

Panduan Pengguna

Alat AWS untuk PowerShell (versi 4)



Alat AWS untuk PowerShell (versi 4): Panduan Pengguna

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

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

Table of Contents

.....	x
Apa yang dimaksud dengan Alat AWS untuk PowerShell?	1
Pemeliharaan dan dukungan untuk versi utama SDK	2
AWS.Tools	2
AWSPowerCangking. NetCore	3
AWSPowerCangking	3
Cara menggunakan panduan ini	4
Topik tambahan di bagian ini	4
Riwayat revisi	4
Apa yang baru	5
Penginstalan	7
Menginstal pada Windows	7
Prasyarat	8
Instal AWS.Tools	8
Instal AWSPower Shell. NetCore	11
Instal AWSPower Shell	12
Aktifkan Eksekusi Skrip	13
Penentuan versi	15
Memperbarui Alat AWS untuk PowerShell	17
Memasang pada Linux atau macOS	18
Gambaran Umum Pengaturan	18
Prasyarat	8
Pasang AWS.Tools	20
Instal AWSPower Shell. NetCore	22
Eksekusi Skrip	13
Mengkonfigurasi Konsol PowerShell	24
Inisialisasi Sesi Anda PowerShell	25
Penentuan versi	15
Memperbarui Alat AWS untuk PowerShell di Linux atau macOS	26
Informasi Terkait	27
Migrasi dari Alat AWS untuk PowerShell Versi 3.3 ke Versi 4	27
Versi AWS.Tools Baru yang Sepenuhnya Termodulasi	28
Cmdlet Get-AWSService baru	28
Parameter -Select baru untuk Mengendalikan Obyek yang Dikembalikan oleh Cmdlet	29

Pembatasan Lebih Konsisten dari Jumlah Item dalam Output	31
Lebih Mudah Menggunakan Parameter Pengaliran	31
Memperluas Alur dengan Nama Properti	32
Parameter Umum Statis	32
AWS.Tools Menyatakan dan Menerapkan Parameter Wajib	33
Semua Parameter Dapat Dibatalkan	33
Menghapus Fitur yang Sudah Tidak Lagi Digunakan	33
Memulai	35
Konfigurasi otentikasi alat	35
Aktifkan dan konfigurasi Pusat Identitas IAM	36
Konfigurasi Alat PowerShell untuk menggunakan IAM Identity Center.	36
Memulai sesi portal AWS akses	38
Contoh	39
Informasi tambahan	39
Gunakan AWS CLI	40
Tentukan AWS Wilayah	44
Mencantumkan Titik Akhir Kustom atau Tidak Standar	46
Informasi tambahan	46
Konfigurasi identitas federasi	46
Prasyarat	47
Bagaimana Pengguna Federasi Identitas Mendapat Akses Federasi ke Layanan AWS	
APIs	47
Cara Kerja SAML Support di Alat AWS untuk PowerShell	49
Cara Menggunakan Cmdlet Konfigurasi PowerShell SAFL	50
Bacaan Tambahan	55
Penemuan dan alias Cmdlet	55
Penemuan Cmdlet	55
Penamaan dan Alias Cmdlet	62
Pipelining, output, dan iterasi	66
Pemipaan pipa	66
Keluaran cmdlet	66
Iterasi melalui data halaman	68
Resolusi kredensi dan profil	70
Urutan Pencarian Kredensial	70
Pengguna dan peran	71
Pengguna dan set izin	71

Peran layanan	72
Menggunakan kredensial warisan	73
Peringatan dan pedoman penting	73
AWS Kredensialnya	74
Kredensial Bersama	84
Fitur	90
Observabilitas	90
Bekerja dengan AWS layanan	94
PowerShell Pengkodean Penggabungan File	94
Objek yang Dikembalikan untuk PowerShell Alat	95
Amazon EC2	95
Amazon S3	95
AWS Lambda dan Alat AWS untuk PowerShell	96
Amazon SNS dan Amazon SQS	96
CloudWatch	96
Lihat Juga	96
Topik	96
Amazon S3 dan Alat untuk Windows PowerShell	97
Membuat Bucket Amazon S3, Memverifikasi Wilayahnya, dan Memilih Menghapusnya	98
Konfigurasi Bucket Amazon S3 sebagai Situs Web dan Aktifkan Pencatatan	99
Unggah Objek ke Bucket Amazon S3	99
Hapus Obyek dan Bucket Amazon S3	102
Unggah Konten Teks Selaras ke Amazon S3	103
Amazon EC2 dan Alat untuk Windows PowerShell	104
Buat Pasangan Kunci	104
Buat Grup Keamanan	107
Temukan AMI	109
Luncurkan sebuah Instans	113
AWS Lambda dan Alat AWS untuk PowerShell	116
Prasyarat	8
Instal AWSLambda PSCore Modul	117
Lihat Juga	96
Amazon SQS, Amazon SNS dan Alat untuk Windows PowerShell	117
Membuat antrean Amazon SQS dan mendapatkan ARN antrean	117
Membuat topik Amazon SNS	118
Memberikan izin untuk topik SNS	118

Berlangganan antrean ke topik SNS	119
Verifikasi izin	119
Verifikasi hasil	120
CloudWatch dari AWS Tools for Windows PowerShell	121
Publikasikan Metrik Kustom ke CloudWatch Dasbor Anda	121
Lihat Juga	96
Menggunakan ClientConfig	122
Menggunakan ClientConfig parameter	122
Menggunakan properti yang tidak ditentukan	123
Menentukan Wilayah AWS	123
Contoh kode	125
ACM	127
Tindakan	127
Penskalaan Otomatis Aplikasi	132
Tindakan	127
AppStream 2.0	139
Tindakan	127
Aurora	165
Tindakan	127
Auto Scaling	166
Tindakan	127
AWS Budgets	201
Tindakan	127
AWS Cloud9	203
Tindakan	127
AWS CloudFormation	209
Tindakan	127
CloudFront	221
Tindakan	127
CloudTrail	229
Tindakan	127
CloudWatch	234
Tindakan	127
CodeCommit	238
Tindakan	127
CodeDeploy	243

Tindakan	127
CodePipeline	262
Tindakan	127
Identitas Amazon Cognito	280
Tindakan	127
AWS Config	284
Tindakan	127
Device Farm	303
Tindakan	127
AWS Directory Service	304
Tindakan	127
AWS DMS	329
Tindakan	127
DynamoDB	330
Tindakan	127
Amazon EC2	345
Tindakan	127
Amazon ECR	476
Tindakan	127
Amazon ECS	477
Tindakan	127
Amazon EFS	483
Tindakan	127
Amazon EKS	490
Tindakan	127
Elastic Load Balancing - Versi 1	503
Tindakan	127
Elastic Load Balancing - Versi 2	523
Tindakan	127
Amazon FSx	546
Tindakan	127
AWS Glue	554
Tindakan	127
AWS Health	556
Tindakan	127
IAM	557

Tindakan	127
Kinesis	630
Tindakan	127
Lambda	634
Tindakan	127
Amazon ML	647
Tindakan	127
Macie	653
Tindakan	127
AWS OpsWorks	654
Tindakan	127
Daftar Harga AWS	655
Tindakan	127
Resource Groups	658
Tindakan	127
API Penandaan Grup Sumber Daya	665
Tindakan	127
Route 53	670
Tindakan	127
Amazon S3	685
Tindakan	127
S3 Glacier	721
Tindakan	127
Security Hub	724
Tindakan	127
Amazon SES	726
Tindakan	127
Amazon SES API v2	727
Tindakan	127
Amazon SNS	728
Tindakan	127
Amazon SQS	730
Tindakan	127
AWS STS	742
Tindakan	127
Dukungan	746

Tindakan	127
Systems Manager	752
Tindakan	127
Amazon Translate	825
Tindakan	127
AWS WAFV2	826
Tindakan	127
WorkSpaces	827
Tindakan	127
Keamanan	843
Perindungan data	843
Enkripsi data	844
Identity and Access Management	845
Audiens	845
Mengautentikasi dengan identitas	846
Mengelola akses menggunakan kebijakan	850
Bagaimana Layanan AWS bekerja dengan IAM	853
Memecahkan masalah AWS identitas dan akses	853
Validasi Kepatuhan	855
Menegakkan versi TLS minimum	856
Pertimbangan keamanan tambahan	856
Pencatatan informasi sensitif	856
Referensi Cmdlet	858
Riwayat dokumen	859
.....	cccclxvi

Alat AWS untuk PowerShell Versi 5 (V5) dalam pratinjau. Untuk melihat konten V5, yang dapat berubah, dan mencoba versi baru, lihat [panduan pengguna versi 5 \(pratinjau\)](#). Untuk informasi spesifik tentang melanggar perubahan dan migrasi ke V5, lihat [topik migrasi dalam panduan tersebut](#).

Apa yang dimaksud dengan Alat AWS untuk PowerShell?

Alat AWS untuk PowerShell Ini adalah satu set PowerShell modul yang dibangun di atas fungsionalitas yang diekspos oleh AWS SDK untuk .NET. Alat AWS untuk PowerShell Memungkinkan Anda untuk skrip operasi pada AWS sumber daya Anda dari baris PowerShell perintah.

Cmdlet memberikan PowerShell pengalaman idiomatik untuk menentukan parameter dan menangani hasil meskipun mereka diimplementasikan menggunakan berbagai kueri HTTP layanan. AWS APIs Misalnya, cmdlet untuk PowerShell pipa Alat AWS untuk PowerShell pendukung—yaitu, Anda dapat menyalurkan PowerShell objek masuk dan keluar dari cmdlet.

Fleksibel dalam cara mereka memungkinkan Anda untuk menangani kredensial, termasuk dukungan untuk infrastruktur AWS Identity and Access Management (IAM). Alat AWS untuk PowerShell Anda dapat menggunakan alat-alat dengan kredensial pengguna IAM, token keamanan sementara, dan peran IAM.

Alat AWS untuk PowerShell Mendukung set layanan dan AWS Wilayah yang sama yang didukung oleh SDK. Anda dapat menginstal Alat AWS untuk PowerShell pada komputer yang menjalankan sistem operasi Windows, Linux, atau macOS.

Note

Alat AWS untuk PowerShell versi 4 (V4) adalah pembaruan yang kompatibel ke versi 3.3. Alat AWS untuk PowerShell Versi ini menambahkan perbaikan yang signifikan dengan tetap mempertahankan perilaku cmdlet yang ada. Skrip Anda yang ada harus terus berfungsi setelah memutakhirkan ke V4, tetapi kami menyarankan Anda mengujinya secara menyeluruh sebelum memutakhirkan. Untuk informasi selengkapnya tentang perubahan di V4, lihat [Migrasi dari Alat AWS untuk PowerShell Versi 3.3 ke Versi 4](#).

Alat AWS untuk PowerShell Tersedia sebagai tiga paket berbeda berikut:

- [AWS.Tools](#)
- [AWSPowerCangKang.NetCore](#)
- [AWSPowerCangKang](#)

Pemeliharaan dan dukungan untuk versi utama SDK

Untuk informasi tentang pemeliharaan dan dukungan untuk versi utama SDK dan dependensi yang mendasarinya, lihat berikut ini di Panduan Referensi [Alat AWS SDKs dan Alat](#) berikut:

- [AWS SDKs dan kebijakan pemeliharaan alat](#)
- [AWS SDKs dan matriks dukungan versi alat](#)

AWS.Tools- Sebuah versi termodulasi dari Alat AWS untuk PowerShell



Versi ini Alat AWS untuk PowerShell adalah versi yang direkomendasikan untuk komputer apa pun yang berjalan PowerShell di lingkungan produksi. Karena termodulasi, Anda hanya perlu mengunduh dan memuat modul-modul untuk layanan yang ingin Anda gunakan. Ini mengurangi waktu pengunduhan, penggunaan memori, dan, dalam banyak kasus, memungkinkan pengimporan otomatis `AWS.Tools` cmdlet tanpa perlu menelepon secara manual terlebih dahulu. `Import-Module`

Ini adalah versi terbaru Alat AWS untuk PowerShell dan berjalan pada semua sistem operasi yang didukung, termasuk Windows, Linux, dan macOS. Paket ini menyediakan satu modul instalasi `AWS.Tools.Installer`, satu modul umum `AWS.Tools.Common`, dan satu modul untuk setiap AWS layanan, misalnya, `AWS.Tools.EC2`, `AWS.Tools.IdentityManagement`, `AWS.Tools.S3`, dan sebagainya.

`AWS.Tools.Installer` Modul ini menyediakan cmdlet yang memungkinkan Anda menginstal, memperbarui, dan menghapus modul untuk setiap layanan. AWS Cmdlet dalam modul ini secara otomatis memastikan bahwa Anda memiliki semua modul tergantung yang diperlukan untuk mendukung modul yang ingin Anda gunakan.

Modul `AWS.Tools.Common` menyediakan cmdlet untuk konfigurasi dan otentikasi yang tidak tergantung jenis layanan. Untuk menggunakan cmdlet untuk suatu AWS layanan, Anda cukup menjalankan perintah. PowerShell secara otomatis mengimpor `AWS.Tools.Common` modul dan

modul untuk AWS layanan yang cmdletnya ingin Anda jalankan. Modul ini secara otomatis diinstal jika Anda menggunakan modul `AWS.Tools.Installer` untuk menginstal modul layanan.

Anda dapat menginstal versi ini Alat AWS untuk PowerShell di komputer yang sedang berjalan:

- PowerShell Core 6.0 atau yang lebih baru di Windows, Linux, atau macOS.
- Windows PowerShell 5.1 atau yang lebih baru di Windows dengan .NET Framework 4.7.2 atau yang lebih baru.

Dalam panduan ini, ketika kami perlu menyebutkan versi ini saja, kami menyebutnya dengan nama modulnya: `AWS.Tools`.

AWSPowerCangkang. NetCore - Sebuah versi modul tunggal dari Alat AWS untuk PowerShell

PowerShell Gallery **AWSPowerShell.NetCore**

ZIP Archive **AWSPowerShell.NetCore**

Versi ini terdiri dari satu modul besar yang berisi dukungan untuk semua AWS layanan. Sebelum Anda dapat menggunakan modul ini, Anda harus mengimpornya secara manual.

Anda dapat menginstal versi ini Alat AWS untuk PowerShell di komputer yang sedang berjalan:

- PowerShell Core 6.0 atau yang lebih baru di Windows, Linux, atau macOS.
- Windows PowerShell 3.0 atau yang lebih baru pada Windows dengan .NET Framework 4.7.2 atau yang lebih baru.

Sepanjang panduan ini, ketika kita perlu menentukan versi ini saja, kita merujuknya dengan nama modulnya: `AWSPowerShell. NetCore`.

AWSPowerShell - Versi modul tunggal untuk Windows PowerShell

PowerShell Gallery **AWSPowerShell**

ZIP Archive **AWSPowerShell**

Versi ini Alat AWS untuk PowerShell kompatibel dengan dan dapat diinstal hanya pada komputer Windows yang menjalankan Windows PowerShell versi 2.0 hingga 5.1. Ini tidak kompatibel dengan PowerShell Core 6.0 atau yang lebih baru, atau sistem operasi lainnya (Linux atau macOS). Versi ini terdiri dari satu modul besar yang berisi dukungan untuk semua AWS layanan.

Sepanjang panduan ini, ketika kita perlu menentukan versi ini saja, kita merujuknya dengan nama modulnya: AWSPowerShell.

Cara menggunakan panduan ini

Panduan ini dibagi menjadi beberapa bagian utama berikut.

[Instalasi Alat AWS untuk PowerShell](#)

Bagian ini menjelaskan cara menginstal Alat AWS untuk PowerShell. Ini termasuk cara mendaftar AWS jika Anda belum memiliki akun, dan cara membuat pengguna IAM yang dapat Anda gunakan untuk menjalankan cmdlet.

[Memulai dengan AWS Tools for Windows PowerShell](#)

Bagian ini menjelaskan dasar-dasar penggunaan Alat AWS untuk PowerShell, seperti menentukan kredensial dan AWS Wilayah, menemukan cmdlet untuk layanan tertentu, dan menggunakan alias untuk cmdlet.

[Bekerja dengan AWS layanan di Alat AWS untuk PowerShell](#)

Bagian ini mencakup informasi tentang penggunaan Alat AWS untuk PowerShell untuk melakukan beberapa AWS tugas yang paling umum.

Topik tambahan di bagian ini

- [Riwayat revisi](#)
- [Apa yang baru di Alat AWS untuk PowerShell](#)

Riwayat revisi

Untuk mengetahui apa yang telah berubah dalam berbagai rilis, lihat yang berikut ini:

- [Log perubahan](#)

- [Apa yang baru di Alat AWS untuk PowerShell](#)
- [Riwayat dokumen](#)

Apa yang baru di Alat AWS untuk PowerShell

Untuk informasi tingkat tinggi tentang perkembangan baru yang terkait dengan Alat AWS untuk PowerShell, lihat halaman produk di <https://aws.amazon.com/powershell/> dan [log perubahan](#).

Berikut ini adalah apa yang baru di Tools for PowerShell.

28 Mei 2025: Panduan pengguna tersedia untuk pratinjau V5 Alat AWS untuk PowerShell

[Panduan pengguna](#) untuk versi 5 (pratinjau) dari Alat AWS untuk PowerShell telah diterbitkan.

10 Februari 2025: Rilis GA untuk observabilitas

Observabilitas adalah sejauh mana keadaan sistem saat ini dapat disimpulkan dari data yang dipancarkannya. Observabilitas telah ditambahkan ke Alat untuk PowerShell, termasuk implementasi penyedia telemetri. Untuk informasi lebih lanjut, lihat [Observabilitas](#) di panduan ini dan posting blog [Mengumumkan ketersediaan umum OpenTelemetry pustaka AWS .NET](#).

15 Januari 2025: Perilaku default baru untuk perlindungan integritas

Dimulai dengan versi 4.1.737 dari Alat AWS untuk PowerShell, alat menyediakan perlindungan integritas default dengan secara otomatis menghitung checksum untuk CRC32 upload. Untuk informasi lebih lanjut, lihat pengumuman GitHub di <https://github.com/aws/aws-tools-for-powershell/issues/370>. Alat ini juga menyediakan pengaturan global untuk perlindungan integritas data yang dapat Anda atur secara eksternal, yang dapat Anda baca di [Perlindungan Integritas Data di Panduan Referensi](#) Alat [AWS SDKs dan](#) Alat.

18 November 2024: Pratinjau 1 rilis untuk versi 5

Note

Ini adalah dokumentasi prarilis untuk fitur dalam rilis pratinjau. Dokumentasi ini dapat berubah.

Alat AWS untuk PowerShell Ini sedang dalam proses diperbarui ke versi 5 dan akan memiliki perubahan yang melanggar. Pratinjau 1 versi 5 telah dirilis. Untuk informasi lebih lanjut tentang

pratinjau ini dan untuk mencobanya, lihat posting blog [Pratinjau 1 dari Alat AWS untuk PowerShell V5](#) dan masalah [Pelacak Pengembangan V5](#) di. GitHub

Instalasi Alat AWS untuk PowerShell

Agar berhasil menginstal dan menggunakan Alat AWS untuk PowerShell cmdlet, lihat langkah-langkah dalam topik berikut.

Topik

- [Menginstal Alat AWS untuk PowerShell pada Windows](#)
- [Menginstal Alat AWS untuk PowerShell di Linux atau macOS](#)
- [Migrasi dari Alat AWS untuk PowerShell Versi 3.3 ke Versi 4](#)

Menginstal Alat AWS untuk PowerShell pada Windows

Komputer berbasis Windows dapat menjalankan salah satu opsi Alat AWS untuk PowerShell paket:

- [AWS.Tools](#) - Versi termodulasi dari. Alat AWS untuk PowerShell Setiap AWS layanan didukung oleh modul kecil individualnya sendiri, dengan modul dukungan bersama `AWS.Tools.Common` dan `AWS.Tools.Installer`.
- [AWSPowerCangkang.NetCore](#) - Versi tunggal, modul besar dari. Alat AWS untuk PowerShell Semua AWS layanan didukung oleh modul tunggal dan besar ini.

Note

Ketahui bahwa modul tunggal mungkin terlalu besar untuk digunakan dengan [AWS Lambda](#) fungsi. Sebagai gantinya, gunakan versi termodulasi yang ditunjukkan di atas.

- [AWSPowerShell](#) - Versi lama khusus Windows, tunggal, modul besar dari. Alat AWS untuk PowerShell Semua AWS layanan didukung oleh modul tunggal dan besar ini.

Paket yang Anda pilih tergantung pada rilis dan edisi Windows yang sedang Anda jalankan.

Note

Alat AWS untuk PowerShell Ini diinstal secara default pada semua Gambar Mesin Amazon berbasis Windows (AMIs). Opsi yang diinstal tergantung pada AMI. Banyak yang AMIs memiliki modul `AWSPowerShell`, tetapi beberapa mungkin memiliki opsi yang berbeda.

Misalnya, Amazon EC2 AMIs untuk Windows Server 2025 menggunakan `AWS.Tools` opsi modular.

Menyiapkan Alat AWS untuk PowerShell melibatkan tugas-tugas tingkat tinggi berikut, dijelaskan secara rinci dalam topik ini.

1. Instal opsi Alat AWS untuk PowerShell paket yang sesuai untuk lingkungan Anda.
2. Verifikasi bahwa eksekusi skrip diaktifkan dengan menjalankan cmdlet `Get-ExecutionPolicy`.
3. Impor Alat AWS untuk PowerShell modul ke PowerShell sesi Anda.

Prasyarat

Versi yang lebih baru PowerShell, termasuk PowerShell Core, tersedia sebagai unduhan dari Microsoft di [Menginstal berbagai versi PowerShell](#) di situs Web Microsoft.

Instal **AWS.Tools** di Windows

Anda dapat menginstal versi termodulasi Alat AWS untuk PowerShell pada komputer yang menjalankan Windows dengan Windows PowerShell 5.1, atau PowerShell Core 6.0 atau yang lebih baru. Untuk informasi tentang cara menginstal PowerShell Core, lihat [Menginstal berbagai versi PowerShell](#) di situs Web Microsoft.

Anda dapat memasang `AWS.Tools` dengan memilih satu dari tiga cara:

- Menggunakan cmdlet di modul `AWS.Tools.Installer`. Modul ini menyederhanakan instalasi dan pembaruan `AWS.Tools` modul lainnya. `AWS.Tools.Installer` membutuhkan `PowerShellGet`, dan secara otomatis mengunduh dan menginstal versi yang diperbarui. `AWS.Tools.Installer` secara otomatis membuat versi modul Anda tetap sinkron. Saat Anda menginstal atau memperbarui ke versi yang lebih baru dari satu modul, cmdlet secara `AWS.Tools.Installer` otomatis memperbarui semua `AWS.Tools` modul Anda yang lain ke versi yang sama.

Metode ini dijelaskan dalam prosedur berikut.

- Mengunduh modul dari [AWS.Tools.zip](#) dan mengekstraknya di salah satu folder modul. Anda dapat menemukan folder modul Anda dengan menampilkan nilai variabel `PSModulePath` lingkungan.

⚠ Warning

Setelah mengunduh file ZIP dan sebelum mengekstraksi konten, Anda mungkin perlu membuka blokir. Ini biasanya dilakukan dengan membuka properti file, melihat tab Umum, dan memilih kotak centang Buka Blokir jika ada.

Jika file ZIP perlu dibuka blokir tetapi Anda tidak melakukannya, Anda mungkin menerima kesalahan yang mirip dengan yang berikut: "Modul Import: Tidak dapat memuat file atau perakitan".

- Menginstal setiap modul layanan dari PowerShell Galeri menggunakan `Install-Module` cmdlet.

Untuk menginstal **AWS.Tools** pada Windows menggunakan **AWS.Tools.Installer** modul

1. Mulai PowerShell sesi.

i Note

Kami menyarankan agar Anda tidak menjalankan PowerShell sebagai administrator dengan izin yang ditinggikan kecuali jika diperlukan oleh tugas yang ada. Hal ini karena potensi risiko keamanan dan tidak sesuai dengan prinsip batasan akses yang paling rendah.

2. Untuk menginstal paket **AWS.Tools** termodulasi, jalankan perintah berikut.

```
PS > Install-Module -Name AWS.Tools.Installer
```

```
Untrusted repository
```

```
You are installing the modules from an untrusted repository. If you trust this repository, change its InstallationPolicy value by running the Set-PSRepository cmdlet. Are you sure
```

```
you want to install the modules from 'PSGallery'?
```

```
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "N"): y
```

Jika Anda diberi tahu bahwa repositori "tidak terpercaya", Anda akan ditanya apakah Anda tetap ingin menginstalnya. Masukkan **y** untuk memungkinkan PowerShell untuk menginstal modul. Untuk menghindari prompt dan menginstal modul tanpa mempercayai repositori, Anda dapat menjalankan perintah dengan parameter `-Force`.

```
PS > Install-Module -Name AWS.Tools.Installer -Force
```

3. Anda sekarang dapat menginstal modul untuk setiap AWS layanan yang ingin Anda gunakan dengan menggunakan `Install-AWSToolsModule` cmdlet. Misalnya, perintah berikut menginstal modul Amazon EC2 dan Amazon S3. Perintah ini juga menginstal setiap modul tergantung yang diperlukan untuk modul tertentu yang akan dikerjakan. Misalnya, saat Anda menginstal modul layanan `AWS.Tools`, maka secara otomatis akan menginstal `AWS.Tools.Common`. Ini adalah modul bersama yang dibutuhkan oleh semua modul AWS layanan. Tindakan ini juga akan menghapus versi modul yang lebih lama, dan memperbarui modul lain ke versi yang sama barunya.

```
PS > Install-AWSToolsModule AWS.Tools.EC2,AWS.Tools.S3 -CleanUp
Confirm
Are you sure you want to perform this action?
Performing the operation "Install-AWSToolsModule" on target "AWS Tools version 4.0.0.0".
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "Y"):
```



```
Installing module AWS.Tools.Common version 4.0.0.0
Installing module AWS.Tools.EC2 version 4.0.0.0
Installing module AWS.Tools.Glacier version 4.0.0.0
Installing module AWS.Tools.S3 version 4.0.0.0
```



```
Uninstalling AWS.Tools version 3.3.618.0
Uninstalling module AWS.Tools.Glacier
Uninstalling module AWS.Tools.S3
Uninstalling module AWS.Tools.SimpleNotificationService
Uninstalling module AWS.Tools.SQS
Uninstalling module AWS.Tools.Common
```

Note

`Install-AWSToolsModuleCmdlet` mengunduh semua modul yang diminta dari PSRepository nama PSGallery (<https://www.powershellgallery.com/>) dan menganggapnya sebagai sumber tepercaya. Gunakan perintah `Get-PSRepository -Name PSGallery` untuk informasi lebih lanjut tentang PSRepository ini.

Secara default, perintah sebelumnya menginstal modul ke dalam %USERPROFILE%\Documents\WindowsPowerShell\Modules folder. Untuk menginstal Alat AWS untuk PowerShell untuk semua pengguna komputer, Anda harus menjalankan perintah berikut dalam PowerShell sesi yang Anda mulai sebagai administrator. Misalnya, perintah berikut menginstal modul IAM ke %ProgramFiles%\WindowsPowerShell\Modules folder yang dapat diakses oleh semua pengguna.

```
PS > Install-AWSToolsModule AWS.Tools.IdentityManagement -Scope AllUsers
```

Untuk menginstal modul lain, jalankan perintah serupa dengan nama modul yang sesuai, seperti yang ditemukan di [PowerShell Galeri](#).

Instal AWSPower Shell. NetCore pada Windows

Anda dapat menginstal AWSPower Shell. NetCore pada komputer yang menjalankan Windows dengan PowerShell versi 3 hingga 5.1, atau PowerShell Core 6.0 atau yang lebih baru. Untuk informasi tentang cara menginstal PowerShell Core, lihat [Menginstal berbagai versi PowerShell](#) di PowerShell situs web Microsoft.

Anda dapat menginstal AWSPower Shell. NetCore Dalam salah satu dari dua cara

- Mengunduh modul dari [AWSPowerShell. NetCore.zip](#) dan mengekstraknya di salah satu direktori modul. Anda dapat menemukan direktori modul Anda dengan menampilkan nilai variabel PSModulePath lingkungan.

Warning

Setelah mengunduh file ZIP dan sebelum mengekstraksi konten, Anda mungkin perlu membuka blokir. Ini biasanya dilakukan dengan membuka properti file, melihat tab Umum, dan memilih kotak centang Buka Blokir jika ada.

Jika file ZIP perlu dibuka blokir tetapi Anda tidak melakukannya, Anda mungkin menerima kesalahan yang mirip dengan yang berikut: “Modul Import: Tidak dapat memuat file atau perakitan”.

- Menginstal dari PowerShell Galeri menggunakan Install-Module cmdlet, seperti yang dijelaskan dalam prosedur berikut.

Untuk menginstal AWSPower Shell. NetCore dari PowerShell Galeri menggunakan cmdlet Install-Module

Untuk menginstal AWSPower Shell. NetCore dari PowerShell Galeri, komputer Anda harus menjalankan PowerShell 5.0 atau lebih baru, atau berjalan [PowerShellGet](#) pada PowerShell 3 atau lebih baru. Jalankan perintah berikut.

```
PS > Install-Module -name AWSPowerShell1.NetCore
```

Jika Anda menjalankan PowerShell sebagai administrator, perintah sebelumnya akan diinstal Alat AWS untuk PowerShell untuk semua pengguna di komputer. Jika Anda menjalankan PowerShell sebagai pengguna standar tanpa izin administrator, perintah yang sama akan diinstal hanya Alat AWS untuk PowerShell untuk pengguna saat ini.

Untuk menginstal untuk hanya pengguna saat ini ketika pengguna tersebut memiliki izin administrator, jalankan perintah dengan parameter `-Scope CurrentUser` yang ditetapkan, sebagai berikut.

```
PS > Install-Module -name AWSPowerShell1.NetCore -Scope CurrentUser
```

Meskipun rilis PowerShell 3.0 dan yang lebih baru biasanya memuat modul ke PowerShell sesi Anda saat pertama kali menjalankan cmdlet di modul, Shell. AWSPower NetCore modul terlalu besar untuk mendukung fungsi ini. Anda harus memuat Shell secara eksplisit. AWSPower NetCore Modul inti ke PowerShell sesi Anda dengan menjalankan perintah berikut.

```
PS > Import-Module AWSPowerShell1.NetCore
```

Untuk memuat AWSPower Shell. NetCore modul ke PowerShell sesi secara otomatis, tambahkan perintah itu ke PowerShell profil Anda. Untuk informasi selengkapnya tentang mengedit PowerShell profil Anda, lihat [Tentang Profil](#) di PowerShell dokumentasi.

Instal AWSPower Shell di Windows PowerShell

Anda dapat menginstal AWS Tools for Windows PowerShell dalam salah satu dari dua cara:

- Mengunduh modul dari [AWSPowerShell.zip](#) dan mengekstraknya di salah satu direktori modul. Anda dapat menemukan direktori modul Anda dengan menampilkan nilai variabel `PSModulePath` lingkungan.

⚠ Warning

Setelah mengunduh file ZIP dan sebelum mengekstraksi konten, Anda mungkin perlu membuka blokir. Ini biasanya dilakukan dengan membuka properti file, melihat tab Umum, dan memilih kotak centang Buka Blokir jika ada.

Jika file ZIP perlu dibuka blokir tetapi Anda tidak melakukannya, Anda mungkin menerima kesalahan yang mirip dengan yang berikut: "Modul Import: Tidak dapat memuat file atau perakitan".

- Menginstal dari PowerShell Galeri menggunakan `Install-Module` cmdlet seperti yang dijelaskan dalam prosedur berikut.

Untuk menginstal AWSPower Shell dari PowerShell Galeri menggunakan cmdlet `Install-Module`

Anda dapat menginstal AWSPower Shell dari PowerShell Galeri jika Anda menjalankan PowerShell 5.0 atau yang lebih baru, atau telah diinstal [PowerShellGet](#) pada PowerShell 3 atau yang lebih baru. Anda dapat menginstal dan memperbarui AWSPower Shell dari [PowerShellGaleri](#) Microsoft dengan menjalankan perintah berikut.

```
PS > Install-Module -Name AWSPowerShell
```

Untuk memuat modul AWSPower Shell ke dalam PowerShell sesi secara otomatis, tambahkan `import-module` cmdlet sebelumnya ke profil Anda PowerShell . Untuk informasi selengkapnya tentang mengedit PowerShell profil Anda, lihat [Tentang Profil](#) di PowerShell dokumentasi.

i Note

Alat untuk Windows diinstal PowerShell secara default pada semua Gambar Mesin Amazon berbasis Windows ()AMIs.

Aktifkan Eksekusi Skrip

Untuk memuat Alat AWS untuk PowerShell modul, Anda harus mengaktifkan eksekusi PowerShell skrip. Untuk mengaktifkan eksekusi skrip, jalankan cmdlet `Set-ExecutionPolicy` untuk menetapkan kebijakan `RemoteSigned`. Untuk informasi selengkapnya, lihat [Tentang Kebijakan Eksekusi](#) di situs web Microsoft Technet.

Note

Ini adalah persyaratan hanya untuk komputer yang menjalankan Windows. Pembatasan keamanan ExecutionPolicy tidak ada pada sistem operasi lain.

Untuk mengaktifkan eksekusi skrip

1. Hak administrator diperlukan untuk menetapkan kebijakan eksekusi. Jika Anda tidak masuk sebagai pengguna dengan hak administrator, buka PowerShell sesi sebagai Administrator. Pilih Mulai, lalu pilih Semua Program. Pilih Aksesoris, lalu pilih Windows PowerShell. Klik kanan Windows PowerShell, dan pada menu konteks, pilih Jalankan sebagai administrator.
2. Di prompt perintah, masukkan perintah berikut.

```
PS > Set-ExecutionPolicy RemoteSigned
```

Note

Pada sistem 64-bit, Anda harus melakukan ini secara terpisah untuk versi 32-bit PowerShell, Windows PowerShell (x86).

Jika kebijakan eksekusi tidak disetel dengan benar, PowerShell menampilkan kesalahan berikut setiap kali Anda mencoba menjalankan skrip, seperti profil Anda.

```
File C:\Users\username\Documents\WindowsPowerShell\Microsoft.PowerShell_profile.ps1
cannot be loaded because the execution
of scripts is disabled on this system. Please see "get-help about_signing" for more
details.
At line:1 char:2
+ . <<<< 'C:\Users\username\Documents\WindowsPowerShell
\Microsoft.PowerShell_profile.ps1'
+ CategoryInfo          : NotSpecified: (:) [], PSSecurityException
+ FullyQualifiedErrorId : RuntimeException
```

PowerShell Penginstal Alat untuk Windows secara otomatis memperbarui [PSModulePath](#) untuk menyertakan lokasi direktori yang berisi AWSPowerShell modul.

Karena PSModulePath menyertakan lokasi direktori AWS modul, `Get-Module -ListAvailable` cmdlet menunjukkan modul.

```
PS > Get-Module -ListAvailable
```

ModuleType	Name	ExportedCommands
Manifest	AppLocker	{}
Manifest	BitsTransfer	{}
Manifest	PSDiagnostics	{}
Manifest	TroubleshootingPack	{}
Manifest	AWSPowerShell	{Update-EBApplicationVersion, Set-DPStatus, Remove-IAMGroupPol...

Penentuan versi

AWS merilis versi baru secara berkala untuk mendukung AWS layanan dan fitur baru. Untuk menentukan versi Tools yang telah Anda instal, jalankan [Get-AWSPowerShellVersion](#) cmdlet.

Misalnya:

```
PS > Get-AWSPowerShellVersion
```

```

AWS Tools for PowerShell
Version 4.1.675
Copyright 2012-2024 Amazon.com, Inc. or its affiliates. All Rights Reserved.

Amazon Web Services SDK for .NET
Core Runtime Version 3.7.400.33
Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.

Release notes: https://github.com/aws/aws-tools-for-powershell/blob/master/CHANGELOG.md

This software includes third party software subject to the following copyrights:
- Logging from log4net, Apache License
[http://logging.apache.org/log4net/license.html]
```

Anda juga dapat menambahkan `-ListServiceVersionInfo` parameter ke `AWSPowerShellVersion` perintah [Get-](#) untuk melihat daftar AWS layanan yang didukung dalam versi alat saat

ini. Jika Anda menggunakan pilihan `AWS.Tools.*` termodulasikan, hanya modul yang saat ini Anda telah impor yang akan ditampilkan.

Misalnya:

```
PS > Get-AWSPowerShellVersion -ListServiceVersionInfo
...

Service                                Noun Prefix Module Name                                SDK
-----
Assembly
Version
-----
-----
AWS IAM Access Analyzer                IAMAA      AWS.Tools.AccessAnalyzer
3.7.400.33
AWS Account                            ACCT       AWS.Tools.Account
3.7.400.33
AWS Certificate Manager Private... PCA        AWS.Tools.ACMPCA
3.7.400.34
AWS Amplify                            AMP        AWS.Tools.Amplify
3.7.401.28
Amplify Backend                        AMPB       AWS.Tools.AmplifyBackend
3.7.400.33
...
```

Untuk menentukan versi PowerShell yang Anda jalankan, masukkan `$PSVersionTable` untuk melihat isi [variabel otomatis](#) `$PSVersionTable`.

Misalnya:

```
PS > $PSVersionTable

Name                                Value
----                                -
PSVersion                           6.2.2
PSEdition                           Core
GitCommitId                         6.2.2
OS                                   Darwin 18.7.0 Darwin Kernel Version 18.7.0: Tue Aug 20
16:57:14 PDT 2019; root:xnu-4903.271.2~2/RELEASE_X86_64
Platform                            Unix
PSCompatibleVersions                {1.0, 2.0, 3.0, 4.0...}
```

PSRemotingProtocolVersion	2.3
SerializationVersion	1.1.0.1
WSManStackVersion	3.0

Memperbarui Alat AWS untuk PowerShell pada Windows

Secara berkala, saat versi terbaru dirilis, Anda harus memperbarui versi yang Anda jalankan secara lokal. Alat AWS untuk PowerShell

Perbarui modul termodulasi **AWS.Tools**

Untuk memperbarui **AWS.Tools** modul Anda ke versi terbaru, jalankan perintah berikut:

```
PS > Update-AWSToolsModule -CleanUp
```

Perintah ini memperbarui semua modul **AWS.Tools** yang saat ini diinstal dan, setelah berhasil melakukan pembaruan, menghapus versi lain yang terpasang.

Note

Update-AWSToolsModuleCmdlet mengunduh semua modul dari PSRepository nama PSGallery (<https://www.powershellgallery.com/>) dan menganggapnya sebagai sumber tepercaya. Gunakan perintah: `Get-PSRepository -Name PSGallery` untuk informasi lebih lanjut tentang PSRepository ini.

Perbarui Alat untuk PowerShell Inti

Jalankan `Get-AWSPowerShellVersion` cmdlet untuk menentukan versi yang Anda jalankan, dan bandingkan dengan versi Alat untuk Windows PowerShell yang tersedia di situs web [PowerShell Galeri](#). Kami sarankan Anda memeriksanya setiap dua sampai tiga minggu. Support untuk perintah dan AWS layanan baru hanya tersedia setelah Anda memperbarui ke versi dengan dukungan itu.

Sebelum Anda menginstal rilis **AWSPowerShell** yang lebih baru. **NetCore**, hapus instalasi modul yang ada. Tutup PowerShell sesi terbuka sebelum Anda menghapus paket yang ada. Jalankan perintah berikut untuk menghapus paket.

```
PS > Uninstall-Module -Name AWSPowerShell.NetCore -AllVersions
```

Setelah paket dihapus, instal modul diperbarui dengan menjalankan perintah berikut.

```
PS > Install-Module -Name AWSPowerShell.NetCore
```

Setelah instalasi, jalankan perintah `Import-Module AWSPowerShell.NetCore` untuk memuat cmdlet yang diperbarui ke sesi Anda PowerShell .

Perbarui Alat untuk Windows PowerShell

Jalankan `Get-AWSPowerShellVersion` cmdlet untuk menentukan versi yang Anda jalankan, dan bandingkan dengan versi Alat untuk Windows PowerShell yang tersedia di situs web [PowerShell Galeri](#). Kami sarankan Anda memeriksanya setiap dua sampai tiga minggu. Support untuk perintah dan AWS layanan baru hanya tersedia setelah Anda memperbarui ke versi dengan dukungan itu.

- Jika Anda menginstal dengan menggunakan `Install-Module`, jalankan perintah berikut.

```
PS > Uninstall-Module -Name AWSPowerShell -AllVersions
PS > Install-Module -Name AWSPowerShell
```

- Jika Anda menginstal dengan menggunakan file ZIP yang diunduh:
 1. Unduh versi terbaru dari [Tools for PowerShell](#) web site. Bandingkan nomor versi paket dalam nama file yang diunduh dengan nomor versi yang Anda dapatkan saat menjalankan cmdlet `Get-AWSPowerShellVersion`.
 2. Jika versi unduhan adalah angka yang lebih tinggi dari versi yang telah Anda instal, tutup semua Alat untuk PowerShell konsol Windows.
 3. Instal versi yang lebih baru dari Tools untuk Windows PowerShell.

Setelah instalasi, jalankan `Import-Module AWSPowerShell` untuk memuat cmdlet yang diperbarui ke sesi Anda PowerShell . Atau jalankan Alat AWS untuk PowerShell konsol khusus dari menu Start Anda.

Menginstal Alat AWS untuk PowerShell di Linux atau macOS

Topik ini memberikan petunjuk tentang cara menginstal Alat AWS untuk PowerShell di Linux atau macOS.

Gambaran Umum Pengaturan

Untuk menginstal Alat AWS untuk PowerShell di komputer Linux atau macOS, Anda dapat memilih dari dua opsi paket:

- [AWS.Tools](#)— Versi termodulasi dari. Alat AWS untuk PowerShell Setiap AWS layanan didukung oleh modul kecil individualnya sendiri, dengan modul dukungan bersama `AWS.Tools.Common`.
- [AWSPowerCangkang.NetCore](#) — Versi modul tunggal dan besar dari. Alat AWS untuk PowerShell Semua AWS layanan didukung oleh modul tunggal dan besar ini.

Note

Ketahui bahwa modul tunggal mungkin terlalu besar untuk digunakan dengan [AWS Lambda](#) fungsi. Sebagai gantinya, gunakan versi termodulasi yang ditunjukkan di atas.

Mengatur salah satu dari pilihan ini pada komputer yang menjalankan Linux atau macOS memerlukan tugas-tugas berikut, dijelaskan secara rinci nanti dalam topik ini:

1. Instal PowerShell Core 6.0 atau yang lebih baru pada sistem yang didukung.
2. Setelah menginstal PowerShell Core, PowerShell mulailah dengan menjalankan `pwsh` shell sistem Anda.
3. Instal salah satu `AWS.Tools` atau `AWSPowerShell.NetCore`.
4. Jalankan `Import-Module` cmdlet yang sesuai untuk mengimpor modul ke sesi Anda PowerShell.
5. Jalankan cmdlet [Initialize-AWSDefault Configuration](#) untuk memberikan kredensial Anda. AWS

Prasyarat

Untuk menjalankan AWS Tools for PowerShell Core, komputer Anda harus menjalankan PowerShell Core 6.0 atau yang lebih baru.

- Untuk daftar rilis platform Linux yang didukung dan untuk informasi tentang cara menginstal versi terbaru PowerShell pada komputer berbasis Linux, lihat [Menginstal PowerShell di Linux di situs web](#) Microsoft. Beberapa sistem operasi berbasis Linux, seperti Arch, Kali, dan Raspbian, tidak didukung secara resmi, tetapi memiliki berbagai tingkat dukungan masyarakat.
- Untuk informasi tentang versi macOS yang didukung dan tentang cara menginstal versi terbaru di PowerShell macOS, lihat [Menginstal PowerShell macOS di situs web Microsoft](#).

Pasang **AWS.Tools** pada Linux atau macOS

Anda dapat menginstal versi termodulasi Alat AWS untuk PowerShell pada komputer yang menjalankan PowerShell Core 6.0 atau yang lebih baru. Untuk informasi tentang cara menginstal PowerShell Core, lihat [Menginstal berbagai versi PowerShell](#) di PowerShell situs web Microsoft.

Anda dapat memasang **AWS.Tools** dengan memilih satu dari tiga cara:

- Menggunakan cmdlet di modul **AWS.Tools.Installer**. Modul ini menyederhanakan instalasi dan pembaruan **AWS.Tools** modul lainnya. **AWS.Tools.Installer** membutuhkan **PowerShellGet**, dan secara otomatis mengunduh dan menginstal versi yang diperbarui. **AWS.Tools.Installer** secara otomatis membuat versi modul Anda tetap sinkron. Saat Anda menginstal atau memperbarui ke versi yang lebih baru dari satu modul, cmdlet secara **AWS.Tools.Installer** otomatis memperbarui semua **AWS.Tools** modul Anda yang lain ke versi yang sama.

Metode ini dijelaskan dalam prosedur berikut.

- Mengunduh modul dari [AWS.Tools.zip](#) dan mengekstraknya di salah satu direktori modul. Anda dapat menemukan direktori modul Anda dengan mencetak nilai variabel `$Env:PSModulePath`.
- Menginstal setiap modul layanan dari PowerShell Galeri menggunakan `Install-Module` cmdlet.

Untuk menginstal **AWS.Tools** di Linux atau macOS menggunakan modul **AWS.Tools.Installer**

1. Mulai sesi PowerShell Core dengan menjalankan perintah berikut.

```
$ pwsh
```

Note

Kami menyarankan agar Anda tidak menjalankan PowerShell sebagai administrator dengan izin yang ditinggikan kecuali jika diperlukan oleh tugas yang ada. Hal ini karena potensi risiko keamanan dan tidak sesuai dengan prinsip batasan akses yang paling rendah.

2. Untuk menginstal paket **AWS.Tools** termodulasi menggunakan modul **AWS.Tools.Installer**, jalankan perintah berikut.

```
PS > Install-Module -Name AWS.Tools.Installer
```

Untrusted repository

You are installing the modules from an untrusted repository. If you trust this repository, change its InstallationPolicy value by running the Set-PSRepository cmdlet. Are you sure you want to install the modules from 'PSGallery'?
 [Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "N"): **y**

Jika Anda diberi tahu bahwa repositori "tidak terpercaya", Anda akan ditanya apakah Anda tetap ingin menginstalnya. Masukkan **y** untuk memungkinkan PowerShell untuk menginstal modul. Untuk menghindari prompt dan menginstal modul tanpa mempercayai repositori, Anda dapat menjalankan perintah berikut.

```
PS > Install-Module -Name AWS.Tools.Installer -Force
```

3. Anda sekarang dapat menginstal modul untuk setiap layanan yang ingin Anda gunakan. Misalnya, perintah berikut menginstal modul Amazon EC2 dan Amazon S3. Perintah ini juga menginstal setiap modul tergantung yang diperlukan untuk modul tertentu yang akan dikerjakan. Misalnya, saat Anda menginstal modul layanan `AWS.Tools`, maka secara otomatis akan menginstal `AWS.Tools.Common`. Ini adalah modul bersama yang dibutuhkan oleh semua modul AWS layanan. Tindakan ini juga akan menghapus versi modul yang lebih lama, dan memperbarui modul lain ke versi yang sama barunya.

```
PS > Install-AWSToolsModule AWS.Tools.EC2,AWS.Tools.S3 -CleanUp
Confirm
Are you sure you want to perform this action?
Performing the operation "Install-AWSToolsModule" on target "AWS Tools version 4.0.0.0".
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "Y"):

Installing module AWS.Tools.Common version 4.0.0.0
Installing module AWS.Tools.EC2 version 4.0.0.0
Installing module AWS.Tools.Glacier version 4.0.0.0
Installing module AWS.Tools.S3 version 4.0.0.0

Uninstalling AWS.Tools version 3.3.618.0
Uninstalling module AWS.Tools.Glacier
Uninstalling module AWS.Tools.S3
Uninstalling module AWS.Tools.SimpleNotificationService
```

```
Uninstalling module AWS.Tools.SQS  
Uninstalling module AWS.Tools.Common
```

Note

`Install-AWSToolsModuleCmdlet` mengunduh semua modul yang diminta dari PSRepository nama PSGallery (<https://www.powershellgallery.com/>) dan menganggap repositori sebagai sumber tepercaya. Gunakan perintah `Get-PSRepository -Name PSGallery` untuk informasi lebih lanjut tentang PSRepository ini.

Perintah sebelumnya menginstal modul ke direktori default pada sistem Anda. Direktori sebenarnya tergantung pada distribusi dan versi sistem operasi Anda dan pada versi yang PowerShell Anda instal. Misalnya, jika Anda menginstal PowerShell 7 pada sistem seperti RHEL, modul default kemungkinan besar terletak di `/opt/microsoft/powershell/7/Modules` (atau `$PSHOME/Modules`) dan modul pengguna kemungkinan besar berada di `~/.local/share/powershell/Modules`. Untuk informasi selengkapnya, lihat [Menginstal PowerShell di Linux](#) di PowerShell situs web Microsoft. Untuk melihat di mana modul diinstal, jalankan perintah berikut:

```
PS > Get-Module -ListAvailable
```

Untuk menginstal modul lain, jalankan perintah serupa dengan nama modul yang sesuai, seperti yang ditemukan di [PowerShell Galeri](#).

Instal AWSPower Shell. NetCore di Linux atau macOS

Untuk meng-upgrade ke rilis AWSPower Shell yang lebih baru. NetCore, ikuti instruksi di [Memperbarui Alat AWS untuk PowerShell di Linux atau macOS](#). Copot pemasangan AWSPower Shell versi sebelumnya. NetCore pertama.

Anda dapat menginstal AWSPower Shell. NetCore dalam salah satu dari dua cara:

- Mengunduh modul dari [AWSPowerShell.NetCore.zip](#) dan mengekstraknya di salah satu direktori modul. Anda dapat menemukan direktori modul Anda dengan mencetak nilai variabel `$Env:PSModulePath`.

- Menginstal dari PowerShell Galeri menggunakan `Install-Module` cmdlet seperti yang dijelaskan dalam prosedur berikut.

Untuk menginstal AWSPower Shell. NetCore di Linux atau macOS menggunakan cmdlet `Install-Module`

Mulai sesi PowerShell Core dengan menjalankan perintah berikut.

```
$ pwsh
```

Note

Kami menyarankan Anda untuk tidak memulai PowerShell dengan menjalankan `sudo pwsh` untuk menjalankan PowerShell dengan hak administrator yang tinggi. Hal ini karena potensi risiko keamanan dan tidak sesuai dengan prinsip batasan akses yang paling rendah.

Untuk menginstal AWSPower Shell. NetCore paket modul tunggal dari PowerShell Galeri, jalankan perintah berikut.

```
PS > Install-Module -Name AWSPowerShell.NetCore
```

```
Untrusted repository
```

```
You are installing the modules from an untrusted repository. If you trust this repository, change its InstallationPolicy value by running the Set-PSRepository cmdlet. Are you sure
```

```
you want to install the modules from 'PSGallery'?
```

```
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "N"): y
```

Jika Anda diberi tahu bahwa repositori "tidak terpercaya", Anda akan ditanya apakah Anda tetap ingin menginstalnya. Masukkan **y** untuk memungkinkan PowerShell untuk menginstal modul. Untuk menghindari prompt tanpa mempercayai repositori, Anda dapat menjalankan perintah berikut.

```
PS > Install-Module -Name AWSPowerShell.NetCore -Force
```

Anda tidak perlu menjalankan perintah ini sebagai root, kecuali jika Anda ingin menginstal Alat AWS untuk PowerShell untuk semua pengguna komputer. Untuk melakukan ini, jalankan perintah berikut dalam PowerShell sesi yang telah Anda mulaisudo `pwsh`.

```
PS > Install-Module -Scope AllUsers -Name AWSPowerShell.NetCore -Force
```

Eksekusi Skrip

Perintah `Set-ExecutionPolicy` tidak tersedia pada sistem non-Windows. Anda dapat menjalankan `Get-ExecutionPolicy`, yang menunjukkan bahwa pengaturan kebijakan eksekusi default di PowerShell Core yang berjalan pada sistem non-Windows adalah `Unrestricted`. Untuk informasi selengkapnya, lihat [Tentang Kebijakan Eksekusi](#) di situs web Microsoft Technet.

Karena `PSModulePath` menyertakan lokasi direktori AWS modul, `Get-Module -ListAvailable` cmdlet menunjukkan modul yang Anda instal.

AWS.Tools

```
PS > Get-Module -ListAvailable
```

Directory: /Users/*username*/.local/share/powershell/Modules

ModuleType	Version	Name	PSEdition	ExportedCommands
Binary	3.3.563.1	AWS.Tools.Common	Desk	{Clear-AWSHistory, Set-AWSHistoryConfiguration, Initialize-AWSDefaultConfiguration, Clear-AWSDefaultConfigurat...

AWSPowerCangkang. NetCore

```
PS > Get-Module -ListAvailable
```

Directory: /Users/*username*/.local/share/powershell/Modules

ModuleType	Version	Name	ExportedCommands
Binary	3.3.563.1	AWSPowerShell.NetCore	

Konfigurasikan PowerShell Konsol untuk Menggunakan AWS Tools for PowerShell Core (AWSPowerShell. NetCore Hanya)

PowerShell Inti biasanya memuat modul secara otomatis setiap kali Anda menjalankan cmdlet dalam modul. Tapi ini tidak berhasil untuk AWSPower Shell. NetCore karena ukurannya yang besar. Untuk

mulai menjalankan AWSPower Shell. NetCore cmdlets, Anda harus terlebih dahulu menjalankan perintah. `Import-Module AWSPowerShell.NetCore` Ini tidak diperlukan untuk cmdlet di modul `AWS.Tools`.

Inisialisasi Sesi Anda PowerShell

Ketika Anda memulai PowerShell pada sistem berbasis Linux atau MacOS setelah Anda menginstal Alat AWS untuk PowerShell, Anda harus menjalankan [Initialize- AWSDefault Configuration](#) untuk menentukan kunci akses mana yang akan digunakan. AWS Untuk informasi selengkapnya tentang `Initialize-AWSDefaultConfiguration`, lihat [Menggunakan AWS Kredensial](#).

Note

Dalam rilis sebelumnya (sebelum 3.3.96.0) dari Alat AWS untuk PowerShell, cmdlet ini diberi nama. `Initialize-AWSDefaults`

Penentuan versi

AWS merilis versi baru secara Alat AWS untuk PowerShell berkala untuk mendukung AWS layanan dan fitur baru. Untuk menentukan versi yang telah Anda instal, jalankan [Get- AWSPowerShellVersion](#) cmdlet. Alat AWS untuk PowerShell

Misalnya:

```
PS > Get-AWSPowerShellVersion
```

```
AWS Tools for PowerShell
```

```
Version 4.1.675
```

```
Copyright 2012-2024 Amazon.com, Inc. or its affiliates. All Rights Reserved.
```

```
Amazon Web Services SDK for .NET
```

```
Core Runtime Version 3.7.400.33
```

```
Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
```

```
Release notes: https://github.com/aws/aws-tools-for-powershell/blob/master/CHANGELOG.md
```

```
This software includes third party software subject to the following copyrights:
```

```
- Logging from log4net, Apache License
```

```
[http://logging.apache.org/log4net/license.html]
```

Untuk melihat daftar AWS layanan yang didukung dalam versi alat saat ini, tambahkan - `ListServiceVersionInfo` parameter ke `AWSPowerShellVersion` cmdlet [Get](#).

Untuk menentukan versi PowerShell yang Anda jalankan, masukkan `$PSVersionTable` untuk melihat konten [variabel \\$PSVersionTable otomatis](#).

Misalnya:

```
PS > $PSVersionTable
Name                           Value
----                           -
PSVersion                      6.2.2
PSEdition                      Core
GitCommitId                    6.2.2
OS                              Darwin 18.7.0 Darwin Kernel Version 18.7.0: Tue Aug 20
 16:57:14 PDT 2019; root:xnu-4903.271.2~2/RELEASE_X86_64
Platform                       Unix
PSCompatibleVersions            {1.0, 2.0, 3.0, 4.0...}
PSRemotingProtocolVersion      2.3
SerializationVersion           1.1.0.1
WSManStackVersion              3.0
```

Memperbarui Alat AWS untuk PowerShell di Linux atau macOS

Secara berkala, saat versi terbaru dirilis, Anda harus memperbarui versi yang Anda jalankan secara lokal. Alat AWS untuk PowerShell

Perbarui modul termodulasi **AWS.Tools**

Untuk memperbarui `AWS.Tools` modul Anda ke versi terbaru, jalankan perintah berikut:

```
PS > Update-AWSToolsModule -Cleanup
```

Perintah ini memperbarui semua modul `AWS.Tools` yang saat ini diinstal dan, untuk modul-modul yang berhasil diperbarui, menghapus versi sebelumnya.

Note

`Update-AWSToolsModuleCmdlet` mengunduh semua modul dari PSRepository nama `PSGallery` (<https://www.powershellgallery.com/>) dan menganggapnya sebagai sumber

tepercaya. Gunakan perintah `Get-PSRepository -Name PSGallery` untuk informasi lebih lanjut tentang PSRepository ini.

Perbarui Alat untuk PowerShell Inti

Jalankan `Get-AWSPowerShellVersion` cmdlet untuk menentukan versi yang Anda jalankan, dan bandingkan dengan versi Alat untuk Windows PowerShell yang tersedia di situs web [PowerShell Galeri](#). Kami sarankan Anda memeriksanya setiap dua sampai tiga minggu. Support untuk perintah dan AWS layanan baru hanya tersedia setelah Anda memperbarui ke versi dengan dukungan itu.

Sebelum Anda menginstal rilis AWSPower Shell yang lebih baru. NetCore, hapus instalasi modul yang ada. Tutup PowerShell sesi terbuka sebelum Anda menghapus paket yang ada. Jalankan perintah berikut untuk menghapus paket.

```
PS > Uninstall-Module -Name AWSPowerShell.NetCore -AllVersions
```

Setelah paket dihapus, instal modul diperbarui dengan menjalankan perintah berikut.

```
PS > Install-Module -Name AWSPowerShell.NetCore
```

Setelah instalasi, jalankan perintah `Import-Module AWSPowerShell.NetCore` untuk memuat cmdlet yang diperbarui ke sesi Anda PowerShell .

Informasi Terkait

- [Memulai dengan AWS Tools for Windows PowerShell](#)
- [Bekerja dengan AWS layanan di Alat AWS untuk PowerShell](#)

Migrasi dari Alat AWS untuk PowerShell Versi 3.3 ke Versi 4

Alat AWS untuk PowerShell versi 4 adalah pembaruan yang kompatibel ke belakang ke Alat AWS untuk PowerShell versi 3.3. Versi ini menambahkan perbaikan yang signifikan dengan tetap mempertahankan perilaku cmdlet yang ada.

Skrip Anda yang ada harus terus bekerja setelah diperbarui ke versi baru, tetapi kami merekomendasikan Anda mengujinya secara menyeluruh sebelum memperbarui lingkungan produksi Anda.

Bagian ini menjelaskan perubahan dan menjelaskan bagaimana kemungkinan dampaknya terhadap skrip Anda.

Versi **AWS.Tools** Baru yang Sepenuhnya Termodulasi

AWSPowerCangkangnya. NetCore dan paket AWSPower Shell adalah “monolitik”. Ini berarti bahwa semua AWS layanan didukung dalam modul yang sama, membuatnya sangat besar, dan tumbuh lebih besar karena setiap AWS layanan dan fitur baru ditambahkan. **AWS.Tools** Paket baru dipecah menjadi modul yang lebih kecil yang memberi Anda fleksibilitas untuk mengunduh dan menginstal hanya yang Anda butuhkan untuk AWS layanan yang Anda gunakan. Paket termasuk modul **AWS.Tools.Common** bersama yang diperlukan oleh semua modul lainnya, dan modul **AWS.Tools.Installer** yang menyederhanakan pemasangan, pembaruan, dan penghapusan modul sesuai kebutuhan.

Hal ini juga memungkinkan pengimporan otomatis cmdlet pada panggilan pertama, tanpa harus terlebih dulu memanggil **Import-Module**. Namun, untuk berinteraksi dengan objek.NET terkait sebelum memanggil cmdlet, Anda tetap harus menelepon **Import-Module** untuk memberi PowerShell tahu tentang jenis.NET yang relevan.

Misalnya, perintah berikut memiliki referensi ke **Amazon.EC2.Model.Filter**. Jenis referensi ini tidak dapat memicu pengimporan otomatis, jadi Anda harus terlebih dulu memanggil **Import-Module** atau perintah ini akan gagal.

```
PS > $filter = [Amazon.EC2.Model.Filter]@{Name="vpc-id";Values="vpc-1234abcd"}
InvalidOperation: Unable to find type [Amazon.EC2.Model.Filter].
```

```
PS > Import-Module AWS.Tools.EC2
PS > $filter = [Amazon.EC2.Model.Filter]@{Name="vpc-id";Values="vpc-1234abcd"}
PS > Get-EC2Instance -Filter $filter -Select Reservations.Instances.InstanceId
i-0123456789abcdefg
i-0123456789hijklmn
```

Cmdlet **Get-AWSService** baru

Untuk membantu Anda menemukan nama-nama modul untuk setiap AWS layanan dalam **AWS.Tools** kumpulan modul, Anda dapat menggunakan **Get-AWSService** cmdlet.

```
PS > Get-AWSService
```

```

Service : ACMPCA
CmdletNounPrefix : PCA
ModuleName : AWS.Tools.ACMPCA
SDKAssemblyVersion : 3.3.101.56
ServiceName : Certificate Manager Private Certificate Authority

Service : AlexaForBusiness
CmdletNounPrefix : ALXB
ModuleName : AWS.Tools.AlexaForBusiness
SDKAssemblyVersion : 3.3.106.26
ServiceName : Alexa For Business
...

```

Parameter **-Select** baru untuk Mengendalikan Obyek yang Dikembalikan oleh Cmdlet

Sebagian besar cmdlet dalam versi 4 mendukung parameter **-Select** baru. Setiap cmdlet memanggil AWS layanan APIs untuk Anda menggunakan. AWS SDK untuk .NET Kemudian Alat AWS untuk PowerShell klien mengubah respons menjadi objek yang dapat Anda gunakan dalam PowerShell skrip dan pipa ke perintah lain. Terkadang PowerShell objek akhir memiliki lebih banyak bidang atau properti dalam respons asli daripada yang Anda butuhkan, dan di lain waktu Anda mungkin ingin objek menyertakan bidang atau properti respons yang tidak ada secara default. Parameter **-Select** memungkinkan Anda untuk menyebutkan yang termasuk dalam objek .NET yang dikembalikan oleh cmdlet.

Misalnya, [Get-S3Object](#) cmdlet memanggil operasi Amazon S3 SDK. [ListObjects](#) Operasi itu mengembalikan [ListObjectsResponse](#) objek. Namun, secara default, `Get-S3Object` cmdlet hanya mengembalikan `S3Objects` elemen respons SDK kepada pengguna. PowerShell Pada contoh berikut, objek adalah array dengan dua elemen.

```

PS > Get-S3Object -BucketName amzn-s3-demo-bucket

ETag : "01234567890123456789012345678901111"
BucketName : amzn-s3-demo-bucket
Key : file1.txt
LastModified : 9/30/2019 1:31:40 PM
Owner : Amazon.S3.Model.Owner
Size : 568
StorageClass : STANDARD

```

```
ETag : "01234567890123456789012345678902222"  
BucketName : amzn-s3-demo-bucket  
Key : file2.txt  
LastModified : 7/15/2019 9:36:54 AM  
Owner : Amazon.S3.Model.Owner  
Size : 392  
StorageClass : STANDARD
```

Di Alat AWS untuk PowerShell versi 4, Anda dapat menentukan `-Select *` untuk mengembalikan objek `respons.NET` lengkap yang dikembalikan oleh panggilan SDK API.

```
PS > Get-S3Object -BucketName amzn-s3-demo-bucket -Select *  
IsTruncated      : False  
NextMarker       :  
S3Objects        : {file1.txt, file2.txt}  
Name             : amzn-s3-demo-bucket  
Prefix          :  
MaxKeys          : 1000  
CommonPrefixes  : {}  
Delimiter        :
```

Anda juga dapat menyebutkan jalur ke properti nest tertentu yang Anda inginkan. Contoh berikut hanya mengembalikan properti `Key` dari setiap elemen dalam array `S3Objects`.

```
PS > Get-S3Object -BucketName amzn-s3-demo-bucket -Select S3Objects.Key  
file1.txt  
file2.txt
```

Dalam situasi tertentu, ini dapat berguna untuk mengembalikan parameter cmdlet. Anda dapat melakukannya dengan `-Select ^ParameterName`. Fitur ini menggantikan parameter `-PassThru`, yang masih tersedia tetapi tidak lagi digunakan.

```
PS > Get-S3Object -BucketName amzn-s3-demo-bucket -Select S3Objects.Key |  
>> Write-S3ObjectTagSet -Select ^Key -BucketName amzn-s3-demo-bucket -Tagging_TagSet  
@{ Key='key'; Value='value'}  
file1.txt  
file2.txt
```

[Topik referensi](#) untuk setiap cmdlet menentukan apakah mendukung parameter `-Select` tersebut atau tidak.

Pembatasan Lebih Konsisten dari Jumlah Item dalam Output

Versi sebelumnya Alat AWS untuk PowerShell memungkinkan Anda untuk menggunakan `-MaxItems` parameter untuk menentukan jumlah maksimum objek yang dikembalikan dalam output akhir.

Perilaku ini dihapus dari `AWS.Tools`.

Perilaku ini tidak digunakan lagi di Shell. `AWSPower NetCore` dan `AWSPower Shell`, dan akan dihapus dari versi tersebut di rilis mendatang.

Jika API layanan yang mendasari mendukung parameter `MaxItems`, maka akan masih tersedia dan berfungsi sebagaimana yang disebutkan API. Tetapi tidak lagi mempunyai perilaku tambahan yang membatasi jumlah item yang dikembalikan dalam output cmdlet.

Untuk membatasi jumlah item yang dikembalikan dalam output akhir, pipa output ke `Select-Object` cmdlet dan tentukan `-First n` parameternya, di mana *n* jumlah maksimum item yang akan dimasukkan dalam output akhir.

```
PS > Get-S3ObjectV2 -BucketName amzn-s3-demo-bucket -Select S3Objects.Key | select -first 2
file1.txt
file2.txt
```

Tidak semua AWS layanan didukung dengan `-MaxItems` cara yang sama, jadi ini menghilangkan ketidakkonsistenan itu dan hasil tak terduga yang terkadang terjadi. Dan juga, `-MaxItems` yang dikombinasikan dengan parameter `-Select` baru terkadang dapat mengakibatkan hasil yang membingungkan.

Lebih Mudah Menggunakan Parameter Pengaliran

Parameter jenis `Stream` atau `byte[]` sekarang dapat menerima nilai `string`, `string[]`, atau `FileInfo`.

Misalnya, Anda dapat menggunakan hal berikut.

```
PS > Invoke-LMFunction -FunctionName MyTestFunction -PayloadStream '{
>> "some": "json"
>> }'
```

```
PS > Invoke-LMFunction -FunctionName MyTestFunction -PayloadStream (ls .\some.json)
```

```
PS > Invoke-LMFunction -FunctionName MyTestFunction -PayloadStream @('{', '"some":  
"json"', '}'')
```

Alat AWS untuk PowerShell mengkonversi semua string untuk `byte[]` menggunakan UTF-8 encoding.

Memperluas Alur dengan Nama Properti

Agar pengalaman pengguna lebih konsisten, Anda sekarang dapat melewati input alur dengan menyebutkan nama properti untuk parameter apa pun.

Pada contoh berikut, kita membuat obyek kustom dengan properti yang memiliki nama yang cocok dengan nama parameter cmdlet target. Ketika cmdlet berjalan, maka secara otomatis menggunakan properti tersebut sebagai parameternya.

```
PS > [pscustomobject] @{ BucketName='amzn-s3-demo-bucket'; Key='file1.txt';  
PartNumber=1 } | Get-S3ObjectMetadata
```

Note

Beberapa properti mendukung ini di versi sebelumnya Alat AWS untuk PowerShell. Versi 4 membuatnya lebih konsisten dengan memungkinkannya untuk semua parameter.

Parameter Umum Statis

Untuk meningkatkan konsistensi di versi 4.0 Alat AWS untuk PowerShell, semua parameter bersifat statis.

Pada versi sebelumnya Alat AWS untuk PowerShell, beberapa parameter umum seperti `AccessKey`, `SecretKey`, atau `ProfileNameRegion`, bersifat [dinamis](#), sementara semua parameter lainnya bersifat statis. Ini dapat menimbulkan masalah karena PowerShell mengikat parameter statis sebelum parameter dinamis. Sebagai contoh, katakanlah Anda menjalankan perintah berikut.

```
PS > Get-EC2Region -Region us-west-2
```

Versi sebelumnya PowerShell mengikat nilai `us-west-2` ke parameter `-RegionName` statis alih-alih parameter `-Region` dinamis. Kemungkinan besar, hal ini dapat membingungkan pengguna.

AWS.Tools Menyatakan dan Menerapkan Parameter Wajib

Modul-modul `AWS.Tools` * saat ini menyatakan dan menerapkan parameter cmdlet wajib. Saat AWS Layanan menyatakan bahwa parameter API diperlukan, akan PowerShell meminta parameter cmdlet yang sesuai jika Anda tidak menentukannya. Ini hanya berlaku untuk `AWS.Tools`. Untuk memastikan kompatibilitas mundur, ini tidak berlaku untuk `AWSPowerShell` `NetCore` atau `AWSPowerShell`.

Semua Parameter Dapat Dibatalkan

Sekarang Anda dapat menetapkan `$null` untuk parameter jenis nilai (angka dan tanggal). Perubahan ini seharusnya tidak mempengaruhi skrip yang ada. Hal ini memungkinkan Anda untuk memotong prompt untuk parameter wajib. Parameter wajib diterapkan hanya di `AWS.Tools`.

Jika Anda menjalankan contoh berikut menggunakan versi 4, maka akan secara efektif memotong validasi sisi klien karena Anda memberikan "value" untuk setiap parameter wajib. Namun, panggilan layanan Amazon EC2 API gagal karena AWS layanan masih memerlukan informasi tersebut.

```
PS > Get-EC2InstanceAttribute -InstanceId $null -Attribute $null
WARNING: You are passing $null as a value for parameter Attribute which is marked as
required.
In case you believe this parameter was incorrectly marked as required, report this by
opening
an issue at https://github.com/aws/aws-tools-for-powershell/issues.
WARNING: You are passing $null as a value for parameter InstanceId which is marked as
required.
In case you believe this parameter was incorrectly marked as required, report this by
opening
an issue at https://github.com/aws/aws-tools-for-powershell/issues.

Get-EC2InstanceAttribute : The request must contain the parameter instanceId
```

Menghapus Fitur yang Sudah Tidak Lagi Digunakan

Fitur-fitur berikut tidak digunakan lagi dalam rilis sebelumnya Alat AWS untuk PowerShell dan dihapus di versi 4:

- Menghapus parameter `-Terminate` dari cmdlet `Stop-EC2Instance`. Sebagai gantinya, gunakan `Remove-EC2Instance`.
- Menghapus `-ProfileName` parameter dari `Clear-AWSCredential` cmdlet. Sebagai gantinya, gunakan `Remove-AWSCredentialProfile`.
- Menghapus cmdlet `Import-EC2Instance` dan `Import-EC2Volume`.

Memulai dengan AWS Tools for Windows PowerShell

Beberapa topik di bagian ini menjelaskan dasar-dasar penggunaan Alat untuk Windows PowerShell setelah Anda [menginstal alat](#). Misalnya, mereka menjelaskan cara menentukan [kredensi](#) dan [AWS Wilayah](#) mana yang PowerShell harus digunakan Alat untuk Windows saat berinteraksi. AWS

Topik lain di bagian ini memberikan informasi tentang cara-cara lanjutan agar Anda dapat mengonfigurasi alat, lingkungan, dan proyek Anda.

Topik

- [Konfigurasi otentikasi alat dengan AWS](#)
- [Tentukan AWS Wilayah](#)
- [Konfigurasi identitas federasi dengan Alat AWS untuk PowerShell](#)
- [Penemuan dan alias Cmdlet](#)
- [Pipelining, output, dan iterasi di Alat AWS untuk PowerShell](#)
- [Resolusi kredensi dan profil](#)
- [Informasi tambahan tentang pengguna dan peran](#)
- [Menggunakan kredensial warisan](#)

Konfigurasi otentikasi alat dengan AWS

Anda harus menetapkan bagaimana kode Anda mengautentikasi AWS saat mengembangkan dengan Layanan AWS. Ada berbagai cara di mana Anda dapat mengonfigurasi akses terprogram ke AWS sumber daya, tergantung pada lingkungan dan AWS akses yang tersedia untuk Anda.

Untuk melihat berbagai metode otentikasi untuk Alat untuk PowerShell, lihat [Otentikasi dan akses](#) di Panduan Referensi Alat AWS SDKs dan Alat.

Topik ini mengasumsikan bahwa pengguna baru sedang berkembang secara lokal, belum diberikan metode otentikasi oleh majikan mereka, dan akan digunakan AWS IAM Identity Center untuk mendapatkan kredensial sementara. Jika lingkungan Anda tidak termasuk dalam asumsi ini, beberapa informasi dalam topik ini mungkin tidak berlaku untuk Anda, atau beberapa informasi mungkin telah diberikan kepada Anda.

Mengkonfigurasi lingkungan ini memerlukan beberapa langkah, yang dirangkum sebagai berikut:

1. [Aktifkan dan konfigurasi Pusat Identitas IAM](#)
2. [Konfigurasi Alat PowerShell untuk menggunakan IAM Identity Center.](#)
3. [Memulai sesi portal AWS akses](#)

Aktifkan dan konfigurasi Pusat Identitas IAM

Untuk menggunakannya AWS IAM Identity Center, pertama-tama harus diaktifkan dan dikonfigurasi. Untuk melihat detail tentang cara melakukannya PowerShell, lihat Langkah 1 dalam topik [otentikasi Pusat Identitas IAM](#) di Panduan Referensi Alat AWS SDKs dan Alat. Secara khusus, ikuti instruksi yang diperlukan di bawah Saya tidak memiliki akses yang ditetapkan melalui Pusat Identitas IAM.

Konfigurasi Alat PowerShell untuk menggunakan IAM Identity Center.

Note

Dimulai dengan versi 4.1.538 dari Tools for PowerShell, metode yang disarankan untuk mengonfigurasi kredensi SSO dan memulai sesi portal AWS akses adalah dengan menggunakan [Initialize-AWSSSOConfiguration](#) dan [Invoke-AWSSSOLogin](#) cmdlet, seperti yang dijelaskan dalam topik ini. Jika Anda tidak memiliki akses ke versi Alat untuk PowerShell (atau yang lebih baru) atau tidak dapat menggunakan cmdlet tersebut, Anda masih dapat melakukan tugas-tugas ini dengan menggunakan AWS CLI Untuk mengetahui caranya, lihat [Gunakan AWS CLI untuk login portal](#).

Prosedur berikut memperbarui AWS config file bersama dengan informasi SSO yang Alat untuk PowerShell digunakan untuk mendapatkan kredensi sementara. Sebagai konsekuensi dari prosedur ini, sesi portal AWS akses juga dimulai. Jika config file bersama sudah memiliki informasi SSO dan Anda hanya ingin tahu cara memulai sesi portal akses menggunakan Alat untuk PowerShell, lihat bagian selanjutnya dalam topik ini, [Memulai sesi portal AWS akses](#).

1. Jika Anda belum melakukannya, buka PowerShell dan instal yang Alat AWS untuk PowerShell sesuai untuk sistem operasi dan lingkungan Anda, termasuk cmdlet umum. Untuk informasi tentang cara melakukan ini, lihat [Instalasi Alat AWS untuk PowerShell](#).

Misalnya, jika menginstal versi Tools termodulasi untuk PowerShell Windows, kemungkinan besar Anda akan menjalankan perintah yang mirip dengan yang berikut ini:

```
Install-Module -Name AWS.Tools.Installer  
Install-AWSToolsModule AWS.Tools.Common
```

2. Jalankan perintah berikut. Ganti nilai properti contoh dengan nilai dari konfigurasi Pusat Identitas IAM Anda. Untuk informasi tentang properti ini dan cara menemukannya, lihat [setelan penyedia kredensi Pusat Identitas IAM di Panduan](#) Referensi Alat AWS SDKs dan Alat.

```
$params = @{  
  ProfileName = 'my-sso-profile'  
  AccountId = '111122223333'  
  RoleName = 'SamplePermissionSet'  
  SessionName = 'my-sso-session'  
  StartUrl = 'https://provided-domain.awsapps.com/start'  
  SSORegion = 'us-west-2'  
  RegistrationScopes = 'sso:account:access'  
};  
Initialize-AWSSSOConfiguration @params
```

Atau, Anda cukup menggunakan cmdlet dengan sendirinya `Initialize-AWSSSOConfiguration`, dan Alat untuk PowerShell meminta Anda untuk nilai properti.

Pertimbangan untuk nilai properti tertentu:

- Jika Anda hanya mengikuti instruksi untuk [mengaktifkan dan mengkonfigurasi IAM Identity Center](#), nilainya `-RoleName` mungkin `PowerUserAccess`. Tetapi jika Anda membuat izin Pusat Identitas IAM yang ditetapkan khusus untuk PowerShell pekerjaan, gunakan itu sebagai gantinya.
 - Pastikan untuk menggunakan Wilayah AWS tempat Anda telah mengkonfigurasi Pusat Identitas IAM.
3. Pada titik ini, AWS config file bersama berisi profil yang dipanggil `my-sso-profile` dengan serangkaian nilai konfigurasi yang dapat direferensikan dari Alat untuk PowerShell. Untuk menemukan lokasi file ini, lihat [Lokasi file bersama di](#) Panduan Referensi Alat AWS SDKs dan.

