aws

Panduan Pengguna

AWS Layanan Injeksi Kesalahan

AWS Layanan Injeksi Kesalahan: Panduan Pengguna

Table of Contents

Apa itu AWS FIS? 1
Konsep 1
Tindakan 2
Target 2
Hentikan kondisi 2
Didukung Layanan AWS 3
Akses AWS FIS 3
Harga 4
Merencanakan eksperimen Anda 5
Prinsip dan pedoman dasar 5
Pedoman perencanaan eksperimen 7
Komponen template percobaan 9
Sintaks template 9
Memulai 10
Tindakan 10
Sintaks tindakan 11
Pengidentifikasi tindakan 12
Parameter tindakan 12
Target aksi 12
Durasi tindakan 13
Contoh tindakan 14
Target 17
Sintaks target 17
Jenis sumber daya 19
Identifikasi sumber daya target 20
Mode pemilihan 24
Contoh target 25
Contoh filter 26
Hentikan kondisi 31
Stop sintaks kondisi 31
Pelajari selengkapnya 32
Peran percobaan 32
Prasyarat 33
Opsi 1: Buat peran eksperimen dan lampirkan kebijakan AWS terkelola 34

Opsi 2: Buat peran eksperimen dan tambahkan dokumen kebijakan sebaris 35
Konfigurasi laporan eksperimen 37
Sintaks konfigurasi laporan eksperimen 39
Izin laporan eksperimen 41
Praktik terbaik laporan eksperimen 43
Opsi percobaan 43
Penargetan akun 44
Mode resolusi target kosong 45
Mode tindakan 46
Referensi tindakan 47
Tindakan injeksi kesalahan 48
aws:fis:inject-api-internal-error 48
aws:fis:inject-api-throttle-error 49
aws:fis:inject-api-unavailable-error 49
Tindakan pemulihan 50
aws:arc:start-zonal-autoshift 50
Tunggu tindakan 52
aws:fis:wait 52
CloudWatch Tindakan Amazon 52
aws:cloudwatch:assert-alarm-state 52
Tindakan Amazon DynamoDB 53
aws:dynamodb:global-table-pause-replication 53
Tindakan Amazon EBS 55
aws:ebs:pause-volume-io 55
EC2 Tindakan Amazon 56
aws:ec2:api-insufficient-instance-capacity-error 56
aws:ec2:asg-insufficient-instance-capacity-error 57
aws:ec2:reboot-instances 58
aws:ec2:send-spot-instance-interruptions 58
aws:ec2:stop-instances 59
aws:ec2:terminate-instances 60
Tindakan Amazon ECS 61
aws:ecs:drain-container-instances 61
aws:ecs:stop-task 62
aws:ecs:task-cpu-stress 63
aws:ecs:task-io-stress 63

aws:ecs:task-kill-process 64
aws:ecs:task-network-blackhole-port 65
aws:ecs:task-network-latency 66
aws:ecs:task-network-packet-loss 67
Tindakan Amazon EKS 68
aws:eks:inject-kubernetes-custom-resource 68
aws:eks:pod-cpu-stress 70
aws:eks:pod-delete 71
aws:eks:pod-io-stress 72
aws:eks:pod-memory-stress 73
aws:eks:pod-network-blackhole-port 74
aws:eks:pod-network-latency 75
aws:eks:pod-network-packet-loss 76
aws:eks:terminate-nodegroup-instances 78
ElastiCache Tindakan Amazon 78
aws:elasticache:replicationgroup-interrupt-az-power 78
AWS Lambda tindakan 79
aws:lambda:invocation-add-delay 80
aws:lambda:invocation-error 80
aws:lambda:invocation-http-integration-response 81
Tindakan jaringan 82
aws:network:disrupt-connectivity 82
aws:network:route-table-disrupt-cross-region-connectivity 84
aws:network:transit-gateway-disrupt-cross-region-connectivity 85
Tindakan Amazon RDS 86
aws:rds:failover-db-cluster 86
aws:rds:reboot-db-instances 87
Tindakan Amazon S3 87
aws:s3:bucket-pause-replication 88
Tindakan Systems Manager 89
aws:ssm:send-command 89
aws:ssm:start-automation-execution 90
Tindakan dokumen SSM 91
Gunakan aws:ssm:send-command tindakan 91
Dokumen SSM AWS FIS yang telah dikonfigurasi sebelumnya 93
Contoh 102

Pemecahan Masalah 102
Tindakan tugas ECS 102
Tindakan 103
Batasan 103
Persyaratan 104
Versi referensi skrip 107
Contoh template percobaan 109
Tindakan EKS Pod 110
Tindakan 111
Batasan 112
Persyaratan 113
Buat peran eksperimen 113
Konfigurasikan akun layanan Kubernetes 113
Berikan akses kepada pengguna dan peran IAM ke Kubernetes APIs 115
Gambar kontainer pod 116
Contoh template percobaan 118
AWS Lambda tindakan 119
Tindakan 120
Batasan 120
Prasyarat 120
Konfigurasikan fungsi Lambda 122
Konfigurasikan AWS FIS eksperimen 122
Pencatatan log 122
Topik lanjutan 124
AWS FIS Versi ekstensi Lambda 130
Mengelola templat eksperimen 134
Buat template eksperimen 134
Lihat templat eksperimen 137
Hasilkan pratinjau target 138
Memulai percobaan dari template 138
Perbarui templat eksperimen 139
Tag template percobaan 140
Hapus templat eksperimen 140
Contoh template 141
Hentikan EC2 instance berdasarkan filter 141
Hentikan sejumlah EC2 instance tertentu 143

Jalankan dokumen SSM AWS FIS yang telah dikonfigurasi sebelumnya 144
Jalankan runbook Otomasi yang telah ditentukan sebelumnya 145
Tindakan Throttle API pada EC2 instance dengan peran IAM target 145
Uji stres CPU pod di klaster Kubernetes 147
Mengelola eksperimen 149
Memulai percobaan 149
Lihat eksperimen Anda 150
Percobaan menyatakan 150
Negara tindakan 151
Tandai eksperimen 151
Menghentikan sebuah percobaan 152
Daftar target yang diselesaikan 152
Tutorial 154
Contoh uji berhenti dan mulai 154
Prasyarat 154
Langkah 1: Buat template percobaan 154
Langkah 2: Mulai percobaan 158
Langkah 3: Lacak kemajuan eksperimen 158
Langkah 4: Verifikasi hasil percobaan 158
Langkah 5: Bersihkan 159
Jalankan stress CPU pada sebuah instance 160
Prasyarat 160
Langkah 1: Buat CloudWatch alarm untuk kondisi berhenti 161
Langkah 2: Buat template percobaan 161
Langkah 3: Mulai percobaan 164
Langkah 4: Lacak kemajuan eksperimen 164
Langkah 5: Verifikasi hasil percobaan 165
Langkah 6: Bersihkan 159
Uji interupsi Instans Spot 167
Prasyarat 167
Langkah 1: Buat template percobaan 169
Langkah 2: Mulai percobaan 171
Langkah 3: Lacak kemajuan eksperimen 172
Langkah 4: Verifikasi hasil percobaan 172
Langkah 5: Bersihkan 173
Simulasikan acara konektivitas 173

Prasyarat 174
Langkah 1: Buat template eksperimen AWS FIS 175
Langkah 2: Ping ke titik akhir Amazon S3 177
Langkah 3: Mulai eksperimen AWS FIS Anda 177
Langkah 4: Lacak kemajuan eksperimen AWS FIS Anda 178
Langkah 5: Verifikasi gangguan jaringan Amazon S3 178
Langkah 5: Bersihkan 179
Jadwalkan percobaan berulang 179
Prasyarat 180
Langkah 1: Buat peran dan kebijakan IAM 180
Langkah 2: Buat Amazon EventBridge Scheduler 182
Langkah 3: Verifikasi eksperimen Anda 183
Langkah 4: Membersihkan 183
Bekerja dengan pustaka skenario 184
Melihat skenario 184
Menggunakan skenario 185
Mengekspor skenario 186
Referensi skenario 186
AZ Availability: Power Interruption 189
Cross-Region: Connectivity 203
Bekerja dengan eksperimen multi-akun 216
Konsep 217
Praktik terbaik 217
Prasyarat 218
Izin 218
Kondisi berhenti (opsional) 221
Tuas pengaman untuk eksperimen multi-akun (opsional) 221
Buat templat eksperimen multi-akun 221
Perbarui konfigurasi akun target 223
Hapus konfigurasi akun target 223
Eksperimen penjadwalan 225
Buat peran penjadwal 225
Buat jadwal percobaan 229
Untuk memperbarui jadwal menggunakan konsol 230
Perbarui jadwal percobaan 230
Menonaktifkan atau menghapus jadwal percobaan 231

Tuas pengaman 232
Konsep untuk tuas pengaman 232
Sumber daya tuas Saftey 233
Bekerja dengan tuas pengaman 233
Melihat tuas pengaman 233
Melibatkan tuas pengaman 234
Melepaskan tuas pengaman 234
Eksperimen pemantauan 236
Monitor menggunakan CloudWatch 237
Pantau AWS eksperimen FIS 237
AWS Metrik penggunaan FIS 238
Monitor menggunakan EventBridge 239
Pencatatan percobaan 241
Izin 241
Skema log 241
Log tujuan 243
Contoh catatan log 243
Aktifkan pencatatan percobaan 248
Nonaktifkan pencatatan percobaan 249
Log panggilan API dengan AWS CloudTrail 249
Gunakan CloudTrail 249
Memahami AWS entri file log FIS 250
Pemecahan Masalah 255
Kode eror 255
Keamanan 258
Perlindungan data 258
Enkripsi diam 260
Enkripsi bergerak 260
Manajemen identitas dan akses 260
Audiens 260
Mengautentikasi dengan identitas 261
Mengelola akses menggunakan kebijakan 265
Bagaimana Layanan Injeksi AWS Kesalahan bekerja dengan IAM 267
Contoh kebijakan 274
Gunakan peran tertaut layanan 284
AWS kebijakan terkelola 287

Keamanan infrastruktur 292
AWS PrivateLink 292
Pertimbangan 293
Membuat titik akhir VPC antarmuka 293
Buat kebijakan titik akhir VPC 293
Pemberian tag pada sumber daya Anda 296
Pembatasan penandaan 296
Bekerja dengan tag 296
Kuota dan Batasan 298
Riwayat dokumen 315
..... cccxxi

Apa itu Layanan Injeksi AWS Kesalahan?

AWS AWS Fault Injection Service (FIS) adalah layanan terkelola yang memungkinkan Anda melakukan eksperimen injeksi kesalahan pada beban AWS kerja Anda. Injeksi kesalahan didasarkan pada prinsip-prinsip rekayasa kekacauan. Eksperimen ini menekankan aplikasi dengan membuat peristiwa yang mengganggu sehingga Anda dapat mengamati bagaimana aplikasi Anda merespons. Anda kemudian dapat menggunakan informasi ini untuk meningkatkan kinerja dan ketahanan aplikasi Anda sehingga mereka berperilaku seperti yang diharapkan.

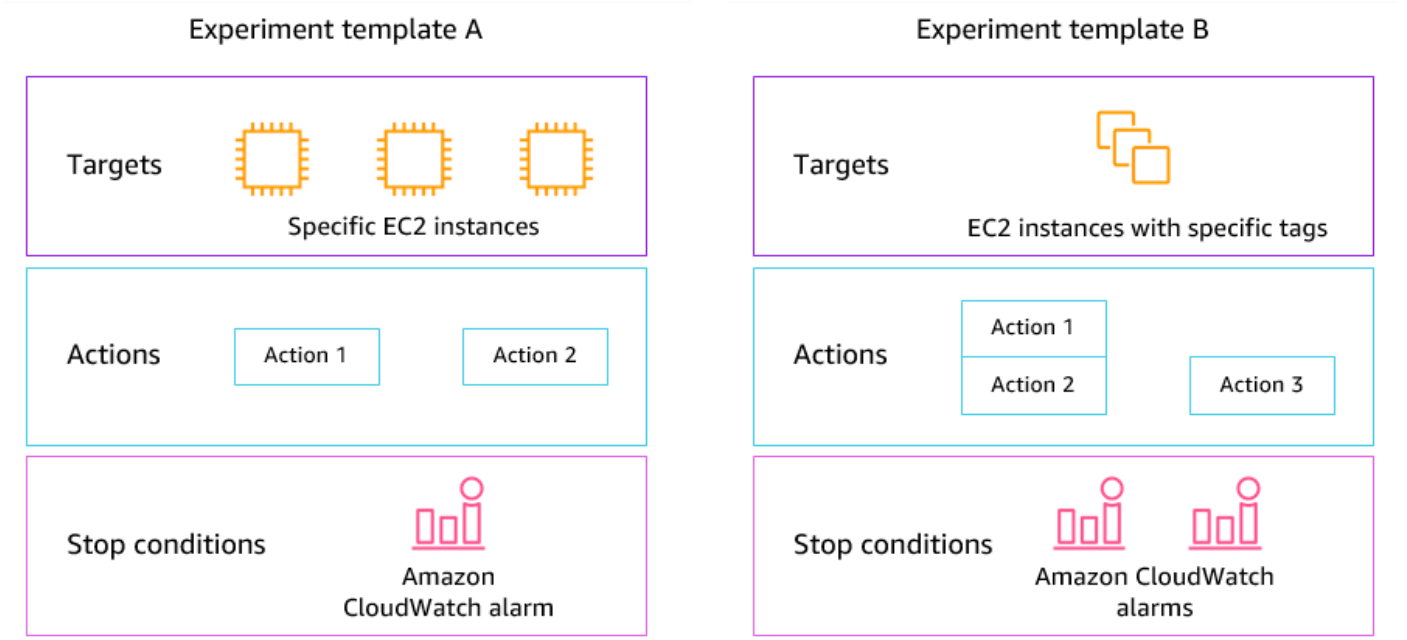
Untuk menggunakan AWS FIS, Anda mengatur dan menjalankan eksperimen yang membantu Anda menciptakan kondisi dunia nyata yang diperlukan untuk mengungkap masalah aplikasi yang mungkin sulit ditemukan sebaliknya. AWS FIS menyediakan templat yang menghasilkan gangguan, serta kontrol serta pagar pembatas yang Anda perlukan untuk menjalankan eksperimen dalam produksi, seperti memutar kembali atau menghentikan eksperimen secara otomatis jika kondisi tertentu terpenuhi.

> **Important**
>
> AWS FIS melakukan tindakan nyata pada AWS sumber daya nyata di sistem Anda. Oleh karena itu, sebelum Anda menggunakan AWS FIS untuk menjalankan eksperimen dalam produksi, kami sangat menyarankan Anda menyelesaikan fase perencanaan dan menjalankan eksperimen di lingkungan pra-produksi.

Untuk informasi selengkapnya tentang merencanakan eksperimen, lihat Uji Reliabilitas danMerencanakan eksperimen AWS FIS Anda. Untuk informasi lebih lanjut tentang AWS FIS, lihat Layanan Injeksi AWS Kesalahan.

AWS Konsep FIS

Untuk menggunakan AWS FIS, Anda menjalankan eksperimen pada AWS sumber daya Anda untuk menguji teori Anda tentang bagaimana kinerja aplikasi atau sistem dalam kondisi kesalahan. Untuk menjalankan eksperimen, pertama-tama Anda membuat template eksperimen. Template eksperimen adalah cetak biru eksperimen Anda. Ini berisi tindakan, target, dan kondisi berhenti untuk percobaan. Setelah Anda membuat template eksperimen, Anda dapat menggunakannya untuk menjalankan eksperimen. Saat eksperimen Anda berjalan, Anda dapat melacak kemajuannya dan melihat statusnya. Eksperimen selesai ketika semua tindakan dalam eksperimen telah berjalan.



Tindakan

Tindakan adalah aktivitas yang AWS dilakukan FIS pada AWS sumber daya selama percobaan. AWS FIS menyediakan serangkaian tindakan yang telah dikonfigurasi berdasarkan jenis sumber daya. AWS Setiap tindakan berjalan selama durasi tertentu selama percobaan, atau sampai Anda menghentikan eksperimen. Tindakan dapat berjalan secara berurutan atau bersamaan (secara paralel).

Target

Target adalah satu atau lebih AWS sumber daya di AWS mana FIS melakukan tindakan selama percobaan. Anda dapat memilih sumber daya tertentu, atau Anda dapat memilih sekelompok sumber daya berdasarkan kriteria tertentu, seperti tag atau status.

Hentikan kondisi

AWS FIS menyediakan kontrol dan pagar pembatas yang Anda butuhkan untuk menjalankan eksperimen dengan aman pada beban kerja Anda. AWS Kondisi berhenti adalah mekanisme untuk menghentikan eksperimen jika mencapai ambang batas yang Anda definisikan sebagai CloudWatch alarm Amazon. Jika kondisi berhenti dipicu saat percobaan sedang berjalan, AWS FIS menghentikan percobaan.

Didukung Layanan AWS

AWS FIS menyediakan tindakan yang telah dikonfigurasi untuk jenis target tertentu di seluruh AWS layanan. AWS FIS mendukung tindakan untuk sumber daya target untuk hal-hal berikut: Layanan AWS

- Amazon CloudWatch
- Amazon DynamoDB
- Amazon EBS
- Amazon EC2
- Amazon ECS
- Amazon EKS
- Amazon ElastiCache
- Amazon RDS
- Amazon S3
- AWS Systems Manager
- Amazon VPC

Untuk eksperimen akun tunggal, sumber daya target harus Akun AWS sama dengan eksperimen. Anda dapat menjalankan eksperimen AWS FIS yang menargetkan sumber daya di akun yang berbeda menggunakan eksperimen Akun AWS multi-akun AWS FIS.

Untuk informasi selengkapnya, lihat Tindakan untuk AWS FIS.

Akses AWS FIS

Anda dapat bekerja dengan AWS FIS dengan salah satu cara berikut:

- AWS Management Console— Menyediakan antarmuka web yang dapat Anda gunakan untuk mengakses AWS FIS. Untuk informasi lebih lanjut, lihat Bekerja dengan AWS Management Console.
- AWS Command Line Interface (AWS CLI) - Menyediakan perintah untuk serangkaian AWS layanan yang luas, termasuk AWS FIS, dan didukung pada Windows, macOS, dan Linux. Untuk informasi selengkapnya, lihat AWS Command Line Interface. Untuk informasi selengkapnya tentang perintah FIS, lihat AWS fis di AWS CLI Command Reference.

- AWS CloudFormation— Buat template yang menggambarkan AWS sumber daya Anda. Anda menggunakan templat untuk menyediakan dan mengelola sumber daya ini sebagai satu unit. Untuk informasi selengkapnya, lihat referensi jenis sumber daya Layanan Injeksi AWS Kesalahan.
- AWS SDKsMenyediakan bahasa khusus APIs dan menangani banyak detail koneksi, seperti menghitung tanda tangan, menangani percobaan ulang permintaan, dan menangani kesalahan. Untuk informasi selengkapnya, lihat AWS SDKs.
- HTTPS API - Menyediakan tindakan API tingkat rendah yang dapat Anda panggil menggunakan permintaan HTTPS. Untuk informasi selengkapnya, lihat Referensi API Layanan Injeksi AWS Kesalahan.

Harga untuk AWS FIS

Anda dikenakan biaya per menit bahwa suatu tindakan berjalan, dari awal hingga akhir, berdasarkan jumlah akun target untuk eksperimen Anda. Untuk informasi lebih lanjut, lihat Harga AWS FIS.

Merencanakan eksperimen AWS FIS Anda

Injeksi kesalahan adalah proses menekankan aplikasi dalam pengujian atau lingkungan produksi dengan menciptakan peristiwa yang mengganggu, seperti pemadaman server atau pelambatan API. Dari mengamati bagaimana sistem merespons, Anda kemudian dapat menerapkan perbaikan. Ketika Anda menjalankan eksperimen pada sistem Anda, ini dapat membantu Anda mengidentifikasi kelemahan sistemik secara terkontrol, sebelum kelemahan tersebut memengaruhi pelanggan yang bergantung pada sistem Anda. Kemudian Anda dapat secara proaktif mengatasi masalah untuk membantu mencegah hasil yang tidak terduga.

Sebelum Anda mulai menjalankan eksperimen injeksi kesalahan menggunakan AWS FIS, kami sarankan Anda membiasakan diri dengan prinsip dan pedoman berikut.

> **Important**
>
> AWS FIS melakukan tindakan nyata pada AWS sumber daya nyata di sistem Anda. Oleh karena itu, sebelum Anda mulai menggunakan AWS FIS untuk menjalankan eksperimen, kami sangat menyarankan Anda untuk menyelesaikan fase perencanaan dan pengujian terlebih dahulu di lingkungan pra-produksi atau pengujian.

Daftar Isi

- Prinsip dan pedoman dasar
- Pedoman perencanaan eksperimen

Prinsip dan pedoman dasar

Sebelum memulai eksperimen dengan AWS FIS, lakukan langkah-langkah berikut:

1. Identifikasi penyebaran target untuk eksperimen — Mulailah dengan mengidentifikasi penyebaran target. Jika ini adalah percobaan pertama Anda, kami sarankan untuk memulai di lingkungan pra-produksi atau pengujian.
2. Tinjau arsitektur aplikasi — Anda harus memastikan bahwa Anda telah mengidentifikasi semua komponen aplikasi, dependensi, dan prosedur pemulihan untuk setiap komponen. Mulailah dengan meninjau arsitektur aplikasi. Tergantung pada aplikasinya, lihat Kerangka AWS Well-Architected.

3. Tentukan perilaku steady-state — Tentukan perilaku kondisi tunak sistem Anda dalam hal metrik teknis dan bisnis yang penting, seperti latensi, beban CPU, login gagal per menit, jumlah percobaan ulang, atau kecepatan pemuatan halaman.
4. Bentuk hipotesis — Bentuk hipotesis tentang bagaimana Anda mengharapkan perilaku sistem berubah selama percobaan. Definisi hipotesis mengikuti format ini:

   Jika *`fault injection action`* dilakukan, tidak *`business or technical metric impact`* boleh melebihi*`value`*.

   Misalnya, hipotesis untuk layanan otentikasi mungkin berbunyi sebagai berikut: “Jika latensi jaringan meningkat sebesar 10%, ada kurang dari 1% peningkatan kegagalan masuk.” Setelah percobaan selesai, Anda mengevaluasi apakah ketahanan aplikasi sesuai dengan harapan bisnis dan teknis Anda.

Kami juga merekomendasikan mengikuti pedoman ini saat bekerja dengan AWS FIS:

- Selalu mulai bereksperimen dengan AWS FIS di lingkungan pengujian. Jangan pernah memulai dengan lingkungan produksi. Saat Anda maju dalam eksperimen injeksi kesalahan Anda, Anda dapat bereksperimen di lingkungan terkontrol lainnya di luar lingkungan pengujian.
- Bangun kepercayaan tim Anda dalam ketahanan aplikasi Anda dengan memulai dengan eksperimen kecil dan sederhana, seperti menjalankan tindakan aws:ec2:stop-instances pada satu target.
- Injeksi kesalahan dapat menyebabkan masalah nyata. Lanjutkan dengan hati-hati dan pastikan bahwa suntikan kesalahan pertama Anda ada pada contoh pengujian sehingga tidak ada pelanggan yang terpengaruh.
- Uji, uji, dan uji lagi. Injeksi kesalahan dimaksudkan untuk diimplementasikan dalam lingkungan yang terkendali dengan eksperimen yang terencana dengan baik. Ini memungkinkan Anda untuk membangun kepercayaan pada kemampuan aplikasi Anda dan alat Anda untuk menahan kondisi yang bergejolak.
- Kami sangat menyarankan agar Anda memiliki program pemantauan dan peringatan yang sangat baik sebelum Anda mulai. Tanpa itu, Anda tidak akan dapat memahami atau mengukur dampak eksperimen Anda, yang sangat penting untuk praktik injeksi kesalahan yang berkelanjutan.

Pedoman perencanaan eksperimen

Dengan AWS FIS, Anda menjalankan eksperimen pada AWS sumber daya Anda untuk menguji teori Anda tentang bagaimana aplikasi atau sistem akan tampil dalam kondisi kesalahan.

Berikut ini adalah panduan yang direkomendasikan untuk merencanakan eksperimen AWS FIS Anda.

- Tinjau riwayat pemadaman - Tinjau pemadaman dan peristiwa sebelumnya untuk sistem Anda. Ini dapat membantu Anda membangun gambaran kesehatan dan ketahanan keseluruhan sistem Anda. Sebelum Anda mulai menjalankan eksperimen pada sistem Anda, Anda harus mengatasi masalah dan kelemahan yang diketahui dalam sistem Anda.
- Identifikasi layanan dengan dampak terbesar — Tinjau layanan Anda dan identifikasi layanan yang memiliki dampak terbesar pada pengguna akhir atau pelanggan Anda jika layanan tersebut turun atau tidak berfungsi dengan benar.
- Identifikasi sistem target — Sistem target adalah sistem di mana Anda akan menjalankan eksperimen. Jika Anda baru mengenal AWS FIS atau belum pernah menjalankan eksperimen injeksi kesalahan sebelumnya, kami sarankan Anda memulai dengan menjalankan eksperimen pada sistem pra-produksi atau pengujian.
- Konsultasikan dengan tim Anda — Tanyakan apa yang mereka khawatirkan. Anda dapat membentuk hipotesis untuk membuktikan atau menyangkal kekhawatiran mereka. Anda juga dapat bertanya kepada tim Anda apa yang tidak mereka khawatirkan. Pertanyaan ini dapat mengungkapkan dua kesalahan umum: kekeliruan biaya tenggelam dan kekeliruan bias konfirmasi. Membentuk hipotesis berdasarkan jawaban tim Anda dapat membantu memberikan lebih banyak informasi tentang realitas keadaan sistem Anda.
- Tinjau arsitektur aplikasi Anda — Lakukan peninjauan sistem atau aplikasi Anda dan pastikan bahwa Anda telah mengidentifikasi semua komponen aplikasi, dependensi, dan prosedur pemulihan untuk setiap komponen.

  Kami menyarankan Anda meninjau Kerangka AWS Well-Architected. Kerangka kerja ini dapat membantu Anda membangun infrastruktur yang aman, berkinerja tinggi, tangguh, dan efisien untuk aplikasi dan beban kerja Anda. Untuk informasi lebih lanjut, lihat AWS Well-Architected.
- Identifikasi metrik yang berlaku — Anda dapat memantau dampak eksperimen pada AWS sumber daya menggunakan CloudWatch metrik Amazon. Anda dapat menggunakan metrik ini untuk menentukan baseline atau “steady state” saat aplikasi Anda berkinerja optimal. Kemudian, Anda dapat memantau metrik ini selama atau setelah percobaan untuk menentukan dampaknya. Untuk informasi selengkapnya, lihat Pantau metrik penggunaan AWS FIS menggunakan Amazon CloudWatch.

- Tentukan ambang kinerja yang dapat diterima untuk sistem Anda — Identifikasi metrik yang mewakili kondisi stabil yang dapat diterima untuk sistem Anda. Anda akan menggunakan metrik ini untuk membuat satu atau beberapa CloudWatch alarm yang mewakili kondisi berhenti untuk eksperimen Anda. Jika alarm dipicu, percobaan dihentikan secara otomatis. Untuk informasi selengkapnya, lihat Kondisi berhenti untuk AWS FIS.

AWS Komponen template eksperimen FIS

Anda menggunakan komponen berikut untuk membuat template eksperimen:

Tindakan

Tindakan AWS FIS yang ingin Anda jalankan. Tindakan dapat dijalankan dalam urutan yang ditetapkan yang Anda tentukan, atau mereka dapat dijalankan secara bersamaan. Untuk informasi selengkapnya, lihat Tindakan.

Target

Sumber AWS daya di mana tindakan tertentu dilakukan. Untuk informasi selengkapnya, lihat Target.

Hentikan kondisi

CloudWatch Alarm yang menentukan ambang batas di mana kinerja aplikasi Anda tidak dapat diterima. Jika kondisi berhenti dipicu saat percobaan sedang berjalan, AWS FIS menghentikan percobaan. Untuk informasi selengkapnya, lihat Hentikan kondisi.

Peran percobaan

Peran IAM yang memberikan AWS FIS izin yang diperlukan sehingga dapat menjalankan eksperimen atas nama Anda. Untuk informasi selengkapnya, lihat Peran percobaan.

Konfigurasi laporan eksperimen

Konfigurasi untuk mengaktifkan laporan eksperimen. Untuk informasi selengkapnya, lihat Konfigurasi laporan eksperimen untuk AWS FIS.

Opsi percobaan

Opsi untuk templat percobaan. Untuk informasi selengkapnya, lihat Opsi percobaan untuk AWS FIS.

Akun Anda memiliki kuota yang terkait dengan AWS FIS. Misalnya, ada kuota jumlah tindakan per templat percobaan. Untuk informasi selengkapnya, lihat Kuota dan Batasan.

Sintaks template

Berikut ini adalah sintaks untuk template percobaan.

```
{
            "description": "string",
            "targets": {},
            "actions": {},
            "stopConditions": [],
            "roleArn": "arn:aws:iam::123456789012:role/AllowFISActions",
            "experimentReportConfiguration":{},
            "experimentOptions":{},
            "tags": {}
        }
```

Sebagai contoh, lihat Contoh template.

Memulai

Untuk membuat template eksperimen menggunakan AWS Management Console, lihatBuat template eksperimen.

Untuk membuat template eksperimen menggunakan AWS CLI, lihatContoh AWS templat eksperimen FIS.

Tindakan untuk AWS FIS

Untuk membuat template eksperimen, Anda harus menentukan satu atau beberapa tindakan. Untuk daftar tindakan yang telah ditentukan sebelumnya yang disediakan oleh AWS FIS, lihat. Referensi tindakan

Anda dapat menjalankan tindakan hanya sekali selama percobaan. Untuk menjalankan tindakan AWS FIS yang sama lebih dari sekali dalam percobaan yang sama, tambahkan ke template beberapa kali menggunakan nama yang berbeda.

Daftar Isi

- Sintaks tindakan
- Pengidentifikasi tindakan
- Parameter tindakan
- Target aksi
- Durasi tindakan
- Contoh tindakan

Sintaks tindakan

Berikut ini adalah sintaks untuk tindakan.

```
{
    "actions": {
        "action_name": {
            "actionId": "aws:service:action-type",
            "description": "string",
            "parameters": {
                "name": "value"
             },
            "startAfter": ["action_name", ...],
            "targets": {
                "ResourceType": "target_name"
            }
        }
    }
}
```

Saat Anda mendefinisikan suatu tindakan, Anda memberikan yang berikut:

action_name

Sebuah nama untuk tindakan.

actionId

Pengidentifikasi tindakan.

description

Deskripsi opsional.

parameters

Parameter tindakan apa pun.

startAfter

Tindakan apa pun yang harus diselesaikan sebelum tindakan ini dapat dimulai. Jika tidak, tindakan berjalan pada awal percobaan.

targets

Target tindakan apa pun.

Sebagai contoh, lihat the section called “Contoh tindakan”.

Pengidentifikasi tindakan

Setiap tindakan AWS FIS memiliki pengenal dengan format berikut:

```
aws:service-name:action-type
```

Misalnya, tindakan berikut menghentikan EC2 instans Amazon target:

```
aws:ec2:stop-instances
```

Untuk daftar lengkap tindakan, lihatAWS FIS Referensi tindakan.

Parameter tindakan

Beberapa tindakan AWS FIS memiliki parameter tambahan yang spesifik untuk tindakan tersebut. Parameter ini digunakan untuk meneruskan informasi ke AWS FIS ketika tindakan dijalankan.

AWS FIS mendukung jenis kesalahan khusus menggunakan `aws:ssm:send-command` tindakan, yang menggunakan Agen SSM dan dokumen perintah SSM untuk membuat kondisi kesalahan pada instance yang ditargetkan. `aws:ssm:send-command`Tindakan tersebut mencakup `documentArn` parameter yang mengambil Amazon Resource Name (ARN) dari dokumen SSM sebagai nilai. Anda menentukan nilai untuk parameter saat menambahkan tindakan ke templat eksperimen.

Untuk informasi selengkapnya tentang menentukan parameter untuk `aws:ssm:send-command` tindakan, lihatGunakan aws:ssm:send-command tindakan.

Jika memungkinkan, Anda dapat memasukkan konfigurasi rollback (juga disebut sebagai tindakan posting) dalam parameter tindakan. Post action mengembalikan target ke keadaan sebelum tindakan dijalankan. Tindakan posting berjalan setelah waktu yang ditentukan dalam durasi tindakan. Tidak semua tindakan dapat mendukung tindakan posting. Misalnya, jika tindakan menghentikan EC2 instance Amazon, Anda tidak dapat memulihkan instance setelah dihentikan.

Target aksi

Tindakan berjalan pada sumber daya target yang Anda tentukan. Setelah Anda menentukan target, Anda dapat menentukan namanya ketika Anda menentukan tindakan.

```
"targets": {
```

```
    "ResourceType": "resource_name"
}
```

AWS Tindakan FIS mendukung jenis sumber daya berikut untuk target tindakan:

- AutoScalingGroups— Grup EC2 Auto Scaling Amazon
- Ember - Ember Amazon S3
- Cluster — Cluster Amazon EKS
- Cluster - Cluster Amazon ECS atau cluster Amazon Aurora DB
- DBInstances- Instans Amazon RDS DB
- Contoh - Instans Amazon EC2
- ManagedResources- Cluster Amazon EKS, EC2 Aplikasi Amazon dan Penyeimbang Beban Jaringan, dan grup Amazon EC2 Auto Scaling yang diaktifkan untuk pergeseran zona ARC.
- Nodegroups - Grup simpul Amazon EKS
- Pod — Pod Kubernetes di Amazon EKS
- ReplicationGroups— Grup ElastiCache Replikasi
- Peran - peran IAM
- SpotInstances— Instans EC2 Spot Amazon
- Subnet - Subnet VPC
- Tabel - Amazon DynamoDB tabel global
- Tugas - Tugas Amazon ECS
- TransitGateways— Gerbang transit
- Volume - Volume Amazon EBS

Sebagai contoh, lihat the section called “Contoh tindakan”.

Durasi tindakan

Jika suatu tindakan menyertakan parameter yang dapat Anda gunakan untuk menentukan durasi tindakan, secara default, tindakan dianggap selesai hanya setelah durasi yang ditentukan telah berlalu. Jika Anda telah menetapkan opsi `emptyTargetResolutionMode` eksperimen`skip`, maka tindakan akan segera selesai dengan status 'dilewati' ketika tidak ada target yang diselesaikan. Misalnya, jika Anda menentukan durasi 5 menit, AWS FIS menganggap tindakan selesai setelah 5 menit. Kemudian memulai tindakan berikutnya, sampai semua tindakan selesai.

Durasi dapat berupa lamanya waktu kondisi tindakan dipertahankan atau lamanya waktu metrik dipantau. Misalnya, latensi disuntikkan selama durasi waktu yang ditentukan. Untuk jenis tindakan yang hampir seketika, seperti menghentikan instance, kondisi berhenti dipantau selama durasi waktu yang ditentukan.

Jika tindakan menyertakan tindakan posting dalam parameter tindakan, tindakan posting berjalan setelah tindakan selesai. Waktu yang diperlukan untuk menyelesaikan tindakan pasca dapat menyebabkan penundaan antara durasi tindakan yang ditentukan dan awal tindakan berikutnya (atau akhir percobaan, jika semua tindakan lain selesai).

Contoh tindakan

Berikut ini adalah contoh tindakan.

Contoh

- Hentikan EC2 contoh
- Interupsi Instans Spot
- Mengganggu lalu lintas jaringan
- Menghentikan pekerja EKS
- Mulai pergeseran otomatis zona ARC

Contoh: Stop EC2 instance

Tindakan berikut menghentikan EC2 instance yang diidentifikasi menggunakan target bernama*targetInstances*. Setelah dua menit, itu memulai ulang instance target.

```
"actions": {
    "stopInstances": {
        "actionId": "aws:ec2:stop-instances",
        "parameters": {
            "startInstancesAfterDuration": "PT2M"
        },
        "targets": {
            "Instances": "targetInstances"
        }
    }
}
```

Contoh: Interupsi Instance Spot

Tindakan berikut menghentikan Instans Spot yang diidentifikasi menggunakan target bernama*targetSpotInstances*. Itu menunggu dua menit sebelum mengganggu Instance Spot.

```
"actions": {
    "interruptSpotInstances": {
        "actionId": "aws:ec2:send-spot-instance-interruptions",
        "parameters": {
            "durationBeforeInterruption": "PT2M"
        },
        "targets": {
            "SpotInstances": "targetSpotInstances"
        }
    }
}
```

Contoh: Mengganggu lalu lintas jaringan

Tindakan berikut menyangkal lalu lintas antara subnet target dan subnet di Availability Zone lainnya.

```
"actions": {
    "disruptAZConnectivity": {
        "actionId": "aws:network:disrupt-connectivity",
        "parameters": {
            "scope": "availability-zone",
            "duration": "PT5M"
        },
        "targets": {
            "Subnets": "targetSubnets"
        }
    }
}
```

Contoh: Menghentikan pekerja EKS

Tindakan berikut mengakhiri 50% EC2 instance di cluster EKS yang diidentifikasi menggunakan target bernama. *targetNodeGroups*

```
"actions": {
    "terminateWorkers": {
```

```
        "actionId": "aws:eks:terminate-nodegroup-instances",
        "parameters": {
            "instanceTerminationPercentage": "50"
        },
        "targets": {
            "Nodegroups": "targetNodeGroups"
        }
    }
}
```

Contoh: Mulai pergeseran otomatis zona ARC

Tindakan berikut memulai autoshift ARC Zonal yang menggeser sumber daya terkelola dari for. *az-in-parameters duration-in-parameteres* Jenis sumber daya `ManagedResources` digunakan sebagai kunci untuk nama target dalam templat eksperimen AWS FIS.

```
{
    "description": "aaa",
    "targets": {
        "ManagedResources-Target-1": {
            "resourceType": "aws:arc:zonal-shift-managed-resource",
            "resourceArns": [
                "arn:aws:elasticloadbalancing:us-east-1:0124567890:loadbalancer/app/
application/11223312312516",
            ],
            "selectionMode": "ALL"
        }
    },
    "actions": {
        "arc": {
            "actionId": "aws:arc:start-zonal-autoshift",
            "parameters": {
                "availabilityZoneIdentifier": "us-east-1a",
                "duration": "PT1M"
            },
            "targets": {
               "ManagedResources": "ManagedResources-Target-1"
            }
        }
    },
    "stopConditions": [
        {
            "source": "none"
```

```
        }
    ],
    "roleArn": "arn:aws:iam::718579638765:role/fis",
    "tags": {},
    "experimentOptions": {
        "accountTargeting": "single-account",
        "emptyTargetResolutionMode": "fail"
    }
}
```

Target untuk AWS FIS

Target adalah satu atau lebih AWS sumber daya di mana tindakan dilakukan oleh AWS AWS Fault Injection Service (FIS) selama percobaan. Target dapat berada di akun AWS yang sama dengan eksperimen, atau di akun lain menggunakan eksperimen multi-akun. Untuk mempelajari lebih lanjut tentang penargetan sumber daya di akun lain, lihatBekerja dengan eksperimen multi-akun.

Anda menentukan target saat membuat templat eksperimen. Anda dapat menggunakan target yang sama untuk beberapa tindakan dalam template eksperimen Anda.

AWS FIS mengidentifikasi semua target pada awal percobaan, sebelum memulai salah satu tindakan dalam tindakan yang ditetapkan. AWS FIS menggunakan sumber daya target yang dipilihnya untuk seluruh percobaan. Jika tidak ada target yang ditemukan, percobaan gagal.

Daftar Isi

- Sintaks target
- Jenis sumber daya
- Identifikasi sumber daya target
  - Filter sumber daya
  - Parameter sumber daya
- Mode pemilihan
- Contoh target
- Contoh filter

Sintaks target

Berikut ini adalah sintaks untuk target.

```
{
    "targets": {
        "target_name": {
            "resourceType": "resource-type",
            "resourceArns": [
                "resource-arn"
            ],
            "resourceTags": {
                "tag-key": "tag-value"
            },
            "parameters": {
                "parameter-name": "parameter-value"
            },
            "filters": [
                {
                    "path": "path-string",
                    "values": ["value-string"]
                }
            ],
            "selectionMode": "value"
        }
    }
}
```

Saat Anda menentukan target, Anda memberikan yang berikut:

target_name

Sebuah nama untuk target.

resourceType

Jenis sumber daya.

resourceArns

Nama Sumber Daya Amazon (ARN) dari sumber daya tertentu.

resourceTags

Tag diterapkan pada sumber daya tertentu.

parameters

Parameter yang mengidentifikasi target menggunakan atribut tertentu.

filters

Filter sumber daya mencakup sumber daya target yang diidentifikasi menggunakan atribut tertentu.

selectionMode

Mode pemilihan untuk sumber daya yang diidentifikasi.

Sebagai contoh, lihat the section called “Contoh target”.

Jenis sumber daya

Setiap tindakan AWS FIS dilakukan pada jenis AWS sumber daya tertentu. Ketika Anda menentukan target, Anda harus menentukan dengan tepat satu jenis sumber daya. Saat Anda menentukan target untuk suatu tindakan, target harus berupa jenis sumber daya yang didukung oleh tindakan tersebut.

Jenis sumber daya berikut didukung oleh AWS FIS:

- aws:arc: zonal-shift-managed-resource — AWS Sumber daya yang terdaftar dengan ARC zonal shift
- aws:dynamodb: global-table - Sebuah tabel global Amazon DynamoDB
- aws:ec2: autoscaling-group - Grup Auto Scaling Amazon EC2
- aws:ec2:ebs-volume - Volume Amazon EBS
- aws:ec2:instance - Sebuah contoh Amazon EC2
- aws:ec2:spot-instance - Instans Spot Amazon EC2
- aws:ec2:subnet - Subnet Amazon VPC
- aws:ec2:transit-gateway — Sebuah gerbang transit
- aws:ecs:cluster - Cluster Amazon ECS
- aws:ecs:task - Tugas Amazon ECS
- aws:eks:cluster - Kluster Amazon EKS
- aws:eks:nodegroup - Grup simpul Amazon EKS
- aws:eks:pod — Sebuah pod Kubernetes
- aws:elasticache:replicationgroup - Grup Replikasi ElastiCache
- aws:iam:role — Peran IAM

- aws:lambda: fungsi - Sebuah fungsi AWS Lambda
- aws:rds:cluster - Cluster Amazon Aurora DB
- aws:rds: db - Sebuah instans Amazon RDS DB
- aws:s3: ember - Ember Amazon S3

Identifikasi sumber daya target

Saat Anda menentukan target di konsol AWS FIS, Anda dapat memilih AWS sumber daya tertentu (dari jenis sumber daya tertentu) untuk ditargetkan. Atau, Anda dapat membiarkan AWS FIS mengidentifikasi sekelompok sumber daya berdasarkan kriteria yang Anda berikan.

Untuk mengidentifikasi sumber daya target Anda, Anda dapat menentukan yang berikut:

- Sumber daya IDs — Sumber daya IDs sumber AWS daya tertentu. Semua sumber daya IDs harus mewakili jenis sumber daya yang sama.
- Tag sumber daya — Tag yang diterapkan pada AWS sumber daya tertentu.
- Filter sumber daya — Jalur dan nilai yang mewakili sumber daya dengan atribut tertentu. Untuk informasi selengkapnya, lihat Filter sumber daya.
- Parameter sumber daya — Parameter yang mewakili sumber daya yang memenuhi kriteria tertentu. Untuk informasi selengkapnya, lihat Parameter sumber daya.

Pertimbangan

- Anda tidak dapat menentukan ID sumber daya dan tag sumber daya untuk target yang sama.
- Anda tidak dapat menentukan ID sumber daya dan filter sumber daya untuk target yang sama.
- Jika Anda menentukan tag sumber daya dengan nilai tag kosong, itu tidak setara dengan wildcard. Ini cocok dengan sumber daya yang memiliki tag dengan kunci tag yang ditentukan dan nilai tag kosong.
- Jika Anda menentukan lebih dari satu tag, semua tag yang ditentukan harus ada pada sumber daya target agar dapat dipilih (`AND`).

Filter sumber daya

Filter sumber daya adalah kueri yang mengidentifikasi sumber daya target sesuai dengan atribut tertentu. AWS FIS menerapkan kueri ke output tindakan API yang berisi deskripsi kanonik AWS

sumber daya, sesuai dengan jenis sumber daya yang Anda tentukan. Sumber daya yang memiliki atribut yang cocok dengan kueri disertakan dalam definisi target.

Setiap filter dinyatakan sebagai jalur atribut dan nilai yang mungkin. Path adalah urutan elemen, dipisahkan oleh periode, yang menggambarkan jalur untuk mencapai atribut dalam output dari tindakan Deskripsikan untuk sumber daya. Setiap periode berarti perluasan elemen. Setiap elemen harus dinyatakan dalam kasus Pascal, bahkan jika output dari tindakan Deskripsikan untuk sumber daya dalam kasus unta. Misalnya, Anda harus menggunakan`AvailabilityZone`, bukan `availablityZone` sebagai elemen atribut.

```
"filters": [
    {
        "path": "Component.Component.Component",
        "values": [
            "string"
        ]
    }
],
```

Logika berikut berlaku untuk semua filter sumber daya:

- Jika beberapa filter disediakan, termasuk filter dengan jalur yang sama, semua filter harus dicocokkan untuk sumber daya yang akan dipilih — `AND`
- Jika beberapa nilai disediakan untuk satu filter, salah satu nilai harus dicocokkan untuk sumber daya yang akan dipilih — `OR`
- Jika beberapa nilai ditemukan di lokasi jalur panggilan API deskripsikan, salah satu nilai harus dicocokkan agar sumber daya dipilih — `OR`
- Untuk mencocokkan pada pasangan kunci tag/nilai, Anda harus memilih sumber daya target dengan tag sebagai gantinya (lihat di atas).

Tabel berikut mencakup tindakan dan AWS CLI perintah API yang dapat Anda gunakan untuk mendapatkan deskripsi kanonik untuk setiap jenis sumber daya. AWS FIS menjalankan tindakan ini atas nama Anda untuk menerapkan filter yang Anda tentukan. Dokumentasi yang sesuai menjelaskan sumber daya yang disertakan dalam hasil secara default. Misalnya, dokumentasi untuk DescribeInstances status bahwa instance yang baru saja dihentikan mungkin muncul di hasil.

Jenis sumber daya	Tindakan API	AWS CLI perintah
aws:arc:zonal-shift-managed-resource	ListManagedResources	list-managed-resources
aws:ec2:autoscaling-group	DescribeAutoScalingGroups	describe-auto-scaling-groups
aws:ec2:ebs-volume	DescribeVolumes	jelaskan-volume
aws:ec2:instance	DescribeInstances	mendeskripsikan-contoh
aws:ec2:subnet	DescribeSubnets	jelaskan-subnet
aws:ec2:transit-gateway	DescribeTransitGateways	describe-transit-gateways
aws:ecs:cluster	DescribeClusters	jelaskan-cluster
aws:ecs:task	DescribeTasks	jelaskan tugas-tugas
aws:eks:cluster	DescribeClusters	jelaskan-cluster
aws:eks:nodegroup	DescribeNodegroup	jelaskan-nodegroup
aws:elasticache:replication group	DescribeReplicationGroups	describe-replication-groups
aws:iam:role	ListRoles	daftar-peran
aws:lambda:function	ListFunctions	daftar-fungsi
aws:rds:cluster	Jelaskan DBClusters	describe-db-clusters
aws:rds:db	Jelaskan DBInstances	describe-db-instances
aws:s3:bucket	ListBuckets	daftar-ember
aws:dynamodb:global-table	DescribeTable	deskripsikan-tabel

Sebagai contoh, lihat the section called “Contoh filter”.

Parameter sumber daya

Parameter sumber daya mengidentifikasi sumber daya target sesuai dengan kriteria tertentu.

Jenis sumber daya berikut mendukung parameter.

aws:ec2:ebs-volume

- `availabilityZoneIdentifier`— Kode (misalnya, us-east-1a) dari Availability Zone yang berisi volume target.

aws:ec2:subnet

- `availabilityZoneIdentifier`— Kode (misalnya, us-east-1a) atau ID AZ (misalnya, use1-az1) dari Availability Zone yang berisi subnet target.
- `vpc`— VPC yang berisi subnet target. Tidak mendukung lebih dari satu VPC per akun.

aws:ecs:task

- `cluster`— Cluster yang berisi tugas target.
- `service`— Layanan yang berisi tugas target.

aws:eks:pod

- `availabilityZoneIdentifier` – Opsional. Availability Zone yang berisi pod target. Misalnya, `us-east-1d`. Kami menentukan Availability Zone dari sebuah pod dengan membandingkan HostIP dan CIDR dari subnet cluster.
- `clusterIdentifier` – Wajib. Nama atau ARN dari kluster EKS target.
- `namespace` – Wajib. Namespace Kubernetes dari pod target.
- `selectorType` – Wajib. Jenis pemilih. Nilai yang mungkin adalah `labelSelector`, `deploymentName`, dan `podName`.
- `selectorValue` – Wajib. Nilai pemilih. Nilai ini tergantung pada nilai`selectorType`.
- `targetContainerName` – Opsional. Nama kontainer target sebagaimana didefinisikan dalam spesifikasi pod. Defaultnya adalah kontainer pertama yang ditentukan dalam setiap spesifikasi pod target.

aws:lambda:function

- `functionQualifier` – Opsional. Versi atau alias fungsi yang ditargetkan. Jika tidak ada kualifikasi yang ditentukan, semua pemanggilan akan dipertimbangkan untuk penargetan. Jika alias dengan beberapa versi ditentukan, semua versi yang disertakan dalam alias akan dipertimbangkan untuk penargetan selama mereka dipanggil menggunakan ARN yang

berisi alias. Jika alias khusus `$LATEST` digunakan, pemanggilan ke fungsi dasar ARN dan pemanggilan termasuk `$LATEST` dalam ARN akan dipertimbangkan untuk injeksi kesalahan. Untuk informasi selengkapnya tentang versi Lambda, lihat Mengelola versi fungsi Lambda di panduan pengguna.AWS Lambda

aws:rds:cluster

- `writerAvailabilityZoneIdentifiers` – Opsional. Availability Zones dari penulis cluster DB. Nilai yang mungkin adalah: daftar pengidentifikasi Availability Zone yang dipisahkan koma,. `all`

aws:rds:db

- `availabilityZoneIdentifiers` – Opsional. Availability Zones dari instans DB akan terpengaruh. Nilai yang mungkin adalah: daftar pengidentifikasi Availability Zone yang dipisahkan koma,. `all`

aws:elasticache:replicationgroup

- `availabilityZoneIdentifier` – Wajib. Kode (misalnya, us-east-1a) atau ID AZ (misalnya, use1-az1) dari Availability Zone yang berisi node target.

Mode pemilihan

Anda mencakup sumber daya yang diidentifikasi dengan menentukan mode pemilihan. AWS FIS mendukung mode pemilihan berikut:

- `ALL`— Jalankan aksi pada semua target.
- `COUNT(n)`— Jalankan tindakan pada jumlah target yang ditentukan, dipilih dari target yang diidentifikasi secara acak. Misalnya, COUNT (1) memilih salah satu target yang diidentifikasi.
- `PERCENT(n)`— Jalankan tindakan pada persentase target yang ditentukan, dipilih dari target yang diidentifikasi secara acak. Misalnya, PERCENT (25) memilih 25% dari target yang diidentifikasi.

Jika Anda memiliki jumlah sumber daya ganjil dan menentukan 50%, AWS FIS akan membulatkan ke bawah. Misalnya, jika Anda menambahkan lima EC2 instans Amazon sebagai target dan cakupan hingga 50%, AWS FIS akan membulatkan ke dua instans. Anda tidak dapat menentukan persentase yang kurang dari satu sumber daya. Misalnya, jika Anda menambahkan empat EC2 instans dan cakupan Amazon menjadi 5%, AWS FIS tidak dapat memilih instans.

Jika Anda menentukan beberapa target menggunakan jenis sumber daya target yang sama, AWS FIS dapat memilih sumber daya yang sama beberapa kali.

Terlepas dari mode pemilihan yang Anda gunakan, jika cakupan yang Anda tentukan tidak mengidentifikasi sumber daya, eksperimen gagal.

Contoh target

Berikut ini adalah contoh target.

Contoh

- Contoh di VPC yang ditentukan dengan tag yang ditentukan
- Tugas dengan parameter yang ditentukan

Contoh: Contoh di VPC yang ditentukan dengan tag yang ditentukan

Target yang mungkin untuk contoh ini adalah EC2 instance Amazon di VPC yang ditentukan dengan tag env=prod. Mode pemilihan menentukan bahwa AWS FIS memilih salah satu target ini secara acak.

```
{
    "targets": {
        "randomInstance": {
            "resourceType": "aws:ec2:instance",
            "resourceTags": {
                "env": "prod"
            },
            "filters": [
                {
                    "path": "VpcId",
                    "values": [
                        "vpc-aabbcc11223344556"
                    ]
                }
            ],
            "selectionMode": "COUNT(1)"
        }
    }
}
```

Contoh: Tugas dengan parameter yang ditentukan

Target yang mungkin untuk contoh ini adalah tugas Amazon ECS dengan cluster dan layanan yang ditentukan. Mode pemilihan menentukan bahwa AWS FIS memilih salah satu target ini secara acak.

```
{
    "targets": {
        "randomTask": {
            "resourceType": "aws:ecs:task",
            "parameters": {
                "cluster": "myCluster",
                "service": "myService"
            },
            "selectionMode": "COUNT(1)"
        }
    }
}
```

Contoh filter

Berikut ini adalah contoh filter.

Contoh

- EC2 contoh
- Cluster DB

Contoh: EC2 contoh

Saat Anda menentukan filter untuk tindakan yang mendukung tipe sumber daya aws:ec2:instance, AWS FIS menggunakan perintah EC2 describe-instances Amazon dan menerapkan filter untuk mengidentifikasi target.

describe-instancesPerintah mengembalikan output JSON di mana setiap instance adalah struktur di bawah`Instances`. Berikut ini adalah output sebagian yang mencakup bidang yang ditandai dengan*`italics`*. Kami akan memberikan contoh yang menggunakan bidang ini untuk menentukan jalur atribut dari struktur output JSON.

```
{
    "Reservations": [
        {
            "Groups": [],
```

```
            "Instances": [
                {
                    "ImageId": "ami-00111111111111111",
                    "InstanceId": "i-00aaaaaaaaaaaaaa",
                    "InstanceType": "t2.micro",
                    "KeyName": "virginia-kp",
                    "LaunchTime": "2020-09-30T11:38:17.000Z",
                    "Monitoring": {
                        "State": "disabled"
                    },
                    "Placement": {
                        "AvailabilityZone": "us-east-1a",
                        "GroupName": "",
                        "Tenancy": "default"
                    },
                    "PrivateDnsName": "ip-10-0-1-240.ec2.internal",
                    "PrivateIpAddress": "10.0.1.240",
                    "ProductCodes": [],
                    "PublicDnsName": "ec2-203-0-113-17.compute-1.amazonaws.com",
                    "PublicIpAddress": "203.0.113.17",
                    "State": {
                        "Code": 16,
                        "Name": "running"
                    },
                    "StateTransitionReason": "",
                    "SubnetId": "subnet-aabbcc11223344556",
                    "VpcId": "vpc-00bbbbbbbbbbbbbbbb",
                    ...
                    "NetworkInterfaces": [
                    {
                        ...
                        "Groups": [
                            {
                                "GroupName": "sec-group-1",
                                "GroupId": "sg-a0011223344556677"
                            },
                            {
                                "GroupName": "sec-group-1",
                                "GroupId": "sg-b9988776655443322"
                            }
                        ],
                        ...
                    },
                    ...
```

```
                },
                ...
                {
                    ...
                }
            ],
            "OwnerId": "123456789012",
            "ReservationId": "r-aaaaaabbbbb111111"
        },
        ...
    ]
}
```

Untuk memilih instance di Availability Zone tertentu menggunakan filter sumber daya, tentukan jalur atribut untuk `AvailabilityZone` dan kode untuk Availability Zone sebagai nilainya. Misalnya:

```
"filters": [
    {
        "path": "Placement.AvailabilityZone",
        "values": [ "us-east-1a" ]
    }
],
```

Untuk memilih instance di subnet tertentu menggunakan filter sumber daya, tentukan jalur atribut untuk `SubnetId` dan ID subnet sebagai nilainya. Misalnya:

```
"filters": [
    {
        "path": "SubnetId",
        "values": [ "subnet-aabbcc11223344556" ]
    }
],
```

Untuk memilih instance yang berada dalam status instance tertentu, tentukan jalur atribut untuk `Name` dan salah satu nama status berikut sebagai nilai: `pending` | | `running` | `shutting-down` | `terminated` `stopping` |`stopped`. Misalnya:

```
"filters": [
    {
        "path": "State.Name",
        "values": [ "running" ]
    }
```

```
],
```

Untuk memilih instance yang memiliki salah satu dari sejumlah grup keamanan yang dilampirkan, tentukan satu filter dengan jalur atribut untuk `GroupId` dan beberapa grup IDs keamanan. Misalnya:

```
"filters": [
    {
        "path": "NetworkInterfaces.Groups.GroupId",
        "values": [
                "sg-a0011223344556677",
                "sg-f1100110011001100"
            ]
    }
],
```

Untuk memilih instance yang memiliki semua grup keamanan yang dilampirkan, tentukan beberapa filter dengan jalur atribut untuk `GroupId` dan satu ID grup keamanan untuk setiap filter. Misalnya:

```
"filters": [
    {
        "path": "NetworkInterfaces.Groups.GroupId",
        "values": [
            "sg-a0011223344556677"
        ]
    },
    {
        "path": "NetworkInterfaces.Groups.GroupId",
        "values": [
            "sg-b9988776655443322"
        ]
    }
],
```

Contoh: Cluster Amazon RDS (cluster DB)

Saat Anda menentukan filter untuk tindakan yang mendukung tipe sumber daya aws:rds:cluster, FIS AWS menjalankan describe-db-clusters perintah Amazon RDS dan menerapkan filter untuk mengidentifikasi target.

describe-db-clustersPerintah mengembalikan output JSON mirip dengan berikut untuk setiap cluster DB. Berikut ini adalah output sebagian yang mencakup bidang yang ditandai dengan*italics*. Kami

akan memberikan contoh yang menggunakan bidang ini untuk menentukan jalur atribut dari struktur output JSON.

```
[
    {
        "AllocatedStorage": 1,
        "AvailabilityZones": [
            "us-east-2a",
            "us-east-2b",
            "us-east-2c"
        ],
        "BackupRetentionPeriod": 7,
        "DatabaseName": "",
        "DBClusterIdentifier": "database-1",
        "DBClusterParameterGroup": "default.aurora-postgresql11",
        "DBSubnetGroup": "default-vpc-01234567abc123456",
        "Status": "available",
        "EarliestRestorableTime": "2020-11-13T15:08:32.211Z",
        "Endpoint": "database-1.cluster-example.us-east-2.rds.amazonaws.com",
        "ReaderEndpoint": "database-1.cluster-ro-example.us-east-2.rds.amazonaws.com",
        "MultiAZ": false,
        "Engine": "aurora-postgresql",
        "EngineVersion": "11.7",
        ...
    }
]
```

Untuk menerapkan filter sumber daya yang hanya mengembalikan cluster DB yang menggunakan mesin DB tertentu, tentukan jalur atribut sebagai `Engine` dan nilai `aurora-postgresql` seperti yang ditunjukkan pada contoh berikut.

```
"filters": [
    {
        "path": "Engine",
        "values": [ "aurora-postgresql" ]
    }
],
```

Untuk menerapkan filter sumber daya yang hanya mengembalikan kluster DB di Availability Zone tertentu, tentukan jalur atribut dan nilai seperti yang ditunjukkan pada contoh berikut.

```
"filters": [
```

```
    {
        "path": "AvailabilityZones",
        "values": [ "us-east-2a" ]
    }
],
```

Kondisi berhenti untuk AWS FIS

AWS AWS Fault Injection Service (FIS) menyediakan kontrol dan pagar pembatas bagi Anda untuk menjalankan eksperimen dengan aman pada beban kerja. AWS Kondisi berhenti adalah mekanisme untuk menghentikan eksperimen jika mencapai ambang batas yang Anda definisikan sebagai CloudWatch alarm Amazon. Jika kondisi berhenti dipicu selama percobaan, AWS FIS menghentikan percobaan. Anda tidak dapat melanjutkan eksperimen yang dihentikan.

Untuk membuat kondisi berhenti, pertama-tama tentukan status tunak untuk aplikasi atau layanan Anda. Steady state adalah ketika aplikasi Anda berkinerja optimal, didefinisikan dalam hal metrik bisnis atau teknis. Misalnya, latensi, beban CPU, atau jumlah percobaan ulang. Anda dapat menggunakan kondisi tunak untuk membuat CloudWatch alarm yang dapat Anda gunakan untuk menghentikan eksperimen jika aplikasi atau layanan Anda mencapai keadaan di mana kinerjanya tidak dapat diterima. Untuk informasi selengkapnya, lihat Menggunakan CloudWatch alarm Amazon di Panduan CloudWatch Pengguna Amazon.

Akun Anda memiliki kuota pada jumlah kondisi berhenti yang dapat Anda tentukan dalam templat percobaan. Untuk informasi selengkapnya, lihat Kuota dan batasan untuk Layanan Injeksi AWS Kesalahan.

Stop sintaks kondisi

Saat membuat templat eksperimen, Anda menentukan satu atau beberapa kondisi penghentian dengan menentukan CloudWatch alarm yang Anda buat.

```
{
    "stopConditions": [
        {
            "source": "aws:cloudwatch:alarm",
            "value": "arn:aws:cloudwatch:region:123456789012:alarm:alarm-name"
        }
    ]
}
```

Contoh berikut menunjukkan bahwa template percobaan tidak menentukan kondisi berhenti.

```
{
    "stopConditions": [
        {
            "source": "none"
        }
    ]
}
```

Pelajari selengkapnya

Untuk tutorial yang menunjukkan cara membuat CloudWatch alarm dan menambahkan kondisi berhenti ke template eksperimen, lihatJalankan stress CPU pada sebuah instance.

Untuk informasi selengkapnya tentang CloudWatch metrik yang tersedia untuk jenis sumber daya yang didukung oleh AWS FIS, lihat berikut ini:

- Pantau instans Anda menggunakan CloudWatch
- Metrik Amazon ECS CloudWatch
- Memantau metrik Amazon RDS menggunakan CloudWatch
- Memantau metrik Run Command menggunakan CloudWatch

Peran IAM untuk eksperimen AWS FIS

AWS Identity and Access Management (IAM) adalah AWS layanan yang membantu administrator mengontrol akses ke AWS sumber daya dengan aman. Untuk menggunakan AWS FIS, Anda harus membuat peran IAM yang memberikan AWS FIS izin yang diperlukan sehingga AWS FIS dapat menjalankan eksperimen atas nama Anda. Anda menentukan peran eksperimen ini saat membuat templat eksperimen. Untuk eksperimen satu akun, kebijakan IAM untuk peran eksperimen harus memberikan izin untuk mengubah sumber daya yang Anda tetapkan sebagai target dalam templat eksperimen Anda. Untuk eksperimen multi-akun, peran eksperimen harus memberikan izin peran orkestrator untuk mengambil peran IAM untuk setiap akun target. Untuk informasi selengkapnya, lihat Izin untuk eksperimen multi-akun.

Kami menyarankan Anda mengikuti praktik keamanan standar dengan memberikan hak istimewa paling sedikit. Anda dapat melakukannya dengan menentukan sumber daya ARNs atau tag tertentu dalam kebijakan Anda.

Untuk membantu Anda memulai AWS FIS dengan cepat, kami menyediakan kebijakan AWS terkelola yang dapat Anda tentukan saat membuat peran eksperimen. Atau, Anda juga dapat menggunakan kebijakan ini sebagai model saat Anda membuat dokumen kebijakan inline Anda sendiri.

Daftar Isi

- Prasyarat
- Opsi 1: Buat peran eksperimen dan lampirkan kebijakan AWS terkelola
- Opsi 2: Buat peran eksperimen dan tambahkan dokumen kebijakan sebaris

Prasyarat

Sebelum Anda mulai, instal AWS CLI dan buat kebijakan kepercayaan yang diperlukan.

Instal AWS CLI

Sebelum Anda mulai, instal dan konfigurasikan file AWS CLI. Ketika Anda mengkonfigurasi AWS CLI, Anda akan diminta untuk AWS kredensialnya. Contoh dalam prosedur ini mengasumsikan bahwa Anda juga mengonfigurasi Wilayah default. Jika tidak, tambahkan `--region` opsi ke setiap perintah. Untuk informasi selengkapnya, lihat Menginstal atau memperbarui AWS CLI dan Mengonfigurasi. AWS CLI

Buat kebijakan hubungan kepercayaan

Peran eksperimen harus memiliki hubungan kepercayaan yang memungkinkan layanan AWS FIS untuk mengambil peran tersebut. Buat file teks bernama `fis-role-trust-policy.json` dan tambahkan kebijakan hubungan kepercayaan berikut.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
                  "fis.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole"
        }
    ]
```

```
}
```

Kami menyarankan Anda menggunakan kunci syarat `aws:SourceAccount` dan `aws:SourceArn` untuk melindungi diri Anda dari masalah wakil yang membingungkan. Akun sumber adalah pemilik eksperimen dan sumber ARN adalah ARN percobaan. Misalnya, Anda harus menambahkan blok kondisi berikut ke kebijakan kepercayaan Anda.

```
"Condition": {
    "StringEquals": {
        "aws:SourceAccount": "account_id"
    },
    "ArnLike": {
        "aws:SourceArn": "arn:aws:fis:region:account_id:experiment/*"
    }
}
```

Tambahkan izin untuk mengambil peran akun target (hanya eksperimen multi-akun)

Untuk eksperimen multi-akun, Anda memerlukan izin yang memungkinkan akun orkestrator untuk mengambil peran akun target. Anda dapat mengubah contoh berikut dan menambahkan sebagai dokumen kebijakan inline untuk mengambil peran akun target:

```
{
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource":[
        "arn:aws:iam::target_account_id:role/role_name"
    ]
}
```

Opsi 1: Buat peran eksperimen dan lampirkan kebijakan AWS terkelola

Gunakan salah satu kebijakan AWS terkelola dari AWS FIS untuk memulai dengan cepat.

Untuk membuat peran eksperimen dan melampirkan kebijakan AWS terkelola

1. Verifikasi bahwa ada kebijakan terkelola untuk tindakan AWS FIS dalam eksperimen Anda. Jika tidak, Anda harus membuat dokumen kebijakan inline Anda sendiri. Untuk informasi selengkapnya, lihat the section called “AWS kebijakan terkelola”.

2. Gunakan perintah create-role berikut untuk membuat peran dan tambahkan kebijakan kepercayaan yang Anda buat dalam prasyarat.

```
aws iam create-role --role-name my-fis-role --assume-role-policy-document
 file://fis-role-trust-policy.json
```

3. Gunakan attach-role-policyperintah berikut untuk melampirkan kebijakan AWS terkelola.

```
aws iam attach-role-policy --role-name my-fis-role --policy-arn fis-policy-arn
```

*fis-policy-arn*Dimana salah satu dari berikut ini:

- arn:aws:iam::aws:policy/service-role/AWSFaultInjectionSimulatorEC2Access
- arn:aws:iam::aws:policy/service-role/AWSFaultInjectionSimulatorECSAccess
- arn:aws:iam::aws:policy/service-role/AWSFaultInjectionSimulatorEKSAccess
- arn:aws:iam::aws:policy/service-role/AWSFaultInjectionSimulatorNetworkAccess
- arn:aws:iam::aws:policy/service-role/AWSFaultInjectionSimulatorRDSAccess
- arn:aws:iam::aws:policy/service-role/AWSFaultInjectionSimulatorSSMAccess

Opsi 2: Buat peran eksperimen dan tambahkan dokumen kebijakan sebaris

Gunakan opsi ini untuk tindakan yang tidak memiliki kebijakan terkelola, atau hanya menyertakan izin yang diperlukan untuk eksperimen spesifik Anda.

Untuk membuat eksperimen dan menambahkan dokumen kebijakan inline

1. Gunakan perintah create-role berikut untuk membuat peran dan tambahkan kebijakan kepercayaan yang Anda buat dalam prasyarat.

