mendukung peran yang terhubung dengan layanan, lihat Layanan AWS yang Berfungsi dengan IAM dan cari layanan yang memiliki Ya di kolom Peran yang Terhubung dengan Layanan. Pilih Ya dengan tautan untuk melihat dokumentasi peran yang terhubung dengan layanan untuk layanan tersebut.

Izin peran yang terhubung dengan layanan untuk Elastic Beanstalk

Elastic Beanstalk menggunakan peran terkait layanan AWSServiceRoleForElasticBeanstalkMaintenancebernama — Memungkinkan Elastic Beanstalk melakukan aktivitas pemeliharaan rutin untuk lingkungan lari Anda.

Peran AWSService RoleForElasticBeanstalkMaintenance terkait layanan mempercayai layanan berikut untuk mengambil peran:

- `maintenance.elasticbeanstalk.amazonaws.com`

Kebijakan izin peran AWSService RoleForElasticBeanstalkMaintenance terkait layanan berisi semua izin yang diperlukan Elastic Beanstalk untuk menyelesaikan tindakan atas nama Anda:

```
{
    "Version": "2012-10-17",
    "Statement":
```

```
        {
            "Sid": "AllowCloudformationChangeSetOperationsOnElasticBeanstalkStacks",
            "Effect": "Allow",
            "Action": [
                "cloudformation:CreateChangeSet",
                "cloudformation:DescribeChangeSet",
                "cloudformation:ExecuteChangeSet",
                "cloudformation:DeleteChangeSet",
                "cloudformation:ListChangeSets",
                "cloudformation:DescribeStacks"
            ],
            "Resource": [
                "arn:aws:cloudformation:*:*:stack/awseb-*",
                "arn:aws:cloudformation:*:*:stack/eb-*"
            ]
        }
}
```

Anda harus mengonfigurasi izin untuk mengizinkan entitas IAM (seperti pengguna, grup, atau peran) untuk membuat, mengedit, atau menghapus peran terkait layanan. Untuk informasi selengkapnya, silakan lihat Izin Peran Tertaut Layanan di Panduan Pengguna IAM.

Atau, Anda dapat menggunakan kebijakan AWS terkelola untuk menyediakan akses penuh ke Elastic Beanstalk.

Membuat peran yang terhubung dengan layanan untuk Elastic Beanstalk

Anda tidak perlu membuat peran yang terhubung dengan layanan secara manual. Bila Anda membuat lingkungan Elastic Beanstalk menggunakan API Elastic Beanstalk dan tidak menentukan profil instans, Elastic Beanstalk membuat peran yang terhubung dengan layanan untuk Anda.

> **Important**
>
> Peran yang terhubung dengan layanan ini dapat muncul di akun Anda jika Anda menyelesaikan tindakan di layanan lain yang menggunakan fitur yang didukung peran ini. Jika Anda menggunakan layanan Elastic Beanstalk sebelum 18 April 2019, ketika layanan tersebut mulai AWSService RoleForElasticBeanstalkMaintenance mendukung peran terkait layanan, dan akun Anda membutuhkannya, maka Elastic Beanstalk membuat peran tersebut di akun Anda. AWSService RoleForElasticBeanstalkMaintenance Untuk mempelajari lebih lanjut, lihat Peran Baru yang Muncul di Akun IAM Saya.

Jika Anda menghapus peran yang terhubung dengan layanan ini, lalu ingin membuatnya lagi, Anda dapat menggunakan proses yang sama untuk membuat ulang peran tersebut di akun Anda. Ketika Anda membuat lingkungan Elastic Beanstalk menggunakan API Elastic Beanstalk dan tidak menentukan profil instans, Elastic Beanstalk membuat peran yang terhubung dengan layanan bagi Anda kembali.

Mengedit peran yang terhubung dengan layanan untuk Elastic Beanstalk

Elastic Beanstalk tidak memungkinkan Anda untuk mengedit peran terkait layanan. AWSService RoleForElasticBeanstalkMaintenance Setelah membuat peran terkait layanan, Anda tidak dapat mengubah nama peran karena berbagai entitas mungkin merujuk peran tersebut. Namun, Anda dapat mengedit penjelasan peran menggunakan IAM. Untuk informasi lebih lanjut, lihat Mengedit Peran yang Terhubung dengan Layanan di Panduan Pengguna IAM.

Menghapus peran yang terhubung dengan layanan untuk Elastic Beanstalk

Jika Anda tidak perlu lagi menggunakan fitur atau layanan yang memerlukan peran yang terhubung dengan layanan, kami menyarankan Anda menghapus peran tersebut. Dengan begitu, Anda tidak memiliki entitas yang tidak digunakan yang tidak dipantau atau dipelihara secara aktif. Tetapi, Anda harus membersihkan sumber daya peran yang terhubung dengan layanan sebelum menghapusnya secara manual.

Membersihkan peran yang terhubung dengan layanan

Sebelum Anda dapat menggunakan IAM untuk menghapus peran yang terhubung dengan layanan, Anda harus menghentikan terlebih dahulu setiap lingkungan Elastic Beanstalk yang menggunakan peran tersebut.

> **Note**
>
> Jika layanan Elastic Beanstalk menggunakan peran yang terhubung dengan layanan saat Anda mencoba untuk menghentikan lingkungan, maka penghentian mungkin gagal. Jika hal itu terjadi, tunggu beberapa menit dan coba mengoperasikannya lagi.

Untuk mengakhiri lingkungan Elastic Beanstalk yang menggunakan (konsol) AWSService RoleForElasticBeanstalkMaintenance

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS

2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pilih Tindakan, lalu pilih Terminate environment.
4. Gunakan kotak dialog di layar untuk mengonfirmasi penghentian lingkungan.

Lihat eb terminate untuk rincian tentang mengakhiri lingkungan Elastic Beanstalk menggunakan EB CLI.

Lihat TerminateEnvironmentdetail tentang mengakhiri lingkungan Elastic Beanstalk menggunakan API.

Menghapus peran tertaut layanan secara manual

Gunakan konsol IAM, IAM CLI, atau IAM API untuk menghapus peran terkait layanan. AWSService RoleForElasticBeanstalkMaintenance Untuk informasi lebih lanjut, lihat Menghapus Peran yang Terhubung Dengan Layanan dalam Panduan Pengguna IAM.

Wilayah yang didukung untuk peran yang terhubung dengan layanan Elastic Beanstalk

Support Elastic Beanstalk menggunakan peran yang terhubung dengan layanan di semua wilayah tempat layanan tersedia. Untuk informasi lebih lanjut, lihat Kuota dan Titik Akhir AWS Elastic Beanstalk.

Peran yang terhubung dengan layanan permbaharuan yang terkelola

AWS Elastic Beanstalk menggunakan AWS Identity and Access Management peran terkait layanan (IAM). Peran yang terhubung dengan layanan adalah jenis IAM role unik yang terhubung langsung ke Elastic Beanstalk. Peran terkait layanan telah ditentukan sebelumnya oleh Elastic Beanstalk dan mencakup semua izin yang diperlukan layanan untuk memanggil layanan lain atas nama Anda. AWS

Peran yang terhubung dengan layanan memudahkan pengaturan Elastic Beanstalk karena Anda tidak perlu menambahkan izin yang diperlukan secara manual. Elastic Beanstalk menentukan izin peran yang terhubung dengan layanan, kecuali jika ditentukan berbeda, hanya Elastic Beanstalk yang dapat mengasumsikan perannya. Izin yang ditentukan mencakup kebijakan kepercayaan dan kebijakan izin, serta bahwa kebijakan izin tidak dapat dilampirkan ke entitas IAM lainnya.

Anda dapat menghapus peran yang terhubung dengan layanan hanya setelah pertama kali menghapus sumber daya terkait. Ini melindungi sumber daya Elastic Beanstalk Anda karena Anda tidak dapat secara tidak sengaja menghapus izin untuk mengakses sumber daya.

Untuk informasi tentang layanan lain yang mendukung peran yang terhubung dengan layanan, lihat Layanan AWS yang Berfungsi dengan IAM dan cari layanan yang memiliki Ya di kolom Peran yang Terhubung dengan Layanan. Pilih Ya dengan tautan untuk melihat dokumentasi peran yang terhubung dengan layanan untuk layanan tersebut.

Izin peran yang terhubung dengan layanan untuk Elastic Beanstalk

Elastic Beanstalk menggunakan peran terkait layanan AWSServiceRoleForElasticBeanstalkManagedUpdatesbernama — Memungkinkan Elastic Beanstalk melakukan pembaruan platform terjadwal di lingkungan berjalan Anda.

Peran AWSService RoleForElasticBeanstalkManagedUpdates terkait layanan mempercayai layanan berikut untuk mengambil peran:

- `managedupdates.elasticbeanstalk.amazonaws.com`

Kebijakan terkelola AWSElasticBeanstalkManagedUpdatesServiceRolePolicymemungkinkan peran AWSService RoleForElasticBeanstalkManagedUpdates terkait layanan semua izin yang diperlukan Elastic Beanstalk untuk menyelesaikan tindakan pembaruan terkelola atas nama Anda. Untuk melihat konten kebijakan terkelola, lihat AWSElasticBeanstalkManagedUpdatesServiceRolePolicyhalaman di Panduan Referensi Kebijakan AWS Terkelola.

Anda harus mengonfigurasi izin untuk mengizinkan entitas IAM (seperti pengguna, grup, atau peran) untuk membuat, mengedit, atau menghapus peran terkait layanan. Untuk informasi selengkapnya, silakan lihat Izin Peran Tertaut Layanan di Panduan Pengguna IAM.

Atau, Anda dapat menggunakan kebijakan AWS terkelola untuk menyediakan akses penuh ke Elastic Beanstalk.

Membuat peran yang terhubung dengan layanan untuk Elastic Beanstalk

Anda tidak perlu membuat peran yang terhubung dengan layanan secara manual. Ketika Anda membuat lingkungan Elastic Beanstalk menggunakan API Elastic Beanstalk, mengaktifkan pembaruan terkelola, dan menentukan `AWSServiceRoleForElasticBeanstalkManagedUpdates` sebagai nilai untuk opsi `ServiceRoleForManagedUpdates` dari namespace `aws:elasticbeanstalk:managedactions`, Elastic Beanstalk membuat peran yang terhubung dengan layanan untuk Anda.

Saat Elastic Beanstalk mencoba AWSService RoleForElasticBeanstalkManagedUpdates membuat peran terkait layanan untuk akun Anda saat membuat lingkungan, Anda harus memiliki izin. `iam:CreateServiceLinkedRole` Jika Anda tidak memiliki izin ini, pembuatan lingkungan gagal, dan Anda melihat pesan yang menjelaskan masalah.

Sebagai alternatif, pengguna lain dengan izin untuk membuat peran yang terhubung dengan layanan dapat menggunakan IAM untuk membuat peran terkait layanan terlebih dahulu. Anda kemudian dapat menciptakan lingkungan Anda bahkan tanpa izin `iam:CreateServiceLinkedRole`.

Anda (atau pengguna lain) dapat menggunakan konsol IAM untuk membuat peran yang terhubung dengan layanan dengan kasus penggunaan Pembaruan yang Dikelola Elastic Beanstalk. Di IAM CLI atau IAM API, buat peran yang terhubung dengan layanan dengan nama layanan `managedupdates.elasticbeanstalk.amazonaws.com`. Untuk informasi lebih lanjut, lihat Membuat Peran yang Terhubung dengan Layanan di Panduan Pengguna IAM. Jika Anda menghapus peran yang terhubung dengan layanan ini, Anda dapat menggunakan proses yang sama untuk membuat ulang peran tersebut.

Jika Anda menghapus peran yang terhubung dengan layanan ini, lalu ingin membuatnya lagi, Anda dapat menggunakan proses yang sama untuk membuat ulang peran tersebut di akun Anda. Ketika Anda membuat lingkungan Elastic Beanstalk menggunakan API Elastic Beanstalk, mengaktifkan pembaruan terkelola, dan menentukan `AWSServiceRoleForElasticBeanstalkManagedUpdates` sebagai nilai untuk pilihan `ServiceRoleForManagedUpdates` dari namespace `aws:elasticbeanstalk:managedactions`, Elastic Beanstalk membuat peran yang terhubung dengan layanan untuk Anda lagi.

Mengedit peran yang terhubung dengan layanan untuk Elastic Beanstalk

Elastic Beanstalk tidak memungkinkan Anda untuk mengedit peran terkait layanan. AWSService RoleForElasticBeanstalkManagedUpdates Setelah membuat peran terkait layanan, Anda tidak dapat mengubah nama peran karena berbagai entitas mungkin merujuk peran tersebut. Namun, Anda dapat mengedit penjelasan peran menggunakan IAM. Untuk informasi lebih lanjut, lihat Mengedit Peran yang Terhubung dengan Layanan di Panduan Pengguna IAM.

Menghapus peran yang terhubung dengan layanan untuk Elastic Beanstalk

Jika Anda tidak perlu lagi menggunakan fitur atau layanan yang memerlukan peran yang terhubung dengan layanan, kami menyarankan Anda menghapus peran tersebut. Dengan begitu, Anda tidak memiliki entitas yang tidak digunakan yang tidak dipantau atau dipelihara secara aktif. Tetapi, Anda

harus membersihkan sumber daya peran yang terhubung dengan layanan sebelum menghapusnya secara manual.

Membersihkan peran yang terhubung dengan layanan

Sebelum Anda dapat menggunakan IAM untuk menghapus peran yang terhubung dengan layanan, Anda harus memastikan terlebih dahulu bahwa lingkungan Elastic Beanstalk dengan pembaruan terkelola diaktifkan menggunakan peran layanan yang berbeda atau dihentikan.

> **Note**
>
> Jika layanan Elastic Beanstalk menggunakan peran yang terhubung dengan layanan saat Anda mencoba untuk menghentikan lingkungan, maka penghentian mungkin gagal. Jika hal itu terjadi, tunggu beberapa menit dan coba mengoperasikannya lagi.

Untuk mengakhiri lingkungan Elastic Beanstalk yang menggunakan (konsol) AWSService RoleForElasticBeanstalkManagedUpdates

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pilih Tindakan, lalu pilih Hentikan Lingkungan.
4. Gunakan kotak dialog di layar untuk mengonfirmasi penghentian lingkungan.

Lihat eb terminate untuk rincian tentang mengakhiri lingkungan Elastic Beanstalk menggunakan EB CLI.

Lihat TerminateEnvironmentdetail tentang mengakhiri lingkungan Elastic Beanstalk menggunakan API.

Menghapus peran tertaut layanan secara manual

Gunakan konsol IAM, IAM CLI, atau IAM API untuk menghapus peran terkait layanan. AWSService RoleForElasticBeanstalkManagedUpdates Untuk informasi lebih lanjut, lihat Menghapus Peran yang Terhubung Dengan Layanan dalam Panduan Pengguna IAM.

Wilayah yang didukung untuk peran yang terhubung dengan layanan Elastic Beanstalk

Support Elastic Beanstalk menggunakan peran yang terhubung dengan layanan di semua wilayah tempat layanan tersedia. Untuk informasi lebih lanjut, lihat Kuota dan Titik Akhir AWS Elastic Beanstalk.

Mengelola kebijakan pengguna Elastic Beanstalk

AWS Elastic Beanstalk menyediakan dua kebijakan terkelola yang memungkinkan Anda menetapkan akses penuh atau akses hanya-baca ke semua sumber daya yang dikelola Elastic Beanstalk. Anda dapat melampirkan kebijakan ke pengguna atau grup AWS Identity and Access Management (IAM), atau ke peran yang diambil oleh pengguna Anda.

Untuk kebijakan yang dikelola

- AdministratorAccess- AWSElastic Beanstalk - Memberikan pengguna izin administratif penuh untuk membuat, memodifikasi, dan menghapus aplikasi Elastic Beanstalk, versi aplikasi, pengaturan konfigurasi, lingkungan, dan sumber daya dasarnya. Untuk melihat konten kebijakan terkelola, lihat halaman AdministratorAccess- AWSElastic Beanstalk di Panduan Referensi Kebijakan AWS Terkelola.
- AWSElasticBeanstalkReadOnly— Memungkinkan pengguna untuk melihat aplikasi dan lingkungan, tetapi tidak melakukan operasi yang memodifikasinya. Ini menyediakan akses hanya-baca ke semua sumber daya Elastic Beanstalk, dan ke sumber daya lain AWS yang diambil oleh konsol Elastic Beanstalk. Perhatikan bahwa akses hanya-baca tidak mengaktifkan tindakan seperti mengunduh Elastic Beanstalk sehingga Anda dapat membacanya. Hal ini karena log dipentaskan di bucket Amazon S3, di mana Elastic Beanstalk akan memerlukan izin tulis. Lihat contoh di akhir topik ini untuk informasi tentang cara mengaktifkan akses ke log Elastic Beanstalk. Untuk melihat konten kebijakan terkelola, lihat AWSElasticBeanstalkReadOnlyhalaman di Panduan Referensi Kebijakan AWS Terkelola.

> **Important**
>
> Kebijakan terkelola Elastic Beanstalk tidak memberikan izin terperinci—mereka memberikan semua izin yang berpotensi diperlukan untuk bekerja dengan aplikasi Elastic Beanstalk. Dalam beberapa kasus, Anda mungkin ingin membatasi izin kebijakan terkelola kami lebih lanjut. Untuk contoh satu kasus penggunaan, lihatMencegah akses bucket Amazon S3 lintas lingkungan.

Kebijakan terkelola kami juga tidak mencakup izin ke sumber daya khusus yang mungkin Anda tambahkan ke solusi Anda, dan yang tidak dikelola oleh Elastic Beanstalk. Untuk menerapkan izin yang lebih rinci, izin minimum yang diperlukan, atau izin sumber daya kustom, gunakan kebijakan khusus.

Kebijakan terkelola tidak lagi digunakan

Sebelumnya, Elastic Beanstalk mendukung dua kebijakan pengguna terkelola lainnya, dan. AWSElasticBeanstalkFullAccessAWSElasticBeanstalkReadOnlyAccess Kami berencana untuk menghentikan kebijakan-kebijakan sebelumnya. Anda mungkin masih dapat melihat dan menggunakannya di konsol IAM. Namun demikian, kami merekomendasikan Anda beralih ke menggunakan kebijakan pengguna terkelola baru, dan menambahkan kebijakan kustom untuk memberikan izin ke sumber daya kustom, jika Anda memiliki.

Kebijakan untuk integrasi dengan layanan lain

Kami juga menyediakan kebijakan yang lebih terperinci yang memungkinkan Anda mengintegrasikan lingkungan Anda dengan layanan lain, jika Anda lebih suka menggunakannya.

- AWSElasticBeanstalkRoleCWL - Memungkinkan lingkungan untuk mengelola grup CloudWatch log Amazon Logs.
- AWSElasticBeanstalkRoleRDS - Memungkinkan lingkungan untuk mengintegrasikan instans Amazon RDS.
- AWSElasticBeanstalkRoleWorkerTier— Memungkinkan tingkat lingkungan pekerja untuk membuat tabel Amazon DynamoDB dan antrian Amazon SQS.
- AWSElasticBeanstalkRoleECS - Memungkinkan lingkungan Docker multicontainer untuk mengelola cluster Amazon ECS.
- AWSElasticBeanstalkRoleCore— Memungkinkan operasi inti dari lingkungan layanan web.
- AWSElasticBeanstalkRoleSNS - Memungkinkan lingkungan untuk mengaktifkan integrasi topik Amazon SNS.

Untuk melihat sumber JSON untuk kebijakan terkelola tertentu, lihat Panduan Referensi Kebijakan AWS Terkelola.

Mengontrol akses dengan kebijakan terkelola

Anda dapat menggunakan kebijakan terkelola untuk memberikan akses penuh atau akses hanya-baca ke Elastic Beanstalk. Elastic Beanstalk memperbarui kebijakan ini secara otomatis ketika izin tambahan diperlukan untuk mengakses fitur baru.

Untuk menerapkan kebijakan terkelola untuk pengguna atau grup IAM

1. Buka halaman kebijakan di Konsol IAM.
2. Di kotak pencarian, ketik **`AWSElasticBeanstalk`** untuk memfilter kebijakan.
3. Dalam daftar kebijakan, pilih kotak centang di sebelah AWSElasticBeanstalkReadOnlyatau AdministratorAccess- AWSElastic Beanstalk.
4. Pilih Tindakan kebijakan, lalu pilih Lampirkan.
5. Pilih satu atau beberapa pengguna dan grup untuk melampirkan kebijakan. Anda bisa menggunakan menu Filter dan kotak pencarian untuk memfilter daftar entitas utama.
6. Pilih Pasang kebijakan.

Membuat kebijakan pengguna kustom

Anda dapat membuat kebijakan IAM Anda sendiri untuk mengizinkan atau menolak tindakan API Elastic Beanstalk tertentu pada sumber Elastic Beanstalk tertentu, dan untuk mengontrol akses ke sumber daya kustom yang tidak dikelola oleh Elastic Beanstalk. Untuk informasi selengkapnya tentang melampirkan kebijakan ke pengguna atau grup, lihat Bekerja dengan Kebijakan di Panduan Pengguna IAM. Untuk informasi lebih lanjut tentang membuat kebijakan IAM, lihat Membuat Kebijakan IAM dalam Panduan Pengguna IAM.

> **Note**
>
> Meskipun Anda dapat membatasi cara pengguna berinteraksi dengan Elastic APIs Beanstalk, saat ini tidak ada cara yang efektif untuk mencegah pengguna yang memiliki izin untuk membuat sumber daya dasar yang diperlukan untuk membuat sumber daya lain di Amazon dan layanan lainnya. EC2
> Pikirkan kebijakan ini sebagai cara yang efektif untuk mendistribusikan tanggung jawab Elastic Beanstalk, bukan sebagai cara untuk mengamankan semua sumber daya yang mendasarinya.

> **Important**
>
> Jika Anda memiliki kebijakan khusus yang ditetapkan ke peran layanan Elastic Beanstalk, penting bagi Anda untuk menetapkan izin yang tepat untuk templat peluncuran. Jika tidak, Anda mungkin tidak memiliki izin yang diperlukan untuk memperbarui lingkungan atau meluncurkan yang baru. Untuk informasi selengkapnya, lihat Izin yang diperlukan untuk template peluncuran.

Kebijakan IAM berisi pernyataan kebijakan yang menjelaskan izin yang ingin Anda berikan. Ketika Anda membuat pernyataan kebijakan untuk Elastic Beanstalk, Anda perlu memahami bagaimana menggunakan empat bagian berikut dari pernyataan kebijakan:

- Efek menentukan apakah untuk mengizinkan atau menolak tindakan dalam pernyataan.
- Tindakan menentukan operasi API yang ingin Anda kendalikan. Misalnya, gunakan `elasticbeanstalk:CreateEnvironment` untuk menentukan operasi `CreateEnvironment`. Operasi tertentu, seperti menciptakan lingkungan, memerlukan izin tambahan untuk melakukan tindakan tersebut. Untuk informasi selengkapnya, lihat Sumber daya dan kondisi untuk tindakan Elastic Beanstalk.

  > **Note**
  >
  > Untuk menggunakan operasi API `UpdateTagsForResource`, menentukan salah satu dari dua tindakan virtual berikut (atau keduanya) bukan nama operasi API:
  >
  > `elasticbeanstalk:AddTags`
  >
  > Mengontrol izin untuk menelepon `UpdateTagsForResource` dan meneruskan daftar tag untuk add-in di parameter `TagsToAdd`.
  >
  > `elasticbeanstalk:RemoveTags`
  >
  > Mengontrol izin untuk menelepon `UpdateTagsForResource` dan meneruskan daftar kunci tag untuk menghapus di parameter `TagsToRemove`.

- Sumber Daya menentukan sumber daya yang Anda inginkan untuk mengontrol akses. Untuk menentukan sumber Elastic Beanstalk, buat daftar Amazon Resource Name (ARN) dari setiap sumber daya.

- (opsional) Ketentuan menentukan pembatasan izin yang diberikan dalam pernyataan. Untuk informasi selengkapnya, lihat Sumber daya dan kondisi untuk tindakan Elastic Beanstalk.

Bagian berikut menunjukkan beberapa kasus di mana Anda mungkin mempertimbangkan kebijakan pengguna kustom.

Memungkinkan penciptaan lingkungan Elastic Beanstalk terbatas

Kebijakan dalam contoh berikut mengizinkan pengguna untuk memanggil tindakan `CreateEnvironment` untuk membuat lingkungan yang namanya dimulai dengan **`Test`** dengan aplikasi dan versi aplikasi yang ditentukan.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid":"CreateEnvironmentPerm",
      "Action": [
        "elasticbeanstalk:CreateEnvironment"
      ],
      "Effect": "Allow",
      "Resource": [
        "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My First Elastic
 Beanstalk Application/Test*"
      ],
      "Condition": {
        "StringEquals": {
          "elasticbeanstalk:InApplication": ["arn:aws:elasticbeanstalk:us-
east-2:123456789012:application/My First Elastic Beanstalk Application"],
          "elasticbeanstalk:FromApplicationVersion": ["arn:aws:elasticbeanstalk:us-
east-2:123456789012:applicationversion/My First Elastic Beanstalk Application/First
 Release"]
        }
      }
    },
    {
      "Sid":"AllNonResourceCalls",
      "Action":[
        "elasticbeanstalk:CheckDNSAvailability",
        "elasticbeanstalk:CreateStorageLocation"
      ],
      "Effect":"Allow",
```

```
      "Resource":[
        "*"
      ]
    }
  ]
}
```

Kebijakan di atas menunjukkan bagaimana untuk memberikan akses terbatas untuk operasi Elastic Beanstalk. Untuk benar-benar meluncurkan lingkungan, pengguna harus memiliki izin untuk membuat AWS sumber daya yang memberi daya pada lingkungan juga. Sebagai contoh, kebijakan berikut memberikan akses ke set default sumber daya untuk lingkungan server web:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:*",
        "ecs:*",
        "elasticloadbalancing:*",
        "autoscaling:*",
        "cloudwatch:*",
        "s3:*",
        "sns:*",
        "cloudformation:*",
        "sqs:*"
        ],
      "Resource": "*"
    }
  ]
}
```

Memungkinkan akses ke log Elastic Beanstalk yang disimpan di Amazon S3

Kebijakan dalam contoh berikut memungkinkan pengguna untuk menarik Elastic Beanstalk log, tahap mereka di Amazon S3, dan mengambil mereka.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
```

```
      "Action": [
        "s3:DeleteObject",
        "s3:GetObjectAcl",
        "s3:PutObjectAcl"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:s3:::elasticbeanstalk-*"
    }
  ]
}
```

> **Note**
>
> Untuk membatasi izin ini untuk hanya lintasan log, gunakan format sumber daya berikut.
>
> ```
> "arn:aws:s3:::elasticbeanstalk-us-east-2-123456789012/resources/environments/
> logs/*"
> ```

Mengaktifkan pengelolaan aplikasi Elastic Beanstalk tertentu

Kebijakan dalam contoh berikut memungkinkan pengguna untuk mengelola lingkungan dan sumber daya lainnya dalam satu aplikasi Elastic Beanstalk tertentu. Kebijakan tersebut menyangkal tindakan Elastic Beanstalk pada sumber daya aplikasi lain, dan juga menyangkal pembuatan dan penghapusan aplikasi Elastic Beanstalk.

> **Note**
>
> Kebijakan ini tidak menolak akses ke sumber daya apa pun melalui layanan lain. Ini menunjukkan cara yang efektif untuk mendistribusikan tanggung jawab untuk mengelola aplikasi Elastic Beanstalk di antara pengguna yang berbeda, bukan sebagai cara untuk mengamankan sumber daya yang mendasarinya.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": [
```

```
        "elasticbeanstalk:CreateApplication",
        "elasticbeanstalk:DeleteApplication"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Effect": "Deny",
      "Action": [
        "elasticbeanstalk:CreateApplicationVersion",
        "elasticbeanstalk:CreateConfigurationTemplate",
        "elasticbeanstalk:CreateEnvironment",
        "elasticbeanstalk:DeleteApplicationVersion",
        "elasticbeanstalk:DeleteConfigurationTemplate",
        "elasticbeanstalk:DeleteEnvironmentConfiguration",
        "elasticbeanstalk:DescribeApplicationVersions",
        "elasticbeanstalk:DescribeConfigurationOptions",
        "elasticbeanstalk:DescribeConfigurationSettings",
        "elasticbeanstalk:DescribeEnvironmentResources",
        "elasticbeanstalk:DescribeEnvironments",
        "elasticbeanstalk:DescribeEvents",
        "elasticbeanstalk:DeleteEnvironmentConfiguration",
        "elasticbeanstalk:RebuildEnvironment",
        "elasticbeanstalk:RequestEnvironmentInfo",
        "elasticbeanstalk:RestartAppServer",
        "elasticbeanstalk:RetrieveEnvironmentInfo",
        "elasticbeanstalk:SwapEnvironmentCNAMEs",
        "elasticbeanstalk:TerminateEnvironment",
        "elasticbeanstalk:UpdateApplicationVersion",
        "elasticbeanstalk:UpdateConfigurationTemplate",
        "elasticbeanstalk:UpdateEnvironment",
        "elasticbeanstalk:RetrieveEnvironmentInfo",
        "elasticbeanstalk:ValidateConfigurationSettings"
      ],
      "Resource": [
        "*"
      ],
      "Condition": {
        "StringNotEquals": {
          "elasticbeanstalk:InApplication": [
            "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/myapplication"
          ]
        }
```

```
      }
    }
  ]
}
```

Format nama sumber Amazon untuk Elastic Beanstalk

Anda menetapkan sumber daya untuk kebijakan IAM menggunakan sumber daya Amazon Resource Name (ARN). Untuk Beanstalk Elastic Beanstalk, ARN memiliki format berikut.

```
arn:aws:elasticbeanstalk:region:account-id:resource-type/resource-path
```

Di mana:

- *region* adalah wilayah tempat sumber daya berada (misalnya, **us-west-2**).
- *account-id*adalah ID AWS akun, tanpa tanda hubung (misalnya,) **123456789012**
- *resource-type* mengidentifikasi jenis Elastic Beanstalk sumber daya—misalnya, `environment`. Lihat tabel di bawah ini untuk daftar semua jenis sumber daya Elastic Beanstalk.
- *resource-path* adalah bagian yang mengidentifikasi sumber spesifik berdasarkan nama. Sebuah sumber Elastic Beanstalk memiliki jalan yang unik mengidentifikasi sumber daya itu. Lihat tabel di bawah ini untuk format jalur sumber daya untuk setiap jenis sumber daya. Misalnya, lingkungan selalu dikaitkan dengan aplikasi. Jalur sumber daya untuk lingkungan **myEnvironment** dalam aplikasi **myApp** akan terlihat seperti ini:

  ```
  myApp/myEnvironment
  ```

Elastic Beanstalk memiliki beberapa jenis sumber daya yang dapat Anda tentukan dalam kebijakan. Tabel berikut menunjukkan format ARN untuk setiap jenis sumber daya dan contoh.

Jenis sumber daya	Format untuk ARN
`application`	`arn:aws:elasticbeanstalk:` *region*`:`*account-id*` :application/` *application-name* Contoh: **arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App**

Jenis sumber daya	Format untuk ARN
applicationversion	arn:aws:elasticbeanstalk: *region*:*account-id* :applicationversion/ *application-name* /*version-label* Contoh: **arn:aws:elasticbeanstalk:us-east-2:123456789012:applicationversion/My App/My Version**
configurationtemplate	arn:aws:elasticbeanstalk: *region*:*account-id* :configurationtemplate/ *application-name* /*template-name* Contoh: **arn:aws:elasticbeanstalk:us-east-2:123456789012:configurationtemplate/My App/My Template**
environment	arn:aws:elasticbeanstalk: *region*:*account-id* :environment/ *application-name* /*environment-name* Contoh: **arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/MyEnvironment**
platform	arn:aws:elasticbeanstalk: *region*:*account-id* :platform/ *platform-name* /*platform-version* Contoh: **arn:aws:elasticbeanstalk:us-east-2:123456789012:platform/MyPlatform/1.0**
solutionstack	arn:aws:elasticbeanstalk: *region*::solutionstack/ *solutionstack-name* Contoh: **arn:aws:elasticbeanstalk:us-east-2::solutionstack/32bit Amazon Linux running Tomcat 7**

Lingkungan, versi aplikasi, dan templat konfigurasi selalu terkandung dalam aplikasi tertentu. Anda akan melihat bahwa semua sumber daya ini memiliki nama aplikasi di jalur sumber daya mereka sehingga mereka diidentifikasi secara unik oleh nama sumber daya mereka dan aplikasi yang memuatnya. Meskipun tumpukan solusi digunakan oleh templat dan lingkungan konfigurasi,

tumpukan solusi tidak spesifik untuk aplikasi atau AWS akun dan tidak memiliki aplikasi atau AWS akun di dalamnya. ARNs

Sumber daya dan kondisi untuk tindakan Elastic Beanstalk

Bagian ini menjelaskan sumber daya dan kondisi yang dapat Anda gunakan dalam pernyataan kebijakan untuk memberikan izin yang memungkinkan tindakan tertentu Elastic Beanstalk untuk dilakukan pada sumber daya tertentu Elastic Beanstalk.

Kondisi memungkinkan Anda untuk menentukan izin ke sumber daya yang perlu diselesaikan oleh tindakan. Misalnya, saat Anda dapat menghubungi tindakan `CreateEnvironment`, Anda juga harus menentukan versi aplikasi untuk men-deploy serta aplikasi yang berisi nama aplikasi tersebut. Ketika Anda menetapkan izin untuk tindakan `CreateEnvironment`, Anda menentukan aplikasi dan aplikasi versi yang Anda ingin tindakan untuk bertindak dengan menggunakan syarat `InApplication` dan `FromApplicationVersion`.

Selain itu, Anda dapat menentukan konfigurasi lingkungan dengan tumpukan solusi (`FromSolutionStack`) atau templat konfigurasi (`FromConfigurationTemplate`). Pernyataan kebijakan berikut memungkinkan tindakan `CreateEnvironment` untuk membuat lingkungan dengan nama **`myenv`** (ditentukan oleh `Resource`) di aplikasi **`My App`** (ditentukan oleh syarat `InApplication`) menggunakan versi aplikasi **`My Version`** (`FromApplicationVersion`) dengan konfigurasi **`32bit Amazon Linux running Tomcat 7`** (`FromSolutionStack`):

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "elasticbeanstalk:CreateEnvironment"
      ],
      "Effect": "Allow",
      "Resource": [
        "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv"
      ],
      "Condition": {
        "StringEquals": {
          "elasticbeanstalk:InApplication": ["arn:aws:elasticbeanstalk:us-
east-2:123456789012:application/My App"],
          "elasticbeanstalk:FromApplicationVersion": ["arn:aws:elasticbeanstalk:us-
east-2:123456789012:applicationversion/My App/My Version"],
```

```
          "elasticbeanstalk:FromSolutionStack": ["arn:aws:elasticbeanstalk:us-
east-2::solutionstack/32bit Amazon Linux running Tomcat 7"]
        }
      }
    }
  ]
}
```

> **Note**
>
> Sebagian besar kunci kondisi yang disebutkan dalam topik ini khusus untuk Elastic Beanstalk, dan namanya berisi prefiks `elasticbeanstalk:`. Untuk singkatnya, kita menghilangkan prefiks ini dari nama kunci kondisi ketika kita menyebutkan mereka di bagian berikut. Sebagai contoh, kami sebutkan `InApplication` alih-alih nama lengkapnya `elasticbeanstalk:InApplication`.
> Sebaliknya, kami menyebutkan beberapa kunci kondisi yang digunakan di seluruh AWS layanan, dan kami menyertakan `aws:` awalan mereka untuk menyoroti pengecualian. Contoh kebijakan selalu menampilkan nama kunci kondisi lengkap, termasuk prefiks.

Bagian

- Informasi kebijakan untuk tindakan Elastic Beanstalk
- Kunci kondisi untuk tindakan Elastic Beanstalk

Informasi kebijakan untuk tindakan Elastic Beanstalk

Tabel berikut mencantumkan semua tindakan Elastic Beanstalk, sumber daya yang setiap tindakan bertindak atas, dan informasi kontekstual tambahan yang dapat diberikan menggunakan syarat.

Informasi kebijakan untuk tindakan Elastic Beanstalk, termasuk sumber daya, kondisi, contoh, dan dependensi

Sumber Daya	Kondisi	Pernyataan Contoh
Tindakan: `AbortEnvironmentUpdate`		

`application`
 	
`environment`
	`aws:ResourceTag/` *`key-name`* (Opsional)	Kebijakan berikut ini memungkinkan pengguna untuk membatalkan operasi pembaruan lingkungan pada lingkungan di aplikasi bernama `My App`.

Sumber Daya	Kondisi	Pernyataan Contoh
	`aws:TagKeys` (Opsional)	`{ "Version": "2012-10-17", "Statement": [ { "Action": [ "elasticbeanstalk:AbortEnvironmentUpdate" ], "Effect": "Allow", "Resource": [ "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App" ] } ] }`
Tindakan: `CheckDNSAvailability`		
`"*"`	N/A	`{ "Version": "2012-10-17", "Statement": [ { "Action": [ "elasticbeanstalk:CheckDNSAvailability" ], "Effect": "Allow", "Resource": "*" } ] }`
Tindakan: `ComposeEnvironments`		

Sumber Daya	Kondisi	Pernyataan Contoh
`application`	`aws:ResourceTag/` *key-name* (Opsional) `aws:TagKeys` (Opsional)	Kebijakan berikut ini memungkinkan pengguna untuk menyusun lingkungan milik aplikasi bernama `My App`. `{ "Version": "2012-10-17", "Statement": [ { "Action": [ "elasticbeanstalk:ComposeEnvironments" ], "Effect": "Allow", "Resource": [ "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App" ] } ] }`
Tindakan: `CreateApplication`		

Sumber Daya	Kondisi	Pernyataan Contoh

`application`	`aws:RequestTag/` *`key-name`* (Opsional)
	
	`aws:TagKeys` (Opsional)
		Contoh ini memungkinkan tindakan `CreateApplication` untuk membuat aplikasi yang namanya dimulai dengan **`DivA`**:
		
		`{`
		`  "Version": "2012-10-17",`
		`  "Statement": [`
		`  {`
		`  "Action": [`
		`  "elasticbeanstalk:CreateApplication"`
		`  ],`
		`  "Effect": "Allow",`
		`  "Resource": [`
		`  "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/DivA*"`
		`  ]`
		`  }`
		`  ]`
		`}`

Tindakan: `CreateApplicationVersion`		

Sumber Daya	Kondisi	Pernyataan Contoh
`applicationversion`	`InApplication` `aws:RequestTag/` *key-name* (Opsional) `aws:TagKeys` (Opsional)	Contoh ini memungkinkan `CreateApplicationVersion` untuk membuat versi aplikasi dengan nama apa pun (*****) di aplikasi **My App**: `{ "Version": "2012-10-17", "Statement": [{ "Action": ["elasticbeanstalk:CreateApplicationVersion"], "Effect": "Allow", "Resource": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:applicationversion/My App/*"], "Condition": { "StringEquals": { "elasticbeanstalk:InApplication": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"] } } }] }`

Tindakan: `CreateConfigurationTemplate`

Sumber Daya	Kondisi	Pernyataan Contoh

`configurationtemplate`	`InApplication`
	
	`FromApplication`
	
	`FromApplicationVersion`
	
	`FromConfigurationTemplate`
	
	`FromEnvironment`
	
	`FromSolutionStack`
	
	`aws:RequestTag/` *key-name* (Opsional)
	
	`aws:TagKeys` (Opsional)
		Kebijakan berikut memungkinkan `CreateConfigurationTemplate` untuk membuat cetakan konfigurasi yang namanya dimulai dengan **`My Template`** (`My Template*`) di aplikasi **`My App`**:
		
		`{ "Version": "2012-10-17", "Statement": [{ "Action": ["elasticbeanstalk:CreateConfigurationTemplate"], "Effect": "Allow", "Resource": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:configurationtemplate/My App/My Template*"], "Condition": { "StringEquals": { "elasticbeanstalk:InApplication": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"], "elasticbeanstalk:FromSolutionStack": ["arn:aws:elasticbeanstalk:us-east-2::solutionstack/32bit Amazon Linux running Tomcat 7"] } } }] }`

Tindakan: `CreateEnvironment`		

Sumber Daya	Kondisi	Pernyataan Contoh

`environment`	`InApplication`
	
	`FromApplicationVersion`
	
	`FromConfigurationTemplate`
	
	`FromSolutionStack`
	
	`aws:RequestTag/` *key-name* (Opsional)
	
	`aws:TagKeys` (Opsional)
		Kebijakan berikut memungkinkan tindakan `CreateEnvironment` untuk membuat lingkungan yang namanya **`myenv`** dalam aplikasi **`My App`** dan menggunakan **`32bit Amazon Linux running Tomcat 7`** tumpukan solusi:
		
		`{ "Version": "2012-10-17", "Statement": [{ "Action": ["elasticbeanstalk:CreateEnvironment"], "Effect": "Allow", "Resource": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv"], "Condition": { "StringEquals": { "elasticbeanstalk:InApplication": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"], "elasticbeanstalk:FromApplicationVersion": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:applicationversion/My App/My Version"], "elasticbeanstalk:FromSolutionStack": ["arn:aws:elasticbeanstalk:us-east-2::solutionstack/32bit Amazon Linux running Tomcat 7"] } } }] }`

Sumber Daya	Kondisi	Pernyataan Contoh
Tindakan: `CreatePlatformVersion`		
`platform`	`aws:RequestTag/` *key-name* (Opsional) `aws:TagKeys` (Opsional)	Contoh ini mengizinkan tindakan `CreatePlatformVersion` untuk membuat versi platform yang menargetkan wilayah `us-east-2` , yang namanya dimulai dengan **`us-east-2_`** : `{ "Version": "2012-10-17", "Statement": [ { "Action": [ "elasticbeanstalk:CreatePlatformVersion" ], "Effect": "Allow", "Resource": [ "arn:aws:elasticbeanstalk:us-east-2:123456789012:platform/us-east-2_*" ] } ] }`
Tindakan: `CreateStorageLocation`		

Sumber Daya	Kondisi	Pernyataan Contoh
`"*"`	N/A	`{ "Version": "2012-10-17", "Statement": [ { "Action": [ "elasticbeanstalk:CreateStorageLocation" ], "Effect": "Allow", "Resource": "*" } ] }`
Tindakan: `DeleteApplication`		
`application`	`aws:ResourceTag/` *key-name* (Opsional) `aws:TagKeys` (Opsional)	Kebijakan berikut memungkinkan tindakan `DeleteApplication` untuk menghapus aplikasi **My App**: `{ "Version": "2012-10-17", "Statement": [ { "Action": [ "elasticbeanstalk:DeleteApplication" ], "Effect": "Allow", "Resource": [ "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App" ] } ] }`
Tindakan: `DeleteApplicationVersion`		

Sumber Daya	Kondisi	Pernyataan Contoh
`applicationversion`	`InApplication` `aws:ResourceTag/` *key-name* (Opsional) `aws:TagKeys` (Opsional)	Kebijakan berikut memungkinkan tindakan `DeleteApplicationVersion` untuk menghapus versi aplikasi yang namanya **`My Version`** dalam aplikasi **`My App`**: `{ "Version": "2012-10-17", "Statement": [{ "Action": ["elasticbeanstalk:DeleteApplicationVersion"], "Effect": "Allow", "Resource": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:applicationversion/My App/My Version"], "Condition": { "StringEquals": { "elasticbeanstalk:InApplication": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"] } } }] }`

Tindakan: `DeleteConfigurationTemplate`

Sumber Daya	Kondisi	Pernyataan Contoh
`configurationtemplate`	`InApplication` (Opsional) `aws:ResourceTag/` *key-name* (Opsional) `aws:TagKeys` (Opsional)	Kebijakan berikut memungkinkan `DeleteConfigurationTemplate` untuk menghapus templat konfigurasi yang namanya **`My Template`** dalam aplikasi **`My App`**. Menentukan nama aplikasi sebagai syarat adalah opsional. `{ "Version": "2012-10-17", "Statement": [ { "Action": [ "elasticbeanstalk:DeleteConfigurationTemplate" ], "Effect": "Allow", "Resource": [ "arn:aws:elasticbeanstalk:us-east-2:123456789012:configurationtemplate/My App/My Template" ] } ] }`

Tindakan: `DeleteEnvironmentConfiguration`

Sumber Daya	Kondisi	Pernyataan Contoh
`environment`	`InApplication` (Opsional)	Kebijakan berikut memungkinkan tindakan `DeleteEnvironmentConfiguration` untuk menghapus konfigurasi draf untuk lingkungan **`myenv`** dalam aplikasi **`My App`**. Menentukan nama aplikasi sebagai syarat adalah opsional. `{ "Version": "2012-10-17", "Statement": [ { "Action": [ "elasticbeanstalk:DeleteEnvironmentConfiguration" ], "Effect": "Allow", "Resource": [ "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv" ] } ] }`
Tindakan: `DeletePlatformVersion`		

Sumber Daya	Kondisi	Pernyataan Contoh
`platform`	`aws:ResourceTag/` *key-name* (Opsional) `aws:TagKeys` (Opsional)	Kebijakan berikut memungkinkan tindakan `DeletePlatformVersion` untuk menghapus versi platform yang menargetkan wilayah `us-east-2`, yang namanya dimulai dengan **`us-east-2_`**: `{ "Version": "2012-10-17", "Statement": [ { "Action": [ "elasticbeanstalk:DeletePlatformVersion" ], "Effect": "Allow", "Resource": [ "arn:aws:elasticbeanstalk:us-east-2:123456789012:platform/us-east-2_*" ] } ] }`
Tindakan: `DescribeApplications`		

Sumber Daya	Kondisi	Pernyataan Contoh
`application`	`aws:ResourceTag/` *key-name* (Opsional) `aws:TagKeys` (Opsional)	Kebijakan berikut memungkinkan tindakan `DescribeApplications` untuk menggambarkan aplikasi sebagai Aplikasi Saya. `{ "Version": "2012-10-17", "Statement": [ { "Action": [ "elasticbeanstalk:DescribeApplications" ], "Effect": "Allow", "Resource": [ "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App" ] } ] }`
Tindakan: `DescribeApplicationVersions`		

Sumber Daya	Kondisi	Pernyataan Contoh
`applicationversion`	`InApplication` (Opsional) `aws:ResourceTag/` *`key-name`* (Opsional) `aws:TagKeys` (Opsional)	Kebijakan berikut memungkinkan tindakan `DescribeApplicationVersions` untuk menggambarkan versi aplikasi **`My Version`** dalam aplikasi **`My App`**. Menentukan nama aplikasi sebagai syarat adalah opsional. `{ "Version": "2012-10-17", "Statement": [ { "Action": [ "elasticbeanstalk:DescribeApplicationVersions" ], "Effect": "Allow", "Resource": [ "arn:aws:elasticbeanstalk:us-east-2:123456789012:applicationversion/My App/My Version" ] } ] }`

Tindakan: `DescribeConfigurationOptions`

Sumber Daya	Kondisi	Pernyataan Contoh

`environment`
 	
`configurationtemplate`
 	
`solutionstack`
	`InApplication` (Opsional)
	
	`aws:ResourceTag/` *`key-name`* (Opsional)
	
	`aws:TagKeys` (Opsional)
		Kebijakan berikut memungkinkan tindakan `DescribeConfigurationOptions` untuk menggambarkan opsi konfigurasi untuk lingkungan **`myenv`** dalam aplikasi **`My App`**. Menentukan nama aplikasi sebagai syarat adalah opsional.
		
		`{`
		`  "Version": "2012-10-17",`
		`  "Statement": [`
		`  {`
		`  "Action": "elasticbeanstalk:DescribeConfigurationOptions",`
		`  "Effect": "Allow",`
		`  "Resource": [`
		`  "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv"`
		`  ]`
		`  }`
		`  ]`
		`}`

Tindakan: `DescribeConfigurationSettings`		

Sumber Daya	Kondisi	Pernyataan Contoh

`environment`
 	
`configurationtemplate`
	`InApplication` (Opsional)
	
	`aws:ResourceTag/` *`key-name`* (Opsional)
	
	`aws:TagKeys` (Opsional)
		Kebijakan berikut memungkinkan tindakan `DescribeConfigurationSettings` untuk menggambarkan pengaturan konfigurasi untuk lingkungan **`myenv`** dalam aplikasi **`My App`**. Menentukan nama aplikasi sebagai syarat adalah opsional.
		
		`{`
		`  "Version": "2012-10-17",`
		`  "Statement": [`
		`  {`
		`  "Action": "elasticbeanstalk:DescribeConfigurationSettings",`
		`  "Effect": "Allow",`
		`  "Resource": [`
		`  "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv"`
		`  ]`
		`  }`
		`  ]`
		`}`

Tindakan: `DescribeEnvironmentHealth`		

Sumber Daya	Kondisi	Pernyataan Contoh
`environment`	`aws:ResourceTag/` *key-name* (Opsional) `aws:TagKeys` (Opsional)	Kebijakan berikut memungkinkan penggunaan `DescribeEnvironmentHealth` untuk mengambil informasi kondisi untuk suatu lingkungan bernama **`myenv`**. `{ "Version": "2012-10-17", "Statement": [ { "Action": "elasticbeanstalk:DescribeEnvironmentHealth", "Effect": "Allow", "Resource": [ "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv" ] } ] }`
Tindakan: `DescribeEnvironmentResources`		

Sumber Daya	Kondisi	Pernyataan Contoh
`environment`	`InApplication` (Opsional) `aws:ResourceTag/` *`key-name`* (Opsional) `aws:TagKeys` (Opsional)	Kebijakan berikut memungkinkan `DescribeEnvironmentResources` tindakan untuk mengembalikan daftar AWS sumber daya untuk lingkungan **`myenv`** dalam aplikasi**`My App`**. Menentukan nama aplikasi sebagai syarat adalah opsional. `{ "Version": "2012-10-17", "Statement": [ { "Action": "elasticbeanstalk:DescribeEnvironmentResources", "Effect": "Allow", "Resource": [ "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv" ] } ] }`
Tindakan: `DescribeEnvironments`		

Sumber Daya	Kondisi	Pernyataan Contoh
`environment`	`InApplication` (Opsional) `aws:ResourceTag/` *key-name* (Opsional) `aws:TagKeys` (Opsional)	Kebijakan berikut memungkinkan tindakan `DescribeEnvironments` untuk menggambarkan lingkungan **`myenv`** dan **`myotherenv`** dalam aplikasi **`My App`**. Menentukan nama aplikasi sebagai syarat adalah opsional. `{ "Version": "2012-10-17", "Statement": [ { "Action": "elasticbeanstalk:DescribeEnvironments", "Effect": "Allow", "Resource": [ "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv", "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App2/myotherenv" ] } ] }`
Tindakan: `DescribeEvents`		

Sumber Daya	Kondisi	Pernyataan Contoh

`application`
 	
`applicationversion`
 	
`configurationtemplate`
 	
`environment`
	`InApplication`
	
	`aws:ResourceTag/` *key-name* (Opsional)
	
	`aws:TagKeys` (Opsional)
		Kebijakan berikut memungkinkan tindakan `DescribeEvents` untuk daftar deskripsi peristiwa untuk lingkungan **`myenv`** dan versi aplikasi **`My Version`** dalam aplikasi **`My App`**.
		
		`{ "Version": "2012-10-17", "Statement": [{ "Action": "elasticbeanstalk:DescribeEvents", "Effect": "Allow", "Resource": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv", "arn:aws:elasticbeanstalk:us-east-2:123456789012:applicationversion/My App/My Version"], "Condition": { "StringEquals": { "elasticbeanstalk:InApplication": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"] } } }] }`

Tindakan: `DescribeInstancesHealth`		

Sumber Daya	Kondisi	Pernyataan Contoh
`environment`	N/A	Kebijakan berikut memungkinkan penggunaan `DescribeInstancesHealth` untuk mengambil informasi kondisi untuk instans di lingkungan bernama **`myenv`**. `{ "Version": "2012-10-17", "Statement": [ { "Action": "elasticbeanstalk:DescribeInstancesHealth", "Effect": "Allow", "Resource": [ "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv" ] } ] }`
Tindakan: `DescribePlatformVersion`		

Sumber Daya	Kondisi	Pernyataan Contoh

`platform`	`aws:ResourceTag/` *`key-name`* (Opsional)
	
	`aws:TagKeys` (Opsional)
		Kebijakan berikut memungkinkan tindakan `DescribePlatformVersion` untuk menggambarkan versi platform yang menargetkan wilayah `us-east-2`, yang namanya dimulai dengan **`us-east-2_`**:
		
		`{`
		`  "Version": "2012-10-17",`
		`  "Statement": [`
		`  {`
		`  "Action": [`
		`  "elasticbeanstalk:DescribePlatformVersion"`
		`  ],`
		`  "Effect": "Allow",`
		`  "Resource": [`
		`  "arn:aws:elasticbeanstalk:us-east-2:123456789012:platform/us-east-2_*"`
		`  ]`
		`  }`
		`  ]`
		`}`

Tindakan: `ListAvailableSolutionStacks`		

Sumber Daya	Kondisi	Pernyataan Contoh
`solutions tack`	N/A	Kebijakan berikut memungkinkan tindakan `ListAvailableSolutionStacks` untuk mengembalikan hanya solusi tumpukan **`32bit Amazon Linux running Tomcat 7`**. `{ "Version": "2012-10-17", "Statement": [ { "Action": [ "elasticbeanstalk:ListAvailableSolutionStacks" ], "Effect": "Allow", "Resource": "arn:aws:elasticbeanstalk:us-east-2::solutionstack/32bit Amazon Linux running Tomcat 7" } ] }`
Tindakan: `ListPlatformVersions`		

Sumber Daya	Kondisi	Pernyataan Contoh
`platform`	`aws:RequestTag/` *key-name* (Opsional) `aws:TagKeys` (Opsional)	Contoh ini mengizinkan tindakan `CreatePlatformVersion` untuk membuat versi platform yang menargetkan wilayah `us-east-2` , yang namanya dimulai dengan **`us-east-2_`** : `{ "Version": "2012-10-17", "Statement": [ { "Action": [ "elasticbeanstalk:ListPlatformVersions" ], "Effect": "Allow", "Resource": [ "arn:aws:elasticbeanstalk:us-east-2:123456789012:platform/us-east-2_*" ] } ] }`
Tindakan: `ListTagsForResource`		

Sumber Daya	Kondisi	Pernyataan Contoh

`application`
 	
`applicationversion`
 	
`configurationtemplate`
 	
`environment`
 	
`platform`
	`aws:ResourceTag/` *`key-name`* (Opsional)
	
	`aws:TagKeys` (Opsional)
		Kebijakan berikut mengizinkan tindakan `ListTagsForResource` untuk mencantumkan tag dari sumber daya yang ada hanya jika mereka memiliki sebuah tag bernama `stage` dengan nilai `test`:
		
		`{`
		`  "Version": "2012-10-17",`
		`  "Statement": [`
		`  {`
		`  "Action": [`
		`  "elasticbeanstalk:ListTagsForResource"`
		`  ],`
		`  "Effect": "Allow",`
		`  "Resource": "*",`
		`  "Condition": {`
		`  "StringEquals": {`
		`  "aws:ResourceTag/stage": ["test"]`
		`  }`
		`  }`
		`  }`
		`  ]`
		`}`

Tindakan: `RebuildEnvironment`		

Sumber Daya	Kondisi	Pernyataan Contoh
`environment`	`InApplication` `aws:ResourceTag/` *key-name* (Opsional) `aws:TagKeys` (Opsional)	Kebijakan berikut mengizinkan tindakan `RebuildEnvironment` untuk membangun kembali lingkungan **`myenv`** dalam aplikasi **`My App`**. `{ "Version": "2012-10-17", "Statement": [{ "Action": ["elasticbeanstalk:RebuildEnvironment"], "Effect": "Allow", "Resource": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv"], "Condition": { "StringEquals": { "elasticbeanstalk:InApplication": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"] } } }] }`
Tindakan: `RequestEnvironmentInfo`		

Sumber Daya	Kondisi	Pernyataan Contoh
`environment`	`InApplication` `aws:ResourceTag/` *key-name* (Opsional) `aws:TagKeys` (Opsional)	Kebijakan berikut mengizinkan tindakan `RequestEnvironmentInfo` untuk mengkompilasi informasi tentang lingkungan **`myenv`** dalam aplikasi **`My App`**. `{ "Version": "2012-10-17", "Statement": [{ "Action": ["elasticbeanstalk:RequestEnvironmentInfo"], "Effect": "Allow", "Resource": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv"], "Condition": { "StringEquals": { "elasticbeanstalk:InApplication": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"] } } }] }`
Tindakan: `RestartAppServer`		

Sumber Daya	Kondisi	Pernyataan Contoh
`environment`	`InApplication`	Kebijakan berikut mengizinkan tindakan `RestartAppServer` untuk memulai ulang server kontainer aplikasi untuk lingkungan **`myenv`** dalam aplikasi **`My App`**. `{ "Version": "2012-10-17", "Statement": [{ "Action": ["elasticbeanstalk:RestartAppServer"], "Effect": "Allow", "Resource": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv"], "Condition": { "StringEquals": { "elasticbeanstalk:InApplication": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"] } } }] }`
Tindakan: `RetrieveEnvironmentInfo`		

Sumber Daya	Kondisi	Pernyataan Contoh
`environment`	`InApplication` `aws:ResourceTag/` *key-name* (Opsional) `aws:TagKeys` (Opsional)	Kebijakan berikut mengizinkan tindakan `RetrieveEnvironmentInfo` untuk mengambil informasi yang dikompilasi untuk lingkungan **`myenv`** dalam aplikasi **`My App`**. `{ "Version": "2012-10-17", "Statement": [{ "Action": ["elasticbeanstalk:RetrieveEnvironmentInfo"], "Effect": "Allow", "Resource": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv"], "Condition": { "StringEquals": { "elasticbeanstalk:InApplication": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"] } } }] }`
Tindakan: `SwapEnvironmentCNAMEs`		

Sumber Daya	Kondisi	Pernyataan Contoh
`environment`	`InApplication` (Opsional) `FromEnvironment` (Opsional)	Kebijakan berikut memungkinkan `SwapEnvironmentCNAMEs` tindakan CNAMEs untuk menukar lingkungan **`mysrcenv`** dan**`mydestenv`** . `{ "Version": "2012-10-17", "Statement": [ { "Action": [ "elasticbeanstalk:SwapEnvironmentCNAMEs" ], "Effect": "Allow", "Resource": [ "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/mysrcenv", "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/mydestenv" ] } ] }`
Tindakan: `TerminateEnvironment`		

Sumber Daya	Kondisi	Pernyataan Contoh
`environment`	`InApplication` `aws:ResourceTag/` *key-name* (Opsional) `aws:TagKeys` (Opsional)	Kebijakan berikut memungkinkan tindakan `TerminateEnvironment` untuk mengakhiri lingkungan **`myenv`** dalam aplikasi **`My App`**. `{ "Version": "2012-10-17", "Statement": [{ "Action": ["elasticbeanstalk:TerminateEnvironment"], "Effect": "Allow", "Resource": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv"], "Condition": { "StringEquals": { "elasticbeanstalk:InApplication": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"] } } }] }`
Tindakan: UpdateApplication		

Sumber Daya	Kondisi	Pernyataan Contoh

`application`	`aws:ResourceTag/` *`key-name`* (Opsional)
	
	`aws:TagKeys` (Opsional)
		Kebijakan berikut memungkinkan tindakan `UpdateApplication` untuk memperbarui properti dari aplikasi **`My App`**.
		
		`{`
		`  "Version": "2012-10-17",`
		`  "Statement": [`
		`  {`
		`  "Action": [`
		`  "elasticbeanstalk:UpdateApplication"`
		`  ],`
		`  "Effect": "Allow",`
		`  "Resource": [`
		`  "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"`
		`  ]`
		`  }`
		`  ]`
		`}`

Tindakan: UpdateApplicationResourceLifecycle		

Sumber Daya	Kondisi	Pernyataan Contoh
`application`	`aws:ResourceTag/` *`key-name`* (Opsional) `aws:TagKeys` (Opsional)	Kebijakan berikut memungkinkan tindakan `UpdateApplicationResourceLifecycle` untuk memperbarui pengaturan siklus hidup aplikasi **`My App`**. `{ "Version": "2012-10-17", "Statement": [ { "Action": [ "elasticbeanstalk:UpdateApplicationResourceLifecycle" ], "Effect": "Allow", "Resource": [ "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App" ] } ] }`
Tindakan: `UpdateApplicationVersion`		

Sumber Daya	Kondisi	Pernyataan Contoh
`applicationversion`	`InApplication` `aws:ResourceTag/` *key-name* (Opsional) `aws:TagKeys` (Opsional)	Kebijakan berikut memungkinkan tindakan `UpdateApplicationVersion` untuk memperbarui properti dari versi aplikasi **`My Version`** dalam aplikasi **`My App`**. `{ "Version": "2012-10-17", "Statement": [{ "Action": ["elasticbeanstalk:UpdateApplicationVersion"], "Effect": "Allow", "Resource": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:applicationversion/My App/My Version"], "Condition": { "StringEquals": { "elasticbeanstalk:InApplication": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"] } } }] }`
Tindakan: `UpdateConfigurationTemplate`		

Sumber Daya	Kondisi	Pernyataan Contoh

`configurationtemplate`	`InApplication`
	
	`aws:ResourceTag/` *key-name* (Opsional)
	
	`aws:TagKeys` (Opsional)
		Kebijakan berikut memungkinkan tindakan `UpdateConfigurationTemplate` untuk memperbarui properti atau pilihan dari templat konfigurasi **`My Template`** dalam aplikasi **`My App`**.
		
		`{`
		`  "Version": "2012-10-17",`
		`  "Statement": [`
		`  {`
		`  "Action": [`
		`  "elasticbeanstalk:UpdateConfigurationTemplate"`
		`  ],`
		`  "Effect": "Allow",`
		`  "Resource": [`
		`  "arn:aws:elasticbeanstalk:us-east-2:123456789012:configurationtemplate/My App/My Template"`
		`  ],`
		`  "Condition": {`
		`  "StringEquals": {`
		`  "elasticbeanstalk:InApplication": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"]`
		`  }`
		`  }`
		`  }`
		`  ]`
		`}`

Tindakan: `UpdateEnvironment`		

Sumber Daya	Kondisi	Pernyataan Contoh

`environment`	`InApplication`
	
	`FromApplicationVersion`
	
	`FromConfigurationTemplate`
	
	`aws:ResourceTag/` *key-name* (Opsional)
	
	`aws:TagKeys` (Opsional)
		Kebijakan berikut memungkinkan tindakan `UpdateEnvironment` untuk memperbarui lingkungan **`myenv`**dalam aplikasi **`My App`** dengan men-deploy versi aplikasi **`My Version`**.
		
		`{ "Version": "2012-10-17", "Statement": [{ "Action": ["elasticbeanstalk:UpdateEnvironment"], "Effect": "Allow", "Resource": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv"], "Condition": { "StringEquals": { "elasticbeanstalk:InApplication": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"], "elasticbeanstalk:FromApplicationVersion": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:applicationversion/My App/My Version"] } } }] }`

Tindakan: **`UpdateTagsForResource`** — AddTags		

Sumber Daya	Kondisi	Pernyataan Contoh

`application`
 	
`applicationversion`
 	
`configurationtemplate`
 	
`environment`
 	
`platform`
	`aws:ResourceTag/` *key-name* (Opsional)
	
	`aws:RequestTag/` *key-name* (Opsional)
	
	`aws:TagKeys` (Opsional)
		Tindakan `AddTags` adalah salah satu dari dua tindakan virtual yang terkait dengan API `UpdateTagsForResource`.
		
		Kebijakan berikut memungkinkan tindakan `AddTags` untuk memodifikasi tag dari sumber daya yang ada hanya jika mereka memiliki tag bernama `stage` dengan nilai `test`:
		
		`{ "Version": "2012-10-17", "Statement": [ { "Action": [ "elasticbeanstalk:AddTags" ], "Effect": "Allow", "Resource": "*", "Condition": { "StringEquals": { "aws:ResourceTag/stage": ["test"] } } } ] }`

Tindakan: **`UpdateTagsForResource`** — `RemoveTags`		

Sumber Daya	Kondisi	Pernyataan Contoh

`application`
 	
`applicationversion`
 	
`configurationtemplate`
 	
`environment`
 	
`platform`
	`aws:ResourceTag/` *`key-name`* (Opsional)
	
	`aws:TagKeys` (Opsional)
		Tindakan `RemoveTags` adalah salah satu dari dua tindakan virtual yang terkait dengan API `UpdateTagsForResource`.
		
		Kebijakan berikut menyangkal tindakan `RemoveTags` untuk meminta penghapusan dari sebuah tag bernama `stage` dari sumber daya yang ada:
		
		`{`
		`  "Version": "2012-10-17",`
		`  "Statement": [`
		`  {`
		`  "Action": [`
		`  "elasticbeanstalk:RemoveTags"`
		`  ],`
		`  "Effect": "Deny",`
		`  "Resource": "*",`
		`  "Condition": {`
		`  "ForAnyValue:StringEquals": {`
		`  "aws:TagKeys": ["stage"]`
		`  }`
		`  }`
		`  }`
		`  ]`
		`}`

Tindakan: `ValidateConfigurationSettings`		

Sumber Daya	Kondisi	Pernyataan Contoh

`template`
 	
`environment`
	`InApplication`
	
	`aws:ResourceTag/` *key-name* (Opsional)
	
	`aws:TagKeys` (Opsional)
		Kebijakan berikut memungkinkan tindakan `ValidateConfigurationSettings` untuk memvalidasi pengaturan konfigurasi terhadap lingkungan **`myenv`** dalam aplikasi **`My App`**.
		
		`{`
		`  "Version": "2012-10-17",`
		`  "Statement": [`
		`  {`
		`  "Action": [`
		`  "elasticbeanstalk:ValidateConfigurationSettings"`
		`  ],`
		`  "Effect": "Allow",`
		`  "Resource": [`
		`  "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv"`
		`  ],`
		`  "Condition": {`
		`  "StringEquals": {`
		`  "elasticbeanstalk:InApplication": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"]`
		`  }`
		`  }`
		`  }`
		`  ]`
		`}`

Kunci kondisi untuk tindakan Elastic Beanstalk

Kunci memungkinkan Anda untuk menentukan kondisi yang mengekspresikan dependensi, membatasi izin, atau menentukan kendala pada parameter input untuk tindakan. Elastic Beanstalk mendukung kunci berikut.

InApplication

Menentukan aplikasi yang berisi sumber daya tempat tindakan beroperasi.

Contoh berikut memungkinkan tindakan UpdateApplicationVersion untuk memperbarui properti dari versi aplikasi **My Version**. Kondisi InApplication menentukan **My App** sebagai kontainer untuk **My Version**.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "elasticbeanstalk:UpdateApplicationVersion"
      ],
      "Effect": "Allow",
      "Resource": [
        "arn:aws:elasticbeanstalk:us-east-2:123456789012:applicationversion/My App/
My Version"
      ],
      "Condition": {
        "StringEquals": {
          "elasticbeanstalk:InApplication": ["arn:aws:elasticbeanstalk:us-
east-2:123456789012:application/My App"]
        }
      }
    }
  ]
}
```