Alat untuk PowerShell menggunakan penyedia token SSO profil untuk memperoleh kredensial sebelum mengirim permintaan ke `AWSsso_role_name` Nilai, yang merupakan peran IAM yang terhubung ke set izin Pusat Identitas IAM, harus memungkinkan akses ke yang Layanan AWS digunakan dalam aplikasi Anda.

Contoh berikut menunjukkan profil yang dibuat dengan menggunakan perintah yang ditunjukkan di atas. Beberapa nilai properti dan urutannya mungkin berbeda di profil Anda yang sebenarnya. `sso-session` properti profil mengacu pada bagian bernama `my-sso-session`, yang berisi pengaturan untuk memulai sesi portal AWS akses.

```
[profile my-sso-profile]
sso_account_id=111122223333
sso_role_name=SamplePermissionSet
sso_session=my-sso-session

[sso-session my-sso-session]
sso_region=us-west-2
sso_registration_scopes=sso:account:access
sso_start_url=https://provided-domain.awsapps.com/start/
```

4. Jika Anda sudah memiliki sesi portal AWS akses aktif, Alat untuk PowerShell memberi tahu Anda bahwa Anda sudah masuk.

Jika bukan itu masalahnya, Alat untuk PowerShell mencoba membuka halaman otorisasi SSO secara otomatis di browser web default Anda. Ikuti petunjuk di browser Anda, yang mungkin termasuk kode otorisasi SSO, nama pengguna dan kata sandi, dan izin untuk mengakses AWS IAM Identity Center akun dan set izin.

Alat untuk PowerShell memberi tahu Anda bahwa login SSO berhasil.

Memulai sesi portal AWS akses

Sebelum menjalankan perintah yang mengakses Layanan AWS, Anda memerlukan sesi portal AWS akses aktif sehingga Alat untuk PowerShell dapat menggunakan autentikasi IAM Identity Center untuk menyelesaikan kredensial. Untuk masuk ke portal AWS akses, jalankan perintah berikut di PowerShell, di mana `-ProfileName my-sso-profile` nama profil yang dibuat di config file bersama saat Anda mengikuti prosedur di bagian sebelumnya dari topik ini.

```
Invoke-AWSSSOLogin -ProfileName my-sso-profile
```

Jika Anda sudah memiliki sesi portal AWS akses aktif, Alat untuk PowerShell memberi tahu Anda bahwa Anda sudah masuk.

Jika bukan itu masalahnya, Alat untuk PowerShell mencoba membuka halaman otorisasi SSO secara otomatis di browser web default Anda. Ikuti petunjuk di browser Anda, yang mungkin termasuk kode otorisasi SSO, nama pengguna dan kata sandi, dan izin untuk mengakses AWS IAM Identity Center akun dan set izin.

Alat untuk PowerShell memberi tahu Anda bahwa login SSO berhasil.

Untuk menguji apakah Anda sudah memiliki sesi aktif, jalankan perintah berikut setelah menginstal atau mengimpor `AWS.Tools.SecurityToken` modul sesuai kebutuhan.

```
Get-STSCallerIdentity -ProfileName my-sso-profile
```

Respons terhadap `Get-STSCallerIdentity` cmdlet melaporkan akun IAM Identity Center dan set izin yang dikonfigurasi dalam file bersama. `config`

Contoh

Berikut ini adalah contoh bagaimana menggunakan IAM Identity Center dengan Tools for PowerShell. Ini mengasumsikan sebagai berikut:

- Anda telah mengaktifkan IAM Identity Center dan mengonfigurasinya seperti yang dijelaskan sebelumnya dalam topik ini. Properti SSO ada di `my-sso-profile` profil, yang telah dikonfigurasi sebelumnya dalam topik ini.
- Saat Anda masuk melalui `Initialize-AWSSSOConfiguration` atau `Invoke-AWSSSOLogin` cmdlet, pengguna memiliki setidaknya izin hanya-baca untuk Amazon S3.
- Beberapa bucket S3 tersedia untuk dilihat pengguna tersebut.

Instal atau impor `AWS.Tools.S3` modul sesuai kebutuhan dan kemudian gunakan PowerShell perintah berikut untuk menampilkan daftar bucket S3.

```
Get-S3Bucket -ProfileName my-sso-profile
```

Informasi tambahan

- Untuk opsi lebih lanjut tentang otentikasi Alat untuk PowerShell, seperti penggunaan profil dan variabel lingkungan, lihat bagian [konfigurasi](#) di Panduan Referensi Alat AWS SDKs dan Alat.
- Beberapa perintah memerlukan AWS Region untuk ditentukan. Ada beberapa cara untuk melakukannya, termasuk opsi `-Region` cmdlet, `[default]` profil, dan variabel `AWS_REGION`

lingkungan. Untuk informasi selengkapnya, lihat [Tentukan AWS Wilayah](#) di panduan ini dan [AWS Wilayah](#) di Panduan Referensi Alat AWS SDKs dan Alat.

- Untuk mempelajari lebih lanjut tentang praktik terbaik, lihat [Praktik terbaik keamanan di IAM](#) di Panduan Pengguna IAM.
- Untuk membuat AWS kredensial jangka pendek, lihat [Kredensial Keamanan Sementara di Panduan Pengguna](#) IAM.
- Untuk mempelajari tentang penyedia kredensi lainnya, lihat Penyedia kredensi [terstandarisasi di Panduan Referensi](#) Alat AWS SDKs dan Alat.

Topik

- [Gunakan AWS CLI untuk login portal](#)

Gunakan AWS CLI untuk login portal

Dimulai dengan versi 4.1.538 dari Tools for PowerShell, metode yang disarankan untuk mengonfigurasi kredensi SSO dan memulai sesi portal AWS akses adalah dengan menggunakan [Initialize-AWSSSOConfiguration](#) dan [Invoke-AWSSSOLogin](#) cmdlet, seperti yang dijelaskan dalam [Konfigurasi otentikasi alat dengan AWS](#). Jika Anda tidak memiliki akses ke versi Alat untuk PowerShell (atau yang lebih baru) atau tidak dapat menggunakan cmdlet tersebut, Anda masih dapat melakukan tugas-tugas ini dengan menggunakan AWS CLI.

Konfigurasi Alat PowerShell untuk menggunakan Pusat Identitas IAM melalui AWS CLI

Jika Anda belum melakukannya, pastikan untuk [Aktifkan dan konfigurasi IAM Identity Center](#) sebelum Anda melanjutkan.

Informasi tentang cara mengkonfigurasi Alat PowerShell untuk menggunakan Pusat Identitas IAM melalui AWS CLI adalah di Langkah 2 dalam topik untuk [otentikasi Pusat Identitas IAM](#) di AWS SDKs dan Panduan Referensi Alat. Setelah Anda menyelesaikan konfigurasi ini, sistem Anda harus berisi elemen-elemen berikut:

- Itu AWS CLI, yang Anda gunakan untuk memulai sesi portal AWS akses sebelum Anda menjalankan aplikasi Anda.
- AWS configFile bersama yang berisi [\[default\]profil](#) dengan serangkaian nilai konfigurasi yang dapat direferensikan dari Alat untuk PowerShell. Untuk menemukan lokasi file ini, lihat [Lokasi](#)

[file bersama di](#) Panduan Referensi Alat AWS SDKs dan. Alat untuk PowerShell menggunakan penyedia token SSO profil untuk memperoleh kredensial sebelum mengirim permintaan ke. `AWSsso_role_name` Nilai, yang merupakan peran IAM yang terhubung ke set izin Pusat Identitas IAM, harus memungkinkan akses ke yang Layanan AWS digunakan dalam aplikasi Anda.

`configFile` contoh berikut menunjukkan `[default]` profil yang disiapkan dengan penyedia token SSO. `sso_session` Pengaturan profil mengacu pada `sso-session` bagian bernama. `sso-session` Bagian ini berisi pengaturan untuk memulai sesi portal AWS akses.

```
[default]
sso_session = my-sso
sso_account_id = 111122223333
sso_role_name = SampleRole
region = us-east-1
output = json

[sso-session my-sso]
sso_region = us-east-1
sso_start_url = https://provided-domain.awsapps.com/start
sso_registration_scopes = sso:account:access
```

Important

PowerShell Sesi Anda harus memiliki modul berikut diinstal dan diimpor sehingga resolusi SSO dapat bekerja:

- `AWS.Tools.SSO`
- `AWS.Tools.SSOIDC`

Jika Anda menggunakan versi Tools yang lebih lama PowerShell dan Anda tidak memiliki modul ini, Anda akan mendapatkan kesalahan yang mirip dengan berikut: “Assembly `AWSSDK.SSOIDC` tidak dapat ditemukan...”.

Memulai sesi portal AWS akses

Sebelum menjalankan perintah yang mengakses Layanan AWS, Anda memerlukan sesi portal AWS akses aktif sehingga Alat untuk Windows PowerShell dapat menggunakan autentikasi IAM

Identity Center untuk menyelesaikan kredensial. Bergantung pada panjang sesi yang dikonfigurasi, akses Anda pada akhirnya akan kedaluwarsa dan Alat untuk Windows PowerShell akan mengalami kesalahan otentikasi. Untuk masuk ke portal AWS akses, jalankan perintah berikut di AWS CLI.

```
aws sso login
```

Karena Anda menggunakan [default] profil, Anda tidak perlu memanggil perintah dengan --profile opsi. Jika konfigurasi penyedia token SSO Anda menggunakan profil bernama, perintahnya adalah `aws sso login --profile named-profile` sebagai gantinya. Untuk informasi selengkapnya tentang profil bernama, lihat bagian [Profil](#) di Panduan Referensi Alat AWS SDKs dan Alat.

Untuk menguji apakah Anda sudah memiliki sesi aktif, jalankan AWS CLI perintah berikut (dengan pertimbangan yang sama untuk profil bernama):

```
aws sts get-caller-identity
```

Respons terhadap perintah ini harus melaporkan akun IAM Identity Center dan set izin yang dikonfigurasi dalam config file bersama.

Note

Jika Anda sudah memiliki sesi portal AWS akses aktif dan menjalankannya `aws sso login`, Anda tidak akan diminta untuk memberikan kredensial.

Proses masuk mungkin meminta Anda untuk mengizinkan AWS CLI akses ke data Anda. Karena AWS CLI dibangun di atas SDK untuk Python, pesan izin mungkin berisi variasi nama. `botocore`

Contoh

Berikut ini adalah contoh bagaimana menggunakan IAM Identity Center dengan Tools for PowerShell. Ini mengasumsikan sebagai berikut:

- Anda telah mengaktifkan IAM Identity Center dan mengonfigurasinya seperti yang dijelaskan sebelumnya dalam topik ini. Properti SSO ada di [default] profil.
- Saat Anda masuk melalui AWS CLI by using `aws sso login`, pengguna tersebut memiliki setidaknya izin hanya-baca untuk Amazon S3.

- Beberapa bucket S3 tersedia untuk dilihat pengguna tersebut.

Gunakan PowerShell perintah berikut untuk menampilkan daftar bucket S3:

```
Install-Module AWS.Tools.Installer
Install-AWSToolsModule S3
# And if using an older version of the AWS Tools for PowerShell:
Install-AWSToolsModule SS0, SS00IDC

# In older versions of the AWS Tools for PowerShell, we're not invoking a cmdlet from
these modules directly,
# so we must import them explicitly:
Import-Module AWS.Tools.SS0
Import-Module AWS.Tools.SS00IDC

# Older versions of the AWS Tools for PowerShell don't support the SS0 login flow, so
login with the CLI
aws sso login

# Now we can invoke cmdlets using the SS0 profile
Get-S3Bucket
```

Seperti disebutkan di atas, karena Anda menggunakan [default] profil, Anda tidak perlu memanggil Get-S3Bucket cmdlet dengan opsi. -ProfileName Jika konfigurasi penyedia token SSO Anda menggunakan profil bernama, perintahnya adalah Get-S3Bucket -ProfileName *named-profile*. Untuk informasi selengkapnya tentang profil bernama, lihat bagian [Profil](#) di Panduan Referensi Alat AWS SDKs dan Alat.

Informasi tambahan

- Untuk opsi lebih lanjut tentang otentikasi Alat untuk PowerShell, seperti penggunaan profil dan variabel lingkungan, lihat bagian [konfigurasi](#) di Panduan Referensi Alat AWS SDKs dan Alat.
- Beberapa perintah memerlukan AWS Region untuk ditentukan. Ada beberapa cara untuk melakukannya, termasuk opsi -Region cmdlet, [default] profil, dan variabel AWS_REGION lingkungan. Untuk informasi selengkapnya, lihat [Tentukan AWS Wilayah](#) di panduan ini dan [AWS Wilayah](#) di Panduan Referensi Alat AWS SDKs dan Alat.
- Untuk mempelajari lebih lanjut tentang praktik terbaik, lihat [Praktik terbaik keamanan di IAM](#) di Panduan Pengguna IAM.

- Untuk membuat AWS kredensial jangka pendek, lihat [Kredensial Keamanan Sementara di Panduan Pengguna](#) IAM.
- Untuk mempelajari tentang penyedia kredensi lainnya, lihat Penyedia kredensi [terstandarisasi di Panduan Referensi](#) Alat AWS SDKs dan Alat.

Tentukan AWS Wilayah

Ada dua cara untuk menentukan AWS Wilayah yang akan digunakan saat menjalankan Alat AWS untuk PowerShell perintah:

- Menggunakan parameter umum `-Region` pada perintah individu.
- Menggunakan perintah `Set-DefaultAWSRegion` untuk mengatur Wilayah default untuk semua perintah.

Banyak AWS cmdlet gagal jika Alat untuk Windows tidak PowerShell dapat mengetahui Wilayah apa yang akan digunakan. Pengecualian termasuk cmdlet untuk Amazon S3, [Amazon](#) SES, AWS Identity and Access Management dan, yang secara otomatis default ke titik akhir global.

Untuk menentukan wilayah untuk satu AWS perintah

Tambahkan parameter `-Region` ke perintah Anda, seperti ini.

```
PS > Get-EC2Image -Region us-west-2
```

Untuk mengatur wilayah default untuk semua perintah AWS CLI di sesi saat ini

Dari PowerShell command prompt, ketik perintah berikut.

```
PS > Set-DefaultAWSRegion -Region us-west-2
```

Note

Pengaturan ini tetap berlanjut hanya untuk sesi saat ini. Untuk menerapkan pengaturan ke semua PowerShell sesi Anda, tambahkan perintah ini ke PowerShell profil Anda seperti yang Anda lakukan untuk `Import-Module` perintah.

Untuk melihat wilayah default saat ini untuk semua perintah AWS CLI

Dari PowerShell command prompt, ketik perintah berikut.

```
PS > Get-DefaultAWSRegion
```

Region	Name	IsShellDefault
-----	----	-----
us-west-2	US West (Oregon)	True

Untuk menghapus Wilayah default saat ini untuk semua perintah AWS CLI

Dari PowerShell command prompt, ketik perintah berikut.

```
PS > Clear-DefaultAWSRegion
```

Untuk melihat daftar semua AWS Wilayah yang tersedia

Dari PowerShell command prompt, ketik perintah berikut. Kolom ketiga dalam output sampel mengidentifikasi Wilayah mana yang merupakan default untuk sesi Anda saat ini.

```
PS > Get-AWSRegion
```

Region	Name	IsShellDefault
-----	----	-----
ap-east-1	Asia Pacific (Hong Kong)	False
ap-northeast-1	Asia Pacific (Tokyo)	False
...		
us-east-2	US East (Ohio)	False
us-west-1	US West (N. California)	False
us-west-2	US West (Oregon)	True
...		

Note

Beberapa Wilayah mungkin didukung tetapi tidak termasuk dalam output dari cmdlet `Get-AWSRegion`. Misalnya, hal ini terkadang berlaku untuk Wilayah yang belum global. Jika Anda tidak dapat menentukan Wilayah dengan menambahkan parameter `-Region` ke perintah, coba sebutkan Wilayah di titik akhir kustom sebagai gantinya, seperti yang ditunjukkan di bagian berikut.

Mencantumkan Titik Akhir Kustom atau Tidak Standar

Tentukan titik akhir kustom sebagai URL dengan menambahkan parameter `-EndpointUrl` umum ke PowerShell perintah Tools for Windows Anda, dalam format contoh berikut.

```
PS > Some-AWS-PowerShellCmdlet -EndpointUrl "custom endpoint URL" -Other -Parameters
```

Berikut ini adalah sebuah contoh menggunakan cmdlet `Get-EC2Instance`. Titik akhir kustom ada di `us-west-2`, atau Wilayah US West (Oregon) dalam contoh ini, tetapi Anda dapat menggunakan Wilayah AWS yang didukung lainnya, termasuk wilayah yang tidak disebutkan oleh `Get-AWSRegion`.

```
PS > Get-EC2Instance -EndpointUrl "https://service-custom-url.us-west-2.amazonaws.com"
-InstanceID "i-0555a30a2000000e1"
```

Informasi tambahan

Untuk informasi tambahan tentang AWS Wilayah, lihat [AWS Wilayah](#) di Panduan Referensi AWS SDKs dan Alat.

Konfigurasi identitas federasi dengan Alat AWS untuk PowerShell

Agar pengguna di organisasi Anda dapat mengakses AWS sumber daya, Anda harus mengonfigurasi metode otentikasi standar dan berulang untuk tujuan keamanan, auditabilitas, kepatuhan, dan kemampuan untuk mendukung pemisahan peran dan akun. Meskipun umum untuk memberi pengguna kemampuan untuk mengakses AWS APIs, tanpa akses API federasi, Anda juga harus membuat pengguna AWS Identity and Access Management (IAM), yang mengalahkan tujuan menggunakan federasi. Topik ini menjelaskan dukungan SAM (Security Assertion Markup Language) Alat AWS untuk PowerShell yang memudahkan solusi akses federasi Anda.

Dukungan SAFL di Alat AWS untuk PowerShell memungkinkan Anda memberikan akses federasi kepada AWS pengguna Anda ke layanan. SAFL adalah format standar terbuka berbasis XML untuk mentransmisikan otentikasi pengguna dan data otorisasi antar layanan; khususnya, antara penyedia identitas (seperti Layanan [Federasi Direktori Aktif](#)), dan [penyedia layanan](#) (seperti). AWS Untuk informasi lebih lanjut tentang SAML dan cara kerjanya, lihat [SAML](#) di Wikipedia, atau [Spesifikasi](#)

[Teknis SAML](#) di situs web Organisasi untuk Kelanjutan Standar Informasi Terstruktur (OASIS). Dukungan SALL di Alat AWS untuk PowerShell kompatibel dengan SAFL 2.0.

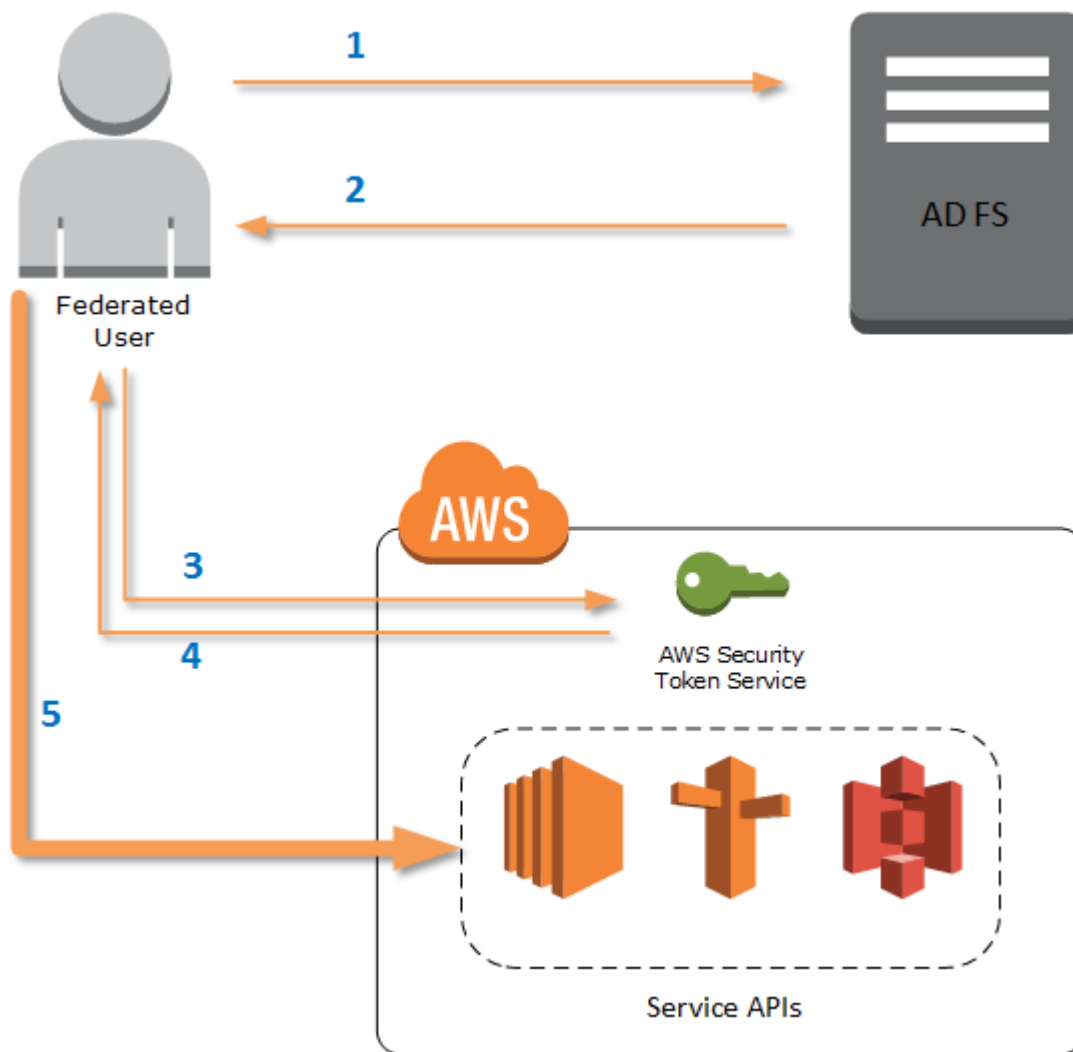
Prasyarat

Anda harus memiliki yang berikut sebelum Anda mencoba untuk menggunakan dukungan SAML untuk pertama kalinya.

- Solusi identitas federasi yang terintegrasi dengan benar dengan akun AWS Anda untuk akses konsol dengan menggunakan hanya kredensial organisasi Anda. Untuk informasi selengkapnya tentang cara melakukan ini secara khusus untuk Layanan Federasi Direktori Aktif, lihat [Tentang Federasi SAM 2.0](#) di Panduan Pengguna IAM, dan posting blog, [Mengaktifkan Federasi untuk AWS Menggunakan Windows Active Directory, AD FS, dan SAFL 2.0](#). Meskipun posting blog mencakup AD FS 2.0, langkah-langkahnya serupa jika Anda menjalankan AD FS 3.0.
- Versi 3.1.31.0 atau yang lebih baru Alat AWS untuk PowerShell diinstal pada workstation lokal Anda.

Bagaimana Pengguna Federasi Identitas Mendapat Akses Federasi ke Layanan AWS APIs

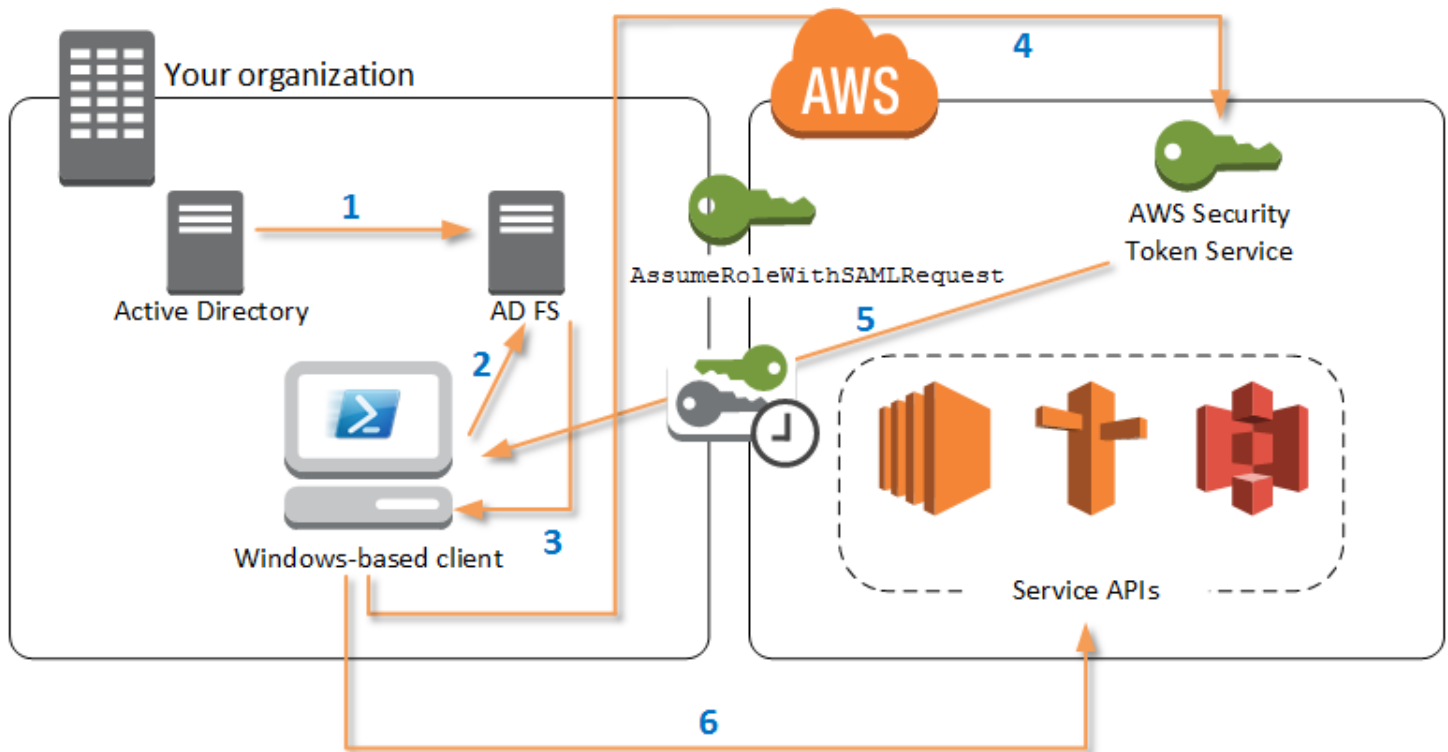
Proses berikut menjelaskan, pada tingkat tinggi, bagaimana pengguna Active Directory (AD) digabungkan oleh AD FS untuk mendapatkan akses ke AWS sumber daya.



1. Klien pada komputer pengguna gabungan mengotentikasi terhadap AD FS.
2. Jika autentikasi berhasil, AD FS mengirimkan pernyataan SAML ke pengguna.
3. Klien pengguna mengirimkan pernyataan SAFL ke AWS Security Token Service (STS) sebagai bagian dari permintaan federasi SAFL.
4. STS mengembalikan respons SAFL yang berisi AWS kredensial sementara untuk peran yang dapat diasumsikan pengguna.
5. Pengguna mengakses AWS layanan APIs dengan memasukkan kredensial sementara tersebut dalam permintaan yang dibuat oleh. Alat AWS untuk PowerShell

Cara Kerja SAML Support di Alat AWS untuk PowerShell

Bagian ini menjelaskan bagaimana Alat AWS untuk PowerShell cmdlet mengaktifkan konfigurasi federasi identitas berbasis SAML untuk pengguna.



1. Alat AWS untuk PowerShell mengautentikasi terhadap AD FS dengan menggunakan kredensial pengguna Windows saat ini, atau secara interaktif, saat pengguna mencoba menjalankan cmdlet yang memerlukan kredensial untuk dipanggil. AWS
2. AD FS mengotentikasi pengguna.
3. AD FS menghasilkan respons otentikasi SAM 2.0 yang mencakup pernyataan; tujuan dari pernyataan ini adalah untuk mengidentifikasi dan memberikan informasi tentang pengguna. Alat AWS untuk PowerShell mengekstrak daftar peran resmi pengguna dari pernyataan SAML.
4. Alat AWS untuk PowerShell meneruskan permintaan SAML, termasuk Amazon Resource Names (ARN) peran yang diminta, ke STS dengan melakukan panggilan API `AssumeRoleWithSAMLRequest`.
5. Jika permintaan SAML valid, STS mengembalikan jawaban yang berisi AWS AccessKeyId, SecretAccessKey, dan SessionToken. Kredensial ini berlangsung selama 3.600 detik (1 jam).
6. Pengguna sekarang memiliki kredensial yang valid untuk bekerja dengan AWS layanan apa pun APIs yang diizinkan untuk diakses oleh peran pengguna. Alat AWS untuk PowerShell

secara otomatis menerapkan kredensial ini untuk setiap panggilan AWS API berikutnya, dan memperbaruinya secara otomatis ketika mereka kedaluwarsa.

Note

Ketika kredensial berakhir, dan mandat baru diperlukan, Alat AWS untuk PowerShell secara otomatis mengotentikasi ulang dengan AD FS, dan memperoleh kredensial baru untuk jam berikutnya. Untuk pengguna dengan akun tergabung domain, proses ini terjadi secara diam-diam. Untuk akun yang tidak bergabung dengan domain, Alat AWS untuk PowerShell meminta pengguna untuk memasukkan kredensialnya sebelum mereka dapat mengotentikasi ulang.

Cara Menggunakan Cmdlet Konfigurasi PowerShell SAFL

Alat AWS untuk PowerShell termasuk dua cmdlet baru yang menyediakan dukungan SAFL.

- `Set-AWSSamlEndpoint` mengkonfigurasi titik akhir AD FS Anda, menetapkan nama yang mudah untuk titik akhir, dan dapat memilih menjelaskan jenis autentikasi titik akhir.
- `Set-AWSSamlRoleProfile` membuat atau mengedit profil akun pengguna yang ingin Anda kaitkan dengan titik akhir AD FS, diidentifikasi dengan menentukan nama yang mudah yang Anda berikan untuk cmdlet `Set-AWSSamlEndpoint`. Setiap profil peran memetakan peran tunggal yang dapat dilaksanakan secara sah oleh pengguna.

Sama seperti profil AWS kredensi, Anda menetapkan nama ramah ke profil peran. Anda dapat menggunakan nama ramah yang sama dengan `Set-AWSCredential` cmdlet, atau sebagai nilai `ProfileName` parameter untuk cmdlet apa pun yang memanggil layanan AWS APIs

Buka Alat AWS untuk PowerShell sesi baru. Jika Anda menjalankan PowerShell 3.0 atau yang lebih baru, Alat AWS untuk PowerShell modul secara otomatis diimpor saat Anda menjalankan salah satu cmdletnya. Jika Anda menjalankan PowerShell 2.0, Anda harus mengimpor modul secara manual dengan menjalankan cmdlet `Import-Module`, seperti yang ditunjukkan pada contoh berikut.

```
PS > Import-Module "C:\Program Files (x86)\AWS Tools\PowerShell\AWSPowerShell\AWSPowerShell.psd1"
```

Cara Menjalankan Cmdlet **Set-AWSSamlEndpoint** dan **Set-AWSSamlRoleProfile**

1. Pertama, konfigurasi pengaturan titik akhir untuk sistem AD FS. Cara termudah untuk melakukan hal ini adalah dengan menyimpan titik akhir dalam sebuah variabel, seperti yang ditunjukkan pada langkah ini. Pastikan untuk mengganti akun placeholder IDs dan nama host AD FS dengan akun Anda sendiri IDs dan nama host AD FS. Menentukan nama host AD FS di parameter Endpoint.

```
PS > $endpoint = "https://adfs.example.com/adfs/ls/IdpInitiatedSignOn.aspx?  
loginToRp=urn:amazon:webservices"
```

2. Untuk membuat pengaturan titik akhir, jalankan cmdlet **Set-AWSSamlEndpoint**, yang menyebutkan nilai yang benar untuk parameter **AuthenticationType**. Nilai yang valid termasuk Basic, Digest, Kerberos, Negotiate, dan NTLM. Jika Anda tidak menentukan parameter ini, maka nilai default adalah Kerberos.

```
PS > $sepName = Set-AWSSamlEndpoint -Endpoint $endpoint -StoreAs ADFS-Demo -  
AuthenticationType NTLM
```

Cmdlet mengembalikan nama mudah yang Anda tetapkan menggunakan parameter **-StoreAs**, sehingga Anda dapat menggunakannya ketika Anda menjalankan **Set-AWSSamlRoleProfile** di baris berikutnya.

3. Sekarang, jalankan cmdlet **Set-AWSSamlRoleProfile** untuk mengotentikasi dengan penyedia identitas AD FS dan mendapatkan set peran (dalam pernyataan SAML) yang dapat secara sah dilakukan oleh pengguna tersebut.

Cmdlet **Set-AWSSamlRoleProfile** menggunakan set peran yang dikembalikan untuk meminta pengguna untuk memilih peran untuk mengasosiasikan dengan profil tertentu, atau memvalidasi bahwa data peran yang disediakan dalam parameter sudah ada (jika tidak, pengguna diminta untuk memilih). Jika pengguna diizinkan hanya untuk satu peran, cmdlet akan mengaitkan peran tersebut dengan profil secara otomatis, tanpa meminta pengguna. Tidak perlu memberikan kredensial untuk mengatur profil untuk penggunaan tergabung domain.

```
PS > Set-AWSSamlRoleProfile -StoreAs SAMLDemoProfile -EndpointName $sepName
```

Atau, untuk non-domain-joined akun, Anda dapat memberikan kredensi Direktori Aktif, lalu pilih AWS peran yang dapat diakses pengguna, seperti yang ditunjukkan pada baris berikut. Ini berguna jika Anda memiliki akun pengguna Direktori Aktif yang berbeda untuk membedakan peran dalam organisasi Anda (misalnya, fungsi administrasi).

```
PS > $credential = Get-Credential -Message "Enter the domain credentials for the endpoint"
PS > Set-AWSSamlRoleProfile -EndpointName $epName -NetworkCredential $credential -StoreAs SAMLDemoProfile
```

4. Dalam kedua kasus, cmdlet `Set-AWSSamlRoleProfile` meminta Anda untuk memilih peran mana yang harus disimpan dalam profil. Contoh berikut menunjukkan dua peran yang tersedia: ADFS-Dev, dan ADFS-Production. Peran IAM terkait dengan kredensial masuk AD Anda oleh administrator AD FS.

```
Select Role
Select the role to be assumed when this profile is active
[1] 1 - ADFS-Dev [2] 2 - ADFS-Production [?] Help (default is "1"):
```

Pilihan lainnya, Anda dapat menentukan peran tanpa permintaan tersebut, dengan memasukkan parameter `RoleARN`, `PrincipalARN`, atau memilih `NetworkCredential`. Jika peran yang ditentukan tidak tercantum dalam pernyataan yang dikembalikan oleh autentikasi, pengguna diminta untuk memilih dari peran yang tersedia.

```
PS > $params = @{ "NetworkCredential"=$credential,
  "PrincipalARN"="{arn:aws:iam::012345678912:saml-provider/ADFS}",
  "RoleARN"="{arn:aws:iam::012345678912:role/ADFS-Dev}"
}
PS > $epName | Set-AWSSamlRoleProfile @params -StoreAs SAMLDemoProfile1 -Verbose
```

5. Anda dapat membuat profil untuk semua peran dalam satu perintah dengan menambahkan parameter `StoreAllRoles`, seperti yang ditunjukkan dalam kode berikut. Perhatikan bahwa nama peran digunakan sebagai nama profil.

```
PS > Set-AWSSamlRoleProfile -EndpointName $epName -StoreAllRoles
ADFS-Dev
ADFS-Production
```

Cara Menggunakan Profil Peran untuk Menjalankan Cmdlet yang Memerlukan Kredensial AWS

Untuk menjalankan cmdlet yang memerlukan AWS kredensial, Anda dapat menggunakan profil peran yang ditentukan dalam file kredensi AWS bersama. Berikan nama profil peran ke `Set-AWSCredential` (atau sebagai nilai untuk `ProfileName` parameter apa pun di Alat AWS untuk PowerShell) untuk mendapatkan AWS kredensial sementara secara otomatis untuk peran yang dijelaskan dalam profil.

Meskipun Anda menggunakan hanya satu profil peran pada satu waktu, Anda dapat beralih profil dalam sesi shell. Cmdlet `Set-AWSCredential` tidak mengotentikasi dan mendapatkan kredensial saat Anda menjalankannya sendiri; catatan cmdlet mencatat bahwa Anda ingin menggunakan profil peran tertentu. Sampai Anda menjalankan cmdlet yang memerlukan kredensial AWS, tidak ada autentikasi atau permintaan untuk kredensial yang terjadi.

Anda sekarang dapat menggunakan AWS kredensial sementara yang Anda peroleh dengan `SAMLDemoProfile` profil untuk bekerja dengan AWS layanan. APIs Bagian berikut menunjukkan contoh cara menggunakan profil peran.

Contoh 1: Mengatur Peran Default dengan **Set-AWSCredential**

Contoh ini menetapkan peran default untuk Alat AWS untuk PowerShell sesi dengan menggunakan `Set-AWSCredential`. Kemudian, Anda dapat menjalankan cmdlet yang memerlukan kredensial, dan diizinkan oleh peran yang ditentukan. Contoh ini berisi daftar semua instans Amazon Elastic Compute Cloud di Wilayah US West (Oregon) yang terkait dengan profil yang Anda tentukan dengan cmdlet `Set-AWSCredential`.

```
PS > Set-AWSCredential -ProfileName SAMLDemoProfile
PS > Get-EC2Instance -Region us-west-2 | Format-Table -Property Instances,GroupNames
```

Instances	GroupNames
-----	-----
{TestInstance1}	{default}
{TestInstance2}	{}
{TestInstance3}	{launch-wizard-6}
{TestInstance4}	{default}
{TestInstance5}	{}
{TestInstance6}	{AWS-OpsWorks-Default-
Server}	

Contoh 2: Ubah Profil Peran Selama PowerShell Sesi

Contoh ini mencantumkan semua bucket Amazon S3 yang tersedia di AWS akun peran yang terkait dengan profil. `SAMLDemoProfile` Contoh menunjukkan bahwa meskipun Anda mungkin telah menggunakan profil lain di awal Alat AWS untuk PowerShell sesi Anda, Anda dapat mengubah profil dengan menentukan nilai yang berbeda untuk `-ProfileName` parameter dengan cmdlet yang mendukungnya. Ini adalah tugas umum bagi administrator yang mengelola Amazon S3 dari baris perintah PowerShell .

```
PS > Get-S3Bucket -ProfileName SAMLDemoProfile
```

CreationDate	BucketName
-----	-----
7/25/2013 3:16:56 AM	<i>amzn-s3-demo-bucket</i>
4/15/2015 12:46:50 AM	<i>amzn-s3-demo-bucket1</i>
4/15/2015 6:15:53 AM	<i>amzn-s3-demo-bucket2</i>
1/12/2015 11:20:16 PM	<i>amzn-s3-demo-bucket3</i>

Perhatikan bahwa cmdlet `Get-S3Bucket` menentukan nama profil yang dibuat dengan menjalankan cmdlet `Set-AWSSamlRoleProfile`. Perintah ini dapat berguna jika Anda telah menetapkan profil peran sebelumnya dalam sesi Anda (misalnya, dengan menjalankan cmdlet `Set-AWSCredential`) dan ingin menggunakan profil peran yang berbeda untuk cmdlet `Get-S3Bucket`. Manajer profil menyediakan kredensial sementara untuk cmdlet `Get-S3Bucket`.

Meskipun kredensialnya kedaluwarsa setelah 1 jam (batas yang diberlakukan oleh STS), Alat AWS untuk PowerShell secara otomatis menyegarkan kredensial dengan meminta pernyataan SAML baru ketika alat mendeteksi bahwa kredensial saat ini telah kedaluwarsa.

Untuk pengguna tergabung domain, proses ini terjadi tanpa gangguan, karena identitas Windows pengguna saat ini digunakan selama autentikasi. Untuk akun non-domain-joined pengguna, Alat AWS untuk PowerShell menampilkan prompt PowerShell kredensial yang meminta kata sandi pengguna. Pengguna menyediakan kredensial yang digunakan untuk melakukan autentikasi ulang terhadap pengguna dan mendapatkan pernyataan baru.

Contoh 3: Mendapatkan Instans di suatu Wilayah

Contoh berikut mencantumkan semua EC2 instans Amazon di Wilayah Asia Pasifik (Sydney) yang terkait dengan akun yang digunakan oleh `ADFS-Production` profil. Ini adalah perintah yang berguna untuk mengembalikan semua EC2 instance Amazon di suatu wilayah.

```
PS > (Get-Ec2Instance -ProfileName ADFS-Production -Region ap-southeast-2).Instances |
Select InstanceType, @{Name="Servername";Expression={$_.tags | where key -eq "Name" |
Select Value -Expand Value}}
```

InstanceType	Servername
-----	-----
t2.small	DC2
t1.micro	NAT1
t1.micro	RDGW1
t1.micro	RDGW2
t1.micro	NAT2
t2.small	DC1
t2.micro	BUILD

Bacaan Tambahan

Untuk informasi umum tentang cara menerapkan akses API tergabung, lihat [Cara Menerapkan Solusi Umum untuk Akses API/CLI Federasi Menggunakan SAML 2.0](#).

Untuk pertanyaan dukungan atau komentar, kunjungi Forum AWS Pengembang untuk [PowerShell Scripting](#) atau [.NET Development](#).

Penemuan dan alias Cmdlet

Bagian ini menunjukkan kepada Anda cara membuat daftar layanan yang didukung oleh Alat AWS untuk PowerShell, cara menampilkan kumpulan cmdlet yang disediakan oleh dukungan layanan tersebut, dan cara menemukan nama cmdlet alternatif (juga disebut alias) untuk mengakses layanan tersebut. Alat AWS untuk PowerShell

Penemuan Cmdlet

Semua operasi AWS layanan (atau APIs) didokumentasikan dalam Panduan Referensi API untuk setiap layanan. Misalnya, lihat bagian [Referensi API IAM](#). Dalam banyak kasus, ada one-to-one korespondensi antara API AWS layanan dan AWS PowerShell cmdlet. Untuk mendapatkan nama cmdlet yang sesuai dengan nama API AWS layanan, jalankan `Get-AWSCmdletName` cmdlet dengan `-ApiOperation` parameter dan nama API layanan. AWS Misalnya, untuk mendapatkan semua kemungkinan nama cmdlet yang didasarkan pada API `DescribeInstances` AWS layanan apa pun yang tersedia, jalankan perintah berikut:

```
PS > Get-AWSCmdletName -ApiOperation DescribeInstances
```

CmdletName	ServiceOperation	ServiceName	CmdletNounPrefix
-----	-----	-----	-----
Get-EC2Instance	DescribeInstances	Amazon Elastic Compute Cloud	EC2
Get-GMLInstance	DescribeInstances	Amazon GameLift Service	GML

Parameter `-ApiOperation` merupakan parameter default, sehingga Anda dapat menghilangkan nama parameter. Contoh berikut setara dengan yang sebelumnya:

```
PS > Get-AWSCmdletName DescribeInstances
```

Jika Anda mengetahui nama API dan layanan, Anda dapat menyertakan `-Service` parameter bersama dengan awalan kata benda cmdlet atau bagian dari nama layanan. AWS Misalnya, awalan kata benda cmdlet untuk Amazon adalah. EC2 Untuk mendapatkan nama cmdlet yang sesuai dengan `DescribeInstances` API di EC2 layanan Amazon, jalankan salah satu perintah berikut. Semuanya menghasilkan output yang sama:

```
PS > Get-AWSCmdletName -ApiOperation DescribeInstances -Service EC2
PS > Get-AWSCmdletName -ApiOperation DescribeInstances -Service Compute
PS > Get-AWSCmdletName -ApiOperation DescribeInstances -Service "Compute Cloud"
```

CmdletName	ServiceOperation	ServiceName	CmdletNounPrefix
-----	-----	-----	-----
Get-EC2Instance	DescribeInstances	Amazon Elastic Compute Cloud	EC2

Nilai parameter dalam perintah ini peka huruf besar-kecil.

Jika Anda tidak tahu nama API AWS layanan yang diinginkan atau AWS layanan, Anda dapat menggunakan `-ApiOperation` parameter, bersama dengan pola yang cocok, dan `-MatchWithRegex` parameter. Misalnya, untuk mendapatkan semua nama cmdlet tersedia yang berisi `SecurityGroup`, jalankan perintah berikut:

```
PS > Get-AWSCmdletName -ApiOperation SecurityGroup -MatchWithRegex
```

CmdletName	ServiceOperation
ServiceName	CmdletNounPrefix
-----	-----
-----	-----
Approve-ECCacheSecurityGroupIngress	AuthorizeCacheSecurityGroupIngress
Amazon ElastiCache	EC

Get-ECCacheSecurityGroup		DescribeCacheSecurityGroups	
Amazon ElastiCache	EC		
New-ECCacheSecurityGroup		CreateCacheSecurityGroup	
Amazon ElastiCache	EC		
Remove-ECCacheSecurityGroup		DeleteCacheSecurityGroup	
Amazon ElastiCache	EC		
Revoke-ECCacheSecurityGroupIngress		RevokeCacheSecurityGroupIngress	
Amazon ElastiCache	EC		
Add-EC2SecurityGroupToClientVpnTargetNetwork			
ApplySecurityGroupsToClientVpnTargetNetwork	Amazon Elastic Compute Cloud		EC2
Get-EC2SecurityGroup		DescribeSecurityGroups	
Amazon Elastic Compute Cloud	EC2		
Get-EC2SecurityGroupReference		DescribeSecurityGroupReferences	
Amazon Elastic Compute Cloud	EC2		
Get-EC2StaleSecurityGroup		DescribeStaleSecurityGroups	
Amazon Elastic Compute Cloud	EC2		
Grant-EC2SecurityGroupEgress		AuthorizeSecurityGroupEgress	
Amazon Elastic Compute Cloud	EC2		
Grant-EC2SecurityGroupIngress		AuthorizeSecurityGroupIngress	
Amazon Elastic Compute Cloud	EC2		
New-EC2SecurityGroup		CreateSecurityGroup	
Amazon Elastic Compute Cloud	EC2		
Remove-EC2SecurityGroup		DeleteSecurityGroup	
Amazon Elastic Compute Cloud	EC2		
Revoke-EC2SecurityGroupEgress		RevokeSecurityGroupEgress	
Amazon Elastic Compute Cloud	EC2		
Revoke-EC2SecurityGroupIngress		RevokeSecurityGroupIngress	
Amazon Elastic Compute Cloud	EC2		
Update-EC2SecurityGroupRuleEgressDescription		UpdateSecurityGroupRuleDescriptionsEgress	
Amazon Elastic Compute Cloud	EC2		
Update-EC2SecurityGroupRuleIngressDescription			
UpdateSecurityGroupRuleDescriptionsIngress	Amazon Elastic Compute Cloud		EC2
Edit-EFSMountTargetSecurityGroup		ModifyMountTargetSecurityGroups	
Amazon Elastic File System	EFS		
Get-EFSMountTargetSecurityGroup		DescribeMountTargetSecurityGroups	
Amazon Elastic File System	EFS		
Join-ELBSecurityGroupToLoadBalancer		ApplySecurityGroupsToLoadBalancer	
Elastic Load Balancing	ELB		
Set-ELB2SecurityGroup		SetSecurityGroups	
Elastic Load Balancing V2	ELB2		
Enable-RDSDBSecurityGroupIngress		AuthorizeDBSecurityGroupIngress	
Amazon Relational Database Service RDS			
Get-RDSDBSecurityGroup		DescribeDBSecurityGroups	
Amazon Relational Database Service RDS			

New-RDSDBSecurityGroup		CreateDBSecurityGroup
Amazon Relational Database Service RDS		
Remove-RDSDBSecurityGroup		DeleteDBSecurityGroup
Amazon Relational Database Service RDS		
Revoke-RDSDBSecurityGroupIngress		RevokeDBSecurityGroupIngress
Amazon Relational Database Service RDS		
Approve-RSClusterSecurityGroupIngress		AuthorizeClusterSecurityGroupIngress
Amazon Redshift	RS	
Get-RSClusterSecurityGroup		DescribeClusterSecurityGroups
Amazon Redshift	RS	
New-RSClusterSecurityGroup		CreateClusterSecurityGroup
Amazon Redshift	RS	
Remove-RSClusterSecurityGroup		DeleteClusterSecurityGroup
Amazon Redshift	RS	
Revoke-RSClusterSecurityGroupIngress		RevokeClusterSecurityGroupIngress
Amazon Redshift	RS	

Jika Anda mengetahui nama AWS layanan tetapi bukan API AWS layanan, sertakan -MatchWithRegex parameter dan -Service parameter untuk cakupan pencarian hingga satu layanan. Misalnya, untuk mendapatkan semua nama cmdlet yang hanya berisi SecurityGroup EC2 layanan Amazon, jalankan perintah berikut

```
PS > Get-AWSCmdletName -ApiOperation SecurityGroup -MatchWithRegex -Service EC2
```

CmdletName	ServiceOperation
ServiceName	CmdletNounPrefix
-----	-----
-----	-----
Add-EC2SecurityGroupToClientVpnTargetNetwrk	
ApplySecurityGroupsToClientVpnTargetNetwork	Amazon Elastic Compute Cloud EC2
Get-EC2SecurityGroup	DescribeSecurityGroups
Amazon Elastic Compute Cloud EC2	
Get-EC2SecurityGroupReference	DescribeSecurityGroupReferences
Amazon Elastic Compute Cloud EC2	
Get-EC2StaleSecurityGroup	DescribeStaleSecurityGroups
Amazon Elastic Compute Cloud EC2	
Grant-EC2SecurityGroupEgress	AuthorizeSecurityGroupEgress
Amazon Elastic Compute Cloud EC2	
Grant-EC2SecurityGroupIngress	AuthorizeSecurityGroupIngress
Amazon Elastic Compute Cloud EC2	
New-EC2SecurityGroup	CreateSecurityGroup
Amazon Elastic Compute Cloud EC2	

Remove-EC2SecurityGroup	DeleteSecurityGroup
Amazon Elastic Compute Cloud EC2	
Revoke-EC2SecurityGroupEgress	RevokeSecurityGroupEgress
Amazon Elastic Compute Cloud EC2	
Revoke-EC2SecurityGroupIngress	RevokeSecurityGroupIngress
Amazon Elastic Compute Cloud EC2	
Update-EC2SecurityGroupRuleEgressDescription	UpdateSecurityGroupRuleDescriptionsEgress
Amazon Elastic Compute Cloud EC2	
Update-EC2SecurityGroupRuleIngressDescription	
UpdateSecurityGroupRuleDescriptionsIngress	Amazon Elastic Compute Cloud EC2

Jika Anda mengetahui nama perintah AWS Command Line Interface (AWS CLI), Anda dapat menggunakan `-AwsCliCommand` parameter dan nama AWS CLI perintah yang diinginkan untuk mendapatkan nama cmdlet yang didasarkan pada API yang sama. Misalnya, untuk mendapatkan nama cmdlet yang sesuai dengan panggilan `authorize-security-group-ingress` AWS CLI perintah di EC2 layanan Amazon, jalankan perintah berikut:

```
PS > Get-AWSCmdletName -AwsCliCommand "aws ec2 authorize-security-group-ingress"
```

CmdletName	ServiceOperation	ServiceName
CmdletNounPrefix		
-----	-----	-----

Grant-EC2SecurityGroupIngress	AuthorizeSecurityGroupIngress	Amazon Elastic Compute Cloud EC2

`Get-AWSCmdletNameCmdlet` hanya membutuhkan cukup nama AWS CLI perintah untuk mengidentifikasi layanan dan API. AWS

Untuk mendapatkan daftar semua cmdlet di Tools for PowerShell Core, jalankan PowerShell `Get-Command cmdlet`, seperti yang ditunjukkan pada contoh berikut.

```
PS > Get-Command -Module AWSPowerShell.NetCore
```

Anda dapat menjalankan perintah yang sama dengan `-Module AWSPowerShell` untuk melihat cmdlet di AWS Tools for Windows PowerShell.

Cmdlet `Get-Command` menghasilkan daftar cmdlet sesuai urutan abjad. Perhatikan bahwa secara default daftar diurutkan berdasarkan PowerShell kata kerja, bukan PowerShell kata benda.

Untuk mengurutkan hasil berdasarkan layanan, jalankan perintah berikut:

```
PS > Get-Command -Module AWSPowerShell.NetCore | Sort-Object Noun,Verb
```

Untuk menyaring cmdlet yang dikembalikan oleh `Get-Command` cmdlet, pipa output ke cmdlet. PowerShell `Select-String` Misalnya, untuk melihat kumpulan cmdlet yang berfungsi dengan AWS region, jalankan perintah berikut:

```
PS > Get-Command -Module AWSPowerShell.NetCore | Select-String region
```

```
Clear-DefaultAWSRegion
Copy-HSM2BackupToRegion
Get-AWSRegion
Get-DefaultAWSRegion
Get-EC2Region
Get-LSRegionList
Get-RDSSourceRegion
Set-DefaultAWSRegion
```

Anda juga dapat menemukan cmdlet untuk layanan tertentu dengan menyaring prefiks layanan kata benda cmdlet. Untuk melihat daftar prefiks layanan yang tersedia, jalankan `Get-AWSPowerShellVersion -ListServiceVersionInfo`. Contoh berikut mengembalikan cmdlet yang mendukung layanan Amazon CloudWatch Events.

```
PS > Get-Command -Module AWSPowerShell -Noun CWE*
```

CommandType	Name	Version	Source
-----	----	-----	-----
Cmdlet	Add-CWResourceTag	3.3.563.1	
	AWSPowerShell.NetCore		
Cmdlet	Disable-CWEventSource	3.3.563.1	
	AWSPowerShell.NetCore		
Cmdlet	Disable-CWERule	3.3.563.1	
	AWSPowerShell.NetCore		
Cmdlet	Enable-CWEventSource	3.3.563.1	
	AWSPowerShell.NetCore		
Cmdlet	Enable-CWERule	3.3.563.1	
	AWSPowerShell.NetCore		
Cmdlet	Get-CWEventBus	3.3.563.1	
	AWSPowerShell.NetCore		
Cmdlet	Get-CWEventBusList	3.3.563.1	
	AWSPowerShell.NetCore		

Cmdlet	Get-CWEEventSource	3.3.563.1
	AWSPowerShell.NetCore	
Cmdlet	Get-CWEEventSourceList	3.3.563.1
	AWSPowerShell.NetCore	
Cmdlet	Get-CWEPartnerEventSource	3.3.563.1
	AWSPowerShell.NetCore	
Cmdlet	Get-CWEPartnerEventSourceAccountList	3.3.563.1
	AWSPowerShell.NetCore	
Cmdlet	Get-CWEPartnerEventSourceList	3.3.563.1
	AWSPowerShell.NetCore	
Cmdlet	Get-CWEResourceTag	3.3.563.1
	AWSPowerShell.NetCore	
Cmdlet	Get-CWERule	3.3.563.1
	AWSPowerShell.NetCore	
Cmdlet	Get-CWERuleDetail	3.3.563.1
	AWSPowerShell.NetCore	
Cmdlet	Get-CWERuleNamesByTarget	3.3.563.1
	AWSPowerShell.NetCore	
Cmdlet	Get-CWETargetsByRule	3.3.563.1
	AWSPowerShell.NetCore	
Cmdlet	New-CWEEventBus	3.3.563.1
	AWSPowerShell.NetCore	
Cmdlet	New-CWEPartnerEventSource	3.3.563.1
	AWSPowerShell.NetCore	
Cmdlet	Remove-CWEEventBus	3.3.563.1
	AWSPowerShell.NetCore	
Cmdlet	Remove-CWEPartnerEventSource	3.3.563.1
	AWSPowerShell.NetCore	
Cmdlet	Remove-CWEPermission	3.3.563.1
	AWSPowerShell.NetCore	
Cmdlet	Remove-CWEResourceTag	3.3.563.1
	AWSPowerShell.NetCore	
Cmdlet	Remove-CWERule	3.3.563.1
	AWSPowerShell.NetCore	
Cmdlet	Remove-CWETarget	3.3.563.1
	AWSPowerShell.NetCore	
Cmdlet	Test-CWEEventPattern	3.3.563.1
	AWSPowerShell.NetCore	
Cmdlet	Write-CWEEvent	3.3.563.1
	AWSPowerShell.NetCore	
Cmdlet	Write-CWEPartnerEvent	3.3.563.1
	AWSPowerShell.NetCore	
Cmdlet	Write-CWEPermission	3.3.563.1
	AWSPowerShell.NetCore	

Cmdlet	Write-CWRule	3.3.563.1
	AWSPowerShell.NetCore	
Cmdlet	Write-CWETarget	3.3.563.1
	AWSPowerShell.NetCore	

Penamaan dan Alias Cmdlet

Cmdlet di Alat AWS untuk PowerShell untuk setiap layanan didasarkan pada metode yang disediakan oleh AWS SDK untuk layanan. Namun, karena konvensi penamaan wajib, nama cmdlet mungkin berbeda dari nama panggilan API atau metode yang menjadi dasarnya. PowerShell Misalnya, `Get-EC2Instance` cmdlet didasarkan pada metode `Amazon EC2DescribeInstances`.

Dalam beberapa kasus, nama cmdlet mungkin mirip dengan nama metode, tetapi keduanya mungkin sebenarnya melakukan fungsi yang berbeda. Misalnya, metode `GetObjectAmazon S3` mengambil objek Amazon S3. Namun, cmdlet `Get-S3Object` mengembalikan Informasi tentang objek Amazon S3 dan bukan objek itu sendiri.

```
PS > Get-S3Object -BucketName text-content -Key aws-tech-docs
```

```
ETag          : "df000002a0fe0000f3c000004EXAMPLE"
BucketName    : aws-tech-docs
Key           : javascript/frameset.js
LastModified  : 6/13/2011 1:24:18 PM
Owner         : Amazon.S3.Model.Owner
Size          : 512
StorageClass  : STANDARD
```

Untuk mendapatkan objek S3 dengan Alat AWS untuk PowerShell, jalankan `Read-S3Object` cmdlet:

```
PS > Read-S3Object -BucketName text-content -Key text-object.txt -file c:\tmp\text-object-download.txt
```

```
Mode                LastWriteTime         Length Name
----                -
-a---             11/5/2012   7:29 PM       20622 text-object-download.txt
```

 Note

Bantuan cmdlet untuk AWS cmdlet memberikan nama API AWS SDK yang menjadi dasar cmdlet.

Untuk informasi selengkapnya tentang PowerShell kata kerja standar dan artinya, lihat [Kata Kerja yang Disetujui untuk PowerShell](#) Perintah.

Semua AWS cmdlet yang menggunakan Remove kata kerja — dan Stop-EC2Instance cmdlet saat Anda menambahkan -Terminate parameter — meminta konfirmasi sebelum melanjutkan. Untuk memotong konfirmasi, tambahkan parameter -Force ke perintah Anda.

 Important

AWS cmdlet tidak mendukung sakelar. -WhatIf

Alias

Pengaturan Alat AWS untuk PowerShell menginstal file alias yang berisi alias untuk banyak cmdlet. AWS Alias ini mungkin lebih intuitif dibandingkan nama cmdlet. Misalnya, nama layanan dan nama metode AWS SDK menggantikan PowerShell kata kerja dan kata benda di beberapa alias. Sebagai contoh adalah alias EC2-DescribeInstances.

Alias lain menggunakan kata kerja yang, meskipun tidak mengikuti PowerShell konvensi standar, dapat lebih deskriptif dari operasi yang sebenarnya. Misalnya, file alias memetakan alias Get-S3Content ke cmdlet Read-S3Object.

```
PS > Set-Alias -Name Get-S3Content -Value Read-S3Object
```

File alias terletak di direktori Alat AWS untuk PowerShell instalasi. Untuk memuat alias ke lingkungan Anda, dot-souce file. Berikut ini adalah contoh berbasis Windows.

```
PS > . "C:\Program Files (x86)\AWS Tools\PowerShell\AWSPowershell\AWSAliases.ps1"
```

Untuk shell Linux atau macOS, mungkin akan terlihat seperti ini:

```
. ~/.local/share/powershell/Modules/AWSPowerShell.NetCore/3.3.563.1/AWSAliases.ps1
```

Untuk menampilkan semua Alat AWS untuk PowerShell alias, jalankan perintah berikut. Perintah ini menggunakan ? alias untuk PowerShell Where-Object cmdlet dan Source properti untuk memfilter hanya alias yang berasal dari modul. `AWSPowerShell.NetCore`

```
PS > Get-Alias | ? Source -like "AWSPowerShell.NetCore"
```

CommandType	Name	Version	Source
-----	----	-----	-----
Alias	Add-ASInstances	3.3.343.0	
AWSPowerShell			
Alias	Add-CTTag	3.3.343.0	
AWSPowerShell			
Alias	Add-DPTags	3.3.343.0	
AWSPowerShell			
Alias	Add-DSIpRoutes	3.3.343.0	
AWSPowerShell			
Alias	Add-ELBTags	3.3.343.0	
AWSPowerShell			
Alias	Add-EMRTag	3.3.343.0	
AWSPowerShell			
Alias	Add-ESTag	3.3.343.0	
AWSPowerShell			
Alias	Add-MLTag	3.3.343.0	
AWSPowerShell			
Alias	Clear-AWSCredentials	3.3.343.0	
AWSPowerShell			
Alias	Clear-AWSDefaults	3.3.343.0	
AWSPowerShell			
Alias	Dismount-ASInstances	3.3.343.0	
AWSPowerShell			
Alias	Edit-EC2Hosts	3.3.343.0	
AWSPowerShell			
Alias	Edit-RSClusterIamRoles	3.3.343.0	
AWSPowerShell			
Alias	Enable-ORGAllFeatures	3.3.343.0	
AWSPowerShell			
Alias	Find-CTEvents	3.3.343.0	
AWSPowerShell			
Alias	Get-ASACases	3.3.343.0	
AWSPowerShell			
Alias	Get-ASAccountLimits	3.3.343.0	
AWSPowerShell			

Alias AWSPowerShell	Get-ASACommunications	3.3.343.0
Alias AWSPowerShell	Get-ASAServices	3.3.343.0
Alias AWSPowerShell	Get-ASASeverityLevels	3.3.343.0
Alias AWSPowerShell	Get-ASATrustedAdvisorCheckRefreshStatuses	3.3.343.0
Alias AWSPowerShell	Get-ASATrustedAdvisorChecks	3.3.343.0
Alias AWSPowerShell	Get-ASATrustedAdvisorCheckSummaries	3.3.343.0
Alias AWSPowerShell	Get-ASLifecycleHooks	3.3.343.0
Alias AWSPowerShell	Get-ASLifecycleHookTypes	3.3.343.0
Alias AWSPowerShell	Get-AWSCredentials	3.3.343.0
Alias AWSPowerShell	Get-CDApplications	3.3.343.0
Alias AWSPowerShell	Get-CDDeployments	3.3.343.0
Alias AWSPowerShell	Get-CFCloudFrontOriginAccessIdentities	3.3.343.0
Alias AWSPowerShell	Get-CFDistributions	3.3.343.0
Alias AWSPowerShell	Get-CFGConfigRules	3.3.343.0
Alias AWSPowerShell	Get-CFGConfigurationRecorders	3.3.343.0
Alias AWSPowerShell	Get-CFGDeliveryChannels	3.3.343.0
Alias AWSPowerShell	Get-CFInvalidations	3.3.343.0
Alias AWSPowerShell	Get-CFNAccountLimits	3.3.343.0
Alias AWSPowerShell	Get-CFNStackEvents	3.3.343.0
...		

Untuk menambahkan alias Anda sendiri ke file ini, Anda mungkin perlu menaikkan nilai [variabel \\$MaximumAliasCount preferensi](#) ke nilai yang lebih besar dari 5500. PowerShell Nilai default

adalah 4096; Anda dapat meningkatkan hingga maksimum 32768. Untuk melakukannya, jalankan yang berikut ini.

```
PS > $MaximumAliasCount = 32768
```

Untuk memverifikasi bahwa perubahan Anda berhasil, masukkan nama variabel untuk menunjukkan nilai saat ini.

```
PS > $MaximumAliasCount  
32768
```

Pipelining, output, dan iterasi di Alat AWS untuk PowerShell

Pemipaan pipa

PowerShell mendorong pengguna untuk menghubungkan cmdlet ke [saluran pipa](#) yang mengarahkan output dari satu cmdlet ke input berikutnya. Contoh berikut menunjukkan perilaku ini saat menggunakan Alat AWS untuk PowerShell. Perintah mendapatkan dan kemudian menghentikan semua EC2 instance Amazon di Wilayah default saat ini.

```
PS > Get-EC2Instance | Stop-EC2Instance
```

Keluaran cmdlet

Untuk mendukung pipelining dengan lebih baik, beberapa data dari tanggapan AWS SDK untuk .NET mungkin dibuang secara default. Output dari Alat AWS untuk PowerShell cmdlet tidak dibentuk ulang untuk menyertakan respons layanan dan instance hasil sebagai Note properti pada objek koleksi yang dipancarkan. Sebaliknya, untuk panggilan yang memancarkan satu koleksi sebagai output, koleksi sekarang disebutkan ke pipeline. PowerShell Ini berarti bahwa respons SDK dan data hasil tidak dapat ada di pipeline karena tidak ada objek koleksi yang dapat dilampirkan.

Meskipun sebagian besar pengguna mungkin tidak memerlukan data ini, ini dapat berguna untuk tujuan diagnostik karena Anda dapat melihat dengan tepat apa yang dikirim dan diterima dari panggilan AWS layanan yang mendasari yang dibuat oleh cmdlet. Dimulai dengan Alat AWS untuk PowerShell V4, cmdlet dapat menggunakan `-Select *` parameter dan argumen untuk mengembalikan seluruh respons layanan.

Note

Dalam versi Alat AWS untuk PowerShell sebelum V4, variabel sesi yang disebut `$AWSHistory` diperkenalkan yang menyimpan catatan pemanggilan AWS cmdlet dan tanggapan layanan yang diterima untuk setiap pemanggilan. Di V4 dari Tools for PowerShell, variabel sesi ini tidak digunakan lagi demi `-Select *` parameter dan argumen, yang dapat digunakan untuk mengembalikan seluruh respons layanan. Parameter ini dijelaskan dalam topik ini.

Note

Ini adalah dokumentasi prarilis untuk fitur dalam rilis pratinjau. Dokumentasi ini dapat berubah.

`$AWSHistory` Variabel akan dihapus di V5 dari file. Alat AWS untuk PowerShell Untuk informasi lebih lanjut, lihat posting blog [Pemberitahuan versi utama 5 AWS Alat yang akan datang untuk PowerShell](#).

Untuk mengilustrasikan bagaimana semua data dari respons dapat dikembalikan, pertimbangkan contoh berikut.

Contoh pertama hanya mengembalikan daftar ember Amazon S3. Ini adalah perilaku default.

```
PS > Get-S3Bucket
```

CreationDate	BucketName
-----	-----
9/22/2023 10:54:35 PM	amzn-s3-demo-bucket1
9/22/2023 11:04:37 AM	amzn-s3-demo-bucket2
9/22/2023 12:54:34 PM	amzn-s3-demo-bucket3

Contoh kedua mengembalikan objek AWS SDK untuk .NET respon. Karena `-Select *` telah ditentukan, output mencakup seluruh respons API, yang berisi kumpulan bucket di `Buckets` properti. Dalam contoh ini, `Format-List` cmdlet tidak sepenuhnya diperlukan, tetapi hadir untuk memastikan bahwa semua properti ditampilkan.

```
PS > Get-S3Bucket -Select * | Format-List
```

```
LoggedAt           : 10/1/2023 9:45:52 AM
Buckets            : {amzn-s3-demo-bucket1, amzn-s3-demo-bucket2,
                    amzn-s3-demo-bucket3}
Owner              : Amazon.S3.Model.Owner
ContinuationToken  :
ResponseMetadata   : Amazon.Runtime.ResponseMetadata
ContentLength      : 0
HttpStatusCode     : OK
```

Iterasi melalui data halaman

Bagian berikut menjelaskan berbagai jenis iterasi yang mungkin.

Iterasi otomatis

Untuk layanan APIs yang memaksakan jumlah maksimum default objek yang dikembalikan untuk panggilan tertentu atau yang mendukung set hasil yang dapat dihalaman, sebagian besar cmdlet menerapkan iterasi otomatis, yang memungkinkan perilaku default `page-to-completion`. Dalam skenario ini, cmdlet membuat panggilan sebanyak yang diperlukan atas nama Anda untuk mengembalikan kumpulan data lengkap ke pipeline.

Dalam contoh berikut, yang menggunakan `Get-S3Object` cmdlet, `$result` variabel berisi `S3Object` instance untuk setiap kunci dalam bucket yang disebut `amzn-s3-demo-bucket1`, yang berpotensi merupakan kumpulan data yang sangat besar.

```
PS > $result = Get-S3Object -BucketName amzn-s3-demo-bucket1
```

Contoh berikut mengurangi jumlah hasil untuk setiap halaman selama iterasi otomatis dari nilai default 1000 menjadi 500. Contoh melakukan dua kali lebih banyak panggilan iterasi otomatis karena hanya setengah dari banyak hasil yang dikembalikan untuk setiap panggilan.

```
PS > $result = Get-S3Object -BucketName amzn-s3-demo-bucket1 -MaxKey 500
```

Note

Di Alat AWS untuk PowerShell V4, beberapa cmdlet untuk operasi paged tidak menerapkan iterasi otomatis. Jika cmdlet tidak memiliki `-NoAutoIteration` parameter, yang dibahas di bagian berikutnya, maka cmdlet tidak menerapkan iterasi otomatis.

Nonaktifkan iterasi otomatis

Jika Anda ingin Alat PowerShell untuk mengembalikan hanya halaman pertama data, Anda dapat menambahkan `-NoAutoIteration` parameter untuk mencegah halaman data tambahan dikembalikan.

Contoh berikut menggunakan `-MaxKey` parameter `-NoAutoIteration` and untuk membatasi jumlah S3Object instance yang dikembalikan tidak lebih dari 500 pertama yang ditemukan di bucket.

```
PS > $result = Get-S3Object -BucketName amzn-s3-demo-bucket1 -MaxKey 500 -
NoAutoIteration
```

Untuk menentukan apakah lebih banyak data tersedia tetapi tidak dikembalikan, gunakan `-Select *` parameter dan argumen dan periksa apakah ada nilai di properti token berikutnya.

Contoh berikut kembali `$true` jika ada lebih dari 500 objek dalam ember dan `$false` sebaliknya.

```
PS > $result = Get-S3Object -BucketName amzn-s3-demo-bucket1 -MaxKey 500 -
NoAutoIteration -Select *
PS > $null -eq $result.NextMarker
```

Note

Nama-nama properti respons token berikutnya dan parameter cmdlet bervariasi antar cmdlet. Untuk detailnya, lihat dokumentasi bantuan untuk setiap cmdlet.

Iterasi manual

Contoh berikut mengembalikan semua objek S3 dari bucket menggunakan [do](#) loop, yang mengevaluasi kondisi setelah setiap iterasi. `doLoop` melakukan iterasi hingga `Get-S3Object` set `$result.NextMarker` ke `$null`, menunjukkan bahwa tidak ada lagi data halaman yang tersisa. Output dari loop ditugaskan ke `$s3objects` variabel.

```
$s3objects = do
{
    $splatParams = @{
        BucketName = 'amzn-s3-demo-bucket1'
```

```
    MaxKey = 500
    Marker = $result.NextMarker
    NoAutoIteration = $true
    Select = '*'
}
$result = Get-S3Object @splatParams

$result.S3Objects
}
while ($null -ne $result.NextMarker)
```

Contoh ini menggunakan PowerShell [percikan](#) untuk menghindari baris panjang kode yang akan disebabkan oleh mendeklarasikan parameter dan argumen in-line.

Resolusi kredensi dan profil

Urutan Pencarian Kredensial

Ketika Anda menjalankan perintah, Alat AWS untuk PowerShell mencari kredensi dalam urutan berikut. Pencarian berhenti ketika menemukan kredensial yang dapat digunakan.

1. Kredensial literal yang melekat sebagai parameter di baris perintah.

Kami sangat menyarankan untuk menggunakan profil bukan menempatkan kredensial literal di baris perintah Anda.

2. Nama profil atau lokasi profil tertentu.

- Jika Anda hanya menentukan nama profil, perintah akan mencari profil yang ditentukan di penyimpanan AWS SDK dan, jika itu tidak ada, profil yang ditentukan dari file AWS kredensi bersama di lokasi default.
- Jika Anda hanya menyebutkan lokasi profil, perintah akan mencari profil default dari file kredensial tersebut.
- Jika Anda hanya menyebutkan nama dan lokasi, perintah akan mencari profil yang disebutkan dari file kredensial tersebut.

Jika profil atau lokasi yang disebutkan tidak ditemukan, perintah akan menunjukkan pengecualian. Pencarian akan dilanjutkan ke langkah-langkah berikut hanya jika Anda tidak menyebutkan profil atau lokasi.

3. Kredensial yang disebutkan oleh parameter `-Credential`.

4. Profil sesi, jika ada.
5. Profil default, dalam urutan sebagai berikut:
 - a. defaultProfil di toko AWS SDK.
 - b. defaultProfil dalam file kredensial AWS bersama.
 - c. AWS PS DefaultProfil di toko AWS SDK.
6. Jika perintah berjalan pada EC2 instans Amazon yang dikonfigurasi untuk menggunakan peran IAM, kredensial sementara instans diakses dari profil EC2 instance.

Untuk informasi selengkapnya tentang penggunaan peran IAM untuk EC2 instans Amazon, lihat [AWS SDK untuk .NET](#)

Jika pencarian ini gagal menemukan kredensial yang disebutkan, perintah akan menunjukkan pengecualian.

Informasi tambahan tentang pengguna dan peran

Untuk menjalankan Alat untuk PowerShell perintah aktif AWS, Anda harus memiliki beberapa kombinasi pengguna, set izin, dan peran layanan yang sesuai untuk tugas Anda.

Pengguna tertentu, set izin, dan peran layanan yang Anda buat, dan cara Anda menggunakannya, akan bergantung pada kebutuhan Anda. Berikut ini adalah beberapa informasi tambahan tentang mengapa mereka dapat digunakan dan cara membuatnya.

Pengguna dan set izin

Meskipun dimungkinkan untuk menggunakan akun pengguna IAM dengan kredensi jangka panjang untuk mengakses AWS layanan, ini bukan lagi praktik terbaik dan harus dihindari. Bahkan selama pengembangan, itu adalah praktik terbaik untuk membuat pengguna dan set izin AWS IAM Identity Center dan menggunakan kredensi sementara yang disediakan oleh sumber identitas.

Untuk pengembangan, Anda dapat menggunakan pengguna yang Anda buat atau diberikan [Konfigurasi otentikasi alat](#). Jika Anda memiliki AWS Management Console izin yang sesuai, Anda juga dapat membuat set izin yang berbeda dengan hak istimewa paling sedikit untuk pengguna tersebut atau membuat pengguna baru khusus untuk proyek pengembangan, memberikan set izin dengan hak istimewa paling sedikit. Tindakan yang Anda pilih, jika ada, tergantung pada keadaan Anda.

Untuk informasi selengkapnya tentang pengguna dan set izin ini serta cara membuatnya, lihat [Otentikasi dan akses](#) di Panduan Referensi Alat AWS SDKs dan [Memulai](#) di Panduan AWS IAM Identity Center Pengguna.

Peran layanan

Anda dapat mengatur peran AWS layanan untuk mengakses AWS layanan atas nama pengguna. Jenis akses ini sesuai jika beberapa orang akan menjalankan aplikasi Anda dari jarak jauh; misalnya, pada EC2 instance Amazon yang telah Anda buat untuk tujuan ini.

Proses untuk membuat peran layanan bervariasi tergantung pada situasinya, tetapi pada dasarnya adalah sebagai berikut.

1. Masuk ke AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Pilih Peran, lalu pilih Buat peran.
3. Pilih AWS layanan, temukan dan pilih EC2(misalnya), lalu pilih kasus EC2penggunaan (misalnya).
4. Pilih Berikutnya dan pilih [kebijakan yang sesuai](#) untuk AWS layanan yang akan digunakan aplikasi Anda.

Warning

JANGAN memilih AdministratorAccesskebijakan karena kebijakan tersebut memungkinkan izin baca dan tulis untuk hampir semua yang ada di akun Anda.

5. Pilih Berikutnya. Masukkan nama Peran, Deskripsi, dan tag apa pun yang Anda inginkan.

Anda dapat menemukan informasi tentang tag di [Mengontrol akses menggunakan tag AWS sumber daya](#) di [Panduan Pengguna IAM](#).

6. Pilih Buat peran.

[Anda dapat menemukan informasi tingkat tinggi tentang peran IAM di Identitas IAM \(pengguna, grup pengguna, dan peran\) di Panduan Pengguna IAM.](#) Temukan informasi terperinci tentang peran dalam topik [peran IAM](#).

Menggunakan kredensial warisan

Topik di bagian ini memberikan informasi tentang penggunaan kredensi jangka panjang atau jangka pendek tanpa menggunakan. AWS IAM Identity Center

Warning

Untuk menghindari risiko keamanan, jangan gunakan pengguna IAM untuk otentikasi saat mengembangkan perangkat lunak yang dibuat khusus atau bekerja dengan data nyata. Sebaliknya, gunakan federasi dengan penyedia identitas seperti [AWS IAM Identity Center](#).

Note

Informasi dalam topik ini adalah untuk keadaan di mana Anda perlu memperoleh dan mengelola kredensi jangka pendek atau jangka panjang secara manual. Untuk informasi tambahan tentang kredensi jangka pendek dan jangka panjang, lihat [Cara lain untuk mengautentikasi](#) di Panduan Referensi Alat AWS SDKs dan Alat.

Untuk praktik keamanan terbaik, gunakan AWS IAM Identity Center, seperti yang dijelaskan dalam [Konfigurasi otentikasi alat](#).