```
aws iam create-role --role-name my-fis-role --assume-role-policy-document
 file://fis-role-trust-policy.json
```

2. Buat file teks bernama `fis-role-permissions-policy.json` dan tambahkan kebijakan izin. Untuk contoh yang dapat Anda gunakan sebagai titik awal, lihat yang berikut ini.

   - Tindakan injeksi kesalahan - Mulai dari kebijakan berikut.

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowFISExperimentRoleFaultInjectionActions",
            "Effect": "Allow",
            "Action": [
                "fis:InjectApiInternalError",
                "fis:InjectApiThrottleError",
                "fis:InjectApiUnavailableError"
            ],
            "Resource": "arn:*:fis:*:*:experiment/*"
        }
    ]
}
```

- Tindakan Amazon EBS - Mulai dari kebijakan berikut.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeVolumes"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:PauseVolumeIO"
            ],
            "Resource": "arn:aws:ec2:*:*:volume/*"
        }
    ]
}
```

- EC2 Tindakan Amazon - Mulai dari kebijakan AWSFaultInjectionSimulatorEC2Akses.
- Tindakan Amazon ECS — Mulai dari AWSFaultInjectionSimulatorECSAccesskebijakan.
- Tindakan Amazon EKS - Mulai dari AWSFaultInjectionSimulatorEKSAccesskebijakan.
- Tindakan jaringan — Mulai dari AWSFaultInjectionSimulatorNetworkAccesskebijakan.
- Tindakan Amazon RDS — Mulai dari AWSFaultInjectionSimulatorRDSAccesskebijakan.

- Tindakan Systems Manager — Mulai dari AWSFaultInjectionSimulatorSSMAccesskebijakan.

3. Gunakan put-role-policyperintah berikut untuk menambahkan kebijakan izin yang Anda buat di langkah sebelumnya.

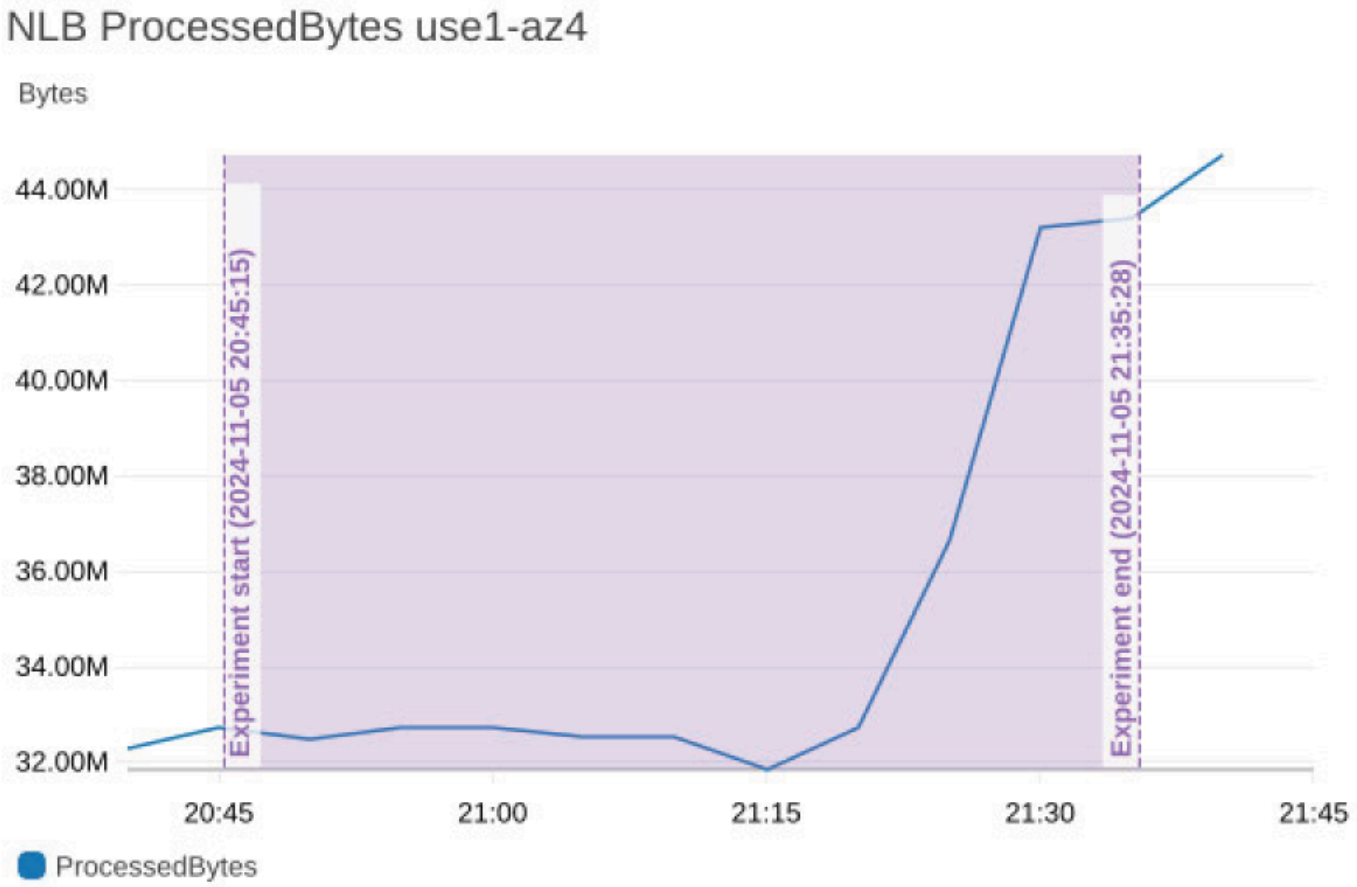
```
aws iam put-role-policy --role-name my-fis-role --policy-name my-fis-policy --
policy-document file://fis-role-permissions-policy.json
```

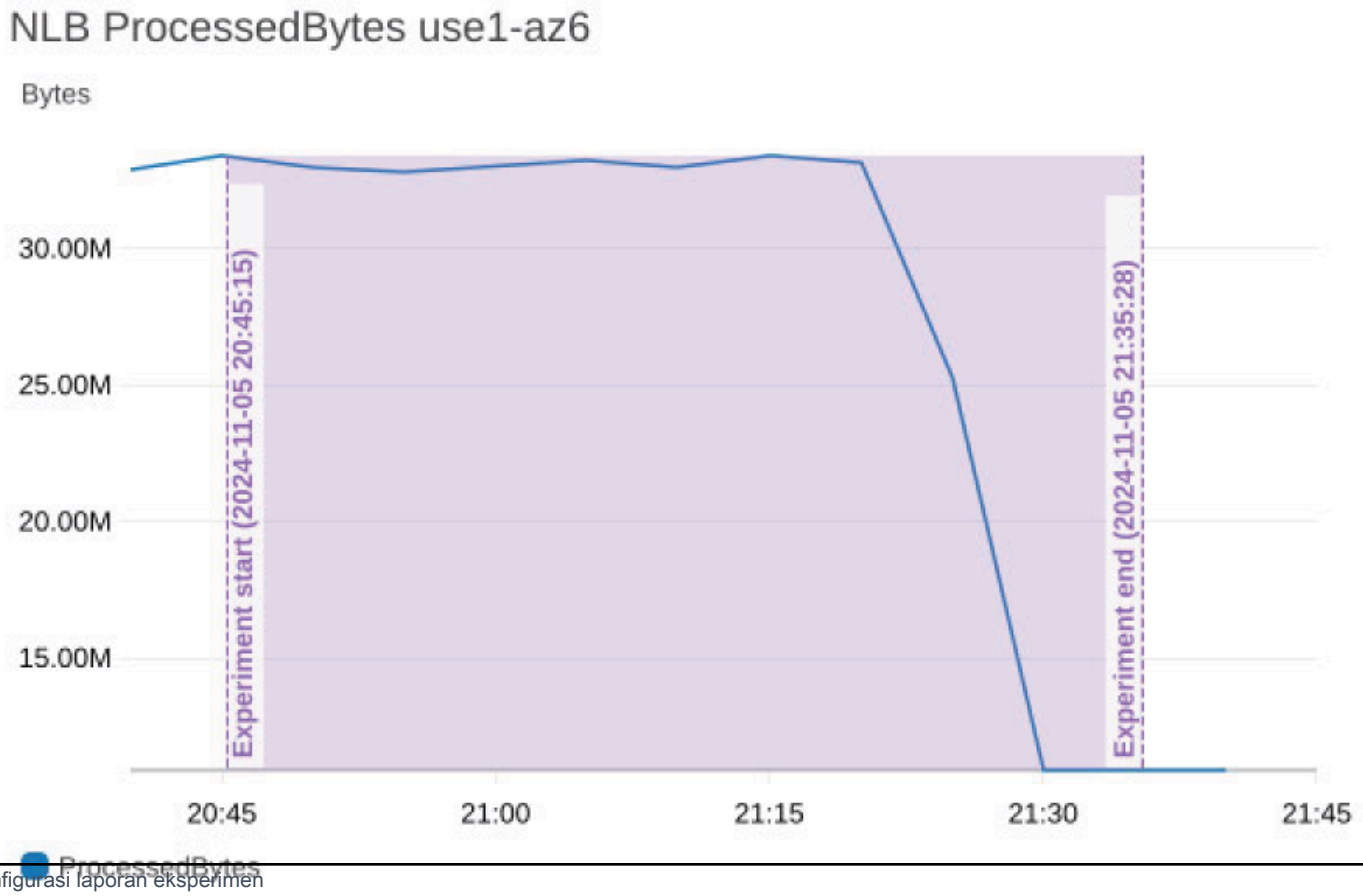
Konfigurasi laporan eksperimen untuk AWS FIS

Anda dapat mengaktifkan AWS Fault Injection Service (FIS) untuk menghasilkan laporan untuk eksperimen, sehingga lebih mudah untuk menghasilkan bukti pengujian ketahanan. Laporan eksperimen adalah dokumen PDF yang merangkum tindakan eksperimen dan secara opsional menangkap respons aplikasi dari CloudWatch dasbor yang Anda tentukan. Untuk melihat contoh laporan percobaan, unduh file zip di sini.

Untuk mengaktifkan dan mengonfigurasi konten laporan yang dihasilkan untuk eksperimen, Anda menentukan konfigurasi laporan eksperimen untuk templat eksperimen. Saat Anda menentukan CloudWatch dasbor, AWS FIS menyertakan grafik snapshot dari semua widget di dasbor yang diberikan yang dianotasi dengan waktu mulai dan berakhir eksperimen selama durasi yang Anda tentukan, seperti yang ditunjukkan pada contoh di bawah ini.

Contoh ini menunjukkan dampak dari percobaan packet loss di Availability Zone (AZ). Ketika packet loss diperkenalkan di AZ use1-az6, lalu lintas bergeser dari use1-az6 dan menuju use1-az4, sehingga jumlah byte yang diproses oleh penyeimbang beban di AZ tersebut menurun.


NLB ProcessedBytes use1-az4
Bytes
44.00M
42.00M
40.00M
38.00M
36.00M
34.00M
32.00M
Experiment start (2024-11-05 20:45:15)
Experiment end (2024-11-05 21:35:28)
20:45
21:00
21:15
21:30
21:45
ProcessedBytes


NLB ProcessedBytes use1-az6
Bytes
30.00M
25.00M
20.00M
15.00M
Experiment start (2024-11-05 20:45:15)
Experiment end (2024-11-05 21:35:28)
20:45
21:00
21:15
21:30
21:45
ProcessedBytes

Ketika percobaan berakhir, laporan dapat diunduh dari konsol AWS FIS dan juga disimpan dalam ember Amazon S3. Jika Anda menyertakan CloudWatch dasbor dalam konfigurasi laporan Anda, gambar dari setiap widget juga dikirimkan. Laporan tidak dibuat untuk eksperimen yang sedang `cancelled` atau dijalankan sebagai bagian dari pratinjau target (dengan ActionsMode disetel ke). `skip-all` Setelah eksperimen melebihi batas retensi data eksperimen, laporan hanya akan tersedia dari bucket Amazon S3. AWS Biaya FIS berlaku untuk setiap laporan yang dikirimkan, kecuali yang gagal dengan kesalahan internal. Untuk informasi selengkapnya, lihat harga Layanan Injeksi AWS Kesalahan danKuota dan batasan untuk Layanan Injeksi AWS Kesalahan. Biaya konsumsi dan penyimpanan untuk Amazon S3 CloudWatch dan biaya API GetMetricWidgetImageuntuk GetDashboarddan permintaan mungkin berlaku. Untuk informasi selengkapnya, lihat harga dan CloudWatch harga Amazon S3.

Daftar Isi

- Sintaks konfigurasi laporan eksperimen
- Izin laporan eksperimen
- Praktik terbaik laporan eksperimen

Sintaks konfigurasi laporan eksperimen

Berikut ini adalah sintaks untuk konfigurasi laporan eksperimen, bagian opsional dari template eksperimen.

```
{
    "experimentReportConfiguration": {
        "outputs": {
            "s3Configuration": {
                "bucketName": "my-bucket-name",
                "prefix": "report-storage-prefix"
            }
        },
        "dataSources": {
            "cloudWatchDashboards": [
                {
                    "dashboardIdentifier": "arn:aws:cloudwatch::123456789012:dashboard/
MyDashboard"
                }
            ]
        },
        "preExperimentDuration": "PT20M",
```

```
        "postExperimentDuration": "PT20M"
    }
}
```

Dengan menggunakan`experimentReportConfiguration`, Anda dapat menyesuaikan tujuan keluaran, data input, dan jendela waktu agar data dapat disertakan dalam laporan eksperimen, yang dapat membantu Anda lebih memahami dampak dan hasil eksperimen AWS FIS Anda. Saat Anda menentukan konfigurasi laporan eksperimen, Anda memberikan yang berikut:

keluaran

Bagian `experimentReportConfiguration` yang menentukan di mana laporan percobaan akan dikirimkan. Di`outputs`, Anda menentukan `s3Configuration` dengan memberikan yang berikut:

- `bucketName`- Nama bucket Amazon S3 tempat laporan akan disimpan. Ember harus berada di wilayah yang sama dengan percobaan.
- `prefix`(Opsional) - Awalan dalam bucket Amazon S3 tempat laporan akan disimpan. Bidang ini sangat disarankan sehingga Anda dapat membatasi akses ke awalan saja.

Sumber Data

Bagian opsional `experimentReportConfiguration` yang menentukan sumber data tambahan yang akan disertakan dalam laporan percobaan.

- `cloudWatchDashboards`- Array CloudWatch dasbor yang akan disertakan dalam laporan. Terbatas untuk satu CloudWatch dasbor.
- `dashboardIdentifier`- ARN dasbor. CloudWatch Grafik snapshot dari setiap widget dengan tipe `metric` di dasbor ini akan disertakan dalam laporan, dengan pengecualian metrik lintas wilayah.

preExperimentDuration

Bagian opsional `experimentReportConfiguration` yang menentukan durasi pra-eksperimen untuk metrik CloudWatch dasbor untuk disertakan dalam laporan, hingga 30 menit. Ini harus menjadi periode yang mewakili kondisi tunak aplikasi Anda. Misalnya, durasi pra-percobaan 5 menit berarti grafik snapshot akan menyertakan metrik 5 menit sebelum percobaan dimulai. Format untuk durasi adalah ISO 8601 dan defaultnya adalah 20 menit.

postExperimentDuration

Bagian opsional `experimentReportConfiguration` yang menentukan durasi pasca-eksperimen untuk metrik CloudWatch dasbor untuk disertakan dalam laporan, hingga 2 jam.

Ini harus menjadi durasi yang mewakili kondisi tunak aplikasi atau periode pemulihan Anda. Misalnya, jika Anda menentukan durasi pasca-percobaan 5 menit, grafik snapshot akan menyertakan metrik hingga 5 menit setelah percobaan berakhir. Format untuk durasi adalah ISO 8601 dan defaultnya adalah 20 menit.

Izin laporan eksperimen

Untuk mengaktifkan AWS FIS menghasilkan dan menyimpan laporan eksperimen, Anda harus mengizinkan operasi berikut dari peran IAM eksperimen AWS FIS Anda:

- `cloudwatch:GetDashboard`
- `cloudwatch:GetMetricWidgetImage`
- `s3:GetObject`
- `s3:PutObject`

Kami menyarankan Anda mengikuti praktik terbaik AWS keamanan dan membatasi peran eksperimen ke bucket dan awalan. Berikut ini adalah contoh pernyataan kebijakan yang membatasi akses peran eksperimen.

```
{
    "Version": "2012-10-17",
    "Statement":
        [
            {
                "Action": [
                    "s3:PutObject",
                    "s3:GetObject"
                    ],
                "Resource": "arn:aws:s3:::my-experiment-report-bucket/my-prefix/*",
                "Effect": "Allow"
            },
            {
                "Action": [
                    "cloudwatch:GetDashboard"
                    ],
                "Resource": "arn:aws:cloudwatch::012345678912:dashboard/my-experiment-
report-dashboard",
                "Effect": "Allow"
            },
```

```
            {
                "Action": [
                    "cloudwatch:GetMetricWidgetImage"
                    ],
                "Resource": "*",
                "Effect": "Allow"
            }
        ]
}
```

Izin tambahan untuk laporan yang dikirimkan ke bucket Amazon S3 yang dienkripsi dengan kunci terkelola pelanggan (CMK)

Jika bucket Amazon S3 yang Anda tentukan `S3Configuration` dienkripsi dengan CMK, Anda harus memberikan izin tambahan berikut untuk peran eksperimen FIS pada kebijakan kunci KMS Anda:

- `kms:GenerateDataKey`
- `kms:Decrypt`

Berikut ini adalah contoh pernyataan kebijakan kunci KMS yang memungkinkan peran eksperimen FIS untuk menulis laporan ke bucket terenkripsi:

```
{
    "Sid": "Allow FIS experiment report",
    "Effect": "Allow",
    "Principal":
    {
        "AWS": [
            "arn:aws:iam::012345678912:role/FISExperimentRole",
        ]
    },
    "Action": [
        "kms:Decrypt",
        "kms:GenerateDataKey"
        ],
    "Resource": "*"
   }
```

Praktik terbaik laporan eksperimen

Berikut ini adalah praktik terbaik untuk menggunakan konfigurasi laporan eksperimen AWS FIS:

- Sebelum memulai eksperimen, buat pratinjau target untuk memverifikasi bahwa templat eksperimen Anda dikonfigurasi seperti yang Anda harapkan. Pratinjau target akan memberi Anda informasi tentang target yang diharapkan dari eksperimen Anda. Untuk mempelajari selengkapnya, lihat Buat pratinjau target dari templat eksperimen.
- Laporan tidak boleh digunakan untuk memecahkan masalah eksperimen yang gagal. Sebagai gantinya, gunakan log eksperimen untuk memecahkan masalah kesalahan eksperimen. Kami menyarankan Anda mengandalkan laporan hanya untuk eksperimen yang sebelumnya telah Anda jalankan dan berhasil diselesaikan.
- Batasi put peran IAM percobaan dan dapatkan akses objek ke bucket dan awalan tujuan S3. Kami menyarankan Anda mendedikasikan awalan bucket/hanya untuk laporan eksperimen AWS FIS, dan tidak memberikan akses AWS layanan lain ke bucket dan awalan ini.
- Gunakan Amazon S3 Object Lock untuk mencegah laporan dihapus atau ditimpa untuk jangka waktu tertentu atau tanpa batas waktu. Untuk mempelajari lebih lanjut, lihat Mengunci objek dengan Object Lock.
- Jika CloudWatch dasbor Anda berada di akun terpisah dalam wilayah yang sama, Anda dapat menggunakan observabilitas CloudWatch lintas akun untuk mengaktifkan akun orkestrator AWS FIS Anda sebagai akun pemantauan dan akun terpisah sebagai akun sumber dari CloudWatch konsol atau perintah Observability Access Manager di API dan. AWS CLI Untuk mempelajari lebih lanjut, lihat CloudWatch observabilitas lintas akun.

Opsi percobaan untuk AWS FIS

Opsi eksperimen adalah pengaturan opsional untuk eksperimen. Anda dapat menentukan opsi eksperimen tertentu pada templat eksperimen. Opsi eksperimen tambahan ditetapkan saat Anda memulai percobaan.

Berikut ini adalah sintaks untuk opsi eksperimen yang Anda tentukan pada template eksperimen.

```
{
        "experimentOptions": {
        "accountTargeting": "single-account | multi-account",
            "emptyTargetResolutionMode": "fail | skip"
    }
```

```
}
```

Jika Anda tidak menentukan opsi eksperimen apa pun saat membuat templat eksperimen, default untuk setiap opsi akan digunakan.

Berikut ini adalah sintaks untuk opsi eksperimen yang Anda tetapkan saat memulai eksperimen.

```
{
        "experimentOptions": {
            "actionsMode": "run-all | skip-all"
     }
}
```

Jika Anda tidak menentukan opsi eksperimen apa pun saat memulai percobaan, default akan `run-all` digunakan.

Daftar Isi

- Penargetan akun
- Mode resolusi target kosong
- Mode tindakan

Penargetan akun

Jika Anda memiliki beberapa AWS akun dengan sumber daya yang ingin Anda targetkan dalam eksperimen, Anda dapat menentukan eksperimen multi-akun menggunakan opsi eksperimen penargetan akun. Anda menjalankan eksperimen multi-akun dari akun orkestrator yang memengaruhi sumber daya di beberapa akun target. Akun orkestrator memiliki templat eksperimen dan AWS FIS eksperimen. Akun target adalah akun AWS individual dengan sumber daya yang dapat dipengaruhi oleh AWS FIS eksperimen. Untuk informasi selengkapnya, lihat Bekerja dengan eksperimen multi-akun untuk AWS FIS.

Anda menggunakan penargetan akun untuk menunjukkan lokasi sumber daya target Anda. Anda dapat memberikan dua nilai untuk penargetan akun:

- akun tunggal — Default. Eksperimen hanya akan menargetkan sumber daya di AWS akun tempat AWS FIS eksperimen berjalan.
- multi-akun — Eksperimen dapat menargetkan sumber daya di beberapa akun AWS.

Konfigurasi akun target

Untuk menjalankan eksperimen multi-akun, Anda harus menentukan satu atau beberapa konfigurasi akun target. Konfigurasi akun target menentukan accountID, roLearn, dan deskripsi untuk setiap akun dengan sumber daya yang ditargetkan dalam percobaan. Akun IDs konfigurasi akun target untuk templat eksperimen harus unik.

Saat Anda membuat templat eksperimen multi-akun, templat eksperimen akan menampilkan bidang hanya-bacatargetAccountConfigurationsCount, yang merupakan hitungan dari semua konfigurasi akun target untuk templat eksperimen.

Berikut ini adalah sintaks untuk konfigurasi akun target.

```
{
    accountId: "123456789012",
    roleArn: "arn:aws:iam::123456789012:role/AllowFISActions",
    description: "fis-ec2-test"
}
```

Saat Anda membuat konfigurasi akun target, Anda memberikan yang berikut:

accountId

12 digit ID akun AWS dari akun target.

roleArn

Peran IAM yang memberikan AWS FIS izin untuk mengambil tindakan di akun target.

description

Deskripsi opsional.

Untuk mempelajari lebih lanjut tentang cara bekerja dengan konfigurasi akun target, lihatBekerja dengan eksperimen multi-akun untuk AWS FIS.

Mode resolusi target kosong

Mode ini memberi Anda opsi untuk memungkinkan eksperimen selesai bahkan ketika sumber daya target tidak diselesaikan.

- gagal — Default. Jika tidak ada sumber daya yang diselesaikan untuk target, percobaan segera dihentikan dengan status. `failed`

- lewati — Jika tidak ada sumber daya yang diselesaikan untuk target, percobaan akan dilanjutkan dan tindakan apa pun tanpa target yang diselesaikan dilewati. Tindakan dengan target yang ditentukan menggunakan pengidentifikasi unik, seperti ARNs, tidak dapat dilewati. Jika target yang ditentukan menggunakan pengenal unik tidak ditemukan, eksperimen segera dihentikan dengan status `failed`

Mode tindakan

Mode tindakan adalah parameter opsional yang dapat Anda tentukan saat memulai eksperimen. Anda dapat mengatur mode tindakan `skip-all` untuk menghasilkan pratinjau target sebelum menyuntikkan kesalahan ke sumber daya target Anda. Pratinjau target memungkinkan Anda memverifikasi hal-hal berikut:

- Bahwa Anda telah mengonfigurasi template eksperimen Anda untuk menargetkan sumber daya yang Anda harapkan. Sumber daya aktual yang ditargetkan saat Anda memulai eksperimen ini mungkin berbeda dari pratinjau karena sumber daya dapat dihapus, diperbarui, atau diambil sampelnya secara acak.
- Bahwa konfigurasi logging Anda diatur dengan benar.
- Untuk eksperimen multi-akun, Anda telah menyiapkan peran IAM dengan benar untuk setiap konfigurasi akun target Anda.

> **Note**
>
> `skip-all`Mode ini tidak memungkinkan Anda untuk memverifikasi bahwa Anda memiliki izin yang diperlukan untuk menjalankan AWS FIS eksperimen dan mengambil tindakan pada sumber daya Anda.

Parameter mode tindakan menerima nilai-nilai berikut:

- `run-all`- (Default) Eksperimen akan mengambil tindakan pada sumber daya target.
- `skip-all`- Eksperimen akan melewati semua tindakan pada sumber daya target.

Untuk mempelajari lebih lanjut tentang cara mengatur parameter mode tindakan saat Anda memulai eksperimen, lihatBuat pratinjau target dari templat eksperimen.

AWS FIS Referensi tindakan

Tindakan adalah aktivitas injeksi kesalahan yang Anda jalankan pada target menggunakan AWS Fault Injection Service (AWS FIS). AWS FIS menyediakan tindakan yang telah dikonfigurasi sebelumnya untuk jenis target tertentu di seluruh AWS layanan. Anda menambahkan tindakan ke templat eksperimen, yang kemudian Anda gunakan untuk menjalankan eksperimen.

Referensi ini menjelaskan tindakan umum AWS FIS, termasuk informasi tentang parameter tindakan dan izin IAM yang diperlukan. Anda juga dapat mencantumkan AWS FIS tindakan yang didukung menggunakan AWS FIS konsol atau perintah list-action dari AWS Command Line Interface ()AWS CLI. Setelah Anda memiliki nama tindakan tertentu, Anda dapat melihat informasi rinci tentang tindakan dengan menggunakan perintah get-action. Untuk informasi selengkapnya tentang penggunaan AWS FIS perintah dengan AWS CLI, lihat Panduan AWS Command Line Interface Pengguna dan fis di Referensi AWS CLI Perintah.

Untuk informasi selengkapnya tentang cara kerja AWS FIS tindakan, lihat Tindakan untuk AWS FIS danBagaimana Layanan Injeksi AWS Kesalahan bekerja dengan IAM.

Tindakan

- Tindakan injeksi kesalahan
- Tindakan pemulihan
- Tunggu tindakan
- CloudWatch Tindakan Amazon
- Tindakan Amazon DynamoDB
- Tindakan Amazon EBS
- EC2 Tindakan Amazon
- Tindakan Amazon ECS
- Tindakan Amazon EKS
- ElastiCache Tindakan Amazon
- AWS Lambda tindakan
- Tindakan jaringan
- Tindakan Amazon RDS
- Tindakan Amazon S3

- Tindakan Systems Manager
- Menggunakan dokumen SSM Systems Manager dengan AWS FIS
- Gunakan tindakan AWS FIS aws:ecs:task
- Gunakan tindakan AWS FIS aws:eks:pod
- Gunakan tindakan AWS FIS aws:lambda: function

Tindakan injeksi kesalahan

AWS FIS mendukung tindakan injeksi kesalahan berikut.

Tindakan

- aws:fis:inject-api-internal-error
- aws:fis:inject-api-throttle-error
- aws:fis:inject-api-unavailable-error

aws:fis:inject-api-internal-error

Menyuntikkan Kesalahan Internal ke dalam permintaan yang dibuat oleh peran IAM target. Respons spesifik tergantung pada setiap layanan dan API. Untuk informasi selengkapnya, silakan tinjau dokumentasi SDK dan API layanan Anda.

Jenis sumber daya

- aws:iam:role

Parameter

- duration— Durasi, dari satu menit hingga 12 jam. Di AWS FIS API, nilainya adalah string dalam format ISO 8601. Misalnya, PT1 M mewakili satu menit. Di AWS FIS konsol, Anda memasukkan jumlah detik, menit, atau jam.
- service— Namespace AWS API target. Nilai yang didukung adalah `ec2`.
- percentage— Persentase (1-100) panggilan untuk menyuntikkan kesalahan ke dalam.
- operations— Operasi untuk menyuntikkan kesalahan ke dalam, dipisahkan menggunakan koma. Untuk daftar tindakan API untuk `ec2` namespace, lihat Tindakan di Referensi EC2 API Amazon.

Izin

- `fis:InjectApiInternalError`

aws:fis:inject-api-throttle-error

Menyuntikkan kesalahan pelambatan ke dalam permintaan yang dibuat oleh peran IAM target. Respons spesifik tergantung pada setiap layanan dan API. Untuk informasi selengkapnya, silakan tinjau dokumentasi SDK dan API layanan Anda.

Jenis sumber daya

- aws:iam:role

Parameter

- duration— Durasi, dari satu menit hingga 12 jam. Di AWS FIS API, nilainya adalah string dalam format ISO 8601. Misalnya, PT1 M mewakili satu menit. Di AWS FIS konsol, Anda memasukkan jumlah detik, menit, atau jam.
- service— Namespace AWS API target. Nilai yang didukung adalah `ec2`.
- percentage— Persentase (1-100) panggilan untuk menyuntikkan kesalahan ke dalam.
- operations— Operasi untuk menyuntikkan kesalahan ke dalam, dipisahkan menggunakan koma. Untuk daftar tindakan API untuk `ec2` namespace, lihat Tindakan di Referensi EC2 API Amazon.

Izin

- `fis:InjectApiThrottleError`

aws:fis:inject-api-unavailable-error

Menyuntikkan kesalahan yang tidak tersedia ke dalam permintaan yang dibuat oleh peran IAM target. Respons spesifik tergantung pada setiap layanan dan API. Untuk informasi selengkapnya, silakan tinjau dokumentasi SDK dan API layanan Anda.

Jenis sumber daya

- aws:iam:role

Parameter

- duration— Durasi, dari satu menit hingga 12 jam. Di AWS FIS API, nilainya adalah string dalam format ISO 8601. Misalnya, PT1 M mewakili satu menit. Di AWS FIS konsol, Anda memasukkan jumlah detik, menit, atau jam.
- service— Namespace AWS API target. Nilai yang didukung adalah `ec2`.
- percentage— Persentase (1-100) panggilan untuk menyuntikkan kesalahan ke dalam.
- operations— Operasi untuk menyuntikkan kesalahan ke dalam, dipisahkan menggunakan koma. Untuk daftar tindakan API untuk `ec2` namespace, lihat Tindakan di Referensi EC2 API Amazon.

Izin

- `fis:InjectApiUnavailableError`

Tindakan pemulihan

Tindakan pemulihan dilakukan untuk mengurangi risiko atau melindungi aplikasi setelah kerusakan.

AWS FIS mendukung tindakan pemulihan berikut.

aws:arc:start-zonal-autoshift

Secara otomatis mengalihkan lalu lintas untuk sumber daya yang didukung dari Availability Zone (AZ) yang berpotensi mengalami gangguan dan merutingnya menjadi sehat AZs di Wilayah AWS yang sama. Hal ini memungkinkan untuk mengalami pergeseran otomatis zona melalui FIS. Zonal autoshift adalah kemampuan di Amazon Application Recovery Controller (ARC) yang memungkinkan AWS untuk mengalihkan lalu lintas untuk sumber daya yang jauh dari AZ, atas nama Anda, ketika AWS menentukan bahwa ada gangguan yang berpotensi mempengaruhi pelanggan di AZ.

Saat Anda menjalankan `aws:arc:start-zonal-autoshift` tindakan, AWS FIS mengelola pergeseran zona menggunakan StartZonalShift, UpdateZonalShift, dan CancelZonalShift APIs dengan `expiresIn` bidang untuk permintaan ini disetel ke 1 menit sebagai mekanisme keamanan. Hal ini memungkinkan AWS FIS untuk dengan cepat mengembalikan pergeseran zona jika terjadi peristiwa tak terduga seperti pemadaman jaringan atau masalah sistem. Di konsol ARC, bidang waktu kedaluwarsa akan menampilkan AWS FIS-managed, dan kadaluarsa yang diharapkan sebenarnya ditentukan oleh durasi yang ditentukan dalam aksi pergeseran zona.

Jenis sumber daya

- aws:arc:zonal-shift-managed-resource

  Sumber daya yang dikelola pergeseran zona adalah jenis sumber daya termasuk kluster Amazon EKS, EC2 Aplikasi Amazon dan Penyeimbang Beban Jaringan, dan grup Auto Scaling Amazon yang dapat EC2 diaktifkan untuk pergeseran otomatis zona ARC. Untuk informasi selengkapnya, lihat sumber daya yang didukung dan mengaktifkan sumber daya pergeseran otomatis zona di Panduan Pengembang ARC.

Parameter

- duration— Lamanya waktu lalu lintas akan digeser. Di AWS FIS API, nilainya adalah string dalam format ISO 8601. Misalnya, PT1 M mewakili satu menit. Di AWS FIS konsol, Anda memasukkan jumlah detik, menit, atau jam.
- availabilityZoneIdentifier— Lalu lintas bergerak menjauh dari AZ ini. Ini bisa berupa nama AZ (us-east-1a) atau ID AZ (use1-az1).
- managedResourceTypes— Jenis sumber daya dari mana lalu lintas akan digeser, dipisahkan dengan koma. Opsi yang memungkinkan adalah `ASG` (Grup Auto Scaling), `ALB` (Application Load Balancer)`NLB`, (Network Load Balancer), dan `EKS` (Amazon EKS).
- zonalAutoshiftStatus— `zonalAutoshiftStatus` Status sumber daya yang ingin Anda targetkan. Opsi yang memungkinkan adalah `ENABLEDDISABLED`, dan`ANY`. Nilai default-nya `ENABLED`.

Izin

- arc-zonal-shift:StartZonalShift
- arc-zonal-shift:GetManagedResource
- arc-zonal-shift:UpdateZonalShift
- arc-zonal-shift:CancelZonalShift
- arc-zonal-shift:ListManagedResources
- penskalaan otomatis: DescribeTags
- Tag: GetResources

Tunggu tindakan

AWS FIS mendukung tindakan tunggu berikut.

aws:fis:wait

Menjalankan aksi AWS FIS tunggu.

Parameter

- duration— Durasi, dari satu menit hingga 12 jam. Di AWS FIS API, nilainya adalah string dalam format ISO 8601. Misalnya, PT1 M mewakili satu menit. Di AWS FIS konsol, Anda memasukkan jumlah detik, menit, atau jam.

Izin

- Tidak ada

CloudWatch Tindakan Amazon

AWS FIS mendukung CloudWatch tindakan Amazon berikut.

aws:cloudwatch:assert-alarm-state

Memverifikasi bahwa alarm yang ditentukan berada di salah satu status alarm yang ditentukan.

Jenis sumber daya

- Tidak ada

Parameter

- alarmArns— ARNs Alarm, dipisahkan dengan koma. Anda dapat menentukan hingga lima alarm.
- alarmStates— Alarm menyatakan, dipisahkan oleh koma. Status alarm yang mungkin adalah`OK`,`ALARM`, dan`INSUFFICIENT_DATA`.

Izin

- `cloudwatch:DescribeAlarms`

Tindakan Amazon DynamoDB

AWS FIS mendukung tindakan Amazon DynamoDB berikut.

aws:dynamodb:global-table-pause-replication

Menjeda replikasi tabel global Amazon DynamoDB ke tabel replika apa pun. Tabel dapat terus direplikasi hingga 5 menit setelah tindakan dimulai.

Pernyataan berikut akan ditambahkan secara dinamis ke kebijakan untuk tabel global DynamoDB target:

```
{
   "Statement":[
      {
         "Sid": "DoNotModifyFisDynamoDbPauseReplicationEXPxxxxxxxxxxxxxx",
         "Effect":"Deny",
         "Principal":{
            "AWS":"arn:aws:iam::123456789012:role/aws-service-role/
replication.dynamodb.amazonaws.com/AWSServiceRoleForDynamoDBReplication"
         },
         "Action":[
            "dynamodb:GetItem",
            "dynamodb:PutItem",
            "dynamodb:UpdateItem",
            "dynamodb:DeleteItem",
            "dynamodb:DescribeTable",
            "dynamodb:UpdateTable",
            "dynamodb:Scan",
            "dynamodb:DescribeTimeToLive",
            "dynamodb:UpdateTimeToLive"
         ],
         "Resource":"arn:aws:dynamodb:us-east-1:123456789012:table/ExampleGlobalTable",
         "Condition": {
            "DateLessThan": {
            "aws:CurrentTime": "2024-04-10T09:51:41.511Z"
         }
       }
      }
   ]
}
```

Pernyataan berikut akan ditambahkan secara dinamis ke kebijakan aliran untuk tabel global DynamoDB target:

```
{
   "Statement":[
      {
         "Sid": "DoNotModifyFisDynamoDbPauseReplicationEXPxxxxxxxxxxxxxx",
         "Effect":"Deny",
         "Principal":{
            "AWS":"arn:aws:iam::123456789012:role/aws-service-role/
replication.dynamodb.amazonaws.com/AWSServiceRoleForDynamoDBReplication"
         },
         "Action":[
            "dynamodb:GetRecords",
            "dynamodb:DescribeStream",
            "dynamodb:GetShardIterator"
         ],
         "Resource":"arn:aws:dynamodb:us-east-1:123456789012:table/ExampleGlobalTable/
stream/2023-08-31T09:50:24.025",
         "Condition": {
            "DateLessThan": {
            "aws:CurrentTime": "2024-04-10T09:51:41.511Z"
         }
      }
   ]
}
```

Jika tabel atau aliran target tidak memiliki kebijakan sumber daya terlampir, kebijakan sumber daya dibuat selama percobaan, dan secara otomatis dihapus saat eksperimen berakhir. Jika tidak, pernyataan kesalahan dimasukkan ke dalam kebijakan yang ada, tanpa modifikasi tambahan pada pernyataan kebijakan yang ada. Pernyataan kesalahan kemudian dihapus dari kebijakan di akhir percobaan.

Jenis sumber daya

- aws:dynamodb:global-table

Parameter

- duration— Di AWS FIS API, nilainya adalah string dalam format ISO 8601. Misalnya, PT1 M mewakili satu menit. Di AWS FIS konsol, Anda memasukkan jumlah detik, menit, atau jam.

Izin

- `dynamodb:PutResourcePolicy`
- `dynamodb:DeleteResourcePolicy`
- `dynamodb:GetResourcePolicy`
- `dynamodb:DescribeTable`
- `tag:GetResources`

Tindakan Amazon EBS

AWS FIS mendukung tindakan Amazon EBS berikut.

aws:ebs:pause-volume-io

Menjeda I/O operasi pada volume EBS target. Volume target harus berada di Availability Zone yang sama dan harus dilampirkan ke instance yang dibangun di Sistem Nitro. Volume tidak dapat dilampirkan ke instance di Outpost.

Untuk memulai eksperimen menggunakan EC2 konsol Amazon, lihat Pengujian kesalahan di Amazon EBS di EC2 Panduan Pengguna Amazon.

Jenis sumber daya

- aws:ec2:ebs-volume

Parameter

- duration— Durasi, dari satu detik hingga 12 jam. Di AWS FIS API, nilainya adalah string dalam format ISO 8601. Misalnya, PT1 M mewakili satu menit, PT5 S mewakili lima detik, dan PT6 H mewakili enam jam. Di AWS FIS konsol, Anda memasukkan jumlah detik, menit, atau jam. Jika durasinya kecil, seperti PT5 S, I/O dijeda selama durasi yang ditentukan, tetapi mungkin perlu waktu lebih lama untuk menyelesaikan eksperimen karena waktu yang diperlukan untuk menginisialisasi eksperimen.

Izin

- `ec2:DescribeVolumes`

- `ec2:PauseVolumeIO`
- `tag:GetResources`

EC2 Tindakan Amazon

AWS FIS mendukung EC2 tindakan Amazon berikut.

Tindakan

- aws:ec2:api-insufficient-instance-capacity-error
- aws:ec2:asg-insufficient-instance-capacity-error
- aws:ec2:reboot-instances
- aws:ec2:send-spot-instance-interruptions
- aws:ec2:stop-instances
- aws:ec2:terminate-instances

AWS FIS juga mendukung tindakan injeksi kesalahan melalui Agen AWS Systems Manager SSM. Systems Manager menggunakan dokumen SSM yang mendefinisikan tindakan yang akan dilakukan pada EC2 instance. Anda dapat menggunakan dokumen Anda sendiri untuk menyuntikkan kesalahan khusus, atau Anda dapat menggunakan dokumen SSM yang telah dikonfigurasi sebelumnya. Untuk informasi selengkapnya, lihat the section called “Tindakan dokumen SSM”.

aws:ec2:api-insufficient-instance-capacity-error

Menyuntikkan respons `InsufficientInstanceCapacity` kesalahan pada permintaan yang dibuat oleh peran IAM target. Operasi yang didukung adalah RunInstances CreateCapacityReservation, StartInstances, CreateFleet panggilan. Permintaan yang menyertakan permintaan kapasitas di beberapa Availability Zone tidak didukung. Tindakan ini tidak mendukung penentuan target menggunakan tag sumber daya, filter, atau parameter.

Jenis sumber daya

- aws:iam:role

Parameter

- duration— Di AWS FIS API, nilainya adalah string dalam format ISO 8601. Misalnya, PT1 M mewakili satu menit. Di AWS FIS konsol, Anda memasukkan jumlah detik, menit, atau jam.
- availabilityZoneIdentifiers— Daftar Availability Zones yang dipisahkan koma. Mendukung nama Zona IDs (misalnya`"use1-az1, use1-az2"`) dan Zona (misalnya`"us-east-1a"`).
- percentage— Persentase (1-100) panggilan untuk menyuntikkan kesalahan ke dalam.

Izin

- `ec2:InjectApiError`dengan `ec2:FisActionId` nilai kunci kondisi disetel ke `aws:ec2:api-insufficient-instance-capacity-error` dan kunci `ec2:FisTargetArns` kondisi disetel ke peran IAM target.

Untuk contoh kebijakan, lihat Contoh: Gunakan tombol kondisi untuk `ec2:InjectApiError`.

aws:ec2:asg-insufficient-instance-capacity-error

Menyuntikkan respons `InsufficientInstanceCapacity` kesalahan pada permintaan yang dibuat oleh grup Auto Scaling target. Tindakan ini hanya mendukung grup Auto Scaling menggunakan template peluncuran. Untuk mempelajari lebih lanjut tentang kesalahan kapasitas instans yang tidak mencukupi, lihat panduan EC2 pengguna Amazon.

Jenis sumber daya

- aws:ec2:autoscaling-group

Parameter

- duration— Di AWS FIS API, nilainya adalah string dalam format ISO 8601. Misalnya, PT1 M mewakili satu menit. Di AWS FIS konsol, Anda memasukkan jumlah detik, menit, atau jam.
- availabilityZoneIdentifiers— Daftar Availability Zones yang dipisahkan koma. Mendukung nama Zona IDs (misalnya`"use1-az1, use1-az2"`) dan Zona (misalnya`"us-east-1a"`).
- percentage – Opsional. Persentase (1-100) permintaan peluncuran grup Auto Scaling target untuk menyuntikkan kesalahan. Secara default, nilainya adalah 100.

Izin

- `ec2:InjectApiError`dengan kunci kondisi ec2: FisActionId nilai disetel ke `aws:ec2:asg-insufficient-instance-capacity-error` dan kunci `ec2:FisTargetArns` kondisi diatur untuk menargetkan grup Auto Scaling.
- `autoscaling:DescribeAutoScalingGroups`

Untuk contoh kebijakan, lihat Contoh: Gunakan tombol kondisi untuk `ec2:InjectApiError`.

aws:ec2:reboot-instances

Menjalankan tindakan Amazon EC2 API RebootInstancespada EC2 instance target.

Jenis sumber daya

- aws:ec2:instance

Parameter

- Tidak ada

Izin

- `ec2:RebootInstances`
- `ec2:DescribeInstances`

AWS kebijakan terkelola

- AWSFaultInjectionSimulatorEC2Akses

aws:ec2:send-spot-instance-interruptions

Menginterupsi Instans Spot target. Mengirim pemberitahuan interupsi Instans Spot untuk menargetkan Instans Spot dua menit sebelum menyela. Waktu interupsi ditentukan oleh durationBeforeInterruptionparameter yang ditentukan. Dua menit setelah waktu interupsi, Instans Spot dihentikan atau dihentikan, tergantung pada perilaku interupsi mereka. Instans Spot yang dihentikan oleh AWS FIS tetap berhenti sampai Anda memulai ulang.

Segera setelah tindakan dimulai, instance target menerima rekomendasi penyeimbangan ulang EC2 instance. Jika Anda menentukan durationBeforeInterruption, mungkin ada penundaan antara rekomendasi penyeimbangan kembali dan pemberitahuan gangguan.

Untuk informasi selengkapnya, lihat the section called “Uji interupsi Instans Spot”. Sebagai alternatif, untuk memulai eksperimen menggunakan EC2 konsol Amazon, lihat Memulai interupsi Instans Spot di Panduan Pengguna Amazon EC2 .

Jenis sumber daya

- aws:ec2:spot-instance

Parameter

- durationBeforeInterruption— Waktu untuk menunggu sebelum menyela instance, dari 2 hingga 15 menit. Di AWS FIS API, nilainya adalah string dalam format ISO 8601. Misalnya, PT2 M mewakili dua menit. Di AWS FIS konsol, Anda memasukkan jumlah menit.

Izin

- `ec2:SendSpotInstanceInterruptions`
- `ec2:DescribeInstances`

AWS kebijakan terkelola

- AWSFaultInjectionSimulatorEC2Akses

aws:ec2:stop-instances

Menjalankan tindakan Amazon EC2 API StopInstancespada EC2 instance target.

Jenis sumber daya

- aws:ec2:instance

Parameter

- startInstancesAfterDuration – Opsional. Waktu untuk menunggu sebelum memulai instance, dari satu menit hingga 12 jam. Di AWS FIS API, nilainya adalah string dalam format ISO 8601. Misalnya, PT1 M mewakili satu menit. Di AWS FIS konsol, Anda memasukkan jumlah detik, menit, atau jam. Jika instance memiliki volume EBS terenkripsi, Anda harus memberikan AWS FIS izin ke kunci KMS yang digunakan untuk mengenkripsi volume, atau menambahkan peran eksperimen ke kebijakan kunci KMS.
- completeIfInstancesTerminated – Opsional. Jika benar, dan jika juga `startInstancesAfterDuration` benar, tindakan ini tidak akan gagal ketika EC2 instance yang ditargetkan telah dihentikan oleh permintaan terpisah di luar FIS dan tidak dapat dimulai ulang. Misalnya, grup Auto Scaling dapat menghentikan EC2 instance yang dihentikan di bawah kendali mereka sebelum tindakan ini selesai. Default-nya adalah salah.

Izin

- `ec2:StopInstances`
- `ec2:StartInstances`
- `ec2:DescribeInstances` – Opsional. Diperlukan dengan completeIfInstancesTerminated untuk memvalidasi status instance di akhir tindakan.
- `kms:CreateGrant` – Opsional. Diperlukan dengan startInstancesAfterDurasi untuk memulai ulang instance dengan volume terenkripsi.

AWS kebijakan terkelola

- AWSFaultInjectionSimulatorEC2Akses

aws:ec2:terminate-instances

Menjalankan tindakan Amazon EC2 API TerminateInstancespada EC2 instance target.

Jenis sumber daya

- aws:ec2:instance

Parameter

- Tidak ada

Izin

- `ec2:TerminateInstances`
- `ec2:DescribeInstances`

AWS kebijakan terkelola

- AWSFaultInjectionSimulatorEC2Akses

Tindakan Amazon ECS

AWS FIS mendukung tindakan Amazon ECS berikut.

Tindakan

- aws:ecs:drain-container-instances
- aws:ecs:stop-task
- aws:ecs:task-cpu-stress
- aws:ecs:task-io-stress
- aws:ecs:task-kill-process
- aws:ecs:task-network-blackhole-port
- aws:ecs:task-network-latency
- aws:ecs:task-network-packet-loss

aws:ecs:drain-container-instances

Menjalankan tindakan Amazon ECS API UpdateContainerInstancesStateuntuk menguras persentase yang ditentukan dari EC2 instans Amazon yang mendasari pada kluster target.

Jenis sumber daya

- aws:ecs:cluster

Parameter

- drainagePercentage— Persentase (1-100).
- duration— Durasi, dari satu menit hingga 12 jam. Di AWS FIS API, nilainya adalah string dalam format ISO 8601. Misalnya, PT1 M mewakili satu menit. Di AWS FIS konsol, Anda memasukkan jumlah detik, menit, atau jam.

Izin

- `ecs:DescribeClusters`
- `ecs:UpdateContainerInstancesState`
- `ecs:ListContainerInstances`
- `tag:GetResources`

AWS kebijakan terkelola

- AWSFaultInjectionSimulatorECSAccess

aws:ecs:stop-task

Menjalankan tindakan Amazon ECS API StopTaskuntuk menghentikan tugas target.

Jenis sumber daya

- aws:ecs:task

Parameter

- Tidak ada

Izin

- `ecs:DescribeTasks`
- `ecs:ListTasks`
- `ecs:StopTask`
- `tag:GetResources`

AWS kebijakan terkelola

- AWSFaultInjectionSimulatorECSAccess

aws:ecs:task-cpu-stress

Menjalankan stress CPU pada tugas target. Menggunakan dokumen SSM AWSFIS-Run-CPU-Stress. Tugas harus dikelola oleh AWS Systems Manager. Untuk informasi selengkapnya, lihat Tindakan tugas ECS.

Jenis sumber daya

- aws:ecs:task

Parameter

- duration— Durasi stress test, dalam format ISO 8601.
- percent – Opsional. Persentase beban target, dari 0 (tanpa beban) hingga 100 (beban penuh). Secara default, nilainya adalah 100.
- workers – Opsional. Jumlah stresor yang digunakan. Defaultnya adalah 0, yang menggunakan semua stresor.
- installDependencies – Opsional. Jika nilai ini`True`, Systems Manager menginstal dependensi yang diperlukan pada wadah sespan untuk agen SSM, jika belum diinstal. Nilai default-nya `True`. Ketergantungan adalahstress-ng.

Izin

- `ssm:SendCommand`
- `ssm:ListCommands`
- `ssm:CancelCommand`

aws:ecs:task-io-stress

Menjalankan I/O stress pada tugas target. Menggunakan dokumen SSM AWSFIS-Run-IO-Stress. Tugas harus dikelola oleh AWS Systems Manager. Untuk informasi selengkapnya, lihat Tindakan tugas ECS.

Jenis sumber daya

- aws:ecs:task

Parameter

- duration— Durasi stress test, dalam format ISO 8601.
- percent – Opsional. Persentase ruang kosong pada sistem file untuk digunakan selama stress test. Defaultnya adalah 80%.
- workers – Opsional. Jumlah pekerja. Pekerja melakukan campuran read/write operasi berurutan, acak, dan dipetakan memori, sinkronisasi paksa, dan penghapusan cache. Beberapa proses anak melakukan I/O operasi yang berbeda pada file yang sama. Default-nya adalah 1.
- installDependencies – Opsional. Jika nilai ini`True`, Systems Manager menginstal dependensi yang diperlukan pada wadah sespan untuk agen SSM, jika belum diinstal. Nilai default-nya `True`. Ketergantungan adalahstress-ng.

Izin

- `ssm:SendCommand`
- `ssm:ListCommands`
- `ssm:CancelCommand`

aws:ecs:task-kill-process

Menghentikan proses yang ditentukan dalam tugas, menggunakan killall perintah. Menggunakan dokumen SSM AWSFIS-Run-Kill-Process. Definisi tugas harus `pidMode` disetel `ketask`. Tugas harus dikelola oleh AWS Systems Manager. Untuk informasi selengkapnya, lihat Tindakan tugas ECS.

Jenis sumber daya

- aws:ecs:task

Parameter

- processName— Nama proses untuk berhenti.

- signal – Opsional. Sinyal untuk mengirim bersama dengan perintah. Nilai yang mungkin adalah `SIGTERM` (yang dapat dipilih penerima untuk diabaikan) dan `SIGKILL` (yang tidak dapat diabaikan). Nilai default-nya `SIGTERM`.
- installDependencies – Opsional. Jika nilai ini`True`, Systems Manager menginstal dependensi yang diperlukan pada wadah sespan untuk agen SSM, jika belum diinstal. Nilai default-nya `True`. Ketergantungan adalahkillall.

Izin

- `ssm:SendCommand`
- `ssm:ListCommands`
- `ssm:CancelCommand`

aws:ecs:task-network-blackhole-port

Menurunkan lalu lintas masuk atau keluar untuk protokol dan port yang ditentukan, menggunakan titik akhir Amazon ECS Fault Injection. Menggunakan dokumen SSM AWSFIS-Run-Network-Blackhole-Port-ECS. Definisi tugas harus `pidMode` disetel `ketask`. Tugas harus dikelola oleh AWS Systems Manager. Anda tidak dapat mengatur `networkMode` ke `bridge` dalam definisi tugas. Untuk informasi selengkapnya, lihat Tindakan tugas ECS.

Ketika `useEcsFaultInjectionEndpoints` diatur `kefalse`, kesalahan menggunakan `iptables` alat, dan menggunakan dokumen AWSFIS-Run-Network-Blackhole-Port SSM.

Jenis sumber daya

- aws:ecs:task

Parameter

- duration— Durasi tes, dalam format ISO 8601.
- port— Nomor port.
- trafficType— Jenis lalu lintas. Nilai yang mungkin adalah `ingress` dan `egress`.
- protocol – Opsional. Protokol. Nilai yang mungkin adalah `tcp` dan udp. Nilai default-nya `tcp`.

- installDependencies – Opsional. Jika nilai ini`True`, Systems Manager menginstal dependensi yang diperlukan pada wadah sespan untuk agen SSM, jika belum diinstal. Nilai default-nya `True`. Dependensi adalahatd,curl-minimal, dig dan. jq
- useEcsFaultInjectionEndpoints – Opsional. Jika disetel ke true, Amazon ECS Fault Injection APIs akan digunakan. Default-nya adalah salah.

Izin

- `ssm:SendCommand`
- `ssm:ListCommands`
- `ssm:CancelCommand`

aws:ecs:task-network-latency

Menambahkan latensi dan jitter ke antarmuka jaringan untuk lalu lintas keluar ke sumber tertentu, menggunakan titik akhir Amazon ECS Fault Injection. Menggunakan dokumen SSM AWSFIS-Run-Network-Latency-ECS. Definisi tugas harus `pidMode` disetel `ketask`. Tugas harus dikelola oleh AWS Systems Manager. Anda tidak dapat mengatur `networkMode` ke `bridge` dalam definisi tugas. Untuk informasi selengkapnya, lihat Tindakan tugas ECS.

Ketika `useEcsFaultInjectionEndpoints` diatur `kefalse`, kesalahan menggunakan `tc` alat, dan menggunakan dokumen SSM AWSFIS-Run-Network-Latency-Sources.

Jenis sumber daya

- aws:ecs:task

Parameter

- duration— Durasi tes, dalam format ISO 8601.
- delayMilliseconds – Opsional. Penundaan, dalam milidetik. Defaultnya adalah 200.
- jitterMilliseconds – Opsional. Jitter, dalam milidetik. Default-nya adalah 10.
- sources – Opsional. Sumber, dipisahkan dengan koma, tanpa spasi. Nilai yang mungkin adalah: IPv4 alamat, blok IPv4 CIDR, nama domain`DYNAMODB`, dan`S3`. Jika Anda menentukan `DYNAMODB` atau`S3`, ini hanya berlaku untuk titik akhir Regional di Wilayah saat ini. Defaultnya adalah 0.0.0.0/0, yang cocok dengan semua lalu lintas. IPv4

- installDependencies – Opsional. Jika nilai ini`True`, Systems Manager menginstal dependensi yang diperlukan pada wadah sespan untuk agen SSM, jika belum diinstal. Nilai default-nya `True`. Dependensi adalahatd,, curl-minimaldig, jq dan. lsof
- useEcsFaultInjectionEndpoints – Opsional. Jika disetel ke true, Amazon ECS Fault Injection APIs akan digunakan. Default-nya adalah salah.

Izin

- `ssm:SendCommand`
- `ssm:ListCommands`
- `ssm:CancelCommand`

aws:ecs:task-network-packet-loss

Menambahkan kehilangan paket ke antarmuka jaringan untuk lalu lintas keluar ke sumber tertentu, menggunakan titik akhir Amazon ECS Fault Injection. Menggunakan dokumen SSM AWSFIS-Run-Network-Packet-loss-ECS. Definisi tugas harus `pidMode` disetel `ketask`. Tugas harus dikelola oleh AWS Systems Manager. Anda tidak dapat mengatur `networkMode` ke `bridge` dalam definisi tugas. Untuk informasi selengkapnya, lihat Tindakan tugas ECS.

Ketika `useEcsFaultInjectionEndpoints` diatur `kefalse`, kesalahan menggunakan `tc` alat, dan menggunakan dokumen SSM AWSFIS-Run-Network-Packet-Loss-Sources.

Jenis sumber daya

- aws:ecs:task

Parameter

- duration— Durasi tes, dalam format ISO 8601.
- lossPercent – Opsional. Persentase kehilangan paket. Defaultnya adalah 7%.
- sources – Opsional. Sumber, dipisahkan dengan koma, tanpa spasi. Nilai yang mungkin adalah: IPv4 alamat, blok IPv4 CIDR, nama domain`DYNAMODB`, dan`S3`. Jika Anda menentukan `DYNAMODB` atau`S3`, ini hanya berlaku untuk titik akhir Regional di Wilayah saat ini. Defaultnya adalah 0.0.0.0/0, yang cocok dengan semua lalu lintas. IPv4

- installDependencies – Opsional. Jika nilai ini`True`, Systems Manager menginstal dependensi yang diperlukan pada wadah sespan untuk agen SSM, jika belum diinstal. Nilai default-nya `True`. Dependensi adalahatd,, curl-minimaldig, jq dan. lsof
- useEcsFaultInjectionEndpoints – Opsional. Jika disetel ke true, Amazon ECS Fault Injection APIs akan digunakan. Default-nya adalah salah.

Izin

- `ssm:SendCommand`
- `ssm:ListCommands`
- `ssm:CancelCommand`

Tindakan Amazon EKS

AWS FIS mendukung tindakan Amazon EKS berikut.

Tindakan

- aws:eks:inject-kubernetes-custom-resource
- aws:eks:pod-cpu-stress
- aws:eks:pod-delete
- aws:eks:pod-io-stress
- aws:eks:pod-memory-stress
- aws:eks:pod-network-blackhole-port
- aws:eks:pod-network-latency
- aws:eks:pod-network-packet-loss
- aws:eks:terminate-nodegroup-instances

aws:eks:inject-kubernetes-custom-resource

Menjalankan percobaan ChaosMesh atau Lakmus pada satu cluster target. Anda harus menginstal ChaosMesh atau Lakmus pada cluster target.

Saat membuat templat eksperimen dan menentukan jenis target`aws:eks:cluster`, Anda harus menargetkan tindakan ini ke satu Nama Sumber Daya Amazon (ARN). Tindakan ini tidak mendukung penentuan target menggunakan tag sumber daya, filter, atau parameter.

Saat Anda menginstal ChaosMesh, Anda harus menentukan runtime kontainer yang sesuai. Dimulai dengan Amazon EKS versi 1.23, runtime default berubah dari Docker menjadi. containerd Dimulai dengan versi 1.24, Docker telah dihapus.

Jenis sumber daya

- aws:eks:cluster

Parameter

- kubernetesApiVersion— Versi API dari sumber daya kustom Kubernetes. Nilai yang mungkin adalah `chaos-mesh.org/v1alpha1` |`litmuschaos.io/v1alpha1`.
- kubernetesKind— Jenis sumber daya kustom Kubernetes. Nilai tergantung pada versi API.
  - `chaos-mesh.org/v1alpha1`— Nilai yang mungkin adalah `AWSChaos` `DNSChaos` | `GCPChaos` | `HTTPChaos` | `IOChaos` `JVMChaos` | `KernelChaos` | `NetworkChaos` | `PhysicalMachineChaos` | `PodChaos` | `PodHttpChaos` | `PodIOChaos` `PodNetworkChaos` | `Schedule` | `StressChaos` | `TimeChaos` |
  - `litmuschaos.io/v1alpha1`Nilai yang mungkin adalah`ChaosEngine`.
- kubernetesNamespace— Namespace Kubernetes.
- kubernetesSpec— `spec` Bagian dari sumber daya kustom Kubernetes, dalam format JSON.
- maxDuration— Waktu maksimum yang diizinkan untuk menyelesaikan eksekusi otomatisasi, dari satu menit hingga 12 jam. Di AWS FIS API, nilainya adalah string dalam format ISO 8601. Misalnya, PT1 M mewakili satu menit. Di AWS FIS konsol, Anda memasukkan jumlah detik, menit, atau jam.

Izin

Tidak diperlukan izin AWS Identity and Access Management (IAM) and Access Management (IAM) untuk tindakan ini. Izin yang diperlukan untuk menggunakan tindakan ini dikendalikan oleh Kubernetes menggunakan otorisasi RBAC. Untuk informasi selengkapnya, lihat Menggunakan Otorisasi RBAC di dokumentasi resmi Kubernetes. Untuk informasi lebih lanjut tentang Chaos Mesh, lihat dokumentasi resmi Chaos Mesh. Untuk informasi lebih lanjut tentang Lakmus, lihat dokumentasi resmi Lakmus.

aws:eks:pod-cpu-stress

Menjalankan stress CPU pada pod target. Untuk informasi selengkapnya, lihat Tindakan EKS Pod.

Jenis sumber daya

- aws:eks:pod

Parameter

- duration— Durasi stress test, dalam format ISO 8601.
- percent – Opsional. Persentase beban target, dari 0 (tanpa beban) hingga 100 (beban penuh). Secara default, nilainya adalah 100.
- workers – Opsional. Jumlah stresor yang digunakan. Defaultnya adalah 0, yang menggunakan semua stresor.
- kubernetesServiceAccount— Akun layanan Kubernetes. Untuk informasi tentang izin yang diperlukan, lihatthe section called “Konfigurasikan akun layanan Kubernetes”.
- fisPodContainerImage – Opsional. Gambar kontainer yang digunakan untuk membuat pod injektor kesalahan. Defaultnya adalah menggunakan gambar yang disediakan oleh AWS FIS. Untuk informasi selengkapnya, lihat the section called “Gambar kontainer pod”.
- maxErrorsPercent – Opsional. Persentase target yang bisa gagal sebelum injeksi kesalahan gagal. Default-nya adalah 0.
- fisPodLabels – Opsional. Label Kubernetes yang melekat pada pod orkestrasi kesalahan yang dibuat oleh FIS.
- fisPodAnnotations – Opsional. Anotasi Kubernetes yang dilampirkan pada pod orkestrasi kesalahan yang dibuat oleh FIS.
- fisPodSecurityPolicy – Opsional. Kebijakan Standar Keamanan Kubernetes digunakan untuk pod orkestrasi kesalahan yang dibuat oleh FIS dan container fana. Nilai yang mungkin adalah`privileged`, `baseline` dan`restricted`. Tindakan ini kompatibel dengan semua tingkat kebijakan.

Izin

- `eks:DescribeCluster`
- `ec2:DescribeSubnets`

- `tag:GetResources`

AWS kebijakan terkelola

- AWSFaultInjectionSimulatorEKSAccess

aws:eks:pod-delete

Menghapus pod target. Untuk informasi selengkapnya, lihat Tindakan EKS Pod.

Jenis sumber daya

- aws:eks:pod

Parameter

- gracePeriodSeconds – Opsional. Durasi, dalam hitungan detik, menunggu pod berakhir dengan anggun. Jika nilainya 0, kami segera melakukan tindakan. Jika nilainya nihil, kita menggunakan masa tenggang default untuk pod.
- kubernetesServiceAccount— Akun layanan Kubernetes. Untuk informasi tentang izin yang diperlukan, lihatthe section called “Konfigurasikan akun layanan Kubernetes”.
- fisPodContainerImage – Opsional. Gambar kontainer yang digunakan untuk membuat pod injektor kesalahan. Defaultnya adalah menggunakan gambar yang disediakan oleh AWS FIS. Untuk informasi selengkapnya, lihat the section called “Gambar kontainer pod”.
- maxErrorsPercent – Opsional. Persentase target yang bisa gagal sebelum injeksi kesalahan gagal. Default-nya adalah 0.
- fisPodLabels – Opsional. Label Kubernetes yang melekat pada pod orkestrasi kesalahan yang dibuat oleh FIS.
- fisPodAnnotations – Opsional. Anotasi Kubernetes yang dilampirkan pada pod orkestrasi kesalahan yang dibuat oleh FIS.
- fisPodSecurityPolicy – Opsional. Kebijakan Standar Keamanan Kubernetes digunakan untuk pod orkestrasi kesalahan yang dibuat oleh FIS dan container fana. Nilai yang mungkin adalah`privileged`, `baseline` dan`restricted`. Tindakan ini kompatibel dengan semua tingkat kebijakan.

Izin

- `eks:DescribeCluster`
- `ec2:DescribeSubnets`
- `tag:GetResources`

AWS kebijakan terkelola

- AWSFaultInjectionSimulatorEKSAccess

aws:eks:pod-io-stress

Menjalankan I/O stress pada pod target. Untuk informasi selengkapnya, lihat Tindakan EKS Pod.

Jenis sumber daya

- aws:eks:pod

Parameter

- duration— Durasi stress test, dalam format ISO 8601.
- workers – Opsional. Jumlah pekerja. Pekerja melakukan campuran read/write operasi berurutan, acak, dan dipetakan memori, sinkronisasi paksa, dan penghapusan cache. Beberapa proses anak melakukan I/O operasi yang berbeda pada file yang sama. Default-nya adalah 1.
- percent – Opsional. Persentase ruang kosong pada sistem file untuk digunakan selama stress test. Defaultnya adalah 80%.
- kubernetesServiceAccount— Akun layanan Kubernetes. Untuk informasi tentang izin yang diperlukan, lihatthe section called “Konfigurasikan akun layanan Kubernetes”.
- fisPodContainerImage – Opsional. Gambar kontainer yang digunakan untuk membuat pod injektor kesalahan. Defaultnya adalah menggunakan gambar yang disediakan oleh AWS FIS. Untuk informasi selengkapnya, lihat the section called “Gambar kontainer pod”.
- maxErrorsPercent – Opsional. Persentase target yang bisa gagal sebelum injeksi kesalahan gagal. Default-nya adalah 0.
- fisPodLabels – Opsional. Label Kubernetes yang melekat pada pod orkestrasi kesalahan yang dibuat oleh FIS.

- fisPodAnnotations – Opsional. Anotasi Kubernetes yang dilampirkan pada pod orkestrasi kesalahan yang dibuat oleh FIS.
- fisPodSecurityPolicy – Opsional. Kebijakan Standar Keamanan Kubernetes digunakan untuk pod orkestrasi kesalahan yang dibuat oleh FIS dan container fana. Nilai yang mungkin adalah`privileged`, `baseline` dan`restricted`. Tindakan ini kompatibel dengan semua tingkat kebijakan.

Izin

- `eks:DescribeCluster`
- `ec2:DescribeSubnets`
- `tag:GetResources`

AWS kebijakan terkelola

- AWSFaultInjectionSimulatorEKSAccess

aws:eks:pod-memory-stress

Menjalankan stress memori pada pod target. Untuk informasi selengkapnya, lihat Tindakan EKS Pod.

Jenis sumber daya

- aws:eks:pod

Parameter

- duration— Durasi stress test, dalam format ISO 8601.
- workers – Opsional. Jumlah stresor yang digunakan. Default-nya adalah 1.
- percent – Opsional. Persentase memori virtual yang digunakan selama stress test. Defaultnya adalah 80%.
- kubernetesServiceAccount— Akun layanan Kubernetes. Untuk informasi tentang izin yang diperlukan, lihatthe section called “Konfigurasikan akun layanan Kubernetes”.
- fisPodContainerImage – Opsional. Gambar kontainer yang digunakan untuk membuat pod injektor kesalahan. Defaultnya adalah menggunakan gambar yang disediakan oleh AWS FIS. Untuk informasi selengkapnya, lihat the section called “Gambar kontainer pod”.

- maxErrorsPercent – Opsional. Persentase target yang bisa gagal sebelum injeksi kesalahan gagal. Default-nya adalah 0.
- fisPodLabels – Opsional. Label Kubernetes yang melekat pada pod orkestrasi kesalahan yang dibuat oleh FIS.
- fisPodAnnotations – Opsional. Anotasi Kubernetes yang dilampirkan pada pod orkestrasi kesalahan yang dibuat oleh FIS.
- fisPodSecurityPolicy – Opsional. Kebijakan Standar Keamanan Kubernetes digunakan untuk pod orkestrasi kesalahan yang dibuat oleh FIS dan container fana. Nilai yang mungkin adalah`privileged`, `baseline` dan`restricted`. Tindakan ini kompatibel dengan semua tingkat kebijakan.

Izin

- `eks:DescribeCluster`
- `ec2:DescribeSubnets`
- `tag:GetResources`

AWS kebijakan terkelola

- AWSFaultInjectionSimulatorEKSAccess

aws:eks:pod-network-blackhole-port

Menurunkan lalu lintas masuk atau keluar untuk protokol dan port yang ditentukan. Hanya kompatibel dengan kebijakan Standar Keamanan Kubernetes. `privileged` Untuk informasi selengkapnya, lihat Tindakan EKS Pod.

Jenis sumber daya

- aws:eks:pod

Parameter

- duration— Durasi tes, dalam format ISO 8601.
- protocol- Protokol. Nilai yang mungkin adalah `tcp` dan `udp`.
- trafficType— Jenis lalu lintas. Nilai yang mungkin adalah `ingress` dan `egress`.

- port— Nomor port.
- kubernetesServiceAccount— Akun layanan Kubernetes. Untuk informasi tentang izin yang diperlukan, lihatthe section called “Konfigurasikan akun layanan Kubernetes”.
- fisPodContainerImage – Opsional. Gambar kontainer yang digunakan untuk membuat pod injektor kesalahan. Defaultnya adalah menggunakan gambar yang disediakan oleh AWS FIS. Untuk informasi selengkapnya, lihat the section called “Gambar kontainer pod”.
- maxErrorsPercent – Opsional. Persentase target yang bisa gagal sebelum injeksi kesalahan gagal. Default-nya adalah 0.
- fisPodLabels – Opsional. Label Kubernetes yang melekat pada pod orkestrasi kesalahan yang dibuat oleh FIS.
- fisPodAnnotations – Opsional. Anotasi Kubernetes yang dilampirkan pada pod orkestrasi kesalahan yang dibuat oleh FIS.

Izin

- `eks:DescribeCluster`
- `ec2:DescribeSubnets`
- `tag:GetResources`

AWS kebijakan terkelola

- AWSFaultInjectionSimulatorEKSAccess

aws:eks:pod-network-latency

Menambahkan latensi dan jitter ke antarmuka jaringan menggunakan tc alat untuk lalu lintas ke atau dari sumber tertentu. Hanya kompatibel dengan kebijakan Standar Keamanan Kubernetes. `privileged` Untuk informasi selengkapnya, lihat Tindakan EKS Pod.

Jenis sumber daya

- aws:eks:pod

Parameter

- duration— Durasi tes, dalam format ISO 8601.

- interface – Opsional. Antarmuka jaringan. Nilai default-nya `eth0`.
- delayMilliseconds – Opsional. Penundaan, dalam milidetik. Defaultnya adalah 200.
- jitterMilliseconds – Opsional. Jitter, dalam milidetik. Default-nya adalah 10.
- sources – Opsional. Sumber, dipisahkan dengan koma, tanpa spasi. Nilai yang mungkin adalah: IPv4 alamat, blok IPv4 CIDR, nama domain`DYNAMODB`, dan`S3`. Jika Anda menentukan `DYNAMODB` atau`S3`, ini hanya berlaku untuk titik akhir Regional di Wilayah saat ini. Defaultnya adalah 0.0.0.0/0, yang cocok dengan semua lalu lintas. IPv4
- kubernetesServiceAccount— Akun layanan Kubernetes. Untuk informasi tentang izin yang diperlukan, lihatthe section called “Konfigurasikan akun layanan Kubernetes”.
- fisPodContainerImage – Opsional. Gambar kontainer yang digunakan untuk membuat pod injektor kesalahan. Defaultnya adalah menggunakan gambar yang disediakan oleh AWS FIS. Untuk informasi selengkapnya, lihat the section called “Gambar kontainer pod”.
- maxErrorsPercent – Opsional. Persentase target yang bisa gagal sebelum injeksi kesalahan gagal. Default-nya adalah 0.
- fisPodLabels – Opsional. Label Kubernetes yang melekat pada pod orkestrasi kesalahan yang dibuat oleh FIS.
- fisPodAnnotations – Opsional. Anotasi Kubernetes yang dilampirkan pada pod orkestrasi kesalahan yang dibuat oleh FIS.

Izin

- `eks:DescribeCluster`
- `ec2:DescribeSubnets`
- `tag:GetResources`

AWS kebijakan terkelola

- AWSFaultInjectionSimulatorEKSAccess

aws:eks:pod-network-packet-loss

Menambahkan packet loss ke antarmuka jaringan menggunakan tc alat ini. Hanya kompatibel dengan kebijakan Standar Keamanan Kubernetes. `privileged` Untuk informasi selengkapnya, lihat Tindakan EKS Pod.

Jenis sumber daya

- aws:eks:pod

Parameter

- duration— Durasi tes, dalam format ISO 8601.
- interface – Opsional. Antarmuka jaringan. Nilai default-nya `eth0`.
- lossPercent – Opsional. Persentase kehilangan paket. Defaultnya adalah 7%.
- sources – Opsional. Sumber, dipisahkan dengan koma, tanpa spasi. Nilai yang mungkin adalah: IPv4 alamat, blok IPv4 CIDR, nama domain`DYNAMODB`, dan`S3`. Jika Anda menentukan `DYNAMODB` atau`S3`, ini hanya berlaku untuk titik akhir Regional di Wilayah saat ini. Defaultnya adalah 0.0.0.0/0, yang cocok dengan semua lalu lintas. IPv4
- kubernetesServiceAccount— Akun layanan Kubernetes. Untuk informasi tentang izin yang diperlukan, lihatthe section called “Konfigurasikan akun layanan Kubernetes”.
- fisPodContainerImage – Opsional. Gambar kontainer yang digunakan untuk membuat pod injektor kesalahan. Defaultnya adalah menggunakan gambar yang disediakan oleh AWS FIS. Untuk informasi selengkapnya, lihat the section called “Gambar kontainer pod”.
- maxErrorsPercent – Opsional. Persentase target yang bisa gagal sebelum injeksi kesalahan gagal. Default-nya adalah 0.
- fisPodLabels – Opsional. Label Kubernetes yang melekat pada pod orkestrasi kesalahan yang dibuat oleh FIS.
- fisPodAnnotations – Opsional. Anotasi Kubernetes yang dilampirkan pada pod orkestrasi kesalahan yang dibuat oleh FIS.

Izin

- `eks:DescribeCluster`
- `ec2:DescribeSubnets`
- `tag:GetResources`

AWS kebijakan terkelola

- AWSFaultInjectionSimulatorEKSAccess

aws:eks:terminate-nodegroup-instances

Menjalankan tindakan Amazon EC2 API TerminateInstancespada grup node target.

Jenis sumber daya

- aws:eks:nodegroup

Parameter

- instanceTerminationPercentage— Persentase (1-100) instance yang akan dihentikan.

Izin

- `ec2:DescribeInstances`
- `ec2:TerminateInstances`
- `eks:DescribeNodegroup`
- `tag:GetResources`

AWS kebijakan terkelola

- AWSFaultInjectionSimulatorEKSAccess

ElastiCache Tindakan Amazon

AWS FIS mendukung ElastiCache tindakan berikut.

aws:elasticache:replicationgroup-interrupt-az-power

Menginterupsi daya ke node di Availability Zone yang ditentukan untuk grup ElastiCache replikasi target dengan Multi-AZ diaktifkan. Hanya satu Availability Zone per grup replikasi yang dapat terpengaruh pada satu waktu. Ketika node primer ditargetkan, replika baca yang sesuai dengan lag replikasi paling sedikit dipromosikan ke primer. Penggantian replika baca di Availability Zone yang ditentukan diblokir selama durasi tindakan ini, yang berarti bahwa Grup Replikasi target beroperasi dengan kapasitas yang berkurang. Target untuk tindakan ini mendukung mesin Redis dan Valkey. Tindakan ini tidak mendukung opsi penerapan “tanpa server”.

Jenis sumber daya

- aws:elasticache:replicationgroup

Parameter

- duration— Durasi, dari satu menit hingga 12 jam. Di AWS FIS API, nilainya adalah string dalam format ISO 8601. Misalnya, PT1 M mewakili satu menit. Di AWS FIS konsol, Anda memasukkan jumlah detik, menit, atau jam.

Izin

- `elasticache:InterruptClusterAzPower`
- `elasticache:DescribeReplicationGroups`
- `tag:GetResources`

> **Note**
>
> Tindakan daya ElastiCache interupsi AZ sekarang mendukung semua jenis grup replikasi, termasuk Valkey dan Redis. Untuk lebih mewakili fungsi ini, tindakan telah diganti namanya. Jika saat ini Anda menggunakan`aws:elasticache:interrupt-cluster-az-power`, kami sarankan Anda bermigrasi ke tindakan baru `aws:elasticache:replicationgroup-interrupt-az-power` untuk memanfaatkan fitur terbaru.

AWS Lambda tindakan

AWS Lambda mendukung tindakan Lambda berikut

Tindakan

- aws:lambda:invocation-add-delay
- aws:lambda:invocation-error
- aws:lambda:invocation-http-integration-response

aws:lambda:invocation-add-delay

Penundaan memulai fungsi untuk sejumlah milidetik yang Anda tentukan. Efek dari tindakan ini mirip dengan Lambda cold start, tetapi waktu tambahan dihabiskan sebagai bagian dari durasi tagihan dan diterapkan ke semua lingkungan eksekusi daripada hanya memengaruhi lingkungan eksekusi baru. Ini berarti Anda mungkin mengalami awal dingin Lambda dan penundaan ini. Dengan menyetel nilai latensi yang lebih tinggi dari batas waktu yang dikonfigurasi pada fungsi Lambda, tindakan ini juga akan menyediakan akses ke peristiwa batas waktu fidelitas tinggi.

Jenis sumber daya

- aws:lambda:function

Parameter

- durasi — Lamanya waktu tindakan berlangsung. Di AWS FIS API, nilainya adalah string dalam format ISO 8601. Misalnya, PT1 M mewakili satu menit. Di AWS FIS konsol, Anda memasukkan jumlah detik, menit, atau jam.
- InvocationPercentage - Opsional. Persentase (1-100) pemanggilan fungsi untuk menyuntikkan kesalahan ke dalam. Secara default, nilainya adalah 100.
- startupDelayMilliseconds – Opsional. Jumlah waktu dalam milidetik (0-900.000) untuk menunggu antara pemanggilan dan eksekusi kode fungsi. Defaultnya adalah 1000.

Izin

- `s3:PutObject`
- `s3:DeleteObject`
- `lambda:GetFunction`
- `tag:GetResources`

aws:lambda:invocation-error

Menandai pemanggilan fungsi Lambda sebagai gagal. Tindakan ini berguna untuk menguji mekanisme penanganan kesalahan, seperti alarm dan konfigurasi coba lagi. Saat menggunakan

tindakan ini, Anda memilih apakah akan menjalankan kode fungsi atau tidak sebelum mengembalikan kesalahan.

Jenis sumber daya

- aws:lambda:function

Parameter

- durasi — Lamanya waktu tindakan berlangsung. Di AWS FIS API, nilainya adalah string dalam format ISO 8601. Misalnya, PT1 M mewakili satu menit. Di AWS FIS konsol, Anda memasukkan jumlah detik, menit, atau jam.
- InvocationPercentage - Opsional. Persentase (1-100) pemanggilan fungsi untuk menyuntikkan kesalahan ke dalam. Secara default, nilainya adalah 100.
- preventExecution - Jika nilainya benar, tindakan akan mengembalikan kesalahan tanpa menjalankan fungsi.

Izin

- `s3:PutObject`
- `s3:DeleteObject`
- `lambda:GetFunction`
- `tag:GetResources`

aws:lambda:invocation-http-integration-response

Memodifikasi perilaku fungsi. Anda memilih jenis konten dan kode respons HTTP untuk mendukung integrasi dengan ALB, API-GW dan VPC Lattice. Untuk mengaktifkan integrasi hulu atau hilir yang berdampak selektif, Anda dapat memilih apakah akan langsung mengembalikan respons yang dimodifikasi atau apakah akan menjalankan fungsi dan mengganti respons setelah fungsi selesai dieksekusi.

Jenis sumber daya

- aws:lambda:function

Parameter

- contentTypeHeader— Nilai string header tipe konten HTTP untuk kembali dari fungsi Lambda.
- durasi — Lamanya waktu tindakan berlangsung. Di AWS FIS API, nilainya adalah string dalam format ISO 8601. Misalnya, PT1 M mewakili satu menit. Di AWS FIS konsol, Anda memasukkan jumlah detik, menit, atau jam.
- InvocationPercentage - Opsional. Persentase (1-100) pemanggilan fungsi untuk menyuntikkan kesalahan ke dalam. Secara default, nilainya adalah 100.
- preventExecution - Jika nilainya benar, tindakan akan mengembalikan respons tanpa menjalankan fungsi.
- StatusCode — Nilai kode status HTTP (000-999) untuk kembali dari fungsi Lambda.

Izin

- `s3:PutObject`
- `s3:DeleteObject`
- `lambda:GetFunction`
- `tag:GetResources`

Tindakan jaringan

AWS FIS mendukung tindakan jaringan berikut.

Tindakan

- aws:network:disrupt-connectivity
- aws:network:route-table-disrupt-cross-region-connectivity
- aws:network:transit-gateway-disrupt-cross-region-connectivity

aws:network:disrupt-connectivity

Menyangkal lalu lintas yang ditentukan ke subnet target dengan sementara mengkloning daftar kontrol akses jaringan asli (ACL jaringan) yang terkait dengan subnet yang ditargetkan. FIS menambahkan aturan penolakan ke ACL jaringan kloning, yang memiliki tag managedByfis=True, dan mengaitkannya dengan subnet selama durasi tindakan. Pada penyelesaian tindakan, FIS menghapus ACL jaringan kloning dan mengembalikan asosiasi ACL jaringan asli.

Jenis sumber daya

- aws:ec2:subnet

Parameter

- scopeJenis lalu lintas yang harus ditolak. Ketika ruang lingkup tidak`all`, jumlah maksimum entri dalam jaringan ACLs adalah 20. Nilai yang mungkin adalah:
  - `all`— Menyangkal semua lalu lintas yang masuk dan keluar dari subnet. Perhatikan bahwa opsi ini memungkinkan lalu lintas intra-subnet, termasuk lalu lintas ke dan dari antarmuka jaringan di subnet.
  - `availability-zone`— Menolak lalu lintas intra-VPC ke dan dari subnet di Availability Zone lainnya. Jumlah maksimum subnet yang dapat ditargetkan dalam VPC adalah 30.
  - `dynamodb`— Menolak lalu lintas ke dan dari titik akhir Regional untuk DynamoDB di Wilayah saat ini.
  - `prefix-list`— Menolak lalu lintas ke dan dari daftar awalan yang ditentukan.
  - `s3`— Menolak lalu lintas ke dan dari titik akhir Regional untuk Amazon S3 di Wilayah saat ini.
  - `vpc`— Menolak lalu lintas yang masuk dan keluar dari VPC.
- duration— Durasi, dari satu menit hingga 12 jam. Di AWS FIS API, nilainya adalah string dalam format ISO 8601. Misalnya, PT1 M mewakili satu menit. Di AWS FIS konsol, Anda memasukkan jumlah detik, menit, atau jam.
- prefixListIdentifier— Jika ruang lingkupnya`prefix-list`, ini adalah pengidentifikasi daftar awalan yang dikelola pelanggan. Anda dapat menentukan nama, ID, atau ARN. Daftar awalan dapat memiliki paling banyak 10 entri.

Izin

- `ec2:CreateNetworkAcl`— Membuat ACL jaringan dengan tag ManagedByfis=True.
- `ec2:CreateNetworkAclEntry`— ACL jaringan harus memiliki tag ManagedByfis=True.
- `ec2:CreateTags`
- `ec2:DeleteNetworkAcl`— ACL jaringan harus memiliki tag ManagedByfis=True.
- `ec2:DescribeManagedPrefixLists`
- `ec2:DescribeNetworkAcls`
- `ec2:DescribeSubnets`

- `ec2:DescribeVpcs`
- `ec2:GetManagedPrefixListEntries`
- `ec2:ReplaceNetworkAclAssociation`

AWS kebijakan terkelola

- AWSFaultInjectionSimulatorNetworkAccess

aws:network:route-table-disrupt-cross-region-connectivity

Memblokir lalu lintas yang berasal dari subnet target dan ditujukan untuk Wilayah yang ditentukan. Membuat tabel rute yang mencakup semua rute untuk Wilayah untuk mengisolasi. Untuk mengizinkan FIS membuat tabel rute ini, naikkan kuota `routes per route table` VPC Amazon menjadi 250 ditambah jumlah rute di tabel rute yang ada.

Jenis sumber daya

- aws:ec2:subnet

Parameter

- `region`— Kode Wilayah untuk mengisolasi (misalnya, eu-west-1).
- `duration`- Lamanya waktu aksi berlangsung. Di AWS FIS API, nilainya adalah string dalam format ISO 8601. Misalnya, PT1 M mewakili satu menit. Di AWS FIS konsol, Anda memasukkan jumlah detik, menit, atau jam.

Izin

- `ec2:AssociateRouteTable`
- `ec2:CreateManagedPrefixList` †
- `ec2:CreateNetworkInterface` †
- `ec2:CreateRoute` †
- `ec2:CreateRouteTable` †
- `ec2:CreateTags` †
- `ec2:DeleteManagedPrefixList` †

- `ec2:DeleteNetworkInterface` †
- `ec2:DeleteRouteTable` †
- `ec2:DescribeManagedPrefixLists`
- `ec2:DescribeNetworkInterfaces`
- `ec2:DescribeRouteTables`
- `ec2:DescribeSubnets`
- `ec2:DescribeVpcPeeringConnections`
- `ec2:DescribeVpcs`
- `ec2:DisassociateRouteTable`
- `ec2:GetManagedPrefixListEntries`
- `ec2:ModifyManagedPrefixList` †
- `ec2:ModifyVpcEndpoint`
- `ec2:ReplaceRouteTableAssociation`

† Cakupan menggunakan tag. managedByFIS=true Anda tidak perlu mengelola tag ini. AWS FIS menambahkan dan menghapus tag ini selama percobaan.

AWS kebijakan terkelola

- AWSFaultInjectionSimulatorNetworkAccess

aws:network:transit-gateway-disrupt-cross-region-connectivity

Memblokir lalu lintas dari lampiran peering gateway transit target yang ditujukan untuk Wilayah yang ditentukan.

Jenis sumber daya

- aws:ec2:transit-gateway

Parameter

- `region`— Kode Wilayah untuk mengisolasi (misalnya, eu-west-1).

- `duration`- Lamanya waktu aksi berlangsung. Di AWS FIS API, nilainya adalah string dalam format ISO 8601. Misalnya, PT1 M mewakili satu menit. Di AWS FIS konsol, Anda memasukkan jumlah detik, menit, atau jam.

Izin

- `ec2:AssociateTransitGatewayRouteTable`
- `ec2:DescribeTransitGatewayAttachments`
- `ec2:DescribeTransitGatewayPeeringAttachments`
- `ec2:DescribeTransitGateways`
- `ec2:DisassociateTransitGatewayRouteTable`

AWS kebijakan terkelola

- AWSFaultInjectionSimulatorNetworkAccess

Tindakan Amazon RDS

AWS FIS mendukung tindakan Amazon RDS berikut.

Tindakan

- aws:rds:failover-db-cluster
- aws:rds:reboot-db-instances

aws:rds:failover-db-cluster

Menjalankan Failover aksi Amazon RDS API DBCluster pada cluster Aurora DB target.

Jenis sumber daya

- aws:rds:cluster

Parameter

- Tidak ada

Izin

- `rds:FailoverDBCluster`
- `rds:DescribeDBClusters`
- `tag:GetResources`

AWS kebijakan terkelola

- AWSFaultInjectionSimulatorRDSAccess

aws:rds:reboot-db-instances

Menjalankan aksi Amazon RDS API Reboot DBInstance pada instans DB target.

Jenis sumber daya

- aws:rds:db

Parameter

- forceFailover – Opsional. Jika nilainya true, dan jika instance Multi-AZ, memaksa failover dari satu Availability Zone ke Availability Zone lainnya. Default-nya adalah salah.

Izin

- `rds:RebootDBInstance`
- `rds:DescribeDBInstances`
- `tag:GetResources`

AWS kebijakan terkelola

- AWSFaultInjectionSimulatorRDSAccess

Tindakan Amazon S3

AWS FIS mendukung tindakan Amazon S3 berikut.

Tindakan

- aws:s3:bucket-pause-replication

aws:s3:bucket-pause-replication

Menjeda replikasi dari bucket sumber target ke bucket tujuan. Bucket tujuan dapat berada di Wilayah AWS yang berbeda atau dalam Wilayah yang sama dengan bucket sumber. Objek yang ada dapat terus direplikasi hingga satu jam setelah tindakan dimulai. Tindakan ini hanya mendukung penargetan berdasarkan tag. Untuk mempelajari lebih lanjut tentang Replikasi Amazon S3, lihat panduan pengguna Amazon S3.

Jenis sumber daya

- aws:s3:bucket

Parameter

- duration— Durasi, dari satu menit hingga 12 jam. Di AWS FIS API, nilainya adalah string dalam format ISO 8601. Misalnya, PT1 M mewakili satu menit. Di AWS FIS konsol, Anda memasukkan jumlah detik, menit, atau jam.
- region— Wilayah AWS tempat bucket tujuan berada.
- destinationBuckets – Opsional. Daftar bucket S3 tujuan yang dipisahkan koma.
- prefixes – Opsional. Daftar koma dipisahkan dari awalan kunci objek S3 dari filter aturan replikasi. Aturan replikasi bucket target dengan filter berdasarkan awalan (es) akan dijeda.

Izin

- `S3:PutReplicationConfiguration`dengan kunci kondisi `S3:IsReplicationPauseRequest` diatur ke `True`
- `S3:GetReplicationConfiguration`dengan kunci kondisi `S3:IsReplicationPauseRequest` diatur ke `True`
- `S3:PauseReplication`
- `S3:ListAllMyBuckets`
- `tag:GetResources`

Untuk contoh kebijakan, lihat Contoh: Gunakan tombol kondisi untuk `aws:s3:bucket-pause-replication`.

Tindakan Systems Manager

AWS FIS mendukung tindakan Systems Manager berikut.

Tindakan

- aws:ssm:send-command
- aws:ssm:start-automation-execution

aws:ssm:send-command

Menjalankan aksi Systems Manager API SendCommandpada EC2 instance target. Dokumen Systems Manager (dokumen SSM) mendefinisikan tindakan yang dilakukan Systems Manager pada instans Anda. Untuk informasi selengkapnya, lihat Gunakan aws:ssm:send-command tindakan.

Jenis sumber daya

- aws:ec2:instance

Parameter

- documentArn— Nama Sumber Daya Amazon (ARN) dari dokumen. Di konsol, parameter ini selesai untuk Anda jika Anda memilih nilai dari Jenis tindakan yang sesuai dengan salah satu dokumen AWS FIS SSM yang telah dikonfigurasi sebelumnya.
- documentVersion – Opsional. Versi dokumen. Jika kosong, versi default berjalan.
- documentParameters— Bersyarat. Parameter yang diperlukan dan opsional yang diterima dokumen. Formatnya adalah objek JSON dengan kunci yang merupakan string dan nilai yang berupa string atau array string.
- duration— Durasi, dari satu menit hingga 12 jam. Di AWS FIS API, nilainya adalah string dalam format ISO 8601. Misalnya, PT1 M mewakili satu menit. Di AWS FIS konsol, Anda memasukkan jumlah detik, menit, atau jam.

Izin

- `ssm:SendCommand`

- `ssm:ListCommands`
- `ssm:CancelCommand`

AWS kebijakan terkelola

- AWSFaultInjectionSimulatorEC2Akses

aws:ssm:start-automation-execution

Menjalankan aksi API Systems Manager StartAutomationExecution.

Jenis sumber daya

- Tidak ada

Parameter

- documentArn— Nama Sumber Daya Amazon (ARN) dari dokumen otomatisasi.
- documentVersion – Opsional. Versi dokumen. Jika kosong, versi default berjalan.
- documentParameters— Bersyarat. Parameter yang diperlukan dan opsional yang diterima dokumen. Formatnya adalah objek JSON dengan kunci yang merupakan string dan nilai yang berupa string atau array string.
- maxDuration— Waktu maksimum yang diizinkan untuk menyelesaikan eksekusi otomatisasi, dari satu menit hingga 12 jam. Di AWS FIS API, nilainya adalah string dalam format ISO 8601. Misalnya, PT1 M mewakili satu menit. Di AWS FIS konsol, Anda memasukkan jumlah detik, menit, atau jam.

Izin

- `ssm:GetAutomationExecution`
- `ssm:StartAutomationExecution`
- `ssm:StopAutomationExecution`
- `iam:PassRole` – Opsional. Diperlukan jika dokumen otomatisasi mengambil peran.

AWS kebijakan terkelola

- AWSFaultInjectionSimulatorSSMAccess

Menggunakan dokumen SSM Systems Manager dengan AWS FIS

AWS FIS mendukung jenis kesalahan kustom melalui Agen AWS Systems Manager SSM dan tindakan AWS FIS. aws:ssm:send-command Dokumen SSM Systems Manager yang telah dikonfigurasi sebelumnya (dokumen SSM) yang dapat digunakan untuk membuat tindakan injeksi kesalahan umum tersedia sebagai AWS dokumen publik yang dimulai dengan awalan -. AWSFIS

Agen SSM adalah perangkat lunak Amazon yang dapat diinstal dan dikonfigurasi di EC2 instans Amazon, server lokal, atau mesin virtual (). VMs Hal ini memungkinkan Systems Manager untuk mengelola sumber daya ini. Agen memproses permintaan dari Systems Manager, dan kemudian menjalankannya seperti yang ditentukan dalam permintaan. Anda dapat menyertakan dokumen SSM Anda sendiri untuk menyuntikkan kesalahan khusus, atau referensi salah satu dokumen milik Amazon publik.

Persyaratan

Untuk tindakan yang mengharuskan Agen SSM untuk menjalankan tindakan pada target, Anda harus memastikan hal-hal berikut:

- Agen dipasang pada target. Agen SSM diinstal secara default pada beberapa Amazon Machine Images (AMIs). Jika tidak, Anda dapat menginstal Agen SSM pada instans Anda. Untuk informasi selengkapnya, lihat Instal Agen SSM secara manual untuk EC2 instance di AWS Systems Manager Panduan Pengguna.
- Systems Manager memiliki izin untuk melakukan tindakan pada instans Anda. Anda memberikan akses menggunakan profil instans IAM. Untuk informasi selengkapnya, lihat Membuat profil instans IAM untuk Systems Manager dan Melampirkan profil instans IAM ke EC2 instance di AWS Systems Manager Panduan Pengguna.

Gunakan aws:ssm:send-command tindakan

Dokumen SSM menentukan tindakan yang dilakukan Systems Manager di instans terkelola Anda. Systems Manager menyertakan sejumlah dokumen yang telah dikonfigurasi sebelumnya, atau Anda dapat membuatnya sendiri. Untuk informasi selengkapnya tentang membuat dokumen SSM Anda

sendiri, lihat Membuat dokumen Systems Manager di Panduan AWS Systems Manager Pengguna. Untuk informasi selengkapnya tentang dokumen SSM secara umum, lihat AWS Systems Manager dokumen di Panduan AWS Systems Manager Pengguna.

AWS FIS menyediakan dokumen SSM pra-konfigurasi. Anda dapat melihat dokumen SSM yang telah dikonfigurasi sebelumnya di bawah Dokumen di AWS Systems Manager konsol: https://console.aws.amazon.com/systems-manager/ dokumen. Anda juga dapat memilih dari pilihan dokumen yang telah dikonfigurasi sebelumnya di konsol AWS FIS. Untuk informasi selengkapnya, lihat Dokumen SSM AWS FIS yang telah dikonfigurasi sebelumnya.

Untuk menggunakan dokumen SSM dalam eksperimen AWS FIS Anda, Anda dapat menggunakan tindakan tersebut. aws:ssm:send-command Tindakan ini mengambil dan menjalankan dokumen SSM yang ditentukan pada instance target Anda.

Saat Anda menggunakan `aws:ssm:send-command` tindakan dalam templat eksperimen, Anda harus menentukan parameter tambahan untuk tindakan tersebut, termasuk yang berikut ini:

- documentArn – Wajib. Nama Sumber Daya Amazon (ARN) dari dokumen SSM.
- documentParameters— Bersyarat. Parameter yang diperlukan dan opsional yang diterima dokumen SSM. Formatnya adalah objek JSON dengan kunci yang merupakan string dan nilai yang berupa string atau array string.
- documentVersion – Opsional. Versi dokumen SSM untuk dijalankan.

Anda dapat melihat informasi untuk dokumen SSM (termasuk parameter untuk dokumen) dengan menggunakan konsol Systems Manager atau baris perintah.

Untuk melihat informasi tentang dokumen SSM menggunakan konsol

1. Buka AWS Systems Manager konsol di https://console.aws.amazon.com/systems-manager/.
2. Di panel navigasi, pilih Dokumen.
3. Pilih dokumen, dan pilih tab Detail.

Untuk melihat informasi tentang dokumen SSM menggunakan baris perintah

Gunakan perintah deskripsi-dokumen SSM.

Dokumen SSM AWS FIS yang telah dikonfigurasi sebelumnya

Anda dapat menggunakan dokumen SSM AWS FIS yang telah dikonfigurasi sebelumnya dengan `aws:ssm:send-command` tindakan di templat eksperimen Anda.

Persyaratan

- Dokumen SSM pra-konfigurasi yang disediakan oleh AWS FIS hanya didukung pada sistem operasi berikut:
  - Amazon Linux 2023, Amazon Linux 2
  - Ubuntu
  - RHEL 8, 9
  - CentOS 9
- Dokumen SSM pra-konfigurasi yang disediakan oleh AWS FIS hanya didukung pada instance. EC2 Mereka tidak didukung pada jenis node terkelola lainnya, seperti server lokal.

Untuk menggunakan dokumen SSM ini dalam eksperimen pada tugas ECS, gunakan yang sesuai. the section called “Tindakan Amazon ECS” Misalnya, aws:ecs:task-cpu-stress tindakan menggunakan AWSFIS-Run-CPU-Stress dokumen.

Dokumen

- AWSFIS-Run-CPU-Stress
- AWSFIS-Run-Disk-Fill
- AWSFIS-Run-IO-Stress
- AWSFIS-Run-Kill-Process
- AWSFIS-Run-Memory-Stress
- AWSFIS-Run-Network-Blackhole-Port
- AWSFIS-Run-Network-Latency
- AWSFIS-Run-Network-Latency-Sources
- AWSFIS-Run-Network-Packet-Loss
- AWSFIS-Run-Network-Packet-Loss-Sources

Perbedaan antara durasi tindakan dan DurationSeconds dalam dokumen SSM AWS FIS

Beberapa dokumen SSM membatasi waktu eksekusi mereka sendiri, misalnya DurationSeconds parameter digunakan oleh beberapa dokumen AWS FIS SSM yang telah dikonfigurasi sebelumnya. Akibatnya, Anda perlu menentukan dua durasi independen dalam definisi tindakan AWS FIS:

- Action duration: Untuk eksperimen dengan satu tindakan, durasi aksi setara dengan durasi percobaan. Dengan beberapa tindakan, durasi eksperimen bergantung pada durasi tindakan individu dan urutan pelaksanaannya. AWS FIS memonitor setiap tindakan hingga durasi aksinya berlalu.
- Parameter dokumenDurationSeconds: Durasi, ditentukan dalam detik, di mana dokumen SSM akan dijalankan.

Anda dapat memilih nilai yang berbeda untuk dua jenis durasi:

- Action duration exceeds DurationSeconds: Eksekusi dokumen SSM selesai sebelum tindakan selesai. AWS FIS menunggu sampai durasi tindakan berlalu sebelum tindakan selanjutnya dimulai.
- Action duration is shorter than DurationSeconds: Dokumen SSM melanjutkan eksekusi setelah tindakan selesai. Jika eksekusi dokumen SSM masih dalam proses dan durasi tindakan telah berlalu maka status tindakan diatur ke Selesai. AWS FIS hanya memonitor eksekusi sampai durasi tindakan berlalu.

Perhatikan bahwa beberapa dokumen SSM memiliki durasi variabel. Misalnya dokumen AWS FIS SSM memiliki opsi untuk menginstal prasyarat, yang dapat memperpanjang durasi eksekusi keseluruhan di luar parameter yang ditentukan. DurationSeconds Jadi, jika Anda menetapkan durasi tindakan dan DurationSeconds nilai yang sama, ada kemungkinan bahwa skrip SSM dapat berjalan lebih lama dari durasi tindakan.

AWSFIS-Run-CPU-Stress

Menjalankan stress CPU pada instance menggunakan stress-ng alat ini. Menggunakan dokumen SSM AWSFIS-Run-CPU-Stress.

Jenis tindakan (hanya konsol)

aws:ssm:send-command/AWSFIS-Run-CPU-Stress

ARN

arn:aws:ssm:region::document/AWSFIS-Run-CPU-Stress

Parameter dokumen

- DurationSeconds – Wajib. Durasi tes stres CPU, dalam hitungan detik.
- CPU – Opsional. Jumlah stresor CPU yang digunakan. Defaultnya adalah 0, yang menggunakan semua stresor CPU.
- LoadPercent – Opsional. Target persentase beban CPU, dari 0 (tanpa beban) hingga 100 (beban penuh). Secara default, nilainya adalah 100.
- InstallDependencies – Opsional. Jika nilainya`True`, Systems Manager menginstal dependensi yang diperlukan pada instance target jika belum diinstal. Nilai default-nya `True`. Ketergantungan adalahstress-ng.

Berikut ini adalah contoh string yang dapat Anda masukkan di konsol.