FromApplicationVersion

Menentukan versi aplikasi sebagai ketergantungan atau kendala pada parameter input.

Contoh berikut memungkinkan tindakan UpdateEnvironment untuk memperbarui lingkungan **myenv** dalam aplikasi **My App**. Syarat FromApplicationVersion mengontrol kondisi parameter VersionLabel untuk mengizinkan hanya versi aplikasi **My Version** untuk memperbarui lingkungan.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
```

```
      "Action": [
        "elasticbeanstalk:UpdateEnvironment"
      ],
      "Effect": "Allow",
      "Resource": [
        "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv"
      ],
      "Condition": {
        "StringEquals": {
          "elasticbeanstalk:InApplication": ["arn:aws:elasticbeanstalk:us-
east-2:123456789012:application/My App"],
          "elasticbeanstalk:FromApplicationVersion": ["arn:aws:elasticbeanstalk:us-
east-2:123456789012:applicationversion/My App/My Version"]
        }
      }
    }
  ]
}
```

`FromConfigurationTemplate`

Menentukan templat konfigurasi sebagai ketergantungan atau kendala pada parameter input.

Contoh berikut memungkinkan tindakan `UpdateEnvironment` untuk memperbarui lingkungan **`myenv`**dalam aplikasi **`My App`**. Syarat `FromConfigurationTemplate` mengontrol parameter `TemplateName` untuk mengizinkan hanya templat konfigurasi **`My Template`** untuk memperbarui lingkungan.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "elasticbeanstalk:UpdateEnvironment"
      ],
      "Effect": "Allow",
      "Resource": [
        "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv"
      ],
      "Condition": {
        "StringEquals": {
          "elasticbeanstalk:InApplication": ["arn:aws:elasticbeanstalk:us-
east-2:123456789012:application/My App"],
```

```
          "elasticbeanstalk:FromConfigurationTemplate":
 ["arn:aws:elasticbeanstalk:us-east-2:123456789012:configurationtemplate/My App/My
 Template"]
        }
      }
    }
  ]
}
```

`FromEnvironment`

Menentukan lingkungan sebagai ketergantungan atau kendala pada parameter input.

Contoh berikut memungkinkan `SwapEnvironmentCNAMEs` tindakan untuk menukar CNAMEs in **`My App`** untuk semua lingkungan yang namanya dimulai dengan **`mysrcenv`** dan **`mydestenv`** tetapi bukan lingkungan yang namanya dimulai dengan **`mysrcenvPROD*`** dan**`mydestenvPROD*`**.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "elasticbeanstalk:SwapEnvironmentCNAMEs"
      ],
      "Effect": "Allow",
      "Resource": [
        "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/
mysrcenv*",
        "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/
mydestenv*"
      ],
      "Condition": {
        "StringNotLike": {
          "elasticbeanstalk:FromEnvironment": [
            "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/
mysrcenvPROD*",
            "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/
mydestenvPROD*"
          ]
        }
      }
    }
  ]
```

```
}
```

`FromSolutionStack`

Menentukan solusi tumpukan sebagai ketergantungan atau kendala pada parameter input.

Kebijakan berikut memungkinkan tindakan `CreateConfigurationTemplate` untuk membuat cetakan konfigurasi yang namanya dimulai dengan **`My Template`** (`My Template*`) di aplikasi **`My App`**. Syarat `FromSolutionStack` mengontrol kondisi parameter `solutionstack` untuk mengizinkan hanya solusi tumpukan **`32bit Amazon Linux running Tomcat 7`** sebagai nilai input untuk parameter tersebut.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "elasticbeanstalk:CreateConfigurationTemplate"
      ],
      "Effect": "Allow",
      "Resource": [
        "arn:aws:elasticbeanstalk:us-east-2:123456789012:configurationtemplate/My
 App/My Template*"
      ],
      "Condition": {
        "StringEquals": {
          "elasticbeanstalk:InApplication": ["arn:aws:elasticbeanstalk:us-
east-2:123456789012:application/My App"],
          "elasticbeanstalk:FromSolutionStack": ["arn:aws:elasticbeanstalk:us-
east-2::solutionstack/32bit Amazon Linux running Tomcat 7"]
        }
      }
    }
  ]
}
```

`aws:ResourceTag/`*`key-name`*`, aws:RequestTag/`*`key-name`*`, aws:TagKeys`

Tentukan ketentuan berbasis tag. Untuk detailnya, lihat Menggunakan tag untuk mengontrol akses ke sumber Elastic Beanstalk.

Menggunakan tag untuk mengontrol akses ke sumber Elastic Beanstalk

Topik ini menjelaskan bagaimana kontrol akses berbasis tag dapat membantu Anda membuat dan mengelola kebijakan IAM.

Kita dapat menggunakan kondisi dalam pernyataan kebijakan pengguna IAM untuk mengonfigurasi izin untuk akses Elastic Beanstalk ke sumber daya. Untuk mempelajari lebih lanjut tentang kondisi pernyataan kebijakan, lihatSumber daya dan kondisi untuk tindakan Elastic Beanstalk. Menggunakan tanda dalam kondisi adalah salah satu cara untuk mengontrol akses ke sumber daya dan permintaan. Untuk informasi tentang pemberian tag sumber Elastic Beanstalk, lihat Pelabelan sumber daya aplikasi Elastic Beanstalk.

Saat merancang kebijakan IAM, Anda mungkin menetapkan izin terperinci dengan memberikan akses ke sumber daya tertentu. Saat jumlah sumber daya yang Anda kelola bertambah, tugas ini menjadi lebih sulit. Menandai sumber daya dan menggunakan tanda dalam kondisi pernyataan kebijakan dapat mempermudah tugas ini. Anda memberikan akses secara massal ke sumber daya dengan tag tertentu. Kemudian Anda menerapkan tag ini berulang kali ke sumber daya yang relevan, selama pembuatan atau yang lebih baru.

Tag dapat dilampirkan ke sumber daya atau diteruskan atas permintaan ke layanan yang mendukung penandaan. Di Elastic Beanstalk, sumber daya dapat memiliki tag, dan beberapa tindakan dapat mencakup tag. Saat membuat kebijakan IAM, Anda dapat menggunakan kunci kondisi tag untuk mengontrol kondisi berikut:

- Manakah pengguna yang dapat melakukan tindakan pada distribusi, berdasarkan tag yang telah dimiliki.
- Tag apa yang dapat diteruskan dalam permintaan tindakan.
- Apakah kunci tag tertentu dapat digunakan dalam permintaan.

Untuk sintaks dan semantik kunci syarat tanda yang lengkap, lihat Controlling Access Using Tags dalam Panduan Pengguna IAM.

Contoh kondisi tag dalam kebijakan

Contoh berikut ini mendemosntrasikan cara menentukan syarat tag dalam kebijakan bagi pengguna Elastic Beanstalk.

Example 1: Batasi tindakan berdasarkan tag dalam permintaan

Kebijakan pengguna yang dikelola AdministratorAccessElastic Beanstalk - AWSElastic Beanstalk memberikan izin tak terbatas kepada pengguna untuk melakukan tindakan Elastic Beanstalk pada sumber daya yang dikelola Elastic Beanstalk.

Kebijakan berikut membatasi kekuatan ini dan menolak izin pengguna yang tidak sah untuk membuat lingkungan produksi Elastic Beanstalk. Untuk melakukan itu, ia menolak tindakan `CreateEnvironment` jika permintaan menentukan tag bernama `stage` dengan salah satu nilai `gamma` atau `prod`. Selain itu, kebijakan ini mencegah pengguna yang tidak berwenang merusak tahap lingkungan produksi dengan tidak mengizinkan tindakan modifikasi tag untuk memasukkan nilai tag yang sama ini atau sepenuhnya menghapus tag `stage`. Administrator pelanggan harus melampirkan kebijakan IAM ini kepada pengguna IAM yang tidak sah, selain kebijakan pengguna yang dikelola.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": [
        "elasticbeanstalk:CreateEnvironment",
        "elasticbeanstalk:AddTags"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:RequestTag/stage": ["gamma", "prod"]
        }
      }
    },
    {
      "Effect": "Deny",
      "Action": [
        "elasticbeanstalk:RemoveTags"
      ],
      "Resource": "*",
      "Condition": {
        "ForAnyValue:StringEquals": {
          "aws:TagKeys": ["stage"]
        }
      }
```

```
    }
  ]
}
```

Example 2: Batasi tindakan berdasarkan tanda sumber daya

Kebijakan pengguna yang dikelola AdministratorAccessElastic Beanstalk - AWSElastic Beanstalk memberikan izin tak terbatas kepada pengguna untuk melakukan tindakan Elastic Beanstalk pada sumber daya yang dikelola Elastic Beanstalk.

Kebijakan berikut membatasi kekuatan ini dan menolak izin pengguna yang tidak sah untuk melakukan tindakan pada lingkungan produksi Elastic Beanstalk. Untuk melakukan itu, ia menyangkal tindakan tertentu jika lingkungan memiliki tag bernama `stage` dengan salah satu nilai `gamma` atau `prod`. Administrator pelanggan harus melampirkan kebijakan IAM ini kepada pengguna IAM yang tidak sah, selain kebijakan pengguna yang dikelola.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": [
        "elasticbeanstalk:AddTags",
        "elasticbeanstalk:RemoveTags",
        "elasticbeanstalk:DescribeEnvironments",
        "elasticbeanstalk:TerminateEnvironment",
        "elasticbeanstalk:UpdateEnvironment",
        "elasticbeanstalk:ListTagsForResource"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/stage": ["gamma", "prod"]
        }
      }
    }
  ]
}
```

Example 3: Izinkan tindakan berdasarkan tag dalam permintaan

Kebijakan berikut memberikan izin pengguna untuk membuat aplikasi pengembangan Elastic Beanstalk.

Untuk melakukan itu, memungkinkan tindakan `CreateApplication` dan `AddTags` jika permintaan menentukan tag bernama `stage` dengan nilai `development`. Syarat `aws:TagKeys` memastikan bahwa pengguna tidak dapat menambahkan kunci tag lainnya. Secara khusus, memastikan sensitivitas kasus kunci tag `stage`. Perhatikan bahwa kebijakan ini berguna bagi pengguna IAM yang tidak memiliki kebijakan pengguna terkelola Elastic Beanstalk AdministratorAccess - AWSElasticBeanstalk. Kebijakan yang terkelola memberikan pengguna izin tak terbatas untuk melakukan tindakan Elastic Beanstalk pada sumber daya yang dikelola Elastic Beanstalk.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "elasticbeanstalk:CreateApplication",
        "elasticbeanstalk:AddTags"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:RequestTag/stage": "development"
        },
        "ForAllValues:StringEquals": {
          "aws:TagKeys": ["stage"]
        }
      }
    }
  ]
}
```

Example 4: Izinkan tindakan berdasarkan tag sumber daya

Kebijakan berikut memberikan pengguna izin untuk melakukan tindakan, dan mendapatkan informasi tentang, aplikasi pengembangan Elastic Beanstalk.

Untuk melakukan itu, memungkinkan tindakan tertentu jika aplikasi memiliki tag bernama `stage` dengan nilai `development`. Syarat `aws:TagKeys` memastikan bahwa pengguna tidak dapat

menambahkan kunci tag lainnya. Secara khusus, memastikan sensitivitas kasus kunci tag `stage`. Perhatikan bahwa kebijakan ini berguna bagi pengguna IAM yang tidak memiliki kebijakan pengguna terkelola Elastic Beanstalk AdministratorAccess - AWSElasticBeanstalk. Kebijakan yang terkelola memberikan pengguna izin tak terbatas untuk melakukan tindakan Elastic Beanstalk pada sumber daya yang dikelola Elastic Beanstalk.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "elasticbeanstalk:UpdateApplication",
        "elasticbeanstalk:DeleteApplication",
        "elasticbeanstalk:DescribeApplications"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/stage": "development"
        },
        "ForAllValues:StringEquals": {
          "aws:TagKeys": ["stage"]
        }
      }
    }
  ]
}
```

Contoh kebijakan berdasarkan kebijakan terkelola

Bagian ini menunjukkan cara mengontrol akses pengguna ke AWS Elastic Beanstalk dan menyertakan contoh kebijakan yang menyediakan akses yang diperlukan untuk skenario umum. Kebijakan ini berasal dari kebijakan dikelola Elastic Beanstalk. Untuk informasi tentang melampirkan kebijakan terkelola ke pengguna dan grup, lihat Mengelola kebijakan pengguna Elastic Beanstalk.

Dalam skenario ini, Example Corp. adalah perusahaan perangkat lunak dengan tiga tim yang bertanggung jawab untuk situs web perusahaan: administrator yang mengelola infrastruktur, developer yang membangun perangkat lunak untuk situs web, dan tim QA yang menguji situs web. Untuk membantu mengelola izin ke sumber daya Elastic Beanstalk mereka, Contoh Corp. menciptakan tiga kelompok yang menjadi anggota masing-masing tim: Admin, Developer, dan

Penguji. Contoh Corp. ingin grup Admin memiliki akses penuh ke semua aplikasi, lingkungan, dan sumber daya yang mendasarinya sehingga mereka dapat membuat, memecahkan masalah, dan menghapus semua aset Elastic Beanstalk. Developer memerlukan izin untuk melihat semua aset Elastic Beanstalk dan untuk membuat dan men-deploy versi aplikasi. Developer seharusnya tidak dapat membuat aplikasi baru atau lingkungan atau mengakhiri lingkungan yang sedang berjalan. Penguji perlu melihat semua sumber daya Elastic Beanstalk untuk memantau dan menguji aplikasi. Penguji seharusnya tidak dapat membuat perubahan pada sumber daya Elastic Beanstalk.

Contoh kebijakan berikut memberikan izin yang diperlukan untuk setiap grup.

Contoh 1: Grup admin — Semua Elastic Beanstalk dan layanan terkait APIs

Kebijakan berikut memberi pengguna izin untuk semua tindakan yang diperlukan untuk menggunakan Elastic Beanstalk. Kebijakan ini juga memungkinkan Elastic Beanstalk untuk menyediakan dan mengelola sumber daya atas nama Anda dalam layanan berikut. Elastic Beanstalk bergantung pada layanan tambahan ini untuk menyediakan sumber daya yang mendasari saat menciptakan lingkungan.

- Amazon Elastic Compute Cloud
- Elastic Load Balancing
- Auto Scaling
- Amazon CloudWatch
- Amazon Simple Storage Service
- Amazon Simple Notification Service
- Amazon Relational Database Service
- AWS CloudFormation

Perhatikan bahwa kebijakan ini adalah contoh. Ini memberikan satu set luas izin untuk layanan AWS yang digunakan Elastic Beanstalk untuk mengelola aplikasi dan lingkungan. Misalnya, `ec2:*` memungkinkan pengguna AWS Identity and Access Management (IAM) untuk melakukan tindakan apa pun pada EC2 sumber daya Amazon apa pun di AWS akun. Izin ini tidak terbatas pada sumber daya yang Anda gunakan dengan Elastic Beanstalk. Sebagai praktik terbaik, Anda harus memberi izin kepada individu saja yang mereka butuhkan untuk menjalankan tugasnya.

```
{
  "Version" : "2012-10-17",
```

```
  "Statement" : [
    {
      "Effect" : "Allow",
      "Action" : [
        "elasticbeanstalk:*",
        "ec2:*",
        "elasticloadbalancing:*",
        "autoscaling:*",
        "cloudwatch:*",
        "s3:*",
        "sns:*",
        "rds:*",
        "cloudformation:*"
      ],
      "Resource" : "*"
    }
  ]
}
```

Contoh 2: Kelompok developer – Semua kecuali operasi yang sangat istimewa

Kebijakan berikut menolak izin untuk membuat aplikasi dan lingkungan, dan memungkinkan semua tindakan Elastic Beanstalk lainnya.

Perhatikan bahwa kebijakan ini adalah contoh. Ini memberikan satu set yang luas izin untuk produk AWS yang digunakan Elastic Beanstalk untuk mengelola aplikasi dan lingkungan. Misalnya, `ec2:*` memungkinkan pengguna IAM untuk melakukan tindakan apa pun pada EC2 sumber daya Amazon apa pun di AWS akun. Izin ini tidak terbatas pada sumber daya yang Anda gunakan dengan Elastic Beanstalk. Sebagai praktik terbaik, Anda harus memberi izin kepada individu saja yang mereka butuhkan untuk menjalankan tugasnya.

```
{
  "Version" : "2012-10-17",
  "Statement" : [
    {
      "Action" : [
        "elasticbeanstalk:CreateApplication",
        "elasticbeanstalk:CreateEnvironment",
        "elasticbeanstalk:DeleteApplication",
        "elasticbeanstalk:RebuildEnvironment",
        "elasticbeanstalk:SwapEnvironmentCNAMEs",
        "elasticbeanstalk:TerminateEnvironment"],
```

```
      "Effect" : "Deny",
      "Resource" : "*"
    },
    {
      "Action" : [
        "elasticbeanstalk:*",
        "ec2:*",
        "elasticloadbalancing:*",
        "autoscaling:*",
        "cloudwatch:*",
        "s3:*",
        "sns:*",
        "rds:*",
        "cloudformation:*"],
      "Effect" : "Allow",
      "Resource" : "*"
    }
  ]
}
```

Contoh 3: Penguji – Lihat saja

Kebijakan berikut memungkinkan akses baca-saja ke seluruh aplikasi, versi aplikasi, peristiwa, dan lingkungan. Ini tidak memungkinkan melakukan tindakan apa pun.

```
{
  "Version" : "2012-10-17",
  "Statement" : [
    {
      "Effect" : "Allow",
      "Action" : [
        "elasticbeanstalk:Check*",
        "elasticbeanstalk:Describe*",
        "elasticbeanstalk:List*",
        "elasticbeanstalk:RequestEnvironmentInfo",
        "elasticbeanstalk:RetrieveEnvironmentInfo",
        "ec2:Describe*",
        "elasticloadbalancing:Describe*",
        "autoscaling:Describe*",
        "cloudwatch:Describe*",
        "cloudwatch:List*",
        "cloudwatch:Get*",
```

```
        "s3:Get*",
        "s3:List*",
        "sns:Get*",
        "sns:List*",
        "rds:Describe*",
        "cloudformation:Describe*",
        "cloudformation:Get*",
        "cloudformation:List*",
        "cloudformation:Validate*",
        "cloudformation:Estimate*"
      ],
      "Resource" : "*"
    }
  ]
}
```

Contoh kebijakan berdasarkan izin sumber daya

Bagian ini berjalan melalui kasus penggunaan untuk mengendalikan izin pengguna untuk tindakan Elastic Beanstalk yang mengakses sumber Elastic Beanstalk tertentu. Kami akan berjalan melalui contoh kebijakan yang mendukung kasus penggunaan. Untuk kebijakan informasi lebih lanjut tentang sumber Elastic Beanstalk, lihat Membuat kebijakan pengguna kustom. Untuk informasi tentang melampirkan kebijakan untuk pengguna dan grup, buka Mengelola Kebijakan IAM di Menggunakan AWS Identity and Access Management.

Dalam kasus penggunaan kami, Example Corp adalah perusahaan konsultan kecil yang mengembangkan aplikasi untuk dua pelanggan yang berbeda. John adalah manajer pengembangan yang mengawasi pengembangan dua aplikasi Elastic Beanstalk, app1 dan app2. John melakukan pengembangan dan beberapa pengujian pada dua aplikasi, dan hanya dia yang dapat memperbarui lingkungan produksi untuk dua aplikasi. Ini adalah izin yang ia butuhkan untuk app1 dan app2:

- Melihat aplikasi, versi aplikasi, lingkungan, dan templat konfigurasi
- Buat versi aplikasi dan men-deploy ke lingkungan pementasan
- Memperbarui lingkungan produksi
- Membuat dan mengakhiri lingkungan

Jill adalah tester yang membutuhkan akses untuk melihat sumber daya berikut untuk memantau dan menguji dua aplikasi: aplikasi, versi aplikasi, lingkungan, dan templat konfigurasi. Namun, dia seharusnya tidak bisa membuat perubahan pada sumber Elastic Beanstalk.

Jack adalah developer untuk app1 yang membutuhkan akses untuk melihat semua sumber daya untuk app1 dan juga perlu membuat versi aplikasi untuk app1 dan men-deploy mereka ke lingkungan pementasan.

Judy adalah administrator AWS akun untuk Example Corp. Dia telah membuat pengguna IAM untuk John, Jill, dan Jack dan melampirkan kebijakan berikut kepada pengguna tersebut untuk memberikan izin yang sesuai ke aplikasi app1 dan app2.

Contoh 1: John – Manajer pengembangan untuk app1, app2

Kami telah menguraikan kebijakan John menjadi tiga kebijakan terpisah sehingga lebih mudah dibaca dan dikelola. Bersama-sama, mereka memberikan John izin yang dia butuhkan untuk melakukan pengembangan, pengujian, dan tindakan deployment pada dua aplikasi.

Kebijakan pertama menentukan tindakan untuk Auto Scaling, Amazon S3, Amazon,, Amazon EC2 SNS, Elastic Load Balancing, CloudWatch Amazon RDS, dan. AWS CloudFormation Elastic Beanstalk bergantung pada layanan tambahan ini untuk menyediakan sumber daya yang mendasari saat menciptakan lingkungan.

Perhatikan bahwa kebijakan ini adalah contoh. Ini memberikan serangkaian izin yang luas untuk AWS produk yang digunakan Elastic Beanstalk untuk mengelola aplikasi dan lingkungan. Misalnya, `ec2:*` memungkinkan pengguna IAM untuk melakukan tindakan apa pun pada EC2 sumber daya Amazon apa pun di AWS akun. Izin ini tidak terbatas pada sumber daya yang Anda gunakan dengan Elastic Beanstalk. Sebagai praktik terbaik, Anda harus memberi izin kepada individu saja yang mereka butuhkan untuk menjalankan tugasnya.

```
{
   "Version": "2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "ec2:*",
            "ecs:*",
            "ecr:*",
            "elasticloadbalancing:*",
            "autoscaling:*",
            "cloudwatch:*",
            "s3:*",
            "sns:*",
            "cloudformation:*",
```

```
            "dynamodb:*",
            "rds:*",
            "sqs:*",
            "logs:*",
            "iam:GetPolicyVersion",
            "iam:GetRole",
            "iam:PassRole",
            "iam:ListRolePolicies",
            "iam:ListAttachedRolePolicies",
            "iam:ListInstanceProfiles",
            "iam:ListRoles",
            "iam:ListServerCertificates",
            "acm:DescribeCertificate",
            "acm:ListCertificates",
            "codebuild:CreateProject",
            "codebuild:DeleteProject",
            "codebuild:BatchGetBuilds",
            "codebuild:StartBuild"
         ],
         "Resource":"*"
      }
   ]
}
```

Kebijakan kedua menentukan tindakan Elastic Beanstalk yang John diperbolehkan untuk melakukan pada sumber daya app1 dan app2. Pernyataan `AllCallsInApplications` memungkinkan semua tindakan Elastic Beanstalk (`"elasticbeanstalk:*"`) dilakukan pada semua sumber daya dalam app1 dan app2 (misalnya, `elasticbeanstalk:CreateEnvironment`). Pernyataan `AllCallsOnApplications` memungkinkan semua tindakan Elastic Beanstalk (`"elasticbeanstalk:*"`) pada sumber daya aplikasi app1 dan app2 (misalnya, `elasticbeanstalk:DescribeApplications`, `elasticbeanstalk:UpdateApplication`, dll.). Pernyataan `AllCallsOnSolutionStacks` memungkinkan semua tindakan Elastic Beanstalk (`"elasticbeanstalk:*"`) untuk sumber daya tumpukan solusi (misalnya, `elasticbeanstalk:ListAvailableSolutionStacks`).

```
{
   "Version": "2012-10-17",
   "Statement":[
      {
         "Sid":"AllCallsInApplications",
         "Action":[
            "elasticbeanstalk:*"
```

```
         ],
         "Effect":"Allow",
         "Resource":[
            "*"
         ],
         "Condition":{
            "StringEquals":{
               "elasticbeanstalk:InApplication":[
                  "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/app1",
                  "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/app2"
               ]
            }
         }
      },
      {
         "Sid":"AllCallsOnApplications",
         "Action":[
            "elasticbeanstalk:*"
         ],
         "Effect":"Allow",
         "Resource":[
            "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/app1",
            "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/app2"
         ]
      },
      {
         "Sid":"AllCallsOnSolutionStacks",
         "Action":[
            "elasticbeanstalk:*"
         ],
         "Effect":"Allow",
         "Resource":[
            "arn:aws:elasticbeanstalk:us-east-2::solutionstack/*"
         ]
      }
   ]
}
```

Kebijakan ketiga menentukan tindakan Elastic Beanstalk bahwa kebijakan kedua membutuhkan izin untuk menyelesaikan tindakan Elastic Beanstalk. Pernyataan `AllNonResourceCalls` memungkinkan tindakan `elasticbeanstalk:CheckDNSAvailability`, yang diperlukan untuk memanggil `elasticbeanstalk:CreateEnvironment` dan tindakan lainnya. Hal ini juga memungkinkan tindakan `elasticbeanstalk:CreateStorageLocation`, yang diperlukan untuk

`elasticbeanstalk:CreateApplication`, `elasticbeanstalk:CreateEnvironment`, dan tindakan lainnya.

```
{
   "Version": "2012-10-17",
   "Statement":[
      {
         "Sid":"AllNonResourceCalls",
         "Action":[
            "elasticbeanstalk:CheckDNSAvailability",
            "elasticbeanstalk:CreateStorageLocation"
         ],
         "Effect":"Allow",
         "Resource":[
            "*"
         ]
      }
   ]
}
```

Contoh 2: Jill – Tester untuk app1, app2

Kami telah menguraikan kebijakan Jill menjadi tiga kebijakan terpisah sehingga lebih mudah untuk membaca dan mengelola. Bersama-sama, mereka memberikan Jill izin yang dia butuhkan untuk melakukan pengujian dan pemantauan tindakan pada dua aplikasi.

Kebijakan pertama menentukan`Describe*`,`List*`, dan `Get*` tindakan pada Auto Scaling, Amazon S3, Amazon,, CloudWatch Amazon SNS EC2, Elastic Load Balancing, Amazon RDS AWS CloudFormation , dan (untuk tipe container non-legacy) sehingga tindakan Elastic Beanstalk dapat mengambil informasi yang relevan tentang sumber daya dasar aplikasi app1 dan app2.

```
{
   "Version": "2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "ec2:Describe*",
            "elasticloadbalancing:Describe*",
            "autoscaling:Describe*",
            "cloudwatch:Describe*",
            "cloudwatch:List*",
```

```
            "cloudwatch:Get*",
            "s3:Get*",
            "s3:List*",
            "sns:Get*",
            "sns:List*",
            "rds:Describe*",
            "cloudformation:Describe*",
         "cloudformation:Get*",
         "cloudformation:List*",
         "cloudformation:Validate*",
         "cloudformation:Estimate*"
         ],
         "Resource":"*"
      }
   ]
}
```

Kebijakan kedua menentukan tindakan Elastic Beanstalk yang diizinkan untuk dilakukan Jill pada sumber daya app1 dan app2 sumber daya. Pernyataan `AllReadCallsInApplications` memungkinkan dia untuk memanggil tindakan `Describe*` dan tindakan info lingkungan. Pernyataan `AllReadCallsOnApplications` memungkinkan dia untuk memanggil tindakan `DescribeApplications` dan `DescribeEvents` pada sumber aplikasi app1 dan app2. Pernyataan `AllReadCallsOnSolutionStacks` memungkinkan melihat tindakan yang melibatkan solusi tumpukan sumber daya (`ListAvailableSolutionStacks`,`DescribeConfigurationOptions`, dan `ValidateConfigurationSettings`).

```
{
   "Version": "2012-10-17",
   "Statement":[
      {
         "Sid":"AllReadCallsInApplications",
         "Action":[
            "elasticbeanstalk:Describe*",
            "elasticbeanstalk:RequestEnvironmentInfo",
            "elasticbeanstalk:RetrieveEnvironmentInfo"
         ],
         "Effect":"Allow",
         "Resource":[
            "*"
         ],
         "Condition":{
            "StringEquals":{
```

```
               "elasticbeanstalk:InApplication":[
                  "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/app1",
                  "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/app2"
               ]
            }
         }
      },
      {
         "Sid":"AllReadCallsOnApplications",
         "Action":[
            "elasticbeanstalk:DescribeApplications",
            "elasticbeanstalk:DescribeEvents"
         ],
         "Effect":"Allow",
         "Resource":[
            "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/app1",
            "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/app2"
         ]
      },
      {
         "Sid":"AllReadCallsOnSolutionStacks",
         "Action":[
            "elasticbeanstalk:ListAvailableSolutionStacks",
            "elasticbeanstalk:DescribeConfigurationOptions",
            "elasticbeanstalk:ValidateConfigurationSettings"
         ],
         "Effect":"Allow",
         "Resource":[
            "arn:aws:elasticbeanstalk:us-east-2::solutionstack/*"
         ]
      }
   ]
}
```

Kebijakan ketiga menentukan tindakan Elastic Beanstalk bahwa kebijakan kedua membutuhkan izin untuk menyelesaikan tindakan Elastic Beanstalk. Pernyataan `AllNonResourceCalls` memungkinkan tindakan `elasticbeanstalk:CheckDNSAvailability`, yang diperlukan untuk beberapa tindakan melihat.

```
{
   "Version": "2012-10-17",
   "Statement":[
      {
```

```
         "Sid":"AllNonResourceCalls",
         "Action":[
            "elasticbeanstalk:CheckDNSAvailability"
         ],
         "Effect":"Allow",
         "Resource":[
            "*"
         ]
      }
   ]
}
```

Contoh 3: Jack – Developer untuk app1

Kami telah memecah kebijakan Jack menjadi tiga kebijakan terpisah sehingga lebih mudah untuk membaca dan mengelola. Bersama-sama, mereka memberikan Jack izin yang dia butuhkan untuk melakukan pengujian, pemantauan, dan tindakan deployment pada sumber daya app1.

Kebijakan pertama menentukan tindakan pada Auto Scaling, Amazon S3, Amazon,, CloudWatch Amazon EC2 SNS, Elastic Load Balancing, Amazon RDS AWS CloudFormation , dan (untuk tipe container non-legacy) sehingga tindakan Elastic Beanstalk dapat melihat dan bekerja dengan sumber daya dasar app1. Untuk daftar jenis kontainer non-warisan yang didukung, lihat the section called “Mengapa beberapa versi platform ditandai sebagai legasi?”

Perhatikan bahwa kebijakan ini adalah contoh. Ini memberikan serangkaian izin yang luas untuk AWS produk yang digunakan Elastic Beanstalk untuk mengelola aplikasi dan lingkungan. Misalnya, `ec2:*` memungkinkan pengguna IAM untuk melakukan tindakan apa pun pada EC2 sumber daya Amazon apa pun di AWS akun. Izin ini tidak terbatas pada sumber daya yang Anda gunakan dengan Elastic Beanstalk. Sebagai praktik terbaik, Anda harus memberi izin kepada individu saja yang mereka butuhkan untuk menjalankan tugasnya.

```
{
   "Version": "2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "ec2:*",
            "elasticloadbalancing:*",
            "autoscaling:*",
            "cloudwatch:*",
```

```
            "s3:*",
            "sns:*",
            "rds:*",
            "cloudformation:*"
         ],
         "Resource":"*"
      }
   ]
}
```

Kebijakan kedua menentukan tindakan Elastic Beanstalk yang Jack diperbolehkan untuk melakukan pada sumber daya app1.

```
{
   "Version": "2012-10-17",
   "Statement":[
      {
         "Sid":"AllReadCallsAndAllVersionCallsInApplications",
         "Action":[
            "elasticbeanstalk:Describe*",
            "elasticbeanstalk:RequestEnvironmentInfo",
            "elasticbeanstalk:RetrieveEnvironmentInfo",
            "elasticbeanstalk:CreateApplicationVersion",
            "elasticbeanstalk:DeleteApplicationVersion",
            "elasticbeanstalk:UpdateApplicationVersion"
         ],
         "Effect":"Allow",
         "Resource":[
            "*"
         ],
         "Condition":{
            "StringEquals":{
               "elasticbeanstalk:InApplication":[
                  "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/app1"
               ]
            }
         }
      },
      {
         "Sid":"AllReadCallsOnApplications",
         "Action":[
            "elasticbeanstalk:DescribeApplications",
            "elasticbeanstalk:DescribeEvents"
```

```
         ],
         "Effect":"Allow",
         "Resource":[
            "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/app1"
         ]
      },
      {
         "Sid":"UpdateEnvironmentInApplications",
         "Action":[
            "elasticbeanstalk:UpdateEnvironment"
         ],
         "Effect":"Allow",
         "Resource":[
            "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/app1/app1-
staging*"
         ],
         "Condition":{
            "StringEquals":{
               "elasticbeanstalk:InApplication":[
                  "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/app1"
               ]
            },
            "StringLike":{
               "elasticbeanstalk:FromApplicationVersion":[
                  "arn:aws:elasticbeanstalk:us-east-2:123456789012:applicationversion/
app1/*"
               ]
            }
         }
      },
      {
         "Sid":"AllReadCallsOnSolutionStacks",
         "Action":[
            "elasticbeanstalk:ListAvailableSolutionStacks",
            "elasticbeanstalk:DescribeConfigurationOptions",
            "elasticbeanstalk:ValidateConfigurationSettings"
         ],
         "Effect":"Allow",
         "Resource":[
            "arn:aws:elasticbeanstalk:us-east-2::solutionstack/*"
         ]
      }
   ]
```

```
}
```

Kebijakan ketiga menentukan tindakan Elastic Beanstalk bahwa kebijakan kedua membutuhkan izin untuk menyelesaikan tindakan Elastic Beanstalk. Pernyataan `AllNonResourceCalls` memungkinkan tindakan `elasticbeanstalk:CheckDNSAvailability`, yang diperlukan untuk memanggil `elasticbeanstalk:CreateEnvironment` dan tindakan lainnya. Hal ini juga memungkinkan tindakan `elasticbeanstalk:CreateStorageLocation`, yang diperlukan untuk `elasticbeanstalk:CreateEnvironment`, dan tindakan lainnya.

```
{
   "Version": "2012-10-17",
   "Statement":[
      {
         "Sid":"AllNonResourceCalls",
         "Action":[
            "elasticbeanstalk:CheckDNSAvailability",
            "elasticbeanstalk:CreateStorageLocation"
         ],
         "Effect":"Allow",
         "Resource":[
            "*"
         ]
      }
   ]
}
```

Mencegah akses bucket Amazon S3 lintas lingkungan

Topik ini menjelaskan bagaimana kebijakan terkelola memungkinkan akses bucket S3 lintas lingkungan dan cara membuat kebijakan khusus untuk mengelola jenis akses ini.

Elastic Beanstalk menyediakan kebijakan terkelola untuk AWS menangani sumber daya yang dibutuhkan oleh lingkungan Elastic Beanstalk di akun Anda. AWS Izin yang diberikan secara default ke satu aplikasi di AWS akun Anda memiliki akses ke sumber daya S3 milik aplikasi lain di akun yang sama AWS .

Jika AWS akun Anda menjalankan beberapa aplikasi Beanstalk, Anda dapat mengurangi keamanan kebijakan Anda dengan membuat kebijakan kustom Anda sendiri untuk dilampirkan ke peran layanan atau profil instans Anda sendiri untuk setiap lingkungan. Anda kemudian dapat membatasi izin S3 dalam kebijakan kustom Anda ke lingkungan tertentu.

> **Note**
>
> Ketahuilah bahwa Anda bertanggung jawab untuk menjaga kebijakan kustom Anda. Jika kebijakan terkelola Elastic Beanstalk yang menjadi dasar kebijakan kustom Anda berubah, Anda harus mengubah kebijakan kustom Anda dengan perubahan masing-masing pada kebijakan dasar. Untuk riwayat perubahan kebijakan terkelola Elastic Beanstalk, lihat. Elastic Beanstalk memperbarui kebijakan terkelola AWS

Contoh izin cakupan bawah

Contoh berikut didasarkan pada kebijakan yang AWSElasticBeanstalkWebTierdikelola.

Kebijakan default menyertakan baris berikut untuk izin ke bucket S3. Kebijakan default ini tidak membatasi tindakan bucket S3 ke lingkungan atau aplikasi tertentu.

```
{
   "Sid" : "BucketAccess",
   "Action" : [
      "s3:Get*",
      "s3:List*",
      "s3:PutObject"
     ],
   "Effect" : "Allow",
   "Resource" : [
      "arn:aws:s3:::elasticbeanstalk-*",
      "arn:aws:s3:::elasticbeanstalk-*/*"
     ]
}
```

Anda dapat mengurangi akses dengan mengkualifikasi sumber daya tertentu ke peran layanan yang ditentukan sebagai`Principal`. Contoh berikut memberikan `aws-elasticbeanstalk-ec2-role-my-example-env` izin peran layanan kustom ke bucket S3 di lingkungan dengan id. `my-example-env-ID`

Example Berikan izin hanya untuk bucket S3 lingkungan tertentu

```
{
   "Sid": "BucketAccess",
   "Action": [
```

```
      "s3:Get*",
      "s3:List*",
      "s3:PutObject"
    ],
   "Effect": "Allow",
   "Principal": {
      "AWS": "arn:aws:iam::...:role/aws-elasticbeanstalk-ec2-role-my-example-env"
     },
   "Resource": [
      "arn:aws:s3:::elasticbeanstalk-my-region-account-id-12345",
      "arn:aws:s3:::elasticbeanstalk-my-region-account-id-12345/resources/environments/
my-example-env-ID/*"
    ]
}
```

> **Note**
>
> ARN Sumber Daya harus menyertakan ID lingkungan Elastic Beanstalk, (bukan nama lingkungan). Anda dapat memperoleh id lingkungan dari konsol Elastic Beanstalk di halaman ikhtisar Lingkungan. Anda juga dapat menggunakan AWS CLI perintah deskripsi-lingkungan untuk mendapatkan informasi ini.

Untuk informasi selengkapnya guna membantu Anda memperbarui izin bucket S3 untuk lingkungan Elastic Beanstalk, lihat sumber daya berikut:

- Menggunakan Elastic Beanstalk dengan Amazon S3 dalam panduan ini
- Jenis sumber daya yang ditentukan oleh Amazon S3 dalam panduan Referensi Otorisasi Layanan
- Format ARN dalam Panduan Pengguna IAM

Menggunakan Elastic Beanstalk dengan Amazon RDS

Bagian ini menjelaskan bagaimana Anda dapat menggunakan Elastic Beanstalk dengan Amazon Relational Database Service (Amazon RDS) untuk menyiapkan, mengoperasikan, dan menskalakan database relasional. Kami menjelaskan beberapa konsep tentang konfigurasi dan memberikan rekomendasi. Kemudian kami akan memandu Anda melalui proses untuk membuat dan terhubung ke Amazon RDS.

Ada dua opsi untuk memulai:

- Buat database baru di Amazon RDS.
- Mulailah dengan database yang sebelumnya dibuat oleh Elastic Beanstalk dan kemudian dipisahkan dari lingkungan Beanstalk. Untuk informasi selengkapnya, lihat the section called “Basis data”.

Pilih pendekatan

Anda dapat menggunakan salah satu pendekatan untuk menjalankan instance database di Amazon RDS dan mengonfigurasi aplikasi Anda untuk menyambungkannya saat diluncurkan. Anda dapat menghubungkan beberapa lingkungan ke database.

> **Note**
>
> Jika Anda belum pernah menggunakan instance database dengan aplikasi Anda sebelumnya, sebaiknya Anda menambahkan database ke lingkungan pengujian dengan konsol Elastic Beanstalk terlebih dahulu. Dengan melakukan ini, Anda dapat memverifikasi bahwa aplikasi Anda dapat membaca properti lingkungan, membuat string koneksi, dan terhubung ke instance database, tanpa pekerjaan konfigurasi tambahan yang diperlukan untuk database mandiri. Untuk informasi selengkapnya, lihat Menambahkan basis data ke lingkungan Elastic Beanstalk Anda.

Konfigurasikan grup keamanan

Untuk mengizinkan EC2 instans Amazon di lingkungan Anda terhubung ke database luar, konfigurasikan grup keamanan tambahan untuk grup Auto Scaling yang terkait dengan lingkungan Anda. Anda dapat melampirkan grup keamanan yang sama yang dilampirkan ke instance database Anda. Atau, Anda dapat menggunakan grup keamanan terpisah. Jika Anda melampirkan grup keamanan yang berbeda, Anda harus mengonfigurasi grup keamanan yang dilampirkan ke database Anda untuk mengizinkan akses masuk dari grup keamanan ini.

> **Note**
>
> Anda dapat menghubungkan lingkungan Anda ke database dengan menambahkan aturan ke grup keamanan yang dilampirkan ke database Anda. Aturan ini harus mengizinkan akses masuk dari grup keamanan yang dibuat otomatis yang dilampirkan Elastic Beanstalk ke grup Auto Scaling untuk lingkungan Anda. Namun, ketahuilah bahwa, dengan membuat aturan ini, Anda juga membuat ketergantungan antara dua grup keamanan. Selanjutnya, ketika Anda

mencoba untuk mengakhiri lingkungan, Elastic Beanstalk tidak akan dapat menghapus grup keamanan lingkungan, karena grup keamanan database bergantung padanya.

Konfigurasikan koneksi database

Setelah meluncurkan instance database dan mengonfigurasi grup keamanan, Anda dapat meneruskan informasi koneksi, seperti titik akhir dan kata sandi, ke aplikasi Anda dengan menggunakan properti lingkungan. Ini adalah mekanisme yang sama yang digunakan Elastic Beanstalk di latar belakang saat Anda menjalankan instance database di lingkungan Anda.

Untuk lapisan keamanan tambahan, Anda dapat menyimpan informasi koneksi di Amazon S3, dan mengonfigurasi Elastic Beanstalk untuk mengambilnya selama penerapan. Dengan file konfigurasi (`.ebextensions`), Anda dapat mengonfigurasi instans di lingkungan Anda untuk secara aman mengambil file dari Amazon S3 ketika Anda men-deploy aplikasi Anda.

Topik

- Meluncurkan dan menghubungkan ke instans Amazon RDS eksternal dalam VPC default
- Menyimpan kredensi Amazon RDS di AWS Secrets Manager
- Membersihkan instans Amazon RDS eksternal

Meluncurkan dan menghubungkan ke instans Amazon RDS eksternal dalam VPC default

Prosedur berikut menjelaskan proses untuk menghubungkan ke instans Amazon RDS eksternal ke VPC default. Prosesnya sama jika Anda menggunakan VPC khusus. Satu-satunya persyaratan tambahan adalah bahwa lingkungan Anda dan DB instans berada di subnet yang sama, atau subnet yang diizinkan untuk berkomunikasi satu sama lain. Untuk informasi selengkapnya tentang mengonfigurasi VPC khusus untuk digunakan dengan Elastic Beanstalk, lihat. Menggunakan Elastic Beanstalk dengan Amazon VPC

> **Note**
>
> - Alternatif untuk meluncurkan instans DB baru, adalah memulai dengan database yang sebelumnya dibuat oleh Elastic Beanstalk dan kemudian dipisahkan dari lingkungan Beanstalk. Untuk informasi selengkapnya, lihat the section called “Basis data”. Dengan opsi

ini, Anda tidak perlu menyelesaikan prosedur untuk meluncurkan database baru. Namun, Anda perlu menyelesaikan prosedur selanjutnya yang dijelaskan dalam topik ini.