Peringatan penting dan panduan untuk kredensial

Peringatan untuk kredensial

- JANGAN gunakan kredensi root akun Anda untuk mengakses AWS sumber daya. Kredensi ini menyediakan akses akun yang tidak terbatas dan sulit dicabut.
- Jangan menaruh kunci akses literal atau informasi kredensi dalam perintah atau skrip Anda. Jika Anda melakukannya, Anda membuat risiko secara tidak sengaja mengekspos kredensial Anda.
- Ketahuilah bahwa setiap kredensial yang disimpan dalam AWS `credentials` file bersama, disimpan dalam teks biasa.

Panduan tambahan untuk mengelola kredensial dengan aman

[Untuk diskusi umum tentang cara mengelola AWS kredensial dengan aman, lihat kredensial AWS keamanan dalam praktik dan kasus penggunaan terbaik Keamanan Referensi Umum AWS dan kasus penggunaan di Panduan Pengguna IAM.](#) Selain diskusi tersebut, pertimbangkan hal berikut:

- Buat pengguna tambahan, seperti pengguna di IAM Identity Center, dan gunakan kredensialnya alih-alih menggunakan kredensi pengguna AWS root Anda. Kredensi untuk pengguna lain dapat dicabut jika perlu atau bersifat sementara. Selain itu, Anda dapat menerapkan kebijakan kepada setiap pengguna untuk akses hanya ke sumber daya dan tindakan tertentu dan dengan demikian mengambil sikap izin hak istimewa paling sedikit.
- Gunakan [peran IAM untuk tugas untuk tugas](#) Amazon Elastic Container Service (Amazon ECS).
- Gunakan [peran IAM](#) untuk aplikasi yang berjalan di EC2 instans Amazon.

Topik

- [Menggunakan AWS Kredensial](#)
- [Kredensial Bersama di Alat AWS untuk PowerShell](#)

Menggunakan AWS Kredensial

Setiap Alat AWS untuk PowerShell perintah harus menyertakan satu set AWS kredensial, yang digunakan untuk menandatangani permintaan layanan web yang sesuai secara kriptografis. Anda dapat menentukan kredensial per perintah, per sesi, atau untuk semua sesi.

Warning

Untuk menghindari risiko keamanan, jangan gunakan pengguna IAM untuk otentikasi saat mengembangkan perangkat lunak yang dibuat khusus atau bekerja dengan data nyata. Sebaliknya, gunakan federasi dengan penyedia identitas seperti [AWS IAM Identity Center](#).

 Note

Informasi dalam topik ini adalah untuk keadaan di mana Anda perlu memperoleh dan mengelola kredensi jangka pendek atau jangka panjang secara manual. Untuk informasi tambahan tentang kredensial jangka pendek dan jangka panjang, lihat [Cara lain untuk mengautentikasi](#) di Panduan Referensi Alat AWS SDKs dan Alat.

Untuk praktik keamanan terbaik, gunakan AWS IAM Identity Center, seperti yang dijelaskan dalam [Konfigurasi otentikasi alat](#).

Sebagai praktik terbaik, agar kredensial Anda tidak terlihat, jangan menempatkan kredensial literal dalam perintah. Sebaliknya, buat profil untuk setiap set kredensial yang ingin Anda gunakan, dan simpan profil di salah satu dari dua penyimpanan kredensial. Tentukan profil yang benar berdasarkan nama dalam perintah Anda, dan ambil kredensial Alat AWS untuk PowerShell terkait. Untuk diskusi umum tentang cara mengelola AWS kredensial dengan aman, lihat [Praktik Terbaik untuk Mengelola Kunci AWS Akses](#) di. Referensi Umum Amazon Web Services

 Note

Anda memerlukan AWS akun untuk mendapatkan kredensial dan menggunakan. Alat AWS untuk PowerShell Untuk membuat AWS akun, lihat [Memulai: Apakah Anda AWS pengguna pertama kali?](#) dalam Panduan AWS Account Management Referensi.

Topik

- [Lokasi Penyimpanan Kredensial](#)
- [Mengelola Profil](#)
- [Menentukan Kredensial](#)
- [Urutan Pencarian Kredensial](#)
- [Penanganan Kredensial di AWS Tools for PowerShell Core](#)

Lokasi Penyimpanan Kredensial

Alat AWS untuk PowerShell Dapat menggunakan salah satu dari dua toko kredensial:

- Toko AWS SDK, yang mengenkripsi kredensial Anda dan menyimpannya di folder rumah Anda. Di Windows, penyimpanan ini terletak di: `C:\Users\username\AppData\Local\AWSToolkit\RegisteredAccounts.json`.

[AWS SDK untuk .NET](#) dan [Toolkit for Visual Studio](#) Juga dapat menggunakan penyimpanan SDK AWS.

- File kredensial bersama, yang juga terletak di folder beranda Anda, tetapi menyimpan kredensial sebagai teks biasa.

Secara default, file kredensial tersebut disimpan di sini:

- Di Windows: `C:\Users\username\.aws\credentials`
- Di Mac/Linux: `~/.aws/credentials`

The AWS SDKs and the juga AWS Command Line Interface dapat menggunakan file kredensial. Jika Anda menjalankan skrip di luar konteks AWS pengguna, pastikan file yang berisi kredensial Anda disalin ke lokasi di mana semua akun pengguna (sistem lokal dan pengguna) dapat mengakses kredensial Anda.

Mengelola Profil

Profil memungkinkan Anda untuk mereferensikan kumpulan kredensial yang berbeda dengan. Alat AWS untuk PowerShell Anda dapat menggunakan Alat AWS untuk PowerShell cmdlet untuk mengelola profil Anda di toko AWS SDK. Anda juga dapat mengelola profil di penyimpanan SDK AWS dengan menggunakan [Toolkit for Visual Studio](#) atau secara pemrograman dengan menggunakan [AWS SDK untuk .NET](#). Untuk petunjuk tentang cara mengelola profil dalam file kredensial, lihat [Praktik Terbaik untuk Mengelola Kunci AWS Akses](#).

Menambahkan Profil Baru

Untuk menambahkan profil baru ke toko AWS SDK, jalankan perintah `Set-AWSCredential`. Tindakan ini akan menyimpan access key dan secret key Anda dalam file kredensial default Anda dengan nama profil yang Anda tentukan.

```
PS > Set-AWSCredential `
    -AccessKey AKIA0123456787EXAMPLE `
    -SecretKey wJalrXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY `
    -StoreAs MyNewProfile
```

- -AccessKey- Access key ID.
- -SecretKey- Secret key.
- -StoreAs- Nama profil, yang harus unik. Untuk menetapkan profil default, gunakan nama default.

Memperbarui Profil

Penyimpanan AWS SDK harus dipelihara secara manual. Jika selanjutnya Anda mengubah kredensial di layanan—misalnya, dengan menggunakan [Konsol IAM](#)—menjalankan perintah dengan kredensial yang disimpan secara lokal gagal dengan pesan galat berikut:

```
The Access Key Id you provided does not exist in our records.
```

Anda dapat memperbarui profil dengan mengulangi perintah `Set-AWSCredential` untuk profil tersebut, dan memberikan access key dan secret key baru.

Membuat Daftar Profil

Anda dapat memeriksa daftar nama terkini dengan perintah berikut. Dalam contoh ini, pengguna bernama Shirley memiliki akses ke tiga profil yang semuanya disimpan dalam file kredensial bersama (`~/.aws/credentials`).

```
PS > Get-AWSCredential -ListProfileDetail
```

ProfileName	StoreTypeName	ProfileLocation
-----	-----	-----
default	SharedCredentialsFile	/Users/shirley/.aws/credentials
production	SharedCredentialsFile	/Users/shirley/.aws/credentials
test	SharedCredentialsFile	/Users/shirley/.aws/credentials

Menghapus Profil

Untuk menghapus profil yang tidak lagi Anda perlukan, gunakan perintah berikut.

```
PS > Remove-AWSCredentialProfile -ProfileName an-old-profile-I-do-not-need
```

Parameter `-ProfileName` menentukan profil yang ingin Anda hapus.

Perintah [Clear-](#) yang tidak digunakan lagi masih `AWSCredential` tersedia untuk kompatibilitas mundur, tetapi lebih disukai. `Remove-AWSCredentialProfile`

Menentukan Kredensial

Ada beberapa cara untuk menentukan kredensial. Cara yang lebih disukai adalah mengidentifikasi profil alih-alih memasukkan kredensial literal ke dalam baris perintah Anda. Alat AWS untuk PowerShell menempatkan profil menggunakan urutan pencarian yang dijelaskan dalam [Credentials Search Order](#).

Di Windows, AWS kredensial yang disimpan di toko AWS SDK dienkripsi dengan identitas pengguna Windows yang masuk. Kredensial-kredensial tersebut tidak dapat didekripsi dengan menggunakan akun lain, atau digunakan di perangkat yang berbeda dari akun yang awalnya dibuat. Untuk melakukan tugas-tugas yang memerlukan kredensial pengguna lain, seperti akun pengguna yang akan menjalankan suatu tugas terjadwal, atur profil kredensial, seperti yang dijelaskan di bagian sebelumnya, yang dapat Anda gunakan ketika Anda log in ke komputer sebagai pengguna. Log in sebagai pengguna yang melaksanakan tugas untuk menyelesaikan langkah-langkah pengaturan kredensial, dan buat profil yang berfungsi untuk pengguna tersebut. Kemudian log out dan log in kembali dengan kredensial Anda sendiri untuk mengatur tugas terjadwal tersebut.

Note

Menggunakan parameter umum `-ProfileName` untuk menentukan profil. Parameter ini setara dengan `-StoredCredentials` parameter dalam Alat AWS untuk PowerShell rilis sebelumnya. Untuk kompatibilitas balik, `-StoredCredentials` masih didukung.

Profil Default (Disarankan)

Semua AWS SDKs dan alat manajemen dapat menemukan kredensial Anda secara otomatis di komputer lokal Anda jika kredensialnya disimpan dalam profil bernama `default`. Misalnya, jika Anda memiliki profil bernama `default` di komputer lokal, Anda tidak perlu menjalankan cmdlet `Initialize-AWSDefaultConfiguration` atau cmdlet `Set-AWSCredential`. Alat secara otomatis menggunakan data access key dan secret key yang tersimpan dalam profil tersebut. Untuk menggunakan Wilayah AWS selain Wilayah default Anda (hasil dari `Get-DefaultAWSRegion`), Anda dapat menjalankan `Set-DefaultAWSRegion` dan menentukan Wilayah.

Jika profil Anda tidak diberi nama `default`, tetapi Anda ingin menggunakannya sebagai profil default untuk sesi saat ini, jalankan `Set-AWSCredential` untuk menetakannya sebagai profil default.

Meskipun menjalankan `Initialize-AWSDefaultConfiguration` memungkinkan Anda menentukan profil default untuk setiap PowerShell sesi, cmdlet memuat kredensial dari profil yang diberi nama khusus, tetapi menimpa profil dengan profil bernama. `default`

Kami menyarankan agar Anda tidak menjalankan `Initialize-AWSDefaultConfiguration` kecuali Anda menjalankan PowerShell sesi pada EC2 instans Amazon yang tidak diluncurkan dengan profil instans, dan Anda ingin mengatur profil kredensialnya secara manual. Perhatikan bahwa profil kredensial dalam skenario ini tidak akan berisi kredensial. Profil kredensi yang dihasilkan dari menjalankan `Initialize-AWSDefaultConfiguration` EC2 instans tidak secara langsung menyimpan kredensial, melainkan menunjuk ke metadata instance (yang menyediakan kredensial sementara yang berputar secara otomatis). Namun, tindakan ini tidak akan menyimpan Wilayah instans. Skenario lain yang mungkin harus menjalankan `Initialize-AWSDefaultConfiguration` terjadi jika Anda ingin menjalankan panggilan terhadap sebuah Wilayah selain dari Wilayah di mana instans tersebut berjalan. Menjalankan perintah tersebut akan secara permanen menimpa Wilayah yang disimpan dalam metadata instans.

```
PS > Initialize-AWSDefaultConfiguration -ProfileName MyProfileName -Region us-west-2
```

Note

Kredensi default disertakan dalam penyimpanan AWS SDK di bawah nama profil. `default`
Perintah tersebut akan menimpa profil yang ada dengan nama itu.

Jika EC2 instans Anda diluncurkan dengan profil instans, PowerShell secara otomatis mendapatkan AWS kredensial dan informasi Wilayah dari profil instans. Anda tidak perlu menjalankan `Initialize-AWSDefaultConfiguration`. Menjalankan `Initialize-AWSDefaultConfiguration` cmdlet pada EC2 instance yang diluncurkan dengan profil instance tidak diperlukan, karena menggunakan data profil instance yang sama yang PowerShell sudah digunakan secara default.

Profil Sesi

Gunakan `Set-AWSCredential` untuk menentukan profil default untuk sesi tertentu. Profil ini menimpa profil default apapun selama sesi berlangsung. Kami merekomendasikan hal ini jika Anda ingin menggunakan profil yang diberi nama khusus di sesi Anda, bukan profil `default` saat ini.

```
PS > Set-AWSCredential -ProfileName MyProfileName
```

Note

Dalam versi Tools untuk Windows PowerShell yang lebih awal dari 1.1, `Set-AWSCredential` cmdlet tidak berfungsi dengan benar, dan akan menimpa profil yang ditentukan oleh `""`. `MyProfileName` Sebaiknya gunakan versi Tools for Windows yang lebih baru PowerShell.

Profil Perintah

Pada perintah individual, Anda dapat menambahkan parameter `-ProfileName` untuk menentukan profil yang berlaku untuk hanya satu perintah tersebut. Profil ini menimpa profil default atau sesi, seperti yang ditunjukkan dalam contoh berikut.

```
PS > Get-EC2Instance -ProfileName MyProfileName
```

Note

Bila Anda menentukan profil default atau sesi, Anda juga dapat menambahkan parameter `-Region` untuk menimpa Wilayah default atau sesi. Untuk informasi lebih lanjut, lihat [Tentukan AWS Wilayah](#). Contoh berikut menentukan profil dan Wilayah default.

```
PS > Initialize-AWSDefaultConfiguration -ProfileName MyProfileName -Region us-west-2
```

Secara default, file kredensi AWS bersama diasumsikan berada di folder beranda pengguna (C:\Users\username\.aws di Windows, atau ~/.aws di Linux). Untuk menentukan file kredensial di lokasi yang berbeda, masukkan parameter `-ProfileLocation` dan tentukan jalur file kredensial. Contoh berikut menentukan file kredensial non default untuk perintah tertentu.

```
PS > Get-EC2Instance -ProfileName MyProfileName -ProfileLocation C:\aws_service_credentials\credentials
```

Note

Jika Anda menjalankan PowerShell skrip selama waktu yang biasanya tidak masuk ke AWS —misalnya, Anda menjalankan PowerShell skrip sebagai tugas terjadwal di luar jam kerja

normal Anda—tambahkan `-ProfileLocation` parameter saat Anda menentukan profil yang ingin Anda gunakan, dan setel nilainya ke jalur file yang menyimpan kredensialnya. Untuk memastikan bahwa Alat AWS untuk PowerShell skrip Anda berjalan dengan kredensi akun yang benar, Anda harus menambahkan `-ProfileLocation` parameter setiap kali skrip Anda berjalan dalam konteks atau proses yang tidak menggunakan akun AWS. Anda juga dapat menyalin file kredensial Anda ke lokasi yang dapat diakses oleh sistem lokal atau akun lain yang digunakan skrip Anda untuk melakukan tugas.

Urutan Pencarian Kredensial

Ketika Anda menjalankan perintah, Alat AWS untuk PowerShell mencari kredensi dalam urutan berikut. Pencarian berhenti ketika menemukan kredensial yang dapat digunakan.

1. Kredensial literal yang melekat sebagai parameter di baris perintah.

Kami sangat menyarankan untuk menggunakan profil bukan menempatkan kredensial literal di baris perintah Anda.

2. Nama profil atau lokasi profil tertentu.

- Jika Anda hanya menentukan nama profil, perintah akan mencari profil yang ditentukan di penyimpanan AWS SDK dan, jika itu tidak ada, profil yang ditentukan dari file AWS kredensial bersama di lokasi default.
- Jika Anda hanya menyebutkan lokasi profil, perintah akan mencari profil default dari file kredensial tersebut.
- Jika Anda hanya menyebutkan nama dan lokasi, perintah akan mencari profil yang disebutkan dari file kredensial tersebut.

Jika profil atau lokasi yang disebutkan tidak ditemukan, perintah akan menunjukkan pengecualian. Pencarian akan dilanjutkan ke langkah-langkah berikut hanya jika Anda tidak menyebutkan profil atau lokasi.

3. Kredensial yang disebutkan oleh parameter `-Credential`.

4. Profil sesi, jika ada.

5. Profil default, dalam urutan sebagai berikut:

- a. defaultProfil di toko AWS SDK.
- b. defaultProfil dalam file kredensial AWS bersama.
- c. AWS PS DefaultProfil di toko AWS SDK.

6. Jika perintah berjalan pada EC2 instans Amazon yang dikonfigurasi untuk menggunakan peran IAM, kredensial sementara instans diakses dari profil EC2 instance.

Untuk informasi selengkapnya tentang penggunaan peran IAM untuk EC2 instans Amazon, lihat [AWS SDK untuk .NET](#)

Jika pencarian ini gagal menemukan kredensial yang disebutkan, perintah akan menunjukkan pengecualian.

Penanganan Kredensial di AWS Tools for PowerShell Core

Cmdlet dalam AWS Tools for PowerShell Core menerima AWS akses dan kunci rahasia atau nama profil kredensi saat dijalankan, mirip dengan file. AWS Tools for Windows PowerShell Ketika berjalan di Windows, kedua modul memiliki akses ke file penyimpanan kredensial AWS SDK untuk .NET (yang disimpan di file AppData\Local\AWSToolkit\RegisteredAccounts.json per-pengguna).

File ini menyimpan kunci Anda dalam format terenkripsi, dan tidak dapat digunakan pada komputer yang berbeda. Ini adalah file pertama yang Alat AWS untuk PowerShell mencari profil kredensi, dan juga merupakan file tempat Alat AWS untuk PowerShell menyimpan profil kredensi. Untuk informasi selengkapnya tentang file penyimpanan AWS SDK untuk .NET kredensi, lihat [Mengonfigurasi AWS Kredensial](#). PowerShellModul Tools for Windows saat ini tidak mendukung kredensi penulisan ke file atau lokasi lain.

Kedua modul dapat membaca profil dari file AWS kredensi bersama yang digunakan oleh yang lain AWS SDKs dan file. AWS CLI Pada Windows, lokasi default untuk file ini adalah C:\Users\<userid>\.aws\credentials. Pada platform selain Windows, file ini disimpan di ~/.aws/credentials. Parameter -ProfileLocation dapat digunakan untuk menunjuk ke nama file non default atau lokasi file.

Toko kredensi SDK menyimpan kredensial Anda dalam bentuk terenkripsi dengan menggunakan kriptografi Windows. APIs Ini tidak APIs tersedia di platform lain, jadi AWS Tools for PowerShell Core modul menggunakan file kredensi AWS bersama secara eksklusif, dan mendukung penulisan profil kredensi baru ke file kredensi bersama.

Contoh skrip berikut yang menggunakan **Set-AWSCredential** cmdlet menunjukkan opsi untuk menangani profil kredensi di Windows dengan Shell atau Shell. AWSPower AWSPower NetCoremodul.

```
# Writes a new (or updates existing) profile with name "myProfileName"
# in the encrypted SDK store file

Set-AWSCredential -AccessKey akey -SecretKey skey -StoreAs myProfileName

# Checks the encrypted SDK credential store for the profile and then
# falls back to the shared credentials file in the default location

Set-AWSCredential -ProfileName myProfileName

# Bypasses the encrypted SDK credential store and attempts to load the
# profile from the ini-format credentials file "mycredentials" in the
# folder C:\MyCustomPath

Set-AWSCredential -ProfileName myProfileName -ProfileLocation C:\MyCustomPath
\mycredentials
```

Contoh berikut menunjukkan perilaku AWSPowerShell. NetCoremodul pada sistem operasi Linux atau macOS.

```
# Writes a new (or updates existing) profile with name "myProfileName"
# in the default shared credentials file ~/.aws/credentials

Set-AWSCredential -AccessKey akey -SecretKey skey -StoreAs myProfileName

# Writes a new (or updates existing) profile with name "myProfileName"
# into an ini-format credentials file "~/mycustompath/mycredentials"

Set-AWSCredential -AccessKey akey -SecretKey skey -StoreAs myProfileName -
ProfileLocation ~/mycustompath/mycredentials

# Reads the default shared credential file looking for the profile "myProfileName"

Set-AWSCredential -ProfileName myProfileName

# Reads the specified credential file looking for the profile "myProfileName"

Set-AWSCredential -ProfileName myProfileName -ProfileLocation ~/mycustompath/
mycredentials
```

Kredensial Bersama di Alat AWS untuk PowerShell

Alat untuk Windows PowerShell mendukung penggunaan file kredensi AWS bersama, mirip dengan AWS CLI dan lainnya. AWS SDKs Tools untuk Windows PowerShell sekarang mendukung membaca dan menulis `basic`, `session`, dan profil kredensialnya ke file `assume_role_kredensial.NET` dan file kredensialnya AWS bersama. Fungsionalitas ini diaktifkan oleh namespace `Amazon.Runtime.CredentialManagement` baru.

Warning

Untuk menghindari risiko keamanan, jangan gunakan pengguna IAM untuk otentikasi saat mengembangkan perangkat lunak yang dibuat khusus atau bekerja dengan data nyata. Sebaliknya, gunakan federasi dengan penyedia identitas seperti [AWS IAM Identity Center](#).

Note

Informasi dalam topik ini adalah untuk keadaan di mana Anda perlu memperoleh dan mengelola kredensi jangka pendek atau jangka panjang secara manual. Untuk informasi tambahan tentang kredensi jangka pendek dan jangka panjang, lihat [Cara lain untuk mengautentikasi](#) di Panduan Referensi Alat AWS SDKs dan Alat.

Untuk praktik keamanan terbaik, gunakan AWS IAM Identity Center, seperti yang dijelaskan dalam [Konfigurasi otentikasi alat](#).

[Jenis profil baru dan akses ke file AWS kredensi bersama didukung oleh parameter berikut yang telah ditambahkan ke cmdlet terkait kredensi, Inisialisasi- AWSDefault Konfigurasi, Baru-, dan Set- AWSCredential AWSCredential](#) Dalam cmdlet layanan, Anda dapat merujuk ke profil Anda dengan menambahkan parameter umum, `-ProfileName`.

Menggunakan IAM role dengan Alat AWS untuk PowerShell

File AWS kredensi bersama memungkinkan jenis akses tambahan. Misalnya, Anda dapat mengakses AWS sumber daya Anda dengan menggunakan peran IAM alih-alih kredensi jangka panjang pengguna IAM. Untuk melakukannya, Anda harus memiliki profil standar yang memiliki izin untuk melaksanakan peran tersebut. Ketika Anda memberitahu Alat AWS untuk PowerShell untuk menggunakan profil yang menentukan peran, Alat AWS untuk PowerShell mencari profil yang

diidentifikasi oleh `SourceProfile` parameter. Kredensial tersebut digunakan untuk meminta kredensial sementara untuk peran yang ditentukan oleh parameter `RoleArn`. Anda dapat memilih meminta penggunaan perangkat autentikasi multi-factor (MFA) atau kode `ExternalId` ketika peran dilaksanakan oleh pihak ketiga.

Nama Parameter	Deskripsi
<code>ExternalId</code>	ID eksternal yang ditetapkan pengguna untuk digunakan ketika melaksanakan peran, jika diperlukan oleh peran. Ini biasanya hanya diperlukan saat Anda mendelegasikan akses akun Anda ke pihak ketiga. Pihak ketiga harus menyertakan <code>ExternalId</code> sebagai parameter saat mengasumsikan peran yang ditugaskan. Untuk informasi selengkapnya, lihat Cara Menggunakan ID Eksternal Saat Memberikan Akses ke AWS Sumber Daya Anda kepada Pihak Ketiga dalam Panduan Pengguna IAM.
<code>MfaSerial</code>	Nomor seri MFA yang akan digunakan ketika melaksanakan peran, jika diperlukan oleh peran. Untuk informasi selengkapnya, lihat Menggunakan Autentikasi Multi-Faktor (MFA) dalam AWS dalam Panduan Pengguna IAM.
<code>RoleArn</code>	ARN peran untuk melaksanakan kredensial peran. Untuk informasi selengkapnya tentang pembuatan dan penggunaan peran, lihat IAM role dalam Panduan Pengguna IAM.
<code>SourceProfile</code>	Nama profil sumber yang akan digunakan dengan melaksanakan kredensial peran. Kredensial yang ditemukan di profil ini digunakan untuk melaksanakan peran yang ditentukan oleh parameter <code>RoleArn</code> .

Pengaturan profil untuk melaksanakan peran

Berikut ini adalah contoh yang menunjukkan cara mengatur profil sumber yang memungkinkan secara langsung melaksanakan IAM role.

Perintah pertama membuat profil sumber yang disebutkan oleh profil peran. Perintah kedua membuat profil peran yang perannya akan dilaksanakan. Perintah ketiga menunjukkan kredensial untuk profil peran.

```
PS > Set-AWSCredential -StoreAs my_source_profile -AccessKey access_key_id -
SecretKey secret_key
PS > Set-AWSCredential -StoreAs my_role_profile -SourceProfile my_source_profile -
RoleArn arn:aws:iam::123456789012:role/role-i-want-to-assume
PS > Get-AWSCredential -ProfileName my_role_profile
```

SourceCredentials	RoleArn
RoleSessionName	Options
-----	-----
-----	-----
Amazon.Runtime.BasicAWSCredentials	arn:aws:iam::123456789012:role/ role-i-want-to-assume aws-dotnet-sdk-session-636238288466144357
Amazon.Runtime.AssumeRoleAWSCredentialsOptions	

Untuk menggunakan profil peran ini dengan cmdlet PowerShell layanan Alat untuk Windows, tambahkan parameter `-ProfileName` umum ke perintah untuk mereferensikan profil peran. Contoh berikut menggunakan profil peran yang ditentukan dalam contoh sebelumnya untuk mengakses [Get-S3Bucket](#) cmdlet. Alat AWS untuk PowerShell mencari kredensialnya `my_source_profile`, menggunakan kredensial tersebut untuk memanggil `AssumeRole` atas nama pengguna, dan kemudian menggunakan kredensial peran sementara tersebut untuk menelepon. `Get-S3Bucket`

```
PS > Get-S3Bucket -ProfileName my_role_profile
```

CreationDate	BucketName
-----	-----
2/27/2017 8:57:53 AM	4ba3578c-f88f-4d8b-b95f-92a8858dac58-bucket1
2/27/2017 10:44:37 AM	2091a504-66a9-4d69-8981-aaef812a02c3-bucket2

Menggunakan Jenis Profil Kredensial

Untuk menetapkan jenis profil kredensial, memahami parameter yang memberikan informasi yang diperlukan oleh jenis profil.

Tipe kredensial	Parameter yang harus Anda gunakan
Basic	-AccessKey
Ini adalah kredensial jangka panjang untuk pengguna IAM	-SecretKey
Sesi:	-AccessKey
Ini adalah kredensi jangka pendek untuk peran IAM yang Anda ambil secara manual, seperti dengan langsung memanggil cmdlet Use-STSRole	-SecretKey
	-SessionToken
Peran:	-SourceProfile
Ini adalah kredensial jangka pendek untuk IAM role yang diambil oleh Alat AWS untuk PowerShell untuk Anda.	-RoleArn
	opsional: -ExternalId
	opsional: -MfaSerial

Parameter Umum **ProfilesLocation**

Anda dapat menggunakan -ProfileLocation untuk menulis ke file kredensial bersama serta memerintahkan cmdlet untuk membaca dari file kredensial. Menambahkan -ProfileLocation parameter mengontrol apakah Alat untuk Windows PowerShell menggunakan file kredensi bersama atau file kredensial.NET. Tabel berikut menjelaskan cara kerja parameter di Alat untuk Windows PowerShell.

Nilai Lokasi Profil	Perilaku Resolusi Profil
nihil (tidak diatur) atau kosong	Pertama, cari file kredensial .NET untuk profil dengan nama yang disebutkan. Jika profil tidak ditemukan, cari file kredensial AWS bersama di. <i>(user's home directory)</i> \.aws\credentials

Nilai Lokasi Profil	Perilaku Resolusi Profil
Path ke file dalam format file AWS kredensi bersama	Cari hanya file yang disebutkan untuk profil dengan nama yang diberikan.

Menyimpan Kredensial ke File Kredensial

Untuk menulis dan menyimpan kredensial ke salah satu dari dua file kredensial tersebut, jalankan perintah cmdlet `Set-AWSCredential`. Contoh berikut menunjukkan cara melakukannya. Perintah pertama menggunakan `Set-AWSCredential` dengan `-ProfileLocation` untuk menambahkan access key dan secret key ke profil yang ditentukan oleh parameter `-ProfileName`. Pada baris kedua, jalankan cmdlet [Get-Content](#) untuk menampilkan isi file kredensial.

```
PS > Set-AWSCredential -ProfileLocation C:\Users\auser\.aws\credentials -ProfileName
    basic_profile -AccessKey access_key2 -SecretKey secret_key2
PS > Get-Content C:\Users\auser\.aws\credentials

aws_access_key_id=access_key2
aws_secret_access_key=secret_key2
```

Menampilkan Profil Kredensial Anda

Jalankan [Get-AWSCredential](#) cmdlet dan tambahkan `-ListProfileDetail` parameter untuk mengembalikan jenis dan lokasi file kredensi, dan daftar nama profil.

```
PS > Get-AWSCredential -ListProfileDetail

ProfileName                StoreTypeName                ProfileLocation
-----
source_profile             NetSDKCredentialsFile
assume_role_profile        NetSDKCredentialsFile
basic_profile              SharedCredentialsFile C:\Users\auser\.aws\credentials
```

Menghapus Profil Kredensial

Untuk menghapus profil kredensi, jalankan cmdlet [Hapus-AWSCredential Profil](#) baru. [Clear-AWSCredential](#) tidak digunakan lagi, tetapi masih tersedia untuk kompatibilitas mundur.

Catatan Penting

Hanya [Inisialisasi- AWSDefault Konfigurasi](#), [Baru- AWSCredential](#), dan [Set- AWSCredential](#) mendukung parameter untuk profil peran. Anda tidak dapat menyebutkan parameter peran secara langsung pada perintah seperti `Get-S3Bucket -SourceProfile source_profile_name -RoleArn arn:aws:iam::999999999999:role/role_name`. Cara ini tidak akan bekerja karena cmdlet layanan tidak secara langsung mendukung parameter `SourceProfile` atau `RoleArn`. Sebaliknya, Anda harus menyimpan parameter tersebut dalam profil, kemudian memanggil perintah dengan parameter `-ProfileName`.

Fitur dari AWS Tools for Windows PowerShell

Beberapa topik di bagian ini memberikan informasi tentang fitur Alat untuk Windows PowerShell yang mungkin perlu Anda pertimbangkan saat membuat proyek dan skrip Anda. Topik lain di bagian ini memberikan informasi tentang cara-cara lanjutan agar Anda dapat mengonfigurasi alat, lingkungan, dan proyek Anda. Pastikan Anda telah [menginstal](#) dan [mengatur](#) alat terlebih dahulu.

Untuk informasi tentang mengembangkan perangkat lunak untuk AWS layanan tertentu bersama dengan contoh kode, lihat [Bekerja dengan AWS layanan](#). Untuk contoh kode tambahan, lihat [Alat untuk contoh kode PowerShell V4](#).

Topik

- [Observabilitas](#)

Observabilitas

Observabilitas adalah sejauh mana keadaan sistem saat ini dapat disimpulkan dari data yang dipancarkannya. Data yang dipancarkan biasanya disebut sebagai telemetri. [Untuk informasi tambahan tentang telemetri saat menggunakan AWS layanan, lihat Observabilitas di Panduan Pengembang.AWS SDK untuk .NET](#)

Kode berikut menunjukkan contoh bagaimana observabilitas dapat diaktifkan di. Alat AWS untuk PowerShell

```
<#
  This is an example of generating telemetry for AWS Tools for PowerShell.
  Each cmdlet that interacts with an Amazon Web Service creates a trace containing
  spans
  for underlying processes and AWS SDK for .NET operations.
  This example is written using PowerShell 7 and .NET 8.
  It requires the installation of the .NET CLI tool.
  Note that implementation varies by the exporter/endpoint, which is not specified in
  this example.
  For more information, see https://opentelemetry.io/docs/languages/net/exporters/.
#>

# Set this value to a common folder path on your computer for local development of code
  repositories.
$devProjectsPath = [System.IO.Path]::Join('C:', 'Dev', 'Repos')
```

```
# If these values are changed, update the hardcoded method invocation toward the end of
this script.
# Values must follow constraints for namespaces and classes.
$telemetryProjectName = 'ExampleAWSPowerShellTelemetryImplementation'
$serviceName = 'ExamplePowerShellService'

# This example supposes that the OTLP exporter requires these two properties,
# but some exporters require different properties or no properties.
$telemetryEndPoint = 'https://example-endpoint-provider.io'
$telemetryHeaders = 'x-example-header=abc123'

$dllsPath = [System.IO.Path]::Join($devProjectsPath, $telemetryProjectName, 'bin',
'Release', 'net8.0', 'publish')

$telemetryProjectPath = [System.IO.Path]::Join($devProjectsPath, $telemetryProjectName)

# This script is designed to recreate the example telemetry project each time it's
executed.
Remove-Item -Path $telemetryProjectPath -Recurse -Force -ErrorAction 'SilentlyContinue'
$null = New-Item -Path $devProjectsPath -Name $telemetryProjectName -ItemType
'Directory'

<#
    Create and build a C#-based .NET 8 project that implements
    OpenTelemetry Instrumentation for the AWS Tools for PowerShell.
#>

Set-Location -Path $telemetryProjectPath

dotnet new classlib

# Other exporters are available.
# For more information, see https://opentelemetry.io/docs/languages/net/exporters/.
dotnet add package OpenTelemetry.Exporter.OpenTelemetryProtocol
dotnet add package OpenTelemetry.Instrumentation.AWS

$classContent = @"
using OpenTelemetry;
using OpenTelemetry.Resources;
using OpenTelemetry.Trace;

namespace Example.Telemetry;
```

```

public class AWSToolsForPowerShellTelemetry
{
    public static void InitializeAWSInstrumentation()
    {
        Sdk.CreateTracerProviderBuilder()
            .ConfigureResource(e => e.AddService("$ServiceName"))
            .AddAWSInstrumentation()
            // Exporters vary so options might need to be changed or omitted.
            .AddOtlpExporter(options =>
            {
                options.Endpoint = new Uri("$telemetryEndPoint");
                options.Headers = "$telemetryHeaders";
            })
            .Build();
    }
}
"@

$csFilePath = [System.IO.Path]::Join($telemetryProjectPath, ($serviceName + '.cs'))
Set-Content -Path $csFilePath -Value $classContent

dotnet build
dotnet publish -c Release

<#
    Add additional modules here for any other cmdlets that you require.
    Beyond this point, additional AWS Tools for PowerShell modules will fail to import
    due to conflicts with the AWS SDK for .NET assemblies that are added next.
#>

Import-Module -Name 'AWS.Tools.Common'
Import-Module -Name 'AWS.Tools.DynamoDBv2'

# Load assemblies for the telemetry project, excluding the AWS SDK for .NET assemblies
# that were already loaded by importing AWS Tools for PowerShell modules.
$dlls = (Get-ChildItem $dllsPath -Filter *.dll -Recurse ).FullName
$AWSSDKAssembliesAlreadyLoaded =
    [Threading.Thread]::GetDomain().GetAssemblies().Location | Where-Object {$_ -like
        '*AWSSDK*' } | Split-Path -Leaf
$dlls.Where{$AWSSDKAssembliesAlreadyLoaded -notcontains ($_ | Split-Path -
    Leaf)}.ForEach{Add-Type -Path $_}

# Invoke the method defined earlier in this script.

```

```
[Example.Telemetry.AWSToolsForPowerShellTelemetry]::InitializeAWSInstrumentation()
```

```
<#
```

```
    Now telemetry will be exported for AWS Tools for PowerShell cmdlets  
    that are invoked directly or indirectly.
```

```
    Execute this cmdlet or execute your own PowerShell script.
```

```
#>
```

```
Get-DDBTable -TableName 'DotNetTests-HashTable' -Region 'us-east-1'
```

Bekerja dengan AWS layanan di Alat AWS untuk PowerShell

Bagian ini memberikan contoh penggunaan Alat AWS untuk PowerShell untuk mengakses AWS layanan. Contoh-contoh ini membantu mendemonstrasikan cara menggunakan cmdlet untuk melakukan tugas yang sebenarnya AWS. Contoh-contoh ini bergantung pada cmdlet yang disediakan Alat untuk PowerShell. Untuk melihat cmdlet apa yang tersedia, lihat Referensi [Alat AWS untuk PowerShell Cmdlet](#).

PowerShell Pengkodean Penggabungan File

Beberapa cmdlet dalam Alat AWS untuk PowerShell mengedit file atau catatan yang ada yang Anda miliki. AWS Contohnya adalah `Edit-R53ResourceRecordSet`, yang memanggil [ChangeResourceRecordSets](#) API untuk Amazon Route 53.

Saat Anda mengedit atau menggabungkan file dalam rilis PowerShell 5.1 atau yang lebih lama, PowerShell mengkodekan output dalam UTF-16, bukan UTF-8. Ini dapat menambahkan karakter yang tidak diinginkan dan membuat hasil yang tidak valid. Editor heksadesimal dapat mengungkapkan karakter yang tidak diinginkan.

Untuk menghindari konversi output file ke UTF-16, Anda dapat menyalurkan perintah Anda ke PowerShell `Out-File` cmdlet dan menentukan pengkodean UTF-8, seperti yang ditunjukkan pada contoh berikut:

```
PS > *some file concatenation command* | Out-File filename.txt -Encoding utf8
```

Jika Anda menjalankan AWS CLI perintah dari dalam PowerShell konsol, perilaku yang sama berlaku. Anda dapat menyalurkan output AWS CLI perintah ke `Out-File` dalam PowerShell konsol. Cmdlet lainnya, seperti `Export-Csv` atau `Export-Clixml`, juga memiliki parameter `Encoding`. Untuk daftar lengkap cmdlet yang memiliki parameter `Encoding`, dan yang memungkinkan Anda untuk memperbaiki pengkodean output dari file tergabung, jalankan perintah berikut:

```
PS > Get-Command -ParameterName "Encoding"
```

Note

PowerShell 6.0 dan yang lebih baru, termasuk PowerShell Core, secara otomatis mempertahankan pengkodean UTF-8 untuk output file gabungan.

Objek yang Dikembalikan untuk PowerShell Alat

Agar Alat AWS untuk PowerShell lebih berguna di PowerShell lingkungan asli, objek yang dikembalikan oleh Alat AWS untuk PowerShell cmdlet adalah objek.NET, bukan objek teks JSON yang biasanya dikembalikan dari API yang sesuai di SDK. AWS Misalnya, `Get-S3Bucket` memancarkan kumpulan Buckets, bukan objek jawaban JSON Amazon S3. BucketsKoleksi dapat ditempatkan di dalam PowerShell pipa dan berinteraksi dengan cara yang tepat. Demikian pula, `Get-EC2Instance` memancarkan Reservation kumpulan obyek .NET, bukan obyek hasil JSON `DescribeEC2Instances`. Perilaku ini dirancang dan memungkinkan Alat AWS untuk PowerShell pengalaman menjadi lebih konsisten dengan idiomatik PowerShell.

Jawaban layanan aktual tersedia untuk Anda jika Anda membutuhkannya. Jawaban-jawaban disimpan sebagai properti note pada objek yang dikembalikan. Untuk tindakan API yang mendukung pembagian dengan menggunakan bidang `NextToken`, ini juga dilampirkan sebagai properti note.

Amazon EC2

Bagian ini membahas langkah-langkah yang diperlukan untuk meluncurkan EC2 instans Amazon termasuk cara:

- Ambil daftar Amazon Machine Images (AMIs).
- Membuat pasangan kunci untuk otentikasi SSH.
- Buat dan konfigurasikan grup EC2 keamanan Amazon.
- Meluncurkan instans dan mengambil informasi tentang hal itu.

Amazon S3

Bagian ini membahas langkah-langkah yang diperlukan untuk membuat situs web statis yang ditempatkan di Amazon S3. Bagian ini menunjukkan bagaimana cara:

- Membuat dan menghapus bucket Amazon S3.
- Mengunggah file ke bucket Amazon S3 sebagai obyek.
- Menghapus obyek dari bucket Amazon S3.
- Memfungsikan bucket Amazon S3 sebagai situs web.

[AWS Lambda dan Alat AWS untuk PowerShell](#)

Bagian ini memberikan gambaran singkat tentang Alat AWS Lambda untuk PowerShell modul dan menjelaskan langkah-langkah yang diperlukan untuk menyiapkan modul.

[Amazon SNS dan Amazon SQS](#)

Bagian ini membahas langkah-langkah yang diperlukan untuk berlangganan antrean Amazon SQS untuk topik Amazon SNS. Bagian ini menunjukkan bagaimana cara:

- Membuat topik Amazon SNS.
- Membuat antrean Amazon SQS.
- Berlangganan antrean ke topik.
- Mengirim pesan ke topik.
- Menerima pesan dari antrean.

[CloudWatch](#)

Bagian ini memberikan contoh cara mempublikasikan data kustom ke CloudWatch.

- Publikasikan Metrik Kustom ke CloudWatch Dasbor Anda.

Lihat Juga

- [Memulai dengan AWS Tools for Windows PowerShell](#)

Topik

- [Amazon S3 dan Alat untuk Windows PowerShell](#)
- [Amazon EC2 dan Alat untuk Windows PowerShell](#)
- [AWS Lambda dan Alat AWS untuk PowerShell](#)
- [Amazon SQS, Amazon SNS dan Alat untuk Windows PowerShell](#)
- [CloudWatch dari AWS Tools for Windows PowerShell](#)

- [Menggunakan ClientConfig parameter dalam cmdlet](#)

Amazon S3 dan Alat untuk Windows PowerShell

Di bagian ini, kami membuat situs web statis AWS Tools for Windows PowerShell menggunakan Amazon S3 dan CloudFront. Dalam prosesnya, kami menunjukkan sejumlah tugas umum dengan layanan ini. Panduan ini dimodelkan setelah Panduan Memulai untuk [Host Situs Web Statis](#), yang menjelaskan proses serupa menggunakan [Konsol Manajemen AWS](#).

Perintah yang ditampilkan di sini mengasumsikan bahwa Anda telah menetapkan kredensi default dan wilayah default untuk sesi Anda PowerShell. Oleh karena itu, kredensial dan wilayah tidak termasuk dalam invokasi cmdlet.

Note

Saat ini tidak ada API Amazon S3 untuk mengganti nama bucket atau objek, dan oleh karena itu, tidak ada satu pun PowerShell cmdlet Alat untuk Windows untuk melakukan tugas ini. Untuk mengganti nama objek di S3, kami sarankan Anda menyalin objek ke objek dengan nama baru, dengan menjalankan [Copy-S3Object](#) cmdlet, dan kemudian menghapus objek asli dengan menjalankan cmdlet [Remove-S3Object](#).

Lihat juga

- [Bekerja dengan AWS layanan di Alat AWS untuk PowerShell](#)
- [Hosting Situs Web Statis di Amazon S3](#)
- [Konsol Amazon S3](#)

Topik

- [Membuat Bucket Amazon S3, Memverifikasi Wilayahnya, dan Memilih Menghapusnya](#)
- [Konfigurasi Bucket Amazon S3 sebagai Situs Web dan Aktifkan Pencatatan](#)
- [Unggah Objek ke Bucket Amazon S3](#)
- [Hapus Obyek dan Bucket Amazon S3](#)
- [Unggah Konten Teks Selaras ke Amazon S3](#)

Membuat Bucket Amazon S3, Memverifikasi Wilayahnya, dan Memilih Menghapusnya

Menggunakan cmdlet `New-S3Bucket` untuk membuat bucket Amazon S3 baru. Contoh berikut membuat bucket dengan nama `website-example`. Nama bucket harus unik di semua wilayah. Contoh tersebut membuat bucket di wilayah `us-west-1`.

```
PS > New-S3Bucket -BucketName website-example -Region us-west-2
```

CreationDate	BucketName
-----	-----
8/16/19 8:45:38 PM	website-example

Anda dapat memverifikasi wilayah tempat bucket berada menggunakan cmdlet `Get-S3BucketLocation`.

```
PS > Get-S3BucketLocation -BucketName website-example
```

Value

us-west-2

Setelah selesai dengan tutorial ini, Anda dapat menggunakan baris berikut untuk menghapus bucket ini. Kami menyarankan agar Anda membiarkan bucket ini di tempatnya karena kami menggunakannya dalam contoh berikutnya.

```
PS > Remove-S3Bucket -BucketName website-example
```

Perhatikan bahwa proses penghapusan bucket dapat memakan waktu lama hingga selesai. Jika Anda mencoba untuk membuat ulang bucket dengan nama yang sama secara langsung, cmdlet `New-S3Bucket` tidak akan dapat melakukannya hingga bucket lama sepenuhnya hilang.

Lihat Juga

- [Bekerja dengan AWS layanan di Alat AWS untuk PowerShell](#)
- [Put Bucket \(Referensi Layanan Amazon S3\)](#)
- [AWS PowerShell Wilayah untuk Amazon S3](#)

Konfigurasi Bucket Amazon S3 sebagai Situs Web dan Aktifkan Pencatatan

Gunakan cmdlet `Write-S3BucketWebsite` untuk mengkonfigurasi bucket Amazon S3 sebagai situs web statis. Contoh berikut menentukan nama `index.html` untuk halaman web konten default dan nama `error.html` untuk halaman web kesalahan default. Perhatikan bahwa cmdlet ini tidak membuat halaman tersebut. Halaman-halaman tersebut harus [diunggah sebagai objek Amazon S3](#).

```
PS > Write-S3BucketWebsite -BucketName website-example -  
WebsiteConfiguration_IndexDocumentSuffix index.html -WebsiteConfiguration_ErrorDocument  
error.html  
RequestId      : A1813E27995FFDDD  
AmazonId2      : T7h1D0eLqA5Q2XfTe8j2q3SLop3/5XwhUU3RyJBGHU/LnC+CIWLeGgP0MY24xA1I  
ResponseStream :  
Headers        : {x-amz-id-2, x-amz-request-id, Content-Length, Date...}  
Metadata       : {}  
ResponseXml    :
```

Lihat Juga

- [Bekerja dengan AWS layanan di Alat AWS untuk PowerShell](#)
- [Letakkan Situs Web Bucket \(Referensi API Amazon S3\)](#)
- [Letakkan Bucket ACL \(Referensi API Amazon S3\)](#)

Unggah Objek ke Bucket Amazon S3

Gunakan cmdlet `Write-S3Object` untuk meng-unggah file dari sistem file lokal ke bucket Amazon S3 sebagai objek. Contoh di bawah ini menciptakan dan meng-unggah dua file HTML sederhana ke bucket Amazon S3, dan memverifikasi keberadaan objek yang di-unggah. Parameter `-File` ke `Write-S3Object` menentukan nama file dalam sistem file lokal. Parameter `-Key` menentukan nama yang akan dimiliki objek terkait di Amazon S3.

Amazon menyimpulkan konten-jenis objek dari ekstensi file, dalam hal ini, ".html".

```
PS > # Create the two files using here-strings and the Set-Content cmdlet  
PS > $index_html = @"  
>> <html>  
>>   <body>  
>>     <p>  
>>       Hello, World!
```

```

>>     </p>
>> </body>
>> </html>
>> "@"
>>
PS > $index_html | Set-Content index.html
PS > $error_html = @"
>> <html>
>>   <body>
>>     <p>
>>       This is an error page.
>>     </p>
>>   </body>
>> </html>
>> @"
>>
>>$error_html | Set-Content error.html
>># Upload the files to Amazon S3 using a foreach loop
>>foreach ($f in "index.html", "error.html") {
>> Write-S3Object -BucketName website-example -File $f -Key $f -CannedACLName public-
read
>> }
>>
PS > # Verify that the files were uploaded
PS > Get-S3BucketWebsite -BucketName website-example

IndexDocumentSuffix                                ErrorDocument
-----
index.html                                              error.html

```

Opsi ACL Kalengan

Nilai untuk menentukan kalengan ACLs dengan Alat untuk Windows PowerShell sama dengan yang digunakan oleh AWS SDK untuk .NET Perhatikan, bagaimanapun, ini berbeda dari nilai-nilai yang digunakan oleh tindakanPut Object Amazon S3. Alat untuk Windows PowerShell mendukung kaleng berikut ini ACLs:

- NoACL
- private
- public-read
- public-read-write

- `aws-exec-read`
- `authenticated-read`
- `bucket-owner-read`
- `bucket-owner-full-control`
- `log-delivery-write`

Untuk informasi selengkapnya tentang pengaturan canned ACL, lihat [Gambaran Umum Daftar Kontrol Akses](#).

Catatan Mengenai Unggahan Multipart

Jika Anda menggunakan API Amazon S3 untuk meng-upload file yang berukuran lebih besar dari 5 GB, Anda perlu menggunakan unggahan multipart. Namun, `Write-S3Object` cmdlet yang disediakan oleh Tools for Windows PowerShell dapat secara transparan menangani unggahan file yang lebih besar dari 5 GB.

Uji Situs Webnya

Pada titik ini, Anda dapat menguji situs web dengan menavigasi ke sana menggunakan browser. URLs untuk situs web statis yang dihosting di Amazon S3 ikuti format standar.

```
http://<bucket-name>.s3-website-<region>.amazonaws.com
```

Misalnya:

```
http://website-example.s3-website-us-west-1.amazonaws.com
```

Lihat Juga

- [Bekerja dengan AWS layanan di Alat AWS untuk PowerShell](#)
- [Letakkan Objek \(Referensi API Amazon S3\)](#)
- [Kalangan ACLs \(Referensi API Amazon S3\)](#)

Hapus Obyek dan Bucket Amazon S3

Bagian ini menjelaskan cara menghapus situs web yang Anda buat di bagian sebelumnya. Anda cukup menghapus objek untuk file HTML, dan kemudian menghapus bucket Amazon S3 untuk situs tersebut.

Pertama, jalankan cmdlet `Remove-S3Object` untuk menghapus objek untuk file HTML dari bucket Amazon S3.

```
PS > foreach ( $obj in "index.html", "error.html" ) {  
>> Remove-S3Object -BucketName website-example -Key $obj  
>> }  
>>  
IsDeleteMarker  
-----  
False
```

Respon `False` merupakan artefak yang diharapkan dari cara Amazon S3 memproses permintaan. Dalam hal ini, ini bukanlah masalah.

Sekarang Anda dapat menjalankan cmdlet `Remove-S3Bucket` untuk menghapus bucket Amazon S3 yang sudah kosong untuk situs tersebut.

```
PS > Remove-S3Bucket -BucketName website-example  
  
RequestId      : E480ED92A2EC703D  
AmazonId2      : k6tqaqC1nMkoeYwbuJXUx1/UDa49BJd6dfLN0Ls1mWYNPHjbc8/Nyvm6AGbWcc2P  
ResponseStream :  
Headers        : {x-amz-id-2, x-amz-request-id, Date, Server}  
Metadata       : {}  
ResponseXml    :
```

Di 1.1 dan versi yang lebih baru Alat AWS untuk PowerShell, Anda dapat menambahkan `DeleteBucketContent` parameter ke `Remove-S3Bucket`, yang pertama-tama menghapus semua objek dan versi objek di bucket yang ditentukan sebelum mencoba menghapus bucket itu sendiri. Bergantung pada jumlah objek atau versi objek dalam bucket, operasi ini dapat memerlukan waktu lama. Dalam versi Tools untuk Windows PowerShell yang lebih tua dari 1.1, bucket harus kosong sebelum `Remove-S3Bucket` bisa menghapusnya.

Note

Kecuali Anda menambahkan `-Force` parameter, Alat AWS untuk PowerShell meminta Anda untuk konfirmasi sebelum cmdlet berjalan.

Lihat Juga

- [Bekerja dengan AWS layanan di Alat AWS untuk PowerShell](#)
- [Hapus Objek \(Referensi API Amazon S3\)](#)
- [DeleteBucket \(Referensi API Amazon S3\)](#)

Unggah Konten Teks Selaras ke Amazon S3

Cmdlet `Write-S3Object` mendukung kemampuan untuk meng-upload konten teks selaras ke Amazon S3. Menggunakan parameter `-Content` (alias `-Text`), Anda dapat menentukan konten berbasis teks yang harus di-unggah ke Amazon S3 tanpa perlu menempatkannya ke dalam file terlebih dahulu. Parameter tersebut menerima string satu baris sederhana serta here string yang berisi beberapa baris.

```
PS > # Specifying content in-line, single line text:
PS > write-s3object amzn-s3-demo-bucket -key myobject.txt -content "file content"

PS > # Specifying content in-line, multi-line text: (note final newline needed to end
in-line here-string)
PS > write-s3object amzn-s3-demo-bucket -key myobject.txt -content @"
>> line 1
>> line 2
>> line 3
>> "@
>>

PS > # Specifying content from a variable: (note final newline needed to end in-line
here-string)
PS > $x = @"
>> line 1
>> line 2
>> line 3
>> "@
>>
```

```
PS > write-s3object amzn-s3-demo-bucket -key myobject.txt -content $x
```

Amazon EC2 dan Alat untuk Windows PowerShell

Anda dapat melakukan tugas-tugas umum yang terkait dengan Amazon EC2 menggunakan file Alat AWS untuk PowerShell.

Contoh perintah yang ditampilkan di sini mengasumsikan bahwa Anda telah menetapkan kredensi default dan wilayah default untuk sesi Anda PowerShell. Oleh karena itu, kami tidak menyertakan kredensial atau wilayah saat kami membuka cmdlet. Lihat informasi yang lebih lengkap di [Memulai dengan AWS Tools for Windows PowerShell](#).

Topik

- [Membuat Pasangan Kunci](#)
- [Buat Grup Keamanan Menggunakan Windows PowerShell](#)
- [Temukan Gambar Mesin Amazon Menggunakan Windows PowerShell](#)
- [Luncurkan EC2 Instans Amazon Menggunakan Windows PowerShell](#)

Membuat Pasangan Kunci

New-EC2KeyPairContoh berikut menciptakan key pair dan menyimpan dalam PowerShell variabel \$myPSKeyPair

```
PS > $myPSKeyPair = New-EC2KeyPair -KeyName myPSKeyPair
```

Kirim objek pasangan kunci ke dalam cmdlet Get-Member untuk melihat struktur objek.

```
PS > $myPSKeyPair | Get-Member
```

```
TypeName: Amazon.EC2.Model.KeyPair
```

Name	MemberType	Definition
----	-----	-----
Equals	Method	bool Equals(System.Object obj)
GetHashCode	Method	int GetHashCode()
GetType	Method	type GetType()
ToString	Method	string ToString()

KeyFingerprint	Property	System.String KeyFingerprint {get;set;}
KeyMaterial	Property	System.String KeyMaterial {get;set;}
KeyName	Property	System.String KeyName {get;set;}

Kirim objek pasangan kunci ke dalam cmdlet `Format-List` untuk melihat nilai-nilai anggota `KeyName`, `KeyFingerprint`, dan `KeyMaterial`. (Output telah dipotong agar lebih mudah dibaca.)

```
PS > $myPSKeyPair | Format-List KeyName, KeyFingerprint, KeyMaterial
```

```

KeyName           : myPSKeyPair
KeyFingerprint    : 09:06:70:8e:26:b6:e7:ef:8f:fe:4a:1d:bc:9c:6a:63:11:ac:ad:3c
KeyMaterial       : -----BEGIN RSA PRIVATE KEY-----
                    MIIIEogIBAAKCAQEAKK+ANYUS9c7niNjYfaCn6KYj/D0I6djnFoQE...
                    Mz6bt0xPcE7EMeH1wySUP8nouAS9xb1917+Vkd74bN9KmNcPa/Mu...
                    Zyn4vVe0Q5il/MpkrRogHq0B0rigeTeV5Yc3lv00RFFPu0Kz4kcm...
                    w3Jg8dKsWn0pl0pX7V3sRC02KgJIbejQUvBFGi50QK9bm4tXBIeC...
                    daxKIAQMtDUdmBDrhR1/YMv8itFe5DiLLbq7Ga+FDcS85NstBa3h...
                    iuskGkcvgWkcFqkLmRHRoDpPb+OdFsZtjHZDpMVfMA9tT8EdbkEF...
                    3SrNeqZPsxJJIX0odb3CxLJpg75JU5kyWnb0+sDNVHoJiZCULCr0...
                    GGLLfEgB95KjGIk7zEv2Q7K6s+DHclrDeMZwa7KFNRZuCuX7jssC...
                    x098abxMr3o3TNU6p1ZYRJEQ0oJr0W+kc+/8SWb8NIwfltwmJEy...
                    1BX9X8WFX/A8VLHrT1elrKmlkNECgYEAwltkV1p0JAFhz9p7ZFev...
                    vvVsPaF0Ev9bk9pqhx269PB50x2KokwCagDMMaYvasWobuLmNu/1...
                    lmwRx7KTeQ7W1J30LgxHA1QNMkip9c4Tb3q9vVc3t/fPf8vwfJ8C...
                    63g6N6rk2FkHZX1E62BgbewUd3eZ0S05Ip4VUdvtGcuc8/qa+e5C...
                    KXgyt9n164pMv+VaXfXkZhdLAdY0Khc9TGB9++VMSG5TrD15YJId...
                    gYALEI7m1jJKpHWAes0hiemw5VmKyIZpzGstSJsFStER1AjiETDH...
                    YAtnI4J8dRyP9I7B0V0n3wNfIjk85gi1/00c+j8S65giLAfndWGR...
                    9R9wIkm5BMUcSRRcDy0yuwKBgEbkOnGGSD0ah4HkvrUkepIbUDTD...
                    AnEBM1cXI5UT7BfKInpUihZi59QhgdK/hk0SmWhlZGWikJ5VizBf...
                    drkBr/vTKVRMTi31VFB7KkIV1xJxC5E/BZ+YdZEpWoCZAoGAC/Cd...
                    TTld5N6opg0XAcQJwzqoGa9ZMwc5Q9f4bfRc67emkw0ZAAwSsvWR...
                    x302duuy7/smTwWwskEWRK5IrUxoMv/VVYaqdzc0ajwieNrblr7c...
                    -----END RSA PRIVATE KEY-----

```

Anggota `KeyMaterial` menyimpan kunci privat untuk pasangan kunci. Kunci publik disimpan di AWS. Anda tidak dapat mengambil kunci publik dari AWS, tetapi Anda dapat memverifikasi kunci publik dengan membandingkan `KeyFingerprint` untuk kunci pribadi dengan yang dikembalikan dari AWS untuk kunci publik.

Melihat Sidik Jari Pasangan Kunci Anda

Anda dapat menggunakan cmdlet `Get-EC2KeyPair` untuk melihat sidik jari untuk pasangan kunci Anda.

```
PS > Get-EC2KeyPair -KeyName myPSKeyPair | format-list KeyName, KeyFingerprint
```

```
KeyName           : myPSKeyPair
KeyFingerprint    : 09:06:70:8e:26:b6:e7:ef:8f:fe:4a:1d:bc:9c:6a:63:11:ac:ad:3c
```

Menyimpan Kunci Privat Anda

Untuk menyimpan kunci privat ke file, kirim anggota `KeyFingerMaterial` ke cmdlet `Out-File`.

```
PS > $myPSKeyPair.KeyMaterial | Out-File -Encoding ascii myPSKeyPair.pem
```

Anda harus menentukan `-Encoding ascii` saat menuliskan kunci privat ke file. Jika tidak, alat seperti `openssl` mungkin tidak dapat membaca file dengan benar. Anda dapat memverifikasi bahwa format file yang dihasilkan benar dengan menggunakan perintah seperti berikut:

```
PS > openssl rsa -check < myPSKeyPair.pem
```

(`openssl` Alat ini tidak disertakan dengan Alat AWS untuk PowerShell atau AWS SDK untuk .NET.)

Menghapus Pasangan Kunci Anda

Anda memerlukan pasangan kunci Anda untuk meluncurkan dan terhubung ke sebuah instans. Setelah selesai menggunakan pasangan kunci, Anda dapat menghapusnya. Untuk menghapus kunci publik dari AWS, gunakan `Remove-EC2KeyPair` cmdlet. Saat diminta, tekan `Enter` untuk menghapus pasangan kunci.

```
PS > Remove-EC2KeyPair -KeyName myPSKeyPair
```

```
Confirm
Performing the operation "Remove-EC2KeyPair (DeleteKeyPair)" on target "myPSKeyPair".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is "Y"):
```

Variabel, `$myPSKeyPair`, masih ada di PowerShell sesi saat ini dan masih berisi informasi key pair. File `myPSKeyPair.pem` juga ada. Namun, kunci privat tidak berlaku lagi karena kunci publik untuk pasangan kunci tidak lagi disimpan di AWS.

Buat Grup Keamanan Menggunakan Windows PowerShell

Anda dapat menggunakan Alat AWS untuk PowerShell untuk membuat dan mengkonfigurasi grup keamanan. Jawabannya adalah ID dari grup keamanan.

Jika Anda perlu terhubung ke instans Anda, Anda harus mengkonfigurasi grup keamanan untuk mengizinkan lalu lintas SSH (Linux) atau lalu lintas RDP (Windows).

Topik

- [Prasyarat](#)
- [Membuat Grup Keamanan untuk EC2 -VPC](#)

Prasyarat

Anda memerlukan alamat IP publik komputer Anda, dalam notasi CIDR. Anda bisa mendapatkan alamat IP publik dari komputer lokal menggunakan layanan. Misalnya, Amazon menyediakan layanan berikut: <http://checkip.amazonaws.com/> atau <https://checkip.amazonaws.com/>. Untuk menemukan layanan lain yang menyediakan alamat IP Anda, gunakan frasa pencarian "apa alamat IP saya". Jika Anda terhubung melalui ISP atau dari belakang firewall Anda tanpa alamat IP statis, Anda perlu mencari tahu rentang alamat IP yang digunakan oleh komputer klien.

Warning

Jika Anda menentukan `0.0.0.0/0`, Anda mengaktifkan lalu lintas dari alamat IP manapun di dunia. Untuk protokol SSH dan RDP, Anda mungkin menganggap ini dapat diterima untuk waktu yang singkat di lingkungan pengujian, tetapi tidak aman untuk lingkungan produksi. Dalam produksi, pastikan untuk mengotorisasi akses hanya dari alamat IP individu atau rentang alamat yang sesuai.

Membuat Grup Keamanan untuk EC2 -VPC

Warning

EC2-Classic pensiun pada 15 Agustus 2022. Kami menyarankan Anda bermigrasi dari EC2 - Classic ke VPC. Untuk informasi lebih lanjut, lihat posting blog [EC2-Jaringan Klasik Pensiun - Inilah Cara Mempersiapkan](#).

Contoh New-EC2SecurityGroup berikut ini menambahkan parameter -VpcId untuk membuat grup keamanan untuk VPC tertentu.

```
PS > $groupid = New-EC2SecurityGroup `
    -VpcId "vpc-da0013b3" `
    -GroupName "myPSSecurityGroup" `
    -GroupDescription "EC2-VPC from PowerShell"
```

Untuk melihat konfigurasi awal grup keamanan, gunakan cmdlet Get-EC2SecurityGroup. Secara default, grup keamanan untuk VPC berisi aturan yang memungkinkan semua lalu lintas ke luar. Perhatikan bahwa Anda tidak dapat mereferensikan grup keamanan untuk EC2 -VPC berdasarkan nama.

```
PS > Get-EC2SecurityGroup -GroupId sg-5d293231

OwnerId           : 123456789012
GroupName         : myPSSecurityGroup
GroupId           : sg-5d293231
Description       : EC2-VPC from PowerShell
IpPermissions     : {}
IpPermissionsEgress : {Amazon.EC2.Model.IpPermission}
VpcId             : vpc-da0013b3
Tags              : {}
```

Untuk menentukan izin untuk lalu lintas masuk pada TCP port 22 (SSH) dan TCP port 3389, gunakan cmdlet New-Object. Contoh skrip berikut menjabarkan izin untuk TCP port 22 dan 3389 dari alamat IP tunggal, 203.0.113.25/32.

```
$ip1 = new-object Amazon.EC2.Model.IpPermission
$ip1.IpProtocol = "tcp"
$ip1.FromPort = 22
$ip1.ToPort = 22
$ip1.IpRanges.Add("203.0.113.25/32")
$ip2 = new-object Amazon.EC2.Model.IpPermission
$ip2.IpProtocol = "tcp"
$ip2.FromPort = 3389
$ip2.ToPort = 3389
$ip2.IpRanges.Add("203.0.113.25/32")
Grant-EC2SecurityGroupIngress -GroupId $groupid -IpPermissions @( $ip1, $ip2 )
```

Untuk memverifikasi grup keamanan telah diperbarui, gunakan cmdlet `Get-EC2SecurityGroup` lagi.

```
PS > Get-EC2SecurityGroup -GroupIds sg-5d293231

OwnerId           : 123456789012
GroupName         : myPSSecurityGroup
GroupId           : sg-5d293231
Description       : EC2-VPC from PowerShell
IpPermissions     : {Amazon.EC2.Model.IpPermission}
IpPermissionsEgress : {Amazon.EC2.Model.IpPermission}
VpcId             : vpc-da0013b3
Tags              : {}
```

Untuk melihat aturan inbound, Anda dapat mengambil proeprti `IpPermissions` dari objek pengambilan yang dikembalikan oleh perintah sebelumnya.

```
PS > (Get-EC2SecurityGroup -GroupIds sg-5d293231).IpPermissions

IpProtocol      : tcp
FromPort        : 22
ToPort          : 22
UserIdGroupPairs : {}
IpRanges        : {203.0.113.25/32}

IpProtocol      : tcp
FromPort        : 3389
ToPort          : 3389
UserIdGroupPairs : {}
IpRanges        : {203.0.113.25/32}
```

Temukan Gambar Mesin Amazon Menggunakan Windows PowerShell

Saat meluncurkan EC2 instans Amazon, Anda menentukan Amazon Machine Image (AMI) untuk dijadikan template instance. Namun, IDs untuk AWS Windows sering AMIs berubah karena AWS menyediakan baru AMIs dengan pembaruan terbaru dan peningkatan keamanan. Anda dapat menggunakan [Get-EC2Image](#) dan [Get-EC2ImageByName](#) cmdlet untuk menemukan Windows saat ini AMIs dan mendapatkannya. IDs

Topik

- [Get-EC2Image](#)

- [Get-EC2ImageByName](#)

Get-EC2Image

Get-EC2ImageCmdlet mengambil daftar yang dapat Anda AMIs gunakan.

Gunakan -Owner parameter dengan nilai array amazon, self sehingga hanya Get-EC2Image mengambil AMIs yang milik Amazon atau milik Anda. Dalam konteks ini, Anda merujuk ke pengguna yang kredensialnya Anda gunakan untuk membuka cmdlet.

```
PS > Get-EC2Image -Owner amazon, self
```

Anda dapat melihat hasilnya menggunakan parameter -Filter. Untuk menentukan filter, buat objek jenis Amazon.EC2.Model.Filter. Misalnya, gunakan filter berikut untuk hanya menampilkan Windows AMIs.

```
$platform_values = New-Object 'collections.generic.list[string]'
$platform_values.add("windows")
$filter_platform = New-Object Amazon.EC2.Model.Filter -Property @{Name = "platform";
    Values = $platform_values}
Get-EC2Image -Owner amazon, self -Filter $filter_platform
```

Berikut ini adalah contoh dari salah satu yang AMIs dikembalikan oleh cmdlet; output aktual dari perintah sebelumnya memberikan informasi bagi banyak orang. AMIs

```
Architecture      : x86_64
BlockDeviceMappings : {/dev/sda1, xvdca, xvdcb, xvdcc...}
CreationDate      : 2019-06-12T10:41:31.000Z
Description       : Microsoft Windows Server 2019 Full Locale English with SQL Web
                  2017 AMI provided by Amazon
EnaSupport        : True
Hypervisor        : xen
ImageId           : ami-000226b77608d973b
ImageLocation     : amazon/Windows_Server-2019-English-Full-SQL_2017_Web-2019.06.12
ImageOwnerAlias   : amazon
ImageType         : machine
KernelId          :
Name              : Windows_Server-2019-English-Full-SQL_2017_Web-2019.06.12
OwnerId          : 801119661308
Platform         : Windows
```

```
ProductCodes      : {}  
Public            : True  
RamdiskId         :  
RootDeviceName    : /dev/sda1  
RootDeviceType    : ebs  
SriovNetSupport   : simple  
State             : available  
StateReason       :  
Tags              : {}  
VirtualizationType : hvm
```

Get-EC2ImageByName

Get-EC2ImageByNameCmdlet memungkinkan Anda untuk memfilter daftar AWS Windows AMIs berdasarkan jenis konfigurasi server yang Anda minati.

Ketika dijalankan tanpa parameter, sebagai berikut, cmdlet memancarkan rangkaian lengkap nama filter saat ini:

```
PS > Get-EC2ImageByName  
  
WINDOWS_2016_BASE  
WINDOWS_2016_NANO  
WINDOWS_2016_CORE  
WINDOWS_2016_CONTAINER  
WINDOWS_2016_SQL_SERVER_ENTERPRISE_2016  
WINDOWS_2016_SQL_SERVER_STANDARD_2016  
WINDOWS_2016_SQL_SERVER_WEB_2016  
WINDOWS_2016_SQL_SERVER_EXPRESS_2016  
WINDOWS_2012R2_BASE  
WINDOWS_2012R2_CORE  
WINDOWS_2012R2_SQL_SERVER_EXPRESS_2016  
WINDOWS_2012R2_SQL_SERVER_STANDARD_2016  
WINDOWS_2012R2_SQL_SERVER_WEB_2016  
WINDOWS_2012R2_SQL_SERVER_EXPRESS_2014  
WINDOWS_2012R2_SQL_SERVER_STANDARD_2014  
WINDOWS_2012R2_SQL_SERVER_WEB_2014  
WINDOWS_2012_BASE  
WINDOWS_2012_SQL_SERVER_EXPRESS_2014  
WINDOWS_2012_SQL_SERVER_STANDARD_2014  
WINDOWS_2012_SQL_SERVER_WEB_2014  
WINDOWS_2012_SQL_SERVER_EXPRESS_2012  
WINDOWS_2012_SQL_SERVER_STANDARD_2012
```

```

WINDOWS_2012_SQL_SERVER_WEB_2012
WINDOWS_2012_SQL_SERVER_EXPRESS_2008
WINDOWS_2012_SQL_SERVER_STANDARD_2008
WINDOWS_2012_SQL_SERVER_WEB_2008
WINDOWS_2008R2_BASE
WINDOWS_2008R2_SQL_SERVER_EXPRESS_2012
WINDOWS_2008R2_SQL_SERVER_STANDARD_2012
WINDOWS_2008R2_SQL_SERVER_WEB_2012
WINDOWS_2008R2_SQL_SERVER_EXPRESS_2008
WINDOWS_2008R2_SQL_SERVER_STANDARD_2008
WINDOWS_2008R2_SQL_SERVER_WEB_2008
WINDOWS_2008RTM_BASE
WINDOWS_2008RTM_SQL_SERVER_EXPRESS_2008
WINDOWS_2008RTM_SQL_SERVER_STANDARD_2008
WINDOWS_2008_BEANSTALK_IIS75
WINDOWS_2012_BEANSTALK_IIS8
VPC_NAT

```

Untuk mempersempit kumpulan gambar yang dikembalikan, tentukan satu atau beberapa nama filter menggunakan parameter `Names`.

```
PS > Get-EC2ImageByName -Names WINDOWS_2016_CORE
```

```

Architecture      : x86_64
BlockDeviceMappings : {/dev/sda1, xvdca, xvdc, xvdc...}
CreationDate       : 2019-08-16T09:36:09.000Z
Description        : Microsoft Windows Server 2016 Core Locale English AMI provided by
  Amazon
EnaSupport         : True
Hypervisor         : xen
ImageId            : ami-06f2a2afca06f15fc
ImageLocation      : amazon/Windows_Server-2016-English-Core-Base-2019.08.16
ImageOwnerAlias     : amazon
ImageType          : machine
KernelId           :
Name               : Windows_Server-2016-English-Core-Base-2019.08.16
OwnerId            : 801119661308
Platform           : Windows
ProductCodes       : {}
Public             : True
RamdiskId          :
RootDeviceName     : /dev/sda1
RootDeviceType     : ebs

```

```
SriovNetSupport    : simple
State              : available
StateReason        :
Tags               : {}
VirtualizationType : hvm
```

Luncurkan EC2 Instans Amazon Menggunakan Windows PowerShell

Untuk meluncurkan EC2 instans Amazon, Anda memerlukan key pair dan grup keamanan yang Anda buat di bagian sebelumnya. Anda juga memerlukan ID dari Amazon Machine Image (AMI). Untuk informasi lebih lanjut, lihat dokumentasi berikut ini:

- [Membuat Pasangan Kunci](#)
- [Buat Grup Keamanan Menggunakan Windows PowerShell](#)
- [Temukan Gambar Mesin Amazon Menggunakan Windows PowerShell](#)

Important

Jika Anda meluncurkan sebuah instans yang tidak berada dalam Tingkat Gratis, Anda akan ditagih setelah Anda meluncurkan instans dan dikenai biaya selama instans tersebut berjalan, meskipun instans dalam posisi siaga.

Topik

- [Meluncurkan Instans di VPC](#)
- [Meluncurkan Instans Spot di VPC](#)

Meluncurkan Instans di VPC

Warning

EC2-Classic pensiun pada 15 Agustus 2022. Kami menyarankan Anda bermigrasi dari EC2 - Classic ke VPC. Untuk informasi lebih lanjut, lihat posting blog [EC2-Jaringan Klasik Pensiun - Inilah Cara Mempersiapkan](#).

Perintah berikut membuat satu instans `m1.small` di subnet privat yang ditentukan. Grup keamanan harus berlaku untuk subnet yang ditentukan.

```
PS > New-EC2Instance `
    -ImageId ami-c49c0dac `
    -MinCount 1 -MaxCount 1 `
    -KeyName myPSKeyPair `
    -SecurityGroupId sg-5d293231 `
    -InstanceType m1.small `
    -SubnetId subnet-d60013bf
```

```
ReservationId    : r-b70a0ef1
OwnerId          : 123456789012
RequesterId      :
Groups           : {}
GroupName        : {}
Instances        : {}
```

Instans Anda awalnya ada dalam keadaan pending, tetapi setelah beberapa menit kemudian dalam keadaan running. Untuk melihat informasi tentang instans Anda, gunakan cmdlet `Get-EC2Instance`. Jika Anda memiliki lebih dari satu instans, Anda dapat memfilter hasil pada ID reservasi menggunakan parameter `Filter`. Pertama, buat obyek tipe `Amazon.EC2.Model.Filter`. Selanjutnya, panggil `Get-EC2Instance` yang menggunakan filter, dan kemudian tampilkan properti `Instances`.

```
PS > $reservation = New-Object 'collections.generic.list[string]'
PS > $reservation.add("r-b70a0ef1")
PS > $filter_reservation = New-Object Amazon.EC2.Model.Filter -Property @{Name =
    "reservation-id"; Values = $reservation}
PS > (Get-EC2Instance -Filter $filter_reservation).Instances
```

```
AmiLaunchIndex    : 0
Architecture      : x86_64
BlockDeviceMappings : {/dev/sda1}
ClientToken       :
EbsOptimized      : False
Hypervisor        : xen
IamInstanceProfile :
ImageId           : ami-c49c0dac
InstanceId        : i-5203422c
InstanceLifecycle :
InstanceType      : m1.small
```

```

KernelId           :
KeyName            : myPSKeyPair
LaunchTime         : 12/2/2018 3:38:52 PM
Monitoring         : Amazon.EC2.Model.Monitoring
NetworkInterfaces  : {}
Placement          : Amazon.EC2.Model.Placement
Platform           : Windows
PrivateDnsName     :
PrivateIpAddress   : 10.25.1.11
ProductCodes       : {}
PublicDnsName      :
PublicIpAddress    : 198.51.100.245
RamdiskId          :
RootDeviceName     : /dev/sda1
RootDeviceType     : ebs
SecurityGroups     : {myPSSecurityGroup}
SourceDestCheck    : True
SpotInstanceRequestId :
SriovNetSupport    :
State              : Amazon.EC2.Model.InstanceState
StateReason        :
StateTransitionReason :
SubnetId           : subnet-d60013bf
Tags               : {}
VirtualizationType : hvm
VpcId              : vpc-a01106c2

```

Meluncurkan Instans Spot di VPC

Contoh skrip berikut meminta Instans Spot di subnet yang ditentukan. Grup keamanan harus menjadi salah satu yang Anda buat untuk VPC yang berisi subnet yang ditentukan.

```

$interface1 = New-Object Amazon.EC2.Model.InstanceNetworkInterfaceSpecification
$interface1.DeviceIndex = 0
$interface1.SubnetId = "subnet-b61f49f0"
$interface1.PrivateIpAddress = "10.0.1.5"
$interface1.Groups.Add("sg-5d293231")
Request-EC2SpotInstance `
    -SpotPrice 0.007 `
    -InstanceCount 1 `
    -Type one-time `
    -LaunchSpecification_ImageId ami-7527031c `
    -LaunchSpecification_InstanceType m1.small `

```

```
-Region us-west-2 `
-LaunchSpecification_NetworkInterfaces $interface1
```

AWS Lambda dan Alat AWS untuk PowerShell

Dengan menggunakan [AWSLambdaPSCore](#) modul, Anda dapat mengembangkan AWS Lambda fungsi di PowerShell Core 6.0 menggunakan runtime .NET Core 2.1. PowerShell pengembang dapat mengelola AWS sumber daya dan menulis skrip otomatisasi di PowerShell lingkungan dengan menggunakan Lambda. PowerShell dukungan di Lambda memungkinkan Anda menjalankan PowerShell skrip atau fungsi sebagai respons terhadap peristiwa Lambda apa pun, seperti acara Amazon S3 atau acara terjadwal Amazon. CloudWatch AWSLambdaPSCore Modul ini adalah AWS modul terpisah untuk PowerShell; itu bukan bagian dari Alat AWS untuk PowerShell, juga tidak menginstal AWSLambda PSCore modul menginstal Alat AWS untuk PowerShell.