```
{"DurationSeconds":"60", "InstallDependencies":"True"}
```

AWSFIS-Run-Disk-Fill

Mengalokasikan ruang disk pada volume root sebuah instance untuk mensimulasikan kesalahan penuh disk. Menggunakan dokumen SSM AWSFIS-Run-Disk-Fill.

Jika percobaan menyuntikkan kesalahan ini dihentikan, baik secara manual atau melalui kondisi berhenti, AWS FIS mencoba untuk memutar kembali dengan membatalkan dokumen SSM yang sedang berjalan. Namun, jika disk 100% penuh, baik karena kesalahan atau kesalahan ditambah aktivitas aplikasi, Systems Manager mungkin tidak dapat menyelesaikan operasi pembatalan. Oleh karena itu, jika Anda mungkin perlu menghentikan percobaan, pastikan disk tidak akan menjadi 100% penuh.

Jenis tindakan (hanya konsol)

aws:ssm:send-command/AWSFIS-Run-Disk-Fill

ARN

arn:aws:ssm:region::document/AWSFIS-Run-Disk-Fill

Parameter dokumen

- DurationSeconds – Wajib. Durasi tes pengisian disk, dalam hitungan detik.

- Percent – Opsional. Persentase disk yang dialokasikan selama tes pengisian disk. Defaultnya adalah 95%.
- InstallDependencies – Opsional. Jika nilainya`True`, Systems Manager menginstal dependensi yang diperlukan pada instance target jika belum diinstal. Nilai default-nya `True`. Dependensi adalahatd, kmod dan. fallocate

Berikut ini adalah contoh string yang dapat Anda masukkan di konsol.

```
{"DurationSeconds":"60", "InstallDependencies":"True"}
```

AWSFIS-Run-IO-Stress

Menjalankan stress IO pada instance menggunakan stress-ng alat. Menggunakan dokumen SSM AWSFIS-Run-IO-Stress.

Jenis tindakan (hanya konsol)

aws:ssm:send-command/AWSFIS-Run-IO-Stress

ARN

arn:aws:ssm:region::document/AWSFIS-Run-IO-Stress

Parameter dokumen

- DurationSeconds – Wajib. Durasi tes stress IO, dalam hitungan detik.
- Workers – Opsional. Jumlah pekerja yang melakukan campuran read/write operasi sekuensial, acak, dan dipetakan memori, sinkronisasi paksa, dan penghapusan cache. Beberapa proses anak melakukan I/O operasi yang berbeda pada file yang sama. Default-nya adalah 1.
- Percent – Opsional. Persentase ruang kosong pada sistem file yang digunakan selama IO stress test. Defaultnya adalah 80%.
- InstallDependencies – Opsional. Jika nilainya`True`, Systems Manager menginstal dependensi yang diperlukan pada instance target jika belum diinstal. Nilai default-nya `True`. Ketergantungan adalahstress-ng.

Berikut ini adalah contoh string yang dapat Anda masukkan di konsol.

```
{"Workers":"1", "Percent":"80", "DurationSeconds":"60", "InstallDependencies":"True"}
```

AWSFIS-Run-Kill-Process

Menghentikan proses yang ditentukan dalam contoh, menggunakan killall perintah. Menggunakan dokumen SSM AWSFIS-Run-Kill-Process.

Jenis tindakan (hanya konsol)

aws:ssm:send-command/AWSFIS-Run-Kill-Process

ARN

arn:aws:ssm:region::document/AWSFIS-Run-Kill-Process

Parameter dokumen

- ProcessName – Wajib. Nama proses untuk berhenti.
- Signal – Opsional. Sinyal untuk mengirim bersama dengan perintah. Nilai yang mungkin adalah `SIGTERM` (yang dapat dipilih penerima untuk diabaikan) dan `SIGKILL` (yang tidak dapat diabaikan). Nilai default-nya `SIGTERM`.
- InstallDependencies – Opsional. Jika nilainya`True`, Systems Manager menginstal dependensi yang diperlukan pada instance target jika belum diinstal. Nilai default-nya `True`. Ketergantungan adalahkillall.

Berikut ini adalah contoh string yang dapat Anda masukkan di konsol.

```
{"ProcessName":"myapplication", "Signal":"SIGTERM"}
```

AWSFIS-Run-Memory-Stress

Menjalankan stress memori pada instance menggunakan stress-ng alat. Menggunakan dokumen SSM AWSFIS-Run-Memory-Stress.

Jenis tindakan (hanya konsol)

aws:ssm:send-command/AWSFIS-Run-Memory-Stress

ARN

arn:aws:ssm:region::document/AWSFIS-Run-Memory-Stress

Parameter dokumen

- DurationSeconds – Wajib. Durasi tes stres memori, dalam hitungan detik.
- Workers – Opsional. Jumlah stresor memori virtual. Default-nya adalah 1.
- Percent – Wajib. Persentase memori virtual yang digunakan selama tes stress memori.
- InstallDependencies – Opsional. Jika nilainya`True`, Systems Manager menginstal dependensi yang diperlukan pada instance target jika belum diinstal. Nilai default-nya `True`. Ketergantungan adalahstress-ng.

Berikut ini adalah contoh string yang dapat Anda masukkan di konsol.

```
{"Percent":"80", "DurationSeconds":"60", "InstallDependencies":"True"}
```

AWSFIS-Run-Network-Blackhole-Port

Menurunkan lalu lintas masuk atau keluar untuk protokol dan port menggunakan alat. iptables Menggunakan dokumen SSM AWSFIS-Run-Network-Blackhole-Port.

Jenis tindakan (hanya konsol)

aws:ssm:send-command/AWSFIS-Run-Network-Blackhole-Port

ARN

arn:aws:ssm:region::document/AWSFIS-Run-Network-Blackhole-Port

Parameter dokumen

- Protocol – Wajib. Protokol. Nilai yang mungkin adalah `tcp` dan `udp`.
- Port – Wajib. Nomor port.
- TrafficType – Opsional. Jenis lalu lintas. Nilai yang mungkin adalah `ingress` dan `egress`. Nilai default-nya `ingress`.
- DurationSeconds – Wajib. Durasi tes blackhole jaringan, dalam hitungan detik.
- InstallDependencies – Opsional. Jika nilainya`True`, Systems Manager menginstal dependensi yang diperlukan pada instance target jika belum diinstal. Nilai default-nya `True`. Dependensi adalahatd,, diglsof, dan. iptables

Berikut ini adalah contoh string yang dapat Anda masukkan di konsol.

```
{"Protocol":"tcp", "Port":"8080", "TrafficType":"egress", "DurationSeconds":"60",
 "InstallDependencies":"True"}
```

AWSFIS-Run-Network-Latency

Menambahkan latensi ke antarmuka jaringan menggunakan tc alat ini. Menggunakan dokumen SSM AWSFIS-Run-Network-Latency.

Jenis tindakan (hanya konsol)

aws:ssm:send-command/AWSFIS-Run-Network-Latency

ARN

arn:aws:ssm:region::document/AWSFIS-Run-Network-Latency

Parameter dokumen

- Interface – Opsional. Antarmuka jaringan. Nilai default-nya `eth0`.
- DelayMilliseconds – Opsional. Penundaan, dalam milidetik. Defaultnya adalah 200.
- DurationSeconds – Wajib. Durasi uji latensi jaringan, dalam hitungan detik.
- InstallDependencies – Opsional. Jika nilainya`True`, Systems Manager menginstal dependensi yang diperlukan pada instance target jika belum diinstal. Nilai default-nya `True`. Dependensi adalahatd,dig, dan. tc

Berikut ini adalah contoh string yang dapat Anda masukkan di konsol.

```
{"DelayMilliseconds":"200", "Interface":"eth0", "DurationSeconds":"60",
 "InstallDependencies":"True"}
```

AWSFIS-Run-Network-Latency-Sources

Menambahkan latensi dan jitter ke antarmuka jaringan menggunakan tc alat untuk lalu lintas ke atau dari sumber tertentu. Menggunakan dokumen SSM AWSFIS-Run-Network-Latency-Sources.

Jenis tindakan (hanya konsol)

aws:ssm:send-command/AWSFIS-Run-Network-Latency-Sources

ARN

arn:aws:ssm:region::document/AWSFIS-Run-Network-Latency-Sources

Parameter dokumen

- Interface – Opsional. Antarmuka jaringan. Nilai default-nya `eth0`.
- DelayMilliseconds – Opsional. Penundaan, dalam milidetik. Defaultnya adalah 200.
- JitterMilliseconds – Opsional. Jitter, dalam milidetik. Default-nya adalah 10.
- Sources – Wajib. Sumber, dipisahkan dengan koma, tanpa spasi. Nilai yang mungkin adalah: IPv4 alamat, blok IPv4 CIDR, nama domain`DYNAMODB`, dan`S3`. Jika Anda menentukan `DYNAMODB` atau`S3`, ini hanya berlaku untuk titik akhir Regional di Wilayah saat ini.
- TrafficType – Opsional. Jenis lalu lintas. Nilai yang mungkin adalah `ingress` dan `egress`. Nilai default-nya `ingress`.
- DurationSeconds – Wajib. Durasi uji latensi jaringan, dalam hitungan detik.
- InstallDependencies – Opsional. Jika nilainya`True`, Systems Manager menginstal dependensi yang diperlukan pada instance target jika belum diinstal. Nilai default-nya `True`. Dependensi adalahatd,,dig, jqlsof, dan. tc

Berikut ini adalah contoh string yang dapat Anda masukkan di konsol.

```
{"DelayMilliseconds":"200", "JitterMilliseconds":"15",
 "Sources":"S3,www.example.com,72.21.198.67", "Interface":"eth0",
 "TrafficType":"egress", "DurationSeconds":"60", "InstallDependencies":"True"}
```

AWSFIS-Run-Network-Packet-Loss

Menambahkan packet loss ke antarmuka jaringan menggunakan tc alat ini. Menggunakan dokumen SSM AWSFIS-Run-Network-Packet-Loss.

Jenis tindakan (hanya konsol)

aws:ssm:send-command/AWSFIS-Run-Network-Packet-Loss

ARN

arn:aws:ssm:region::document/AWSFIS-Run-Network-Packet-Loss

Parameter dokumen

- Interface – Opsional. Antarmuka jaringan. Nilai default-nya `eth0`.

- LossPercent – Opsional. Persentase kehilangan paket. Defaultnya adalah 7%.
- DurationSeconds – Wajib. Durasi tes kehilangan paket jaringan, dalam hitungan detik.
- InstallDependencies – Opsional. Jika nilainya`True`, Systems Manager menginstal dependensi yang diperlukan pada instance target. Nilai default-nya `True`. Dependensi adalahatd,, lsofdig, dan. tc

Berikut ini adalah contoh string yang dapat Anda masukkan di konsol.

```
{"LossPercent":"15", "Interface":"eth0", "DurationSeconds":"60",
 "InstallDependencies":"True"}
```

AWSFIS-Run-Network-Packet-Loss-Sources

Menambahkan packet loss ke antarmuka jaringan menggunakan tc alat untuk lalu lintas ke atau dari sumber tertentu. Menggunakan dokumen SSM AWSFIS-Run-Network-Packet-Loss-Sources.

Jenis tindakan (hanya konsol)

aws:ssm:send-command/AWSFIS-Run-Network-Packet-Loss-Sources

ARN

arn:aws:ssm:region::document/AWSFIS-Run-Network-Packet-Loss-Sources

Parameter dokumen

- Interface – Opsional. Antarmuka jaringan. Nilai default-nya `eth0`.
- LossPercent – Opsional. Persentase kehilangan paket. Defaultnya adalah 7%.
- Sources – Wajib. Sumber, dipisahkan dengan koma, tanpa spasi. Nilai yang mungkin adalah: IPv4 alamat, blok IPv4 CIDR, nama domain`DYNAMODB`, dan`S3`. Jika Anda menentukan `DYNAMODB` atau`S3`, ini hanya berlaku untuk titik akhir Regional di Wilayah saat ini.
- TrafficType – Opsional. Jenis lalu lintas. Nilai yang mungkin adalah `ingress` dan `egress`. Nilai default-nya `ingress`.
- DurationSeconds – Wajib. Durasi tes kehilangan paket jaringan, dalam hitungan detik.
- InstallDependencies – Opsional. Jika nilainya`True`, Systems Manager menginstal dependensi yang diperlukan pada instance target. Nilai default-nya `True`. Dependensi adalahatd,,dig, jqlsof, dan. tc

Berikut ini adalah contoh string yang dapat Anda masukkan di konsol.

```
{"LossPercent":"15", "Sources":"S3,www.example.com,72.21.198.67", "Interface":"eth0",
 "TrafficType":"egress", "DurationSeconds":"60", "InstallDependencies":"True"}
```

Contoh

Untuk contoh templat eksperimen, lihatthe section called “Jalankan dokumen SSM AWS FIS yang telah dikonfigurasi sebelumnya”.

Untuk tutorial contoh, lihat Jalankan stress CPU pada sebuah instance.

Pemecahan Masalah

Gunakan prosedur berikut untuk memecahkan masalah.

Untuk memecahkan masalah dengan dokumen SSM

1. Buka AWS Systems Manager konsol di https://console.aws.amazon.com/systems-manager/.
2. Di panel navigasi, pilih Manajemen Node, Jalankan Perintah.
3. Pada tab Riwayat perintah, gunakan filter untuk menemukan proses dokumen.
4. Pilih ID perintah untuk membuka halaman detailnya.
5. Pilih ID instance. Tinjau output dan kesalahan untuk setiap langkah.

Gunakan tindakan AWS FIS aws:ecs:task

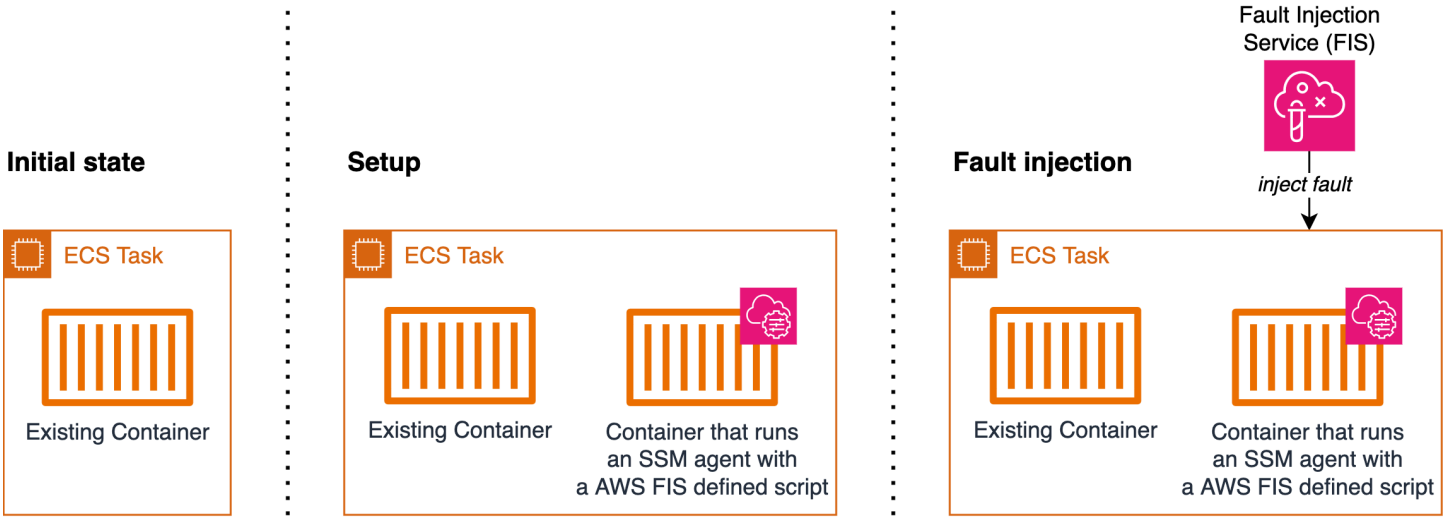
Anda dapat menggunakan tindakan aws:ecs:task untuk menyuntikkan kesalahan ke dalam tugas Amazon ECS Anda. Jenis kapasitas Amazon EC2 dan Fargate didukung.

Tindakan ini menggunakan dokumen AWS Systems Manager (SSM) untuk menyuntikkan kesalahan. Untuk menggunakan `aws:ecs:task` tindakan, Anda perlu menambahkan kontainer dengan Agen SSM ke definisi tugas Amazon Elastic Container Service (Amazon ECS) Container Service (Amazon ECS). Container menjalankan skrip yang AWS ditentukan FIS yang mendaftarkan tugas Amazon ECS sebagai Instans Terkelola dalam layanan SSM. Selain itu, skrip mengambil metadata tugas untuk menambahkan tag ke Instans Terkelola. Pengaturan akan memungkinkan AWS FIS untuk menyelesaikan tugas target. Paragraf ini mengacu pada Setup pada diagram di bawah ini.

Saat Anda menjalankan penargetan eksperimen AWS FIS`aws:ecs:task`, AWS FIS memetakan tugas Amazon ECS target yang Anda tentukan dalam templat eksperimen AWS FIS ke sekumpulan

instance terkelola SSM menggunakan tag sumber daya. `ECS_TASK_ARN` Nilai tag adalah ARN dari tugas Amazon ECS terkait di mana dokumen SSM harus dijalankan. Paragraf ini mengacu pada Injeksi Kesalahan pada diagram di bawah ini.

Diagram berikut mencontohkan pengaturan dan injeksi kesalahan pada tugas dengan satu wadah yang ada.



Tindakan

- the section called “aws:ecs:task-cpu-stress”
- the section called “aws:ecs:task-io-stress”
- the section called “aws:ecs:task-kill-process”
- the section called “aws:ecs:task-network-blackhole-port”
- the section called “aws:ecs:task-network-latency”
- the section called “aws:ecs:task-network-packet-loss”

Batasan

- Tindakan berikut tidak dapat berjalan secara paralel:
  - aws:ecs:task-network-blackhole-port
  - aws:ecs:task-network-latency
  - aws:ecs:task-network-packet-loss
- Jika Anda mengaktifkan Amazon ECS Exec, Anda harus menonaktifkannya sebelum dapat menggunakan tindakan ini.

- Eksekusi dokumen SSM mungkin memiliki Status Dibatalkan bahkan jika eksperimen memiliki Status Selesai. Saat menjalankan tindakan Amazon ECS, durasi yang disediakan pelanggan digunakan baik untuk durasi tindakan dalam eksperimen maupun durasi dokumen Amazon EC2 Systems Manager (SSM). Setelah tindakan dimulai, dibutuhkan beberapa waktu untuk dokumen SSM untuk mulai berjalan. Akibatnya, pada saat durasi tindakan yang ditentukan tercapai, dokumen SSM mungkin masih memiliki beberapa detik tersisa untuk menyelesaikan pelaksanaannya. Ketika durasi tindakan percobaan tercapai, tindakan dihentikan, dan eksekusi dokumen SSM dibatalkan. Injeksi kesalahan berhasil.

Persyaratan

- Tambahkan izin berikut ke peran eksperimen AWS FIS:
  - `ssm:SendCommand`
  - `ssm:ListCommands`
  - `ssm:CancelCommand`
- Tambahkan izin berikut ke peran IAM tugas Amazon ECS:
  - `ssm:CreateActivation`
  - `ssm:AddTagsToResource`
  - `iam:PassRole`

  Perhatikan bahwa Anda dapat menentukan ARN dari peran instans terkelola sebagai sumber daya untuk. `iam:PassRole`
- Buat peran IAM eksekusi tugas Amazon ECS dan tambahkan kebijakan ECSTask ExecutionRolePolicy terkelola Amazon.
- Dalam definisi tugas, atur variabel lingkungan `MANAGED_INSTANCE_ROLE_NAME` ke nama peran instance terkelola. Ini adalah peran yang akan dilampirkan pada tugas yang terdaftar sebagai instance terkelola di SSM.
- Tambahkan izin berikut ke peran instans terkelola:
  - `ssm:DeleteActivation`
  - `ssm:DeregisterManagedInstance`
- Tambahkan kebijakan SSMManaged InstanceCore terkelola Amazon ke peran instans terkelola.
- Tambahkan wadah agen SSM ke definisi tugas Amazon ECS. Skrip perintah mendaftarkan tugas Amazon ECS sebagai instance terkelola.

```
{
    "name": "amazon-ssm-agent",
    "image": "public.ecr.aws/amazon-ssm-agent/amazon-ssm-agent:latest",
    "cpu": 0,
    "links": [],
    "portMappings": [],
    "essential": false,
    "entryPoint": [],
    "command": [
        "/bin/bash",
        "-c",
        "set -e; dnf upgrade -y; dnf install jq procps awscli -y; term_handler()
 { echo \"Deleting SSM activation $ACTIVATION_ID\"; if ! aws ssm delete-
activation --activation-id $ACTIVATION_ID --region $ECS_TASK_REGION; then
 echo \"SSM activation $ACTIVATION_ID failed to be deleted\" 1>&2; fi;
 MANAGED_INSTANCE_ID=$(jq -e -r .ManagedInstanceID /var/lib/amazon/ssm/registration);
 echo \"Deregistering SSM Managed Instance $MANAGED_INSTANCE_ID\"; if ! aws
 ssm deregister-managed-instance --instance-id $MANAGED_INSTANCE_ID --region
 $ECS_TASK_REGION; then echo \"SSM Managed Instance $MANAGED_INSTANCE_ID
 failed to be deregistered\" 1>&2; fi; kill -SIGTERM $SSM_AGENT_PID; }; trap
 term_handler SIGTERM SIGINT; if [[ -z $MANAGED_INSTANCE_ROLE_NAME ]]; then
 echo \"Environment variable MANAGED_INSTANCE_ROLE_NAME not set, exiting\"
 1>&2; exit 1; fi; if ! ps ax | grep amazon-ssm-agent | grep -v grep > /dev/
null; then if [[ -n $ECS_CONTAINER_METADATA_URI_V4 ]] ; then echo \"Found ECS
 Container Metadata, running activation with metadata\"; TASK_METADATA=$(curl
 \"${ECS_CONTAINER_METADATA_URI_V4}/task\"); ECS_TASK_AVAILABILITY_ZONE=$(echo
 $TASK_METADATA | jq -e -r '.AvailabilityZone'); ECS_TASK_ARN=$(echo $TASK_METADATA
 | jq -e -r '.TaskARN'); ECS_TASK_REGION=$(echo $ECS_TASK_AVAILABILITY_ZONE | sed
 's/.$//'); ECS_TASK_AVAILABILITY_ZONE_REGEX='^(af|ap|ca|cn|eu|me|sa|us|us-gov)-
(central|north|(north(east|west))|south|south(east|west)|east|west)-[0-9]{1}[a-z]
{1}$'; if ! [[ $ECS_TASK_AVAILABILITY_ZONE =~ $ECS_TASK_AVAILABILITY_ZONE_REGEX ]];
 then echo \"Error extracting Availability Zone from ECS Container Metadata,
 exiting\" 1>&2; exit 1; fi; ECS_TASK_ARN_REGEX='^arn:(aws|aws-cn|aws-us-gov):ecs:
[a-z0-9-]+:[0-9]{12}:task/[a-zA-Z0-9_-]+/[a-zA-Z0-9]+$'; if ! [[ $ECS_TASK_ARN
 =~ $ECS_TASK_ARN_REGEX ]]; then echo \"Error extracting Task ARN from ECS
 Container Metadata, exiting\" 1>&2; exit 1; fi; CREATE_ACTIVATION_OUTPUT=
$(aws ssm create-activation --iam-role $MANAGED_INSTANCE_ROLE_NAME --
tags Key=ECS_TASK_AVAILABILITY_ZONE,Value=$ECS_TASK_AVAILABILITY_ZONE
 Key=ECS_TASK_ARN,Value=$ECS_TASK_ARN Key=FAULT_INJECTION_SIDECAR,Value=true --
region $ECS_TASK_REGION); ACTIVATION_CODE=$(echo $CREATE_ACTIVATION_OUTPUT | jq
 -e -r .ActivationCode); ACTIVATION_ID=$(echo $CREATE_ACTIVATION_OUTPUT | jq -e
 -r .ActivationId); if ! amazon-ssm-agent -register -code $ACTIVATION_CODE -id
 $ACTIVATION_ID -region $ECS_TASK_REGION; then echo \"Failed to register with AWS
```

```
 Systems Manager (SSM), exiting\" 1>&2; exit 1; fi; amazon-ssm-agent & SSM_AGENT_PID=
$!; wait $SSM_AGENT_PID; else echo \"ECS Container Metadata not found, exiting\"
 1>&2; exit 1; fi; else echo \"SSM agent is already running, exiting\" 1>&2; exit 1;
 fi"
    ],
    "environment": [
        {
            "name": "MANAGED_INSTANCE_ROLE_NAME",
            "value": "SSMManagedInstanceRole"
        }
    ],
    "environmentFiles": [],
    "mountPoints": [],
    "volumesFrom": [],
    "secrets": [],
    "dnsServers": [],
    "dnsSearchDomains": [],
    "extraHosts": [],
    "dockerSecurityOptions": [],
    "dockerLabels": {},
    "ulimits": [],
    "logConfiguration": {},
    "systemControls": []
}
```

Untuk versi skrip yang lebih mudah dibaca, lihatthe section called “Versi referensi skrip”.

- Aktifkan Injeksi Kesalahan Amazon ECS APIs, dengan mengatur `enableFaultInjection` bidang dalam definisi tugas Amazon ECS:

```
"enableFaultInjection": true,
```

- Saat menggunakan`aws:ecs:task-network-blackhole-port`,`aws:ecs:task-network-latency`, atau `aws:ecs:task-network-packet-loss` tindakan pada tugas Fargate, tindakan harus memiliki `useEcsFaultInjectionEndpoints` parameter yang disetel ke. `true`
- Saat menggunakan`aws:ecs:task-kill-process`,, `aws:ecs:task-network-blackhole-port`aws:ecs:task-network-latency, dan `aws:ecs:task-network-packet-loss` tindakan, definisi tugas Amazon ECS harus `pidMode` disetel ke`task`.
- Saat menggunakan`aws:ecs:task-network-blackhole-port`,`aws:ecs:task-network-latency`, dan `aws:ecs:task-network-packet-loss` tindakan, definisi tugas Amazon ECS harus `networkMode` disetel ke nilai selain`bridge`.

Versi referensi skrip

Berikut ini adalah versi skrip yang lebih mudah dibaca di bagian Persyaratan, untuk referensi Anda.

```
#!/usr/bin/env bash

# This is the activation script used to register ECS tasks as Managed Instances in SSM
# The script retrieves information form the ECS task metadata endpoint to add three
 tags to the Managed Instance
#  - ECS_TASK_AVAILABILITY_ZONE: To allow customers to target Managed Instances / Tasks
 in a specific Availability Zone
#  - ECS_TASK_ARN: To allow customers to target Managed Instances / Tasks by using the
 Task ARN
#  - FAULT_INJECTION_SIDECAR: To make it clear that the tasks were registered as
 managed instance for fault injection purposes. Value is always 'true'.
# The script will leave the SSM Agent running in the background
# When the container running this script receives a SIGTERM or SIGINT signal, it will
 do the following cleanup:
#  - Delete SSM activation
#  - Deregister SSM managed instance

set -e # stop execution instantly as a query exits while having a non-zero

dnf upgrade -y
dnf install jq procps awscli -y

term_handler() {
  echo "Deleting SSM activation $ACTIVATION_ID"
  if ! aws ssm delete-activation --activation-id $ACTIVATION_ID --region
 $ECS_TASK_REGION; then
    echo "SSM activation $ACTIVATION_ID failed to be deleted" 1>&2
  fi

  MANAGED_INSTANCE_ID=$(jq -e -r .ManagedInstanceID /var/lib/amazon/ssm/registration)
  echo "Deregistering SSM Managed Instance $MANAGED_INSTANCE_ID"
  if ! aws ssm deregister-managed-instance --instance-id $MANAGED_INSTANCE_ID --region
 $ECS_TASK_REGION; then
    echo "SSM Managed Instance $MANAGED_INSTANCE_ID failed to be deregistered" 1>&2
  fi

  kill -SIGTERM $SSM_AGENT_PID
}
trap term_handler SIGTERM SIGINT
```

```
# check if the required IAM role is provided
if [[ -z $MANAGED_INSTANCE_ROLE_NAME ]] ; then
  echo "Environment variable MANAGED_INSTANCE_ROLE_NAME not set, exiting" 1>&2
  exit 1
fi

# check if the agent is already running (it will be if ECS Exec is enabled)
if ! ps ax | grep amazon-ssm-agent | grep -v grep > /dev/null; then

  # check if ECS Container Metadata is available
  if [[ -n $ECS_CONTAINER_METADATA_URI_V4 ]] ; then

    # Retrieve info from ECS task metadata endpoint
    echo "Found ECS Container Metadata, running activation with metadata"
    TASK_METADATA=$(curl "${ECS_CONTAINER_METADATA_URI_V4}/task")
    ECS_TASK_AVAILABILITY_ZONE=$(echo $TASK_METADATA | jq -e -r '.AvailabilityZone')
    ECS_TASK_ARN=$(echo $TASK_METADATA | jq -e -r '.TaskARN')
    ECS_TASK_REGION=$(echo $ECS_TASK_AVAILABILITY_ZONE | sed 's/.$//')

    # validate ECS_TASK_AVAILABILITY_ZONE
    ECS_TASK_AVAILABILITY_ZONE_REGEX='^(af|ap|ca|cn|eu|me|sa|us|us-gov)-(central|north|
(north(east|west))|south|south(east|west)|east|west)-[0-9]{1}[a-z]{1}$'
    if ! [[ $ECS_TASK_AVAILABILITY_ZONE =~ $ECS_TASK_AVAILABILITY_ZONE_REGEX ]] ; then
      echo "Error extracting Availability Zone from ECS Container Metadata, exiting"
 1>&2
      exit 1
    fi

    # validate ECS_TASK_ARN
    ECS_TASK_ARN_REGEX='^arn:(aws|aws-cn|aws-us-gov):ecs:[a-z0-9-]+:[0-9]{12}:task/[a-
zA-Z0-9_-]+/[a-zA-Z0-9]+$'
    if ! [[ $ECS_TASK_ARN =~ $ECS_TASK_ARN_REGEX ]] ; then
      echo "Error extracting Task ARN from ECS Container Metadata, exiting" 1>&2
      exit 1
    fi

    # Create activation tagging with Availability Zone and Task ARN
    CREATE_ACTIVATION_OUTPUT=$(aws ssm create-activation \
      --iam-role $MANAGED_INSTANCE_ROLE_NAME \
      --tags Key=ECS_TASK_AVAILABILITY_ZONE,Value=$ECS_TASK_AVAILABILITY_ZONE
 Key=ECS_TASK_ARN,Value=$ECS_TASK_ARN Key=FAULT_INJECTION_SIDECAR,Value=true \
      --region $ECS_TASK_REGION)
```

```
    ACTIVATION_CODE=$(echo $CREATE_ACTIVATION_OUTPUT | jq -e -r .ActivationCode)
    ACTIVATION_ID=$(echo $CREATE_ACTIVATION_OUTPUT | jq -e -r .ActivationId)

    # Register with AWS Systems Manager (SSM)
    if ! amazon-ssm-agent -register -code $ACTIVATION_CODE -id $ACTIVATION_ID -region
 $ECS_TASK_REGION; then
      echo "Failed to register with AWS Systems Manager (SSM), exiting" 1>&2
      exit 1
    fi

    # the agent needs to run in the background, otherwise the trapped signal
    # won't execute the attached function until this process finishes
    amazon-ssm-agent &
    SSM_AGENT_PID=$!

    # need to keep the script alive, otherwise the container will terminate
    wait $SSM_AGENT_PID

  else
    echo "ECS Container Metadata not found, exiting" 1>&2
    exit 1
  fi

else
  echo "SSM agent is already running, exiting" 1>&2
  exit 1
fi
```

Contoh template percobaan

Berikut ini adalah contoh template eksperimen untuk the section called “aws:ecs:task-cpu-stress” tindakan tersebut.

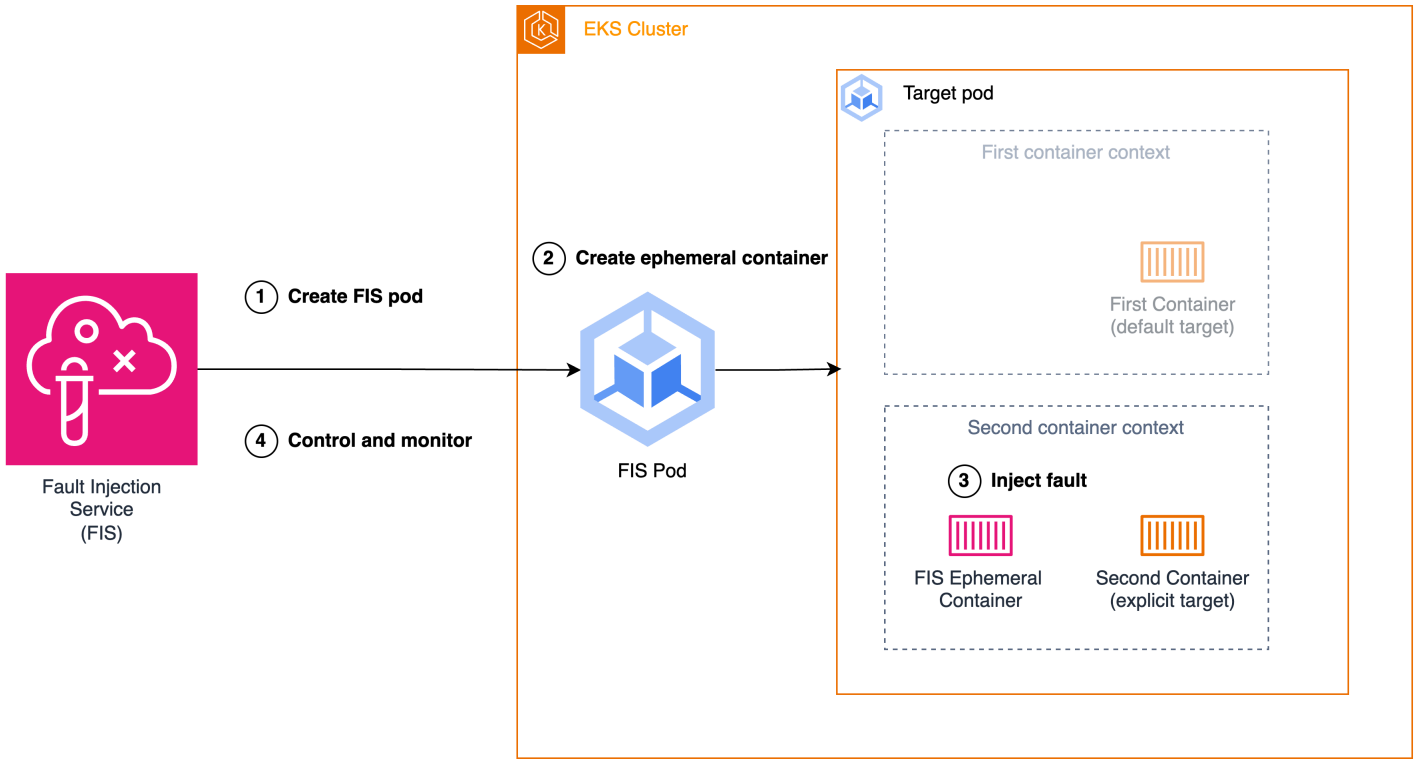
```
{
    "description": "Run CPU stress on the target ECS tasks",
    "targets": {
        "myTasks": {
            "resourceType": "aws:ecs:task",
            "resourceArns": [
                "arn:aws:ecs:us-east-1:111122223333:task/my-
cluster/09821742c0e24250b187dfed8EXAMPLE"
            ],
            "selectionMode": "ALL"
```

```
        }
    },
    "actions": {
        "EcsTask-cpu-stress": {
            "actionId": "aws:ecs:task-cpu-stress",
            "parameters": {
                "duration": "PT1M"
            },
            "targets": {
                "Tasks": "myTasks"
            }
        }
    },
    "stopConditions": [
        {
            "source": "none",
        }
    ],
    "roleArn": "arn:aws:iam::111122223333:role/fis-experiment-role",
    "tags": {}
}
```

Gunakan tindakan AWS FIS aws:eks:pod

Kamu dapat menggunakan tindakan aws:eks:pod untuk menyuntikkan kesalahan ke dalam Pod Kubernetes yang berjalan di kluster EKS kamu.

Ketika sebuah tindakan dimulai, FIS mengambil gambar kontainer FIS Pod. Gambar ini kemudian digunakan untuk membuat Pod di cluster EKS yang ditargetkan. Pod yang baru dibuat bertanggung jawab untuk menyuntikkan, mengendalikan, dan memantau kesalahan. Untuk semua action FIS EKS, kecuali aws:eks:pod-delete, injeksi kesalahan dicapai melalui penggunaan kontainer ephemeral, sebuah fitur Kubernetes yang memungkinkan pembuatan kontainer temporer di dalam Pod yang sudah ada. Wadah fana dimulai di namespace yang sama dengan wadah target dan menjalankan tugas injeksi kesalahan yang diinginkan. Jika tidak ada kontainer target yang ditentukan, kontainer pertama dalam spesifikasi Pod dipilih sebagai target.



1. FIS membuat Pod FIS di cluster target yang ditentukan dalam template percobaan.
2. FIS Pod menciptakan sebuah kontainer sementara di Target Pod dalam namespace yang sama dengan kontainer target.
3. Wadah fana menyuntikkan kesalahan di namespace wadah target.
4. FIS Pod mengontrol dan memonitor injeksi kesalahan kontainer sementara dan kontrol FIS dan memonitor Pod FIS.

Setelah percobaan selesai atau jika terjadi kesalahan, kontainer fana dan Pod FIS dihapus.

Tindakan

- the section called “aws:eks:pod-cpu-stress”
- the section called “aws:eks:pod-delete”
- the section called “aws:eks:pod-io-stress”
- the section called “aws:eks:pod-memory-stress”
- the section called “aws:eks:pod-network-blackhole-port”
- the section called “aws:eks:pod-network-latency”

- the section called “aws:eks:pod-network-packet-loss”

Batasan

- Tindakan berikut tidak berfungsi dengan AWS Fargate:
  - aws:eks:pod-network-blackhole-port
  - aws:eks:pod-network-latency
  - aws:eks:pod-network-packet-loss
- Tindakan berikut tidak mendukung mode `bridge` jaringan:
  - aws:eks:pod-network-blackhole-port
  - aws:eks:pod-network-latency
  - aws:eks:pod-network-packet-loss
- Tindakan berikut memerlukan izin root dalam wadah sementara.
  - aws:eks:pod-network-blackhole-port
  - aws:eks:pod-network-latency
  - aws:eks:pod-network-packet-loss

  Container ephemeral akan mewarisi izinnya dari konteks keamanan Pod target. Jika Anda perlu menjalankan kontainer di Pod sebagai pengguna non-root, Anda dapat mengatur konteks keamanan terpisah untuk kontainer di Pod target.
- Anda tidak dapat mengidentifikasi target tipe aws:eks:pod di templat eksperimen menggunakan tag sumber daya atau sumber daya ARNs . Anda harus mengidentifikasi target menggunakan parameter sumber daya yang diperlukan.
- Actions aws:eks: pod-network-latency dan aws:eks: tidak pod-network-packet-loss boleh dijalankan secara paralel dan menargetkan Pod yang sama. Bergantung pada nilai `maxErrors` parameter yang Anda tentukan, tindakan dapat berakhir dengan keadaan selesai atau gagal:
  - Jika `maxErrorsPercent` 0 (default), tindakan akan berakhir dalam keadaan gagal.
  - Jika tidak, kegagalan akan menambah `maxErrorsPercent` anggaran. Jika jumlah suntikan yang gagal tidak mencapai yang disediakan`maxErrors`, tindakan akan berakhir dalam keadaan selesai.
  - Anda dapat mengidentifikasi kegagalan ini dari log kontainer sementara yang disuntikkan di Pod target. Ini akan gagal dengan`Exit Code: 16`.

- Action aws:eks: tidak pod-network-blackhole-port boleh dijalankan secara paralel dengan action lain yang menargetkan Pod yang sama dan menggunakan Pod yang sama. `trafficType` Tindakan paralel menggunakan jenis lalu lintas yang berbeda didukung.
- FIS hanya dapat memantau status injeksi kesalahan ketika `securityContext` Pod target disetel `kereadOnlyRootFilesystem: false`. Tanpa konfigurasi ini, semua tindakan EKS Pod akan gagal.

Persyaratan

- Instal AWS CLI di komputer Anda. Ini diperlukan hanya jika Anda akan menggunakan AWS CLI untuk membuat peran IAM. Untuk informasi selengkapnya, lihat Menginstal atau memperbarui AWS CLI.
- Instal kubectl di komputer Anda. Ini diperlukan hanya untuk berinteraksi dengan cluster EKS untuk mengkonfigurasi atau memantau aplikasi target. Untuk informasi lebih lanjut, lihat https://kubernetes. io/docs/tasks/tools/.
- Versi EKS minimum yang didukung adalah 1.23.

Buat peran eksperimen

Untuk menjalankan eksperimen, Anda perlu mengonfigurasi peran IAM untuk eksperimen. Untuk informasi selengkapnya, lihat the section called “Peran percobaan”. Izin yang diperlukan untuk peran ini bergantung pada tindakan yang Anda gunakan. Lihat tindakan AWS FIS yang menargetkan `aws:eks:pod` untuk menemukan izin yang diperlukan untuk tindakan Anda.

Konfigurasikan akun layanan Kubernetes

Konfigurasikan akun layanan Kubernetes untuk menjalankan eksperimen dengan target di namespace Kubernetes yang ditentukan. Dalam contoh berikut, akun layanan adalah *`myserviceaccount`* dan namespace adalah. *`default`* Perhatikan bahwa default ini adalah salah satu ruang nama Kubernetes standar.

Untuk mengonfigurasi akun layanan Kubernetes

1. Buat file bernama `rbac.yaml` dan tambahkan yang berikut ini.