- Jika Anda memulai dengan database yang dibuat oleh Elastic Beanstalk dan kemudian dipisahkan dari lingkungan Beanstalk, Anda dapat melewati kelompok langkah pertama dan melanjutkan dengan langkah-langkah yang dikelompokkan di bawah Untuk memodifikasi aturan masuk pada grup keamanan instans RDS Anda.
- Jika Anda berencana untuk menggunakan database yang Anda pisahkan untuk lingkungan produksi, verifikasi jenis penyimpanan yang digunakan database sesuai dengan beban kerja Anda. Untuk informasi selengkapnya, lihat Penyimpanan Instans DB dan Memodifikasi instans DB di Panduan Pengguna Amazon RDS.

Untuk meluncurkan instans DB RDS pada VPC default

1. Buka konsol RDS.
2. Di panel navigasi, pilih Basis Data.
3. Pilih Buat basis data.
4. Pilih Pembuatan Standar.

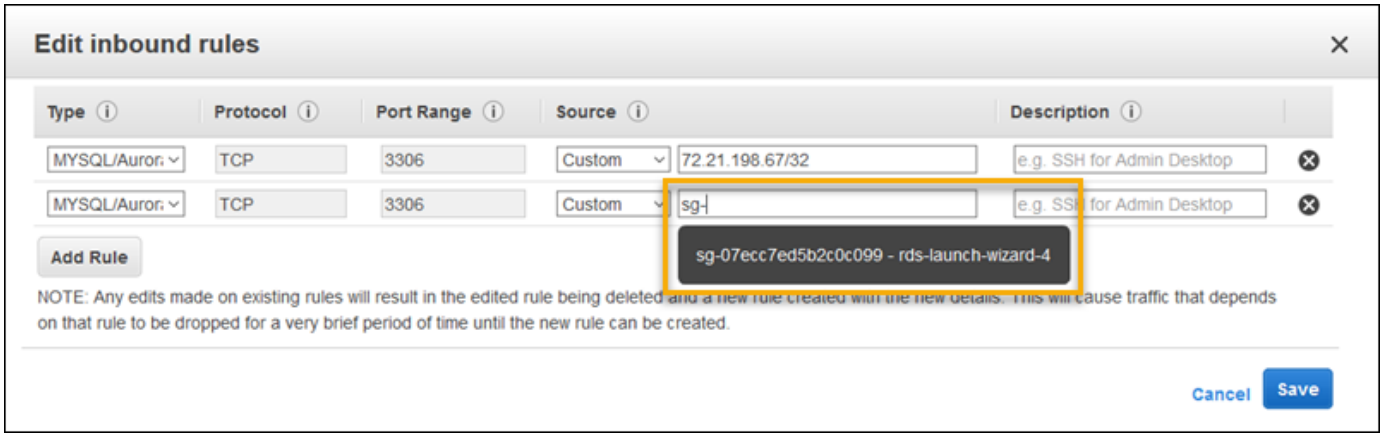
> **Important**
>
> Jangan pilih Pembuatan Mudah. Jika Anda memilihnya, Anda tidak dapat mengonfigurasi pengaturan yang diperlukan untuk meluncurkan RDS DB ini.

5. Dalam Konfigurasi tambahan, untuk Nama basis data awal, ketik **ebdb**.
6. Tinjau pengaturan default dan sesuaikan pengaturan ini sesuai dengan kebutuhan spesifik Anda. Perhatikan opsi berikut:
   - Kelas instans DB – Memilih ukuran instans yang memiliki jumlah memori dan daya CPU yang sesuai dengan beban kerja Anda.
   - Penerapan multi-AZ — Untuk ketersediaan tinggi, atur ini ke Buat Replica/Reader simpul Aurora di AZ yang berbeda.
   - Nama pengguna utama dan Kata sandi utama – Nama pengguna dan kata sandi basis data. Catat pengaturan ini karena Anda akan menggunakannya nanti.
7. Verifikasi pengaturan default untuk opsi lainnya, dan kemudian pilih Buat basis data.

Selanjutnya, ubah grup keamanan yang dilampirkan ke instans DB Anda untuk memungkinkan lalu lintas masuk pada port yang sesuai. Ini adalah grup keamanan yang sama yang akan Anda lampirkan ke lingkungan Elastic Beanstalk Anda nanti. Akibatnya, aturan yang Anda tambahkan akan memberikan izin akses masuk ke sumber daya lain di grup keamanan yang sama.

Untuk mengubah aturan masuk pada grup keamanan yang dilampirkan ke instans RDS Anda

1. Buka konsol Amazon RDS.
2. Pilih Basis data.
3. Pilih nama instans DB Anda untuk menampilkan detailnya.
4. Di bagian Konektivitas, catat Subnet, grup Keamanan, dan Titik Akhir yang ditampilkan di halaman ini. Ini agar Anda dapat menggunakan informasi ini nanti.
5. Di bawah Keamanan, Anda dapat melihat grup keamanan yang terkait dengan instans DB. Buka tautan untuk melihat grup keamanan di EC2 konsol Amazon.
6. Pada detail grup keamanan, pilih Masuk.
7. Pilih Edit.
8. Pilih Tambahkan Aturan.
9. Untuk Jenis, pilih mesin DB yang digunakan aplikasi Anda.
10. Untuk Sumber, ketik **`sg-`** untuk melihat daftar grup keamanan yang tersedia. Pilih grup keamanan yang terkait dengan grup Auto Scaling yang digunakan dengan lingkungan Elastic Beanstalk Anda. Ini agar EC2 instance Amazon di lingkungan dapat memiliki akses ke database.



11. Pilih Simpan.

Selanjutnya, tambahkan grup keamanan untuk instans DB ke lingkungan berjalan Anda. Dalam prosedur ini Elastic Beanstalk menyediakan kembali semua instance di lingkungan Anda dengan grup keamanan tambahan yang terpasang.

Untuk menambahkan grup keamanan ke lingkungan Anda

- Lakukan salah satu dari berikut ini:
  - Untuk menambahkan grup keamanan menggunakan konsol Elastic Beanstalk
    a. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
    b. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
    c. Pada panel navigasi, pilih Konfigurasi.
    d. Pada kategori konfigurasi Instans, pilih Edit.
    e. Di bawah grup EC2 keamanan, pilih grup keamanan untuk dilampirkan ke instance, selain grup keamanan instans yang dibuat Elastic Beanstalk.
    f. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.
    g. Baca peringatan, kemudian pilih Konfirmasi.
  - Untuk menambahkan grup keamanan menggunakan file konfigurasi, gunakan file `securitygroup-addexisting.config`contoh.

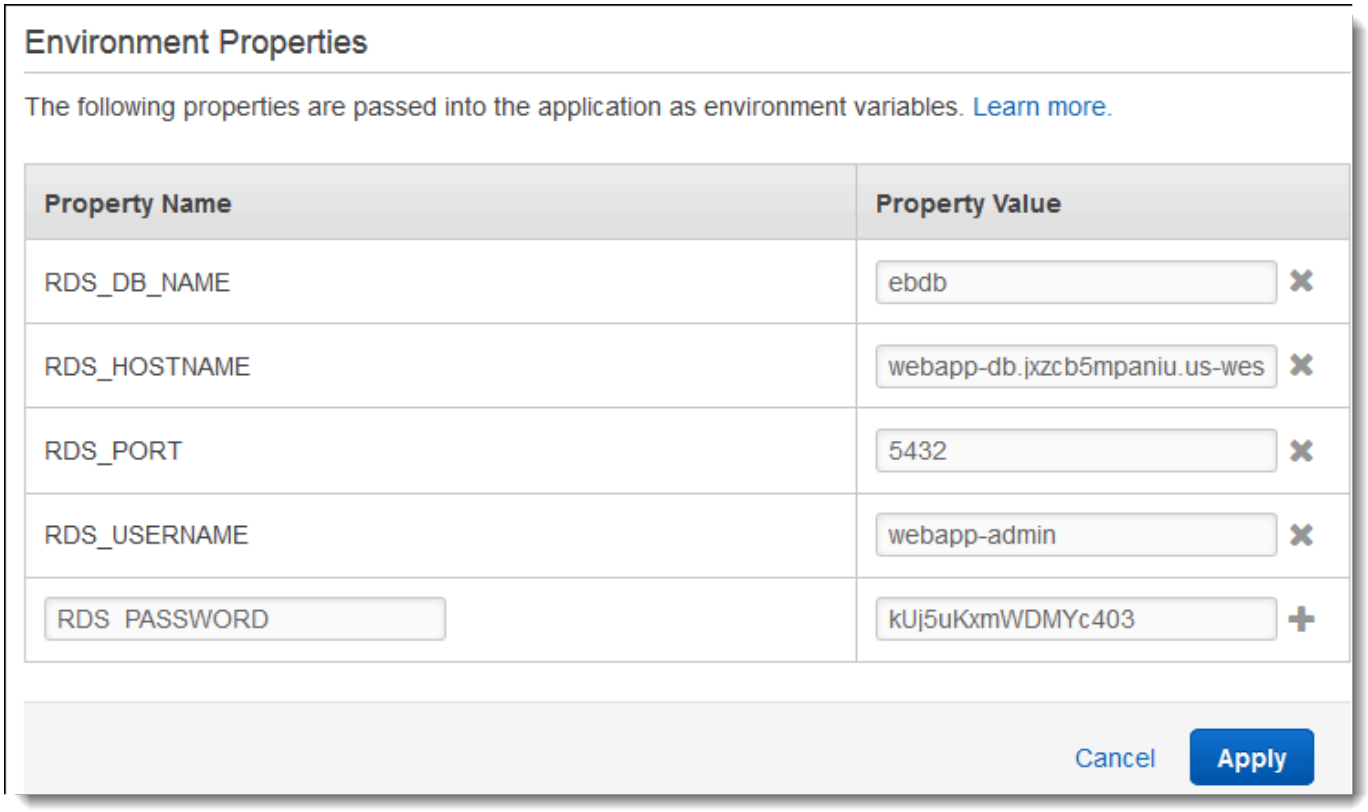
Selanjutnya, lulus informasi koneksi ke lingkungan Anda dengan menggunakan properti lingkungan. Saat Anda menambahkan instans DB ke lingkungan Anda dengan konsol Elastic Beanstalk, Elastic Beanstalk menggunakan properti lingkungan, seperti RDS_HOSTNAME, untuk meneruskan informasi koneksi ke aplikasi Anda. Anda dapat menggunakan properti yang sama. Dengan melakukan ini, Anda menggunakan kode aplikasi yang sama dengan instans DB terintegrasi dan instance DB eksternal. Atau, sebagai alternatif, Anda dapat memilih nama properti Anda sendiri.

Untuk mengonfigurasi properti lingkungan bagi instans DB Amazon RDS

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Dalam kategori konfigurasi Pembaruan, pemantauan, dan pencatatan, pilih Edit.

5. Pada bagian Properti lingkungan, tentukan variabel yang dibaca aplikasi Anda untuk membangun string koneksi. Untuk kompatibilitas dengan lingkungan yang memiliki instans DB RDS terintegrasi, gunakan nama dan nilai-nilai berikut. Anda dapat menemukan semua nilai, kecuali untuk kata sandi Anda, di Konsol RDS.

Nama properti	Deskripsi	Nilai properti
`RDS_HOSTNAME`	Nama host instans DB.	Di tab Konektivitas & keamanan di konsol Amazon RDS: Titik akhir.
`RDS_PORT`	Port tempat instans DB menerima koneksi. Nilai default bervariasi di antara mesin DB.	Di tab Konektivitas & keamanan di konsol Amazon RDS: Port.
`RDS_DB_NAME`	Nama basis data, **ebdb**.	Di tab Konfigurasi di konsol Amazon RDS: Nama DB.
`RDS_USERNAME`	Nama pengguna yang Anda konfigurasi untuk basis data Anda.	Di tab Konfigurasi di konsol Amazon RDS: Nama pengguna utama.
`RDS_PASSWORD`	Kata sandi yang Anda konfigurasi untuk basis data Anda.	Tidak tersedia untuk referensi di konsol Amazon RDS.



6. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Jika Anda belum memprogram aplikasi untuk membaca properti lingkungan dan membuat string koneksi, lihat topik khusus bahasa berikut untuk petunjuk:

- Java SE – Menghubungkan ke basis data (platform Java SE)
- Java dengan Tomcat – Menghubungkan ke basis data (platform Tomcat)
- Node.js – Menyambungkan ke basis data
- .NET – Menghubungkan ke basis data
- PHP – Menghubungkan ke database dengan PDO atau My SQLi
- Python – Menghubungkan ke basis data
- Ruby – Menyambungkan ke basis data

Akhirnya, tergantung pada ketika aplikasi Anda membaca variabel lingkungan, Anda mungkin perlu untuk memulai ulang server aplikasi pada instans di lingkungan Anda.

Memulai ulang server aplikasi lingkungan

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pilih Tindakan, lalu pilih Mulai ulang server aplikasi.

Menyimpan kredensi Amazon RDS di AWS Secrets Manager

Topik ini menjelaskan bagaimana AWS Secrets Manager dapat meningkatkan postur keamanan Anda untuk pengambilan kredensi dengan Elastic Beanstalk. Ini juga menyediakan referensi ke sumber daya tertentu yang dapat membantu Anda mengonfigurasi kredensil untuk aplikasi Elastic Beanstalk Anda.

AWS Secrets Manager membantu Anda meningkatkan postur keamanan Anda, dengan menyediakan kemampuan untuk menyimpan dan mengambil kredensi terenkripsi. Menyimpan kredensil di Secrets Manager membantu menghindari kemungkinan kompromi oleh siapa saja yang dapat memeriksa aplikasi Anda atau komponen yang terkait dengannya. Kode Anda dapat membuat panggilan runtime ke layanan Secrets Manager untuk mengambil kredensi secara dinamis. Secrets Manager juga menawarkan fitur seperti komponen caching rahasia sisi klien untuk bahasa runtime, yang meliputi Python, Go, dan Java.

Untuk informasi selengkapnya, lihat topik berikut di Panduan AWS Secrets Manager Pengguna.

- Bagaimana Amazon RDS menggunakan AWS Secrets Manager
- Buat rahasia AWS Secrets Manager database
- Ambil rahasia dari AWS Secrets Manager

Membersihkan instans Amazon RDS eksternal

Saat Anda menghubungkan instans Amazon RDS eksternal ke lingkungan Elastic Beanstalk, instans database tidak bergantung pada siklus hidup lingkungan Anda, dan karenanya, instans tersebut tidak dihapus saat Anda mengakhiri lingkungan. Untuk memastikan bahwa informasi pribadi yang mungkin Anda simpan dalam instance database tidak perlu disimpan, hapus catatan apa pun yang tidak Anda perlukan lagi. Atau, hapus instance database.

Menggunakan Elastic Beanstalk dengan Amazon S3

Topik ini menjelaskan bagaimana Elastic Beanstalk memanfaatkan Amazon Simple Storage Service (Amazon S3) dan jenis objek yang disimpannya di bucket S3. Ini juga mencatat objek mana yang harus Anda hapus secara manual setelah Anda mengakhiri lingkungan Elastic Beanstalk Anda dan memberikan instruksi untuk melakukannya.

Bucket akun pelanggan Elastic Beanstalk Amazon S3

Elastic Beanstalk membuat `elasticbeanstalk-`*`region`*`-`*`account-id`* bucket Amazon S3 terenkripsi yang diberi nama untuk setiap wilayah tempat Anda membuat lingkungan. AWS Akun Anda memiliki bucket ini. Elastic Beanstalk menyimpan file konfigurasi sementara dan objek lain untuk pengoperasian aplikasi Anda dengan benar di bucket ini. Elastic Beanstalk ACLs perlu diaktifkan untuk bucket yang dikelola layanan dan karenanya mengaktifkan Access Control List (ACL) bucket ini.

Ketahuilah bahwa Amazon S3 menonaktifkan bucket Access Control Lists (ACLs) secara default. Selain itu, topik ikhtisar ACL di Panduan Pengguna Amazon S3 merekomendasikan agar Anda ACLs tetap dinonaktifkan, kecuali untuk kasus penggunaan tertentu. Bucket yang dikelola layanan Elastic Beanstalk termasuk dalam kasus penggunaan yang perlu diaktifkan. ACLs Untuk menjaga keamanan, penerapan Elastic Beanstalk memberlakukan bahwa bucket ini dimiliki oleh akun yang menjalankan aplikasi.

Elastic Beanstalk mempertahankan enkripsi default yang disediakan oleh bucket Amazon S3. Untuk informasi selengkapnya tentang enkripsi bucket, lihat enkripsi default Amazon S3 di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon.

Isi ember akun pelanggan Elastic Beanstalk Amazon S3

Tabel berikut mencantumkan beberapa objek yang disimpan Elastic Beanstalk di bucket akun pelanggan Anda. Tabel juga menunjukkan objek mana yang harus dihapus secara manual. Untuk menghindari biaya penyimpanan yang tidak perlu, dan untuk memastikan bahwa informasi pribadi tidak disimpan, pastikan untuk menghapus objek ini secara manual bila Anda tidak membutuhkannya lagi.

Objek	Kapan disimpan?	Kapan dihapus?
Versi aplikasi	Bila Anda membuat lingkungan atau men-deploy kode aplikasi Anda ke lingkungan	Selama penghapusan aplikasi, dan menurut Siklus hidup versi.

Objek	Kapan disimpan?	Kapan dihapus?
	yang ada, Elastic Beanstalk menyimpan versi aplikasi di Amazon S3 dan mengaitkannya dengan lingkungan.	
Bundel sumber	Ketika Anda meng-upload versi aplikasi baru menggunakan konsol Elastic Beanstalk atau EB CLI, Elastic Beanstalk menyimpan salinannya di Amazon S3, dan menetapkan sebagai paket sumber lingkungan Anda.	Secara manual. Bila Anda menghapus versi aplikasi, Anda dapat memilih Hapus versi dari Amazon S3 untuk juga menghapus paket sumber terkait. Untuk rincian selengkapnya, lihat Mengelola versi aplikasi.
Platform khusus	Bila Anda membuat platform kustom, Elastic Beanstalk menyimpan sementara data terkait di Amazon S3.	Setelah berhasil menyelesaikan pembuatan platform kustom.
File log	Anda dapat meminta Elastic Beanstalk untuk mengambil instans berkas log (ekor atau paket log) dan menyimpannya di Amazon S3. Anda juga dapat mengaktifkan rotasi log dan mengonfigurasi lingkungan Anda untuk mempublikasikan log secara otomatis ke Amazon S3 setelah mereka diputar.	Log ekor dan paket: 15 menit setelah dibuat. Log yang diputar: Secara Manual.
Konfigurasi tersimpan	Secara manual.	Secara manual.

Menghapus objek dalam bucket Elastic Beanstalk Amazon S3

Ketika Anda mengakhiri lingkungan atau menghapus aplikasi, Elastic Beanstalk menghapus objek yang paling terkait dari Amazon S3. Untuk meminimalkan biaya penyimpanan aplikasi berjalan, secara rutin menghapus objek yang aplikasi Anda tidak perlu. Selain itu, perhatikan objek yang harus Anda hapus secara manual, seperti yang tercantum dalam Isi ember akun pelanggan Elastic

Beanstalk Amazon S3. Untuk memastikan bahwa informasi pribadi tidak perlu disimpan, hapus objek ini ketika Anda tidak membutuhkannya lagi.

- Hapus versi aplikasi yang tidak Anda harapkan untuk digunakan dalam aplikasi Anda lagi. Bila Anda menghapus versi aplikasi, Anda dapat memilih Hapus versi dari Amazon S3 untuk juga menghapus paket sumber terkait—salinan kode sumber dan file konfigurasi aplikasi Anda, yang Elastic Beanstalk diunggah ke Amazon S3 saat Anda men-deploy aplikasi atau mengunggah versi aplikasi. Untuk mempelajari cara menghapus versi aplikasi, lihat Mengelola versi aplikasi.
- Hapus log yang diputar yang tidak Anda perlukan. Sebagai alternatif, unduh atau pindahkan ke Amazon S3 Glacier untuk analisis lebih lanjut.
- Hapus konfigurasi tersimpan yang tidak akan Anda gunakan di lingkungan apapun lagi.

Menghapus bucket Elastic Beanstalk Amazon S3

Saat Elastic Beanstalk membuat bucket, ia juga membuat kebijakan bucket yang berlaku untuk bucket baru. Kebijakan ini memiliki dua tujuan:

- Untuk memungkinkan lingkungan menulis ke ember.
- Untuk mencegah penghapusan ember secara tidak sengaja.

Karena kebijakan Elastic Beanstalk berlaku untuk bucket yang dibuat untuk lingkungan Anda, Anda tidak diizinkan untuk menghapus bucket ini, kecuali jika Anda sengaja menghapus kebijakan bucket terlebih dahulu. Anda dapat menghapus kebijakan bucket dari bagian Izin properti bucket di konsol Amazon S3.

> **Warning**
>
> Kami menyarankan Anda menghapus objek tertentu yang tidak perlu dari bucket Elastic Beanstalk Amazon S3 Anda, alih-alih menghapus seluruh bucket.
> Jika Anda menghapus bucket yang dibuat Elastic Beanstalk di akun Anda, dan Anda masih memiliki aplikasi yang ada dan lingkungan yang berjalan di wilayah yang sesuai, aplikasi Anda mungkin berhenti berfungsi dengan benar. Misalnya:
>
> - Saat skala lingkungan habis, Elastic Beanstalk harus dapat menemukan versi aplikasi lingkungan di bucket Amazon S3 dan menggunakannya untuk memulai instans Amazon baru. EC2

- Bila Anda membuat platform kustom, Elastic Beanstalk menggunakan penyimpanan Amazon S3 sementara selama proses pembuatan.

Untuk informasi selengkapnya tentang implikasi penghapusan bucket S3, lihat pertimbangan yang tercantum dalam Menghapus bucket di Panduan Pengguna Amazon S3.

Untuk menghapus bucket penyimpanan Elastic Beanstalk (konsol)

Prosedur umum untuk menghapus bucket S3 juga dijelaskan dalam Menghapus bucket di Panduan Pengguna Amazon S3. Karena kami menghapus bucket yang dibuat oleh Elastic Beanstalk dalam prosedur berikut, kami menyertakan langkah-langkah tambahan untuk menghapus kebijakan bucket terlebih dahulu.

1. Buka konsol Amazon S3.
2. Buka halaman ember penyimpanan Elastic Beanstalk dengan memilih nama bucket.
3. Pilih tab Izin.
4. Pilih Kebijakan Bucket.
5. Pilih Hapus.
6. Kembali ke halaman utama konsol Amazon S3, lalu pilih ember penyimpanan Elastic Beanstalk.
7. Pilih Hapus Bucket.
8. Konfirmasikan bahwa Anda ingin menghapus bucket dengan memasukkan nama bucket ke dalam kolom teks, lalu pilih Hapus bucket.

Menggunakan Elastic Beanstalk dengan dan AWS Secrets Manager Parameter Store AWS Systems Manager

Topik ini menjelaskan bagaimana Anda dapat menggunakan AWS Secrets Manager dan AWS Systems Manager Parameter Store dengan lingkungan Elastic Beanstalk Anda untuk menyimpan dan mengambil informasi sensitif dengan aman, seperti kredensional dan kunci API. Aplikasi Anda dapat mengambil rahasia dan parameter yang tersimpan langsung dari toko-toko ini, menggunakan alat baris perintah APIs atau layanan ini.

Elastic Beanstalk juga menawarkan kemampuan untuk mereferensikan Secrets Manager dan Systems Manager Parameter Store data dalam variabel lingkungan. Ini adalah opsi aman untuk

aplikasi Anda untuk mengakses rahasia dan parameter asli yang disimpan oleh layanan ini tanpa harus mengelola panggilan API ke mereka.

Topik

- Mengambil rahasia dan parameter ke variabel lingkungan Elastic Beanstalk
- Izin IAM yang diperlukan untuk Elastic Beanstalk untuk mengakses rahasia dan parameter
- Menggunakan AWS Secrets Manager dan AWS Systems Manager Parameter Store
- Pemecahan masalah integrasi rahasia dengan variabel lingkungan Elastic Beanstalk

Mengambil rahasia dan parameter ke variabel lingkungan Elastic Beanstalk

Elastic Beanstalk dapat mengambil AWS Secrets Manager nilai AWS Systems Manager dari dan Parameter Store selama bootstrap instance dan menetapkannya ke variabel lingkungan untuk digunakan aplikasi Anda.

Poin-poin berikut merangkum konfigurasi, sinkronisasi, dan akses untuk menggunakan variabel lingkungan sebagai rahasia:

- Konfigurasikan variabel lingkungan Anda untuk menyimpan rahasia dengan menentukan Amazon Resource Names (ARNs) untuk rahasia dan parameter yang akan disimpan.
- Ketika nilai rahasia diperbarui atau diputar di Secrets Manager atau Systems Manager Parameter Store, Anda harus menyegarkan variabel lingkungan secara manual.
- Variabel lingkungan rahasia tersedia untuk perintah wadah ebextension dan kait platform.

Versi platform yang didukung

Versi platform yang dirilis pada atau setelah 26 Maret 2025 mendukung AWS Secrets Manager rahasia dan AWS Systems Manager parameter Parameter Store yang dikonfigurasi sebagai variabel lingkungan.

> **Note**
>
> Dengan pengecualian platform docker berbasis Docker dan ECS, versi platform Amazon Linux 2 tidak mendukung nilai variabel multiline. Untuk informasi selengkapnya tentang dukungan variabel multiline, lihatNilai multiline.

Topik

- Harga
- Konfigurasikan rahasia sebagai variabel lingkungan Elastic Beanstalk
- Praktik terbaik untuk sinkronisasi rahasia dengan variabel lingkungan Elastic Beanstalk
- Nilai multiline dalam variabel lingkungan Amazon Linux 2

Harga

Biaya standar berlaku untuk penggunaan Secrets Manager dan Systems Manager Parameter Store. Untuk informasi selengkapnya tentang harga, lihat situs web berikut:

- AWS Secrets Manager harga
- AWS Systems Manager harga (pilih Parameter Store dari daftar konten)

Elastic Beanstalk tidak mengenakan biaya untuk aplikasi Anda untuk mereferensikan rahasia lingkungan melalui variabel lingkungan. Namun, biaya standar berlaku untuk permintaan yang dilakukan Elastic Beanstalk untuk layanan ini atas nama Anda.

Konfigurasikan rahasia sebagai variabel lingkungan Elastic Beanstalk

Anda dapat menggunakan konsol Elastic Beanstalk, file konfigurasi `.ebextensions` di, CLI, AWS AWS dan SDK untuk mengonfigurasi rahasia dan parameter sebagai variabel lingkungan.

Topik

- Prasyarat
- Menggunakan konsol
- Konfigurasi menggunakan file di.ebextensions
- Konfigurasi menggunakan AWS CLI
- Konfigurasi menggunakan AWS SDK

Prasyarat

Sebelum Anda dapat mengatur variabel lingkungan Anda untuk referensi rahasia, Anda harus terlebih dahulu menyelesaikan langkah-langkah berikut.

Prosedur umum sebelum konfigurasi variabel lingkungan

1. Buat rahasia Secrets Manager atau parameter Parameter Store untuk menyimpan data sensitif Anda. Untuk informasi selengkapnya, lihat salah satu atau kedua topik berikut:
   - Menciptakan rahasia di the section called “Menggunakan Secrets Manager”
   - Membuat parameter di the section called “Menggunakan Systems Manager Parameter Store”
2. Siapkan izin IAM yang diperlukan untuk EC2 instance lingkungan Anda untuk mengambil rahasia dan parameter. Untuk informasi selengkapnya, lihat Izin IAM yang diperlukan.

Menggunakan konsol

Anda dapat menggunakan konsol Elastic Beanstalk untuk mengonfigurasi rahasia sebagai variabel lingkungan.

Untuk mengonfigurasi rahasia sebagai variabel lingkungan di konsol Elastic Beanstalk

1. Buka konsol Elastic Beanstalk, dan di daftar Wilayah, pilih konsol Elastic Beanstalk. Wilayah AWS
2. Di panel navigasi, pilih Lingkungan, dan kemudian pilih nama lingkungan Anda dari daftar.
3. Pada panel navigasi, pilih Konfigurasi.
4. Dalam kategori konfigurasi Pembaruan, pemantauan, dan pencatatan, pilih Edit.
5. Gulir ke bawah ke variabel lingkungan Runtime.
6. Pilih Tambahkan variabel lingkungan.
7. Untuk Source pilih Secrets Manager atau SSM Parameter Store.

   > **Note**
   >
   > Untuk informasi selengkapnya tentang opsi Teks biasa di drop-down, lihatMengkonfigurasi properti lingkungan (variabel lingkungan).

8. Untuk nama variabel Environment masukkan nama variabel lingkungan untuk menyimpan nilai rahasia atau parameter.
9. Untuk nilai variabel Environment masukkan ARN dari parameter Systems Manager Parameter Store atau rahasia Secrets Manager. Selama bootstrap instance Elastic Beanstalk akan memulai nilai variabel yang Anda masukkan di Langkah 8 ke nilai yang disimpan dalam sumber daya ARN ini.

Konsol memvalidasi jika nilai yang Anda masukkan adalah format ARN yang valid untuk toko yang Anda pilih di Langkah 7. Namun, itu tidak memvalidasi keberadaan sumber daya yang ditentukan oleh ARN atau jika Anda memiliki izin IAM yang diperlukan untuk mengaksesnya.

10. Jika Anda perlu menambahkan lebih banyak variabel ulangi Langkah 6 hingga Langkah 9.
11. Untuk menyimpan perubahan pilih Terapkan di bagian bawah halaman.

Konfigurasi menggunakan file di.ebextensions

Anda dapat menggunakan file konfigurasi Elastic Beanstalk untuk mengonfigurasi rahasia sebagai variabel lingkungan. Gunakan aws:elasticbeanstalk:aplikasi:environmentsecrets namespace untuk menentukan properti lingkungan.

Example .ebextensions/options.config untuk rahasia lingkungan (sintaks singkatan)

```
option_settings:
  aws:elasticbeanstalk:application:environmentsecrets:
    MY_SECRET: arn:aws:secretsmanager:us-east-1:111122223333:secret:mysecret
    MY_PARAMETER: arn:aws:ssm:us-east-1:111122223333:parameter/myparam
```

Example .ebextensions/options.config untuk rahasia lingkungan (sintaks standar)

```
option_settings:
  - namespace: aws:elasticbeanstalk:application:environmentsecrets
    option_name: MY_SECRET
    value: arn:aws:secretsmanager:us-east-1:111122223333:secret:mysecret
  - namespace: aws:elasticbeanstalk:application:environmentsecrets
    option_name: MY_PARAMETER
    value: arn:aws:ssm:us-east-1:111122223333:parameter/myparam
```

Konfigurasi menggunakan AWS CLI

Anda dapat menggunakan AWS Command Line Interface (AWS CLI) untuk mengonfigurasi rahasia sebagai variabel lingkungan Elastic Beanstalk. Bagian ini memberikan contoh perintah create-environment dan update-environment dengan namespace. aws:elasticbeanstalk:aplikasi:environmentsecrets Ketika Elastic Beanstalk EC2 mem-bootstrap instance untuk lingkungan yang direferensikan perintah ini, ia menginisialisasi variabel lingkungan dengan rahasia yang diambil dan nilai parameter. Ini mengambil nilai-nilai ini dari masing-masing ARNs Secrets Manager dan Systems Manager Parameter Store.

Dua contoh berikut menggunakan perintah create-environment untuk menambahkan rahasia dan parameter, dikonfigurasi sebagai variabel lingkungan bernama dan. `MY_SECRET` `MY_PARAMETER`

Example dari create-environment dengan rahasia yang dikonfigurasi sebagai variabel lingkungan (opsi namespace sebaris)

```
aws elasticbeanstalk create-environment \
--region us-east-1 \
--application-name my-app \
--environment-name my-env \
--solution-stack-name "64bit Amazon Linux 2023 v6.5.0 running Node.js 20" \
--option-settings \
Namespace=aws:elasticbeanstalk:application:environmentsecrets,OptionName=MY_SECRET,Value=arn:aw
east-1:111122223333:secret:mysecret \
Namespace=aws:elasticbeanstalk:application:environmentsecrets,OptionName=MY_PARAMETER,Value=arn
east-1:111122223333:parameter/myparam
```

Sebagai alternatif, gunakan `options.json` file untuk menentukan opsi namespace alih-alih memasukkannya sebaris.

Example dari create-environment dengan rahasia yang dikonfigurasi sebagai variabel lingkungan (opsi namespace dalam file) **`options.json`**

```
aws elasticbeanstalk create-environment \
--region us-east-1 \
--application-name my-app \
--environment-name my-env \
--solution-stack-name "64bit Amazon Linux 2023 v6.5.0 running Node.js 20" \
--option-settings file://options.json
```

Example

```
### example options.json ###
[
  {
    "Namespace": "aws:elasticbeanstalk:application:environmentsecrets",
    "OptionName": "MY_SECRET",
    "Value": "arn:aws:secretsmanager:us-east-1:111122223333:secret:mysecret"
  },
  {
```

```
    "Namespace": "aws:elasticbeanstalk:application:environmentsecrets",
    "OptionName": "MY_PARAMETER",
    "Value": "arn:aws:ssm:us-east-1:111122223333:parameter/myparam"
  }
]
```

Contoh berikutnya mengkonfigurasi variabel lingkungan, bernama `MY_SECRET` dan`MY_PARAMETER`, untuk menyimpan rahasia dan parameter untuk lingkungan yang ada. Perintah update-environment meneruskan opsi dengan sintaks yang sama dengan `create-environment` perintah, baik inline atau dengan file. `options.json` Contoh berikut menunjukkan perintah menggunakan `options.json` file yang sama yang juga digunakan dalam contoh sebelumnya.

Example dari update-environment dengan rahasia yang dikonfigurasi sebagai variabel lingkungan (opsi namespace dalam file) **`options.json`**

```
aws elasticbeanstalk update-environment \
--region us-east-1 \
--application-name my-app \
--environment-name my-env \
--solution-stack-name "64bit Amazon Linux 2023 v6.5.0 running Node.js 20" \
--option-settings file://options.json
```

Konfigurasi menggunakan AWS SDK

Anda dapat mengonfigurasi rahasia dan parameter sebagai variabel lingkungan menggunakan file AWS SDKs. Mirip dengan `create-environment` AWS CLI perintah `update-environment` dan yang disebutkan di bagian sebelumnya, Anda dapat menggunakan tindakan CreateEnvironmentdan UpdateEnvironmentAPI. Gunakan parameter `OptionSettings` permintaan untuk menentukan opsi aws:elasticbeanstalk:aplikasi:environmentsecrets namespace.

Praktik terbaik untuk sinkronisasi rahasia dengan variabel lingkungan Elastic Beanstalk

Topik ini merekomendasikan praktik terbaik bagi aplikasi Anda untuk menggunakan rahasia lingkungan dengan Secrets Manager atau Systems Manager Parameter Store. Aplikasi Elastic Beanstalk Anda tidak akan secara otomatis menerima nilai yang diperbarui jika data penyimpanan rahasia diperbarui atau diputar. Elastic Beanstalk hanya menarik rahasia ke dalam variabel lingkungan pada saat bootstrap instance.

Menyegarkan variabel lingkungan Anda

Untuk memicu lingkungan Elastic Beanstalk Anda untuk memperbaiki nilai rahasia terbaru dari toko rahasia mereka, kami sarankan Anda menjalankan atau operasi. `UpdateEnvironment` `RestartAppServer` Anda dapat menjalankan operasi ini menggunakan konsol Elastic Beanstalk, the AWS CLI, atau Elastic Beanstalk API. Untuk informasi selengkapnya, lihat AWS CLI contoh Elastic Beanstalk, atau Referensi API.AWS Elastic Beanstalk

Mengelola efek penskalaan otomatis pada sinkronisasi rahasia

Jika peristiwa skala atau penggantian instance terjadi setelah pembaruan toko rahasia, instance baru yang muncul akan memiliki nilai rahasia terbaru dari Secrets Manager atau Systems Manager Parameter Store. Peristiwa semacam itu dapat terjadi bahkan jika tidak semua contoh lain di lingkungan telah disegarkan untuk mengambil rahasia baru.

> **Important**
>
> Anda harus memastikan bahwa aplikasi Anda dapat menggunakan dua nilai rahasia yang berbeda untuk variabel lingkungan yang sama. Ini mengakomodasi peristiwa di mana pembaruan rahasia terjadi di Secrets Manager atau Systems Manager Parameter Store, diikuti dengan skala keluar atau penggantian instance di lingkungan Anda, sementara instance lainnya sedang menunggu penyegaran variabel lingkungan. Selama periode tunggu untuk penyegaran, tidak semua instance lingkungan akan memiliki nilai yang sama untuk variabel lingkungan penyimpanan rahasia.

Contoh kasus penggunaan seperti itu adalah rotasi kredenal database. Ketika peristiwa skala keluar mengikuti rotasi kredensi, rahasia lingkungan yang direferensikan oleh instance yang baru di-bootstrap berisi kredenal database yang diperbarui. Namun, rahasia lingkungan yang direferensikan oleh instance yang ada mempertahankan nilai lama sampai disegarkan oleh operasi atau. `UpdateEnvironment` `RestartAppServer`

Nilai multiline dalam variabel lingkungan Amazon Linux 2

Nilai multiline terdiri dari lebih dari satu baris dan termasuk karakter baris baru. Dengan pengecualian Docker dan platform Docker berbasis ECS, platform yang berjalan di Amazon Linux 2 tidak mendukung nilai multiline untuk variabel lingkungan

> **Note**
>
> Elastic Beanstalk akan gagal dalam penerapan lingkungan yang terpengaruh jika mendeteksi nilai multiline.

Opsi berikut dapat berfungsi sebagai solusi atau solusi untuk masalah multiline:

- Tingkatkan lingkungan Amazon Linux 2 Anda ke Amazon Linux 2023. Untuk informasi selengkapnya, lihat Migrasi dari Amazon Linux 2 ke Amazon Linux 2023.
- Hapus karakter baris baru dari nilai rahasia Anda. Salah satu contoh pendekatan adalah Base64 menyandikan nilai Anda sebelum menyimpannya di toko rahasia. Aplikasi Anda kemudian perlu memecahkan kode nilai kembali ke format asli ketika mereferensikannya dari variabel rahasia lingkungan.
- Rancang kode aplikasi Anda untuk mengambil data langsung dari Secrets Manager atau Systems Manager Parameter Store. Untuk informasi selengkapnya, lihat Mengambil rahasia di Menggunakan Secrets Manager atau Mengambil parameter. Menggunakan Systems Manager Parameter Store

Izin IAM yang diperlukan untuk Elastic Beanstalk untuk mengakses rahasia dan parameter

Anda harus memberikan izin yang diperlukan untuk EC2 instance lingkungan Anda untuk mengambil rahasia dan parameter untuk AWS Secrets Manager dan AWS Systems Manager Parameter Store. Izin diberikan ke EC2 instance melalui peran profil EC2 instans.

Bagian berikut mencantumkan izin khusus yang perlu Anda tambahkan ke profil EC2 instance, tergantung pada layanan yang Anda gunakan. Ikuti langkah-langkah yang disediakan di Perbarui kebijakan izin untuk peran dalam Panduan Pengguna IAM untuk menambahkan izin ini.

> **Izin IAM untuk platform Docker yang dikelola ECS**
>
> Platform Docker yang dikelola ECS memerlukan izin IAM tambahan untuk yang disediakan dalam topik ini. Untuk informasi selengkapnya tentang semua izin yang diperlukan untuk lingkungan platform Docker terkelola ECS Anda guna mendukung integrasi variabel lingkungan Elastic Beanstalk dengan rahasia, lihat. Format ARN Peran Eksekusi

Topik

- Izin IAM yang diperlukan untuk Secrets Manager
- Izin IAM yang diperlukan Systems Manager Parameter Store

Izin IAM yang diperlukan untuk Secrets Manager

Izin berikut memberikan akses untuk mengambil rahasia terenkripsi dari toko: AWS Secrets Manager

- manajer rahasia: GetSecretValue
- kms:Decrypt

Izin untuk mendekripsi hanya AWS KMS key diperlukan jika rahasia Anda menggunakan kunci yang dikelola pelanggan alih-alih kunci default. Penambahan kunci khusus ARN Anda menambahkan izin untuk mendekripsi kunci yang dikelola pelanggan.

Example kebijakan dengan Secrets Manager dan izin kunci KMS

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetSecretValue",
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:secretsmanager:us-east-1:111122223333:secret:my-secret",
                "arn:aws:kms:us-east-1:111122223333:key/my-key"
            ]
        }
    ]
}
```

Izin IAM yang diperlukan Systems Manager Parameter Store

Izin berikut memberikan akses untuk mengambil parameter terenkripsi dari Parameter Store: AWS Systems Manager

- ssm: GetParameter

- kms:Decrypt

Izin untuk mendekripsi hanya AWS KMS key diperlukan untuk tipe `SecureString` parameter yang menggunakan kunci yang dikelola pelanggan, bukan kunci default. Penambahan kunci khusus ARN Anda menambahkan izin untuk mendekripsi kunci yang dikelola pelanggan. Jenis parameter reguler yang tidak dienkripsi, `String` dan`StringList`, tidak memerlukan file. AWS KMS key

Example kebijakan dengan Systems Manager dan izin AWS KMS kunci

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetParameter",
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:ssm:us-east-1:111122223333:parameter/my-parameter",
                "arn:aws:kms:us-east-1:111122223333:key/my-key"
            ]
        }
    ]
}
```

Menggunakan AWS Secrets Manager dan AWS Systems Manager Parameter Store

Topik ini memberikan pengenalan singkat tentang AWS Secrets Manager dan AWS Systems Manager Parameter Store, informasi harga, dan referensi untuk mempelajari lebih lanjut tentang membuat dan mengambil rahasia, menggunakan konsol dan opsi terprogram.

Tentang Secrets Manager

AWS Secrets Manager membantu Anda mengelola, mengambil, dan memutar rahasia sepanjang siklus hidupnya. Contoh data rahasia yang dapat Anda kelola termasuk kredensi basis data, kredensi aplikasi, OAuth token, dan kunci API. Secrets Manager memungkinkan Anda mengonfigurasi jadwal rotasi otomatis untuk rahasia Anda.

Tentang Systems Manager Parameter Store

Parameter Store adalah alat di AWS Systems Manager. Ini menyediakan penyimpanan hierarkis yang aman untuk manajemen data konfigurasi dan manajemen rahasia. Anda dapat mengelola data konfigurasi penting sebagai nilai parameter. Contoh data yang dapat Anda kelola dengan Parameter Store mencakup Amazon Machine Image (AMI) IDs, kode lisensi, kata sandi, dan string database.

> **Harga**
>
> Biaya standar berlaku untuk penggunaan Secrets Manager dan Systems Manager Parameter Store. Untuk informasi selengkapnya tentang harga, lihat situs web berikut:
>
> - AWS Secrets Manager harga
> - AWS Systems Manager harga (pilih Parameter Store dari daftar konten)

Topik

- Menggunakan Secrets Manager untuk membuat dan mengambil rahasia
- Menggunakan Systems Manager Parameter Store untuk membuat dan mengambil parameter

Menggunakan Secrets Manager untuk membuat dan mengambil rahasia

Anda dapat membuat dan mengambil rahasia Secrets Manager menggunakan AWS Secrets Manager konsol, SDK AWS CLI, atau AWS SDK. Lihat sumber daya berikut untuk mempelajari lebih lanjut tentang berbagai metode untuk membuat dan mengambil rahasia Secrets Manager.

Menciptakan rahasia

- Konsol — Buat AWS Secrets Manager rahasia (konsol) di Panduan AWS Secrets Manager Pengguna
- AWS CLI - AWS CLI dalam Panduan Pengguna AWS Secrets Manager
- AWS SDK — AWS SDK di Panduan Pengguna AWS Secrets Manager

Mengambil rahasia

- Konsol — Dapatkan nilai rahasia (konsol) di Panduan AWS Secrets Manager Pengguna
- AWS CLI - Dapatkan nilai rahasia (AWS CLI) di Panduan Pengguna AWS Secrets Manager
- AWS SDK — Contoh kode untuk Secrets Manager yang digunakan AWS SDKs di AWS SDK Code Examples Code Library

- Metode lain — Dapatkan rahasia dari AWS Secrets Manager dalam Panduan AWS Secrets Manager Pengguna

Untuk informasi lebih lanjut tentang AWS Secrets Manager, lihat Apa itu AWS Secrets Manager? dalam AWS Secrets Manager User Guide.

Menggunakan Systems Manager Parameter Store untuk membuat dan mengambil parameter

Anda dapat membuat dan mengambil parameter Parameter Store menggunakan AWS Systems Manager konsol, SDK AWS CLI, atau AWS SDK. Lihat sumber daya berikut untuk mempelajari lebih lanjut tentang berbagai metode untuk membuat dan mengambil parameter Parameter Store.

Membuat parameter

- Konsol — Membuat parameter Systems Manager (konsol) di Panduan AWS Systems Manager Pengguna
- AWS CLI - Buat parameter Systems Manager (AWS CLI) di Panduan Pengguna AWS Systems Manager
- AWS SDK — Gunakan PutParameter dengan AWS SDK atau AWS CLI di SDK Code AWS Examples Code Library

Mengambil parameter

- Konsol — Mencari parameter (konsol) di Panduan AWS Systems Manager Pengguna
- AWS CLI - Gunakan GetParameter dengan AWS SDK atau AWS CLI di Perpustakaan Kode Contoh Kode AWS SDK
- AWS SDK — Gunakan GetParameter dengan AWS SDK atau AWS CLI di SDK Code AWS Examples Code Library

Untuk informasi selengkapnya, lihat Penyimpanan Parameter AWS Systems Manager dalam Panduan Pengguna AWS Systems Manager .

Pemecahan masalah integrasi rahasia dengan variabel lingkungan Elastic Beanstalk

Acara: Penerapan instance gagal mendapatkan satu atau lebih rahasia

Pesan ini menunjukkan bahwa Elastic Beanstalk tidak dapat mengambil satu atau lebih rahasia yang ditentukan selama penerapan aplikasi Anda.

- Periksa apakah sumber daya yang ditentukan oleh nilai ARN dalam konfigurasi variabel lingkungan Anda ada.
- Konfirmasikan bahwa peran profil instans EC2 Elastic Beanstalk Anda memiliki izin IAM yang diperlukan untuk mengakses sumber daya.
- Jika peristiwa ini dipicu melalui `RestartAppServer` operasi, setelah masalah diperbaiki, coba lagi `RestartAppServer` panggilan untuk menyelesaikan masalah.
- Jika acara dipicu melalui `UpdateEnvironment` panggilan, coba lagi `UpdateEnvironment` operasi.

Untuk contoh perintah ini, lihat AWS CLI contoh untuk Elastic Beanstalk. Untuk informasi selengkapnya tentang tindakan API untuk operasi ini, lihat Referensi AWS Elastic Beanstalk API.

Peristiwa: Penerapan instance mendeteksi satu atau lebih nilai lingkungan multiline, yang tidak didukung untuk platform ini

Variabel multiline tidak didukung untuk platform Amazon Linux 2, tidak termasuk platform Docker yang dikelola Docker dan ECS. Untuk opsi yang tersedia untuk melanjutkan, lihatNilai multiline.

Acara: CreateEnvironment gagal saat rahasia ditentukan

Ketika `CreateEnvironment` gagal dan Anda memiliki rahasia sebagai variabel lingkungan, Anda perlu mengatasi masalah mendasar dan kemudian menggunakannya `UpdateEnvironment` untuk menyelesaikan pengaturan lingkungan. Jangan gunakan`RestartAppServer`, karena tidak akan cukup untuk memunculkan lingkungan dalam situasi ini. Untuk contoh perintah ini, lihat AWS CLI contoh untuk Elastic Beanstalk. Untuk informasi selengkapnya tentang tindakan API untuk operasi ini, lihat Referensi AWS Elastic Beanstalk API.

Menggunakan Elastic Beanstalk dengan Amazon VPC

Topik ini menjelaskan manfaat menggunakan titik akhir VPC dengan Elastic Beanstalk dan berbagai jenis konfigurasi yang dapat Anda terapkan.

Anda dapat menggunakan Amazon Virtual Private Cloud (Amazon VPC) untuk membuat jaringan aman untuk aplikasi Elastic Beanstalk dan sumber daya terkait. AWS Ketika Anda membuat lingkungan Anda, Anda memilih VPC, subnet, dan grup keamanan yang digunakan untuk instans

aplikasi Anda dan penyeimbang beban. Anda dapat menggunakan konfigurasi VPC yang Anda sukai selama memenuhi persyaratan berikut.

Persyaratan VPC

- Akses Internet – Instans dapat memiliki akses ke internet melalui salah satu metode berikut:
  - Subnet Publik – Instans memiliki alamat IP publik dan menggunakan Gateway Internet untuk mengakses internet.
  - Subnet Pribadi – Instans menggunakan perangkat NAT untuk mengakses internet.

  > **Note**
  >
  > Jika Anda mengonfigurasi VPC endpoint di VPC Anda untuk terhubung ke layanan `elasticbeanstalk` dan `elasticbeanstalk-health`, akses internet bersifat opsional, dan hanya diperlukan jika aplikasi Anda secara khusus membutuhkannya. Tanpa VPC endpoint, VPC Anda harus memiliki akses ke internet.
  > Default VPC yang Elastic Beanstalk atur untuk Anda menyediakan akses internet.

  Elastic Beanstalk tidak mendukung pengaturan proksi seperti `HTTPS_PROXY` untuk mengonfigurasi proksi web.
- NTP – Instans dalam lingkungan Elastic Beanstalk Anda menggunakan Network Time Protocol (NTP) untuk menyinkronkan jam sistem. Jika instans tidak dapat berkomunikasi pada UDP port 123, jam mungkin tidak sinkron, menyebabkan masalah dengan pelaporan kondisi Elastic Beanstalk. Pastikan grup dan jaringan keamanan VPC Anda ACLs mengizinkan lalu lintas UDP masuk dan keluar di port 123 untuk menghindari masalah ini.

elastic-beanstalk-samplesRepositori menyediakan AWS CloudFormation template yang dapat Anda gunakan untuk membuat VPC untuk digunakan dengan lingkungan Elastic Beanstalk Anda.

Untuk membuat sumber daya dengan AWS CloudFormation template

1. Klon repositori sampel atau unduh templat menggunakan tautan di README.
2. Buka konsol AWS CloudFormation.
3. Pilih Membuat tumpukan.
4. Pilih Unggah templat ke Amazon S3.
5. Pilih Unggah file dan mengungah file templat dari mesin lokal Anda.

6. Pilih Selanjutnya dan ikuti petunjuk untuk membuat tumpukan dengan sumber daya dalam templat.

Saat pembuatan tumpukan selesai, periksa tab Output untuk menemukan ID VPC dan subnet. IDs Gunakan ini untuk mengonfigurasi VPC di wizard lingkungan baru kategori konfigurasi jaringan.

Topik

- VPC publik
- VPC publik/privat
- VPC privat
- Contoh: Meluncurkan aplikasi Elastic Beanstalk di VPC dengan host bastion
- Contoh: Meluncurkan Elastic Beanstalk di VPC dengan Amazon RDS
- Menggunakan Elastic Beanstalk dengan VPC endpoint
- Menggunakan kebijakan titik akhir untuk mengontrol akses dengan VPC endpoint

VPC publik

AWS CloudFormation template — vpc-public.yaml

Pengaturan (beban seimbang)

- Visibilitas penyeimbang beban – Publik
- Subnet penyeimbang beban – Kedua subnet publik
- IP publik instans – Diaktifkan
- Subnet instans – Kedua subnet publik
- Grup keamanan instans – Tambahkan grup keamanan default

Pengaturan (instans tunggal)

- Subnets instance – Salah satu subnet publik
- Grup keamanan instans – Tambahkan grup keamanan default

Dasar tata letak VPC hanya-publik mencakup satu atau lebih subnet publik, gateway internet, dan grup keamanan default yang memungkinkan lalu lintas antara sumber daya di VPC. Ketika Anda

membuat lingkungan di VPC, Elastic Beanstalk menciptakan sumber daya tambahan yang bervariasi tergantung pada jenis lingkungannya.

Sumber daya VPC

- Satu instans – Elastic Beanstalk membuat grup keamanan untuk instans aplikasi yang memungkinkan lalu lintas di port 80 dari internet, dan memberikan instans sebuah IP Elastic untuk memberikan alamat IP publik. Nama domain lingkungan ditetapkan ke alamat IP publik instans.
- Beban seimbang – Elastic Beanstalk menciptakan grup keamanan untuk penyeimbang beban yang memungkinkan lalu lintas di port 80 dari internet, dan grup keamanan untuk instans aplikasi yang memungkinkan lalu lintas dari grup keamanan penyeimbang beban. Nama domain lingkungan menyelesaikan nama domain publik penyeimbang beban.

Hal ini serupa dengan cara bahwa Elastic Beanstalk mengelola jaringan saat menggunakan VPC default. Keamanan dalam subnet publik bergantung pada penyeimbang beban dan grup keamanan instans yang dibuat oleh Elastic Beanstalk. Ini adalah konfigurasi paling mahal karena tidak memerlukan NAT Gateway.

VPC publik/privat

AWS CloudFormation template — vpc-privatepublic.yaml

Pengaturan (beban seimbang)

- Visibilitas penyeimbang beban – Publik
- Subnet penyeimbang beban – Kedua subnet publik
- IP publik instans – Nonaktifkan
- Subnets instans – Kedua subnet privat
- Grup keamanan instans – Tambahkan grup keamanan default

Untuk keamanan tambahan, tambahkan subnet privat ke VPC Anda untuk membuat tata letak publik-privat. Tata letak ini memerlukan penyeimbang beban dan NAT gateway di subnet publik, dan memungkinkan Anda menjalankan instans aplikasi, basis data, dan sumber daya lainnya di subnet privat. Intans dalam subnet privat hanya dapat berkomunikasi dengan internet melalui penyeimbang beban dan gateway NAT.

VPC privat

AWS CloudFormation template — vpc-private.yaml

Pengaturan (beban seimbang)

- Visibilitas penyeimbang beban – Privat
- Subnet penyeimbang beban – Kedua subnet pribadi
- IP publik instans – Nonaktifkan
- Subnets instans – Kedua subnet privat
- Grup keamanan instans – Tambahkan grup keamanan default

Untuk aplikasi internal yang seharusnya tidak memiliki akses dari internet, Anda dapat menjalankan semuanya di subnet pribadi dan mengonfigurasi penyeimbang beban agar menghadap secara internal (ubah visibilitas Penyeimbang beban ke Internal). Templat ini membuat VPC tanpa subnet publik dan tidak ada gateway internet. Gunakan tata letak ini untuk aplikasi yang seharusnya hanya dapat diakses dari VPC yang sama atau VPN terlampir.