Setelah Anda menginstal AWSLambda PSCore modul, Anda dapat menggunakan PowerShell cmdlet yang tersedia — atau mengembangkan sendiri — untuk membuat fungsi tanpa server. Alat AWS Lambda untuk PowerShell modul mencakup templat proyek untuk aplikasi tanpa server PowerShell berbasis, dan alat untuk mempublikasikan proyek. AWS

AWSLambdaPSCore dukungan modul tersedia di semua wilayah yang mendukung Lambda. Untuk informasi selengkapnya tentang wilayah yang didukung, lihat [tabel wilayah AWS](#).

Prasyarat

Langkah-langkah berikut diperlukan sebelum Anda dapat menginstal dan menggunakan AWSLambda PSCore modul. Untuk detail selengkapnya tentang langkah-langkah ini, lihat [Menyiapkan Lingkungan PowerShell Pengembangan](#) di Panduan AWS Lambda Pengembang.

- Instal rilis yang benar PowerShell — Dukungan Lambda untuk PowerShell didasarkan pada rilis PowerShell Core 6.0 lintas platform. Anda dapat mengembangkan fungsi PowerShell Lambda di Windows, Linux, atau Mac. Jika Anda tidak PowerShell menginstal setidaknya rilis ini, instruksi tersedia di situs [web PowerShell dokumentasi Microsoft](#).
- Instal .NET Core 2.1 SDK — Karena PowerShell Core didasarkan pada .NET Core, dukungan Lambda PowerShell untuk menggunakan runtime .NET Core 2.1 Lambda yang sama untuk fungsi.NET Core dan Lambda. PowerShell Cmdlet PowerShell penerbitan Lambda menggunakan .NET Core 2.1 SDK untuk membuat paket penyebaran Lambda. .NET Core 2.1 SDK tersedia dari [Pusat Unduhan Microsoft](#). Pastikan untuk menginstal SDK, bukan Runtime.

Instal AWSLambda PSCore Modul

Setelah menyelesaikan prasyarat, Anda siap untuk menginstal modul. AWSLambda PSCore Jalankan perintah berikut dalam sesi PowerShell Core.

```
PS> Install-Module AWSLambdaPSCore -Scope CurrentUser
```

Anda siap untuk mulai mengembangkan fungsi Lambda di. PowerShell Untuk informasi selengkapanya tentang cara memulai, lihat [Model Pemrograman untuk Menulis Fungsi Lambda PowerShell](#) di Panduan AWS Lambda Pengembang.

Lihat Juga

- [Mengumumkan PowerShell Dukungan Lambda untuk Core AWS di Blog Pengembang](#)
- [AWSLambdaPSCore modul di situs PowerShell Galeri](#)
- [Menyiapkan Lingkungan PowerShell Pembangunan](#)
- [AWS Alat Lambda untuk Powershell aktif GitHub](#)
- [AWS Konsol Lambda](#)

Amazon SQS, Amazon SNS dan Alat untuk Windows PowerShell

Bagian ini menyajikan contoh yang menunjukkan cara:

- Membuat antrean Amazon SQS dan mendapatkan antrean ARN (Amazon Resource Name).
- Membuat topik Amazon SNS.
- Memberikan izin untuk topik Amazon SNS sehingga dapat mengirim pesan ke antrean.
- Berlangganan antrean ke topik SNS
- Berikan izin kepada pengguna atau AWS akun IAM untuk mempublikasikan ke topik SNS dan membaca pesan dari antrian SQS.
- Memverifikasi hasil dengan menerbitkan pesan ke topik dan membaca pesan dari antrean.

Membuat antrean Amazon SQS dan mendapatkan ARN antrean

Perintah berikut membuat antrean SQS di wilayah default Anda. Output menunjukkan URL dari antrean baru.

```
PS > New-SQSQueue -QueueName myQueue  
https://sqs.us-west-2.amazonaws.com/123456789012/myQueue
```

Perintah berikut mengambil ARN antrean.

```
PS > Get-SQSQueueAttribute -QueueUrl https://sqs.us-west-2.amazonaws.com/123456789012/  
myQueue -AttributeName QueueArn  
...  
QueueARN                : arn:aws:sqs:us-west-2:123456789012:myQueue  
...
```

Membuat topik Amazon SNS

Perintah berikut membuat topik SNS di wilayah default Anda, dan mengembalikan ARN topik baru.

```
PS > New-SNSTopic -Name myTopic  
arn:aws:sns:us-west-2:123456789012:myTopic
```

Memberikan izin untuk topik SNS

Skrip contoh berikut membuat antrean SQS dan topik SNS, dan memberikan izin untuk topik SNS sehingga dapat mengirim pesan ke antrean SQS:

```
# create the queue and topic to be associated  
$qurl = New-SQSQueue -QueueName "myQueue"  
$topicarn = New-SNSTopic -Name "myTopic"  
  
# get the queue ARN to inject into the policy; it will be returned  
# in the output's QueueARN member but we need to put it into a variable  
# so text expansion in the policy string takes effect  
$qarn = (Get-SQSQueueAttribute -QueueUrl $qurl -AttributeNames "QueueArn").QueueARN  
  
# construct the policy and inject arns  
$policy = @"  
{  
  "Version": "2012-10-17",  
  "Statement": {  
    "Effect": "Allow",  
    "Principal": "*",  
    "Action": "SQS:SendMessage",
```

```

        "Resource": "$qarn",
        "Condition": { "ArnEquals": { "aws:SourceArn": "$topicarn" } }
    }
}
"@

# set the policy
Set-SQSQueueAttribute -QueueUrl $qurl -Attribute @{ Policy=$policy }

```

Berlangganan antrean ke topik SNS

Perintah berikut berlangganan antrian myQueue ke topik SNS myTopic, dan mengembalikan ID Langganan:

```

PS > Connect-SNSNotification `
    -TopicARN arn:aws:sns:us-west-2:123456789012:myTopic `
    -Protocol SQS `
    -Endpoint arn:aws:sqs:us-west-2:123456789012:myQueue
arn:aws:sns:us-west-2:123456789012:myTopic:f8ff77c6-e719-4d70-8e5c-a54d41feb754

```

Verifikasi izin

Perintah berikut memberikan izin untuk melakukan tindakan sns:Publish pada topik myTopic

```

PS > Add-SNSPermission `
    -TopicArn arn:aws:sns:us-west-2:123456789012:myTopic `
    -Label ps-cmdlet-topic `
    -AWSAccountIds 123456789012 `
    -ActionNames publish

```

Perintah berikut memberikan izin untuk melakukan tindakan sqs:ReceiveMessage dan sqs:DeleteMessage pada antrean myQueue.

```

PS > Add-SQSPermission `
    -QueueUrl https://sqs.us-west-2.amazonaws.com/123456789012/myQueue `
    -AWSAccountId "123456789012" `
    -Label queue-permission `
    -ActionName SendMessage, ReceiveMessage

```

Verifikasi hasil

Perintah berikut menguji antrian dan topik baru Anda dengan menerbitkan pesan ke topik SNS myTopic dan mengembalikan MessageId.

```
PS > Publish-SNSMessage `
    -TopicArn arn:aws:sns:us-west-2:123456789012:myTopic `
    -Message "Have A Nice Day!"
728180b6-f62b-49d5-b4d3-3824bb2e77f4
```

Perintah berikut mengambil pesan dari antrian SQS myQueue dan menampilkannya.

```
PS > Receive-SQSMessage -QueueUrl https://sqs.us-west-2.amazonaws.com/123456789012/
myQueue
```

```
Attributes      : {}
Body            : {
    "Type" : "Notification",
    "MessageId" : "491c687d-b78d-5c48-b7a0-3d8d769ee91b",
    "TopicArn" : "arn:aws:sns:us-west-2:123456789012:myTopic",
    "Message" : "Have A Nice Day!",
    "Timestamp" : "2019-09-09T21:06:27.201Z",
    "SignatureVersion" : "1",
    "Signature" :
    "11E17A2+X0uJZnw3TlgcXz4C4KPLXZxbxoEMIirelh13u/oxkWmz5+9tJKFMns1Z0qQvKxk
+ExfEZcD5yWt6biVuBb8pyRmZ1b03hUENl3ayv2WQiQT1vpLpM7VEQN5m+hLIiPFcs
vyuGkJReV710JWPHnCN
+qTE2lId2RPkF0eGtLGawTsSPTWEvJdDbLl1f7E0zZ0q1niXTUtpsZ8Swx01X3Q06u9i9qBFt0ekJFZNJp6Avu05hIklb4yo
y0a8Yl9lWp7a7EoWaBn0zhCESe7o
kZC6ncBJWphX7KCGVYD0qhVf/5VDgBuv9w8T+higJyvr3WbaSvg==",
    "SigningCertURL" : "https://sns.us-west-2.amazonaws.com/
SimpleNotificationService-6aad65c2f9911b05cd53efda11f913f9.pem",
    "UnsubscribeURL" :
    "https://sns.us-west-2.amazonaws.com/?
Action=Unsubscribe&SubscriptionArn=arn:aws:sns:us-west-2:123456789012:myTopic:22b77de7-
a216-4000-9a23-bf465744ca84"
}
MD5ofBody      : 5b5ee4f073e9c618eda3718b594fa257
MD5ofMessageAttributes :
MessageAttributes : {}
MessageId      : 728180b6-f62b-49d5-b4d3-3824bb2e77f4
```

```

ReceiptHandle      :
AQEB2vvk1e5c0KFjeIWJticabkc664yuDEjhucnIOqdVUmie7bX7GiJb17F0enABUgaI2XjEcNPxixhVc/
wfsAJZLNHN18S1bQa0R/kD+Saqa40Ivfj8x3M40h1yM1cVKpYmhAzsYrAwAD5g5FvxNBD6zs
+HmXdkax2Wd+9AxrHlQZV5ur1MoByKWwBDbSqYJTJquCc10gWIak/sBx/
daBRMTiVQ4GHsrQWMVHtNC14q7Jy/0L2dkmb4dzJfJq0VbFSX1G+u/lrSLpgae+Dfux646y8yFiPFzY4ua4mCF/
SVUn63Spy
sHN12776axknhg3j9K/Xwj54DixdsegrKoLx+ctI
+0jzAetBR66Q1VhIoJAq7s0a2Msey0eM/Jjucg6Sr9VUnTWVhV8ErXmotoiEg==

```

CloudWatch dari AWS Tools for Windows PowerShell

Bagian ini menunjukkan contoh cara menggunakan Alat untuk Windows PowerShell untuk mempublikasikan data metrik kustom ke CloudWatch.

Contoh ini mengasumsikan bahwa Anda telah menetapkan kredensi default dan wilayah default untuk sesi Anda. PowerShell

Publikasikan Metrik Kustom ke CloudWatch Dasbor Anda

PowerShell Kode berikut menginisialisasi CloudWatch `MetricDatum` objek dan mempostingnya ke layanan. [Anda dapat melihat hasil operasi ini dengan menavigasi ke konsol. CloudWatch](#)

```

$dat = New-Object Amazon.CloudWatch.Model.MetricDatum
$dat.Timestamp = (Get-Date).ToUniversalTime()
$dat.MetricName = "New Posts"
$dat.Unit = "Count"
$dat.Value = ".50"
Write-CWMetricData -Namespace "Usage Metrics" -MetricData $dat

```

Perhatikan hal berikut:

- Informasi tanggal-waktu yang Anda gunakan untuk menginisialisasi `$dat.Timestamp` harus dalam Waktu Universal (UTC).
- Nilai yang Anda gunakan untuk menginisialisasi `$dat.Value` dapat berupa nilai string yang diberi tanda kutip, atau nilai numerik (tanpa tanda kutip). Contoh ini menunjukkan nilai string.

Lihat Juga

- [Bekerja dengan AWS layanan di Alat AWS untuk PowerShell](#)

- [AmazonCloudWatchClient.PutMetricData](#)(.NET SDK Referensi)
- [MetricDatum](#)(Referensi API Layanan)
- [CloudWatch Konsol Amazon](#)

Menggunakan ClientConfig parameter dalam cmdlet

`ClientConfigParameter` dapat digunakan untuk menentukan pengaturan konfigurasi tertentu ketika Anda terhubung ke layanan. Sebagian besar properti yang mungkin dari parameter ini didefinisikan di [Amazon.Runtime.ClientConfig](#) kelas, yang diwarisi ke APIs untuk AWS layanan. Untuk contoh pewarisan sederhana, lihat [Amazon.Keyspaces.AmazonKeyspacesConfig](#) kelasnya. Selain itu, beberapa layanan mendefinisikan properti tambahan yang hanya sesuai untuk layanan itu. Untuk contoh properti tambahan yang telah didefinisikan, lihat [Amazon.S3.AmazonS3Config](#) kelas, khususnya `ForcePathStyle` properti.

Menggunakan **ClientConfig** parameter

Untuk menggunakan `ClientConfig` parameter, Anda dapat menentukannya pada baris perintah sebagai `ClientConfig` objek atau menggunakan PowerShell percikan untuk meneruskan kumpulan nilai parameter ke perintah sebagai unit. Metode-metode ini ditunjukkan dalam contoh berikut. Contoh mengasumsikan bahwa `AWS.Tools.S3` modul telah diinstal dan diimpor, dan bahwa Anda memiliki profil `[default]` kredensial dengan izin yang sesuai.

Mendefinisikan objek **ClientConfig**

```
$s3Config = New-Object -TypeName Amazon.S3.AmazonS3Config
$s3Config.ForcePathStyle = $true
$s3Config.Timeout = [TimeSpan]::FromMilliseconds(150000)
Get-S3Object -BucketName <BUCKET_NAME> -ClientConfig $s3Config
```

Menambahkan **ClientConfig** properti dengan menggunakan PowerShell percikan

```
$params=@{
    ClientConfig=@{
        ForcePathStyle=$true
        Timeout=[TimeSpan]::FromMilliseconds(150000)
    }
    BucketName="<BUCKET_NAME>"
}
```

```
}  
  
Get-S3Object @params
```

Menggunakan properti yang tidak ditentukan

Saat menggunakan PowerShell percikan, jika Anda menentukan `ClientConfig` properti yang tidak ada, Alat AWS untuk PowerShell tidak mendeteksi kesalahan hingga runtime, pada saat itu ia mengembalikan pengecualian. Memodifikasi contoh dari atas:

```
$params=@{  
    ClientConfig=@{  
        ForcePathStyle=$true  
        UndefinedProperty="Value"  
        Timeout=[TimeSpan]::FromMilliseconds(150000)  
    }  
    BucketName="<BUCKET_NAME>"  
}  
  
Get-S3Object @params
```

Contoh ini menghasilkan pengecualian yang mirip dengan berikut ini:

```
Cannot bind parameter 'ClientConfig'. Cannot create object of type  
"Amazon.S3.AmazonS3Config". The UndefinedProperty property was not found for the  
Amazon.S3.AmazonS3Config object.
```

Menentukan Wilayah AWS

Anda dapat menggunakan `ClientConfig` parameter Wilayah AWS untuk mengatur perintah. Wilayah diatur melalui `RegionEndpoint` properti. Alat AWS untuk PowerShell Menghitung Wilayah yang akan digunakan sesuai dengan prioritas berikut:

1. `-RegionParameter`nya
2. Wilayah dilewatkan dalam `ClientConfig` parameter
3. Keadaan PowerShell sesi
4. AWS `configFile` yang dibagikan
5. Variabel lingkungan

6. Metadata EC2 instans Amazon, jika diaktifkan.

Alat untuk contoh kode PowerShell V4

Contoh kode dalam topik ini menunjukkan cara menggunakan Alat AWS untuk PowerShell V4 dengan AWS.

Dasar-dasar adalah contoh kode yang menunjukkan kepada Anda bagaimana melakukan operasi penting dalam suatu layanan.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Skenario adalah contoh kode yang menunjukkan kepada Anda bagaimana menyelesaikan tugas tertentu dengan memanggil beberapa fungsi dalam layanan atau dikombinasikan dengan yang lain Layanan AWS.

Beberapa layanan berisi kategori contoh tambahan yang menunjukkan cara memanfaatkan pustaka atau fungsi khusus untuk layanan.

Layanan

- [Contoh ACM menggunakan Alat untuk V4 PowerShell](#)
- [Contoh Application Auto Scaling menggunakan Alat untuk V4 PowerShell](#)
- [AppStream 2.0 contoh menggunakan Alat untuk PowerShell V4](#)
- [Contoh Aurora menggunakan Alat untuk V4 PowerShell](#)
- [Contoh Auto Scaling menggunakan Alat untuk V4 PowerShell](#)
- [AWS Budgets contoh menggunakan Alat untuk PowerShell V4](#)
- [AWS Cloud9 contoh menggunakan Alat untuk PowerShell V4](#)
- [AWS CloudFormation contoh menggunakan Alat untuk PowerShell V4](#)
- [CloudFront contoh menggunakan Alat untuk PowerShell V4](#)
- [CloudTrail contoh menggunakan Alat untuk PowerShell V4](#)
- [CloudWatch contoh menggunakan Alat untuk PowerShell V4](#)
- [CodeCommit contoh menggunakan Alat untuk PowerShell V4](#)
- [CodeDeploy contoh menggunakan Alat untuk PowerShell V4](#)

- [CodePipeline contoh menggunakan Alat untuk PowerShell V4](#)
- [Contoh Identitas Amazon Cognito menggunakan Alat untuk V4 PowerShell](#)
- [AWS Config contoh menggunakan Alat untuk PowerShell V4](#)
- [Contoh Device Farm menggunakan Alat untuk PowerShell V4](#)
- [AWS Directory Service contoh menggunakan Alat untuk PowerShell V4](#)
- [AWS DMS contoh menggunakan Alat untuk PowerShell V4](#)
- [Contoh DynamoDB menggunakan Alat untuk V4 PowerShell](#)
- [EC2 Contoh Amazon menggunakan Alat untuk PowerShell V4](#)
- [Contoh Amazon ECR menggunakan Alat untuk V4 PowerShell](#)
- [Contoh Amazon ECS menggunakan Alat untuk PowerShell V4](#)
- [Amazon EFS contoh menggunakan Alat untuk PowerShell V4](#)
- [Contoh Amazon EKS menggunakan Alat untuk PowerShell V4](#)
- [Elastic Load Balancing - Contoh versi 1 menggunakan Alat untuk V4 PowerShell](#)
- [Elastic Load Balancing - Contoh versi 2 menggunakan Alat untuk V4 PowerShell](#)
- [FSx Contoh Amazon menggunakan Alat untuk PowerShell V4](#)
- [AWS Glue contoh menggunakan Alat untuk PowerShell V4](#)
- [AWS Health contoh menggunakan Alat untuk PowerShell V4](#)
- [Contoh IAM menggunakan Alat untuk V4 PowerShell](#)
- [Contoh Kinesis menggunakan Alat untuk V4 PowerShell](#)
- [Contoh Lambda menggunakan Alat untuk V4 PowerShell](#)
- [Contoh Amazon ML menggunakan Alat untuk PowerShell V4](#)
- [Contoh Macie menggunakan Alat untuk V4 PowerShell](#)
- [AWS OpsWorks contoh menggunakan Alat untuk PowerShell V4](#)
- [Daftar Harga AWS contoh menggunakan Alat untuk PowerShell V4](#)
- [Contoh Resource Groups menggunakan Alat untuk PowerShell V4](#)
- [Contoh API Penandaan Resource Groups menggunakan Alat untuk V4 PowerShell](#)
- [Contoh rute 53 menggunakan Alat untuk PowerShell V4](#)
- [Contoh Amazon S3 menggunakan Alat untuk V4 PowerShell](#)

- [Contoh S3 Glacier menggunakan Alat untuk V4 PowerShell](#)
- [Contoh Security Hub menggunakan Alat untuk PowerShell V4](#)
- [Amazon SES contoh menggunakan Alat untuk PowerShell V4](#)
- [Amazon SES API v2 contoh menggunakan Alat untuk PowerShell V4](#)
- [Contoh Amazon SNS menggunakan Alat untuk V4 PowerShell](#)
- [Contoh Amazon SQS menggunakan Alat untuk V4 PowerShell](#)
- [AWS STS contoh menggunakan Alat untuk PowerShell V4](#)
- [Dukungan contoh menggunakan Alat untuk PowerShell V4](#)
- [Contoh Systems Manager menggunakan Alat untuk PowerShell V4](#)
- [Contoh Amazon Translate menggunakan Alat untuk PowerShell V4](#)
- [AWS WAFV2 contoh menggunakan Alat untuk PowerShell V4](#)
- [WorkSpaces contoh menggunakan Alat untuk PowerShell V4](#)

Contoh ACM menggunakan Alat untuk V4 PowerShell

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan ACM.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

Get-ACMCertificate

Contoh kode berikut menunjukkan cara menggunakan `Get-ACMCertificate`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menunjukkan cara mengembalikan sertifikat dan rantainya menggunakan ARN sertifikat.

```
Get-ACMCertificate -CertificateArn "arn:aws:acm:us-east-1:123456789012:certificate/12345678-1234-1234-1234-123456789012"
```

- Untuk detail API, lihat [GetCertificatedi Referensi Alat AWS untuk PowerShell Cmdlet \(V4\)](#).

Get-ACMCertificateDetail

Contoh kode berikut menunjukkan cara menggunakan `Get-ACMCertificateDetail`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan rincian sertifikat yang ditentukan.

```
Get-ACMCertificateDetail -CertificateArn "arn:aws:acm:us-east-1:123456789012:certificate/12345678-1234-1234-1234-123456789012"
```

Output:

```
CertificateArn      : arn:aws:acm:us-east-1:123456789012:certificate/12345678-1234-1234-1234-123456789012
CreatedAt          : 1/21/2016 5:55:59 PM
DomainName         : www.example.com
DomainValidationOptions : {www.example.com}
InUseBy            : {}
IssuedAt           : 1/1/0001 12:00:00 AM
Issuer             :
KeyAlgorithm        : RSA-2048
NotAfter           : 1/1/0001 12:00:00 AM
NotBefore          : 1/1/0001 12:00:00 AM
RevocationReason    :
RevokedAt          : 1/1/0001 12:00:00 AM
Serial             :
SignatureAlgorithm   : SHA256WITHRSA
Status             : PENDING_VALIDATION
Subject            : CN=www.example.com
SubjectAlternativeNames : {www.example.net}
```

- Untuk detail API, lihat [DescribeCertificate](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-ACMCertificateList

Contoh kode berikut menunjukkan cara menggunakan `Get-ACMCertificateList`.

Alat untuk PowerShell V4

Contoh 1: Mengambil daftar semua sertifikat Anda ARNs dan nama domain untuk masing-masing. Cmdlet akan secara otomatis melakukan paginasi untuk mengambil semua file. ARNs Untuk mengontrol pagination secara manual, gunakan `MaxItems` parameter - untuk mengontrol berapa banyak sertifikat ARNs yang dikembalikan untuk setiap panggilan layanan dan `NextToken` parameter - untuk menunjukkan titik awal untuk setiap panggilan.

```
Get-ACMCertificateList
```

Output:

```
CertificateArn
DomainName
-----
-----
arn:aws:acm:us-east-1:123456789012:certificate/12345678-1234-1234-1234-123456789012
www.example.com
```

Contoh 2: Mengambil daftar semua sertifikat Anda ARNs di mana status sertifikat cocok pada status yang disediakan.

```
Get-ACMCertificateList -CertificateStatus "VALIDATION_TIMED_OUT","FAILED"
```

Contoh 3: Contoh ini mengembalikan daftar semua sertifikat di wilayah us-east-1 yang memiliki tipe kunci RSA_2048, dan penggunaan kunci yang diperluas, atau tujuan, dari CODE_SIGNING. Anda dapat menemukan nilai untuk parameter pemfilteran ini di topik referensi `ListCertificates Filter API`: https://docs.aws.amazon.com/acm/latest/APIReference/API_Filters.html.

```
Get-ACMCertificateList -Region us-east-1 -Includes_KeyType RSA_2048 -
Includes_ExtendedKeyUsage CODE_SIGNING
```

Output:

```

CertificateArn
DomainName
-----
-----
arn:aws:acm:us-east-1:8xxxxxxxxxxx:certificate/xxxxxxxx-d7c0-48c1-af8d-2133d8f30zzz
*.route53docs.com
arn:aws:acm:us-east-1:8xxxxxxxxxxx:certificate/xxxxxxxx-98a5-443d-a734-800430c80zzz
nerdzizm.net
arn:aws:acm:us-east-1:8xxxxxxxxxxx:certificate/xxxxxxxx-2be6-4376-8fa7-bad559525zzz

arn:aws:acm:us-east-1:8xxxxxxxxxxx:certificate/xxxxxxxx-e7ca-44c5-803e-24d9f2f36zzz

arn:aws:acm:us-east-1:8xxxxxxxxxxx:certificate/xxxxxxxx-1241-4b71-80b1-090305a62zzz

arn:aws:acm:us-east-1:8xxxxxxxxxxx:certificate/xxxxxxxx-8709-4568-8c64-f94617c99zzz

arn:aws:acm:us-east-1:8xxxxxxxxxxx:certificate/xxxxxxxx-a8fa-4a61-98cf-e08ccc0eezzz

arn:aws:acm:us-east-1:8xxxxxxxxxxx:certificate/xxxxxxxx-fa47-40fe-a714-2d277d3eezzz
*.route53docs.com

```

- Untuk detail API, lihat [ListCertificates](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-ACMCertificate

Contoh kode berikut menunjukkan cara menggunakan `New-ACMCertificate`.

Alat untuk PowerShell V4

Contoh 1: Membuat sertifikat baru. Layanan mengembalikan ARN sertifikat baru.

```
New-ACMCertificate -DomainName "www.example.com"
```

Output:

```
arn:aws:acm:us-east-1:123456789012:certificate/12345678-1234-1234-1234-123456789012
```

Contoh 2: Membuat sertifikat baru. Layanan mengembalikan ARN sertifikat baru.

```
New-ACMCertificate -DomainName "www.example.com" -SubjectAlternativeName
"example.com", "www.example.net"
```

Output:

```
arn:aws:acm:us-east-1:123456789012:certificate/12345678-1234-1234-1234-123456789012
```

- Untuk detail API, lihat [RequestCertificate](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-ACMCertificate

Contoh kode berikut menunjukkan cara menggunakan `Remove-ACMCertificate`.

Alat untuk PowerShell V4

Contoh 1: Menghapus sertifikat yang diidentifikasi oleh ARN yang disediakan dan kunci pribadi terkait. Cmdlet akan meminta konfirmasi sebelum melanjutkan; tambahkan sakelar `-Force` untuk menekan konfirmasi.

```
Remove-ACMCertificate -CertificateArn "arn:aws:acm:us-east-1:123456789012:certificate/12345678-1234-1234-1234-123456789012"
```

- Untuk detail API, lihat [DeleteCertificate](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Send-ACMValidationEmail

Contoh kode berikut menunjukkan cara menggunakan `Send-ACMValidationEmail`.

Alat untuk PowerShell V4

Contoh 1: Permintaan agar email untuk memvalidasi kepemilikan domain untuk 'www.example.com' dikirim. Jika \$ shell Anda `ConfirmPreference` disetel ke 'Medium' atau lebih rendah, cmdlet akan meminta konfirmasi sebelum memproses. Tambahkan sakelar `-Force` untuk menekan permintaan konfirmasi.

```
$params = @{  
    CertificateArn="arn:aws:acm:us-east-1:123456789012:certificate/12345678-1234-1234-1234-123456789012"  
    Domain="www.example.com"  
    ValidationDomain="example.com"  
}  
Send-ACMValidationEmail @params
```

- Untuk detail API, lihat [ResendValidationEmail](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Contoh Application Auto Scaling menggunakan Alat untuk V4 PowerShell

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan Application Auto Scaling.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

Add-AASScalableTarget

Contoh kode berikut menunjukkan cara menggunakan Add-AASScalableTarget.

Alat untuk PowerShell V4

Contoh 1: Cmdlet ini mendaftarkan atau memperbarui target yang dapat diskalakan. Target yang dapat diskalakan adalah sumber daya yang dapat diskalakan dan diskalakan oleh Application Auto Scaling.

```
Add-AASScalableTarget -ServiceNamespace AppStream -ResourceId fleet/MyFleet -  
ScalableDimension appstream:fleet:DesiredCapacity -MinCapacity 2 -MaxCapacity 10
```

- Untuk detail API, lihat [RegisterScalableTarget](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-AASScalableTarget

Contoh kode berikut menunjukkan cara menggunakan `Get-AASScalableTarget`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini akan memberikan informasi tentang target Application Autoscaling Scaling Scalable di namespace yang ditentukan.

```
Get-AASScalableTarget -ServiceNamespace "AppStream"
```

Output:

```
CreationTime      : 11/7/2019 2:30:03 AM
MaxCapacity       : 5
MinCapacity       : 1
ResourceId        : fleet/Test
RoleARN           : arn:aws:iam::012345678912:role/aws-
service-role/appstream.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_AppStreamFleet
ScalableDimension : appstream:fleet:DesiredCapacity
ServiceNamespace  : appstream
SuspendedState    : Amazon.ApplicationAutoScaling.Model.SuspendedState
```

- Untuk detail API, lihat [DescribeScalableTargets](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-AASScalingActivity

Contoh kode berikut menunjukkan cara menggunakan `Get-AASScalingActivity`.

Alat untuk PowerShell V4

Contoh 1: Memberikan informasi deskriptif tentang aktivitas penskalaan di namespace yang ditentukan dari enam minggu sebelumnya.

```
Get-AASScalingActivity -ServiceNamespace AppStream
```

Output:

```
ActivityId        : 2827409f-b639-4cdb-a957-8055d5d07434
```

```

Cause           : monitor alarm Appstream2-MyFleet-default-scale-in Alarm in state
                  ALARM triggered policy default-scale-in
Description      : Setting desired capacity to 2.
Details          :
EndTime          : 12/14/2019 11:32:49 AM
ResourceId       : fleet/MyFleet
ScalableDimension : appstream:fleet:DesiredCapacity
ServiceNamespace : appstream
StartTime        : 12/14/2019 11:32:14 AM
StatusCode       : Successful
StatusMessage    : Successfully set desired capacity to 2. Change successfully
                  fulfilled by appstream.

```

- Untuk detail API, lihat [DescribeScalingActivities](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-AASScalingPolicy

Contoh kode berikut menunjukkan cara menggunakan `Get-AASScalingPolicy`.

Alat untuk PowerShell V4

Contoh 1: Cmdlet ini menjelaskan kebijakan penskalaan Application Auto Scaling untuk namespace layanan yang ditentukan.

```
Get-AASScalingPolicy -ServiceNamespace AppStream
```

Output:

```

Alarms           : {Appstream2-LabFleet-default-scale-out-
Alarm}
CreationTime      : 9/3/2019 2:48:15 AM
PolicyARN         : arn:aws:autoscaling:us-
west-2:012345678912:scalingPolicy:5659b069-b5cd-4af1-9f7f-3e956d36233e:resource/
appstream/fleet/LabFleet:
                  policyName/default-scale-out
PolicyName        : default-scale-out
PolicyType        : StepScaling
ResourceId        : fleet/LabFleet
ScalableDimension : appstream:fleet:DesiredCapacity
ServiceNamespace  : appstream

```

```

StepScalingPolicyConfiguration      :
  Amazon.ApplicationAutoScaling.Model.StepScalingPolicyConfiguration
TargetTrackingScalingPolicyConfiguration :

Alarms                             : {Appstream2-LabFleet-default-scale-in-
Alarm}
CreationTime                       : 9/3/2019 2:48:15 AM
PolicyARN                          : arn:aws:autoscaling:us-
west-2:012345678912:scalingPolicy:5659b069-b5cd-4af1-9f7f-3e956d36233e:resource/
appstream/fleet/LabFleet:
                                   policyName/default-scale-in
PolicyName                         : default-scale-in
PolicyType                         : StepScaling
ResourceId                         : fleet/LabFleet
ScalableDimension                  : appstream:fleet:DesiredCapacity
ServiceNamespace                   : appstream
StepScalingPolicyConfiguration      :
  Amazon.ApplicationAutoScaling.Model.StepScalingPolicyConfiguration
TargetTrackingScalingPolicyConfiguration :

```

- Untuk detail API, lihat [DescribeScalingPolicies](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-AASScheduledAction

Contoh kode berikut menunjukkan cara menggunakan `Get-AASScheduledAction`.

Alat untuk PowerShell V4

Contoh 1: Cmdlet ini mencantumkan tindakan yang dijadwalkan untuk grup Auto Scaling Anda yang belum berjalan atau yang belum mencapai waktu akhirnya.

```
Get-AASScheduledAction -ServiceNamespace AppStream
```

Output:

```

CreationTime      : 12/22/2019 9:25:52 AM
EndTime           : 1/1/0001 12:00:00 AM
ResourceId        : fleet/MyFleet
ScalableDimension : appstream:fleet:DesiredCapacity
ScalableTargetAction : Amazon.ApplicationAutoScaling.Model.ScalableTargetAction

```

```

Schedule           : cron(0 0 8 ? * MON-FRI *)
ScheduledActionARN : arn:aws:autoscaling:us-
west-2:012345678912:scheduledAction:4897ca24-3caa-4bf1-8484-851a089b243c:resource/
appstream/fleet/MyFleet:scheduledActionName
                    /WeekDaysFleetScaling
ScheduledActionName : WeekDaysFleetScaling
ServiceNamespace   : appstream
StartTime           : 1/1/0001 12:00:00 AM

```

- Untuk detail API, lihat [DescribeScheduledActions](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-AASScalableTarget

Contoh kode berikut menunjukkan cara menggunakan `Remove-AASScalableTarget`.

Alat untuk PowerShell V4

Contoh 1: Cmdlet ini membatalkan pendaftaran target yang dapat diskalakan Application Auto Scaling. Membatalkan pendaftaran target yang dapat diskalakan akan menghapus kebijakan penskalaan yang terkait dengannya.

```

Remove-AASScalableTarget -ResourceId fleet/MyFleet -ScalableDimension
appstream:fleet:DesiredCapacity -ServiceNamespace AppStream

```

Output:

```

Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-AASScalableTarget (DeregisterScalableTarget)" on
target "fleet/MyFleet".
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is
"Y"): Y

```

- Untuk detail API, lihat [DeregisterScalableTarget](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-AASScalingPolicy

Contoh kode berikut menunjukkan cara menggunakan `Remove-AASScalingPolicy`.

Alat untuk PowerShell V4

Contoh 1: Cmdlet ini menghapus kebijakan penskalaan yang ditentukan untuk target scalable Application Auto Scaling.

```
Remove-AASScalingPolicy -ServiceNamespace AppStream -PolicyName "default-scale-out"
-ResourceId fleet/Test -ScalableDimension appstream:fleet:DesiredCapacity
```

- Untuk detail API, lihat [DeleteScalingPolicy](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-AASScheduledAction

Contoh kode berikut menunjukkan cara menggunakan `Remove-AASScheduledAction`.

Alat untuk PowerShell V4

Contoh 1: Cmdlet ini menghapus tindakan terjadwal yang ditentukan untuk target scalable Application Auto Scaling.

```
Remove-AASScheduledAction -ServiceNamespace AppStream -ScheduledActionName
WeekDaysFleetScaling -ResourceId fleet/MyFleet -ScalableDimension
appstream:fleet:DesiredCapacity
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-AASScheduledAction (DeleteScheduledAction)" on
target "WeekDaysFleetScaling".
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is
"Y"): Y
```

- Untuk detail API, lihat [DeleteScheduledAction](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Set-AASScalingPolicy

Contoh kode berikut menunjukkan cara menggunakan `Set-AASScalingPolicy`.

Alat untuk PowerShell V4

Contoh 1: Cmdlet ini membuat atau memperbarui kebijakan untuk target scalable Application Auto Scaling. Setiap target yang dapat diskalakan diidentifikasi oleh namespace layanan, ID sumber daya, dan dimensi yang dapat diskalakan.

```
Set-AASScalingPolicy -ServiceNamespace AppStream -PolicyName ASFleetScaleInPolicy
-PolicyType StepScaling -ResourceId fleet/MyFleet -ScalableDimension
appstream:fleet:DesiredCapacity -StepScalingPolicyConfiguration_AdjustmentType
ChangeInCapacity -StepScalingPolicyConfiguration_Cooldown 360
-StepScalingPolicyConfiguration_MetricAggregationType Average -
StepScalingPolicyConfiguration_StepAdjustments @{ScalingAdjustment = -1;
MetricIntervalUpperBound = 0}
```

Output:

```
Alarms      PolicyARN
-----
{}          arn:aws:autoscaling:us-
west-2:012345678912:scalingPolicy:4897ca24-3caa-4bf1-8484-851a089b243c:resource/
appstream/fleet/MyFleet:policyName/ASFleetScaleInPolicy
```

- Untuk detail API, lihat [PutScalingPolicy](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Set-AASScheduledAction

Contoh kode berikut menunjukkan cara menggunakan Set-AASScheduledAction.

Alat untuk PowerShell V4

Contoh 1: Cmdlet ini membuat atau memperbarui tindakan terjadwal untuk target scalable Application Auto Scaling. Setiap target yang dapat diskalakan diidentifikasi oleh namespace layanan, ID sumber daya, dan dimensi yang dapat diskalakan.

```
Set-AASScheduledAction -ServiceNamespace AppStream -ResourceId fleet/
MyFleet -Schedule "cron(0 0 8 ? * MON-FRI *)" -ScalableDimension
appstream:fleet:DesiredCapacity -ScheduledActionName WeekDaysFleetScaling -
ScalableTargetAction_MinCapacity 5 -ScalableTargetAction_MaxCapacity 10
```

- Untuk detail API, lihat [PutScheduledAction](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

AppStream 2.0 contoh menggunakan Alat untuk PowerShell V4

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan AppStream 2.0.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

Add-APSResourceTag

Contoh kode berikut menunjukkan cara menggunakan Add-APSResourceTag.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menambahkan Tag sumber daya ke AppStream sumber daya

```
Add-APSResourceTag -ResourceArn arn:aws:appstream:us-east-1:123456789012:stack/SessionScriptTest -Tag @{StackState='Test'} -Select ^Tag
```

Output:

Name	Value
----	-----
StackState	Test

- Untuk detail API, lihat [TagResource](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Copy-APSImage

Contoh kode berikut menunjukkan cara menggunakan Copy-APSImage.

Alat untuk PowerShell V4

Contoh 1: Sampel ini menyalin gambar ke wilayah lain

```
Copy-APSIImage -DestinationImageName TestImageCopy -DestinationRegion us-west-2 -  
SourceImageName Powershell
```

Output:

```
TestImageCopy
```

- Untuk detail API, lihat [CopyImage](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Disable-APSUser

Contoh kode berikut menunjukkan cara menggunakan `Disable-APSUser`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menonaktifkan pengguna di USERPOOL

```
Disable-APSUser -AuthenticationType USERPOOL -UserName TestUser@lab.com
```

- Untuk detail API, lihat [DisableUser](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Enable-APSUser

Contoh kode berikut menunjukkan cara menggunakan `Enable-APSUser`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memungkinkan pengguna yang dinonaktifkan di USERPOOL

```
Enable-APSUser -AuthenticationType USERPOOL -UserName TestUser@lab.com
```

- Untuk detail API, lihat [EnableUser](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-APSAssociatedFleetList

Contoh kode berikut menunjukkan cara menggunakan `Get-APSAssociatedFleetList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menampilkan armada yang terkait dengan tumpukan

```
Get-APSAssociatedFleetList -StackName PowershellStack
```

Output:

```
PowershellFleet
```

- Untuk detail API, lihat [ListAssociatedFleets](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-APSAssociatedStackList

Contoh kode berikut menunjukkan cara menggunakan `Get-APSAssociatedStackList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menampilkan tumpukan yang terkait dengan armada

```
Get-APSAssociatedStackList -FleetName PowershellFleet
```

Output:

```
PowershellStack
```

- Untuk detail API, lihat [ListAssociatedStacks](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-APSDirectoryConfigList

Contoh kode berikut menunjukkan cara menggunakan `Get-APSDirectoryConfigList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menampilkan Konfigurasi Direktori yang dibuat di AppStream

```
Get-APSDirectoryConfigList | Select DirectoryName,  
OrganizationalUnitDistinguishedNames, CreatedTime
```

Output:

DirectoryName	OrganizationalUnitDistinguishedNames	CreatedTime
Test.com	{OU=AppStream,DC=Test,DC=com}	9/6/2019 10:56:40 AM
contoso.com	{OU=AppStream,OU=contoso,DC=contoso,DC=com}	8/9/2019 9:08:50 AM

- Untuk detail API, lihat [DescribeDirectoryConfigs](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-APSFleetList

Contoh kode berikut menunjukkan cara menggunakan `Get-APSFleetList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menampilkan detail armada

```
Get-APSFleetList -Name Test
```

Output:

```

Arn                : arn:aws:appstream:us-east-1:1234567890:fleet/Test
ComputeCapacityStatus : Amazon.AppStream.Model.ComputeCapacityStatus
CreatedTime         : 9/12/2019 5:00:45 PM
Description         : Test
DisconnectTimeoutInSeconds : 900
DisplayName         : Test
DomainJoinInfo      :
EnableDefaultInternetAccess : False
FleetErrors         : {}
FleetType           : ON_DEMAND
IamRoleArn          :
IdleDisconnectTimeoutInSeconds : 900
ImageArn            : arn:aws:appstream:us-east-1:1234567890:image/Test
ImageName           : Test
InstanceType        : stream.standard.medium
MaxUserDurationInSeconds : 57600
Name                : Test
State               : STOPPED
VpcConfig           : Amazon.AppStream.Model.VpcConfig

```

- Untuk detail API, lihat [DescribeFleets](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-APSIImageBuilderList

Contoh kode berikut menunjukkan cara menggunakan `Get-APSIImageBuilderList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menampilkan rincian ImageBuilder

```
Get-APSIImageBuilderList -Name TestImage
```

Output:

```
AccessEndpoints           : {}
AppstreamAgentVersion     : 06-19-2019
Arn                       : arn:aws:appstream:us-east-1:1234567890:image-builder/
TestImage
CreatedTime               : 1/14/2019 4:33:05 AM
Description               :
DisplayName               : TestImage
DomainJoinInfo            :
EnableDefaultInternetAccess : False
IamRoleArn                :
ImageArn                  : arn:aws:appstream:us-east-1::image/Base-Image-
Builder-05-02-2018
ImageBuilderErrors        : {}
InstanceType              : stream.standard.large
Name                      : TestImage
NetworkAccessConfiguration : Amazon.AppStream.Model.NetworkAccessConfiguration
Platform                  : WINDOWS
State                     : STOPPED
StateChangeReason          :
VpcConfig                  : Amazon.AppStream.Model.VpcConfig
```

- Untuk detail API, lihat [DescribeImageBuilders](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-APSIImageList

Contoh kode berikut menunjukkan cara menggunakan `Get-APSIImageList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menampilkan AppStream Gambar pribadi

```
Get-APSIImageList -Type PRIVATE | select DisplayName, ImageBuilderName, Visibility,
arn
```

Output:

DisplayName	ImageBuilderName	Visibility	Arn
-----	-----	-----	---
OfficeApps	OfficeApps	PRIVATE	arn:aws:appstream:us-
east-1:123456789012:image/OfficeApps			
SessionScriptV2	SessionScriptTest	PRIVATE	arn:aws:appstream:us-
east-1:123456789012:image/SessionScriptV2			

- Untuk detail API, lihat [DescribeImages](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-APSIImagePermission

Contoh kode berikut menunjukkan cara menggunakan `Get-APSIImagePermission`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menampilkan izin Gambar pada Gambar bersama AppStream

```
Get-APSIImagePermission -Name Powershell | select SharedAccountId,
@{n="AllowFleet";e={$_.ImagePermissions.AllowFleet}},
@{n="AllowImageBuilder";e={$_.ImagePermissions.AllowImageBuilder}}
```

Output:

SharedAccountId	AllowFleet	AllowImageBuilder
-----	-----	-----
123456789012	True	True

- Untuk detail API, lihat [DescribeImagePermissions](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-APSSessionList

Contoh kode berikut menunjukkan cara menggunakan `Get-APSSessionList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menampilkan daftar sesi ke armada

```
Get-APSSessionList -FleetName PowershellFleet -StackName PowershellStack
```

Output:

```
AuthenticationType      : API
ConnectionState         : CONNECTED
FleetName                : PowershellFleet
Id                       : d8987c70-4394-4324-a396-2d485c26f2a2
MaxExpirationTime       : 12/27/2019 4:54:07 AM
NetworkAccessConfiguration : Amazon.AppStream.Model.NetworkAccessConfiguration
StackName               : PowershellStack
StartTime               : 12/26/2019 12:54:12 PM
State                   : ACTIVE
UserId                  : Test
```

- Untuk detail API, lihat [DescribeSessions](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-APSStackList

Contoh kode berikut menunjukkan cara menggunakan `Get-APSStackList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menampilkan daftar AppStream Stack

```
Get-APSStackList | Select DisplayName, Arn, CreatedTime
```

Output:

```
DisplayName      Arn
-----
CreatedTime
-----
```

PowershellStack	arn:aws:appstream:us-east-1:123456789012:stack/
PowershellStack	4/24/2019 8:49:29 AM
SessionScriptTest	arn:aws:appstream:us-east-1:123456789012:stack/
SessionScriptTest	9/12/2019 3:23:12 PM

- Untuk detail API, lihat [DescribeStacks](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-APSTagsForResourceList

Contoh kode berikut menunjukkan cara menggunakan `Get-APSTagsForResourceList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menampilkan tag pada sumber AppStream daya

```
Get-APSTagsForResourceList -ResourceArn arn:aws:appstream:us-east-1:123456789012:stack/SessionScriptTest
```

Output:

Key	Value
---	-----
StackState	Test

- Untuk detail API, lihat [ListTagsForResource](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-APSUsageReportSubscription

Contoh kode berikut menunjukkan cara menggunakan `Get-APSUsageReportSubscription`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menampilkan detail AppStreamUsageReport konfigurasi

```
Get-APSUsageReportSubscription
```

Output:

LastGeneratedReportDate	S3BucketName	Schedule
SubscriptionErrors		

```
-----
1/1/0001 12:00:00 AM    appstream-logs-us-east-1-123456789012-sik1hnxe DAILY    {}
```

- Untuk detail API, lihat [DescribeUsageReportSubscriptions](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-APSUser

Contoh kode berikut menunjukkan cara menggunakan `Get-APSUser`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menampilkan daftar pengguna dengan status diaktifkan

```
Get-APSUser -AuthenticationType USERPOOL | Select-Object UserName,
AuthenticationType, Enabled
```

Output:

UserName	AuthenticationType	Enabled
-----	-----	-----
foo1@contoso.com	USERPOOL	True
foo2@contoso.com	USERPOOL	True
foo3@contoso.com	USERPOOL	True
foo4@contoso.com	USERPOOL	True
foo5@contoso.com	USERPOOL	True

- Untuk detail API, lihat [DescribeUsers](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-APSUserStackAssociation

Contoh kode berikut menunjukkan cara menggunakan `Get-APSUserStackAssociation`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menampilkan daftar pengguna yang ditugaskan ke tumpukan

```
Get-APSUserStackAssociation -StackName PowershellStack
```

Output:

AuthenticationType	SendEmailNotification	StackName	UserName
-----	-----	-----	-----
USERPOOL	False	PowershellStack	TestUser1@lab.com
USERPOOL	False	PowershellStack	TestUser2@lab.com

- Untuk detail API, lihat [DescribeUserStackAssociations](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-APSDirectoryConfig

Contoh kode berikut menunjukkan cara menggunakan `New-APSDirectoryConfig`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat konfigurasi direktori di AppStream

```
New-APSDirectoryConfig -ServiceAccountCredentials_AccountName contoso\ServiceAccount
-ServiceAccountCredentials_AccountPassword MyPass -DirectoryName contoso.com -
OrganizationalUnitDistinguishedName "OU=AppStream,OU=Contoso,DC=Contoso,DC=com"
```

Output:

CreatedTime	DirectoryName	OrganizationalUnitDistinguishedNames
-----	-----	-----
12/27/2019 11:00:30 AM	contoso.com	{OU=AppStream,OU=Contoso,DC=Contoso,DC=com}

Amazon.AppStream.Model.ServiceAccountCredentials

- Untuk detail API, lihat [CreateDirectoryConfig](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-APSFleet

Contoh kode berikut menunjukkan cara menggunakan `New-APSFleet`.

Alat untuk PowerShell V4

Contoh 1: Sampel ini membuat AppStream armada baru

```
New-APSFleet -ComputeCapacity_DesiredInstance 1 -InstanceType stream.standard.medium
-Name TestFleet -DisplayName TestFleet -FleetType ON_DEMAND -
EnableDefaultInternetAccess $True -VpcConfig_SubnetIds "subnet-123ce32","subnet-
a1234cfd" -VpcConfig_SecurityGroupIds sg-4d012a34 -ImageName SessionScriptTest -
Region us-west-2
```

Output:

```
Arn : arn:aws:appstream:us-west-2:123456789012:fleet/
TestFleet
ComputeCapacityStatus : Amazon.AppStream.Model.ComputeCapacityStatus
CreatedTime : 12/27/2019 11:24:42 AM
Description :
DisconnectTimeoutInSeconds : 900
DisplayName : TestFleet
DomainJoinInfo :
EnableDefaultInternetAccess : True
FleetErrors : {}
FleetType : ON_DEMAND
IamRoleArn :
IdleDisconnectTimeoutInSeconds : 0
ImageArn : arn:aws:appstream:us-west-2:123456789012:image/
SessionScriptTest
ImageName : SessionScriptTest
InstanceType : stream.standard.medium
MaxUserDurationInSeconds : 57600
Name : TestFleet
State : STOPPED
VpcConfig : Amazon.AppStream.Model.VpcConfig
```

- Untuk detail API, lihat [CreateFleet](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-APSIImageBuilder

Contoh kode berikut menunjukkan cara menggunakan New-APSIImageBuilder.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat Image Builder di AppStream

```
New-APSIImageBuilder -InstanceType stream.standard.medium -Name TestIB -DisplayName
TestIB -ImageName AppStream-WinServer2012R2-12-12-2019 -EnableDefaultInternetAccess
```

```
$True -VpcConfig_SubnetId subnet-a1234cfd -VpcConfig_SecurityGroupIds sg-2d012a34 -
Region us-west-2
```

Output:

```
AccessEndpoints           : {}
AppstreamAgentVersion     : 12-16-2019
Arn                       : arn:aws:appstream:us-west-2:123456789012:image-
builder/TestIB
CreatedTime               : 12/27/2019 11:39:24 AM
Description               :
DisplayName               : TestIB
DomainJoinInfo           :
EnableDefaultInternetAccess : True
IamRoleArn                :
ImageArn                  : arn:aws:appstream:us-west-2::image/AppStream-
WinServer2012R2-12-12-2019
ImageBuilderErrors        : {}
InstanceType              : stream.standard.medium
Name                      : TestIB
NetworkAccessConfiguration :
Platform                  : WINDOWS
State                     : PENDING
StateChangeReason         :
VpcConfig                  : Amazon.AppStream.Model.VpcConfig
```

- Untuk detail API, lihat [CreateImageBuilder](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-APSIImageBuilderStreamingURL

Contoh kode berikut menunjukkan cara menggunakan `New-APSIImageBuilderStreamingURL`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat URL ImageBuilder streaming dengan validitas 2 jam

```
New-APSIImageBuilderStreamingURL -Name TestIB -Validity 7200 -Region us-west-2
```

Output:

Expires	StreamingURL
---------	--------------

```
-----
12/27/2019 1:49:13 PM https://appstream2.us-west-2.aws.amazon.com/authenticate?
parameters=eyJ0eXB1IjoiQURNSU4iLCJleHBpcmVzIjoimTU3NzQ1NDU1MyIsImF3c0FjY291bnRjZCI6IjM5MzQwM
```

- Untuk detail API, lihat [CreateImageBuilderStreamingURL](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-APSSStack

Contoh kode berikut menunjukkan cara menggunakan New-APSSStack.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat AppStream Stack baru

```
New-APSSStack -Name TestStack -DisplayName TestStack -ApplicationSettings_Enabled
$True -ApplicationSettings_SettingsGroup TestStack -Region us-west-2
```

Output:

```
AccessEndpoints      : {}
ApplicationSettings  : Amazon.AppStream.Model.ApplicationSettingsResponse
Arn                  : arn:aws:appstream:us-west-2:123456789012:stack/TestStack
CreatedTime          : 12/27/2019 12:34:19 PM
Description           :
DisplayName           : TestStack
EmbedHostDomains     : {}
FeedbackURL          :
Name                  : TestStack
RedirectURL           :
StackErrors           : {}
StorageConnectors    : {}
UserSettings         : {Amazon.AppStream.Model.UserSetting,
  Amazon.AppStream.Model.UserSetting, Amazon.AppStream.Model.UserSetting,
  Amazon.AppStream.Model.UserSetting}
```

- Untuk detail API, lihat [CreateStack](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-APSSStreamingURL

Contoh kode berikut menunjukkan cara menggunakan New-APSSStreamingURL.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat URL streaming Stack

```
New-APSSstreamingURL -StackName SessionScriptTest -FleetName SessionScriptNew -UserId
TestUser
```

Output:

```
Expires                StreamingURL
-----                -
12/27/2019 12:43:37 PM https://appstream2.us-east-1.aws.amazon.com/authenticate?
parameters=eyJ0eXB1IjoiRU5EX1VTRVIiLCJleHBpcmVzIjoiMTU3NzQ1MDYxNyIsImF3c0FjY291bnRJZCI6IjM5M...
```

- Untuk detail API, lihat [CreateStreamingURL](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-APSUsageReportSubscription

Contoh kode berikut menunjukkan cara menggunakan New-APSUsageReportSubscription.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengaktifkan Laporan AppStream Penggunaan

```
New-APSUsageReportSubscription
```

Output:

```
S3BucketName                Schedule
-----                -
appstream-logs-us-east-1-123456789012-sik2hnxe DAILY
```

- Untuk detail API, lihat [CreateUsageReportSubscription](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-APSUser

Contoh kode berikut menunjukkan cara menggunakan New-APSUser.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat pengguna di USERPOOL

```
New-APSUser -UserName Test@lab.com -AuthenticationType USERP00L -FirstName 'kt' -
LastName 'aws' -Select ^UserName
```

Output:

```
Test@lab.com
```

- Untuk detail API, lihat [CreateUser](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Register-APSFleet

Contoh kode berikut menunjukkan cara menggunakan `Register-APSFleet`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendaftarkan armada dengan tumpukan

```
Register-APSFleet -StackName TestStack -FleetName TestFleet -Region us-west-2
```

- Untuk detail API, lihat [AssociateFleet](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Register-APSUserStackBatch

Contoh kode berikut menunjukkan cara menggunakan `Register-APSUserStackBatch`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menetapkan tumpukan ke pengguna di USERPOOL

```
Register-APSUserStackBatch -UserStackAssociation
@{AuthenticationType="USERP00L";SendEmailNotification=
$False;StackName="PowershellStack";UserName="TestUser1@lab.com"}
```

- Untuk detail API, lihat [BatchAssociateUserStack](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-APSDirectoryConfig

Contoh kode berikut menunjukkan cara menggunakan `Remove-APSDirectoryConfig`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus konfigurasi AppStream Direktori

```
Remove-APSDirectoryConfig -DirectoryName contoso.com
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-APSDirectoryConfig (DeleteDirectoryConfig)" on
target "contoso.com".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"): A
```

- Untuk detail API, lihat [DeleteDirectoryConfig](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-APSFleet

Contoh kode berikut menunjukkan cara menggunakan `Remove-APSFleet`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus armada AppStream

```
Remove-APSFleet -Name TestFleet -Region us-west-2
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-APSFleet (DeleteFleet)" on target "TestFleet".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"): A
```

- Untuk detail API, lihat [DeleteFleet](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-APSIImage

Contoh kode berikut menunjukkan cara menggunakan `Remove-APSIImage`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus Gambar

```
Remove-APSIImage -Name TestImage -Region us-west-2
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-APSIImage (DeleteImage)" on target "TestImage".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"): A

Applications                : {}
AppstreamAgentVersion       : LATEST
Arn                          : arn:aws:appstream:us-west-2:123456789012:image/
TestImage
BaseImageArn                :
CreatedTime                  : 12/27/2019 1:34:10 PM
Description                  :
DisplayName                  : TestImage
ImageBuilderName             :
ImageBuilderSupported        : True
ImagePermissions             :
Name                         : TestImage
Platform                     : WINDOWS
PublicBaseImageReleasedDate : 6/12/2018 12:00:00 AM
State                        : AVAILABLE
StateChangeReason            :
Visibility                    : PRIVATE
```

- Untuk detail API, lihat [DeleteImage](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-APSIImageBuilder

Contoh kode berikut menunjukkan cara menggunakan `Remove-APSIImageBuilder`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus ImageBuilder

```
Remove-APSIImageBuilder -Name TestIB -Region us-west-2
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-APSIImageBuilder (DeleteImageBuilder)" on target
"TestIB".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"): A

AccessEndpoints           : {}
AppstreamAgentVersion     : 12-16-2019
Arn                       : arn:aws:appstream:us-west-2:123456789012:image-
builder/TestIB
CreatedTime               : 12/27/2019 11:39:24 AM
Description               :
DisplayName               : TestIB
DomainJoinInfo            :
EnableDefaultInternetAccess : True
IamRoleArn                :
ImageArn                  : arn:aws:appstream:us-west-2::image/AppStream-
WinServer2012R2-12-12-2019
ImageBuilderErrors        : {}
InstanceType              : stream.standard.medium
Name                      : TestIB
NetworkAccessConfiguration : Amazon.AppStream.Model.NetworkAccessConfiguration
Platform                  : WINDOWS
State                     : DELETING
StateChangeReason          :
VpcConfig                  : Amazon.AppStream.Model.VpcConfig
```

- Untuk detail API, lihat [DeleteImageBuilder](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-APSIImagePermission

Contoh kode berikut menunjukkan cara menggunakan `Remove-APSIImagePermission`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus izin Gambar

```
Remove-APSIImagePermission -Name Powershell -SharedAccountId 123456789012
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-APSIImagePermission (DeleteImagePermissions)" on
target "Powershell".
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is
"Y"): A
```

- Untuk detail API, lihat [DeleteImagePermissions](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-APSResourceTag

Contoh kode berikut menunjukkan cara menggunakan Remove-APSResourceTag.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus tag sumber daya dari AppStream sumber daya

```
Remove-APSResourceTag -ResourceArn arn:aws:appstream:us-east-1:123456789012:stack/
SessionScriptTest -TagKey StackState
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-APSResourceTag (UntagResource)" on target
"arn:aws:appstream:us-east-1:123456789012:stack/SessionScriptTest".
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is
"Y"): A
```

- Untuk detail API, lihat [UntagResource](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-APSSStack

Contoh kode berikut menunjukkan cara menggunakan `Remove-APSSStack`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus Stack

```
Remove-APSSStack -Name TestStack -Region us-west-2
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-APSSStack (DeleteStack)" on target "TestStack".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"): A
```

- Untuk detail API, lihat [DeleteStack](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-APSUsageReportSubscription

Contoh kode berikut menunjukkan cara menggunakan `Remove-APSUsageReportSubscription`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menonaktifkan langganan Laporan AppStream Penggunaan

```
Remove-APSUsageReportSubscription
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-APSUsageReportSubscription
(DeleteUsageReportSubscription)" on target "".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"): A
```

- Untuk detail API, lihat [DeleteUsageReportSubscription](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-APSUser

Contoh kode berikut menunjukkan cara menggunakan `Remove-APSUser`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus pengguna dari USERPOOL

```
Remove-APSUser -UserName TestUser@lab.com -AuthenticationType USERPOOL
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-APSUser (DeleteUser)" on target "TestUser@lab.com".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is "Y"): A
```

- Untuk detail API, lihat [DeleteUser](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Revoke-APSSession

Contoh kode berikut menunjukkan cara menggunakan `Revoke-APSSession`.

Alat untuk PowerShell V4

Contoh 1: Sampel ini mencabut sesi ke AppStream armada

```
Revoke-APSSession -SessionId 6cd2f9a3-f948-4aa1-8014-8a7dcde14877
```

- Untuk detail API, lihat [ExpireSession](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Start-APSFleet

Contoh kode berikut menunjukkan cara menggunakan `Start-APSFleet`.

Alat untuk PowerShell V4

Contoh 1: Sampel ini memulai armada

```
Start-APSFleet -Name PowershellFleet
```

- Untuk detail API, lihat [StartFleet](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Start-APSIImageBuilder

Contoh kode berikut menunjukkan cara menggunakan `Start-APSIImageBuilder`.

Alat untuk PowerShell V4

Contoh 1: Sampel ini memulai ImageBuilder

```
Start-APSIImageBuilder -Name TestImage
```

Output:

```
AccessEndpoints           : {}
AppstreamAgentVersion     : 06-19-2019
Arn                       : arn:aws:appstream:us-east-1:123456789012:image-
builder/TestImage
CreatedTime               : 1/14/2019 4:33:05 AM
Description               :
DisplayName               : TestImage
DomainJoinInfo            :
EnableDefaultInternetAccess : False
IamRoleArn                :
ImageArn                  : arn:aws:appstream:us-east-1::image/Base-Image-
Builder-05-02-2018
ImageBuilderErrors        : {}
InstanceType              : stream.standard.large
Name                      : TestImage
NetworkAccessConfiguration : Amazon.AppStream.Model.NetworkAccessConfiguration
Platform                  : WINDOWS
State                     : PENDING
StateChangeReason          :
VpcConfig                  : Amazon.AppStream.Model.VpcConfig
```

- Untuk detail API, lihat [StartImageBuilder](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Stop-APSFleet

Contoh kode berikut menunjukkan cara menggunakan `Stop-APSFleet`.

Alat untuk PowerShell V4

Contoh 1: Sampel ini menghentikan armada

```
Stop-APSFleet -Name PowershellFleet
```

- Untuk detail API, lihat [StopFleet](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Stop-APSImageBuilder

Contoh kode berikut menunjukkan cara menggunakan Stop-APSImageBuilder.

Alat untuk PowerShell V4

Contoh 1: Sampel ini menghentikan ImageBuilder

```
Stop-APSImageBuilder -Name TestImage
```

Output:

```
AccessEndpoints           : {}
AppstreamAgentVersion     : 06-19-2019
Arn                       : arn:aws:appstream:us-east-1:123456789012:image-
builder/TestImage
CreatedTime               : 1/14/2019 4:33:05 AM
Description               :
DisplayName               : TestImage
DomainJoinInfo           :
EnableDefaultInternetAccess : False
IamRoleArn                :
ImageArn                 : arn:aws:appstream:us-east-1::image/Base-Image-
Builder-05-02-2018
ImageBuilderErrors        : {}
InstanceType              : stream.standard.large
Name                     : TestImage
NetworkAccessConfiguration : Amazon.AppStream.Model.NetworkAccessConfiguration
Platform                  : WINDOWS
State                     : STOPPING
StateChangeReason         :
VpcConfig                 : Amazon.AppStream.Model.VpcConfig
```

- Untuk detail API, lihat [StopImageBuilder](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Unregister-APSFleet

Contoh kode berikut menunjukkan cara menggunakan `Unregister-APSFleet`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membatalkan registrasi armada dari tumpukan

```
Unregister-APSFleet -StackName TestStack -FleetName TestFleet -Region us-west-2
```

- Untuk detail API, lihat [DisassociateFleet](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Unregister-APSUserStackBatch

Contoh kode berikut menunjukkan cara menggunakan `Unregister-APSUserStackBatch`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus pengguna dari Stack yang ditetapkan

```
Unregister-APSUserStackBatch -UserStackAssociation  
@{AuthenticationType="USERPOOL";SendEmailNotification=  
$False;StackName="PowershellStack";UserName="TestUser1@lab.com"}
```

- Untuk detail API, lihat [BatchDisassociateUserStack](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-APSDirectoryConfig

Contoh kode berikut menunjukkan cara menggunakan `Update-APSDirectoryConfig`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memperbarui konfigurasi Direktori yang dibuat AppStream

```
Update-APSDirectoryConfig -ServiceAccountCredentials_AccountName contoso  
\ServiceAccount -ServiceAccountCredentials_AccountPassword MyPass@1$@#  
-DirectoryName contoso.com -OrganizationalUnitDistinguishedName  
"OU=AppStreamNew,OU=Contoso,DC=Contoso,DC=com"
```

Output:

```

CreatedTime      DirectoryName  OrganizationalUnitDistinguishedNames
ServiceAccountCredentials
-----
-----
12/27/2019 3:50:02 PM contoso.com    {OU=AppStreamNew,OU=Contoso,DC=Contoso,DC=com}
Amazon.AppStream.Model.ServiceAccountCredentials

```

- Untuk detail API, lihat [UpdateDirectoryConfig](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-APSFleet

Contoh kode berikut menunjukkan cara menggunakan `Update-APSFleet`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memperbarui properti armada

```

Update-APSFleet -Name PowershellFleet -EnableDefaultInternetAccess $True -
DisconnectTimeoutInSeconds 950

```

Output:

```

Arn                : arn:aws:appstream:us-east-1:123456789012:fleet/
PowershellFleet
ComputeCapacityStatus : Amazon.AppStream.Model.ComputeCapacityStatus
CreatedTime         : 4/24/2019 8:39:41 AM
Description          : PowershellFleet
DisconnectTimeoutInSeconds : 950
DisplayName           : PowershellFleet
DomainJoinInfo       :
EnableDefaultInternetAccess : True
FleetErrors           : {}
FleetType             : ON_DEMAND
IamRoleArn            :
IdleDisconnectTimeoutInSeconds : 900
ImageArn              : arn:aws:appstream:us-east-1:123456789012:image/
Powershell
ImageName             : Powershell
InstanceType          : stream.standard.medium
MaxUserDurationInSeconds : 57600

```

```
Name           : PowershellFleet
State          : STOPPED
VpcConfig      : Amazon.AppStream.Model.VpcConfig
```

- Untuk detail API, lihat [UpdateFleet](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-APSIImagePermission

Contoh kode berikut menunjukkan cara menggunakan `Update-APSIImagePermission`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membagikan AppStream Gambar dengan akun lain

```
Update-APSIImagePermission -Name Powershell -SharedAccountId 123456789012 -
ImagePermissions_AllowFleet $True -ImagePermissions_AllowImageBuilder $True
```

- Untuk detail API, lihat [UpdateImagePermissions](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-APSSStack

Contoh kode berikut menunjukkan cara menggunakan `Update-APSSStack`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memperbarui (mengaktifkan) Persistensi pengaturan aplikasi dan Folder Home pada Stack

```
Update-APSSStack -Name PowershellStack -ApplicationSettings_Enabled $True
-ApplicationSettings_SettingsGroup PowershellStack -StorageConnector
@{ConnectorType="HOMEFOLDERS"}
```

Output:

```
AccessEndpoints      : {}
ApplicationSettings  : Amazon.AppStream.Model.ApplicationSettingsResponse
Arn                  : arn:aws:appstream:us-east-1:123456789012:stack/PowershellStack
CreatedTime          : 4/24/2019 8:49:29 AM
Description           : PowershellStack
DisplayName           : PowershellStack
```

```

EmbedHostDomains      : {}
FeedbackURL           :
Name                  : PowershellStack
RedirectURL           :
StackErrors           : {}
StorageConnectors     : {Amazon.AppStream.Model.StorageConnector,
                        Amazon.AppStream.Model.StorageConnector}
UserSettings          : {Amazon.AppStream.Model.UserSetting,
                        Amazon.AppStream.Model.UserSetting, Amazon.AppStream.Model.UserSetting,
                        Amazon.AppStream.Model.UserSetting}

```

- Untuk detail API, lihat [UpdateStack](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Contoh Aurora menggunakan Alat untuk V4 PowerShell

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan Aurora.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

Get-RDSOrderableDBInstanceOption

Contoh kode berikut menunjukkan cara menggunakan `Get-RDSOrderableDBInstanceOption`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan versi mesin DB yang mendukung kelas instans DB tertentu dalam file Wilayah AWS.

```
$params = @{
```

```
Engine = 'aurora-postgresql'
DBInstanceClass = 'db.r5.large'
Region = 'us-east-1'
}
Get-RDSOrderableDBInstanceOption @params
```

Contoh 2: Contoh ini mencantumkan kelas instans DB yang didukung untuk versi mesin DB tertentu dalam file Wilayah AWS.

```
$params = @{
    Engine = 'aurora-postgresql'
    EngineVersion = '13.6'
    Region = 'us-east-1'
}
Get-RDSOrderableDBInstanceOption @params
```

- Untuk detail API, lihat [DescribeOrderableDBInstanceOps](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Contoh Auto Scaling menggunakan Alat untuk V4 PowerShell

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan Auto Scaling.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

Add-ASLoadBalancer

Contoh kode berikut menunjukkan cara menggunakan Add-ASLoadBalancer.

Alat untuk PowerShell V4

Contoh 1: Contoh ini melampirkan penyeimbang beban yang ditentukan ke grup Auto Scaling yang ditentukan.

```
Add-ASLoadBalancer -LoadBalancerName my-lb -AutoScalingGroupName my-asg
```

- Untuk detail API, lihat [AttachLoadBalancers](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Complete-ASLifecycleAction

Contoh kode berikut menunjukkan cara menggunakan `Complete-ASLifecycleAction`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini melengkapi tindakan siklus hidup yang ditentukan.

```
Complete-ASLifecycleAction -LifecycleHookName myLifecycleHook -  
AutoScalingGroupName my-asg -LifecycleActionResult CONTINUE -LifecycleActionToken  
bcd2f1b8-9a78-44d3-8a7a-4dd07d7cf635
```

- Untuk detail API, lihat [CompleteLifecycleAction](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Disable-ASMetricsCollection

Contoh kode berikut menunjukkan cara menggunakan `Disable-ASMetricsCollection`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menonaktifkan pemantauan metrik yang ditentukan untuk grup Auto Scaling yang ditentukan.

```
Disable-ASMetricsCollection -AutoScalingGroupName my-asg -Metric @("GroupMinSize",  
"GroupMaxSize")
```

Contoh 2: Contoh ini menonaktifkan pemantauan semua metrik untuk grup Auto Scaling yang ditentukan.

```
Disable-ASMetricsCollection -AutoScalingGroupName my-asg
```

- Untuk detail API, lihat [DisableMetricsCollection](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Dismount-ASInstance

Contoh kode berikut menunjukkan cara menggunakan `Dismount-ASInstance`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini melepaskan instance yang ditentukan dari grup Auto Scaling yang ditentukan dan mengurangi kapasitas yang diinginkan sehingga Auto Scaling tidak meluncurkan instance pengganti.

```
Dismount-ASInstance -InstanceId i-93633f9b -AutoScalingGroupName my-asg -
ShouldDecrementDesiredCapacity $true
```

Output:

```
ActivityId           : 06733445-ce94-4039-be1b-b9f1866e276e
AutoScalingGroupName : my-asg
Cause                : At 2015-11-20T22:34:59Z instance i-93633f9b was detached in
                      response to a user request, shrinking
                      the capacity from 2 to 1.
Description          : Detaching EC2 instance: i-93633f9b
Details              : {"Availability Zone":"us-west-2b","Subnet
                      ID":"subnet-5264e837"}
EndTime              :
Progress              : 50
StartTime             : 11/20/2015 2:34:59 PM
StatusCode            : InProgress
StatusMessage        :
```

Contoh 2: Contoh ini melepaskan instance yang ditentukan dari grup Auto Scaling yang ditentukan tanpa mengurangi kapasitas yang diinginkan. Auto Scaling meluncurkan instance pengganti.

```
Dismount-ASInstance -InstanceId i-7bf746a2 -AutoScalingGroupName my-asg -
ShouldDecrementDesiredCapacity $false
```

Output:

```

ActivityId           : f43a3cd4-d38c-4af7-9fe0-d76ec2307b6d
AutoScalingGroupName : my-asg
Cause                : At 2015-11-20T22:34:59Z instance i-7bf746a2 was detached in
                      response to a user request.
Description          : Detaching EC2 instance: i-7bf746a2
Details              : {"Availability Zone":"us-west-2b","Subnet
                      ID":"subnet-5264e837"}
EndTime              :
Progress             : 50
StartTime            : 11/20/2015 2:34:59 PM
StatusCode           : InProgress
StatusMessage        :

```

- Untuk detail API, lihat [DetachInstances](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Dismount-ASLoadBalancer

Contoh kode berikut menunjukkan cara menggunakan `Dismount-ASLoadBalancer`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini melepaskan penyeimbang beban yang ditentukan dari grup Auto Scaling yang ditentukan.

```
Dismount-ASLoadBalancer -LoadBalancerName my-lb -AutoScalingGroupName my-asg
```

- Untuk detail API, lihat [DetachLoadBalancers](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Enable-ASMetricsCollection

Contoh kode berikut menunjukkan cara menggunakan `Enable-ASMetricsCollection`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memungkinkan pemantauan metrik yang ditentukan untuk grup Auto Scaling yang ditentukan.

```
Enable-ASMetricsCollection -Metric @("GroupMinSize", "GroupMaxSize") -
AutoScalingGroupName my-asg -Granularity 1Minute
```

Contoh 2: Contoh ini memungkinkan pemantauan semua metrik untuk grup Auto Scaling yang ditentukan.

```
Enable-ASMetricsCollection -AutoScalingGroupName my-asg -Granularity 1Minute
```

- Untuk detail API, lihat [EnableMetricsCollection](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Enter-ASStandby

Contoh kode berikut menunjukkan cara menggunakan Enter-ASStandby.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menempatkan instance yang ditentukan ke mode siaga dan mengurangi kapasitas yang diinginkan sehingga Auto Scaling tidak meluncurkan instance pengganti.

```
Enter-ASStandby -InstanceId i-93633f9b -AutoScalingGroupName my-asg -
ShouldDecrementDesiredCapacity $true
```

Output:

```
ActivityId           : e36a5a54-ced6-4df8-bd19-708e2a59a649
AutoScalingGroupName : my-asg
Cause                : At 2015-11-22T15:48:06Z instance i-95b8484f was moved to
                      standby in response to a user request,
                      shrinking the capacity from 2 to 1.
Description          : Moving EC2 instance to Standby: i-95b8484f
Details              : {"Availability Zone":"us-west-2b","Subnet
                      ID":"subnet-5264e837"}
EndTime              :
Progress              : 50
StartTime             : 11/22/2015 7:48:06 AM
StatusCode            : InProgress
StatusMessage        :
```

Contoh 2: Contoh ini menempatkan instance yang ditentukan ke mode siaga tanpa mengurangi kapasitas yang diinginkan. Auto Scaling meluncurkan instance pengganti.

```
Enter-ASStandby -InstanceId i-93633f9b -AutoScalingGroupName my-asg -
ShouldDecrementDesiredCapacity $false
```

Output:

```
ActivityId           : e36a5a54-ced6-4df8-bd19-708e2a59a649
AutoScalingGroupName : my-asg
Cause                : At 2015-11-22T15:48:06Z instance i-95b8484f was moved to
                      standby in response to a user request.
Description          : Moving EC2 instance to Standby: i-95b8484f
Details              : {"Availability Zone":"us-west-2b","Subnet
                      ID":"subnet-5264e837"}
EndTime              :
Progress              : 50
StartTime             : 11/22/2015 7:48:06 AM
StatusCode            : InProgress
StatusMessage         :
```

- Untuk detail API, lihat [EnterStandby](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Exit-ASStandby

Contoh kode berikut menunjukkan cara menggunakan `Exit-ASStandby`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memindahkan instance yang ditentukan keluar dari mode siaga.

```
Exit-ASStandby -InstanceId i-93633f9b -AutoScalingGroupName my-asg
```

Output:

```
ActivityId           : 1833d3e8-e32f-454e-b731-0670ad4c6934
AutoScalingGroupName : my-asg
Cause                : At 2015-11-22T15:51:21Z instance i-95b8484f was moved out of
                      standby in response to a user
                      request, increasing the capacity from 1 to 2.
Description          : Moving EC2 instance out of Standby: i-95b8484f
```

```

Details           : {"Availability Zone":"us-west-2b","Subnet
ID":"subnet-5264e837"}
EndTime           :
Progress          : 30
StartTime         : 11/22/2015 7:51:21 AM
StatusCode        : PreInService
StatusMessage     :

```

- Untuk detail API, lihat [ExitStandby](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-ASAccountLimit

Contoh kode berikut menunjukkan cara menggunakan `Get-ASAccountLimit`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan batas sumber daya Auto Scaling untuk akun Anda AWS .

```
Get-ASAccountLimit
```

Output:

```

MaxNumberOfAutoScalingGroups    : 20
MaxNumberOfLaunchConfigurations : 100

```

- Untuk detail API, lihat [DescribeAccountLimits](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-ASAdjustmentType

Contoh kode berikut menunjukkan cara menggunakan `Get-ASAdjustmentType`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan jenis penyesuaian yang didukung oleh Auto Scaling.

```
Get-ASAdjustmentType
```

Output:

```
Type
```

```

----
ChangeInCapacity
ExactCapacity
PercentChangeInCapacity

```

- Untuk detail API, lihat [DescribeAdjustmentTypes](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-ASAutoScalingGroup

Contoh kode berikut menunjukkan cara menggunakan `Get-ASAutoScalingGroup`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan nama grup Auto Scaling Anda.

```
Get-ASAutoScalingGroup | format-table -property AutoScalingGroupName
```

Output:

```

AutoScalingGroupName
-----
my-asg-1
my-asg-2
my-asg-3
my-asg-4
my-asg-5
my-asg-6

```

Contoh 2: Contoh ini menjelaskan grup Auto Scaling yang ditentukan.

```
Get-ASAutoScalingGroup -AutoScalingGroupName my-asg-1
```

Output:

```

AutoScalingGroupARN      : arn:aws:autoscaling:us-
west-2:123456789012:autoScalingGroup:930d940e-891e-4781-a11a-7b0acd480
                           f03:autoScalingGroupName/my-asg-1
AutoScalingGroupName     : my-asg-1
AvailabilityZones        : {us-west-2b, us-west-2a}
CreatedTime              : 3/1/2015 9:05:31 AM

```

```

DefaultCooldown      : 300
DesiredCapacity      : 2
EnabledMetrics       : {}
HealthCheckGracePeriod : 300
HealthCheckType      : EC2
Instances            : {my-lc}
LaunchConfigurationName : my-lc
LoadBalancerNames    : {}
MaxSize              : 0
MinSize              : 0
PlacementGroup       :
Status               :
SuspendedProcesses   : {}
Tags                 : {}
TerminationPolicies  : {Default}
VPCZoneIdentifier     : subnet-e4f33493,subnet-5264e837

```

Contoh 3: Contoh ini menjelaskan dua grup Auto Scaling yang ditentukan.

```
Get-ASAutoScalingGroup -AutoScalingGroupName @"my-asg-1", "my-asg-2")
```

Contoh 4: Contoh ini menjelaskan instance Auto Scaling untuk grup Auto Scaling yang ditentukan.

```
(Get-ASAutoScalingGroup -AutoScalingGroupName my-asg-1).Instances
```

Contoh 5: Contoh ini menjelaskan semua grup Auto Scaling Anda.

```
Get-ASAutoScalingGroup
```

Contoh 6: Contoh ini menjelaskan LaunchTemplate untuk grup Auto Scaling yang ditentukan.