```
kind: ServiceAccount
apiVersion: v1
```

```
metadata:
  namespace: default
  name: myserviceaccount

---
kind: Role
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  namespace: default
  name: role-experiments
rules:
- apiGroups: [""]
  resources: ["configmaps"]
  verbs: [ "get", "create", "patch", "delete"]
- apiGroups: [""]
  resources: ["pods"]
  verbs: ["create", "list", "get", "delete", "deletecollection"]
- apiGroups: [""]
  resources: ["pods/ephemeralcontainers"]
  verbs: ["update"]
- apiGroups: [""]
  resources: ["pods/exec"]
  verbs: ["create"]
- apiGroups: ["apps"]
  resources: ["deployments"]
  verbs: ["get"]

---
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: bind-role-experiments
  namespace: default
subjects:
- kind: ServiceAccount
  name: myserviceaccount
  namespace: default
- apiGroup: rbac.authorization.k8s.io
  kind: User
  name: fis-experiment
roleRef:
  kind: Role
  name: role-experiments
```

```
  apiGroup: rbac.authorization.k8s.io
```

2. Jalankan perintah berikut.

```
kubectl apply -f rbac.yaml
```

Berikan akses kepada pengguna dan peran IAM ke Kubernetes APIs

Ikuti langkah-langkah yang dijelaskan dalam Identitas IAM Asosiasi dengan Izin Kubernetes dalam dokumentasi. EKS

Opsi 1: Buat entri akses

Sebaiknya gunakan Access Entries dengan mengikuti langkah-langkah yang dijelaskan dalam Grant IAM akses pengguna ke Kubernetes dengan entri akses EKS.

```
aws eks create-access-entry \
                --principal-arn arn:aws:iam::123456789012:role/fis-experiment-role \
                --username fis-experiment \
                --cluster-name my-cluster
```

> **Important**
>
> Untuk memanfaatkan entri akses, mode otentikasi kluster EKS harus dikonfigurasi ke mode `API_AND_CONFIG_MAP` atau`API`.

Opsi 2: Tambahkan entri ke aws-auth ConfigMap

Anda juga dapat menggunakan perintah berikut untuk membuat pemetaan identitas. Untuk informasi selengkapnya, lihat Mengelola pengguna dan peran IAM dalam eksctl dokumentasi.

```
eksctl create iamidentitymapping \
                --arn arn:aws:iam::123456789012:role/fis-experiment-role \
                --username fis-experiment \
                --cluster my-cluster
```

> **Important**
>
> Memanfaatkan toolkit eksctl untuk mengonfigurasi pemetaan identitas akan menghasilkan pembuatan entri di dalam file. `aws-auth` ConfigMap Penting untuk dicatat bahwa entri yang dihasilkan ini tidak mendukung penyertaan komponen jalur. Akibatnya, ARN yang disediakan sebagai input tidak boleh berisi segmen jalur (misalnya,`arn:aws:iam::123456789012:role/service-role/fis-experiment-role`).

Gambar kontainer pod

Gambar kontainer Pod yang disediakan oleh AWS FIS di-host di Amazon ECR. Saat Anda mereferensikan gambar dari Amazon ECR, Anda harus menggunakan URI gambar lengkap.

Gambar kontainer Pod juga tersedia di AWS ECR Public Gallery.

Wilayah AWS	URI citra
AS Timur (Ohio)	`051821878176.dkr.ecr.us-east-2.amazonaws.com/aws-fis-pod:0.1`
AS Timur (Virginia Utara)	`731367659002.dkr.ecr.us-east-1.amazonaws.com/aws-fis-pod:0.1`
AS Barat (California Utara)	`080694859247.dkr.ecr.us-west-1.amazonaws.com/aws-fis-pod:0.1`
AS Barat (Oregon)	`864386544765.dkr.ecr.us-west-2.amazonaws.com/aws-fis-pod:0.1`
Afrika (Cape Town)	`056821267933.dkr.ecr.af-south-1.amazonaws.com/aws-fis-pod:0.1`
Asia Pasifik (Hong Kong)	`246405402639.dkr.ecr.ap-east-1.amazonaws.com/aws-fis-pod:0.1`
Asia Pasifik (Mumbai)	`524781661239.dkr.ecr.ap-south-1.amazonaws.com/aws-fis-pod:0.1`

Wilayah AWS	URI citra
Asia Pasifik (Seoul)	526524659354.dkr.ecr.ap-northeast-2.amazonaws.com/aws-fis-pod:0.1
Asia Pasifik (Singapura)	316401638346.dkr.ecr.ap-southeast-1.amazonaws.com/aws-fis-pod:0.1
Asia Pasifik (Sydney)	488104106298.dkr.ecr.ap-southeast-2.amazonaws.com/aws-fis-pod:0.1
Asia Pasifik (Tokyo)	635234321696.dkr.ecr.ap-northeast-1.amazonaws.com/aws-fis-pod:0.1
Kanada (Pusat)	490658072207.dkr.ecr.ca-central-1.amazonaws.com/aws-fis-pod:0.1
Eropa (Frankfurt)	713827034473.dkr.ecr.eu-central-1.amazonaws.com/aws-fis-pod:0.1
Eropa (Irlandia)	205866052826.dkr.ecr.eu-west-1.amazonaws.com/aws-fis-pod:0.1
Eropa (London)	327424803546.dkr.ecr.eu-west-2.amazonaws.com/aws-fis-pod:0.1
Eropa (Milan)	478809367036.dkr.ecr.eu-south-1.amazonaws.com/aws-fis-pod:0.1
Eropa (Paris)	154605889247.dkr.ecr.eu-west-3.amazonaws.com/aws-fis-pod:0.1
Eropa (Spanyol)	395402409451.dkr.ecr.eu-south-2.amazonaws.com/aws-fis-pod:0.1
Eropa (Stockholm)	263175118295.dkr.ecr.eu-north-1.amazonaws.com/aws-fis-pod:0.1
Timur Tengah (Bahrain)	065825543785.dkr.ecr.me-south-1.amazonaws.com/aws-fis-pod:0.1

Wilayah AWS	URI citra
Amerika Selatan (Sao Paulo)	`767113787785.dkr.ecr.sa-east-1.amazonaws.com/aws-fis-pod:0.1`
AWS GovCloud (AS-Timur)	`246533647532.dkr.ecr.us-gov-east-1.amazonaws.com/aws-fis-pod:0.1`
AWS GovCloud (AS-Barat)	`246529956514.dkr.ecr.us-gov-west-1.amazonaws.com/aws-fis-pod:0.1`

Contoh template percobaan

Berikut ini adalah contoh template eksperimen untuk the section called “aws:eks:pod-network-latency” tindakan tersebut.

```
{
    "description": "Add latency and jitter to the network interface for the target EKS
 Pods",
    "targets": {
        "myPods": {
            "resourceType": "aws:eks:pod",
            "parameters": {
                "clusterIdentifier": "mycluster",
                "namespace": "default",
                "selectorType": "labelSelector",
                "selectorValue": "mylabel=mytarget"
            },
            "selectionMode": "COUNT(3)"
        }
    },
    "actions": {
        "EksPod-latency": {
            "actionId": "aws:eks:pod-network-latency",
            "description": "Add latency",
            "parameters": {
                "kubernetesServiceAccount": "myserviceaccount",
                "duration": "PT5M",
                "delayMilliseconds": "200",
                "jitterMilliseconds": "10",
                "sources": "0.0.0.0/0"
```

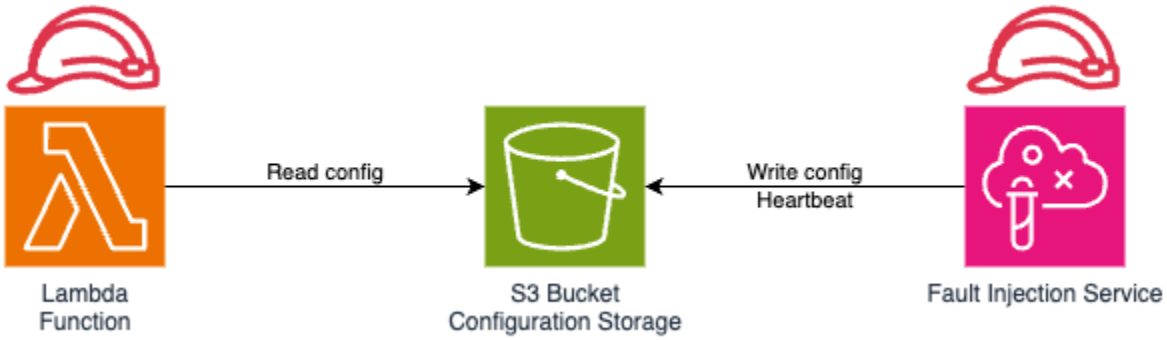
```
            },
            "targets": {
                "Pods": "myPods"
            }
        }
    },
    "stopConditions": [
        {
            "source": "none",
        }
    ],
    "roleArn": "arn:aws:iam::111122223333:role/fis-experiment-role",
    "tags": {
        "Name": "EksPodNetworkLatency"
    }
}
```

Gunakan tindakan AWS FIS aws:lambda: function

Anda dapat menggunakan tindakan aws:lambda:function untuk menyuntikkan kesalahan ke dalam pemanggilan fungsi Anda. AWS Lambda

Tindakan ini menggunakan ekstensi AWS FIS terkelola untuk menyuntikkan kesalahan. Untuk menggunakan tindakan aws:lambda:function, Anda harus melampirkan ekstensi sebagai lapisan ke fungsi Lambda Anda dan mengonfigurasi bucket Amazon S3 untuk berkomunikasi antara dan ekstensi. AWS FIS

Saat Anda menjalankan AWS FIS eksperimen yang menargetkan aws:lambda:function, baca konfigurasi AWS FIS Amazon S3 dari fungsi Lambda dan menulis informasi injeksi kesalahan ke lokasi Amazon S3 yang ditentukan, seperti yang ditunjukkan pada diagram di bawah ini.



Tindakan

- the section called “aws:lambda:invocation-add-delay”
- the section called “aws:lambda:invocation-error”
- the section called “aws:lambda:invocation-http-integration-response”

Batasan

- Ekstensi AWS FIS Lambda tidak dapat digunakan dengan fungsi yang menggunakan streaming respons. Bahkan ketika tidak ada kesalahan yang diterapkan, ekstensi AWS FIS Lambda akan menekan konfigurasi streaming. Untuk informasi selengkapnya, lihat Streaming respons untuk fungsi Lambda di panduan AWS Lambda pengguna.

Prasyarat

Sebelum menggunakan tindakan AWS FIS Lambda, pastikan Anda telah menyelesaikan tugas satu kali ini:

- Buat bucket Amazon S3 di wilayah yang Anda rencanakan untuk memulai eksperimen - Anda dapat menggunakan satu bucket Amazon S3 untuk beberapa eksperimen dan berbagi bucket di antara beberapa akun. AWS Namun, Anda harus memiliki ember terpisah untuk masing-masing Wilayah AWS.
- Buat kebijakan IAM untuk memberikan akses baca untuk ekstensi Lambda ke bucket Amazon S3 - Di template berikut, `my-config-distribution-bucket` ganti dengan nama bucket Amazon S3 yang Anda buat di atas `FisConfigs` dan dengan nama folder di bucket Amazon S3 yang ingin Anda gunakan.

```
{
  "Version": "2012-10-17",
  "Statement": [
      {
          "Sid": "AllowListingConfigLocation",
          "Effect": "Allow",
          "Action": ["s3:ListBucket"],
          "Resource": ["arn:aws:s3:::my-config-distribution-bucket"],
          "Condition": {
```

```
                "StringLike": {
                    "s3:prefix": ["FisConfigs/*"]
                }
            }
        },
        {
            "Sid": "AllowReadingObjectFromConfigLocation",
            "Effect": "Allow",
            "Action": "s3:GetObject",
            "Resource": ["arn:aws:s3:::my-config-distribution-bucket/FisConfigs/*"]
        }
    ]
}
```

- Buat kebijakan IAM untuk memberikan akses tulis AWS FIS percobaan ke bucket Amazon S3 - Di template berikut, `my-config-distribution-bucket` ganti dengan nama bucket Amazon S3 yang Anda buat di atas `FisConfigs` dan dengan nama folder di bucket Amazon S3 yang ingin Anda gunakan.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowFisToWriteAndDeleteFaultConfigurations",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:DeleteObject"
            ],
            "Resource": "arn:aws:s3:::my-config-distribution-bucket/FisConfigs/*"
        },
        {
            "Sid": "AllowFisToInspectLambdaFunctions",
            "Effect": "Allow",
            "Action": [
                "lambda:GetFunction"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowFisToDoTagLookups",
```

```
            "Effect": "Allow",
            "Action": [
                "tag:GetResources"
            ],
            "Resource": "*"
        }
    ]
}
```

Konfigurasikan fungsi Lambda

Ikuti langkah-langkah di bawah ini untuk setiap fungsi Lambda yang ingin Anda pengaruhi:

1. Lampirkan kebijakan akses baca Amazon S3 yang dibuat di atas ke fungsi Lambda.
2. Pasang AWS FIS ekstensi sebagai lapisan ke fungsi. Untuk informasi lebih lanjut tentang layer ARNs, lihatVersi AWS FIS ekstensi yang tersedia untuk Lambda.
3. Setel `AWS_FIS_CONFIGURATION_LOCATION` variabel ke ARN dari folder konfigurasi Amazon S3, misalnya. `arn:aws:s3:::my-config-distribution-bucket/FisConfigs/`
4. Atur `AWS_LAMBDA_EXEC_WRAPPER` variabel ke`/opt/aws-fis/bootstrap`.

Konfigurasikan AWS FIS eksperimen

Sebelum menjalankan eksperimen, pastikan Anda telah melampirkan kebijakan akses tulis Amazon S3 yang Anda buat dalam prasyarat ke peran eksperimen yang akan menggunakan tindakan Lambda. AWS FIS Untuk informasi selengkapnya tentang cara menyiapkan AWS FIS eksperimen, lihatMengelola AWS templat eksperimen FIS.

Pencatatan log

Ekstensi AWS FIS Lambda menulis log ke konsol dan CloudWatch log. Logging dapat dikonfigurasi menggunakan `AWS_FIS_LOG_LEVEL` variabel. Nilai yang didukung adalah `INFO`, `WARN`, dan `ERROR`. Log akan ditulis dalam format log yang dikonfigurasi untuk fungsi Lambda Anda.

Berikut ini adalah contoh log in format teks:

```
2024-08-09T18:51:38.599984Z INFO AWS FIS EXTENSION - extension enabled 1.0.1
```

Berikut ini adalah contoh log dalam format JSON:

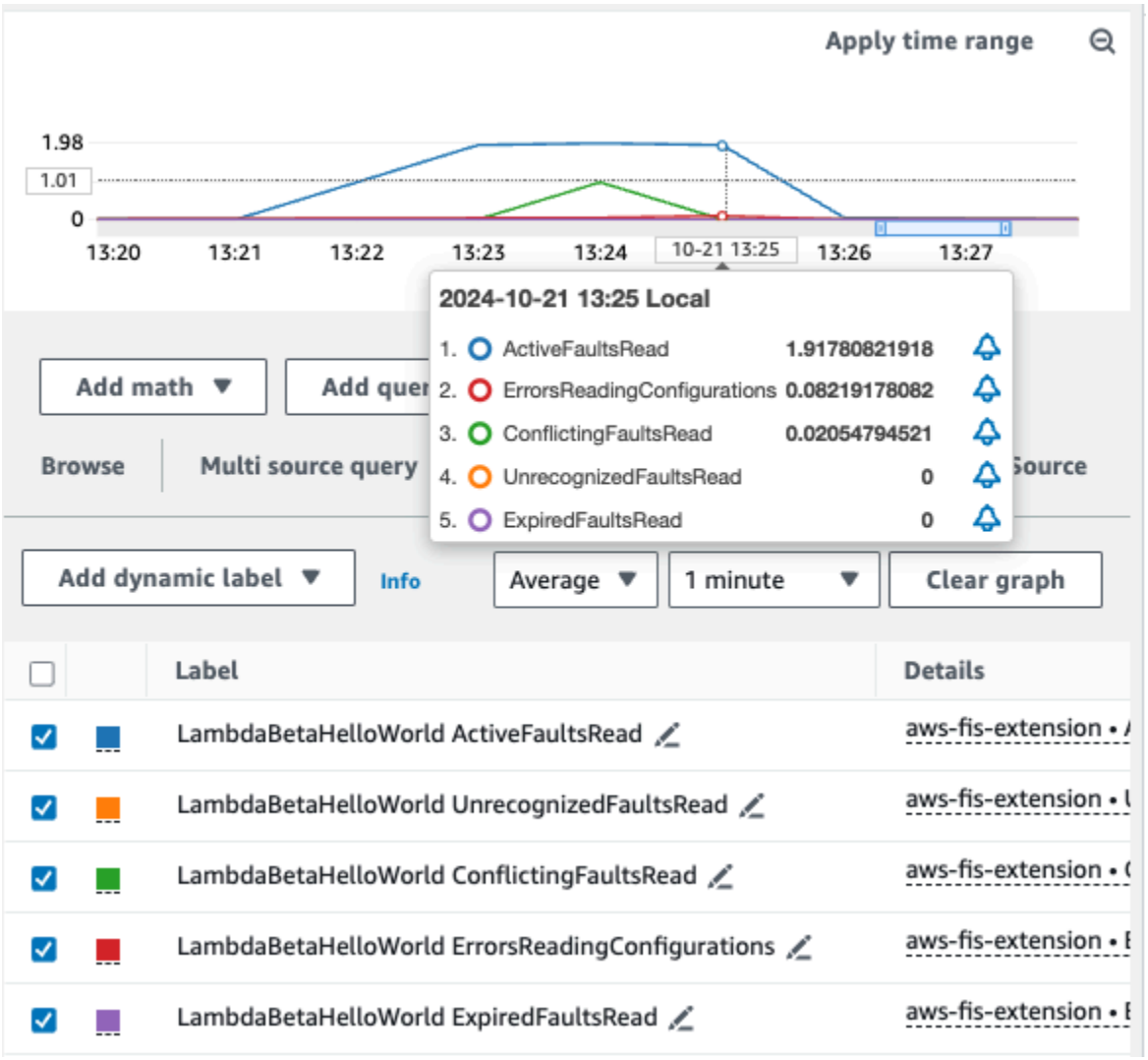
```
{
  "timestamp": "2024-10-08T17:15:36.953905Z",
  "level": "INFO",
  "fields": {
    "message": "AWS FIS EXTENSION - adding 5000 milliseconds of latency to function
 invocation",
    "requestId":"0608bf70-908f-4a17-bbfe-3782cd783d8b"
  }
}
```

Log yang dipancarkan dapat digunakan dengan filter CloudWatch metrik Amazon untuk menghasilkan metrik khusus. Untuk informasi selengkapnya tentang filter metrik, lihat Membuat metrik dari peristiwa log menggunakan filter di panduan pengguna Amazon CloudWatch Logs.

Menggunakan Format Metrik CloudWatch Tertanam (EMF)

Anda dapat mengonfigurasi ekstensi AWS FIS Lambda untuk memancarkan log EMF dengan menyetel variabel ke. `AWS_FIS_EXTENSION_METRICS` `all` Secara default, ekstensi tidak memancarkan log EMF, dan `AWS_FIS_EXTENSION_METRICS` defaultnya. `none` Log EMF diterbitkan `aws-fis-extension namespace` di CloudWatch konsol.

Di dalam `aws-fis-extension` namespace, Anda dapat memilih metrik tertentu untuk ditampilkan dalam grafik. Contoh di bawah ini menunjukkan beberapa metrik yang tersedia di `aws-fis-extension` namespace.



Topik lanjutan

Bagian ini memberikan informasi tambahan tentang cara AWS FIS kerja dengan ekstensi Lambda dan kasus penggunaan khusus.

Topik

- Memahami polling
- Memahami konkurensi
- Memahami persentase pemanggilan
- Pertimbangan khusus untuk SnapStart
- Pertimbangan khusus untuk fungsi cepat yang jarang terjadi
- Mengonfigurasi beberapa ekstensi menggunakan proxy API Lambda Runtime

- Menggunakan AWS FIS dengan runtime kontainer
- AWS FIS Variabel lingkungan Lambda

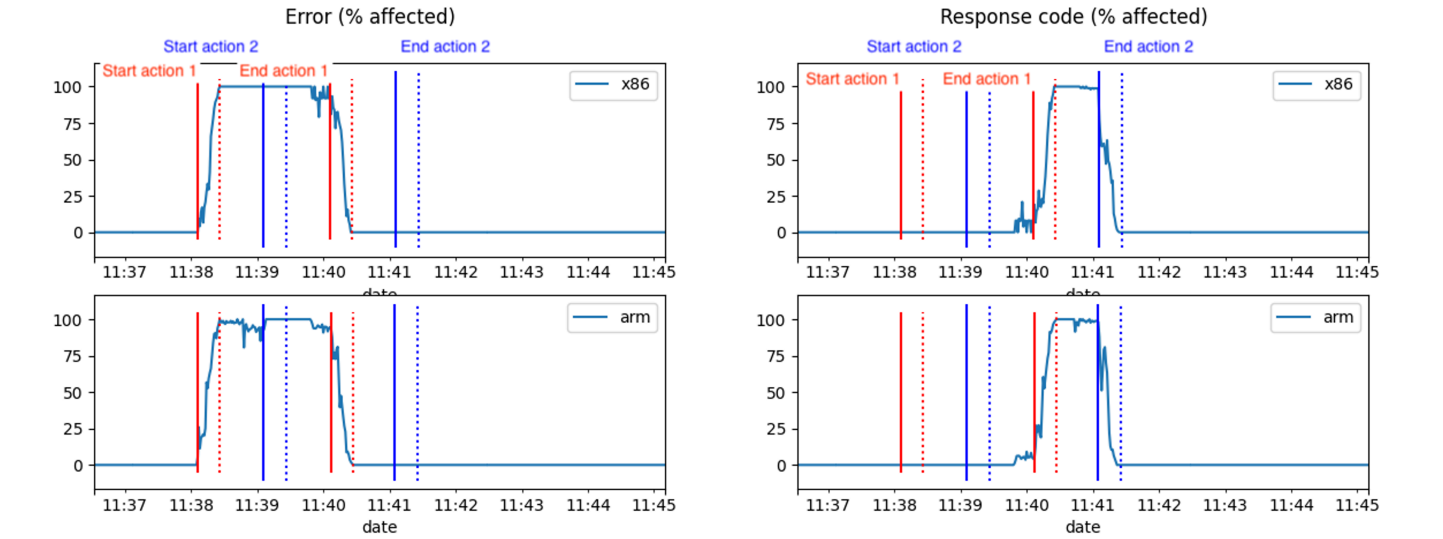
Memahami polling

Anda mungkin melihat periode peningkatan hingga 60-an sebelum kesalahan mulai memengaruhi semua pemanggilan. Ini karena ekstensi Lambda jarang melakukan polling untuk informasi konfigurasi sambil menunggu percobaan dimulai. Anda dapat menyesuaikan interval polling dengan menyetel variabel `AWS_FIS_SLOW_POLL_INTERVAL_SECONDS` lingkungan (default 60an). Nilai yang lebih rendah akan lebih sering melakukan polling tetapi menimbulkan dampak kinerja dan biaya yang lebih besar. Anda mungkin juga melihat periode ramp-down hingga 20-an setelah kesalahan disuntikkan. Ini karena jajak pendapat ekstensi lebih sering saat eksperimen sedang berjalan.

Memahami konkurensi

Anda dapat menargetkan fungsi Lambda yang sama dengan beberapa tindakan secara bersamaan. Jika semua tindakan berbeda satu sama lain, maka semua tindakan akan diterapkan. Misalnya, Anda dapat menambahkan penundaan awal sebelum mengembalikan kesalahan. Jika dua tindakan identik atau bertentangan diterapkan ke fungsi yang sama, maka hanya tindakan dengan tanggal mulai paling awal yang akan diterapkan.

Gambar di bawah ini menunjukkan dua tindakan yang saling bertentangan, aws:lambda:invocation-error dan aws:lambda:, overlapping. invocation-http-integration-response Awalnya, aws:lambda:invocation-error meningkat pada 11:38 dan berjalan selama 2 menit. Kemudian, aws:lambda: invocation-http-integration-response upaya untuk memulai pada 11:39, tetapi tidak mulai berlaku sampai 11:40 setelah tindakan pertama selesai. Untuk mempertahankan waktu percobaan, aws:lambda: invocation-http-integration-response masih selesai pada waktu yang dimaksudkan pada 11:41.



Memahami persentase pemanggilan

Tindakan AWS Fault Injection Service Lambda menggunakan target aws:lambda:function yang memungkinkan Anda memilih satu atau beberapa fungsi. AWS Lambda ARNs Dengan menggunakan ini ARNs, tindakan AWS Fault Injection Service Lambda dapat menyuntikkan kesalahan di setiap pemanggilan fungsi Lambda yang dipilih. Untuk memungkinkan Anda menyuntikkan kesalahan hanya ke dalam sebagian kecil pemanggilan, setiap tindakan memungkinkan Anda menentukan `invocationPercentage` parameter dengan nilai dari 0 hingga 100. Dengan menggunakan `invocationPercentage` parameter, Anda dapat memastikan bahwa tindakan bersamaan bahkan untuk persentase pemanggilan di bawah 100%.

Pertimbangan khusus untuk SnapStart

AWS Lambda fungsi dengan SnapStart diaktifkan akan memiliki kemungkinan lebih tinggi untuk menunggu durasi penuh `AWS_FIS_SLOW_POLL_INTERVAL_SECONDS` sebelum mengambil konfigurasi kesalahan pertama, bahkan jika percobaan sudah berjalan. Ini karena Lambda SnapStart menggunakan satu snapshot sebagai status awal untuk beberapa lingkungan eksekusi dan mempertahankan penyimpanan sementara. Untuk ekstensi AWS Fault Injection Service Lambda itu akan mempertahankan frekuensi polling dan melewati pemeriksaan konfigurasi awal pada inisialisasi lingkungan eksekusi. Untuk informasi selengkapnya tentang Lambda SnapStart, lihat Meningkatkan kinerja startup dengan Lambda SnapStart di panduan pengguna.AWS Lambda

Pertimbangan khusus untuk fungsi cepat yang jarang terjadi

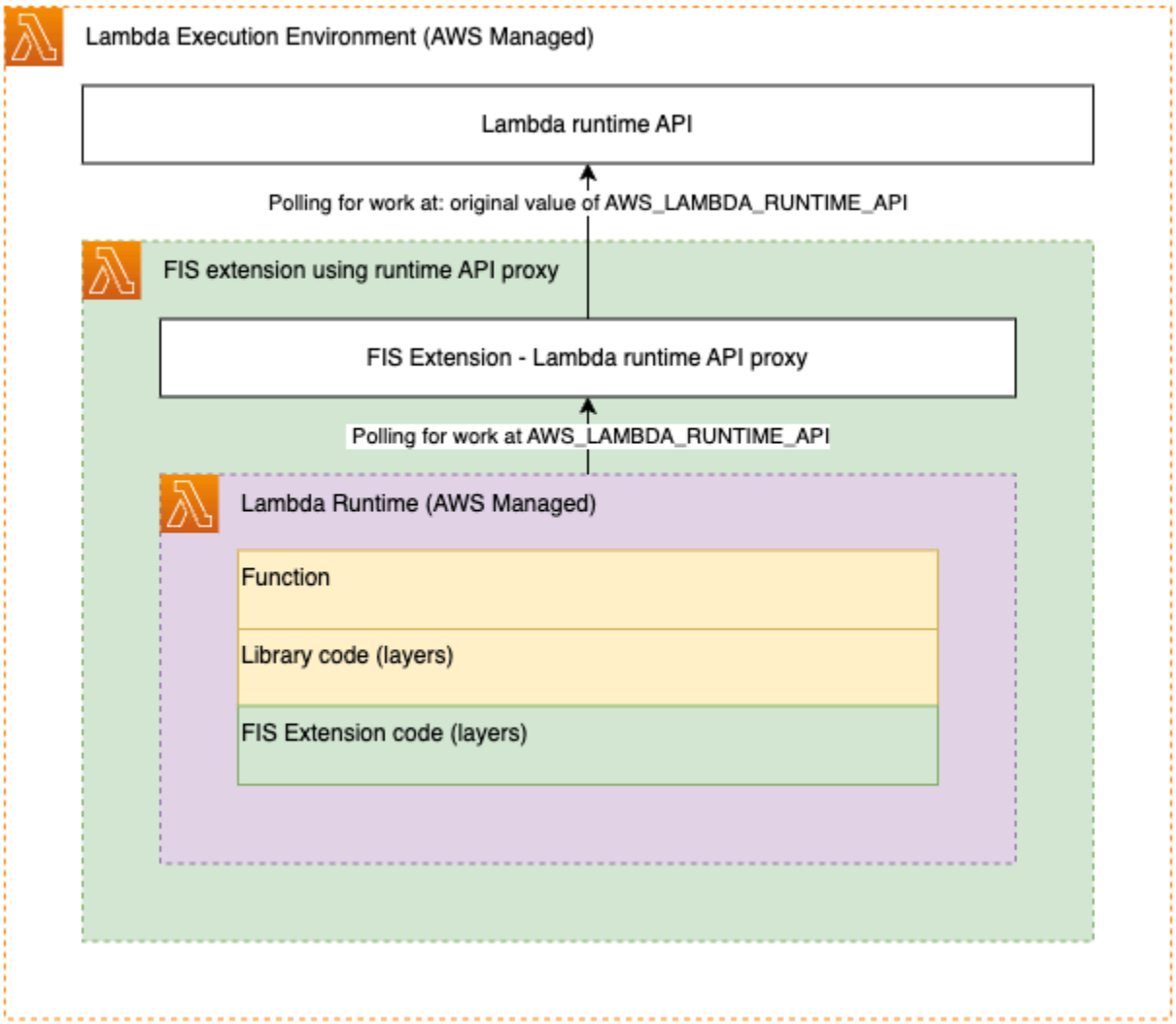
Jika fungsi Lambda Anda berjalan kurang dari durasi polling rata-rata 70 milidetik, maka utas polling mungkin memerlukan beberapa pemanggilan untuk mendapatkan konfigurasi kesalahan. Jika fungsi

berjalan jarang, misalnya setiap 15 menit sekali, maka jajak pendapat tidak akan pernah selesai. Untuk memastikan utas polling dapat selesai, atur `AWS_FIS_POLL_MAX_WAIT_MILLISECONDS` parameternya. Ekstensi akan menunggu hingga durasi yang Anda tetapkan untuk menyelesaikan polling dalam penerbangan sebelum memulai fungsi. Perhatikan bahwa ini akan meningkatkan durasi fungsi yang ditagih dan menyebabkan penundaan tambahan pada beberapa pemanggilan.

Mengonfigurasi beberapa ekstensi menggunakan proxy API Lambda Runtime

Ekstensi Lambda menggunakan proxy AWS Lambda Runtime API untuk mencegat pemanggilan fungsi sebelum mencapai runtime. Ini dilakukan dengan mengekspos proxy untuk Runtime API ke AWS Lambda runtime dan mengiklankan lokasinya di variabel. `AWS_LAMBDA_RUNTIME_API`

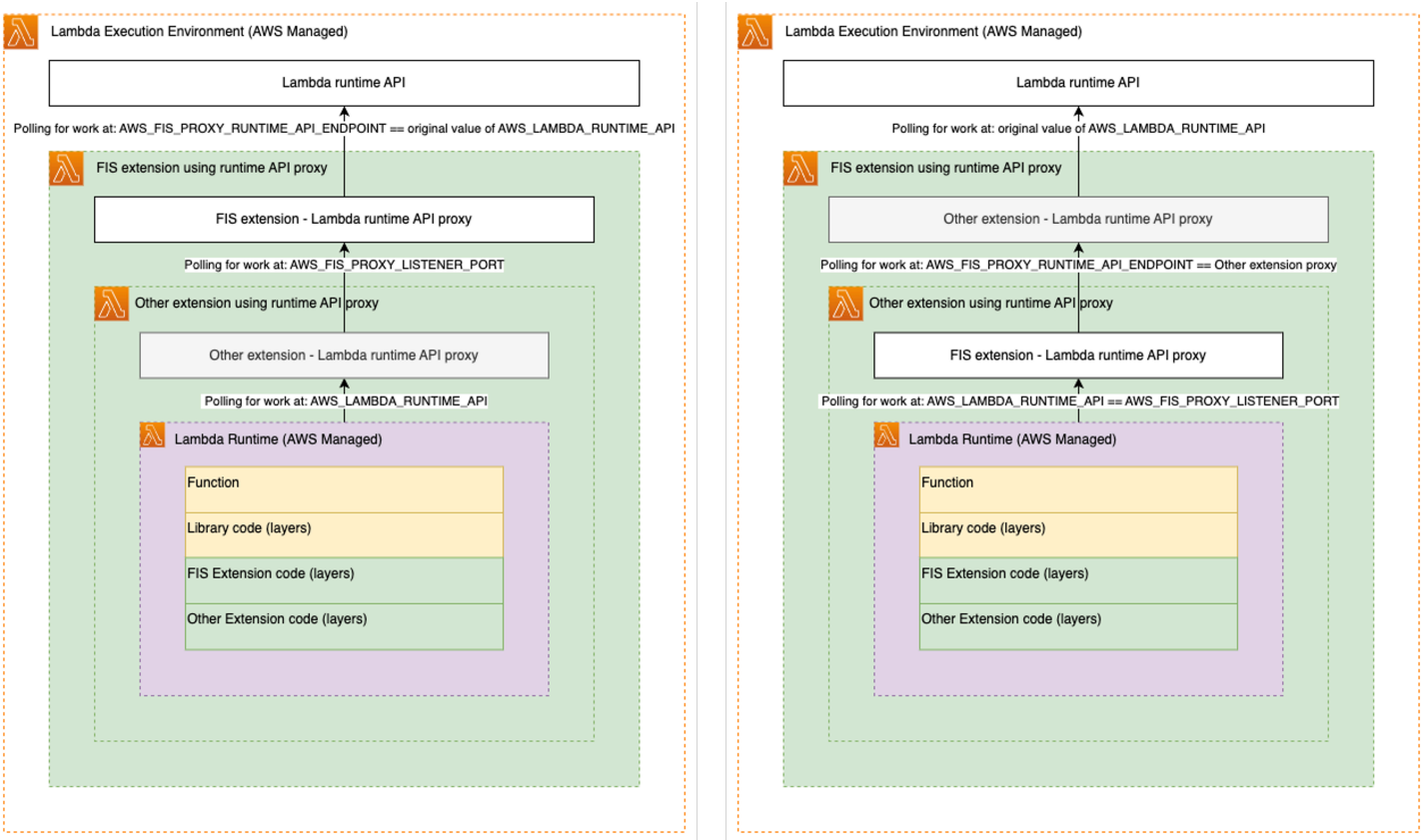
Diagram berikut menunjukkan konfigurasi untuk satu ekstensi menggunakan proxy Lambda Runtime API:



Untuk menggunakan ekstensi AWS FIS Lambda dengan ekstensi lain menggunakan pola proxy AWS Lambda Runtime API, Anda harus menghubungkan proxy menggunakan skrip bootstrap khusus. Ekstensi AWS FIS Lambda menerima variabel lingkungan berikut:

- `AWS_FIS_PROXY_RUNTIME_API_ENDPOINT`- Mengambil string dalam bentuk yang `127.0.0.1:9876` mewakili IP lokal dan port listener untuk AWS Lambda Runtime API. Ini bisa menjadi nilai asli `AWS_LAMBDA_RUNTIME_API` atau lokasi proxy lain.
- `AWS_FIS_PROXY_LISTENER_PORT`- Mengambil nomor port di mana AWS FIS ekstensi harus memulai proxy sendiri, secara default`9100`.

Dengan pengaturan ini, Anda dapat menghubungkan AWS FIS ekstensi dengan ekstensi lain menggunakan proxy API Lambda Runtime dalam dua urutan berbeda.



Untuk informasi selengkapnya tentang proxy AWS Lambda Runtime API, lihat Meningkatkan keamanan dan tata kelola runtime dengan ekstensi proxy AWS Lambda Runtime API dan Menggunakan API runtime Lambda untuk runtime kustom dalam panduan pengguna.AWS Lambda

Menggunakan AWS FIS dengan runtime kontainer

Untuk AWS Lambda fungsi yang menggunakan gambar kontainer yang menerima variabel `AWS_LAMBDA_RUNTIME_API` lingkungan, Anda dapat mengemas ekstensi AWS FIS Lambda ke dalam gambar kontainer Anda dengan mengikuti langkah-langkah di bawah ini:

1. Tentukan ARN dari lapisan untuk mengekstrak ekstensi. Untuk informasi lebih lanjut tentang cara menemukan ARN, lihat. Konfigurasikan fungsi Lambda
2. Gunakan AWS Command Line Interface (CLI) untuk meminta detail tentang ekstensi. `aws lambda get-layer-version-by-arn --arn fis-extension-arn` Respons akan berisi `Location` bidang yang berisi URL yang telah ditandatangani sebelumnya dari mana Anda dapat mengunduh ekstensi FIS sebagai file ZIP.
3. Buka zip konten ekstensi ke dalam `/opt` sistem file Docker Anda. Berikut ini adalah contoh Dockerfile berdasarkan runtime NodeJS Lambda:

```
# extension installation #
FROM amazon/aws-lambda-nodejs:12 AS builder
COPY extension.zip extension.zip
RUN yum install -y unzip
RUN mkdir -p /opt
RUN unzip extension.zip -d /opt
RUN rm -f extension.zip
FROM amazon/aws-lambda-nodejs:12
WORKDIR /opt
COPY --from=builder /opt .
# extension installation finished #
# JS example. Modify as required by your runtime
WORKDIR ${LAMBDA_TASK_ROOT}
COPY index.js package.json .
RUN npm install
CMD [ "index.handler" ]
```

Untuk informasi selengkapnya tentang gambar kontainer, lihat Membuat fungsi Lambda menggunakan gambar kontainer di panduan AWS Lambda pengguna.

AWS FIS Variabel lingkungan Lambda

Berikut ini adalah daftar variabel lingkungan untuk ekstensi AWS FIS Lambda

- `AWS_FIS_CONFIGURATION_LOCATION`- Diperlukan. Lokasi di mana AWS FIS akan menulis konfigurasi kesalahan aktif dan ekstensi akan membaca konfigurasi kesalahan. Lokasi harus dalam format Amazon S3 ARN termasuk bucket dan path. Misalnya, `arn:aws:s3:::my-fis-config-bucket/FisConfigs/`.
- `AWS_LAMBDA_EXEC_WRAPPER`- Diperlukan. Lokasi skrip AWS Lambda pembungkus yang digunakan untuk mengkonfigurasi ekstensi AWS FIS Lambda. Ini harus diatur ke `/opt/aws-fis/bootstrap` skrip yang disertakan dengan ekstensi.
- `AWS_FIS_LOG_LEVEL`- Opsional. Tingkat log untuk pesan yang dipancarkan oleh ekstensi Lambda AWS FIS . Nilai yang didukung adalah `INFO`, `WARN`, dan `ERROR`. Jika tidak disetel, AWS FIS ekstensi akan default ke`INFO`.
- `AWS_FIS_EXTENSION_METRICS`- Opsional. Kemungkinan nilainya adalah `all` and `none`. Jika disetel `all` ke ekstensi akan memancarkan metrik EMF di bawah. `aws-fis-extension namespace`
- `AWS_FIS_SLOW_POLL_INTERVAL_SECONDS`- Opsional. Jika set akan mengganti interval polling (dalam hitungan detik) sementara ekstensi tidak menyuntikkan kesalahan dan menunggu konfigurasi kesalahan ditambahkan ke lokasi konfigurasi. Default ke `60`.
- `AWS_FIS_PROXY_RUNTIME_API_ENDPOINT`- Opsional. Jika set akan mengganti nilai `AWS_LAMBDA_RUNTIME_API` untuk menentukan di mana AWS FIS ekstensi berinteraksi dengan API AWS Lambda runtime untuk mengontrol pemanggilan fungsi. Mengharapkan IP: PORT, misalnya,. `127.0.0.1:9000` Untuk informasi selengkapnya*`AWS_LAMBDA_RUNTIME_API`*, lihat Menggunakan API runtime Lambda untuk runtime kustom dalam panduan pengguna.AWS Lambda
- `AWS_FIS_PROXY_LISTENER_PORT`- Opsional. Mendefinisikan port tempat ekstensi AWS FIS Lambda mengekspos proxy API runtime AWS Lambda yang dapat digunakan oleh ekstensi lain atau runtime. Default ke `9100`.
- `AWS_FIS_POLL_MAX_WAIT_MILLISECONDS`- Opsional. Jika disetel ke nilai bukan nol, variabel ini menentukan jumlah milidetik ekstensi akan menunggu polling asinkron dalam penerbangan selesai sebelum mengevaluasi konfigurasi kesalahan dan memulai pemanggilan runtime. Default ke `0`.

Versi AWS FIS ekstensi yang tersedia untuk Lambda

Bagian ini mencakup informasi tentang versi ekstensi AWS FIS Lambda. Ekstensi ini mendukung fungsi Lambda yang dikembangkan untuk platform x86-64 dan ARM64 (Graviton2). Fungsi Lambda Anda harus dikonfigurasi untuk menggunakan Nama Sumber Daya Amazon (ARN) tertentu untuk Wilayah AWS tempat saat ini di-host. Anda dapat melihat Wilayah AWS dan detail ARN di bawah ini.

Topik

- AWS FIS Catatan rilis ekstensi Lambda
- Panduan Akses untuk Ekstensi Lambda ARNs

AWS FIS Catatan rilis ekstensi Lambda

Tabel berikut menjelaskan perubahan yang dibuat pada versi terbaru dari ekstensi AWS FIS Lambda

Versi	Tanggal peluncuran	Catatan
1.0.0	2024-10-29	Rilis awal

Panduan Akses untuk Ekstensi Lambda ARNs

Anda harus memiliki setidaknya satu parameter di Anda Akun AWS dan Wilayah AWS sebelum Anda dapat mencari parameter publik menggunakan konsol. Untuk menemukan parameter publik, lihat Menemukan parameter publik di Parameter Store.

Akses Konsol:

1. Buka AWS Systems Manager konsol di https://console.aws.amazon.com/systems-manager/.
2. Di panel navigasi, pilih Penyimpanan Parameter.
3. Pilih tab Parameter publik.
4. Pilih dropdown Pilih layanan. Dari opsi dropdown, pilih. `fis`
5. (Opsional) Filter parameter yang Anda pilih dengan memasukkan informasi lebih lanjut ke dalam bilah pencarian. Untuk arsitektur arm64, filter parameter dengan memasukkan “arm64". Untuk arsitektur x86_64, filter parameter dengan memasukkan “x86_64".
6. Pilih parameter publik yang ingin Anda gunakan.
7. Dari detail parameter, cari nilai ARN. Salin ARN untuk digunakan dalam mengonfigurasi ekstensi lapisan pada fungsi Lambda target Anda.

AWS CLI Akses:

Nama Parameter SSM

Nama parameter SSM berikut tersedia untuk arsitektur yang berbeda:

1. arm64: `/aws/service/fis/lambda-extension/AWS-FIS-extension-arm64/1.x.x`
2. x86_64: `/aws/service/fis/lambda-extension/AWS-FIS-extension-x86_64/1.x.x`

AWS CLI Format Perintah

Untuk mengambil ekstensi ARNs, gunakan format AWS CLI perintah berikut di mana ParameterName adalah nama untuk arsitektur dan wilayah adalah target: Wilayah AWS

```
aws ssm get-parameter --name parameterName --region region
```

Contoh Penggunaan

```
aws ssm get-parameter --name /aws/service/fis/lambda-extension/AWS-FIS-extension-
x86_64/1.x.x --region ap-southeast-2
```

Format Respons

Perintah mengembalikan objek JSON yang berisi rincian parameter seperti berikut ini. ARN dari lapisan Lambda disertakan dalam bidang Nilai objek Parameter. Salin ARN untuk digunakan dalam mengonfigurasi ekstensi lapisan pada fungsi Lambda target Anda.

```
 {
     "Parameter":
        {
             "Name": "/aws/service/fis/lambda-extension/AWS-FIS-extension-
x86_64/1.x.x",
             "Type": "String",
             "Value": "arn:aws:lambda:ap-southeast-2:211125361907:layer:aws-fis-
extension-x86_64:9",
             "Version": 1,
             "LastModifiedDate": "2025-01-02T15:13:54.465000-05:00",
             "ARN": "arn:aws:ssm:ap-southeast-2::parameter/aws/service/fis/lambda-
extension/AWS-FIS-extension-x86_64/1.x.x",
             "DataType": "text"
        }
 }
```

Akses Terprogram:

Ambil parameter publik ini secara terprogram saat membangun atau mengonfigurasi fungsi Lambda Anda menggunakan Infrastructure as Code (IAc). Pendekatan ini membantu mempertahankan fungsi Lambda Anda dengan ARN versi lapisan terbaru tanpa memerlukan pembaruan kode manual yang akan diperlukan jika AWS FIS lapisan ekstensi ARN di-hardcode. Sumber daya berikut menunjukkan cara mengambil parameter publik menggunakan platform IAc umum:

- Dapatkan parameter publik menggunakan AWS SDK
- Dapatkan parameter publik dari AWS Systems Manager Parameter Store
- Dapatkan parameter publik menggunakan Terraform

Mengelola AWS templat eksperimen FIS

Anda dapat membuat dan mengelola template eksperimen menggunakan konsol AWS FIS atau baris perintah. Template eksperimen berisi satu atau beberapa tindakan untuk dijalankan pada target tertentu selama percobaan. Ini juga berisi kondisi berhenti yang mencegah eksperimen keluar dari batas. Untuk informasi selengkapnya tentang komponen templat eksperimen, lihatKomponen template percobaan. Setelah Anda membuat template eksperimen, Anda dapat menggunakannya untuk menjalankan eksperimen.

Tugas

- Buat template eksperimen
- Lihat templat eksperimen
- Buat pratinjau target dari templat eksperimen
- Memulai percobaan dari template
- Perbarui templat eksperimen
- Tag template percobaan
- Hapus templat eksperimen
- Contoh AWS templat eksperimen FIS

Buat template eksperimen

Sebelum memulai, selesaikan tugas berikut:

- Rencanakan eksperimen Anda.
- Buat peran IAM yang memberikan izin layanan AWS FIS untuk melakukan tindakan atas nama Anda. Untuk informasi selengkapnya, lihat Peran IAM untuk eksperimen AWS FIS.
- Pastikan Anda memiliki akses ke AWS FIS. Untuk informasi selengkapnya, lihat contoh kebijakan AWS FIS.

Untuk membuat template eksperimen menggunakan konsol

1. Buka konsol AWS FIS di https://console.aws.amazon.com/fis/.
2. Di panel navigasi, pilih Template eksperimen.
3. Pilih Buat template eksperimen.

4. Untuk Langkah 1, Tentukan detail template, lakukan hal berikut:

   a. Untuk Deskripsi dan nama, masukkan deskripsi untuk templat, sepertiAmazon S3 Network Disrupt Connectivity.

   b. (Opsional) Untuk penargetan Akun, pilih Beberapa akun untuk mengonfigurasi templat eksperimen multi-akun.

   c. Pilih Berikutnya, dan pindah ke Langkah 2, Tentukan tindakan dan target.

5. Untuk Tindakan, tentukan kumpulan tindakan untuk templat. Untuk setiap tindakan, pilih Tambah tindakan dan selesaikan yang berikut ini:

   - Untuk Nama, masukkan nama untuk tindakan tersebut.

     Karakter yang diizinkan adalah karakter alfanumerik, tanda hubung (-), dan garis bawah (_). Nama harus dimulai dengan huruf. Tidak ada spasi yang diizinkan. Setiap nama tindakan harus unik dalam template ini.

   - (Opsional) Untuk Deskripsi, masukkan deskripsi untuk tindakan tersebut. Panjang maksimum adalah 512 karakter.

   - (Opsional) Untuk Mulai setelahnya, pilih tindakan lain yang ditentukan dalam templat ini yang harus diselesaikan sebelum tindakan saat ini dimulai. Jika tidak, tindakan berjalan pada awal percobaan.

   - Untuk tipe Action, pilih tindakan AWS FIS.

   - Untuk Target, pilih target yang Anda tentukan di bagian Target. Jika Anda belum menentukan target untuk tindakan ini, AWS FIS menciptakan target baru untuk Anda.

   - Untuk parameter Tindakan, tentukan parameter untuk tindakan tersebut. Bagian ini hanya muncul jika tindakan AWS FIS memiliki parameter.

   - Pilih Simpan.

6. Untuk Target, tentukan sumber daya target untuk melakukan tindakan. Anda harus menentukan setidaknya satu ID sumber daya atau satu tag sumber daya sebagai target. Pilih Edit untuk mengedit target yang AWS FIS buat untuk Anda pada langkah sebelumnya, atau pilih Tambah target. Untuk setiap target, lakukan hal berikut:

   - Untuk Nama, masukkan nama untuk target.

     Karakter yang diizinkan adalah karakter alfanumerik, tanda hubung (-), dan garis bawah (_). Nama harus dimulai dengan huruf. Tidak ada spasi yang diizinkan. Setiap nama target harus unik dalam template ini.

- Untuk jenis Sumber Daya, pilih jenis sumber daya yang didukung untuk tindakan tersebut.
- Untuk metode Target, lakukan salah satu hal berikut:
  - Pilih Resource IDs dan kemudian pilih atau tambahkan sumber daya IDs.
  - Pilih tag sumber daya, filter, dan parameter lalu tambahkan tag dan filter yang Anda butuhkan. Untuk informasi selengkapnya, lihat the section called “Identifikasi sumber daya target”.
- Untuk mode Seleksi, pilih Hitung untuk menjalankan tindakan pada jumlah target yang diidentifikasi yang ditentukan atau pilih Persen untuk menjalankan tindakan pada persentase tertentu dari target yang diidentifikasi. Secara default, tindakan berjalan pada semua target yang diidentifikasi.
- Pilih Simpan.

7. Untuk memperbarui tindakan dengan target yang Anda buat, temukan tindakan di bawah Tindakan, pilih Edit, lalu perbarui Target. Anda dapat menggunakan target yang sama untuk beberapa tindakan.
8. (Opsional) Untuk opsi percobaan, pilih perilaku mode resolusi target kosong.
9. Pilih Berikutnya untuk pindah ke Langkah 3, Konfigurasikan akses layanan.
10. Untuk Akses Layanan, pilih Gunakan peran IAM yang ada, lalu pilih peran IAM yang Anda buat seperti yang dijelaskan dalam prasyarat untuk tutorial ini. Jika peran Anda tidak ditampilkan, verifikasi bahwa ia memiliki hubungan kepercayaan yang diperlukan. Untuk informasi selengkapnya, lihat the section called “Peran percobaan”.
11. (Hanya eksperimen multi-akun) Untuk konfigurasi akun Target, tambahkan ARN Peran dan deskripsi opsional untuk setiap akun target. Untuk mengunggah peran akun target ARNs dengan file CSV, pilih Peran Unggah ARNs untuk semua akun target, lalu pilih Pilih file.CSV
12. Pilih Berikutnya untuk pindah ke Langkah 4, Konfigurasikan pengaturan opsional.
13. (Opsional) Untuk kondisi Berhenti, pilih CloudWatch alarm Amazon untuk kondisi berhenti. Untuk informasi selengkapnya, lihat Kondisi berhenti untuk AWS FIS.
14. (Opsional) Untuk Log, konfigurasikan opsi tujuan. Untuk mengirim log ke bucket S3, pilih Kirim ke bucket Amazon S3 dan masukkan nama bucket dan awalan. Untuk mengirim CloudWatch log ke Log, pilih Kirim ke CloudWatch Log dan masukkan grup log.
15. (Opsional) Untuk Tag, pilih Tambahkan tag baru dan tentukan kunci tag dan nilai tag. Tag yang Anda tambahkan diterapkan ke template eksperimen Anda, bukan eksperimen yang dijalankan menggunakan template.
16. Pilih Berikutnya untuk pindah ke Langkah 5, Tinjau dan buat.

17. Tinjau template dan pilih Buat template eksperimen. Ketika diminta untuk konfirmasi, masukkan`create`, Lalu pilih Buat template percobaan.

Untuk membuat template eksperimen menggunakan CLI

Gunakan perintah create-experiment-template.

Anda dapat memuat template eksperimen dari file JSON.

Gunakan parameter `--cli-input-json`.

```
aws fis create-experiment-template --cli-input-json fileb://<path-to-json-file>
```

Untuk informasi selengkapnya, lihat Membuat template kerangka CLI di Panduan Pengguna.AWS Command Line Interface Misalnya template, lihatContoh AWS templat eksperimen FIS.

Lihat templat eksperimen

Anda dapat melihat template eksperimen yang Anda buat.

Untuk melihat template eksperimen menggunakan konsol

1. Buka konsol AWS FIS di https://console.aws.amazon.com/fis/.
2. Di panel navigasi, pilih Template eksperimen.
3. Untuk melihat informasi tentang templat tertentu, pilih ID templat Eksperimen.
4. Di bagian Detail, Anda dapat melihat deskripsi dan kondisi berhenti untuk template.
5. Untuk melihat tindakan untuk templat eksperimen, pilih Tindakan.
6. Untuk melihat target template eksperimen, pilih Target.
7. Untuk melihat tag untuk templat eksperimen, pilih Tag.

Untuk melihat template eksperimen menggunakan CLI

Gunakan list-experiment-templatesperintah untuk mendapatkan daftar templat eksperimen, dan gunakan get-experiment-templateperintah untuk mendapatkan informasi tentang templat eksperimen tertentu.

Buat pratinjau target dari templat eksperimen

Sebelum memulai eksperimen, Anda dapat membuat pratinjau target untuk memverifikasi bahwa templat eksperimen Anda dikonfigurasi untuk menargetkan sumber daya yang diharapkan. Sumber daya yang ditargetkan saat Anda memulai eksperimen sebenarnya mungkin berbeda dari yang ada di pratinjau, karena sumber daya dapat dihapus, diperbarui, atau diambil sampelnya secara acak. Saat membuat pratinjau target, Anda memulai eksperimen yang melewatkan semua tindakan.

> **Note**
>
> Membuat pratinjau target tidak memungkinkan Anda memverifikasi bahwa Anda memiliki izin yang diperlukan untuk mengambil tindakan pada sumber daya Anda.

Untuk memulai pratinjau target menggunakan konsol

1. Buka konsol AWS FIS di https://console.aws.amazon.com/fis/.
2. Di panel navigasi, pilih Template eksperimen.
3. Untuk melihat target template eksperimen, pilih Target.
4. Untuk memverifikasi sumber daya target untuk templat eksperimen, pilih Hasilkan Pratinjau. Saat Anda menjalankan eksperimen, pratinjau target ini akan diperbarui secara otomatis dengan target dari eksperimen terbaru.

Untuk memulai pratinjau target menggunakan CLI

- Jalankan perintah start-experiment berikut. Ganti nilai dalam huruf miring dengan nilai Anda sendiri.

```
aws fis start-experiment \
    --experiment-options actionsMode=skip-all \
    --experiment-template-id EXTxxxxxxxxx
```

Memulai percobaan dari template

Setelah Anda membuat template eksperimen, Anda dapat memulai eksperimen menggunakan template itu.

Saat Anda memulai eksperimen, kami membuat snapshot dari template yang ditentukan dan menggunakan snapshot itu untuk menjalankan eksperimen. Oleh karena itu, jika template eksperimen diperbarui atau dihapus saat percobaan sedang berjalan, perubahan tersebut tidak berdampak pada eksperimen yang sedang berjalan.

Saat Anda memulai eksperimen, AWS FIS membuat peran terkait layanan atas nama Anda. Untuk informasi selengkapnya, lihat Gunakan peran terkait layanan untuk Layanan Injeksi AWS Kesalahan.

Setelah Anda memulai percobaan, Anda dapat menghentikannya kapan saja. Untuk informasi selengkapnya, lihat Menghentikan sebuah percobaan.

Untuk memulai percobaan menggunakan konsol

1. Buka konsol AWS FIS di https://console.aws.amazon.com/fis/.
2. Di panel navigasi, pilih Template eksperimen.
3. Pilih template eksperimen, dan pilih Mulai eksperimen.
4. (Opsional) Untuk menambahkan tag ke eksperimen Anda, pilih Tambahkan tag baru dan masukkan kunci tag dan nilai tag.
5. Pilih Mulai percobaan. Saat diminta konfirmasi, masukkan **`start`** dan pilih Mulai eksperimen.

Untuk memulai percobaan menggunakan CLI

Gunakan perintah start-experiment.

Perbarui templat eksperimen

Anda dapat memperbarui template eksperimen yang ada. Saat Anda memperbarui templat eksperimen, perubahan tidak memengaruhi eksperimen yang sedang berjalan yang menggunakan templat.

Untuk memperbarui template eksperimen menggunakan konsol

1. Buka konsol AWS FIS di https://console.aws.amazon.com/fis/.
2. Di panel navigasi, pilih Template eksperimen.
3. Pilih template eksperimen, lalu pilih Tindakan, Perbarui templat eksperimen.
4. Ubah detail templat sesuai kebutuhan, dan pilih Perbarui templat eksperimen.

Untuk memperbarui template eksperimen menggunakan CLI

Gunakan perintah update-experiment-template.

Tag template percobaan

Anda dapat menerapkan tag Anda sendiri ke templat percobaan untuk membantu Anda mengaturnya. Anda juga dapat menerapkan kebijakan IAM berbasis tag untuk mengontrol akses ke templat eksperimen.

Untuk menandai template eksperimen menggunakan konsol

1. Buka konsol AWS FIS di https://console.aws.amazon.com/fis/.
2. Di panel navigasi, pilih Template eksperimen.
3. Pilih template eksperimen dan pilih Tindakan, Kelola tag.
4. Untuk menambahkan tag baru, pilih Tambahkan tag baru, lalu tentukan kunci dan nilai.

   Untuk menghapus tag, pilih Hapus untuk tag.
5. Pilih Simpan.

Untuk menandai template eksperimen menggunakan CLI

Gunakan perintah tag-resource.

Hapus templat eksperimen

Jika Anda tidak lagi membutuhkan templat percobaan, Anda dapat menghapusnya. Saat Anda menghapus templat eksperimen, eksperimen apa pun yang sedang berjalan yang menggunakan templat tidak akan terpengaruh. Eksperimen terus berjalan sampai selesai atau berhenti. Namun, templat eksperimen yang dihapus tidak tersedia untuk dilihat dari halaman Eksperimen di konsol.

Untuk menghapus template eksperimen menggunakan konsol

1. Buka konsol AWS FIS di https://console.aws.amazon.com/fis/.
2. Di panel navigasi, pilih Template eksperimen.
3. Pilih template eksperimen, lalu pilih Tindakan, Hapus template eksperimen.
4. Saat diminta konfirmasi, masukkan **`delete`** dan pilih Hapus templat eksperimen.

Untuk menghapus template percobaan menggunakan CLI

Gunakan perintah delete-experiment-template.

Contoh AWS templat eksperimen FIS

Jika Anda menggunakan AWS FIS API atau alat baris perintah untuk membuat template eksperimen, Anda dapat membuat template di JavaScript Object Notation (JSON). Untuk informasi selengkapnya tentang komponen templat eksperimen, lihatAWS Komponen template eksperimen FIS.

Untuk membuat eksperimen menggunakan salah satu contoh templat, simpan ke file JSON (misalnya,`my-template.json`), ganti nilai placeholder *`italics`* dengan nilai Anda sendiri, lalu jalankan perintah berikut. create-experiment-template

```
aws fis create-experiment-template --cli-input-json file://my-template.json
```

Contoh template

- Hentikan EC2 instance berdasarkan filter
- Hentikan sejumlah EC2 instance tertentu
- Jalankan dokumen SSM AWS FIS yang telah dikonfigurasi sebelumnya
- Jalankan runbook Otomasi yang telah ditentukan sebelumnya
- Tindakan Throttle API pada EC2 instance dengan peran IAM target
- Uji stres CPU pod di klaster Kubernetes

Hentikan EC2 instance berdasarkan filter

Contoh berikut menghentikan semua EC2 instance Amazon yang berjalan di Wilayah tertentu dengan tag yang ditentukan di VPC yang ditentukan. Ini restart mereka setelah dua menit.

```
{
    "tags": {
        "Name": "StopEC2InstancesWithFilters"
    },
    "description": "Stop and restart all instances in us-east-1b with the tag env=prod
 in the specified VPC",
    "targets": {
        "myInstances": {
            "resourceType": "aws:ec2:instance",
```

```
            "resourceTags": {
                "env": "prod"
            },
            "filters": [
                {
                    "path": "Placement.AvailabilityZone",
                    "values": ["us-east-1b"]
                },
                {
                    "path": "State.Name",
                    "values": ["running"]
                },
                {
                    "path": "VpcId",
                    "values": [ "vpc-aabbcc11223344556"]
                }
            ],
            "selectionMode": "ALL"
        }
    },
    "actions": {
        "StopInstances": {
            "actionId": "aws:ec2:stop-instances",
            "description": "stop the instances",
            "parameters": {
                "startInstancesAfterDuration": "PT2M"
            },
            "targets": {
                "Instances": "myInstances"
            }
        }
    },
    "stopConditions": [
        {
            "source": "aws:cloudwatch:alarm",
            "value": "arn:aws:cloudwatch:us-east-1:111122223333:alarm:alarm-name"
        }
    ],
    "roleArn": "arn:aws:iam::111122223333:role/role-name"
}
```

Hentikan sejumlah EC2 instance tertentu

Contoh berikut menghentikan tiga contoh dengan tag yang ditentukan. AWS FIS memilih instance tertentu untuk berhenti secara acak. Ini memulai ulang contoh ini setelah dua menit.

```
{
    "tags": {
        "Name": "StopEC2InstancesByCount"
    },
    "description": "Stop and restart three instances with the specified tag",
    "targets": {
        "myInstances": {
            "resourceType": "aws:ec2:instance",
            "resourceTags": {
                "env": "prod"
            },
            "selectionMode": "COUNT(3)"
        }
    },
    "actions": {
        "StopInstances": {
            "actionId": "aws:ec2:stop-instances",
            "description": "stop the instances",
            "parameters": {
                "startInstancesAfterDuration": "PT2M"
            },
            "targets": {
                "Instances": "myInstances"
            }
        }
    },
    "stopConditions": [
        {
            "source": "aws:cloudwatch:alarm",
            "value": "arn:aws:cloudwatch:us-east-1:111122223333:alarm:alarm-name"
        }
    ],
    "roleArn": "arn:aws:iam::111122223333:role/role-name"
}
```

Jalankan dokumen SSM AWS FIS yang telah dikonfigurasi sebelumnya

Contoh berikut menjalankan injeksi kesalahan CPU selama 60 detik pada EC2 instance yang ditentukan menggunakan dokumen FIS SSM yang telah dikonfigurasi sebelumnya, AWS -CPU-Stress. AWSFIS-Run AWS FIS memantau percobaan selama dua menit.

```
{
    "tags": {
        "Name": "CPUStress"
    },
    "description": "Run a CPU fault injection on the specified instance",
    "targets": {
        "myInstance": {
            "resourceType": "aws:ec2:instance",
            "resourceArns": ["arn:aws:ec2:us-east-1:111122223333:instance/instance-
id"],
            "selectionMode": "ALL"
        }
    },
    "actions": {
        "CPUStress": {
            "actionId": "aws:ssm:send-command",
            "description": "run cpu stress using ssm",
            "parameters": {
                "duration": "PT2M",
                "documentArn": "arn:aws:ssm:us-east-1::document/AWSFIS-Run-CPU-Stress",
                "documentParameters": "{\"DurationSeconds\": \"60\",
 \"InstallDependencies\": \"True\", \"CPU\": \"0\"}"
            },
            "targets": {
                "Instances": "myInstance"
            }
        }
    },
    "stopConditions": [
        {
            "source": "aws:cloudwatch:alarm",
            "value": "arn:aws:cloudwatch:us-east-1:111122223333:alarm:alarm-name"
        }
    ],
    "roleArn": "arn:aws:iam::111122223333:role/role-name"
}
```

Jalankan runbook Otomasi yang telah ditentukan sebelumnya

Contoh berikut menerbitkan pemberitahuan ke Amazon SNS menggunakan runbook yang disediakan oleh Systems Manager, AWS-Publish. SNSNotification Peran harus memiliki izin untuk mempublikasikan pemberitahuan ke topik SNS yang ditentukan.

```
{
    "description": "Publish event through SNS",
    "stopConditions": [
        {
            "source": "none"
        }
    ],
    "targets": {
    },
    "actions": {
        "sendToSns": {
            "actionId": "aws:ssm:start-automation-execution",
            "description": "Publish message to SNS",
            "parameters": {
                "documentArn": "arn:aws:ssm:us-east-1::document/AWS-
PublishSNSNotification",
                "documentParameters": "{\"Message\": \"Hello, world\", \"TopicArn\":
 \"arn:aws:sns:us-east-1:111122223333:topic-name\"}",
                "maxDuration": "PT1M"
            },
            "targets": {
            }
        }
    },
    "roleArn": "arn:aws:iam::111122223333:role/role-name"
}
```

Tindakan Throttle API pada EC2 instance dengan peran IAM target

Contoh berikut membatasi 100% panggilan API yang ditentukan dalam definisi tindakan untuk panggilan API yang dibuat oleh peran IAM yang ditentukan dalam definisi target.