Menjalankan lingkungan Elastic Beanstalk di VPC pribadi

Ketika Anda membuat lingkungan Elastic Beanstalk di VPC pribadi, lingkungan tidak memiliki akses ke internet. Aplikasi Anda mungkin memerlukan akses ke layanan Elastic Beanstalk atau layanan lainnya. Lingkungan Anda mungkin menggunakan pelaporan kondisi yang ditingkatkan, dan dalam hal ini instans lingkungan mengirimkan informasi kondisi ke layanan kondisi yang ditingkatkan. Dan kode Elastic Beanstalk pada instance lingkungan mengirimkan lalu lintas AWS ke layanan lain, dan lalu lintas lainnya ke AWS non-endpoint (misalnya, untuk mengunduh paket dependensi untuk aplikasi Anda). Berikut adalah beberapa langkah yang mungkin perlu Anda ambil dalam kasus ini untuk memastikan bahwa lingkungan Anda bekerja dengan baik.

- Mengonfigurasi VPC endpoint untuk Elastic Beanstalk – Elastic Beanstalk dan layanan kondisi yang ditingkatkan mendukung titik akhir VPC, yang memastikan bahwa lalu lintas ke layanan ini tetap berada di dalam jaringan Amazon dan tidak memerlukan akses internet. Untuk informasi selengkapnya, lihat the section called “Titik akhir VPC”.
- Konfigurasikan titik akhir VPC untuk layanan tambahan — Instans Elastic Beanstalk mengirim lalu lintas ke beberapa layanan AWS lain atas nama Anda: Amazon Simple Storage Service (Amazon S3) Service, Amazon Simple Queue Service (Amazon Simple Queue Service),, dan Amazon Logs.

AWS CloudFormation CloudWatch Anda harus mengonfigurasi VPC endpoint untuk layanan ini juga. Untuk informasi rinci tentang VPC endpoint, termasuk tautan per layanan, lihat VPC endpoint di Panduan Pengguna Amazon VPC.

> **Note**
>
> Beberapa AWS layanan, termasuk Elastic Beanstalk, mendukung titik akhir VPC di sejumlah Wilayah terbatas. AWS Saat Anda mendesain solusi VPC pribadi Anda, verifikasi bahwa Elastic Beanstalk dan layanan dependen lainnya yang disebutkan di sini mendukung titik akhir VPC di Wilayah yang Anda pilih. AWS

- Menyediakan gambar Docker privat – Dalam lingkungan Docker, kode pada instans lingkungan mungkin mencoba menarik gambar Docker yang dikonfigurasi dari internet selama pembuatan lingkungan dan gagal. Untuk menghindari kegagalan ini, buat gambar Docker kustom di lingkungan Anda, atau gunakan gambar Docker yang disimpan di Amazon Elastic Container Registry (Amazon ECR) dan mengonfigurasi VPC endpoint untuk layanan Amazon ECR.
- Aktifkan nama DNS — Kode Elastic Beanstalk pada instance lingkungan mengirimkan lalu lintas AWS ke semua layanan menggunakan titik akhir publiknya. Untuk memastikan bahwa lalu lintas ini berjalan, pilih opsi Aktifkan nama DNS ketika Anda mengonfigurasi semua antarmuka VPC endpoint. Ini menambahkan entri DNS di VPC Anda yang memetakan titik akhir layanan publik untuk antarmuka VPC endpoint.

> **Important**
>
> Jika VPC Anda tidak bersifat privat dan memiliki akses internet publik, dan jika Aktifkan nama DNS dinonaktifkan untuk setiap VPC endpoint, lalu lintas ke layanan masing-masing perjalanan melalui internet publik. Ini mungkin bukan apa yang Anda inginkan. Sangat mudah untuk mendeteksi masalah ini dengan VPC privat, karena mencegah lalu lintas ini melewati dan Anda menerima kesalahan. Namun, dengan VPC menghadap publik, Anda tidak mendapatkan indikasi.

- Sertakan dependensi aplikasi – Jika aplikasi Anda memiliki dependensi seperti paket waktu aktif bahasa, mungkin mencoba untuk mengunduh dan menginstalnya dari internet selama pembuatan lingkungan dan gagal. Untuk menghindari kegagalan ini, sertakan semua paket ketergantungan dalam paket sumber aplikasi Anda.
- Gunakan versi platform saat ini – Pastikan lingkungan Anda menggunakan versi platform yang dirilis pada 24 Februari 2020 atau lebih baru. Secara khusus, gunakan versi platform yang dirilis

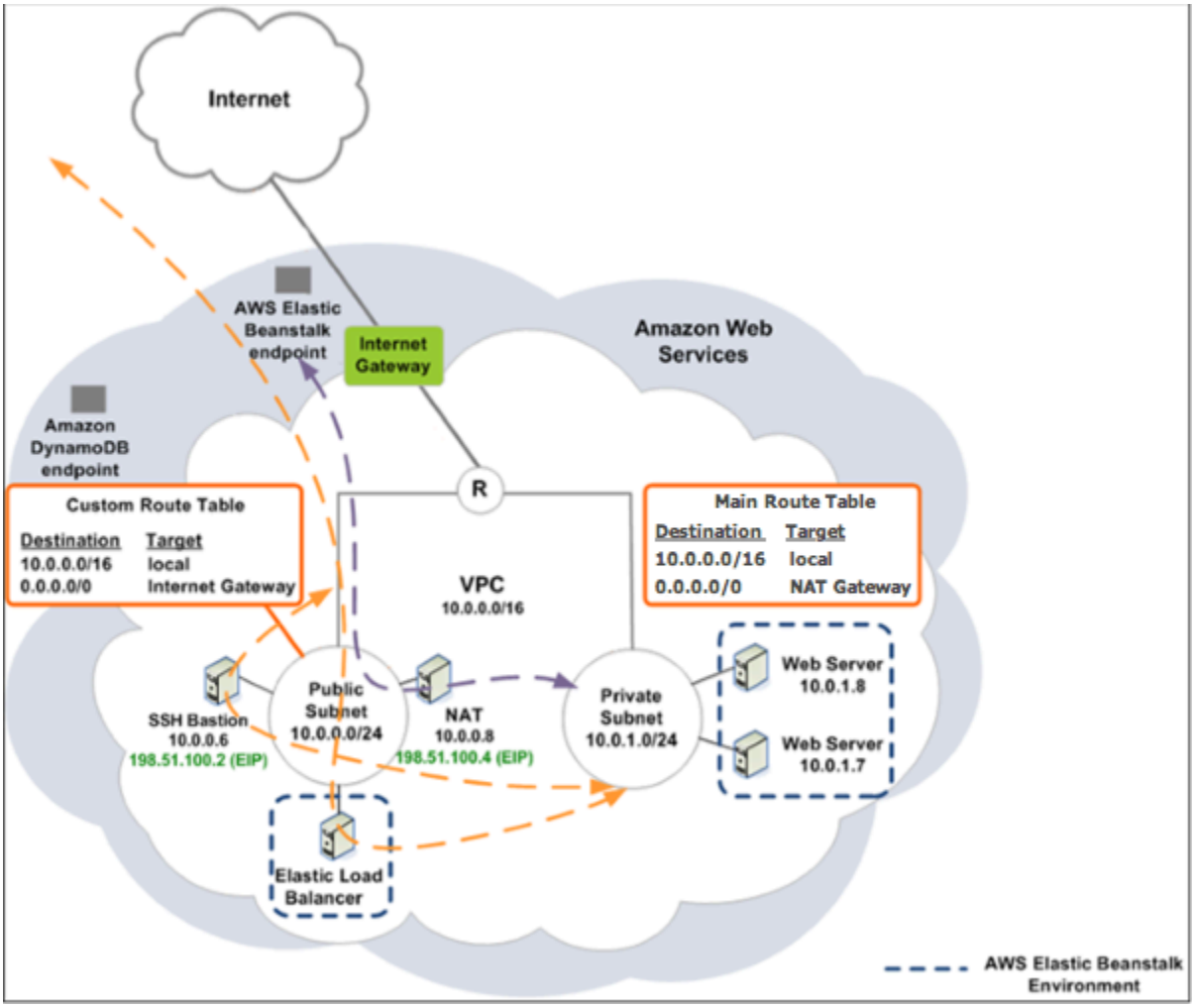
di atau setelah salah satu dari dua pembaruan ini: Pembaruan Linux 2020-02-28, Pemutakhiran Windows 2020-02-24.

> **Note**
>
> Alasan untuk membutuhkan versi platform yang diperbarui adalah bahwa versi lama memiliki masalah yang akan mencegah entri DNS yang dibuat oleh pilihan Aktifkan nama DNS dari bekerja dengan baik untuk Amazon SQS.

Contoh: Meluncurkan aplikasi Elastic Beanstalk di VPC dengan host bastion

Bagian ini menjelaskan cara menerapkan aplikasi Elastic Beanstalk di dalam VPC menggunakan host bastion dan mengapa Anda menerapkan topologi ini.

Jika EC2 instans Amazon Anda berada di dalam subnet pribadi, Anda tidak akan dapat menghubungkannya dari jarak jauh. Untuk menhubungkan ke instans Anda, Anda dapat mengatur server bastion di subnet publik untuk bertindak sebagai proksi. Misalnya, Anda dapat mengatur SSH port forwarders atau RDP gateway di subnet publik untuk proksi lalu lintas pergi ke server basis data Anda dari jaringan Anda sendiri. Bagian ini memberikan contoh bagaimana untuk membuat VPC dengan subnet privat dan publik. Instans terletak di dalam subnet privat, dan host bastion, NAT gateway, dan penyeimbang beban terletak di dalam subnet publik. Infrastruktur Anda akan terlihat serupa dengan diagram berikut.



Untuk men-deploy aplikasi Elastic Beanstalk di dalam VPC menggunakan host bastion, menyelesaikan langkah-langkah yang dijelaskan dalam subbagian berikut.

Langkah-langkah

- VPC dengan subnet publik dan privat
- Membuat dan mengonfigurasi grup keamanan host bastion
- Memperbarui grup keamanan instans
- Buat host bastion

VPC dengan subnet publik dan privat

Lengkapi semua prosedur di VPC publik/privat. Saat menerapkan aplikasi, Anda harus menentukan EC2 key pair Amazon untuk instance sehingga Anda dapat menghubungkannya dari jarak jauh.

Untuk informasi lebih lanjut tentang cara menentukan pasangan kunci instans, lihat EC2 Contoh Amazon untuk lingkungan Elastic Beanstalk Anda.

Membuat dan mengonfigurasi grup keamanan host bastion

Buat grup keamanan untuk host bastion, dan tambahkan aturan yang memungkinkan lalu lintas SSH masuk dari Internet, dan lalu lintas SSH keluar ke subnet pribadi yang berisi instans Amazon. EC2

Untuk membuat grup keamanan host bastion

1. Buka konsol VPC Amazon di. https://console.aws.amazon.com/vpc/
2. Pada panel navigasi, pilih Grup Keamanan.
3. Pilih Buat Grup Keamanan.
4. Di kotak dialog Buat Grup Keamanan, masukkan berikut dan pilih Ya, Buat.

   Tag nama (Opsional)

   Masukkan tag nama untuk grup keamanan.

   Nama grup

   Memasukkan nama grup keamanan.

   Deskripsi

   Memasukkan nama untuk grup keamanan baru dan deskripsinya.

   VPC

   Pilih VPC Anda.

   Grup keamanan di buat dan muncul di halaman Grup Keamanan. Perhatikan bahwa ia memiliki ID (misalnya, `sg-xxxxxxxx`). Anda mungkin harus mengaktifkan kolom ID Grup dengan mengklik Show/Sembunyikan di pojok kanan atas halaman.

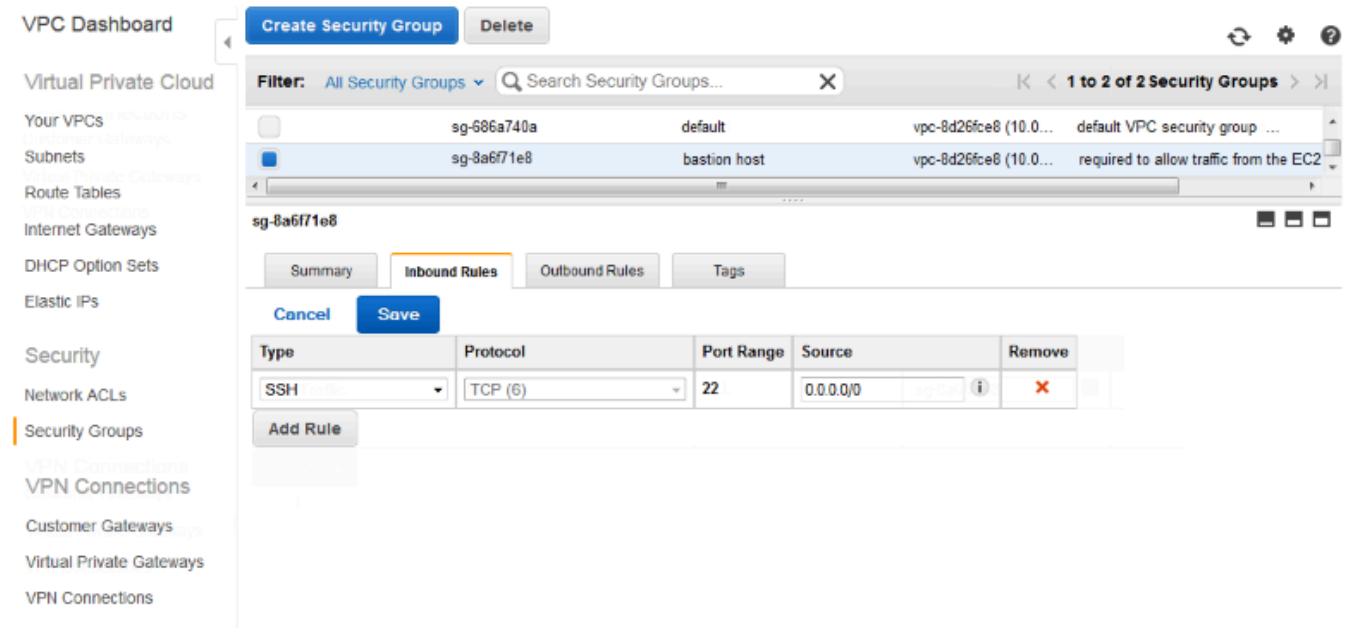
Untuk mengonfigurasi grup keamanan host bastion

1. Dalam daftar grup keamanan, pilih kotak centang untuk grup keamanan yang baru saja dibuat untuk host bastion Anda.
2. Pada tab Aturan Masuk, pilih Edit.
3. Jika diperlukan, pilih Tambahkan aturan lain.

4. Jika host bastion anda adalah sebuah instans Linux, di bawah Jenis, pilih SSH.

   Jika host bastion Anda adalah instans Windows, di bawah Jenis, pilih RDP.

5. Masukkan kisaran sumber CIDR yang diinginkan di bidang Sumber dan pilih Simpan.

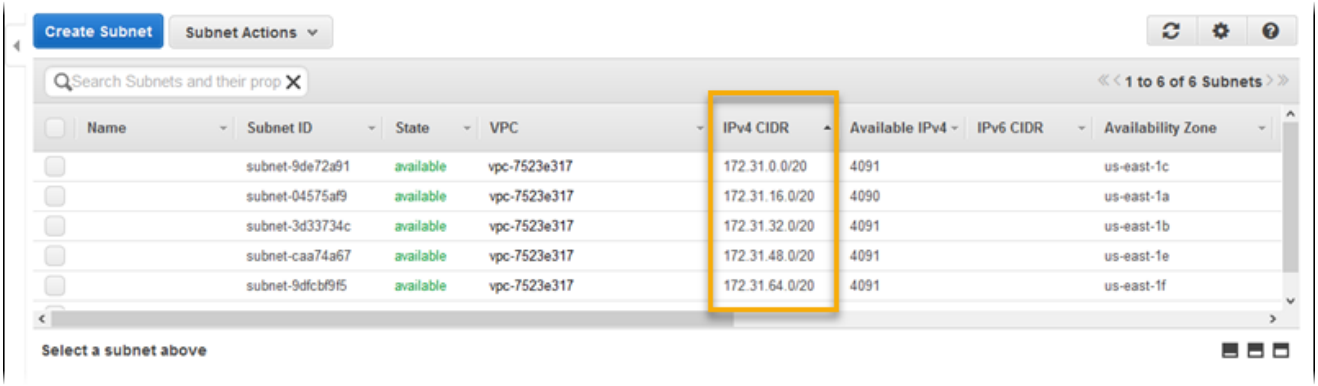


6. Pada tab Aturan Outbound, pilih Edit
7. Jika diperlukan, pilih Tambahkan aturan lain.
8. Di bawah Jenis, pilih jenis yang Anda tentukan untuk aturan inbound.
9. Di bidang Sumber, masukkan kisaran CIDR dari subnet host di subnet privat VPC.

   Untuk menemukannya:

   a. Buka konsol VPC Amazon di. https://console.aws.amazon.com/vpc/
   b. Di panel navigasi, pilih Pengguna.
   c. Perhatikan nilai di bawah IPv4 CIDR untuk setiap Availability Zone di mana Anda memiliki host yang Anda inginkan untuk dijembatani oleh host bastion.

> **Note**
>
> Jika Anda memiliki host di beberapa Availability Zone, membuat aturan keluar untuk masing-masing Availability Zone ini.



10. Pilih Simpan.

Memperbarui grup keamanan instans

Secara default, grup keamanan yang Anda buat untuk instans Anda tidak mengizinkan lalu lintas masuk. Sementara Elastic Beanstalk akan memodifikasi grup default untuk instans memungkinkan lalu lintas SSH, Anda harus mengubah grup keamanan instans kustom Anda untuk memungkinkan lalu lintas RDP jika instans Anda adalah instans Windows.

Untuk memperbarui grup keamanan instans untuk RDP

1. Dalam daftar grup keamanan, pilih kotak centang untuk grup keamanan instans.
2. Pada tab Inbound, pilih Edit.
3. Jika diperlukan, pilih Tambahkan aturan lain.
4. Masukkan nilai berikut, dan pilih Simpan.

   Jenis

   RDP

   Protokol

   TCP

   Rentang Port

   3389

Sumber

Masukkan ID dari grup keamanan host bastion (misalnya, `sg-8a6f71e8`) dan pilih Simpan.

Buat host bastion

Untuk membuat host bastion, Anda meluncurkan EC2 instans Amazon di subnet publik Anda yang akan bertindak sebagai host bastion.

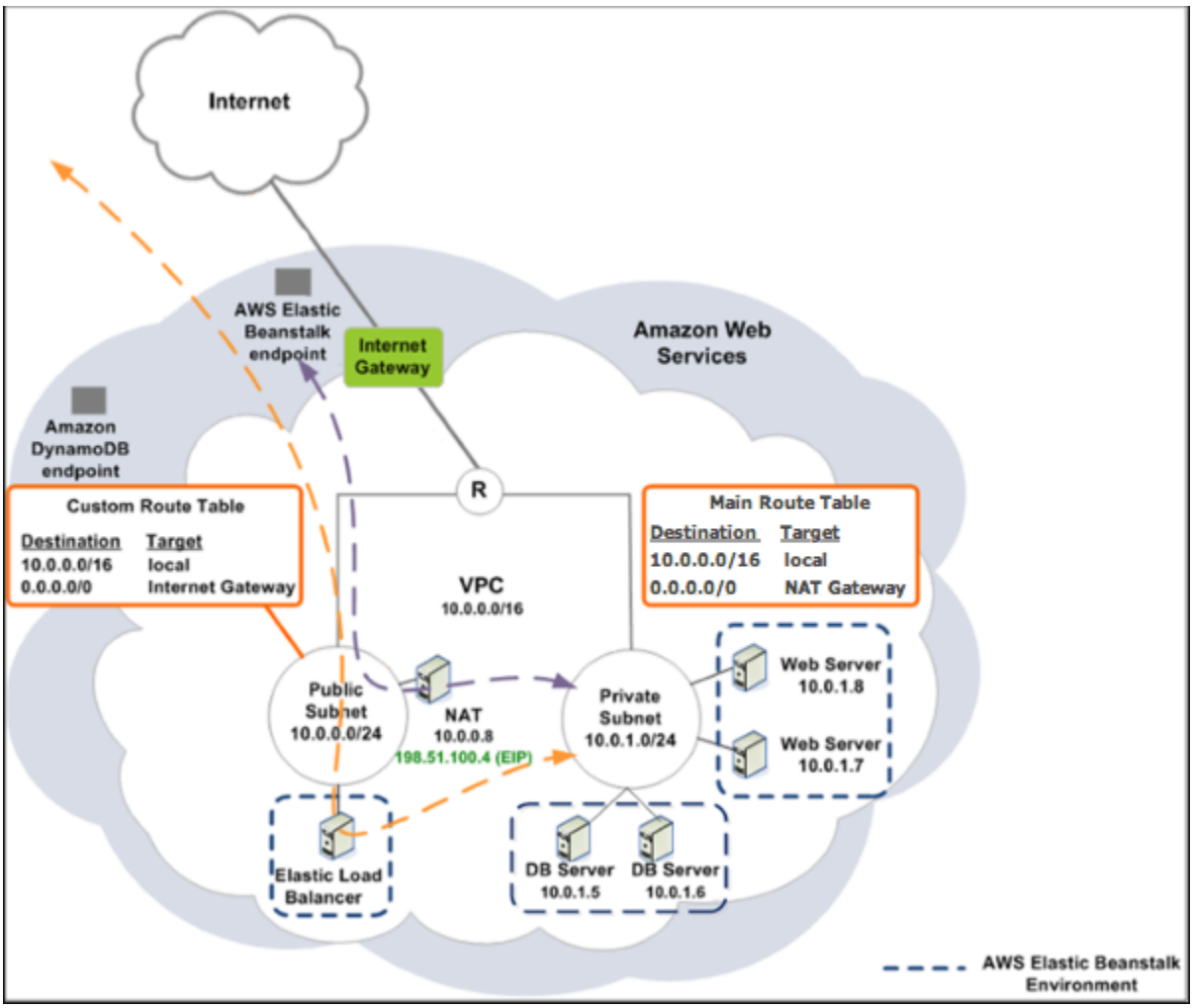
Untuk informasi selengkapnya tentang menyiapkan host bastion untuk instance Windows di subnet pribadi, lihat Mengontrol Akses Jaringan ke EC2 Instans Menggunakan Server Bastion.

Untuk informasi lebih lanjut tentang pengaturan host benteng untuk instans Linux di subnet privat, lihat Hubungkan Secara Aman ke Instans Linux yang Berjalan di Amazon VPC Privat .

Contoh: Meluncurkan Elastic Beanstalk di VPC dengan Amazon RDS

Bagian ini memandu Anda melalui tugas-tugas untuk men-deploy aplikasi Elastic Beanstalk dengan Amazon RDS di VPC menggunakan gateway NAT.

Infrastruktur Anda akan terlihat serupa dengan diagram berikut.



> **Note**
>
> Jika Anda belum menggunakan instans DB dengan aplikasi Anda sebelumnya, coba menambahkan satu ke lingkungan percobaan, dan menghubungkan ke contoh DB eksternal sebelum menambahkan konfigurasi VPC ke mix.

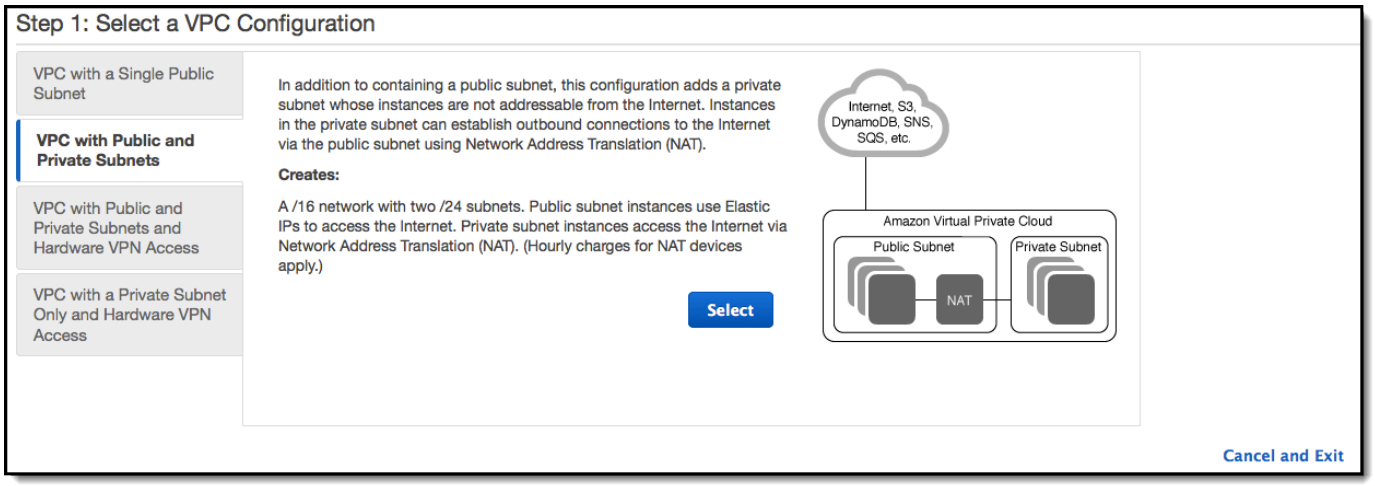
VPC dengan subnet publik dan privat

Anda bisa menggunakan konsol Amazon VPC untuk membuat VPC.

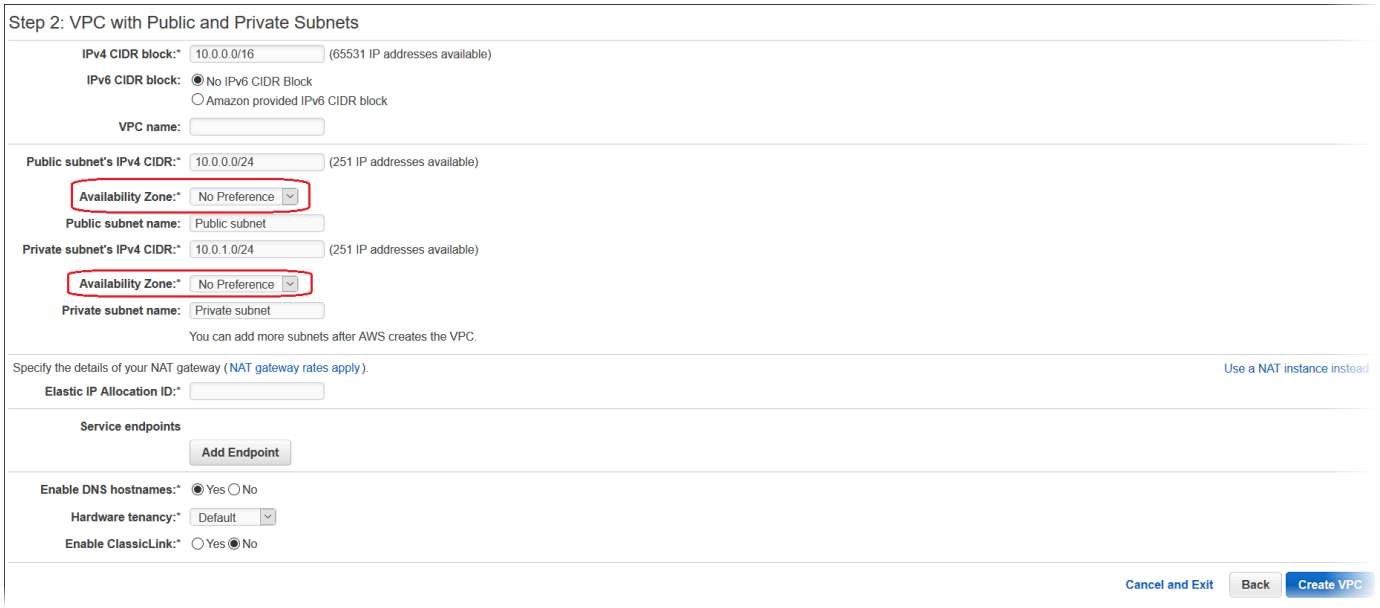
Untuk membuat VPC

1. Masuk ke Amazon konsol Amazon VPC.
2. Di panel navigasi, pilih VPC Dasbor. Kemudian pilih Buat VPC.

3. Pilih VPC dengan Publik dan Subnet Privat, dan kemudian, pilih Pilih.



4. Penyeimbang beban Elastic Load Balancing dan EC2 instans Amazon Anda harus berada di Availability Zone yang sama sehingga mereka dapat berkomunikasi satu sama lain. Pilih Availability Zone yang sama dari setiap daftar Availability Zone.



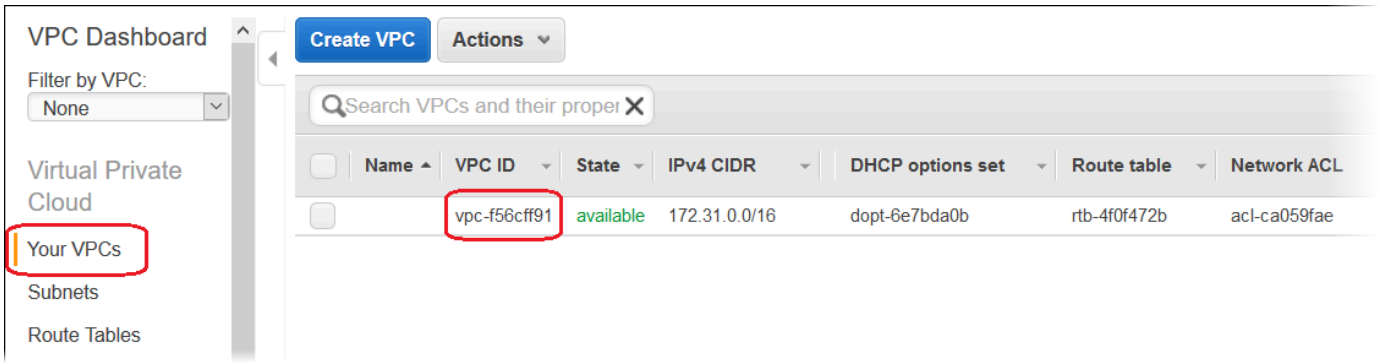
5. Pilih alamat IP Elastic untuk gateway NAT Anda.

6. Pilih Buat VPC.

   Wizard mulai membuat VPC, subnet, dan gateway internet. Hal ini juga memperbarui tabel rute utama dan menciptakan tabel rute kustom. Akhirnya, wizard menciptakan gateway NAT di subnet publik.

> **Note**
>
> Anda dapat memilih untuk meluncurkan instans NAT di subnet publik bukan gateway NAT. Untuk informasi selengkapnya, lihat Skenario 2: VPC dengan subnet publik dan privat (NAT) di Panduan Pengguna Amazon VPC.

7. Setelah VPC berhasil dibuat, Anda mendapatkan ID VPC. Anda memerlukan nilai ini untuk langkah berikutnya. Untuk melihat ID VPC Anda, pilih Anda VPCs di panel kiri konsol VPC Amazon.

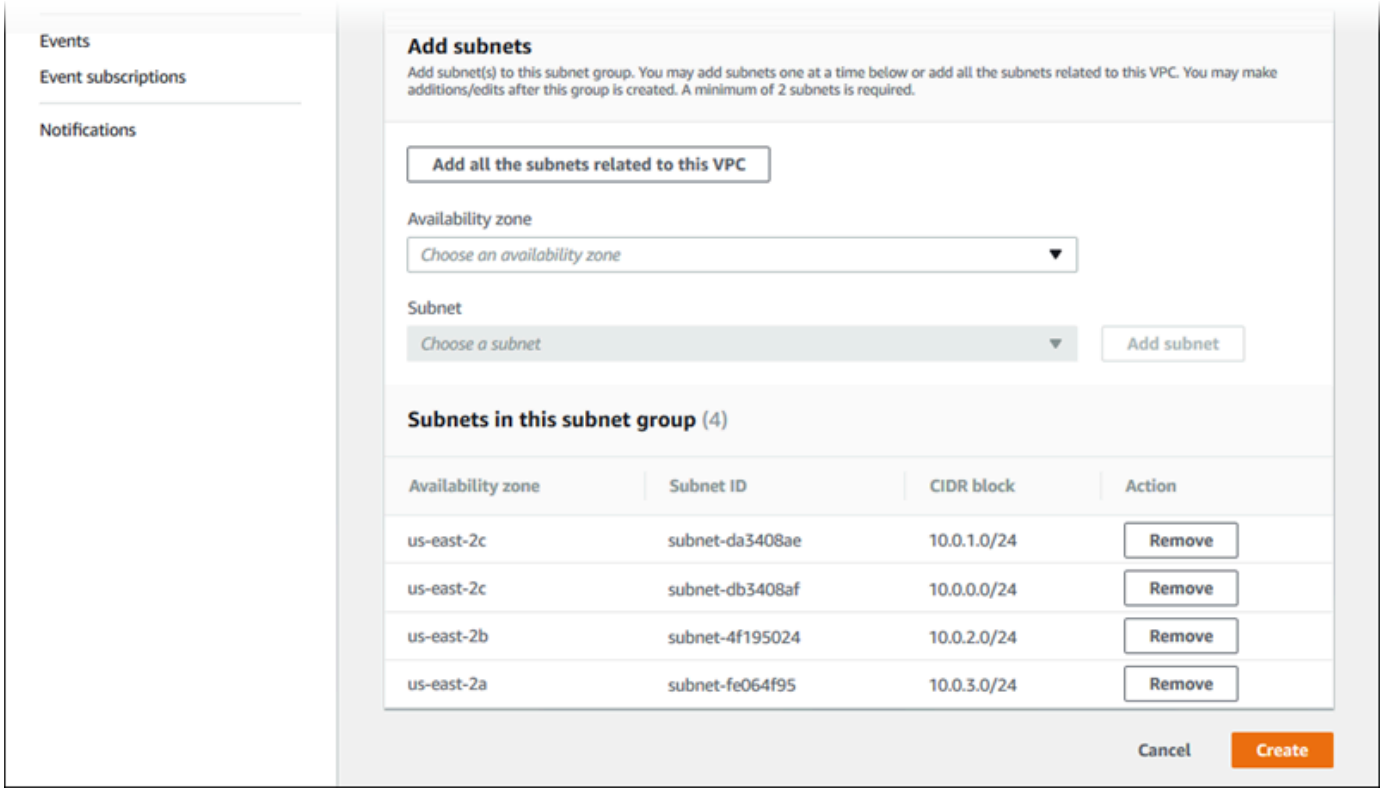


Buat grup subnet DB

Grup subnet DB adalah kumpulan subnet (biasanya privat) yang Anda buat untuk VPC dan yang kemudian Anda tetapkan untuk instans backend RDS DB Anda. Setiap grup subnet DB harus memiliki setidaknya satu subnet untuk setiap Availability Zone di Wilayah tertentu AWS . Untuk mempelajari lebih lanjut, lihat Membuat Subnet di VPC Anda.

Buat grup subnet DB

1. Buka konsol Amazon RDS.
2. Di panel navigasi, pilih Grup Subnet.
3. Pilih Buat Grup Subnet DB.
4. Pilih Nama, dan ketik nama grup subnet DB Anda.
5. Pilih Deskripsi, dan kemudian menjelaskan grup subnet DB Anda.
6. Untuk VPC, pilih ID VPC yang Anda buat.
7. Di Menambahkan subnet, pilih Tambahkan semua subnet yang terkait dengan VPC ini.



8. Setelah selesai, pilih Buat.

   Grup subnet DB baru Anda muncul dalam daftar grup subnet di konsol Amazon RDS. Anda dapat memilihnya untuk melihat detail, seperti semua subnet yang terkait dengan grup ini, di panel detail di bagian bawah halaman.

Men-deploy ke Elastic Beanstalk

Setelah Anda mengatur VPC Anda, Anda dapat membuat lingkungan Anda di dalamnya dan men-deploy aplikasi Anda ke Elastic Beanstalk. Anda dapat melakukan ini menggunakan konsol Elastic Beanstalk, atau Anda dapat AWS menggunakan toolkit AWS CLI,, EB CLI, atau Elastic Beanstalk API. Jika Anda menggunakan konsol Elastic Beanstalk, Anda hanya perlu mengunggah `.war` atau file `.zip` dan pilih pengaturan VPC di dalam wizard. Elastic Beanstalk kemudian menciptakan lingkungan Anda di dalam VPC Anda dan men-deploy aplikasi Anda. Atau, Anda dapat menggunakan AWS toolkit, EB CLI AWS CLI, atau Elastic Beanstalk API untuk menyebarkan aplikasi Anda. Untuk melakukan ini, Anda perlu menentukan pengaturan opsi VPC Anda dalam file konfigurasi dan men-deploy file ini dengan paket sumber Anda. Topik ini memberikan petunjuk untuk kedua metode tersebut.

Men-deploy dengan konsol Elastic Beanstalk

Konsol Elastic Beanstalk memandu Anda menciptakan lingkungan baru Anda di dalam VPC Anda. Anda harus menyediakan file `.war`(untuk aplikasi Java) atau file `.zip` (untuk semua aplikasi lainnya). Pada halaman Konfigurasi VPC dari wizard lingkungan Elastic Beanstalk, Anda harus membuat pilihan berikut:

VPC

Pilih VPC Anda.

Grup keamanan VPC

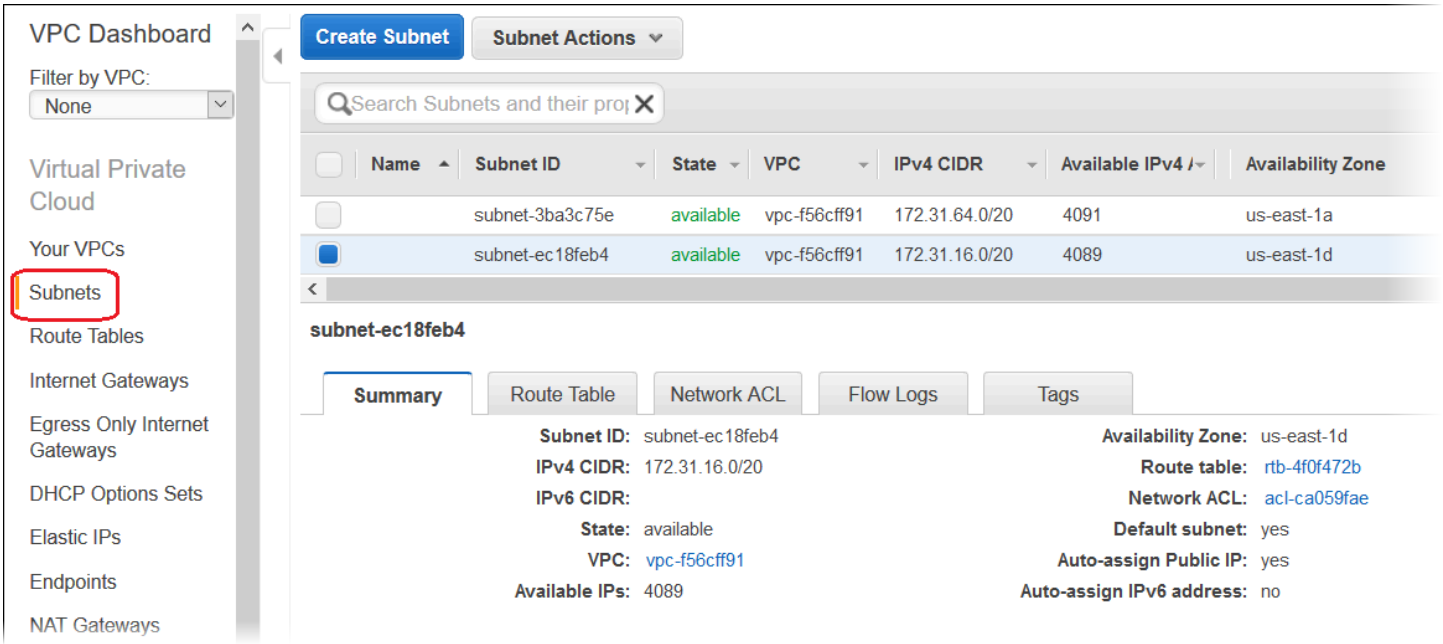
Pilih grup keamanan instans yang Anda buat di atas.

Visibilitas ELB

Pilih `External` jika penyeimbang beban Anda harus tersedia untuk umum, atau pilih `Internal` jika penyeimbang beban harus tersedia hanya dalam VPC Anda.

Pilih subnet untuk penyeimbang beban dan EC2 instance Anda. Pastikan Anda memilih subnet publik untuk penyeimbang beban, dan subnet pribadi untuk instans Amazon Anda. EC2 Secara default, pembuatan VPC menciptakan subnet publik di `10.0.0.0/24`dan subnet pribadi di `10.0.1.0/24`.

Anda dapat melihat subnet Anda IDs dengan memilih Subnet di konsol VPC Amazon.



Menerapkan dengan AWS toolkit, EB CLI, atau API AWS CLI

Saat menerapkan aplikasi Anda ke Elastic Beanstalk menggunakan AWS toolkit, EB AWS CLI CLI, atau API, Anda dapat menentukan pengaturan opsi VPC dalam file dan menerapkannya dengan bundel sumber Anda. Lihat Penyesuaian lingkungan lanjutan dengan file konfigurasi (`.ebextensions`) untuk informasi selengkapnya.

Saat Anda memperbarui pengaturan opsi, Anda harus menentukan setidaknya yang berikut ini:

- VPCId—Berisi ID VPC.
- Subnets—Berisi ID dari subnet grup Auto Scaling. Dalam contoh ini, ini adalah ID dari subnet privat.
- ELBSubnets—Berisi ID subnet untuk penyeimbang beban. Dalam contoh ini, ini adalah ID dari subnet publik.
- SecurityGroups—Berisi ID grup keamanan.
- DBSubnets—Berisi ID subnet DB.

> **Note**
>
> Saat menggunakan subnet DB, Anda perlu membuat subnet tambahan di VPC Anda untuk mencakup semua Availability Zone di Region. AWS

Opsional, Anda juga dapat menentukan informasi berikut:

- ELBScheme— Tentukan `internal` untuk membuat penyeimbang beban internal di dalam VPC Anda sehingga aplikasi Elastic Beanstalk Anda tidak dapat diakses dari luar VPC Anda.

Berikut ini adalah contoh pengaturan opsi yang dapat Anda gunakan ketika men-deploy aplikasi Elastic Beanstalk Anda di dalam VPC. Untuk informasi lebih lanjut tentang pengaturan opsi VPC (termasuk contoh untuk cara menentukan mereka, nilai default, dan nilai-nilai yang valid), lihat tabel namespace aws:ec2:vpc di Opsi konfigurasi.

```
option_settings:
  - namespace: aws:autoscaling:launchconfiguration
    option_name: EC2KeyName
    value: ec2keypair
```

```
  - namespace: aws:ec2:vpc
    option_name: VPCId
    value: vpc-170647c

  - namespace: aws:ec2:vpc
    option_name: Subnets
    value: subnet-4f195024

  - namespace: aws:ec2:vpc
    option_name: ELBSubnets
    value: subnet-fe064f95

  - namespace: aws:ec2:vpc
    option_name: DBSubnets
    value: subnet-fg148g78

  - namespace: aws:autoscaling:launchconfiguration
    option_name: InstanceType
    value: m1.small

  - namespace: aws:autoscaling:launchconfiguration
    option_name: SecurityGroups
    value: sg-7f1ef110
```

> **Note**
>
> Saat menggunakan subnet DB, pastikan Anda memiliki subnet di VPC Anda untuk mencakup semua Availability Zone di Region. AWS

Menggunakan Elastic Beanstalk dengan VPC endpoint

Topik ini menjelaskan manfaat yang dapat ditawarkan oleh titik akhir VPC untuk aplikasi Elastic Beanstalk Anda. Ini juga memberikan instruksi untuk membuat titik akhir VPC antarmuka ke layanan Elastic Beanstalk.

Titik akhir VPC memungkinkan Anda menghubungkan VPC Anda secara pribadi ke layanan yang didukung AWS dan layanan titik akhir VPC yang didukung oleh AWS PrivateLink, tanpa memerlukan gateway internet, perangkat NAT, koneksi VPN, atau koneksi. AWS Direct Connect

Instans di VPC Anda tidak memerlukan alamat IP publik untuk berkomunikasi dengan sumber daya dalam layanan. Lalu lintas antara VPC Anda dan layanan lainnya tidak meninggalkan jaringan

Amazon. Untuk informasi lebih lanjut tentang VPC endpoint, lihat VPC Endpoints dalam Panduan Pengguna Amazon VPC.

AWS Elastic Beanstalk mendukung AWS PrivateLink, yang menyediakan konektivitas pribadi ke layanan Elastic Beanstalk dan menghilangkan paparan lalu lintas ke internet publik. Untuk mengaktifkan aplikasi Anda mengirim permintaan ke Elastic AWS PrivateLink Beanstalk menggunakan, Anda mengonfigurasi jenis titik akhir VPC yang dikenal sebagai titik akhir VPC antarmuka (titik akhir antarmuka). Untuk informasi selengkapnya, lihat VPC Endpoint Antarmuka (AWS PrivateLink) dalam Panduan Pengguna Amazon VPC.

> **Note**
>
> Elastic AWS PrivateLink Beanstalk mendukung dan antarmuka titik akhir VPC dalam jumlah terbatas. Wilayah AWS Kami bekerja untuk memperluas dukungan ke lebih banyak Wilayah AWS dalam waktu dekat.

IPv6 dukungan

Elastic Beanstalk mendukung lalu lintas masuk berulang-ulang. IPv4 IPv6 Bagian ini menjelaskan titik akhir publik yang mendukung IPV6 dan juga menjelaskan cara mengonfigurasi titik akhir VPC Elastic Beanstalk Anda untuk mendukung lalu lintas dual-stack.

Untuk informasi umum selengkapnya IPv6, lihat AWS layanan yang mendukung IPv6 di Panduan Pengguna Amazon VPC dan AWS IPv6 whitepaper aktif. AWS

Titik akhir publik

Layanan Elastic Beanstalk memiliki dua set titik akhir yang terdiri dari titik akhir yang lebih tua dan titik akhir yang lebih IPv4 baru dengan kemampuan dual-stack. Kedua set titik akhir mengikuti standar AWS penamaan:

- IPv4 endpoint menggunakan format domain `amazonaws.com` — untuk endpoint layanan umum: `elasticbeanstalk.region.amazonaws.com`
- Titik akhir tumpukan ganda menggunakan format domain `api.aws` — untuk titik akhir layanan umum:: `elasticbeanstalk.region.api.aws`

Titik akhir untuk kesehatan layanan dan FIPS memiliki nama host yang berbeda, tetapi mereka mengikuti pola nama domain yang sama. Untuk daftar titik akhir, lihat titik akhir layanan Elastic Beanstalk di. Referensi Umum Amazon Web Services

Permintaan untuk Elastic Beanstalk

Saat Anda mengirim permintaan ke layanan Elastic Beanstalk AWS CLIdengan atau SDK, Anda dapat IPv4 menentukan AWS titik akhir atau titik akhir tumpukan ganda. AWS SDK AWS CLI dan menggunakan titik akhir IPv4 -only secara default jika URL titik akhir tidak ditentukan.

Contoh berikut menunjukkan AWS CLI pengiriman permintaan ke endpoint dual-stack:

Example

```
aws elasticbeanstalk list-available-solution-stacks \
    --endpoint-url "https://elasticbeanstalk.us-east-1.api.aws"
```

Contoh berikut menunjukkan SDK AWS Python mengirim permintaan ke endpoint dual-stack:

Example

```
import boto3

dual_stack_eb_client = boto3.client(
    service_name='elasticbeanstalk',
    region_name='us-east-1',
    endpoint_url='https://elasticbeanstalk.us-east-1.api.aws';
)

print(dual_stack_eb_client.list_available_solution_stacks())
```

Titik akhir VPC untuk dual-stack IPs

Untuk mengonfigurasi titik akhir VPC Elastic Beanstalk Anda untuk mendukung lalu lintas dual-stack, tentukan dualstack untuk parameter tipe alamat IP titik akhir VPC. Anda dapat menentukan bidang ini melalui AWS CLI, AWS SDK, atau AWS PrivateLink konsol. Untuk petunjuk melakukannya di AWS PrivateLink konsol, lihat Membuat titik akhir VPC di Panduan.AWS PrivateLink

> **Note**
>
> Anda harus menentukan jenis alamat IP dari titik akhir VPC sebagai salah satu atau IPv4dualstack. Saat ini titik akhir VPC Elastic Beanstalk tidak mendukung jenis alamat

IPv6IP, yang akan menunjukkan dukungan -only. IPv6 Opsi dualstack memungkinkan untuk protokol IPv6 internet IPv4 dan protokol.

Contoh berikut menunjukkan cara membuat endpoint VPC dual-stack dengan: AWS CLI

Example

```
aws ec2 create-vpc-endpoint \
    --vpc-id "vpc-example"
    --service-name "com.amazonaws.us-east-1.elasticbeanstalk"
    --ip-address-type "dualstack"
```

Menyiapkan VPC endpoint untuk Elastic Beanstalk

Untuk membuat antarmuka VPC endpoint untuk layanan Elastic Beanstalk di VPC Anda, ikuti prosedur Membuat Antarmuka Titik Akhir.

- Untuk Nama Layanan, pilih com.amazonaws. ***region***.elasticbeanstalk.
- Untuk jenis alamat IP, pilih salah satu IPv4atau Dualstack. Saat ini titik akhir VPC Elastic Beanstalk tidak mendukung jenis alamat IPv6IP, yang akan menunjukkan dukungan -only. IPv6 Opsi Dualstack memungkinkan untuk protokol IPv4 dan IPv6 internet.

Jika VPC Anda dikonfigurasi dengan akses internet publik, aplikasi Anda masih dapat mengakses Elastic Beanstalk melalui internet menggunakan titik akhir atau publik. `elasticbeanstalk.region.amazonaws.com` `elasticbeanstalk.region.api.aws` Anda dapat mencegah hal ini dengan memastikannya Aktifkan nama DNS diaktifkan selama pembuatan endpoint (true secara default). Ini menambahkan entri DNS di VPC Anda yang memetakan titik akhir layanan publik untuk antarmuka VPC endpoint.

Menyiapkan VPC endpoint untuk kondisi yang ditingkatkan

Jika Anda mengaktifkan pelaporan kesehatan yang disempurnakan untuk lingkungan Anda, Anda dapat mengonfigurasi informasi kesehatan yang disempurnakan untuk dikirim AWS PrivateLink juga. Informasi kondisi yang ditingkatkan dikirim oleh daemon `healthd`, komponen Elastic Beanstalk pada instans lingkungan Anda, ke layanan kondisi yang ditingkatkan Elastic Beanstalk terpisah. Untuk membuat VPC endpoint antarmuka untuk layanan ini di VPC Anda, ikuti prosedur Membuat Antarmuka Titik Akhir.

- Untuk Nama Layanan, pilih com.amazonaws. ***region***.elasticbeanstalk-kesehatan.
- Untuk jenis alamat IP, pilih salah satu IPv4atau Dualstack. Saat ini titik akhir VPC Elastic Beanstalk tidak mendukung jenis alamat IPv6IP, yang akan menunjukkan dukungan -only. IPv6 Opsi Dualstack memungkinkan untuk protokol IPv4 dan IPv6 internet.

> **Important**
>
> `healthd`Daemon mengirimkan informasi kesehatan yang ditingkatkan ke titik akhir `elasticbeanstalk-health.`*`region`*`.amazonaws.com` publik atau. `elasticbeanstalk-health.`*`region`*`.api.aws` Jika VPC Anda dikonfigurasi dengan akses internet publik, dan Aktifkan nama DNS dinonaktifkan untuk VPC endpoint, informasi kondisi yang ditingkatkan melalui internet publik. Ini mungkin bukan niat Anda ketika Anda mengatur VPC endpoint kondisi yang ditingkatkan. Pastikan bahwa Aktifkan nama DNS diaktifkan (benar secara default).

Menggunakan VPC endpoint di VPC privat

Sebuah VPC pribadi, atau subnet pribadi di VPC, tidak memiliki akses internet publik. Anda mungkin ingin menjalankan lingkungan Elastic Beanstalk Anda dalam VPC pribadi dan konfigurasikan VPC endpoint antarmuka untuk keamanan yang ditingkatkan. Dalam hal ini, perhatikan bahwa lingkungan Anda mungkin mencoba untuk terhubung ke internet karena alasan lain selain menghubungi layanan Elastic Beanstalk. Untuk mempelajari lebih lanjut tentang menjalankan lingkungan di VPC pribadi, lihat the section called “Menjalankan lingkungan Elastic Beanstalk di VPC pribadi”.

Menggunakan kebijakan titik akhir untuk mengontrol akses dengan VPC endpoint

Topik ini menjelaskan bagaimana Anda dapat melampirkan kebijakan ke titik akhir VPC untuk mengontrol akses ke aplikasi Anda (layanan Anda) dan lingkungan Elastic Beanstalk Anda.

Kebijakan endpoint adalah kebijakan sumber daya AWS Identity and Access Management (IAM) yang mengontrol akses dari titik akhir ke layanan yang ditentukan. Kebijakan titik akhir khusus untuk titik akhir. Ini terpisah dari pengguna atau instans kebijakan IAM yang mungkin dimiliki lingkungan Anda dan tidak menimpa atau menggantikannya.

Secara default, VPC endpoint memungkinkan akses penuh ke layanan yang terkait. Saat membuat atau memodifikasi titik akhir, Anda dapat melampirkan kebijakan titik akhir untuk mengontrol akses ke

sumber daya tertentu yang terkait dengan layanan. Untuk detail tentang pembuatan dan penggunaan kebijakan titik akhir VPC, lihat Mengontrol akses ke titik akhir VPC menggunakan kebijakan titik akhir di Panduan.AWS PrivateLink

> **Note**
>
> Saat membuat kebijakan titik akhir yang membatasi, Anda mungkin perlu menambahkan izin khusus ke sumber daya yang diperlukan, sehingga akses ke sumber daya ini tidak diblokir oleh kebijakan titik akhir. Melakukannya memastikan bahwa lingkungan Anda terus menyebar dan berfungsi dengan baik.

Contoh berikut menyangkal semua pengguna izin untuk mengakhiri lingkungan melalui VPC endpoint, dan memungkinkan akses penuh ke semua tindakan lainnya.

```
{
    "Statement": [
        {
            "Action": "*",
            "Effect": "Allow",
            "Resource": "*",
            "Principal": "*"
        },
        {
            "Action": "elasticbeanstalk:TerminateEnvironment",
            "Effect": "Deny",
            "Resource": "*",
            "Principal": "*"
        }
    ]
}
```

Izin bucket Amazon S3 yang diperlukan untuk kebijakan titik akhir VPC yang membatasi

Jika menambahkan batasan pada kebijakan titik akhir VPC, Anda harus menyertakan izin bucket Amazon S3 tertentu untuk memastikan bahwa lingkungan Anda terus diterapkan dan berfungsi dengan benar. Bagian ini menjelaskan bucket S3 yang diperlukan dan menyertakan contoh kebijakan.