Contoh ini mengasumsikan bahwa “Opsi pembelian instans” diatur ke “Patuhi templat peluncuran”. Jika opsi ini diatur ke “Gabungkan opsi pembelian dan jenis instance”, LaunchTemplate dapat diakses menggunakan "MixedInstancesPolicy". LaunchTemplate “properti.

```
(Get-ASAutoScalingGroup -AutoScalingGroupName my-ag-1).LaunchTemplate
```

Output:

LaunchTemplateId	LaunchTemplateName	Version
-----	-----	-----

```
lt-06095fd619cb40371 test-launch-template $Default
```

- Untuk detail API, lihat [DescribeAutoScalingGroups](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-ASAutoScalingInstance

Contoh kode berikut menunjukkan cara menggunakan `Get-ASAutoScalingInstance`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan instance IDs Auto Scaling Anda.

```
Get-ASAutoScalingInstance | format-table -property InstanceId
```

Output:

```
InstanceId
-----
i-12345678
i-87654321
i-abcd1234
```

Contoh 2: Contoh ini menjelaskan contoh Auto Scaling yang ditentukan.

```
Get-ASAutoScalingInstance -InstanceId i-12345678
```

Output:

```
AutoScalingGroupName    : my-asg
AvailabilityZone          : us-west-2b
HealthStatus             : HEALTHY
InstanceId                : i-12345678
LaunchConfigurationName  : my-lc
LifecycleState            : InService
```

Contoh 3: Contoh ini menjelaskan dua instance Auto Scaling yang ditentukan.

```
Get-ASAutoScalingInstance -InstanceId @( "i-12345678", "i-87654321" )
```

Contoh 4: Contoh ini menjelaskan instance Auto Scaling untuk grup Auto Scaling yang ditentukan.

```
(Get-ASAutoScalingGroup -AutoScalingGroupName my-asg).Instances | Get-ASAutoScalingInstance
```

Contoh 5: Contoh ini menjelaskan semua instance Auto Scaling Anda.

```
Get-ASAutoScalingInstance
```

- Untuk detail API, lihat [DescribeAutoScalingInstances](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-ASAutoScalingNotificationType

Contoh kode berikut menunjukkan cara menggunakan `Get-ASAutoScalingNotificationType`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan jenis notifikasi yang didukung oleh Auto Scaling.

```
Get-ASAutoScalingNotificationType
```

Output:

```
autoscaling:EC2_INSTANCE_LAUNCH
autoscaling:EC2_INSTANCE_LAUNCH_ERROR
autoscaling:EC2_INSTANCE_TERMINATE
autoscaling:EC2_INSTANCE_TERMINATE_ERROR
autoscaling:TEST_NOTIFICATION
```

- Untuk detail API, lihat [DescribeAutoScalingNotificationTypes](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-ASLaunchConfiguration

Contoh kode berikut menunjukkan cara menggunakan `Get-ASLaunchConfiguration`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan nama konfigurasi peluncuran Anda.

```
Get-ASLaunchConfiguration | format-table -property LaunchConfigurationName
```

Output:

```
LaunchConfigurationName
-----
my-lc-1
my-lc-2
my-lc-3
my-lc-4
my-lc-5
```

Contoh 2: Contoh ini menjelaskan konfigurasi peluncuran yang ditentukan.

```
Get-ASLaunchConfiguration -LaunchConfigurationName my-lc-1
```

Output:

```
AssociatePublicIpAddress      : True
BlockDeviceMappings           : {/dev/xvda}
ClassicLinkVPCId              :
ClassicLinkVPCSecurityGroups  : {}
CreatedTime                   : 12/12/2014 3:22:08 PM
EbsOptimized                   : False
IamInstanceProfile            :
ImageId                       : ami-043a5034
InstanceMonitoring             : Amazon.AutoScaling.Model.InstanceMonitoring
InstanceType                  : t2.micro
KernelId                      :
KeyName                       :
LaunchConfigurationARN         : arn:aws:autoscaling:us-
                                west-2:123456789012:launchConfiguration:7e5f31e4-693b-4604-9322-
                                e6f68d7fafad:launchConfigurationName/my-lc-1
LaunchConfigurationName       : my-lc-1
PlacementTenancy              :
RamdiskId                     :
SecurityGroups                 : {sg-67ef0308}
SpotPrice                     :
UserData                      :
```

Contoh 3: Contoh ini menjelaskan dua konfigurasi peluncuran yang ditentukan.

```
Get-ASLaunchConfiguration -LaunchConfigurationName @("my-lc-1", "my-lc-2")
```

Contoh 4: Contoh ini menjelaskan semua konfigurasi peluncuran Anda.

```
Get-ASLaunchConfiguration
```

- Untuk detail API, lihat [DescribeLaunchConfigurations](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-ASLifecycleHook

Contoh kode berikut menunjukkan cara menggunakan `Get-ASLifecycleHook`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan hook siklus hidup yang ditentukan.

```
Get-ASLifecycleHook -AutoScalingGroupName my-asg -LifecycleHookName myLifecycleHook
```

Output:

```
AutoScalingGroupName : my-asg
DefaultResult         : ABANDON
GlobalTimeout         : 172800
HeartbeatTimeout      : 3600
LifecycleHookName     : myLifecycleHook
LifecycleTransition    : auto-scaling:EC2_INSTANCE_LAUNCHING
NotificationMetadata  :
NotificationTargetARN : arn:aws:sns:us-west-2:123456789012:my-topic
RoleARN               : arn:aws:iam::123456789012:role/my-iam-role
```

Contoh 2: Contoh ini menjelaskan semua kait siklus hidup untuk grup Auto Scaling yang ditentukan.

```
Get-ASLifecycleHook -AutoScalingGroupName my-asg
```

Contoh 3: Contoh ini menjelaskan semua kait siklus hidup untuk semua grup Auto Scaling Anda.

```
Get-ASLifecycleHook
```

- Untuk detail API, lihat [DescribeLifecycleHooks](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-ASLifecycleHookType

Contoh kode berikut menunjukkan cara menggunakan `Get-ASLifecycleHookType`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan jenis kait siklus hidup yang didukung oleh Auto Scaling.

```
Get-ASLifecycleHookType
```

Output:

```
autoscaling:EC2_INSTANCE_LAUNCHING
auto-scaling:EC2_INSTANCE_TERMINATING
```

- Untuk detail API, lihat [DescribeLifecycleHookTypes](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-ASLoadBalancer

Contoh kode berikut menunjukkan cara menggunakan `Get-ASLoadBalancer`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan penyeimbang beban untuk grup Auto Scaling yang ditentukan.

```
Get-ASLoadBalancer -AutoScalingGroupName my-asg
```

Output:

LoadBalancerName	State
-----	----
my-lb	Added

- Untuk detail API, lihat [DescribeLoadBalancers](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-ASMetricCollectionType

Contoh kode berikut menunjukkan cara menggunakan `Get-ASMetricCollectionType`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan jenis koleksi metrik yang didukung oleh Auto Scaling.

```
(Get-ASMetricCollectionType).Metrics
```

Output:

```
Metric
-----
GroupMinSize
GroupMaxSize
GroupDesiredCapacity
GroupInServiceInstances
GroupPendingInstances
GroupTerminatingInstances
GroupStandbyInstances
GroupTotalInstances
```

Contoh 2: Contoh ini mencantumkan granularitas yang sesuai.

```
(Get-ASMetricCollectionType).Granularities
```

Output:

```
Granularity
-----
1Minute
```

- Untuk detail API, lihat [DescribeMetricCollectionTypes](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-ASNotificationConfiguration

Contoh kode berikut menunjukkan cara menggunakan `Get-ASNotificationConfiguration`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan tindakan notifikasi yang terkait dengan grup Auto Scaling yang ditentukan.

```
Get-ASNotificationConfiguration -AutoScalingGroupName my-asg | format-list
```

Output:

```
AutoScalingGroupName : my-asg
NotificationType      : auto-scaling:EC2_INSTANCE_LAUNCH
TopicARN              : arn:aws:sns:us-west-2:123456789012:my-topic

AutoScalingGroupName : my-asg
NotificationType      : auto-scaling:EC2_INSTANCE_TERMINATE
TopicARN              : arn:aws:sns:us-west-2:123456789012:my-topic
```

Contoh 2: Contoh ini menjelaskan tindakan notifikasi yang terkait dengan semua grup Auto Scaling Anda.

```
Get-ASNotificationConfiguration
```

- Untuk detail API, lihat [DescribeNotificationConfigurations](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-ASPolicy

Contoh kode berikut menunjukkan cara menggunakan `Get-ASPolicy`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan semua kebijakan untuk grup Auto Scaling yang ditentukan.

```
Get-ASPolicy -AutoScalingGroupName my-asg
```

Output:

```
AdjustmentType      : ChangeInCapacity
Alarms              : {}
AutoScalingGroupName : my-asg
Cooldown            : 0
EstimatedInstanceWarmup : 0
MetricAggregationType :
MinAdjustmentMagnitude : 0
MinAdjustmentStep   : 0
```

```
PolicyARN      : arn:aws:auto-scaling:us-west-2:123456789012:scalingPolicy:aa3836ab-5462-42c7-adab-e1d769fc24ef
                : autoScalingGroupName/my-asg:policyName/myScaleInPolicy
PolicyName     : myScaleInPolicy
PolicyType     : SimpleScaling
ScalingAdjustment : -1
StepAdjustments : {}
```

Contoh 2: Contoh ini menjelaskan kebijakan yang ditentukan untuk grup Auto Scaling yang ditentukan.

```
Get-ASPolicy -AutoScalingGroupName my-asg -PolicyName @("myScaleOutPolicy",
"myScaleInPolicy")
```

Contoh 3: Contoh ini menjelaskan semua kebijakan untuk semua grup Auto Scaling Anda.

```
Get-ASPolicy
```

- Untuk detail API, lihat [DescribePolicies](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-ASScalingActivity

Contoh kode berikut menunjukkan cara menggunakan `Get-ASScalingActivity`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan aktivitas penskalaan selama enam minggu terakhir untuk grup Auto Scaling yang ditentukan.

```
Get-ASScalingActivity -AutoScalingGroupName my-asg
```

Output:

```
ActivityId      : 063308ae-aa22-4a9b-94f4-9fae4EXAMPLE
AutoScalingGroupName : my-asg
Cause          : At 2015-11-22T15:45:16Z a user request explicitly set group
                desired capacity changing the desired
                capacity from 1 to 2. At 2015-11-22T15:45:34Z an instance
                was started in response to a difference
                between desired and actual capacity, increasing the capacity
                from 1 to 2.
```

```

Description      : Launching a new EC2 instance: i-26e715fc
Details          : {"Availability Zone":"us-west-2b","Subnet
  ID":"subnet-5264e837"}
EndTime         : 11/22/2015 7:46:09 AM
Progress        : 100
StartTime       : 11/22/2015 7:45:35 AM
StatusCode      : Successful
StatusMessage   :

ActivityId       : ce719997-086d-4c73-a2f1-ab703EXAMPLE
AutoScalingGroupName : my-asg
Cause           : At 2015-11-20T22:57:53Z a user request created an
  AutoScalingGroup changing the desired capacity
                  from 0 to 1. At 2015-11-20T22:57:58Z an instance was
  started in response to a difference betwe
                  en desired and actual capacity, increasing the capacity from
  0 to 1.
Description     : Launching a new EC2 instance: i-93633f9b
Details         : {"Availability Zone":"us-west-2b","Subnet
  ID":"subnet-5264e837"}
EndTime        : 11/20/2015 2:58:32 PM
Progress       : 100
StartTime      : 11/20/2015 2:57:59 PM
StatusCode     : Successful
StatusMessage  :

```

Contoh 2: Contoh ini menjelaskan aktivitas penskalaan yang ditentukan.

```
Get-ASScalingActivity -ActivityId "063308ae-aa22-4a9b-94f4-9fae4EXAMPLE"
```

Contoh 3: Contoh ini menjelaskan aktivitas penskalaan selama enam minggu terakhir untuk semua grup Auto Scaling Anda.

```
Get-ASScalingActivity
```

- Untuk detail API, lihat [DescribeScalingActivities](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-ASScalingProcessType

Contoh kode berikut menunjukkan cara menggunakan `Get-ASScalingProcessType`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan jenis proses yang didukung oleh Auto Scaling.

```
Get-ASScalingProcessType
```

Output:

```
ProcessName
-----
AZRebalance
AddToLoadBalancer
AlarmNotification
HealthCheck
Launch
ReplaceUnhealthy
ScheduledActions
Terminate
```

- Untuk detail API, lihat [DescribeScalingProcessTypes](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-ASScheduledAction

Contoh kode berikut menunjukkan cara menggunakan `Get-ASScheduledAction`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan tindakan penskalaan terjadwal untuk grup Auto Scaling yang ditentukan.

```
Get-ASScheduledAction -AutoScalingGroupName my-asg
```

Output:

```
AutoScalingGroupName : my-asg
DesiredCapacity       : 10
EndTime               :
MaxSize               :
MinSize               :
Recurrence            :
```

```
ScheduledActionARN : arn:aws:autoscaling:us-
west-2:123456789012:scheduledUpdateGroupAction:8a4c5f24-6ec6-4306-a2dd-f7
2c3af3a4d6:autoScalingGroupName/my-asg:scheduledActionName/
myScheduledAction
ScheduledActionName : myScheduledAction
StartTime           : 11/30/2015 8:00:00 AM
Time                : 11/30/2015 8:00:00 AM
```

Contoh 2: Contoh ini menjelaskan tindakan penskalaan terjadwal yang ditentukan.

```
Get-ASScheduledAction -ScheduledActionName @"(\"myScheduledScaleOut\",
\"myScheduledScaleIn\")
```

Contoh 3: Contoh ini menjelaskan tindakan penskalaan terjadwal yang dimulai pada waktu yang ditentukan.

```
Get-ASScheduledAction -StartTime "2015-12-01T08:00:00Z"
```

Contoh 4: Contoh ini menjelaskan tindakan penskalaan terjadwal yang berakhir pada waktu yang ditentukan.

```
Get-ASScheduledAction -EndTime "2015-12-30T08:00:00Z"
```

Contoh 5: Contoh ini menjelaskan tindakan penskalaan terjadwal untuk semua grup Auto Scaling Anda.

```
Get-ASScheduledAction
```

- Untuk detail API, lihat [DescribeScheduledActions](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-ASTag

Contoh kode berikut menunjukkan cara menggunakan `Get-ASTag`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan tag dengan nilai kunci baik 'myTag' atau 'myTag2'. Nilai yang mungkin untuk nama filter adalah 'auto-scaling-group', 'kunci', 'nilai', dan 'propagate-at-launch'. Sintaks yang digunakan oleh contoh ini memerlukan PowerShell versi 3 atau yang lebih baru.

```
Get-ASTag -Filter @( @{ Name="key"; Values=@("myTag", "myTag2") } )
```

Output:

```
Key           : myTag2
PropagateAtLaunch : True
ResourceId     : my-asg
ResourceType    : auto-scaling-group
Value          : myTagValue2

Key           : myTag
PropagateAtLaunch : True
ResourceId     : my-asg
ResourceType    : auto-scaling-group
Value          : myTagValue
```

Contoh 2: Dengan PowerShell versi 2, Anda harus menggunakan New-Object untuk membuat filter untuk parameter Filter.

```
$keys = New-Object string[] 2
$keys[0] = "myTag"
$keys[1] = "myTag2"
$filter = New-Object Amazon.AutoScaling.Model.Filter
$filter.Name = "key"
$filter.Values = $keys
Get-ASTag -Filter @( $filter )
```

Contoh 3: Contoh ini menjelaskan semua tag untuk semua grup Auto Scaling Anda.

```
Get-ASTag
```

- Untuk detail API, lihat [DescribeTags](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-ASTerminationPolicyType

Contoh kode berikut menunjukkan cara menggunakan Get-ASTerminationPolicyType.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan kebijakan penghentian yang didukung oleh Auto Scaling.

```
Get-ASTerminationPolicyType
```

Output:

```
ClosestToNextInstanceHour  
Default  
NewestInstance  
OldestInstance  
OldestLaunchConfiguration
```

- Untuk detail API, lihat [DescribeTerminationPolicyTypes](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Mount-ASInstance

Contoh kode berikut menunjukkan cara menggunakan `Mount-ASInstance`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini melampirkan instance yang ditentukan ke grup Auto Scaling yang ditentukan. Auto Scaling secara otomatis meningkatkan kapasitas yang diinginkan dari grup Auto Scaling.

```
Mount-ASInstance -InstanceId i-93633f9b -AutoScalingGroupName my-asg
```

- Untuk detail API, lihat [AttachInstances](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-ASAutoScalingGroup

Contoh kode berikut menunjukkan cara menggunakan `New-ASAutoScalingGroup`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat grup Auto Scaling dengan nama dan atribut yang ditentukan. Kapasitas default yang diinginkan adalah ukuran minimum. Oleh karena itu, grup Auto Scaling ini meluncurkan dua instance, satu di masing-masing dari dua Availability Zone yang ditentukan.

```
New-ASAutoScalingGroup -AutoScalingGroupName my-asg -LaunchConfigurationName my-lc -  
MinSize 2 -MaxSize 6 -AvailabilityZone @("us-west-2a", "us-west-2b")
```

- Untuk detail API, lihat [CreateAutoScalingGroup](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-ASLaunchConfiguration

Contoh kode berikut menunjukkan cara menggunakan `New-ASLaunchConfiguration`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat konfigurasi peluncuran bernama 'my-lc'. EC2 Instans yang diluncurkan oleh grup Auto Scaling yang menggunakan konfigurasi peluncuran ini menggunakan tipe instans tertentu, AML, grup keamanan, dan peran IAM.

```
New-ASLaunchConfiguration -LaunchConfigurationName my-lc -InstanceType "m3.medium" -ImageId "ami-12345678" -SecurityGroup "sg-12345678" -IamInstanceProfile "myIamRole"
```

- Untuk detail API, lihat [CreateLaunchConfiguration](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-ASAutoScalingGroup

Contoh kode berikut menunjukkan cara menggunakan `Remove-ASAutoScalingGroup`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus grup Auto Scaling yang ditentukan jika tidak memiliki instance yang berjalan. Anda diminta untuk konfirmasi sebelum operasi berlangsung.

```
Remove-ASAutoScalingGroup -AutoScalingGroupName my-asg
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-ASAutoScalingGroup (DeleteAutoScalingGroup)" on Target "my-asg".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is "Y"):
```

Contoh 2: Jika Anda menentukan parameter Force, Anda tidak diminta untuk konfirmasi sebelum operasi berlangsung.

```
Remove-ASAutoScalingGroup -AutoScalingGroupName my-asg -Force
```

Contoh 3: Contoh ini menghapus grup Auto Scaling yang ditentukan dan mengakhiri semua instance yang sedang berjalan yang dikandungnya.

```
Remove-ASAutoScalingGroup -AutoScalingGroupName my-asg -ForceDelete $true -Force
```

- Untuk detail API, lihat [DeleteAutoScalingGroup](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-ASLaunchConfiguration

Contoh kode berikut menunjukkan cara menggunakan Remove-ASLaunchConfiguration.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus konfigurasi peluncuran yang ditentukan jika tidak dilampirkan ke grup Auto Scaling. Anda diminta untuk konfirmasi sebelum operasi berlangsung.

```
Remove-ASLaunchConfiguration -LaunchConfigurationName my-lc
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-ASLaunchConfiguration (DeleteLaunchConfiguration)" on
Target "my-lc".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"):
```

Contoh 2: Jika Anda menentukan parameter Force, Anda tidak diminta untuk konfirmasi sebelum operasi berlangsung.

```
Remove-ASLaunchConfiguration -LaunchConfigurationName my-lc -Force
```

- Untuk detail API, lihat [DeleteLaunchConfiguration](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-ASLifecycleHook

Contoh kode berikut menunjukkan cara menggunakan `Remove-ASLifecycleHook`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus hook siklus hidup yang ditentukan untuk grup Auto Scaling yang ditentukan. Anda diminta untuk konfirmasi sebelum operasi berlangsung.

```
Remove-ASLifecycleHook -AutoScalingGroupName my-asg -LifecycleHookName  
myLifecycleHook
```

Output:

```
Confirm  
Are you sure you want to perform this action?  
Performing operation "Remove-ASLifecycleHook (DeleteLifecycleHook)" on Target  
"myLifecycleHook".  
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is  
"Y"):
```

Contoh 2: Jika Anda menentukan parameter `Force`, Anda tidak diminta untuk konfirmasi sebelum operasi berlangsung.

```
Remove-ASLifecycleHook -AutoScalingGroupName my-asg -LifecycleHookName  
myLifecycleHook -Force
```

- Untuk detail API, lihat [DeleteLifecycleHook](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-ASNotificationConfiguration

Contoh kode berikut menunjukkan cara menggunakan `Remove-ASNotificationConfiguration`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus tindakan notifikasi yang ditentukan. Anda diminta untuk konfirmasi sebelum operasi berlangsung.

```
Remove-ASNotificationConfiguration -AutoScalingGroupName my-asg -TopicARN  
"arn:aws:sns:us-west-2:123456789012:my-topic"
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-ASNotificationConfiguration
(DeleteNotificationConfiguration)" on Target
"arn:aws:sns:us-west-2:123456789012:my-topic".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"):
```

Contoh 2: Jika Anda menentukan parameter Force, Anda tidak diminta untuk konfirmasi sebelum operasi berlangsung.

```
Remove-ASNotificationConfiguration -AutoScalingGroupName my-asg -TopicARN
"arn:aws:sns:us-west-2:123456789012:my-topic" -Force
```

- Untuk detail API, lihat [DeleteNotificationConfiguration](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-ASPolicy

Contoh kode berikut menunjukkan cara menggunakan Remove-ASPolicy.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus kebijakan yang ditentukan untuk grup Auto Scaling yang ditentukan. Anda diminta untuk konfirmasi sebelum operasi berlangsung.

```
Remove-ASPolicy -AutoScalingGroupName my-asg -PolicyName myScaleInPolicy
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-ASPolicy (DeletePolicy)" on Target "myScaleInPolicy".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"):
```

Contoh 2: Jika Anda menentukan parameter Force, Anda tidak diminta untuk konfirmasi sebelum operasi berlangsung.

```
Remove-ASPolicy -AutoScalingGroupName my-asg -PolicyName myScaleInPolicy -Force
```

- Untuk detail API, lihat [DeletePolicy](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-ASScheduledAction

Contoh kode berikut menunjukkan cara menggunakan `Remove-ASScheduledAction`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus tindakan terjadwal yang ditentukan untuk grup Auto Scaling yang ditentukan. Anda diminta untuk konfirmasi sebelum operasi berlangsung.

```
Remove-ASScheduledAction -AutoScalingGroupName my-asg -ScheduledAction
"myScheduledAction"
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-ASScheduledAction (DeleteScheduledAction)" on Target
"myScheduledAction".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"):
```

Contoh 2: Jika Anda menentukan parameter `Force`, Anda tidak diminta untuk konfirmasi sebelum operasi berlangsung.

```
Remove-ASScheduledAction -AutoScalingGroupName my-asg -ScheduledAction
"myScheduledAction" -Force
```

- Untuk detail API, lihat [DeleteScheduledAction](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-ASTag

Contoh kode berikut menunjukkan cara menggunakan `Remove-ASTag`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus tag yang ditentukan dari grup Auto Scaling yang ditentukan. Anda diminta untuk konfirmasi sebelum operasi berlangsung. Sintaks yang digunakan oleh contoh ini memerlukan PowerShell versi 3 atau yang lebih baru.

```
Remove-ASTag -Tag @( @{ResourceType="auto-scaling-group"; ResourceId="my-asg";
  Key="myTag" } )
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-ASTag (DeleteTags)" on target
  "Amazon.AutoScaling.Model.Tag".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
  "Y"):
```

Contoh 2: Jika Anda menentukan parameter Force, Anda tidak diminta untuk konfirmasi sebelum operasi berlangsung.

```
Remove-ASTag -Tag @( @{ResourceType="auto-scaling-group"; ResourceId="my-asg";
  Key="myTag" } ) -Force
```

Contoh 3: Dengan Powershell versi 2, Anda harus menggunakan New-Object untuk membuat tag untuk parameter Tag.

```
$tag = New-Object Amazon.AutoScaling.Model.Tag
$tag.ResourceType = "auto-scaling-group"
$tag.ResourceId = "my-asg"
$tag.Key = "myTag"
Remove-ASTag -Tag $tag -Force
```

- Untuk detail API, lihat [DeleteTags](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Resume-ASProcess

Contoh kode berikut menunjukkan cara menggunakan Resume-ASProcess.

Alat untuk PowerShell V4

Contoh 1: Contoh ini melanjutkan proses Auto Scaling yang ditentukan untuk grup Auto Scaling yang ditentukan.

```
Resume-ASProcess -AutoScalingGroupName my-asg -ScalingProcess "AlarmNotification"
```

Contoh 2: Contoh ini melanjutkan semua proses Auto Scaling yang ditangguhkan untuk grup Auto Scaling yang ditentukan.

```
Resume-ASProcess -AutoScalingGroupName my-asg
```

- Untuk detail API, lihat [ResumeProcesses](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Set-ASDesiredCapacity

Contoh kode berikut menunjukkan cara menggunakan `Set-ASDesiredCapacity`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menetapkan ukuran grup Auto Scaling yang ditentukan.

```
Set-ASDesiredCapacity -AutoScalingGroupName my-asg -DesiredCapacity 2
```

Contoh 2: Contoh ini menetapkan ukuran grup Auto Scaling yang ditentukan dan menunggu periode cooldown selesai sebelum penskalaan ke ukuran baru.

```
Set-ASDesiredCapacity -AutoScalingGroupName my-asg -DesiredCapacity 2 -HonorCooldown $true
```

- Untuk detail API, lihat [SetDesiredCapacity](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Set-ASInstanceHealth

Contoh kode berikut menunjukkan cara menggunakan `Set-ASInstanceHealth`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menetapkan status instance yang ditentukan menjadi 'Tidak Sehat', mengeluarkannya dari layanan. Auto Scaling mengakhiri dan menggantikan instance.

```
Set-ASInstanceHealth -HealthStatus Unhealthy -InstanceId i-93633f9b
```

Contoh 2: Contoh ini menetapkan status instance yang ditentukan ke 'Sehat', menjaganya tetap dalam layanan. Masa tenggang pemeriksaan kesehatan apa pun untuk grup Auto Scaling tidak dihormati.

```
Set-ASInstanceHealth -HealthStatus Healthy -InstanceId i-93633f9b -  
ShouldRespectGracePeriod $false
```

- Untuk detail API, lihat [SetInstanceHealth](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Set-ASInstanceProtection

Contoh kode berikut menunjukkan cara menggunakan `Set-ASInstanceProtection`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memungkinkan perlindungan instance untuk instance tertentu.

```
Set-ASInstanceProtection -AutoScalingGroupName my-asg -InstanceId i-12345678 -  
ProtectedFromScaleIn $true
```

Contoh 2: Contoh ini menonaktifkan perlindungan instance untuk instance tertentu.

```
Set-ASInstanceProtection -AutoScalingGroupName my-asg -InstanceId i-12345678 -  
ProtectedFromScaleIn $false
```

- Untuk detail API, lihat [SetInstanceProtection](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Set-ASTag

Contoh kode berikut menunjukkan cara menggunakan `Set-ASTag`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menambahkan satu tag ke grup Auto Scaling yang ditentukan. Kunci tag adalah 'MyTag' dan nilai tag adalah 'myTagValue'. Auto Scaling menyebarkan tag ini ke EC2 instance berikutnya yang diluncurkan oleh grup Auto Scaling. Sintaks yang digunakan oleh contoh ini memerlukan PowerShell versi 3 atau yang lebih baru.

```
Set-ASTag -Tag @( @{ResourceType="auto-scaling-group"; ResourceId="my-asg";
Key="myTag"; Value="myTagValue"; PropagateAtLaunch=$true} )
```

Contoh 2: Dengan PowerShell versi 2, Anda harus menggunakan New-Object untuk membuat tag untuk parameter Tag.

```
$tag = New-Object Amazon.AutoScaling.Model.Tag
$tag.ResourceType = "auto-scaling-group"
$tag.ResourceId = "my-asg"
$tag.Key = "myTag"
$tag.Value = "myTagValue"
$tag.PropagateAtLaunch = $true
Set-ASTag -Tag $tag
```

- Untuk detail API, lihat [CreateOrUpdateTags](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Start-ASPolicy

Contoh kode berikut menunjukkan cara menggunakan Start-ASPolicy.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengeksekusi kebijakan yang ditentukan untuk grup Auto Scaling yang ditentukan.

```
Start-ASPolicy -AutoScalingGroupName my-asg -PolicyName "myScaleInPolicy"
```

Contoh 2: Contoh ini mengeksekusi kebijakan yang ditentukan untuk grup Auto Scaling yang ditentukan, setelah menunggu periode cooldown selesai.

```
Start-ASPolicy -AutoScalingGroupName my-asg -PolicyName "myScaleInPolicy" -
HonorCooldown $true
```

- Untuk detail API, lihat [ExecutePolicy](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Stop-ASInstanceInAutoScalingGroup

Contoh kode berikut menunjukkan cara menggunakan Stop-ASInstanceInAutoScalingGroup.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengakhiri instance yang ditentukan dan mengurangi kapasitas yang diinginkan dari grup Auto Scaling sehingga Auto Scaling tidak meluncurkan instance pengganti.

```
Stop-ASInstanceInAutoScalingGroup -InstanceId i-93633f9b -
ShouldDecrementDesiredCapacity $true
```

Output:

```
ActivityId      : 2e40d9bd-1902-444c-abf3-6ea0002efdc5
AutoScalingGroupName :
Cause          : At 2015-11-22T16:09:03Z instance i-93633f9b was taken out of
                 service in response to a user
                 request, shrinking the capacity from 2 to 1.
Description    : Terminating EC2 instance: i-93633f9b
Details        : {"Availability Zone":"us-west-2b","Subnet
                 ID":"subnet-5264e837"}
EndTime       :
Progress      : 0
StartTime     : 11/22/2015 8:09:03 AM
StatusCode    : InProgress
StatusMessage :
```

Contoh 2: Contoh ini mengakhiri instance yang ditentukan tanpa mengurangi kapasitas yang diinginkan dari grup Auto Scaling. Auto Scaling meluncurkan instance pengganti.

```
Stop-ASInstanceInAutoScalingGroup -InstanceId i-93633f9b -
ShouldDecrementDesiredCapacity $false
```

Output:

```
ActivityId      : 2e40d9bd-1902-444c-abf3-6ea0002efdc5
AutoScalingGroupName :
Cause          : At 2015-11-22T16:09:03Z instance i-93633f9b was taken out of
                 service in response to a user
                 request.
Description    : Terminating EC2 instance: i-93633f9b
Details        : {"Availability Zone":"us-west-2b","Subnet
                 ID":"subnet-5264e837"}
EndTime       :
```

```
Progress           : 0
StartTime          : 11/22/2015 8:09:03 AM
StatusCode         : InProgress
StatusMessage      :
```

- Untuk detail API, lihat [TerminateInstanceInAutoScalingGroup](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Suspend-ASProcess

Contoh kode berikut menunjukkan cara menggunakan `Suspend-ASProcess`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menangguhkan proses Auto Scaling yang ditentukan untuk grup Auto Scaling yang ditentukan.

```
Suspend-ASProcess -AutoScalingGroupName my-asg -ScalingProcess "AlarmNotification"
```

Contoh 2: Contoh ini menangguhkan semua proses Auto Scaling untuk grup Auto Scaling yang ditentukan.

```
Suspend-ASProcess -AutoScalingGroupName my-asg
```

- Untuk detail API, lihat [SuspendProcesses](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-ASAutoScalingGroup

Contoh kode berikut menunjukkan cara menggunakan `Update-ASAutoScalingGroup`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memperbarui ukuran minimum dan maksimum grup Auto Scaling yang ditentukan.

```
Update-ASAutoScalingGroup -AutoScalingGroupName my-asg -MaxSize 5 -MinSize 1
```

Contoh 2: Contoh ini memperbarui periode cooldown default dari grup Auto Scaling yang ditentukan.

```
Update-ASAutoScalingGroup -AutoScalingGroupName my-asg -DefaultCooldown 10
```

Contoh 3: Contoh ini memperbarui Availability Zones dari grup Auto Scaling yang ditentukan.

```
Update-ASAutoScalingGroup -AutoScalingGroupName my-asg -AvailabilityZone @("us-west-2a", "us-west-2b")
```

Contoh 4: Contoh ini memperbarui grup Auto Scaling yang ditentukan untuk menggunakan pemeriksaan kesehatan Elastic Load Balancing.

```
Update-ASAutoScalingGroup -AutoScalingGroupName my-asg -HealthCheckType ELB -  
HealthCheckGracePeriod 60
```

- Untuk detail API, lihat [UpdateAutoScalingGroup](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Write-ASLifecycleActionHeartbeat

Contoh kode berikut menunjukkan cara menggunakan `Write-ASLifecycleActionHeartbeat`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencatat detak jantung untuk tindakan siklus hidup yang ditentukan. Ini membuat instance dalam status tertunda hingga Anda menyelesaikan tindakan kustom.

```
Write-ASLifecycleActionHeartbeat -AutoScalingGroupName my-asg -LifecycleHookName  
myLifecycleHook -LifecycleActionToken bcd2f1b8-9a78-44d3-8a7a-4dd07d7cf635
```

- Untuk detail API, lihat [RecordLifecycleActionHeartbeat](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Write-ASLifecycleHook

Contoh kode berikut menunjukkan cara menggunakan `Write-ASLifecycleHook`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menambahkan hook siklus hidup yang ditentukan ke grup Auto Scaling yang ditentukan.

```
Write-ASLifecycleHook -AutoScalingGroupName my-asg -LifecycleHookName
"myLifecycleHook" -LifecycleTransition "autoscaling:EC2_INSTANCE_LAUNCHING" -
NotificationTargetARN "arn:aws:sns:us-west-2:123456789012:my-sns-topic" -RoleARN
"arn:aws:iam::123456789012:role/my-iam-role"
```

- Untuk detail API, lihat [PutLifecycleHook](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Write-ASNotificationConfiguration

Contoh kode berikut menunjukkan cara menggunakan `Write-ASNotificationConfiguration`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengonfigurasi grup Auto Scaling yang ditentukan untuk mengirim pemberitahuan ke topik SNS yang ditentukan saat meluncurkan instance. EC2

```
Write-ASNotificationConfiguration -AutoScalingGroupName my-asg -NotificationType
"autoscaling:EC2_INSTANCE_LAUNCH" -TopicARN "arn:aws:sns:us-west-2:123456789012:my-
topic"
```

Contoh 2: Contoh ini mengonfigurasi grup Auto Scaling yang ditentukan untuk mengirim pemberitahuan ke topik SNS yang ditentukan saat meluncurkan atau mengakhiri instance. EC2

```
Write-ASNotificationConfiguration -AutoScalingGroupName my-asg -NotificationType
@("autoscaling:EC2_INSTANCE_LAUNCH", "autoscaling:EC2_INSTANCE_TERMINATE") -
TopicARN "arn:aws:sns:us-west-2:123456789012:my-topic"
```

- Untuk detail API, lihat [PutNotificationConfiguration](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Write-ASScalingPolicy

Contoh kode berikut menunjukkan cara menggunakan `Write-ASScalingPolicy`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menambahkan kebijakan yang ditentukan ke grup Auto Scaling yang ditentukan. Jenis penyesuaian yang ditentukan menentukan bagaimana menafsirkan `ScalingAdjustment` parameter. Dengan `ChangeInCapacity`, nilai positif meningkatkan kapasitas

dengan jumlah instance yang ditentukan dan nilai negatif menurunkan kapasitas dengan jumlah instance yang ditentukan.

```
Write-ASScalingPolicy -AutoScalingGroupName my-asg -AdjustmentType  
"ChangeInCapacity" -PolicyName "myScaleInPolicy" -ScalingAdjustment -1
```

Output:

```
arn:aws:autoscaling:us-west-2:123456789012:scalingPolicy:aa3836ab-5462-42c7-adab-  
e1d769fc24ef:autoScalingGroupName/my-asg  
:policyName/myScaleInPolicy
```

- Untuk detail API, lihat [PutScalingPolicy](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Write-ASScheduledUpdateGroupAction

Contoh kode berikut menunjukkan cara menggunakan `Write-ASScheduledUpdateGroupAction`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat atau memperbarui tindakan terjadwal satu kali untuk mengubah kapasitas yang diinginkan pada waktu mulai yang ditentukan.

```
Write-ASScheduledUpdateGroupAction -AutoScalingGroupName my-asg -ScheduledActionName  
"myScheduledAction" -StartTime "2015-12-01T00:00:00Z" -DesiredCapacity 10
```

- Untuk detail API, lihat [PutScheduledUpdateGroupAction](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

AWS Budgets contoh menggunakan Alat untuk PowerShell V4

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan AWS Budgets.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

New-BGTBudget

Contoh kode berikut menunjukkan cara menggunakan New-BGTBudget.

Alat untuk PowerShell V4

Contoh 1: Membuat anggaran baru dengan batasan anggaran dan waktu yang ditentukan dengan pemberitahuan email.

```
$notification = @{
    NotificationType = "ACTUAL"
    ComparisonOperator = "GREATER_THAN"
    Threshold = 80
}

$addressObject = @{
    Address = @"user@domain.com"
    SubscriptionType = "EMAIL"
}

$subscriber = New-Object Amazon.Budgets.Model.NotificationWithSubscribers
$subscriber.Notification = $notification
$subscriber.Subscribers.Add($addressObject)

$startDate = [datetime]::new(2017,09,25)
$endDate = [datetime]::new(2017,10,25)

New-BGTBudget -Budget_BudgetName "Tester" -Budget_BudgetType COST -
CostTypes_IncludeTax $true -Budget_TimeUnit MONTHLY -BudgetLimit_Unit USD -
TimePeriod_Start $startDate -TimePeriod_End $endDate -AccountId 123456789012 -
BudgetLimit_Amount 200 -NotificationsWithSubscriber $subscriber
```

- Untuk detail API, lihat [CreateBudget](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

AWS Cloud9 contoh menggunakan Alat untuk PowerShell V4

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan AWS Cloud9.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

Get-C9EnvironmentData

Contoh kode berikut menunjukkan cara menggunakan `Get-C9EnvironmentData`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan informasi tentang lingkungan pengembangan AWS Cloud9 yang ditentukan.

```
Get-C9EnvironmentData -EnvironmentId
685f892f431b45c2b28cb69eadcdb0EX,1980b80e5f584920801c09086667f0EX
```

Output:

```
Arn          : arn:aws:cloud9:us-
east-1:123456789012:environment:685f892f431b45c2b28cb69eadcdb0EX
Description  : Created from CodeStar.
Id           : 685f892f431b45c2b28cb69eadcdb0EX
Lifecycle    : Amazon.Cloud9.Model.EnvironmentLifecycle
Name         : my-demo-ec2-env
OwnerArn     : arn:aws:iam::123456789012:user/MyDemoUser
Type         : ec2
```

```

Arn      : arn:aws:cloud9:us-
east-1:123456789012:environment:1980b80e5f584920801c09086667f0EX
Description :
Id       : 1980b80e5f584920801c09086667f0EX
Lifecycle : Amazon.Cloud9.Model.EnvironmentLifecycle
Name     : my-demo-ssh-env
OwnerArn  : arn:aws:iam::123456789012:user/MyDemoUser
Type     : ssh

```

Contoh 2: Contoh ini mendapatkan informasi tentang status siklus hidup lingkungan pengembangan Cloud9 AWS yang ditentukan.

```
(Get-C9EnvironmentData -EnvironmentId 685f892f431b45c2b28cb69eadcdb0EX).Lifecycle
```

Output:

```

FailureResource Reason Status
-----
                        CREATED

```

- Untuk detail API, lihat [DescribeEnvironments](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-C9EnvironmentList

Contoh kode berikut menunjukkan cara menggunakan `Get-C9EnvironmentList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan daftar pengidentifikasi lingkungan pengembangan AWS Cloud9 yang tersedia.

```
Get-C9EnvironmentList
```

Output:

```

685f892f431b45c2b28cb69eadcdb0EX
1980b80e5f584920801c09086667f0EX

```

- Untuk detail API, lihat [ListEnvironments](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-C9EnvironmentMembershipList

Contoh kode berikut menunjukkan cara menggunakan `Get-C9EnvironmentMembershipList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan informasi tentang anggota lingkungan untuk lingkungan pengembangan AWS Cloud9 yang ditentukan.

```
Get-C9EnvironmentMembershipList -EnvironmentId ffd88420d4824eeeeaeaa8a04bfde8cEX
```

Output:

```
EnvironmentId : ffd88420d4824eeeeaeaa8a04bfde8cEX
LastAccess    : 1/1/0001 12:00:00 AM
Permissions   : read-write
UserArn       : arn:aws:iam::123456789012:user/AnotherDemoUser
UserId       : AIDAJ3BA602FMJWCWXHEX

EnvironmentId : ffd88420d4824eeeeaeaa8a04bfde8cEX
LastAccess    : 1/1/0001 12:00:00 AM
Permissions   : owner
UserArn       : arn:aws:iam::123456789012:user/MyDemoUser
UserId       : AIDAJ3LOROMOUCTBSU6EX
```

Contoh 2: Contoh ini mendapatkan informasi tentang pemilik lingkungan pengembangan AWS Cloud9 yang ditentukan.

```
Get-C9EnvironmentMembershipList -EnvironmentId ffd88420d4824eeeeaeaa8a04bfde8cEX -
Permission owner
```

Output:

```
EnvironmentId : ffd88420d4824eeeeaeaa8a04bfde8cEX
LastAccess    : 1/1/0001 12:00:00 AM
Permissions   : owner
UserArn       : arn:aws:iam::123456789012:user/MyDemoUser
UserId       : AIDAJ3LOROMOUCTBSU6EX
```

Contoh 3: Contoh ini mendapatkan informasi tentang anggota lingkungan yang ditentukan untuk beberapa lingkungan pengembangan AWS Cloud9.

```
Get-C9EnvironmentMembershipList -UserArn arn:aws:iam::123456789012:user/MyDemoUser
```

Output:

```
EnvironmentId : ffd88420d4824eeeeaaa8a04bfde8cEX
LastAccess    : 1/17/2018 7:48:14 PM
Permissions   : owner
UserArn       : arn:aws:iam::123456789012:user/MyDemoUser
UserId        : AIDAJ3LOROMOUXTBSUGEX

EnvironmentId : 1980b80e5f584920801c09086667f0EX
LastAccess    : 1/16/2018 11:21:24 PM
Permissions   : owner
UserArn       : arn:aws:iam::123456789012:user/MyDemoUser
UserId        : AIDAJ3LOROMOUXTBSUGEX
```

- Untuk detail API, lihat [DescribeEnvironmentMemberships](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-C9EnvironmentStatus

Contoh kode berikut menunjukkan cara menggunakan `Get-C9EnvironmentStatus`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan informasi status untuk lingkungan pengembangan AWS Cloud9 yang ditentukan.

```
Get-C9EnvironmentStatus -EnvironmentId 349c86d4579e4e7298d500ff57a6b2EX
```

Output:

```
Message                Status
-----
Environment is ready to use ready
```

- Untuk detail API, lihat [DescribeEnvironmentStatus](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-C9EnvironmentEC2

Contoh kode berikut menunjukkan cara menggunakan `New-C9EnvironmentEC2`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat lingkungan pengembangan AWS Cloud9 dengan setelan yang ditentukan, meluncurkan instance Amazon Elastic Compute Cloud (EC2Amazon), dan kemudian menghubungkan dari instance ke lingkungan.

```
New-C9EnvironmentEC2 -Name my-demo-env -AutomaticStopTimeMinutes 60 -Description
"My demonstration development environment." -InstanceType t2.micro -OwnerArn
arn:aws:iam::123456789012:user/MyDemoUser -SubnetId subnet-d43a46EX
```

Output:

```
ffd88420d4824eeeeaaa8a04bfde8cEX
```

- Untuk detail API, lihat [CreateEnvironmentEc2](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-C9EnvironmentMembership

Contoh kode berikut menunjukkan cara menggunakan `New-C9EnvironmentMembership`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menambahkan anggota lingkungan yang ditentukan ke lingkungan pengembangan AWS Cloud9 yang ditentukan.

```
New-C9EnvironmentMembership -UserArn arn:aws:iam::123456789012:user/AnotherDemoUser
-EnvironmentId ffd88420d4824eeeeaaa8a04bfde8cEX -Permission read-write
```

Output:

```
EnvironmentId : ffd88420d4824eeeeaaa8a04bfde8cEX
LastAccess    : 1/1/0001 12:00:00 AM
Permissions   : read-write
UserArn       : arn:aws:iam::123456789012:user/AnotherDemoUser
UserId       : AIDAJ3BA602FMJWCXHEX
```

- Untuk detail API, lihat [CreateEnvironmentMembership](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-C9Environment

Contoh kode berikut menunjukkan cara menggunakan `Remove-C9Environment`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus lingkungan pengembangan AWS Cloud9 yang ditentukan. Jika EC2 instans Amazon terhubung ke lingkungan, juga menghentikan instance.

```
Remove-C9Environment -EnvironmentId ffd88420d4824eeeeaea8a04bfde8cEX
```

- Untuk detail API, lihat [DeleteEnvironment](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-C9EnvironmentMembership

Contoh kode berikut menunjukkan cara menggunakan `Remove-C9EnvironmentMembership`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus anggota lingkungan tertentu dari lingkungan pengembangan AWS Cloud9 yang ditentukan.

```
Remove-C9EnvironmentMembership -UserArn arn:aws:iam::123456789012:user/  
AnotherDemoUser -EnvironmentId ffd88420d4824eeeeaea8a04bfde8cEX
```

- Untuk detail API, lihat [DeleteEnvironmentMembership](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-C9Environment

Contoh kode berikut menunjukkan cara menggunakan `Update-C9Environment`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengubah pengaturan yang ditentukan dari lingkungan pengembangan AWS Cloud9 yang sudah ditentukan.

```
Update-C9Environment -EnvironmentId ffd88420d4824eeeeaea8a04bfde8cEX -Description
"My changed demonstration development environment." -Name my-changed-demo-env
```

- Untuk detail API, lihat [UpdateEnvironment](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-C9EnvironmentMembership

Contoh kode berikut menunjukkan cara menggunakan `Update-C9EnvironmentMembership`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengubah pengaturan anggota lingkungan yang ada yang ditentukan untuk lingkungan pengembangan AWS Cloud9 yang ditentukan.

```
Update-C9EnvironmentMembership -UserArn arn:aws:iam::123456789012:user/
AnotherDemoUser -EnvironmentId ffd88420d4824eeeeaea8a04bfde8cEX -Permission read-
only
```

Output:

```
EnvironmentId : ffd88420d4824eeeeaea8a04bfde8cEX
LastAccess    : 1/1/0001 12:00:00 AM
Permissions   : read-only
UserArn       : arn:aws:iam::123456789012:user/AnotherDemoUser
UserId        : AIDAJ3BA602FMJWCWHEX
```

- Untuk detail API, lihat [UpdateEnvironmentMembership](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

AWS CloudFormation contoh menggunakan Alat untuk PowerShell V4

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan AWS CloudFormation.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

Get-CFNStack

Contoh kode berikut menunjukkan cara menggunakan `Get-CFNStack`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan koleksi instance Stack yang menjelaskan semua tumpukan pengguna.

```
Get-CFNStack
```

Contoh 2: Mengembalikan instance Stack yang menjelaskan tumpukan yang ditentukan

```
Get-CFNStack -StackName "myStack"
```

- Untuk detail API, lihat [DescribeStacks](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CFNStackEvent

Contoh kode berikut menunjukkan cara menggunakan `Get-CFNStackEvent`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan semua peristiwa terkait tumpukan untuk tumpukan tertentu.

```
Get-CFNStackEvent -StackName "myStack"
```

- Untuk detail API, lihat [DescribeStackEvents](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CFNStackResource

Contoh kode berikut menunjukkan cara menggunakan `Get-CFNStackResource`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan deskripsi sumber daya yang diidentifikasi dalam template yang terkait dengan tumpukan yang ditentukan oleh ID logis DBInstance “Saya”.

```
Get-CFNStackResource -StackName "myStack" -LogicalResourceId "MyDBInstance"
```

- Untuk detail API, lihat [DescribeStackResource](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CFNStackResourceList

Contoh kode berikut menunjukkan cara menggunakan `Get-CFNStackResourceList`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan deskripsi AWS sumber daya hingga 100 sumber daya yang terkait dengan tumpukan yang ditentukan. Untuk mendapatkan rincian semua sumber daya yang terkait dengan tumpukan gunakan `Get-CFNStackResourceSummary`, yang juga mendukung paging manual hasil.

```
Get-CFNStackResourceList -StackName "myStack"
```

Contoh 2: Mengembalikan deskripsi EC2 instance Amazon yang diidentifikasi dalam template yang terkait dengan tumpukan yang ditentukan oleh ID logis “Ec2Instance”.

```
Get-CFNStackResourceList -StackName "myStack" -LogicalResourceId "Ec2Instance"
```

Contoh 3: Mengembalikan deskripsi hingga 100 sumber daya yang terkait dengan tumpukan yang berisi instance Amazon yang diidentifikasi oleh ID EC2 instance “i-123456”. Untuk mendapatkan rincian semua sumber daya yang terkait dengan tumpukan gunakan `Get-CFNStackResourceSummary`, yang juga mendukung paging manual hasil.

```
Get-CFNStackResourceList -PhysicalResourceId "i-123456"
```

Contoh 4: Mengembalikan deskripsi EC2 instance Amazon yang diidentifikasi oleh ID logis “Ec2Instance” dalam template untuk tumpukan. Tumpukan diidentifikasi menggunakan ID sumber daya fisik dari sumber daya yang dikandungnya, dalam hal ini juga instance Amazon

dengan ID EC2 instance "i-123456". Sumber daya fisik yang berbeda juga dapat digunakan untuk mengidentifikasi tumpukan tergantung pada konten template, misalnya bucket Amazon S3.

```
Get-CFNStackResourceList -PhysicalResourceId "i-123456" -LogicalResourceId  
"Ec2Instance"
```

- Untuk detail API, lihat [DescribeStackResources](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CFNStackResourceSummary

Contoh kode berikut menunjukkan cara menggunakan `Get-CFNStackResourceSummary`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan deskripsi dari semua sumber daya yang terkait dengan tumpukan tertentu.

```
Get-CFNStackResourceSummary -StackName "myStack"
```

- Untuk detail API, lihat [ListStackResources](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CFNStackSummary

Contoh kode berikut menunjukkan cara menggunakan `Get-CFNStackSummary`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan informasi ringkasan untuk semua tumpukan.

```
Get-CFNStackSummary
```

Contoh 2: Mengembalikan informasi ringkasan untuk semua tumpukan yang sedang dibuat.

```
Get-CFNStackSummary -StackStatusFilter "CREATE_IN_PROGRESS"
```

Contoh 3: Mengembalikan informasi ringkasan untuk semua tumpukan yang sedang dibuat atau diperbarui.

```
Get-CFNStackSummary -StackStatusFilter @"CREATE_IN_PROGRESS", "UPDATE_IN_PROGRESS")
```

- Untuk detail API, lihat [ListStacks](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CFNTemplate

Contoh kode berikut menunjukkan cara menggunakan `Get-CFNTemplate`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan template yang terkait dengan tumpukan tertentu.

```
Get-CFNTemplate -StackName "myStack"
```

- Untuk detail API, lihat [GetTemplate](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Measure-CFNTemplateCost

Contoh kode berikut menunjukkan cara menggunakan `Measure-CFNTemplateCost`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan URL Kalkulator Bulanan AWS Sederhana dengan string kueri yang menjelaskan sumber daya yang diperlukan untuk menjalankan template. Template diperoleh dari URL Amazon S3 yang ditentukan dan parameter penyesuaian tunggal diterapkan. Parameter juga dapat ditentukan menggunakan 'Kunci' dan 'Nilai' bukan 'ParameterKey' dan 'ParameterValue'.

```
Measure-CFNTemplateCost -TemplateURL https://s3.amazonaws.com/amzn-s3-demo-bucket/
templatefile.template `
                        -Region us-west-1 `
                        -Parameter @{ ParameterKey="KeyName";
ParameterKey="myKeyPairName" }
```

Contoh 2: Mengembalikan URL Kalkulator Bulanan AWS Sederhana dengan string kueri yang menjelaskan sumber daya yang diperlukan untuk menjalankan template. Template diurai dari konten yang disediakan dan parameter kustomisasi diterapkan (contoh ini mengasumsikan konten template akan mendeklarasikan dua parameter, " dan KeyName 'InstanceType'). Parameter kustomisasi juga dapat ditentukan menggunakan 'Kunci' dan 'Nilai' alih-alih 'ParameterKey' dan 'ParameterValue'.

```
Measure-CFNTemplateCost -TemplateBody "{TEMPLATE CONTENT HERE}" `
    -Parameter @( @{ ParameterKey="KeyName";
    ParameterValue="myKeyPairName" }, `
    @{ ParameterKey="InstanceType";
    ParameterValue="m1.large" })
```

Contoh 3: Menggunakan New-Object untuk membangun set parameter template dan mengembalikan URL Kalkulator Bulanan AWS Sederhana dengan string kueri yang menjelaskan sumber daya yang diperlukan untuk menjalankan template. Template diurai dari konten yang disediakan, dengan parameter kustomisasi (contoh ini mengasumsikan konten template akan mendeklarasikan dua parameter, " dan KeyName 'InstanceType').

```
$p1 = New-Object -Type Amazon.CloudFormation.Model.Parameter
$p1.ParameterKey = "KeyName"
$p1.ParameterValue = "myKeyPairName"

$p2 = New-Object -Type Amazon.CloudFormation.Model.Parameter
$p2.ParameterKey = "InstanceType"
$p2.ParameterValue = "m1.large"

Measure-CFNTemplateCost -TemplateBody "{TEMPLATE CONTENT HERE}" -Parameter @( $p1,
    $p2 )
```

- Untuk detail API, lihat [EstimateTemplateCost](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-CFNStack

Contoh kode berikut menunjukkan cara menggunakan New-CFNStack.

Alat untuk PowerShell V4

Contoh 1: Membuat tumpukan baru dengan nama yang ditentukan. Template diurai dari konten yang disediakan dengan parameter kustomisasi ('PK1' dan 'PK2' mewakili nama parameter yang dideklarasikan dalam konten template, 'PV1' dan 'PV2' mewakili nilai untuk parameter tersebut. Parameter kustomisasi juga dapat ditentukan menggunakan 'Kunci' dan 'Nilai' alih-alih 'ParameterKey' dan 'ParameterValue'. Jika pembuatan tumpukan gagal, itu tidak akan digulung kembali.

```
New-CFNStack -StackName "myStack" `
```

```
-TemplateBody "{TEMPLATE CONTENT HERE}" `
-Parameter @( @{ ParameterKey="PK1"; ParameterValue="PV1" },
@{ ParameterKey="PK2"; ParameterValue="PV2" }) `
-DisableRollback $true
```

Contoh 2: Membuat tumpukan baru dengan nama yang ditentukan. Template diurai dari konten yang disediakan dengan parameter kustomisasi ('PK1' dan 'PK2' mewakili nama parameter yang dideklarasikan dalam konten template, 'PV1' dan 'PV2' mewakili nilai untuk parameter tersebut. Parameter kustomisasi juga dapat ditentukan menggunakan 'Kunci' dan 'Nilai' alih-alih 'ParameterKey' dan 'ParameterValue'. Jika pembuatan tumpukan gagal, itu akan digulung kembali.

```
$p1 = New-Object -Type Amazon.CloudFormation.Model.Parameter
$p1.ParameterKey = "PK1"
$p1.ParameterValue = "PV1"

$p2 = New-Object -Type Amazon.CloudFormation.Model.Parameter
$p2.ParameterKey = "PK2"
$p2.ParameterValue = "PV2"

New-CFNStack -StackName "myStack" `
-TemplateBody "{TEMPLATE CONTENT HERE}" `
-Parameter @( $p1, $p2 ) `
-OnFailure "ROLLBACK"
```

Contoh 3: Membuat tumpukan baru dengan nama yang ditentukan. Template diperoleh dari URL Amazon S3 dengan parameter kustomisasi ('PK1' mewakili nama parameter yang dideklarasikan dalam konten template, 'PV1' mewakili nilai untuk parameter. Parameter kustomisasi juga dapat ditentukan menggunakan 'Kunci' dan 'Nilai' alih-alih 'ParameterKey' dan 'ParameterValue'. Jika pembuatan tumpukan gagal, itu akan digulung kembali (sama seperti menentukan -DisableRollback \$false).

```
New-CFNStack -StackName "myStack" `
-TemplateURL https://s3.amazonaws.com/amzn-s3-demo-bucket/
templatefile.template `
-Parameter @{ ParameterKey="PK1"; ParameterValue="PV1" }
```

Contoh 4: Membuat tumpukan baru dengan nama yang ditentukan. Template diperoleh dari URL Amazon S3 dengan parameter kustomisasi ('PK1' mewakili nama parameter yang dideklarasikan dalam konten template, 'PV1' mewakili nilai untuk parameter. Parameter kustomisasi juga dapat

ditentukan menggunakan 'Kunci' dan 'Nilai' alih-alih 'ParameterKey' dan 'ParameterValue'. Jika pembuatan tumpukan gagal, itu akan digulung kembali (sama seperti menentukan -DisableRollback \$false). Pemberitahuan yang ditentukan AENs akan menerima acara terkait tumpukan yang dipublikasikan.

```
New-CFNStack -StackName "myStack" `
              -TemplateURL https://s3.amazonaws.com/amzn-s3-demo-bucket/
templatefile.template `
              -Parameter @{ ParameterKey="PK1"; ParameterValue="PV1" } `
              -NotificationARN @( "arn1", "arn2" )
```

- Untuk detail API, lihat [CreateStack](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-CFNStack

Contoh kode berikut menunjukkan cara menggunakan `Remove-CFNStack`.

Alat untuk PowerShell V4

Contoh 1: Menghapus tumpukan yang ditentukan.

```
Remove-CFNStack -StackName "myStack"
```

- Untuk detail API, lihat [DeleteStack](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Resume-CFNUpdateRollback

Contoh kode berikut menunjukkan cara menggunakan `Resume-CFNUpdateRollback`.

Alat untuk PowerShell V4

Contoh 1: Melanjutkan rollback dari tumpukan bernama, yang seharusnya dalam status 'UPDATE_ROLLBACK_FAILED'. Jika rollback lanjutan berhasil, tumpukan akan memasukkan status 'UPDATE_ROLLBACK_COMPLETE'.

```
Resume-CFNUpdateRollback -StackName "myStack"
```

- Untuk detail API, lihat [ContinueUpdateRollback](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Stop-CFNUpdateStack

Contoh kode berikut menunjukkan cara menggunakan `Stop-CFNUpdateStack`.

Alat untuk PowerShell V4

Contoh 1: Membatalkan pembaruan pada tumpukan yang ditentukan.

```
Stop-CFNUpdateStack -StackName "myStack"
```

- Untuk detail API, lihat [CancelUpdateStack](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Test-CFNStack

Contoh kode berikut menunjukkan cara menggunakan `Test-CFNStack`.

Alat untuk PowerShell V4

Contoh 1: Menguji apakah tumpukan telah mencapai salah satu status `UPDATE_ROLLBACK_COMPLETE`, `CREATE_COMPLETE`, `ROLLBACK_COMPLETE` atau `UPDATE_COMPLETE`.

```
Test-CFNStack -StackName MyStack
```

Output:

```
False
```

Contoh 2: Menguji apakah tumpukan telah mencapai status `UPDATE_COMPLETE` atau `UPDATE_ROLLBACK_COMPLETE`.

```
Test-CFNStack -StackName MyStack -Status UPDATE_COMPLETE,UPDATE_ROLLBACK_COMPLETE
```

Output:

```
True
```

- Untuk detail API, lihat [Menguji- CFNStack](#) dalam Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Test-CFNTemplate

Contoh kode berikut menunjukkan cara menggunakan `Test-CFNTemplate`.

Alat untuk PowerShell V4

Contoh 1: Memvalidasi konten template yang ditentukan. Output merinci kemampuan, deskripsi, dan parameter template.

```
Test-CFNTemplate -TemplateBody "{TEMPLATE CONTENT HERE}"
```

Contoh 2: Memvalidasi template tertentu yang diakses melalui URL Amazon S3. Output merinci kemampuan, deskripsi, dan parameter template.

```
Test-CFNTemplate -TemplateURL https://s3.amazonaws.com/amzn-s3-demo-bucket/
templatefile.template
```

- Untuk detail API, lihat [ValidateTemplate](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-CFNStack

Contoh kode berikut menunjukkan cara menggunakan `Update-CFNStack`.

Alat untuk PowerShell V4

Contoh 1: Memperbarui tumpukan 'MyStack' dengan template dan parameter kustomisasi yang ditentukan. 'PK1' mewakili nama parameter yang dideklarasikan dalam template dan 'PV1' mewakili nilainya. Parameter kustomisasi juga dapat ditentukan menggunakan 'Kunci' dan 'Nilai' alih-alih 'ParameterKey' dan 'ParameterValue'.

```
Update-CFNStack -StackName "myStack" `
    -TemplateBody "{Template Content Here}" `
    -Parameter @{ ParameterKey="PK1"; ParameterValue="PV1" }
```

Contoh 2: Memperbarui tumpukan 'MyStack' dengan template dan parameter kustomisasi yang ditentukan. 'PK1' dan 'PK2' mewakili nama parameter yang dideklarasikan dalam templat, 'PV1' dan 'PV2' mewakili nilai yang diminta. Parameter kustomisasi juga dapat ditentukan menggunakan 'Kunci' dan 'Nilai' alih-alih 'ParameterKey' dan 'ParameterValue'.

```
Update-CFNStack -StackName "myStack" `
```

```
-TemplateBody "{Template Content Here}" `
-Parameter @( @{ ParameterKey="PK1"; ParameterValue="PV1" },
@{ ParameterKey="PK2"; ParameterValue="PV2" } )
```

Contoh 3: Memperbarui tumpukan 'MyStack' dengan template dan parameter kustomisasi yang ditentukan. 'PK1' mewakili nama parameter yang dideklarasikan dalam template dan 'PV2' mewakili nilainya. Parameter kustomisasi juga dapat ditentukan menggunakan 'Kunci' dan 'Nilai' alih-alih 'ParameterKey' dan 'ParameterValue'.

```
Update-CFNStack -StackName "myStack" -TemplateBody "{Template Content Here}" -
Parameters @{ ParameterKey="PK1"; ParameterValue="PV1" }
```

Contoh 4: Memperbarui tumpukan 'MyStack' dengan templat yang ditentukan, diperoleh dari Amazon S3, dan parameter penyesuaian. 'PK1' dan 'PK2' mewakili nama parameter yang dideklarasikan dalam templat, 'PV1' dan 'PV2' mewakili nilai yang diminta. Parameter kustomisasi juga dapat ditentukan menggunakan 'Kunci' dan 'Nilai' alih-alih 'ParameterKey' dan 'ParameterValue'.

```
Update-CFNStack -StackName "myStack" `
-TemplateURL https://s3.amazonaws.com/amzn-s3-demo-bucket/
templatefile.template `
-Parameter @( @{ ParameterKey="PK1"; ParameterValue="PV1" },
@{ ParameterKey="PK2"; ParameterValue="PV2" } )
```

Contoh 5: Memperbarui tumpukan 'MyStack', yang diasumsikan dalam contoh ini berisi sumber daya IAM, dengan templat yang ditentukan, diperoleh dari Amazon S3, dan parameter penyesuaian. 'PK1' dan 'PK2' mewakili nama parameter yang dideklarasikan dalam templat, 'PV1' dan 'PV2' mewakili nilai yang diminta. Parameter kustomisasi juga dapat ditentukan menggunakan 'Kunci' dan 'Nilai' alih-alih 'ParameterKey' dan 'ParameterValue'. Tumpukan yang berisi sumber daya IAM mengharuskan Anda untuk menentukan parameter -Capabilities "CAPABILITY_IAM" jika tidak pembaruan akan gagal dengan kesalahan ". InsufficientCapabilities

```
Update-CFNStack -StackName "myStack" `
-TemplateURL https://s3.amazonaws.com/amzn-s3-demo-bucket/
templatefile.template `
-Parameter @( @{ ParameterKey="PK1"; ParameterValue="PV1" },
@{ ParameterKey="PK2"; ParameterValue="PV2" } ) `
-Capabilities "CAPABILITY_IAM"
```

- Untuk detail API, lihat [UpdateStack](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Wait-CFNStack

Contoh kode berikut menunjukkan cara menggunakan Wait-CFNStack.

Alat untuk PowerShell V4

Contoh 1: Menguji apakah tumpukan telah mencapai salah satu status UPDATE_ROLLBACK_COMPLETE, CREATE_COMPLETE, ROLLBACK_COMPLETE atau UPDATE_COMPLETE. Jika tumpukan tidak berada di salah satu status, perintah tidur selama dua detik sebelum menguji status lagi. Ini diulang sampai tumpukan mencapai salah satu status yang diminta atau periode batas waktu default 60 detik berlalu. Jika periode batas waktu terlampaui, pengecualian dilemparkan. Jika tumpukan mencapai salah satu status yang diminta dalam periode batas waktu, tumpukan tersebut dikembalikan ke pipeline.

```
$stack = Wait-CFNStack -StackName MyStack
```

Contoh 2: Contoh ini menunggu total 5 menit (300 detik) hingga tumpukan mencapai salah satu status yang ditentukan. Dalam contoh ini status tercapai sebelum batas waktu dan oleh karena itu objek tumpukan dikembalikan ke pipeline.

```
Wait-CFNStack -StackName MyStack -Timeout 300 -Status  
CREATE_COMPLETE,ROLLBACK_COMPLETE
```

Output:

```
Capabilities      : {CAPABILITY_IAM}
ChangeSetId       :
CreationTime      : 6/1/2017 9:29:33 AM
Description       : AWS CloudFormation Sample Template
ec2_instance_with_instance_profile: Create an EC2 instance with an associated
instance profile. **WARNING** This template creates one or more Amazon EC2
instances and an Amazon SQS queue. You will be billed for the
AWS resources used if you create a stack from this template.
DisableRollback   : False
LastUpdatedTime   : 1/1/0001 12:00:00 AM
NotificationARNs  : {}
Outputs           : {}
Parameters        : {}
RoleARN           :
StackId           : arn:aws:cloudformation:us-west-2:123456789012:stack/
MyStack/7ea87b50-46e7-11e7-9c9b-503a90a9c4d1
```

```
StackName      : MyStack
StackStatus    : CREATE_COMPLETE
StackStatusReason :
Tags           : {}
TimeoutInMinutes : 0
```

Contoh 3: Contoh ini menunjukkan keluaran kesalahan ketika tumpukan tidak mencapai salah satu status yang diminta dalam periode batas waktu (dalam hal ini periode default 60 detik).

```
Wait-CFNStack -StackName MyStack -Status CREATE_COMPLETE,ROLLBACK_COMPLETE
```

Output:

```
Wait-CFNStack : Timed out after 60 seconds waiting for CloudFormation
stack MyStack in region us-west-2 to reach one of state(s):
UPDATE_ROLLBACK_COMPLETE,CREATE_COMPLETE,ROLLBACK_COMPLETE,UPDATE_COMPLETE
At line:1 char:1
+ Wait-CFNStack -StackName MyStack -State CREATE_COMPLETE,ROLLBACK_COMPLETE
+ ~~~~~
+ CategoryInfo          : InvalidOperation:
(Amazon.PowerShe...tCFNStackCmdlet:WaitCFNStackCmdlet) [Wait-CFNStack],
InvalidOperationException
+ FullyQualifiedErrorId :
InvalidOperationException,Amazon.PowerShell.Cmdlets.CFN.WaitCFNStackCmdlet
```

- Untuk detail API, lihat [Tunggu- CFNStack](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

CloudFront contoh menggunakan Alat untuk PowerShell V4

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan CloudFront.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

Get-CFCloudFrontOriginAccessIdentity

Contoh kode berikut menunjukkan cara menggunakan `Get-CFCloudFrontOriginAccessIdentity`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan identitas akses CloudFront asal Amazon tertentu, yang ditentukan oleh parameter `-Id`. Meskipun parameter `-Id` tidak diperlukan, jika Anda tidak menentukannya, tidak ada hasil yang dikembalikan.

```
Get-CFCloudFrontOriginAccessIdentity -Id E3XXXXXXXXXXRT
```

Output:

```
CloudFrontOriginAccessIdentityConfig  Id
S3CanonicalUserId
-----
Amazon.CloudFront.Model.CloudFrontOr... E3XXXXXXXXXXRT
4b6e...
```

- Untuk detail API, lihat [GetCloudFrontOriginAccessIdentity](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CFCloudFrontOriginAccessIdentityConfig

Contoh kode berikut menunjukkan cara menggunakan `Get-CFCloudFrontOriginAccessIdentityConfig`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan informasi konfigurasi tentang identitas akses CloudFront asal Amazon tunggal, yang ditentukan oleh parameter `-Id`. Kesalahan terjadi jika tidak ada parameter `-Id` yang ditentukan..

```
Get-CFCloudFrontOriginAccessIdentityConfig -Id E3XXXXXXXXXXRT
```

Output:

CallerReference	Comment
-----	-----
mycallerreference: 2/1/2011 1:16:32 PM	Caller reference:
2/1/2011 1:16:32 PM	

- Untuk detail API, lihat [GetCloudFrontOriginAccessIdentityConfig](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CFCloudFrontOriginAccessIdentityList

Contoh kode berikut menunjukkan cara menggunakan `Get-CFCloudFrontOriginAccessIdentityList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan daftar identitas akses CloudFront asal Amazon. Karena `MaxItem` parameter menentukan nilai 2, hasilnya mencakup dua identitas.

```
Get-CFCloudFrontOriginAccessIdentityList -MaxItem 2
```

Output:

```
IsTruncated : True
Items       : {E326XXXXXXXXXT, E1YXXXXXXXX9B}
Marker      :
MaxItems    : 2
NextMarker  : E1YXXXXXXXX9B
Quantity    : 2
```

- Untuk detail API, lihat [ListCloudFrontOriginAccessIdentities](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CFDistribution

Contoh kode berikut menunjukkan cara menggunakan `Get-CFDistribution`.

Alat untuk PowerShell V4

Contoh 1: Mengambil informasi untuk distribusi tertentu.

```
Get-CFDistribution -Id EXAMPLE0000ID
```

- Untuk detail API, lihat [GetDistribution](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CFDistributionConfig

Contoh kode berikut menunjukkan cara menggunakan `Get-CFDistributionConfig`.

Alat untuk PowerShell V4

Contoh 1: Mengambil konfigurasi untuk distribusi tertentu.

```
Get-CFDistributionConfig -Id EXAMPLE0000ID
```

- Untuk detail API, lihat [GetDistributionConfig](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CFDistributionList

Contoh kode berikut menunjukkan cara menggunakan `Get-CFDistributionList`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan distribusi.

```
Get-CFDistributionList
```

- Untuk detail API, lihat [ListDistributions](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-CFDistribution

Contoh kode berikut menunjukkan cara menggunakan `New-CFDistribution`.

Alat untuk PowerShell V4

Contoh 1: Membuat CloudFront distribusi dasar, dikonfigurasi dengan logging dan caching.

```

$origin = New-Object Amazon.CloudFront.Model.Origin
$origin.DomainName = "amzn-s3-demo-bucket.s3.amazonaws.com"
$origin.Id = "UniqueOrigin1"
$origin.S3originConfig = New-Object Amazon.CloudFront.Model.S3originConfig
$origin.S3originConfig.OriginAccessIdentity = ""
New-CFDistribution `
    -DistributionConfig_Enabled $true `
    -DistributionConfig_Comment "Test distribution" `
    -Origins_Item $origin `
    -Origins_Quantity 1 `
    -Logging_Enabled $true `
    -Logging_IncludeCookie $true `
    -Logging_Bucket amzn-s3-demo-logging-bucket.s3.amazonaws.com `
    -Logging_Prefix "help/" `
    -DistributionConfig_CallerReference Client1 `
    -DistributionConfig_DefaultRootObject index.html `
    -DefaultCacheBehavior_TargetOriginId $origin.Id `
    -ForwardedValues_QueryString $true `
    -Cookies_Forward all `
    -WhitelistedNames_Quantity 0 `
    -TrustedSigners_Enabled $false `
    -TrustedSigners_Quantity 0 `
    -DefaultCacheBehavior_ViewerProtocolPolicy allow-all `
    -DefaultCacheBehavior_MinTTL 1000 `
    -DistributionConfig_PriceClass "PriceClass_All" `
    -CacheBehaviors_Quantity 0 `
    -Aliases_Quantity 0

```

- Untuk detail API, lihat [CreateDistribution](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-CFInvalidation

Contoh kode berikut menunjukkan cara menggunakan `New-CFInvalidation`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat pembatalan baru pada distribusi dengan ID `EXAMPLENSTXAXE`. `CallerReference` ini adalah ID unik yang dipilih oleh pengguna; dalam hal ini, cap waktu yang mewakili 15 Mei 2019 pukul 9:00 pagi digunakan. Variabel `$Paths` menyimpan tiga jalur ke file gambar dan media yang tidak diinginkan pengguna sebagai bagian dari cache distribusi. Nilai parameter `-Paths_Quantity` adalah jumlah total jalur yang ditentukan dalam parameter `-Paths_Item`.

```
$Paths = "/images/*.gif", "/images/image1.jpg", "/videos/*.mp4"
New-CFInvalidation -DistributionId "EXAMPLENSTXAXE" -
InvalidationBatch_CallerReference 20190515090000 -Paths_Item $Paths -Paths_Quantity
3
```

Output:

Invalidation	Location
-----	-----
Amazon.CloudFront.Model.Invalidation	https://cloudfront.amazonaws.com/2018-11-05/distribution/EXAMPLENSTXAXE/invalidation/EXAMPLE8NOK9H

- Untuk detail API, lihat [CreateInvalidation](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-CFSignedCookie

Contoh kode berikut menunjukkan cara menggunakan `New-CFSignedCookie`.

Alat untuk PowerShell V4

Contoh 1: Membuat cookie yang ditandatangani ke sumber daya yang ditentukan menggunakan kebijakan kalengan. Cookie akan berlaku selama satu tahun.

```
$params = @{
  "ResourceUri"="http://xyz.cloudfront.net/image1.jpeg"
  "KeyPairId"="AKIAIOSFODNN7EXAMPLE"
  "PrivateKeyFile"="C:\pk-AKIAIOSFODNN7EXAMPLE.pem"
  "ExpiresOn"=(Get-Date).AddYears(1)
}
New-CFSignedCookie @params
```

Output:

Expires

[CloudFront-Expires, 1472227284]

Contoh 2: Membuat cookie yang ditandatangani ke sumber daya yang ditentukan menggunakan kebijakan khusus. Cookie akan berlaku dalam 24 jam dan akan kedaluwarsa satu minggu sesudahnya.

```
$start = (Get-Date).AddHours(24)
$params = @{
    "ResourceUri"="http://xyz.cloudfront.net/content/*.jpeg"
    "KeyPairId"="AKIAIOSFODNN7EXAMPLE"
    "PrivateKeyFile"="C:\pk-AKIAIOSFODNN7EXAMPLE.pem"
    "ExpiresOn"=$start.AddDays(7)
    "ActiveFrom"=$start
}

New-CFSignedCookie @params
```

Output:

```
Policy
-----
[CloudFront-Policy, eyJTd...wIjo...
```

Contoh 3: Membuat cookie yang ditandatangani ke sumber daya yang ditentukan menggunakan kebijakan khusus. Cookie akan berlaku dalam 24 jam dan akan kedaluwarsa satu minggu sesudahnya. Akses ke sumber daya dibatasi pada rentang ip yang ditentukan.

```
$start = (Get-Date).AddHours(24)
$params = @{
    "ResourceUri"="http://xyz.cloudfront.net/content/*.jpeg"
    "KeyPairId"="AKIAIOSFODNN7EXAMPLE"
    "PrivateKeyFile"="C:\pk-AKIAIOSFODNN7EXAMPLE.pem"
    "ExpiresOn"=$start.AddDays(7)
    "ActiveFrom"=$start
    "IpRange"="192.0.2.0/24"
}

New-CFSignedCookie @params
```

Output:

```
Policy
-----
```

```
[CloudFront-Policy, eyJTd...wIjo...
```

- Untuk detail API, lihat [CFSignedCookie Baru di Referensi Alat AWS untuk PowerShell Cmdlet \(V4\)](#).

New-CFSignedUrl

Contoh kode berikut menunjukkan cara menggunakan `New-CFSignedUrl`.

Alat untuk PowerShell V4

Contoh 1: Membuat url yang ditandatangani ke sumber daya yang ditentukan menggunakan kebijakan kalengan. Url akan berlaku selama satu jam. Objek `System.Uri` yang berisi url yang ditandatangani dipancarkan ke pipeline.