> **Note**
>
> Jika Anda ingin menargetkan EC2 instans yang merupakan anggota grup Auto Scaling, gunakan tindakan aws:ec2asg-insufficient-instance-capacity: -error, dan targetkan

berdasarkan grup Auto Scaling sebagai gantinya. Untuk informasi selengkapnya, lihat aws:ec2:asg-insufficient-instance-capacity-error.

```
{
    "tags": {
        "Name": "ThrottleEC2APIActions"
    },
    "description": "Throttle the specified EC2 API actions on the specified IAM role",
    "targets": {
        "myRole": {
            "resourceType": "aws:iam:role",
            "resourceArns": ["arn:aws:iam::111122223333:role/role-name"],
            "selectionMode": "ALL"
        }
    },
    "actions": {
        "ThrottleAPI": {
            "actionId": "aws:fis:inject-api-throttle-error",
            "description": "Throttle APIs for 5 minutes",
            "parameters": {
                "service": "ec2",
                "operations": "DescribeInstances,DescribeVolumes",
                "percentage": "100",
                "duration": "PT2M"
            },
            "targets": {
                "Roles": "myRole"
            }
        }
    },
    "stopConditions": [
        {
            "source": "aws:cloudwatch:alarm",
            "value": "arn:aws:cloudwatch:us-east-1:111122223333:alarm:alarm-name"
        }
    ],
    "roleArn": "arn:aws:iam::111122223333:role/role-name"
}
```

Uji stres CPU pod di klaster Kubernetes

Contoh berikut menggunakan Chaos Mesh untuk stress test CPU pod di cluster Amazon EKS Kubernetes selama satu menit.

```
{
    "description": "ChaosMesh StressChaos example",
    "targets": {
        "Cluster-Target-1": {
            "resourceType": "aws:eks:cluster",
            "resourceArns": [
                "arn:aws:eks:arn:aws::111122223333:cluster/cluster-id"
            ],
            "selectionMode": "ALL"
        }
    },
    "actions": {
        "TestCPUStress": {
            "actionId": "aws:eks:inject-kubernetes-custom-resource",
            "parameters": {
                "maxDuration": "PT2M",
                "kubernetesApiVersion": "chaos-mesh.org/v1alpha1",
                "kubernetesKind": "StressChaos",
                "kubernetesNamespace": "default",
                "kubernetesSpec": "{\"selector\":{\"namespaces\":[\"default\"],
\"labelSelectors\":{\"run\":\"nginx\"}},\"mode\":\"all\",\"stressors\": {\"cpu\":
{\"workers\":1,\"load\":50}},\"duration\":\"1m\"}"
            },
            "targets": {
                "Cluster": "Cluster-Target-1"
            }
        }
    },
    "stopConditions": [{
        "source": "none"
    }],
    "roleArn": "arn:aws:iam::111122223333:role/role-name",
    "tags": {}
}
```

Contoh berikut menggunakan Lakmus untuk menguji stres CPU pod di cluster Amazon EKS Kubernetes selama satu menit.

```
{
    "description": "Litmus CPU Hog",
    "targets": {
        "MyCluster": {
            "resourceType": "aws:eks:cluster",
            "resourceArns": [
                "arn:aws:eks:arn:aws::111122223333:cluster/cluster-id"
            ],
            "selectionMode": "ALL"
        }
    },
    "actions": {
        "MyAction": {
            "actionId": "aws:eks:inject-kubernetes-custom-resource",
            "parameters": {
                "maxDuration": "PT2M",
                "kubernetesApiVersion": "litmuschaos.io/v1alpha1",
                "kubernetesKind": "ChaosEngine",
                "kubernetesNamespace": "litmus",
                "kubernetesSpec": "{\"engineState\":\"active\",\"appinfo\":
{\"appns\":\"default\",\"applabel\":\"run=nginx\",\"appkind\":\"deployment\"},
\"chaosServiceAccount\":\"litmus-admin\",\"experiments\":[{\"name\":\"pod-cpu-hog
\",\"spec\":{\"components\":{\"env\":[{\"name\":\"TOTAL_CHAOS_DURATION\",\"value\":
\"60\"},{\"name\":\"CPU_CORES\",\"value\":\"1\"},{\"name\":\"PODS_AFFECTED_PERC\",
\"value\":\"100\"},{\"name\":\"CONTAINER_RUNTIME\",\"value\":\"docker\"},{\"name\":
\"SOCKET_PATH\",\"value\":\"/var/run/docker.sock\"}]},\"probe\":[]}}],\"annotationCheck
\":\"false\"}"
            },
            "targets": {
                "Cluster": "MyCluster"
            }
        }
    },
    "stopConditions": [{
        "source": "none"
    }],
    "roleArn": "arn:aws:iam::111122223333:role/role-name",
    "tags": {}
}
```

Mengelola eksperimen AWS FIS Anda

AWS FIS memungkinkan Anda untuk melakukan eksperimen injeksi kesalahan pada beban AWS kerja Anda. Untuk memulai, buat template eksperimen. Setelah Anda membuat template eksperimen, Anda dapat menggunakannya untuk memulai percobaan.

Eksperimen selesai ketika salah satu dari berikut ini terjadi:

- Semua tindakan dalam template selesai dengan sukses.
- Kondisi berhenti dipicu.
- Tindakan tidak dapat diselesaikan karena kesalahan. Misalnya, jika target tidak dapat ditemukan.
- Eksperimen dihentikan secara manual.

Anda tidak dapat melanjutkan eksperimen yang dihentikan atau gagal. Anda juga tidak dapat menjalankan kembali eksperimen yang telah selesai. Namun, Anda dapat memulai eksperimen baru dari templat eksperimen yang sama. Anda dapat memperbarui templat eksperimen secara opsional sebelum menentukannya lagi dalam eksperimen baru.

Tugas

- Memulai percobaan
- Lihat eksperimen Anda
- Tandai eksperimen
- Menghentikan sebuah percobaan
- Daftar target yang diselesaikan

Memulai percobaan

Anda memulai eksperimen dari templat eksperimen. Untuk informasi selengkapnya, lihat Memulai percobaan dari template.

Anda dapat menjadwalkan eksperimen Anda sebagai tugas satu kali atau tugas berulang menggunakan. Amazon EventBridge Untuk informasi selengkapnya, lihat Tutorial: Jadwalkan percobaan berulang.

Anda dapat memantau eksperimen Anda menggunakan salah satu fitur berikut:

- Lihat eksperimen Anda di konsol AWS FIS. Untuk informasi selengkapnya, lihat Lihat eksperimen Anda.
- Lihat CloudWatch metrik Amazon untuk sumber daya target dalam eksperimen Anda atau lihat metrik penggunaan AWS FIS. Untuk informasi selengkapnya, lihat Monitor menggunakan CloudWatch.
- Aktifkan pencatatan eksperimen untuk menangkap informasi terperinci tentang eksperimen Anda saat dijalankan. Untuk mengetahui informasi selengkapnya, lihat Pencatatan percobaan.

Lihat eksperimen Anda

Anda dapat melihat kemajuan eksperimen yang sedang berjalan, dan Anda dapat melihat eksperimen yang telah selesai, dihentikan, atau gagal.

Eksperimen yang dihentikan, selesai, dan gagal secara otomatis dihapus dari akun Anda setelah 120 hari.

Untuk melihat eksperimen menggunakan konsol

1. Buka konsol AWS FIS di https://console.aws.amazon.com/fis/.
2. Di panel navigasi, pilih Eksperimen.
3. Pilih ID Eksperimen eksperimen untuk membuka halaman detailnya.
4. Lakukan salah satu atau beberapa hal berikut:
   - Periksa Detail, Nyatakan status percobaan.
   - Pilih tab Tindakan untuk informasi tentang tindakan eksperimen.
   - Pilih tab Target untuk informasi tentang target eksperimen.
   - Pilih tab Timeline untuk representasi visual dari tindakan berdasarkan waktu mulai dan berakhir.

Untuk melihat eksperimen menggunakan CLI

Gunakan perintah list-experiments untuk mendapatkan daftar eksperimen, dan gunakan perintah get-experiment untuk mendapatkan informasi tentang eksperimen tertentu.

Percobaan menyatakan

Eksperimen dapat berada di salah satu keadaan berikut:

- Tertunda — Eksperimen sedang tertunda.
- memulai — Eksperimen sedang bersiap untuk memulai.
- berjalan — Eksperimen sedang berjalan.
- selesai — Semua tindakan dalam percobaan selesai dengan sukses.
- berhenti — Kondisi berhenti dipicu atau percobaan dihentikan secara manual.
- berhenti — Semua tindakan yang berjalan atau tertunda dalam percobaan dihentikan.
- gagal — Eksperimen gagal karena kesalahan, seperti izin yang tidak memadai atau sintaks yang salah.
- dibatalkan — Eksperimen dihentikan atau dicegah untuk memulai karena tuas pengaman yang terpasang.

Negara tindakan

Suatu tindakan dapat berada di salah satu keadaan berikut:

- Tertunda — Tindakan tertunda, baik karena percobaan belum dimulai atau tindakan akan dimulai nanti dalam percobaan.
- Memulai — Tindakan sedang bersiap untuk memulai.
- berlari — Aksi sedang berjalan.
- Selesai — Tindakan selesai dengan sukses.
- dibatalkan — Eksperimen berhenti sebelum tindakan dimulai.
- dilewati — Aksi telah dilewati.
- berhenti — Tindakan berhenti.
- berhenti — Semua tindakan yang berjalan atau tertunda dalam percobaan dihentikan.
- gagal — Tindakan gagal karena kesalahan klien, seperti izin yang tidak memadai atau sintaks yang salah.

Tandai eksperimen

Anda dapat menerapkan tag ke eksperimen untuk membantu Anda mengaturnya. Anda juga dapat menerapkan kebijakan IAM berbasis tag untuk mengontrol akses ke eksperimen.

Untuk menandai eksperimen menggunakan konsol

1. Buka konsol AWS FIS di https://console.aws.amazon.com/fis/.
2. Di panel navigasi, pilih Eksperimen.
3. Pilih eksperimen dan pilih Tindakan, Kelola tag.
4. Untuk menambahkan tag baru, pilih Tambahkan tag baru, dan tentukan kunci dan nilai.

   Untuk menghapus tag, pilih Hapus untuk tag.
5. Pilih Simpan.

Untuk menandai eksperimen menggunakan CLI

Gunakan perintah tag-resource.

Menghentikan sebuah percobaan

Anda dapat menghentikan eksperimen yang sedang berjalan kapan saja. Saat Anda menghentikan eksperimen, tindakan posting apa pun yang belum selesai untuk suatu tindakan akan diselesaikan sebelum eksperimen berhenti. Anda tidak dapat melanjutkan eksperimen yang dihentikan.

Untuk menghentikan percobaan menggunakan konsol

1. Buka konsol AWS FIS di https://console.aws.amazon.com/fis/.
2. Di panel navigasi, pilih Eksperimen.
3. Pilih eksperimen, dan pilih Hentikan eksperimen.
4. Di kotak dialog konfirmasi, pilih Hentikan eksperimen.

Untuk menghentikan percobaan menggunakan CLI

Gunakan perintah stop-experiment.

Daftar target yang diselesaikan

Anda dapat melihat informasi untuk target yang diselesaikan untuk eksperimen setelah resolusi target berakhir.

Untuk melihat target yang diselesaikan menggunakan konsol

1. Buka konsol AWS FIS di https://console.aws.amazon.com/fis/.
2. Di panel navigasi, pilih Eksperimen.
3. Pilih eksperimen, dan pilih Laporkan.
4. Lihat informasi target yang diselesaikan di bawah Sumber Daya.

Untuk melihat target yang diselesaikan menggunakan CLI

Gunakan perintah list-experiment-resolved-targets.

Tutorial untuk Layanan Injeksi AWS Kesalahan

Tutorial berikut menunjukkan cara membuat dan menjalankan eksperimen menggunakan AWS Fault Injection Service (AWS FIS).

Tutorial

- Tutorial: Uji contoh berhenti dan mulai menggunakan AWS FIS
- Tutorial: Jalankan stress CPU pada instance menggunakan AWS FIS
- Tutorial: Uji interupsi Instans Spot menggunakan FIS AWS
- Tutorial: Simulasikan acara konektivitas
- Tutorial: Jadwalkan percobaan berulang

Tutorial: Uji contoh berhenti dan mulai menggunakan AWS FIS

Anda dapat menggunakan AWS AWS Fault Injection Service (FIS) untuk menguji bagaimana aplikasi Anda menangani instance stop dan start. Gunakan tutorial ini untuk membuat template eksperimen yang menggunakan `aws:ec2:stop-instances` tindakan AWS FIS untuk menghentikan satu instance dan kemudian instance kedua.

Prasyarat

Untuk menyelesaikan tutorial ini, pastikan Anda melakukan hal berikut:

- Luncurkan dua EC2 contoh pengujian di akun Anda. Setelah meluncurkan instans, perhatikan kedua IDs instance tersebut.
- Buat peran IAM yang memungkinkan layanan AWS FIS untuk melakukan `aws:ec2:stop-instances` tindakan atas nama Anda. Untuk informasi selengkapnya, lihat Peran IAM untuk eksperimen AWS FIS.
- Pastikan Anda memiliki akses ke AWS FIS. Untuk informasi selengkapnya, lihat contoh kebijakan AWS FIS.

Langkah 1: Buat template percobaan

Buat template percobaan menggunakan konsol AWS FIS. Dalam template, Anda menentukan dua tindakan yang akan berjalan secara berurutan selama tiga menit masing-masing. Tindakan pertama

menghentikan salah satu contoh pengujian, yang dipilih AWS FIS secara acak. Tindakan kedua menghentikan kedua contoh pengujian.

Untuk membuat template percobaan

1. Buka konsol AWS FIS di https://console.aws.amazon.com/fis/.
2. Di panel navigasi, pilih Template eksperimen.
3. Pilih Buat template eksperimen.
4. Untuk Langkah 1, Tentukan detail template, lakukan hal berikut:
   a. Untuk Deskripsi dan nama, masukkan deskripsi untuk templat, seperti`Amazon S3 Network Disrupt Connectivity`.
   b. Pilih Berikutnya, dan pindah ke Langkah 2, Tentukan tindakan dan target.
5. Untuk Tindakan, lakukan hal berikut:
   a. Pilih Tambahkan tindakan.
   b. Masukkan nama untuk tindakan tersebut. Misalnya, masukkan **`stopOneInstance`**.
   c. Untuk tipe Action, pilih aws:ec2:stop-instance.
   d. Untuk Target, pertahankan target yang AWS dibuat FIS untuk Anda.
   e. Untuk parameter Tindakan, Mulai instance setelah durasi, tentukan 3 menit (PT3M).
   f. Pilih Simpan.
6. Untuk Target, lakukan langkah berikut:
   a. Pilih Edit untuk target yang AWS dibuat FIS secara otomatis untuk Anda pada langkah sebelumnya.
   b. Ganti nama default dengan nama yang lebih deskriptif. Misalnya, masukkan **`oneRandomInstance`**.
   c. Verifikasi bahwa tipe Resource adalah aws:ec2:instance.
   d. Untuk metode Target, pilih Resource IDs, dan kemudian pilih IDs dari dua contoh pengujian.
   e. Untuk mode Seleksi, pilih Hitung. Untuk Jumlah sumber daya, masukkan**1**.
   f. Pilih Simpan.
7. Pilih Tambah target dan lakukan hal berikut:
   a. Masukkan nama untuk target. Misalnya, masukkan **`bothInstances`**.
   b. Untuk tipe Resource, pilih aws:ec2:instance.

c. Untuk metode Target, pilih Resource IDs, dan kemudian pilih IDs dari dua contoh pengujian.

d. Untuk mode Seleksi, pilih Semua.

e. Pilih Simpan.

8. Dari bagian Tindakan, pilih Tambah tindakan. Lakukan hal-hal berikut:

a. Untuk Nama, masukkan nama untuk tindakan tersebut. Misalnya, masukkan **`stopBothInstances`**.

b. Untuk tipe Action, pilih aws:ec2:stop-instance.

c. Untuk Mulai setelah, pilih tindakan pertama yang Anda tambahkan (**`stopOneInstance`**).

d. Untuk Target, pilih target kedua yang Anda tambahkan (**`bothInstances`**).

e. Untuk parameter Tindakan, Mulai instance setelah durasi, tentukan 3 menit (PT3M).

f. Pilih Simpan.

9. Pilih Berikutnya untuk pindah ke Langkah 3, Konfigurasikan akses layanan.

10. Untuk Akses Layanan, pilih Gunakan peran IAM yang ada, lalu pilih peran IAM yang Anda buat seperti yang dijelaskan dalam prasyarat untuk tutorial ini. Jika peran Anda tidak ditampilkan, verifikasi bahwa ia memiliki hubungan kepercayaan yang diperlukan. Untuk informasi selengkapnya, lihat the section called “Peran percobaan”.

11. Pilih Berikutnya untuk pindah ke Langkah 4, Konfigurasikan pengaturan opsional.

12. (Opsional) Untuk Tag, pilih Tambahkan tag baru dan tentukan kunci tag dan nilai tag. Tag yang Anda tambahkan diterapkan ke template eksperimen Anda, bukan eksperimen yang dijalankan menggunakan template.

13. Pilih Berikutnya untuk pindah ke Langkah 5, Tinjau dan buat.

14. Tinjau template dan pilih Buat template eksperimen. Ketika diminta untuk konfirmasi, masukkan`create`, Lalu pilih Buat template percobaan.

(Opsional) Untuk melihat template eksperimen JSON

Pilih tab Ekspor. Berikut ini adalah contoh dari JSON yang dibuat oleh prosedur konsol sebelumnya.

```
{
    "description": "Test instance stop and start",
    "targets": {
        "bothInstances": {
            "resourceType": "aws:ec2:instance",
            "resourceArns": [
```

```
                "arn:aws:ec2:region:123456789012:instance/instance_id_1",
                "arn:aws:ec2:region:123456789012:instance/instance_id_2"
            ],
            "selectionMode": "ALL"
        },
        "oneRandomInstance": {
            "resourceType": "aws:ec2:instance",
            "resourceArns": [
                "arn:aws:ec2:region:123456789012:instance/instance_id_1",
                "arn:aws:ec2:region:123456789012:instance/instance_id_2"
            ],
            "selectionMode": "COUNT(1)"
        }
    },
    "actions": {
        "stopBothInstances": {
            "actionId": "aws:ec2:stop-instances",
            "parameters": {
                "startInstancesAfterDuration": "PT3M"
            },
            "targets": {
                "Instances": "bothInstances"
            },
            "startAfter": [
                "stopOneInstance"
            ]
        },
        "stopOneInstance": {
            "actionId": "aws:ec2:stop-instances",
            "parameters": {
                "startInstancesAfterDuration": "PT3M"
            },
            "targets": {
                "Instances": "oneRandomInstance"
            }
        }
    },
    "stopConditions": [
        {
            "source": "none"
        }
    ],
    "roleArn": "arn:aws:iam::123456789012:role/AllowFISEC2Actions",
    "tags": {}
```

```
}
```

Langkah 2: Mulai percobaan

Setelah selesai membuat templat eksperimen, Anda dapat menggunakannya untuk memulai eksperimen.

Untuk memulai percobaan

1. Anda harus berada di halaman detail untuk template eksperimen yang baru saja Anda buat. Jika tidak, pilih Templat eksperimen lalu pilih ID templat eksperimen untuk membuka halaman detail.
2. Pilih Mulai percobaan.
3. (Opsional) Untuk menambahkan tag ke eksperimen Anda, pilih Tambahkan tag baru dan masukkan kunci tag dan nilai tag.
4. Pilih Mulai percobaan. Saat diminta konfirmasi, masukkan **`start`** dan pilih Mulai eksperimen.

Langkah 3: Lacak kemajuan eksperimen

Anda dapat melacak kemajuan eksperimen yang sedang berjalan hingga percobaan selesai, dihentikan, atau gagal.

Untuk melacak kemajuan eksperimen

1. Anda harus berada di halaman detail untuk eksperimen yang baru saja Anda mulai. Jika tidak, pilih Eksperimen lalu pilih ID eksperimen untuk membuka halaman detail.
2. Untuk melihat status percobaan, periksa Status di panel Detail. Untuk informasi lebih lanjut, lihat status eksperimen.
3. Ketika keadaan percobaan sedang berjalan, lanjutkan ke langkah berikutnya.

Langkah 4: Verifikasi hasil percobaan

Anda dapat memverifikasi bahwa instance dihentikan dan dimulai oleh eksperimen seperti yang diharapkan.

Untuk memverifikasi hasil percobaan

1. Buka EC2 konsol Amazon https://console.aws.amazon.com/ec2/di tab atau jendela browser baru. Ini memungkinkan Anda untuk terus melacak kemajuan percobaan di konsol AWS FIS sambil melihat hasil percobaan di EC2 konsol Amazon.
2. Di panel navigasi, pilih Instans.
3. Saat status tindakan pertama berubah dari Pending ke Running (konsol AWS FIS), status salah satu instance target berubah dari Running ke Stopped (EC2 konsol Amazon).
4. Setelah tiga menit, status tindakan pertama berubah menjadi Selesai, status tindakan kedua berubah menjadi Running, dan status instance target lainnya berubah menjadi Berhenti.
5. Setelah tiga menit, status tindakan kedua berubah menjadi Selesai, status instance target berubah menjadi Running, dan status eksperimen berubah menjadi Selesai.

Langkah 5: Bersihkan

Jika Anda tidak lagi memerlukan EC2 instance pengujian yang Anda buat untuk eksperimen ini, Anda dapat menghentikannya.

Untuk mengakhirkan instans

1. Buka EC2 konsol Amazon di https://console.aws.amazon.com/ec2/.
2. Di panel navigasi, pilih Instans.
3. Pilih kedua instance pengujian dan pilih status Instance, Terminate instance.
4. Saat diminta konfirmasi, pilih Akhiri.

Jika Anda tidak lagi membutuhkan templat percobaan, Anda dapat menghapusnya.

Untuk menghapus template percobaan menggunakan konsol AWS FIS

1. Buka konsol AWS FIS di https://console.aws.amazon.com/fis/.
2. Di panel navigasi, pilih Template eksperimen.
3. Pilih template eksperimen, lalu pilih Tindakan, Hapus template eksperimen.
4. Saat diminta konfirmasi, masukkan **`delete`** lalu pilih Hapus templat eksperimen.

Tutorial: Jalankan stress CPU pada instance menggunakan AWS FIS

Anda dapat menggunakan AWS AWS Fault Injection Service (FIS) untuk menguji bagaimana aplikasi Anda menangani stres CPU. Gunakan tutorial ini untuk membuat template eksperimen yang menggunakan AWS FIS untuk menjalankan dokumen SSM pra-konfigurasi yang menjalankan stres CPU pada sebuah instance. Tutorial menggunakan kondisi berhenti untuk menghentikan percobaan ketika pemanfaatan CPU dari instance melebihi ambang batas yang dikonfigurasi.

Untuk informasi selengkapnya, lihat the section called “Dokumen SSM AWS FIS yang telah dikonfigurasi sebelumnya”.

Prasyarat

Sebelum Anda dapat menggunakan AWS FIS untuk menjalankan stress CPU, selesaikan prasyarat berikut.

Membuat peran IAM

Buat peran dan lampirkan kebijakan yang memungkinkan AWS FIS untuk menggunakan `aws:ssm:send-command` tindakan atas nama Anda. Untuk informasi selengkapnya, lihat Peran IAM untuk eksperimen AWS FIS.

Verifikasi akses ke AWS FIS

Pastikan Anda memiliki akses ke AWS FIS. Untuk informasi selengkapnya, lihat contoh kebijakan AWS FIS.

Siapkan EC2 contoh pengujian

- Luncurkan EC2 instance menggunakan Amazon Linux 2 atau Ubuntu, seperti yang dipersyaratkan oleh dokumen SSM yang telah dikonfigurasi sebelumnya.
- Instans harus dikelola oleh SSM. Untuk memverifikasi bahwa instans dikelola oleh SSM, buka konsol Fleet Manager. Jika instans tidak dikelola oleh SSM, verifikasi bahwa Agen SSM diinstal dan instans memiliki peran IAM terlampir dengan kebijakan Amazon. SSMManaged InstanceCore Untuk memverifikasi Agen SSM yang diinstal, sambungkan ke instans Anda dan jalankan perintah berikut.

  Amazon Linux 2

```
yum info amazon-ssm-agent
```

Ubuntu

```
apt list amazon-ssm-agent
```

- Aktifkan pemantauan terperinci untuk contoh ini. Ini memberikan data dalam periode 1 menit, dengan biaya tambahan. Pilih instans dan pilih Tindakan, Pantau dan pemecahan masalah, Kelola pemantauan terperinci.

Langkah 1: Buat CloudWatch alarm untuk kondisi berhenti

Konfigurasikan CloudWatch alarm sehingga Anda dapat menghentikan percobaan jika pemanfaatan CPU melebihi ambang batas yang Anda tentukan. Prosedur berikut menetapkan ambang batas hingga 50% pemanfaatan CPU untuk instance target. Untuk informasi selengkapnya, lihat Hentikan kondisi.

Untuk membuat alarm yang menunjukkan kapan pemanfaatan CPU melebihi ambang batas

1. Buka EC2 konsol Amazon di https://console.aws.amazon.com/ec2/.
2. Di panel navigasi, pilih Instans.
3. Pilih instance target dan pilih Actions, Monitor dan troubleshoot, Manage CloudWatch alarm.
4. Untuk pemberitahuan Alarm, gunakan sakelar untuk mematikan notifikasi Amazon SNS.
5. Untuk ambang Alarm, gunakan pengaturan berikut:
   - Kelompokkan sampel berdasarkan: Maksimum
   - Jenis data untuk sampel: Pemanfaatan CPU
   - Persen: **`50`**
   - Periode: **`1 Minute`**
6. Setelah selesai mengonfigurasi alarm, pilih Buat.

Langkah 2: Buat template percobaan

Buat template percobaan menggunakan konsol AWS FIS. Dalam template, Anda menentukan tindakan berikut untuk dijalankan: aws:ssm:send-command/ -cpu-stress AWSFIS-Run.

Untuk membuat template percobaan

1. Buka konsol AWS FIS di https://console.aws.amazon.com/fis/.
2. Di panel navigasi, pilih Template eksperimen.
3. Pilih Buat template eksperimen.
4. Untuk Langkah 1, Tentukan detail template, lakukan hal berikut:
   a. Untuk Deskripsi dan nama, masukkan deskripsi untuk templat.
   b. Pilih Berikutnya, dan pindah ke Langkah 2, Tentukan tindakan dan target.
5. Untuk Tindakan, lakukan hal berikut:
   a. Pilih Tambahkan tindakan.
   b. Masukkan nama untuk tindakan tersebut. Misalnya, masukkan **`runCpuStress`**.
   c. Untuk tipe Action, pilih AWSFIS-Runaws:ssm:send-command/ -CPU-stress. Ini secara otomatis menambahkan ARN dokumen SSM ke Dokumen ARN.
   d. Untuk Target, pertahankan target yang AWS dibuat FIS untuk Anda.
   e. Untuk parameter Tindakan, Parameter dokumen, masukkan yang berikut ini:

      ```
      {"DurationSeconds":"120"}
      ```

   f. Untuk parameter Tindakan, Durasi, tentukan 5 menit (PT5M).
   g. Pilih Simpan.
6. Untuk Target, lakukan langkah berikut:
   a. Pilih Edit untuk target yang AWS dibuat FIS secara otomatis untuk Anda pada langkah sebelumnya.
   b. Ganti nama default dengan nama yang lebih deskriptif. Misalnya, masukkan **`testInstance`**.
   c. Verifikasi bahwa tipe Resource adalah aws:ec2:instance.
   d. Untuk metode Target, pilih Resource IDs, lalu pilih ID dari instance pengujian.
   e. Untuk mode Seleksi, pilih Semua.
   f. Pilih Simpan.
7. Pilih Berikutnya untuk pindah ke Langkah 3, Konfigurasikan akses layanan.
8. Untuk Akses Layanan, pilih Gunakan peran IAM yang ada, lalu pilih peran IAM yang Anda buat seperti yang dijelaskan dalam prasyarat untuk tutorial ini. Jika peran Anda tidak ditampilkan,

verifikasi bahwa ia memiliki hubungan kepercayaan yang diperlukan. Untuk informasi selengkapnya, lihat the section called “Peran percobaan”.

9. Pilih Berikutnya untuk pindah ke Langkah 4, Konfigurasikan pengaturan opsional.
10. Untuk kondisi Stop, pilih CloudWatch alarm yang Anda buat di Langkah 1.
11. (Opsional) Untuk Tag, pilih Tambahkan tag baru dan tentukan kunci tag dan nilai tag. Tag yang Anda tambahkan diterapkan ke template eksperimen Anda, bukan eksperimen yang dijalankan menggunakan template.
12. Pilih Berikutnya untuk pindah ke Langkah 5, Tinjau dan buat.
13. Tinjau template dan pilih Buat template eksperimen. Ketika diminta untuk konfirmasi, masukkan`create`, Lalu pilih Buat template percobaan.

(Opsional) Untuk melihat template eksperimen JSON

Pilih tab Ekspor. Berikut ini adalah contoh dari JSON yang dibuat oleh prosedur konsol sebelumnya.

```
{
    "description": "Test CPU stress predefined SSM document",
    "targets": {
        "testInstance": {
            "resourceType": "aws:ec2:instance",
            "resourceArns": [
                "arn:aws:ec2:region:123456789012:instance/instance_id"
            ],
            "selectionMode": "ALL"
        }
    },
    "actions": {
        "runCpuStress": {
            "actionId": "aws:ssm:send-command",
            "parameters": {
                "documentArn": "arn:aws:ssm:region::document/AWSFIS-Run-CPU-Stress",
                "documentParameters": "{\"DurationSeconds\":\"120\"}",
                "duration": "PT5M"
            },
            "targets": {
                "Instances": "testInstance"
            }
        }
    },
    "stopConditions": [
```

```
        {
            "source": "aws:cloudwatch:alarm",
            "value": "arn:aws:cloudwatch:region:123456789012:alarm:awsec2-instance_id-
GreaterThanOrEqualToThreshold-CPUUtilization"
        }
    ],
    "roleArn": "arn:aws:iam::123456789012:role/AllowFISSSMActions",
    "tags": {}
}
```

Langkah 3: Mulai percobaan

Setelah selesai membuat templat eksperimen, Anda dapat menggunakannya untuk memulai eksperimen.

Untuk memulai percobaan

1. Anda harus berada di halaman detail untuk template eksperimen yang baru saja Anda buat. Jika tidak, pilih Templat eksperimen lalu pilih ID templat eksperimen untuk membuka halaman detail.
2. Pilih Mulai percobaan.
3. (Opsional) Untuk menambahkan tag ke eksperimen Anda, pilih Tambahkan tag baru dan masukkan kunci tag dan nilai tag.
4. Pilih Mulai percobaan. Saat diminta mengonfirmasi, pilih **`start`**. Pilih Mulai percobaan.

Langkah 4: Lacak kemajuan eksperimen

Anda dapat melacak kemajuan eksperimen yang sedang berjalan hingga eksperimen selesai, berhenti, atau gagal.

Untuk melacak kemajuan eksperimen

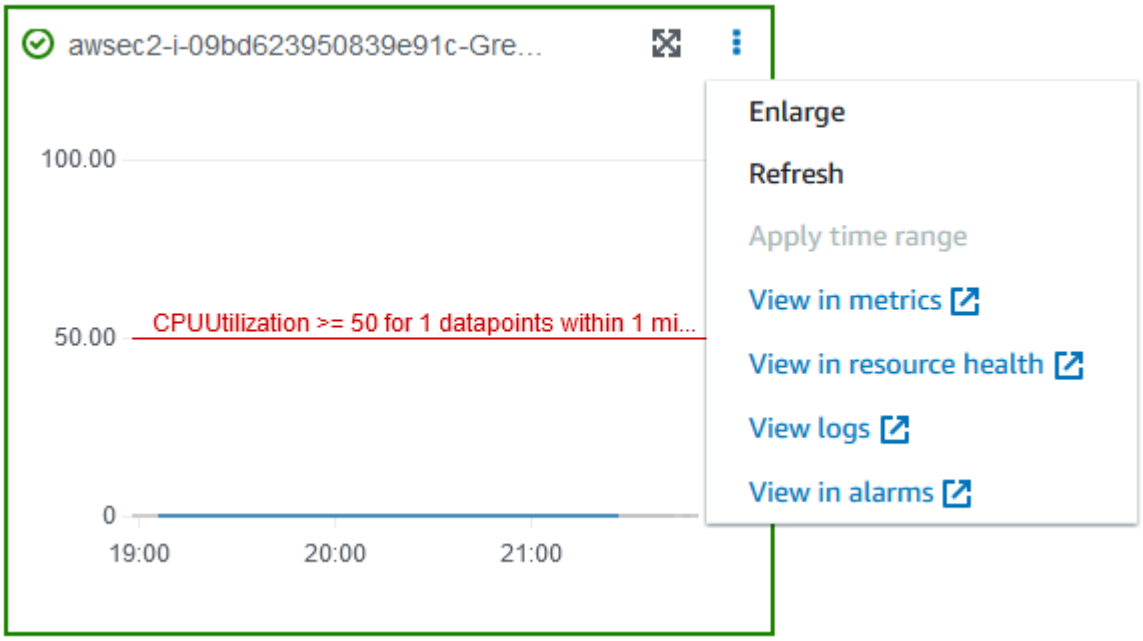
1. Anda harus berada di halaman detail untuk eksperimen yang baru saja Anda mulai. Jika tidak, pilih Eksperimen lalu pilih ID eksperimen untuk membuka halaman detail eksperimen.
2. Untuk melihat status percobaan, periksa Status di panel Detail. Untuk informasi lebih lanjut, lihat status eksperimen.
3. Saat status percobaan sedang berjalan, lanjutkan ke langkah berikutnya.

Langkah 5: Verifikasi hasil percobaan

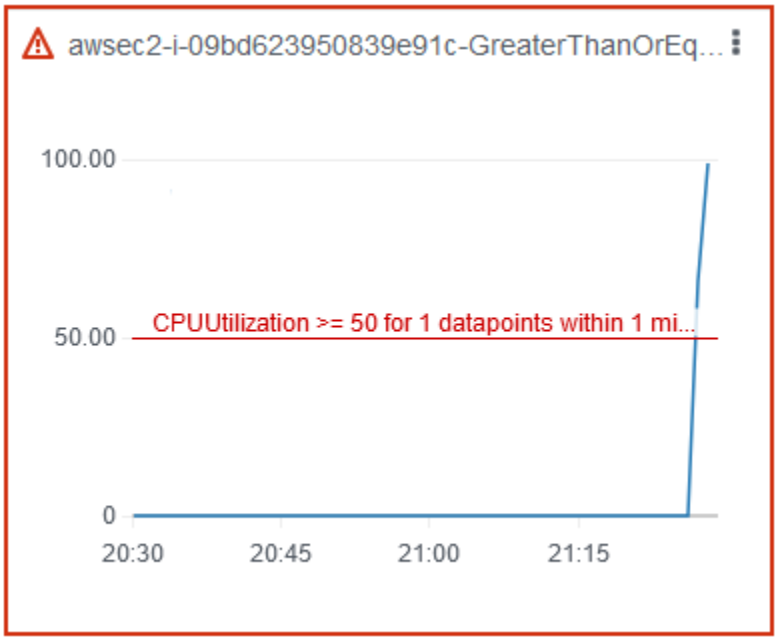
Anda dapat memantau pemanfaatan CPU instance Anda saat percobaan sedang berjalan. Ketika pemanfaatan CPU mencapai ambang batas, alarm dipicu dan percobaan dihentikan oleh kondisi berhenti.

Untuk memverifikasi hasil percobaan

1. Pilih tab Stop conditions. Batas hijau dan ikon tanda centang hijau menunjukkan bahwa keadaan awal alarm adalah`OK`. Garis merah menunjukkan ambang alarm. Jika Anda lebih suka grafik yang lebih detail, pilih Perbesar dari menu widget.



2. Ketika pemanfaatan CPU melebihi ambang batas, batas merah dan ikon tanda seru merah di tab Stop conditions menunjukkan bahwa status alarm berubah menjadi. `ALARM` Di panel Detail, status percobaan Dihentikan. Jika Anda memilih status, pesan yang ditampilkan adalah “Eksperimen dihentikan oleh kondisi berhenti”.



3. Ketika pemanfaatan CPU menurun di bawah ambang batas, batas hijau dan ikon tanda centang hijau menunjukkan bahwa status alarm berubah menjadi. OK
4. (Opsional) Pilih Lihat di alarm dari menu widget. Ini membuka halaman detail alarm di CloudWatch konsol, di mana Anda bisa mendapatkan detail lebih lanjut tentang alarm atau mengedit pengaturan alarm.

Langkah 6: Bersihkan

Jika Anda tidak lagi memerlukan EC2 instance pengujian yang Anda buat untuk eksperimen ini, Anda dapat menghentikannya.

Untuk menghentikan instans

1. Buka EC2 konsol Amazon di https://console.aws.amazon.com/ec2/.
2. Di panel navigasi, pilih Instans.
3. Pilih contoh pengujian dan pilih status Instance, Terminate instance.
4. Saat diminta konfirmasi, pilih Akhiri.

Jika Anda tidak lagi membutuhkan templat percobaan, Anda dapat menghapusnya.

Untuk menghapus template percobaan menggunakan konsol AWS FIS

1. Buka konsol AWS FIS di https://console.aws.amazon.com/fis/.
2. Di panel navigasi, pilih Template eksperimen.
3. Pilih template eksperimen, lalu pilih Tindakan, Hapus template eksperimen.
4. Saat diminta konfirmasi, masukkan **`delete`** lalu pilih Hapus templat eksperimen.

Tutorial: Uji interupsi Instans Spot menggunakan FIS AWS

Instans Spot menggunakan EC2 kapasitas cadangan yang tersedia, hingga diskon 90% dibandingkan dengan harga On-Demand. Namun, Amazon EC2 dapat mengganggu Instans Spot Anda saat membutuhkan kapasitas kembali. Saat menggunakan Instans Spot, Anda harus siap menghadapi potensi gangguan. Untuk informasi selengkapnya, lihat Interupsi Instans Spot di EC2 Panduan Pengguna Amazon.

Anda dapat menggunakan AWS AWS Fault Injection Service (FIS) untuk menguji bagaimana aplikasi Anda menangani interupsi Instans Spot. Gunakan tutorial ini untuk membuat template eksperimen yang menggunakan `aws:ec2:send-spot-instance-interruptions` tindakan AWS FIS untuk mengganggu salah satu Instans Spot Anda.

Atau, untuk memulai eksperimen menggunakan EC2 konsol Amazon, lihat Memulai interupsi Instans Spot di Panduan Pengguna Amazon EC2 .

Prasyarat

Sebelum Anda dapat menggunakan AWS FIS untuk menginterupsi Instance Spot, selesaikan prasyarat berikut.

1. Membuat peran IAM

Buat peran dan lampirkan kebijakan yang memungkinkan AWS FIS untuk melakukan `aws:ec2:send-spot-instance-interruptions` tindakan atas nama Anda. Untuk informasi selengkapnya, lihat Peran IAM untuk eksperimen AWS FIS.

2. Verifikasi akses ke AWS FIS

Pastikan Anda memiliki akses ke AWS FIS. Untuk informasi selengkapnya, lihat contoh kebijakan AWS FIS.

3. (Opsional) Buat permintaan Instans Spot

Jika Anda ingin Instance Spot baru digunakan untuk eksperimen ini, gunakan perintah run-instance untuk meminta Instance Spot. Defaultnya adalah menghentikan Instans Spot yang terputus. Jika Anda menyetel perilaku interupsi`stop`, Anda juga harus menyetel jenisnya. `persistent` Untuk tutorial ini, jangan atur perilaku interupsi`hibernate`, karena proses hibernasi segera dimulai.

```
aws ec2 run-instances \
    --image-id ami-0ab193018fEXAMPLE \
    --instance-type "t2.micro" \
    --count 1 \
    --subnet-id subnet-1234567890abcdef0 \
    --security-group-ids sg-111222333444aaab \
    --instance-market-options file://spot-options.json \
    --query Instances[*].InstanceId
```

Berikut ini adalah contoh file `spot-options.json`.

```
{
    "MarketType": "spot",
    "SpotOptions": {
        "SpotInstanceType": "persistent",
        "InstanceInterruptionBehavior": "stop"
    }
}
```

`--query`Opsi dalam perintah contoh membuatnya sehingga perintah hanya mengembalikan ID instance dari Instance Spot. Berikut ini adalah output contoh.

```
[
    "i-0abcdef1234567890"
]
```

4. Tambahkan tag sehingga AWS FIS dapat mengidentifikasi Instance Spot target

Gunakan perintah create-tags untuk menambahkan tag Name=interruptMe ke Instance Spot target Anda.

```
aws ec2 create-tags \
    --resources i-0abcdef1234567890 \
    --tags Key=Name,Value=interruptMe
```

Langkah 1: Buat template percobaan

Buat template percobaan menggunakan konsol AWS FIS. Dalam template, Anda menentukan tindakan yang akan berjalan. Tindakan menginterupsi Instance Spot dengan tag yang ditentukan. Jika ada lebih dari satu Instance Spot dengan tag, AWS FIS memilih salah satunya secara acak.

Untuk membuat template percobaan

1. Buka konsol AWS FIS di https://console.aws.amazon.com/fis/.
2. Di panel navigasi, pilih Template eksperimen.
3. Pilih Buat template eksperimen.
4. Untuk Langkah 1, Tentukan detail template, lakukan hal berikut:
    a. Untuk Deskripsi dan nama, masukkan deskripsi dan nama untuk templat.
    b. Pilih Berikutnya, dan pindah ke Langkah 2, Tentukan tindakan dan target.
5. Untuk Tindakan, lakukan hal berikut:
    a. Pilih Tambahkan tindakan.
    b. Masukkan nama untuk tindakan tersebut. Misalnya, masukkan **`interruptSpotInstance`**.
    c. Untuk tipe Action, pilih aws:ec2:. send-spot-instance-interruptions
    d. Untuk Target, pertahankan target yang AWS dibuat FIS untuk Anda.
    e. Untuk parameter Tindakan, Durasi sebelum interupsi, tentukan 2 Menit (PT2M).
    f. Pilih Simpan.
6. Untuk Target, lakukan langkah berikut:
    a. Pilih Edit untuk target yang AWS dibuat FIS secara otomatis untuk Anda pada langkah sebelumnya.
    b. Ganti nama default dengan nama yang lebih deskriptif. Misalnya, masukkan **`oneSpotInstance`**.
    c. Verifikasi bahwa tipe Resource adalah aws:ec2:spot-instance.
    d. Untuk metode Target, pilih Tag sumber daya, filter, dan parameter.
    e. Untuk tag Resource, pilih Tambahkan tag baru, dan masukkan kunci tag dan nilai tag. Gunakan tag yang Anda tambahkan ke Instance Spot untuk menginterupsi, seperti yang dijelaskan dalam Prasyarat untuk tutorial ini.

f. Untuk filter Sumber daya pilih Tambahkan filter baru dan masukkan **`State.Name`** sebagai jalur dan **`running`** sebagai nilai.

g. Untuk mode Seleksi, pilih Hitung. Untuk Jumlah sumber daya, masukkan**`1`**.

h. Pilih Simpan.

7. Pilih Berikutnya untuk pindah ke Langkah 3, Konfigurasikan akses layanan.
8. Untuk Akses Layanan, pilih Gunakan peran IAM yang ada, lalu pilih peran IAM yang Anda buat seperti yang dijelaskan dalam prasyarat untuk tutorial ini. Jika peran Anda tidak ditampilkan, verifikasi bahwa ia memiliki hubungan kepercayaan yang diperlukan. Untuk informasi selengkapnya, lihat the section called “Peran percobaan”.
9. Pilih Berikutnya untuk pindah ke Langkah 4, Konfigurasikan pengaturan opsional.
10. (Opsional) Untuk Tag, pilih Tambahkan tag baru dan tentukan kunci tag dan nilai tag. Tag yang Anda tambahkan diterapkan ke template eksperimen Anda, bukan eksperimen yang dijalankan menggunakan template.
11. Pilih Berikutnya untuk pindah ke Langkah 5, Tinjau dan buat.
12. Tinjau template dan pilih Buat template eksperimen. Ketika diminta untuk konfirmasi, masukkan`create`, Lalu pilih Buat template percobaan.

(Opsional) Untuk melihat template eksperimen JSON

Pilih tab Ekspor. Berikut ini adalah contoh dari JSON yang dibuat oleh prosedur konsol sebelumnya.

```
{
    "description": "Test Spot Instance interruptions",
    "targets": {
        "oneSpotInstance": {
            "resourceType": "aws:ec2:spot-instance",
            "resourceTags": {
                "Name": "interruptMe"
            },
            "filters": [
                {
                    "path": "State.Name",
                    "values": [
                        "running"
                    ]
                }
            ],
```

```
            "selectionMode": "COUNT(1)"
        }
    },
    "actions": {
        "interruptSpotInstance": {
            "actionId": "aws:ec2:send-spot-instance-interruptions",
            "parameters": {
                "durationBeforeInterruption": "PT2M"
            },
            "targets": {
                "SpotInstances": "oneSpotInstance"
            }
        }
    },
    "stopConditions": [
        {
            "source": "none"
        }
    ],
    "roleArn": "arn:aws:iam::123456789012:role/AllowFISSpotInterruptionActions",
    "tags": {
        "Name": "my-template"
    }
}
```

Langkah 2: Mulai percobaan

Setelah selesai membuat templat eksperimen, Anda dapat menggunakannya untuk memulai eksperimen.

Untuk memulai percobaan

1. Anda harus berada di halaman detail untuk template eksperimen yang baru saja Anda buat. Jika tidak, pilih Templat eksperimen lalu pilih ID templat eksperimen untuk membuka halaman detail.
2. Pilih Mulai percobaan.
3. (Opsional) Untuk menambahkan tag ke eksperimen Anda, pilih Tambahkan tag baru dan masukkan kunci tag dan nilai tag.
4. Pilih Mulai percobaan. Saat diminta konfirmasi, masukkan **`start`** dan pilih Mulai eksperimen.

Langkah 3: Lacak kemajuan eksperimen

Anda dapat melacak kemajuan eksperimen yang sedang berjalan hingga percobaan selesai, dihentikan, atau gagal.

Untuk melacak kemajuan eksperimen

1. Anda harus berada di halaman detail untuk eksperimen yang baru saja Anda mulai. Jika tidak, pilih Eksperimen lalu pilih ID eksperimen untuk membuka halaman detail.
2. Untuk melihat status percobaan, periksa Status di panel Detail. Untuk informasi lebih lanjut, lihat status eksperimen.
3. Ketika keadaan percobaan sedang berjalan, lanjutkan ke langkah berikutnya.

Langkah 4: Verifikasi hasil percobaan

Ketika tindakan untuk percobaan ini selesai, berikut ini terjadi:

- Instance Spot target menerima rekomendasi penyeimbangan ulang instans.
- Pemberitahuan interupsi Instans Spot dikeluarkan dua menit sebelum Amazon EC2 menghentikan atau menghentikan instans Anda.
- Setelah dua menit, Instans Spot dihentikan atau dihentikan.
- Instance Spot yang dihentikan oleh AWS FIS tetap dihentikan sampai Anda memulai ulang.

Untuk memverifikasi bahwa instance terputus oleh percobaan

1. Buka EC2 konsol Amazon di https://console.aws.amazon.com/ec2/.
2. Dari panel navigasi, buka Permintaan Spot dan Instans di tab atau jendela peramban yang terpisah.
3. Untuk Permintaan Spot, pilih permintaan Instans Spot. Status awal adalah `fulfilled`. Setelah percobaan selesai, status berubah sebagai berikut:
   - `terminate`- Status berubah menjadi`instance-terminated-by-experiment`.
   - `stop`- Status berubah menjadi `marked-for-stop-by-experiment` dan kemudian`instance-stopped-by-experiment`.
4. Untuk Instans, pilih Instans Spot. Status awal adalah `Running`. Dua menit setelah Anda menerima pemberitahuan interupsi Instans Spot, statusnya berubah sebagai berikut:

- `stop`- Status berubah menjadi `Stopping` dan kemudian`Stopped`.
- `terminate`- Status berubah menjadi `Shutting-down` dan kemudian`Terminated`.

Langkah 5: Bersihkan

Jika Anda membuat Instance Spot pengujian untuk eksperimen ini dengan perilaku interupsi `stop` dan Anda tidak lagi membutuhkannya, Anda dapat membatalkan permintaan Instans Spot dan menghentikan Instans Spot.

Untuk membatalkan permintaan dan menghentikan instance menggunakan AWS CLI

1. Gunakan cancel-spot-instance-requestsperintah untuk membatalkan permintaan Instans Spot.

```
aws ec2 cancel-spot-instance-requests --spot-instance-request-ids sir-ksie869j
```

2. Gunakan perintah terminate-instance untuk mengakhiri instance.

```
aws ec2 terminate-instances --instance-ids i-0abcdef1234567890
```

Jika Anda tidak lagi membutuhkan templat percobaan, Anda dapat menghapusnya.

Untuk menghapus template percobaan menggunakan konsol AWS FIS

1. Buka konsol AWS FIS di https://console.aws.amazon.com/fis/.
2. Di panel navigasi, pilih Template eksperimen.
3. Pilih template eksperimen, lalu pilih Tindakan, Hapus template eksperimen.
4. Saat diminta konfirmasi, masukkan **`delete`** lalu pilih Hapus templat eksperimen.

Tutorial: Simulasikan acara konektivitas

Anda dapat menggunakan AWS AWS Fault Injection Service (FIS) untuk mensimulasikan berbagai peristiwa konektivitas. AWS FIS mensimulasikan peristiwa konektivitas dengan memblokir koneksi jaringan dengan salah satu cara berikut:

- `all`— Menyangkal semua lalu lintas yang masuk dan keluar dari subnet. Perhatikan bahwa opsi ini memungkinkan lalu lintas intra-subnet, termasuk lalu lintas ke dan dari antarmuka jaringan di subnet.
- `availability-zone`— Menolak lalu lintas intra-VPC ke dan dari subnet di Availability Zone lainnya.
- `dynamodb`— Menolak lalu lintas ke dan dari titik akhir Regional untuk DynamoDB di Wilayah saat ini.
- `prefix-list`— Menolak lalu lintas ke dan dari daftar awalan yang ditentukan.
- `s3`— Menolak lalu lintas ke dan dari titik akhir Regional untuk Amazon S3 di Wilayah saat ini.
- `vpc`— Menolak lalu lintas masuk dan keluar dari VPC.

Gunakan tutorial ini untuk membuat template eksperimen yang menggunakan `aws:network:disrupt-connectivity` tindakan AWS FIS untuk memperkenalkan kehilangan konektivitas dengan Amazon S3 di subnet target.

Topik

- Prasyarat
- Langkah 1: Buat template eksperimen AWS FIS
- Langkah 2: Ping ke titik akhir Amazon S3
- Langkah 3: Mulai eksperimen AWS FIS Anda
- Langkah 4: Lacak kemajuan eksperimen AWS FIS Anda
- Langkah 5: Verifikasi gangguan jaringan Amazon S3
- Langkah 5: Bersihkan

Prasyarat

Sebelum memulai tutorial ini, Anda memerlukan peran dengan izin yang sesuai di Anda Akun AWS, dan EC2 contoh uji Amazon:

Peran dengan izin di Akun AWS

Buat peran dan lampirkan kebijakan yang memungkinkan AWS FIS untuk melakukan `aws:network:disrupt-connectivity` tindakan atas nama Anda.

Peran IAM Anda memerlukan kebijakan berikut:

- AWSFaultInjectionSimulatorNetworkAccess— Memberikan izin layanan AWS FIS di EC2 jaringan Amazon dan layanan lain yang diperlukan untuk melakukan tindakan AWS FIS yang terkait dengan infrastruktur jaringan.

> **Note**
>
> Untuk mempermudah, tutorial ini menggunakan kebijakan AWS terkelola. Untuk penggunaan produksi, sebaiknya Anda hanya memberikan izin minimum yang diperlukan untuk kasus penggunaan Anda.
> Untuk informasi selengkapnya tentang cara membuat peran IAM, lihat peran IAM untuk eksperimen AWS FIS (AWS CLI) atau Membuat peran IAM (konsol) di Panduan Pengguna IAM.

EC2 Contoh uji Amazon

Luncurkan dan sambungkan ke EC2 instance Amazon pengujian. Anda dapat menggunakan tutorial berikut untuk meluncurkan dan menyambung ke EC2 instans Amazon: Tutorial: Memulai instans Amazon EC2 Linux di Panduan EC2 Pengguna Amazon.

Langkah 1: Buat template eksperimen AWS FIS

Buat template percobaan dengan menggunakan AWS FIS AWS Management Console. Template AWS FIS terdiri dari tindakan, target, kondisi berhenti, dan peran eksperimen. Untuk informasi selengkapnya tentang cara kerja template, lihat Template eksperimen untuk AWS FIS.

Sebelum Anda mulai, pastikan Anda memiliki yang berikut ini siap:

- Peran IAM dengan izin yang benar.
- EC2 Contoh Amazon.
- ID subnet dari EC2 instans Amazon Anda.

Untuk membuat template percobaan

1. Buka konsol AWS FIS di https://console.aws.amazon.com/fis/.
2. Di panel navigasi kiri, pilih Template eksperimen.

3. Pilih Buat template eksperimen.
4. Untuk Langkah 1, Tentukan detail template, lakukan hal berikut:
   a. Untuk Deskripsi dan nama, masukkan deskripsi untuk templat, seperti`Amazon S3 Network Disrupt Connectivity`.
   b. Pilih Berikutnya, dan pindah ke Langkah 2, Tentukan tindakan dan target.
5. Di bawah Tindakan, pilih Tambah tindakan.
   a. Untuk Nama, masukkan`disruptConnectivity`.
   b. Untuk tipe Action, pilih aws:network:disrupt-connectivity.
   c. Di bawah Parameter tindakan, atur Durasi ke`2 minutes`.
   d. Di bawah Lingkup, pilih s3.
   e. Di bagian atas, pilih Simpan.
6. Di bawah Target, Anda akan melihat target yang telah dibuat secara otomatis. Pilih Edit.
   a. Verifikasi bahwa jenis Resource adalah`aws:ec2:subnet`.
   b. Di bawah metode Target, pilih Resource IDs, lalu pilih subnet yang Anda gunakan saat membuat EC2 instance Amazon Anda dalam langkah-langkah Prasyarat.
   c. Verifikasi bahwa mode Seleksi adalah Semua.
   d. Pilih Simpan.
7. Pilih Berikutnya untuk pindah ke Langkah 3, Konfigurasikan akses layanan.
8. Di bawah Service Access, pilih peran IAM yang Anda buat seperti yang dijelaskan dalam Prasyarat untuk tutorial ini. Jika peran Anda tidak ditampilkan, verifikasi bahwa ia memiliki hubungan kepercayaan yang diperlukan. Untuk informasi selengkapnya, lihat the section called “Peran percobaan”.
9. Pilih Berikutnya untuk pindah ke Langkah 4, Konfigurasikan pengaturan opsional.
10. (Opsional) Dalam kondisi Stop, Anda dapat memilih CloudWatch alarm untuk menghentikan percobaan jika kondisi terjadi. Untuk informasi selengkapnya, lihat Kondisi berhenti untuk AWS FIS.
11. (Opsional) Di bawah Log, Anda dapat memilih bucket Amazon S3, atau mengirim log CloudWatch untuk eksperimen Anda.
12. Pilih Berikutnya untuk pindah ke Langkah 5, Tinjau dan buat.
13. Tinjau template dan pilih Buat template eksperimen. Ketika diminta untuk konfirmasi, masukkan`create`, Lalu pilih Buat template percobaan.

Langkah 2: Ping ke titik akhir Amazon S3

Pastikan EC2 instans Amazon Anda dapat mencapai titik akhir Amazon S3.

1. Connect ke EC2 instans Amazon yang Anda buat di langkah-langkah Prasyarat.

   Untuk pemecahan masalah, lihat Memecahkan masalah saat menyambung ke instans Anda di Panduan Pengguna Amazon. EC2

2. Periksa untuk melihat di Wilayah AWS mana instans Anda berada. Anda dapat melakukan ini di EC2 konsol Amazon atau dengan menjalankan perintah berikut.

   ```
   hostname
   ```

   Misalnya, jika Anda meluncurkan EC2 instans Amazon di`us-west-2`, Anda akan melihat output berikut.

   ```
   [ec2-user@ip-172.16.0.0 ~]$ hostname
   ip-172.16.0.0.us-west-2.compute.internal
   ```

3. Ping titik akhir Amazon S3 di. Wilayah AWS Ganti *`Wilayah AWS`* dengan Wilayah Anda.

   ```
   ping -c 1 s3.Wilayah AWS.amazonaws.com
   ```

   Untuk output, Anda akan melihat ping sukses dengan 0% packet loss, seperti yang ditunjukkan pada contoh berikut.

   ```
   PING s3.us-west-2.amazonaws.com (x.x.x.x) 56(84) bytes of data.
   64 bytes from s3-us-west-2.amazonaws.com (x.x.x.x: icmp_seq=1 ttl=249 time=1.30 ms

   --- s3.us-west-2.amazonaws.com ping statistics ---
   1 packets transmitted, 1 received, 0% packet loss, time 0ms
   rtt min/avg/max/mdev = 1.306/1.306/1.306/0.000 ms
   ```

Langkah 3: Mulai eksperimen AWS FIS Anda

Mulai percobaan dengan template eksperimen yang baru saja Anda buat.

1. Buka konsol AWS FIS di https://console.aws.amazon.com/fis/.

2. Di panel navigasi kiri, pilih Template eksperimen.
3. Pilih ID template eksperimen yang Anda buat untuk membuka halaman detailnya.
4. Pilih Mulai percobaan.
5. (Opsional) Di halaman konfirmasi, tambahkan tag untuk eksperimen Anda.
6. Di halaman konfirmasi, pilih Mulai eksperimen.

Langkah 4: Lacak kemajuan eksperimen AWS FIS Anda

Anda dapat melacak kemajuan eksperimen yang sedang berjalan hingga percobaan selesai, dihentikan, atau gagal.

1. Anda harus berada di halaman detail untuk eksperimen yang baru saja Anda mulai. Jika tidak, pilih Eksperimen, lalu pilih ID eksperimen untuk membuka halaman detailnya.
2. Untuk melihat status percobaan, periksa Status di panel detail. Untuk informasi selengkapnya, lihat Status eksperimen.
3. Saat keadaan percobaan sedang berjalan, lanjutkan ke langkah berikutnya.

Langkah 5: Verifikasi gangguan jaringan Amazon S3

Anda dapat memvalidasi kemajuan eksperimen dengan melakukan ping ke titik akhir Amazon S3.

- Dari EC2 instans Amazon Anda, ping titik akhir Amazon S3 di titik akhir Anda. Wilayah AWS Ganti *`Wilayah AWS`* dengan Wilayah Anda.

  ```
  ping -c 1 s3.Wilayah AWS.amazonaws.com
  ```

  Untuk output, Anda akan melihat ping yang gagal dengan 100% packet loss, seperti yang ditunjukkan pada contoh berikut.

  ```
  ping -c 1 s3.us-west-2.amazonaws.com
  PING s3.us-west-2.amazonaws.com (x.x.x.x) 56(84) bytes of data.

  --- s3.us-west-2.amazonaws.com ping statistics ---
  1 packets transmitted, 0 received, 100% packet loss, time 0ms
  ```

Langkah 5: Bersihkan

Jika Anda tidak lagi memerlukan EC2 instans Amazon yang Anda buat untuk eksperimen ini atau templat AWS FIS, Anda dapat menghapusnya.

Untuk menghapus EC2 instance Amazon

1. Buka EC2 konsol Amazon di https://console.aws.amazon.com/ec2/.
2. Di panel navigasi, pilih Instans.
3. Pilih contoh pengujian, pilih status Instance, dan kemudian pilih Terminate instance.
4. Saat diminta konfirmasi, pilih Akhiri.

Untuk menghapus template percobaan menggunakan konsol AWS FIS

1. Buka konsol AWS FIS di https://console.aws.amazon.com/fis/.
2. Di panel navigasi, pilih Template eksperimen.
3. Pilih template eksperimen, lalu pilih Tindakan, Hapus template eksperimen.
4. Saat diminta konfirmasi, masukkan`delete`, lalu pilih Hapus templat eksperimen.

Tutorial: Jadwalkan percobaan berulang

Dengan AWS AWS Fault Injection Service (FIS), Anda dapat melakukan eksperimen injeksi kesalahan pada beban AWS kerja Anda. Eksperimen ini berjalan pada template yang berisi satu atau beberapa tindakan untuk dijalankan pada target tertentu. Saat Anda juga menggunakan Amazon EventBridge, Anda dapat menjadwalkan eksperimen Anda sebagai tugas satu kali atau tugas berulang.

Gunakan tutorial ini untuk membuat EventBridge jadwal yang menjalankan template eksperimen AWS FIS setiap 5 menit.

Tugas

- Prasyarat
- Langkah 1: Buat peran dan kebijakan IAM
- Langkah 2: Buat Amazon EventBridge Scheduler
- Langkah 3: Verifikasi eksperimen Anda

- Langkah 4: Membersihkan

Prasyarat

Sebelum memulai tutorial ini, harus memiliki template eksperimen AWS FIS yang ingin Anda jalankan sesuai jadwal. Jika Anda sudah memiliki template percobaan yang berfungsi, catat ID template dan Wilayah AWS. Jika tidak, Anda dapat membuat template dengan mengikuti instruksi dithe section called “Contoh uji berhenti dan mulai”, dan kemudian kembali ke tutorial ini.

Langkah 1: Buat peran dan kebijakan IAM

Untuk membuat peran dan kebijakan IAM

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi kiri, pilih Peran, lalu Buat Peran.
3. Pilih Kebijakan kepercayaan khusus, lalu masukkan cuplikan berikut untuk memungkinkan Amazon EventBridge Scheduler mengambil peran atas nama Anda.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                  "Service": "scheduler.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

   Pilih Berikutnya.
4. Di bawah Tambahkan izin, pilih Buat kebijakan.
5. Pilih JSON, lalu masukkan kebijakan berikut. Ganti *your-experiment-template-id* nilai dengan ID templat eksperimen Anda dari langkah-langkah Prasyarat.

```
{
    "Version": "2012-10-17",
```

```
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "fis:StartExperiment",
            "Resource": [
                "arn:aws:fis:*:*:experiment-template/your-experiment-template-id",
                "arn:aws:fis:*:*:experiment/*"
            ]
        }
    ]
}
```

Anda dapat membatasi penjadwal untuk hanya menjalankan templat eksperimen AWS FIS yang memiliki nilai tag tertentu. Misalnya, kebijakan berikut memberikan `StartExperiment` izin untuk semua eksperimen AWS FIS, tetapi membatasi penjadwal untuk hanya menjalankan templat eksperimen yang diberi tag. `Purpose=Schedule`

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "fis:StartExperiment",
            "Resource": "arn:aws:fis:*:*:experiment/*"
        },
        {
            "Effect": "Allow",
            "Action": "fis:StartExperiment",
            "Resource": "arn:aws:fis:*:*:experiment-template/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/Purpose": "Schedule"
                }
            }
        }
    ]
}
```

Pilih Selanjutnya: Tag.

6. Pilih Berikutnya: Tinjau.

7. Di bawah Kebijakan tinjauan, beri nama kebijakan Anda`FIS_RecurringExperiment`, lalu pilih Buat kebijakan.
8. Di bawah Tambahkan izin, tambahkan `FIS_RecurringExperiment` kebijakan baru ke peran Anda, lalu pilih Berikutnya.
9. Di bawah Nama, tinjau, dan buat, beri nama peran`FIS_RecurringExperiment_role`, lalu pilih Buat peran.

Langkah 2: Buat Amazon EventBridge Scheduler

Untuk membuat Amazon EventBridge Scheduler

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi kiri, pilih Jadwal.
3. Verifikasi bahwa Anda Wilayah AWS sama dengan templat eksperimen AWS FIS Anda.
4. Pilih Buat jadwal, dan isi yang berikut ini:
   - Di bawah nama Jadwal, masukkan`FIS_recurring_experiment_tutorial`.
   - Di bawah Pola jadwal, pilih Jadwal berulang.
   - Di bawah Jenis jadwal, pilih Jadwal berdasarkan tarif.
   - Di bawah Ekspresi Rate, pilih 5 menit.
   - Di bawah jendela waktu fleksibel, pilih Mati.
   - (Opsional) Di bawah Jangka Waktu, pilih zona waktu Anda.
   - Pilih Berikutnya.
5. Di bawah Pilih target, pilih Semua APIs, lalu cari AWS FIS.
6. Pilih AWS FIS, lalu pilih StartExperiment.
7. Di bawah Input, masukkan payload JSON berikut. Ganti *`your-experiment-template-id`* nilainya dengan ID templat eksperimen Anda. `ClientToken`Ini adalah pengidentifikasi unik untuk penjadwal. Dalam tutorial ini, kita menggunakan kata kunci konteks yang diizinkan oleh Amazon EventBridge Scheduler. Untuk informasi selengkapnya, lihat Menambahkan atribut konteks di Panduan EventBridge Pengguna Amazon.