Topik

- S3 Bucket yang menyimpan aset untuk mengelola platform lingkungan
- Bucket S3 dimiliki oleh AWS CloudFormation
- S3 Bucket yang dimiliki oleh akun pelanggan untuk menyimpan kode sumber dan item lainnya
- Bucket S3 yang dimiliki oleh akun pelanggan untuk mendukung otentikasi registri Docker
- Memperbarui kebijakan titik akhir VPC Anda

S3 Bucket yang menyimpan aset untuk mengelola platform lingkungan

Layanan Elastic Beanstalk memiliki bucket S3 yang menyimpan aset yang terkait dengan tumpukan solusi (versi platform). Aset ini mencakup file konfigurasi, aplikasi sampel, dan jenis instance yang tersedia. Saat Elastic Beanstalk membuat dan mengelola lingkungan Anda, Elastic Beanstalk akan mengambil informasi yang diperlukan untuk versi platform tertentu dari bucket aset untuk setiap yang sesuai. Wilayah AWS

S3 Bucket ARN

`arn:aws:s3:::elasticbeanstalk-samples-`*`region`*

Amazon Linux 2 dan yang lebih baru

- `arn:aws:s3:::elasticbeanstalk-platform-assets-`*`region`*

> **Note**
>
> Nama bucket mengikuti konvensi yang berbeda untuk wilayah BJS. String public-beta-cn-north-1 digunakan sebagai pengganti*`region`*. Misalnya, `arn:aws:s3:::elasticbeanstalk-platform-assets-public-beta-cn-north-1`.

Windows Server, Amazon Linux (AMI), Amazon Linux 2 dan yang lebih baru

- `arn:aws:s3:::elasticbeanstalk-env-resources-`*`region`*
- `arn:aws:s3:::elasticbeanstalk-`*`region`*

> **Note**
>
> Nama bucket untuk bucket platform-assets dan env-resources mengikuti konvensi yang berbeda di beberapa wilayah. Lihat bagian pola penamaan bucket khusus wilayah di bawah ini untuk detailnya.

Pola ARN bucket khusus wilayah

Wilayah	Jenis Bucket	Pola ARN Bucket
me-central-1	platform-aset	`arn:aws:s3:::elasticbeanstalk-platform-assets-me-central-1-f08b818c`
	env-sumber daya	`arn:aws:s3:::elasticbeanstalk-env-resources-me-central-1-f08b818c`

Operasi

GetObject

Contoh kebijakan titik akhir VPC

Contoh berikut menggambarkan cara menyediakan akses ke bucket S3 yang diperlukan untuk operasi Elastic Beanstalk di Wilayah Timur AS (Ohio) (us-east-2). Contoh ini mencantumkan semua bucket untuk platform Amazon Linux dan Windows Server. Perbarui kebijakan Anda untuk hanya menyertakan bucket yang berlaku untuk sistem operasi lingkungan Anda.

Contoh kebijakan

> **Important**
>
> Kami menyarankan Anda untuk menghindari menggunakan karakter wildcard (*) di tempat Wilayah tertentu dalam kebijakan ini. Misalnya, gunakan `arn:aws:s3:::cloudformation-waitcondition-us-east-2/*` dan jangan gunakan `arn:aws:s3:::cloudformation-waitcondition-*/*`. Menggunakan wildcard dapat menyediakan akses ke bucket S3 yang tidak ingin Anda berikan akses. Jika Anda ingin

menggunakan kebijakan untuk lebih dari satu Wilayah, sebaiknya ulangi `Statement` blok pertama untuk setiap Wilayah.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowRequestsToAWSResources",
            "Effect": "Allow",
            "Principal": {"AWS": "*"},
            "Action": ["s3:GetObject"],
            "Resource": [
                "arn:aws:s3:::elasticbeanstalk-platform-assets-us-east-2/*",
                "arn:aws:s3:::elasticbeanstalk-env-resources-us-east-2/*",
                "arn:aws:s3:::elasticbeanstalk-env-resources-us-east-2/*",
                "arn:aws:s3:::elasticbeanstalk-samples-us-east-2/*"
            ]
         }
    ]
}
```

Bucket S3 dimiliki oleh AWS CloudFormation

Elastic AWS CloudFormation Beanstalk digunakan untuk menciptakan sumber daya bagi lingkungan Anda. CloudFormation memiliki ember S3 di masing-masing Wilayah AWS untuk memantau respons terhadap kondisi menunggu.

Layanan seperti Elastic Beanstalk CloudFormation berkomunikasi dengan mengirimkan permintaan ke URL Amazon S3 yang telah ditetapkan sebelumnya untuk bucket S3 yang dimiliki. CloudFormation CloudFormation membuat URL Amazon S3 yang telah ditetapkan sebelumnya menggunakan prinsip layanan`cloudformation.amazonaws.com`.

Untuk informasi selengkapnya, lihat Pertimbangan untuk titik akhir CloudFormation VPC di AWS CloudFormation Panduan Pengguna. Untuk mempelajari selengkapnya tentang presigned URLs, lihat Bekerja dengan presigned URLs di Panduan Pengguna Amazon S3.

S3 Bucket ARN

- `arn:aws:s3:::cloudformation-waitcondition-region`

Ketika menggunakan kondisi menunggu, nama region mengandung tanda hubung. Misalnya, us-west-2.

- `arn:aws:s3:::cloudformation-custom-resource-response-region`

  Saat menggunakan sumber daya kustom, nama region tidak mengandung tanda hubung. Misalnya, uswest2.

Operasi

GetObject

Contoh kebijakan titik akhir VPC

Contoh berikut menggambarkan cara menyediakan akses ke bucket S3 yang diperlukan untuk operasi Elastic Beanstalk di Wilayah Timur AS (Ohio) (us-east-2).

Contoh kebijakan

> **Important**
>
> Kami menyarankan Anda untuk menghindari menggunakan karakter wildcard (*) di tempat Wilayah tertentu dalam kebijakan ini. Misalnya, gunakan `arn:aws:s3:::cloudformation-waitcondition-us-east-2/*` dan jangan gunakan `arn:aws:s3:::cloudformation-waitcondition-*/*`. Menggunakan wildcard dapat menyediakan akses ke bucket S3 yang tidak ingin Anda berikan akses. Jika Anda ingin menggunakan kebijakan untuk lebih dari satu Wilayah, sebaiknya ulangi `Statement` blok pertama untuk setiap Wilayah.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowRequestsToCloudFormation",
            "Effect": "Allow",
            "Principal": {"AWS": "*"},
            "Action": ["s3:GetObject"],
            "Resource": [
```

```
                "arn:aws:s3:::cloudformation-waitcondition-us-east-2/*",
                "arn:aws:s3:::cloudformation-custom-resource-response-us-east-2/*"
            ]
        }
    ]
}
```

S3 Bucket yang dimiliki oleh akun pelanggan untuk menyimpan kode sumber dan item lainnya

Bucket ini dimiliki oleh akun AWS pelanggan yang memiliki lingkungan. Ini menyimpan sumber daya yang spesifik untuk lingkungan Anda, seperti kode sumber dan log yang diminta.

S3 Bucket ARN

`arn:aws:s3:::elasticbeanstalk-region-account-id`

Operasi

- GetObject
- GetObjectAcl
- PutObject
- PutObjectAcl
- ListBucket

Contoh kebijakan titik akhir VPC

Contoh berikut menggambarkan cara menyediakan akses ke bucket S3 yang diperlukan untuk operasi Elastic Beanstalk di Wilayah Timur AS (Ohio) (us-east-2) dan untuk contoh id akun 123456789012. AWS

Contoh kebijakan

> **Important**
>
> Kami menyarankan Anda untuk menghindari menggunakan karakter wildcard (*) di tempat Wilayah tertentu dalam kebijakan ini. Misalnya, gunakan `arn:aws:s3:::cloudformation-waitcondition-us-east-2/*` dan jangan gunakan `arn:aws:s3:::cloudformation-waitcondition-*/*`. Menggunakan wildcard dapat

menyediakan akses ke bucket S3 yang tidak ingin Anda berikan akses. Jika Anda ingin menggunakan kebijakan untuk lebih dari satu Wilayah, sebaiknya ulangi `Statement` blok pertama untuk setiap Wilayah.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowRequestsToCustomerItems",
            "Effect": "Allow",
            "Principal": {"AWS": "*"},
            "Action": ["GetObject",
                       "GetObjectAcl",
                       "PutObject",
                       "PutObjectAcl",
                       "ListBucket"
                       ],
            "Resource": [
                "arn:aws:s3:::elasticbeanstalk-us-east-2-123456789012/*"
            ]
         }
    ]
}
```

Bucket S3 yang dimiliki oleh akun pelanggan untuk mendukung otentikasi registri Docker

Bucket ini hanya berlaku untuk lingkungan berdasarkan platform Docker. Bucket menyimpan file yang digunakan untuk mengautentikasi ke registri Docker pribadi yang berada di bucket S3 yang disediakan oleh pelanggan. Untuk informasi selengkapnya, lihat Menggunakan file `Dockerrun.aws.json` v3 di bagian platform Docker dari panduan ini.

S3 Bucket ARN

ARN bervariasi menurut akun pelanggan.

Bucket S3 ARN memiliki format berikut: `arn:aws:s3:::`*`bucket-name`*

Operasi

GetObject

Contoh kebijakan titik akhir VPC

Contoh berikut menggambarkan cara menyediakan akses ke bucket S3 dengan nama amzn-s3-demo-bucket1.

Contoh kebijakan

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowRequestsToDockerRegistryAuth",
            "Effect": "Allow",
            "Principal": {"AWS": "*"},
            "Action": ["GetObject"],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket1"
            ]
         }
    ]
}
```

Memperbarui kebijakan titik akhir VPC Anda

Karena titik akhir VPC hanya memiliki satu kebijakan yang dilampirkan, Anda harus menggabungkan semua izin ke dalam satu kebijakan. Contoh berikut memberikan semua contoh sebelumnya digabungkan menjadi satu.

Untuk detail tentang pembuatan dan penggunaan kebijakan titik akhir VPC, lihat Mengontrol akses ke titik akhir VPC menggunakan kebijakan titik akhir di Panduan.AWS PrivateLink

Seperti contoh sebelumnya, contoh berikut menggambarkan cara menyediakan akses ke bucket S3 yang diperlukan untuk operasi Elastic Beanstalk di Wilayah AS Timur (Ohio) (us-east-2). Ini juga termasuk bucket dengan contoh id AWS akun 123456789012 dan contoh nama bucket amzn-s3-demo-bucket1.

> **Important**
>
> Kami menyarankan Anda untuk menghindari menggunakan karakter wildcard (*) di tempat Wilayah tertentu dalam kebijakan ini. Misalnya, gunakan `arn:aws:s3:::cloudformation-waitcondition-us-east-2/*` dan jangan gunakan

`arn:aws:s3:::cloudformation-waitcondition-*/*`. Menggunakan wildcard dapat menyediakan akses ke bucket S3 yang tidak ingin Anda berikan akses. Jika Anda ingin menggunakan kebijakan untuk lebih dari satu Wilayah, sebaiknya ulangi `Statement` blok pertama untuk setiap Wilayah.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowRequestsToAWSResources",
            "Effect": "Allow",
            "Principal": {"AWS": "*"},
            "Action": ["s3:GetObject"],
            "Resource": [
                "arn:aws:s3:::elasticbeanstalk-platform-assets-us-east-2/*",
                "arn:aws:s3:::elasticbeanstalk-env-resources-us-east-2/*",
                "arn:aws:s3:::elasticbeanstalk-env-resources-us-east-2/*",
                "arn:aws:s3:::elasticbeanstalk-samples-us-east-2/*"
            ]
         },
        {
            "Sid": "AllowRequestsToCloudFormation",
            "Effect": "Allow",
            "Principal": {"AWS": "*"},
            "Action": ["s3:GetObject"],
            "Resource": [
                "arn:aws:s3:::cloudformation-waitcondition-us-east-2/*",
                "arn:aws:s3:::cloudformation-custom-resource-response-us-east-2/*"
            ]
         },
         {
            "Sid": "AllowRequestsToCustomerItems",
            "Effect": "Allow",
            "Principal": {"AWS": "*"},
            "Action": ["GetObject",
                       "GetObjectAcl",
                       "PutObject",
                       "PutObjectAcl",
                       "ListBucket"
                       ],
            "Resource": [
```

```
                "arn:aws:s3:::elasticbeanstalk-us-east-2-123456789012/*"
            ]
         },
        {
            "Sid": "AllowRequestsToDockerRegistryAuth",
            "Effect": "Allow",
            "Principal": {"AWS": "*"},
            "Action": ["GetObject"],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket1""
            ]
         }
    ]
}
```

AWS Elastic Beanstalk keamanan

Gunakan bagian ini untuk mempelajari lebih lanjut tentang tugas keamanan Elastic Beanstalk yang bertanggung jawab, bersama dengan konfigurasi keamanan yang harus Anda pertimbangkan saat menggunakan Elastic Beanstalk untuk memenuhi tujuan keamanan dan kepatuhan Anda.

Keamanan cloud di AWS adalah prioritas tertinggi. Sebagai AWS pelanggan, Anda mendapat manfaat dari pusat data dan arsitektur jaringan yang dibangun untuk memenuhi persyaratan organisasi yang paling sensitif terhadap keamanan.

Keamanan adalah tanggung jawab bersama antara Anda AWS dan Anda. Model Tanggung Jawab Bersama menggambarkan ini sebagai Keamanan dari Cloud dan Keamanan dalam Cloud.

Security of the Cloud - AWS bertanggung jawab untuk melindungi infrastruktur yang menjalankan semua layanan yang ditawarkan di AWS Cloud dan memberi Anda layanan yang dapat Anda gunakan dengan aman. Tanggung jawab keamanan kami adalah prioritas tertinggi di AWS, dan efektivitas keamanan kami secara teratur diuji dan diverifikasi oleh auditor pihak ketiga sebagai bagian dari Program AWS Kepatuhan. Tinjau AWS Layanan dalam Lingkup program AWS jaminan untuk informasi yang berkaitan dengan Elastic Beanstalk.

Keamanan di Cloud — Tanggung jawab Anda ditentukan oleh AWS layanan yang Anda gunakan, dan faktor-faktor lain termasuk sensitivitas data Anda, persyaratan organisasi Anda, serta undang-undang dan peraturan yang berlaku. Dokumentasi ini diharapkan dapat membantu Anda memahami cara menerapkan Model Tanggung Jawab Bersama saat menggunakan Elastic Beanstalk.

Topik

- Perlindungan data di Elastic Beanstalk
- Identity and access management untuk Elastic Beanstalk
- Pencatatan dan pemantauan di Elastic Beanstalk
- Validasi kepatuhan untuk Elastic Beanstalk
- Ketahanan dalam Elastic Beanstalk
- Keamanan infrastruktur di Elastic Beanstalk
- Analisis konfigurasi dan kerentanan di Elastic Beanstalk
- Praktik terbaik keamanan untuk Elastic Beanstalk

Perlindungan data di Elastic Beanstalk

Model tanggung jawab AWS bersama model berlaku untuk perlindungan data di AWS Elastic Beanstalk. Sebagaimana dijelaskan dalam model AWS ini, bertanggung jawab untuk melindungi infrastruktur global yang menjalankan semua AWS Cloud. Anda bertanggung jawab untuk mempertahankan kendali atas konten yang di-host pada infrastruktur ini. Anda juga bertanggung jawab atas tugas-tugas konfigurasi dan manajemen keamanan untuk Layanan AWS yang Anda gunakan. Lihat informasi yang lebih lengkap tentang privasi data dalam Pertanyaan Umum Privasi Data. Lihat informasi tentang perlindungan data di Eropa di pos blog Model Tanggung Jawab Bersama dan GDPR AWS di Blog Keamanan AWS .

Untuk tujuan perlindungan data, kami merekomendasikan agar Anda melindungi Akun AWS kredensyal dan mengatur pengguna individu dengan AWS IAM Identity Center atau AWS Identity and Access Management (IAM). Dengan cara itu, setiap pengguna hanya diberi izin yang diperlukan untuk memenuhi tanggung jawab tugasnya. Kami juga menyarankan supaya Anda mengamankan data dengan cara-cara berikut:

- Gunakan autentikasi multi-faktor (MFA) pada setiap akun.
- Gunakan SSL/TLS untuk berkomunikasi dengan sumber daya. AWS Kami mensyaratkan TLS 1.2 dan menganjurkan TLS 1.3.
- Menyiapkan API dan pencatatan aktivitas pengguna dengan AWS CloudTrail. Untuk informasi tentang penggunaan CloudTrail jejak untuk menangkap AWS aktivitas, lihat Bekerja dengan CloudTrail jejak di AWS CloudTrail Panduan Pengguna.
- Gunakan solusi AWS enkripsi, bersama semua kontrol keamanan standar di dalamnya Layanan AWS.
- Gunakan layanan keamanan terkelola tingkat lanjut seperti Amazon Macie, yang membantu menemukan dan mengamankan data sensitif yang disimpan di Amazon S3.
- Jika Anda memerlukan modul kriptografi yang divalidasi FIPS 140-3 ketika mengakses AWS melalui antarmuka baris perintah atau API, gunakan titik akhir FIPS. Lihat informasi selengkapnya tentang titik akhir FIPS yang tersedia di Standar Pemrosesan Informasi Federal (FIPS) 140-3.

Kami sangat merekomendasikan agar Anda tidak pernah memasukkan informasi identifikasi yang sensitif, seperti nomor rekening pelanggan Anda, ke dalam tanda atau bidang isian bebas seperti bidang Nama. Ini termasuk ketika Anda bekerja dengan Elastic Beanstalk Layanan AWS atau lainnya menggunakan konsol, API, atau. AWS CLI AWS SDKs Data apa pun yang Anda masukkan ke dalam tanda atau bidang isian bebas yang digunakan untuk nama dapat digunakan untuk log penagihan

atau log diagnostik. Saat Anda memberikan URL ke server eksternal, kami sangat menganjurkan supaya Anda tidak menyertakan informasi kredensial di dalam URL untuk memvalidasi permintaan Anda ke server itu.

Untuk topik keamanan Elastic Beanstalk lainnya, lihat AWS Elastic Beanstalk keamanan.

Topik

- Melindungi data menggunakan enkripsi
- Privasi lalu lintas kerja internet

Melindungi data menggunakan enkripsi

Anda dapat menggunakan berbagai bentuk enkripsi data untuk melindungi data Elastic Beanstalk Anda. Perlindungan data mengacu pada perlindungan data saat dalam transit (saat melakukan perjalanan ke dan dari Elastic Beanstalk) dan saat diam (sementara data disimpan di pusat data). AWS

Enkripsi bergerak

Anda dapat mencapai perlindungan data dalam transit dengan dua cara: mengenkripsi koneksi menggunakan Lapisan Soket Aman (SSL), atau menggunakan enkripsi di sisi klien (di mana objek dienkripsi sebelum dikirim). Kedua metode ini berlaku untuk melindungi data aplikasi Anda. Untuk mengamankan koneksi, mengenkripsi menggunakan SSL setiap kali aplikasi Anda, developer dan administrator, dan pengguna akhir mengirim atau menerima objek apa pun. Untuk detail tentang mengenkripsi lalu lintas web ke dan dari aplikasi Anda, lihat the section called “HTTPS”.

Enkripsi di sisi klien bukanlah metode yang valid untuk melindungi kode sumber Anda dalam versi aplikasi dan paket sumber yang Anda upload. Elastic Beanstalk membutuhkan akses ke objek-objek ini, sehingga mereka tidak dapat dienkripsikan. Oleh karena itu, pastikan untuk mengamankan koneksi antara lingkungan pengembangan atau deployment Anda dan Elastic Beanstalk.

Enkripsi saat istirahat

Untuk melindungi aplikasi data at rest Anda, pelajari tentang perlindungan data di layanan penyimpanan yang aplikasi Anda gunakan. Misalnya, lihat Perlindungan Data di Amazon RDS di Panduan Pengguna Amazon RDS, Perlindungan Data di Amazon S3 di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon, atau Mengenkripsi Data dan Metadata di EFS di Panduan Pengguna Amazon Elastic File System.

Elastic Beanstalk menyimpan berbagai objek di bucket Amazon Simple Storage Service (Amazon S3) yang menciptakan untuk setiap Wilayah di mana Anda menciptakan lingkungan. AWS Karena Elastic Beanstalk mempertahankan enkripsi default yang disediakan oleh Amazon S3, Elastic Beanstalk membuat bucket Amazon S3 terenkripsi. Lihat perinciannya di the section called “Amazon S3”. Anda menyediakan beberapa objek yang disimpan dan mengirimkannya ke Elastic Beanstalk, misalnya, versi aplikasi dan paket sumber. Elastic Beanstalk menghasilkan benda lain, misalnya berkas log. Selain data yang disimpan Elastic Beanstalk, aplikasi Anda dapat mentransfer dan/atau menyimpan data sebagai bagian dari operasinya.

Untuk melindungi data yang disimpan di Amazon Elastic Block Store (Amazon EBS) volume yang dilampirkan ke instans lingkungan Anda, aktifkan enkripsi Amazon EBS secara default di AWS akun dan Wilayah Anda. Saat diaktifkan, semua volume Amazon EBS baru dan snapshot mereka secara otomatis dienkripsi menggunakan kunci. AWS Key Management Service Untuk informasi selengkapnya, lihat Enkripsi secara default di Panduan Pengguna Amazon EBS.

Untuk informasi tentang perlindungan data, lihat postingan blog Model Tanggung Jawab Bersama AWS dan Peraturan Perlindungan Data Umum (GDPR) di Blog Keamanan AWS .

Untuk topik keamanan Elastic Beanstalk lainnya, lihat AWS Elastic Beanstalk keamanan.

Privasi lalu lintas kerja internet

Anda dapat menggunakan Amazon Virtual Private Cloud (Amazon VPC) untuk membuat batas antara sumber daya dalam aplikasi Elastic Beanstalk dan mengontrol lalu lintas antara mereka, jaringan on-premise Anda, dan internet. Untuk rincian selengkapnya, lihat the section called “Amazon VPC”.

Untuk informasi lebih lanjut tentang Keamanan Amazon VPC, lihat Keamanan dalam Panduan Pengguna Amazon VPC.

Untuk informasi tentang perlindungan data, lihat postingan blog Model Tanggung Jawab Bersama AWS dan Peraturan Perlindungan Data Umum (GDPR) di Blog Keamanan AWS .

Untuk topik keamanan Elastic Beanstalk lainnya, lihat AWS Elastic Beanstalk keamanan.

Identity and access management untuk Elastic Beanstalk

AWS Identity and Access Management (IAM) adalah AWS layanan yang membantu administrator mengontrol akses ke AWS sumber daya dengan aman. Administrator IAM mengontrol siapa yang dapat terautentikasi (masuk) dan berwenang (memiliki izin) untuk menggunakan sumber daya AWS Elastic Beanstalk . IAM adalah AWS layanan yang dapat Anda gunakan tanpa biaya tambahan.

Untuk rincian tentang bekerja dengan IAM, lihat Menggunakan Elastic Beanstalk dengan AWS Identity and Access Management.

Untuk topik keamanan Elastic Beanstalk lainnya, lihat AWS Elastic Beanstalk keamanan.

AWS kebijakan terkelola untuk AWS Elastic Beanstalk

Kebijakan AWS terkelola adalah kebijakan mandiri yang dibuat dan dikelola oleh AWS. AWS Kebijakan terkelola dirancang untuk memberikan izin bagi banyak kasus penggunaan umum sehingga Anda dapat mulai menetapkan izin kepada pengguna, grup, dan peran.

Perlu diingat bahwa kebijakan AWS terkelola mungkin tidak memberikan izin hak istimewa paling sedikit untuk kasus penggunaan spesifik Anda karena tersedia untuk digunakan semua pelanggan. AWS Kami menyarankan Anda untuk mengurangi izin lebih lanjut dengan menentukan kebijakan yang dikelola pelanggan yang khusus untuk kasus penggunaan Anda.

Anda tidak dapat mengubah izin yang ditentukan dalam kebijakan AWS terkelola. Jika AWS memperbarui izin yang ditentukan dalam kebijakan AWS terkelola, pembaruan akan memengaruhi semua identitas utama (pengguna, grup, dan peran) yang dilampirkan kebijakan tersebut. AWS kemungkinan besar akan memperbarui kebijakan AWS terkelola saat baru Layanan AWS diluncurkan atau operasi API baru tersedia untuk layanan yang ada.

Untuk informasi selengkapnya, lihat Kebijakan terkelola AWS dalam Panduan Pengguna IAM.

Elastic Beanstalk memperbarui kebijakan terkelola AWS

Lihat detail tentang pembaruan kebijakan AWS terkelola untuk Elastic Beanstalk sejak 1 Maret 2021.

Untuk melihat sumber JSON untuk kebijakan terkelola tertentu, lihat Panduan Referensi Kebijakan AWS Terkelola.

Perubahan	Deskripsi	Tanggal
AWSElasticBeanstal kManagedUpdatesCus	Kebijakan ini diperbaru i untuk memungkinkan Elastic Beanstalk melakukan	Februari 27, 2025

Perubahan	Deskripsi	Tanggal
tomerRolePolicy—Memperbarui kebijakan yang ada	pembaruan terkelola saat diaktifkan. Tag propagasi untuk meluncurkan template Untuk informasi selengkapnya, lihat Kebijakan peran layanan yang terkelola.	
AdministratorAccess-AWSElastic Beanstalk — Memperbarui kebijakan yang ada	Kebijakan ini diperbarui untuk mengganti StringLikeoperator dengan ArnLikeoperator guna mengevaluasi kunci tipe ARN di blok kondisi. `iam:PolicyArn` Ini memberikan penegakan yang lebih aman. Untuk informasi selengkapnya, lihat Mengelola kebijakan pengguna Elastic Beanstalk.	Desember 11, 2024

Perubahan	Deskripsi	Tanggal
Kebijakan berikut telah diperbarui: • AWSElasticBeanstalkInternalMaintenanceRolePolicy • AWSElasticBeanstalkMaintenance • AWSElasticBeanstalkManagedUpdatesInternalServiceRolePolicy • AWSElasticBeanstalkManagedUpdatesServiceRolePolicy • AWSElasticBeanstalkRoleCore	Kebijakan ini diperbarui untuk memungkinkan Elastic Beanstalk menambah atau menghapus tag saat membuat atau AWS CloudFormation memperbarui kumpulan tumpukan atau perubahan. Untuk informasi selengkapnya tentang `AWSElasticBeanstalkManagedUpdatesServiceRolePolicy` , lihat Izin peran yang terhubung dengan layanan untuk Elastic Beanstalk. Untuk informasi selengkapnya tentang `AWSElasticBeanstalkRoleCore` , lihat Kebijakan untuk integrasi dengan layanan lain.	April 30, 2024

Perubahan	Deskripsi	Tanggal
AWSElasticBeanstalkService—Memperbarui kebijakan yang ada	Kebijakan ini diperbarui untuk memungkinkan Elastic Beanstalk menandai sumber daya saat pembuatan untuk Elastic Load Balancing, grup Auto Scaling (ASG), dan Amazon ECS. **Note** Kebijakan ini sebelumnya telah digantikan oleh. `AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy` Meskipun kebijakan ini tidak lagi tersedia untuk dilampirkan ke pengguna, grup, atau peran IAM baru, kebijakan ini mungkin masih dilampirkan ke yang sudah ada sebelumnya. Untuk informasi selengkapnya, lihat Kebijakan peran layanan yang terkelola.	10 Mei 2023

Perubahan	Deskripsi	Tanggal
AWSElasticBeanstalkMulticontainerDocker—Memperbarui kebijakan yang ada	Kebijakan ini diperbarui untuk memungkinkan Elastic Beanstalk menandai sumber daya saat pembuatan untuk Amazon ECS. Untuk informasi selengkapnya, lihat Mengelola profil instans Elastic Beanstalk.	Maret 23, 2023
AWSElasticBeanstalkRoleECS —Kebijakan yang ada diperbarui	Kebijakan ini diperbarui untuk memungkinkan Elastic Beanstalk menandai sumber daya saat pembuatan untuk Amazon ECS. Untuk informasi selengkapnya, lihat Kebijakan untuk integrasi dengan layanan lain.	Maret 23, 2023
AdministratorAccess-AWSElastic Beanstalk — Memperbarui kebijakan yang ada	Kebijakan ini diperbarui untuk memungkinkan Elastic Beanstalk menandai sumber daya saat pembuatan untuk Amazon ECS. Untuk informasi selengkapnya, lihat Mengelola kebijakan pengguna Elastic Beanstalk.	Maret 23, 2023

Perubahan	Deskripsi	Tanggal
AWSElasticBeanstalkManagedUpdatesServiceRolePolicy —Memperbarui kebijakan yang ada	Kebijakan ini diperbarui agar Elastic Beanstalk dapat menambahkan tag ke resource Amazon ECS saat dibuat. Untuk informasi selengkapnya, lihat Izin peran yang terhubung dengan layanan untuk Elastic Beanstalk.	Maret 23, 2023
AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy—Memperbarui kebijakan yang ada	Kebijakan ini diperbarui agar Elastic Beanstalk dapat menambahkan tag ke resource Amazon ECS saat dibuat. Untuk informasi selengkapnya, lihat Kebijakan peran layanan yang terkelola.	Maret 23, 2023
AWSElasticBeanstalkManagedUpdatesServiceRolePolicy—Memperbarui kebijakan yang ada	Kebijakan ini diperbarui untuk memungkinkan Elastic Beanstalk menambahkan tag ke grup Auto Scaling saat dibuat. Untuk informasi selengkapnya, lihat Peran yang terhubung dengan layanan permbaharuan yang terkelola.	27 Januari 2023

Perubahan	Deskripsi	Tanggal
AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy—Memperbarui kebijakan yang ada	Kebijakan ini telah diperbarui untuk memungkinkan Elastic Beanstalk menambahkan tag pada pembuatan grup Auto Scaling (ASG). Untuk informasi selengkapnya, lihat Kebijakan peran layanan yang terkelola.	23 Januari 2023
AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy—Memperbarui kebijakan yang ada	Kebijakan ini diperbarui untuk memungkinkan Elastic Beanstalk menambahkan tag pada pembuatan penyeimbang beban elastis (ELB). Untuk informasi selengkapnya, lihat Kebijakan peran layanan yang terkelola.	21 Desember 2022

Perubahan	Deskripsi	Tanggal
AWSElasticBeanstalkManagedUpdatesServiceRolePolicy—Memperbarui kebijakan yang ada	Izin ditambahkan ke kebijakan ini untuk memungkinkan Elastic Beanstalk melakukan hal berikut selama pembaruan terkelola: • Buat dan hapus templat peluncuran dan versi templat. • Luncurkan EC2 instans Amazon dengan templat peluncuran. • Jika Amazon RDS hadir, ambil daftar mesin DB yang tersedia dan informasi tentang instans RDS yang disediakan. Untuk informasi selengkapnya, lihat Peran yang terhubung dengan layanan permbaharuan yang terkelola.	23 Agustus 2022

Perubahan	Deskripsi	Tanggal
AWSElasticBeanstalkReadOnlyAccess— Usang GovCloud (AS) Wilayah AWS	Kebijakan ini telah diganti dengan `AWSElasticBeanstalkReadOnly` . Kebijakan ini akan dihapus di GovCloud (AS) Wilayah AWS. Ketika kebijakan ini dihapus, kebijakan ini tidak akan lagi tersedia untuk lampiran ke pengguna IAM baru, grup, atau peran setelah 17 Juni 2021. Untuk informasi selengkapnya, lihat Kebijakan pengguna.	17 Juni 2021
AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy—Memperbarui kebijakan yang ada	Kebijakan ini diperbarui untuk memungkinkan Elastic Beanstalk membaca atribut untuk Availability Zones. EC2 Hal ini memungkinkan Elastic Beanstalk untuk memberikan validasi yang lebih efektif dari pilihan jenis instans Anda di seluruh Availability Zone. Untuk informasi selengkapnya, lihat Kebijakan peran layanan yang terkelola.	16 Juni 2021

Perubahan	Deskripsi	Tanggal
AWSElasticBeanstalkFullAccess— Usang GovCloud (AS) Wilayah AWS	Kebijakan ini telah diganti dengan `AdministratorAccess-AWSElasticBeanstalk` . Kebijakan ini akan dihapus di GovCloud (AS) Wilayah AWS. Ketika kebijakan ini dihapus, kebijakan ini tidak akan lagi tersedia untuk lampiran ke pengguna IAM baru, grup, atau peran setelah 10 Juni 2021. Untuk informasi selengkapnya, lihat Kebijakan pengguna.	10 Juni 2021

Perubahan	Deskripsi	Tanggal
Kebijakan terkelola berikut tidak digunakan lagi di semua negara Tiongkok: Wilayah AWS • AWSElasticBeanstalkFullAccess • AWSElasticBeanstalkReadOnlyAccess	Kebijakan `AWSElasticBeanstalkFullAccess` ini telah digantikan oleh `AdministratorAccess-AWSElasticBeanstalk` . Kebijakan `AWSElasticBeanstalkReadOnlyAccess` ini telah digantikan oleh `AWSElasticBeanstalkReadOnly` . Kebijakan-kebijakan ini dihapus di seluruh Tiongkok Wilayah AWS. Kebijakan ini tidak akan lagi tersedia untuk lampiran ke pengguna, grup, atau peran IAM baru setelah 3 Juni 2021. Untuk informasi selengkapnya, lihat Kebijakan pengguna.	Juni 3, 2021

Perubahan	Deskripsi	Tanggal
AWSElasticBeanstalkService— Usang	Kebijakan ini telah digantikan oleh. `AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy` Kebijakan ini dihapus dan tidak lagi tersedia untuk dilampirkan ke pengguna, grup, atau peran IAM baru. Untuk informasi selengkapnya, lihat Kebijakan peran layanan yang terkelola.	Juni 2021 - Januari 2022

Perubahan	Deskripsi	Tanggal
Kebijakan terkelola berikut tidak digunakan lagi di semua Wilayah AWS s, kecuali untuk Tiongkok dan GovCloud (AS): • AWSElasticBeanstalkFullAccess • AWSElasticBeanstalkReadOnlyAccess	Kebijakan `AWSElasticBeanstalkFullAccess` ini telah digantikan oleh `AdministratorAccess-AWSElasticBeanstalk` . Kebijakan `AWSElasticBeanstalkReadOnlyAccess` ini telah digantikan oleh `AWSElasticBeanstalkReadOnly` . Kebijakan ini dihapus di semua Wilayah AWS s, kecuali China dan GovCloud (AS). Kebijakan ini tidak akan lagi tersedia untuk lampiran ke pengguna, grup, atau peran IAM baru setelah 16 April 2021. Untuk informasi selengkapnya, lihat Kebijakan pengguna.	16 April 2021

Perubahan	Deskripsi	Tanggal
Kebijakan yang terkelola berikut diperbarui: • AdministratorAccess-Batang AWSElastic kacang • AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy	Kedua kebijakan ini sekarang mendukung PassRole izin di Tiongkok Wilayah AWS. Untuk informasi selengkapnya tentang `AdministratorAccess-AWSElasticBeanstalk` , lihat Kebijakan pengguna. Untuk informasi lebih lanjut tentang `AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy` , lihat Kebijakan peran layanan yang terkelola.	Maret 9, 2021
AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy – Kebijakan baru	Elastic Beanstalk menambahkan kebijakan baru untuk menggantikan kebijakan `AWSElasticBeanstalkService` yang terkelola. Kebijakan terkelola baru ini meningkatkan keamanan untuk sumber daya Anda dengan menerapkan serangkaian izin yang lebih ketat. Untuk informasi selengkapnya, lihat Kebijakan peran layanan yang terkelola.	Maret 3, 2021

Perubahan	Deskripsi	Tanggal
Elastic Beanstalk mulai melacak perubahan	Elastic Beanstalk mulai melacak perubahan untuk kebijakan terkelola. AWS	Maret 1, 2021

Pencatatan dan pemantauan di Elastic Beanstalk

AWS menyediakan beberapa alat untuk memantau sumber daya Elastic Beanstalk Anda dan menanggapi potensi insiden. Pemantauan penting untuk menjaga keandalan, ketersediaan, dan kinerja AWS Elastic Beanstalk dan AWS solusi Anda. Anda harus mengumpulkan data pemantauan dari semua bagian AWS solusi Anda sehingga Anda dapat lebih mudah men-debug kegagalan multipoint jika terjadi.

Untuk informasi selengkapnya tentang pemantauan, lihat Memantau lingkungan di Elastic Beanstalk.

Untuk topik keamanan Elastic Beanstalk lainnya, lihat AWS Elastic Beanstalk keamanan.

Pelaporan kondisi yang ditingkatkan

Pelaporan kondisi yang ditingkatkan adalah fitur yang dapat Anda aktifkan pada lingkungan Anda untuk memungkinkan Elastic Beanstalk untuk mengumpulkan informasi tambahan tentang sumber daya di lingkungan Anda. Elastic Beanstalk menganalisis informasi untuk memberikan gambaran yang lebih baik tentang kesehatan lingkungan secara keseluruhan dan membantu mengidentifikasi masalah yang dapat menyebabkan aplikasi Anda menjadi tidak tersedia. Untuk informasi selengkapnya, lihat Peningkatan pelaporan dan pemantauan kesehatan di Elastic Beanstalk.

Log EC2 instans Amazon

EC2 Instans Amazon di lingkungan Elastic Beanstalk Anda menghasilkan log yang dapat Anda lihat untuk memecahkan masalah dengan aplikasi atau file konfigurasi Anda. Log yang dibuat oleh server web, server aplikasi, skrip platform Elastic Beanstalk AWS CloudFormation , dan disimpan secara lokal pada instance individual. Anda dapat dengan mudah mengambilnya dengan menggunakan konsol manajemen lingkungan atau EB CLI. Anda juga dapat mengonfigurasi lingkungan Anda untuk mengalirkan log ke Amazon CloudWatch Logs secara real time. Untuk informasi selengkapnya, lihat Melihat log dari EC2 instans Amazon di lingkungan Elastic Beanstalk Anda.

Notifikasi lingkungan

Anda dapat mengonfigurasi lingkungan Elastic Beanstalk Anda untuk menggunakan Amazon Simple Notification Service (Amazon SNS) untuk memberi tahu Anda tentang peristiwa penting yang mempengaruhi aplikasi Anda. Tentukan alamat email selama atau setelah pembuatan lingkungan untuk menerima email dari AWS saat terjadi kesalahan, atau ketika kesehatan lingkungan Anda berubah. Untuk informasi selengkapnya, lihat Pemberitahuan lingkungan Elastic Beanstalk dengan Amazon SNS.

CloudWatch Alarm Amazon

Menggunakan CloudWatch alarm, Anda menonton satu metrik selama periode waktu yang Anda tentukan. Jika metrik melebihi ambang batas tertentu, pemberitahuan akan dikirim ke topik atau AWS Auto Scaling kebijakan Amazon SNS. CloudWatch alarm tidak memanggil tindakan karena mereka berada dalam keadaan tertentu. Sebaliknya, alarm memanggil tindakan ketika kondisi diubah dan dipertahankan selama jangka waktu tertentu. Untuk informasi selengkapnya, lihat Menggunakan Elastic Beanstalk dengan Amazon CloudWatch.

AWS CloudTrail log

CloudTrail menyediakan catatan tindakan yang diambil oleh pengguna, peran, atau AWS layanan di Elastic Beanstalk. Dengan menggunakan informasi yang dikumpulkan oleh CloudTrail, Anda dapat menentukan permintaan yang dibuat untuk Elastic Beanstalk, alamat IP dari mana permintaan dibuat, siapa yang membuat permintaan, kapan dibuat, dan detail tambahan. Untuk informasi selengkapnya, lihat Logging panggilan Elastic Beanstalk API dengan AWS CloudTrail.

AWS X-Ray debugging

X-Ray adalah AWS layanan yang mengumpulkan data tentang permintaan yang dilayani aplikasi Anda, dan menggunakannya untuk membuat peta layanan yang dapat Anda gunakan untuk mengidentifikasi masalah dengan aplikasi Anda dan peluang untuk pengoptimalan. Anda dapat menggunakan AWS Elastic Beanstalk konsol atau file konfigurasi untuk menjalankan daemon X-Ray pada instance di lingkungan Anda. Untuk informasi selengkapnya, lihat Mengkonfigurasi debugging AWS X-Ray.

Validasi kepatuhan untuk Elastic Beanstalk

Keamanan dan kepatuhan dinilai AWS Elastic Beanstalk oleh auditor pihak ketiga sebagai bagian dari beberapa program AWS kepatuhan. Ini termasuk SOC, PCI, FedRAMP, HIPAA, dan lainnya.

AWS menyediakan daftar AWS layanan yang sering diperbarui dalam lingkup program kepatuhan khusus di AWS Layanan dalam Lingkup oleh Program Kepatuhan.

Laporan audit pihak ketiga tersedia untuk Anda unduh AWS Artifact. Untuk informasi selengkapnya, lihat Mengunduh Laporan di AWS Artifact.

Untuk informasi selengkapnya tentang program AWS kepatuhan, lihat Program AWS Kepatuhan.

Tanggung jawab kepatuhan Anda saat menggunakan Elastic Beanstalk ditentukan oleh sensitivitas data Anda, tujuan kepatuhan organisasi Anda, serta undang-undang dan peraturan yang berlaku. Jika penggunaan Elastic Beanstalk Anda tunduk pada kepatuhan dengan standar seperti HIPAA, PCI, atau AWS FedRAMP, menyediakan sumber daya untuk membantu:

- Panduan Memulai Cepat Keamanan dan Kepatuhan — Panduan penerapan yang membahas pertimbangan arsitektur dan memberikan langkah-langkah untuk menerapkan lingkungan dasar yang berfokus pada keamanan dan berfokus pada kepatuhan. AWS
- Arsitektur untuk Keamanan dan Kepatuhan HIPAA di Amazon Web Services — Whitepaper yang menjelaskan bagaimana perusahaan dapat menggunakan AWS untuk membuat aplikasi yang sesuai dengan HIPAA.
- AWS Sumber Daya Kepatuhan — Kumpulan buku kerja dan panduan kepatuhan yang mungkin berlaku untuk industri dan lokasi Anda.
- AWS Config – Layanan yang menilai seberapa baik konfigurasi sumber daya Anda mematuhi praktik internal, pedoman industri, dan peraturan.
- AWS Security Hub— Pandangan komprehensif tentang keadaan keamanan Anda di dalamnya AWS yang membantu Anda memeriksa kepatuhan Anda terhadap standar industri keamanan dan praktik terbaik.

Untuk topik keamanan Elastic Beanstalk lainnya, lihat AWS Elastic Beanstalk keamanan.

Ketahanan dalam Elastic Beanstalk

AWS Elastic Beanstalk mengelola dan mengotomatiskan penggunaan infrastruktur AWS global atas nama Anda. Saat menggunakan Elastic Beanstalk, Anda mendapat manfaat dari ketersediaan dan mekanisme toleransi kesalahan yang ditawarkan. AWS

Infrastruktur AWS global dibangun di sekitar AWS Wilayah dan Zona Ketersediaan.

AWS Wilayah menyediakan beberapa Availability Zone yang terpisah secara fisik dan terisolasi, yang terhubung dengan latensi rendah, throughput tinggi, dan jaringan yang sangat redundan.

Dengan Zona Ketersediaan, Anda dapat merancang serta mengoperasikan aplikasi dan basis data yang secara otomatis melakukan failover di antara Zona Ketersediaan tanpa gangguan. Zona Ketersediaan memiliki ketersediaan dan toleransi kesalahan yang lebih baik, dan dapat diskalakan dibandingkan infrastruktur biasa yang terdiri dari satu atau beberapa pusat data.

Untuk informasi selengkapnya tentang AWS Wilayah dan Availability Zone, lihat Infrastruktur AWS Global.

Untuk topik keamanan Elastic Beanstalk lainnya, lihat AWS Elastic Beanstalk keamanan.

Keamanan infrastruktur di Elastic Beanstalk

Sebagai layanan terkelola, AWS Elastic Beanstalk dilindungi oleh prosedur keamanan jaringan AWS global yang dijelaskan dalam situs web Praktik Terbaik untuk Keamanan, Identitas, dan Kepatuhan kami.

Anda menggunakan panggilan API yang AWS dipublikasikan untuk mengakses Elastic Beanstalk melalui jaringan. Klien harus mendukung Keamanan Lapisan Pengangkutan (TLS) 1.2 atau versi yang lebih baru. Selain itu, klien harus mendukung cipher suites dengan perfect forward secrecy (PFS) seperti Ephemeral Diffie-Hellman (DHE) atau Elliptic Curve Diffie-Hellman Ephemeral (ECDHE). Sebagian besar platform modern mendukung mode ini.

Selain itu, permintaan harus ditandatangani menggunakan ID kunci akses dan kunci akses rahasia yang terkait dengan prinsipal IAM. Atau Anda bisa menggunakan AWS Security Token Service (AWS STS) untuk membuat kredensial keamanan sementara untuk menandatangani permintaan.

Untuk topik keamanan Elastic Beanstalk lainnya, lihat AWS Elastic Beanstalk keamanan.

Analisis konfigurasi dan kerentanan di Elastic Beanstalk

AWS dan pelanggan kami berbagi tanggung jawab untuk mencapai tingkat keamanan dan kepatuhan komponen perangkat lunak yang tinggi. AWS Elastic Beanstalk membantu Anda melakukan sisi Anda dari model tanggung jawab bersama dengan menyediakan fitur pembaruan terkelola. Fitur ini secara otomatis menerapkan pembaruan kecil dan patch untuk versi platform yang didukung Elastic Beanstalk.

Selengkapnya, lihat Model tanggung jawab bersama untuk pemeliharaan platform Elastic Beanstalk.

Untuk topik keamanan Elastic Beanstalk lainnya, lihat AWS Elastic Beanstalk keamanan.

Praktik terbaik keamanan untuk Elastic Beanstalk

AWS Elastic Beanstalk menyediakan beberapa fitur keamanan untuk dipertimbangkan saat Anda mengembangkan dan menerapkan kebijakan keamanan Anda sendiri. Praktik terbaik berikut adalah pedoman umum dan tidak mewakili solusi keamanan yang lengkap. Karena praktik terbaik ini mungkin tidak sesuai atau cukup untuk lingkungan Anda, anggap sebagai pertimbangan yang membantu, dan bukan sebagai resep.

Untuk topik keamanan Elastic Beanstalk lainnya, lihat AWS Elastic Beanstalk keamanan.

Praktik terbaik keamanan pencegahan

Kontrol keamanan preventif berusaha mencegah insiden sebelum terjadi.

Terapkan akses hak akses paling rendah

Elastic AWS Identity and Access Management Beanstalk menyediakan (IAM) kebijakan terkelola untuk profil misalnya, peran layanan, dan pengguna IAM. Kebijakan terkelola ini menentukan semua izin yang mungkin diperlukan untuk pengoperasian lingkungan dan aplikasi yang benar.

Aplikasi Anda mungkin tidak memerlukan semua izin dalam kebijakan terkelola kami. Anda dapat menyesuaikan mereka dan memberikan hanya izin yang diperlukan untuk lingkungan instans Anda, layanan Elastic Beanstalk, dan pengguna Anda untuk melakukan tugas-tugas mereka. Hal ini sangat relevan untuk kebijakan pengguna, di mana peran pengguna yang berbeda mungkin memiliki kebutuhan izin yang berbeda. Menerapkan akses hak akses paling rendah adalah hal mendasar dalam mengurangi risiko keamanan dan dampak yang dapat diakibatkan oleh kesalahan atau niat jahat.

Lindungi data aplikasi sensitif

Saat aplikasi Anda perlu mengakses informasi sensitif seperti kredensional, kunci API, atau data konfigurasi, ikuti praktik berikut untuk menjaga keamanan:

- Ambil data sensitif langsung dari AWS Secrets Manager atau AWS Systems Manager Parameter Store menggunakan masing-masing SDKs atau APIs dalam kode aplikasi Anda. Ini memberikan cara yang paling aman dan fleksibel untuk mengakses informasi sensitif.

- Jika Anda meneruskan data sensitif dari AWS Secrets Manager atau AWS Systems Manager Parameter Store sebagai variabel lingkungan (lihatAmbil rahasia ke variabel lingkungan), batasi akses ke pasangan EC2 kunci dengan hati-hati dan konfigurasikan peran IAM yang sesuai dengan izin hak istimewa paling sedikit untuk instance Anda.
- Jangan pernah mencetak, mencatat, atau mengekspos data sensitif dalam kode aplikasi Anda, karena nilai-nilai ini dapat berakhir di file log atau pesan kesalahan yang mungkin terlihat oleh pengguna yang tidak sah.

Perbarui platform Anda secara teratur

Elastic Beanstalk secara teratur merilis versi platform baru untuk memperbarui semua platformnya. Versi platform baru menyediakan sistem operasi, waktu aktif, server aplikasi, dan pembaruan server web, dan pembaruan komponen Elastic Beanstalk. Banyak pembaruan platform ini mencakup perbaikan keamanan yang penting. Pastikan lingkungan Elastic Beanstalk Anda berjalan pada versi platform yang didukung (biasanya versi terbaru untuk platform Anda). Untuk rincian selengkapnya, lihat Memperbarui versi platform lingkungan Elastic Beanstalk Anda.

Cara termudah untuk menjaga platform lingkungan Anda tetap up to date adalah mengonfigurasi lingkungan untuk menggunakan pembaruan platform terkelola.

Menegakkan IMDSv2 pada contoh lingkungan

Instans Amazon Elastic Compute Cloud (Amazon EC2) di lingkungan Elastic Beanstalk Anda menggunakan layanan metadata instans (IMDS), komponen on-instance, untuk mengakses metadata instance dengan aman. IMDS mendukung dua metode untuk mengakses data: IMDSv1 dan. IMDSv2 IMDSv2 menggunakan permintaan berorientasi sesi dan mengurangi beberapa jenis kerentanan yang dapat digunakan untuk mencoba mengakses IMDS. Untuk detail tentang keuntungan IMDSv2, lihat penyempurnaan untuk menambahkan pertahanan secara mendalam ke Layanan Metadata EC2 Instance.

IMDSv2 lebih aman, jadi itu ide yang baik untuk menegakkan penggunaan IMDSv2 pada instans Anda. Untuk menegakkan IMDSv2, pastikan bahwa semua komponen dukungan aplikasi Anda IMDSv2, dan kemudian nonaktifkan IMDSv1. Untuk informasi selengkapnya, lihat the section called “IMDS”.

Praktik terbaik keamanan detective

Kontrol keamanan detective mengidentifikasi pelanggaran keamanan setelah mereka telah terjadi. Mereka dapat membantu Anda mendeteksi potensi ancaman keamanan atau insiden.

Melaksanakan pemantauan

Pemantauan merupakan bagian penting dalam menjaga keandalan, keamanan, ketersediaan, dan kinerja solusi Elastic Beanstalk Anda. AWS menyediakan beberapa alat dan layanan untuk membantu Anda memantau AWS layanan Anda.

Berikut adalah beberapa instans item yang perlu dipantau:

- CloudWatch Metrik Amazon untuk Elastic Beanstalk — Atur alarm untuk metrik Elastic Beanstalk utama dan untuk metrik kustom aplikasi Anda. Lihat perinciannya di Menggunakan Elastic Beanstalk dengan Amazon CloudWatch.
- AWS CloudTrail entri — Melacak tindakan yang mungkin memengaruhi ketersediaan, seperti `UpdateEnvironment` atau`TerminateEnvironment`. Lihat perinciannya di Logging panggilan Elastic Beanstalk API dengan AWS CloudTrail.

Aktifkan AWS Config

AWS Config memberikan tampilan terperinci tentang konfigurasi AWS sumber daya di akun Anda. Anda dapat melihat bagaimana sumber daya terkait, mendapatkan riwayat perubahan konfigurasi, dan melihat bagaimana hubungan dan konfigurasi berubah seiring waktu.

Anda dapat menggunakan AWS Config untuk menentukan aturan yang mengevaluasi konfigurasi sumber daya untuk kepatuhan data. AWS Config aturan mewakili pengaturan konfigurasi ideal untuk sumber daya Elastic Beanstalk Anda. Jika sumber daya melanggar aturan dan ditandai sebagai tidak sesuai, AWS Config Anda dapat memperingatkan Anda menggunakan topik Amazon Simple Notification Service (Amazon SNS). Lihat perinciannya di Menemukan dan melacak sumber daya Elastic Beanstalk dengan AWS Config.

Peran Elastic Beanstalk Service, profil instans, dan kebijakan pengguna

Peran adalah entitas yang Anda buat dengan AWS Identity and Access Management (IAM) untuk menerapkan izin. Ada peran yang diperlukan agar lingkungan Elastic Beanstalk Anda berfungsi dengan baik. Anda juga memiliki opsi untuk membuat kebijakan dan peran kustom Anda sendiri yang dapat Anda tetapkan ke pengguna atau grup.

Peran yang diperlukan untuk lingkungan Elastic Beanstalk Anda

Saat Anda membuat lingkungan, AWS Elastic Beanstalk meminta Anda untuk memberikan peran berikut AWS Identity and Access Management (IAM):

- Peran layanan: Elastic Beanstalk mengasumsikan peran layanan untuk menggunakan orang lain atas nama Anda. Layanan AWS
- Profil instans Elastic Beanstalk menerapkan profil instance ke instans EC2 Amazon di lingkungan Anda. Tindakan ini memungkinkan mereka untuk melakukan tugas yang diperlukan, seperti mengambil informasi dari Amazon Simple Storage Service (Amazon S3) dan mengunggah log ke S3.

Buat peran layanan dan peran profil EC2 instance

Jika AWS akun Anda tidak memiliki profil EC2 instans atau peran layanan, Anda harus membuat salah satu dari masing-masing menggunakan layanan IAM. Anda kemudian dapat menetapkan profil EC2 instance dan peran layanan ke lingkungan baru yang Anda buat. Wizard Buat lingkungan memandu Anda ke layanan IAM, sehingga Anda dapat membuat peran ini dengan izin yang diperlukan.

Kebijakan dan peran opsional untuk mengelola lingkungan Elastic Beanstalk Anda

Anda dapat membuat kebijakan pengguna secara opsional dan menerapkannya pada pengguna dan grup IAM di akun Anda. Melakukannya memungkinkan pengguna untuk membuat dan mengelola aplikasi dan lingkungan Elastic Beanstalk. Anda juga dapat menetapkan kebijakan terkelola Elastic

Beanstalk untuk akses penuh dan akses hanya-baca ke pengguna atau grup. Untuk informasi selengkapnya tentang kebijakan ini, lihatthe section called “Kebijakan pengguna”.

Anda dapat membuat profil instans Anda sendiri dan kebijakan pengguna untuk skenario lanjutan. Jika instans Anda perlu mengakses layanan yang tidak disertakan dalam kebijakan default, Anda dapat membuat kebijakan baru atau menambahkan kebijakan tambahan ke kebijakan default. Jika kebijakan terkelola terlalu permisif untuk kebutuhan Anda, Anda juga dapat membuat kebijakan pengguna yang lebih ketat. Untuk informasi selengkapnya tentang AWS izin, lihat Panduan Pengguna IAM.

Peran layanan Elastic Beanstalk

Peran layanan adalah IAM role yang diasumsikan Elastic Beanstalk saat memanggil layanan lain atas nama Anda. Misalnya, Elastic Beanstalk menggunakan peran layanan saat memanggil Amazon Elastic Compute Cloud (Amazon), Elastic Load Balancing EC2, dan EC2 Amazon Auto Scaling untuk mengumpulkan informasi. APIs Peran layanan yang digunakan Elastic Beanstalk adalah peran yang Anda tentukan saat membuat lingkungan Elastic Beanstalk.

Ada dua kebijakan terkelola yang melekat pada peran layanan. Kebijakan ini memberikan izin yang memungkinkan Elastic Beanstalk mengakses sumber daya yang diperlukan untuk membuat dan mengelola AWS lingkungan Anda. Satu kebijakan terkelola memberikan izin untuk pemantauan kesehatan yang ditingkatkan dan dukungan Amazon SQS tingkat pekerja, dan kebijakan lainnya memberikan izin tambahan yang diperlukan untuk pembaruan platform terkelola.

`AWSElasticBeanstalkEnhancedHealth`

Kebijakan ini memberikan izin untuk Elastic Beanstalk untuk memantau instans dan kondisi lingkungan. Ini juga mencakup tindakan Amazon SQS untuk memungkinkan Elastic Beanstalk memantau aktivitas antrian untuk lingkungan pekerja. Untuk melihat konten kebijakan terkelola ini, lihat AWSElasticBeanstalkEnhancedHealthhalaman di Panduan Referensi Kebijakan AWS Terkelola.

`AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy`

Kebijakan ini memberikan izin untuk Elastic Beanstalk untuk memperbarui lingkungan atas nama Anda untuk melakukan pembaruan platform yang terkelola. Untuk melihat konten kebijakan terkelola ini, lihat AWSElasticBeanstalkManagedUpdatesCustomerRolePolicyhalaman di Panduan Referensi Kebijakan AWS Terkelola.

Pengelompokan izin tingkat layanan

Kebijakan ini dikelompokkan ke dalam pernyataan berdasarkan kumpulan izin yang diberikan.

- *ElasticBeanstalkPermissions*— Kelompok izin ini untuk memanggil tindakan layanan Elastic Beanstalk (Elastic Beanstalk). APIs
- *AllowPassRoleToElasticBeanstalkAndDownstreamServices* – Kelompok izin ini mengizinkan peran apa pun untuk diteruskan ke Elastic Beanstalk dan ke layanan hilir lainnya seperti AWS CloudFormation.
- *ReadOnlyPermissions* – Kelompok izin ini adalah untuk mengumpulkan informasi tentang lingkungan berjalan.
- **OperationPermissions* – Grup dengan pola penamaan ini adalah untuk memanggil operasi yang diperlukan untuk melakukan pembaruan platform.
- **BroadOperationPermissions* – Grup dengan pola penamaan ini adalah untuk memanggil operasi yang diperlukan untuk melakukan pembaruan platform. Mereka juga menyertakan izin yang luas untuk mendukung lingkungan lama.
- **TagResource*— Grup dengan pola penamaan ini adalah untuk panggilan yang menggunakan tag-on-create APIs untuk melampirkan tag pada sumber daya yang sedang dibuat di lingkungan Elastic Beanstalk.

Anda dapat membuat lingkungan Elastic Beanstalk dengan salah satu pendekatan berikut. Setiap bagian menjelaskan bagaimana pendekatan menangani peran layanan.

Konsol Elastic Beanstalk

Jika Anda membuat lingkungan menggunakan konsol Elastic Beanstalk, Elastic Beanstalk meminta Anda untuk membuat peran layanan yang diberi nama. `aws-elasticbeanstalk-service-role` Ketika dibuat melalui Elastic Beanstalk, peran ini mencakup kebijakan kepercayaan yang memungkinkan Elastic Beanstalk untuk mengambil peran layanan. Dua kebijakan terkelola yang dijelaskan sebelumnya dalam topik ini juga melekat pada peran tersebut.

Antarmuka Baris Perintah Elastic Beanstalk (EB CLI)

Anda dapat membuat lingkungan menggunakan the section called “eb create” perintah Elastic Beanstalk Command Line Interface (EB CLI). Jika Anda tidak menentukan peran layanan melalui `--service-role` opsi. Elastic Beanstalk menciptakan peran layanan default yang sama. `aws-elasticbeanstalk-service-role` Jika peran layanan default sudah ada, Elastic Beanstalk

menggunakannya untuk lingkungan baru. Ketika dibuat melalui Elastic Beanstalk, peran ini mencakup kebijakan kepercayaan yang memungkinkan Elastic Beanstalk untuk mengambil peran layanan. Dua kebijakan terkelola yang dijelaskan sebelumnya dalam topik ini juga melekat pada peran tersebut.

Elastic Beanstalk API

Anda dapat membuat lingkungan menggunakan `CreateEnvironment` aksi Elastic Beanstalk API. Jika Anda tidak menentukan peran layanan, Elastic Beanstalk akan membuat peran terkait layanan pemantauan. Ini adalah jenis peran layanan unik yang telah ditentukan sebelumnya oleh Elastic Beanstalk untuk menyertakan semua izin yang diperlukan layanan untuk memanggil orang lain atas nama Anda. Layanan AWS Peran terkait layanan berkaitan dengan akun Anda. Elastic Beanstalk membuatnya sekali, dan kemudian menggunakannya kembali saat membuat lingkungan tambahan. Anda juga dapat menggunakan IAM untuk membuat peran terkait layanan pemantauan untuk akun Anda terlebih dahulu. Jika akun Anda memiliki peran terkait layanan pemantauan, Anda dapat menggunakannya untuk membuat lingkungan menggunakan konsol Elastic Beanstalk, Elastic Beanstalk API, atau EB CLI. Untuk petunjuk tentang cara menggunakan peran terkait layanan dengan lingkungan Elastic Beanstalk, lihat. Menggunakan peran yang terhubung dengan layanan untuk Elastic Beanstalk

Untuk informasi selengkapnya tentang peran layanan, lihat Mengelola peran layanan Elastic Beanstalk.

Profil instans Elastic Beanstalk

Profil instans adalah peran IAM yang diterapkan ke EC2 instans Amazon yang diluncurkan di lingkungan Elastic Beanstalk Anda. Saat membuat lingkungan Elastic Beanstalk, Anda menentukan profil instance yang digunakan EC2 saat instance Anda melakukan tindakan berikut:

- Menerima kembali versi aplikasi dari Amazon Simple Storage Service (Amazon S3)
- Menulis log ke Amazon S3
- Pada lingkungan terpadu AWS X-Ray, unggah data debugging ke X-Ray
- Di lingkungan Docker yang dikelola Amazon ECS, koordinasikan penerapan kontainer dengan Amazon Elastic Container Service (Amazon ECS)
- Di lingkungan pekerja, baca mulai dari antrean Amazon Simple Queue Service (Amazon SQS)
- Di lingkungan pekerja, lakukan pemilihan pemimpin dengan Amazon DynamoDB
- Di lingkungan pekerja, publikasikan metrik kesehatan instans ke Amazon CloudWatch

Kebijakan terkelola

Elastic Beanstalk menyediakan serangkaian kebijakan terkelola yang EC2 memungkinkan instans di lingkungan Anda melakukan operasi yang diperlukan. Kebijakan terkelola yang diperlukan untuk kasus penggunaan dasar adalah sebagai berikut.

- `AWSElasticBeanstalkWebTier`
- `AWSElasticBeanstalkWorkerTier`
- `AWSElasticBeanstalkMulticontainerDocker`

Jika aplikasi web Anda memerlukan akses ke tambahan lainnya Layanan AWS, tambahkan pernyataan atau kebijakan terkelola ke profil instans yang memungkinkan akses ke layanan tersebut. Untuk informasi selengkapnya, lihat Menambahkan izin ke profil instans default.

Membuat profil EC2 instance

Jika AWS akun Anda tidak memiliki profil EC2 instance, Anda harus membuatnya menggunakan layanan IAM. Anda kemudian dapat menetapkan profil EC2 instance ke lingkungan baru yang Anda buat. Langkah-langkah Create environment di konsol Elastic Beanstalk memberi Anda akses ke konsol IAM, sehingga Anda dapat EC2 membuat profil instance dengan izin yang diperlukan.

Anda juga dapat membuat profil EC2 instance dengan langsung mengakses konsol IAM, tanpa melalui konsol Elastic Beanstalk. Untuk langkah-langkah mendetail untuk membuat profil instans EC2 Elastic Beanstalk di konsol IAM, lihat. Membuat profil instans

Kebijakan pengguna Elastic Beanstalk

Buat pengguna IAM untuk setiap pengguna yang menggunakan Elastic Beanstalk untuk menghindari penggunaan akun root Anda atau berbagi kredensi. Sebagai praktik terbaik keamanan, hanya berikan izin kepada pengguna ini untuk mengakses layanan dan fitur yang mereka butuhkan.

Elastic Beanstalk memerlukan izin tidak hanya untuk tindakan API-nya sendiri, tetapi juga untuk beberapa layanan lainnya. AWS Elastic Beanstalk menggunakan izin pengguna untuk meluncurkan sumber daya di lingkungan. Sumber daya ini mencakup EC2 instans, penyeimbang beban Elastic Load Balancing, dan grup Auto Scaling. Elastic Beanstalk juga menggunakan izin pengguna untuk menyimpan log dan template ke Amazon Simple Storage Service (Amazon S3), mengirim notifikasi ke Amazon SNS, menetapkan profil instans, dan menerbitkan metrik. CloudWatch Elastic

AWS CloudFormation Beanstalk memerlukan izin untuk mengatur penyebaran dan pembaruan sumber daya. Elastic Beanstalk juga memerlukan izin Amazon RDS untuk membuat basis data bila diperlukan, dan izin Amazon SQS untuk membuat antrean bagi lingkungan pekerja.

Untuk informasi selengkapnya tentang kebijakan pengguna, lihat Mengelola kebijakan pengguna Elastic Beanstalk.

Tutorial dan sampel

Tutorial khusus bahasa dan kerangka kerja tersebar di seluruh Panduan AWS Elastic Beanstalk Pengembang. Tutorial yang baru dan diperbarui akan ditambahkan ke daftar ini saat dipublikasikan. Pembaruan terbaru ditampilkan terlebih dahulu.

Tutorial ini ditargetkan pada pengguna tingkat menengah dan tidak akan berisi petunjuk tentang langkah-langkah dasar seperti mendaftar pada AWS. Jika ini adalah pertama kalinya Anda menggunakan AWS atau Elastic Beanstalk, lihat panduan Memulai untuk mengaktifkan dan menjalankan lingkungan Elastic Beanstalk pertama Anda.

- Ruby on Rails - Men-deploy aplikasi rails ke Elastic Beanstalk
- Ruby dan Sinatra - Men-deploy aplikasi sinatra ke Elastic Beanstalk
- PHP dan Konfigurasi HA MySQL - Men-deploy aplikasi PHP ketersediaan tinggi dengan basis data Amazon RDS eksternal ke Elastic Beanstalk
- PHP dan Laravel - Men-deploy aplikasi Laravel ke Elastic Beanstalk
- PHP dan CakePHP - Men-deploy aplikasi CakePHP ke Elastic Beanstalk
- PHP dan Konfigurasi HA Drupal - Men-deploy situs web Drupal ketersediaan tinggi dengan basis data Amazon RDS eksternal ke Elastic Beanstalk
- Konfigurasi PHP dan WordPress HA - Menyebarkan WordPress situs web dengan ketersediaan tinggi dengan database Amazon RDS eksternal ke Elastic Beanstalk
- Node.js dengan Konfigurasi HA DynamoDB - Men-deploy aplikasi Node.js dengan DynamoDB ke Elastic Beanstalk
- ASP.NET Core - QuickStart: Menyebarkan aplikasi ASP.NET ke Elastic Beanstalk
- Python dan Flask - Men-deploy aplikasi Flask ke Elastic Beanstalk
- Python dan Django - Men-deploy aplikasi Django ke Elastic Beanstalk
- Node.js dan Express - Menyebarkan aplikasi Node.js Express ke Elastic Beanstalk
- Docker, PHP dan nginx - Membuat lingkungan Docker yang dikelola ECS dengan konsol Elastic Beanstalk

Anda dapat mengunduh aplikasi sampel yang digunakan oleh Elastic Beanstalk ketika Anda membuat lingkungan tanpa memberikan paket sumber beserta tautan berikut:

- Docker – docker.zip

- Multicontainer Docker - 2.zip docker-multicontainer-v
- Docker yang telah dikonfigurasi sebelumnya (Glassfish) - 1.zip docker-glassfish-v
- Go – go.zip
- Corretto – corretto.zip
- Tomcat – tomcat.zip
- .NET Core di Linux — dotnet-core-linux.zip
- .NET Inti — dotnet-asp-windows.zip
- Node.js – nodejs.zip
- PHP – php.zip
- Python – python.zip
- Ruby – ruby.zip

Contoh aplikasi yang lebih terlibat yang menunjukkan penggunaan kerangka kerja web, pustaka, dan alat tambahan tersedia sebagai proyek open source pada: GitHub

- Load-balanced WordPress (tutorial) - File konfigurasi untuk menginstal dengan WordPress aman dan menjalankannya di lingkungan Elastic Beanstalk yang seimbang beban.
- Drupal dengan Keseimbangan beban (tutorial) – File konfigurasi dan instruksi untuk menginstal Drupal dengan aman dan menjalankannya di lingkungan Elastic Beanstalk dengan keseimbangan beban.
- Scorekeep - API RESTful web yang menggunakan framework Spring dan AWS SDK untuk Java untuk menyediakan antarmuka untuk membuat dan mengelola pengguna, sesi, dan game. API dipaket dengan aplikasi web Angular 1.5 yang memakai API melalui HTTP. Termasuk cabang yang menunjukkan integrasi dengan Amazon Cognito, AWS X-Ray, dan Amazon Relational Database Service.

  Aplikasi ini menggunakan fitur platform Java SE untuk mengunduh dependensi dan membangun pada instans, meminimalkan ukuran paket sumber. Aplikasi ini juga mencakup file konfigurasi nginx yang mengganti konfigurasi default untuk menyajikan aplikasi web frontend secara statis di port 80 melalui proksi, dan merutekan permintaan ke jalur dalam `/api` ke API yang berjalan di `localhost:5000`.
- Apakah ia memiliki ular? - Aplikasi Tomcat yang menunjukkan penggunaan RDS dalam aplikasi web Java EE di Elastic Beanstalk. Proyek ini menunjukkan penggunaan Servlets,, Simple Tag

Support JSPs, Tag Files, JDBC, SQL, Log4J, Bootstrap, Jackson, dan file konfigurasi Elastic Beanstalk.

- Load Generator Locust - Proyek ini menunjukkan tentang penggunaan fitur platform Java SE untuk menginstal dan menjalankan Locust, alat penghasil muatan yang ditulis dalam Python. Proyek ini mencakup file konfigurasi yang menginstal dan mengonfigurasi Locust, skrip pembuatan yang mengonfigurasikan tabel DynamoDB, dan Procfile yang menjalankan Locust.
- Bagikan Pikiran Anda (tutorial) - aplikasi PHP yang menunjukkan penggunaan MySQL pada Amazon RDS, Composer, dan file konfigurasi.
- Sebuah Startup Baru (tutorial) - Node.js contoh aplikasi yang menunjukkan penggunaan DynamoDB, SDK JavaScript untuk Node.js, AWS manajemen paket npm, dan file konfigurasi.

Migrasi aplikasi IIS ke Elastic Beanstalk

AWS Elastic Beanstalk menyediakan jalur migrasi yang efisien untuk aplikasi Windows Anda yang berjalan di Internet Information Services (IIS). Kemampuan migrasi yang dijelaskan dalam Bab ini secara signifikan mengurangi waktu dan kompleksitas yang biasanya terkait dengan migrasi cloud, membantu Anda mempertahankan fungsionalitas aplikasi dan integritas konfigurasi selama transisi ke AWS.

eb migrateOperasi

Gunakan eb migrate perintah di Elastic Beanstalk Command Line Interface (EB CLI), untuk secara otomatis menemukan, mengemas, dan menyebarkan aplikasi IIS Anda ke file. AWS Cloud Proses ini mempertahankan fungsionalitas aplikasi dan mempertahankan konfigurasi Anda, termasuk binding, kumpulan aplikasi, dan pengaturan otentikasi.

Langkah-langkah berikut merangkum proses yang dilakukan `eb migrate` operasi untuk mentransisikan aplikasi Anda ke AWS Cloud:

1. Temukan situs IIS dan konfigurasinya.
2. Konten dan konfigurasi aplikasi Package.
3. Buat lingkungan dan aplikasi Elastic Beanstalk.
4. Menyebarkan aplikasi dengan pengaturan yang diawetkan.

Alur kerja dan opsi eksekusi lokasi

eb migratePerintah ini menyediakan opsi untuk alur kerja migrasi yang fleksibel dan lokasi eksekusi. Secara default, jalankan perintah di server target yang berisi aplikasi yang ingin Anda migrasikan ke Elastic Beanstalk. Jika Anda tidak dapat menjalankan perintah secara langsung di server aplikasi, gunakan `remote` opsi untuk menjalankan perintah dari host bastion yang terhubung ke server target yang berisi aplikasi dan konfigurasi Anda. Untuk menyelesaikan migrasi dalam dua langkah, Anda juga dapat membuat paket migrasi tanpa menerapkannya menggunakan `archive-only` opsi dan kemudian menerapkannya nanti sesuai keinginan Anda menggunakan opsi. `archive`

Untuk informasi referensi tentang eb migrate perintah, lihatthe section called “eb migrate”.

Topik

Topik berikut memberikan informasi rinci tentang migrasi aplikasi IIS ke Elastic Beanstalk:

- the section called “Prasyarat”- Memahami perangkat lunak, akses, dan izin yang diperlukan untuk memigrasikan aplikasi Windows Anda ke AWS Elastic Beanstalk lingkungan.
- the section called “Glosarium migrasi”- Memahami bagaimana komponen IIS memetakan ke sumber daya Elastic Beanstalk
- the section called “Pemetaan migrasi IIS ke Elastic Beanstalk”- Memahami bagaimana komponen IIS memetakan ke sumber daya Elastic Beanstalk
- the section called “Migrasi IIS dasar”- Pelajari cara melakukan migrasi dasar
- the section called “Skenario migrasi lanjutan”- Menangani skenario migrasi yang kompleks
- the section called “Konfigurasi keamanan”- Konfigurasikan pengaturan keamanan selama migrasi
- the section called “Konfigurasi jaringan”- Mengelola konfigurasi jaringan dan port
- the section called “Pemecahan masalah dan diagnostik”- Memecahkan masalah migrasi umum
- the section called “Opsi migrasi: EB CLI vs MGN”- Bandingkan dua opsi migrasi utama.

Prasyarat

Sebelum menggunakan eb migrate perintah, pastikan lingkungan Anda memenuhi persyaratan ini:

Instalasi dan versi IIS

Server tempat Anda bermigrasi harus menjalankan Internet Information Services (IIS) versi 7.0 atau yang lebih baru. IIS 10.0 pada Windows Server 2016 atau yang lebih baru menyediakan lingkungan yang paling kompatibel untuk migrasi.

Untuk memverifikasi versi IIS Anda, jalankan perintah berikut:

```
PS C:\migrations_workspace> Get-ItemProperty "HKLM:\SOFTWARE\Microsoft\InetStp\"
...
SetupString             : IIS 10.0
VersionString           : Version 10.0
...
```

Persyaratan Windows Server

Lingkungan sumber Anda harus menjalankan Windows Server 2016 atau yang lebih baru untuk kompatibilitas optimal. Elastic Beanstalk mendukung versi Windows Server ini sebagai platform target:

- Windows Server 2025

- Windows Server 2022
- Windows Server 2019
- Windows Server 2016

Instalasi EB CLI

- Alur kerja default (tanpa `--remote` opsi):
  - Python dan Elastic Beanstalk Command Line Interface (EB CLI) harus diinstal pada server yang berisi aplikasi yang ingin Anda migrasikan ke Elastic Beanstalk. Meskipun tidak diperlukan, kami sarankan menginstal EB CLI di dalam kotak pasir `virtualenv` seperti yang dijelaskan dalam. the section called “Instal di virtualenv”
- Menggunakan `--remote` opsi:
  - Python dan Elastic Beanstalk Command Line Interface (EB CLI) harus diinstal pada host bastion Anda. Meskipun tidak diperlukan, kami sarankan menginstal EB CLI di dalam kotak pasir `virtualenv` seperti yang dijelaskan dalam. the section called “Instal di virtualenv”

Izin yang diperlukan

Anda memerlukan kredensi dan izin berikut:

- Hak istimewa administrator di server IIS sumber atau di host bastion (jika menggunakan opsi). `--remote`
- AWS kredensil dengan izin untuk membuat dan mengelola sumber daya Elastic Beanstalk

Penyebaran Web 3.6

Alat Microsoft Web Deploy (versi 3.6 atau yang lebih baru) harus diinstal pada server sumber Anda atau pada host bastion (jika menggunakan opsi). `--remote` Alat ini digunakan oleh eb migrate untuk mengemas aplikasi Anda.

Untuk memverifikasi instalasi, jalankan perintah berikut:

:

```
PS C:\migrations_workspace> Get-ItemProperty "HKLM:\SOFTWARE\Microsoft\IIS
 Extensions\MSDeploy\3" -Name InstallPath

InstallPath  : C:\Program Files\IIS\Microsoft Web Deploy V3\
...
```

Untuk petunjuk penginstalan, lihat Menginstal dan Mengonfigurasi Penerapan Web di IIS 8.0 atau Nanti di situs web dokumentasi produk Microsoft Windows.

Persyaratan jaringan

- Alur kerja default (tanpa `--remote` opsi):
  - Server sumber Anda harus memiliki akses internet keluar ke AWS layanan.
- Menggunakan `--remote` opsi:
  - Server sumber Anda harus memiliki akses internet keluar ke AWS layanan.
  - Konfigurasikan aturan masuk grup keamanan yang tepat yang memungkinkan koneksi jaringan keluar dari host bastion Anda dan koneksi masuk ke mesin jarak jauh. Pastikan IP host bastion terdaftar melalui TCP pada port 22 untuk mengakses mesin jarak jauh.
  - Pastikan klien SSH Anda diinstal dan berjalan di mesin jarak jauh Anda serta host bastion Anda.
  - Pastikan konfigurasi firewall Anda berisi aturan yang sesuai yang membuka port 22 atau mengizinkan koneksi ke klien.
  - Uji koneksi Anda dengan melakukan SSH secara manual ke host jarak jauh dari host bastion sebelum mencoba migrasi.

Glosarium migrasi

Glosarium ini memberikan definisi untuk istilah dan konsep kunci yang terkait dengan IIS, Elastic Beanstalk, dan migrasi aplikasi IIS ke Elastic Beanstalk.

Istilah Windows, IIS, dan .NET

IIS

Internet Information Services, perangkat lunak server web yang dikembangkan oleh Microsoft untuk digunakan dengan Windows Server. IIS menghosting situs web, aplikasi web, dan layanan web, menyediakan platform untuk menjalankan ASP.NET dan teknologi web lainnya. Selama migrasi ke Elastic Beanstalk, situs IIS dan konfigurasinya dikemas dan digunakan ke instance Windows Server di Cloud. AWS

IIS versi 7.0 dan yang lebih baru didukung untuk migrasi, dengan IIS 10.0 pada Windows Server 2016 atau yang lebih baru menyediakan lingkungan yang paling kompatibel.

.NET Kerangka

Platform pengembangan perangkat lunak yang dikembangkan oleh Microsoft untuk membangun dan menjalankan aplikasi Windows. Ini menyediakan perpustakaan kelas besar yang disebut

Framework Class Library (FCL) dan mendukung interoperabilitas bahasa di beberapa bahasa pemrograman.

Saat bermigrasi ke Elastic Beanstalk, aplikasi yang dibangun di atas .NET Framework terus berjalan pada versi framework yang sama di lingkungan cloud. Elastic Beanstalk mendukung beberapa versi.NET Framework (4.x) pada platform Windows Server-nya.

.NET Inti

Sebuah cross-platform, penerus open-source untuk .NET Framework, dirancang agar lebih modular dan ringan. .NET Core (sekarang hanya disebut .NET 5 dan yang lebih baru) memungkinkan pengembang untuk membangun aplikasi yang berjalan di Windows, Linux, dan macOS.

Saat memigrasikan aplikasi yang dibangun di.NET Core ke Elastic Beanstalk, Anda dapat memilih antara platform Windows Server atau platform berbasis Linux, tergantung pada persyaratan dan dependensi aplikasi Anda.

Runtime Bahasa Umum (CLR)

Komponen mesin virtual dari .NET Framework yang mengelola eksekusi program.NET. CLR menyediakan layanan seperti manajemen memori, keamanan tipe, penanganan pengecualian, pengumpulan sampah, dan manajemen benang.

Saat bermigrasi ke Elastic Beanstalk, versi CLR yang sesuai secara otomatis tersedia di platform Windows Server yang Anda pilih, memastikan kompatibilitas dengan persyaratan aplikasi Anda.

Situs

Wadah logis di IIS yang mewakili aplikasi atau layanan web, diidentifikasi dengan pengikatan unik alamat IP, port, dan header host. Setiap situs IIS memiliki kumpulan aplikasi, binding, dan pengaturan konfigurasi sendiri, dan dapat berisi satu atau lebih aplikasi.

Aplikasi

Pengelompokan file konten dan kode dalam situs IIS yang menangani permintaan untuk ruang URL tertentu. Aplikasi di IIS dapat memiliki pengaturan konfigurasi sendiri, yang biasanya disimpan dalam `web.config` file.

Saat bermigrasi ke Elastic Beanstalk, aplikasi dipertahankan dengan struktur jalur dan pengaturan konfigurasinya. Proses migrasi memastikan bahwa aplikasi bersarang mempertahankan hierarki dan jalur URL mereka di lingkungan cloud.

ApplicationPool

Fitur IIS yang mengisolasi aplikasi web untuk keamanan, keandalan, dan manajemen kinerja yang lebih baik. Kumpulan aplikasi menentukan lingkungan runtime untuk aplikasi, termasuk versi.NET Framework, mode pipeline, dan pengaturan identitas.

VirtualDirectory

Pemetaan direktori di IIS yang memungkinkan konten disajikan dari lokasi di luar direktori root situs. Direktori virtual memungkinkan Anda untuk mengatur konten di berbagai lokasi fisik sambil menyajikan struktur URL terpadu kepada pengguna.

Saat bermigrasi ke Elastic Beanstalk, direktori virtual dipertahankan dengan pemetaan jalurnya. eb migratePerintah membuat struktur direktori dan konfigurasi yang diperlukan di lingkungan cloud untuk mempertahankan jalur URL yang sama.

ARR

Application Request Routing, ekstensi IIS yang menyediakan load balancing dan kemampuan proxy untuk server web. ARR memungkinkan routing berbasis URL, penerusan permintaan HTTP, dan distribusi beban di beberapa server.

Selama migrasi ke Elastic Beanstalk, konfigurasi ARR dipertahankan melalui instalasi EC2 fitur ARR pada instance dan konfigurasi aturan routing yang sesuai. Untuk skenario perutean yang kompleks, proses migrasi juga dapat memanfaatkan aturan Application Load Balancer untuk mengimplementasikan fungsionalitas serupa.

Tulis Ulang URL

Modul IIS yang memodifikasi permintaan URLs berdasarkan aturan yang ditetapkan sebelum mencapai aplikasi web. URL Rewrite memungkinkan manipulasi URL, pengalihan, dan pengiriman konten berdasarkan pola dan kondisi.

Saat bermigrasi ke Elastic Beanstalk, aturan penulisan ulang URL `web.config` dari file Anda diterjemahkan ke dalam aturan perutean ALB jika memungkinkan, atau disimpan dalam konfigurasi IIS pada instance. EC2 Ini memastikan bahwa pola URL dan pengalihan terus berfungsi seperti yang diharapkan di lingkungan cloud.

msdeploy.exe

Alat baris perintah yang digunakan untuk menyebarkan aplikasi web dan situs web ke server IIS. Juga dikenal sebagai Web Deploy, ini menyediakan cara untuk mengemas, menyinkronkan, dan menyebarkan aplikasi web, situs web, dan konfigurasi server.

eb migratePerintah menggunakan Web Deploy (versi 3.6 atau yang lebih baru) untuk mengemas aplikasi Anda selama migrasi ke Elastic Beanstalk. Alat ini harus diinstal di server sumber Anda agar proses migrasi berfungsi dengan benar.

Jalur Fisik

Lokasi sistem file yang sebenarnya di mana situs IIS atau file konten aplikasi disimpan. Jalur fisik dapat menunjuk ke direktori lokal, berbagi jaringan, atau lokasi penyimpanan lain yang dapat diakses oleh server IIS.

Selama migrasi ke Elastic Beanstalk, jalur fisik dipetakan ke lokasi EC2 yang sesuai pada instance di lingkungan Anda. Proses migrasi mempertahankan struktur konten sambil memastikan bahwa semua file disebarkan dengan benar ke lingkungan cloud.

ApplicationHost.config

File konfigurasi root untuk IIS yang mendefinisikan pengaturan server-wide dan berisi konfigurasi untuk semua situs, aplikasi, dan direktori virtual. File ini terletak di `%windir%\System32\inetsrv\config` direktori dan mengontrol perilaku keseluruhan server IIS.

Saat bermigrasi ke Elastic Beanstalk, `applicationHost.config` pengaturan yang relevan dari diekstraksi dan diterapkan ke konfigurasi IIS pada instance di lingkungan Anda. EC2 Ini memastikan bahwa pengaturan seluruh server dipertahankan selama migrasi.

web.config

File konfigurasi berbasis XML yang digunakan dalam aplikasi ASP.NET untuk mengontrol pengaturan aplikasi, keamanan, dan perilaku di tingkat aplikasi atau direktori. `web.config`file dapat berisi pengaturan untuk otentikasi, otorisasi, status sesi, kompilasi, dan parameter aplikasi khusus.

Selama migrasi ke Elastic `web.config` Beanstalk, file disimpan dan digunakan dengan aplikasi Anda. Proses migrasi memastikan bahwa konfigurasi khusus aplikasi terus berfungsi seperti yang diharapkan di lingkungan cloud.

DefaultDocument

Fitur IIS yang menentukan file default untuk melayani ketika pengguna meminta direktori tanpa menentukan nama file. Dokumen default diaktifkan secara default, dan IIS 7 mendefinisikan file dokumen default berikut dalam `applicationHost.config` file sebagai default server-wide: Default.htm, Default.asp, Index.htm, Index.html, Iisstart.htm.

Saat bermigrasi ke Elastic Beanstalk, pengaturan dokumen default dipertahankan dalam konfigurasi IIS EC2 pada instance, memastikan bahwa permintaan direktori ditangani secara konsisten di lingkungan cloud.

Sistem.webserver

Bagian konfigurasi di `web.config` atau `applicationHost.config` yang berisi pengaturan khusus IIS untuk modul, penangan, dan perilaku server lainnya. Bagian ini mengontrol bagaimana IIS memproses permintaan, mengelola modul, dan mengkonfigurasi fitur server.

Selama migrasi ke Elastic Beanstalk, konfigurasi System.WebServer dipertahankan dalam file `web.config` aplikasi Anda dan diterapkan ke instalasi IIS pada instance di lingkungan Anda. EC2 Ini memastikan bahwa perilaku khusus IIS dipertahankan di lingkungan cloud.

Istilah Elastic Beanstalk

Platform

Kombinasi sistem operasi, runtime bahasa pemrograman, server web, server aplikasi, dan komponen Elastic Beanstalk yang menentukan tumpukan perangkat lunak untuk menjalankan aplikasi.

Untuk migrasi Windows, Elastic Beanstalk menyediakan platform berbasis Windows Server 2016, 2019, dan 2022 dengan IIS dan berbagai versi.NET Framework untuk memastikan kompatibilitas dengan lingkungan sumber Anda.

SolutionStack

Konfigurasi platform yang telah ditentukan sebelumnya di Elastic Beanstalk yang menentukan sistem operasi, runtime, dan komponen lain yang diperlukan untuk menjalankan aplikasi. Secara konseptual identik dengan platform dan digunakan secara bergantian untuk mengoperasikan lingkungan.

Selama migrasi, eb migrate perintah memilih tumpukan solusi yang sesuai berdasarkan konfigurasi lingkungan sumber Anda, memastikan kompatibilitas dengan aplikasi IIS Anda.

CreateEnvironment

Tindakan API Elastic Beanstalk yang menciptakan lingkungan baru untuk meng-host versi aplikasi. API ini digunakan oleh eb migrate perintah untuk menyediakan AWS sumber daya yang diperlukan untuk aplikasi yang dimigrasi.

Proses migrasi mengonfigurasi parameter lingkungan yang sesuai berdasarkan lingkungan IIS sumber Anda, termasuk jenis instans, variabel lingkungan, dan pengaturan opsi.

CreateApplicationVersion

Tindakan API Elastic Beanstalk yang membuat versi aplikasi baru dari bundel sumber yang disimpan di Amazon S3. eb migratePerintah menggunakan API ini untuk mendaftarkan aplikasi IIS paket Anda sebagai versi di Elastic Beanstalk.

Selama migrasi, file dan konfigurasi aplikasi Anda dikemas, diunggah ke Amazon S3, dan didaftarkan sebagai versi aplikasi sebelum penerapan.

DescribeEvents

Tindakan API Elastic Beanstalk yang mengambil daftar peristiwa untuk lingkungan, termasuk penerapan, perubahan konfigurasi, dan masalah operasional. eb migratePerintah menggunakan API ini untuk memantau kemajuan migrasi Anda.

Anda juga dapat menggunakan eb events perintah setelah migrasi untuk melihat riwayat peristiwa untuk lingkungan Anda.

DescribeEnvironmentHealth

Tindakan API Elastic Beanstalk yang memberikan informasi kesehatan terperinci tentang instance dan komponen lingkungan lainnya. API ini digunakan untuk memverifikasi kesehatan aplikasi yang dimigrasi setelah penerapan.

Setelah migrasi, Anda dapat menggunakan eb health perintah untuk memeriksa status lingkungan Anda dan mengidentifikasi masalah apa pun yang perlu diperhatikan.

HealthD

Agen pemantauan di Elastic Beanstalk yang mengumpulkan metrik, memantau log, dan melaporkan status kesehatan untuk kejadian di lingkungan. EC2 HealthD menyediakan pelaporan kesehatan yang disempurnakan untuk aplikasi yang dimigrasi.

Setelah migrasi, HealthD memantau kinerja aplikasi Anda, pemanfaatan sumber daya, dan tingkat keberhasilan permintaan, memberikan pandangan komprehensif tentang kesehatan lingkungan Anda.

Log Bundel

Fitur di Elastic Beanstalk yang mengompres dan mengunggah log EC2 dari instance ke Amazon S3 untuk penyimpanan dan analisis terpusat. Fitur ini membantu Anda memecahkan masalah dengan aplikasi yang dimigrasi.

Setelah migrasi, Anda dapat menggunakan eb logs perintah untuk mengambil dan melihat log dari lingkungan Anda.

aws-windows-deployment-manifest.json

File yang menjelaskan isi, dependensi, dan konfigurasi paket perangkat lunak atau aplikasi. Manifes ini dihasilkan selama proses migrasi untuk menentukan bagaimana aplikasi IIS Anda harus diterapkan di Elastic Beanstalk.

bagian manifes kustom

Bagian di dalamnya `aws-windows-deployment-manifest.json` menyediakan kontrol khusus atas penerapan aplikasi. Bagian ini berisi PowerShell skrip dan perintah yang dijalankan selama proses penyebaran.

Selama migrasi, bagian manifes kustom dibuat untuk menangani aspek tertentu dari konfigurasi IIS Anda, seperti penyiapan direktori virtual, manajemen izin, dan konfigurasi kumpulan aplikasi.

CLI EB

Alat baris perintah yang menyediakan perintah untuk membuat, mengonfigurasi, dan mengelola aplikasi dan lingkungan Elastic Beanstalk. EB CLI menyertakan perintah khusus untuk memigrasikan eb migrate aplikasi IIS ke Elastic Beanstalk.

Setelah migrasi, Anda dapat terus menggunakan CLI EB untuk mengelola lingkungan Anda, menyebarkan pembaruan, memantau kesehatan, dan melakukan tugas administratif lainnya.

Pengaturan opsi

Nilai konfigurasi yang menentukan cara Elastic Beanstalk menyediakan AWS dan mengonfigurasi sumber daya di lingkungan Anda. Pengaturan opsi diatur ke dalam ruang nama yang mewakili berbagai komponen lingkungan Anda, seperti penyeimbang beban, instance, dan proses lingkungan.

Selama migrasi, eb migrate perintah menghasilkan pengaturan opsi yang sesuai berdasarkan konfigurasi IIS Anda untuk memastikan bahwa lingkungan cloud Anda sesuai dengan kemampuan lingkungan sumber Anda. Untuk informasi selengkapnya, lihat Opsi konfigurasi di Panduan Pengembang Elastic Beanstalk.

aws:elbv2:listener:default

Namespace konfigurasi Elastic Beanstalk untuk pendengar default pada Application Load Balancer. Selama migrasi, namespace ini dikonfigurasi berdasarkan binding situs IIS Anda untuk memastikan perutean lalu lintas yang tepat.

Pendengar default biasanya menangani lalu lintas HTTP pada port 80, yang kemudian diteruskan ke instance aplikasi Anda sesuai dengan aturan perutean.

aws:elbv2:listener:listener_port

Namespace konfigurasi Elastic Beanstalk untuk port listener tertentu pada Application Load Balancer. Namespace ini digunakan untuk mengonfigurasi pendengar tambahan untuk aplikasi yang dimigrasi, seperti HTTPS pada port 443.

Selama migrasi, pendengar dibuat berdasarkan pengikatan port situs IIS Anda, memastikan bahwa aplikasi Anda tetap dapat diakses di port yang sama seperti di lingkungan sumber Anda.

aws:elbv2:listenerrule:rule_name

Namespace konfigurasi Elastic Beanstalk untuk mendefinisikan aturan routing untuk pendengar Application Load Balancer. Aturan ini menentukan bagaimana permintaan masuk dirutekan ke grup target yang berbeda berdasarkan pola jalur atau header host.

Selama migrasi, aturan listener dibuat agar sesuai dengan struktur URL aplikasi IIS Anda, memastikan bahwa permintaan dirutekan ke jalur aplikasi yang benar.

aws:elasticbeanstalk:lingkungan: proses:default

Namespace konfigurasi Elastic Beanstalk untuk proses default di lingkungan. Namespace ini mendefinisikan bagaimana proses aplikasi web default dikonfigurasi, termasuk pengaturan pemeriksaan kesehatan, pemetaan port, dan pengaturan proxy.

Selama migrasi, proses default dikonfigurasi berdasarkan pengaturan situs IIS utama Anda, memastikan pemantauan kesehatan yang tepat dan penanganan permintaan.

aws:elasticbeanstalk: lingkungan: proses: process_name

Namespace konfigurasi Elastic Beanstalk untuk proses bernama tertentu di lingkungan. Namespace ini memungkinkan Anda untuk menentukan beberapa proses dengan konfigurasi yang berbeda, mirip dengan memiliki beberapa kumpulan aplikasi di IIS.

Selama migrasi, proses tambahan dapat dibuat untuk mewakili binding situs yang berbeda dari lingkungan sumber Anda.

> **Note**
>
> Untuk informasi selengkapnya tentang beberapa istilah yang dijelaskan dalam topik ini, lihat sumber daya berikut:

- Tindakan API Elastic Beanstalk - Referensi API AWS Elastic Beanstalk
- Platform Elastic Beanstalk, termasuk versi platform yang didukung - Platform yang Didukung dalam panduan Platform AWS Elastic Beanstalk
- Ruang nama konfigurasi Elastic Beanstalk - dalam panduan ini Opsi umum untuk semua lingkungan
- EB CLI atau perintah EB CLI tertentu - dalam panduan ini Menyiapkan antarmuka baris perintah EB (EB CLI) untuk mengelola Elastic Beanstalk

Istilah Python

pip

Penginstal paket untuk Python, digunakan untuk menginstal dan mengelola paket perangkat lunak yang ditulis dengan Python. CLI EB diinstal dan diperbarui menggunakan pip.

Selama proses migrasi, pip digunakan untuk menginstal paket EB CLI dan dependensinya di server sumber Anda, menyediakan alat yang diperlukan untuk migrasi.

PyPI

Python Package Index, repositori resmi untuk paket perangkat lunak Python pihak ketiga, dari mana pip mengambil dan menginstal paket. CLI EB dan dependensinya di-host di PyPI.

Saat menginstal CLI EB untuk migrasi, pip terhubung ke PyPI untuk mengunduh dan menginstal paket yang diperlukan.

virtualenv

Alat untuk membuat lingkungan Python yang terisolasi, memungkinkan proyek yang berbeda untuk memiliki dependensi dan paket mereka sendiri tanpa konflik. Menggunakan virtualenv dianjurkan ketika menginstal EB CLI untuk menghindari konflik dengan aplikasi Python lainnya.

Membuat lingkungan virtual sebelum menginstal EB CLI memastikan bahwa alat migrasi memiliki lingkungan yang bersih dan terisolasi dengan dependensi yang benar.

pywin32

Satu set ekstensi Python yang menyediakan akses ke banyak Windows APIs, memungkinkan interaksi dengan sistem operasi Windows dan komponennya. CLI EB menggunakan pywin32 untuk berinteraksi dengan fitur khusus Windows selama migrasi.

Selama proses migrasi, pywin32 digunakan untuk mengakses konfigurasi IIS, pengaturan registri Windows, dan informasi sistem lain yang diperlukan untuk mengemas dan memigrasi aplikasi Anda dengan benar.

pythonnet

Sebuah paket yang memungkinkan kode Python untuk berinteraksi dengan aplikasi.NET Framework dan .NET Core. Integrasi ini memungkinkan CLI EB untuk bekerja dengan komponen.NET selama proses migrasi.

Proses migrasi dapat menggunakan pythonnet untuk berinteraksi dengan rakitan dan komponen .NET saat menganalisis dan mengemas aplikasi Anda untuk penyebaran ke Elastic Beanstalk.

Melakukan migrasi IIS dasar

Bagian ini memandu Anda melalui proses migrasi aplikasi IIS Anda ke Elastic Beanstalk menggunakan perintah. eb migrate

Menjelajahi lingkungan IIS Anda

Sebelum membuat perubahan apa pun, Anda harus memahami sumber daya apa yang ada di server Anda. Mulailah dengan menjelajahi situs IIS Anda dengan menjalankaneb migrate explore, seperti yang ditunjukkan pada contoh berikut:

```
PS C:\migrations_workspace> eb migrate explore
```

Perintah ini mengungkapkan situs IIS Anda. Lihat daftar berikut:

```
Default Web Site
Intranet
API.Internal
Reports
```

Untuk tampilan terperinci dari setiap konfigurasi situs, termasuk binding, aplikasi, dan direktori virtual, tambahkan `--verbose` opsi, seperti yang ditunjukkan dalam contoh ini:

```
PS C:\migrations_workspace> eb migrate explore --verbose
```

Daftar berikut menunjukkan informasi komprehensif tentang lingkungan Anda yang disediakan perintah:

```
1: Default Web Site:
  - Bindings:
    - *:80:www.example.com
    - *:443:www.example.com
  - Application '/':
    - Application Pool: DefaultAppPool
    - Enabled Protocols: http
    - Virtual Directories:
      - /:
        - Physical Path: C:\inetpub\wwwroot
        - Logon Method: ClearText
  - Application '/api':
    - Application Pool: ApiPool
    - Enabled Protocols: http
    - Virtual Directories:
      - /:
        - Physical Path: C:\websites\api
        - Logon Method: ClearText
2: Intranet:
...
3. API.Internal:
...
4. Reports:
...
```

Memahami output penemuan

Output verbose memberikan informasi penting berikut untuk perencanaan migrasi:

Situs

Output penemuan mencantumkan semua situs IIS di server Anda. Setiap situs diidentifikasi dengan namanya (misalnya, “Situs Web Default”, “Intranet”, “API.internal”) dan diberi nomor secara berurutan. Ketika beberapa situs ada di server, `eb migrate` perintah dapat mengemas dan menyebarkan masing-masing secara terpisah atau bersama-sama, tergantung pada strategi migrasi Anda.

Binding

Pengikatan protokol mengungkapkan protokol mana (HTTP/HTTPS) yang digunakan situs Anda dan port mana yang mereka operasikan. Informasi yang mengikat mencakup persyaratan header host yang menentukan konfigurasi perutean berbasis domain.

Aplikasi

Jalur aplikasi menunjukkan struktur aplikasi root dan bersarang dalam konfigurasi IIS Anda. Penugasan kumpulan aplikasi menunjukkan bagaimana aplikasi Anda diisolasi satu sama lain untuk keamanan dan manajemen sumber daya.

Direktori virtual

Pemetaan jalur fisik menunjukkan di mana konten Anda berada di sistem file. Pengaturan otentikasi menunjukkan persyaratan akses khusus yang perlu dipertahankan setelah migrasi.

Mempersiapkan migrasi

Dengan pemahaman tentang lingkungan Anda, pastikan bahwa server Anda memenuhi prasyarat. Pertama, verifikasi versi IIS Anda dengan PowerShell perintah berikut:

```
PS C:\migrations_workspace> Get-ItemProperty "HKLM:\SOFTWARE\Microsoft\InetStp\" -Name
 MajorVersion
```

Anda membutuhkan IIS 7.0 atau yang lebih baru. Alat migrasi menggunakan Web Deploy 3.6 untuk mengemas aplikasi Anda. Verifikasi instalasinya dengan perintah berikut:

```
PS C:\migrations_workspace> Get-ItemProperty "HKLM:\SOFTWARE\Microsoft\IIS Extensions
\MSDeploy\3" -Name InstallPath
```

Jika Web Deploy tidak diinstal di server Anda, Anda dapat mengunduhnya dari halaman unduhan Microsoft Web Platform Installer.

Migrasi pertama Anda

Mari kita mulai dengan migrasi dasar dari Situs Web Default. Contoh berikut menunjukkan perintah yang paling sederhana,eb migrate.

```
PS C:\migrations_workspace> eb migrate
```

Perintah ini memulai serangkaian langkah otomatis, yang ditunjukkan dalam contoh output berikut:

```
Identifying VPC configuration of this EC2 instance (i-0123456789abcdef0)
  id: vpc-1234567890abcdef0
  publicip: true
  elbscheme: public
  ec2subnets: subnet-123,subnet-456,subnet-789
  securitygroups: sg-123,sg-456
  elbsubnets: subnet-123,subnet-456,subnet-789

Using .\migrations\latest to contain artifacts for this migration run.
Generating source bundle for sites, applications, and virtual directories...
  Default Web Site/ -> .\migrations\latest\upload_target\DefaultWebSite.zip
```

Alat migrasi membuat direktori terstruktur yang berisi artefak penerapan Anda. Daftar berikut menunjukkan struktur direktori:

```
C:\migration_workspace\
### .\migrations\latest\
    ### upload_target\
        ### DefaultWebSite.zip
        ### aws-windows-deployment-manifest.json
        ### ebmigrateScripts\
            ### site_installer.ps1
            ### permission_handler.ps1
            ### >other helper scripts<
```

Mengontrol migrasi

Untuk kontrol lebih lanjut atas proses migrasi, Anda dapat menentukan dengan tepat situs mana yang akan dimigrasi dengan perintah berikut:

```
PS C:\migrations_workspace> eb migrate --sites "Default Web Site,Intranet"
```

Anda juga dapat menyesuaikan nama lingkungan dan nama aplikasi, seperti yang ditunjukkan pada contoh perintah berikut:

```
PS C:\migrations_workspace> eb migrate `
    --sites "Default Web Site" `
    --application-name "CorporateApp" `
    --environment-name "Production"
```

Untuk daftar lengkap opsi, lihatthe section called “eb migrate”.

Memantau progres

Selama migrasi, eb migrate menyediakan pembaruan status waktu nyata. Lihat contoh keluaran berikut:

```
...
Creating application version
Creating environment... This may take a few minutes

2024-03-18 18:12:15    INFO    Environment details for: Production
  Application name: CorporateApp
  Region: us-west-2
  Deployed Version: app-230320_153045
  Environment ID: e-abcdef1234
  Platform: 64bit Windows Server 2019 v2.7.0 running IIS 10.0
  Tier: WebServer-Standard-1.0
  CNAME: production.us-west-2.elasticbeanstalk.com
  Updated: 2024-03-20 15:30:45
2025-03-18 18:12:17    INFO    createEnvironment is starting.
2025-03-18 18:12:19    INFO    Using elasticbeanstalk-us-east-1-180301529717 as Amazon
 S3 storage bucket for environment data.
2025-03-18 18:12:40    INFO    Created security group named: sg-0fdd4d696a26b086a
2025-03-18 18:12:48    INFO    Environment health has transitioned to Pending.
 Initialization in progress (running for 7 seconds). There are no instances.
...
2025-03-18 18:23:59    INFO    Application available at EBMigratedEnv-arrreal3.us-
east-1.elasticbeanstalk.com.
2025-03-18 18:24:00    INFO    Successfully launched environment: EBMigratedEnv-
arrreal3
```

Memverifikasi migrasi

Setelah lingkungan siap, Elastic Beanstalk menyediakan beberapa cara untuk memverifikasi penerapan Anda.

Akses aplikasi Anda

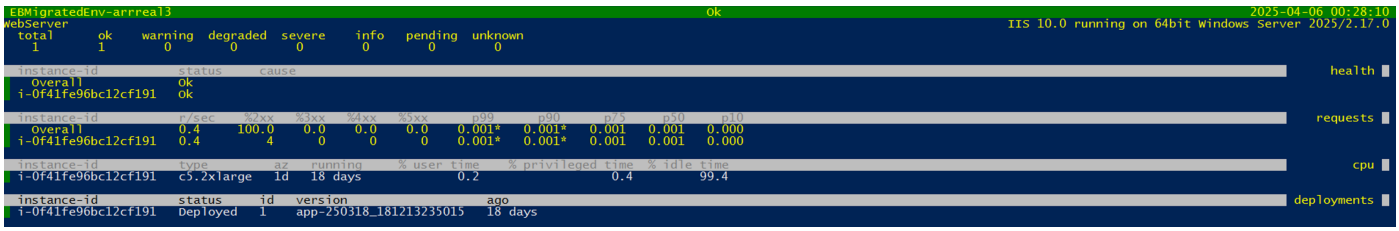
Buka URL aplikasi Anda (CNAME) di browser web untuk memverifikasi itu berfungsi dengan benar.

Periksa kesehatan lingkungan

Gunakan eb health perintah untuk melihat kesehatan lingkungan Anda.