```
$params = @{  
    "ResourceUri"="https://cdn.example.com/index.html"  
    "KeyPairId"="AKIAIOSFODNN7EXAMPLE"  
    "PrivateKeyFile"="C:\pk-AKIAIOSFODNN7EXAMPLE.pem"  
    "ExpiresOn"=(Get-Date).AddHours(1)  
}  
New-CFSignedUrl @params
```

Contoh 2: Membuat url yang ditandatangani ke sumber daya yang ditentukan menggunakan kebijakan khusus. Url akan berlaku mulai dalam 24 jam dan akan kedaluwarsa satu minggu kemudian.

```
$start = (Get-Date).AddHours(24)  
$params = @{  
    "ResourceUri"="https://cdn.example.com/index.html"  
    "KeyPairId"="AKIAIOSFODNN7EXAMPLE"  
    "PrivateKeyFile"="C:\pk-AKIAIOSFODNN7EXAMPLE.pem"  
    "ExpiresOn"=(Get-Date).AddDays(7)  
    "ActiveFrom"=$start  
}  
New-CFSignedUrl @params
```

Contoh 3: Membuat url yang ditandatangani ke sumber daya yang ditentukan menggunakan kebijakan khusus. Url akan berlaku mulai dalam 24 jam dan akan kedaluwarsa satu minggu kemudian. Akses ke sumber daya dibatasi pada rentang ip yang ditentukan.

```
$start = (Get-Date).AddHours(24)
$params = @{
    "ResourceUri"="https://cdn.example.com/index.html"
    "KeyPairId"="AKIAIOSFODNN7EXAMPLE"
    "PrivateKeyFile"="C:\pk-AKIAIOSFODNN7EXAMPLE.pem"
    "ExpiresOn"=(Get-Date).AddDays(7)
    "ActiveFrom"=$start
    "IpRange"="192.0.2.0/24"
}
New-CFSignedUrl @params
```

- Untuk detail API, lihat [CFSignedUrl Baru di](#) Referensi Alat AWS untuk PowerShell Cmdlet (V4).

CloudTrail contoh menggunakan Alat untuk PowerShell V4

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan CloudTrail.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

Find-CTEvent

Contoh kode berikut menunjukkan cara menggunakan Find-CTEvent.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan semua peristiwa yang telah terjadi selama tujuh hari terakhir. Cmdlet secara default secara otomatis membuat beberapa panggilan untuk mengirimkan semua peristiwa, keluar ketika layanan menunjukkan tidak ada data lebih lanjut yang tersedia.

```
Find-CTEvent
```

Contoh 2: Mengembalikan semua peristiwa yang telah terjadi selama tujuh hari terakhir menentukan wilayah yang bukan default shell saat ini.

```
Find-CTEvent -Region eu-central-1
```

Contoh 3: Mengembalikan semua peristiwa yang terkait dengan panggilan RunInstances API.

```
Find-CTEvent -LookupAttribute @{ AttributeKey="EventName";  
    AttributeValue="RunInstances" }
```

Contoh 4: Mengembalikan 5 peristiwa pertama yang tersedia.

```
Find-CTEvent -MaxResult 5
```

- Untuk detail API, lihat [LookupEvents](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CTTrail

Contoh kode berikut menunjukkan cara menggunakan `Get-CTTrail`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan pengaturan semua jejak yang terkait dengan wilayah saat ini untuk akun Anda.

```
Get-CTTrail
```

Contoh 2: Mengembalikan pengaturan untuk jalur yang ditentukan.

```
Get-CTTrail -TrailNameList trail1, trail2
```

Contoh 3: Mengembalikan pengaturan untuk jejak tertentu yang dibuat di wilayah selain default shell saat ini (dalam hal ini wilayah Frankfurt (eu-central-1)).

```
Get-CTTrail -TrailNameList trailABC, trailDEF -Region eu-central-1
```

- Untuk detail API, lihat [DescribeTrails](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CTTrailStatus

Contoh kode berikut menunjukkan cara menggunakan `Get-CTTrailStatus`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan informasi status untuk jejak dengan nama 'myExampleTrail'. Data yang dikembalikan mencakup informasi tentang kesalahan pengiriman, kesalahan Amazon SNS, dan Amazon S3, serta waktu mulai dan hentikan pencatatan untuk jejak. Contoh ini mengasumsikan jejak dibuat di wilayah yang sama dengan default shell saat ini.

```
Get-CTTrailStatus -Name myExampleTrail
```

Contoh 2: Mengembalikan informasi status untuk jejak yang dibuat di wilayah selain default shell saat ini (dalam hal ini, wilayah Frankfurt (eu-central-1)).

```
Get-CTTrailStatus -Name myExampleTrail -Region eu-central-1
```

- Untuk detail API, lihat [GetTrailStatus](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-CTTrail

Contoh kode berikut menunjukkan cara menggunakan `New-CTTrail`.

Alat untuk PowerShell V4

Contoh 1: Membuat jejak yang akan menggunakan bucket 'mycloudtrailbucket' untuk penyimpanan file log.

```
New-CTTrail -Name "awscloudtrail-example" -S3BucketName "amzn-s3-demo-bucket"
```

Contoh 2: Membuat jejak yang akan menggunakan bucket 'mycloudtrailbucket' untuk penyimpanan file log. Objek S3 yang mewakili log akan memiliki key prefix umum 'mylogs'. Saat log baru dikirim ke bucket, notifikasi akan dikirim ke topik SNS 'mlog-deliverytopic'. Contoh ini menggunakan percikan untuk memasok nilai parameter ke cmdlet.

```
$params = @{
```

```
Name="awscloudtrail-example"
S3BucketName="amzn-s3-demo-bucket"
S3KeyPrefix="mylogs"
SnsTopicName="mlog-deliverytopic"
}
New-CTTrail @params
```

- Untuk detail API, lihat [CreateTrail](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-CTTrail

Contoh kode berikut menunjukkan cara menggunakan `Remove-CTTrail`.

Alat untuk PowerShell V4

Contoh 1: Menghapus jejak yang ditentukan. Anda akan diminta konfirmasi sebelum perintah dijalankan. Untuk menekan konfirmasi, tambahkan parameter sakelar `-Force`.

```
Remove-CTTrail -Name "awscloudtrail-example"
```

- Untuk detail API, lihat [DeleteTrail](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Start-CTLogging

Contoh kode berikut menunjukkan cara menggunakan `Start-CTLogging`.

Alat untuk PowerShell V4

Contoh 1: Memulai perekaman panggilan AWS API dan pengiriman file log untuk jejak bernama 'myExampleTrail'. Contoh ini mengasumsikan jejak dibuat di wilayah yang sama dengan default shell saat ini.

```
Start-CTLogging -Name myExampleTrail
```

Contoh 2: Memulai perekaman panggilan AWS API dan pengiriman file log untuk jejak yang dibuat di wilayah selain default shell saat ini (dalam hal ini, wilayah Frankfurt (eu-central-1)).

```
Start-CTLogging -Name myExampleTrail -Region eu-central-1
```

- Untuk detail API, lihat [StartLogging](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Stop-CTLogging

Contoh kode berikut menunjukkan cara menggunakan `Stop-CTLogging`.

Alat untuk PowerShell V4

Contoh 1: Menangguhkan perekaman panggilan AWS API dan pengiriman file log untuk jejak bernama 'myExampleTrail'. Contoh ini mengasumsikan jejak dibuat di wilayah yang sama dengan default shell saat ini.

```
Stop-CTLogging -Name myExampleTrail
```

Contoh 2: Menangguhkan perekaman panggilan AWS API dan pengiriman file log untuk jejak yang dibuat di wilayah selain default shell saat ini (dalam hal ini, wilayah Frankfurt (eu-central-1)).

```
Stop-CTLogging -Name myExampleTrail -Region eu-central-1
```

- Untuk detail API, lihat [StopLogging](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-CTTrail

Contoh kode berikut menunjukkan cara menggunakan `Update-CTTrail`.

Alat untuk PowerShell V4

Contoh 1: Memperbarui jejak yang ditentukan sehingga peristiwa layanan global (seperti yang dari IAM) direkam dan mengubah awalan kunci umum dari file log yang akan maju menjadi 'globallogs'.

```
Update-CTTrail -Name "awscloudtrail-example" -IncludeGlobalServiceEvents $true -  
S3KeyPrefix "globallogs"
```

Contoh 2: Memperbarui jejak yang ditentukan sehingga pemberitahuan tentang pengiriman log baru dikirim ke topik SNS yang ditentukan.

```
Update-CTTrail -Name "awscloudtrail-example" -SnsTopicName "mlog-deliverytopic2"
```

Contoh 3: Memperbarui jejak yang ditentukan sehingga log dikirim ke ember yang berbeda.

```
Update-CTTrail -Name "awscloudtrail-example" -S3BucketName "otherlogs"
```

- Untuk detail API, lihat [UpdateTrail](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

CloudWatch contoh menggunakan Alat untuk PowerShell V4

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan CloudWatch.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

Get-CWDashboard

Contoh kode berikut menunjukkan cara menggunakan `Get-CWDashboard`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan arn badan dasbor yang ditentukan.

```
Get-CWDashboard -DashboardName Dashboard1
```

Output:

DashboardArn	DashboardBody
-----	-----
arn:aws:cloudwatch::123456789012:dashboard/Dashboard1	{...

- Untuk detail API, lihat [GetDashboard](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CWDashboardList

Contoh kode berikut menunjukkan cara menggunakan `Get-CWDashboardList`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan koleksi dasbor untuk akun Anda.

```
Get-CWDashboardList
```

Output:

DashboardArn	DashboardName	LastModified	Size
-----	-----	-----	----
arn:...	Dashboard1	7/6/2017 8:14:15 PM	252

Contoh 2: Mengembalikan koleksi dasbor untuk akun Anda yang namanya dimulai dengan awalan 'dev'.

```
Get-CWDashboardList -DashboardNamePrefix dev
```

- Untuk detail API, lihat [ListDashboards](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-CWDashboard

Contoh kode berikut menunjukkan cara menggunakan `Remove-CWDashboard`.

Alat untuk PowerShell V4

Contoh 1: Menghapus dasbor yang ditentukan, mempromosikan konfirmasi sebelum melanjutkan. Untuk melewati konfirmasi, tambahkan sakelar `-Force` ke perintah.

```
Remove-CWDashboard -DashboardName Dashboard1
```

- Untuk detail API, lihat [DeleteDashboards](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Write-CWDashboard

Contoh kode berikut menunjukkan cara menggunakan `Write-CWDashboard`.

Alat untuk PowerShell V4

Contoh 1: Membuat atau memperbarui dasbor bernama 'Dashboard1' untuk menyertakan dua widget metrik secara berdampingan.

```
$dashBody = @"
{
  "widgets":[
    {
      "type":"metric",
      "x":0,
      "y":0,
      "width":12,
      "height":6,
      "properties":{"
        "metrics":[
          [
            "AWS/EC2",
            "CPUUtilization",
            "InstanceId",
            "i-012345"
          ]
        ],
        "period":300,
        "stat":"Average",
        "region":"us-east-1",
        "title":"EC2 Instance CPU"
      }
    },
    {
      "type":"metric",
      "x":12,
      "y":0,
      "width":12,
      "height":6,
      "properties":{"
        "metrics":[
          [
            "AWS/S3",
            "BucketSizeBytes",
            "BucketName",
            "amzn-s3-demo-bucket"
          ]
        ]
      }
    }
  ],

```

```

        "period":86400,
        "stat":"Maximum",
        "region":"us-east-1",
        "title":"amzn-s3-demo-bucket bytes"
    }
}
]
}
"@

Write-CWDashboard -DashboardName Dashboard1 -DashboardBody $dashBody

```

Contoh 2: Membuat atau memperbarui dasbor, menyalurkan konten yang menjelaskan dasbor ke dalam cmdlet.

```

$dashBody = @"
{
...
}
"@

$dashBody | Write-CWDashboard -DashboardName Dashboard1

```

- Untuk detail API, lihat [PutDashboard](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Write-CWMetricData

Contoh kode berikut menunjukkan cara menggunakan `Write-CWMetricData`.

Alat untuk PowerShell V4

Contoh 1: Membuat `MetricDatum` objek baru, dan menuliskannya ke Amazon Web Services CloudWatch Metrics.

```

### Create a MetricDatum .NET object
$Metric = New-Object -TypeName Amazon.CloudWatch.Model.MetricDatum
$Metric.Timestamp = [DateTime]::UtcNow
$Metric.MetricName = 'CPU'
$Metric.Value = 50

### Write the metric data to the CloudWatch service

```

```
Write-CWMetricData -Namespace instance1 -MetricData $Metric
```

- Untuk detail API, lihat [PutMetricData](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

CodeCommit contoh menggunakan Alat untuk PowerShell V4

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan CodeCommit.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

Get-CCBranch

Contoh kode berikut menunjukkan cara menggunakan `Get-CCBranch`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapat informasi tentang cabang yang ditentukan untuk repositori yang ditentukan.

```
Get-CCBranch -RepositoryName MyDemoRepo -BranchName MyNewBranch
```

Output:

BranchName	CommitId
-----	-----
MyNewBranch	7763222d...561fc9c9

- Untuk detail API, lihat [GetBranch](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CCBranchList

Contoh kode berikut menunjukkan cara menggunakan `Get-CCBranchList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan daftar nama cabang untuk repositori yang ditentukan.

```
Get-CCBranchList -RepositoryName MyDemoRepo
```

Output:

```
master
MyNewBranch
```

- Untuk detail API, lihat [ListBranches](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CCRepository

Contoh kode berikut menunjukkan cara menggunakan `Get-CCRepository`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan informasi untuk repositori yang ditentukan.

```
Get-CCRepository -RepositoryName MyDemoRepo
```

Output:

```
AccountId      : 80398EXAMPLE
Arn            : arn:aws:codecommit:us-east-1:80398EXAMPLE:MyDemoRepo
CloneUrlHttp   : https://git-codecommit.us-east-1.amazonaws.com/v1/repos/
MyDemoRepo
CloneUrlSsh    : ssh://git-codecommit.us-east-1.amazonaws.com/v1/repos/
MyDemoRepo
CreationDate   : 9/8/2015 3:21:33 PM
DefaultBranch  :
LastModifiedDate : 9/8/2015 3:21:33 PM
RepositoryDescription : This is a repository for demonstration purposes.
RepositoryId    : c7d0d2b0-ce40-4303-b4c3-38529EXAMPLE
RepositoryName  : MyDemoRepo
```

- Untuk detail API, lihat [GetRepository](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CCRepositoryBatch

Contoh kode berikut menunjukkan cara menggunakan `Get-CCRepositoryBatch`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengonfirmasi repositori mana yang ditemukan dan tidak ditemukan.

```
Get-CCRepositoryBatch -RepositoryName MyDemoRepo, MyNewRepo, AMissingRepo
```

Output:

Repositories	RepositoriesNotFound
-----	-----
{MyDemoRepo, MyNewRepo}	{AMissingRepo}

- Untuk detail API, lihat [BatchGetRepositories](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CCRepositoryList

Contoh kode berikut menunjukkan cara menggunakan `Get-CCRepositoryList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan semua repositori dalam urutan menaik berdasarkan nama repositori.

```
Get-CCRepositoryList -Order Ascending -SortBy RepositoryName
```

Output:

RepositoryId	RepositoryName
-----	-----
c7d0d2b0-ce40-4303-b4c3-38529EXAMPLE	MyDemoRepo
05f30c66-e3e3-4f91-a0cd-1c84aEXAMPLE	MyNewRepo

- Untuk detail API, lihat [ListRepositories](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-CCBranch

Contoh kode berikut menunjukkan cara menggunakan `New-CCBranch`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat cabang baru dengan nama yang ditentukan untuk repositori tertentu dan ID komit yang ditentukan.

```
New-CCBranch -RepositoryName MyDemoRepo -BranchName MyNewBranch -CommitId
7763222d...561fc9c9
```

- Untuk detail API, lihat [CreateBranch](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-CCRepository

Contoh kode berikut menunjukkan cara menggunakan `New-CCRepository`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat repositori baru dengan nama yang ditentukan dan deskripsi yang ditentukan.

```
New-CCRepository -RepositoryName MyDemoRepo -RepositoryDescription "This is a
repository for demonstration purposes."
```

Output:

```
AccountId           : 80398EXAMPLE
Arn                 : arn:aws:codecommit:us-east-1:80398EXAMPLE:MyDemoRepo
CloneUrlHttp       : https://git-codecommit.us-east-1.amazonaws.com/v1/repos/
MyDemoRepo
CloneUrlSsh        : ssh://git-codecommit.us-east-1.amazonaws.com/v1/repos/
MyDemoRepo
CreationDate        : 9/18/2015 4:13:25 PM
DefaultBranch       :
LastModifiedDate    : 9/18/2015 4:13:25 PM
RepositoryDescription : This is a repository for demonstration purposes.
RepositoryId        : 43ef2443-3372-4b12-9e78-65c27EXAMPLE
RepositoryName       : MyDemoRepo
```

- Untuk detail API, lihat [CreateRepository](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-CCRepository

Contoh kode berikut menunjukkan cara menggunakan `Remove-CCRepository`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini secara paksa menghapus repositori yang ditentukan. Perintah akan meminta konfirmasi sebelum melanjutkan. Tambahkan parameter `-Force` untuk menghapus repositori tanpa prompt.

```
Remove-CCRepository -RepositoryName MyDemoRepo
```

Output:

```
43ef2443-3372-4b12-9e78-65c27EXAMPLE
```

- Untuk detail API, lihat [DeleteRepository](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-CCDefaultBranch

Contoh kode berikut menunjukkan cara menggunakan `Update-CCDefaultBranch`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengubah cabang default untuk repositori yang ditentukan ke cabang yang ditentukan.

```
Update-CCDefaultBranch -RepositoryName MyDemoRepo -DefaultBranchName MyNewBranch
```

- Untuk detail API, lihat [UpdateDefaultBranch](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-CCRepositoryDescription

Contoh kode berikut menunjukkan cara menggunakan `Update-CCRepositoryDescription`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengubah deskripsi untuk repositori yang ditentukan.

```
Update-CCRepositoryDescription -RepositoryName MyDemoRepo -RepositoryDescription  
"This is an updated description."
```

- Untuk detail API, lihat [UpdateRepositoryDescription](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-CCRepositoryName

Contoh kode berikut menunjukkan cara menggunakan `Update-CCRepositoryName`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengubah nama repositori yang ditentukan.

```
Update-CCRepositoryName -NewName MyDemoRepo2 -OldName MyDemoRepo
```

- Untuk detail API, lihat [UpdateRepositoryName](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

CodeDeploy contoh menggunakan Alat untuk PowerShell V4

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan CodeDeploy.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

Add-CDOnPremiseInstanceTag

Contoh kode berikut menunjukkan cara menggunakan `Add-CDOnPremiseInstanceTag`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menambahkan tag instans lokal dengan kunci dan nilai yang ditentukan untuk instans lokal yang ditentukan.

```
Add-CDOnPremiseInstanceTag -InstanceName AssetTag12010298EX -Tag @{"Key" = "Name";
"Value" = "CodeDeployDemo-OnPrem"}
```

- Untuk detail API, lihat [AddTagsToOnPremisesInstances](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CDApplication

Contoh kode berikut menunjukkan cara menggunakan `Get-CDApplication`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan informasi tentang aplikasi yang ditentukan.

```
Get-CDApplication -ApplicationName CodeDeployDemoApplication
```

Output:

ApplicationId LinkedToGitHub	ApplicationName	CreateTime
-----	-----	-----
e07fb938-091e-4f2f-8963-4d3e8EXAMPLE 9:49:48 PM False	CodeDeployDemoApplication	7/20/2015

- Untuk detail API, lihat [GetApplication](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CDApplicationBatch

Contoh kode berikut menunjukkan cara menggunakan `Get-CDApplicationBatch`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan informasi tentang aplikasi yang ditentukan.

```
Get-CDApplicationBatch -ApplicationName CodeDeployDemoApplication,
CodePipelineDemoApplication
```

Output:

ApplicationId LinkedToGitHub ----- -----	ApplicationName -----	CreateTime -----
e07fb938-091e-4f2f-8963-4d3e8EXAMPLE 9:49:48 PM False	CodeDeployDemoApplication	7/20/2015
1ecfd602-62f1-4038-8f0d-06688EXAMPLE 5:53:26 PM False	CodePipelineDemoApplication	8/13/2015

- Untuk detail API, lihat [BatchGetApplications](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CDApplicationList

Contoh kode berikut menunjukkan cara menggunakan `Get-CDApplicationList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan daftar aplikasi yang tersedia.

```
Get-CDApplicationList
```

Output:

```
CodeDeployDemoApplication
CodePipelineDemoApplication
```

- Untuk detail API, lihat [ListApplications](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CDApplicationRevision

Contoh kode berikut menunjukkan cara menggunakan `Get-CDApplicationRevision`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan informasi tentang revisi aplikasi yang ditentukan.

```
$revision = Get-CDApplicationRevision -ApplicationName CodeDeployDemoApplication
-S3Location_Bucket amzn-s3-demo-bucket -Revision_RevisionType S3 -
S3Location_Key 5xd27EX.zip -S3Location_BundleType zip -S3Location_ETag
4565c1ac97187f190c1a90265EXAMPLE
Write-Output ("Description = " + $revision.RevisionInfo.Description + ",
RegisterTime = " + $revision.RevisionInfo.RegisterTime)
```

Output:

```
Description = Application revision registered by Deployment ID: d-CX9CHN3EX,
RegisterTime = 07/20/2015 23:46:42
```

- Untuk detail API, lihat [GetApplicationRevision](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CDApplicationRevisionList

Contoh kode berikut menunjukkan cara menggunakan `Get-CDApplicationRevisionList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan informasi tentang revisi yang tersedia untuk aplikasi yang ditentukan.

```
ForEach ($revision in (Get-CDApplicationRevisionList -ApplicationName
CodeDeployDemoApplication -Deployed Ignore)) {
>> If ($revision.RevisionType -Eq "S3") {
>>   Write-Output ("Type = S3, Bucket = " + $revision.S3Location.Bucket
+ ", BundleType = " + $revision.S3Location.BundleType + ", ETag = " +
$revision.S3Location.ETag + ", Key = " + $revision.S3Location.Key)
>> }
>> If ($revision.RevisionType -Eq "GitHub") {
>>   Write-Output ("Type = GitHub, CommitId = " +
$revision.GitHubLocation.CommitId + ", Repository = " +
$revision.GitHubLocation.Repository)
>> }
>> }
>>
```

Output:

```
Type = S3, Bucket = MyBucket, BundleType = zip, ETag =
4565c1ac97187f190c1a90265EXAMPLE, Key = 5xd27EX.zip
Type = GitHub, CommitId = f48933c3...76405362, Repository = MyGitHubUser/
CodeDeployDemoRepo
```

- Untuk detail API, lihat [ListApplicationRevisions](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CDDeployment

Contoh kode berikut menunjukkan cara menggunakan `Get-CDDeployment`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan informasi ringkasan tentang penerapan yang ditentukan.

```
Get-CDDeployment -DeploymentId d-QZMRGSTEX
```

Output:

```
ApplicationName      : CodeDeployDemoApplication
CompleteTime         : 7/23/2015 11:26:04 PM
CreateTime           : 7/23/2015 11:24:43 PM
Creator              : user
DeploymentConfigName  : CodeDeployDefault.OneAtATime
DeploymentGroupName   : CodeDeployDemoFleet
DeploymentId          : d-QZMRGSTEX
DeploymentOverview    : Amazon.CodeDeploy.Model.DeploymentOverview
Description           :
ErrorInformation      :
IgnoreApplicationStopFailures : False
Revision             : Amazon.CodeDeploy.Model.RevisionLocation
StartTime            : 1/1/0001 12:00:00 AM
Status               : Succeeded
```

Contoh 2: Contoh ini mendapatkan informasi tentang status instance yang berpartisipasi dalam penerapan yang ditentukan.

```
(Get-CDDeployment -DeploymentId d-QZMRGSTEX).DeploymentOverview
```

Output:

```
Failed      : 0
InProgress  : 0
Pending     : 0
Skipped     : 0
Succeeded   : 3
```

Contoh 3: Contoh ini mendapatkan informasi tentang revisi aplikasi untuk penerapan yang ditentukan.

```
(Get-CDDeployment -DeploymentId d-QZMRGSTEX).Revision.S3Location
```

Output:

```
Bucket      : MyBucket
BundleType   : zip
ETag        : cfbb81b304ee5e27efc21adaed3EXAMPLE
Key         : clzfqEX
Version     :
```

- Untuk detail API, lihat [GetDeployment](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CDDeploymentBatch

Contoh kode berikut menunjukkan cara menggunakan `Get-CDDeploymentBatch`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan informasi tentang penerapan yang ditentukan.

```
Get-CDDeploymentBatch -DeploymentId d-QZMRGSTEX, d-RR0T5KTEX
```

Output:

```
ApplicationName      : CodeDeployDemoApplication
CompleteTime         : 7/23/2015 11:26:04 PM
CreateTime           : 7/23/2015 11:24:43 PM
Creator              : user
DeploymentConfigName  : CodeDeployDefault.OneAtATime
DeploymentGroupName   : CodeDeployDemoFleet
```

```

DeploymentId           : d-QZMRGSTEX
DeploymentOverview     : Amazon.CodeDeploy.Model.DeploymentOverview
Description           :
ErrorInformation       :
IgnoreApplicationStopFailures : False
Revision              : Amazon.CodeDeploy.Model.RevisionLocation
StartTime             : 1/1/0001 12:00:00 AM
Status                : Succeeded

ApplicationName        : CodePipelineDemoApplication
CompleteTime          : 7/23/2015 6:07:30 PM
CreateTime            : 7/23/2015 6:06:29 PM
Creator               : user
DeploymentConfigName   : CodeDeployDefault.OneAtATime
DeploymentGroupName    : CodePipelineDemoFleet
DeploymentId           : d-RR0T5KTEX
DeploymentOverview     : Amazon.CodeDeploy.Model.DeploymentOverview
Description           :
ErrorInformation       :
IgnoreApplicationStopFailures : False
Revision              : Amazon.CodeDeploy.Model.RevisionLocation
StartTime             : 1/1/0001 12:00:00 AM
Status                : Succeeded

```

- Untuk detail API, lihat [BatchGetDeployments](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CDDeploymentConfig

Contoh kode berikut menunjukkan cara menggunakan `Get-CDDeploymentConfig`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan informasi ringkasan tentang konfigurasi penerapan yang ditentukan.

```
Get-CDDeploymentConfig -DeploymentConfigName ThreeQuartersHealthy
```

Output:

CreateTime	DeploymentConfigId	DeploymentConfigName
MinimumHealthyHosts		

```

-----
-----
-----
10/3/2014 4:32:30 PM    518a3950-d034-46a1-9d2c-3c949EXAMPLE    ThreeQuartersHealthy
    Amazon.CodeDeploy.Model.MinimumHealthyHosts

```

Contoh 2: Contoh ini mendapatkan informasi tentang definisi konfigurasi penerapan yang ditentukan.

```
Write-Output ((Get-CDDeploymentConfig -DeploymentConfigName
    ThreeQuartersHealthy).MinimumHealthyHosts)
```

Output:

Type	Value
----	-----
FLEET_PERCENT	75

- Untuk detail API, lihat [GetDeploymentConfig](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CDDeploymentConfigList

Contoh kode berikut menunjukkan cara menggunakan `Get-CDDeploymentConfigList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan daftar konfigurasi penerapan yang tersedia.

```
Get-CDDeploymentConfigList
```

Output:

```

ThreeQuartersHealthy
CodeDeployDefault.OneAtATime
CodeDeployDefault.AllAtOnce
CodeDeployDefault.HalfAtATime

```

- Untuk detail API, lihat [ListDeploymentConfigs](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CDDeploymentGroup

Contoh kode berikut menunjukkan cara menggunakan `Get-CDDeploymentGroup`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan informasi tentang grup penyebaran yang ditentukan.

```
Get-CDDeploymentGroup -ApplicationName CodeDeployDemoApplication -
DeploymentGroupName CodeDeployDemoFleet
```

Output:

```
ApplicationName           : CodeDeployDemoApplication
AutoScalingGroups         : {}
DeploymentConfigName       : CodeDeployDefault.OneAtATime
DeploymentGroupId          : 7d7c098a-b444-4b27-96ef-22791EXAMPLE
DeploymentGroupName        : CodeDeployDemoFleet
Ec2TagFilters              : {Name}
OnPremisesInstanceTagFilters : {}
ServiceRoleArn             : arn:aws:iam::80398EXAMPLE:role/
CodeDeploySampleStack-4ph6EX-CodeDeployTrustRole-09MWP7XTL8EX
TargetRevision             : Amazon.CodeDeploy.Model.RevisionLocation
```

- Untuk detail API, lihat [GetDeploymentGroup](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CDDeploymentGroupList

Contoh kode berikut menunjukkan cara menggunakan `Get-CDDeploymentGroupList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan daftar grup penyebaran untuk aplikasi yang ditentukan.

```
Get-CDDeploymentGroupList -ApplicationName CodeDeployDemoApplication
```

Output:

```
ApplicationName      DeploymentGroups
NextToken
```

```

-----
-----
CodeDeployDemoApplication    {CodeDeployDemoFleet, CodeDeployProductionFleet}

```

- Untuk detail API, lihat [ListDeploymentGroups](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CDDeploymentInstance

Contoh kode berikut menunjukkan cara menggunakan `Get-CDDeploymentInstance`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan informasi tentang instance yang ditentukan untuk penerapan yang ditentukan.

```
Get-CDDeploymentInstance -DeploymentId d-QZMRGSTEX -InstanceId i-254e22EX
```

Output:

```

DeploymentId      : d-QZMRGSTEX
InstanceId        : arn:aws:ec2:us-east-1:80398EXAMPLE:instance/i-254e22EX
LastUpdatedAt     : 7/23/2015 11:25:24 PM
LifecycleEvents   : {ApplicationStop, DownloadBundle, BeforeInstall, Install...}
Status            : Succeeded

```

- Untuk detail API, lihat [GetDeploymentInstance](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CDDeploymentInstanceList

Contoh kode berikut menunjukkan cara menggunakan `Get-CDDeploymentInstanceList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan daftar instance IDs untuk penerapan yang ditentukan.

```
Get-CDDeploymentInstanceList -DeploymentId d-QZMRGSTEX
```

Output:

```
i-254e22EX  
i-274e22EX  
i-3b4e22EX
```

- Untuk detail API, lihat [ListDeploymentInstances](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CDDeploymentList

Contoh kode berikut menunjukkan cara menggunakan `Get-CDDeploymentList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan daftar penerapan IDs untuk grup aplikasi dan penyebaran yang ditentukan.

```
Get-CDDeploymentList -ApplicationName CodeDeployDemoApplication -DeploymentGroupName  
CodeDeployDemoFleet
```

Output:

```
d-QZMRGSTEX  
d-RR0T5KTEX
```

- Untuk detail API, lihat [ListDeployments](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CDOnPremiseInstance

Contoh kode berikut menunjukkan cara menggunakan `Get-CDOnPremiseInstance`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan informasi tentang instans lokal yang ditentukan.

```
Get-CDOnPremiseInstance -InstanceName AssetTag12010298EX
```

Output:

```
DeregisterTime : 1/1/0001 12:00:00 AM
```

```

IamUserArn      : arn:aws:iam::80398EXAMPLE:user/CodeDeployDemoUser
InstanceArn     : arn:aws:codedeploy:us-east-1:80398EXAMPLE:instance/
AssetTag12010298EX_rDH556dxEX
InstanceName    : AssetTag12010298EX
RegisterTime    : 4/3/2015 6:36:24 PM
Tags            : {Name}

```

- Untuk detail API, lihat [GetOnPremisesInstances](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CDOnPremiseInstanceBatch

Contoh kode berikut menunjukkan cara menggunakan `Get-CDOnPremiseInstanceBatch`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan informasi tentang instance lokal yang ditentukan.

```
Get-CDOnPremiseInstanceBatch -InstanceName AssetTag12010298EX, AssetTag12010298EX-2
```

Output:

```

DeregisterTime : 1/1/0001 12:00:00 AM
IamUserArn     : arn:aws:iam::80398EXAMPLE:user/CodeDeployFRWUser
InstanceArn    : arn:aws:codedeploy:us-east-1:80398EXAMPLE:instance/
AssetTag12010298EX-2_XmeSz18rEX
InstanceName   : AssetTag12010298EX-2
RegisterTime   : 4/3/2015 6:38:52 PM
Tags           : {Name}

DeregisterTime : 1/1/0001 12:00:00 AM
IamUserArn     : arn:aws:iam::80398EXAMPLE:user/CodeDeployDemoUser
InstanceArn    : arn:aws:codedeploy:us-east-1:80398EXAMPLE:instance/
AssetTag12010298EX_rDH556dxEX
InstanceName   : AssetTag12010298EX
RegisterTime   : 4/3/2015 6:36:24 PM
Tags           : {Name}

```

- Untuk detail API, lihat [BatchGetOnPremisesInstances](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CDOnPremiseInstanceList

Contoh kode berikut menunjukkan cara menggunakan `Get-CDOnPremiseInstanceList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan daftar nama instans lokal yang tersedia.

```
Get-CDOnPremiseInstanceList
```

Output:

```
AssetTag12010298EX  
AssetTag12010298EX-2
```

- Untuk detail API, lihat [ListOnPremisesInstances](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-CDApplication

Contoh kode berikut menunjukkan cara menggunakan `New-CDApplication`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat aplikasi baru dengan nama yang ditentukan.

```
New-CDApplication -ApplicationName MyNewApplication
```

Output:

```
f19e4b61-2231-4328-b0fd-e57f5EXAMPLE
```

- Untuk detail API, lihat [CreateApplication](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-CDDeployment

Contoh kode berikut menunjukkan cara menggunakan `New-CDDeployment`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat penyebaran baru untuk grup aplikasi dan penyebaran yang ditentukan dengan konfigurasi penerapan dan revisi aplikasi yang ditentukan.

```
New-CDDeployment -ApplicationName MyNewApplication -S3Location_Bucket amzn-s3-demo-bucket -S3Location_BundleType zip -DeploymentConfigName CodeDeployDefault.OneAtATime -DeploymentGroupName MyNewDeploymentGroup -IgnoreApplicationStopFailures $True -S3Location_Key aws-codedeploy_linux-master.zip -RevisionType S3
```

Output:

```
d-ZHR0G7UEX
```

Contoh 2: Contoh ini menunjukkan cara menentukan grup tag EC2 instance yang harus diidentifikasi oleh sebuah instance agar dapat disertakan dalam lingkungan pengganti untuk penerapan biru/hijau.

```
New-CDDeployment -ApplicationName MyNewApplication -S3Location_Bucket amzn-s3-demo-bucket -S3Location_BundleType zip -DeploymentConfigName CodeDeployDefault.OneAtATime -DeploymentGroupName MyNewDeploymentGroup -IgnoreApplicationStopFailures $True -S3Location_Key aws-codedeploy_linux-master.zip -RevisionType S3 -Ec2TagSetList @(@{Key="key1";Type="KEY_ONLY"},@{Key="Key2";Type="KEY_AND_VALUE";Value="Value2"}),@(@{Key="
```

Output:

```
d-ZHR0G7UEX
```

- Untuk detail API, lihat [CreateDeployment](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-CDDeploymentConfig

Contoh kode berikut menunjukkan cara menggunakan `New-CDDeploymentConfig`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat konfigurasi penerapan baru dengan nama dan perilaku yang ditentukan.

```
New-CDDeploymentConfig -DeploymentConfigName AtLeastTwoHealthyHosts -
MinimumHealthyHosts_Type HOST_COUNT -MinimumHealthyHosts_Value 2
```

Output:

```
0f3e8187-44ef-42da-aeed-b6823EXAMPLE
```

- Untuk detail API, lihat [CreateDeploymentConfig](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-CDDeploymentGroup

Contoh kode berikut menunjukkan cara menggunakan New-CDDeploymentGroup.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat grup penyebaran dengan nama yang ditentukan, grup Auto Scaling, konfigurasi penerapan, tag, dan peran layanan, untuk aplikasi yang ditentukan.

```
New-CDDeploymentGroup -ApplicationName MyNewApplication -AutoScalingGroup
CodeDeployDemo-ASG -DeploymentConfigName CodeDeployDefault.OneAtATime
-DeploymentGroupName MyNewDeploymentGroup -Ec2TagFilter @{Key="Name";
Type="KEY_AND_VALUE"; Value="CodeDeployDemo"} -ServiceRoleArn
arn:aws:iam::80398EXAMPLE:role/CodeDeployDemo
```

Output:

```
16bbf199-95fd-40fc-a909-0bbcfEXAMPLE
```

Contoh 2: Contoh ini menunjukkan cara menentukan grup tag EC2 instance yang harus diidentifikasi oleh sebuah instance agar dapat disertakan dalam lingkungan pengganti untuk penerapan biru/hijau.

```
New-CDDeploymentGroup -ApplicationName MyNewApplication -AutoScalingGroup
CodeDeployDemo-ASG -DeploymentConfigName CodeDeployDefault.OneAtATime
-DeploymentGroupName MyNewDeploymentGroup -Ec2TagFilter @{Key="Name";
Type="KEY_AND_VALUE"; Value="CodeDeployDemo"} -ServiceRoleArn
arn:aws:iam::80398EXAMPLE:role/CodeDeployDemo -Ec2TagSetList
@(@{Key="key1";Type="KEY_ONLY"},@{Key="Key2";Type="KEY_AND_VALUE";Value="Value2"}),@(@{Key="
```

Output:

```
16bbf199-95fd-40fc-a909-0bbcfEXAMPLE
```

- Untuk detail API, lihat [CreateDeploymentGroup](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Register-CDApplicationRevision

Contoh kode berikut menunjukkan cara menggunakan `Register-CDApplicationRevision`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendaftarkan revisi aplikasi dengan lokasi Amazon S3 yang ditentukan, untuk aplikasi yang ditentukan.

```
Register-CDApplicationRevision -ApplicationName MyNewApplication -S3Location_Bucket  
amzn-s3-demo-bucket -S3Location_BundleType zip -S3Location_Key aws-  
codedeploy_linux-master.zip -Revision_RevisionType S3
```

- Untuk detail API, lihat [RegisterApplicationRevision](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Register-CDOnPremiseInstance

Contoh kode berikut menunjukkan cara menggunakan `Register-CDOnPremiseInstance`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendaftarkan instance lokal dengan nama yang ditentukan dan pengguna IAM.

```
Register-CDOnPremiseInstance -IamUserArn arn:aws:iam::80398EXAMPLE:user/  
CodeDeployDemoUser -InstanceName AssetTag12010298EX
```

- Untuk detail API, lihat [RegisterOnPremisesInstance](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-CDApplication

Contoh kode berikut menunjukkan cara menggunakan `Remove-CDApplication`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus aplikasi dengan nama yang ditentukan. Perintah akan meminta konfirmasi sebelum melanjutkan. Tambahkan parameter `-Force` untuk menghapus aplikasi tanpa prompt.

```
Remove-CDApplication -ApplicationName MyNewApplication
```

- Untuk detail API, lihat [DeleteApplication](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-CDDeploymentConfig

Contoh kode berikut menunjukkan cara menggunakan `Remove-CDDeploymentConfig`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus konfigurasi penerapan dengan nama yang ditentukan. Perintah akan meminta konfirmasi sebelum melanjutkan. Tambahkan parameter `-Force` untuk menghapus konfigurasi penerapan tanpa prompt.

```
Remove-CDDeploymentConfig -DeploymentConfigName AtLeastTwoHealthyHosts
```

- Untuk detail API, lihat [DeleteDeploymentConfig](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-CDDeploymentGroup

Contoh kode berikut menunjukkan cara menggunakan `Remove-CDDeploymentGroup`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus grup penyebaran dengan nama yang ditentukan untuk aplikasi yang ditentukan. Perintah akan meminta konfirmasi sebelum melanjutkan. Tambahkan parameter `-Force` untuk menghapus grup penyebaran tanpa prompt.

```
Remove-CDDeploymentGroup -ApplicationName MyNewApplication -DeploymentGroupName
MyNewDeploymentGroup
```

- Untuk detail API, lihat [DeleteDeploymentGroup](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-CDOnPremiseInstanceTag

Contoh kode berikut menunjukkan cara menggunakan `Remove-CDOnPremiseInstanceTag`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus tag yang ditentukan untuk instance lokal dengan nama yang ditentukan. Perintah akan meminta konfirmasi sebelum melanjutkan. Tambahkan parameter `-Force` untuk menghapus tag tanpa prompt.

```
Remove-CDOnPremiseInstanceTag -InstanceName AssetTag12010298EX -Tag @{"Key" =
"Name"; "Value" = "CodeDeployDemo-OnPrem"}
```

- Untuk detail API, lihat [RemoveTagsFromOnPremisesInstances](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Stop-CDDeployment

Contoh kode berikut menunjukkan cara menggunakan `Stop-CDDeployment`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencoba menghentikan penerapan dengan ID penerapan yang ditentukan.

```
Stop-CDDeployment -DeploymentId d-LJQNREYEX
```

Output:

```
Status      StatusMessage
-----      -
Pending     Stopping Pending. Stopping to schedule commands in the deployment
instances
```

- Untuk detail API, lihat [StopDeployment](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Unregister-CDOnPremiseInstance

Contoh kode berikut menunjukkan cara menggunakan `Unregister-CDOnPremiseInstance`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membatalkan pendaftaran instans lokal dengan nama yang ditentukan.

```
Unregister-CDOnPremiseInstance -InstanceName AssetTag12010298EX
```

- Untuk detail API, lihat [DeregisterOnPremisesInstance](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-CDApplication

Contoh kode berikut menunjukkan cara menggunakan `Update-CDApplication`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengubah nama aplikasi yang ditentukan.

```
Update-CDApplication -ApplicationName MyNewApplication -NewApplicationName  
MyNewApplication-2
```

- Untuk detail API, lihat [UpdateApplication](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-CDDeploymentGroup

Contoh kode berikut menunjukkan cara menggunakan `Update-CDDeploymentGroup`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengubah nama grup penyebaran yang ditentukan untuk aplikasi yang ditentukan.

```
Update-CDDeploymentGroup -ApplicationName MyNewApplication -  
CurrentDeploymentGroupName MyNewDeploymentGroup -NewDeploymentGroupName  
MyNewDeploymentGroup-2
```

Contoh 2: Contoh ini menunjukkan cara menentukan grup tag EC2 instance yang harus diidentifikasi oleh sebuah instance agar dapat disertakan dalam lingkungan pengganti untuk penerapan biru/hijau.

```
Update-CDDeploymentGroup -ApplicationName MyNewApplication -
CurrentDeploymentGroupName MyNewDeploymentGroup -NewDeploymentGroupName
MyNewDeploymentGroup-2 -Ec2TagSetList
@(@{Key="key1";Type="KEY_ONLY"},@{Key="Key2";Type="KEY_AND_VALUE";Value="Value2"}),@(@{Key=
```

- Untuk detail API, lihat [UpdateDeploymentGroup](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

CodePipeline contoh menggunakan Alat untuk PowerShell V4

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan CodePipeline.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

Confirm-CPJob

Contoh kode berikut menunjukkan cara menggunakan Confirm-CPJob.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan status pekerjaan yang ditentukan.

```
Confirm-CPJob -JobId f570dc12-5ef3-44bc-945a-6e133EXAMPLE -Nonce 3
```

Output:

```
Value
-----
InProgress
```

- Untuk detail API, lihat [AcknowledgeJob](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Disable-CPStageTransition

Contoh kode berikut menunjukkan cara menggunakan `Disable-CPStageTransition`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menonaktifkan transisi masuk untuk tahap tertentu dalam pipeline yang ditentukan.

```
Disable-CPStageTransition -PipelineName CodePipelineDemo -Reason "Disabling temporarily." -StageName Beta -TransitionType Inbound
```

- Untuk detail API, lihat [DisableStageTransition](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Enable-CPStageTransition

Contoh kode berikut menunjukkan cara menggunakan `Enable-CPStageTransition`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memungkinkan transisi inbound untuk tahap tertentu dalam pipeline yang ditentukan.

```
Enable-CPStageTransition -PipelineName CodePipelineDemo -StageName Beta -TransitionType Inbound
```

- Untuk detail API, lihat [EnableStageTransition](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CPActionType

Contoh kode berikut menunjukkan cara menggunakan `Get-CPActionType`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan informasi tentang semua tindakan yang tersedia untuk pemilik yang ditentukan.

```
ForEach ($actionType in (Get-CPActionType -ActionOwnerFilter AWS)) {
    Write-Output ("For Category = " + $actionType.Id.Category + ", Owner = " +
    $actionType.Id.Owner + ", Provider = " + $actionType.Id.Provider + ", Version = " +
    $actionType.Id.Version + ":")
    Write-Output ("  ActionConfigurationProperties:")
    ForEach ($acp in $actionType.ActionConfigurationProperties) {
        Write-Output ("    For " + $acp.Name + ":")
        Write-Output ("      Description = " + $acp.Description)
        Write-Output ("      Key = " + $acp.Key)
        Write-Output ("      Queryable = " + $acp.Queryable)
        Write-Output ("      Required = " + $acp.Required)
        Write-Output ("      Secret = " + $acp.Secret)
    }
    Write-Output ("  InputArtifactDetails:")
    Write-Output ("    MaximumCount = " +
    $actionType.InputArtifactDetails.MaximumCount)
    Write-Output ("    MinimumCount = " +
    $actionType.InputArtifactDetails.MinimumCount)
    Write-Output ("  OutputArtifactDetails:")
    Write-Output ("    MaximumCount = " +
    $actionType.OutputArtifactDetails.MaximumCount)
    Write-Output ("    MinimumCount = " +
    $actionType.OutputArtifactDetails.MinimumCount)
    Write-Output ("  Settings:")
    Write-Output ("    EntityUrlTemplate = " + $actionType.Settings.EntityUrlTemplate)
    Write-Output ("    ExecutionUrlTemplate = " +
    $actionType.Settings.ExecutionUrlTemplate)
}
```

Output:

```
For Category = Deploy, Owner = AWS, Provider = ElasticBeanstalk, Version = 1:
ActionConfigurationProperties:
  For ApplicationName:
```

```

        Description = The AWS Elastic Beanstalk Application name
        Key = True
        Queryable = False
        Required = True
        Secret = False
    For EnvironmentName:
        Description = The AWS Elastic Beanstalk Environment name
        Key = True
        Queryable = False
        Required = True
        Secret = False
    InputArtifactDetails:
        MaximumCount = 1
        MinimumCount = 1
    OutputArtifactDetails:
        MaximumCount = 0
        MinimumCount = 0
    Settings:
        EntityUrlTemplate = https://console.aws.amazon.com/elasticbeanstalk/r/
application/{Config:ApplicationName}
        ExecutionUrlTemplate = https://console.aws.amazon.com/elasticbeanstalk/r/
application/{Config:ApplicationName}
    For Category = Deploy, Owner = AWS, Provider = CodeDeploy, Version = 1:
        ActionConfigurationProperties:
            For ApplicationName:
                Description = The AWS CodeDeploy Application name
                Key = True
                Queryable = False
                Required = True
                Secret = False
            For DeploymentGroupName:
                Description = The AWS CodeDeploy Deployment Group name
                Key = True
                Queryable = False
                Required = True
                Secret = False
        InputArtifactDetails:
            MaximumCount = 1
            MinimumCount = 1
        OutputArtifactDetails:
            MaximumCount = 0
            MinimumCount = 0
        Settings:

```

```
EntityUrlTemplate = https://console.aws.amazon.com/codedeploy/home?#/
applications/{Config:ApplicationName}/deployment-groups/{Config:DeploymentGroupName}
ExecutionUrlTemplate = https://console.aws.amazon.com/codedeploy/home?#/
deployments/{ExternalExecutionId}
```

- Untuk detail API, lihat [ListActionTypes](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CPActionableJobList

Contoh kode berikut menunjukkan cara menggunakan `Get-CPActionableJobList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan informasi tentang semua pekerjaan yang dapat ditindaklanjuti untuk kategori tindakan tertentu, pemilik, penyedia, versi, dan parameter kueri.

```
Get-CPActionableJobList -ActionTypeId_Category Build -ActionTypeId_Owner Custom
-ActionTypeId_Provider MyCustomProviderName -ActionTypeId_Version 1 -QueryParam
@{"ProjectName" = "MyProjectName"}
```

Output:

AccountId	Data	Id
----- -----	----	--
80398EXAMPLE f57a0EXAMPLE	Amazon.CodePipeline.Model.JobData 3	0de392f5-712d-4f41-ace3-

- Untuk detail API, lihat [PollForJobs](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CPJobDetail

Contoh kode berikut menunjukkan cara menggunakan `Get-CPJobDetail`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan informasi umum tentang pekerjaan yang ditentukan.

```
Get-CPJobDetail -JobId f570dc12-5ef3-44bc-945a-6e133EXAMPLE
```

Output:

AccountId	Data	Id
-----	----	--
80398EXAMPLE	Amazon.CodePipeline.Model.JobData	
f570dc12-5ef3-44bc-945a-6e133EXAMPLE		

Contoh 2: Contoh ini mendapatkan informasi rinci tentang pekerjaan yang ditentukan.

```
$jobDetails = Get-CPJobDetail -JobId f570dc12-5ef3-44bc-945a-6e133EXAMPLE
Write-Output ("For Job " + $jobDetails.Id + ":")
Write-Output (" AccountId = " + $jobDetails.AccountId)
$jobData = $jobDetails.Data
Write-Output (" Configuration:")
ForEach ($key in $jobData.ActionConfiguration.Keys) {
    $value = $jobData.ActionConfiguration.$key
    Write-Output ("    " + $key + " = " + $value)
}
Write-Output (" ActionTypeId:")
Write-Output ("    Category = " + $jobData.ActionTypeId.Category)
Write-Output ("    Owner = " + $jobData.ActionTypeId.Owner)
Write-Output ("    Provider = " + $jobData.ActionTypeId.Provider)
Write-Output ("    Version = " + $jobData.ActionTypeId.Version)
Write-Output (" ArtifactCredentials:")
Write-Output ("    AccessKeyId = " + $jobData.ArtifactCredentials.AccessKeyId)
Write-Output ("    SecretAccessKey = " +
    $jobData.ArtifactCredentials.SecretAccessKey)
Write-Output ("    SessionToken = " + $jobData.ArtifactCredentials.SessionToken)
Write-Output (" InputArtifacts:")
ForEach ($ia in $jobData.InputArtifacts) {
    Write-Output ("    " + $ia.Name)
}
Write-Output (" OutputArtifacts:")
ForEach ($oa in $jobData.OutputArtifacts) {
    Write-Output ("    " + $oa.Name)
}
Write-Output (" PipelineContext:")
$context = $jobData.PipelineContext
Write-Output ("    Name = " + $context.Action.Name)
Write-Output ("    PipelineName = " + $context.PipelineName)
Write-Output ("    Stage = " + $context.Stage.Name)
```

Output:

```

For Job f570dc12-5ef3-44bc-945a-6e133EXAMPLE:
  AccountId = 80398EXAMPLE
  Configuration:
    ActionTypeId:
      Category = Build
      Owner = Custom
      Provider = MyCustomProviderName
      Version = 1
    ArtifactCredentials:
      AccessKeyId = ASIAIEI3...IXI6YREX
      SecretAccessKey = cqAFDhEi...RdQyfa2u
      SessionToken = AQoDYXdz...5u+lsAU=
    InputArtifacts:
      MyApp
    OutputArtifacts:
      MyAppBuild
    PipelineContext:
      Name = Build
      PipelineName = CodePipelineDemo
      Stage = Build

```

- Untuk detail API, lihat [GetJobDetails](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CPPipeline

Contoh kode berikut menunjukkan cara menggunakan `Get-CPPipeline`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan informasi umum tentang pipeline yang ditentukan.

```
Get-CPPipeline -Name CodePipelineDemo -Version 1
```

Output:

```

ArtifactStore : Amazon.CodePipeline.Model.ArtifactStore
Name          : CodePipelineDemo
RoleArn       : arn:aws:iam::80398EXAMPLE:role/CodePipelineServiceRole
Stages        : {Source, Build, Beta, TestStage}
Version       : 1

```

Contoh 2: Contoh ini mendapatkan informasi rinci tentang pipeline yang ditentukan.

```

$pipeline = Get-CPPipeline -Name CodePipelineDemo
Write-Output ("Name = " + $pipeline.Name)
Write-Output ("RoleArn = " + $pipeline.RoleArn)
Write-Output ("Version = " + $pipeline.Version)
Write-Output ("ArtifactStore:")
Write-Output ("  Location = " + $pipeline.ArtifactStore.Location)
Write-Output ("  Type = " + $pipeline.ArtifactStore.Type.Value)
Write-Output ("Stages:")
ForEach ($stage in $pipeline.Stages) {
  Write-Output ("  Name = " + $stage.Name)
  Write-Output ("    Actions:")
  ForEach ($action in $stage.Actions) {
    Write-Output ("      Name = " + $action.Name)
    Write-Output ("      Category = " + $action.ActionTypeId.Category)
    Write-Output ("      Owner = " + $action.ActionTypeId.Owner)
    Write-Output ("      Provider = " + $action.ActionTypeId.Provider)
    Write-Output ("      Version = " + $action.ActionTypeId.Version)
    Write-Output ("      Configuration:")
    ForEach ($key in $action.Configuration.Keys) {
      $value = $action.Configuration.$key
      Write-Output ("        " + $key + " = " + $value)
    }
    Write-Output ("      InputArtifacts:")
    ForEach ($ia in $action.InputArtifacts) {
      Write-Output ("        " + $ia.Name)
    }
    ForEach ($oa in $action.OutputArtifacts) {
      Write-Output ("        " + $oa.Name)
    }
    Write-Output ("      RunOrder = " + $action.RunOrder)
  }
}

```

Output:

```

Name = CodePipelineDemo
RoleArn = arn:aws:iam::80398EXAMPLE:role/CodePipelineServiceRole
Version = 3
ArtifactStore:
  Location = MyBucketName
  Type = S3
Stages:
  Name = Source

```

Actions:

```
Name = Source
Category = Source
Owner = ThirdParty
Provider = GitHub
Version = 1
Configuration:
  Branch = master
  OAuthToken = ****
  Owner = my-user-name
  Repo = MyRepoName
InputArtifacts:
  MyApp
RunOrder = 1
```

```
Name = Build
```

Actions:

```
Name = Build
Category = Build
Owner = Custom
Provider = MyCustomProviderName
Version = 1
Configuration:
  ProjectName = MyProjectName
InputArtifacts:
  MyApp
  MyAppBuild
RunOrder = 1
```

```
Name = Beta
```

Actions:

```
Name = CodePipelineDemoFleet
Category = Deploy
Owner = AWS
Provider = CodeDeploy
Version = 1
Configuration:
  ApplicationName = CodePipelineDemoApplication
  DeploymentGroupName = CodePipelineDemoFleet
InputArtifacts:
  MyAppBuild
RunOrder = 1
```

```
Name = TestStage
```

Actions:

```
Name = MyJenkinsTestAction
Category = Test
```

```

Owner = Custom
Provider = MyCustomTestProvider
Version = 1
Configuration:
  ProjectName = MyJenkinsProjectName
InputArtifacts:
  MyAppBuild
RunOrder = 1

```

- Untuk detail API, lihat [GetPipeline](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CPPipelineList

Contoh kode berikut menunjukkan cara menggunakan `Get-CPPipelineList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan daftar pipeline yang tersedia.

```
Get-CPPipelineList
```

Output:

Created	Name	Updated	Version
-----	----	-----	-----
8/13/2015 10:17:54 PM	CodePipelineDemo	8/13/2015 10:17:54 PM	3
7/8/2015 2:41:53 AM	MyFirstPipeline	7/22/2015 9:06:37 PM	7

- Untuk detail API, lihat [ListPipelines](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CPPipelineState

Contoh kode berikut menunjukkan cara menggunakan `Get-CPPipelineState`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan informasi umum tentang tahapan untuk pipeline yang ditentukan.

```
Get-CPPipelineState -Name CodePipelineDemo
```

Output:

```

Created           : 8/13/2015 10:17:54 PM
PipelineName      : CodePipelineDemo
PipelineVersion   : 1
StageStates       : {Source, Build, Beta, TestStage}
Updated           : 8/13/2015 10:17:54 PM

```

Contoh 2: Contoh ini mendapatkan informasi rinci tentang keadaan pipa yang ditentukan.

```

ForEach ($stageState in (Get-CCPipelineState -Name $arg).StageStates) {
    Write-Output ("For " + $stageState.StageName + ":")
    Write-Output ("  InboundTransitionState:")
    Write-Output ("    DisabledReason = " +
$stageState.InboundTransitionState.DisabledReason)
    Write-Output ("    Enabled = " + $stageState.InboundTransitionState.Enabled)
    Write-Output ("    LastChangedAt = " +
$stageState.InboundTransitionState.LastChangedAt)
    Write-Output ("    LastChangedBy = " +
$stageState.InboundTransitionState.LastChangedBy)
    Write-Output ("  ActionStates:")
    ForEach ($actionState in $stageState.ActionStates) {
        Write-Output ("    For " + $actionState.ActionName + ":")
    Write-Output ("      CurrentRevision:")
        Write-Output ("        Created = " + $actionState.CurrentRevision.Created)
    Write-Output ("        RevisionChangeId = " +
$actionState.CurrentRevision.RevisionChangeId)
    Write-Output ("        RevisionId = " + $actionState.CurrentRevision.RevisionId)
    Write-Output ("        EntityUrl = " + $actionState.EntityUrl)
    Write-Output ("      LatestExecution:")
        Write-Output ("        ErrorDetails:")
        Write-Output ("          Code = " +
$actionState.LatestExecution.ErrorDetails.Code)
    Write-Output ("          Message = " +
$actionState.LatestExecution.ErrorDetails.Message)
    Write-Output ("        ExternalExecutionId = " +
$actionState.LatestExecution.ExternalExecutionId)
    Write-Output ("        ExternalExecutionUrl = " +
$actionState.LatestExecution.ExternalExecutionUrl)
    Write-Output ("        LastStatusChange = " +
$actionState.LatestExecution.LastStatusChange)
    Write-Output ("        PercentComplete = " +
$actionState.LatestExecution.PercentComplete)

```

```

Write-Output ("          Status = " + $actionState.LatestExecution.Status)
Write-Output ("          Summary = " + $actionState.LatestExecution.Summary)
Write-Output ("          RevisionUrl = " + $actionState.RevisionUrl)
    }
}

```

Output:

```

For Source:
  InboundTransitionState:
    DisabledReason =
    Enabled =
    LastChangedAt =
    LastChangedBy =
  ActionStates:
    For Source:
      CurrentRevision:
        Created =
        RevisionChangeId =
        RevisionId =
      EntityUrl = https://github.com/my-user-name/MyRepoName/tree/master
      LatestExecution:
        ErrorDetails:
          Code =
          Message =
          ExternalExecutionId =
          ExternalExecutionUrl =
          LastStatusChange = 07/20/2015 23:28:45
          PercentComplete = 0
          Status = Succeeded
          Summary =
        RevisionUrl =
    For Build:
      InboundTransitionState:
        DisabledReason =
        Enabled = True
        LastChangedAt = 01/01/0001 00:00:00
        LastChangedBy =
      ActionStates:
        For Build:
          CurrentRevision:
            Created =
            RevisionChangeId =

```

```

        RevisionId =
        EntityUrl = http://54.174.131.1EX/job/MyJenkinsDemo
        LatestExecution:
            ErrorDetails:
                Code = TimeoutError
                Message = The action failed because a job worker exceeded its time limit.
If this is a custom action, make sure that the job worker is configured correctly.
            ExternalExecutionId =
            ExternalExecutionUrl =
            LastStatusChange = 07/21/2015 00:29:29
            PercentComplete = 0
            Status = Failed
            Summary =
        RevisionUrl =
For Beta:
    InboundTransitionState:
        DisabledReason =
        Enabled = True
        LastChangedAt = 01/01/0001 00:00:00
        LastChangedBy =
    ActionStates:
        For CodePipelineDemoFleet:
            CurrentRevision:
                Created =
                RevisionChangeId =
                RevisionId =
                EntityUrl = https://console.aws.amazon.com/codedeploy/home?#/applications/
CodePipelineDemoApplication/deployment-groups/CodePipelineDemoFleet
            LatestExecution:
                ErrorDetails:
                    Code =
                    Message =
                    ExternalExecutionId = d-D5LTCZXEX
                    ExternalExecutionUrl = https://console.aws.amazon.com/codedeploy/home?#/
deployments/d-D5LTCZXEX
                    LastStatusChange = 07/08/2015 22:07:42
                    PercentComplete = 0
                    Status = Succeeded
                    Summary = Deployment Succeeded
                RevisionUrl =
For TestStage:
    InboundTransitionState:
        DisabledReason =
        Enabled = True

```

```

    LastChangedAt = 01/01/0001 00:00:00
    LastChangedBy =
    ActionStates:
    For MyJenkinsTestAction25:
    CurrentRevision:
    Created =
    RevisionChangeId =
    RevisionId =
    EntityUrl = http://54.174.131.1EX/job/MyJenkinsDemo
    LatestExecution:
    ErrorDetails:
    Code =
    Message =
    ExternalExecutionId = 5
    ExternalExecutionUrl = http://54.174.131.1EX/job/MyJenkinsDemo/5
    LastStatusChange = 07/08/2015 22:09:03
    PercentComplete = 0
    Status = Succeeded
    Summary = Finished
    RevisionUrl =

```

- Untuk detail API, lihat [GetPipelineState](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-CPCustomActionType

Contoh kode berikut menunjukkan cara menggunakan `New-CPCustomActionType`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat tindakan kustom baru dengan properti yang ditentukan.

```

New-CPCustomActionType -Category Build -ConfigurationProperty @{"Description"
= "The name of the build project must be provided when this action is added
to the pipeline."; "Key" = $True; "Name" = "ProjectName"; "Queryable"
= $False; "Required" = $True; "Secret" = $False; "Type" = "String"} -
Settings_EntityUrlTemplate "https://my-build-instance/job/{Config:ProjectName}/"
-Settings_ExecutionUrlTemplate "https://my-build-instance/job/mybuildjob/
lastSuccessfulBuild{ExternalExecutionId}/" -InputArtifactDetails_MaximumCount
1 -OutputArtifactDetails_MaximumCount 1 -InputArtifactDetails_MinimumCount 0 -
OutputArtifactDetails_MinimumCount 0 -Provider "MyBuildProviderName" -Version 1

```

Output:

```

ActionConfigurationProperties : {ProjectName}
Id                           : Amazon.CodePipeline.Model.ActionTypeId
InputArtifactDetails         : Amazon.CodePipeline.Model.ArtifactDetails
OutputArtifactDetails        : Amazon.CodePipeline.Model.ArtifactDetails
Settings                     : Amazon.CodePipeline.Model.ActionTypeSettings

```

- Untuk detail API, lihat [CreateCustomActionType](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-CPPipeline

Contoh kode berikut menunjukkan cara menggunakan `New-CPPipeline`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat pipeline baru dengan pengaturan yang ditentukan.

```

$pipeline = New-Object Amazon.CodePipeline.Model.PipelineDeclaration

$sourceStageAction = New-Object Amazon.CodePipeline.Model.ActionDeclaration
$deployStageAction = New-Object Amazon.CodePipeline.Model.ActionDeclaration

$sourceStageAction.OutputArtifact = New-Object
    Amazon.CodePipeline.Model.OutputArtifact
$sourceStageAction.OutputArtifact.Name = "MyApp"

$sourceStageAction.ActionTypeId = @{"Category" = "Source"; "Owner" = "AWS";
    "Provider" = "S3"; "Version" = 1}
$sourceStageAction.Configuration.Add("S3Bucket", "amzn-s3-demo-bucket")
$sourceStageAction.Configuration.Add("S3ObjectKey", "my-object-key-name.zip")
$sourceStageAction.OutputArtifacts.Add($sourceStageAction.OutputArtifact)
$sourceStageAction.Name = "Source"

$deployStageAction.InputArtifact = New-Object Amazon.CodePipeline.Model.InputArtifact
$deployStageAction.InputArtifact.Name = "MyApp"

$deployStageAction.ActionTypeId = @{"Category" = "Deploy"; "Owner" = "AWS";
    "Provider" = "CodeDeploy"; "Version" = 1}
$deployStageAction.Configuration.Add("ApplicationName",
    "CodePipelineDemoApplication")
$deployStageAction.Configuration.Add("DeploymentGroupName", "CodePipelineDemoFleet")
$deployStageAction.InputArtifacts.Add($deployStageAction.InputArtifact)

```

```

$deployStageAction.Name = "CodePipelineDemoFleet"

$sourceStage = New-Object Amazon.CodePipeline.Model.StageDeclaration
$deployStage = New-Object Amazon.CodePipeline.Model.StageDeclaration

$sourceStage.Name = "Source"
$deployStage.Name = "Beta"

$sourceStage.Actions.Add($sourceStageAction)
$deployStage.Actions.Add($deployStageAction)

$pipeline.ArtifactStore = @{"Location" = "amzn-s3-demo-bucket"; "Type" = "S3"}
$pipeline.Name = "CodePipelineDemo"
$pipeline.RoleArn = "arn:aws:iam::80398EXAMPLE:role/CodePipelineServiceRole"
$pipeline.Stages.Add($sourceStage)
$pipeline.Stages.Add($deployStage)
$pipeline.Version = 1

New-CPPipeline -Pipeline $pipeline

```

Output:

```

ArtifactStore : Amazon.CodePipeline.Model.ArtifactStore
Name           : CodePipelineDemo
RoleArn        : arn:aws:iam::80398EXAMPLE:role/CodePipelineServiceRole
Stages         : {Source, Beta}
Version        : 1

```

- Untuk detail API, lihat [CreatePipeline](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-CPCustomActionType

Contoh kode berikut menunjukkan cara menggunakan `Remove-CPCustomActionType`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus tindakan kustom yang ditentukan. Perintah akan meminta konfirmasi sebelum melanjutkan. Tambahkan parameter `-Force` untuk menghapus tindakan kustom tanpa prompt.

```
Remove-CPCustomActionType -Category Build -Provider MyBuildProviderName -Version 1
```

- Untuk detail API, lihat [DeleteCustomActionType](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-CPPipeline

Contoh kode berikut menunjukkan cara menggunakan `Remove-CPPipeline`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus pipeline yang ditentukan. Perintah akan meminta konfirmasi sebelum melanjutkan. Tambahkan parameter `-Force` untuk menghapus pipeline tanpa prompt.

```
Remove-CPPipeline -Name CodePipelineDemo
```

- Untuk detail API, lihat [DeletePipeline](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Start-CPPipelineExecution

Contoh kode berikut menunjukkan cara menggunakan `Start-CPPipelineExecution`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mulai menjalankan pipeline yang ditentukan.

```
Start-CPPipelineExecution -Name CodePipelineDemo
```

- Untuk detail API, lihat [StartPipelineExecution](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-CPPipeline

Contoh kode berikut menunjukkan cara menggunakan `Update-CPPipeline`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memperbarui pipeline yang ada yang ditentukan dengan pengaturan yang ditentukan.

```
$pipeline = New-Object Amazon.CodePipeline.Model.PipelineDeclaration
$sourceStageAction = New-Object Amazon.CodePipeline.Model.ActionDeclaration
```

```

$deployStageAction = New-Object Amazon.CodePipeline.Model.ActionDeclaration

$sourceStageActionOutputArtifact = New-Object
    Amazon.CodePipeline.Model.OutputArtifact
$sourceStageActionOutputArtifact.Name = "MyApp"

$sourceStageAction.ActionTypeId = @{"Category" = "Source"; "Owner" = "AWS";
    "Provider" = "S3"; "Version" = 1}
$sourceStageAction.Configuration.Add("S3Bucket", "amzn-s3-demo-bucket")
$sourceStageAction.Configuration.Add("S3ObjectKey", "my-object-key-name.zip")
$sourceStageAction.OutputArtifacts.Add($sourceStageActionOutputArtifact)
$sourceStageAction.Name = "Source"

$deployStageActionInputArtifact = New-Object Amazon.CodePipeline.Model.InputArtifact
$deployStageActionInputArtifact.Name = "MyApp"

$deployStageAction.ActionTypeId = @{"Category" = "Deploy"; "Owner" = "AWS";
    "Provider" = "CodeDeploy"; "Version" = 1}
$deployStageAction.Configuration.Add("ApplicationName",
    "CodePipelineDemoApplication")
$deployStageAction.Configuration.Add("DeploymentGroupName", "CodePipelineDemoFleet")
$deployStageAction.InputArtifacts.Add($deployStageActionInputArtifact)
$deployStageAction.Name = "CodePipelineDemoFleet"

$sourceStage = New-Object Amazon.CodePipeline.Model.StageDeclaration
$deployStage = New-Object Amazon.CodePipeline.Model.StageDeclaration

$sourceStage.Name = "MyInputFiles"
$deployStage.Name = "MyTestDeployment"

$sourceStage.Actions.Add($sourceStageAction)
$deployStage.Actions.Add($deployStageAction)

$pipeline.ArtifactStore = @{"Location" = "amzn-s3-demo-bucket"; "Type" = "S3"}
$pipeline.Name = "CodePipelineDemo"
$pipeline.RoleArn = "arn:aws:iam::80398EXAMPLE:role/CodePipelineServiceRole"
$pipeline.Stages.Add($sourceStage)
$pipeline.Stages.Add($deployStage)
$pipeline.Version = 1

Update-CPPipeline -Pipeline $pipeline

```

Output:

```
ArtifactStore : Amazon.CodePipeline.Model.ArtifactStore
Name          : CodePipelineDemo
RoleArn       : arn:aws:iam::80398EXAMPLE:role/CodePipelineServiceRole
Stages        : {InputFiles, TestDeployment}
Version       : 2
```

- Untuk detail API, lihat [UpdatePipeline](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Contoh Identitas Amazon Cognito menggunakan Alat untuk V4 PowerShell

Contoh kode berikut menunjukkan cara melakukan tindakan dan menerapkan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan Identitas Amazon Cognito.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

Get-CGIIIdentityPool

Contoh kode berikut menunjukkan cara menggunakan `Get-CGIIIdentityPool`.

Alat untuk PowerShell V4

Contoh 1: Mengambil informasi tentang Identity Pool tertentu dengan idnya.

```
Get-CGIIIdentityPool -IdentityPoolId us-east-1:0de2af35-2988-4d0b-b22d-EXAMPLEGUID1
```

Output:

```

LoggedAt           : 8/12/2015 4:29:40 PM
AllowUnauthenticatedIdentities : True
DeveloperProviderName      :
IdentityPoolId           : us-east-1:0de2af35-2988-4d0b-b22d-EXAMPLEGUID1
IdentityPoolName          : CommonTests1
OpenIdConnectProviderARNs : {}
SupportedLoginProviders    : {}
ResponseMetadata          : Amazon.Runtime.ResponseMetadata
ContentLength           : 142
HttpStatusCode           : OK

```

- Untuk detail API, lihat [DescribeIdentityPool](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CGIIIdentityPoolList

Contoh kode berikut menunjukkan cara menggunakan `Get-CGIIIdentityPoolList`.

Alat untuk PowerShell V4

Contoh 1: Mengambil daftar Identity Pools yang ada.

```
Get-CGIIIdentityPoolList
```

Output:

IdentityPoolId	IdentityPoolName
-----	-----
us-east-1:0de2af35-2988-4d0b-b22d-EXAMPLEGUID1	CommonTests1
us-east-1:118d242d-204e-4b88-b803-EXAMPLEGUID2	Tests2
us-east-1:15d49393-ab16-431a-b26e-EXAMPLEGUID3	CommonTests13

- Untuk detail API, lihat [ListIdentityPools](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CGIIIdentityPoolRole

Contoh kode berikut menunjukkan cara menggunakan `Get-CGIIIdentityPoolRole`.

Alat untuk PowerShell V4

Contoh 1: Mendapatkan informasi tentang peran untuk Identity Pool tertentu.

```
Get-CGIIIdentityPoolRole -IdentityPoolId us-east-1:0de2af35-2988-4d0b-b22d-EXAMPLEGUID1
```

Output:

```

LoggedAt           : 8/12/2015 4:33:51 PM
IdentityPoolId     : us-east-1:0de2af35-2988-4d0b-b22d-EXAMPLEGUID1
Roles              : {[unauthenticated, arn:aws:iam::123456789012:role/CommonTests1Role]}
ResponseMetadata   : Amazon.Runtime.ResponseMetadata
ContentLength      : 165
HttpStatusCode     : OK

```

- Untuk detail API, lihat [GetIdentityPoolRoles](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-CGIIIdentityPool

Contoh kode berikut menunjukkan cara menggunakan `New-CGIIIdentityPool`.

Alat untuk PowerShell V4

Contoh 1: Membuat Identity Pool baru yang memungkinkan identitas yang tidak diautentikasi.

```
New-CGIIIdentityPool -AllowUnauthenticatedIdentities $true -IdentityPoolName CommonTests13
```

Output:

```

LoggedAt           : 8/12/2015 4:56:07 PM
AllowUnauthenticatedIdentities : True
DeveloperProviderName       :
IdentityPoolId           : us-east-1:15d49393-ab16-431a-b26e-EXAMPLEGUID3
IdentityPoolName         : CommonTests13
OpenIdConnectProviderARNs  : {}
SupportedLoginProviders    : {}
ResponseMetadata         : Amazon.Runtime.ResponseMetadata
ContentLength          : 136
HttpStatusCode          : OK

```

- Untuk detail API, lihat [CreateIdentityPool](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-CGIIIdentityPool

Contoh kode berikut menunjukkan cara menggunakan `Remove-CGIIIdentityPool`.

Alat untuk PowerShell V4

Contoh 1: Menghapus Identity Pool tertentu.

```
Remove-CGIIIdentityPool -IdentityPoolId us-east-1:0de2af35-2988-4d0b-b22d-EXAMPLEGUID1
```

- Untuk detail API, lihat [DeleteIdentityPool](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Set-CGIIIdentityPoolRole

Contoh kode berikut menunjukkan cara menggunakan `Set-CGIIIdentityPoolRole`.

Alat untuk PowerShell V4

Contoh 1: Mengkonfigurasi Identity Pool tertentu untuk memiliki peran IAM yang tidak diautentikasi.

```
Set-CGIIIdentityPoolRole -IdentityPoolId us-east-1:0de2af35-2988-4d0b-b22d-EXAMPLEGUID1 -Role @{ "unauthenticated" = "arn:aws:iam::123456789012:role/CommonTests1Role" }
```

- Untuk detail API, lihat [SetIdentityPoolRoles](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-CGIIIdentityPool

Contoh kode berikut menunjukkan cara menggunakan `Update-CGIIIdentityPool`.

Alat untuk PowerShell V4

Contoh 1: Memperbarui beberapa properti Identity Pool, dalam hal ini nama Identity Pool.

```
Update-CGIIIdentityPool -IdentityPoolId us-east-1:0de2af35-2988-4d0b-b22d-EXAMPLEGUID1 -IdentityPoolName NewPoolName
```

Output:

```

LoggedAt                : 8/12/2015 4:53:33 PM
AllowUnauthenticatedIdentities : False
DeveloperProviderName   :
IdentityPoolId          : us-east-1:0de2af35-2988-4d0b-b22d-EXAMPLEGUID1
IdentityPoolName        : NewPoolName
OpenIdConnectProviderARNs : {}
SupportedLoginProviders  : {}
ResponseMetadata        : Amazon.Runtime.ResponseMetadata
ContentLength            : 135
HttpStatusCode           : OK

```

- Untuk detail API, lihat [UpdateIdentityPool](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

AWS Config contoh menggunakan Alat untuk PowerShell V4

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan AWS Config.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

Add-CFGResourceTag

Contoh kode berikut menunjukkan cara menggunakan Add-CFGResourceTag.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengaitkan tag tertentu ke ARN sumber daya, yaitu config-rule/config-rule-16iyn0 dalam kasus ini.

```
Add-CFGResourceTag -ResourceArn arn:aws:config:eu-west-1:123456789012:config-rule/
config-rule-16iyn0 -Tag @{Key="Release";Value="Beta"}
```

- Untuk detail API, lihat [TagResource](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CFGAggregateComplianceByConfigRuleList

Contoh kode berikut menunjukkan cara menggunakan `Get-CFGAggregateComplianceByConfigRuleList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil detail dari pemfilteran ConfigurationAggregator 'kaju' untuk aturan konfigurasi yang diberikan dan memperluas/mengembalikan 'Kepatuhan' aturan.

```
Get-CFGAggregateComplianceByConfigRuleList -ConfigurationAggregatorName kaju
-Filters_ConfigRuleName ALB_HTTP_TO_HTTPS_REDIRECTION_CHECK | Select-Object -
ExpandProperty Compliance
```

Output:

ComplianceContributorCount	ComplianceType
-----	-----
Amazon.ConfigService.Model.ComplianceContributorCount	NON_COMPLIANT

Contoh 2: Contoh ini mengambil detail dari yang diberikan ConfigurationAggregator, menyaringnya untuk akun yang diberikan untuk semua wilayah yang tercakup dalam agregator dan selanjutnya menyetel kembali kepatuhan untuk semua aturan.