```
{
    "ClientToken": "<aws.scheduler.execution-id>",
    "ExperimentTemplateId": "your-experiment-template-id"
```

```
}
```

Pilih Berikutnya.

8. (Opsional) Di bawah Pengaturan, Anda dapat mengatur kebijakan Coba lagi, antrian Dead-letter (DLQ), dan pengaturan Enkripsi. Atau, Anda dapat menyimpan nilai default.
9. Di bawah Izin, pilih Gunakan peran yang ada, lalu cari`FIS_RecurringExperiment_role`.
10. Pilih Berikutnya.
11. Di bawah Tinjau dan buat jadwal, tinjau detail penjadwal Anda, lalu pilih Buat jadwal.

Langkah 3: Verifikasi eksperimen Anda

Untuk memverifikasi bahwa eksperimen AWS FIS Anda berjalan sesuai jadwal

1. Buka konsol AWS FIS di https://console.aws.amazon.com/fis/.
2. Di panel navigasi kiri, pilih Eksperimen.
3. Lima menit setelah Anda membuat jadwal, Anda akan melihat eksperimen Anda berjalan.

Langkah 4: Membersihkan

Untuk menonaktifkan Amazon EventBridge Scheduler

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi kiri, pilih Jadwal.
3. Pilih penjadwal yang baru dibuat, lalu pilih Nonaktifkan.

Bekerja dengan pustaka AWS FIS skenario

Skenario menentukan peristiwa atau kondisi yang dapat diterapkan pelanggan untuk menguji ketahanan aplikasi mereka, seperti gangguan sumber daya komputasi tempat aplikasi berjalan. Skenario dibuat dan dimiliki oleh AWS, dan meminimalkan pengangkatan berat yang tidak berdiferensiasi dengan memberi Anda sekelompok target dan tindakan kesalahan yang telah ditentukan sebelumnya (misalnya, menghentikan 30% instance dalam grup penskalaan otomatis) untuk gangguan aplikasi umum.

Skenario disediakan melalui pustaka skenario khusus konsol dan dijalankan menggunakan templat eksperimen AWS FIS . Untuk menjalankan eksperimen menggunakan skenario, Anda akan memilih skenario dari pustaka, menentukan parameter yang cocok dengan detail beban kerja Anda, dan menyimpannya sebagai templat eksperimen di akun Anda.

Topik

- Melihat skenario
- Menggunakan skenario
- Mengekspor skenario
- Referensi skenario

Melihat skenario

Untuk melihat skenario menggunakan konsol:

1. Buka AWS FIS konsol dihttps://console.aws.amazon.com/fis/.
2. Di panel navigasi, pilih Pustaka skenario.
3. Untuk melihat informasi tentang skenario tertentu, pilih kartu skenario untuk memunculkan panel terpisah.
   - Di tab Deskripsi di panel terpisah di bagian bawah halaman, Anda dapat melihat deskripsi singkat tentang skenario. Anda juga dapat menemukan ringkasan singkat prasyarat yang berisi ringkasan sumber daya target yang diperlukan dan tindakan apa pun yang perlu Anda ambil untuk menyiapkan sumber daya untuk digunakan dengan skenario. Terakhir, Anda juga dapat melihat informasi tambahan tentang target dan tindakan dalam skenario serta durasi yang diantisipasi saat eksperimen berhasil berjalan dengan pengaturan default.

- Di tab Konten di panel terpisah di bagian bawah halaman, Anda dapat melihat pratinjau versi templat eksperimen yang terisi sebagian yang akan dibuat dari skenario.
- Di tab Detail di panel split di bagian bawah halaman, Anda dapat menemukan penjelasan rinci bagaimana skenario diterapkan. Ini mungkin berisi informasi rinci tentang bagaimana aspek individu dari skenario diperkirakan. Jika berlaku, Anda juga dapat membaca tentang metrik apa yang akan digunakan sebagai kondisi berhenti dan untuk memberikan pengamatan untuk belajar dari eksperimen. Akhirnya Anda akan menemukan rekomendasi cara memperluas template eksperimen yang dihasilkan.

Menggunakan skenario

Untuk menggunakan skenario menggunakan konsol:

1. Buka AWS FIS konsol dihttps://console.aws.amazon.com/fis/.
2. Di panel navigasi, pilih Pustaka skenario.
3. Untuk melihat informasi tentang skenario tertentu, pilih kartu skenario untuk memunculkan panel terpisah
4. Untuk menggunakan skenario, pilih kartu skenario dan pilih Buat template dengan skenario.
5. Dalam tampilan Buat template eksperimen, isi item yang hilang.
   a. Beberapa skenario memungkinkan Anda mengedit parameter massal yang dibagikan di beberapa tindakan atau target. Fungsionalitas ini akan dinonaktifkan setelah Anda membuat perubahan apa pun pada skenario, termasuk perubahan dengan pengeditan parameter massal. Untuk menggunakan fitur ini pilih tombol Edit parameter massal. Edit parameter di modal dan pilih tombol Simpan.
   b. Beberapa templat eksperimen mungkin memiliki parameter tindakan atau target yang hilang, disorot pada setiap tindakan dan kartu target. Pilih tombol Edit untuk setiap kartu, tambahkan informasi yang hilang, dan pilih tombol Simpan pada kartu.
   c. Semua template memerlukan peran eksekusi akses Layanan. Anda dapat memilih peran yang ada atau membuat peran baru untuk templat eksperimen ini.
   d. Sebaiknya tentukan satu atau beberapa kondisi Berhenti opsional dengan memilih CloudWatch alarm AWS yang ada. Pelajari lebih lanjut tentang Kondisi berhenti untuk AWS FIS. Jika alarm belum dikonfigurasi, Anda dapat mengikuti petunjuk di Menggunakan CloudWatch Alarm Amazon dan memperbarui templat eksperimen nanti.

e. Sebaiknya aktifkan Log percobaan opsional ke CloudWatch log Amazon atau ke bucket Amazon S3. Pelajari lebih lanjut tentang Pencatatan percobaan untuk AWS FIS. Jika sumber daya yang sesuai belum dikonfigurasi, Anda dapat memperbarui template eksperimen nanti.

6. Dalam template Buat eksperimen pilih Buat template eksperimen.
7. Dari tampilan templat Eksperimen AWS FIS konsol, pilih Mulai eksperimen. Pelajari lebih lanjut tentang Mengelola AWS templat eksperimen FIS.

Mengekspor skenario

Skenario adalah pengalaman khusus konsol. Meskipun mirip dengan templat eksperimen, skenario bukanlah templat eksperimen lengkap dan tidak dapat langsung diimpor ke dalamnya AWS FIS. Jika Anda ingin menggunakan skenario sebagai bagian dari otomatisasi Anda sendiri, Anda dapat menggunakan salah satu dari dua jalur:

1. Ikuti langkah-langkah Menggunakan skenario untuk membuat template AWS FIS eksperimen yang valid dan ekspor template tersebut.
2. Ikuti langkah-langkah di dalam Melihat skenario dan di langkah 3, dari tab Konten, salin dan simpan konten skenario, lalu tambahkan parameter yang hilang secara manual untuk membuat templat eksperimen yang valid.

Referensi skenario

Skenario yang disertakan dalam pustaka skenario dirancang untuk menggunakan tag jika memungkinkan dan setiap skenario menjelaskan tag yang diperlukan di bagian Prasyarat dan Cara kerjanya dari deskripsi skenario. Anda dapat menandai sumber daya Anda dengan tag yang telah ditentukan sebelumnya atau Anda dapat mengatur tag Anda sendiri menggunakan pengalaman pengeditan parameter massal (lihatMenggunakan skenario).

Referensi ini menjelaskan skenario umum di pustaka skenario AWS FIS. Anda juga dapat membuat daftar skenario yang didukung menggunakan konsol AWS FIS.

Untuk informasi selengkapnya, lihat Bekerja dengan pustaka AWS FIS skenario.

AWS FIS mendukung EC2 skenario Amazon berikut. Skenario ini menargetkan instance menggunakan tag. Anda dapat menggunakan tag Anda sendiri atau menggunakan tag default yang termasuk dalam skenario. Beberapa skenario ini menggunakan dokumen SSM.

- EC2 stress: kegagalan instance - Jelajahi efek kegagalan instance dengan menghentikan satu atau lebih EC2 instance.

  Instance target di wilayah saat ini yang memiliki tag tertentu yang dilampirkan. Dalam skenario ini kita akan menghentikan instance tersebut dan memulai ulang pada akhir durasi tindakan, secara default 5 menit.
- EC2 stress: Disk - Jelajahi dampak peningkatan pemanfaatan disk pada aplikasi EC2 berbasis Anda.

  Dalam skenario ini kita akan menargetkan EC2 instance di wilayah saat ini yang memiliki tag tertentu yang dilampirkan. Dalam skenario ini, Anda dapat menyesuaikan peningkatan jumlah pemanfaatan disk yang disuntikkan pada EC2 instance yang ditargetkan untuk durasi aksi, secara default 5 menit untuk setiap aksi stress disk.
- EC2 stress: CPU - Jelajahi dampak peningkatan CPU pada aplikasi EC2 berbasis Anda.

  Dalam skenario ini kita akan menargetkan EC2 instance di wilayah saat ini yang memiliki tag tertentu yang dilampirkan. Dalam skenario ini, Anda dapat menyesuaikan peningkatan jumlah stres CPU yang disuntikkan pada EC2 instans yang ditargetkan selama durasi aksi, secara default 5 menit untuk setiap aksi stres CPU.
- EC2 stress: Memori - Jelajahi dampak peningkatan pemanfaatan memori pada aplikasi EC2 berbasis Anda.

  Dalam skenario ini kita akan menargetkan EC2 instance di wilayah saat ini yang memiliki tag tertentu yang dilampirkan. Dalam skenario ini Anda dapat menyesuaikan peningkatan jumlah stres memori yang disuntikkan pada EC2 instance yang ditargetkan untuk durasi aksi, secara default 5 menit untuk setiap aksi stres memori.
- EC2 stress: Latensi Jaringan - Jelajahi dampak peningkatan latensi jaringan pada aplikasi EC2 berbasis Anda.

  Dalam skenario ini kita akan menargetkan EC2 instance di wilayah saat ini yang memiliki tag tertentu yang dilampirkan. Dalam skenario ini, Anda dapat menyesuaikan peningkatan jumlah latensi jaringan yang disuntikkan pada EC2 instans yang ditargetkan selama durasi tindakan, secara default 5 menit untuk setiap tindakan latensi.

AWS FIS mendukung skenario Amazon EKS berikut. Skenario ini menargetkan pod EKS menggunakan label aplikasi Kubernetes. Anda dapat menggunakan label Anda sendiri atau

menggunakan label default yang disertakan dalam skenario. Untuk informasi lebih lanjut tentang EKS dengan FIS, lihatTindakan EKS Pod.

- EKS stress: Pod Delete - Jelajahi efek kegagalan pod EKS dengan menghapus satu atau beberapa pod.

  Dalam skenario ini kita akan menargetkan pod di wilayah saat ini yang terkait dengan label aplikasi. Dalam skenario ini kita akan menghentikan semua pod yang cocok. Pembuatan ulang pod akan dikontrol oleh konfigurasi kubernetes.

- EKS stress: CPU - Jelajahi dampak peningkatan CPU pada aplikasi berbasis EKS Anda.

  Dalam skenario ini kita akan menargetkan pod di wilayah saat ini yang terkait dengan label aplikasi. Dalam skenario ini, Anda dapat menyesuaikan peningkatan jumlah stres CPU yang disuntikkan pada pod EKS yang ditargetkan selama durasi aksi, secara default 5 menit untuk setiap aksi stres CPU.

- EKS stress: Disk - Jelajahi dampak peningkatan pemanfaatan disk pada aplikasi berbasis EKS Anda.

  Dalam skenario ini kita akan menargetkan pod di wilayah saat ini yang terkait dengan label aplikasi. Dalam skenario ini, Anda dapat menyesuaikan peningkatan jumlah stres disk yang disuntikkan pada pod EKS yang ditargetkan selama durasi aksi, secara default 5 menit untuk setiap aksi stres CPU.

- EKS stress: Memori - Jelajahi dampak peningkatan pemanfaatan memori pada aplikasi berbasis EKS Anda.

  Dalam skenario ini kita akan menargetkan pod di wilayah saat ini yang terkait dengan label aplikasi. Dalam skenario ini, Anda dapat menyesuaikan peningkatan jumlah stress memori yang disuntikkan pada pod EKS yang ditargetkan selama durasi aksi, secara default 5 menit untuk setiap aksi stres memori.

- Stres EKS: Latensi jaringan - Jelajahi dampak peningkatan latensi jaringan pada aplikasi berbasis EKS Anda.

  Dalam skenario ini kita akan menargetkan pod di wilayah saat ini yang terkait dengan label aplikasi. Dalam skenario ini, Anda dapat menyesuaikan peningkatan jumlah latensi jaringan yang disuntikkan pada pod EKS yang ditargetkan selama durasi aksi, secara default 5 menit untuk setiap tindakan latensi.

AWS FIS mendukung skenario berikut untuk aplikasi Multi-AZ dan Multi-region. Skenario ini menargetkan beberapa jenis sumber daya.

- AZ Availability: Power Interruption- Suntikkan gejala yang diharapkan dari gangguan daya total di Availability Zone (AZ). Pelajari lebih lanjut tentang AZ Availability: Power Interruption.
- Cross-Region: Connectivity- Blokir lalu lintas jaringan aplikasi dari Wilayah percobaan ke Wilayah tujuan dan jeda replikasi data lintas wilayah. Pelajari lebih lanjut tentang menggunakanCross-Region: Connectivity.

AZ Availability: Power Interruption

Anda dapat menggunakan AZ Availability: Power Interruption skenario untuk menginduksi gejala yang diharapkan dari gangguan daya total di Availability Zone (AZ).

Skenario ini dapat digunakan untuk menunjukkan bahwa aplikasi Multi-AZ beroperasi seperti yang diharapkan selama gangguan daya AZ tunggal yang lengkap. Ini termasuk hilangnya komputasi zonal (Amazon EC2, EKS, dan ECS), tidak ada penskalaan ulang komputasi di AZ, kehilangan konektivitas subnet, failover RDS, failover, dan volume EBS yang tidak responsif. ElastiCache Secara default, tindakan yang tidak ditemukan targetnya akan dilewati.

Tindakan

Bersama-sama, tindakan berikut menciptakan banyak gejala yang diharapkan dari gangguan daya lengkap dalam satu AZ. Ketersediaan AZ: Gangguan Daya hanya memengaruhi layanan yang diperkirakan akan berdampak selama gangguan daya AZ tunggal. Secara default, skenario menyuntikkan gejala gangguan daya selama 30 menit dan kemudian, selama 30 menit tambahan, menyuntikkan gejala yang mungkin terjadi selama pemulihan.

Instans Berhenti

Selama gangguan daya AZ, EC2 instans di AZ yang terpengaruh akan mati. Setelah daya dipulihkan, instance akan reboot. AZ Availability: Power Interruptionmenyertakan aws:ec2:stop-instances untuk menghentikan semua instans di AZ yang terpengaruh selama durasi interupsi. Setelah durasi, instance dimulai ulang. Menghentikan EC2 instans yang dikelola oleh Amazon EKS menyebabkan pod EKS dependen dihapus. Menghentikan EC2 instans yang dikelola oleh Amazon ECS menyebabkan tugas ECS dependen dihentikan.

Tindakan ini menargetkan EC2 instance yang berjalan di AZ yang terpengaruh. Secara default, ini menargetkan instance dengan tag bernama `AzImpairmentPower` dengan nilai. `StopInstances`

Anda dapat menambahkan tag ini ke instance Anda atau mengganti tag default dengan tag Anda sendiri di template eksperimen. Secara default, jika tidak ditemukan instance valid, tindakan ini akan dilewati.

Instans Stop-ASG

Selama gangguan daya AZ, EC2 instans yang dikelola oleh grup Auto Scaling di AZ yang terpengaruh akan dimatikan. Setelah daya dipulihkan, instance akan reboot. AZ Availability: Power Interruptionmenyertakan aws:ec2:stop-instances untuk menghentikan semua instans, termasuk yang dikelola oleh Auto Scaling, di AZ yang terpengaruh selama durasi interupsi. Setelah durasi, instance dimulai ulang.

Tindakan ini menargetkan EC2 instance yang berjalan di AZ yang terpengaruh. Secara default, ini menargetkan instance dengan tag bernama `AzImpairmentPower` dengan nilai. `IceAsg` Anda dapat menambahkan tag ini ke instance Anda atau mengganti tag default dengan tag Anda sendiri di template eksperimen. Secara default, jika tidak ditemukan instance valid, tindakan ini akan dilewati.

Jeda Peluncuran Instans

Selama gangguan daya AZ, panggilan EC2 API ke kapasitas penyediaan di AZ akan gagal. Secara khusus, hal-hal berikut APIs akan terpengaruh:`ec2:StartInstances`,`ec2:CreateFleet`, dan`ec2:RunInstances`. AZ Availability: Power Interruption includestermasuk aws:ec2: api-insufficient-instance-capacity -error untuk mencegah instance baru disediakan di AZ yang terpengaruh.

Tindakan ini menargetkan peran IAM yang digunakan untuk menyediakan instance. Ini harus ditargetkan menggunakan ARN. Secara default, jika tidak ditemukan peran IAM yang valid, tindakan ini akan dilewati.

Jeda Penskalaan ASG

Selama gangguan daya AZ, panggilan EC2 API yang dilakukan oleh pesawat kontrol Auto Scaling untuk memulihkan kapasitas yang hilang di AZ akan gagal. Secara khusus, hal-hal berikut APIs akan terpengaruh:`ec2:StartInstances`,`ec2:CreateFleet`, dan`ec2:RunInstances`. AZ Availability: Power Interruptiontermasuk aws:ec2: asg-insufficient-instance-capacity -error untuk mencegah instance baru disediakan di AZ yang terpengaruh. Ini juga mencegah Amazon EKS dan Amazon ECS dari penskalaan di AZ yang terpengaruh.

Tindakan ini menargetkan grup Auto Scaling. Secara default, ini menargetkan grup Auto Scaling dengan tag bernama `AzImpairmentPower` dengan nilai. `IceAsg` Anda dapat menambahkan tag

ini ke grup Auto Scaling atau mengganti tag default dengan tag Anda sendiri di template eksperimen. Secara default, jika tidak ditemukan grup Auto Scaling yang valid, tindakan ini akan dilewati.

Jeda Konektivitas Jaringan

Selama gangguan daya AZ, jaringan di AZ tidak akan tersedia. Ketika ini terjadi, beberapa layanan AWS mungkin memerlukan waktu hingga beberapa menit untuk memperbarui DNS agar mencerminkan bahwa titik akhir pribadi di AZ yang terpengaruh tidak tersedia. Selama waktu ini, pencarian DNS dapat mengembalikan alamat IP yang tidak dapat diakses. AZ Availability: Power Interruptiontermasuk aws:network:disrupt-connectivity untuk memblokir semua konektivitas jaringan untuk semua subnet di AZ yang terpengaruh selama 2 menit. Ini akan memaksa batas waktu dan penyegaran DNS untuk sebagian besar aplikasi. Mengakhiri tindakan setelah 2 menit memungkinkan pemulihan DNS layanan regional selanjutnya sementara AZ terus tidak tersedia.

Tindakan ini menargetkan subnet. Secara default, ini menargetkan cluster dengan tag bernama `AzImpairmentPower` dengan nilai. `DisruptSubnet` Anda dapat menambahkan tag ini ke subnet Anda atau mengganti tag default dengan tag Anda sendiri di template percobaan. Secara default, jika tidak ditemukan subnet yang valid, tindakan ini akan dilewati.

Kegagalan RDS

Selama gangguan daya AZ, node RDS di AZ yang terpengaruh akan mati. Node RDS AZ tunggal di AZ yang terpengaruh akan sepenuhnya tidak tersedia. Untuk cluster multi-AZ, node penulis akan gagal menjadi AZ yang tidak terpengaruh dan node pembaca di AZ yang terpengaruh tidak akan tersedia. Untuk cluster multi-AZ, AZ Availability: Power Interruption sertakan aws:rds: failover-db-cluster untuk failover jika penulis berada di AZ yang terpengaruh.

Tindakan ini menargetkan cluster RDS. Secara default, ini menargetkan cluster dengan tag bernama `AzImpairmentPower` dengan nilai. `DisruptRds` Anda dapat menambahkan tag ini ke kluster atau mengganti tag default dengan tag Anda sendiri di templat eksperimen. Secara default, jika tidak ditemukan cluster yang valid, tindakan ini akan dilewati.

Jeda Grup ElastiCache Replikasi

Selama gangguan daya AZ, ElastiCache node di AZ tidak tersedia. AZ Availability: Power Interruptiontermasuk aws:elasticache: replicationgroup-interrupt-az-power untuk mengakhiri ElastiCache node di AZ yang terpengaruh. Selama durasi interupsi, instans baru tidak akan disediakan di AZ yang terpengaruh, sehingga grup replikasi akan tetap pada kapasitas yang berkurang.

Tindakan ini menargetkan kelompok ElastiCache replikasi. Secara default, ini menargetkan grup replikasi dengan tag bernama `AzImpairmentPower` dengan nilai. `ElasticacheImpact` Anda dapat menambahkan tag ini ke grup replikasi atau mengganti tag default dengan tag Anda sendiri di templat eksperimen. Secara default, jika tidak ditemukan grup replikasi yang valid, tindakan ini akan dilewati. Perhatikan bahwa hanya grup replikasi dengan node penulis di AZ yang terpengaruh yang akan dianggap sebagai target yang valid.

Mulai ARC Zonal Autoshift

Lima menit setelah gangguan daya AZ dimulai, tindakan pemulihan `aws:arc:start-zonal-autoshift` secara otomatis menggeser lalu lintas sumber daya dari AZ yang ditentukan selama 25 menit sisa gangguan daya. Setelah durasi itu, lalu lintas bergeser kembali ke AZ asli. Perhatikan bahwa selama gangguan daya AZ dunia nyata AWS akan mendeteksi gangguan dan mengalihkan lalu lintas sumber daya jika pergeseran otomatis diaktifkan. Sementara waktu pergeseran ini bervariasi, diperkirakan terjadi lima menit sejak penurunan nilai dimulai.

Tindakan ini menargetkan sumber daya berkemampuan autoshift Amazon Application Recovery Controller (ARC). Secara default, ini menargetkan sumber daya dengan kunci tag `AzImpairmentPower` dan nilai`RecoverAutoshiftResources`. Anda dapat menambahkan tag ini ke sumber daya Anda atau mengganti tag default dengan tag Anda sendiri di template eksperimen. Misalnya, Anda mungkin ingin menggunakan tag khusus aplikasi. Secara default, jika tidak ditemukan sumber daya yang valid, tindakan ini akan dilewati.

Jeda EBS I/O

Setelah gangguan daya AZ, setelah daya dipulihkan, persentase instans yang sangat kecil mungkin mengalami volume EBS yang tidak responsif. AZ Availability: Power Interruptionmenyertakan aws:ebs:pause-io untuk membiarkan 1 volume EBS dalam keadaan tidak responsif.

Secara default, hanya volume yang disetel untuk bertahan setelah instance dihentikan yang ditargetkan. Tindakan ini menargetkan volume dengan tag bernama `AzImpairmentPower` dengan nilai`APIPauseVolume`. Anda dapat menambahkan tag ini ke volume Anda atau mengganti tag default dengan tag Anda sendiri di template eksperimen. Secara default, jika tidak ditemukan volume yang valid, tindakan ini akan dilewati.

Batasan

- Skenario ini tidak termasuk kondisi berhenti. Kondisi berhenti yang benar untuk aplikasi Anda harus ditambahkan ke template percobaan.

- Di AZ yang ditargetkan, Pod Amazon EKS yang berjalan EC2 akan dihentikan dengan node EC2 pekerja dan awal EC2 node baru akan diblokir. Namun, Pod Amazon EKS yang berjalan di AWS Fargate tidak didukung.
- Di AZ yang ditargetkan, tugas Amazon ECS yang berjalan pada EC2 akan dihentikan dengan node EC2 pekerja dan memulai EC2 node baru akan diblokir. Namun, tugas Amazon ECS yang berjalan di AWS Fargate tidak didukung.
- Amazon RDS Multi-AZ dengan dua instans DB siaga yang dapat dibaca tidak didukung. Dalam hal ini, instans akan dihentikan, RDS akan gagal, dan kapasitas akan segera disediakan kembali di AZ yang terpengaruh. Siaga yang dapat dibaca di AZ yang terpengaruh akan tetap tersedia.

Persyaratan

- Tambahkan izin yang diperlukan ke peran eksperimen AWS FIS.
- Tag sumber daya harus diterapkan pada sumber daya yang akan ditargetkan oleh eksperimen. Ini dapat menggunakan konvensi penandaan Anda sendiri atau tag default yang ditentukan dalam skenario.

Izin

Autoshift zona ARC menggunakan peran terkait layanan IAM `AWSServiceRoleForZonalAutoshiftPracticeRun` untuk melakukan pergeseran zona atas nama Anda. Peran ini menggunakan kebijakan `AWSZonalAutoshiftPracticeRunSLRPolicy`terkelola IAM. Anda tidak perlu membuat peran secara manual. Saat Anda membuat template eksperimen dari skenario Gangguan Daya AZ di AWS Management Console, the, atau AWS SDK AWS CLI, ARC akan membuat peran terkait layanan untuk Anda. Untuk informasi selengkapnya, lihat Menggunakan peran terkait layanan untuk pergeseran otomatis zona di ARC.

Kebijakan berikut memberikan AWS FIS izin yang diperlukan untuk menjalankan eksperimen dengan skenario. AZ Availability: Power Interruption Kebijakan ini harus dilampirkan pada peran percobaan.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowFISExperimentLoggingActionsCloudwatch",
            "Effect": "Allow",
```

```
            "Action": [
                "logs:CreateLogDelivery",
                "logs:PutResourcePolicy",
                "logs:DescribeResourcePolicies",
                "logs:DescribeLogGroups"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "ec2:CreateTags",
            "Resource": "arn:aws:ec2:*:*:network-acl/*",
            "Condition": {
                "StringEquals": {
                    "ec2:CreateAction": "CreateNetworkAcl",
                    "aws:RequestTag/managedByFIS": "true"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "ec2:CreateNetworkAcl",
            "Resource": "arn:aws:ec2:*:*:network-acl/*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/managedByFIS": "true"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateNetworkAclEntry",
                "ec2:DeleteNetworkAcl"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:network-acl/*",
                "arn:aws:ec2:*:*:vpc/*"
            ],
            "Condition": {
                "StringEquals": {
                    "ec2:ResourceTag/managedByFIS": "true"
                }
            }
```

```
        },
        {
            "Effect": "Allow",
            "Action": "ec2:CreateNetworkAcl",
            "Resource": "arn:aws:ec2:*:*:vpc/*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeVpcs",
                "ec2:DescribeManagedPrefixLists",
                "ec2:DescribeSubnets",
                "ec2:DescribeNetworkAcls"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "ec2:ReplaceNetworkAclAssociation",
            "Resource": [
                "arn:aws:ec2:*:*:subnet/*",
                "arn:aws:ec2:*:*:network-acl/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "rds:FailoverDBCluster"
            ],
            "Resource": [
                "arn:aws:rds:*:*:cluster:*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "rds:RebootDBInstance"
            ],
            "Resource": [
                "arn:aws:rds:*:*:db:*"
            ]
        },
        {
            "Effect": "Allow",
```

```
            "Action": [
                "elasticache:DescribeReplicationGroups",
                "elasticache:InterruptClusterAzPower"
            ],
            "Resource": [
                "arn:aws:elasticache:*:*:replicationgroup:*"
            ]
        },
        {
            "Sid": "TargetResolutionByTags",
            "Effect": "Allow",
            "Action": [
                "tag:GetResources"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:StartInstances",
                "ec2:StopInstances"
            ],
            "Resource": "arn:aws:ec2:*:*:instance/*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeInstances"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "kms:CreateGrant"
            ],
            "Resource": [
                "arn:aws:kms:*:*:key/*"
            ],
            "Condition": {
                "StringLike": {
                    "kms:ViaService": "ec2.*.amazonaws.com"
                },
                "Bool": {
```

```
                "kms:GrantIsForAWSResource": "true"
            }
        }
    },
    {
        "Effect": "Allow",
        "Action": [
            "ec2:DescribeVolumes"
        ],
        "Resource": "*"
    },
    {
        "Effect": "Allow",
        "Action": [
            "ec2:PauseVolumeIO"
        ],
        "Resource": "arn:aws:ec2:*:*:volume/*"
    },
    {
        "Sid": "AllowInjectAPI",
        "Effect": "Allow",
        "Action": [
            "ec2:InjectApiError"
        ],
        "Resource": [
            "*"
        ],
        "Condition": {
            "ForAnyValue:StringEquals": {
                "ec2:FisActionId": [
                    "aws:ec2:api-insufficient-instance-capacity-error",
                    "aws:ec2:asg-insufficient-instance-capacity-error"
                ]
            }
        }
    },
    {
        "Sid": "DescribeAsg",
        "Effect": "Allow",
        "Action": [
            "autoscaling:DescribeAutoScalingGroups"
        ],
        "Resource": [
            "*"
```

```
            ]
        }
    ]
}
```

Skenario Konten

Konten berikut mendefinisikan skenario. JSON ini dapat disimpan dan digunakan untuk membuat template eksperimen menggunakan create-experiment-templateperintah dari AWS Command Line Interface (AWS CLI). Untuk versi skenario terbaru, kunjungi pustaka skenario di konsol FIS.

```
{
    "targets": {
        "IAM-role": {
            "resourceType": "aws:iam:role",
            "resourceArns": [],
            "selectionMode": "ALL"
        },
        "EBS-Volumes": {
            "resourceType": "aws:ec2:ebs-volume",
            "resourceTags": {
                "AzImpairmentPower": "ApiPauseVolume"
            },
            "selectionMode": "COUNT(1)",
            "parameters": {
                "availabilityZoneIdentifier": "us-east-1a"
            },
            "filters": [
                {
                    "path": "Attachments.DeleteOnTermination",
                    "values": [
                        "false"
                    ]
                }
            ]
        },
        "EC2-Instances": {
            "resourceType": "aws:ec2:instance",
            "resourceTags": {
                "AzImpairmentPower": "StopInstances"
            },
            "filters": [
```

```
                {
                    "path": "State.Name",
                    "values": [
                        "running"
                    ]
                },
                {
                    "path": "Placement.AvailabilityZone",
                    "values": [
                        "us-east-1a"
                    ]
                }
            ],
            "selectionMode": "ALL"
        },
        "ASG": {
            "resourceType": "aws:ec2:autoscaling-group",
            "resourceTags": {
                "AzImpairmentPower": "IceAsg"
            },
            "selectionMode": "ALL"
        },
        "ASG-EC2-Instances": {
            "resourceType": "aws:ec2:instance",
            "resourceTags": {
                "AzImpairmentPower": "IceAsg"
            },
            "filters": [
                {
                    "path": "State.Name",
                    "values": [
                        "running"
                    ]
                },
                {
                    "path": "Placement.AvailabilityZone",
                    "values": [
                        "us-east-1a"
                    ]
                }
            ],
            "selectionMode": "ALL"
        },
        "Subnet": {
```

```
            "resourceType": "aws:ec2:subnet",
            "resourceTags": {
                "AzImpairmentPower": "DisruptSubnet"
            },
            "filters": [
                {
                    "path": "AvailabilityZone",
                    "values": [
                        "us-east-1a"
                    ]
                }
            ],
            "selectionMode": "ALL",
            "parameters": {}
        },
        "RDS-Cluster": {
            "resourceType": "aws:rds:cluster",
            "resourceTags": {
                "AzImpairmentPower": "DisruptRds"
            },
            "selectionMode": "ALL",
            "parameters": {
                "writerAvailabilityZoneIdentifiers": "us-east-1a"
            }
        },
        "ElastiCache-Cluster": {
            "resourceType": "aws:elasticache:replicationgroup",
            "resourceTags": {
                "AzImpairmentPower": "DisruptElasticache"
            },
            "selectionMode": "ALL",
            "parameters": {
                "availabilityZoneIdentifier": "us-east-1a"
            }
        }
    },
    "actions": {
        "Pause-Instance-Launches": {
            "actionId": "aws:ec2:api-insufficient-instance-capacity-error",
            "parameters": {
                "availabilityZoneIdentifiers": "us-east-1a",
                "duration": "PT30M",
                "percentage": "100"
            },
```

```
            "targets": {
                "Roles": "IAM-role"
            }
        },
        "Pause-EBS-IO": {
            "actionId": "aws:ebs:pause-volume-io",
            "parameters": {
                "duration": "PT30M"
            },
            "targets": {
                "Volumes": "EBS-Volumes"
            },
            "startAfter": [
                "Stop-Instances",
                "Stop-ASG-Instances"
            ]
        },
        "Stop-Instances": {
            "actionId": "aws:ec2:stop-instances",
            "parameters": {
                "completeIfInstancesTerminated": "true",
                "startInstancesAfterDuration": "PT30M"
            },
            "targets": {
                "Instances": "EC2-Instances"
            }
        },
        "Pause-ASG-Scaling": {
            "actionId": "aws:ec2:asg-insufficient-instance-capacity-error",
            "parameters": {
                "availabilityZoneIdentifiers": "us-east-1a",
                "duration": "PT30M",
                "percentage": "100"
            },
            "targets": {
                "AutoScalingGroups": "ASG"
            }
        },
        "Stop-ASG-Instances": {
            "actionId": "aws:ec2:stop-instances",
            "parameters": {
                "completeIfInstancesTerminated": "true",
                "startInstancesAfterDuration": "PT30M"
            },
```

```
            "targets": {
                "Instances": "ASG-EC2-Instances"
            }
        },
        "Pause-network-connectivity": {
            "actionId": "aws:network:disrupt-connectivity",
            "parameters": {
                "duration": "PT2M",
                "scope": "all"
            },
            "targets": {
                "Subnets": "Subnet"
            }
        },
        "Failover-RDS": {
            "actionId": "aws:rds:failover-db-cluster",
            "parameters": {},
            "targets": {
                "Clusters": "RDS-Cluster"
            }
        },
        "Pause-ElastiCache": {
            "actionId": "aws:elasticache:replicationgroup-interrupt-az-power",
            "parameters": {
                "duration": "PT30M"
            },
            "targets": {
                "ReplicationGroups": "ElastiCache-Cluster"
            }
        }
    },
    "stopConditions": [
        {
            "source": "aws:cloudwatch:alarm",
            "value": ""
        }
    ],
    "roleArn": "",
    "tags": {
        "Name": "AZ Impairment: Power Interruption"
    },
    "logConfiguration": {
        "logSchemaVersion": 2
    },
```

```
    "experimentOptions": {
        "accountTargeting": "single-account",
        "emptyTargetResolutionMode": "skip"
    },
    "description": "Affect multiple resource types in a single AZ, targeting by tags
 and explicit ARNs, to approximate power interruption in one AZ."
}
```

Cross-Region: Connectivity

Anda dapat menggunakan Cross-Region: Connectivity skenario untuk memblokir lalu lintas jaringan aplikasi dari Wilayah percobaan ke Wilayah tujuan dan menjeda replikasi lintas wilayah untuk Amazon S3 dan Amazon DynamoDB. Lintas Wilayah: Konektivitas memengaruhi lalu lintas aplikasi keluar dari Wilayah tempat Anda menjalankan eksperimen (Region percobaan). Lalu lintas masuk tanpa kewarganegaraan dari Wilayah yang ingin Anda isolasi dari wilayah percobaan (Wilayah tujuan) tidak dapat diblokir. Lalu lintas dari AWS managed services mungkin tidak diblokir.

Skenario ini dapat digunakan untuk menunjukkan bahwa aplikasi Multi-wilayah beroperasi seperti yang diharapkan ketika sumber daya di Wilayah tujuan tidak dapat diakses dari Wilayah percobaan. Ini termasuk memblokir lalu lintas jaringan dari Wilayah percobaan ke Wilayah tujuan dengan menargetkan gateway transit dan tabel rute. Ini juga menghentikan replikasi lintas wilayah untuk S3 dan DynamoDB. Secara default, tindakan yang tidak ditemukan targetnya akan dilewati.

Tindakan

Bersama-sama, tindakan berikut memblokir konektivitas lintas wilayah untuk layanan AWS yang disertakan. Tindakan dijalankan secara paralel. Secara default, skenario memblokir lalu lintas selama 3 jam, yang dapat Anda tingkatkan hingga durasi maksimum 12 jam.

Mengganggu Konektivitas Transit Gateway

Cross Region: Connectivitytermasuk aws:network: transit-gateway-disrupt-cross -region-connectivity untuk memblokir lalu lintas jaringan lintas wilayah dari VPCs Wilayah percobaan ke VPCs Wilayah tujuan yang dihubungkan oleh gateway transit. Ini tidak memengaruhi akses ke titik akhir VPC dalam Wilayah eksperimen tetapi akan memblokir lalu lintas dari Wilayah eksperimen yang ditujukan untuk titik akhir VPC di Wilayah tujuan.

Tindakan ini menargetkan gateway transit yang menghubungkan Wilayah percobaan dan Wilayah tujuan. Secara default, ini menargetkan gateway transit dengan tag bernama `DisruptTransitGateway` dengan nilai. `Allowed` Anda dapat menambahkan tag ini ke gateway

transit atau mengganti tag default dengan tag Anda sendiri di templat eksperimen. Secara default, jika tidak ditemukan gateway transit yang valid, tindakan ini akan dilewati.

Mengganggu Konektivitas Subnet

Cross Region: Connectivitymenyertakan aws:network: route-table-disrupt-cross -region-connectivity untuk memblokir lalu lintas jaringan lintas wilayah dari Wilayah percobaan ke blok IP AWS publik VPCs di Wilayah tujuan. Blok IP publik ini mencakup titik akhir layanan AWS di Wilayah tujuan, misalnya titik akhir regional S3, dan blok IP AWS untuk layanan terkelola, misalnya alamat IP yang digunakan untuk penyeimbang beban dan Amazon API Gateway. Tindakan ini juga memblokir konektivitas jaringan melalui koneksi Peering VPC Lintas wilayah dari Wilayah percobaan ke Wilayah tujuan. Ini tidak memengaruhi akses ke titik akhir VPC di Wilayah percobaan tetapi akan memblokir lalu lintas dari Wilayah eksperimen yang ditujukan untuk titik akhir VPC di Wilayah tujuan.

Tindakan ini menargetkan subnet di Wilayah percobaan. Secara default, ini menargetkan subnet dengan tag bernama `DisruptSubnet` dengan nilai. `Allowed` Anda dapat menambahkan tag ini ke subnet Anda atau mengganti tag default dengan tag Anda sendiri di template percobaan. Secara default, jika tidak ditemukan subnet yang valid, tindakan ini akan dilewati.

Jeda Replikasi S3

Cross Region: Connectivitytermasuk aws:s3: bucket-pause-replication untuk menjeda replikasi S3 dari Wilayah percobaan ke Wilayah tujuan untuk ember yang ditargetkan. Replikasi dari Wilayah tujuan ke Wilayah percobaan tidak akan terpengaruh. Setelah skenario berakhir, replikasi bucket akan dilanjutkan dari titik dijeda. Perhatikan bahwa waktu yang diperlukan untuk replikasi agar semua objek tetap sinkron akan bervariasi berdasarkan durasi percobaan, dan laju unggahan objek ke bucket.

Tindakan ini menargetkan bucket S3 di Region percobaan dengan Replikasi Lintas Wilayah (CRR) yang diaktifkan ke bucket S3 di Wilayah tujuan. Secara default, ini menargetkan bucket dengan tag bernama `DisruptS3` dengan nilai. `Allowed` Anda dapat menambahkan tag ini ke bucket atau mengganti tag default dengan tag Anda sendiri di template eksperimen. Secara default, jika tidak ditemukan bucket yang valid, tindakan ini akan dilewati.

Jeda Replikasi DynamoDB

Cross-Region: Connectivitytermasuk aws:dynamodb: global-table-pause-replication untuk menjeda replikasi antara Wilayah percobaan dan semua Wilayah lainnya, termasuk Wilayah tujuan. Ini mencegah replikasi masuk dan keluar dari Wilayah percobaan tetapi tidak mempengaruhi replikasi antara Wilayah lain. Setelah skenario berakhir, replikasi tabel akan dilanjutkan dari titik itu dijeda.

Perhatikan bahwa waktu yang diperlukan untuk replikasi untuk menjaga semua data tetap sinkron akan bervariasi berdasarkan durasi percobaan dan tingkat perubahan pada tabel.

Tindakan ini menargetkan tabel global DynamoDB di Wilayah percobaan. Secara default, ini menargetkan tabel dengan tag bernama `DisruptDynamoDb` dengan nilai`Allowed`. Anda dapat menambahkan tag ini ke tabel Anda atau mengganti tag default dengan tag Anda sendiri di template percobaan. Secara default, jika tidak ditemukan tabel global yang valid, tindakan ini akan dilewati.

Batasan

- Skenario ini tidak termasuk kondisi berhenti. Kondisi berhenti yang benar untuk aplikasi Anda harus ditambahkan ke template percobaan.

Persyaratan

- Tambahkan izin yang diperlukan ke peran eksperimen AWS FIS.
- Tag sumber daya harus diterapkan pada sumber daya yang akan ditargetkan oleh eksperimen. Ini dapat menggunakan konvensi penandaan Anda sendiri atau tag default yang ditentukan dalam skenario.

Izin

Kebijakan berikut memberikan AWS FIS izin yang diperlukan untuk menjalankan eksperimen dengan skenario. Cross-Region: Connectivity Kebijakan ini harus dilampirkan pada peran percobaan.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "RouteTableDisruptConnectivity1",
            "Effect": "Allow",
            "Action": "ec2:CreateRouteTable",
            "Resource": "arn:aws:ec2:*:*:route-table/*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/managedByFIS": "true"
                }
            }
        },
        {
```

```
            "Sid": "RouteTableDisruptConnectivity2",
            "Effect": "Allow",
            "Action": "ec2:CreateRouteTable",
            "Resource": "arn:aws:ec2:*:*:vpc/*"
        },
        {
            "Sid": "RouteTableDisruptConnectivity21",
            "Effect": "Allow",
            "Action": "ec2:CreateTags",
            "Resource": "arn:aws:ec2:*:*:route-table/*",
            "Condition": {
                "StringEquals": {
                    "ec2:CreateAction": "CreateRouteTable",
                    "aws:RequestTag/managedByFIS": "true"
                }
            }
        },
        {
            "Sid": "RouteTableDisruptConnectivity3",
            "Effect": "Allow",
            "Action": "ec2:CreateTags",
            "Resource": "arn:aws:ec2:*:*:network-interface/*",
            "Condition": {
                "StringEquals": {
                    "ec2:CreateAction": "CreateNetworkInterface",
                    "aws:RequestTag/managedByFIS": "true"
                }
            }
        },
        {
            "Sid": "RouteTableDisruptConnectivity4",
            "Effect": "Allow",
            "Action": "ec2:CreateTags",
            "Resource": "arn:aws:ec2:*:*:prefix-list/*",
            "Condition": {
                "StringEquals": {
                    "ec2:CreateAction": "CreateManagedPrefixList",
                    "aws:RequestTag/managedByFIS": "true"
                }
            }
        },
        {
            "Sid": "RouteTableDisruptConnectivity5",
            "Effect": "Allow",
```

```
            "Action": "ec2:DeleteRouteTable",
            "Resource": [
                "arn:aws:ec2:*:*:route-table/*",
                "arn:aws:ec2:*:*:vpc/*"
            ],
            "Condition": {
                "StringEquals": {
                    "ec2:ResourceTag/managedByFIS": "true"
                }
            }
        },
        {
            "Sid": "RouteTableDisruptConnectivity6",
            "Effect": "Allow",
            "Action": "ec2:CreateRoute",
            "Resource": "arn:aws:ec2:*:*:route-table/*",
            "Condition": {
                "StringEquals": {
                    "ec2:ResourceTag/managedByFIS": "true"
                }
            }
        },
        {
            "Sid": "RouteTableDisruptConnectivity7",
            "Effect": "Allow",
            "Action": "ec2:CreateNetworkInterface",
            "Resource": "arn:aws:ec2:*:*:network-interface/*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/managedByFIS": "true"
                }
            }
        },
        {
            "Sid": "RouteTableDisruptConnectivity8",
            "Effect": "Allow",
            "Action": "ec2:CreateNetworkInterface",
            "Resource": [
                "arn:aws:ec2:*:*:subnet/*",
                "arn:aws:ec2:*:*:security-group/*"
            ]
        },
        {
            "Sid": "RouteTableDisruptConnectivity9",
```

```
            "Effect": "Allow",
            "Action": "ec2:DeleteNetworkInterface",
            "Resource": "arn:aws:ec2:*:*:network-interface/*",
            "Condition": {
                "StringEquals": {
                    "ec2:ResourceTag/managedByFIS": "true"
                }
            }
        },
        {
            "Sid": "RouteTableDisruptConnectivity10",
            "Effect": "Allow",
            "Action": "ec2:CreateManagedPrefixList",
            "Resource": "arn:aws:ec2:*:*:prefix-list/*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/managedByFIS": "true"
                }
            }
        },
        {
            "Sid": "RouteTableDisruptConnectivity11",
            "Effect": "Allow",
            "Action": "ec2:DeleteManagedPrefixList",
            "Resource": "arn:aws:ec2:*:*:prefix-list/*",
            "Condition": {
                "StringEquals": {
                    "ec2:ResourceTag/managedByFIS": "true"
                }
            }
        },
        {
            "Sid": "RouteTableDisruptConnectivity12",
            "Effect": "Allow",
            "Action": "ec2:ModifyManagedPrefixList",
            "Resource": "arn:aws:ec2:*:*:prefix-list/*",
            "Condition": {
                "StringEquals": {
                    "ec2:ResourceTag/managedByFIS": "true"
                }
            }
        },
        {
            "Sid": "RouteTableDisruptConnectivity13",
```

```
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeNetworkInterfaces",
                "ec2:DescribeVpcs",
                "ec2:DescribeVpcPeeringConnections",
                "ec2:DescribeManagedPrefixLists",
                "ec2:DescribeSubnets",
                "ec2:DescribeRouteTables",
                "ec2:DescribeVpcEndpoints"
            ],
            "Resource": "*"
        },
        {
            "Sid": "RouteTableDisruptConnectivity14",
            "Effect": "Allow",
            "Action": "ec2:ReplaceRouteTableAssociation",
            "Resource": [
                "arn:aws:ec2:*:*:subnet/*",
                "arn:aws:ec2:*:*:route-table/*"
            ]
        },
        {
            "Sid": "RouteTableDisruptConnectivity15",
            "Effect": "Allow",
            "Action": "ec2:GetManagedPrefixListEntries",
            "Resource": "arn:aws:ec2:*:*:prefix-list/*"
        },
        {
            "Sid": "RouteTableDisruptConnectivity16",
            "Effect": "Allow",
            "Action": "ec2:AssociateRouteTable",
            "Resource": [
                "arn:aws:ec2:*:*:subnet/*",
                "arn:aws:ec2:*:*:route-table/*"
            ]
        },
        {
            "Sid": "RouteTableDisruptConnectivity17",
            "Effect": "Allow",
            "Action": "ec2:DisassociateRouteTable",
            "Resource": [
                "arn:aws:ec2:*:*:route-table/*"
            ],
            "Condition": {
```

```
                "StringEquals": {
                    "ec2:ResourceTag/managedByFIS": "true"
                }
            }
        },
        {
            "Sid": "RouteTableDisruptConnectivity18",
            "Effect": "Allow",
            "Action": "ec2:DisassociateRouteTable",
            "Resource": [
                "arn:aws:ec2:*:*:subnet/*"
            ]
        },
        {
            "Sid": "RouteTableDisruptConnectivity19",
            "Effect": "Allow",
            "Action": "ec2:ModifyVpcEndpoint",
            "Resource": [
                "arn:aws:ec2:*:*:route-table/*"
            ],
            "Condition": {
                "StringEquals": {
                    "ec2:ResourceTag/managedByFIS": "true"
                }
            }
        },
        {
            "Sid": "RouteTableDisruptConnectivity20",
            "Effect": "Allow",
            "Action": "ec2:ModifyVpcEndpoint",
            "Resource": [
                "arn:aws:ec2:*:*:vpc-endpoint/*"
            ]
        },
        {
            "Sid": "TransitGatewayDisruptConnectivity1",
            "Effect": "Allow",
            "Action": [
                "ec2:DisassociateTransitGatewayRouteTable",
                "ec2:AssociateTransitGatewayRouteTable"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:transit-gateway-route-table/*",
                "arn:aws:ec2:*:*:transit-gateway-attachment/*"
```

```
            ]
        },
        {
            "Sid": "TransitGatewayDisruptConnectivity2",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeTransitGatewayPeeringAttachments",
                "ec2:DescribeTransitGatewayAttachments",
                "ec2:DescribeTransitGateways"
            ],
            "Resource": "*"
        },
        {
            "Sid": "S3CrossRegion1",
            "Effect": "Allow",
            "Action": [
                "s3:ListAllMyBuckets"
            ],
            "Resource": "*"
        },
        {
            "Sid": "S3CrossRegion2",
            "Effect": "Allow",
            "Action": [
                "tag:GetResources"
            ],
            "Resource": "*"
        },
        {
            "Sid": "S3CrossRegion3",
            "Effect": "Allow",
            "Action": [
                "s3:PauseReplication"
            ],
            "Resource": "arn:aws:s3:::*",
            "Condition": {
                "StringLike": {
                    "s3:DestinationRegion": "*"
                }
            }
        },
        {
            "Sid": "S3CrossRegion4",
            "Effect": "Allow",
```

```
            "Action": [
                "s3:GetReplicationConfiguration",
                "s3:PutReplicationConfiguration"
            ],
            "Resource": "arn:aws:s3:::*",
            "Condition": {
                "BoolIfExists": {
                    "s3:isReplicationPauseRequest": "true"
                }
            }
        },
        {
            "Sid": "DdbCrossRegion1",
            "Effect": "Allow",
            "Action": [
                "tag:GetResources"
            ],
            "Resource": "*"
        },
        {
            "Sid": "DdbCrossRegion",
            "Effect": "Allow",
            "Action": [
                "dynamodb:DescribeTable",
                "dynamodb:PutResourcePolicy",
                "dynamodb:GetResourcePolicy",
                "dynamodb:DeleteResourcePolicy"
            ],
            "Resource": [
                "arn:aws:dynamodb:*:*:table/*",
            ]
        }
    ]
}
```

Skenario Konten

Konten berikut mendefinisikan skenario. JSON ini dapat disimpan dan digunakan untuk membuat template eksperimen menggunakan create-experiment-templateperintah dari AWS Command Line Interface (AWS CLI). Untuk versi skenario terbaru, kunjungi pustaka skenario di konsol FIS.

```
{
```

```
        "targets": {
                "Transit-Gateway": {
                        "resourceType": "aws:ec2:transit-gateway",
                        "resourceTags": {
                                "TgwTag": "TgwValue"
                        },
                        "selectionMode": "ALL"
                },
                "Subnet": {
                        "resourceType": "aws:ec2:subnet",
                        "resourceTags": {
                                "SubnetKey": "SubnetValue"
                        },
                        "selectionMode": "ALL",
                        "parameters": {}
                },
                "S3-Bucket": {
                        "resourceType": "aws:s3:bucket",
                        "resourceTags": {
                                "S3Impact": "Allowed"
                        },
                        "selectionMode": "ALL"
                },
                "DynamoDB-Global-Table": {
                        "resourceType": "aws:dynamodb:global-table",
                        "resourceTags": {
                                "DisruptDynamoDb": "Allowed"
                        },
                        "selectionMode": "ALL"
                }
        },
        "actions": {
                "Disrupt-Transit-Gateway-Connectivity": {
                        "actionId": "aws:network:transit-gateway-disrupt-cross-region-
connectivity",
                        "parameters": {
                                "duration": "PT3H",
                                "region": "eu-west-1"
                        },
                        "targets": {
                                "TransitGateways": "Transit-Gateway"
                        }
                },
                "Disrupt-Subnet-Connectivity": {
```

```
                        "actionId": "aws:network:route-table-disrupt-cross-region-
connectivity",
                        "parameters": {
                                "duration": "PT3H",
                                "region": "eu-west-1"
                        },
                        "targets": {
                                "Subnets": "Subnet"
                        }
                },
                "Pause-S3-Replication": {
                        "actionId": "aws:s3:bucket-pause-replication",
                        "parameters": {
                                "duration": "PT3H",
                                "region": "eu-west-1"
                        },
                        "targets": {
                                "Buckets": "S3-Bucket"
                        }
                },
                "Pause-DynamoDB-Replication": {
                        "actionId": "aws:dynamodb:global-table-pause-replication",
                        "parameters": {
                                "duration": "PT3H"
                        },
                        "targets": {
                                "Tables": "DynamoDB-Global-Table"
                        }
                }
        },
        "stopConditions": [
                {
                        "source": "none"
                }
        ],
        "roleArn": "",
        "logConfiguration": {
                "logSchemaVersion": 2
        },
        "tags": {
                "Name": "Cross-Region: Connectivity"
        },
        "experimentOptions": {
                "accountTargeting": "single-account",
```

```
                "emptyTargetResolutionMode": "skip"
        },
        "description": "Block application network traffic from experiment Region to
 target Region and pause cross-Region replication"
}
```

Bekerja dengan eksperimen multi-akun untuk AWS FIS

Anda dapat membuat dan mengelola templat eksperimen multi-akun menggunakan AWS FIS konsol atau baris perintah. Anda membuat eksperimen multi-akun dengan menentukan opsi eksperimen penargetan akun sebagai`"multi-account"`, dan menambahkan konfigurasi akun target. Setelah membuat template eksperimen multi-akun, Anda dapat menggunakannya untuk menjalankan eksperimen.

Dengan eksperimen multi-akun, Anda dapat mengatur dan menjalankan skenario kegagalan dunia nyata pada aplikasi yang mencakup beberapa AWS akun dalam Wilayah. Anda menjalankan eksperimen multi-akun dari akun orkestrator yang memengaruhi sumber daya di beberapa akun target.

Saat Anda menjalankan eksperimen multi-akun, akun target dengan sumber daya yang terpengaruh akan diberitahukan melalui dasbor AWS Health mereka, memberikan kesadaran kepada pengguna di akun target. Dengan eksperimen multi-akun, Anda dapat:

- Jalankan skenario kegagalan dunia nyata pada aplikasi yang menjangkau beberapa akun dengan kontrol pusat dan pagar pembatas yang AWS FIS menyediakan.
- Kontrol efek eksperimen multi-akun menggunakan peran IAM dengan izin dan tag berbutir halus untuk menentukan cakupan setiap target.
- Secara terpusat melihat tindakan yang AWS FIS diambil di setiap akun dari AWS Management Console dan melalui AWS FIS log.
- Pantau dan audit panggilan API AWS FIS yang dilakukan di setiap akun dengan AWS CloudTrail.

Bagian ini membantu Anda memulai eksperimen multi-akun.

Topik

- Konsep untuk eksperimen multi-akun
- Praktik terbaik untuk eksperimen multi-akun
- Prasyarat untuk eksperimen multi-akun
- Buat templat eksperimen multi-akun
- Perbarui konfigurasi akun target
- Hapus konfigurasi akun target

Konsep untuk eksperimen multi-akun

Berikut ini adalah konsep kunci untuk eksperimen multi-akun:

- Akun orkestrator - Akun orkestrator bertindak sebagai akun pusat untuk mengonfigurasi dan mengelola eksperimen di AWS FIS Konsol, serta untuk memusatkan pencatatan. Akun orkestrator memiliki templat eksperimen dan AWS FIS eksperimen.
- Akun target - Akun target adalah akun AWS individual dengan sumber daya yang dapat dipengaruhi oleh eksperimen AWS FIS multi-akun.
- Konfigurasi akun target - Anda menentukan akun target yang merupakan bagian dari eksperimen dengan menambahkan konfigurasi akun target ke templat eksperimen. Konfigurasi akun target adalah elemen template eksperimen yang diperlukan untuk eksperimen multi-akun. Anda menentukan satu untuk setiap akun target dengan menetapkan ID AWS akun, peran IAM, dan deskripsi opsional.

Praktik terbaik untuk eksperimen multi-akun

Berikut ini adalah praktik terbaik untuk menggunakan eksperimen multi-akun:

- Saat Anda mengonfigurasi target untuk eksperimen multi-akun, sebaiknya penargetan dengan tag sumber daya yang konsisten di semua akun target. AWS FIS Eksperimen akan menyelesaikan sumber daya dengan tag yang konsisten di setiap akun target. Tindakan harus menyelesaikan setidaknya satu sumber daya target di akun target apa pun atau akan gagal, kecuali untuk eksperimen dengan `emptyTargetResolutionMode` set to`skip`. Kuota tindakan berlaku per akun. Jika Anda ingin menargetkan sumber daya berdasarkan sumber daya ARNs, batas akun tunggal yang sama per tindakan berlaku.
- Bila Anda menargetkan sumber daya di satu atau beberapa zona ketersediaan menggunakan parameter atau filter, Anda harus menentukan ID AZ, bukan nama AZ. ID AZ adalah pengidentifikasi unik dan konsisten untuk Availability Zone di seluruh akun. Untuk mempelajari cara menemukan ID AZ untuk zona ketersediaan di akun Anda, lihat Availability Zone IDs untuk sumber daya AWS Anda.

Prasyarat untuk eksperimen multi-akun

Untuk menggunakan kondisi berhenti untuk eksperimen multi-akun, Anda harus terlebih dahulu mengonfigurasi alarm lintas akun. Peran IAM ditentukan saat Anda membuat templat eksperimen multi-akun. Anda dapat membuat peran IAM yang diperlukan sebelum Anda membuat template.

Daftar isi

- Izin untuk eksperimen multi-akun
- Kondisi berhenti untuk eksperimen multi-akun (opsional)
- Tuas pengaman untuk eksperimen multi-akun (opsional)

Izin untuk eksperimen multi-akun

Eksperimen multi-akun menggunakan rantai peran IAM untuk memberikan izin AWS FIS untuk mengambil tindakan pada sumber daya di akun target. Untuk eksperimen multi-akun, Anda mengatur peran IAM di setiap akun target dan akun orkestrator. Peran IAM ini memerlukan hubungan kepercayaan antara akun target dan akun orkestrator, dan antara akun orkestra dan akun orkestra. AWS FIS

Peran IAM untuk akun target berisi izin yang diperlukan untuk mengambil tindakan pada sumber daya dan dibuat untuk templat eksperimen dengan menambahkan konfigurasi akun target. Anda akan membuat peran IAM untuk akun orkestrator dengan izin untuk mengambil peran akun target dan membangun hubungan kepercayaan dengan. AWS FIS Peran IAM ini digunakan sebagai template percobaan. `roleArn`

Untuk mempelajari lebih lanjut tentang rantai peran, lihat Syarat dan konsep Peran. di Panduan Pengguna IAM

Dalam contoh berikut, Anda akan menyiapkan izin untuk akun orkestrator A untuk menjalankan eksperimen dengan `aws:ebs:pause-volume-io` di akun target B.

1. Di akun B, buat peran IAM dengan izin yang diperlukan untuk menjalankan tindakan. Untuk izin yang diperlukan untuk setiap tindakan, lihatReferensi tindakan. Contoh berikut menunjukkan izin yang diberikan akun target untuk menjalankan tindakan EBS Jeda Volume IO. the section called “aws:ebs:pause-volume-io”

```
{
    "Version": "2012-10-17",
```

```
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeVolumes"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:PauseVolumeIO"
            ],
            "Resource": "arn:aws:ec2:region:accountIdB:volume/*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "tag:GetResources"
            ],
            "Resource": "*"
        }
    ]
}
```

2. Selanjutnya, tambahkan kebijakan kepercayaan di akun B yang menciptakan hubungan kepercayaan dengan akun A. Pilih nama untuk peran IAM untuk akun A, yang akan Anda buat di langkah 3.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
           "Effect": "Allow",
            "Principal": {
                "AWS": "AccountIdA"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringLike":{
                    "sts:ExternalId": "arn:aws:fis:region:accountIdA:experiment/*"
                },
                "ArnEquals": {
```

```
                    "aws:PrincipalArn": "arn:aws:iam::accountIdA:role/role_name"
                }
            }
        }
    ]
}
```

3. Di akun A, buat peran IAM. Nama peran ini harus cocok dengan peran yang Anda tentukan dalam kebijakan kepercayaan di langkah 2. Untuk menargetkan beberapa akun, Anda memberikan izin orkestrator untuk mengambil setiap peran. Contoh berikut menunjukkan izin untuk akun A untuk mengambil akun B. Jika Anda memiliki akun target tambahan, Anda akan menambahkan peran tambahan ARNs ke kebijakan ini. Anda hanya dapat memiliki satu peran ARN per akun target.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": [
                "arn:aws:iam::accountIdB:role/role_name"
            ]
        }
    ]
}
```

4. Peran IAM untuk akun A ini digunakan sebagai template percobaan. `roleArn` Contoh berikut menunjukkan kebijakan kepercayaan yang diperlukan dalam peran IAM yang memberikan AWS FIS izin untuk mengasumsikan akun A, akun orkestrator.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "fis.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole"
        }
```

```
    ]
}
```

Anda juga dapat menggunakan Stacksets untuk menyediakan beberapa peran IAM sekaligus. Untuk menggunakannya CloudFormation StackSets, Anda perlu mengatur StackSet izin yang diperlukan di AWS akun Anda. Untuk mempelajari lebih lanjut, lihat Bekerja dengan AWS CloudFormation StackSets.

Kondisi berhenti untuk eksperimen multi-akun (opsional)

Kondisi berhenti adalah mekanisme untuk menghentikan eksperimen jika mencapai ambang batas yang Anda definisikan sebagai alarm. Untuk mengatur kondisi berhenti untuk eksperimen multi-akun, Anda dapat menggunakan alarm lintas akun. Anda harus mengaktifkan berbagi di setiap akun target agar alarm tersedia ke akun orkestrator menggunakan izin hanya-baca. Setelah dibagikan, Anda dapat menggabungkan metrik dari akun target yang berbeda menggunakan Metric Math. Kemudian, Anda dapat menambahkan alarm ini sebagai kondisi berhenti untuk percobaan.

Untuk mempelajari selengkapnya tentang dasbor lintas akun, lihat Mengaktifkan fungsionalitas lintas akun di. CloudWatch

Tuas pengaman untuk eksperimen multi-akun (opsional)

Tuas pengaman digunakan untuk menghentikan semua eksperimen yang berjalan dan mencegah eksperimen baru dimulai. Anda mungkin ingin menggunakan tuas pengaman untuk mencegah eksperimen FIS selama periode waktu tertentu atau sebagai respons terhadap alarm kesehatan aplikasi. Setiap AWS akun memiliki tuas pengaman per Wilayah AWS. Ketika tuas pengaman diaktifkan, itu berdampak pada semua eksperimen yang berjalan di akun dan wilayah yang sama dengan tuas pengaman. Untuk menghentikan dan mencegah eksperimen multi-akun, tuas pengaman harus terlibat dalam akun dan wilayah yang sama di mana eksperimen berjalan.

Buat templat eksperimen multi-akun

Untuk mempelajari cara membuat template eksperimen melalui AWS Management Console

Lihat Buat template eksperimen.

Untuk membuat template eksperimen menggunakan CLI

1. Buka AWS Command Line Interface

2. Untuk membuat eksperimen dari file JSON yang disimpan dengan opsi eksperimen penargetan akun yang disetel ke `"multi-account"` (misalnya,`my-template.json`), ganti nilai placeholder *`italics`* dengan nilai Anda sendiri, lalu jalankan perintah berikut. create-experiment-template

```
aws fis create-experiment-template --cli-input-json file://my-template.json
```

   Ini akan mengembalikan template percobaan dalam respons. Salin `id` dari respon, yang merupakan ID dari template percobaan.

3. Jalankan create-target-account-configurationperintah untuk menambahkan konfigurasi akun target ke template percobaan. Ganti nilai placeholder *`italics`* dengan nilai Anda sendiri, menggunakan `id` dari langkah 2 sebagai nilai untuk `--experiment-template-id` parameter, dan kemudian jalankan yang berikut ini. Parameter `--description` bersifat opsional. Ulangi langkah ini untuk setiap akun target.

```
aws fis create-target-account-configuration --experiment-template-id EXTxxxxxxxxx
 --account-id 111122223333 --role-arn arn:aws:iam::111122223333:role/role-name --
description "my description"
```

4. Jalankan get-target-account-configurationperintah untuk mengambil detail untuk konfigurasi akun target tertentu.

```
aws fis get-target-account-configuration --experiment-template-id EXTxxxxxxxxx --
account-id 111122223333
```

5. Setelah Anda menambahkan semua konfigurasi akun target Anda, Anda dapat menjalankan list-target-account-configurationsperintah perintah untuk melihat bahwa konfigurasi akun target Anda telah dibuat.

```
aws fis list-target-account-configurations --experiment-template-id EXTxxxxxxxxx
```

   Anda juga dapat memverifikasi bahwa Anda telah menambahkan konfigurasi akun target dengan menjalankan get-experiment-templateperintah. Template akan mengembalikan field read-only `targetAccountConfigurationsCount` yang merupakan hitungan dari semua konfigurasi akun target pada template percobaan.

6. Saat Anda siap, Anda dapat menjalankan template eksperimen menggunakan perintah start-experiment.

```
aws fis start-experiment --experiment-template-id EXTxxxxxxxxx
```

Perbarui konfigurasi akun target

Anda dapat memperbarui konfigurasi akun target yang ada jika Anda ingin mengubah peran ARN atau deskripsi untuk akun tersebut. Saat Anda memperbarui konfigurasi akun target, perubahan tidak memengaruhi eksperimen yang sedang berjalan yang menggunakan templat.

Untuk memperbarui konfigurasi akun target menggunakan AWS Management Console

1. Buka AWS FIS konsol di https://console.aws.amazon.com/fis/.
2. Di panel navigasi, pilih Template eksperimen
3. Pilih template eksperimen, lalu pilih Tindakan, Perbarui templat eksperimen.
4. Di panel samping, pilih Langkah 3, Konfigurasikan akses layanan.
5. Ubah konfigurasi akun target, dan pilih Perbarui templat eksperimen.
6. Pilih Langkah 5, Tinjau dan buat.

Untuk memperbarui konfigurasi akun target menggunakan CLI

Jalankan update-target-account-configurationperintah ke perintah, ganti nilai placeholder *italics* dengan nilai Anda sendiri. `--description`Parameter `--role-arn` dan bersifat opsional, dan tidak akan diperbarui jika tidak disertakan.

```
aws fis update-target-account-configuration --experiment-template-id EXTxxxxxxxxx
 --account-id 111122223333 --role-arn arn:aws:iam::111122223333:role/role-name --
description "my description"
```

Hapus konfigurasi akun target

Jika Anda tidak lagi memerlukan konfigurasi akun target, Anda dapat menghapusnya. Saat Anda menghapus konfigurasi akun target, eksperimen apa pun yang berjalan yang menggunakan templat tidak akan terpengaruh. Eksperimen terus berjalan sampai selesai atau berhenti.

Untuk menghapus konfigurasi akun target menggunakan AWS Management Console

1. Buka AWS FIS konsol di https://console.aws.amazon.com/fis/.

2. Di panel navigasi, pilih Template eksperimen.
3. Pilih template eksperimen, dan pilih Tindakan, Perbarui.
4. Di panel samping, pilih Langkah 3, Konfigurasikan akses layanan.
5. Di bawah Konfigurasi akun target, pilih Hapus untuk akun target Peran ARN yang ingin Anda hapus.
6. Pilih Langkah 5, Tinjau dan buat.
7. Tinjau template dan pilih Perbarui template eksperimen. Saat diminta konfirmasi, masukkan `update` dan pilih Perbarui templat eksperimen.

Untuk menghapus konfigurasi akun target menggunakan CLI

Jalankan delete-target-account-configurationperintah, ganti nilai placeholder *`italics`* dengan nilai Anda sendiri.

```
aws fis update-target-account-configuration --experiment-template-id EXTxxxxxxxxx --
account-id 111122223333
```

Eksperimen penjadwalan

Dengan AWS Fault Injection Service (FIS), Anda dapat melakukan eksperimen injeksi kesalahan pada beban kerja AWS Anda. Eksperimen ini berjalan pada template yang berisi satu atau beberapa tindakan untuk dijalankan pada target tertentu. Anda sekarang dapat menjadwalkan eksperimen Anda sebagai tugas satu kali atau tugas berulang secara native dari Konsol FIS. Selain aturan terjadwal, FIS sekarang menawarkan kemampuan penjadwalan baru. FIS sekarang terintegrasi dengan EventBridge Scheduler dan membuat aturan atas nama Anda. EventBridge Scheduler adalah penjadwal tanpa server yang memungkinkan Anda membuat, menjalankan, dan mengelola tugas dari satu layanan terpusat yang dikelola.

Important

Penjadwal Eksperimen dengan tidak AWS Fault Injection Service tersedia di AWS GovCloud (AS-Timur) dan AWS GovCloud (AS-Barat).

Topik

- Buat peran penjadwal
- Buat jadwal percobaan
- Perbarui jadwal percobaan
- Menonaktifkan atau menghapus jadwal percobaan

Buat peran penjadwal

Peran eksekusi adalah peran IAM yang AWS FIS mengasumsikan untuk berinteraksi dengan EventBridge penjadwal dan untuk penjadwal Event Bridge untuk Memulai Eksperimen FIS. Anda melampirkan kebijakan izin ke peran ini untuk memberikan EventBridge Scheduler akses untuk menjalankan Eksperimen FIS. Langkah-langkah berikut menjelaskan cara membuat peran eksekusi baru dan kebijakan EventBridge untuk memungkinkan Memulai Eksperimen.

Buat peran penjadwal menggunakan AWS CLI

Ini adalah peran IAM yang diperlukan agar Event Bridge dapat menjadwalkan percobaan atas nama pelanggan.

1. Salin kebijakan JSON asumsi peran berikut dan simpan secara lokal sebagai. `fis-execution-role.json` Kebijakan kepercayaan ini memungkinkan EventBridge Scheduler untuk mengambil peran atas nama Anda.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "scheduler.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

2. Dari AWS Command Line Interface (AWS CLI), masukkan perintah berikut untuk membuat peran baru. Ganti `FisSchedulerExecutionRole` dengan nama yang ingin Anda berikan peran ini.

```
aws iam create-role --role-name FisSchedulerExecutionRole --assume-role-policy-
document file://fis-execution-role.json
```

Jika berhasil, Anda akan melihat output berikut:

```
{
    "Role": {
        "Path": "/",
        "RoleName": "FisSchedulerExecutionRole",
        "RoleId": "AROAZL22PDN5A6WKRBQNU",
        "Arn": "arn:aws:iam::123456789012:role/FisSchedulerExecutionRole",
        "CreateDate": "2023-08-24T17:23:05+00:00",
        "AssumeRolePolicyDocument": {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Principal": {
                        "Service": "scheduler.amazonaws.com"
                    },
                    "Action": "sts:AssumeRole"
```

```
                }
            ]
        }
    }
}
```

3. Untuk membuat kebijakan baru yang memungkinkan EventBridge Scheduler menjalankan eksperimen, salin JSON berikut dan simpan secara lokal sebagai. `fis-start-experiment-permissions.json` Kebijakan berikut memungkinkan EventBridge Scheduler untuk memanggil `fis:StartExperiment` tindakan pada semua templat eksperimen di akun Anda. Ganti `*` di bagian akhir `"arn:aws:fis:*:*:experiment-template/*"` dengan ID templat eksperimen Anda jika Anda ingin membatasi peran ke templat eksperimen tunggal.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "fis:StartExperiment",
            "Resource": [
                "arn:aws:fis:*:*:experiment-template/*",
                "arn:aws:fis:*:*:experiment/*"
            ]
        }
    ]
}
```

4. Jalankan perintah berikut untuk membuat kebijakan izin baru. Ganti `FisSchedulerPolicy` dengan nama yang ingin Anda berikan pada kebijakan ini.

```
aws iam create-policy --policy-name FisSchedulerPolicy --policy-document file://fis-start-experiment-permissions.json
```

Jika berhasil, Anda akan melihat output berikut. Perhatikan kebijakan ARN. Anda menggunakan ARN ini di langkah berikutnya untuk melampirkan kebijakan ke peran eksekusi kami.

```
{
    "Policy": {
        "PolicyName": "FisSchedulerPolicy",
```

```
        "PolicyId": "ANPAZL22PDN5ESVUWXLBD",
        "Arn": "arn:aws:iam::123456789012:policy/FisSchedulerPolicy",
        "Path": "/",
        "DefaultVersionId": "v1",
        "AttachmentCount": 0,
        "PermissionsBoundaryUsageCount": 0,
        "IsAttachable": true,
        "CreateDate": "2023-08-24T17:34:45+00:00",
        "UpdateDate": "2023-08-24T17:34:45+00:00"
    }
}
```

5. Jalankan perintah berikut untuk melampirkan kebijakan ke peran eksekusi Anda. Ganti `your-policy-arn` dengan ARN dari kebijakan yang Anda buat di langkah sebelumnya. Ganti `FisSchedulerExecutionRole` dengan nama peran eksekusi Anda.

```
aws iam attach-role-policy --policy-arn your-policy-arn --role-name
 FisSchedulerExecutionRole
```

   `attach-role-policy`Operasi tidak mengembalikan respons pada baris perintah.

6. Anda dapat membatasi penjadwal untuk hanya menjalankan template AWS FIS eksperimen yang memiliki nilai tag tertentu. Misalnya, kebijakan berikut memberikan `fis:StartExperiment` izin untuk semua AWS FIS eksperimen, tetapi membatasi penjadwal untuk hanya menjalankan templat eksperimen yang diberi tag. `Purpose=Schedule`

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "fis:StartExperiment",
            "Resource": "arn:aws:fis:*:*:experiment/*"
        },
        {
            "Effect": "Allow",
            "Action": "fis:StartExperiment",
            "Resource": "arn:aws:fis:*:*:experiment-template/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/Purpose": "Schedule"
```

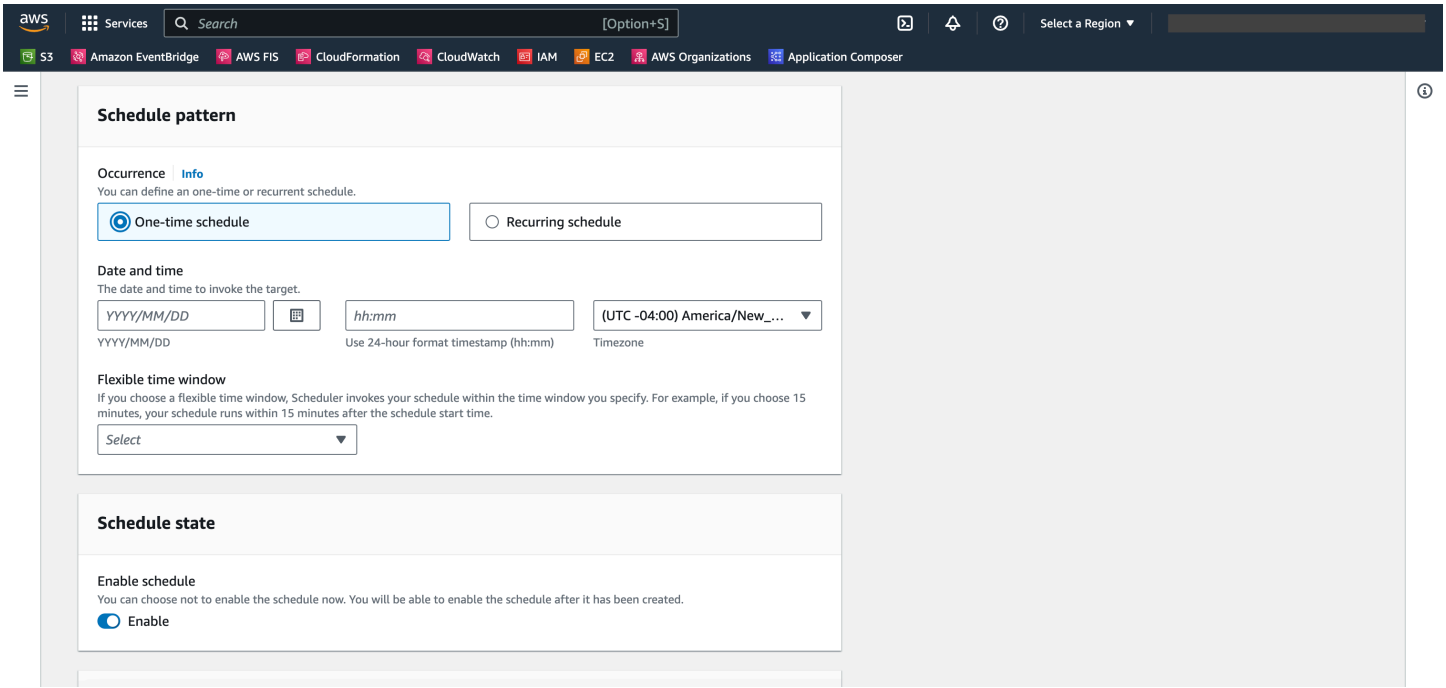
```
                }
            }
        }
    ]
}
```

Buat jadwal percobaan

Sebelum Anda menjadwalkan percobaan, Anda memerlukan satu atau lebih Komponen template percobaan untuk jadwal Anda untuk dipanggil. Anda dapat menggunakan sumber daya AWS yang ada, atau membuat yang baru.