```
PS C:\migrations_workspace> eb health
```

Gambar layar berikut menunjukkan kesehatan instance, metrik respons aplikasi, dan pemanfaatan sumber daya sistem.



Gunakan eb logs perintah untuk mengakses log untuk memecahkan masalah apa pun:

```
PS C:\migrations_workspace> eb logs --zip
```

eb logsPerintah mengunduh log ke `.elasticbeanstalk/logs` direktori. Untuk informasi selengkapnya, lihat the section called “CloudWatch Log”.

Connect ke instance

Jika Anda menetapkan key pair selama migrasi, Anda dapat menyambung ke instans menggunakan RDP untuk pemecahan masalah langsung.

Akses konsol Elastic Beanstalk

Anda dapat melihat kesehatan lingkungan, log, dan properti konfigurasi melalui konsol manajemen lingkungan untuk lingkungan tersebut.

Mengelola artefak migrasi

eb migratePerintah membuat artefak lokal selama proses migrasi. Artefak ini berisi informasi sensitif dan dapat menghabiskan ruang disk yang signifikan dari waktu ke waktu. Gunakan cleanup subperintah untuk mengelola artefak ini, seperti yang ditunjukkan pada contoh berikut:

```
PS C:\migrations_workspace> eb migrate cleanup
Are you sure you would like to cleanup older artifacts within ./migrations/? (Y/N):
```

Untuk memaksa pembersihan tanpa konfirmasi gunakan --force opsi:

```
PS C:\migrations_workspace> eb migrate cleanup --force
```

Proses pembersihan mempertahankan migrasi sukses terbaru di `./migrations/latest` direktori dan menghapus direktori migrasi yang lebih lama

Konfigurasi jaringan dan pengaturan port

Bagian ini mencakup opsi konfigurasi jaringan untuk migrasi IIS, termasuk pengaturan VPC, konfigurasi port, dan penerapan multi-situs.

Konfigurasi VPC

eb migratePerintah ini menyediakan opsi konfigurasi VPC yang fleksibel untuk lingkungan Elastic Beanstalk Anda. Alat ini dapat mendeteksi pengaturan VPC dari EC2 instance sumber atau menerima konfigurasi VPC khusus melalui parameter baris perintah. Tinjau Menggunakan Elastic Beanstalk dengan Amazon VPC untuk memahami cara mengkonfigurasi Elastic Beanstalk dengan VPC.

Deteksi VPC otomatis

Ketika eb migrate berjalan pada sebuah EC2 instance, secara otomatis menemukan dan menggunakan konfigurasi VPC dari instance lingkungan EC2 sumber. Contoh output berikut menggambarkan informasi konfigurasi yang dideteksi:

```
PS C:\migrations_workspace > eb migrate
Identifying VPC configuration of this EC2 instance (i-0123456789abcdef0):
  id: vpc-1234567890abcdef0
  publicip: true
  elbscheme: public
  ec2subnets: subnet-123,subnet-456,subnet-789
  securitygroups: sg-123,sg-456
  elbsubnets: subnet-123,subnet-456,subnet-789
...
```

Konfigurasi yang terdeteksi meliputi:

- Pengidentifikasi VPC
- Pengaturan penetapan IP publik
- Skema penyeimbang beban (publik/swasta)

- EC2 penugasan subnet contoh
- Asosiasi kelompok keamanan
- Tugas subnet penyeimbang beban

Host lokal atau AWS non-cloud

Saat eb migrate dijalankan dari server lokal atau host AWS non-cloud, layanan Elastic Beanstalk akan menggunakan VPC default di akun Anda. AWS Daftar berikut menunjukkan contoh perintah dan output:

```
PS C:\migrations_worspace> eb migrate `
      -k windows-test-pem `
      --region us-east-1 `
      -a EBMigratedEnv `
      -e EBMigratedEnv-test2 `
      --copy-firewall-config
Determining EB platform based on host machine properties
Using .\migrations\latest to contain artifacts for this migration run.
...
```

Tinjau Menggunakan Elastic Beanstalk dengan Amazon VPC untuk memahami bagaimana Elastic Beanstalk mengonfigurasi VPC default untuk lingkungan Anda.

Konfigurasi VPC kustom

Untuk lingkungan sumber apa pun (EC2, lokal, atau AWS non-cloud) yang memerlukan pengaturan VPC tertentu, sediakan file konfigurasi VPC seperti yang ada di contoh berikut:

```
{
    "id": "vpc-12345678",
    "publicip": "true",
    "elbscheme": "public",
    "ec2subnets": ["subnet-a1b2c3d4", "subnet-e5f6g7h8"],
    "securitygroups": "sg-123456,sg-789012",
    "elbsubnets": ["subnet-a1b2c3d4", "subnet-e5f6g7h8"]
}
```

Terapkan konfigurasi ini menggunakan perintah berikut:

```
PS C:\migrations_workspace> eb migrate --vpc-config vpc-config.json
```

> **Note**
>
> File konfigurasi VPC memerlukan `id` bidang yang menentukan ID VPC. Semua bidang lainnya bersifat opsional, dan Elastic Beanstalk akan menggunakan nilai default untuk bidang apa pun yang tidak Anda tentukan.

> **Important**
>
> Migrasi akan mengabaikan pengaturan VPC yang ada dari lingkungan sumber saat Anda menentukan parameter. `--vpc-config` Saat Anda menggunakan parameter ini, migrasi hanya akan menggunakan pengaturan VPC yang ditentukan dalam file konfigurasi yang Anda lewati. Menggunakan parameter ini mengesampingkan perilaku default untuk menemukan konfigurasi VPC instance sumber atau menggunakan VPC default.

Gunakan `--vpc-config` parameter dalam skenario ini:

- Saat Anda memigrasikan EC2 lingkungan non-lingkungan yang tidak memiliki pengaturan VPC yang dapat ditemukan
- Saat Anda bermigrasi ke VPC yang berbeda dari yang digunakan oleh lingkungan sumber
- Saat Anda perlu menyesuaikan pilihan subnet atau konfigurasi grup keamanan
- Ketika penemuan otomatis tidak mengidentifikasi pengaturan VPC yang diinginkan dengan benar
- Saat Anda bermigrasi dari lokal, dan Anda tidak ingin menggunakan VPC default

Konfigurasi keamanan jaringan

Secara default, eb migrate membuka port 80 pada instance target tetapi tidak menyalin aturan Windows Firewall lainnya dari mesin sumber. Untuk menyertakan semua konfigurasi firewall gunakan perintah berikut:

```
PS C:\migrations_workspace> eb migrate --copy-firewall-config
```

Perintah ini melakukan tindakan berikut:

- Mengidentifikasi port yang digunakan oleh binding situs IIS

- Mengambil aturan firewall yang sesuai
- Menghasilkan PowerShell skrip untuk membuat ulang aturan pada instance target
- Mempertahankan aturan DENY untuk port 80 dari mesin sumber (jika tidak port 80 diizinkan secara default)

Pertimbangkan kasus penggunaan, di mana mesin sumber Anda memiliki aturan firewall yang ditentukan dalam contoh berikut:

```
# Source machine firewall configuration
Get-NetFirewallRule | Where-Object {$_.Enabled -eq 'True'} | Get-NetFirewallPortFilter
 | Where-Object {$_.LocalPort -eq 80 -or $_.LocalPort -eq 443 -or $_.LocalPort -eq
 8081}
# Output shows rules for ports 80, 443, and 8081
```

Migrasi membuat script (`modify_firewall_config.ps1`) yang berisi konfigurasi berikut:

```
New-NetFirewallRule -DisplayName "Allow Web Traffic" -Direction Inbound -Action Allow -
Protocol TCP -LocalPort 80,443
New-NetFirewallRule -DisplayName "Allow API Traffic" -Direction Inbound -Action Allow -
Protocol TCP -LocalPort 8081
```

Alat migrasi secara otomatis melakukan tindakan berikut:

- Mengekstrak port HTTP/HTTPS dari semua binding situs IIS
- Menggunakan antarmuka Windows Firewall INetFwPolicy2 untuk menghitung aturan firewall
- Menyaring aturan untuk hanya menyertakan aturan yang secara eksplisit mereferensikan port yang ditentukan
- Hanya memproses binding situs HTTP dan HTTPS dan aturan firewall yang terkait
- Mempertahankan properti aturan termasuk nama tampilan, tindakan, protokol, dan status diaktifkan
- Menangani port individu dan rentang port dalam aturan firewall
- Menambahkan skrip konfigurasi firewall ke manifes penerapan

Konfigurasi penyeimbang beban

Anda dapat menentukan konfigurasi Load Balancer melalui argumen. `--vpc-config` Contoh-contoh berikut menunjukkan parameter.

Pemilihan skema

Pilih antara skema penyeimbang beban publik dan pribadi:

```
{
    "id": "vpc-12345678",
    "elbscheme": "private",
    "elbsubnets": ["subnet-private1", "subnet-private2"]
}
```

Distribusi subnet

Untuk ketersediaan tinggi, distribusikan subnet penyeimbang beban di seluruh Availability Zone:

```
{
    "elbsubnets": [
        "subnet-az1", // Availability Zone 1
        "subnet-az2", // Availability Zone 2
        "subnet-az3"  // Availability Zone 3
    ]
}
```

> **Note**
>
> Sementara Elastic Beanstalk mendukung pembuatan lingkungan dengan Application Load Balancers, Network Load Balancers, dan Classic Load Balancers, perintah ini hanya mendukung Application Load Balancers. eb migrate Untuk informasi selengkapnya tentang tipe penyeimbang beban, lihat Penyeimbang beban untuk lingkungan Elastic Beanstalk Anda.

Penerapan multi-situs dengan konfigurasi port

eb migratePerintah menangani penyebaran IIS multi-situs yang kompleks di mana aplikasi mungkin berbagi dependensi atau menggunakan port non-standar. Pertimbangkan contoh berikut dari pengaturan perusahaan tipikal dengan beberapa situs:

```
<!-- IIS Configuration -->
<sites>
    <site name="Default Web Site" id="1">
        <bindings>
```

```
            <binding protocol="http" bindingInformation="*:80:www.example.com" />
        </bindings>
    </site>
    <site name="InternalAPI" id="2">
        <bindings>
            <binding protocol="http" bindingInformation="*:8081:api.internal" />
        </bindings>
    </site>
    <site name="ReportingPortal" id="3">
        <bindings>
            <binding protocol="http" bindingInformation="*:8082:reports.internal" />
        </bindings>
    </site>
</sites>
```

Untuk memigrasikan konfigurasi ini, gunakan contoh perintah dan parameter berikut:

```
PS C:\migrations_workspace> eb migrate `
    --sites "Default Web Site,InternalAPI,ReportingPortal" `
    --copy-firewall-config `
    --instance-type "c5.large"
```

eb migratePerintah membuat paket penyebaran yang mempertahankan identitas dan konfigurasi setiap situs. Perintah menghasilkan sebuah `aws-windows-deployment-manifest.json` yang mendefinisikan bagaimana situs-situs ini harus digunakan. Contoh berikut menunjukkan file json dihasilkan:

```
{
    "manifestVersion": 1,
    "deployments": {
        "msDeploy": [
            {
                "name": "DefaultWebSite",
                "parameters": {
                    "appBundle": "DefaultWebSite.zip",
                    "iisPath": "/",
                    "iisWebSite": "Default Web Site"
                }
            }
        ],
        "custom": [
            {
```

```
                "name": "InternalAPI",
                "scripts": {
                    "install": {
                        "file": "ebmigrateScripts\\install_site_InternalAPI.ps1"
                    },
                    "restart": {
                        "file": "ebmigrateScripts\\restart_site_InternalAPI.ps1"
                    },
                    "uninstall": {
                        "file": "ebmigrateScripts\\uninstall_site_InternalAPI.ps1"
                    }
                }
            },
            {
                "name": "ReportingPortal",
                "scripts": {
                    "install": {
                        "file": "ebmigrateScripts\\install_site_ReportingPortal.ps1"
                    },
                    "restart": {
                        "file": "ebmigrateScripts\\restart_site_ReportingPortal.ps1"
                    },
                    "uninstall": {
                        "file": "ebmigrateScripts\\uninstall_site_ReportingPortal.ps1"
                    }
                }
            }
        ]
    }
}
```

Proses migrasi membuat aturan pendengar Application Load Balancer berikut yang mempertahankan logika perutean asli Anda:

- Port 80 rute lalu lintas ke Situs Web Default
- Rute lalu lintas Port 8081 ke InternalAPI
- Rute lalu lintas Port 8082 ke ReportingPortal

Konfigurasi dan dependensi bersama

Ketika situs berbagi konfigurasi atau dependensi, eb migrate menangani hubungan ini dengan tepat. Referensi contoh berikut di mana beberapa situs berbagi konfigurasi umum:

```
<!-- Shared configuration in applicationHost.config -->
<location path="Default Web Site">
    <system.webServer>
        <asp enableSessionState="true" />
        <caching enabled="true" enableKernelCache="true" />
    </system.webServer>
</location>
```

Proses migrasi menyelesaikan tugas-tugas berikut:

1. Mengidentifikasi konfigurasi bersama di seluruh situs
2. Menghasilkan PowerShell skrip yang sesuai untuk menerapkan pengaturan ini
3. Mempertahankan hierarki konfigurasi dan pewarisan

Praktik terbaik

Kami menyarankan Anda mengikuti praktik terbaik untuk konfigurasi jaringan aplikasi yang dimigrasi. Pengelompokan berikut memberikan panduan ringkasan.

Desain VPC

- Ikuti AWS Praktik Terbaik Desain VPC
- Gunakan subnet terpisah untuk penyeimbang beban dan instance EC2
- Menerapkan tabel rute yang tepat dan NACLs
- Pertimbangkan titik akhir VPC untuk layanan AWS

Ketersediaan tinggi

- Terapkan di beberapa Availability Zone
- Gunakan setidaknya dua subnet untuk penyeimbang beban
- Konfigurasikan auto-scaling di seluruh AZs
- Menerapkan pemeriksaan kesehatan yang tepat

Keamanan

- Ikuti Praktik Terbaik Keamanan

- Gunakan grup keamanan sebagai kontrol akses utama
- Menerapkan Jaringan Access Control Lists (ACLs) untuk keamanan tambahan
- Pantau Log Aliran VPC

Pemecahan Masalah

Masalah konfigurasi jaringan yang umum termasuk area berikut. Mengikuti setiap subjek adalah contoh perintah untuk mendapatkan informasi lebih lanjut tentang konfigurasi jaringan dan kesehatan lingkungan Anda.

Konfigurasi subnet

```
# Verify subnet availability
PS C:\migrations_workspace> aws ec2 describe-subnets --subnet-ids subnet-id

# Check available IP addresses
PS C:\migrations_workspace>aws ec2 describe-subnets --subnet-ids subnet-id --query
 'Subnets[].AvailableIpAddressCount'
```

Akses grup keamanan

```
# Verify security group rules
PS C:\migrations_workspace> aws ec2 describe-security-groups --group-ids sg-id

# Test network connectivity
PS C:\migrations_workspace> aws ec2 describe-network-interfaces --filters
 Name=group-id,Values=sg-id
```

Kesehatan penyeimbang beban

```
# Check load balancer health
PS C:\migrations_workspace> aws elbv2 describe-target-health --target-group-arn
 arn:aws:elasticloadbalancing:region:account-id:targetgroup/group-name/group-id
```

Konfigurasi keamanan dan peran IAM

eb migratePerintah mengelola konfigurasi AWS keamanan melalui peran IAM, profil instance, dan peran layanan. Memahami komponen-komponen ini memastikan kontrol akses yang tepat dan kepatuhan keamanan selama migrasi.

Konfigurasi profil instance

Profil instans berfungsi sebagai wadah untuk peran IAM yang dilampirkan Elastic Beanstalk EC2 ke instance di lingkungan Anda. Saat mengeksekusieb migrate, Anda dapat menentukan profil instance khusus:

```
PS C:\migrations_workspace> eb migrate --instance-profile "CustomInstanceProfile"
```

Jika Anda tidak menentukan profil eb migrate instance, buat profil default dengan izin berikut:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:GetObjectVersion",
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::elasticbeanstalk-*",
                "arn:aws:s3:::elasticbeanstalk-*/*"
            ]
        }
    ]
}
```

Manajemen peran layanan

Peran layanan memungkinkan Elastic Beanstalk mengelola sumber daya atas nama AWS Anda. Tentukan peran layanan kustom selama migrasi dengan perintah berikut:

```
PS C:\migrations_workspace> eb migrate --service-role "CustomServiceRole"
```

Jika tidak ditentukan, eb migrate buat peran layanan default yang diberi nama `aws-elasticbeanstalk-service-role` dengan kebijakan kepercayaan yang memungkinkan Elastic Beanstalk untuk mengambil peran tersebut. Peran layanan ini sangat penting bagi Elastic Beanstalk untuk memantau kesehatan lingkungan Anda dan melakukan pembaruan platform terkelola. Peran layanan memerlukan dua kebijakan terkelola:

- `AWSElasticBeanstalkEnhancedHealth`- Memungkinkan Elastic Beanstalk untuk memantau instans dan kesehatan lingkungan menggunakan sistem pelaporan kesehatan yang ditingkatkan
- `AWSElasticBeanstalkManagedUpdates`- Memungkinkan Elastic Beanstalk untuk melakukan pembaruan platform terkelola, termasuk memperbarui sumber daya lingkungan saat versi platform baru tersedia

Dengan kebijakan ini, peran layanan memiliki izin untuk:

- Membuat dan mengelola grup Auto Scaling
- Membuat dan mengelola Application Load Balancers
- Unggah log ke Amazon CloudWatch
- Kelola EC2 instance

Untuk informasi selengkapnya tentang peran layanan, lihat Peran layanan Elastic Beanstalk di Panduan Pengembang Elastic Beanstalk.

Konfigurasi grup keamanan

eb migratePerintah secara otomatis mengkonfigurasi grup keamanan berdasarkan binding situs IIS Anda. Misalnya, jika lingkungan sumber Anda memiliki situs yang menggunakan port 80, 443, dan 8081 hasil konfigurasi berikut:

```
<site name="Default Web Site">
    <bindings>
        <binding protocol="http" bindingInformation="*:80:" />
        <binding protocol="https" bindingInformation="*:443:" />
    </bindings>
</site>
<site name="InternalAPI">
    <bindings>
        <binding protocol="http" bindingInformation="*:8081:" />
    </bindings>
```

```
</site>
```

Proses migrasi menyelesaikan tindakan berikut:

- Membuat grup keamanan penyeimbang beban yang memungkinkan lalu lintas masuk pada port 80 dan 443 dari internet (0.0.0.0/0)
- Membuat grup EC2 keamanan yang memungkinkan lalu lintas dari penyeimbang beban
- Mengkonfigurasi port tambahan (seperti 8081) jika ditentukan `--copy-firewall-config`

Secara default, Application Load Balancer dikonfigurasi dengan akses publik dari internet. Jika Anda perlu menyesuaikan perilaku ini, seperti membatasi akses ke rentang IP tertentu atau menggunakan penyeimbang beban pribadi, Anda dapat mengganti konfigurasi VPC dan grup keamanan default menggunakan parameter: `--vpc-config`

```
PS C:\migrations_workspace> eb migrate --vpc-config vpc-config.json
```

Misalnya, `vpc-config.json` konfigurasi berikut membuat penyeimbang beban pribadi di subnet pribadi:

```
{
    "id": "vpc-12345678",
    "publicip": "false",
    "elbscheme": "internal",
    "ec2subnets": ["subnet-private1", "subnet-private2"],
    "elbsubnets": ["subnet-private1", "subnet-private2"]
}
```

Untuk informasi selengkapnya tentang opsi konfigurasi VPC, lihat. Konfigurasi VPC

Integrasi sertifikat SSL

Saat memigrasi situs dengan binding HTTPS, integrasikan sertifikat SSL melalui AWS Certificate Manager (ACM):

```
PS C:\migrations_workspace> eb migrate --ssl-certificates
 "arn:aws:acm:region:account:certificate/certificate-id"
```

Konfigurasi ini melengkapi tindakan berikut:

- Mengaitkan sertifikat dengan Application Load Balancer
- Mempertahankan penghentian HTTPS di penyeimbang beban
- Mempertahankan komunikasi HTTP internal antara penyeimbang beban dan instance EC2

Otentikasi Windows

Untuk aplikasi yang menggunakan Windows Authentication, eb migrate mempertahankan pengaturan otentikasi dalam aplikasi sebagai berikut: `web.config`

```
<configuration>
    <system.webServer>
        <security>
            <authentication>
                <windowsAuthentication enabled="true">
                    <providers>
                        <add value="Negotiate" />
                        <add value="NTLM" />
                    </providers>
                </windowsAuthentication>
            </authentication>
        </security>
    </system.webServer>
</configuration>
```

> **Important**
>
> eb migratePerintah tidak menyalin profil pengguna atau akun dari lingkungan sumber Anda ke instance Elastic Beanstalk target. Akun atau grup pengguna kustom apa pun yang telah Anda buat di server sumber Anda perlu dibuat ulang di lingkungan target setelah migrasi.

Akun Windows bawaan seperti `IUSR` dan grup sepertiIIS_IUSRS, serta semua akun dan grup bawaan lainnya, disertakan secara default pada instance Windows Server target. Untuk informasi selengkapnya tentang akun dan grup IIS bawaan, lihat Memahami Akun Pengguna dan Grup Bawaan di IIS dalam dokumentasi Microsoft.

Jika aplikasi Anda bergantung pada akun pengguna Windows kustom atau integrasi Active Directory, Anda perlu mengonfigurasi aspek-aspek ini secara terpisah setelah migrasi selesai.

Praktik terbaik dan pemecahan masalah

Manajemen peran

Terapkan praktik terbaik AWS IAM saat mengelola peran untuk lingkungan Elastic Beanstalk Anda:

Pembuatan dan manajemen peran

- Buat peran menggunakan kebijakan AWS terkelola jika memungkinkan
- Ikuti Praktik Terbaik Keamanan IAM
- Gunakan AWS Policy Generator untuk kebijakan kustom
- Menerapkan batas izin untuk keamanan tambahan

Pemantauan dan audit

Aktifkan AWS CloudTrail untuk memantau penggunaan peran:

- Ikuti Panduan AWS CloudTrail Pengguna
- Konfigurasikan integrasi CloudWatch Log untuk pemantauan waktu nyata
- Menyiapkan peringatan untuk panggilan API yang tidak sah

Proses peninjauan rutin

Tetapkan siklus tinjauan triwulanan untuk melakukan tugas-tugas berikut:

- Audit izin yang tidak digunakan menggunakan IAM Access Analyzer
- Hapus izin yang sudah ketinggalan zaman
- Perbarui peran berdasarkan prinsip hak istimewa paling sedikit

Manajemen sertifikat

Terapkan praktik ini untuk sertifikat SSL/TLS di lingkungan Elastic Beanstalk Anda:

Siklus hidup sertifikat

- Gunakan AWS Certificate Manageruntuk manajemen sertifikat
- Aktifkan perpanjangan otomatis untuk sertifikat yang dikeluarkan ACM
- Siapkan pemberitahuan kedaluwarsa

Standar keamanan

- Gunakan TLS 1.2 atau yang lebih baru

- Ikuti kebijakan AWS keamanan untuk pendengar HTTPS
- Menerapkan HTTP Strict Transport Security (HSTS) jika diperlukan

Manajemen kelompok keamanan

Terapkan praktik terbaik grup keamanan ini:

Manajemen aturan

- Dokumentasikan semua persyaratan port khusus
- Gunakan VPC Flow Logs untuk memantau lalu lintas
- Gunakan aturan referensi Grup Keamanan alih-alih rentang IP jika memungkinkan

Audit reguler

Tetapkan ulasan bulanan untuk melakukan tugas-tugas berikut:

- Mengidentifikasi dan menghapus aturan yang tidak digunakan
- Validasi persyaratan sumber/tujuan
- Periksa aturan yang tumpang tindih

Pencatatan log dan pemantauan

Untuk pemantauan keamanan yang efektif, konfigurasikan log berikut:

Log peristiwa Windows pada EC2 instance

```
# Review Security event log
PS C:\migrations_workspace> Get-EventLog -LogName Security -Newest 50

# Check Application event log
PS C:\migrations_workspace> Get-EventLog -LogName Application -Source "IIS*"
```

CloudWatch Integrasi log

Konfigurasikan agen CloudWatch Log untuk mengalirkan log peristiwa Windows CloudWatch untuk pemantauan dan peringatan terpusat.

Untuk masalah yang terus-menerus, kumpulkan log ini dan kontak AWS Dukungan dengan informasi berikut:

- ID Lingkungan
- ID Deployment (jika ada)
- Pesan kesalahan yang relevan
- Garis waktu perubahan keamanan

Memahami IIS ke pemetaan migrasi Elastic Beanstalk

Migrasi dari IIS ke Elastic Beanstalk melibatkan pemetaan konfigurasi server Windows lokal Anda ke sumber daya cloud. AWS Memahami pemetaan ini sangat penting untuk keberhasilan migrasi dan manajemen pasca-migrasi.

Situs dan aplikasi IIS di Elastic Beanstalk

Dalam IIS, situs web mewakili kumpulan aplikasi web dan direktori virtual, masing-masing dengan konfigurasi dan kontennya sendiri. Saat bermigrasi ke Elastic Beanstalk, komponen-komponen ini diubah sebagai berikut:

Situs web IIS

Situs web IIS Anda menjadi aplikasi dalam Elastic Beanstalk. Setiap konfigurasi situs web, termasuk binding, kumpulan aplikasi, dan pengaturan otentikasi, dipertahankan melalui manifes penyebaran () Elastic Beanstalk. `aws-windows-deployment-manifest.json`

Misalnya, jika Anda memiliki beberapa situs seperti Situs Web Default dan IntranetSite, eb migrate paket konten dan konfigurasi setiap situs sambil mempertahankan isolasi mereka.

Perintah ini membuat aturan pendengar Application Load Balancer (ALB) yang sesuai untuk menangani permintaan perutean ke aplikasi Anda. Ini juga mengkonfigurasi grup keamanan untuk memastikan akses port yang tepat berdasarkan binding IIS asli Anda.

Kolam aplikasi

Kumpulan aplikasi IIS menyediakan isolasi proses pekerja, manajemen runtime, dan kemampuan daur ulang untuk aplikasi Anda. Dalam Elastic Beanstalk, ini dipetakan ke proses lingkungan yang ditentukan `aws:elasticbeanstalk:environment:process` melalui namespace dan dikonfigurasi melalui IIS pada instance. EC2

Migrasi mempertahankan setelan kumpulan aplikasi penting termasuk yang berikut ini:

- Konfigurasi model proses - Identitas (ApplicationPoolIdentity, NetworkService, atau akun khusus), pengaturan batas waktu idle, dan interval daur ulang proses
- Pengaturan versi.NET CLR - Mempertahankan versi.NET Framework yang ditentukan (v2.0, v4.0, atau No Managed Code) untuk memastikan kompatibilitas aplikasi
- Mode pipeline terkelola - Mempertahankan pengaturan mode pipeline Terintegrasi atau Klasik untuk mempertahankan arsitektur pemrosesan permintaan HTTP Anda
- Pengaturan lanjutan - Panjang antrian, batas CPU, ambang batas perlindungan yang cepat gagal, dan batas waktu startup

eb migratePerintah mempertahankan pemetaan antara situs dan kumpulan aplikasi selama migrasi ke lingkungan Elastic Beanstalk Anda.

Jika kumpulan aplikasi Anda menggunakan jadwal daur ulang kustom (waktu tertentu atau ambang batas memori), ini diimplementasikan melalui PowerShell skrip dalam paket penerapan yang mengonfigurasi pengaturan IIS yang sesuai pada instance. EC2

Binding situs web

Binding situs web IIS, yang menentukan bagaimana klien mengakses aplikasi Anda, diubah menjadi konfigurasi Application Load Balancer (ALB) berikut:

- Binding port dipetakan ke aturan pendengar ALB yang sesuai
- Konfigurasi header host diterjemahkan ke dalam aturan perutean ALB
- Situs berkemampuan SSL menggunakan AWS Certificate Manager (ACM) untuk manajemen sertifikat

Direktori virtual dan manajemen jalur aplikasi

Direktori dan aplikasi virtual IIS menyediakan pemetaan jalur URL ke direktori fisik. Elastic Beanstalk mempertahankan hubungan ini melalui konstruksi berikut:

Direktori virtual

Proses migrasi mempertahankan jalur fisik direktori virtual Anda dalam paket penyebaran.

Pemetaan jalur dikonfigurasi dalam konfigurasi IIS pada EC2 instance, memastikan bahwa struktur URL Anda tetap utuh setelah migrasi.

Jalur fisik penggerak non-sistem

> **Important**
>
> Secara default, lingkungan Elastic Beanstalk Windows hanya menyediakan drive C:\ (volume root). Dalam versi saat ini, aplikasi dengan konten pada drive non-sistem (D:\, E:\, dll.) Tidak didukung untuk migrasi.

eb migratePerintah secara otomatis mendeteksi jalur fisik yang terletak di drive non-sistem dan memperingatkan Anda tentang potensi masalah seperti contoh berikut:

```
ERROR: Detected physical paths on drive D:\ which are not supported in the current
 version:
  - D:\websites\intranet
  - D:\shared\images

Migration of content from non-system drives is not supported. Please relocate this
 content to the C:\ drive before migration. Otherwise, select only those sites that
 are on C:\.
```

Jika aplikasi Anda memiliki dependensi pada drive non-sistem, Anda perlu memodifikasi aplikasi Anda untuk menyimpan semua konten di drive C:\ sebelum migrasi.

Aplikasi bersarang

Aplikasi yang bersarang di bawah situs web digunakan dengan konfigurasi jalur yang benar dan penugasan kumpulan aplikasi yang sesuai. Proses migrasi mempertahankan semua `web.config` pengaturan, memastikan bahwa konfigurasi khusus aplikasi terus berfungsi seperti yang diharapkan di lingkungan cloud.

Penulisan ulang URL dan perutean permintaan aplikasi (ARR)

Jika penyebaran IIS Anda menggunakan URL Rewrite atau Application Request Routing (ARR), eb migrate menangani konfigurasi ini melalui aturan dan konfigurasi berikut:

Aturan penulisan ulang URL

Aturan penulisan ulang URL dari `web.config` file Anda diterjemahkan ke dalam aturan perutean ALB jika memungkinkan. Misalnya, entri berikut menjadi aturan pendengar ALB yang mengarahkan lalu lintas berdasarkan header host dan pola jalur. :

```
<!-- Original IIS URL Rewrite Rule -->
<rule name="Redirect to WWW" stopProcessing="true">
    <match url="(.*)" />
    <conditions>
        <add input="{HTTP_HOST}" pattern="^example.com$" />
    </conditions>
    <action type="Redirect" url="http://www.example.com/{R:1}" />
</rule>
```

Perutean permintaan aplikasi

Konfigurasi ARR dipertahankan melalui instalasi fitur ARR pada instance. EC2 Proses migrasi menyelesaikan tugas-tugas berikut:

- Mengonfigurasi pengaturan proxy agar sesuai dengan lingkungan sumber Anda
- Mempertahankan aturan penulisan ulang URL yang terkait dengan ARR

Struktur artefak migrasi

Ketika Anda menjalankaneb migrate, itu membuat direktori terstruktur yang berisi semua komponen penyebaran yang diperlukan. Daftar berikut menjelaskan struktur direktori:

```
C:\migration_workspace\
### .\migrations\latest\
    ### upload_target\
        ### [SiteName].zip                # One ZIP per IIS site
        ### aws-windows-deployment-manifest.json
        ### ebmigrateScripts\
            ### site_installer.ps1         # Site installation scripts
            ### arr_configuration.ps1      # ARR configuration scripts
            ### permission_handler.ps1     # Permission management
            ### firewall_config.ps1        # Windows Firewall rules
```

`aws-windows-deployment-manifest.json`File ini adalah file konfigurasi inti yang menginstruksikan Elastic Beanstalk cara menerapkan aplikasi Anda. Lihat contoh struktur berikut:

```
{
    "manifestVersion": 1,
    "deployments": {
        "msDeploy": [
            {
                "name": "Primary Site",
                "parameters": {
                    "appBundle": "DefaultWebSite.zip",
                    "iisPath": "/",
                    "iisWebSite": "Default Web Site"
                }
            }
        ],
        "custom": [
            {
                "name": "ConfigureARR",
                "scripts": {
                    "install": {
                        "file": "ebmigrateScripts\\arr_configuration.ps1"
                    },
                    "uninstall": {
                        "file": "ebmigrateScripts\\noop.ps1"
                    },
                    "restart": {
                        "file": "ebmigrateScripts\\noop.ps1"
                    }
                }
            }
        ]
    }
}
```

Manifes ini memastikan hasil ini untuk migrasi Anda:

- Aplikasi digunakan untuk memperbaiki jalur IIS
- Konfigurasi kustom diterapkan
- Pengaturan khusus situs dipertahankan
- Urutan penyebaran dipertahankan

Skenario migrasi lanjutan

Bagian ini mencakup skenario migrasi lanjutan untuk penerapan IIS yang kompleks.

Migrasi multi-situs dengan Application Request Routing (ARR)

eb migratePerintah secara otomatis mendeteksi dan mempertahankan konfigurasi ARR selama migrasi. Ketika mengidentifikasi pengaturan ARR di IIS Andaapplication`Host.config`, itu menghasilkan PowerShell skrip yang diperlukan untuk menginstal ulang dan mengkonfigurasi ARR pada instance target. EC2

Deteksi konfigurasi ARR

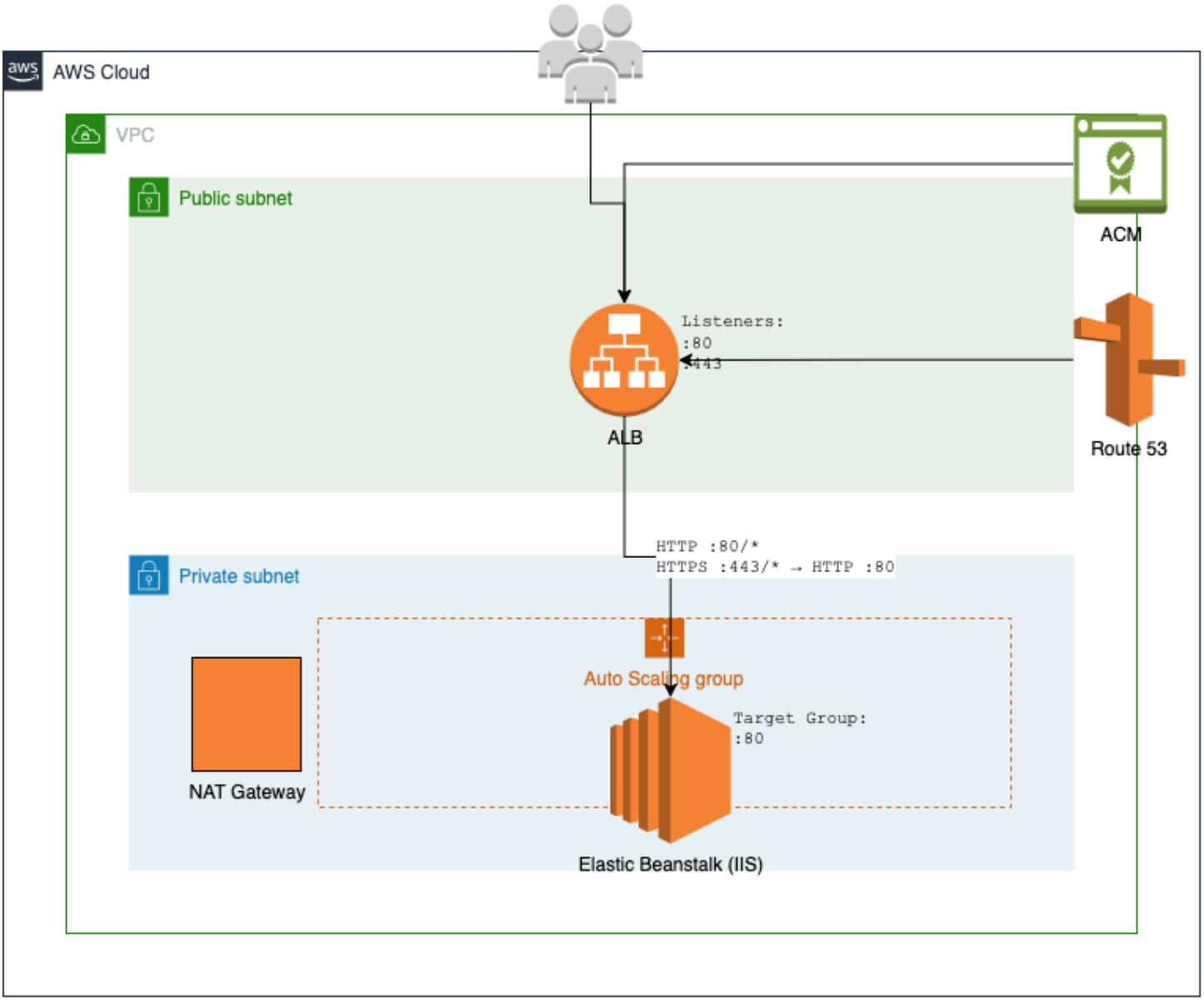
Proses migrasi memeriksa tiga bagian konfigurasi utama di IIS:

- `system.webServer/proxy`: Pengaturan proxy ARR inti
- `system.webServer/rewrite`: Aturan penulisan ulang URL
- `system.webServer/caching`: Konfigurasi caching

Misalnya, pertimbangkan konfigurasi ARR umum di mana proxy port 80 yang `RouterSite` berjalan meminta `APIService` dan `AdminPortal` berjalan pada port 8081 dan 8082 masing-masing:

```
<!-- Original IIS ARR Configuration -->
<rewrite>
    <rules>
        <rule name="Route to API" stopProcessing="true">
            <match url="^api/(.*)$" />
            <action type="Rewrite" url="http://backend:8081/api/{R:1}" />
        </rule>
        <rule name="Route to Admin" stopProcessing="true">
            <match url="^admin/(.*)$" />
            <action type="Rewrite" url="http://backend:8082/admin/{R:1}" />
        </rule>
    </rules>
</rewrite>
```

Diagram berikut menggambarkan bagaimana aturan ini tersembunyi di balik port 80 di server IIS dan tidak diekspos melalui Grup EC2 Keamanan. Hanya port 80 yang dapat diakses oleh Application Load Balancer dan semua lalu lintas darinya dialihkan ke grup target di port 80.



Perintah berikut dapat memigrasikan konfigurasi ini:

```
PS C:\migrations_workspace> eb migrate --sites "RouterSite,APIService,AdminPortal" `
    --copy-firewall-config
```

Proses migrasi ARR

Proses migrasi mempertahankan konfigurasi ARR Anda melalui beberapa langkah.

Ekspor konfigurasi

Alat ini mengekspor pengaturan ARR Anda yang ada dari tiga bagian konfigurasi kunci ke dalam file XHTML terpisah yang disimpan dalam direktori: `ebmigrateScripts`

```
ebmigrateScripts\
### arr_config_proxy.xml
### arr_config_rewrite.xml
### arr_config_caching.xml
```

Skrip instalasi

Dua PowerShell skrip dihasilkan untuk menangani penyiapan ARR:

1. `arr_msi_installer.ps1`: Mengunduh dan menginstal modul ARR
2. `arr_configuration_importer_script.ps1`: Mengimpor konfigurasi ARR Anda yang diekspor

Integrasi manifes penerapan

Skrip diintegrasikan ke dalam proses penyebaran melalui entri di: `aws-windows-deployment-manifest.json`

```
{
    "manifestVersion": 1,
    "deployments": {
        "custom": [
            {
                "name": "WindowsProxyFeatureEnabler",
                "scripts": {
                    "install": {
                        "file": "ebmigrateScripts\
\windows_proxy_feature_enabler.ps1"
                    }
                }
            },
            {
                "name": "ArrConfigurationImporterScript",
                "scripts": {
                    "install": {
                        "file": "ebmigrateScripts\
\arr_configuration_importer_script.ps1"
```

```
                    }
                }
            }
        ]
    }
}
```

Integrasi penyeimbang beban

Proses migrasi menerjemahkan aturan ARR Anda ke dalam aturan pendengar Application Load Balancer (ALB) jika memungkinkan. Misalnya, konfigurasi ARR di atas menghasilkan aturan ALB yang merutekan lalu lintas berdasarkan pola jalur URL sambil mempertahankan perutean internal pada instance. EC2

Lingkungan yang dihasilkan mempertahankan logika perutean ARR Anda sambil memanfaatkan infrastruktur elastis. AWS Aplikasi Anda terus berfungsi seperti sebelumnya, dengan ARR menangani perutean internal sementara Application Load Balancer mengelola distribusi lalu lintas eksternal.

Migrasi multi-situs tanpa ARR menggunakan perutean berbasis host

Sementara Application Request Routing (ARR) adalah pendekatan umum untuk mengelola beberapa situs di IIS, Anda juga dapat memigrasikan penyebaran multi-situs langsung ke Elastic Beanstalk tanpa ARR dengan memanfaatkan kemampuan routing berbasis host Application Load Balancer. Pendekatan ini dapat mengurangi kompleksitas dan meningkatkan kinerja dengan menghilangkan lapisan routing tambahan.

Ikhtisar perutean berbasis host

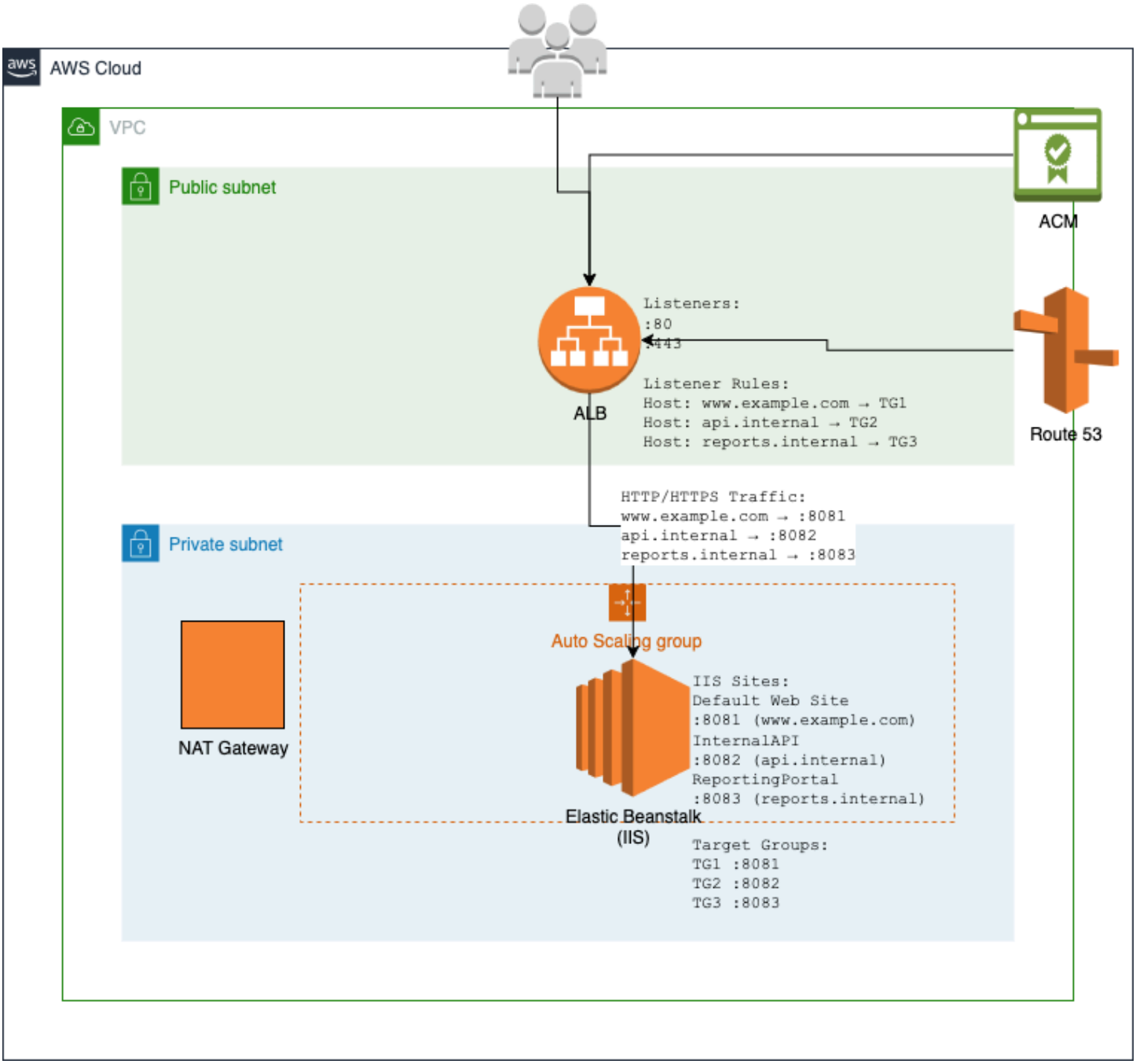
Dalam pendekatan ini, setiap situs IIS diekspos di luar EC2 instance, dan Application Load Balancer mengarahkan lalu lintas langsung ke port yang sesuai berdasarkan header host. Ini menghilangkan kebutuhan akan ARR sambil mempertahankan pemisahan antara aplikasi Anda.

Pertimbangkan konfigurasi IIS multi-situs dengan tiga situs, masing-masing dengan pengikatan nama hostnya sendiri:

```
<sites>
    <site name="Default Web Site" id="1">
        <bindings>
            <binding protocol="http" bindingInformation="*:8081:www.example.com" />
```

```
        </bindings>
    </site>
    <site name="InternalAPI" id="2">
        <bindings>
            <binding protocol="http" bindingInformation="*:8082:api.internal" />
        </bindings>
    </site>
    <site name="ReportingPortal" id="3">
        <bindings>
            <binding protocol="http" bindingInformation="*:8083:reports.internal" />
        </bindings>
    </site>
</sites>
```

Situs-situs ini diekspos di port 8081, 8082, dan 8083 melalui Grup Keamanan. EC2 Application Load Balancer merutekan ke mereka berdasarkan konfigurasi aturan pendengar Load Balancer.



Proses migrasi

Untuk memigrasikan konfigurasi ini ke Elastic Beanstalk tanpa menggunakan ARR eb migrate gunakan perintah dalam contoh berikut:

```
PS C:\migrations_workspace> eb migrate --sites "Default Web
 Site,InternalAPI,ReportingPortal"
```

Proses migrasi secara otomatis mengonfigurasi Application Load Balancer dengan aturan perutean berbasis host yang mengarahkan lalu lintas ke grup target yang sesuai berdasarkan header host. Setiap grup target meneruskan lalu lintas ke port yang sesuai pada EC2 instans Anda:

1. Header host www.example.com → Grup Sasaran di port 8081
2. Header host api.internal → Grup Sasaran pada port 8082
3. Host header reports.internal → Grup Sasaran di port 8083

Konfigurasi SSL/TLS

Untuk mengamankan aplikasi Anda dengan SSL/TLS lakukan langkah-langkah berikut:

1. Minta sertifikat untuk domain Anda melalui AWS Certificate Manager(ACM).
2. Konfigurasikan listener HTTPS pada Application Load Balancer Anda menggunakan sertifikat ini.
3. Perbarui konfigurasi lingkungan Anda untuk menyertakan pendengar HTTPS dengan pengaturan opsi konfigurasi berikut.

```
option_settings:
  aws:elb:listener:443:
    ListenerProtocol: HTTPS
    SSLCertificateId: arn:aws:acm:region:account-id:certificate/certificate-id
    InstancePort: 80
    InstanceProtocol: HTTP
```

Dengan konfigurasi ini, penghentian SSL terjadi di penyeimbang beban, dan lalu lintas diteruskan ke instance Anda melalui HTTP. Ini menyederhanakan manajemen sertifikat sambil mempertahankan koneksi aman dengan klien.

Praktik terbaik

Grup keamanan

> Konfigurasikan grup keamanan untuk mengizinkan lalu lintas masuk hanya pada port yang digunakan oleh situs IIS Anda (8081, 8082, 8083 dalam contoh ini) dari grup keamanan Application Load Balancer.

Pemeriksaan kondisi

Konfigurasikan pemeriksaan kesehatan untuk setiap grup sasaran untuk memastikan lalu lintas hanya diarahkan ke instance yang sehat. Buat titik akhir pemeriksaan kesehatan untuk setiap aplikasi jika belum ada.

Pemantauan

Siapkan CloudWatch alarm untuk memantau kesehatan dan kinerja masing-masing kelompok sasaran secara terpisah. Ini memungkinkan Anda mengidentifikasi masalah khusus untuk aplikasi individual.

Penskalaan

Pertimbangkan persyaratan sumber daya dari semua aplikasi saat mengonfigurasi kebijakan penskalaan otomatis. Jika satu aplikasi memiliki kebutuhan sumber daya yang berbeda secara signifikan, pertimbangkan untuk memigrasikannya ke lingkungan yang terpisah.

Manajemen direktori virtual

eb migratePerintah mempertahankan struktur direktori virtual saat memigrasikan aplikasi IIS Anda ke Elastic Beanstalk.

Konfigurasi izin default

Saat memigrasi direktori virtual, eb migrate tetapkan sekumpulan izin dasar dengan memberikan akses ke: ReadAndExecute

- IIS_IUSRS
- IUSR
- Pengguna yang Diautentikasi

Misalnya, pertimbangkan struktur direktori virtual yang khas:

```
<site name="CorporatePortal">
    <application path="/" applicationPool="CorporatePortalPool">
        <virtualDirectory path="/" physicalPath="C:\sites\portal" />
        <virtualDirectory path="/shared" physicalPath="C:\shared\content" />
        <virtualDirectory path="/reports" physicalPath="D:\reports" />
    </application>
```

```
</site>
```

Direktori virtual yang dilindungi kata sandi

Ketika eb migrate menemukan direktori virtual yang dilindungi kata sandi, itu mengeluarkan peringatan dan memerlukan intervensi manual.

Contoh konfigurasi berikut akan menyebabkan respons peringatan yang mengikuti contoh.

```
<virtualDirectory path="/secure"
                  physicalPath="C:\secure\content"
                  userName="DOMAIN\User"
                  password="[encrypted]" />
```

```
[WARNING] CorporatePortal/secure is hosted at C:\secure\content which is password-
protected and won't be copied.
```

Untuk mempertahankan perlindungan kata sandi, buat skrip penerapan kustom seperti berikut ini:

```
# PS C:\migrations_workspace> cat secure_vdir_config.ps1

$vdirPath = "C:\secure\content"
$siteName = "CorporatePortal"
$vdirName = "secure"
$username = "DOMAIN\User"
$password = "SecurePassword"

# Ensure directory exists
if (-not (Test-Path $vdirPath)) {
    Write-Host "Creating directory: $vdirPath"
    New-Item -Path $vdirPath -ItemType Directory -Force
}

# Configure virtual directory with credentials
Write-Host "Configuring protected virtual directory: $vdirName"
New-WebVirtualDirectory -Site $siteName -Name $vdirName `
    -PhysicalPath $vdirPath -UserName $username -Password $password

# Set additional permissions as needed
$acl = Get-Acl $vdirPath
$rule = New-Object System.Security.AccessControl.FileSystemAccessRule(
```

```
    $username, "ReadAndExecute", "ContainerInherit,ObjectInherit", "None", "Allow"
)
$acl.AddAccessRule($rule)
Set-Acl $vdirPath $acl
```

Tambahkan skrip ini ke penerapan Anda dengan memasukkannya ke dalam manifes:

```
{
    "manifestVersion": 1,
    "deployments": {
        "custom": [
            {
                "name": "SecureVirtualDirectory",
                "scripts": {
                    "install": {
                        "file": "secure_vdir_config.ps1"
                    }
                }
            }
        ]
    }
}
```

Manajemen izin khusus

eb migratePerintah ini menyediakan kerangka kerja untuk skrip izin khusus untuk mengakomodasi aplikasi yang memerlukan izin selain default.

```
$paths = @(
    "C:\sites\portal\uploads",
    "C:\shared\content"
)

foreach ($path in $paths) {
    if (-not (Test-Path $path)) {
        Write-Host "Creating directory: $path"
        New-Item -Path $path -ItemType Directory -Force
    }

    $acl = Get-Acl $path
```

```
    # Add custom permissions
    $customRules = @(
        # Application Pool Identity - Full Control
        [System.Security.AccessControl.FileSystemAccessRule]::new(
            "IIS AppPool\CorporatePortalPool",
            "FullControl",
            "ContainerInherit,ObjectInherit",
            "None",
            "Allow"
        ),
        # Custom Service Account
        [System.Security.AccessControl.FileSystemAccessRule]::new(
            "NT SERVICE\CustomService",
            "Modify",
            "ContainerInherit,ObjectInherit",
            "None",
            "Allow"
        )
    )

    foreach ($rule in $customRules) {
        $acl.AddAccessRule($rule)
    }

    Set-Acl $path $acl
    Write-Host "Custom permissions applied to: $path"
}
```

Praktik terbaik

Ikuti praktik terbaik ini untuk merencanakan, menjalankan, memantau, dan memverifikasi migrasi Anda.

Perencanaan pra-migrasi

Dokumentasikan izin dan persyaratan otentikasi yang ada sebelum migrasi. Uji skrip izin khusus di lingkungan pengembangan sebelum menerapkan ke produksi.

Manajemen konten bersama

Untuk direktori konten bersama, pastikan semua izin sistem file yang diperlukan dikonfigurasi dengan benar melalui skrip khusus. Pertimbangkan untuk menggunakan Amazon FSx untuk Windows File Server untuk persyaratan penyimpanan bersama.

Pemantauan dan verifikasi

Pantau log aplikasi setelah migrasi untuk memverifikasi akses yang tepat ke direktori virtual. Berikan perhatian khusus pada bidang-bidang berikut:

- Akses identitas kumpulan aplikasi
- Izin akun layanan kustom
- Konektivitas berbagi jaringan
- Kegagalan otentikasi

Pengaturan kolam aplikasi kustom

eb migratePerintah tidak menyalin pengaturan kumpulan aplikasi kustom secara default. Untuk mempertahankan konfigurasi kumpulan aplikasi kustom, ikuti prosedur ini untuk membuat dan menerapkan bagian manifes kustom.

1. Buat arsip artefak migrasi Anda.

```
PS C:\migrations_workspace> eb migrate --archive
```

2. Buat PowerShell skrip kustom untuk mengkonfigurasi kumpulan aplikasi.

```
# PS C:\migrations_workspace> cat .\migrations\latest\upload_target
\customize_application_pool_config.ps1

$configPath = "$env:windir\System32\inetsrv\config\applicationHost.config"

[xml]$config = Get-Content -Path $configPath

$newPoolXml = @"
<!-- Original IIS Configuration -->
<applicationPools>
    <add name="CustomPool"
         managedRuntimeVersion="v4.0"
         managedPipelineMode="Integrated">
        <processModel identityType="SpecificUser"
                      userName="AppPoolUser"
                      password="[encrypted]" />
        <recycling>
            <periodicRestart time="00:00:00">
                <schedule>
```

```
                    <add value="02:00:00" />
                    <add value="14:00:00" />
                </schedule>
            </periodicRestart>
        </recycling>
    </add>
</applicationPools>
"@
$newPoolXmlNode = [xml]$newPoolXml

# Find the applicationPools section
$applicationPools = $config.SelectSingleNode("//configuration/
system.applicationHost/applicationPools")

# Import the new node into the document
$importedNode = $config.ImportNode($newPoolXmlNode.DocumentElement, $true)
$applicationPools.AppendChild($importedNode)

# Save the changes
$config.Save($configPath)

Write-Host "ApplicationHost.config has been updated successfully."
```

3. Perbarui `aws-windows-deployment-manifest.json` file untuk menyertakan skrip kustom Anda.

```
{
    "manifestVersion": 1,
    "deployments": {
        ...
        "custom": [
            ...,
            {
                "name": "ModifyApplicationPoolConfig",
                "description": "Modify application pool configuration from source
 machine to remove",
                "scripts": {
                    "install": {
                        "file": "customize_application_pool_config.ps1"
                    },
                    "restart": {
                        "file": "ebmigrateScripts\\noop.ps1"
                    },
```

```
                    "uninstall": {
                        "file": "ebmigrateScripts\\noop.ps1"
                    }
                }
            }
        ]
    }
}
```

4. Buat lingkungan dengan direktori arsip yang diperbarui.

```
PS C:\migrations_workspace> eb migrate `
    --archive-dir '.\migrations\latest\upload_target\'
```

`--archive-dir`Argumen mengatakan eb migrate untuk menggunakan kode sumber yang sebelumnya dibuat, menghindari pembuatan arsip baru.

Menerapkan versi sebelumnya

The eb migrate mempertahankan riwayat migrasi Anda melalui direktori stempel waktu dan versi aplikasi di Elastic Beanstalk. Setiap migrasi membuat file zip unik yang dapat digunakan jika diperlukan.

```
PS C:\migrations_workspace> ls .\migrations\

Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d----l        3/18/2025  10:34 PM                latest
d-----        3/16/2025   5:47 AM                migration_1742104049.479849
d-----        3/17/2025   9:18 PM                migration_1742246303.18056
d-----        3/17/2025   9:22 PM                migration_1742246546.565739
...
d-----        3/18/2025  10:34 PM                migration_1742337258.30742
```

Tautan `latest` simbolis selalu menunjuk ke direktori artefak migrasi yang paling baru dibuat. Selain log aplikasi dan kesalahan yang relevan, setiap direktori artefak migrasi juga berisi `upload_target.zip` file yang dapat Anda gunakan ke Elastic Beanstalk.

```
PS C:\migrations_workspace> ls .\migrations\latest\
```

```
Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----        3/18/2025  10:34 PM                upload_target
-a----        3/18/2025  10:34 PM          13137 application.log
-a----        3/18/2025  10:34 PM              0 error.log
-a----        3/18/2025  10:34 PM        1650642 upload_target.zip
```

Anda dapat menyebarkan `upload_target.zip` file menggunakaneb migrate:

```
PS C:\migrations_workspace> eb migrate --zip .\migrations\latest\upload_target.zip
```

Pemecahan masalah dan diagnostik

Bagian ini memberikan panduan untuk memecahkan masalah umum yang mungkin timbul selama migrasi aplikasi IIS ke Elastic Beanstalk.

Mengaitkan EC2 keypair dengan lingkungan Anda

Anda dapat masuk dengan aman ke instans Amazon Elastic Compute Cloud EC2 (Amazon) yang disediakan untuk aplikasi Elastic Beanstalk Anda dengan key pair Amazon. EC2 Untuk petunjuk cara membuat key pair, lihat Membuat Pasangan Kunci Menggunakan Amazon EC2 di Panduan EC2 Pengguna Amazon.