```
Get-CFGAggregateComplianceByConfigRuleList -ConfigurationAggregatorName
kaju -Filters_AccountId 123456789012 | Select-Object ConfigRuleName,
@{N="Compliance";E={$_.Compliance.ComplianceType}}
```

Output:

ConfigRuleName	Compliance
-----	-----
ALB_HTTP_TO_HTTPS_REDIRECTION_CHECK	NON_COMPLIANT
ec2-instance-no-public-ip	NON_COMPLIANT
desired-instance-type	NON_COMPLIANT

- Untuk detail API, lihat [DescribeAggregateComplianceByConfigRules](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CFGAggregateComplianceDetailsByConfigRule

Contoh kode berikut menunjukkan cara menggunakan `Get-CFGAggregateComplianceDetailsByConfigRule`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan hasil evaluasi yang memilih output dengan resource-id dan resource-type untuk aturan AWS Config " yang berada dalam status 'COMPLIANTdesired-instance-type' untuk akun, agregator, wilayah, dan aturan konfigurasi yang diberikan

```
Get-CFGAggregateComplianceDetailsByConfigRule -AccountId 123456789012 -
AwsRegion eu-west-1 -ComplianceType COMPLIANT -ConfigRuleName desired-
instance-type -ConfigurationAggregatorName raju | Select-Object -
ExpandProperty EvaluationResultIdentifier | Select-Object -ExpandProperty
EvaluationResultQualifier
```

Output:

ConfigRuleName	ResourceId	ResourceType
-----	-----	-----
desired-instance-type	i-0f1bf2f34c5678d12	AWS::EC2::Instance
desired-instance-type	i-0fd12dd3456789123	AWS::EC2::Instance

- Untuk detail API, lihat [GetAggregateComplianceDetailsByConfigRule](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CFGAggregateConfigRuleComplianceSummary

Contoh kode berikut menunjukkan cara menggunakan `Get-CFGAggregateConfigRuleComplianceSummary`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan jumlah aturan yang tidak sesuai untuk agregator yang diberikan.

```
(Get-CFGAggregateConfigRuleComplianceSummary -ConfigurationAggregatorName
raju).AggregateComplianceCounts.ComplianceSummary.NonCompliantResourceCount
```

Output:

```
CapExceeded CappedCount
-----
False      5
```

- Untuk detail API, lihat [GetAggregateConfigRuleComplianceSummary](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CFGAggregateDiscoveredResourceCount

Contoh kode berikut menunjukkan cara menggunakan `Get-CFGAggregateDiscoveredResourceCount`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan jumlah sumber daya untuk agregator yang diberikan yang difilter untuk wilayah us-east-1.

```
Get-CFGAggregateDiscoveredResourceCount -ConfigurationAggregatorName Master -
Filters_Region us-east-1
```

Output:

```
GroupByKey GroupedResourceCounts NextToken TotalDiscoveredResources
-----
{} 455
```

Contoh 2: Contoh ini mengembalikan jumlah sumber daya yang dikelompokkan berdasarkan `RESOURCE_TYPE` untuk wilayah yang difilter untuk agregator yang diberikan.

```
Get-CFGAggregateDiscoveredResourceCount -ConfigurationAggregatorName Master -
Filters_Region us-east-1 -GroupByKey RESOURCE_TYPE |
Select-Object -ExpandProperty GroupedResourceCounts
```

Output:

GroupName	ResourceCount
-----	-----
AWS::CloudFormation::Stack	12
AWS::CloudFront::Distribution	1
AWS::CloudTrail::Trail	1
AWS::DynamoDB::Table	1
AWS::EC2::EIP	2
AWS::EC2::FlowLog	2
AWS::EC2::InternetGateway	4
AWS::EC2::NatGateway	2
AWS::EC2::NetworkAcl	4
AWS::EC2::NetworkInterface	12
AWS::EC2::RouteTable	13
AWS::EC2::SecurityGroup	18
AWS::EC2::Subnet	16
AWS::EC2::VPC	4
AWS::EC2::VPCEndpoint	2
AWS::EC2::VPCPeeringConnection	1
AWS::IAM::Group	2
AWS::IAM::Policy	51
AWS::IAM::Role	78
AWS::IAM::User	7
AWS::Lambda::Function	3
AWS::RDS::DBSecurityGroup	1
AWS::S3::Bucket	3
AWS::SSM::AssociationCompliance	107
AWS::SSM::ManagedInstanceInventory	108

- Untuk detail API, lihat [GetAggregateDiscoveredResourceCounts](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CFGAggregateDiscoveredResourceList

Contoh kode berikut menunjukkan cara menggunakan `Get-CFGAggregateDiscoveredResourceList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan pengidentifikasi sumber daya untuk jenis sumber daya tertentu yang dikumpulkan dalam agregator 'Irelandia'. Untuk daftar jenis sumber daya, silakan periksa <https://docs.aws.amazon.com/sdkfornet/v3/apidocs/index.html?page=ConfigService/TConfigServiceResourceType> `ConfigService .html&tocid=Amazon_ . ResourceType`

```
Get-CFGAggregateDiscoveredResourceList -ConfigurationAggregatorName Ireland -
ResourceType ([Amazon.ConfigService.ResourceType]::AWSAutoScalingAutoScalingGroup)
```

Output:

```
ResourceId      : arn:aws:autoscaling:eu-
west-1:123456789012:autoScalingGroup:12e3b4fc-1234-1234-
a123-1d2ba3c45678:autoScalingGroupName/asg-1
ResourceName    : asg-1
ResourceType    : AWS::AutoScaling::AutoScalingGroup
SourceAccountId : 123456789012
SourceRegion    : eu-west-1
```

Contoh 2: Contoh ini mengembalikan jenis sumber daya **AwsEC2SecurityGroup** bernama 'default' untuk agregator yang diberikan disaring dengan wilayah us-east-1.

```
Get-CFGAggregateDiscoveredResourceList -ConfigurationAggregatorName raju -
ResourceType ([Amazon.ConfigService.ResourceType]::AWSEC2SecurityGroup) -
Filters_Region us-east-1 -Filters_ResourceName default
```

Output:

```
ResourceId      : sg-01234bd5dbfa67c89
ResourceName    : default
ResourceType    : AWS::EC2::SecurityGroup
SourceAccountId : 123456789102
SourceRegion    : us-east-1

ResourceId      : sg-0123a4ebbf56789be
ResourceName    : default
ResourceType    : AWS::EC2::SecurityGroup
SourceAccountId : 123456789102
SourceRegion    : us-east-1

ResourceId      : sg-4fc1d234
ResourceName    : default
ResourceType    : AWS::EC2::SecurityGroup
SourceAccountId : 123456789102
SourceRegion    : us-east-1
```

- Untuk detail API, lihat [ListAggregateDiscoveredResources](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CFGAggregateResourceConfig

Contoh kode berikut menunjukkan cara menggunakan `Get-CFGAggregateResourceConfig`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan Item Konfigurasi untuk sumber daya yang diberikan digabungkan dan memperluas Konfigurasi.

```
(Get-CFGAggregateResourceConfig -ResourceIdentifier_SourceRegion
us-east-1 -ResourceIdentifier_SourceAccountId 123456789012 -
ResourceIdentifier_ResourceId sg-4fc1d234 -ResourceIdentifier_ResourceType
([Amazon.ConfigService.ResourceType]::AWSEC2SecurityGroup) -
ConfigurationAggregatorName raju).Configuration | ConvertFrom-Json
```

Output:

```
{
  "description": "default VPC security group",
  "groupName": "default",
  "ipPermissions": [
    {
      "ipProtocol": "-1",
      "ipv6Ranges": [],
      "prefixListIds": [],
      "userIdGroupPairs": [
        {
          "groupId": "sg-4fc1d234",
          "userId": "123456789012"
        }
      ],
      "ipv4Ranges": [],
      "ipRanges": []
    },
    {
      "fromPort": 3389,
      "ipProtocol": "tcp",
      "ipv6Ranges": [],
      "prefixListIds": [],
      "toPort": 3389,
      "userIdGroupPairs": [],
      "ipv4Ranges": [
        {
          "cidrIp": "54.240.197.224/29",
          "description": "office subnet",
          "cidrIp": "72.21.198.65/32",
          "description": "home pc"
        }
      ],
      "ipRanges": [
        "54.240.197.224/29",
        "72.21.198.65/32"
      ]
    }
  ],
  "ownerId": "123456789012",
  "groupId": "sg-4fc1d234",
  "ipPermissions": [
    {
      "ipProtocol": "-1",
      "ipv6Ranges": [],
      "prefixListIds": [],
      "userIdGroupPairs": [
        {
          "groupId": "sg-4fc1d234",
          "userId": "123456789012"
        }
      ],
      "ipv4Ranges": [
        {
          "cidrIp": "0.0.0.0/0"
        }
      ],
      "ipRanges": [
        "0.0.0.0/0"
      ]
    }
  ],
  "tags": [
    {
      "key": "vpcId",
      "value": "vpc-2d1c2e34"
    }
  ]
}
```

- Untuk detail API, lihat [GetAggregateResourceconfig-service](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CFGAggregateResourceConfigBatch

Contoh kode berikut menunjukkan cara menggunakan `Get-CFGAggregateResourceConfigBatch`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil item konfigurasi saat ini untuk sumber daya (diidentifikasi) yang ada di agregator yang diberikan.

```
$resIdentifier=[Amazon.ConfigService.Model.AggregateResourceIdentifier]@{
    ResourceId= "i-012e3cb4df567e8aa"
    ResourceName = "arn:aws:ec2:eu-west-1:123456789012:instance/i-012e3cb4df567e8aa"
    ResourceType = [Amazon.ConfigService.ResourceType]::AWSEC2Instance
    SourceAccountId = "123456789012"
    SourceRegion = "eu-west-1"
}

Get-CFGAggregateResourceConfigBatch -ResourceIdentifier $resIdentifier -
ConfigurationAggregatorName raju
```

Output:

```
BaseConfigurationItems  UnprocessedResourceIdentifiers
-----
{}                      {arn:aws:ec2:eu-west-1:123456789012:instance/
i-012e3cb4df567e8aa}
```

- Untuk detail API, lihat [BatchGetAggregateResourceconfig-service](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CFGAggregationAuthorizationList

Contoh kode berikut menunjukkan cara menggunakan `Get-CFGAggregationAuthorizationList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil otorisasi yang diberikan kepada agregator.

```
Get-CFGAggregationAuthorizationList
```

Output:

```
AggregationAuthorizationArn
    AuthorizedAccountId AuthorizedAwsRegion CreationTime
```

```

-----
-----
arn:aws:config-service:eu-west-1:123456789012:aggregation-
authorization/123456789012/eu-west-1 123456789012 eu-west-1
8/26/2019 12:55:27 AM

```

- Untuk detail API, lihat [DescribeAggregationAuthorizations](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CFGComplianceByConfigRule

Contoh kode berikut menunjukkan cara menggunakan `Get-CFGComplianceByConfigRule`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil rincian kepatuhan untuk aturan `ebs-optimized-instance`, yang tidak ada hasil evaluasi saat ini untuk aturan, sehingga mengembalikan `INSUFFICIENT_DATA`

```
(Get-CFGComplianceByConfigRule -ConfigRuleName ebs-optimized-instance).Compliance
```

Output:

```

ComplianceContributorCount  ComplianceType
-----
                                INSUFFICIENT_DATA

```

Contoh 2: Contoh ini mengembalikan jumlah sumber daya yang tidak sesuai untuk aturan `ALB_HTTP_TO_HTTPS_REDIRECTION_CHECK`.

```
(Get-CFGComplianceByConfigRule -ConfigRuleName ALB_HTTP_TO_HTTPS_REDIRECTION_CHECK -
ComplianceType NON_COMPLIANT).Compliance.ComplianceContributorCount
```

Output:

```

CapExceeded  CappedCount
-----
False        2

```

- Untuk detail API, lihat [DescribeComplianceByConfigRule](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CFGComplianceByResource

Contoh kode berikut menunjukkan cara menggunakan `Get-CFGComplianceByResource`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memeriksa jenis **AWS::SSM::ManagedInstanceInventory** sumber daya untuk jenis kepatuhan 'COMPLIANT'.

```
Get-CFGComplianceByResource -ComplianceType COMPLIANT -ResourceType
AWS::SSM::ManagedInstanceInventory
```

Output:

Compliance	ResourceId	ResourceType
-----	-----	-----
Amazon.ConfigService.Model.Compliance	i-0123bcf4b567890e3	
AWS::SSM::ManagedInstanceInventory		
Amazon.ConfigService.Model.Compliance	i-0a1234f6f5d6b78f7	
AWS::SSM::ManagedInstanceInventory		

- Untuk detail API, lihat [DescribeComplianceByResource](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CFGComplianceDetailsByConfigRule

Contoh kode berikut menunjukkan cara menggunakan `Get-CFGComplianceDetailsByConfigRule`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memperoleh hasil evaluasi untuk aturan `access-keys-rotated` dan mengembalikan output yang dikelompokkan berdasarkan tipe kepatuhan

```
Get-CFGComplianceDetailsByConfigRule -ConfigRuleName access-keys-rotated | Group-
Object ComplianceType
```

Output:

Count	Name	Group
-------	------	-------

```

-----
2 COMPLIANT {Amazon.ConfigService.Model.EvaluationResult,
Amazon.ConfigService.Model.EvaluationResult}
5 NON_COMPLIANT {Amazon.ConfigService.Model.EvaluationResult,
Amazon.ConfigService.Model.EvaluationResult,
Amazon.ConfigService.Model.EvaluationRes...

```

Contoh 2: Contoh ini menanyakan detail kepatuhan access-keys-rotated untuk aturan sumber daya COMPLIANT.

```

Get-CFGComplianceDetailsByConfigRule -ConfigRuleName access-
keys-rotated -ComplianceType COMPLIANT | ForEach-Object
{$_ .EvaluationResultIdentifier.EvaluationResultQualifier}

```

Output:

ConfigRuleName	ResourceId	ResourceType
-----	-----	-----
access-keys-rotated	BCAB1CDJ2LITAPVEW3JAH	AWS::IAM::User
access-keys-rotated	BCAB1CDJ2LITL3EHREM4Q	AWS::IAM::User

- Untuk detail API, lihat [GetComplianceDetailsByConfigRule](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CFGComplianceDetailsByResource

Contoh kode berikut menunjukkan cara menggunakan `Get-CFGComplianceDetailsByResource`.

Alat untuk PowerShell V4

Contoh 1: Contoh evaluasi ini menghasilkan sumber daya yang diberikan.

```

Get-CFGComplianceDetailsByResource -ResourceId ABCD5STJ4EFGHIVEW6JAH -ResourceType
'AWS::IAM::User'

```

Output:

Annotation	:
ComplianceType	: COMPLIANT
ConfigRuleInvokedTime	: 8/25/2019 11:34:56 PM

```
EvaluationResultIdentifier : Amazon.ConfigService.Model.EvaluationResultIdentifier
ResultRecordedTime       : 8/25/2019 11:34:56 PM
ResultToken               :
```

- Untuk detail API, lihat [GetComplianceDetailsByResource](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CFGComplianceSummaryByConfigRule

Contoh kode berikut menunjukkan cara menggunakan `Get-CFGComplianceSummaryByConfigRule`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan jumlah aturan Config yang tidak sesuai.

```
Get-CFGComplianceSummaryByConfigRule -Select
ComplianceSummary.NonCompliantResourceCount
```

Output:

```
CapExceeded CappedCount
-----
False      9
```

- Untuk detail API, lihat [GetComplianceSummaryByConfigRule](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CFGComplianceSummaryByResourceType

Contoh kode berikut menunjukkan cara menggunakan `Get-CFGComplianceSummaryByResourceType`.

Alat untuk PowerShell V4

Contoh 1: Sampel ini mengembalikan jumlah sumber daya yang sesuai atau tidak sesuai dan mengubah output menjadi json.

```
Get-CFGComplianceSummaryByResourceType -Select
ComplianceSummariesByResourceType.ComplianceSummary | ConvertTo-Json
```

```
{
  "ComplianceSummaryTimestamp": "2019-12-14T06:14:49.778Z",
  "CompliantResourceCount": {
    "CapExceeded": false,
    "CappedCount": 2
  },
  "NonCompliantResourceCount": {
    "CapExceeded": true,
    "CappedCount": 100
  }
}
```

- Untuk detail API, lihat [GetComplianceSummaryByResourceType](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CFGConfigRule

Contoh kode berikut menunjukkan cara menggunakan `Get-CFGConfigRule`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan aturan konfigurasi untuk akun, dengan properti yang dipilih.

```
Get-CFGConfigRule | Select-Object ConfigRuleName, ConfigRuleId, ConfigRuleArn,
ConfigRuleState
```

Output:

ConfigRuleName	ConfigRuleId	ConfigRuleArn	ConfigRuleState
-----	-----	-----	-----
ALB_REDIRECTION_CHECK	config-rule-12iyn3	arn:aws:config-	
service:eu-west-1:123456789012:config-rule/config-rule-12iyn3	ACTIVE		
access-keys-rotated	config-rule-aospfr	arn:aws:config-	
service:eu-west-1:123456789012:config-rule/config-rule-aospfr	ACTIVE		
autoscaling-group-elb-healthcheck-required	config-rule-cn1f2x	arn:aws:config-	
service:eu-west-1:123456789012:config-rule/config-rule-cn1f2x	ACTIVE		

- Untuk detail API, lihat [DescribeConfigRules](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CFGConfigRuleEvaluationStatus

Contoh kode berikut menunjukkan cara menggunakan `Get-CFGConfigRuleEvaluationStatus`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan informasi status untuk aturan konfigurasi yang diberikan.

```
Get-CFGConfigRuleEvaluationStatus -ConfigRuleName root-account-mfa-enabled, vpc-
flow-logs-enabled
```

Output:

```
ConfigRuleArn      : arn:aws:config:eu-west-1:123456789012:config-rule/
config-rule-kvq1wk
ConfigRuleId       : config-rule-kvq1wk
ConfigRuleName     : root-account-mfa-enabled
FirstActivatedTime  : 8/27/2019 8:05:17 AM
FirstEvaluationStarted : True
LastErrorCode      :
LastErrorMessage    :
LastFailedEvaluationTime : 1/1/0001 12:00:00 AM
LastFailedInvocationTime : 1/1/0001 12:00:00 AM
LastSuccessfulEvaluationTime : 12/13/2019 8:12:03 AM
LastSuccessfulInvocationTime : 12/13/2019 8:12:03 AM

ConfigRuleArn      : arn:aws:config:eu-west-1:123456789012:config-rule/
config-rule-z1s23b
ConfigRuleId       : config-rule-z1s23b
ConfigRuleName     : vpc-flow-logs-enabled
FirstActivatedTime  : 8/14/2019 6:23:44 AM
FirstEvaluationStarted : True
LastErrorCode      :
LastErrorMessage    :
LastFailedEvaluationTime : 1/1/0001 12:00:00 AM
LastFailedInvocationTime : 1/1/0001 12:00:00 AM
LastSuccessfulEvaluationTime : 12/13/2019 7:12:01 AM
LastSuccessfulInvocationTime : 12/13/2019 7:12:01 AM
```

- Untuk detail API, lihat [DescribeConfigRuleEvaluationStatus](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CFGConfigurationAggregatorList

Contoh kode berikut menunjukkan cara menggunakan `Get-CFGConfigurationAggregatorList`.

Alat untuk PowerShell V4

Contoh 1: Sampel ini mengembalikan semua agregator untuk wilayah/akun.

```
Get-CFGConfigurationAggregatorList
```

Output:

```
AccountAggregationSources      :
  {Amazon.ConfigService.Model.AccountAggregationSource}
ConfigurationAggregatorArn     : arn:aws:config-service:eu-
west-1:123456789012:config-aggregator/config-aggregator-xabca1me
ConfigurationAggregatorName    : IrelandMaster
CreationTime                   : 8/25/2019 11:42:39 PM
LastUpdatedTime                : 8/25/2019 11:42:39 PM
OrganizationAggregationSource  :

AccountAggregationSources     : {}
ConfigurationAggregatorArn     : arn:aws:config-service:eu-
west-1:123456789012:config-aggregator/config-aggregator-qubqabcd
ConfigurationAggregatorName    : raju
CreationTime                   : 8/11/2019 8:39:25 AM
LastUpdatedTime                : 8/11/2019 8:39:25 AM
OrganizationAggregationSource  :
  Amazon.ConfigService.Model.OrganizationAggregationSource
```

- Untuk detail API, lihat [DescribeConfigurationAggregators](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CFGConfigurationAggregatorSourcesStatus

Contoh kode berikut menunjukkan cara menggunakan `Get-CFGConfigurationAggregatorSourcesStatus`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menampilkan bidang yang diminta untuk sumber dalam agregator yang diberikan.

```
Get-CFGConfigurationAggregatorSourcesStatus -ConfigurationAggregatorName raju |
select SourceType, LastUpdateStatus, LastUpdateTime, SourceId
```

Output:

SourceType	LastUpdateStatus	LastUpdateTime	SourceId
ORGANIZATION	SUCCEEDED	12/31/2019 7:45:06 AM	Organization
ACCOUNT	SUCCEEDED	12/31/2019 7:09:38 AM	612641234567
ACCOUNT	SUCCEEDED	12/31/2019 7:12:53 AM	933301234567
ACCOUNT	SUCCEEDED	12/31/2019 7:18:10 AM	933301234567
ACCOUNT	SUCCEEDED	12/31/2019 7:25:17 AM	933301234567
ACCOUNT	SUCCEEDED	12/31/2019 7:25:49 AM	612641234567
ACCOUNT	SUCCEEDED	12/31/2019 7:26:11 AM	612641234567

- Untuk detail API, lihat [DescribeConfigurationAggregatorSourcesStatus](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CFGConfigurationRecorder

Contoh kode berikut menunjukkan cara menggunakan `Get-CFGConfigurationRecorder`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan rincian perekam konfigurasi.

```
Get-CFGConfigurationRecorder | Format-List
```

Output:

```
Name           : default
RecordingGroup  : Amazon.ConfigService.Model.RecordingGroup
RoleARN         : arn:aws:iam::123456789012:role/aws-service-role/
config.amazonaws.com/AWSServiceRoleForConfig
```

- Untuk detail API, lihat [DescribeConfigurationRecorders](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CFGConfigurationRecorderStatus

Contoh kode berikut menunjukkan cara menggunakan `Get-CFGConfigurationRecorderStatus`.

Alat untuk PowerShell V4

Contoh 1: Sampel ini mengembalikan status perekam konfigurasi.

```
Get-CFGConfigurationRecorderStatus
```

Output:

```
LastErrorCode      :  
LastErrorMessage   :  
LastStartTime      : 10/11/2019 10:13:51 AM  
LastStatus         : Success  
LastStatusChangeTime : 12/31/2019 6:14:12 AM  
LastStopTime       : 10/11/2019 10:13:46 AM  
Name               : default  
Recording          : True
```

- Untuk detail API, lihat [DescribeConfigurationRecorderStatus](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CFGConformancePack

Contoh kode berikut menunjukkan cara menggunakan `Get-CFGConformancePack`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan semua paket kesesuaian.

```
Get-CFGConformancePack
```

Output:

```
ConformancePackArn      : arn:aws:config:eu-west-1:123456789012:conformance-  
pack/dono/conformance-pack-p0acq8bpz  
ConformancePackId       : conformance-pack-p0acabcde  
ConformancePackInputParameters : {}  
ConformancePackName     : dono  
CreatedBy               :
```

```
DeliveryS3Bucket           : kt-ps-examples
DeliveryS3KeyPrefix        :
LastUpdateRequestedTime    : 12/31/2019 8:45:31 AM
```

- Untuk detail API, lihat [DescribeConformancePacks](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CFGDeliveryChannel

Contoh kode berikut menunjukkan cara menggunakan `Get-CFGDeliveryChannel`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil saluran pengiriman untuk wilayah tersebut dan menampilkan detailnya.

```
Get-CFGDeliveryChannel -Region eu-west-1 | Select-Object Name, S3BucketName,
S3KeyPrefix,
@{N="DeliveryFrequency";E={$_.ConfigSnapshotDeliveryProperties.DeliveryFrequency}}
```

Output:

Name	S3BucketName	S3KeyPrefix	DeliveryFrequency
----	-----	-----	-----
default	config-bucket-NA	my	TwentyFour_Hours

- Untuk detail API, lihat [DescribeDeliveryChannels](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-CFGResourceTag

Contoh kode berikut menunjukkan cara menggunakan `Get-CFGResourceTag`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan tag terkait untuk sumber daya yang diberikan

```
Get-CFGResourceTag -ResourceArn $rules[0].ConfigRuleArn
```

Output:

Key	Value
---	-----
Version	1.3

- Untuk detail API, lihat [ListTagsForResource](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-CFGConformancePack

Contoh kode berikut menunjukkan cara menggunakan `Remove-CFGConformancePack`.

Alat untuk PowerShell V4

Contoh 1: Sampel ini menghapus paket kesesuaian yang diberikan, bersama dengan semua aturan, tindakan remediasi, dan hasil evaluasi untuk paket tersebut.

```
Remove-CFGConformancePack -ConformancePackName dono
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-CFGConformancePack (DeleteConformancePack)" on
target "dono".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"): Y
```

- Untuk detail API, lihat [DeleteConformancePack](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Write-CFGConformancePack

Contoh kode berikut menunjukkan cara menggunakan `Write-CFGConformancePack`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat paket kesesuaian, mengambil template dari file yaml yang diberikan.

```
Write-CFGConformancePack -ConformancePackName dono -DeliveryS3Bucket amzn-s3-demo-bucket -TemplateBody (Get-Content C:\windows\temp\template.yaml -Raw)
```

- Untuk detail API, lihat [PutConformancePack](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Write-CFGDeliveryChannel

Contoh kode berikut menunjukkan cara menggunakan `Write-CFGDeliveryChannel`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengubah properti `DeliveryFrequency` dari saluran pengiriman yang ada.

```
Write-CFGDeliveryChannel -ConfigSnapshotDeliveryProperties_DeliveryFrequency  
TwentyFour_Hours -DeliveryChannelName default -DeliveryChannel_S3BucketName amzn-  
s3-demo-bucket -DeliveryChannel_S3KeyPrefix my
```

- Untuk detail API, lihat [PutDeliveryChannel](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Contoh Device Farm menggunakan Alat untuk PowerShell V4

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan Device Farm.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

New-DFUpload

Contoh kode berikut menunjukkan cara menggunakan New-DFUpload.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat upload AWS Device Farm untuk aplikasi Android. Anda bisa mendapatkan proyek ARN dari output New-DFProject atau Get-DFProject List. Gunakan URL yang ditandatangani di DFUpload keluaran Baru untuk mengunggah file ke Device Farm.

```
New-DFUpload -ContentType "application/octet-stream" -ProjectArn  
"arn:aws:devicefarm:us-west-2:123456789012:project:EXAMPLEa-7ec1-4741-9c1f-  
d3e04EXAMPLE" -Name "app.apk" -Type ANDROID_APP
```

- Untuk detail API, lihat [CreateUpload](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

AWS Directory Service contoh menggunakan Alat untuk PowerShell V4

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan AWS Directory Service.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

Add-DSIpRoute

Contoh kode berikut menunjukkan cara menggunakan Add-DSIpRoute.

Alat untuk PowerShell V4

Contoh 1: Perintah ini menghapus Tag Sumber Daya yang ditetapkan ke ID Direktori yang ditentukan

```
Add-DSIpRoute -DirectoryId d-123456ijkl -IpRoute @{CidrIp ="203.0.113.5/32"} -  
UpdateSecurityGroupForDirectoryController $true
```

- Untuk detail API, lihat [AddIpRoutes](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Add-DSResourceTag

Contoh kode berikut menunjukkan cara menggunakan `Add-DSResourceTag`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini menambahkan Tag Sumber Daya ke ID Direktori yang ditentukan

```
Add-DSResourceTag -ResourceId d-123456ijkl -Tag @{Key="myTag"; Value="mytgValue"}
```

- Untuk detail API, lihat [AddTagsToResource](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Approve-DSTrust

Contoh kode berikut menunjukkan cara menggunakan `Approve-DSTrust`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memanggil operasi AWS Directory Service `VerifyTrust` API untuk Trustid tertentu.

```
Approve-DSTrust -TrustId t-9067157123
```

- Untuk detail API, lihat [VerifyTrust](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Confirm-DSSharedDirectory

Contoh kode berikut menunjukkan cara menggunakan `Confirm-DSSharedDirectory`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menerima permintaan berbagi direktori yang dikirim dari pemilik Akun AWS direktori.

```
Confirm-DSSharedDirectory -SharedDirectoryId d-9067012345
```

Output:

```
CreatedDateTime      : 12/30/2019 4:20:27 AM
LastUpdatedDateTime : 12/30/2019 4:21:40 AM
OwnerAccountId       : 123456781234
OwnerDirectoryId     : d-123456ijkl
SharedAccountId      : 123456784321
SharedDirectoryId    : d-9067012345
ShareMethod          :
ShareNotes           : This is test sharing
ShareStatus          : Sharing
```

- Untuk detail API, lihat [AcceptSharedDirectory](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Connect-DSDirectory

Contoh kode berikut menunjukkan cara menggunakan `Connect-DSDirectory`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat AD Connector untuk terhubung ke direktori lokal.

```
Connect-DSDirectory -Name contoso.com -ConnectSettings_CustomerUserName
Administrator -Password $Password -ConnectSettings_CustomerDnsIp 172.31.36.96
-ShortName CONTOSO -Size Small -ConnectSettings_VpcId vpc-123459da -
ConnectSettings_SubnetId subnet-1234ccaa, subnet-5678ffbb
```

- Untuk detail API, lihat [ConnectDirectory](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Deny-DSSharedDirectory

Contoh kode berikut menunjukkan cara menggunakan `Deny-DSSharedDirectory`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menolak permintaan berbagi direktori yang dikirim dari akun pemilik direktori.

```
Deny-DSSharedDirectory -SharedDirectoryId d-9067012345
```

Output:

```
d-9067012345
```

- Untuk detail API, lihat [RejectSharedDirectory](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Disable-DSDirectoryShare

Contoh kode berikut menunjukkan cara menggunakan `Disable-DSDirectoryShare`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghentikan pembagian direktori antara pemilik direktori dan akun konsumen.

```
Disable-DSDirectoryShare -DirectoryId d-123456ijkl -UnshareTarget_Id 123456784321 -  
UnshareTarget_Type ACCOUNT
```

Output:

```
d-9067012345
```

- Untuk detail API, lihat [UnshareDirectory](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Disable-DSLDAPs

Contoh kode berikut menunjukkan cara menggunakan `Disable-DSLDAPs`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menonaktifkan panggilan aman LDAP untuk direktori yang ditentukan.

```
Disable-DSLDAPs -DirectoryId d-123456ijkl -Type Client
```

- Untuk detail API, lihat [menonaktifkan DAPS di Referensi Alat AWS untuk PowerShell Cmdlet \(V4\)](#).

Disable-DSRadius

Contoh kode berikut menunjukkan cara menggunakan `Disable-DSRadius`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menonaktifkan server RADIUS yang dikonfigurasi untuk direktori AD Connector atau Microsoft AD.

```
Disable-DSRadius -DirectoryId d-123456ijkl
```

- Untuk detail API, lihat [DisableRadius](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Disable-DSSso

Contoh kode berikut menunjukkan cara menggunakan `Disable-DSSso`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menonaktifkan sistem masuk tunggal untuk sebuah direktori.

```
Disable-DSSso -DirectoryId d-123456ijkl
```

- Untuk detail API, lihat [DisableSso](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Enable-DSDirectoryShare

Contoh kode berikut menunjukkan cara menggunakan `Enable-DSDirectoryShare`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membagikan direktori tertentu di AWS akun Anda dengan AWS Akun lain menggunakan metode Handshake.

```
Enable-DSDirectoryShare -DirectoryId d-123456ijkl -ShareTarget_Id 123456784321 -  
ShareMethod HANDSHAKE -ShareTarget_Type ACCOUNT
```

Output:

```
d-9067012345
```

- Untuk detail API, lihat [ShareDirectory](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Enable-DSLDAPS

Contoh kode berikut menunjukkan cara menggunakan `Enable-DSLDAPS`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengaktifkan sakelar untuk direktori tertentu agar selalu menggunakan panggilan aman LDAP.

```
Enable-DSLDAPS -DirectoryId d-123456ijkl -Type Client
```

- Untuk detail API, lihat [EnableLDAPS](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Enable-DSRadius

Contoh kode berikut menunjukkan cara menggunakan `Enable-DSRadius`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memungkinkan otentikasi multi-faktor (MFA) dengan konfigurasi server RADIUS yang disediakan untuk Konektor AD atau direktori Microsoft AD.

```
Enable-DSRadius -DirectoryId d-123456ijkl  
-RadiusSettings_AuthenticationProtocol PAP  
-RadiusSettings_DisplayLabel Radius  
-RadiusSettings_RadiusPort 1812  
-RadiusSettings_RadiusRetry 4  
-RadiusSettings_RadiusServer 10.4.185.113  
-RadiusSettings_RadiusTimeout 50  
-RadiusSettings_SharedSecret wJalrXUtnFEMI
```

- Untuk detail API, lihat [EnableRadius](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Enable-DSSso

Contoh kode berikut menunjukkan cara menggunakan `Enable-DSSso`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memungkinkan sistem masuk tunggal untuk sebuah direktori.

```
Enable-DSSso -DirectoryId d-123456ijkl
```

- Untuk detail API, lihat [EnableSso](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-DSCertificate

Contoh kode berikut menunjukkan cara menggunakan `Get-DSCertificate`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menampilkan informasi tentang sertifikat yang terdaftar untuk koneksi LDAP yang aman.

```
Get-DSCertificate -DirectoryId d-123456ijkl -CertificateId c-906731e34f
```

Output:

```
CertificateId      : c-906731e34f
CommonName         : contoso-EC2AMAZ-CTGG2NM-CA
ExpiryDateTime     : 4/15/2025 6:34:15 PM
RegisteredDateTime : 4/15/2020 6:38:56 PM
State              : Registered
StateReason        : Certificate registered successfully.
```

- Untuk detail API, lihat [DescribeCertificate](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-DSCertificateList

Contoh kode berikut menunjukkan cara menggunakan `Get-DSCertificateList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan semua sertifikat yang terdaftar untuk koneksi LDAP aman untuk direktori tertentu.

```
Get-DSCertificateList -DirectoryId d-123456ijkl
```

Output:

CertificateId	CommonName	ExpiryDateTime	State
-----	-----	-----	----
c-906731e34f	contoso-EC2AMAZ-CTGG2NM-CA	4/15/2025 6:34:15 PM	Registered

- Untuk detail API, lihat [ListCertificates](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-DSConditionalForwarder

Contoh kode berikut menunjukkan cara menggunakan `Get-DSConditionalForwarder`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini mendapatkan semua Forwarder Bersyarat yang dikonfigurasi dari ID Direktori yang diberikan.

```
Get-DSConditionalForwarder -DirectoryId d-123456ijkl
```

Output:

DnsIpAddress	RemoteDomainName	ReplicationScope
-----	-----	-----
{172.31.77.239}	contoso.com	Domain

- Untuk detail API, lihat [DescribeConditionalForwarders](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-DSDirectory

Contoh kode berikut menunjukkan cara menggunakan `Get-DSDirectory`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini Memperoleh informasi tentang direktori milik akun ini.

```
Get-DSDirectory | Select-Object DirectoryId, Name, DnsIpAddrs, Type
```

Output:

DirectoryId	Name	DnsIpAddrs	Type
-----	----	-----	----
d-123456abcd	abcd.example.com	{172.31.74.189, 172.31.13.145}	SimpleAD
d-123456efgh	wifi.example.com	{172.31.16.108, 172.31.10.56}	ADConnector
d-123456ijkl	lan2.example.com	{172.31.10.56, 172.31.16.108}	MicrosoftAD

- Untuk detail API, lihat [DescribeDirectories](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-DSDirectoryLimit

Contoh kode berikut menunjukkan cara menggunakan `Get-DSDirectoryLimit`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghilangkan informasi batas direktori untuk wilayah us-east-1.

```
Get-DSDirectoryLimit -Region us-east-1
```

Output:

```
CloudOnlyDirectoriesCurrentCount : 1
CloudOnlyDirectoriesLimit        : 10
CloudOnlyDirectoriesLimitReached : False
CloudOnlyMicrosoftADCurrentCount : 1
CloudOnlyMicrosoftADLimit       : 20
CloudOnlyMicrosoftADLimitReached : False
ConnectedDirectoriesCurrentCount : 1
ConnectedDirectoriesLimit        : 10
```

- Untuk detail API, lihat [GetDirectoryLimits](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-DSDomainControllerList

Contoh kode berikut menunjukkan cara menggunakan `Get-DSDomainControllerList`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini mendapatkan daftar rinci Pengontrol Domain yang diluncurkan untuk direktori-id yang disebutkan

```
Get-DSDomainControllerList -DirectoryId d-123456ijkl
```

Output:

```
AvailabilityZone      : us-east-1b
DirectoryId           : d-123456ijkl
DnsIpAddr             : 172.31.16.108
DomainControllerId    : dc-1234567aa6
LaunchTime            : 4/4/2019 4:53:43 AM
Status                : Active
StatusLastUpdatedDateTime : 4/24/2019 1:37:54 PM
StatusReason          :
SubnetId              : subnet-1234kkaa
VpcId                 : vpc-123459d

AvailabilityZone      : us-east-1d
DirectoryId           : d-123456ijkl
DnsIpAddr             : 172.31.10.56
DomainControllerId    : dc-1234567aa7
LaunchTime            : 4/4/2019 4:53:43 AM
Status                : Active
StatusLastUpdatedDateTime : 4/4/2019 5:14:31 AM
StatusReason          :
SubnetId              : subnet-5678ffbb
VpcId                 : vpc-123459d
```

- Untuk detail API, lihat [DescribeDomainControllers](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-DSEventTopic

Contoh kode berikut menunjukkan cara menggunakan `Get-DSEventTopic`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini menunjukkan informasi Topik SNS yang dikonfigurasi untuk pemberitahuan sementara status direktori berubah.

```
Get-DSEventTopic -DirectoryId d-123456ijkl
```

Output:

```
CreatedDateTime : 12/13/2019 11:15:32 AM
DirectoryId     : d-123456ijkl
Status         : Registered
TopicArn       : arn:aws:sns:us-east-1:123456781234:snstopicname
TopicName      : snstopicname
```

- Untuk detail API, lihat [DescribeEventTopics](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-DSIpRouteList

Contoh kode berikut menunjukkan cara menggunakan `Get-DSIpRouteList`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini mendapatkan blok alamat IP publik yang dikonfigurasi di Directory IP Routing

```
Get-DSIpRouteList -DirectoryId d-123456ijkl
```

Output:

```
AddedDateTime    : 12/13/2019 12:27:22 PM
CidrIp           : 203.0.113.5/32
Description      : Public IP of On-Prem DNS Server
DirectoryId      : d-123456ijkl
IpRouteStatusMsg : Added
IpRouteStatusReason :
```

- Untuk detail API, lihat [ListIpRoutes](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-DSLAPSSetting

Contoh kode berikut menunjukkan cara menggunakan `Get-DSLAPSSetting`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan status keamanan LDAP untuk direktori yang ditentukan.

```
Get-DSLAPSSetting -DirectoryId d-123456ijkl
```

Output:

```
LastUpdatedDateTime  LDAPSStatus LDAPSStatusReason
-----
4/15/2020 6:51:03 PM Enabled      LDAPS is enabled successfully.
```

- Untuk detail API, lihat [Menjelaskan LDAPSSettings](#) dalam Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-DSLogSubscriptionList

Contoh kode berikut menunjukkan cara menggunakan `Get-DSLogSubscriptionList`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini mendapatkan informasi log langganan direktori-id yang ditentukan

```
Get-DSLogSubscriptionList -DirectoryId d-123456ijkl
```

Output:

```
DirectoryId  LogGroupName
SubscriptionCreatedDateTime
-----
d-123456ijkl /aws/directoryservice/d-123456ijkl-lan2.example.com 12/14/2019 9:05:23
AM
```

- Untuk detail API, lihat [ListLogSubscriptions](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-DSResourceTag

Contoh kode berikut menunjukkan cara menggunakan `Get-DSResourceTag`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini mendapatkan semua Tag Direktori tertentu.

```
Get-DSResourceTag -ResourceId d-123456ijkl
```

Output:

```
Key      Value
---      -
myTag    myTagValue
```

- Untuk detail API, lihat [ListTagsForResource](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-DSSchemaExtension

Contoh kode berikut menunjukkan cara menggunakan `Get-DSSchemaExtension`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan semua ekstensi skema yang diterapkan ke Direktori Microsoft AD.

```
Get-DSSchemaExtension -DirectoryId d-123456ijkl
```

Output:

```
Description           : ManagedADSchemaExtension
DirectoryId           : d-123456ijkl
EndDateTime           : 4/12/2020 10:30:49 AM
SchemaExtensionId      : e-9067306643
SchemaExtensionStatus  : Completed
SchemaExtensionStatusReason : Schema updates are complete.
StartDateTime          : 4/12/2020 10:28:42 AM
```

- Untuk detail API, lihat [ListSchemaExtensions](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-DSSharedDirectory

Contoh kode berikut menunjukkan cara menggunakan `Get-DSSharedDirectory`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan direktori bersama Akun Anda AWS

```
Get-DSSharedDirectory -OwnerDirectoryId d-123456ijkl -SharedDirectoryId d-9067012345
```

Output:

```
CreatedDateTime      : 12/30/2019 4:34:37 AM
LastUpdatedDateTime  : 12/30/2019 4:35:22 AM
OwnerAccountId       : 123456781234
OwnerDirectoryId     : d-123456ijkl
SharedAccountId      : 123456784321
SharedDirectoryId    : d-9067012345
ShareMethod          : HANDSHAKE
ShareNotes           : This is a test Sharing
ShareStatus          : Shared
```

- Untuk detail API, lihat [DescribeSharedDirectories](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-DSSnapshot

Contoh kode berikut menunjukkan cara menggunakan `Get-DSSnapshot`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini mendapatkan informasi tentang snapshot direktori tertentu milik akun ini.

```
Get-DSSnapshot -DirectoryId d-123456ijkl
```

Output:

```
DirectoryId : d-123456ijkl
```

```

Name      :
SnapshotId : s-9064bd1234
StartTime  : 12/13/2019 6:33:01 PM
Status     : Completed
Type       : Auto

DirectoryId : d-123456ijkl
Name      :
SnapshotId : s-9064bb4321
StartTime  : 12/9/2019 9:48:11 PM
Status     : Completed
Type       : Auto

```

- Untuk detail API, lihat [DescribeSnapshots](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-DSSnapshotLimit

Contoh kode berikut menunjukkan cara menggunakan `Get-DSSnapshotLimit`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini mendapatkan batas snapshot manual untuk direktori tertentu.

```
Get-DSSnapshotLimit -DirectoryId d-123456ijkl
```

Output:

```

ManualSnapshotsCurrentCount ManualSnapshotsLimit ManualSnapshotsLimitReached
-----
0                           5                           False

```

- Untuk detail API, lihat [GetSnapshotLimits](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-DSTrust

Contoh kode berikut menunjukkan cara menggunakan `Get-DSTrust`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini mendapatkan informasi hubungan kepercayaan yang dibuat untuk direktori-id tertentu.

```
Get-DSTrust -DirectoryId d-123456abcd
```

Output:

```

CreatedDateTime      : 7/5/2019 4:55:42 AM
DirectoryId         : d-123456abcd
LastUpdatedDateTime : 7/5/2019 4:56:04 AM
RemoteDomainName    : contoso.com
SelectiveAuth       : Disabled
StateLastUpdatedDateTime : 7/5/2019 4:56:04 AM
TrustDirection      : One-Way: Incoming
TrustId             : t-9067157123
TrustState          : Created
TrustStateReason    :
TrustType           : Forest

```

- Untuk detail API, lihat [DescribeTrusts](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-DSAlias

Contoh kode berikut menunjukkan cara menggunakan `New-DSAlias`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini membuat alias untuk direktori dan menetapkan alias ke direktori-id yang ditentukan.

```
New-DSAlias -DirectoryId d-123456ijkl -Alias MyOrgName
```

Output:

```

Alias      DirectoryId
-----
myorgname d-123456ijkl

```

- Untuk detail API, lihat [CreateAlias](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-DSComputer

Contoh kode berikut menunjukkan cara menggunakan `New-DSComputer`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat objek komputer Active Directory baru.

```
New-DSComputer -DirectoryId d-123456ijkl -ComputerName ADMemberServer -Password
$password
```

Output:

ComputerAttributes	ComputerId
ComputerName	
-----	-----

{WindowsSamName, DistinguishedName}	S-1-5-21-1191241402-978882507-2717148213-1662
ADMemberServer	

- Untuk detail API, lihat [CreateComputer](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-DSConditionalForwarder

Contoh kode berikut menunjukkan cara menggunakan `New-DSConditionalForwarder`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat kondisional forwarder di direktori-ID tertentu. AWS

```
New-DSConditionalForwarder -DirectoryId d-123456ijkl -DnsIpAddress
172.31.36.96,172.31.10.56 -RemoteDomainName contoso.com
```

- Untuk detail API, lihat [CreateConditionalForwarder](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-DSDirectory

Contoh kode berikut menunjukkan cara menggunakan `New-DSDirectory`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat direktori Simple AD baru.

```
New-DSDirectory -Name corp.example.com -Password $Password -Size Small -  
VpcSettings_VpcId vpc-123459d -VpcSettings_SubnetIds subnet-1234kkaa,subnet-5678ffbb
```

- Untuk detail API, lihat [CreateDirectory](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-DSLogSubscription

Contoh kode berikut menunjukkan cara menggunakan New-DSLogSubscription.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat langganan untuk meneruskan log keamanan pengontrol domain Directory Service real-time ke grup log Amazon yang ditentukan di grup CloudWatch log Amazon Anda Akun AWS.

```
New-DSLogSubscription -DirectoryId d-123456ijkl -LogGroupName /aws/directoryservice/  
d-123456ijkl-lan2.example.com
```

- Untuk detail API, lihat [CreateLogSubscription](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-DSMicrosoftAD

Contoh kode berikut menunjukkan cara menggunakan New-DSMicrosoftAD.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat Direktori Microsoft AD baru di AWS Cloud.

```
New-DSMicrosoftAD -Name corp.example.com -Password $Password -edition Standard -  
VpcSettings_VpcId vpc-123459d -VpcSettings_SubnetIds subnet-1234kkaa,subnet-5678ffbb
```

- Untuk detail API, lihat [CreateMicrosoftAD](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-DSSnapshot

Contoh kode berikut menunjukkan cara menggunakan New-DSSnapshot.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat snapshot direktori

```
New-DSSnapshot -DirectoryId d-123456ijkl
```

- Untuk detail API, lihat [CreateSnapshot](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-DSTrust

Contoh kode berikut menunjukkan cara menggunakan `New-DSTrust`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menciptakan kepercayaan Two-Way Forestwide antara direktori AD AWS Microsoft Terkelola dan Microsoft Active Directory lokal yang ada.

```
New-DSTrust -DirectoryId d-123456ijkl -RemoteDomainName contoso.com -TrustDirection  
Two-Way -TrustType Forest -TrustPassword $Password -ConditionalForwarderIpAddr  
172.31.36.96
```

Output:

```
t-9067157123
```

- Untuk detail API, lihat [CreateTrust](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Register-DSCertificate

Contoh kode berikut menunjukkan cara menggunakan `Register-DSCertificate`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendaftarkan sertifikat untuk koneksi LDAP yang aman.

```
$Certificate = Get-Content contoso.cer -Raw  
Register-DSCertificate -DirectoryId d-123456ijkl -CertificateData $Certificate
```

Output:

```
c-906731e350
```

- Untuk detail API, lihat [RegisterCertificate](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Register-DSEventTopic

Contoh kode berikut menunjukkan cara menggunakan `Register-DSEventTopic`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengaitkan direktori sebagai penerbit dengan topik SNS.

```
Register-DSEventTopic -DirectoryId d-123456ijkl -TopicName snstopicname
```

- Untuk detail API, lihat [RegisterEventTopic](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-DSConditionalForwarder

Contoh kode berikut menunjukkan cara menggunakan `Remove-DSConditionalForwarder`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus forwarder bersyarat yang telah disiapkan untuk Directory Anda. AWS

```
Remove-DSConditionalForwarder -DirectoryId d-123456ijkl -RemoteDomainName  
contoso.com
```

- Untuk detail API, lihat [DeleteConditionalForwarder](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-DSDirectory

Contoh kode berikut menunjukkan cara menggunakan `Remove-DSDirectory`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus AWS direktori layanan Direktori (AD/Microsoft AD/ADKonektor Sederhana)

```
Remove-DSDirectory -DirectoryId d-123456ijkl
```

- Untuk detail API, lihat [DeleteDirectory](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-DSIpRoute

Contoh kode berikut menunjukkan cara menggunakan `Remove-DSIpRoute`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini menghapus IP yang ditentukan dari rute IP Dikonfigurasi dari ID Direktori.

```
Remove-DSIpRoute -DirectoryId d-123456ijkl -CidrIp 203.0.113.5/32
```

- Untuk detail API, lihat [RemoveIpRoutes](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-DSLogSubscription

Contoh kode berikut menunjukkan cara menggunakan `Remove-DSLogSubscription`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini menghapus Langganan Log dari ID Direktori yang ditentukan

```
Remove-DSLogSubscription -DirectoryId d-123456ijkl
```

- Untuk detail API, lihat [DeleteLogSubscription](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-DSResourceTag

Contoh kode berikut menunjukkan cara menggunakan `Remove-DSResourceTag`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini menghapus Tag Sumber Daya yang ditetapkan ke ID Direktori yang ditentukan

```
Remove-DSResourceTag -ResourceId d-123456ijkl -TagKey myTag
```

- Untuk detail API, lihat [RemoveTagsFromResource](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-DSSnapshot

Contoh kode berikut menunjukkan cara menggunakan `Remove-DSSnapshot`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus snapshot yang dibuat secara manual.

```
Remove-DSSnapshot -SnapshotId s-9068b488kc
```

- Untuk detail API, lihat [DeleteSnapshot](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-DSTrust

Contoh kode berikut menunjukkan cara menggunakan `Remove-DSTrust`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus hubungan kepercayaan yang ada antara Direktori AD AWS Terkelola dan domain eksternal.

```
Get-DSTrust -DirectoryId d-123456ijkl -Select Trusts.TrustId | Remove-DSTrust
```

Output:

```
t-9067157123
```

- Untuk detail API, lihat [DeleteTrust](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Reset-DSUserPassword

Contoh kode berikut menunjukkan cara menggunakan `Reset-DSUserPassword`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengatur ulang kata sandi pengguna Active Directory yang dinamai `ADUser` di Microsoft AD yang AWS dikelola atau Simple AD Directory

```
Reset-DSUserPassword -UserName ADuser -DirectoryId d-123456ijkl -NewPassword  
$Password
```

- Untuk detail API, lihat [ResetUserPassword](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Restore-DSFromSnapshot

Contoh kode berikut menunjukkan cara menggunakan `Restore-DSFromSnapshot`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan direktori menggunakan snapshot direktori yang ada.

```
Restore-DSFromSnapshot -SnapshotId s-9068b488kc
```

- Untuk detail API, lihat [RestoreFromSnapshot](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Set-DSDomainControllerCount

Contoh kode berikut menunjukkan cara menggunakan `Set-DSDomainControllerCount`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menetapkan jumlah pengontrol domain ke 3 untuk direktori-id tertentu.

```
Set-DSDomainControllerCount -DirectoryId d-123456ijkl -DesiredNumber 3
```

- Untuk detail API, lihat [UpdateNumberOfDomainControllers](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Start-DSSchemaExtension

Contoh kode berikut menunjukkan cara menggunakan `Start-DSSchemaExtension`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini Menerapkan ekstensi skema ke direktori Microsoft AD.

```
$ldif = Get-Content D:\Users\Username\Downloads\ExtendedSchema.ldf -Raw
Start-DSSchemaExtension -DirectoryId d-123456ijkl -
CreateSnapshotBeforeSchemaExtension $true -Description ManagedADSchemaExtension -
LdifContent $ldif
```

Output:

```
e-9067306643
```

- Untuk detail API, lihat [StartSchemaExtension](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Stop-DSSchemaExtension

Contoh kode berikut menunjukkan cara menggunakan `Stop-DSSchemaExtension`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membatalkan ekstensi skema yang sedang berlangsung ke direktori Microsoft AD.

```
Stop-DSSchemaExtension -DirectoryId d-123456ijkl -SchemaExtensionId e-9067306643
```

- Untuk detail API, lihat [CancelSchemaExtension](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Unregister-DSCertificate

Contoh kode berikut menunjukkan cara menggunakan `Unregister-DSCertificate`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus dari sistem sertifikat yang terdaftar untuk koneksi LDAP yang aman..

```
Unregister-DSCertificate -DirectoryId d-123456ijkl -CertificateId c-906731e34f
```

- Untuk detail API, lihat [DeregisterCertificate](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Unregister-DSEventTopic

Contoh kode berikut menunjukkan cara menggunakan `Unregister-DSEventTopic`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus direktori specified sebagai penerbit ke topik SNS yang ditentukan.

```
Unregister-DSEventTopic -DirectoryId d-123456ijkl -TopicName snstopicname
```

- Untuk detail API, lihat [DeregisterEventTopic](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-DSConditionalForwarder

Contoh kode berikut menunjukkan cara menggunakan `Update-DSConditionalForwarder`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memperbarui forwarder bersyarat yang telah disiapkan untuk direktori Anda. AWS

```
Update-DSConditionalForwarder -DirectoryId d-123456ijkl -DnsIpAddr  
172.31.36.96,172.31.16.108 -RemoteDomainName contoso.com
```

- Untuk detail API, lihat [UpdateConditionalForwarder](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-DSRadius

Contoh kode berikut menunjukkan cara menggunakan `Update-DSRadius`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memperbarui informasi server RADIUS untuk direktori AD Connector atau Microsoft AD.

```
Update-DSRadius -DirectoryId d-123456ijkl -RadiusSettings_RadiusRetry 3
```

- Untuk detail API, lihat [UpdateRadius](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-DSTrust

Contoh kode berikut menunjukkan cara menggunakan `Update-DSTrust`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memperbarui `SelectiveAuth` parameter `trust-id` tertentu dari `Disabled` ke `Enabled`.

```
Update-DSTrust -TrustId t-9067157123 -SelectiveAuth Enabled
```

Output:

RequestId	TrustId
-----	-----
138864a7-c9a8-4ad1-a828-eae479e85b45	t-9067157123

- Untuk detail API, lihat [UpdateTrust](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

AWS DMS contoh menggunakan Alat untuk PowerShell V4

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan AWS DMS.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

New-DMSReplicationTask

Contoh kode berikut menunjukkan cara menggunakan `New-DMSReplicationTask`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat tugas replikasi AWS Database Migration Service baru yang menggunakan CdcStartTime sebagai ganti. CdcStartPosition MigrationType Ini diatur ke "full-load-and-cdc", yang berarti tabel target harus kosong. Tugas baru ditandai dengan tag yang memiliki kunci Stage dan nilai kunci Test. Untuk informasi selengkapnya tentang nilai yang digunakan oleh cmdlet ini, lihat Membuat Tugas (https://docs.aws.amazon.com/dms/latest/userguide/CHAP_Tasks.creating.html) di Panduan Pengguna Layanan Migrasi Database. AWS

```
New-DMSReplicationTask -ReplicationInstanceArn "arn:aws:dms:us-east-1:123456789012:rep:EXAMPLE66XFJUWATDJGBEXAMPLE" `
  -CdcStartTime "2019-08-08T12:12:12" `
  -CdcStopPosition "server_time:2019-08-09T12:12:12" `
  -MigrationType "full-load-and-cdc" `
  -ReplicationTaskIdentifier "task1" `
  -ReplicationTaskSetting "" `
  -SourceEndpointArn "arn:aws:dms:us-east-1:123456789012:endpoint:EXAMPLEW5UANC7Y3P4EEXAMPLE" `
  -TableMapping "file:///home/testuser/table-mappings.json" `
  -Tag @{"Key"="Stage";"Value"="Test"} `
  -TargetEndpointArn "arn:aws:dms:us-east-1:123456789012:endpoint:EXAMPLEJZASXWHTWCLNEXAMPLE"
```

- Untuk detail API, lihat [CreateReplicationTask](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Contoh DynamoDB menggunakan Alat untuk V4 PowerShell

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan DynamoDB.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

Add-DDBIndexSchema

Contoh kode berikut menunjukkan cara menggunakan `Add-DDBIndexSchema`.

Alat untuk PowerShell V4

Contoh 1: Membuat `TableSchema` objek kosong dan menambahkan definisi indeks sekunder lokal baru sebelum menulis `TableSchema` objek ke pipeline.

```
$schema | Add-DDBIndexSchema -IndexName "LastPostIndex" -RangeKeyName
"LastPostDateTime" -RangeKeyDataType "S" -ProjectionType "keys_only"
$schema = New-DDBTableSchema
```

Output:

AttributeSchema	KeySchema
LocalSecondaryIndexSchema	
-----	-----

{LastPostDateTime}	{}
{LastPostIndex}	

Contoh 2: Menambahkan definisi indeks sekunder lokal baru ke `TableSchema` objek yang disediakan sebelum menulis `TableSchema` objek kembali ke pipeline. `TableSchema` Objek juga dapat diberikan menggunakan parameter `-Schema`.

```
New-DDBTableSchema | Add-DDBIndexSchema -IndexName "LastPostIndex" -RangeKeyName
"LastPostDateTime" -RangeKeyDataType "S" -ProjectionType "keys_only"
```

Output:

AttributeSchema	KeySchema
LocalSecondaryIndexSchema	
-----	-----

{LastPostDateTime}	{}
{LastPostIndex}	

- Untuk detail API, lihat [Menambahkan DDBIndex Skema](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Add-DDBKeySchema

Contoh kode berikut menunjukkan cara menggunakan Add-DDBKeySchema.

Alat untuk PowerShell V4

Contoh 1: Membuat TableSchema objek kosong dan menambahkan entri definisi kunci dan atribut ke dalamnya menggunakan data kunci yang ditentukan sebelum menulis TableSchema objek ke pipeline. Tipe kunci dinyatakan sebagai 'HASH' secara default; gunakan - KeyType paameter dengan nilai 'RANGE' untuk mendeklarasikan kunci rentang.

```
$schema = New-DDBTableSchema
$schema | Add-DDBKeySchema -KeyName "ForumName" -KeyDataType "S"
```

Output:

AttributeSchema	KeySchema
LocalSecondaryIndexSchema	
-----	-----

{ForumName}	{ForumName}
{}	

Contoh 2: Menambahkan entri definisi kunci dan atribut baru ke TableSchema objek yang disediakan sebelum menulis TableSchema objek ke pipeline. Tipe kunci dinyatakan sebagai 'HASH' secara default; gunakan - KeyType paameter dengan nilai 'RANGE' untuk mendeklarasikan kunci rentang. TableSchema Objek juga dapat diberikan menggunakan parameter -Schema.

```
New-DDBTableSchema | Add-DDBKeySchema -KeyName "ForumName" -KeyDataType "S"
```

Output:

AttributeSchema	KeySchema
LocalSecondaryIndexSchema	

```

-----
-----
{ForumName}          {ForumName}
{}

```

- Untuk detail API, lihat [Menambahkan DDBKey Skema](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

ConvertFrom-DDBItem

Contoh kode berikut menunjukkan cara menggunakan `ConvertFrom-DDBItem`.

Alat untuk PowerShell V4

Contoh 1: `ConvertFrom - DDBItem` digunakan untuk mengonversi hasil `Get-DDBItem` dari hashtable `AttributeValues` DynamoDB ke hashtable tipe umum seperti string dan double.

```

@{
    SongTitle = 'Somewhere Down The Road'
    Artist    = 'No One You Know'
} | ConvertTo-DDBItem

Get-DDBItem -TableName 'Music' -Key $key | ConvertFrom-DDBItem

```

Output:

Name	Value
----	----
Genre	Country
Artist	No One You Know
Price	1.94
CriticRating	9
SongTitle	Somewhere Down The Road
AlbumTitle	Somewhat Famous

- Untuk detail API, lihat [ConvertFrom- DDBItem](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

ConvertTo-DDBItem

Contoh kode berikut menunjukkan cara menggunakan `ConvertTo-DDBItem`.

Alat untuk PowerShell V4

Contoh 1: Contoh untuk mengubah hashtable menjadi kamus nilai atribut DynamoDB.

```
@{
    SongTitle = 'Somewhere Down The Road'
    Artist    = 'No One You Know'
} | ConvertTo-DDBItem
```

Key	Value
---	-----
SongTitle	Amazon.DynamoDBv2.Model.AttributeValue
Artist	Amazon.DynamoDBv2.Model.AttributeValue

Contoh 2: Contoh untuk mengubah hashtable menjadi kamus nilai atribut DynamoDB.

```
@{
    MyMap      = @{
        MyString = 'my string'
    }
    MyStringSet = [System.Collections.Generic.HashSet[String]]@('my', 'string')
    MyNumericSet = [System.Collections.Generic.HashSet[Int]]@(1, 2, 3)
    MyBinarySet = [System.Collections.Generic.HashSet[System.IO.MemoryStream]]@(
        ([IO.MemoryStream]::new([Text.Encoding]::UTF8.GetBytes('my'))),
        ([IO.MemoryStream]::new([Text.Encoding]::UTF8.GetBytes('string'))))
    )
    MyList1     = @('my', 'string')
    MyList2     = [System.Collections.Generic.List[Int]]@(1, 2)
    MyList3     = [System.Collections.ArrayList]@('one', 2, $true)
} | ConvertTo-DDBItem
```

Output:

Key	Value
---	-----
MyStringSet	Amazon.DynamoDBv2.Model.AttributeValue
MyList1	Amazon.DynamoDBv2.Model.AttributeValue
MyNumericSet	Amazon.DynamoDBv2.Model.AttributeValue
MyList2	Amazon.DynamoDBv2.Model.AttributeValue
MyBinarySet	Amazon.DynamoDBv2.Model.AttributeValue
MyMap	Amazon.DynamoDBv2.Model.AttributeValue
MyList3	Amazon.DynamoDBv2.Model.AttributeValue

- Untuk detail API, lihat [ConvertTo- DDBItem](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-DDBBatchItem

Contoh kode berikut menunjukkan cara menggunakan `Get-DDBBatchItem`.

Alat untuk PowerShell V4

Contoh 1: Mendapatkan item dengan `SongTitle` “Somewhere Down The Road” dari tabel DynamoDB 'Music' dan 'Songs'.

```
$key = @{
    SongTitle = 'Somewhere Down The Road'
    Artist = 'No One You Know'
} | ConvertTo-DDBItem

$keysAndAttributes = New-Object Amazon.DynamoDBv2.Model.KeysAndAttributes
$list = New-Object
'System.Collections.Generic.List[System.Collections.Generic.Dictionary[String,
Amazon.DynamoDBv2.Model.AttributeValue]]'
$list.Add($key)
$keysAndAttributes.Keys = $list

$requestItem = @{
    'Music' = [Amazon.DynamoDBv2.Model.KeysAndAttributes]$keysAndAttributes
    'Songs' = [Amazon.DynamoDBv2.Model.KeysAndAttributes]$keysAndAttributes
}

$batchItems = Get-DDBBatchItem -RequestItem $requestItem
$batchItems.GetEnumerator() | ForEach-Object {$PSItem.Value} | ConvertFrom-DDBItem
```

Output:

Name	Value
----	-----
Artist	No One You Know
SongTitle	Somewhere Down The Road
AlbumTitle	Somewhat Famous
CriticRating	10
Genre	Country
Price	1.94

Artist	No One You Know
SongTitle	Somewhere Down The Road
AlbumTitle	Somewhat Famous
CriticRating	10
Genre	Country
Price	1.94

- Untuk detail API, lihat [BatchGetItem](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-DDBItem

Contoh kode berikut menunjukkan cara menggunakan `Get-DDBItem`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan item DynamoDB dengan `SongTitle` kunci partisi dan kunci sort `Artist`.

```
$key = @{
    SongTitle = 'Somewhere Down The Road'
    Artist = 'No One You Know'
} | ConvertTo-DDBItem

Get-DDBItem -TableName 'Music' -Key $key | ConvertFrom-DDBItem
```

Output:

Name	Value
----	-----
Genre	Country
SongTitle	Somewhere Down The Road
Price	1.94
Artist	No One You Know
CriticRating	9
AlbumTitle	Somewhat Famous

- Untuk detail API, lihat [GetItem](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-DDBTable

Contoh kode berikut menunjukkan cara menggunakan `Get-DDBTable`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan rincian tabel yang ditentukan.

```
Get-DDBTable -TableName "myTable"
```

- Untuk detail API, lihat [DescribeTable](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-DDBTableList

Contoh kode berikut menunjukkan cara menggunakan `Get-DDBTableList`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan rincian semua tabel, secara otomatis iterasi sampai layanan menunjukkan tidak ada tabel lebih lanjut.

```
Get-DDBTableList
```

- Untuk detail API, lihat [ListTables](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Invoke-DDBQuery

Contoh kode berikut menunjukkan cara menggunakan `Invoke-DDBQuery`.

Alat untuk PowerShell V4

Contoh 1: Memanggil query yang mengembalikan item DynamoDB dengan yang ditentukan dan `Artist`. `SongTitle`

```
$invokeDDBQuery = @{
    TableName = 'Music'
    KeyConditionExpression = ' SongTitle = :SongTitle and Artist = :Artist'
    ExpressionAttributeValues = @{
        ':SongTitle' = 'Somewhere Down The Road'
        ':Artist' = 'No One You Know'
    } | ConvertTo-DDBItem
}
Invoke-DDBQuery @invokeDDBQuery | ConvertFrom-DDBItem
```

Output:

Name	Value
----	-----
Genre	Country
Artist	No One You Know
Price	1.94
CriticRating	9
SongTitle	Somewhere Down The Road
AlbumTitle	Somewhat Famous

- Untuk detail API, lihat [Kueri](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Invoke-DDBScan

Contoh kode berikut menunjukkan cara menggunakan Invoke-DDBScan.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan semua item dalam tabel Musik.

```
Invoke-DDBScan -TableName 'Music' | ConvertFrom-DDBItem
```

Output:

Name	Value
----	-----
Genre	Country
Artist	No One You Know
Price	1.94
CriticRating	9
SongTitle	Somewhere Down The Road
AlbumTitle	Somewhat Famous
Genre	Country
Artist	No One You Know
Price	1.98
CriticRating	8.4
SongTitle	My Dog Spot
AlbumTitle	Hey Now

Contoh 2: Mengembalikan item dalam tabel Musik dengan CriticRating lebih besar dari atau sama dengan sembilan.

```
$scanFilter = @{
    CriticRating = [Amazon.DynamoDBv2.Model.Condition]{
        AttributeValueList = @(@{N = '9'})
        ComparisonOperator = 'GE'
    }
}
Invoke-DDBScan -TableName 'Music' -ScanFilter $scanFilter | ConvertFrom-DDBItem
```

Output:

Name	Value
----	-----
Genre	Country
Artist	No One You Know
Price	1.94
CriticRating	9
SongTitle	Somewhere Down The Road
AlbumTitle	Somewhat Famous

- Untuk detail API, lihat [Memindai](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-DDBTable

Contoh kode berikut menunjukkan cara menggunakan New-DDBTable.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat tabel bernama Thread yang memiliki kunci utama yang terdiri dari 'ForumName' (hash tipe kunci) dan 'Subject' (rentang tipe kunci). Skema yang digunakan untuk membangun tabel dapat disalurkan ke setiap cmdlet seperti yang ditunjukkan atau ditentukan menggunakan parameter -Schema.

```
$schema = New-DDBTableSchema
$schema | Add-DDBKeySchema -KeyName "ForumName" -KeyDataType "S"
$schema | Add-DDBKeySchema -KeyName "Subject" -KeyType RANGE -KeyDataType "S"
$schema | New-DDBTable -TableName "Thread" -ReadCapacity 10 -WriteCapacity 5
```

Output:

```
AttributeDefinitions : {ForumName, Subject}
```

```

TableName      : Thread
KeySchema      : {ForumName, Subject}
TableStatus    : CREATING
CreationDateTime : 10/28/2013 4:39:49 PM
ProvisionedThroughput : Amazon.DynamoDBv2.Model.ProvisionedThroughputDescription
TableSizeBytes : 0
ItemCount      : 0
LocalSecondaryIndexes : {}

```

Contoh 2: Contoh ini membuat tabel bernama Thread yang memiliki kunci utama yang terdiri dari 'ForumName' (hash tipe kunci) dan 'Subject' (rentang tipe kunci). Indeks sekunder lokal juga didefinisikan. Kunci indeks sekunder lokal akan diatur secara otomatis dari kunci hash utama pada tabel (ForumName). Skema yang digunakan untuk membangun tabel dapat disalurkan ke setiap cmdlet seperti yang ditunjukkan atau ditentukan menggunakan parameter -Schema.

```

$schema = New-DDBTableSchema
$schema | Add-DDBKeySchema -KeyName "ForumName" -KeyDataType "S"
$schema | Add-DDBKeySchema -KeyName "Subject" -KeyDataType "S"
$schema | Add-DDBIndexSchema -IndexName "LastPostIndex" -RangeKeyName
    "LastPostDateTime" -RangeKeyDataType "S" -ProjectionType "keys_only"
$schema | New-DDBTable -TableName "Thread" -ReadCapacity 10 -WriteCapacity 5

```

Output:

```

AttributeDefinitions : {ForumName, LastPostDateTime, Subject}
TableName            : Thread
KeySchema            : {ForumName, Subject}
TableStatus          : CREATING
CreationDateTime      : 10/28/2013 4:39:49 PM
ProvisionedThroughput : Amazon.DynamoDBv2.Model.ProvisionedThroughputDescription
TableSizeBytes        : 0
ItemCount             : 0
LocalSecondaryIndexes : {LastPostIndex}

```

Contoh 3: Contoh ini menunjukkan cara menggunakan pipeline tunggal untuk membuat tabel bernama Thread yang memiliki kunci utama yang terdiri dari 'ForumName' (hash tipe kunci) dan 'Subjek' (rentang tipe kunci) dan indeks sekunder lokal. DDBKeyAdd-Schema dan Add- DDBIndex Schema membuat TableSchema objek baru untuk Anda jika tidak disediakan dari pipeline atau parameter -Schema.

```
New-DDBTableSchema |
```

```
Add-DDBKeySchema -KeyName "ForumName" -KeyDataType "S" |
Add-DDBKeySchema -KeyName "Subject" -KeyDataType "S" |
Add-DDBIndexSchema -IndexName "LastPostIndex" `
                    -RangeKeyName "LastPostDateTime" `
                    -RangeKeyDataType "S" `
                    -ProjectionType "keys_only" |
New-DDBTable -TableName "Thread" -ReadCapacity 10 -WriteCapacity 5
```

Output:

```
AttributeDefinitions : {ForumName, LastPostDateTime, Subject}
TableName            : Thread
KeySchema            : {ForumName, Subject}
TableStatus          : CREATING
CreationDateTime      : 10/28/2013 4:39:49 PM
ProvisionedThroughput : Amazon.DynamoDBv2.Model.ProvisionedThroughputDescription
TableSizeBytes       : 0
ItemCount            : 0
LocalSecondaryIndexes : {LastPostIndex}
```

- Untuk detail API, lihat [CreateTable](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-DDBTableSchema

Contoh kode berikut menunjukkan cara menggunakan `New-DDBTableSchema`.

Alat untuk PowerShell V4

Contoh 1: Membuat `TableSchema` objek kosong yang siap menerima definisi kunci dan indeks untuk digunakan dalam membuat tabel Amazon DynamoDB baru. Objek yang dikembalikan dapat disalurkan ke dalam `Add-DDBKeySchema`, `Add-DDBIndexSchema` dan `New-DDBTable` cmdlet atau diteruskan ke mereka menggunakan parameter `-Schema` pada setiap cmdlet.

```
New-DDBTableSchema
```

Output:

```
AttributeSchema      KeySchema
  LocalSecondaryIndexSchema
-----
-----
```

```
{}
```

```
{}
```

- Untuk detail API, lihat [DDBTableSkema Baru di](#) Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-DDBItem

Contoh kode berikut menunjukkan cara menggunakan `Remove-DDBItem`.

Alat untuk PowerShell V4

Contoh 1: Menghapus item DynamoDB yang cocok dengan kunci yang disediakan.

```
$key = @{
    SongTitle = 'Somewhere Down The Road'
    Artist = 'No One You Know'
} | ConvertTo-DDBItem
Remove-DDBItem -TableName 'Music' -Key $key -Confirm:$false
```

- Untuk detail API, lihat [DeleteItem](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-DDBTable

Contoh kode berikut menunjukkan cara menggunakan `Remove-DDBTable`.

Alat untuk PowerShell V4

Contoh 1: Menghapus tabel yang ditentukan. Anda diminta untuk konfirmasi sebelum operasi berlangsung.

```
Remove-DDBTable -TableName "myTable"
```

Contoh 2: Menghapus tabel yang ditentukan. Anda tidak diminta untuk konfirmasi sebelum operasi berlangsung.

```
Remove-DDBTable -TableName "myTable" -Force
```

- Untuk detail API, lihat [DeleteTable](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Set-DDBBatchItem

Contoh kode berikut menunjukkan cara menggunakan `Set-DDBBatchItem`.

Alat untuk PowerShell V4

Contoh 1: Membuat item baru, atau mengganti item yang ada dengan item baru di tabel DynamoDB Musik dan Lagu.

```
$item = @{
    SongTitle = 'Somewhere Down The Road'
    Artist = 'No One You Know'
    AlbumTitle = 'Somewhat Famous'
    Price = 1.94
    Genre = 'Country'
    CriticRating = 10.0
} | ConvertTo-DDBItem

$writeRequest = New-Object Amazon.DynamoDBv2.Model.WriteRequest
$writeRequest.PutRequest = [Amazon.DynamoDBv2.Model.PutRequest]$item
```

Output:

```
$requestItem = @{
    'Music' = [Amazon.DynamoDBv2.Model.WriteRequest]($writeRequest)
    'Songs' = [Amazon.DynamoDBv2.Model.WriteRequest]($writeRequest)
}

Set-DDBBatchItem -RequestItem $requestItem
```

- Untuk detail API, lihat [BatchWriteItem](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Set-DDBItem

Contoh kode berikut menunjukkan cara menggunakan `Set-DDBItem`.

Alat untuk PowerShell V4

Contoh 1: Membuat item baru, atau mengganti item yang sudah ada dengan item baru.

```
$item = @{
```

```

    SongTitle = 'Somewhere Down The Road'
    Artist = 'No One You Know'
        AlbumTitle = 'Somewhat Famous'
        Price = 1.94
        Genre = 'Country'
        CriticRating = 9.0
} | ConvertTo-DDBItem
Set-DDBItem -TableName 'Music' -Item $item

```

- Untuk detail API, lihat [PutItem](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-DDBItem

Contoh kode berikut menunjukkan cara menggunakan `Update-DDBItem`.

Alat untuk PowerShell V4

Contoh 1: Menetapkan atribut genre ke 'Rap' pada item DynamoDB dengan `SongTitle` kunci partisi dan `Artist` kunci sortir.

```

$key = @{
    SongTitle = 'Somewhere Down The Road'
    Artist = 'No One You Know'
} | ConvertTo-DDBItem

$updateDdbItem = @{
    TableName = 'Music'
    Key = $key
    UpdateExpression = 'set Genre = :val1'
    ExpressionAttributeValue = (@{
        ':val1' = ([Amazon.DynamoDBv2.Model.AttributeValue]'Rap')
    })
}
Update-DDBItem @updateDdbItem

```

Output:

Name	Value
----	-----
Genre	Rap

- Untuk detail API, lihat [UpdateItem](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-DDBTable

Contoh kode berikut menunjukkan cara menggunakan `Update-DDBTable`.

Alat untuk PowerShell V4

Contoh 1: Memperbarui throughput yang disediakan untuk tabel yang diberikan.

```
Update-DDBTable -TableName "myTable" -ReadCapacity 10 -WriteCapacity 5
```

- Untuk detail API, lihat [UpdateTable](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

EC2 Contoh Amazon menggunakan Alat untuk PowerShell V4

Contoh kode berikut menunjukkan cara melakukan tindakan dan menerapkan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan Amazon EC2.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

Add-EC2CapacityReservation

Contoh kode berikut menunjukkan cara menggunakan `Add-EC2CapacityReservation`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat Reservasi Kapasitas baru dengan atribut yang ditentukan

```
Add-EC2CapacityReservation -InstanceType m4.xlarge -InstanceCount 2 -  
AvailabilityZone eu-west-1b -EbsOptimized True -InstancePlatform Windows
```

Output:

```

AvailabilityZone      : eu-west-1b
AvailableInstanceCount : 2
CapacityReservationId : cr-0c1f2345db6f7cdba
CreateDate            : 3/28/2019 9:29:41 AM
EbsOptimized          : True
EndDate               : 1/1/0001 12:00:00 AM
EndDateType           : unlimited
EphemeralStorage      : False
InstanceMatchCriteria : open
InstancePlatform      : Windows
InstanceType          : m4.xlarge
State                 : active
Tags                  : {}
Tenancy                : default
TotalInstanceCount    : 2

```

- Untuk detail API, lihat [CreateCapacityReservation](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Add-EC2InternetGateway

Contoh kode berikut menunjukkan cara menggunakan `Add-EC2InternetGateway`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini melampirkan gateway Internet yang ditentukan ke VPC yang ditentukan.

```
Add-EC2InternetGateway -InternetGatewayId igw-1a2b3c4d -VpcId vpc-12345678
```

Contoh 2: Contoh ini membuat VPC dan gateway Internet, dan kemudian melampirkan gateway Internet ke VPC.

```
$vpc = New-EC2Vpc -CidrBlock 10.0.0.0/16
New-EC2InternetGateway | Add-EC2InternetGateway -VpcId $vpc.VpcId
```

- Untuk detail API, lihat [AttachInternetGateway](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Add-EC2NetworkInterface

Contoh kode berikut menunjukkan cara menggunakan `Add-EC2NetworkInterface`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini melampirkan antarmuka jaringan yang ditentukan ke instance yang ditentukan.