Setelah template eksperimen dibuat, klik Tindakan dan pilih Jadwalkan eksperimen. Anda akan dialihkan ke halaman percobaan jadwal. Nama jadwal akan diisi untuk Anda.

Ikuti bagian pola jadwal dan pilih jadwal satu kali atau berulang. Isi kolom input yang diperlukan dan arahkan ke izin.



Status jadwal akan diaktifkan secara default. Catatan: jika Anda menonaktifkan status jadwal, percobaan tidak akan dijadwalkan bahkan jika Anda membuat jadwal.

AWS FIS Penjadwal Eksperimen dibangun di atas EventBridge Scheduler. Anda dapat merujuk dokumentasi untuk berbagai jenis jadwal yang didukung.

Untuk memperbarui jadwal menggunakan konsol

1. Buka konsol AWS FIS.
2. Di panel navigasi kiri, pilih Template Eksperimen.
3. Pilih Template Eksperimen yang ingin Anda buat jadwalnya.
4. Klik Tindakan, dan pilih Jadwalkan Eksperimen dari menu tarik-turun.
   a. Di bawah nama Jadwal, nama diisi secara otomatis.
   b. Di bawah Pola jadwal, pilih Jadwal berulang.
   c. Di bawah Jenis jadwal, Anda dapat memilih jadwal berdasarkan tarif, lihat jenis jadwal.
   d. Di bawah Ekspresi Rate, pilih tingkat yang lebih lambat dari waktu eksekusi eksperimen Anda, misalnya 5 menit.
   e. Di bawah Jangka Waktu, pilih Zona Waktu Anda.
   f. Di bawah Tanggal dan Waktu Mulai, tentukan tanggal dan waktu mulai.
   g. Di bawah Tanggal dan Waktu Berakhir, tentukan tanggal dan waktu berakhir
   h. Di bawah Status Jadwal, alihkan Opsi Aktifkan Jadwal.
   i. Di bawah Izin, pilih Gunakan peran yang ada, lalu cari `FisSchedulerExecutionRole`.
   j. Pilih Berikutnya.
5. Pilih Tinjau dan buat jadwal, tinjau detail penjadwal Anda, lalu pilih Buat jadwal.

Perbarui jadwal percobaan

Anda dapat memperbarui jadwal percobaan sehingga terjadi pada tanggal dan waktu tertentu yang cocok untuk Anda.

Untuk memperbarui eksekusi eksperimen menggunakan konsol

1. Buka konsol Amazon FIS.
2. Di panel navigasi, pilih Template Eksperimen.
3. Pilih Jenis sumber daya: Template Eksperimen yang jadwalnya sudah dibuat.
4. Klik pada ID Eksperimen untuk template. Kemudian arahkan ke jadwal Tab.
5. Periksa apakah ada jadwal yang ada terkait dengan eksperimen. Pilih jadwal yang terkait dan Klik tombolnya Perbarui Jadwal.

Menonaktifkan atau menghapus jadwal percobaan

Untuk menghentikan eksperimen mengeksekusi atau berjalan sesuai jadwal, Anda dapat menghapus atau menonaktifkan aturan. Langkah-langkah berikut memandu Anda melalui cara menghapus atau menonaktifkan Eksekusi Eksperimen menggunakan AWS konsol.

Untuk menghapus atau menonaktifkan aturan

1. Buka konsol Amazon FIS.
2. Di panel navigasi, pilih Template Eksperimen.
3. Pilih Jenis sumber daya: Template Eksperimen yang jadwalnya sudah dibuat.
4. Klik pada ID Eksperimen untuk template. Kemudian arahkan ke jadwal Tab.
5. Periksa apakah ada jadwal yang ada terkait dengan eksperimen. Pilih jadwal yang terkait dan Klik tombolnya Perbarui Jadwal.
6. Lakukan salah satu hal berikut ini:
   a. Untuk menghapus jadwal, pilih tombol di sebelah aturan Hapus Jadwal. Ketik `delete` dan klik tombol Hapus Jadwal.
   b. Untuk menonaktifkan jadwal, pilih tombol di sebelah aturan Nonaktifkan Jadwal. Ketik `disable` dan klik tombol Nonaktifkan Jadwal.

Tuas Pengaman untuk AWS FIS

Tuas pengaman digunakan untuk menghentikan semua eksperimen yang berjalan dan mencegah eksperimen baru dimulai. Anda mungkin ingin menggunakan tuas pengaman untuk mencegah eksperimen FIS selama periode waktu tertentu atau sebagai respons terhadap alarm kesehatan aplikasi. Setiap AWS akun memiliki tuas pengaman per Wilayah AWS.

Untuk eksperimen yang sedang berlangsung yang dihentikan oleh tuas pengaman, Anda hanya membayar durasi tindakan yang berjalan sebelum percobaan dihentikan. Eksperimen yang dicegah untuk memulai tidak akan menimbulkan biaya apa pun. Bagian berikut memberikan informasi tentang cara memulai menggunakan tuas pengaman.

Topik

- Konsep untuk tuas pengaman
- Bekerja dengan tuas pengaman

Konsep untuk tuas pengaman

Tuas pengaman dapat dipasang atau dilepaskan.

- Jika terlepas, eksperimen FIS diperbolehkan. Secara default, tuas pengaman dilepaskan.
- Jika terlibat, eksperimen yang sedang berlangsung dihentikan dan tidak ada eksperimen baru yang diizinkan untuk dimulai.

Eksperimen yang dipengaruhi oleh tuas pengaman akan berakhir di salah satu status berikut:

- Berhenti, jika percobaan berjalan ketika tuas pengaman diaktifkan.
- Dibatalkan, jika percobaan dimulai ketika tuas pengaman sudah diaktifkan.

Anda tidak dapat melanjutkan atau menjalankan kembali eksperimen yang telah dihentikan atau dibatalkan. Namun, Anda dapat memulai eksperimen baru menggunakan templat eksperimen yang sama setelah tuas pengaman dilepaskan.

Sumber daya tuas Saftey

Tuas pengaman adalah sumber daya yang ditentukan oleh Amazon Resource Name (ARN). Tuas pengaman meliputi parameter berikut:

- Status, yang terlibat atau tidak terlibat.
- Alasan, yang merupakan input string oleh pengguna untuk mencatat mengapa status tuas pengaman diubah.

Bekerja dengan tuas pengaman

Bagian ini akan merinci cara melihat, menggunakan, dan melepaskan tuas pengaman menggunakan AWS FIS konsol atau baris perintah.

Melihat tuas pengaman

Anda dapat melihat status tuas pengaman untuk akun dan wilayah Anda dengan mengikuti langkah-langkah di bawah ini.

Untuk melihat tuas pengaman menggunakan konsol

1. Buka AWS FIS konsol
2. Di panel navigasi, pilih Eksperimen.
3. Jika tuas pengaman diaktifkan, Anda akan melihat spanduk peringatan di bagian atas halaman. Jika tidak ada spanduk peringatan, tuas pengaman dilepaskan.

Untuk melihat tuas pengaman menggunakan CLI

- Gunakan perintah berikut ini.

```
aws fis get-safety-lever --id "default"
```

Tuas pengaman dapat berada di salah satu kondisi berikut:

- Dilepas - Tuas pengaman tidak memengaruhi eksperimen apa pun. Eksperimen dapat berjalan dengan bebas. Tuas pengaman dilepaskan secara default.

- Melibatkan - Tuas pengaman berubah dari yang dirusak menjadi aktif. Mungkin masih ada eksperimen yang belum dihentikan. Tuas pengaman tidak dapat diubah saat dalam kondisi ini.
- Terlibat - Tuas pengaman aktif dan tidak ada eksperimen yang berjalan. Eksperimen baru apa pun yang mencoba memulai saat tuas pengaman diaktifkan akan dibatalkan.

Melibatkan tuas pengaman

Untuk menggunakan tuas pengaman menggunakan konsol

1. Buka AWS FIS konsol
2. Di panel navigasi, pilih Eksperimen.
3. Pilih tombol Hentikan semua eksperimen.
4. Masukkan alasan untuk menggunakan tuas pengaman.
5. Pilih Konfirmasi.

Untuk meningkatkan tuas pengaman menggunakan CLI

- Gunakan perintah berikut ini. Isi bidang alasan dengan tanggapan Anda sendiri.

```
aws fis  update-safety-lever-state --id "default" --state
 "status=engaged,reason=xxxxx"
```

Melepaskan tuas pengaman

Untuk melepaskan tuas pengaman menggunakan konsol

1. Buka AWS FIS konsol
2. Di panel navigasi, pilih Eksperimen.
3. Pilih tombol tuas pengaman Lepaskan.
4. Masukkan alasan untuk melepaskan tuas pengaman.
5. Pilih Konfirmasi.

Untuk melepaskan tuas pengaman menggunakan CLI

- Gunakan perintah berikut ini.

```
aws fis  update-safety-lever-state --id "default" --state
 "status=disengaged,reason=recovered"
```

Memantau AWS eksperimen FIS

Anda dapat menggunakan alat berikut untuk memantau kemajuan dan dampak eksperimen AWS Fault Injection Service (AWS FIS) Anda.

AWS Konsol FIS dan AWS CLI

Gunakan konsol AWS FIS atau AWS CLI untuk memantau kemajuan eksperimen yang sedang berjalan. Anda dapat melihat status setiap tindakan dalam percobaan, dan hasil dari setiap tindakan. Untuk informasi selengkapnya, lihat the section called “Lihat eksperimen Anda”.

CloudWatch metrik penggunaan dan alarm

Gunakan metrik CloudWatch penggunaan untuk memberikan visibilitas ke dalam penggunaan sumber daya akun Anda. AWS Metrik penggunaan FIS sesuai dengan kuota AWS layanan. Anda dapat mengonfigurasi alarm yang memberi tahu Anda saat penggunaan mendekati kuota layanan. Untuk informasi selengkapnya, lihat Monitor menggunakan CloudWatch.

Anda juga dapat membuat kondisi berhenti untuk eksperimen AWS FIS Anda dengan membuat CloudWatch alarm yang menentukan kapan eksperimen keluar dari batas. Ketika alarm dipicu, percobaan berhenti. Untuk informasi selengkapnya, lihat Hentikan kondisi. Untuk informasi selengkapnya tentang membuat CloudWatch CloudWatch alarm, lihat Membuat Alarm Berdasarkan Ambang Statis dan Membuat CloudWatch Alarm Berdasarkan Deteksi Anomali di Panduan Pengguna Amazon CloudWatch .

AWS Pencatatan percobaan FIS

Aktifkan pencatatan eksperimen untuk menangkap informasi terperinci tentang eksperimen Anda saat dijalankan. Untuk informasi selengkapnya, lihat Pencatatan percobaan.

Peristiwa perubahan status eksperimen

Amazon EventBridge memungkinkan Anda merespons secara otomatis peristiwa sistem atau perubahan sumber daya. AWS FIS memancarkan pemberitahuan ketika keadaan eksperimen berubah. Anda dapat membuat aturan untuk acara yang Anda minati yang menentukan tindakan otomatis yang akan diambil saat acara cocok dengan aturan. Misalnya, mengirim pemberitahuan ke topik Amazon SNS atau menjalankan fungsi Lambda. Untuk informasi selengkapnya, lihat Monitor menggunakan EventBridge.

CloudTrail log

Gunakan AWS CloudTrail untuk menangkap informasi terperinci tentang panggilan yang dilakukan ke AWS FIS API dan menyimpannya sebagai file log di Amazon S3. CloudTrail juga mencatat panggilan yang dibuat ke layanan APIs untuk sumber daya tempat Anda menjalankan eksperimen. Anda dapat menggunakan CloudTrail log ini untuk menentukan panggilan mana yang dilakukan, alamat IP sumber dari mana panggilan itu berasal, siapa yang melakukan panggilan, kapan panggilan dilakukan, dan sebagainya.

AWS Pemberitahuan Dasbor Kesehatan

AWS Health menyediakan visibilitas berkelanjutan ke kinerja sumber daya Anda dan ketersediaan AWS layanan dan akun Anda. Ketika Anda memulai percobaan, AWS FIS memancarkan pemberitahuan ke Dasbor AWS Kesehatan Anda. Pemberitahuan hadir selama percobaan di setiap akun yang berisi sumber daya yang ditargetkan dalam eksperimen, termasuk eksperimen multi-akun. Eksperimen multi-akun dengan hanya tindakan yang tidak menyertakan target, seperti `aws:ssm:start-automation-execution` dan`aws:fis:wait`, tidak akan memancarkan pemberitahuan. Informasi tentang peran yang digunakan untuk memungkinkan eksperimen akan dicantumkan di bawah Sumber daya yang terpengaruh. Untuk mempelajari lebih lanjut tentang Dasbor AWS Health, lihat Dasbor AWS Health di Panduan Pengguna AWS Health.

> **Note**
>
> AWS Health menghadirkan acara dengan upaya terbaik.

Pantau metrik penggunaan AWS FIS menggunakan Amazon CloudWatch

Anda dapat menggunakan Amazon CloudWatch untuk memantau dampak eksperimen AWS FIS pada target. Anda juga dapat memantau penggunaan AWS FIS Anda.

Untuk informasi lebih lanjut tentang melihat keadaan eksperimen, lihatLihat eksperimen Anda.

Pantau AWS eksperimen FIS

Saat Anda merencanakan eksperimen AWS FIS Anda, identifikasi CloudWatch metrik yang dapat Anda gunakan untuk mengidentifikasi baseline atau “kondisi tunak” untuk jenis sumber daya target

untuk eksperimen. Setelah memulai eksperimen, Anda dapat memantau CloudWatch metrik tersebut untuk target yang dipilih melalui templat eksperimen.

Untuk informasi selengkapnya tentang CloudWatch metrik yang tersedia untuk jenis sumber daya target yang didukung oleh AWS FIS, lihat berikut ini:

- Pantau instans Anda menggunakan CloudWatch
- Metrik Amazon ECS CloudWatch
- Memantau metrik Amazon RDS menggunakan CloudWatch
- Memantau metrik Run Command menggunakan CloudWatch

AWS Metrik penggunaan FIS

Anda dapat menggunakan metrik CloudWatch penggunaan untuk memberikan visibilitas ke dalam penggunaan sumber daya akun Anda. Gunakan metrik ini untuk memvisualisasikan penggunaan layanan Anda saat ini pada CloudWatch grafik dan dasbor.

AWS Metrik penggunaan FIS sesuai dengan kuota AWS layanan. Anda dapat mengonfigurasi alarm yang memberi tahu Anda saat penggunaan mendekati kuota layanan. Untuk informasi selengkapnya tentang CloudWatch alarm, lihat Panduan CloudWatch Pengguna Amazon.

AWS FIS menerbitkan metrik berikut di ruang nama AWS/Usage.

Metrik	Deskripsi
`ResourceCount`	Jumlah sumber daya tertentu yang berjalan di akun Anda. Sumber daya ditentukan oleh dimensi yang terkait dengan metrik.

Dimensi berikut digunakan untuk menyempurnakan metrik penggunaan yang diterbitkan oleh AWS FIS.

Dimensi	Deskripsi
`Service`	Nama AWS layanan yang berisi sumber daya. Untuk metrik penggunaan AWS FIS, nilai untuk dimensi ini adalah. `FIS`

Dimensi	Deskripsi
`Type`	Tipe entitas yang dilaporkan. Saat ini, satu-satu nya nilai yang valid untuk metrik penggunaan AWS FIS adalah. `Resource`
`Resource`	Tipe sumber daya yang sedang berjalan. Nilai yang mungkin adalah `ExperimentTemplate s` untuk templat eksperimen, dan `ActiveExp eriments` untuk eksperimen aktif.
`Class`	Dimensi ini dicadangkan untuk penggunaan masa depan.

Pantau eksperimen AWS FIS menggunakan Amazon EventBridge

Ketika keadaan eksperimen berubah, AWS FIS memancarkan pemberitahuan. Pemberitahuan ini tersedia sebagai acara melalui Amazon EventBridge (sebelumnya disebut CloudWatch Acara). AWS FIS memancarkan peristiwa ini atas dasar upaya terbaik. Acara dikirimkan ke EventBridge dalam waktu dekat.

Dengan EventBridge, Anda dapat membuat aturan yang memicu tindakan terprogram sebagai respons terhadap suatu peristiwa. Misalnya, Anda dapat mengonfigurasi aturan yang memanggil topik SNS untuk mengirim pemberitahuan email atau memanggil fungsi Lambda untuk mengambil beberapa tindakan.

Untuk informasi selengkapnya EventBridge, lihat Memulai Amazon EventBridge di Panduan EventBridge Pengguna Amazon.

Berikut ini adalah sintaks dari peristiwa perubahan status eksperimen:

```
{
    "version": "0",
    "id": "12345678-1234-1234-1234-123456789012",
    "detail-type": "FIS Experiment State Change",
    "source": "aws.fis",
    "account": "123456789012",
    "time": "yyyy-mm-ddThh:mm:ssZ",
    "region": "region",
```

```
    "resources": [
        "arn:aws:fis:region:account_id:experiment/experiment-id"
    ],
    "detail": {
        "experiment-id": "EXPaBCD1efg2HIJkL3",
        "experiment-template-id": "EXTa1b2c3de5f6g7h",
        "new-state": {
            "status": "new_value",
            "reason": "reason_string"
        },
        "old-state": {
           "status": "old_value",
           "reason": "reason_string"
        }
    }
}
```

`experiment-id`

ID percobaan yang statusnya berubah.

`experiment-template-id`

ID template percobaan yang digunakan oleh percobaan.

`new_value`

Keadaan baru percobaan. Nilai yang mungkin adalah:

- `completed`
- `failed`
- `initiating`
- `running`
- `stopped`
- `stopping`

`old_value`

Keadaan percobaan sebelumnya. Nilai yang mungkin adalah:

- `initiating`
- `pending`
- `running`

- `stopping`

Pencatatan percobaan untuk AWS FIS

Anda dapat menggunakan pencatatan eksperimen untuk menangkap informasi terperinci tentang eksperimen Anda saat dijalankan.

Anda dikenakan biaya untuk pencatatan percobaan berdasarkan biaya yang terkait dengan setiap jenis tujuan log. Untuk informasi selengkapnya, lihat CloudWatch Harga Amazon (di bawah Tingkat Berbayar, Log, Log Penjual) dan Harga Amazon S3.

Izin

Anda harus memberikan izin AWS FIS untuk mengirim log ke setiap tujuan log yang Anda konfigurasikan. Untuk informasi selengkapnya, lihat berikut ini di Panduan Pengguna Amazon CloudWatch Logs:

- Log dikirim ke CloudWatch Log
- Log dikirim ke Amazon S3

Skema log

Berikut ini adalah skema yang digunakan dalam pencatatan percobaan. Versi skema saat ini adalah 2. Bidang untuk `details` tergantung pada nilai`log_type`. Bidang untuk `resolved_targets` tergantung pada nilai`target_type`. Untuk informasi selengkapnya, lihat the section called “Contoh catatan log”.

```
{
    "id": "EXP123abc456def789",
    "log_type": "experiment-start | target-resolution-start | target-resolution-detail
 | target-resolution-end | action-start | action-error | action-end | experiment-end",
    "event_timestamp": "yyyy-mm-ddThh:mm:ssZ",
    "version": "2",
    "details": {
        "account_id":"123456789012",
        "action_end_time": "yyyy-mm-ddThh:mm:ssZ",
        "action_id": "String",
        "action_name": "String",
        "action_start_time": "yyyy-mm-ddThh:mm:ssZ",
```

```
        "action_state": {
            "status": "pending | initiating | running | completed | cancelled |
 stopping | stopped | failed",
            "reason": "String"
        },
        "action_targets": "String to string map",
        "error_information": "String",
        "experiment_end_time": "yyyy-mm-ddThh:mm:ssZ",
        "experiment_state": {
            "status": "pending | initiating | running | completed | stopping | stopped
 | failed",
            "reason": "String"
        },
        "experiment_start_time": "yyyy-mm-ddThh:mm:ssZ",
        "experiment_template_id": "String",
        "page": Number,
        "parameters": "String to string map",
        "resolved_targets": [
            {
               "field": "value"
            }
        ],
        "resolved_targets_count": Number,
        "status": "failed | completed",
        "target_name": "String",
        "target_resolution_end_time": "yyyy-mm-ddThh:mm:ssZ",
        "target_resolution_start_time": "yyyy-mm-ddThh:mm:ssZ",
        "target_type": "String",
        "total_pages": Number,
        "total_resolved_targets_count": Number

    }
}
```

Catatan perilisan

- Versi 2 memperkenalkan:
  - `target_type`Bidang dan mengubah `resolved_targets` bidang dari daftar ARNs ke daftar objek. Bidang yang valid untuk `resolved_targets` objek bergantung pada nilai`target_type`, yang merupakan jenis sumber daya target.
  - Jenis `action-error` dan `target-resolution-detail` acara yang menambahkan `account_id` bidang.

- Versi 1 adalah rilis awal.

Log tujuan

AWS FIS mendukung pengiriman log ke tujuan berikut:

- Bucket Amazon S3
- Grup CloudWatch log Amazon Logs

Pengiriman log S3

Log dikirim ke lokasi berikut.

```
bucket-and-optional-prefix/AWSLogs/account-id/fis/region/experiment-
id/YYYY/MM/DD/account-id_awsfislogs_region_experiment-id_YYYYMMDDHHMMZ_hash.log
```

Ini bisa memakan waktu beberapa menit sebelum log dikirim ke ember.

CloudWatch Log pengiriman log

Log dikirim ke aliran lognamed /aws/fis/*experiment-id*.

Log dikirim ke grup log dalam waktu kurang dari satu menit.

Contoh catatan log

Berikut ini adalah contoh catatan log untuk eksperimen yang menjalankan aws:ec2:reboot-instances tindakan pada EC2 instance yang dipilih secara acak.

Catatan

- percobaan-mulai
- target-resolution-start
- target-resolution-detail
- target-resolution-end
- aksi-mulai
- aksi-akhir
- tindakan-kesalahan

- ujung-eksperimen

percobaan-mulai

Berikut ini adalah contoh catatan untuk `experiment-start` acara tersebut.

```
{
    "id": "EXPhjAXCGY78HV2a4A",
    "log_type": "experiment-start",
    "event_timestamp": "2023-05-31T18:50:45Z",
    "version": "2",
    "details": {
        "experiment_template_id": "EXTCDh1M8HHkhxoaQ",
        "experiment_start_time": "2023-05-31T18:50:43Z"
    }
}
```

target-resolution-start

Berikut ini adalah contoh catatan untuk `target-resolution-start` acara tersebut.

```
{
    "id": "EXPhjAXCGY78HV2a4A",
    "log_type": "target-resolution-start",
    "event_timestamp": "2023-05-31T18:50:45Z",
    "version": "2",
    "details": {
        "target_resolution_start_time": "2023-05-31T18:50:45Z",
        "target_name": "EC2InstancesToReboot"
    }
}
```

target-resolution-detail

Berikut ini adalah contoh catatan untuk `target-resolution-detail` acara tersebut. Jika resolusi target gagal, catatan juga menyertakan `error_information` bidang.

```
{
    "id": "EXPhjAXCGY78HV2a4A",
```

```
    "log_type": "target-resolution-detail",
    "event_timestamp": "2023-05-31T18:50:45Z",
    "version": "2",
    "details": {
        "target_resolution_end_time": "2023-05-31T18:50:45Z",
        "target_name": "EC2InstancesToReboot",
        "target_type": "aws:ec2:instance",
        "account_id": "123456789012",
        "resolved_targets_count": 2,
        "status": "completed"

    }
}
```

target-resolution-end

Jika resolusi target gagal, catatan juga menyertakan `error_information` bidang. Jika `total_pages` lebih besar dari 1, jumlah target yang diselesaikan melebihi batas ukuran untuk satu catatan. Ada `target-resolution-end` catatan tambahan yang berisi target yang diselesaikan yang tersisa.

Berikut ini adalah contoh catatan untuk `target-resolution-end` acara untuk suatu EC2 tindakan.

```
{
    "id": "EXPhjAXCGY78HV2a4A",
    "log_type": "target-resolution-end",
    "event_timestamp": "2023-05-31T18:50:45Z",
    "version": "2",
    "details": {
        "target_resolution_end_time": "2023-05-31T18:50:46Z",
        "target_name": "EC2InstanceToReboot",
        "target_type": "aws:ec2:instance",
        "resolved_targets": [
            {
                "arn": "arn:aws:ec2:us-east-1:123456789012:instance/
i-0f7ee2abffc330de5"
            }
        ],
        "page": 1,
        "total_pages": 1
    }
}
```

Berikut ini adalah contoh catatan untuk `target-resolution-end` acara untuk tindakan EKS.

```
{
    "id": "EXP24YfiucfyVPJpEJn",
    "log_type": "target-resolution-end",
    "event_timestamp": "2023-05-31T18:50:45Z",
    "version": "2",
    "details": {
        "target_resolution_end_time": "2023-05-31T18:50:46Z",
        "target_name": "myPods",
        "target_type": "aws:eks:pod",
        "resolved_targets": [
            {
                "pod_name": "example-696fb6498b-sxhw5",
                "namespace": "default",
                "cluster_arn": "arn:aws:eks:us-east-1:123456789012:cluster/fis-demo-
cluster",
                "target_container_name": "example"
            }
        ],
        "page": 1,
        "total_pages": 1
    }
}
```

aksi-mulai

Berikut ini adalah contoh catatan untuk `action-start` acara tersebut. Jika template percobaan menentukan parameter untuk tindakan, catatan juga menyertakan `parameters` bidang.

```
{
    "id": "EXPhjAXCGY78HV2a4A",
    "log_type": "action-start",
    "event_timestamp": "2023-05-31T18:50:56Z",
    "version": "2",
    "details": {
        "action_name": "Reboot",
        "action_id": "aws:ec2:reboot-instances",
        "action_start_time": "2023-05-31T18:50:56Z",
        "action_targets": {"Instances":"EC2InstancesToReboot"}
    }
}
```

tindakan-kesalahan

Berikut ini adalah contoh catatan untuk `action-error` acara tersebut. Acara ini hanya dikembalikan ketika suatu tindakan gagal. Itu dikembalikan untuk setiap akun di mana tindakan gagal.

```
{
    "id": "EXPhjAXCGY78HV2a4A",
    "log_type": "action-error",
    "event_timestamp": "2023-05-31T18:50:56Z",
    "version": "2",
    "details": {
        "action_name": "pause-io",
        "action_id": "aws:ebs:pause-volume-io",
        "account_id": "123456789012",
        "action_state": {
            "status": "failed",
            "reason":"Unable to start Pause Volume IO. Target volumes must be attached
 to an instance type based on the Nitro system. VolumeId(s): [vol-1234567890abcdef0]:"
        }
    }
}
```

aksi-akhir

Berikut ini adalah contoh catatan untuk `action-end` acara tersebut.

```
{
    "id": "EXPhjAXCGY78HV2a4A",
    "log_type": "action-end",
    "event_timestamp": "2023-05-31T18:50:56Z",
    "version": "2",
    "details": {
        "action_name": "Reboot",
        "action_id": "aws:ec2:reboot-instances",
        "action_end_time": "2023-05-31T18:50:56Z",
        "action_state": {
            "status": "completed",
            "reason": "Action was completed."
        }
    }
}
```

ujung-eksperimen

Berikut ini adalah contoh catatan untuk `experiment-end` acara tersebut.

```
{
    "id": "EXPhjAXCGY78HV2a4A",
    "log_type": "experiment-end",
    "event_timestamp": "2023-05-31T18:50:57Z",
    "version": "2",
    "details": {
        "experiment_end_time": "2023-05-31T18:50:57Z",
        "experiment_state": {
            "status": "completed",
            "reason": "Experiment completed"
        }
    }
}
```

Aktifkan pencatatan percobaan

Pencatatan percobaan dinonaktifkan secara default. Untuk menerima log eksperimen untuk eksperimen, Anda harus membuat eksperimen dari templat eksperimen dengan pencatatan diaktifkan. Pertama kali Anda menjalankan eksperimen yang dikonfigurasi untuk menggunakan tujuan yang belum pernah digunakan sebelumnya untuk logging, kami menunda eksperimen untuk mengonfigurasi pengiriman log ke tujuan ini, yang memakan waktu sekitar 15 detik.

Untuk mengaktifkan pencatatan eksperimen menggunakan konsol

1. Buka konsol AWS FIS di https://console.aws.amazon.com/fis/.
2. Di panel navigasi, pilih Template eksperimen.
3. Pilih template eksperimen, lalu pilih Tindakan, Perbarui templat eksperimen.
4. Untuk Log, konfigurasikan opsi tujuan. Untuk mengirim log ke bucket S3, pilih Kirim ke bucket Amazon S3 dan masukkan nama bucket dan awalan. Untuk mengirim CloudWatch log ke Log, pilih Kirim ke CloudWatch Log dan masukkan grup log.
5. Pilih Perbarui templat eksperimen.

Untuk mengaktifkan pencatatan eksperimen menggunakan AWS CLI

Gunakan update-experiment-templateperintah dan tentukan konfigurasi log.

Nonaktifkan pencatatan percobaan

Jika Anda tidak lagi ingin menerima log untuk eksperimen Anda, Anda dapat menonaktifkan pencatatan eksperimen.

Untuk menonaktifkan pencatatan eksperimen menggunakan konsol

1. Buka konsol AWS FIS di https://console.aws.amazon.com/fis/.
2. Di panel navigasi, pilih Template eksperimen.
3. Pilih template eksperimen, lalu pilih Tindakan, Perbarui templat eksperimen.
4. Untuk Log, hapus Kirim ke bucket Amazon S3 dan Kirim ke CloudWatch Log.
5. Pilih Perbarui templat eksperimen.

Untuk menonaktifkan pencatatan eksperimen menggunakan AWS CLI

Gunakan update-experiment-templateperintah dan tentukan konfigurasi log kosong.

Log panggilan API dengan AWS CloudTrail

AWS AWS Fault Injection Service (FIS) terintegrasi dengan AWS CloudTrail, layanan yang menyediakan catatan tindakan yang diambil oleh pengguna, peran, atau AWS layanan di AWS FIS. CloudTrail menangkap semua panggilan API untuk AWS FIS sebagai peristiwa. Panggilan yang ditangkap termasuk panggilan dari konsol AWS FIS dan panggilan kode ke operasi API AWS FIS. Jika Anda membuat jejak, Anda dapat mengaktifkan pengiriman CloudTrail acara secara berkelanjutan ke bucket Amazon S3, termasuk acara untuk AWS FIS. 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 AWS FIS, alamat IP dari mana permintaan dibuat, siapa yang membuat permintaan, kapan dibuat, dan detail tambahan.

Untuk mempelajari selengkapnya CloudTrail, lihat Panduan AWS CloudTrail Pengguna.

Gunakan CloudTrail

CloudTrail diaktifkan pada Akun AWS saat Anda membuat akun. Ketika aktivitas terjadi di AWS FIS, aktivitas tersebut dicatat dalam suatu CloudTrail peristiwa bersama dengan peristiwa AWS layanan lainnya dalam riwayat Acara. Anda dapat melihat, mencari, dan mengunduh acara terbaru di situs

Anda Akun AWS. Untuk informasi selengkapnya, lihat Melihat Acara dengan Riwayat CloudTrail Acara.

Untuk catatan acara yang sedang berlangsung di Anda Akun AWS, termasuk acara untuk AWS FIS, buat jejak. Jejak memungkinkan CloudTrail untuk mengirimkan file log ke bucket Amazon S3. Secara default, saat Anda membuat jejak di konsol, jejak tersebut berlaku untuk semua AWS Wilayah. Jejak mencatat peristiwa dari semua Wilayah di AWS partisi 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 berikut ini:

- Buat Jejak untuk AWS Akun Anda
- CloudTrail Layanan dan Integrasi yang Didukung
- Mengkonfigurasi Notifikasi Amazon SNS untuk CloudTrail
- Menerima File CloudTrail Log dari Beberapa Wilayah dan Menerima File CloudTrail Log dari Beberapa Akun

Semua tindakan AWS FIS dicatat oleh CloudTrail dan didokumentasikan dalam Referensi API Layanan Injeksi AWS Kesalahan. Untuk tindakan percobaan yang dilakukan pada sumber daya target, lihat dokumentasi referensi API untuk layanan yang memiliki sumber daya. Misalnya, untuk tindakan yang dilakukan pada EC2 instance Amazon, lihat Referensi Amazon EC2 API.

Setiap entri peristiwa atau log berisi informasi tentang siapa yang membuat permintaan tersebut. Informasi identitas membantu Anda menentukan hal berikut:

- Baik permintaan tersebut dibuat dengan kredensial pengguna atau root.
- Apakah permintaan tersebut dibuat dengan kredensial keamanan sementara untuk satu peran atau pengguna gabungan.
- Apakah permintaan itu dibuat oleh AWS layanan lain.

Untuk informasi lain, lihat Elemen userIdentity CloudTrail .

Memahami AWS entri file log FIS

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, jadi file tersebut tidak muncul dalam urutan tertentu.

Berikut ini adalah contoh entri CloudTrail log untuk panggilan ke `StopExperiment` tindakan AWS FIS.


```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AKIAIOSFODNN7EXAMPLE:jdoe",
    "arn": "arn:aws:sts::111122223333:assumed-role/example/jdoe",
    "accountId": "111122223333",
    "accessKeyId": "AKIAI44QH8DHBEXAMPLE",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AKIAIOSFODNN7EXAMPLE",
        "arn": "arn:aws:iam::111122223333:role/example",
        "accountId": "111122223333",
        "userName": "example"
      },
      "webIdFederationData": {},
      "attributes": {
        "creationDate": "2020-12-03T09:40:42Z",
        "mfaAuthenticated": "false"
      }
    }
  },
  "eventTime": "2020-12-03T09:44:20Z",
  "eventSource": "fis.amazonaws.com",
  "eventName": "StopExperiment",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "192.51.100.25",
  "userAgent": "Boto3/1.22.9 Python/3.8.13 Linux/5.4.186-113.361.amzn2int.x86_64
 Botocore/1.25.9",
  "requestParameters": {
    "clientToken": "1234abc5-6def-789g-012h-ijklm34no56p",
    "experimentTemplateId": "ABCDE1fgHIJkLmNop",
    "tags": {}
  },
```

```
  "responseElements": {
    "experiment": {
      "actions": {
        "exampleAction1": {
          "actionId": "aws:ec2:stop-instances",
          "duration": "PT10M",
          "state": {
            "reason": "Initial state",
            "status": "pending"
          },
          "targets": {
            "Instances": "exampleTag1"
          }
        },
        "exampleAction2": {
          "actionId": "aws:ec2:stop-instances",
          "duration": "PT10M",
          "state": {
            "reason": "Initial state",
            "status": "pending"
          },
          "targets": {
            "Instances": "exampleTag2"
          }
        }
      },
      "creationTime": 1605788649.95,
      "endTime": 1606988660.846,
      "experimentTemplateId": "ABCDE1fgHIJkLmNop",
      "id": "ABCDE1fgHIJkLmNop",
      "roleArn": "arn:aws:iam::111122223333:role/AllowFISActions",
      "startTime": 1605788650.109,
      "state": {
        "reason": "Experiment stopped",
        "status": "stopping"
      },
      "stopConditions": [
        {
          "source": "aws:cloudwatch:alarm",
          "value": "arn:aws:cloudwatch:us-east-1:111122223333:alarm:example"
        }
      ],
      "tags": {},
      "targets": {
```

```
        "ExampleTag1": {
          "resourceTags": {
            "Example": "tag1"
          },
          "resourceType": "aws:ec2:instance",
          "selectionMode": "RANDOM(1)"
        },
        "ExampleTag2": {
          "resourceTags": {
            "Example": "tag2"
          },
          "resourceType": "aws:ec2:instance",
          "selectionMode": "RANDOM(1)"
        }
      }
    }
  },
  "requestID": "1abcd23e-f4gh-567j-klm8-9np01q234r56",
  "eventID": "1234a56b-c78d-9e0f-g1h2-34jk56m7n890",
  "readOnly": false,
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "111122223333",
  "eventCategory": "Management"

}
```

Berikut ini adalah contoh entri CloudTrail log untuk tindakan API yang AWS dipanggil FIS sebagai bagian dari eksperimen yang mencakup tindakan `aws:ssm:send-command` AWS FIS. `userIdentity`Elemen mencerminkan permintaan yang dibuat dengan kredensi sementara yang diperoleh dengan mengasumsikan peran. Nama peran yang diasumsikan muncul di`userName`. ID percobaan, EXP21n T17 WMz A6DNugz, muncul di dan `principalId` sebagai bagian dari ARN dari peran yang diasumsikan.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROATZZZ4JPIXUEXAMPLE:EXP21nT17WMzA6dnUgz",
        "arn": "arn:aws:sts::111122223333:assumed-role/AllowActions/
EXP21nT17WMzA6dnUgz",
        "accountId": "111122223333",
```

```
        "accessKeyId": "AKIAI44QH8DHBEXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROATZZZ4JPIXUEXAMPLE",
                "arn": "arn:aws:iam::111122223333:role/AllowActions",
                "accountId": "111122223333",
                "userName": "AllowActions"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-05-30T13:23:19Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "fis.amazonaws.com"
    },
    "eventTime": "2022-05-30T13:23:19Z",
    "eventSource": "ssm.amazonaws.com",
    "eventName": "ListCommands",
    "awsRegion": "us-east-2",
    "sourceIPAddress": "fis.amazonaws.com",
    "userAgent": "fis.amazonaws.com",
    "requestParameters": {
        "commandId": "51dab97f-489b-41a8-a8a9-c9854955dc65"
    },
    "responseElements": null,
    "requestID": "23709ced-c19e-471a-9d95-cf1a06b50ee6",
    "eventID": "145fe5a6-e9d5-45cc-be25-b7923b950c83",
    "readOnly": true,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

Pemecahan masalah AWS FIS

Untuk memecahkan masalah kesalahan, AWS FIS mengembalikan kesalahan terperinci dari log eksperimen `GetExperiment` API dan FIS. Kesalahan dikembalikan sebagai bagian dari status eksperimen ketika status percobaan gagal. Ketika beberapa tindakan gagal, tindakan gagal pertama dikembalikan sebagai kesalahan percobaan. Anda dapat meninjau log eksperimen FIS Anda untuk kesalahan lainnya. Untuk mempelajari cara mencatat dan memantau AWS FIS eksperimen, lihatMemantau AWS eksperimen FIS.

Tergantung pada jenis kegagalan, Anda mungkin menerima salah satu kesalahan berikut:

- Alasan: Deskripsi terperinci tentang kegagalan spesifik. Nilai alasan tidak boleh digunakan untuk otomatisasi, karena dapat berubah.
- Kode: Jenis kegagalan. Nilai kode tidak boleh digunakan untuk otomatisasi, karena dapat berubah, kecuali ditentukan lain dalam tabel di bawah ini.
- Lokasi: Konteks untuk bagian template percobaan yang gagal, seperti tindakan atau target.
- ID Akun: AWS Akun tempat kegagalan terjadi.

Kode eror

Kode Kesalahan	Deskripsi Kode
`ConfigurationFailure`	Tindakan, target, eksperimen, atau log tidak dikonfigurasi dengan benar. Periksa kesalahan `location` dan pastikan bahwa parameter dan konfigurasi sudah benar.
`DependentServiceFailure`	Ada kegagalan dari AWS layanan lain. Coba jalankan eksperimen lagi.
`InternalFailure`	Terjadi kesalahan internal saat menjalank an percobaan. Anda dapat mengotomatisasi berdasarkan kode kesalahan ini.

Kode Kesalahan	Deskripsi Kode
`InvalidTarget`	Target tidak dapat diselesaikan selama resolusi target atau pada awal tindakan. Ini mungkin karena salah satu alasan berikut: • Target tidak ada, misalnya jika dihapus atau jika ARN salah. • Ada tag untuk target Anda yang tidak menyelesaikan sumber daya apa pun. • Ada tindakan yang tidak terkait dengan target. Untuk memecahkan masalah, tinjau log Anda untuk mengidentifikasi target mana yang tidak terselesaikan. Periksa apakah semua tindakan ditautkan ke target, dan apakah ID atau tag sumber daya Anda ada dan belum salah eja.

Kode Kesalahan	Deskripsi Kode
`AuthorizationFailure`	Ada dua penyebab utama kegagalan percobaan karena kesalahan izin: • Peran IAM yang Anda targetkan tidak memiliki izin yang tepat untuk menyelesa ikan target atau mengambil tindakan pada sumber daya Anda. Untuk memperbaiki kesalahan ini, tinjau izin yang diperlukan untuk tindakan Anda di Referensi Tindakan FIS dan tambahkan ke peran IAM eksperime n Anda. • Pembuatan peran AWS terkait layanan (SLR) untuk FIS ditolak oleh kebijakan kontrol layanan (SCP) di organisasi Anda. FIS menggunakan SLR untuk mengelola pemantauan dan pemilihan sumber daya untuk eksperimen. Untuk informasi selengkapnya, lihat Izin peran terkait layanan untuk FIS AWS.
`QuotaExceededFailure`	Kuota untuk jenis sumber daya telah terlampau i. Untuk menentukan apakah kuota dapat ditingkatkan, lihatKuota dan batasan untuk Layanan Injeksi AWS Kesalahan.

Keamanan dalam Layanan Injeksi AWS Kesalahan

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 menjelaskan hal ini sebagai keamanan dari cloud dan keamanan dalam cloud:

- Keamanan cloud — AWS bertanggung jawab untuk melindungi infrastruktur yang menjalankan AWS layanan di AWS Cloud. AWS juga memberi Anda layanan yang dapat Anda gunakan dengan aman. Auditor pihak ketiga secara teratur menguji dan memverifikasi efektivitas keamanan kami sebagai bagian dari Program AWS Kepatuhan Program AWS Kepatuhan . Untuk mempelajari tentang program kepatuhan yang berlaku untuk Layanan Injeksi AWS Kesalahan, lihat AWS Layanan dalam Lingkup oleh AWS Layanan Program Kepatuhan .
- Keamanan di cloud — Tanggung jawab Anda ditentukan oleh AWS layanan yang Anda gunakan. Anda juga bertanggung jawab atas faktor lain, yang mencakup sensitivitas data Anda, persyaratan perusahaan Anda, serta undang-undang dan peraturan yang berlaku.

Dokumentasi ini membantu Anda memahami cara menerapkan model tanggung jawab bersama saat menggunakan AWS FIS. Topik berikut menunjukkan cara mengkonfigurasi AWS FIS untuk memenuhi tujuan keamanan dan kepatuhan Anda. Anda juga belajar cara menggunakan AWS layanan lain yang membantu Anda memantau dan mengamankan sumber daya AWS FIS Anda.

Daftar Isi

- Perlindungan data di Layanan Injeksi AWS Kesalahan
- Manajemen identitas dan akses untuk Layanan Injeksi AWS Kesalahan
- Keamanan infrastruktur di Layanan Injeksi AWS Kesalahan
- Akses AWS FIS menggunakan antarmuka VPC endpoint ()AWS PrivateLink

Perlindungan data di Layanan Injeksi AWS Kesalahan

Model tanggung jawab AWS bersama model berlaku untuk perlindungan data di Layanan Injeksi AWS Kesalahan. Seperti yang 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 menyarankan 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 AWS sumber daya. Kami mensyaratkan TLS 1.2 dan menganjurkan TLS 1.3.
- Siapkan 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 dengan semua kontrol keamanan default 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 tervalidasi FIPS 140-3 saat 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 AWS FIS atau lainnya Layanan AWS menggunakan konsol, API AWS CLI, atau 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.

Enkripsi diam

AWS FIS selalu mengenkripsi data Anda saat istirahat. Data dalam AWS FIS dienkripsi saat istirahat menggunakan enkripsi sisi server transparan. Hal ini membantu mengurangi beban operasional dan kompleksitas yang terlibat dalam melindungi data sensitif. Dengan enkripsi saat istirahat, Anda dapat membangun aplikasi yang sensitif terhadap keamanan yang memenuhi persyaratan kepatuhan enkripsi dan peraturan.

Enkripsi bergerak

AWS FIS mengenkripsi data dalam perjalanan antara layanan dan layanan terintegrasi lainnya. AWS Semua data yang melewati antara AWS FIS dan layanan terintegrasi dienkripsi menggunakan Transport Layer Security (TLS). Untuk informasi selengkapnya tentang AWS layanan terintegrasi lainnya, lihatDidukung Layanan AWS.

Manajemen identitas dan akses untuk Layanan Injeksi AWS Kesalahan

AWS Identity and Access Management (IAM) adalah Layanan AWS yang membantu administrator mengontrol akses ke AWS sumber daya dengan aman. Administrator IAM mengontrol siapa yang dapat diautentikasi (masuk) dan diberi wewenang (memiliki izin) untuk menggunakan sumber daya FIS. AWS IAM adalah Layanan AWS yang dapat Anda gunakan tanpa biaya tambahan.

Daftar Isi

- Audiens
- Mengautentikasi dengan identitas
- Mengelola akses menggunakan kebijakan
- Bagaimana Layanan Injeksi AWS Kesalahan bekerja dengan IAM
- AWS Contoh kebijakan Layanan Injeksi Kesalahan
- Gunakan peran terkait layanan untuk Layanan Injeksi AWS Kesalahan
- AWS kebijakan terkelola untuk Layanan Injeksi AWS Kesalahan

Audiens

Bagaimana Anda menggunakan AWS Identity and Access Management (IAM) berbeda, tergantung pada pekerjaan yang Anda lakukan di AWS FIS.

Pengguna layanan — Jika Anda menggunakan layanan AWS FIS untuk melakukan pekerjaan Anda, maka administrator Anda memberi Anda kredensi dan izin yang Anda butuhkan. Saat Anda menggunakan lebih banyak AWS fitur FIS untuk melakukan pekerjaan Anda, Anda mungkin memerlukan izin tambahan. Memahami cara akses dikelola dapat membantu Anda meminta izin yang tepat dari administrator Anda.

Administrator layanan — Jika Anda bertanggung jawab atas sumber daya AWS FIS di perusahaan Anda, Anda mungkin memiliki akses penuh ke AWS FIS. Tugas Anda adalah menentukan fitur dan sumber daya AWS FIS mana yang harus diakses pengguna layanan Anda. Kemudian, Anda harus mengirimkan permintaan kepada administrator IAM untuk mengubah izin pengguna layanan Anda. Tinjau informasi di halaman ini untuk memahami konsep dasar IAM.

Administrator IAM - Jika Anda seorang administrator IAM, Anda mungkin ingin mempelajari detail tentang bagaimana Anda dapat menulis kebijakan untuk mengelola akses ke AWS FIS.

Mengautentikasi dengan identitas

Otentikasi adalah cara Anda masuk AWS menggunakan kredensi identitas Anda. Anda harus diautentikasi (masuk ke AWS) sebagai Pengguna root akun AWS, sebagai pengguna IAM, atau dengan mengasumsikan peran IAM.

Anda dapat masuk AWS sebagai identitas federasi dengan menggunakan kredensil yang disediakan melalui sumber identitas. AWS IAM Identity Center Pengguna (IAM Identity Center), autentikasi masuk tunggal perusahaan Anda, dan kredensi Google atau Facebook Anda adalah contoh identitas federasi. Saat Anda masuk sebagai identitas terfederasi, administrator Anda sebelumnya menyiapkan federasi identitas menggunakan peran IAM. Ketika Anda mengakses AWS dengan menggunakan federasi, Anda secara tidak langsung mengambil peran.

Bergantung pada jenis pengguna Anda, Anda dapat masuk ke AWS Management Console atau portal AWS akses. Untuk informasi selengkapnya tentang masuk AWS, lihat Cara masuk ke Panduan AWS Sign-In Pengguna Anda Akun AWS.

Jika Anda mengakses AWS secara terprogram, AWS sediakan kit pengembangan perangkat lunak (SDK) dan antarmuka baris perintah (CLI) untuk menandatangani permintaan Anda secara kriptografis dengan menggunakan kredensil Anda. Jika Anda tidak menggunakan AWS alat, Anda harus menandatangani permintaan sendiri. Guna mengetahui informasi selengkapnya tentang penggunaan metode yang disarankan untuk menandatangani permintaan sendiri, lihat AWS Signature Version 4 untuk permintaan API dalam Panduan Pengguna IAM.

Apa pun metode autentikasi yang digunakan, Anda mungkin diminta untuk menyediakan informasi keamanan tambahan. Misalnya, AWS merekomendasikan agar Anda menggunakan otentikasi multi-faktor (MFA) untuk meningkatkan keamanan akun Anda. Untuk mempelajari selengkapnya, lihat Autentikasi multi-faktor dalam Panduan Pengguna AWS IAM Identity Center dan Autentikasi multi-faktor AWS di IAM dalam Panduan Pengguna IAM.

Akun AWS pengguna root

Saat Anda membuat Akun AWS, Anda mulai dengan satu identitas masuk yang memiliki akses lengkap ke semua Layanan AWS dan sumber daya di akun. Identitas ini disebut pengguna Akun AWS root dan diakses dengan masuk dengan alamat email dan kata sandi yang Anda gunakan untuk membuat akun. Kami sangat menyarankan agar Anda tidak menggunakan pengguna root untuk tugas sehari-hari. Lindungi kredensial pengguna root Anda dan gunakan kredensial tersebut untuk melakukan tugas yang hanya dapat dilakukan pengguna root. Untuk daftar lengkap tugas yang mengharuskan Anda masuk sebagai pengguna root, lihat Tugas yang memerlukan kredensial pengguna root dalam Panduan Pengguna IAM.

Identitas gabungan

Sebagai praktik terbaik, mewajibkan pengguna manusia, termasuk pengguna yang memerlukan akses administrator, untuk menggunakan federasi dengan penyedia identitas untuk mengakses Layanan AWS dengan menggunakan kredensi sementara.

Identitas federasi adalah pengguna dari direktori pengguna perusahaan Anda, penyedia identitas web, direktori Pusat Identitas AWS Directory Service, atau pengguna mana pun yang mengakses Layanan AWS dengan menggunakan kredensyal yang disediakan melalui sumber identitas. Ketika identitas federasi mengakses Akun AWS, mereka mengambil peran, dan peran memberikan kredensyal sementara.

Untuk manajemen akses terpusat, kami sarankan Anda menggunakan AWS IAM Identity Center. Anda dapat membuat pengguna dan grup di Pusat Identitas IAM, atau Anda dapat menghubungkan dan menyinkronkan ke sekumpulan pengguna dan grup di sumber identitas Anda sendiri untuk digunakan di semua aplikasi Akun AWS dan aplikasi Anda. Untuk informasi tentang Pusat Identitas IAM, lihat Apakah itu Pusat Identitas IAM? dalam Panduan Pengguna AWS IAM Identity Center .

Pengguna dan grup IAM

Pengguna IAM adalah identitas dalam diri Anda Akun AWS yang memiliki izin khusus untuk satu orang atau aplikasi. Jika memungkinkan, kami merekomendasikan untuk mengandalkan kredensial

sementara, bukan membuat pengguna IAM yang memiliki kredensial jangka panjang seperti kata sandi dan kunci akses. Namun, jika Anda memiliki kasus penggunaan tertentu yang memerlukan kredensial jangka panjang dengan pengguna IAM, kami merekomendasikan Anda merotasi kunci akses. Untuk informasi selengkapnya, lihat Merotasi kunci akses secara teratur untuk kasus penggunaan yang memerlukan kredensial jangka panjang dalam Panduan Pengguna IAM.

Grup IAM adalah identitas yang menentukan sekumpulan pengguna IAM. Anda tidak dapat masuk sebagai grup. Anda dapat menggunakan grup untuk menentukan izin bagi beberapa pengguna sekaligus. Grup mempermudah manajemen izin untuk sejumlah besar pengguna sekaligus. Misalnya, Anda dapat meminta kelompok untuk menyebutkan IAMAdmins dan memberikan izin kepada grup tersebut untuk mengelola sumber daya IAM.

Pengguna berbeda dari peran. Pengguna secara unik terkait dengan satu orang atau aplikasi, tetapi peran dimaksudkan untuk dapat digunakan oleh siapa pun yang membutuhkannya. Pengguna memiliki kredensial jangka panjang permanen, tetapi peran memberikan kredensial sementara. Untuk mempelajari selengkapnya, lihat Kasus penggunaan untuk pengguna IAM dalam Panduan Pengguna IAM.

Peran IAM

Peran IAM adalah identitas dalam diri Anda Akun AWS yang memiliki izin khusus. Peran ini mirip dengan pengguna IAM, tetapi tidak terkait dengan orang tertentu. Untuk mengambil peran IAM sementara AWS Management Console, Anda dapat beralih dari pengguna ke peran IAM (konsol). Anda dapat mengambil peran dengan memanggil operasi AWS CLI atau AWS API atau dengan menggunakan URL kustom. Untuk informasi selengkapnya tentang cara menggunakan peran, lihat Metode untuk mengambil peran dalam Panduan Pengguna IAM.

Peran IAM dengan kredensial sementara berguna dalam situasi berikut:

- Akses pengguna terfederasi – Untuk menetapkan izin ke identitas terfederasi, Anda membuat peran dan menentukan izin untuk peran tersebut. Ketika identitas terfederasi mengautentikasi, identitas tersebut terhubung dengan peran dan diberi izin yang ditentukan oleh peran. Untuk informasi tentang peran untuk federasi, lihat Buat peran untuk penyedia identitas pihak ketiga dalam Panduan Pengguna IAM. Jika menggunakan Pusat Identitas IAM, Anda harus mengonfigurasi set izin. Untuk mengontrol apa yang dapat diakses identitas Anda setelah identitas tersebut diautentikasi, Pusat Identitas IAM akan mengorelasikan set izin ke peran dalam IAM. Untuk informasi tentang set izin, lihat Set izin dalam Panduan Pengguna AWS IAM Identity Center .
- Izin pengguna IAM sementara – Pengguna atau peran IAM dapat mengambil peran IAM guna mendapatkan berbagai izin secara sementara untuk tugas tertentu.

- Akses lintas akun – Anda dapat menggunakan peran IAM untuk mengizinkan seseorang (prinsipal tepercaya) di akun lain untuk mengakses sumber daya di akun Anda. Peran adalah cara utama untuk memberikan akses lintas akun. Namun, dengan beberapa Layanan AWS, Anda dapat melampirkan kebijakan langsung ke sumber daya (alih-alih menggunakan peran sebagai proxy). Untuk mempelajari perbedaan antara peran dan kebijakan berbasis sumber daya untuk akses lintas akun, lihat Akses sumber daya lintas akun di IAM dalam Panduan Pengguna IAM.
- Akses lintas layanan — Beberapa Layanan AWS menggunakan fitur lain Layanan AWS. Misalnya, saat Anda melakukan panggilan dalam suatu layanan, biasanya layanan tersebut menjalankan aplikasi di Amazon EC2 atau menyimpan objek di Amazon S3. Sebuah layanan mungkin melakukannya menggunakan izin prinsipal yang memanggil, menggunakan peran layanan, atau peran terkait layanan.
  - Sesi akses teruskan (FAS) — Saat Anda menggunakan pengguna atau peran IAM untuk melakukan tindakan AWS, Anda dianggap sebagai prinsipal. Ketika Anda menggunakan beberapa layanan, Anda mungkin melakukan sebuah tindakan yang kemudian menginisiasi tindakan lain di layanan yang berbeda. FAS menggunakan izin dari pemanggilan utama Layanan AWS, dikombinasikan dengan permintaan Layanan AWS untuk membuat permintaan ke layanan hilir. Permintaan FAS hanya dibuat ketika layanan menerima permintaan yang memerlukan interaksi dengan orang lain Layanan AWS atau sumber daya untuk menyelesaikannya. Dalam hal ini, Anda harus memiliki izin untuk melakukan kedua tindakan tersebut. Untuk detail kebijakan ketika mengajukan permintaan FAS, lihat Sesi akses maju.
  - Peran layanan – Peran layanan adalah peran IAM yang dijalankan oleh layanan untuk melakukan tindakan atas nama Anda. Administrator IAM dapat membuat, mengubah, dan menghapus peran layanan dari dalam IAM. Untuk informasi selengkapnya, lihat Buat sebuah peran untuk mendelegasikan izin ke Layanan AWS dalam Panduan pengguna IAM.
  - Peran terkait layanan — Peran terkait layanan adalah jenis peran layanan yang ditautkan ke. Layanan AWS Layanan tersebut dapat menjalankan peran untuk melakukan tindakan atas nama Anda. Peran terkait layanan muncul di Anda Akun AWS dan dimiliki oleh layanan. Administrator IAM dapat melihat, tetapi tidak dapat mengedit izin untuk peran terkait layanan.
- Aplikasi yang berjalan di Amazon EC2 — Anda dapat menggunakan peran IAM untuk mengelola kredensyal sementara untuk aplikasi yang berjalan pada EC2 instance dan membuat AWS CLI atau AWS permintaan API. Ini lebih baik untuk menyimpan kunci akses dalam EC2 instance. Untuk menetapkan AWS peran ke EC2 instance dan membuatnya tersedia untuk semua aplikasinya, Anda membuat profil instans yang dilampirkan ke instance. Profil instance berisi peran dan memungkinkan program yang berjalan pada EC2 instance untuk mendapatkan kredensyal

sementara. Untuk informasi selengkapnya, lihat Menggunakan peran IAM untuk memberikan izin ke aplikasi yang berjalan di EC2 instans Amazon di Panduan Pengguna IAM.

Mengelola akses menggunakan kebijakan

Anda mengontrol akses AWS dengan membuat kebijakan dan melampirkannya ke AWS identitas atau sumber daya. Kebijakan adalah objek AWS yang, ketika dikaitkan dengan identitas atau sumber daya, menentukan izinnya. AWS mengevaluasi kebijakan ini ketika prinsipal (pengguna, pengguna root, atau sesi peran) membuat permintaan. Izin dalam kebijakan menentukan apakah permintaan diizinkan atau ditolak. Sebagian besar kebijakan disimpan AWS sebagai dokumen JSON. Untuk informasi selengkapnya tentang struktur dan isi dokumen kebijakan JSON, lihat Gambaran umum kebijakan JSON dalam Panduan Pengguna IAM.

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Artinya, prinsipal manakah yang dapat melakukan tindakan pada sumber daya apa, dan dengan kondisi apa.

Secara default, pengguna dan peran tidak memiliki izin. Untuk memberikan izin kepada pengguna untuk melakukan tindakan di sumber daya yang mereka perlukan, administrator IAM dapat membuat kebijakan IAM. Administrator kemudian dapat menambahkan kebijakan IAM ke peran, dan pengguna dapat mengambil peran.

Kebijakan IAM mendefinisikan izin untuk suatu tindakan terlepas dari metode yang Anda gunakan untuk melakukan operasinya. Misalnya, anggaplah Anda memiliki kebijakan yang mengizinkan tindakan `iam:GetRole`. Pengguna dengan kebijakan tersebut bisa mendapatkan informasi peran dari AWS Management Console, API AWS CLI, atau AWS API.

Kebijakan berbasis identitas

Kebijakan berbasis identitas adalah dokumen kebijakan izin JSON yang dapat Anda lampirkan ke sebuah identitas, seperti pengguna IAM, grup pengguna IAM, atau peran IAM. Kebijakan ini mengontrol jenis tindakan yang dapat dilakukan oleh pengguna dan peran, di sumber daya mana, dan berdasarkan kondisi seperti apa. Untuk mempelajari cara membuat kebijakan berbasis identitas, lihat Tentukan izin IAM kustom dengan kebijakan terkelola pelanggan dalam Panduan Pengguna IAM.

Kebijakan berbasis identitas dapat dikategorikan lebih lanjut sebagai kebijakan inline atau kebijakan yang dikelola. Kebijakan inline disematkan langsung ke satu pengguna, grup, atau peran. Kebijakan

terkelola adalah kebijakan mandiri yang dapat Anda lampirkan ke beberapa pengguna, grup, dan peran dalam. Akun AWS Kebijakan AWS terkelola mencakup kebijakan terkelola dan kebijakan yang dikelola pelanggan. Untuk mempelajari cara memilih antara kebijakan yang dikelola atau kebijakan inline, lihat Pilih antara kebijakan yang dikelola dan kebijakan inline dalam Panduan Pengguna IAM.

Kebijakan berbasis sumber daya

Kebijakan berbasis sumber daya adalah dokumen kebijakan JSON yang Anda lampirkan ke sumber daya. Contoh kebijakan berbasis sumber daya adalah kebijakan kepercayaan peran IAM dan kebijakan bucket Amazon S3. Dalam layanan yang mendukung kebijakan berbasis sumber daya, administrator layanan dapat menggunakannya untuk mengontrol akses ke sumber daya tertentu. Untuk sumber daya tempat kebijakan dilampirkan, kebijakan menentukan tindakan apa yang dapat dilakukan oleh prinsipal tertentu pada sumber daya tersebut dan dalam kondisi apa. Anda harus menentukan prinsipal dalam kebijakan berbasis sumber daya. Prinsipal dapat mencakup akun, pengguna, peran, pengguna federasi, atau. Layanan AWS

Kebijakan berbasis sumber daya merupakan kebijakan inline yang terletak di layanan tersebut. Anda tidak dapat menggunakan kebijakan AWS terkelola dari IAM dalam kebijakan berbasis sumber daya.

Daftar kontrol akses (ACLs)

Access control lists (ACLs) mengontrol prinsipal mana (anggota akun, pengguna, atau peran) yang memiliki izin untuk mengakses sumber daya. ACLs mirip dengan kebijakan berbasis sumber daya, meskipun mereka tidak menggunakan format dokumen kebijakan JSON.

Amazon S3, AWS WAF, dan Amazon VPC adalah contoh layanan yang mendukung. ACLs Untuk mempelajari selengkapnya ACLs, lihat Ringkasan daftar kontrol akses (ACL) di Panduan Pengembang Layanan Penyimpanan Sederhana Amazon.

Jenis-jenis kebijakan lain

AWS mendukung jenis kebijakan tambahan yang kurang umum. Jenis-jenis kebijakan ini dapat mengatur izin maksimum yang diberikan kepada Anda oleh jenis kebijakan yang lebih umum.

- Batasan izin – Batasan izin adalah fitur lanjutan tempat Anda mengatur izin maksimum yang dapat diberikan oleh kebijakan berbasis identitas ke entitas IAM (pengguna IAM atau peran IAM). Anda dapat menetapkan batasan izin untuk suatu entitas. Izin yang dihasilkan adalah perpotongan antara kebijakan berbasis identitas milik entitas dan batasan izinnya. Kebijakan berbasis sumber daya yang menentukan pengguna atau peran dalam bidang `Principal` tidak dibatasi oleh

batasan izin. Penolakan eksplisit dalam salah satu kebijakan ini akan menggantikan pemberian izin. Untuk informasi selengkapnya tentang batasan izin, lihat Batasan izin untuk entitas IAM dalam Panduan Pengguna IAM.

- Kebijakan kontrol layanan (SCPs) — SCPs adalah kebijakan JSON yang menentukan izin maksimum untuk organisasi atau unit organisasi (OU) di. AWS Organizations AWS Organizations adalah layanan untuk mengelompokkan dan mengelola secara terpusat beberapa Akun AWS yang dimiliki bisnis Anda. Jika Anda mengaktifkan semua fitur dalam organisasi, Anda dapat menerapkan kebijakan kontrol layanan (SCPs) ke salah satu atau semua akun Anda. SCP membatasi izin untuk entitas di akun anggota, termasuk masing-masing. Pengguna root akun AWS Untuk informasi selengkapnya tentang Organizations dan SCPs, lihat Kebijakan kontrol layanan di Panduan AWS Organizations Pengguna.
- Kebijakan kontrol sumber daya (RCPs) — RCPs adalah kebijakan JSON yang dapat Anda gunakan untuk menetapkan izin maksimum yang tersedia untuk sumber daya di akun Anda tanpa memperbarui kebijakan IAM yang dilampirkan ke setiap sumber daya yang Anda miliki. RCP membatasi izin untuk sumber daya di akun anggota dan dapat memengaruhi izin efektif untuk identitas, termasuk Pengguna root akun AWS, terlepas dari apakah itu milik organisasi Anda. Untuk informasi selengkapnya tentang Organizations dan RCPs, termasuk daftar dukungan Layanan AWS tersebut RCPs, lihat Kebijakan kontrol sumber daya (RCPs) di Panduan AWS Organizations Pengguna.
- Kebijakan sesi – Kebijakan sesi adalah kebijakan lanjutan yang Anda berikan sebagai parameter ketika Anda membuat sesi sementara secara programatis untuk peran atau pengguna terfederasi. Izin sesi yang dihasilkan adalah perpotongan antara kebijakan berbasis identitas pengguna atau peran dan kebijakan sesi. Izin juga bisa datang dari kebijakan berbasis sumber daya. Penolakan eksplisit dalam salah satu kebijakan ini akan menggantikan pemberian izin. Untuk informasi selengkapnya, lihat Kebijakan sesi dalam Panduan Pengguna IAM.

Berbagai jenis kebijakan

Ketika beberapa jenis kebijakan berlaku pada suatu permintaan, izin yang dihasilkan lebih rumit untuk dipahami. Untuk mempelajari cara AWS menentukan apakah akan mengizinkan permintaan saat beberapa jenis kebijakan terlibat, lihat Logika evaluasi kebijakan di Panduan Pengguna IAM.

Bagaimana Layanan Injeksi AWS Kesalahan bekerja dengan IAM

Sebelum Anda menggunakan IAM untuk mengelola akses ke AWS FIS, pelajari fitur IAM apa yang tersedia untuk digunakan dengan FIS. AWS

Fitur IAM yang dapat Anda gunakan dengan Layanan Injeksi AWS Kesalahan

Fitur IAM	AWS Dukungan FIS
Kebijakan berbasis identitas	Ya
Kebijakan berbasis sumber daya	Tidak
Tindakan kebijakan	Ya
Sumber daya kebijakan	Ya
kunci-kunci persyaratan kebijakan (spesifik layanan)	Ya
ACLs	Tidak
ABAC (tanda dalam kebijakan)	Ya
Kredensial sementara	Ya
Izin principal	Ya
Peran layanan	Ya
Peran terkait layanan	Ya

Untuk mendapatkan pandangan tingkat tinggi tentang bagaimana AWS FIS dan AWS layanan lainnya bekerja dengan sebagian besar fitur IAM, lihat AWS layanan yang bekerja dengan IAM di Panduan Pengguna IAM.

Kebijakan berbasis identitas untuk FIS AWS

Mendukung kebijakan berbasis identitas: Ya

Kebijakan berbasis identitas adalah dokumen kebijakan izin JSON yang dapat Anda lampirkan ke sebuah identitas, seperti pengguna IAM, grup pengguna IAM, atau peran IAM. Kebijakan ini mengontrol jenis tindakan yang dapat dilakukan oleh pengguna dan peran, di sumber daya mana, dan berdasarkan kondisi seperti apa. Untuk mempelajari cara membuat kebijakan berbasis identitas, lihat Tentukan izin IAM kustom dengan kebijakan terkelola pelanggan dalam Panduan Pengguna IAM.

Dengan kebijakan berbasis identitas IAM, Anda dapat menentukan secara spesifik apakah tindakan dan sumber daya diizinkan atau ditolak, serta kondisi yang menjadi dasar dikabulkan atau ditolaknya tindakan tersebut. Anda tidak dapat menentukan secara spesifik prinsipal dalam sebuah kebijakan berbasis identitas karena prinsipal berlaku bagi pengguna atau peran yang melekat kepadanya. Untuk mempelajari semua elemen yang dapat Anda gunakan dalam kebijakan JSON, lihat Referensi elemen kebijakan JSON IAM dalam Panduan Pengguna IAM.

Contoh kebijakan berbasis identitas untuk FIS AWS

Untuk melihat contoh kebijakan berbasis identitas AWS FIS, lihat. AWS Contoh kebijakan Layanan Injeksi Kesalahan

Kebijakan berbasis sumber daya dalam FIS AWS

Mendukung kebijakan berbasis sumber daya: Tidak

Kebijakan berbasis sumber daya adalah dokumen kebijakan JSON yang Anda lampirkan ke sumber daya. Contoh kebijakan berbasis sumber daya adalah kebijakan kepercayaan peran IAM dan kebijakan bucket Amazon S3. Dalam layanan yang mendukung kebijakan berbasis sumber daya, administrator layanan dapat menggunakannya untuk mengontrol akses ke sumber daya tertentu. Untuk sumber daya tempat kebijakan dilampirkan, kebijakan menentukan tindakan apa yang dapat dilakukan oleh prinsipal tertentu pada sumber daya tersebut dan dalam kondisi apa. Anda harus menentukan prinsipal dalam kebijakan berbasis sumber daya. Prinsipal dapat mencakup akun, pengguna, peran, pengguna federasi, atau. Layanan AWS

Untuk mengaktifkan akses lintas akun, Anda dapat menentukan secara spesifik seluruh akun atau entitas IAM di akun lain sebagai prinsipal dalam kebijakan berbasis sumber daya. Menambahkan prinsipal akun silang ke kebijakan berbasis sumber daya hanya setengah dari membangun hubungan kepercayaan. Ketika prinsipal dan sumber daya berbeda Akun AWS, administrator IAM di akun tepercaya juga harus memberikan izin entitas utama (pengguna atau peran) untuk mengakses sumber daya. Mereka memberikan izin dengan melampirkan kebijakan berbasis identitas kepada entitas. Namun, jika kebijakan berbasis sumber daya memberikan akses ke principal dalam akun yang sama, tidak diperlukan kebijakan berbasis identitas tambahan. Untuk informasi selengkapnya, lihat Akses sumber daya lintas akun di IAM dalam Panduan Pengguna IAM.

Tindakan kebijakan untuk AWS FIS

Mendukung tindakan kebijakan: Ya

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Artinya, prinsipal manakah yang dapat melakukan tindakan pada sumber daya apa, dan dengan kondisi apa.