Menentukan nama kunci untuk eb migrate memiliki efek mengaitkan lingkungan Elastic Beanstalk Anda dengan keypair. Untuk tujuan keamanan, ini tidak akan membuka port 3389 pada grup keamanan EC2 instans Anda. Anda dapat mengaitkan grup EC2 keamanan tambahan yang memungkinkan lalu lintas di port 3389 eb config setelah migrasi awal.

```
PS C:\migrations_workspace> eb migrate  `
    --keyname "my-keypair"  `
    --verbose
```

Saat Anda membuat key pair, Amazon EC2 menyimpan salinan kunci publik Anda. Jika Anda tidak perlu lagi menggunakannya untuk terhubung ke instance lingkungan apa pun, Anda dapat menghapusnya dari Amazon EC2. Untuk detailnya, lihat Menghapus Pasangan Kunci Anda di Panduan EC2 Pengguna Amazon.

Untuk informasi selengkapnya tentang menghubungkan ke EC2 instans Windows Amazon, lihat Menghubungkan ke Instans Windows.

Mengakses log

EB CLI menyediakan fasilitas eb logs yang dapat Anda gunakan untuk mengambil log dari lingkungan Elastic Beanstalk tanpa masuk ke instance-nya. EC2 Setelah eksekusieb migrate, Anda dapat mengeluarkan eb logs --zip perintah yang akan mengunduh dan menyimpan log ke `.elasticbeanstalk\logs` direktori.

Atau, Anda dapat melihat log melalui konsol AWS Elastic Beanstalk. Untuk informasi selengkapnya, lihat Melihat log dari EC2 instans Amazon di lingkungan Elastic Beanstalk Anda.

Mengakses artefak sisi klien

eb migratePerintah menyimpan log aplikasi dan kesalahan yang dihasilkan oleh direktori artefak migrasi msdeploy di dalam.

```
./migrations/
### latest -> migration_20240308_123456/
### migration_20240308_123456/
    ### application.log
    ### error.log
    ### upload_target\
```

Memantau kesehatan lingkungan

Elastic Beanstalk membantu Anda memantau kesehatan menggunakan kemampuan pemantauan kesehatan yang ditingkatkan. Ini adalah sistem pemantauan kesehatan otomatis yang terus melacak status operasional instance aplikasi, memanfaatkan metrik bawaan seperti pemanfaatan CPU, latensi, jumlah permintaan, dan kode respons.

Sistem pemantauan kesehatan menggunakan pendekatan berbasis agen untuk mengumpulkan data tingkat instans dan terintegrasi dengan pencatatan dan peringatan waktu nyata. Elastic Load Balancing (ELB) dan Auto Scaling merespons perubahan status kesehatan secara dinamis, memastikan ketersediaan tinggi dan toleransi kesalahan. Mode pemantauan lanjutan, termasuk pelaporan kesehatan yang ditingkatkan, memberikan visibilitas terperinci ke dalam perilaku aplikasi, memungkinkan pemecahan masalah proaktif dan mekanisme pemulihan otomatis.

Jalankan perintah EB eb health CLI untuk menampilkan kesehatan lingkungan. Informasi berikut menampilkan:

- Status kesehatan instance

- Metrik respons aplikasi
- Pemanfaatan sumber daya sistem
- Acara penyebaran terbaru

EC2 optimasi kinerja

Secara default, eb migrate pilih jenis instans c5.2xlarge untuk memberikan pengalaman pertama kali yang optimal dengan Elastic Beanstalk. Anda dapat mengganti perilaku ini dengan --instance-type argumen:

```
PS C:\migrations_workspace> eb migrate `
    --instance-type "t3.large"
```

Untuk lingkungan produksi, pertimbangkan faktor-faktor ini saat memilih jenis instans:

- Persyaratan memori aplikasi Anda
- Persyaratan CPU untuk memproses beban kerja
- Kebutuhan kinerja jaringan
- Tujuan pengoptimalan biaya

Konfigurasi volume EBS

Secara default, Elastic Beanstalk hanya akan membuat root `C:\` block-device volume () untuk lingkungan Anda. Anda dapat meneruskan volume snapshot Amazon Elastic Block Store tambahan dengan --ebs-snapshots opsi:

```
PS C:\migrations_workspace> eb migrate `
    --ebs-snapshots "snap-123456789abc"
```

Untuk contoh bagaimana Anda dapat mengonfigurasi pemetaan blok-perangkat dengan Elastic Beanstalk, lihat artikel blog Sesuaikan Volume Ephemeral dan EBS di Lingkungan Elastic Beanstalk.

Untuk aplikasi dengan persyaratan penyimpanan tinggi, pertimbangkan opsi berikut:

- Menggunakan volume EBS untuk data persisten
- Menerapkan Amazon S3 untuk konten statis
- Menggunakan Amazon FSx untuk Windows File Server untuk sistem file bersama

Masalah dan solusi umum

Acara: Instalasi Deploy Web Hilang

Jika Anda menemukan kesalahan terkait Web Deploy tidak ditemukan, maka instal Web Deploy 3.6 atau yang lebih baru dari Microsoft Web Platform Installer. Contoh berikut menampilkan pesan kesalahan yang mungkin.

```
Couldn't find msdeploy.exe. Follow instructions here: https://learn.microsoft.com/en-us/iis/install/installing-publishing-technologies/installing-and-configuring-web-deploy
```

Acara: Masalah izin selama migrasi

Jika Anda mengalami kesalahan terkait izin, pastikan Anda menjalankan CLI EB dengan hak administratif. Contoh berikut menampilkan pesan kesalahan yang mungkin.

```
[ERROR] Access to the path 'C:\inetpub\wwwroot\web.config' is denied.
```

Acara: Masalah identitas kumpulan aplikasi

Jika aplikasi Anda gagal dimulai karena masalah identitas kumpulan aplikasi, buat skrip kustom untuk mengonfigurasi identitas kumpulan aplikasi seperti yang ditunjukkan padathe section called “Pengaturan kolam aplikasi kustom”.

Acara: Kesalahan konfigurasi sertifikat SSL

Jika binding HTTPS gagal berfungsi, pastikan Anda telah menetapkan ARN sertifikat ACM yang valid menggunakan parameter opsi. eb mibrate `--ssl-certificates`

Acara: Batas waktu pembuatan lingkungan

Jika waktu pembuatan lingkungan habis, periksa AWS CloudFormation peristiwa di Konsol AWS Manajemen untuk mengetahui kegagalan pembuatan sumber daya tertentu. Penyebab umum termasuk masalah konfigurasi VPC atau batas layanan.

Mendapatkan Dukungan

Jika Anda mengalami masalah yang tidak dapat Anda selesaikan, sebelum menghubungi, AWS Dukungan kumpulkan informasi berikut:

- ID Lingkungan (`eb status`)
- Log aplikasi (`eb logs --zip`)
- Artefak migrasi dari `.\migrations\latest\`
- Konfigurasi sumber IIS (output dari`eb migrate explore --verbose`)
- Pesan kesalahan terperinci

Untuk informasi selengkapnya tentang pemecahan masalah Elastic Beanstalk, lihat. Memecahkan masalah lingkungan Elastic Beanstalk Anda

Membandingkan opsi migrasi: EB CLI vs. AWS Application Migration Service

AWS menawarkan beberapa jalur untuk memigrasikan aplikasi Windows ke cloud. Bagian ini membandingkan dua opsi utama: eb migrate perintah di EB CLI dan AWS Application Migration Service (MGN). Memahami perbedaan antara pendekatan ini akan membantu Anda memilih strategi migrasi yang paling tepat untuk kebutuhan spesifik Anda.

Perbandingan opsi migrasi

Fitur	eb migrateCLI EB ()	AWS Application Migration Service (MGN)
Fokus utama	Migrasi tingkat aplikasi situs web dan aplikasi IIS	Rehosting tingkat server dari seluruh mesin (server fisik, virtual, atau cloud)
Paling cocok untuk	Aplikasi IIS yang ingin Anda migrasikan langsung ke Elastic Beanstalk dengan konfigurasi ulang minimal	Migrasi skala besar yang melibatkan banyak server atau infrastruktur kompleks
Pendekatan penemuan	Penemuan tingkat aplikasi situs, aplikasi, dan konfigurasi IIS	Replikasi tingkat server dari seluruh mesin, termasuk sistem operasi dan aplikasi
Lingkungan target	Langsung membuat dan mengkonfigurasi lingkungan Elastic Beanstalk	Membuat EC2 instance yang memerlukan konfigurasi tambahan

Fitur	eb migrateCLI EB ()	AWS Application Migration Service (MGN)
	yang dioptimalkan untuk aplikasi Windows	untuk bekerja dengan Elastic Beanstalk
Konfigurasi pelestarian	Secara otomatis mempertahankan konfigurasi khusus IIS (situs, kumpulan aplikasi, binding)	Mempertahankan seluruh konfigura si server, yang mungkin termasuk komponen yang tidak perlu
Model penyebaran	Menciptakan lingkungan Elastic Beanstalk yang bersih dengan aplikasi Anda yang digunakan menggunakan praktik terbaik Elastic Beanstalk	Membuat replika server sumber Anda yang mungkin memerlukan pengoptimalan untuk operasi cloud
Skala migrasi	Ideal untuk migrasi yang ditargetkan dari aplikasi tertentu	Dirancang untuk migrasi skala besar dari banyak server
Langkah-langkah pasca-migrasi	Minimal; lingkungan siap digunakan dengan alat manajemen Elastic Beanstalk	Memerlukan langkah-langkah tambahan untuk berintegrasi dengan Elastic Beanstalk, seperti menjalank an tindakan pasca-peluncuran SSM

Kapan menggunakan setiap opsi migrasi

Pilih eb migrate kapan Anda memiliki persyaratan berikut:

- Anda ingin memigrasikan aplikasi IIS tertentu daripada seluruh server
- Tujuan Anda adalah mengadopsi Elastic Beanstalk sebagai platform manajemen aplikasi Anda
- Anda ingin memanfaatkan fitur platform terkelola Elastic Beanstalk seperti penskalaan, penerapan, dan pemantauan yang mudah
- Anda lebih memilih penerapan bersih yang mengikuti praktik AWS terbaik untuk operasi cloud-native
- Anda ingin meminimalkan pekerjaan konfigurasi pasca-migrasi

Pilih AWS Application Migration Service kapan Anda memiliki persyaratan berikut:

- Anda perlu memigrasi sejumlah besar server

- Anda memiliki konfigurasi server yang kompleks yang harus dipertahankan dengan tepat
- Aplikasi Anda memiliki masalah kompatibilitas yang memerlukan pemeliharaan lingkungan server yang tepat
- Anda ingin “mengangkat dan menggeser” dengan sedikit perubahan pada aplikasi Anda
- Anda berencana untuk memfaktorkan ulang atau mengoptimalkan aplikasi Anda setelah migrasi

Perbandingan alur kerja migrasi

Alur kerja EB CLI eb migrate ():

1. Instal CLI EB di server IIS sumber Anda atau host bastion.
2. Jalankan eb migrate untuk menemukan aplikasi IIS.
3. Perintah ini mengemas aplikasi dan konfigurasi Anda.
4. Lingkungan Elastic Beanstalk dibuat dengan sumber daya yang sesuai.
5. Aplikasi Anda dikerahkan ke lingkungan baru.
6. Anda dapat langsung mengelola aplikasi Anda menggunakan alat Elastic Beanstalk.

AWS Application Migration Service alur kerja:

1. Instal Agen AWS Replikasi di server sumber.
2. Konfigurasikan dan uji replikasi data.
3. Luncurkan contoh pengujian untuk memverifikasi fungsionalitas.
4. Jadwalkan cutover ke AWS.
5. Luncurkan instance produksi.
6. Jalankan tindakan pasca-peluncuran untuk mengoptimalkan cloud.
7. Jika Elastic Beanstalk adalah platform target, konfigurasi tambahan diperlukan untuk mengintegrasikan dengan Elastic Beanstalk.

Kesimpulan

Elastic Beanstalk adalah tujuan pilihan untuk AWS aplikasi platform Windows, menawarkan lingkungan terkelola yang menyederhanakan penerapan, penskalaan, dan manajemen. eb migratePerintah ini menyediakan jalur langsung ke Elastic Beanstalk untuk aplikasi IIS, dengan penemuan dan konfigurasi otomatis yang mempertahankan pengaturan aplikasi Anda.

Meskipun AWS Application Migration Service menawarkan kemampuan yang kuat untuk migrasi server skala besar, diperlukan langkah-langkah tambahan untuk berintegrasi dengan Elastic Beanstalk. Untuk sebagian besar migrasi aplikasi IIS di mana Elastic Beanstalk adalah platform eb migrate target, menawarkan pendekatan yang lebih efisien yang selaras dengan model layanan terkelola Elastic Beanstalk.

Pilih pendekatan migrasi yang paling sesuai dengan kebutuhan spesifik Anda, dengan mempertimbangkan faktor-faktor seperti skala, kompleksitas, dan arsitektur status akhir yang Anda inginkan. AWS

Untuk informasi lebih lanjut tentang AWS Application Migration Service, lihat Apa itu AWS Application Migration Service? dalam AWS Application Migration Service User Guide.

Memecahkan masalah lingkungan Elastic Beanstalk Anda

Bab ini memberikan panduan untuk memecahkan masalah dengan lingkungan Elastic Beanstalk Anda. Ini memberikan informasi berikut.

- Pengantar alat AWS Systems Manager, ditambah prosedur untuk menjalankan runbook Elastic Beanstalk yang telah ditentukan sebelumnya yang menghasilkan langkah dan rekomendasi pemecahan masalah.
- Panduan umum untuk tindakan yang dapat Anda ambil dan sumber daya yang dapat Anda lihat jika status lingkungan Anda menurun.
- Kiat pemecahan masalah yang lebih spesifik berdasarkan kategori subjek.

> **Note**
>
> Jika kesehatan lingkungan Anda berubah menjadi merah, kami sarankan Anda terlebih dahulu menggunakan AWS Systems Manager alat yang menyertakan runbook yang telah ditentukan untuk memecahkan masalah Elastic Beanstalk. Untuk informasi lebih lanjut, lihat Menggunakan alat Systems Manager.

Topik

- Menggunakan runbook AWS Systems Manager Elastic Beanstalk
- Panduan umum untuk mengatasi masalah lingkungan Elastic Beanstalk Anda
- Lingkungan yang mengakses rahasia dan parameter dengan variabel lingkungan
- Pembuatan lingkungan dan peluncuran instans
- Deployment
- Kondisi
- Konfigurasi
- Mengatasi masalah kontainer Docker
- Pertanyaan yang Sering Diajukan
- Kesalahan Umum
- Kesalahan penerapan

Menggunakan runbook AWS Systems Manager Elastic Beanstalk

Anda dapat menggunakan Systems Manager untuk memecahkan masalah lingkungan Elastic Beanstalk Anda. Untuk membantu Anda memulai dengan cepat, Systems Manager menyediakan runbook Otomasi yang telah ditentukan untuk Elastic Beanstalk. Runbook Otomasi adalah jenis dokumen Systems Manager yang mendefinisikan tindakan yang harus dilakukan pada instance lingkungan Anda dan sumber daya lainnya. AWS

Dokumen `AWSSupport-TroubleshootElasticBeanstalk` ini adalah runbook Otomasi yang dirancang untuk membantu mengidentifikasi sejumlah masalah umum yang dapat menurunkan lingkungan Elastic Beanstalk Anda. Untuk melakukannya, ia memeriksa komponen lingkungan Anda, termasuk yang berikut: EC2 instance, VPC, tumpukan, penyeimbang beban AWS CloudFormation , grup Auto Scaling, dan konfigurasi jaringan yang terkait dengan aturan grup keamanan, tabel rute, dan. ACLs

Ini juga menyediakan opsi untuk mengunggah file log yang dibundel dari lingkungan Anda ke AWS Support.

Untuk informasi selengkapnya, lihat `AWSSupport-TroubleshootElasticBeanstalk` di referensi buku runbook Otomatisasi AWS Systems Manager .

Menggunakan Systems Manager untuk menjalankan **`AWSSupport-TroubleshootElasticBeanstalk`** runbook

> **Note**
>
> Jalankan prosedur ini di tempat yang sama Wilayah AWS di mana lingkungan Elastic Beanstalk Anda berada.

1. Buka konsol AWS Systems Manager.
2. Dari panel navigasi, di bagian Manajemen Perubahan, pilih Otomatisasi.
3. Pilih Eksekusi otomatisasi.
4. Pada tab Dimiliki oleh Amazon, di kotak pencarian dokumen Otomasi, masukkan`AWSSupport-TroubleshootElasticBeanstalk`.
5. Pilih AWSSupport-TroubleshootElasticBeanstalkkartu, lalu pilih Berikutnya.
6. Pilih Jalankan.

7. Di bagian Parameter input:

   a. Dari AutomationAssumeRoledropdown, pilih ARN dari peran yang memungkinkan Systems Manager untuk melakukan tindakan atas nama Anda.

   b. Untuk ApplicationName, masukkan nama aplikasi Elastic Beanstalk.

   c. Untuk Nama Lingkungan, masukkan lingkungan Elastic Beanstalk.

   d. (Opsional) Untuk S3 UploaderLink, masukkan tautan jika AWS Support Engineer telah memberi Anda tautan S3 untuk pengumpulan log.

8. Pilih Eksekusi.

   Jika salah satu langkah gagal, pilih tautan di bawah kolom ID Langkah untuk langkah yang gagal. Ini menampilkan halaman detail Eksekusi untuk langkah tersebut. VerificationErrorMessageBagian ini akan menampilkan ringkasan langkah-langkah yang membutuhkan perhatian. Misalnya, `IAMPermissionCheck` dapat menampilkan pesan Peringatan. Dalam hal ini, Anda dapat memeriksa apakah peran yang dipilih di AutomationAssumeRoledropdown memiliki izin yang diperlukan.

Setelah semua langkah berhasil diselesaikan, output memberikan langkah pemecahan masalah dan rekomendasi untuk memulihkan lingkungan Anda ke keadaan sehat.

Panduan umum untuk mengatasi masalah lingkungan Elastic Beanstalk Anda

Pesan galat dapat muncul di halaman Acara di konsol, di log, atau di halaman Kesehatan. Anda juga dapat mengambil tindakan untuk pulih dari lingkungan terdegradasi yang disebabkan oleh perubahan baru-baru ini. Jika kesehatan lingkungan Anda berubah menjadi Merah, coba yang berikut ini:

- Jika operasi di lingkungan Anda menampilkan kesalahan yang berisi teks`The stack stack_id associated with environment environment-ID is in stack-status state`, lihat bantuan Memulihkan lingkungan Elastic Beanstalk Anda dari keadaan tidak valid pemecahan masalah.
- Jika operasi di lingkungan Anda mengembalikan kesalahan yang berisi teks`Environment environment-name associated CloudFormation stack stack_arn does not exist`, hentikan lingkungan Anda dan buat yang lain.
- Tinjau peristiwa lingkungan terbaru. Pesan dari Elastic Beanstalk tentang masalah deployment, beban, dan konfigurasi sering muncul di sini.

- Tinjau riwayat perubahan lingkungan terbaru. Riwayat perubahan mencantumkan semua perubahan konfigurasi yang dibuat pada lingkungan Anda dan termasuk informasi lainnya, seperti pengguna IAM yang membuat perubahan dan parameter konfigurasi yang ditetapkan.
- Tarik log untuk melihat entri berkas log terbaru. Log server web berisi informasi tentang permintaan dan kesalahan yang masuk.
- Connect ke instans dan periksa sumber daya sistem.
- Gulung kembali ke versi aplikasi yang berfungsi sebelumnya.
- Membatalkan perubahan konfigurasi terbaru atau memulihkan konfigurasi tersimpan.
- Men-deploy lingkungan baru. Jika lingkungan tampak sehat, lakukan swap CNAME untuk merutekan lalu lintas ke lingkungan baru dan terus men-debug yang sebelumnya.

Lingkungan yang mengakses rahasia dan parameter dengan variabel lingkungan

Acara: Penerapan instance gagal mendapatkan satu atau lebih rahasia

Pesan ini menunjukkan bahwa Elastic Beanstalk tidak dapat mengambil satu atau lebih rahasia yang ditentukan selama penerapan aplikasi Anda.

- Periksa apakah sumber daya yang ditentukan oleh nilai ARN dalam konfigurasi variabel lingkungan Anda ada.
- Konfirmasikan bahwa peran profil instans EC2 Elastic Beanstalk Anda memiliki izin IAM yang diperlukan untuk mengakses sumber daya.
- Jika peristiwa ini dipicu melalui `RestartAppServer` operasi, setelah masalah diperbaiki, coba lagi `RestartAppServer` panggilan untuk menyelesaikan masalah.
- Jika acara dipicu melalui `UpdateEnvironment` panggilan, coba lagi `UpdateEnvironment` operasi.

Untuk contoh perintah ini, lihat AWS CLI contoh untuk Elastic Beanstalk. Untuk informasi selengkapnya tentang tindakan API untuk operasi ini, lihat Referensi AWS Elastic Beanstalk API.

Peristiwa: Penerapan instance mendeteksi satu atau lebih nilai lingkungan multiline, yang tidak didukung untuk platform ini

Variabel multiline tidak didukung untuk platform Amazon Linux 2, tidak termasuk platform Docker yang dikelola Docker dan ECS. Untuk opsi yang tersedia untuk melanjutkan, lihatNilai multiline.

Acara: CreateEnvironment gagal saat rahasia ditentukan

Ketika `CreateEnvironment` gagal dan Anda memiliki rahasia sebagai variabel lingkungan, Anda perlu mengatasi masalah mendasar dan kemudian menggunakannya `UpdateEnvironment` untuk menyelesaikan pengaturan lingkungan. Jangan gunakan`RestartAppServer`, karena tidak akan cukup untuk memunculkan lingkungan dalam situasi ini. Untuk contoh perintah ini, lihat AWS CLI contoh untuk Elastic Beanstalk. Untuk informasi selengkapnya tentang tindakan API untuk operasi ini, lihat Referensi AWS Elastic Beanstalk API.

Pembuatan lingkungan dan peluncuran instans

Peristiwa: Gagal Meluncurkan Lingkungan

Peristiwa ini terjadi ketika Elastic Beanstalk mencoba untuk meluncurkan lingkungan dan menemukan kegagalan di sepanjang jalan. Peristiwa sebelumnya di halaman Peristiwa akan memberi tahu Anda akar masalahnya.

Peristiwa: Membuat operasi lingkungan selesai, tetapi dengan batas waktu perintah. Coba tingkatkan batas waktu periode.

Aplikasi Anda mungkin membutuhkan waktu lama untuk men-deploy jika Anda menggunakan file konfigurasi yang menjalankan perintah di instans, mengunduh file besar, atau memasang paket. Tingkatkan batas wkatu perintah untuk memberikan aplikasi Anda lebih banyak waktu untuk mulai berjalan selama deployment.

Peristiwa: Sumber daya berikut gagal dibuat: [AWSEBInstanceLaunchWaitCondition]

Pesan ini menunjukkan bahwa EC2 instans Amazon lingkungan Anda tidak berkomunikasi dengan Elastic Beanstalk bahwa mereka berhasil diluncurkan. Hal ini dapat terjadi jika instans tidak memiliki konektivitas Internet. Jika Anda mengonfigurasi lingkungan Anda untuk meluncurkan instans di subnet VPC pribadi, pastikan subnet memiliki NAT untuk mengizinkan instans terhubung ke Elastic Beanstalk.

Peristiwa: Peran Layanan diperlukan di wilayah ini. Harap tambahkan opsi Peran Layanan ke lingkungan.

Elastic Beanstalk menggunakan peran layanan untuk memantau sumber daya di lingkungan dan mendukung pembaruan platform terkelola. Lihat Mengelola peran layanan Elastic Beanstalk untuk informasi selengkapnya.

Deployment

Masalah: Aplikasi menjadi tidak tersedia selama deployment

Karena Elastic Beanstalk menggunakan proses peningkatan drop-in, mungkin ada waktu henti beberapa detik. Gunakan deployment bergulir untuk meminimalkan efek deployment di lingkungan produksi Anda.

Acara: Gagal membuat versi aplikasi AWS Elastic Beanstalk

Paket sumber aplikasi Anda mungkin terlalu besar, atau Anda mungkin telah mencapai kuota versi aplikasi.

Peristiwa: Pembaruan operasi lingkungan selesai, tetapi dengan batas waktu perintah. Coba tingkatkan batas waktu periode.

Aplikasi Anda mungkin membutuhkan waktu lama untuk men-deploy jika Anda menggunakan file konfigurasi yang menjalankan perintah di instans, mengunduh file besar, atau memasang paket. Tingkatkan batas wkatu perintah untuk memberikan aplikasi Anda lebih banyak waktu untuk mulai berjalan selama deployment.

Kondisi

Peristiwa: Penggunaan CPU melebihi 95,00%

Coba jalankan lebih banyak instans, atau pilih tipe instans yang berbeda.

Acara: Elastic Load Balancer awseb- *myapp* Memiliki Nol Contoh Sehat

Jika aplikasi Anda tampaknya berfungsi, pastikan bahwa URL pemeriksaan kondisi aplikasi Anda dikonfigurasi dengan benar. Jika tidak, periksa layar Kondisi dan log lingkungan untuk informasi selengkapnya.

Acara: Elastic Load Balancer awseb - Tidak Dapat Ditemukan *myapp*

Penyeimbang beban lingkungan Anda mungkin telah dihapus out-of-band. Hanya buat perubahan pada sumber daya lingkungan Anda dengan opsi konfigurasi dan perpanjangan yang disediakan oleh Elastic Beanstalk. Bangun kembali lingkungan Anda atau luncurkan yang baru.

Acara: Kegagalan Peluncuran EC2 Instance. Menunggu EC2 Instance Baru untuk Diluncurkan...

Ketersediaan untuk tipe instans lingkungan Anda mungkin rendah, atau Anda mungkin telah mencapai kuota instans akun Anda. Periksa dasbor kesehatan layanan untuk memastikan bahwa layanan Elastic Compute Cloud (Amazon EC2) berwarna hijau, atau minta peningkatan kuota.

Konfigurasi

Acara: Tumpukan *`stack_id`* yang terkait dengan lingkungan *`environment-ID`* dalam *`stack-status`* status

AWS CloudFormation Tumpukan yang mendasari lingkungan Anda mungkin dalam status *_FAILED. Status ini harus diperbaiki untuk melanjutkan operasi Elastic Beanstalk di lingkungan Anda. Untuk informasi selengkapnya, lihat Memulihkan lingkungan Elastic Beanstalk Anda dari keadaan tidak valid.

Peristiwa: Anda tidak dapat mengonfigurasi lingkungan Elastic Beanstalk dengan nilai-nilai untuk kedua opsi Target Elastic Load Balancing dan opsi URL Pemeriksaan Kondisi Aplikasi

Opsi `Target` di namespace `aws:elb:healthcheck` tidak lagi digunakan. Menghapus namespace opsi `Target`) dari lingkungan Anda dan coba perbarui lagi.

Acara: ELB tidak dapat dilampirkan ke beberapa subnet di AZ yang sama

Pesan ini dapat dilihat jika Anda mencoba untuk memindahkan penyeimbang beban antar subnet di Availability Zone yang sama. Mengubah subnet di penyeimbang beban perlu memindahkannya keluar dari availability zone aslinya dan kemudian kembali ke aslinya dengan subnet yang diinginkan. Selama proses berlangsung, semua instans Anda akan dimigrasikan antara AZs, menyebabkan downtime yang signifikan. Sebagai gantinya, pertimbangkan untuk membuat lingkungan baru dan lakukan penggantian CNAME.

Mengatasi masalah kontainer Docker

Peristiwa: Gagal menarik gambar Docker: terbaru: Nama repositori tidak valid (), hanya [a-z0-9-_.] yang diizinkan. Ekor log untuk detail selengkapnya.

Periksa sintaks file `dockerrun.aws.json` menggunakan validator JSON. Verifikasi juga konten dockerfile terhadap persyaratan yang dijelaskan di Mempersiapkan image Docker Anda untuk penyebaran ke Elastic Beanstalk

Peristiwa: Tidak ada arahan EXPOSE ditemukan di Dockerfile, batalkan deployment

`Dockerfile` atau file `dockerrun.aws.json` tidak menyatakan port kontainer. Gunakan petunjuk `EXPOSE` (`Dockerfile`) atau blok `Ports` (file `dockerrun.aws.json`) untuk mengekspos port lalu lintas masuk.

Acara: Gagal mengunduh kredensyal *`repository`* otentikasi dari *`bucket name`*

`dockerrun.aws.json`Ini menyediakan bucket and/or S3 EC2 key pair yang tidak valid untuk file tersebut. `.dockercfg` Atau, profil instance tidak memiliki GetObject otorisasi untuk bucket S3. Verifikasi bahwa `.dockercfg` file tersebut berisi bucket S3 dan EC2 key pair yang valid. Berikan izin untuk tindakan `s3:GetObject` ke IAM role di profil instans. Untuk detail, lanjutkan ke Mengelola profil instans Elastic Beanstalk

Peristiwa: Aktivitas pelaksanaan gagal, karena: PERINGATAN: File konfigurasi autentikasi tidak valid

File autentikasi Anda (`config.json`) tidak diformat dengan benar. Lihat Menggunakan gambar dari repositori pribadi di Elastic Beanstalk

Pertanyaan yang Sering Diajukan

Pertanyaan: Bagaimana cara mengubah URL aplikasi saya dari myapp.us-west-2.elasticbeanstalk.com ke www.myapp.com?

Di server DNS, daftarkan data catatan CNAME, seperti **`www.mydomain.com CNAME mydomain.elasticbeanstalk.com`**.

Pertanyaan: Bagaimana cara menentukan Availability Zone spesifik untuk aplikasi Elastic Beanstalk saya?

Anda dapat memilih Availability Zone tertentu dengan menggunakan plugin APIs, CLI, Eclipse, atau Visual Studio. Untuk petunjuk tentang cara menggunakan konsol Elastic Beanstalk untuk menentukan Availability Zone, lihat Auto Scaling instance lingkungan Elastic Beanstalk Anda.

Pertanyaan: Bagaimana cara mengubah tipe instans lingkungan saya?

Untuk mengubah tipe instans lingkungan Anda, lanjutkan ke halaman konfigurasi lingkungan dan pilih Edit di kategori konfigurasi Instans. Kemudian, pilih tipe instans baru dan pilih Terapkan untuk memperbarui lingkungan Anda. Setelah ini, Elastic Beanstalk mengakhiri semua instans berjalan dan menggantikannya dengan yang baru.

Pertanyaan: Bagaimana cara menentukan apakah ada yang membuat perubahan konfigurasi ke lingkungan?

Untuk melihat informasi ini, di panel navigasi konsol Elastic Beanstalk, pilih Riwayat perubahan untuk menampilkan daftar perubahan konfigurasi untuk semua lingkungan. Daftar ini mencakup tanggal dan waktu perubahan, parameter konfigurasi dan nilai yang diubah, dan pengguna IAM yang membuat perubahan. Untuk informasi selengkapnya, lihat Riwayat perubahan.

Pertanyaan: Dapatkah saya mencegah volume Amazon EBS dihapus ketika instans diakhiri?

Instans di lingkungan Anda menggunakan Amazon EBS untuk penyimpanan; namun, volume akar dihapus ketika instans diakhiri oleh Auto Scaling. Kami tidak merekomendasikan Anda menyimpan status atau data lain di instans Anda. Jika diperlukan, Anda dapat mencegah volume dihapus dengan AWS CLI: `$ aws ec2 modify-instance-attribute -b '/dev/sdc=<vol-id>:false` seperti yang dijelaskan dalam AWS CLI Referensi.

Pertanyaan: Bagaimana cara menghapus informasi pribadi dari aplikasi Elastic Beanstalk saya?

AWS sumber daya yang digunakan aplikasi Elastic Beanstalk Anda dapat menyimpan informasi pribadi. Ketika Anda mengakhiri lingkungan Anda, Elastic Beanstalk mengakhiri semua sumber daya yang dibuatnya. Sumber daya yang Anda tambahkan menggunakan file konfigurasi juga diakhiri. Namun, jika Anda membuat AWS sumber daya di luar lingkungan Elastic Beanstalk dan menghubungkannya dengan aplikasi Anda, Anda mungkin perlu memeriksa secara manual bahwa informasi pribadi yang mungkin disimpan aplikasi Anda tidak disimpan. Sepanjang panduan developer ini, setiap kali kita membahas pembuatan sumber daya tambahan, kami juga menyebutkan kapan Anda harus mempertimbangkan untuk menghapusnya.

Kesalahan Umum

Topik ini berisi daftar pesan kesalahan umum yang ditemui saat menggunakan EB CLI dan solusi yang memungkinkan. Jika Anda mengalami pesan kesalahan yang tidak ditampilkan di sini, gunakan tautan Umpan Balik untuk memberi tahu kami tentang hal itu.

ERROR: Terjadi kesalahan saat menangani perintah git. Kode kesalahan: Kesalahan 128: fatal: Bukan nama objek valid HEAD

Penyebab: Pesan kesalahan ini ditampilkan ketika Anda telah menginisialisasi repositori Git tetapi belum berkomitmen. EB CLI mencari revisi HEAD ketika folder proyek Anda berisi repositori Git.

Solusi: tambahkan file di folder proyek Anda ke area persiapan dan lakukan:

```
~/my-app$ git add .
~/my-app$ git commit -m "First commit"
```

KESALAHAN: Cabang ini tidak memiliki lingkungan default. Anda harus menentukan lingkungan dengan mengetikkan “status eb my-env-name" atau mengatur lingkungan default dengan mengetik “eb use my-env-name”.

Penyebab: Ketika Anda membuat cabang baru di git, itu tidak terlampir ke lingkungan Elastic Beanstalk secara default.

Solusi: Jalankan eb list untuk melihat daftar lingkungan yang tersedia. Kemudian jalankan eb use ***env-name*** untuk menggunakan salah satu lingkungan yang tersedia.

EROR: 2.0+ Platform memerlukan peran layanan. Anda dapat memberikan satu dengan --opsi peran layanan

Penyebab: Jika Anda menentukan nama lingkungan dengan eb create (misalnya, eb create my-env), EB CLI tidak akan mencoba membuat peran layanan untuk Anda. Jika Anda tidak memiliki peran layanan default, kesalahan di atas akan ditampilkan.

Solusi: Jalankan eb create tanpa nama lingkungan dan ikuti prompt untuk membuat peran layanan default.

Kesalahan penerapan

Penerapan Elastic Beanstalk Anda mungkin merespons dengan respons 404 (jika aplikasi Anda gagal diluncurkan) atau 500 (jika aplikasi Anda gagal selama runtime). Untuk mengatasi banyak masalah umum, Anda dapat menggunakan EB CLI untuk memeriksa status penerapan, melihat lognya, mendapatkan akses ke instans EC2 Anda dengan SSH, atau membuka halaman Management Console untuk lingkungan aplikasi AWS Anda.

Untuk menggunakan EB CLI untuk membantu memecahkan masalah deployment

1. Jalankan eb status untuk melihat status penyebaran dan kesehatan EC2 host Anda saat ini. Misalnya:

```
$ eb status --verbose

Environment details for: python_eb_app
  Application name: python_eb_app
  Region: us-west-2
  Deployed Version: app-150206_035343
  Environment ID: e-wa8u6rrmqy
  Platform: 64bit Amazon Linux 2014.09 v1.1.0 running Python 2.7
  Tier: WebServer-Standard-
  CNAME: python_eb_app.elasticbeanstalk.com
  Updated: 2015-02-06 12:00:08.557000+00:00
  Status: Ready
  Health: Green
  Running instances: 1
      i-8000528c: InService
```

> **Note**
>
> Menggunakan switch `--verbose` memberikan informasi tentang status instans berjalan Anda. Tanpa itu, eb status hanya akan mencetak informasi umum tentang lingkungan Anda.

2. Jalankan eb health untuk melihat informasi kondisi tentang lingkungan Anda:

```
$ eb health --refresh
 elasticBeanstalkExa-env                                  Degraded
 2016-03-28 23:13:20
WebServer
    Ruby 2.1 (Puma)
  total      ok    warning  degraded  severe    info   pending  unknown
    5        2        0        2        1        0        0        0

  instance-id   status     cause
    Overall     Degraded  Incorrect application version found on 3 out of 5
 instances. Expected version "Sample Application" (deployment 1).
  i-d581497d    Degraded  Incorrect application version "v2" (deployment 2).
 Expected version "Sample Application" (deployment 1).
  i-d481497c    Degraded  Incorrect application version "v2" (deployment 2).
 Expected version "Sample Application" (deployment 1).
  i-136e00c0    Severe    Instance ELB health has not been available for 5 minutes.
  i-126e00c1    Ok
```

```
 i-8b2cf575    Ok

 instance-id   r/sec    %2xx   %3xx   %4xx   %5xx      p99      p90      p75
p50     p10
   Overall     646.7   100.0    0.0    0.0    0.0    0.003    0.002    0.001
0.001   0.000
 i-dac3f859    167.5    1675      0      0      0    0.003    0.002    0.001
0.001   0.000
 i-05013a81    161.2    1612      0      0      0    0.003    0.002    0.001
0.001   0.000
 i-04013a80    0.0         -      -      -      -        -        -       -
-       -
 i-3ab524a1    155.9    1559      0      0      0    0.003    0.002    0.001
0.001   0.000
 i-bf300d3c    162.1    1621      0      0      0    0.003    0.002    0.001
0.001   0.000

 instance-id   type       az   running     load 1  load 5      user%  nice%
system%  idle%   iowait%
 i-d581497d    t2.micro   1a   25 mins       0.16     0.1        7.0    0.0
1.7   91.0       0.1
 i-d481497c    t2.micro   1a   25 mins       0.14     0.1        7.2    0.0
1.6   91.1       0.0
 i-136e00c0    t2.micro   1b   25 mins        0.0    0.01        0.0    0.0
0.0   99.9       0.1
 i-126e00c1    t2.micro   1b   25 mins       0.03    0.08        6.9    0.0
2.1   90.7       0.1
 i-8b2cf575    t2.micro   1c   1 hour        0.05    0.41        6.9    0.0
2.0   90.9       0.0

 instance-id   status     id   version             ago
     deployments
 i-d581497d    Deployed   2    v2                  9 mins
 i-d481497c    Deployed   2    v2                  7 mins
 i-136e00c0    Failed     2    v2                  5 mins
 i-126e00c1    Deployed   1    Sample Application  25 mins
 i-8b2cf575    Deployed   1    Sample Application  1 hour
```

Contoh di atas menunjukkan lingkungan dengan lima instans yang deployment versi "v2" gagal di instans ketiga. Setelah deployment gagal, versi yang diharapkan diatur ulang ke versi terakhir yang berhasil, yang dalam hal ini adalah "Aplikasi Sampel" dari deployment pertama. Lihat Menggunakan EB CLI untuk memantau kondisi lingkungan untuk informasi selengkapnya.

3. Jalankan eb logs untuk mengunduh dan melihat log yang terkait dengan deployment aplikasi Anda.

```
$ eb logs
```

4. Jalankan eb ssh untuk terhubung dengan EC2 instance yang menjalankan aplikasi Anda dan memeriksanya secara langsung. Di instans, aplikasi yang di-deploy dapat ditemukan di direktori `/opt/python/current/app`, dan lingkungan Python Anda akan ditemukan di `/opt/python/run/venv/`.
5. Jalankan eb console untuk melihat lingkungan aplikasi Anda di Konsol Manajemen AWS. Anda dapat menggunakan antarmuka web untuk dengan mudah memeriksa berbagai aspek deployment Anda, termasuk konfigurasi, status, peristiwa, log aplikasi Anda. Anda juga dapat mengunduh versi aplikasi saat ini atau sebelumnya yang telah Anda deploy ke server.

Sumber daya Elastic Beanstalk

Sumber daya terkait berikut dapat membantu Anda ketika bekerja dengan layanan ini.

- Referensi API Elastic Beanstalk Deskripsi lengkap dari semua SOAP dan Query. APIs Selain itu, referensi tersebut berisi daftar semua jenis data SOAP.
- elastic-beanstalk-samples on GitHub — Sebuah GitHub repositori dengan file konfigurasi sampel Elastic Beanstalk (.ebextensions). `README.md`File repositori memiliki tautan ke GitHub repositori tambahan dengan aplikasi sampel.
- FAQ Teknis Elastic Beanstalk – Pertanyaan teratas yang telah ditanyakan developer tentang produk ini.
- AWS Elastic Beanstalk Catatan Rilis - Detail tentang fitur, pembaruan, dan perbaikan baru dalam layanan Elastic Beanstalk, platform, konsol, dan rilis EB CLI.

- Kelas & Lokakarya - Tautan ke kursus berbasis peran dan khusus, selain laboratorium mandiri untuk membantu mempertajam keterampilan Anda AWS dan mendapatkan pengalaman praktis.
- AWS Pusat Pengembang — Jelajahi tutorial, unduh alat, dan pelajari tentang acara AWS pengembang.
- AWS Alat Pengembang — Tautan ke alat pengembang SDKs, toolkit IDE, dan alat baris perintah untuk mengembangkan dan mengelola AWS aplikasi.
- Memulai Pusat Sumber Daya — Pelajari cara menyiapkan Akun AWS, bergabung dengan AWS komunitas, dan meluncurkan aplikasi pertama Anda.
- Hands-On Tutorial - Ikuti step-by-step tutorial untuk meluncurkan aplikasi pertama Anda. AWS
- AWS Whitepaper — Tautan ke daftar lengkap AWS whitepaper teknis, yang mencakup topik-topik seperti arsitektur, keamanan, dan ekonomi dan ditulis oleh AWS Solutions Architects atau pakar teknis lainnya.
- AWS Dukungan Pusat — Hub untuk membuat dan mengelola AWS Dukungan kasus Anda. Juga termasuk tautan ke sumber daya bermanfaat lainnya, seperti forum, teknis FAQs, status kesehatan layanan, dan AWS Trusted Advisor.
- Dukungan— Halaman web utama untuk informasi tentang Dukungan, saluran dukungan respons cepat untuk membantu Anda membangun dan menjalankan aplikasi di cloud. one-on-one
- Hubungi Kami – Titik kontak pusat untuk pertanyaan tentang tandaihan AWS , akun, peristiwa, penyalahgunaan, dan masalah lainnya.

- AWS Ketentuan Situs — Informasi terperinci tentang hak cipta dan merek dagang kami; akun, lisensi, dan akses situs Anda; dan topik lainnya.

Aplikasi sampel

Berikut ini adalah tautan unduhan ke aplikasi sampel yang di-deploy sebagai bagian dari Pelajari cara memulai dengan Elastic Beanstalk.

> **Note**
>
> Beberapa sampel menggunakan fitur yang mungkin telah dirilis sejak rilis platform yang sedang Anda gunakan. Jika sampel gagal dijalankan, coba perbarui platform Anda ke versi saat ini, seperti yang dijelaskan di the section called “Platform yang didukung”.

- Docker – docker.zip
- Multicontainer Docker - 2.zip docker-multicontainer-v
- Docker yang telah dikonfigurasi sebelumnya (Glassfish) - 1.zip docker-glassfish-v
- Go – go.zip
- Corretto – corretto.zip
- Tomcat – tomcat.zip
- .NET Core di Linux — dotnet-core-linux.zip
- .NET Inti — dotnet-asp-windows.zip
- Node.js – nodejs.zip
- PHP – php.zip
- Python – python.zip
- Ruby – ruby.zip

AWS SDK untuk Java

AWS SDK untuk Java Ini menyediakan Java API yang dapat Anda gunakan untuk membangun aplikasi yang menggunakan layanan AWS infrastruktur. Dengan ini AWS SDK untuk Java, Anda dapat memulai dalam hitungan menit dengan satu paket yang dapat diunduh yang mencakup pustaka AWS Java, contoh kode, dan dokumentasi.

AWS SDK untuk Java Membutuhkan Kit Pengembangan J2SE. Anda dapat mengunduh perangkat lunak Java terbaru dari http://developers.sun.com/downloads/. SDK juga membutuhkan Apache Commons (Codec, HTTPClient, dan Logging) dan paket pihak ketiga Saxon-HE, yang termasuk dalam direktori pihak ketiga SDK.

Untuk informasi selengkapnya, lihat AWS SDK for Java. Untuk informasi selengkapnya tentang SDK, contoh kode, dokumentasi, alat, dan sumber daya tambahan, lihat Java di Pusat AWS Pengembang.

AWS SDK untuk .NET

Dengan ini AWS SDK untuk .NET, Anda dapat memulai dengan cepat dengan satu paket yang dapat diunduh yang mencakup pustaka AWS .NET, contoh kode, dan dokumentasi.

Untuk informasi selengkapnya, lihat AWS SDK for .NET. Untuk versi .NET Framework dan Visual Studio yang didukung, lihat Panduan Developer AWS SDK untuk .NET.

Untuk informasi selengkapnya tentang SDK, contoh kode, dokumentasi, alat, dan sumber daya tambahan, lihat .NET di Pusat AWS Pengembang.

AWS Toolkit for Visual Studio

Dengan AWS Toolkit for Visual Studio plug-in, Anda dapat menyebarkan aplikasi.NET yang ada ke Elastic Beanstalk. Anda juga dapat membuat proyek menggunakan AWS templat yang telah dikonfigurasi sebelumnya dengan. AWS SDK untuk .NET

Untuk informasi prasyarat dan pemasangan, lihat AWS Toolkit for Visual Studio. Untuk memulai pembuatan aplikasi Elastic Beanstalk Anda menggunakan Visual Studio, lihat Menyebarkan aplikasi.NET Windows dengan Elastic Beanstalk.

AWS SDK untuk JavaScript di Node.js

Dengan AWS SDK for JavaScript di Node.js, Anda dapat memulai dengan cepat dengan satu paket yang dapat diunduh yang mencakup pustaka AWS Node.js, contoh kode, dan dokumentasi.

Untuk informasi selengkapnya, lihat AWS SDK untuk JavaScript di Node.js.

AWS SDK untuk PHP

Dengan ini AWS SDK untuk PHP, Anda dapat memulai dengan cepat dengan satu paket yang dapat diunduh yang mencakup perpustakaan AWS PHP, contoh kode, dan dokumentasi.

Untuk informasi selengkapnya, lihat AWS SDK for PHP. Untuk informasi selengkapnya tentang SDK, contoh kode, dokumentasi, alat, dan sumber daya tambahan, lihat PHP di Pusat AWS Pengembang.

AWS SDK for Python (Boto)

Dengan ini AWS SDK for Python (Boto), Anda dapat memulai dengan cepat dengan satu paket yang dapat diunduh yang mencakup pustaka AWS Python, contoh kode, dan dokumentasi. Anda dapat membangun aplikasi Python di atas APIs yang menghilangkan kompleksitas pengkodean langsung terhadap antarmuka layanan web.

all-in-onePustaka menyediakan Python yang ramah pengembang APIs yang menyembunyikan banyak tugas tingkat rendah yang terkait dengan pemrograman untuk AWS Cloud, termasuk otentikasi, percobaan ulang permintaan, dan penanganan kesalahan. SDK memberikan contoh praktis di Python tentang bagaimana menggunakan pustaka untuk membangun aplikasi.

Untuk informasi tentang Boto, contoh kode, dokumentasi, alat, dan sumber daya tambahan, lihat Python AWS di Pusat Pengembang.

AWS SDK untuk Ruby

Dengan pustaka AWS Ruby, contoh kode, dan dokumentasi, Anda dapat membangun aplikasi Ruby di atas APIs yang menghilangkan kompleksitas pengkodean langsung terhadap antarmuka layanan web.

all-in-onePustaka menyediakan Ruby yang ramah pengembang APIs yang menyembunyikan banyak tugas tingkat rendah yang terkait dengan pemrograman untuk AWS Cloud, termasuk otentikasi, percobaan ulang permintaan, dan penanganan kesalahan. SDK memberikan contoh praktis di Ruby tentang cara menggunakan pustaka untuk membangun aplikasi.

Untuk informasi tentang SDK, contoh kode, dokumentasi, alat, dan sumber daya tambahan, lihat Ruby di Pusat AWS Pengembang.

Riwayat dokumen

Tabel berikut menjelaskan perubahan penting pada Panduan AWS Elastic Beanstalk Pengembang sejak April 2024.

Perubahan	Deskripsi	Tanggal
Topik yang direvisi: Apa itu? AWS Elastic Beanstalk	Diagram arsitektur baru dan pengantar kental untuk kejelasan.	Juni 5, 2025
Topik yang direvisi: Menyiapkan CLI EB	Petunjuk pemasangan yang kental dan terfokus.	Juni 5, 2025
Topik yang direvisi: PHP Quickstart	Menambahkan langkah untuk memperbarui dan menyebarkan kembali aplikasi dan meningkatkan beberapa instruksi.	Juni 5, 2025
Topik yang direvisi: Memulai tutorial	Merampingkan tutorial Memulai untuk kejelasan dan orientasi yang lebih cepat.	Juni 5, 2025
Konten yang direorganisasi	Struktur konten yang direorganisasi untuk menyoroti informasi penting dan meningkatkan kemampuan untuk ditemukan.	Juni 5, 2025
Topik baru: Mengelola grup EC2 keamanan	Elastic Beanstalk menambahkan dukungan untuk memilih keluar lingkungan dari grup keamanan default. EC2	April 30, 2025
Topik baru: Memigrasi aplikasi IIS ke Elastic Beanstalk	Elastic Beanstalk menambahkan fitur untuk memigrasikan aplikasi Windows IIS Anda ke. AWS Elastic Beanstalk	April 18, 2025

Topik baru: Menggunakan Elastic Beanstalk dengan dan AWS Secrets Manager Parameter Store AWS Systems Manager	Elastic Beanstalk menambahkan dukungan AWS Secrets Manager untuk referensi AWS Systems Manager rahasia dan parameter Parameter Store dengan variabel lingkungan.	Maret 31, 2025
AWS pembaruan kebijakan terkelola untuk`AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy` .	Izin yang diperbarui dalam kebijakan AWS terkelola.	Februari 27, 2025
Konfigurasi peluncuran yang tidak digunakan lagi demi templat peluncuran	Merevisi topik ini untuk menjelaskan bagaimana lingkungan Elastic Beanstalk dipengaruhi oleh penghentian EC2 Amazon Auto Scaling terhadap konfigurasi peluncuran yang mendukung template peluncuran. Kami juga merevisi konten lain yang terkait dengan konfigurasi peluncuran dan templat peluncuran.	Januari 23, 2025
Topik baru: Menggunakan Peluncuran EC2 Cepat dengan cabang platform Windows	Rilis platform Elastic Beanstalk Windows AMIs menyertakan EC2 basis dengan Peluncuran Cepat diaktifkan.	Januari 22, 2025
Topik baru: Strategi alokasi Instans Spot	Elastic Beanstalk menambahkan dukungan untuk konfigurasi Strategi Alokasi Spot selama pembuatan lingkungan.	Januari 15, 2025

AdministratorAccess-AWSElasticBeanstalk AWS kebijakan terkelola	Izin yang diperbarui dalam kebijakan AWS terkelola.	Desember 11, 2024
Rekomendasi untuk lingkungan gelombang pertama Graviton arm64 ke Bab konten yang diarsipkan	Rekomendasi EB yang dipindahkan untuk lingkungan gelombang pertama Graviton arm64 ke Bab konten yang diarsipkan	Oktober 5, 2024
Luncurkan template	Topik baru: Luncurkan templat. Topik ini menjelaskan bagaimana Amazon EC2 Auto Scaling menghapus konfigurasi peluncuran secara bertahap demi template peluncuran.	Oktober 1, 2024
QuickStartfor Python	Topik baru: QuickStart untuk Python	September 24, 2024
Izin bucket Amazon S3 yang diperlukan untuk kebijakan titik akhir VPC yang membatasi	Topik baru: Izin bucket Amazon S3 yang diperlukan untuk kebijakan titik akhir VPC yang membatasi	September 18, 2024
QuickStart untuk Java di Tomcat	Topik baru: QuickStart untuk aplikasi Java JSP berjalan di Tomcat	September 12, 2024
QuickStart untuk Java	Topik baru: QuickStart untuk Java	September 12, 2024

Menggunakan Amazon EC2 Systems Manager untuk repositori pribadi Docker	Topik yang dipulihkan: Menggunakan Amazon EC2 Systems Manager untuk repositori pribadi Docker. Mengganti topik Menggunakan repositori pribadi AWS Secrets Manager for Docker hingga integrasi Elastic Beanstalk/ SSM didukung.	September 8, 2024
EB CLI 2.6 (pensiun) ke Bab konten yang diarsipkan	Memindahkan EB CLI 2.6 (pensiun) ke Bab konten yang diarsipkan	Agustus 15, 2024
EB API (pensiun) ke Bab konten yang diarsipkan	Memindahkan EB API (pensiun) ke Bab konten yang diarsipkan	Agustus 15, 2024
Menyebarkan aplikasi Elastic Beanstalk dari wadah Docker	Diperbarui dan direorganisasi: Menyebarkan aplikasi Elastic Beanstalk dari wadah Docker	Agustus 15, 2024
Konten yang diarsipkan	Bab baru: Konten yang diarsipkan	Agustus 15, 2024
QuickStart untuk Windows ASP.NET	Topik baru: QuickStart untuk Windows ASP.NET	Juli 5, 2024
QuickStart untuk .NET Core di Windows	Topik baru: QuickStart untuk.NET Core di Windows	Juni 28, 2024
QuickStart untuk Docker	Topik baru: QuickStart untuk Docker	Juni 19, 2024
Mencegah akses bucket Amazon S3 lintas lingkungan	Topik baru: Mencegah akses bucket Amazon S3 lintas lingkungan	12 Juni 2024

QuickStart untuk .NET Core di Linux	Topik baru: QuickStart untuk.NET Core di Windows	28 Mei 2024
QuickStart untuk PHP	Topik baru: QuickStart untuk PHP	10 Mei 2024
QuickStart untuk Node.js	Topik baru: QuickStart untuk Node.js	5 Mei 2024
QuickStart untuk Go	Topik baru: QuickStart untuk Go	5 Mei 2024
Jadwal rilis platform Elastic Beanstalk	Ditambahkan topik baru yang mencakup jadwalRilis cabang platform mendatang. Pindah Jadwal cabang platform yang pensiun dan Sejarah cabang platform pensiunan ke topik ini.	1 Mei 2024
AWSElasticBeanstalkRoleCore AWS kebijakan terkelola	Izin yang diperbarui dalam kebijakan AWS terkelola.	April 30, 2024
AWSElasticBeanstalkManagedUpdatesServiceRolePolicy AWS kebijakan terkelola	Izin yang diperbarui dalam kebijakan AWS terkelola.	April 30, 2024
AWSElasticBeanstalkManagedUpdatesInternalServiceRolePolicy AWS kebijakan terkelola	Izin yang diperbarui dalam kebijakan AWS terkelola.	April 30, 2024
AWSElasticBeanstalkMaintenance AWS kebijakan terkelola	Izin yang diperbarui dalam kebijakan AWS terkelola.	April 30, 2024

AWSElasticBeanstalkInternalMaintenanceRolePolicy AWS kebijakan terkelola	Izin yang diperbarui dalam kebijakan AWS terkelola.	April 30, 2024

Terjemahan disediakan oleh mesin penerjemah. Jika konten terjemahan yang diberikan bertentangan dengan versi bahasa Inggris aslinya, utamakan versi bahasa Inggris.