Elemen `Action` dari kebijakan JSON menjelaskan tindakan yang dapat Anda gunakan untuk mengizinkan atau menolak akses dalam sebuah kebijakan. Tindakan kebijakan biasanya memiliki nama yang sama dengan operasi AWS API terkait. Ada beberapa pengecualian, misalnya tindakan hanya izin yang tidak memiliki operasi API yang cocok. Ada juga beberapa operasi yang memerlukan beberapa tindakan dalam suatu kebijakan. Tindakan tambahan ini disebut tindakan dependen.

Sertakan tindakan dalam kebijakan untuk memberikan izin untuk melakukan operasi terkait.

Untuk melihat daftar tindakan AWS FIS, lihat Tindakan yang ditentukan oleh Layanan Injeksi AWS Kesalahan di Referensi Otorisasi Layanan.

Tindakan kebijakan di AWS FIS menggunakan awalan berikut sebelum tindakan:

```
fis
```

Untuk menetapkan secara spesifik beberapa tindakan dalam satu pernyataan, pisahkan tindakan tersebut dengan koma.

```
"Action": [
    "fis:action1",
    "fis:action2"
]
```

Anda juga dapat menentukan beberapa tindakan menggunakan wildcard (*). Sebagai contoh, untuk menentukan semua tindakan yang dimulai dengan kata `List`, sertakan tindakan berikut:

```
"Action": "fis:List*"
```

Sumber daya kebijakan untuk AWS FIS

Mendukung sumber daya kebijakan: Ya

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Artinya, prinsipal manakah yang dapat melakukan tindakan pada sumber daya apa, dan dengan kondisi apa.

Elemen kebijakan JSON `Resource` menentukan objek yang menjadi target penerapan tindakan. Pernyataan harus menyertakan elemen `Resource` atau `NotResource`. Praktik terbaiknya, tentukan sumber daya menggunakan Amazon Resource Name (ARN). Anda dapat melakukan ini untuk tindakan yang mendukung jenis sumber daya tertentu, yang dikenal sebagai izin tingkat sumber daya.

Untuk tindakan yang tidak mendukung izin di tingkat sumber daya, misalnya operasi pencantuman, gunakan wildcard (*) untuk menunjukkan bahwa pernyataan tersebut berlaku untuk semua sumber daya.

```
"Resource": "*"
```

Beberapa tindakan API AWS FIS mendukung banyak sumber daya. Untuk menentukan beberapa sumber daya dalam satu pernyataan, pisahkan ARNs dengan koma.

```
"Resource": [
    "resource1",
    "resource2"
]
```

Untuk melihat daftar jenis sumber daya AWS FIS dan jenisnya ARNs, lihat Jenis sumber daya yang ditentukan oleh Layanan Injeksi AWS Kesalahan di Referensi Otorisasi Layanan. Untuk mempelajari tindakan mana yang dapat Anda tentukan ARN dari setiap sumber daya, lihat Tindakan yang ditentukan oleh Layanan Injeksi AWS Kesalahan.

Kunci kondisi kebijakan untuk AWS FIS

Mendukung kunci kondisi kebijakan khusus layanan: Yes

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Artinya, prinsipal manakah yang dapat melakukan tindakan pada sumber daya apa, dan dengan kondisi apa.

Elemen `Condition` (atau blok `Condition`) akan memungkinkan Anda menentukan kondisi yang menjadi dasar suatu pernyataan berlaku. Elemen `Condition` bersifat opsional. Anda dapat membuat ekspresi bersyarat yang menggunakan operator kondisi, misalnya sama dengan atau kurang dari, untuk mencocokkan kondisi dalam kebijakan dengan nilai-nilai yang diminta.

Jika Anda menentukan beberapa elemen `Condition` dalam sebuah pernyataan, atau beberapa kunci dalam elemen `Condition` tunggal, maka AWS akan mengevaluasinya menggunakan operasi

AND logis. Jika Anda menentukan beberapa nilai untuk satu kunci kondisi, AWS mengevaluasi kondisi menggunakan OR operasi logis. Semua kondisi harus dipenuhi sebelum izin pernyataan diberikan.

Anda juga dapat menggunakan variabel placeholder saat menentukan kondisi. Sebagai contoh, Anda dapat memberikan izin kepada pengguna IAM untuk mengakses sumber daya hanya jika izin tersebut mempunyai tanda yang sesuai dengan nama pengguna IAM mereka. Untuk informasi selengkapnya, lihat Elemen kebijakan IAM: variabel dan tanda dalam Panduan Pengguna IAM.

AWS mendukung kunci kondisi global dan kunci kondisi khusus layanan. Untuk melihat semua kunci kondisi AWS global, lihat kunci konteks kondisi AWS global di Panduan Pengguna IAM.

Untuk melihat daftar kunci kondisi AWS FIS, lihat Kunci kondisi untuk Layanan Injeksi AWS Kesalahan di Referensi Otorisasi Layanan. Untuk mempelajari tindakan dan sumber daya yang dapat Anda gunakan kunci kondisi, lihat Tindakan yang ditentukan oleh Layanan Injeksi AWS Kesalahan.

Untuk melihat contoh kebijakan berbasis identitas AWS FIS, lihat. AWS Contoh kebijakan Layanan Injeksi Kesalahan

ACLs di AWS FIS

Mendukung ACLs: Tidak

Access control lists (ACLs) mengontrol prinsipal mana (anggota akun, pengguna, atau peran) yang memiliki izin untuk mengakses sumber daya. ACLs mirip dengan kebijakan berbasis sumber daya, meskipun mereka tidak menggunakan format dokumen kebijakan JSON.

ABAC dengan AWS FIS

Mendukung ABAC (tanda dalam kebijakan): Ya

Kontrol akses berbasis atribut (ABAC) adalah strategi otorisasi yang menentukan izin berdasarkan atribut. Dalam AWS, atribut ini disebut tag. Anda dapat melampirkan tag ke entitas IAM (pengguna atau peran) dan ke banyak AWS sumber daya. Penandaan ke entitas dan sumber daya adalah langkah pertama dari ABAC. Kemudian rancanglah kebijakan ABAC untuk mengizinkan operasi ketika tanda milik prinsipal cocok dengan tanda yang ada di sumber daya yang ingin diakses.

ABAC sangat berguna di lingkungan yang berkembang dengan cepat dan berguna di situasi saat manajemen kebijakan menjadi rumit.

Untuk mengendalikan akses berdasarkan tanda, berikan informasi tentang tanda di elemen kondisi dari kebijakan menggunakan kunci kondisi `aws:ResourceTag/`*`key-name`*, `aws:RequestTag/`*`key-name`*, atau `aws:TagKeys`.

Jika sebuah layanan mendukung ketiga kunci kondisi untuk setiap jenis sumber daya, nilainya adalah Ya untuk layanan tersebut. Jika suatu layanan mendukung ketiga kunci kondisi untuk hanya beberapa jenis sumber daya, nilainya adalah Parsial.

Untuk informasi selengkapnya tentang ABAC, lihat Tentukan izin dengan otorisasi ABAC dalam Panduan Pengguna IAM. Untuk melihat tutorial yang menguraikan langkah-langkah pengaturan ABAC, lihat Menggunakan kontrol akses berbasis atribut (ABAC) dalam Panduan Pengguna IAM.

Untuk melihat contoh kebijakan berbasis identitas untuk membatasi akses ke sumber daya berdasarkan tag untuk sumber daya tersebut, lihat. Contoh: Gunakan tag untuk mengontrol penggunaan sumber daya

Menggunakan kredensyal sementara dengan FIS AWS

Mendukung kredensial sementara: Ya

Beberapa Layanan AWS tidak berfungsi saat Anda masuk menggunakan kredensil sementara. Untuk informasi tambahan, termasuk yang Layanan AWS bekerja dengan kredensi sementara, lihat Layanan AWS yang bekerja dengan IAM di Panduan Pengguna IAM.

Anda menggunakan kredensyal sementara jika Anda masuk AWS Management Console menggunakan metode apa pun kecuali nama pengguna dan kata sandi. Misalnya, ketika Anda mengakses AWS menggunakan tautan masuk tunggal (SSO) perusahaan Anda, proses tersebut secara otomatis membuat kredensyal sementara. Anda juga akan secara otomatis membuat kredensial sementara ketika Anda masuk ke konsol sebagai seorang pengguna lalu beralih peran. Untuk informasi selengkapnya tentang peralihan peran, lihat Beralih dari pengguna ke peran IAM (konsol) dalam Panduan Pengguna IAM.

Anda dapat membuat kredensyal sementara secara manual menggunakan API AWS CLI atau AWS . Anda kemudian dapat menggunakan kredensil sementara tersebut untuk mengakses. AWS AWS merekomendasikan agar Anda secara dinamis menghasilkan kredensi sementara alih-alih menggunakan kunci akses jangka panjang. Untuk informasi selengkapnya, lihat Kredensial keamanan sementara di IAM.

Izin utama lintas layanan untuk FIS AWS

Mendukung sesi akses maju (FAS): Ya

Saat Anda menggunakan pengguna atau peran IAM untuk melakukan tindakan AWS, Anda dianggap sebagai prinsipal. Ketika Anda menggunakan beberapa layanan, Anda mungkin melakukan sebuah

tindakan yang kemudian menginisiasi tindakan lain di layanan yang berbeda. FAS menggunakan izin dari pemanggilan utama Layanan AWS, dikombinasikan dengan permintaan Layanan AWS untuk membuat permintaan ke layanan hilir. Permintaan FAS hanya dibuat ketika layanan menerima permintaan yang memerlukan interaksi dengan orang lain Layanan AWS atau sumber daya untuk menyelesaikannya. Dalam hal ini, Anda harus memiliki izin untuk melakukan kedua tindakan tersebut. Untuk detail kebijakan ketika mengajukan permintaan FAS, lihat Sesi akses maju.

Peran layanan untuk AWS FIS

Mendukung peran layanan: Ya

Peran layanan adalah peran IAM yang diambil oleh sebuah layanan untuk melakukan tindakan atas nama Anda. Administrator IAM dapat membuat, mengubah, dan menghapus peran layanan dari dalam IAM. Untuk informasi selengkapnya, lihat Buat sebuah peran untuk mendelegasikan izin ke Layanan AWS dalam Panduan pengguna IAM.

Peran terkait layanan untuk FIS AWS

Mendukung peran terkait layanan: Ya

Peran terkait layanan adalah jenis peran layanan yang ditautkan ke. Layanan AWS Layanan tersebut dapat menjalankan peran untuk melakukan tindakan atas nama Anda. Peran terkait layanan muncul di Anda Akun AWS dan dimiliki oleh layanan. Administrator IAM dapat melihat, tetapi tidak dapat mengedit izin untuk peran terkait layanan.

Untuk detail tentang membuat atau mengelola peran terkait layanan AWS FIS, lihat. Gunakan peran terkait layanan untuk Layanan Injeksi AWS Kesalahan

AWS Contoh kebijakan Layanan Injeksi Kesalahan

Secara default, pengguna dan peran tidak memiliki izin untuk membuat atau memodifikasi sumber daya AWS FIS. Mereka juga tidak dapat melakukan tugas dengan menggunakan AWS Management Console, AWS Command Line Interface (AWS CLI), atau AWS API. Untuk memberikan izin kepada pengguna untuk melakukan tindakan di sumber daya yang mereka perlukan, administrator IAM dapat membuat kebijakan IAM. Administrator kemudian dapat menambahkan kebijakan IAM ke peran, dan pengguna dapat mengambil peran.

Untuk mempelajari cara membuat kebijakan berbasis identitas IAM dengan menggunakan contoh dokumen kebijakan JSON ini, lihat Membuat kebijakan IAM (konsol) di Panduan Pengguna IAM.

Untuk detail tentang tindakan dan jenis sumber daya yang ditentukan oleh AWS FIS, termasuk format ARNs untuk setiap jenis sumber daya, lihat Tindakan, sumber daya, dan kunci kondisi untuk Layanan Injeksi AWS Kesalahan dalam Referensi Otorisasi Layanan.

Daftar Isi

- Praktik terbaik kebijakan
- Contoh: Gunakan konsol AWS FIS
- Contoh: Daftar tindakan AWS FIS yang tersedia
- Contoh: Membuat template eksperimen untuk tindakan tertentu
- Contoh: Mulai percobaan
- Contoh: Gunakan tag untuk mengontrol penggunaan sumber daya
- Contoh: Hapus template eksperimen dengan tag tertentu
- Contoh: Izinkan pengguna untuk melihat izin mereka sendiri
- Contoh: Gunakan tombol kondisi untuk ec2:InjectApiError
- Contoh: Gunakan tombol kondisi untuk aws:s3:bucket-pause-replication

Praktik terbaik kebijakan

Kebijakan berbasis identitas menentukan apakah seseorang dapat membuat, mengakses, atau menghapus sumber daya AWS FIS di akun Anda. Tindakan ini membuat Akun AWS Anda dikenai biaya. Ketika Anda membuat atau mengedit kebijakan berbasis identitas, ikuti panduan dan rekomendasi ini:

- Mulailah dengan kebijakan AWS terkelola dan beralih ke izin hak istimewa paling sedikit — Untuk mulai memberikan izin kepada pengguna dan beban kerja Anda, gunakan kebijakan AWS terkelola yang memberikan izin untuk banyak kasus penggunaan umum. Mereka tersedia di Anda Akun AWS. Kami menyarankan Anda mengurangi izin lebih lanjut dengan menentukan kebijakan yang dikelola AWS pelanggan yang khusus untuk kasus penggunaan Anda. Untuk informasi selengkapnya, lihat Kebijakan yang dikelola AWS atau Kebijakan yang dikelola AWS untuk fungsi tugas dalam Panduan Pengguna IAM.
- Menerapkan izin dengan hak akses paling rendah – Ketika Anda menetapkan izin dengan kebijakan IAM, hanya berikan izin yang diperlukan untuk melakukan tugas. Anda melakukannya dengan mendefinisikan tindakan yang dapat diambil pada sumber daya tertentu dalam kondisi tertentu, yang juga dikenal sebagai izin dengan hak akses paling rendah. Untuk informasi

selengkapnya tentang cara menggunakan IAM untuk mengajukan izin, lihat Kebijakan dan izin dalam IAM dalam Panduan Pengguna IAM.

- Gunakan kondisi dalam kebijakan IAM untuk membatasi akses lebih lanjut – Anda dapat menambahkan suatu kondisi ke kebijakan Anda untuk membatasi akses ke tindakan dan sumber daya. Sebagai contoh, Anda dapat menulis kondisi kebijakan untuk menentukan bahwa semua permintaan harus dikirim menggunakan SSL. Anda juga dapat menggunakan ketentuan untuk memberikan akses ke tindakan layanan jika digunakan melalui yang spesifik Layanan AWS, seperti AWS CloudFormation. Untuk informasi selengkapnya, lihat Elemen kebijakan JSON IAM: Kondisi dalam Panduan Pengguna IAM.
- Gunakan IAM Access Analyzer untuk memvalidasi kebijakan IAM Anda untuk memastikan izin yang aman dan fungsional – IAM Access Analyzer memvalidasi kebijakan baru dan yang sudah ada sehingga kebijakan tersebut mematuhi bahasa kebijakan IAM (JSON) dan praktik terbaik IAM. IAM Access Analyzer menyediakan lebih dari 100 pemeriksaan kebijakan dan rekomendasi yang dapat ditindaklanjuti untuk membantu Anda membuat kebijakan yang aman dan fungsional. Untuk informasi selengkapnya, lihat Validasi kebijakan dengan IAM Access Analyzer dalam Panduan Pengguna IAM.
- Memerlukan otentikasi multi-faktor (MFA) - Jika Anda memiliki skenario yang mengharuskan pengguna IAM atau pengguna root di Anda, Akun AWS aktifkan MFA untuk keamanan tambahan. Untuk meminta MFA ketika operasi API dipanggil, tambahkan kondisi MFA pada kebijakan Anda. Untuk informasi selengkapnya, lihat Amankan akses API dengan MFA dalam Panduan Pengguna IAM.

Untuk informasi selengkapnya tentang praktik terbaik dalam IAM, lihat Praktik terbaik keamanan di IAM dalam Panduan Pengguna IAM.

Contoh: Gunakan konsol AWS FIS

Untuk mengakses konsol Layanan Injeksi AWS Kesalahan, Anda harus memiliki set izin minimum. Izin ini harus memungkinkan Anda untuk daftar dan melihat rincian tentang sumber daya AWS FIS di Anda. Akun AWS Jika Anda membuat kebijakan berbasis identitas yang lebih ketat daripada izin minimum yang diperlukan, konsol tidak akan berfungsi sebagaimana mestinya untuk entitas (pengguna atau peran) dengan kebijakan tersebut.

Anda tidak perlu mengizinkan izin konsol minimum untuk pengguna yang melakukan panggilan hanya ke AWS CLI atau AWS API. Sebagai gantinya, izinkan akses hanya ke tindakan yang sesuai dengan operasi API yang coba mereka lakukan.

Contoh kebijakan berikut memberikan izin untuk membuat daftar dan melihat semua sumber daya AWS FIS menggunakan konsol AWS FIS, tetapi tidak untuk membuat, memperbarui, atau menghapusnya. Ini juga memberikan izin untuk melihat sumber daya yang tersedia yang digunakan oleh semua tindakan AWS FIS yang dapat Anda tentukan dalam templat eksperimen.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "FISReadOnlyActions",
            "Effect": "Allow",
            "Action": [
                "fis:List*",
                "fis:Get*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AdditionalReadOnlyActions",
            "Effect": "Allow",
            "Action": [
                "ssm:Describe*",
                "ssm:Get*",
                "ssm:List*",
                "ec2:DescribeInstances",
                "rds:DescribeDBClusters",
                "ecs:DescribeClusters",
                "ecs:ListContainerInstances",
                "eks:DescribeNodegroup",
                "cloudwatch:DescribeAlarms",
                "iam:ListRoles"
            ],
            "Resource": "*"
        },
        {
            "Sid": "PermissionsToCreateServiceLinkedRole",
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:AWSServiceName": "fis.amazonaws.com"
                }
```

```
            }
        }
    ]
}
```

Contoh: Daftar tindakan AWS FIS yang tersedia

Kebijakan berikut memberikan izin untuk mencantumkan tindakan AWS FIS yang tersedia.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "fis:ListActions"
            ],
            "Resource": "arn:aws:fis:*:*:action/*"
        }
    ]
}
```

Contoh: Membuat template eksperimen untuk tindakan tertentu

Kebijakan berikut memberikan izin untuk membuat templat eksperimen untuk tindakan `aws:ec2:stop-instances` tersebut.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "PolicyExample",
      "Effect": "Allow",
      "Action": [
          "fis:CreateExperimentTemplate"
      ],
      "Resource": [
        "arn:aws:fis:*:*:action/aws:ec2:stop-instances",
        "arn:aws:fis:*:*:experiment-template/*"
      ]
    },
    {
```

```
      "Sid": "PolicyPassRoleExample",
      "Effect": "Allow",
      "Action": [
          "iam:PassRole"
      ],
      "Resource": [
        "arn:aws:iam::account-id:role/role-name"
      ]
    }
  ]
}
```

Contoh: Mulai percobaan

Kebijakan berikut memberikan izin untuk memulai eksperimen menggunakan peran IAM dan templat eksperimen yang ditentukan. Ini juga memungkinkan AWS FIS untuk membuat peran terkait layanan atas nama pengguna. Untuk informasi selengkapnya, lihat Gunakan peran terkait layanan untuk Layanan Injeksi AWS Kesalahan.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "PolicyExample",
      "Effect": "Allow",
      "Action": [
          "fis:StartExperiment"
      ],
      "Resource": [
        "arn:aws:fis:*:*:experiment-template/experiment-template-id",
        "arn:aws:fis:*:*:experiment/*"
      ]
    },
    {
        "Sid": "PolicyExampleforServiceLinkedRole",
        "Effect": "Allow",
        "Action": "iam:CreateServiceLinkedRole",
        "Resource": "*",
        "Condition": {
            "StringEquals": {
                "iam:AWSServiceName": "fis.amazonaws.com"
            }
        }
```

```
    }
  ]
}
```

Contoh: Gunakan tag untuk mengontrol penggunaan sumber daya

Kebijakan berikut memberikan izin untuk menjalankan eksperimen dari template eksperimen yang memiliki tag`Purpose=Test`. Itu tidak memberikan izin untuk membuat atau memodifikasi templat eksperimen, atau menjalankan eksperimen menggunakan templat yang tidak memiliki tag yang ditentukan.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "fis:StartExperiment",
            "Resource": "arn:aws:fis:*:*:experiment-template/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/Purpose": "Test"
                }
            }
        }
    ]
}
```

Contoh: Hapus template eksperimen dengan tag tertentu

Kebijakan berikut memberikan izin untuk menghapus templat eksperimen dengan tag`Purpose=Test`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "fis:DeleteExperimentTemplate"
            ],
            "Resource": "*",
```

```
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/Purpose": "Test"
                }
            }
        }
    ]
}
```

Contoh: Izinkan pengguna untuk melihat izin mereka sendiri

Contoh ini menunjukkan cara membuat kebijakan yang mengizinkan pengguna IAM melihat kebijakan inline dan terkelola yang dilampirkan ke identitas pengguna mereka. Kebijakan ini mencakup izin untuk menyelesaikan tindakan ini di konsol atau menggunakan API atau secara terprogram. AWS CLI AWS

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
```

```
            ],
            "Resource": "*"
        }
    ]
}
```

Contoh: Gunakan tombol kondisi untuk **ec2:InjectApiError**

Contoh kebijakan berikut menggunakan kunci `ec2:FisTargetArns` kondisi untuk cakupan sumber daya target. Kebijakan ini memungkinkan tindakan AWS FIS `aws:ec2:api-insufficient-instance-capacity-error` dan`aws:ec2:asg-insufficient-instance-capacity-error`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:InjectApiError",
            "Resource": "*",
            "Condition": {
                "ForAllValues:StringEquals": {
                    "ec2:FisActionId": [
                        "aws:ec2:api-insufficient-instance-capacity-error",
                    ],
                    "ec2:FisTargetArns": [
                        "arn:aws:iam:*:*:role:role-name"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "ec2:InjectApiError",
            "Resource": "*",
            "Condition": {
                "ForAllValues:StringEquals": {
                    "ec2:FisActionId": [
                        "aws:ec2:asg-insufficient-instance-capacity-error"
                    ],
                    "ec2:FisTargetArns": [

 "arn:aws:autoscaling:*:*:autoScalingGroup:uuid:autoScalingGroupName/asg-name"
                    ]
```

```
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "autoscaling:DescribeAutoScalingGroups",
            "Resource": "*"
        }
    ]
}
```

Contoh: Gunakan tombol kondisi untuk **aws:s3:bucket-pause-replication**

Contoh kebijakan berikut menggunakan kunci `S3:IsReplicationPauseRequest` kondisi untuk mengizinkan `PutReplicationConfiguration` dan `GetReplicationConfiguration` hanya bila digunakan oleh AWS FIS dalam konteks tindakan AWS FIS. `aws:s3:bucket-pause-replication`

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "S3:PauseReplication"
            ],
            "Resource": "arn:aws:s3:::mybucket",
            "Condition": {
                "StringEquals": {
                    "s3:DestinationRegion": "region"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "S3:PutReplicationConfiguration",
                "S3:GetReplicationConfiguration"
            ],
            "Resource": "arn:aws:s3:::mybucket",
            "Condition": {
                "BoolIfExists": {
                    "s3:IsReplicationPauseRequest": "true"
```

```
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "S3:ListBucket"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "tag:GetResources"
            ],
            "Resource": "*"
        }
    ]
    }
```

Gunakan peran terkait layanan untuk Layanan Injeksi AWS Kesalahan

AWS Layanan Injeksi Kesalahan menggunakan AWS Identity and Access Management peran terkait layanan (IAM). Peran terkait layanan adalah jenis unik peran IAM yang terhubung langsung ke FIS. AWS Peran terkait layanan telah ditentukan sebelumnya oleh AWS FIS dan mencakup semua izin yang diperlukan layanan untuk memanggil layanan lain AWS atas nama Anda.

Peran terkait layanan membuat pengaturan AWS FIS lebih mudah karena Anda tidak perlu menambahkan izin yang diperlukan secara manual untuk mengelola pemantauan dan pemilihan sumber daya untuk eksperimen. AWS FIS mendefinisikan izin dari peran terkait layanan, dan kecuali ditentukan lain, hanya AWS FIS yang dapat mengambil perannya. Izin yang ditentukan mencakup kebijakan kepercayaan dan kebijakan izin, serta bahwa kebijakan izin tidak dapat dilampirkan ke entitas IAM lainnya.

Selain peran terkait layanan, Anda juga harus menentukan peran IAM yang memberikan izin untuk mengubah sumber daya yang Anda tetapkan sebagai target dalam templat eksperimen. Untuk informasi selengkapnya, lihat Peran IAM untuk eksperimen AWS FIS.

Anda dapat menghapus peran tertaut layanan hanya setelah menghapus sumber daya terkait terlebih dahulu. Ini melindungi sumber daya AWS FIS Anda karena Anda tidak dapat secara tidak sengaja menghapus izin untuk mengakses sumber daya.

Izin peran terkait layanan untuk FIS AWS

AWS FIS menggunakan peran terkait layanan bernama AWSServiceRoleForFIS untuk memungkinkannya mengelola pemantauan dan pemilihan sumber daya untuk eksperimen.

Peran AWSServiceRoleForterkait layanan FIS mempercayai layanan berikut untuk mengambil peran:

- `fis.amazonaws.com`

AWSServiceRoleForPeran terkait layanan FIS menggunakan kebijakan terkelola Amazon. FISService RolePolicy Kebijakan ini memungkinkan AWS FIS untuk mengelola pemantauan dan pemilihan sumber daya untuk eksperimen. Untuk informasi selengkapnya, lihat Amazon FISService RolePolicy di Referensi Kebijakan AWS Terkelola.

Anda harus mengonfigurasi izin agar entitas IAM (seperti pengguna, grup, atau peran) dapat membuat, mengedit, atau menghapus peran terkait layanan. Agar peran terkait layanan AWSServiceRoleForFIS berhasil dibuat, identitas IAM yang Anda gunakan AWS FIS harus memiliki izin yang diperlukan. Untuk memberikan izin yang diperlukan, lampirkan kebijakan berikut untuk identitas IAM.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "*",
            "Condition": {
                "StringLike": {
                    "iam:AWSServiceName": "fis.amazonaws.com"
                }
            }
        }
    ]
}
```

Untuk informasi lebih lanjut, lihat Izin Peran Terkait Layanan dalam Panduan Pengguna IAM.

Buat peran terkait layanan untuk FIS AWS

Anda tidak perlu membuat peran terkait layanan secara manual. Saat Anda memulai eksperimen AWS FIS di AWS Management Console, the AWS CLI, atau AWS API, AWS FIS membuat peran terkait layanan untuk Anda.

Jika Anda menghapus peran terkait layanan ini, dan ingin membuatnya lagi, Anda dapat mengulangi proses yang sama untuk membuat kembali peran tersebut di akun Anda. Ketika Anda memulai eksperimen AWS FIS, AWS FIS menciptakan peran terkait layanan untuk Anda lagi.

Mengedit peran terkait layanan untuk FIS AWS

AWS FIS tidak memungkinkan Anda untuk mengedit peran terkait layanan AWSServiceRoleForFIS. 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 selengkapnya, lihat Mengedit peran terkait layanan dalam Panduan Pengguna IAM.

Menghapus peran terkait layanan untuk FIS AWS

Jika Anda tidak perlu lagi menggunakan fitur atau layanan yang memerlukan peran terkait layanan, sebaiknya hapus 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.

> **Note**
>
> Jika layanan AWS FIS menggunakan peran ketika Anda mencoba untuk membersihkan sumber daya, maka pembersihan mungkin gagal. Jika hal itu terjadi, tunggu beberapa menit dan coba mengoperasikannya lagi.

Untuk membersihkan sumber daya AWS FIS yang digunakan oleh FIS AWSService RoleFor

Pastikan tidak ada eksperimen Anda yang sedang berjalan. Jika perlu, hentikan eksperimen Anda. Untuk informasi selengkapnya, lihat Menghentikan sebuah percobaan.

Untuk menghapus peran tertaut layanan secara manual menggunakan IAM

Gunakan konsol IAM, the AWS CLI, atau AWS API untuk menghapus peran terkait layanan AWSServiceRoleForFIS. Untuk informasi selengkapnya, lihat Menghapus peran terkait layanan dalam Panduan Pengguna IAM.

Wilayah yang Didukung untuk AWS peran terkait layanan FIS

AWS FIS mendukung penggunaan peran terkait layanan di semua Wilayah di mana layanan tersedia. Untuk informasi selengkapnya, lihat titik akhir dan kuota Layanan Injeksi AWS Kesalahan.

AWS kebijakan terkelola untuk Layanan Injeksi AWS Kesalahan

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, pemutakhiran 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.

AWS kebijakan terkelola: Amazon FISService RolePolicy

Kebijakan ini dilampirkan pada peran terkait layanan bernama AWSServiceRoleForFIS untuk memungkinkan AWS FIS mengelola pemantauan dan pemilihan sumber daya untuk eksperimen. Untuk informasi selengkapnya, lihat Gunakan peran terkait layanan untuk Layanan Injeksi AWS Kesalahan.

AWS kebijakan terkelola: AWSFault InjectionSimulator EC2 Akses

Gunakan kebijakan ini dalam peran eksperimen untuk memberikan izin AWS FIS untuk menjalankan eksperimen yang menggunakan tindakan AWS FIS untuk Amazon. EC2 Untuk informasi selengkapnya, lihat the section called “Peran percobaan”.

Untuk melihat izin kebijakan ini, lihat AWSFaultInjectionSimulatorEC2Akses di Referensi Kebijakan AWS Terkelola.

AWS kebijakan terkelola: AWSFault InjectionSimulator ECSAccess

Gunakan kebijakan ini dalam peran eksperimen untuk memberikan izin AWS FIS untuk menjalankan eksperimen yang menggunakan tindakan AWS FIS untuk Amazon ECS. Untuk informasi selengkapnya, lihat the section called “Peran percobaan”.

Untuk melihat izin kebijakan ini, lihat AWSFaultInjectionSimulatorECSAccessdi Referensi Kebijakan AWS Terkelola.

AWS kebijakan terkelola: AWSFault InjectionSimulator EKSAccess

Gunakan kebijakan ini dalam peran eksperimen untuk memberikan izin AWS FIS untuk menjalankan eksperimen yang menggunakan tindakan AWS FIS untuk Amazon EKS. Untuk informasi selengkapnya, lihat the section called “Peran percobaan”.

Untuk melihat izin kebijakan ini, lihat AWSFaultInjectionSimulatorEKSAccessdi Referensi Kebijakan AWS Terkelola.

AWS kebijakan terkelola: AWSFault InjectionSimulatorNetworkAccess

Gunakan kebijakan ini dalam peran eksperimen untuk memberikan izin AWS FIS untuk menjalankan eksperimen yang menggunakan tindakan jaringan AWS FIS. Untuk informasi selengkapnya, lihat the section called “Peran percobaan”.

Untuk melihat izin kebijakan ini, lihat AWSFaultInjectionSimulatorNetworkAccessdi Referensi Kebijakan AWS Terkelola.

AWS kebijakan terkelola: AWSFault InjectionSimulator RDSAccess

Gunakan kebijakan ini dalam peran eksperimen untuk memberikan izin AWS FIS untuk menjalankan eksperimen yang menggunakan tindakan AWS FIS untuk Amazon RDS. Untuk informasi selengkapnya, lihat the section called “Peran percobaan”.

Untuk melihat izin kebijakan ini, lihat AWSFaultInjectionSimulatorRDSAccessdi Referensi Kebijakan AWS Terkelola.

AWS kebijakan terkelola: AWSFault InjectionSimulator SSMAccess

Gunakan kebijakan ini dalam peran eksperimen untuk memberikan izin AWS FIS untuk menjalankan eksperimen yang menggunakan tindakan AWS FIS untuk Systems Manager. Untuk informasi selengkapnya, lihat the section called “Peran percobaan”.

Untuk melihat izin kebijakan ini, lihat AWSFaultInjectionSimulatorSSMAccessdi Referensi Kebijakan AWS Terkelola.

AWS Pembaruan FIS untuk kebijakan AWS terkelola

Lihat detail tentang pembaruan kebijakan AWS terkelola untuk AWS FIS sejak layanan ini mulai melacak perubahan ini.

Perubahan	Deskripsi	Tanggal
AWSFaultInjectionSimulatorECSAccess — Permbaruan ke kebijakan yang sudah ada	Menambahkan izin untuk memungkinkan AWS FIS menyelesaikan target ECS.	Januari 25, 2024
AWSFaultInjectionSimulatorNetworkAccess – Pembaruan ke kebijakan yang ada	Menambahkan izin untuk memungkinkan AWS FIS menjalankan eksperimen menggunakan aws:network:route-table-disrupt-cross-region-connectivity dan aws:network:transit-gateway-disrupt-cross-region-connectivity tindakan.	Januari 25, 2024
AWSFaultInjectionSimulatorEC2Akses — Perbarui ke kebijakan yang ada	Menambahkan izin untuk memungkinkan AWS FIS menyelesaikan EC2 instance.	13 November 2023
AWSFaultInjectionSimulatorEKSAccess – Pembaruan ke kebijakan yang ada	Menambahkan izin untuk memungkinkan AWS FIS menyelesaikan target EKS.	13 November 2023
AWSFaultInjectionSimulatorRDSAccess – Pembaruan ke kebijakan yang ada	Menambahkan izin untuk memungkinkan AWS FIS menyelesaikan target RDS.	13 November 2023

Perubahan	Deskripsi	Tanggal
AWSFaultInjectionSimulatorEC2Akses — Perbarui ke kebijakan yang ada	Menambahkan izin untuk memungkinkan AWS FIS menjalankan dokumen SSM pada EC2 instance dan untuk menghentikan instance. EC2	Juni 2, 2023
AWSFaultInjectionSimulatorSSMAccess – Pembaruan ke kebijakan yang ada	Menambahkan izin untuk memungkinkan AWS FIS menjalankan dokumen SSM pada instance. EC2	Juni 2, 2023
AWSFaultInjectionSimulatorECSAccess – Pembaruan ke kebijakan yang ada	Menambahkan izin untuk memungkinkan AWS FIS menjalankan eksperimen menggunakan tindakan baruaws:ecs:task.	1 Juni 2023
AWSFaultInjectionSimulatorEKSAccess – Pembaruan ke kebijakan yang ada	Menambahkan izin untuk memungkinkan AWS FIS menjalankan eksperimen menggunakan tindakan baruaws:eks:pod.	1 Juni 2023
AWSFaultInjectionSimulatorEC2Akses — Kebijakan baru	Menambahkan kebijakan untuk memungkinkan AWS FIS menjalankan eksperimen yang menggunakan tindakan AWS FIS untuk Amazon. EC2	26 Oktober 2022
AWSFaultInjectionSimulatorECSAccess – Kebijakan baru	Menambahkan kebijakan untuk memungkinkan AWS FIS menjalankan eksperimen yang menggunakan tindakan AWS FIS untuk Amazon ECS.	26 Oktober 2022
AWSFaultInjectionSimulatorEKSAccess – Kebijakan baru	Menambahkan kebijakan untuk mengizinkan AWS FIS menjalankan eksperimen yang menggunakan tindakan AWS FIS untuk Amazon EKS.	26 Oktober 2022

Perubahan	Deskripsi	Tanggal
AWSFaultInjectionSimulatorNetworkAccess – Kebijakan baru	Menambahkan kebijakan untuk memungkinkan AWS FIS menjalankan eksperimen yang menggunakan tindakan jaringan AWS FIS.	26 Oktober 2022
AWSFaultInjectionSimulatorRDSAccess – Kebijakan baru	Menambahkan kebijakan untuk memungkinkan AWS FIS menjalankan eksperimen yang menggunakan tindakan AWS FIS untuk Amazon RDS.	26 Oktober 2022
AWSFaultInjectionSimulatorSSMAccess – Kebijakan baru	Menambahkan kebijakan untuk memungkinkan AWS FIS menjalankan eksperimen yang menggunakan tindakan AWS FIS untuk Systems Manager.	26 Oktober 2022
Amazon FISService RolePolicy - Perbarui ke kebijakan yang ada	Menambahkan izin untuk memungkinkan AWS FIS mendeskripsikan subnet.	26 Oktober 2022
Amazon FISService RolePolicy - Perbarui ke kebijakan yang ada	Menambahkan izin untuk memungkinkan AWS FIS mendeskripsikan kluster EKS.	Juli 7, 2022
Amazon FISService RolePolicy - Perbarui ke kebijakan yang ada	Menambahkan izin untuk memungkinkan AWS FIS untuk daftar dan menjelaskan tugas-tugas di cluster Anda.	Februari 7, 2022
Amazon FISService RolePolicy - Perbarui ke kebijakan yang ada	Menghapus `events:ManagedBy` kondisi untuk `events:DescribeRule` tindakan.	6 Januari 2022

Perubahan	Deskripsi	Tanggal
Amazon FISService RolePolicy - Perbarui ke kebijakan yang ada	Menambahkan izin untuk memungkinkan AWS FIS mengambil riwayat CloudWatch alarm yang digunakan dalam kondisi berhenti.	30 Juni 2021
AWS FIS mulai melacak perubahan	AWS FIS mulai melacak perubahan pada kebijakan yang AWS dikelola	1 Maret 2021

Keamanan infrastruktur di Layanan Injeksi AWS Kesalahan

Sebagai layanan terkelola, AWS Fault Injection Service dilindungi oleh keamanan jaringan AWS global. Untuk informasi tentang layanan AWS keamanan dan cara AWS melindungi infrastruktur, lihat Keamanan AWS Cloud. Untuk mendesain AWS lingkungan Anda menggunakan praktik terbaik untuk keamanan infrastruktur, lihat Perlindungan Infrastruktur dalam Kerangka Kerja yang AWS Diarsiteksikan dengan Baik Pilar Keamanan.

Anda menggunakan panggilan API yang AWS dipublikasikan untuk mengakses AWS FIS melalui jaringan. Klien harus mendukung hal-hal berikut:

- Keamanan Lapisan Pengangkutan (TLS). Kami mensyaratkan TLS 1.2 dan menganjurkan TLS 1.3.
- Sandi cocok dengan sistem kerahasiaan maju sempurna (perfect forward secrecy, PFS) seperti DHE (Ephemeral Diffie-Hellman) atau ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Sebagian besar sistem modern seperti Java 7 dan versi lebih baru mendukung mode-mode ini.

Selain itu, permintaan harus ditandatangani menggunakan ID kunci akses dan kunci akses rahasia yang terkait dengan prinsipal IAM. Atau Anda dapat menggunakan AWS Security Token Service (AWS STS) untuk menghasilkan kredensial keamanan sementara untuk menandatangani permintaan.

Akses AWS FIS menggunakan antarmuka VPC endpoint ()AWS PrivateLink

Anda dapat membuat koneksi pribadi antara VPC dan Layanan Injeksi AWS Kesalahan Anda dengan membuat titik akhir VPC antarmuka. Titik akhir VPC didukung oleh AWS PrivateLink, teknologi yang

memungkinkan Anda mengakses AWS FIS secara pribadi APIs tanpa gateway internet, perangkat NAT, koneksi VPN, atau koneksi Direct Connect. AWS Instans di VPC Anda tidak memerlukan alamat IP publik untuk berkomunikasi AWS dengan FIS. APIs

Setiap titik akhir antarmuka diwakili oleh satu atau lebih antarmuka jaringan elastis dalam subnet Anda.

Untuk informasi selengkapnya, lihat Akses Layanan AWS melalui AWS PrivateLink di AWS PrivateLink Panduan.

Pertimbangan untuk titik akhir AWS VPC FIS

Sebelum Anda menyiapkan titik akhir VPC antarmuka untuk AWS FIS, tinjau Akses menggunakan titik akhir VPC Layanan AWS antarmuka di Panduan.AWS PrivateLink

AWS FIS mendukung panggilan ke semua tindakan API-nya dari VPC Anda.

Buat antarmuka VPC endpoint untuk FIS AWS

Anda dapat membuat titik akhir VPC untuk layanan AWS FIS menggunakan konsol VPC Amazon atau (). AWS Command Line Interface AWS CLI Untuk informasi selengkapnya, lihat Membuat titik akhir VPC di Panduan Pengguna AWS PrivateLink .

Buat titik akhir VPC untuk AWS FIS menggunakan nama layanan berikut:. `com.amazonaws.region.fis`

Jika Anda mengaktifkan DNS pribadi untuk titik akhir, Anda dapat membuat permintaan API ke AWS FIS menggunakan nama DNS default untuk Wilayah, misalnya,. `fis.us-east-1.amazonaws.com`

Membuat kebijakan titik akhir VPC untuk FIS AWS

Anda dapat melampirkan kebijakan titik akhir ke titik akhir VPC Anda yang mengontrol akses ke FIS. AWS Kebijakan titik akhir menentukan informasi berikut:

- Prinsipal yang dapat melakukan tindakan.
- Tindakan yang dapat dilakukan.
- Sumber daya yang menjadi target tindakan.

Untuk informasi selengkapnya, lihat Mengontrol akses ke titik akhir VPC menggunakan kebijakan titik akhir di Panduan.AWS PrivateLink

Contoh: Kebijakan titik akhir VPC untuk tindakan FIS tertentu AWS

Kebijakan titik akhir VPC berikut memberikan akses ke tindakan AWS FIS yang terdaftar pada semua sumber daya ke semua prinsipal.

```
{
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "fis:ListExperimentTemplates",
            "fis:StartExperiment",
            "fis:StopExperiment",
            "fis:GetExperiment"
         ],
         "Resource":"*",
         "Principal":"*"
      }
   ]
}
```

Contoh: Kebijakan titik akhir VPC yang menolak akses dari yang spesifik Akun AWS

Kebijakan titik akhir VPC berikut menolak Akun AWS akses yang ditentukan ke semua tindakan dan sumber daya, tetapi memberikan semua Akun AWS akses lain ke semua tindakan dan sumber daya.

```
{
   "Statement":[
      {
         "Effect": "Allow",
         "Action": "*",
         "Resource": "*",
         "Principal": "*"
      },
      {
         "Effect":"Deny",
         "Action": "*",
         "Resource": "*",
         "Principal": {
           "AWS": [ "123456789012" ]
         }
      }
   ]
```

```
}
```

Menandai sumber daya AWS FIS Anda

Tag adalah label metadata yang Anda atau tetapkan ke AWS sumber daya. AWS Setiap tanda terdiri atas kunci dan nilai. Untuk tanda yang Anda tetapkan, Anda menentukan kunci dan nilai. Misalnya, Anda mungkin mendefinisikan kunci sebagai `purpose` dan nilai sebagai `test` sumber daya.

Tanda membantu Anda melakukan hal berikut:

- Identifikasi dan atur AWS sumber daya Anda. Banyak AWS layanan mendukung penandaan, sehingga Anda dapat menetapkan tag yang sama ke sumber daya dari layanan yang berbeda untuk menunjukkan bahwa sumber daya terkait.
- Kontrol akses ke AWS sumber daya Anda. Untuk informasi selengkapnya, lihat Mengontrol Akses Menggunakan Tanda di Panduan Pengguna IAM.

Pembatasan penandaan

Pembatasan dasar berikut berlaku untuk tag pada sumber daya AWS FIS:

- Jumlah tanda maksimum tag yang dapat Anda tetapkan ke sumber daya: 50
- Panjang kunci maksimum: 128 karakter Unicode
- Panjang nilai maksimum: 256 karakter Unicode
- Karakter yang valid untuk kunci dan nilai: a-z, A-Z, 0-9, spasi, dan karakter berikut: _.:/= + - dan @
- Kunci dan nilai tanda peka huruf besar-kecil
- Anda tidak dapat menggunakan `aws:` sebagai awalan untuk kunci, karena itu dicadangkan untuk AWS digunakan

Bekerja dengan tag

Sumber daya AWS Fault Injection Service (AWS FIS) berikut mendukung penandaan:

- Tindakan
- Eksperimen
- Templat eksperimen

Anda dapat menggunakan konsol untuk bekerja dengan tag untuk eksperimen dan templat eksperimen. Untuk informasi selengkapnya, lihat berikut ini:

- Tandai eksperimen
- Tag template percobaan

Anda dapat menggunakan AWS CLI perintah berikut untuk bekerja dengan tag untuk tindakan, eksperimen, dan templat eksperimen:

- tag-resource - Tambahkan tag ke sumber daya.
- untag-resource — Hapus tag dari sumber daya.
- list-tags-for-resource— Daftar tag untuk sumber daya tertentu.

Kuota dan batasan untuk Layanan Injeksi AWS Kesalahan

Anda Akun AWS memiliki kuota default, sebelumnya disebut sebagai batas, untuk setiap layanan. AWS Kecuali dinyatakan lain, setiap kuota bersifat spesifik wilayah. Anda dapat meminta kenaikan kuota yang ditandai sebagai dapat disesuaikan pada tabel di bawah ini.

Untuk melihat kuota AWS FIS di akun Anda, buka konsol Service Quotas. Di panel navigasi, pilih AWS layanan dan pilih Layanan Injeksi AWS Kesalahan. Nilai hingga dan termasuk kuota yang disetujui secara otomatis diterapkan secara instan. Kuota yang disetujui secara otomatis diuraikan dalam kolom deskripsi pada tabel di bawah ini. Jika Anda memerlukan kuota yang melebihi batas yang disetujui secara otomatis, kirimkan permintaan. Nilai di atas batas yang disetujui secara otomatis ditinjau oleh dukungan pelanggan dan disetujui jika memungkinkan.

Untuk meminta penambahan kuota, lihat Meminta penambahan kuota di Panduan Pengguna Service Quotas.

Anda Akun AWS memiliki kuota berikut yang terkait dengan AWS FIS.

Nama	Default	Dapat disesuaikan	Deskripsi
Durasi aksi dalam jam	Setiap Wilayah yang didukung: 12	Tidak	Jumlah maksimum jam yang diizinkan untuk menjalankan satu tindakan di akun ini di Wilayah saat ini.
Tindakan per templat eksperimen	Setiap Wilayah yang didukung: 20	Tidak	Jumlah maksimum tindakan yang dapat Anda buat dalam template percobaan di akun ini di Wilayah saat ini.
Eksperimen aktif	Setiap Wilayah yang didukung: 5	Tidak	Jumlah maksimum eksperimen aktif yang dapat Anda jalankan

Nama	Default	Dapat disesu an	Deskripsi
			secara bersamaan di akun ini di Wilayah saat ini.
Menyelesaikan retensi data eksperimen dalam beberapa hari	Setiap Wilayah yang didukung: 120	Tidak	Jumlah hari maksimum yang diizinkan AWS FIS untuk menyimpan data tentang eksperimen yang diselesaikan dalam akun ini di Wilayah saat ini.
Durasi percobaan dalam beberapa jam	Setiap Wilayah yang didukung: 12	Tidak	Jumlah maksimum jam yang diizinkan untuk menjalankan satu percobaan di akun ini di Wilayah saat ini.
Templat eksperimen	Setiap Wilayah yang didukung: 500	Tidak	Jumlah maksimum templat eksperimen yang dapat Anda buat di akun ini di Wilayah saat ini.
Jumlah maksimum Daftar Awalan Terkelola di aws:network: -region-c onnectivity route-table-disrupt-cross	Setiap Wilayah yang didukung: 15	Tidak	Jumlah maksimum Daftar Awalan Terkelola yang aws:network: route-table-disrupt-cross -region-connectivity akan memungkinkan, per tindakan.

Nama	Default	Dapat disesuaikan	Deskripsi
Jumlah maksimum Tabel Rute di aws:network: -region-connectivity route-table-disrupt-cross	Setiap Wilayah yang didukung: 10	Tidak	Jumlah maksimum Tabel Rute yang aws:network: route-table-disrupt-cross -region-connectivity akan memungkinkan, per tindakan.
Jumlah maksimum rute di aws:network: -region-connectivity route-table-disrupt-cross	Setiap Wilayah yang didukung: 200	Tidak	Jumlah maksimum rute yang aws:network: route-table-disrupt-cross -region-connectivity akan memungkinkan, per tindakan.
Tindakan paralel per eksperimen	Setiap Wilayah yang didukung: 10	Tidak	Jumlah maksimum tindakan yang dapat Anda jalankan secara paralel dalam percobaan di akun ini di Wilayah saat ini.
Syarat Stop per templat eksperimen	Setiap Wilayah yang didukung: 5	Tidak	Jumlah maksimum kondisi berhenti yang dapat Anda tambahkan ke templat eksperimen di akun ini di Wilayah saat ini.

Nama	Default	Dapat disesuaikan	Deskripsi
Target grup Auto Scaling untuk aws:ec2: -error asg-insufficient-instance-capacity	Setiap Wilayah yang didukung: 5	Ya	Jumlah maksimum grup Auto Scaling yang aws:ec2: asg-insufficient-instance-capacity -error dapat menargetkan saat Anda mengidentifikasi target menggunakan tag, per percobaan. Permintaan kenaikan kuota akan secara otomatis disetujui untuk nilai hingga 500.
Bucket Target untuk aws:s3: bucket-pause-replication	Setiap Wilayah yang didukung: 20	Ya	Jumlah maksimum Bucket S3 yang aws:s3: bucket-pause-replication dapat menargetkan saat Anda mengidentifikasi target menggunakan tag, per percobaan. Permintaan kenaikan kuota akan secara otomatis disetujui untuk nilai hingga 25.
Cluster Target untuk aws:ecs: drain-container-instances	Setiap Wilayah yang didukung: 5	Ya	Jumlah maksimum Cluster yang aws:ecs: drain-container-instances dapat menargetkan ketika Anda mengidentifikasi target menggunakan tag, per percobaan. Permintaan kenaikan kuota akan secara otomatis disetujui untuk nilai hingga 100.

Nama	Default	Dapat disesuaikan	Deskripsi
Cluster Target untuk aws:rds: failover-db-cluster	Setiap Wilayah yang didukung: 5	Ya	Jumlah maksimum Cluster yang aws:rds: failover-db-cluster dapat menargetkan ketika Anda mengidentifikasi target menggunakan tag, per percobaan. Permintaan kenaikan kuota akan secara otomatis disetujui untuk nilai hingga 160.
Target DBInstances untuk aws:rds: reboot-db-instances	Setiap Wilayah yang didukung: 5	Ya	Jumlah maksimum aws:rds: reboot-db-instances dapat menargetkan saat Anda mengidentifikasi target menggunakan tag, per percobaan. DBInstances Permintaan kenaikan kuota akan secara otomatis disetujui untuk nilai hingga 130.
Instance Target untuk aws:ec2:reboot-instances	Setiap Wilayah yang didukung: 5	Ya	Jumlah maksimum Instans yang aws:ec2:reboot-instances dapat menargetkan saat Anda mengidentifikasi target menggunakan tag, per percobaan. Permintaan kenaikan kuota akan secara otomatis disetujui untuk nilai hingga 600.

Nama	Default	Dapat disesuaikan	Deskripsi
Instance Target untuk aws:ec2:stop-instance	Setiap Wilayah yang didukung: 5	Ya	Jumlah maksimum Instans yang aws:ec2:stop-instances dapat menargetkan saat Anda mengidentifikasi target menggunakan tag, per percobaan. Permintaan kenaikan kuota akan secara otomatis disetujui untuk nilai hingga 400.
Instance Target untuk aws:ec2:terminate-instance	Setiap Wilayah yang didukung: 5	Ya	Jumlah maksimum Instans yang aws:ec2:terminate-instances dapat menargetkan saat Anda mengidentifikasi target menggunakan tag, per percobaan. Permintaan kenaikan kuota akan secara otomatis disetujui untuk nilai hingga 300.
Contoh Target untuk aws:ssm:send-command	Setiap Wilayah yang didukung: 5	Ya	Jumlah maksimum Instans yang aws:ssm:send-command dapat menargetkan saat Anda mengidentifikasi target menggunakan tag, per percobaan. Permintaan kenaikan kuota akan secara otomatis disetujui untuk nilai hingga 50.

Nama	Default	Dapat disesuaikan	Deskripsi
Target Nodegroups untuk aws:eks: terminate-nodegroup-instances	Setiap Wilayah yang didukung: 5	Ya	Jumlah maksimum Nodegroups yang aws:eks: terminate-nodegroup-instances dapat menargetkan ketika Anda mengidentifikasi target menggunakan tag, per percobaan. Permintaan kenaikan kuota akan secara otomatis disetujui untuk nilai hingga 100.
Target Pod untuk aws:eks: pod-cpu-stress	Setiap Wilayah yang didukung: 50	Ya	Jumlah maksimum Pod yang aws:eks: pod-cpu-stress dapat menargetkan ketika Anda mengidentifikasi target menggunakan parameter, per percobaan. Permintaan kenaikan kuota akan secara otomatis disetujui untuk nilai hingga 1000.
Target Pod untuk aws:eks:pod-delete	Setiap Wilayah yang didukung: 50	Ya	Jumlah maksimum Pod yang aws:eks:pod-delete dapat menargetkan ketika Anda mengidentifikasi target menggunakan parameter, per percobaan. Permintaan kenaikan kuota akan secara otomatis disetujui untuk nilai hingga 1000.

Nama	Default	Dapat disesuaikan	Deskripsi
Target Pod untuk aws:eks: pod-io-stress	Setiap Wilayah yang didukung: 50	Ya	Jumlah maksimum Pod yang aws:eks: pod-io-stress dapat menargetkan ketika Anda mengidentifikasi target menggunakan parameter, per percobaan. Permintaan kenaikan kuota akan secara otomatis disetujui untuk nilai hingga 1000.
Target Pod untuk aws:eks: pod-memory-stress	Setiap Wilayah yang didukung: 50	Ya	Jumlah maksimum Pod yang aws:eks: pod-memory-stress dapat menargetkan ketika Anda mengidentifikasi target menggunakan parameter, per percobaan. Permintaan kenaikan kuota akan secara otomatis disetujui untuk nilai hingga 1000.
Target Pod untuk aws:eks: pod-network-blackhole-port	Setiap Wilayah yang didukung: 50	Ya	Jumlah maksimum Pod yang aws:eks: pod-network-blackhole-port dapat menargetkan ketika Anda mengidentifikasi target menggunakan parameter, per percobaan. Permintaan kenaikan kuota akan secara otomatis disetujui untuk nilai hingga 1000.

Nama	Default	Dapat disesuaikan	Deskripsi
Target Pod untuk aws:eks: pod-network-latency	Setiap Wilayah yang didukung: 50	Ya	Jumlah maksimum Pod yang aws:eks: pod-network-latency dapat menargetkan ketika Anda mengidentifikasi target menggunakan parameter, per percobaan. Permintaan kenaikan kuota akan secara otomatis disetujui untuk nilai hingga 1000.
Target Pod untuk aws:eks: pod-network-packet-loss	Setiap Wilayah yang didukung: 50	Ya	Jumlah maksimum Pod yang aws:eks: pod-network-packet-loss dapat menargetkan ketika Anda mengidentifikasi target menggunakan parameter, per percobaan. Permintaan kenaikan kuota akan secara otomatis disetujui untuk nilai hingga 1000.

Nama	Default	Dapat disesuaikan	Deskripsi
Target ReplicationGroups untuk aws:elasticache: - Penghentian Direncanakan interrupt-cluster-az-power	Setiap Wilayah yang didukung: 5	Ya	Jumlah maksimum aws:elasticache: interrupt-cluster-az-power dapat menargetkan saat Anda mengidentifikasi target menggunakan tag/parameter, per percobaan. ReplicationGroups Permintaan kenaikan kuota akan secara otomatis disetujui untuk nilai hingga 5.
Target ReplicationGroups untuk aws:elasticache: replicationgroup-interrupt-az-power	Setiap Wilayah yang didukung: 5	Ya	Jumlah maksimum aws:elasticache: replicationgroup-interrupt-az-power dapat menargetkan per percobaan. ReplicationGroups Batas harian berlaku untuk penargetan. ReplicationGroups Untuk informasi lebih lanjut, kunjungi: https://docs.aws.amazon.com/fis/ latest/userguide/fis-quotas.html. Permintaan kenaikan kuota akan secara otomatis disetujui untuk nilai hingga 20.

Nama	Default	Dapat disesuaikan	Deskripsi
Target SpotInstances untuk aws:ec2:send-spot-instance-interruptions	Setiap Wilayah yang didukung: 5	Ya	Jumlah maksimum aws:ec2: send-spot-instance-interruptions dapat menargetkan saat Anda mengidentifikasi target menggunakan tag, per percobaan. SpotInstances Permintaan kenaikan kuota akan secara otomatis disetujui untuk nilai hingga 100.
Subnet Target untuk aws:network:disrupt-connectivity	Setiap Wilayah yang didukung: 5	Ya	Jumlah maksimum Subnet yang aws:network:disrupt-connectivity dapat menargetkan saat Anda mengidentifikasi target menggunakan tag, per percobaan. Kuota di atas 5 hanya berlaku untuk cakupan parameter: semua. Jika Anda memerlukan kuota yang lebih tinggi untuk jenis cakupan lain, hubungi dukungan pelanggan. Permintaan kenaikan kuota akan secara otomatis disetujui untuk nilai hingga 100.

Nama	Default	Dapat disesuaikan	Deskripsi
Subnet Target untuk aws:network: -region-connectivity route-table-disrupt-cross	Setiap Wilayah yang didukung: 6	Ya	Jumlah maksimum Subnet yang aws:network: route-table-disrupt-cross -region-connectivity dapat menargetkan saat Anda mengidentifikasi target menggunakan tag, per percobaan. Permintaan kenaikan kuota akan secara otomatis disetujui untuk nilai hingga 50.
Tugas Target untuk aws:ecs:stop-task	Setiap Wilayah yang didukung: 5	Ya	Jumlah maksimum Tugas yang aws:ecs:stop-task dapat menargetkan saat Anda mengidentifikasi target menggunakan tag, per percobaan. Permintaan kenaikan kuota akan secara otomatis disetujui untuk nilai hingga 500.
Tugas Target untuk aws:ecs: task-cpu-stress	Setiap Wilayah yang didukung: 5	Ya	Jumlah maksimum Tugas yang aws:ecs: task-cpu-stress dapat menargetkan ketika Anda mengidentifikasi target menggunakan tag/parameter, per percobaan. Permintaan kenaikan kuota akan secara otomatis disetujui untuk nilai hingga 50.

Nama	Default	Dapat disesuaikan	Deskripsi
Tugas Target untuk aws:ecs: task-io-stress	Setiap Wilayah yang didukung: 5	Ya	Jumlah maksimum Tugas yang aws:ecs: task-io-stress dapat menargetkan ketika Anda mengidentifikasi target menggunakan tag/parameter, per percobaan. Permintaan kenaikan kuota akan secara otomatis disetujui untuk nilai hingga 50.
Tugas Target untuk aws:ecs: task-kill-process	Setiap Wilayah yang didukung: 5	Ya	Jumlah maksimum Tugas yang aws:ecs: task-kill-process dapat menargetkan ketika Anda mengidentifikasi target menggunakan tag/parameter, per percobaan. Permintaan kenaikan kuota akan secara otomatis disetujui untuk nilai hingga 50.
Tugas Target untuk aws:ecs: task-network-blackhole-port	Setiap Wilayah yang didukung: 5	Ya	Jumlah maksimum Tugas yang aws:ecs: task-network-blackhole-port dapat menargetkan ketika Anda mengidentifikasi target menggunakan tag/parameter, per percobaan. Permintaan kenaikan kuota akan secara otomatis disetujui untuk nilai hingga 50.

Nama	Default	Dapat disesuaikan	Deskripsi
Tugas Target untuk aws:ecs: task-network-latency	Setiap Wilayah yang didukung: 5	Ya	Jumlah maksimum Tugas yang aws:ecs: task-network-latency dapat menargetkan ketika Anda mengidentifikasi target menggunakan tag/parameter, per percobaan. Permintaan kenaikan kuota akan secara otomatis disetujui untuk nilai hingga 50.
Tugas Target untuk aws:ecs: task-network-packet-loss	Setiap Wilayah yang didukung: 5	Ya	Jumlah maksimum Tugas yang aws:ecs: task-network-packet-loss dapat menargetkan ketika Anda mengidentifikasi target menggunakan tag/parameter, per percobaan. Permintaan kenaikan kuota akan secara otomatis disetujui untuk nilai hingga 50.

Nama	Default	Dapat disesuaikan	Deskripsi
Target TransitGateways untuk aws:network: -region-connectivity transit-gateway-disrupt-cross	Setiap Wilayah yang didukung: 5	Ya	Jumlah maksimum Transit Gateway yang aws:network: transit-gateway-disrupt-cross -region-connectivity dapat menargetkan saat Anda mengidentifikasi target menggunakan tag, per percobaan. Permintaan kenaikan kuota akan secara otomatis disetujui untuk nilai hingga 50.
Volume Target untuk aws:ebs: pause-volume-io	ca-central-1:160 eu-central-1:160 Masing-masing Wilayah yang didukung lainnya: 5	Ya	Jumlah maksimum Volume yang aws:ebs: pause-volume-io dapat menargetkan ketika Anda mengidentifikasi target menggunakan tag, per percobaan. Permintaan kenaikan kuota akan secara otomatis disetujui untuk nilai hingga 160.
Konfigurasi akun target per templat percobaan	Setiap Wilayah yang didukung: 10	Ya	Jumlah maksimum konfigurasi akun target yang dapat Anda buat untuk templat eksperimen di akun ini di Wilayah saat ini. Permintaan kenaikan kuota akan secara otomatis disetujui untuk nilai hingga 40.

Nama	Default	Dapat disesuaikan	Deskripsi
Fungsi target untuk aws:lambda: action. invocation-add-delay	Setiap Wilayah yang didukung: 5	Ya	Jumlah maksimum fungsi Lambda yang aws:lambda: invocation-add-delay dapat menargetkan saat Anda mengidentifikasi target menggunakan tag, per percobaan. Permintaan kenaikan kuota akan secara otomatis disetujui untuk nilai hingga 140.
Fungsi target untuk tindakan aws:lambda:invocation-error.	Setiap Wilayah yang didukung: 5	Ya	Jumlah maksimum fungsi Lambda yang aws:lambda:invocation-error dapat menargetkan saat Anda mengidentifikasi target menggunakan tag, per percobaan. Permintaan kenaikan kuota akan secara otomatis disetujui untuk nilai hingga 140.
Fungsi target untuk aws:lambda: action. invocation-http-integration-response	Setiap Wilayah yang didukung: 5	Ya	Jumlah maksimum fungsi Lambda yang aws:lambda: invocation-http-integration-response dapat menargetkan saat Anda mengidentifikasi target menggunakan tag, per percobaan. Permintaan kenaikan kuota akan secara otomatis disetujui untuk nilai hingga 140.

Nama	Default	Dapat disesuaikan	Deskripsi
Tabel target untuk aws:dynamodb: action global-table-pause-replication	Setiap Wilayah yang didukung: 5	Ya	Jumlah maksimum tabel global yang aws:dynamodb: global-table-pause-replication dapat menargetkan, per percobaan. Permintaan kenaikan kuota akan secara otomatis disetujui untuk nilai hingga 60.

Penggunaan AWS FIS Anda tunduk pada batasan tambahan berikut:

Nama	Batasan
Batas harian pada target ReplicationGroups untuk aws:elasticache: replicationgroup-interrupt-az-power	Batasnya adalah 20 ReplicationGroups ditargetkan per akun per wilayah per hari. Anda dapat meminta peningkatan dengan membuat kasus dukungan di AWS Support Center Console.

Riwayat dokumen

Tabel berikut menjelaskan pembaruan dokumentasi penting dalam Panduan Pengguna Layanan Injeksi AWS Kesalahan.

Perubahan	Deskripsi	Tanggal
Dukungan ARC di AWS FIS	Anda dapat menggunakan AWS FIS untuk menguji bagaimana ARC zonal autoshift secara otomatis memulihkan aplikasi Anda selama gangguan daya AZ.	Maret 26, 2025
Konfigurasi laporan eksperimen baru	Anda sekarang dapat mengaktifkan AWS FIS untuk menghasilkan laporan untuk eksperimen yang merangkum tindakan eksperimen dan tanggapan dari CloudWatch dasbor.	November 12, 2024
Tindakan Lambda baru	Anda sekarang dapat menggunakan tindakan aws:lambda:function untuk menyuntikkan kesalahan ke dalam pemanggilan fungsi Lambda Anda.	Oktober 31, 2024
Fitur tuas pengaman baru	AWS FIS sekarang mendukung tuas pengaman yang memungkinkan Anda menghentikan semua eksperimen yang berjalan dengan cepat dan mencegah eksperimen baru dimulai.	September 3, 2024

Bab Pemecahan Masalah Baru	AWS FIS menambahkan panduan pemecahan masalah yang mencakup kode kesalahan dan konteks untuk eksperimen yang gagal.	Agustus 13, 2024
Aksi baru	Anda sekarang dapat menggunakan `aws:dynamodb:global-table-pause-replication` tindakan untuk menjeda replikasi data antara tabel global target dan tabel replika. `aws:dynamodb:encrypted-global-table-pause-replication` Tindakan tidak akan lagi didukung.	April 24, 2024
Opsi percobaan mode tindakan baru	Anda dapat mengatur mode tindakan `skip-all` untuk menghasilkan pratinjau target sebelum menjalankan eksperimen.	Maret 13, 2024
AWS pembaruan kebijakan terkelola	AWS FIS memperbarui kebijakan terkelola yang ada.	Januari 25, 2024
Skenario dan tindakan baru	Anda sekarang dapat menggunakan skenario AWS FIS Lintas Wilayah: Konektivitas dan Ketersediaan AZ: Gangguan Daya.	30 November 2023

Aksi baru	Anda sekarang dapat menggunakan aws:ec2:asg-insufficient-instance-capacity-error aksinya.	30 November 2023
Aksi baru	Anda sekarang dapat menggunakan aws:ec2:api-insufficient-instance-capacity-error aksinya.	30 November 2023
Aksi baru	Anda sekarang dapat menggunakan aws:network:route-table-disrupt-cross-region-connectivity aksinya.	30 November 2023
Aksi baru	Anda sekarang dapat menggunakan aws:network:transit-gateway-disrupt-cross-region-connectivity aksinya.	30 November 2023
Aksi baru	Anda sekarang dapat menggunakan aws:dynamodb:encrypted-global-table-pause-replication aksinya.	30 November 2023
Aksi baru	Anda sekarang dapat menggunakan aws:s3:bucket-pause-replication aksinya.	30 November 2023
Aksi baru	Anda sekarang dapat menggunakan aws:elasticache:interrupt-cluster-az-power aksinya.	30 November 2023

Opsi percobaan baru	Anda sekarang dapat menggunakan opsi eksperimen AWS FIS untuk penargetan akun dan resolusi target kosong.	27 November 2023
Perubahan nama AWS FIS	Nama layanan yang diperbarui ke Layanan Injeksi AWS Kesalahan.	15 November 2023
AWS pembaruan kebijakan terkelola	AWS FIS memperbarui kebijakan terkelola yang ada.	13 November 2023
Pustaka skenario baru	Anda sekarang dapat menggunakan fitur AWS pustaka skenario FIS.	7 November 2023
Penjadwal percobaan baru	Anda sekarang dapat menggunakan fitur penjadwal eksperimen AWS FIS.	7 November 2023
AWS pembaruan kebijakan terkelola	AWS FIS memperbarui kebijakan terkelola yang ada.	Juni 2, 2023
Tindakan baru	Anda dapat menggunakan yang baru aws:ecs:task dan aws:eks:pod tindakan.	1 Juni 2023
AWS pembaruan kebijakan terkelola	AWS FIS memperbarui kebijakan terkelola yang ada.	1 Juni 2023
Dokumen SSM pra-konfigurasi baru	Anda dapat menggunakan dokumen SSM pra-konfigurasi berikut: AWSFIS-Run -Disk-Fill.	28 April 2023

Aksi baru	Anda dapat menggunakan aws:ebs:pause-volume-io tindakan untuk menjeda I/O antara volume target dan instance yang dilampirkan.	27 Januari 2023
Aksi baru	Anda dapat menggunakan aws:network:disrupt-connectivity tindakan untuk menolak jenis lalu lintas tertentu ke subnet target.	26 Oktober 2022
Aksi baru	Anda dapat menggunakan aws:eks:inject-kubernetes-custom-resource tindakan untuk menjalankan eksperimen ChaosMesh atau Lakmus pada satu cluster target.	Juli 7, 2022
Pencatatan percobaan	Anda dapat mengonfigurasi templat eksperimen untuk mengirim log aktivitas eksperimen ke CloudWatch Log atau ke bucket S3.	28 Februari 2022
Pemberitahuan baru	Ketika keadaan eksperimen berubah, AWS FIS memancarkan pemberitahuan. Pemberitahuan ini tersedia sebagai acara melalui Amazon EventBridge.	Februari 24, 2022
Aksi baru	Anda dapat menggunakan aws:ecs:stop-task tindakan untuk menghentikan tugas yang ditentukan.	Februari 9, 2022

Aksi baru	Anda dapat menggunak an aws:cloudwatch:assert-alarm-state tindakan untuk memverifikasi bahwa alarm yang ditentukan berada di salah satu status alarm yang ditentukan.	November 5, 2021
Dokumen SSM pra-konfigurasi baru	Anda dapat menggunakan dokumen SSM pra-konfigurasi berikut: AWSFIS-Run -IO-Stress, -Network-Blackhold -Port, -Network-Latency-S ources, -Network-Packet-Loss, dan AWSFIS-Run -Network-Packet-Loss-Source s. AWSFIS-Run AWSFIS-Run AWSFIS-Run	4 November 2021
Aksi baru	Anda dapat menggunakan aws:ec2:send-spot-instance-interruptions tindakan untuk mengirim pemberitahuan interupsi Instans Spot untuk menargetkan Instans Spot dan kemudian menginterupsi Instans Spot target.	20 Oktober 2021
Aksi baru	Anda dapat menggunakan aws:ssm:start-automation-execution tindakan untuk memulai eksekusi runbook Otomasi.	September 17, 2021
Rilis awal	Rilis awal Panduan Pengguna Layanan Injeksi AWS Kesalahan.	15 Maret 2021

Terjemahan disediakan oleh mesin penerjemah. Jika konten terjemahan yang diberikan bertentangan dengan versi bahasa Inggris aslinya, utamakan versi bahasa Inggris.