```
Add-EC2NetworkInterface -NetworkInterfaceId eni-12345678 -InstanceId i-1a2b3c4d -DeviceIndex 1
```

Output:

```
eni-attach-1a2b3c4d
```

- Untuk detail API, lihat [AttachNetworkInterface](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Add-EC2Volume

Contoh kode berikut menunjukkan cara menggunakan `Add-EC2Volume`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini melampirkan volume yang ditentukan ke instance yang ditentukan dan memaparkannya dengan nama perangkat yang ditentukan.

```
Add-EC2Volume -VolumeId vol-12345678 -InstanceId i-1a2b3c4d -Device /dev/sdh
```

Output:

```
AttachTime       : 12/22/2015 1:53:58 AM
DeleteOnTermination : False
Device           : /dev/sdh
InstanceId       : i-1a2b3c4d
State            : attaching
VolumeId        : vol-12345678
```

- Untuk detail API, lihat [AttachVolume](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Add-EC2VpnGateway

Contoh kode berikut menunjukkan cara menggunakan `Add-EC2VpnGateway`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini melampirkan gateway pribadi virtual yang ditentukan ke VPC yang ditentukan.

```
Add-EC2VpnGateway -VpnGatewayId vgw-1a2b3c4d -VpcId vpc-12345678
```

Output:

State	VpcId
-----	-----
attaching	vpc-12345678

- Untuk detail API, lihat [AttachVpnGateway](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Approve-EC2VpcPeeringConnection

Contoh kode berikut menunjukkan cara menggunakan `Approve-EC2VpcPeeringConnection`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menyetujui `pcx-1dfad234b56ff78be` yang diminta `VpcPeeringConnectionId`

```
Approve-EC2VpcPeeringConnection -VpcPeeringConnectionId pcx-1dfad234b56ff78be
```

Output:

```

AccepterVpcInfo      : Amazon.EC2.Model.VpcPeeringConnectionVpcInfo
ExpirationTime       : 1/1/0001 12:00:00 AM
RequesterVpcInfo     : Amazon.EC2.Model.VpcPeeringConnectionVpcInfo
Status               : Amazon.EC2.Model.VpcPeeringConnectionStateReason
Tags                 : {}
VpcPeeringConnectionId : pcx-1dfad234b56ff78be
  
```

- Untuk detail API, lihat [AcceptVpcPeeringConnection](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Confirm-EC2ProductInstance

Contoh kode berikut menunjukkan cara menggunakan `Confirm-EC2ProductInstance`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menentukan apakah kode produk yang ditentukan dikaitkan dengan instance yang ditentukan.

```
Confirm-EC2ProductInstance -ProductCode 774F4FF8 -InstanceId i-12345678
```

- Untuk detail API, lihat [ConfirmProductInstance](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Copy-EC2Image

Contoh kode berikut menunjukkan cara menggunakan `Copy-EC2Image`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menyalin AMI yang ditentukan di wilayah 'UE (Irlandia)' ke wilayah 'AS Barat (Oregon)'. Jika `-Region` tidak ditentukan, wilayah default saat ini digunakan sebagai wilayah tujuan.

```
Copy-EC2Image -SourceRegion eu-west-1 -SourceImageId ami-12345678 -Region us-west-2  
-Name "Copy of ami-12345678"
```

Output:

```
ami-87654321
```

- Untuk detail API, lihat [CopyImage](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Copy-EC2Snapshot

Contoh kode berikut menunjukkan cara menggunakan `Copy-EC2Snapshot`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menyalin snapshot yang ditentukan dari wilayah UE (Irlandia) ke wilayah AS Barat (Oregon).

```
Copy-EC2Snapshot -SourceRegion eu-west-1 -SourceSnapshotId snap-12345678 -Region us-west-2
```

Contoh 2: Jika Anda menetapkan wilayah default dan menghilangkan parameter Region, wilayah tujuan default adalah wilayah default.

```
Set-DefaultAWSRegion us-west-2  
Copy-EC2Snapshot -SourceRegion eu-west-1 -SourceSnapshotId snap-12345678
```

- Untuk detail API, lihat [CopySnapshot](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Deny-EC2VpcPeeringConnection

Contoh kode berikut menunjukkan cara menggunakan Deny-EC2VpcPeeringConnection.

Alat untuk PowerShell V4

Contoh 1: Contoh di atas menolak permintaan VpcPeering permintaan id pcx-01a2b3ce45fe67eb8

```
Deny-EC2VpcPeeringConnection -VpcPeeringConnectionId pcx-01a2b3ce45fe67eb8
```

- Untuk detail API, lihat [RejectVpcPeeringConnection](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Disable-EC2VgwRoutePropagation

Contoh kode berikut menunjukkan cara menggunakan Disable-EC2VgwRoutePropagation.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menonaktifkan VGW dari menyebarkan rute secara otomatis ke tabel perutean yang ditentukan.

```
Disable-EC2VgwRoutePropagation -RouteTableId rtb-12345678 -GatewayId vgw-1a2b3c4d
```

- Untuk detail API, lihat [DisableVgwRoutePropagation](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Disable-EC2VpcClassicLink

Contoh kode berikut menunjukkan cara menggunakan `Disable-EC2VpcClassicLink`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menonaktifkan EC2 VpcClassicLink untuk `vpc-01e23c4a5d6db78e9`. Ia mengembalikan `True` atau `False`

```
Disable-EC2VpcClassicLink -VpcId vpc-01e23c4a5d6db78e9
```

- Untuk detail API, lihat [DisableVpcClassicLink](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Disable-EC2VpcClassicLinkDnsSupport

Contoh kode berikut menunjukkan cara menggunakan `Disable-EC2VpcClassicLinkDnsSupport`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menonaktifkan dukungan ClassicLink DNS untuk `vpc-0b12d3456a7e8910d`

```
Disable-EC2VpcClassicLinkDnsSupport -VpcId vpc-0b12d3456a7e8910d
```

- Untuk detail API, lihat [DisableVpcClassicLinkDnsSupport](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Dismount-EC2InternetGateway

Contoh kode berikut menunjukkan cara menggunakan `Dismount-EC2InternetGateway`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini melepaskan gateway Internet yang ditentukan dari VPC yang ditentukan.

```
Dismount-EC2InternetGateway -InternetGatewayId igw-1a2b3c4d -VpcId vpc-12345678
```

- Untuk detail API, lihat [DetachInternetGateway](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Dismount-EC2NetworkInterface

Contoh kode berikut menunjukkan cara menggunakan `Dismount-EC2NetworkInterface`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus lampiran yang ditentukan antara antarmuka jaringan dan instance.

```
Dismount-EC2NetworkInterface -AttachmentId eni-attach-1a2b3c4d -Force
```

- Untuk detail API, lihat [DetachNetworkInterface](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Dismount-EC2Volume

Contoh kode berikut menunjukkan cara menggunakan `Dismount-EC2Volume`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini melepaskan volume yang ditentukan.

```
Dismount-EC2Volume -VolumeId vol-12345678
```

Output:

```
AttachTime       : 12/22/2015 1:53:58 AM
DeleteOnTermination : False
Device           : /dev/sdh
InstanceId       : i-1a2b3c4d
State            : detaching
VolumeId        : vol-12345678
```

Contoh 2: Anda juga dapat menentukan ID instans dan nama perangkat untuk memastikan bahwa Anda melepaskan volume yang benar.

```
Dismount-EC2Volume -VolumeId vol-12345678 -InstanceId i-1a2b3c4d -Device /dev/sdh
```

- Untuk detail API, lihat [DetachVolume](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Dismount-EC2VpnGateway

Contoh kode berikut menunjukkan cara menggunakan `Dismount-EC2VpnGateway`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini melepaskan gateway pribadi virtual yang ditentukan dari VPC yang ditentukan.

```
Dismount-EC2VpnGateway -VpnGatewayId vgw-1a2b3c4d -VpcId vpc-12345678
```

- Untuk detail API, lihat [DetachVpnGateway](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Edit-EC2CapacityReservation

Contoh kode berikut menunjukkan cara menggunakan `Edit-EC2CapacityReservation`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memodifikasi `CapacityReservationId` `cr-0c1f2345db6f7cdba` dengan mengubah hitungan instane menjadi 1

```
Edit-EC2CapacityReservation -CapacityReservationId cr-0c1f2345db6f7cdba -  
InstanceCount 1
```

Output:

```
True
```

- Untuk detail API, lihat [ModifyCapacityReservation](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Edit-EC2Host

Contoh kode berikut menunjukkan cara menggunakan `Edit-EC2Host`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memodifikasi `AutoPlacement` pengaturan ke off untuk host khusus `h-01e23f4cd567890f3`

```
Edit-EC2Host -HostId h-03e09f8cd681609f3 -AutoPlacement off
```

Output:

```
Successful          Unsuccessful
-----
{h-01e23f4cd567890f3} {}
```

- Untuk detail API, lihat [ModifyHosts](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Edit-EC2IdFormat

Contoh kode berikut menunjukkan cara menggunakan `Edit-EC2IdFormat`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memungkinkan format ID yang lebih panjang untuk jenis sumber daya yang ditentukan.

```
Edit-EC2IdFormat -Resource instance -UseLongId $true
```

Contoh 2: Contoh ini menonaktifkan format ID yang lebih panjang untuk jenis sumber daya yang ditentukan.

```
Edit-EC2IdFormat -Resource instance -UseLongId $false
```

- Untuk detail API, lihat [ModifyIdFormat](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Edit-EC2ImageAttribute

Contoh kode berikut menunjukkan cara menggunakan `Edit-EC2ImageAttribute`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memperbarui deskripsi untuk AMI yang ditentukan.

```
Edit-EC2ImageAttribute -ImageId ami-12345678 -Description "New description"
```

Contoh 2: Contoh ini membuat AMI menjadi publik (misalnya, jadi siapa pun Akun AWS dapat menggunakannya).

```
Edit-EC2ImageAttribute -ImageId ami-12345678 -Attribute launchPermission -  
OperationType add -UserGroup all
```

Contoh 3: Contoh ini menjadikan AMI pribadi (misalnya, sehingga hanya Anda sebagai pemilik yang dapat menggunakannya).

```
Edit-EC2ImageAttribute -ImageId ami-12345678 -Attribute launchPermission -  
OperationType remove -UserGroup all
```

Contoh 4: Contoh ini memberikan izin peluncuran ke yang ditentukan Akun AWS.

```
Edit-EC2ImageAttribute -ImageId ami-12345678 -Attribute launchPermission -  
OperationType add -UserId 111122223333
```

Contoh 5: Contoh ini menghapus izin peluncuran dari yang ditentukan Akun AWS.

```
Edit-EC2ImageAttribute -ImageId ami-12345678 -Attribute launchPermission -  
OperationType remove -UserId 111122223333
```

- Untuk detail API, lihat [ModifyImageAttribute](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Edit-EC2InstanceAttribute

Contoh kode berikut menunjukkan cara menggunakan `Edit-EC2InstanceAttribute`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memodifikasi jenis instance dari instance yang ditentukan.

```
Edit-EC2InstanceAttribute -InstanceId i-12345678 -InstanceType m3.medium
```

Contoh 2: Contoh ini memungkinkan peningkatan jaringan untuk instance tertentu, dengan menentukan “sederhana” sebagai nilai dari parameter dukungan jaringan virtualisasi I/O root tunggal (SR-IOV), `-SriovNetSupport`

```
Edit-EC2InstanceAttribute -InstanceId i-12345678 -SriovNetSupport "simple"
```

Contoh 3: Contoh ini memodifikasi grup keamanan untuk instance tertentu. Instans harus dalam VPC. Anda harus menentukan ID dari setiap grup keamanan, bukan nama.

```
Edit-EC2InstanceAttribute -InstanceId i-12345678 -Group @( "sg-12345678",  
"sg-45678901" )
```

Contoh 4: Contoh ini memungkinkan pengoptimalan EBS I/O untuk instance yang ditentukan. Fitur ini tidak tersedia dengan semua jenis instance. Biaya penggunaan tambahan berlaku saat menggunakan instans yang dioptimalkan EBS.

```
Edit-EC2InstanceAttribute -InstanceId i-12345678 -EbsOptimized $true
```

Contoh 5: Contoh ini memungkinkan pemeriksaan sumber/tujuan untuk contoh yang ditentukan. Untuk instance NAT untuk melakukan NAT, nilainya harus 'salah'.

```
Edit-EC2InstanceAttribute -InstanceId i-12345678 -SourceDestCheck $true
```

Contoh 6: Contoh ini menonaktifkan penghentian untuk contoh yang ditentukan.

```
Edit-EC2InstanceAttribute -InstanceId i-12345678 -DisableApiTermination $true
```

Contoh 7: Contoh ini mengubah instance tertentu sehingga berakhir ketika shutdown dimulai dari instance.

```
Edit-EC2InstanceAttribute -InstanceId i-12345678 -InstanceInitiatedShutdownBehavior  
terminate
```

- Untuk detail API, lihat [ModifyInstanceAttribute](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Edit-EC2InstanceCreditSpecification

Contoh kode berikut menunjukkan cara menggunakan `Edit-EC2InstanceCreditSpecification`.

Alat untuk PowerShell V4

Contoh 1: Ini memungkinkan kredit tak terbatas T2 misalnya i-01234567890abcdef.

```
$Credit = New-Object -TypeName Amazon.EC2.Model.InstanceCreditSpecificationRequest
$Credit.InstanceId = "i-01234567890abcdef"
$Credit.CpuCredits = "unlimited"
Edit-EC2InstanceCreditSpecification -InstanceCreditSpecification $Credit
```

- Untuk detail API, lihat [ModifyInstanceCreditSpecification](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Edit-EC2NetworkInterfaceAttribute

Contoh kode berikut menunjukkan cara menggunakan `Edit-EC2NetworkInterfaceAttribute`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memodifikasi antarmuka jaringan yang ditentukan sehingga lampiran yang ditentukan dihapus pada penghentian.

```
Edit-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-1a2b3c4d -
Attachment_AttachmentId eni-attach-1a2b3c4d -Attachment_DeleteOnTermination $true
```

Contoh 2: Contoh ini memodifikasi deskripsi antarmuka jaringan yang ditentukan.

```
Edit-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-1a2b3c4d -Description "my
description"
```

Contoh 3: Contoh ini memodifikasi grup keamanan untuk antarmuka jaringan yang ditentukan.

```
Edit-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-1a2b3c4d -Groups
sg-1a2b3c4d
```

Contoh 4: Contoh ini menonaktifkan pemeriksaan sumber/tujuan untuk antarmuka jaringan yang ditentukan.

```
Edit-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-1a2b3c4d -SourceDestCheck
$false
```

- Untuk detail API, lihat [ModifyNetworkInterfaceAttribute](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Edit-EC2ReservedInstance

Contoh kode berikut menunjukkan cara menggunakan `Edit-EC2ReservedInstance`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memodifikasi Availability Zone, jumlah instans, dan platform untuk instans Cadangan yang ditentukan.

```
$config = New-Object Amazon.EC2.Model.ReservedInstancesConfiguration
$config.AvailabilityZone = "us-west-2a"
$config.InstanceCount = 1
$config.Platform = "EC2-VPC"

Edit-EC2ReservedInstance `
-ReservedInstancesId @"FE32132D-70D5-4795-B400-AE435EXAMPLE", "0CC556F3-7AB8-4C00-B0E5-98666EXAMPLE" `
-TargetConfiguration $config
```

- Untuk detail API, lihat [ModifyReservedInstances](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Edit-EC2SnapshotAttribute

Contoh kode berikut menunjukkan cara menggunakan `Edit-EC2SnapshotAttribute`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat snapshot yang ditentukan publik dengan menyetel `CreateVolumePermission` atributnya.

```
Edit-EC2SnapshotAttribute -SnapshotId snap-12345678 -Attribute
CreateVolumePermission -OperationType Add -GroupName all
```

- Untuk detail API, lihat [ModifySnapshotAttribute](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Edit-EC2SpotFleetRequest

Contoh kode berikut menunjukkan cara menggunakan `Edit-EC2SpotFleetRequest`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memperbarui kapasitas target permintaan armada Spot yang ditentukan.

```
Edit-EC2SpotFleetRequest -SpotFleetRequestId sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE -TargetCapacity 10
```

Output:

```
True
```

- Untuk detail API, lihat [ModifySpotFleetRequest](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Edit-EC2SubnetAttribute

Contoh kode berikut menunjukkan cara menggunakan `Edit-EC2SubnetAttribute`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memungkinkan pengalamatan IP publik untuk subnet yang ditentukan.

```
Edit-EC2SubnetAttribute -SubnetId subnet-1a2b3c4d -MapPublicIpOnLaunch $true
```

Contoh 2: Contoh ini menonaktifkan alamat IP publik untuk subnet yang ditentukan.

```
Edit-EC2SubnetAttribute -SubnetId subnet-1a2b3c4d -MapPublicIpOnLaunch $false
```

- Untuk detail API, lihat [ModifySubnetAttribute](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Edit-EC2VolumeAttribute

Contoh kode berikut menunjukkan cara menggunakan `Edit-EC2VolumeAttribute`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memodifikasi atribut tertentu dari volume yang ditentukan. Operasi I/O untuk volume secara otomatis dilanjutkan setelah ditangguhkan karena data yang berpotensi tidak konsisten.

```
Edit-EC2VolumeAttribute -VolumeId vol-12345678 -AutoEnableIO $true
```

- Untuk detail API, lihat [ModifyVolumeAttribute](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Edit-EC2VpcAttribute

Contoh kode berikut menunjukkan cara menggunakan `Edit-EC2VpcAttribute`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memungkinkan dukungan untuk nama host DNS untuk VPC yang ditentukan.

```
Edit-EC2VpcAttribute -VpcId vpc-12345678 -EnableDnsHostnames $true
```

Contoh 2: Contoh ini menonaktifkan dukungan untuk nama host DNS untuk VPC yang ditentukan.

```
Edit-EC2VpcAttribute -VpcId vpc-12345678 -EnableDnsHostnames $false
```

Contoh 3: Contoh ini memungkinkan dukungan untuk resolusi DNS untuk VPC yang ditentukan.

```
Edit-EC2VpcAttribute -VpcId vpc-12345678 -EnableDnsSupport $true
```

Contoh 4: Contoh ini menonaktifkan dukungan untuk resolusi DNS untuk VPC yang ditentukan.

```
Edit-EC2VpcAttribute -VpcId vpc-12345678 -EnableDnsSupport $false
```

- Untuk detail API, lihat [ModifyVpcAttribute](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Enable-EC2VgwRoutePropagation

Contoh kode berikut menunjukkan cara menggunakan `Enable-EC2VgwRoutePropagation`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memungkinkan VGW yang ditentukan untuk menyebarkan rute secara otomatis ke tabel perutean yang ditentukan.

```
Enable-EC2VgwRoutePropagation -RouteTableId rtb-12345678 -GatewayId vgw-1a2b3c4d
```

- Untuk detail API, lihat [EnableVgwRoutePropagation](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Enable-EC2VolumeIO

Contoh kode berikut menunjukkan cara menggunakan `Enable-EC2VolumeIO`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memungkinkan operasi I/O untuk volume yang ditentukan, jika operasi I/O dinonaktifkan.

```
Enable-EC2VolumeIO -VolumeId vol-12345678
```

- Untuk detail API, lihat [EnableVolumeIO](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Enable-EC2VpcClassicLink

Contoh kode berikut menunjukkan cara menggunakan `Enable-EC2VpcClassicLink`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memungkinkan VPC `vpc-0123456b789b0d12f` untuk ClassicLink

```
Enable-EC2VpcClassicLink -VpcId vpc-0123456b789b0d12f
```

Output:

```
True
```

- Untuk detail API, lihat [EnableVpcClassicLink](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Enable-EC2VpcClassicLinkDnsSupport

Contoh kode berikut menunjukkan cara menggunakan `Enable-EC2VpcClassicLinkDnsSupport`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memungkinkan vpc-0b12d3456a7e8910d untuk mendukung resolusi nama host DNS untuk ClassicLink

```
Enable-EC2VpcClassicLinkDnsSupport -VpcId vpc-0b12d3456a7e8910d -Region eu-west-1
```

- Untuk detail API, lihat [EnableVpcClassicLinkDnsSupport](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2AccountAttribute

Contoh kode berikut menunjukkan cara menggunakan Get-EC2AccountAttribute.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan apakah Anda dapat meluncurkan instance ke EC2 -Classic dan EC2 -VPC di wilayah tersebut, atau hanya ke -VPC. EC2

```
(Get-EC2AccountAttribute -AttributeName supported-platforms).AttributeValues
```

Output:

```
AttributeValue
-----
EC2
VPC
```

Contoh 2: Contoh ini menjelaskan VPC default Anda, atau 'tidak ada' jika Anda tidak memiliki VPC default di wilayah tersebut.

```
(Get-EC2AccountAttribute -AttributeName default-vpc).AttributeValues
```

Output:

```
AttributeValue
-----
vpc-12345678
```

Contoh 3: Contoh ini menjelaskan jumlah maksimum instans On-Demand yang dapat Anda jalankan.

```
(Get-EC2AccountAttribute -AttributeName max-instances).AttributeValues
```

Output:

```
AttributeValue
-----
20
```

- Untuk detail API, lihat [DescribeAccountAttributes](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2Address

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2Address`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan alamat IP Elastis yang ditentukan untuk instance di EC2 - Classic.

```
Get-EC2Address -AllocationId eipalloc-12345678
```

Output:

```
AllocationId      : eipalloc-12345678
AssociationId     : eipassoc-12345678
Domain           : vpc
InstanceId        : i-87654321
NetworkInterfaceId : eni-12345678
NetworkInterfaceOwnerId : 12345678
PrivateIpAddress  : 10.0.2.172
PublicIp          : 198.51.100.2
```

Contoh 2: Contoh ini menjelaskan alamat IP Elastis Anda untuk instance di VPC. Sintaks ini membutuhkan PowerShell versi 3 atau yang lebih baru.

```
Get-EC2Address -Filter @{ Name="domain";Values="vpc" }
```

Contoh 3: Contoh ini menjelaskan alamat IP Elastis yang ditentukan untuk instance di EC2 -Classic.

```
Get-EC2Address -PublicIp 203.0.113.17
```

Output:

```
AllocationId      :
AssociationId     :
Domain           : standard
InstanceId        : i-12345678
NetworkInterfaceId :
NetworkInterfaceOwnerId :
PrivateIpAddress  :
PublicIp         : 203.0.113.17
```

Contoh 4: Contoh ini menjelaskan alamat IP Elastis Anda untuk instance di EC2 -Classic. Sintaks ini membutuhkan PowerShell versi 3 atau yang lebih baru.

```
Get-EC2Address -Filter @{ Name="domain";Values="standard" }
```

Contoh 5: Contoh ini menjelaskan semua alamat IP Elastis Anda.

```
Get-EC2Address
```

Contoh 6: Contoh ini mengembalikan IP publik dan pribadi untuk id contoh yang disediakan dalam filter

```
Get-EC2Address -Region eu-west-1 -Filter @{Name="instance-id";Values="i-0c12d3f4f567ffb89"} | Select-Object PrivateIpAddress, PublicIp
```

Output:

```
PrivateIpAddress PublicIp
-----
10.0.0.99         63.36.5.227
```

Contoh 7: Contoh ini mengambil semua Elastic IPs dengan id alokasi, id asosiasi dan id instance-nya

```
Get-EC2Address -Region eu-west-1 | Select-Object InstanceId, AssociationId,
AllocationId, PublicIp
```

Output:

InstanceId	AssociationId	AllocationId	PublicIp
-----	-----	-----	-----
		eipalloc-012e3b456789e1fad	
17.212.120.178			
i-0c123dfd3415bac67	eipassoc-0e123456bb7890bdb	eipalloc-01cd23ebf45f7890c	
17.212.124.77			
		eipalloc-012345678eeabcfad	
17.212.225.7			
i-0123d405c67e89a0c	eipassoc-0c123b456783966ba	eipalloc-0123cdd456a8f7892	
37.216.52.173			
i-0f1bf2f34c5678d09	eipassoc-0e12934568a952d96	eipalloc-0e1c23e4d5e6789e4	
37.218.222.278			
i-012e3cb4df567e8aa	eipassoc-0d1b2fa4d67d03810	eipalloc-0123f456f78a01b58	
37.210.82.27			
i-0123bcf4b567890e1	eipassoc-01d2345f678903fb1	eipalloc-0e1db23cfef5c45c7	
37.215.222.270			

Contoh 8: Contoh ini mengambil daftar alamat EC2 IP yang cocok dengan kunci tag 'Kategori' dengan nilai 'Prod'

```
Get-EC2Address -Filter @{"Name"="tag:Category";Values="Prod"}
```

Output:

```
AllocationId      : eipalloc-0123f456f81a01b58
AssociationId     : eipassoc-0d1b23a456d103810
CustomerOwnedIp   :
CustomerOwnedIpv4Pool :
Domain           : vpc
InstanceId        : i-012e3cb4df567e1aa
NetworkBorderGroup : eu-west-1
NetworkInterfaceId : eni-0123f41d5a60d5f40
NetworkInterfaceOwnerId : 123456789012
PrivateIpAddress  : 192.168.1.84
PublicIp          : 34.250.81.29
PublicIpv4Pool    : amazon
```

```
Tags : {Category, Name}
```

- Untuk detail API, lihat [DescribeAddresses](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2AvailabilityZone

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2AvailabilityZone`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan Availability Zone untuk wilayah saat ini yang tersedia untuk Anda.

```
Get-EC2AvailabilityZone
```

Output:

Messages	RegionName	State	ZoneName
-----	-----	----	-----
{}	us-west-2	available	us-west-2a
{}	us-west-2	available	us-west-2b
{}	us-west-2	available	us-west-2c

Contoh 2: Contoh ini menjelaskan Availability Zone yang berada dalam keadaan terganggu. Sintaks yang digunakan oleh contoh ini membutuhkan PowerShell versi 3 atau lebih tinggi.

```
Get-EC2AvailabilityZone -Filter @{ Name="state";Values="impaired" }
```

Contoh 3: Dengan PowerShell versi 2, Anda harus menggunakan `New-Object` untuk membuat filter.

```
$filter = New-Object Amazon.EC2.Model.Filter
$filter.Name = "state"
$filter.Values = "impaired"

Get-EC2AvailabilityZone -Filter $filter
```

- Untuk detail API, lihat [DescribeAvailabilityZones](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2BundleTask

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2BundleTask`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan tugas bundel yang ditentukan.

```
Get-EC2BundleTask -BundleId bun-12345678
```

Contoh 2: Contoh ini menjelaskan tugas bundel yang statusnya 'lengkap' atau 'gagal'.

```
$filter = New-Object Amazon.EC2.Model.Filter
$filter.Name = "state"
$filter.Values = @( "complete", "failed" )

Get-EC2BundleTask -Filter $filter
```

- Untuk detail API, lihat [DescribeBundleTasks](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2CapacityReservation

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2CapacityReservation`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan satu atau beberapa Reservasi Kapasitas Anda untuk wilayah tersebut

```
Get-EC2CapacityReservation -Region eu-west-1
```

Output:

```
AvailabilityZone      : eu-west-1b
AvailableInstanceCount : 2
CapacityReservationId : cr-0c1f2345db6f7cdba
CreateDate           : 3/28/2019 9:29:41 AM
EbsOptimized         : True
EndDate              : 1/1/0001 12:00:00 AM
EndDateType          : unlimited
```

```

EphemeralStorage      : False
InstanceMatchCriteria : open
InstancePlatform      : Windows
InstanceType          : m4.xlarge
State                 : active
Tags                  : {}
Tenancy               : default
TotalInstanceCount    : 2

```

- Untuk detail API, lihat [DescribeCapacityReservations](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2ConsoleOutput

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2ConsoleOutput`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan output konsol untuk instance Linux yang ditentukan. Output konsol dikodekan.

```
Get-EC2ConsoleOutput -InstanceId i-0e19abcd47c123456
```

Output:

InstanceId	Output
i-0e194d3c47c123637	WyAgICAwLjAwMDAwMF0gQ29tbW...bGU9dHR5UzAgc2Vs

Contoh 2: Contoh ini menyimpan output konsol yang dikodekan dalam variabel dan kemudian menerjemahkannya.

```
$Output_encoded = (Get-EC2ConsoleOutput -InstanceId i-0e19abcd47c123456).Output
[System.Text.Encoding]::UTF8.GetString([System.Convert]::FromBase64String($Output_encoded))
```

- Untuk detail API, lihat [GetConsoleOutput](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2CustomerGateway

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2CustomerGateway`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan gateway pelanggan yang ditentukan.

```
Get-EC2CustomerGateway -CustomerGatewayId cgw-1a2b3c4d
```

Output:

```
BgpAsn          : 65534
CustomerGatewayId : cgw-1a2b3c4d
IpAddress        : 203.0.113.12
State            : available
Tags             : {}
Type             : ipsec.1
```

Contoh 2: Contoh ini menjelaskan gateway pelanggan yang statusnya tertunda atau tersedia.

```
$filter = New-Object Amazon.EC2.Model.Filter
$filter.Name = "state"
$filter.Values = @( "pending", "available" )

Get-EC2CustomerGateway -Filter $filter
```

Contoh 3: Contoh ini menjelaskan semua gateway pelanggan Anda.

```
Get-EC2CustomerGateway
```

- Untuk detail API, lihat [DescribeCustomerGateways](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2DhcpOption

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2DhcpOption`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan set opsi DHCP Anda.

```
Get-EC2DhcpOption
```

Output:

DhcpConfigurations	DhcpOptionsId	Tag
-----	-----	---
{domain-name, domain-name-servers}	dopt-1a2b3c4d	{}
{domain-name, domain-name-servers}	dopt-2a3b4c5d	{}
{domain-name-servers}	dopt-3a4b5c6d	{}

Contoh 2: Contoh ini mendapatkan detail konfigurasi untuk set opsi DHCP yang ditentukan.

```
(Get-EC2DhcpOption -DhcpOptionsId dopt-1a2b3c4d).DhcpConfigurations
```

Output:

Key	Values
---	-----
domain-name	{abc.local}
domain-name-servers	{10.0.0.101, 10.0.0.102}

- Untuk detail API, lihat [DescribeDhcpOptions](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2FlowLog

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2FlowLog`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan satu atau lebih flow log dengan tipe tujuan log 's3'

```
Get-EC2FlowLog -Filter @{"Name="log-destination-type";Values="s3"}
```

Output:

```
CreationTime       : 2/25/2019 9:07:36 PM
DeliverLogsErrorMessage :
DeliverLogsPermissionArn :
DeliverLogsStatus  : SUCCESS
FlowLogId          : fl-01b2e3d45f67f8901
FlowLogStatus      : ACTIVE
LogDestination     : arn:aws:s3:::my-bucket-dd-tata
```

```
LogDestinationType      : s3
LogGroupName            :
ResourceId              : eni-01d2dda3456b7e890
TrafficType             : ALL
```

- Untuk detail API, lihat [DescribeFlowLogs](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2Host

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2Host`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan rincian EC2 host

```
Get-EC2Host
```

Output:

```
AllocationTime      : 3/23/2019 4:55:22 PM
AutoPlacement       : off
AvailabilityZone     : eu-west-1b
AvailableCapacity   : Amazon.EC2.Model.AvailableCapacity
ClientToken         :
HostId              : h-01e23f4cd567890f1
HostProperties       : Amazon.EC2.Model.HostProperties
HostReservationId   :
Instances           : {}
ReleaseTime         : 1/1/0001 12:00:00 AM
State               : available
Tags                : {}
```

Contoh 2: Contoh ini menanyakan host `AvailableInstanceCapacity` `h-01e23f4cd567899f1`

```
Get-EC2Host -HostId h-01e23f4cd567899f1 | Select-Object -ExpandProperty
AvailableCapacity | Select-Object -expand AvailableInstanceCapacity
```

Output:

```
AvailableCapacity InstanceType TotalCapacity
-----
```

11	m4.xlarge	11
----	-----------	----

- Untuk detail API, lihat [DescribeHosts](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2HostReservationOffering

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2HostReservationOffering`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan reservasi Host Khusus yang tersedia untuk dibeli untuk filter yang diberikan 'instance-family' di mana "PaymentOption NoUpfront"

```
Get-EC2HostReservationOffering -Filter @{Name="instance-family";Values="m4"} |
Where-Object PaymentOption -eq NoUpfront
```

Output:

```
CurrencyCode      :
Duration          : 94608000
HourlyPrice       : 1.307
InstanceFamily    : m4
OfferingId        : hro-0c1f234567890d9ab
PaymentOption     : NoUpfront
UpfrontPrice      : 0.000

CurrencyCode      :
Duration          : 31536000
HourlyPrice       : 1.830
InstanceFamily    : m4
OfferingId        : hro-04ad12aaaf34b5a67
PaymentOption     : NoUpfront
UpfrontPrice      : 0.000
```

- Untuk detail API, lihat [DescribeHostReservationOfferings](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2HostReservationPurchasePreview

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2HostReservationPurchasePreview`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menampilkan pratinjau pembelian reservasi dengan konfigurasi yang cocok dengan Host Khusus Anda h-01e23f4cd567890f1

```
Get-EC2HostReservationPurchasePreview -OfferingId hro-0c1f23456789d0ab -HostIdSet
h-01e23f4cd567890f1
```

Output:

CurrencyCode	Purchase	TotalHourlyPrice	TotalUpfrontPrice
-----	-----	-----	-----
	{}	1.307	0.000

- Untuk detail API, lihat [GetHostReservationPurchasePreview](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2IdFormat

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2IdFormat`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan format ID untuk jenis sumber daya yang ditentukan.

```
Get-EC2IdFormat -Resource instance
```

Output:

Resource	UseLongIds
-----	-----
instance	False

Contoh 2: Contoh ini menjelaskan format ID untuk semua jenis sumber daya yang mendukung lebih lama IDs.

```
Get-EC2IdFormat
```

Output:

Resource	UseLongIds
-----	-----
reservation	False
instance	False

- Untuk detail API, lihat [DescribeIdFormat](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2IdentityIdFormat

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2IdentityIdFormat`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan format ID untuk sumber 'image' untuk peran yang diberikan

```
Get-EC2IdentityIdFormat -PrincipalArn arn:aws:iam::123456789511:role/JDBC -Resource image
```

Output:

Deadline	Resource	UseLongIds
-----	-----	-----
8/2/2018 11:30:00 PM	image	True

- Untuk detail API, lihat [DescribeIdentityIdFormat](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2Image

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2Image`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan AMI yang ditentukan.

```
Get-EC2Image -ImageId ami-12345678
```

Output:

Architecture	: x86_64
--------------	----------

```

BlockDeviceMappings : {/dev/xvda}
CreationDate         : 2014-10-20T00:56:28.000Z
Description          : My image
Hypervisor           : xen
ImageId              : ami-12345678
ImageLocation        : 123456789012/my-image
ImageOwnerAlias      :
ImageType            : machine
KernelId             :
Name                 : my-image
OwnerId              : 123456789012
Platform             :
ProductCodes         : {}
Public               : False
RamdiskId            :
RootDeviceName       : /dev/xvda
RootDeviceType       : ebs
SriovNetSupport      : simple
State                : available
StateReason          :
Tags                 : {Name}
VirtualizationType   : hvm

```

Contoh 2: Contoh ini menggambarkan AMIs yang Anda miliki.

```
Get-EC2Image -owner self
```

Contoh 3: Contoh ini menjelaskan publik AMIs yang menjalankan Microsoft Windows Server.

```
Get-EC2Image -Filter @{ Name="platform"; Values="windows" }
```

Contoh 4: Contoh ini menjelaskan semua publik AMIs di wilayah 'us-west-2'.

```
Get-EC2Image -Region us-west-2
```

- Untuk detail API, lihat [DescribeImages](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2ImageAttribute

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2ImageAttribute`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan deskripsi untuk AMI yang ditentukan.

```
Get-EC2ImageAttribute -ImageId ami-12345678 -Attribute description
```

Output:

```
BlockDeviceMappings : {}  
Description          : My image description  
ImageId              : ami-12345678  
KernelId             :  
LaunchPermissions    : {}  
ProductCodes         : {}  
RamdiskId            :  
SriovNetSupport      :
```

Contoh 2: Contoh ini mendapatkan izin peluncuran untuk AMI yang ditentukan.

```
Get-EC2ImageAttribute -ImageId ami-12345678 -Attribute launchPermission
```

Output:

```
BlockDeviceMappings : {}  
Description          :  
ImageId              : ami-12345678  
KernelId             :  
LaunchPermissions    : {all}  
ProductCodes         : {}  
RamdiskId            :  
SriovNetSupport      :
```

Contoh 3: Contoh ini menguji apakah jaringan yang ditingkatkan diaktifkan.

```
Get-EC2ImageAttribute -ImageId ami-12345678 -Attribute sriovNetSupport
```

Output:

```
BlockDeviceMappings : {}
```

```

Description      :
ImageId          : ami-12345678
KernelId        :
LaunchPermissions : {}
ProductCodes     : {}
RamdiskId        :
SriovNetSupport  : simple

```

- Untuk detail API, lihat [DescribeImageAttribute](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2ImageByName

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2ImageByName`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan kumpulan lengkap nama filter yang saat ini didukung.

```
Get-EC2ImageByName
```

Output:

```

WINDOWS_2016_BASE
WINDOWS_2016_NANO
WINDOWS_2016_CORE
WINDOWS_2016_CONTAINER
WINDOWS_2016_SQL_SERVER_ENTERPRISE_2016
WINDOWS_2016_SQL_SERVER_STANDARD_2016
WINDOWS_2016_SQL_SERVER_WEB_2016
WINDOWS_2016_SQL_SERVER_EXPRESS_2016
WINDOWS_2012R2_BASE
WINDOWS_2012R2_CORE
WINDOWS_2012R2_SQL_SERVER_EXPRESS_2016
WINDOWS_2012R2_SQL_SERVER_STANDARD_2016
WINDOWS_2012R2_SQL_SERVER_WEB_2016
WINDOWS_2012R2_SQL_SERVER_EXPRESS_2014
WINDOWS_2012R2_SQL_SERVER_STANDARD_2014
WINDOWS_2012R2_SQL_SERVER_WEB_2014
WINDOWS_2012_BASE
WINDOWS_2012_SQL_SERVER_EXPRESS_2014
WINDOWS_2012_SQL_SERVER_STANDARD_2014

```

```

WINDOWS_2012_SQL_SERVER_WEB_2014
WINDOWS_2012_SQL_SERVER_EXPRESS_2012
WINDOWS_2012_SQL_SERVER_STANDARD_2012
WINDOWS_2012_SQL_SERVER_WEB_2012
WINDOWS_2012_SQL_SERVER_EXPRESS_2008
WINDOWS_2012_SQL_SERVER_STANDARD_2008
WINDOWS_2012_SQL_SERVER_WEB_2008
WINDOWS_2008R2_BASE
WINDOWS_2008R2_SQL_SERVER_EXPRESS_2012
WINDOWS_2008R2_SQL_SERVER_STANDARD_2012
WINDOWS_2008R2_SQL_SERVER_WEB_2012
WINDOWS_2008R2_SQL_SERVER_EXPRESS_2008
WINDOWS_2008R2_SQL_SERVER_STANDARD_2008
WINDOWS_2008R2_SQL_SERVER_WEB_2008
WINDOWS_2008RTM_BASE
WINDOWS_2008RTM_SQL_SERVER_EXPRESS_2008
WINDOWS_2008RTM_SQL_SERVER_STANDARD_2008
WINDOWS_2008_BEANSTALK_IIS75
WINDOWS_2012_BEANSTALK_IIS8
VPC_NAT

```

Contoh 2: Contoh ini menjelaskan AMI yang ditentukan. Menggunakan perintah ini untuk menemukan AMI sangat membantu karena AWS merilis Windows baru AMIs dengan pembaruan terbaru setiap bulan. Anda dapat menentukan 'ImageId' New-EC2Instance untuk meluncurkan instance menggunakan AMI saat ini untuk filter yang ditentukan.

```
Get-EC2ImageByName -Names WINDOWS_2016_BASE
```

Output:

```

Architecture      : x86_64
BlockDeviceMappings : {/dev/sda1, xvdca, xvdcb, xvdcc...}
CreationDate       : yyyy.mm.ddThh:mm:ss.000Z
Description        : Microsoft Windows Server 2016 with Desktop Experience Locale
                    English AMI provided by Amazon
Hypervisor         : xen
ImageId            : ami-xxxxxxxx
ImageLocation      : amazon/Windows_Server-2016-English-Full-Base-yyyy.mm.dd
ImageOwnerAlias     : amazon
ImageType          : machine
KernelId           :
Name               : Windows_Server-2016-English-Full-Base-yyyy.mm.dd

```

```

OwnerId      : 801119661308
Platform     : Windows
ProductCodes : {}
Public       : True
RamdiskId    :
RootDeviceName : /dev/sda1
RootDeviceType : ebs
SriovNetSupport : simple
State        : available
StateReason  :
Tags         : {}
VirtualizationType : hvm

```

- Untuk detail API, lihat [Get-EC2ImageByName](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2ImportImageTask

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2ImportImageTask`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan tugas impor gambar yang ditentukan.

```
Get-EC2ImportImageTask -ImportTaskId import-ami-hgfedcba
```

Output:

```

Architecture : x86_64
Description   : Windows Image 2
Hypervisor    :
ImageId       : ami-1a2b3c4d
ImportTaskId  : import-ami-hgfedcba
LicenseType   : AWS
Platform     : Windows
Progress      :
SnapshotDetails : {/dev/sda1}
Status        : completed
StatusMessage :

```

Contoh 2: Contoh ini menjelaskan semua tugas impor gambar Anda.

Get-EC2ImportImageTask**Output:**

```
Architecture      :  
Description       : Windows Image 1  
Hypervisor       :  
ImageId          :  
ImportTaskId     : import-ami-abcdefgh  
LicenseType      : AWS  
Platform         : Windows  
Progress         :  
SnapshotDetails  : {}  
Status           : deleted  
StatusMessage    : User initiated task cancelation
```

```
Architecture      : x86_64  
Description       : Windows Image 2  
Hypervisor       :  
ImageId          : ami-1a2b3c4d  
ImportTaskId     : import-ami-hgfedcba  
LicenseType      : AWS  
Platform         : Windows  
Progress         :  
SnapshotDetails  : {/dev/sda1}  
Status           : completed  
StatusMessage    :
```

- Untuk detail API, lihat [DescribeImportImageTasks](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2ImportSnapshotTask

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2ImportSnapshotTask`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan tugas impor snapshot yang ditentukan.

```
Get-EC2ImportSnapshotTask -ImportTaskId import-snap-abcdefgh
```

Output:

Description	ImportTaskId	SnapshotTaskDetail
-----	-----	-----
Disk Image Import 1	import-snap-abcdefgh	Amazon.EC2.Model.SnapshotTaskDetail

Contoh 2: Contoh ini menjelaskan semua tugas impor snapshot Anda.

```
Get-EC2ImportSnapshotTask
```

Output:

Description	ImportTaskId	SnapshotTaskDetail
-----	-----	-----
Disk Image Import 1	import-snap-abcdefgh	Amazon.EC2.Model.SnapshotTaskDetail
Disk Image Import 2	import-snap-hgfedcba	Amazon.EC2.Model.SnapshotTaskDetail

- Untuk detail API, lihat [DescribeImportSnapshotTasks](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2Instance

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2Instance`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan contoh yang ditentukan.

```
(Get-EC2Instance -InstanceId i-12345678).Instances
```

Output:

```
AmiLaunchIndex      : 0
Architecture        : x86_64
```

```

BlockDeviceMappings      : {/dev/sda1}
ClientToken               : TleEy1448154045270
EbsOptimized              : False
Hypervisor                : xen
IamInstanceProfile        : Amazon.EC2.Model.IamInstanceProfile
ImageId                   : ami-12345678
InstanceId                : i-12345678
InstanceLifecycle         :
InstanceType              : t2.micro
KernelId                  :
KeyName                   : my-key-pair
LaunchTime                : 12/4/2015 4:44:40 PM
Monitoring                : Amazon.EC2.Model.Monitoring
NetworkInterfaces         : {ip-10-0-2-172.us-west-2.compute.internal}
Placement                 : Amazon.EC2.Model.Placement
Platform                  : Windows
PrivateDnsName            : ip-10-0-2-172.us-west-2.compute.internal
PrivateIpAddress          : 10.0.2.172
ProductCodes              : {}
PublicDnsName             :
PublicIpAddress           :
RamdiskId                 :
RootDeviceName            : /dev/sda1
RootDeviceType            : ebs
SecurityGroups            : {default}
SourceDestCheck           : True
SpotInstanceRequestId    :
SriovNetSupport           :
State                     : Amazon.EC2.Model.InstanceState
StateReason               :
StateTransitionReason     :
SubnetId                  : subnet-12345678
Tags                      : {Name}
VirtualizationType        : hvm
VpcId                     : vpc-12345678

```

Contoh 2: Contoh ini menjelaskan semua instans Anda di wilayah saat ini, dikelompokkan berdasarkan reservasi. Untuk melihat detail instance, perluas koleksi Instances dalam setiap objek reservasi.

```
Get-EC2Instance
```

Output:

```

GroupNames      : {}
Groups          : {}
Instances       : {}
OwnerId         : 123456789012
RequesterId     : 226008221399
ReservationId   : r-c5df370c

GroupNames      : {}
Groups          : {}
Instances       : {}
OwnerId         : 123456789012
RequesterId     : 854251627541
ReservationId   : r-63e65bab
...

```

Contoh 3: Contoh ini menggambarkan penggunaan filter untuk query untuk EC2 instance dalam subnet tertentu dari VPC.

```
(Get-EC2Instance -Filter @{Name="vpc-id";Values="vpc-1a2bc34d"},@{Name="subnet-id";Values="subnet-1a2b3c4d"}).Instances
```

Output:

InstanceId	InstanceType	Platform	PrivateIpAddress	PublicIpAddress
SecurityGroups	SubnetId	VpcId		
i-01af...82cf180e19	t2.medium	Windows	10.0.0.98	...
subnet-1a2b3c4d	vpc-1a2b3c4d			
i-0374...7e9d5b0c45	t2.xlarge	Windows	10.0.0.53	...
subnet-1a2b3c4d	vpc-1a2b3c4d			

Contoh 4: Contoh ini menggambarkan penggunaan filter dengan beberapa nilai untuk query untuk EC2 instance yang keduanya berjalan dan berhenti

```

$instanceParams = @{
    Filter = @(
        @{'Name' = 'instance-state-name'; 'Values' = @("running","stopped")}
    )
}

```

```
(Get-EC2Instance @InstanceParams).Instances
```

Output:

InstanceId	InstanceType	Platform	PrivateIpAddress	PublicIpAddress
SecurityGroups	SubnetId	VpcId		
i-05a9...f6c5f46e18	t3.medium		10.0.1.7	...
subnet-1a2b3c4d	vpc-1a2b3c4d			
i-02cf...945c4fdd07	t3.medium	Windows	10.0.1.8	...
subnet-1a2b3c4d	vpc-1a2b3c4d			
i-0ac0...c037f9f3a1	t3.xlarge	Windows	10.0.1.10	...
subnet-1a2b3c4d	vpc-1a2b3c4d			
i-066b...57b7b08888	t3.medium	Windows	10.0.1.11	...
subnet-1a2b3c4d	vpc-1a2b3c4d			
i-0fee...82e83ccd72	t3.medium	Windows	10.0.1.5	...
subnet-1a2b3c4d	vpc-1a2b3c4d			
i-0a68...274cc5043b	t3.medium	Windows	10.0.1.6	...
subnet-1a2b3c4d	vpc-1a2b3c4d			

Contoh 5: Contoh ini menggambarkan penggunaan filter dengan beberapa nilai untuk kueri EC2 instance yang berjalan dan dihentikan dan menggunakan cmdlet `Select-Object` untuk memilih nilai tertentu yang akan dikeluarkan.

```
$InstanceParams = @{
    Filter = @(
        @{'Name' = 'instance-state-name'; 'Values' = @("running","stopped")}
    )
}

$SelectParams = @{
    Property = @(
        "InstanceId", "InstanceType", "Platform", "PrivateIpAddress",
        @{'Name'="Name";Expression={$_.Tags[$_].Tags.Key.IndexOf("Name")}.Value}},
        @{'Name'="State";Expression={$_.State.Name}}
    )
}

$result = Get-EC2Instance @InstanceParams
$result.Instances | Select-Object @SelectParams | Format-Table -AutoSize
```

Output:

InstanceId	InstanceType	Platform	PrivateIpAddress	Name	State
-----	-----	-----	-----	----	----
i-05a9...f6c5f46e18	t3.medium		10.0.1.7	ec2-name-01	running
i-02cf...945c4fdd07	t3.medium	Windows	10.0.1.8	ec2-name-02	stopped
i-0ac0...c037f9f3a1	t3.xlarge	Windows	10.0.1.10	ec2-name-03	running
i-066b...57b7b08888	t3.medium	Windows	10.0.1.11	ec2-name-04	stopped
i-0fee...82e83ccd72	t3.medium	Windows	10.0.1.5	ec2-name-05	running
i-0a68...274cc5043b	t3.medium	Windows	10.0.1.6	ec2-name-06	stopped

- Untuk detail API, lihat [DescribeInstances](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2InstanceAttribute

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2InstanceAttribute`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan jenis instance dari instance yang ditentukan.

```
Get-EC2InstanceAttribute -InstanceId i-12345678 -Attribute instanceType
```

Output:

```
InstanceType           : t2.micro
```

Contoh 2: Contoh ini menjelaskan apakah jaringan yang disempurnakan diaktifkan untuk instance yang ditentukan.

```
Get-EC2InstanceAttribute -InstanceId i-12345678 -Attribute sriovNetSupport
```

Output:

```
SriovNetSupport        : simple
```

Contoh 3: Contoh ini menjelaskan grup keamanan untuk instance tertentu.

```
(Get-EC2InstanceAttribute -InstanceId i-12345678 -Attribute groupSet).Groups
```

Output:

```
GroupId
-----
sg-12345678
sg-45678901
```

Contoh 4: Contoh ini menjelaskan apakah optimasi EBS diaktifkan untuk instance yang ditentukan.

```
Get-EC2InstanceAttribute -InstanceId i-12345678 -Attribute ebsOptimized
```

Output:

```
EbsOptimized           : False
```

Contoh 5: Contoh ini menjelaskan atribut disableApiTermination " dari contoh yang ditentukan.

```
Get-EC2InstanceAttribute -InstanceId i-12345678 -Attribute disableApiTermination
```

Output:

```
DisableApiTermination   : False
```

Contoh 6: Contoh ini menjelaskan atribut 'instanceInitiatedShutdownPerilaku' dari contoh yang ditentukan.

```
Get-EC2InstanceAttribute -InstanceId i-12345678 -Attribute
instanceInitiatedShutdownBehavior
```

Output:

```
InstanceInitiatedShutdownBehavior : stop
```

- Untuk detail API, lihat [DescribeInstanceAttribute](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2InstanceMetadata

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2InstanceMetadata`.

Alat untuk PowerShell V4

Contoh 1: Daftar kategori metadata instance yang tersedia yang dapat ditanyakan.

```
Get-EC2InstanceMetadata -ListCategory
```

Output:

```
AmiId
LaunchIndex
ManifestPath
AncestorAmiId
BlockDeviceMapping
InstanceId
InstanceType
LocalHostname
LocalIpv4
KernelId
AvailabilityZone
ProductCode
PublicHostname
PublicIpv4
PublicKey
RamdiskId
Region
ReservationId
SecurityGroup
UserData
InstanceMonitoring
IdentityDocument
IdentitySignature
IdentityPkcs7
```

Contoh 2: Mengembalikan id Amazon Machine Image (AMI) yang digunakan untuk meluncurkan instance.

```
Get-EC2InstanceMetadata -Category AmiId
```

Output:

```
ami-b2e756ca
```

Contoh 3: Contoh ini menanyakan dokumen identitas berformat JSON untuk instance tersebut.

```
Get-EC2InstanceMetadata -Category IdentityDocument
{
  "availabilityZone" : "us-west-2a",
  "devpayProductCodes" : null,
  "marketplaceProductCodes" : null,
  "version" : "2017-09-30",
  "instanceId" : "i-01ed50f7e2607f09e",
  "billingProducts" : [ "bp-6ba54002" ],
  "instanceType" : "t2.small",
  "pendingTime" : "2018-03-07T16:26:04Z",
  "imageId" : "ami-b2e756ca",
  "privateIp" : "10.0.0.171",
  "accountId" : "111122223333",
  "architecture" : "x86_64",
  "kernelId" : null,
  "ramdiskId" : null,
  "region" : "us-west-2"
}
```

Contoh 4: Contoh ini menggunakan kueri jalur untuk mendapatkan macs antarmuka jaringan untuk instance.

```
Get-EC2InstanceMetadata -Path "/network/interfaces/macs"
```

Output:

```
02:80:7f:ef:4c:e0/
```

Contoh 5: Jika ada peran IAM yang terkait dengan instance, mengembalikan informasi tentang terakhir kali profil instance diperbarui, termasuk LastUpdated tanggal instans, InstanceProfileArn, dan InstanceProfileId.

```
Get-EC2InstanceMetadata -Path "/iam/info"
```

Output:

```
{
  "Code" : "Success",
  "LastUpdated" : "2018-03-08T03:38:40Z",
  "InstanceProfileArn" : "arn:aws:iam::111122223333:instance-profile/
MyLaunchRole_Profile",
  "InstanceProfileId" : "AIPAI4...WVK2RW"
}
```

- Untuk detail API, lihat [Get-EC2InstanceMetadata](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2InstanceStatus

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2InstanceStatus`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan status instance yang ditentukan.

```
Get-EC2InstanceStatus -InstanceId i-12345678
```

Output:

```
AvailabilityZone : us-west-2a
Events           : {}
InstanceId       : i-12345678
InstanceState    : Amazon.EC2.Model.InstanceState
Status          : Amazon.EC2.Model.InstanceStatusSummary
SystemStatus     : Amazon.EC2.Model.InstanceStatusSummary
```

```
$status = Get-EC2InstanceStatus -InstanceId i-12345678
$status.InstanceState
```

Output:

```
Code    Name
----    -
16      running
```

```
$status.Status
```

Output:

Details	Status
-----	-----
{reachability}	ok

```
$status.SystemStatus
```

Output:

Details	Status
-----	-----
{reachability}	ok

- Untuk detail API, lihat [DescribeInstanceStatus](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2InternetGateway

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2InternetGateway`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan gateway Internet yang ditentukan.

```
Get-EC2InternetGateway -InternetGatewayId igw-1a2b3c4d
```

Output:

Attachments	InternetGatewayId	Tags
-----	-----	----
{vpc-1a2b3c4d}	igw-1a2b3c4d	{}

Contoh 2: Contoh ini menjelaskan semua gateway Internet Anda.

```
Get-EC2InternetGateway
```

Output:

Attachments	InternetGatewayId	Tags
-----	-----	----
{vpc-1a2b3c4d}	igw-1a2b3c4d	{}
{}	igw-2a3b4c5d	{}

- Untuk detail API, lihat [DescribeInternetGateways](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2KeyPair

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2KeyPair`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan key pair yang ditentukan.

```
Get-EC2KeyPair -KeyName my-key-pair
```

Output:

KeyFingerprint	KeyName
-----	-----
1f:51:ae:28:bf:89:e9:d8:1f:25:5d:37:2d:7d:b8:ca:9f:f5:f1:6f	my-key-pair

Contoh 2: Contoh ini menjelaskan semua pasangan kunci Anda.

```
Get-EC2KeyPair
```

- Untuk detail API, lihat [DescribeKeyPairs](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2NetworkAcl

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2NetworkAcl`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan ACL jaringan tertentu.

```
Get-EC2NetworkAcl -NetworkAclId acl-12345678
```

Output:

```
Associations : {aclassoc-1a2b3c4d}
Entries      : {Amazon.EC2.Model.NetworkAclEntry, Amazon.EC2.Model.NetworkAclEntry}
IsDefault    : False
NetworkAclId : acl-12345678
Tags         : {Name}
VpcId        : vpc-12345678
```

Contoh 2: Contoh ini menjelaskan aturan untuk ACL jaringan yang ditentukan.

```
(Get-EC2NetworkAcl -NetworkAclId acl-12345678).Entries
```

Output:

```
CidrBlock      : 0.0.0.0/0
Egress         : True
IcmpTypeCode    :
PortRange      :
Protocol       : -1
RuleAction     : deny
RuleNumber     : 32767

CidrBlock      : 0.0.0.0/0
Egress         : False
IcmpTypeCode    :
PortRange      :
Protocol       : -1
RuleAction     : deny
RuleNumber     : 32767
```

Contoh 3: Contoh ini menjelaskan semua jaringan Anda ACLs.

```
Get-EC2NetworkAcl
```

- Untuk detail API, lihat [DescribeNetworkAcls](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2NetworkInterface

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2NetworkInterface`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan antarmuka jaringan yang ditentukan.

```
Get-EC2NetworkInterface -NetworkInterfaceId eni-12345678
```

Output:

```
Association      :  
Attachment       : Amazon.EC2.Model.NetworkInterfaceAttachment  
AvailabilityZone  : us-west-2c  
Description      :  
Groups           : {my-security-group}  
MacAddress        : 0a:e9:a6:19:4c:7f  
NetworkInterfaceId : eni-12345678  
OwnerId          : 123456789012  
PrivateDnsName    : ip-10-0-0-107.us-west-2.compute.internal  
PrivateIpAddress  : 10.0.0.107  
PrivateIpAddresses : {ip-10-0-0-107.us-west-2.compute.internal}  
RequesterId      :  
RequesterManaged : False  
SourceDestCheck   : True  
Status            : in-use  
SubnetId          : subnet-1a2b3c4d  
TagSet            : {}  
VpcId             : vpc-12345678
```

Contoh 2: Contoh ini menjelaskan semua antarmuka jaringan Anda.

```
Get-EC2NetworkInterface
```

- Untuk detail API, lihat [DescribeNetworkInterfaces](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2NetworkInterfaceAttribute

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2NetworkInterfaceAttribute`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan antarmuka jaringan yang ditentukan.

```
Get-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-12345678 -Attribute Attachment
```

Output:

```
Attachment      : Amazon.EC2.Model.NetworkInterfaceAttachment
```

Contoh 2: Contoh ini menjelaskan antarmuka jaringan yang ditentukan.

```
Get-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-12345678 -Attribute Description
```

Output:

```
Description      : My description
```

Contoh 3: Contoh ini menjelaskan antarmuka jaringan yang ditentukan.

```
Get-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-12345678 -Attribute GroupSet
```

Output:

```
Groups           : {my-security-group}
```

Contoh 4: Contoh ini menjelaskan antarmuka jaringan yang ditentukan.

```
Get-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-12345678 -Attribute SourceDestCheck
```

Output:

```
SourceDestCheck  : True
```

- Untuk detail API, lihat [DescribeNetworkInterfaceAttribute](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2PasswordData

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2PasswordData`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendekripsi kata sandi yang EC2 ditetapkan Amazon ke akun Administrator untuk instance Windows yang ditentukan. Sebagai file PEM ditentukan, pengaturan sakelar `-Decrypt` secara otomatis diasumsikan.

```
Get-EC2PasswordData -InstanceId i-12345678 -PemFile C:\path\my-key-pair.pem
```

Output:

```
mYZ(PA9?C)Q
```

Contoh 2: (PowerShell Hanya Windows) Memeriksa instance untuk menentukan nama keypair yang digunakan untuk meluncurkan instance dan kemudian mencoba menemukan data keypair yang sesuai di penyimpanan konfigurasi Toolkit for Visual AWS Studio. Jika data keypair ditemukan, kata sandi didekripsi.

```
Get-EC2PasswordData -InstanceId i-12345678 -Decrypt
```

Output:

```
mYZ(PA9?C)Q
```

Contoh 3: Mengembalikan data kata sandi terenkripsi untuk contoh.

```
Get-EC2PasswordData -InstanceId i-12345678
```

Output:

```
iVz3BAK/WAXV.....dqt8WeMA==
```

- Untuk detail API, lihat [GetPasswordData](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2PlacementGroup

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2PlacementGroup`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan grup penempatan yang ditentukan.

```
Get-EC2PlacementGroup -GroupName my-placement-group
```

Output:

GroupName	State	Strategy
-----	-----	-----
my-placement-group	available	cluster

- Untuk detail API, lihat [DescribePlacementGroups](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2PrefixList

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2PrefixList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil yang tersedia Layanan AWS dalam format daftar awalan untuk wilayah tersebut

```
Get-EC2PrefixList
```

Output:

Cidrs	PrefixListId	PrefixListName
-----	-----	-----
{52.94.5.0/24, 52.119.240.0/21, 52.94.24.0/23}	pl-6fa54006	com.amazonaws.eu-west-1.dynamodb
{52.218.0.0/17, 54.231.128.0/19}	pl-6da54004	com.amazonaws.eu-west-1.s3

- Untuk detail API, lihat [DescribePrefixLists](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2Region

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2Region`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan wilayah yang tersedia untuk Anda.

```
Get-EC2Region
```

Output:

Endpoint	RegionName
-----	-----
ec2.eu-west-1.amazonaws.com	eu-west-1
ec2.ap-southeast-1.amazonaws.com	ap-southeast-1
ec2.ap-southeast-2.amazonaws.com	ap-southeast-2
ec2.eu-central-1.amazonaws.com	eu-central-1
ec2.ap-northeast-1.amazonaws.com	ap-northeast-1
ec2.us-east-1.amazonaws.com	us-east-1
ec2.sa-east-1.amazonaws.com	sa-east-1
ec2.us-west-1.amazonaws.com	us-west-1
ec2.us-west-2.amazonaws.com	us-west-2

- Untuk detail API, lihat [DescribeRegions](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2RouteTable

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2RouteTable`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan semua tabel rute Anda.

```
Get-EC2RouteTable
```

Output:

```
DestinationCidrBlock    : 10.0.0.0/16
DestinationPrefixListId :
GatewayId               : local
InstanceId              :
InstanceOwnerId         :
NetworkInterfaceId     :
Origin                  : CreateRouteTable
State                   : active
```

```
VpcPeeringConnectionId :
DestinationCidrBlock   : 0.0.0.0/0
DestinationPrefixListId :
GatewayId               : igw-1a2b3c4d
InstanceId              :
InstanceOwnerId         :
NetworkInterfaceId     :
Origin                  : CreateRoute
State                   : active
VpcPeeringConnectionId :
```

Contoh 2: Contoh ini mengembalikan rincian untuk tabel rute yang ditentukan.

```
Get-EC2RouteTable -RouteTableId rtb-1a2b3c4d
```

Contoh 3: Contoh ini menjelaskan tabel rute untuk VPC yang ditentukan.

```
Get-EC2RouteTable -Filter @{ Name="vpc-id"; Values="vpc-1a2b3c4d" }
```

Output:

```
Associations      : {rtbassoc-12345678}
PropagatingVgws   : {}
Routes            : {, }
RouteTableId      : rtb-1a2b3c4d
Tags              : {}
VpcId             : vpc-1a2b3c4d
```

- Untuk detail API, lihat [DescribeRouteTables](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2ScheduledInstance

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2ScheduledInstance`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan Instance Terjadwal yang ditentukan.

```
Get-EC2ScheduledInstance -ScheduledInstanceId sci-1234-1234-1234-1234-123456789012
```

Output:

```

AvailabilityZone      : us-west-2b
CreateDate            : 1/25/2016 1:43:38 PM
HourlyPrice           : 0.095
InstanceCount        : 1
InstanceType          : c4.large
NetworkPlatform      : EC2-VPC
NextSlotStartTime     : 1/31/2016 1:00:00 AM
Platform             : Linux/UNIX
PreviousSlotEndTime   :
Recurrence            : Amazon.EC2.Model.ScheduledInstanceRecurrence
ScheduledInstanceId   : sci-1234-1234-1234-1234-123456789012
SlotDurationInHours   : 32
TermEndDate           : 1/31/2017 1:00:00 AM
TermStartDate         : 1/31/2016 1:00:00 AM
TotalScheduledInstanceHours : 1696

```

Contoh 2: Contoh ini menjelaskan semua Instans Terjadwal Anda.

```
Get-EC2ScheduledInstance
```

- Untuk detail API, lihat [DescribeScheduledInstances](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2ScheduledInstanceAvailability

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2ScheduledInstanceAvailability`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan jadwal yang terjadi setiap minggu pada hari Minggu, dimulai pada tanggal yang ditentukan.

```

Get-EC2ScheduledInstanceAvailability -Recurrence_Frequency
Weekly -Recurrence_Interval 1 -Recurrence_OccurrenceDay 1 -
FirstSlotStartTimeRange_EarliestTime 2016-01-31T00:00:00Z -
FirstSlotStartTimeRange_LatestTime 2016-01-31T04:00:00Z

```

Output:

```

AvailabilityZone      : us-west-2b
AvailableInstanceCount : 20
FirstSlotStartTime    : 1/31/2016 8:00:00 AM
HourlyPrice           : 0.095
InstanceType          : c4.large
MaxTermDurationInDays : 366
MinTermDurationInDays : 366
NetworkPlatform       : EC2-VPC
Platform              : Linux/UNIX
PurchaseToken         : eyJ2IjoiMSIsInMiOjEsImMiOi...
Recurrence            : Amazon.EC2.Model.ScheduledInstanceRecurrence
SlotDurationInHours   : 23
TotalScheduledInstanceHours : 1219

...

```

Contoh 2: Untuk mempersempit hasil, Anda dapat menambahkan filter untuk kriteria seperti sistem operasi, jaringan, dan jenis instance.

```
-Filter @{ Name="platform";Values="Linux/UNIX" },@{ Name="network-
platform";Values="EC2-VPC" },@{ Name="instance-type";Values="c4.large" }
```

- Untuk detail API, lihat [DescribeScheduledInstanceAvailability](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2SecurityGroup

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2SecurityGroup`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan grup keamanan yang ditentukan untuk VPC. Saat bekerja dengan grup keamanan milik VPC, Anda harus menggunakan ID grup keamanan (`-GroupId` parameter), bukan nama (`-GroupName` parameter), untuk mereferensikan grup.

```
Get-EC2SecurityGroup -GroupId sg-12345678
```

Output:

```
Description      : default VPC security group
```

```

GroupId       : sg-12345678
GroupName     : default
IpPermissions : {Amazon.EC2.Model.IpPermission}
IpPermissionsEgress : {Amazon.EC2.Model.IpPermission}
OwnerId      : 123456789012
Tags         : {}
VpcId        : vpc-12345678

```

Contoh 2: Contoh ini menjelaskan grup keamanan yang ditentukan untuk EC2 -Classic. Saat bekerja dengan grup keamanan untuk EC2 -Classic Anda dapat menggunakan nama grup (- GroupName parameter) atau ID grup (- GroupId parameter) untuk mereferensikan grup keamanan.

```
Get-EC2SecurityGroup -GroupName my-security-group
```

Output:

```

Description   : my security group
GroupId       : sg-45678901
GroupName     : my-security-group
IpPermissions : {Amazon.EC2.Model.IpPermission, Amazon.EC2.Model.IpPermission}
IpPermissionsEgress : {}
OwnerId      : 123456789012
Tags         : {}
VpcId        :

```

Contoh 3: Contoh ini mengambil semua grup keamanan untuk vpc-0fc1ff23456b789eb

```
Get-EC2SecurityGroup -Filter @{"Name"="vpc-id";Values="vpc-0fc1ff23456b789eb"}
```

- Untuk detail API, lihat [DescribeSecurityGroups](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2Snapshot

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2Snapshot`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan snapshot yang ditentukan.

```
Get-EC2Snapshot -SnapshotId snap-12345678
```

Output:

```
DataEncryptionKeyId :
Description          : Created by CreateImage(i-1a2b3c4d) for ami-12345678 from
                        vol-12345678
Encrypted            : False
KmsKeyId             :
OwnerAlias           :
OwnerId              : 123456789012
Progress             : 100%
SnapshotId           : snap-12345678
StartTime            : 10/23/2014 6:01:28 AM
State                : completed
StateMessage         :
Tags                 : {}
VolumeId             : vol-12345678
VolumeSize           : 8
```

Contoh 2: Contoh ini menjelaskan snapshot yang memiliki tag 'Nama'.

```
Get-EC2Snapshot | ? { $_.Tags.Count -gt 0 -and $_.Tags.Key -eq "Name" }
```

Contoh 3: Contoh ini menjelaskan snapshot yang memiliki tag 'Nama' dengan nilai 'TestValue'.

```
Get-EC2Snapshot | ? { $_.Tags.Count -gt 0 -and $_.Tags.Key -eq "Name" -and
                        $_.Tags.Value -eq "TestValue" }
```

Contoh 4: Contoh ini menjelaskan semua snapshot Anda.

```
Get-EC2Snapshot -Owner self
```

- Untuk detail API, lihat [DescribeSnapshots](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2SnapshotAttribute

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2SnapshotAttribute`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan atribut tertentu dari snapshot yang ditentukan.

```
Get-EC2SnapshotAttribute -SnapshotId snap-12345678 -Attribute ProductCodes
```

Output:

CreateVolumePermissions	ProductCodes	SnapshotId
-----	-----	-----
{}	{}	snap-12345678

Contoh 2: Contoh ini menjelaskan atribut tertentu dari snapshot yang ditentukan.

```
(Get-EC2SnapshotAttribute -SnapshotId snap-12345678 -Attribute
CreateVolumePermission).CreateVolumePermissions
```

Output:

Group	UserId
-----	-----
all	

- Untuk detail API, lihat [DescribeSnapshotAttribute](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2SpotDatafeedSubscription

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2SpotDatafeedSubscription`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan umpan data instans Spot Anda.

```
Get-EC2SpotDatafeedSubscription
```

Output:

```
Bucket   : my-s3-bucket
Fault    :
OwnerId  : 123456789012
```

```
Prefix : spotdata
State  : Active
```

- Untuk detail API, lihat [DescribeSpotDatafeedSubscription](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2SpotFleetInstance

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2SpotFleetInstance`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan instance yang terkait dengan permintaan armada Spot yang ditentukan.

```
Get-EC2SpotFleetInstance -SpotFleetRequestId sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE
```

Output:

InstanceId	InstanceType	SpotInstanceRequestId
-----	-----	-----
i-f089262a	c3.large	sir-12345678
i-7e8b24a4	c3.large	sir-87654321

- Untuk detail API, lihat [DescribeSpotFleetInstances](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2SpotFleetRequest

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2SpotFleetRequest`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan permintaan armada Spot yang ditentukan.

```
Get-EC2SpotFleetRequest -SpotFleetRequestId sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE
| format-list
```

Output:

```

ConfigData          : Amazon.EC2.Model.SpotFleetRequestConfigData
CreateTime          : 12/26/2015 8:23:33 AM
SpotFleetRequestId  : sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE
SpotFleetRequestState : active

```

Contoh 2: Contoh ini menjelaskan semua permintaan armada Spot Anda.

```
Get-EC2SpotFleetRequest
```

- Untuk detail API, lihat [DescribeSpotFleetRequests](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2SpotFleetRequestHistory

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2SpotFleetRequestHistory`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan riwayat permintaan armada Spot yang ditentukan.

```

Get-EC2SpotFleetRequestHistory -SpotFleetRequestId sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE -StartTime 2015-12-26T00:00:00Z

```

Output:

```

HistoryRecords      : {Amazon.EC2.Model.HistoryRecord,
  Amazon.EC2.Model.HistoryRecord...}
LastEvaluatedTime   : 12/26/2015 8:29:11 AM
NextToken           :
SpotFleetRequestId  : sfr-088bc5f1-7e7b-451a-bd13-757f10672b93
StartTime           : 12/25/2015 8:00:00 AM

```

```

(Get-EC2SpotFleetRequestHistory -SpotFleetRequestId sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE -StartTime 2015-12-26T00:00:00Z).HistoryRecords

```

Output:

EventInformation	EventType	Timestamp
-----	-----	-----

Amazon.EC2.Model.EventInformation	fleetRequestChange	12/26/2015 8:23:33 AM
Amazon.EC2.Model.EventInformation	fleetRequestChange	12/26/2015 8:23:33 AM
Amazon.EC2.Model.EventInformation	fleetRequestChange	12/26/2015 8:23:33 AM
Amazon.EC2.Model.EventInformation	launched	12/26/2015 8:25:34 AM
Amazon.EC2.Model.EventInformation	launched	12/26/2015 8:25:05 AM

- Untuk detail API, lihat [DescribeSpotFleetRequestHistory](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2SpotInstanceRequest

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2SpotInstanceRequest`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan permintaan instance Spot yang ditentukan.

```
Get-EC2SpotInstanceRequest -SpotInstanceRequestId sir-12345678
```

Output:

```
ActualBlockHourlyPrice :
AvailabilityZoneGroup   :
BlockDurationMinutes    : 0
CreateTime              : 4/8/2015 2:51:33 PM
Fault                  :
InstanceId              : i-12345678
LaunchedAvailabilityZone : us-west-2b
LaunchGroup            :
LaunchSpecification     : Amazon.EC2.Model.LaunchSpecification
ProductDescription      : Linux/UNIX
SpotInstanceRequestId   : sir-12345678
SpotPrice               : 0.020000
State                  : active
Status                 : Amazon.EC2.Model.SpotInstanceStatus
Tags                   : {Name}
Type                   : one-time
```

Contoh 2: Contoh ini menjelaskan semua permintaan instans Spot Anda.

```
Get-EC2SpotInstanceRequest
```

- Untuk detail API, lihat [DescribeSpotInstanceRequests](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2SpotPriceHistory

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2SpotPriceHistory`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan 10 entri terakhir dalam riwayat harga Spot untuk jenis instans tertentu dan Availability Zone. Perhatikan bahwa nilai yang ditentukan untuk AvailabilityZone parameter - harus valid untuk nilai wilayah yang diberikan ke parameter -Region cmdlet (tidak ditampilkan dalam contoh) atau ditetapkan sebagai default di shell. Perintah contoh ini mengasumsikan wilayah default 'us-west-2' telah disetel di lingkungan.

```
Get-EC2SpotPriceHistory -InstanceType c3.large -AvailabilityZone us-west-2a -
MaxResult 10
```

Output:

```
AvailabilityZone : us-west-2a
InstanceType     : c3.large
Price            : 0.017300
ProductDescription : Linux/UNIX (Amazon VPC)
Timestamp        : 12/25/2015 7:39:49 AM

AvailabilityZone : us-west-2a
InstanceType     : c3.large
Price            : 0.017200
ProductDescription : Linux/UNIX (Amazon VPC)
Timestamp        : 12/25/2015 7:38:29 AM

AvailabilityZone : us-west-2a
InstanceType     : c3.large
Price            : 0.017300
ProductDescription : Linux/UNIX (Amazon VPC)
Timestamp        : 12/25/2015 6:57:13 AM
...
```

- Untuk detail API, lihat [DescribeSpotPriceHistory](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2Subnet

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2Subnet`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan subnet yang ditentukan.

```
Get-EC2Subnet -SubnetId subnet-1a2b3c4d
```

Output:

```
AvailabilityZone      : us-west-2c
AvailableIpAddressCount : 251
CidrBlock             : 10.0.0.0/24
DefaultForAz          : False
MapPublicIpOnLaunch   : False
State                 : available
SubnetId              : subnet-1a2b3c4d
Tags                  : {}
VpcId                 : vpc-12345678
```

Contoh 2: Contoh ini menjelaskan semua subnet Anda.

```
Get-EC2Subnet
```

- Untuk detail API, lihat [DescribeSubnets](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2Tag

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2Tag`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil tag untuk 'image' tipe sumber daya

```
Get-EC2Tag -Filter @{"Name"="resource-type";Values="image"}
```

Output:

Key	ResourceId	ResourceType	Value
---	-----	-----	----
Name	ami-0a123b4ccb567a8ea	image	Win7-Imported
auto-delete	ami-0a123b4ccb567a8ea	image	never

Contoh 2: Contoh ini mengambil semua tag untuk semua sumber daya dan mengelompokkannya berdasarkan jenis sumber daya

```
Get-EC2Tag | Group-Object resourcetype
```

Output:

```
Count Name
-----
    9 subnet
    53 instance
    3 route-table
    5 security-group
   30 volume
    1 internet-gateway
    3 network-interface
    4 elastic-ip
    1 dhcp-options
    2 image
    3 vpc

Group
-----
{Amazon.EC2.Model.TagDescription,
 Amazon.EC2.Model.TagDescription, Amazon.EC2.Model.TagDescription,
 Amazon.EC2.Model.TagDescription...}
{Amazon.EC2.Model.TagDescription,
 Amazon.EC2.Model.TagDescription, Amazon.EC2.Model.TagDescription,
 Amazon.EC2.Model.TagDescription...}
{Amazon.EC2.Model.TagDescription,
 Amazon.EC2.Model.TagDescription, Amazon.EC2.Model.TagDescription}
{Amazon.EC2.Model.TagDescription,
 Amazon.EC2.Model.TagDescription, Amazon.EC2.Model.TagDescription,
 Amazon.EC2.Model.TagDescription...}
{Amazon.EC2.Model.TagDescription,
 Amazon.EC2.Model.TagDescription, Amazon.EC2.Model.TagDescription,
 Amazon.EC2.Model.TagDescription...}
{Amazon.EC2.Model.TagDescription}
{Amazon.EC2.Model.TagDescription,
 Amazon.EC2.Model.TagDescription, Amazon.EC2.Model.TagDescription}
{Amazon.EC2.Model.TagDescription,
 Amazon.EC2.Model.TagDescription, Amazon.EC2.Model.TagDescription,
 Amazon.EC2.Model.TagDescription}
{Amazon.EC2.Model.TagDescription}
{Amazon.EC2.Model.TagDescription,
 Amazon.EC2.Model.TagDescription}
{Amazon.EC2.Model.TagDescription,
 Amazon.EC2.Model.TagDescription, Amazon.EC2.Model.TagDescription}
```

Contoh 3: Contoh ini menampilkan semua sumber daya dengan tag 'auto-delete' dengan nilai 'no' untuk wilayah tertentu

```
Get-EC2Tag -Region eu-west-1 -Filter @{"Name":"tag:auto-delete";Values="no"}
```

Output:

Key	ResourceId	ResourceType	Value
---	-----	-----	----
auto-delete	i-0f1bce234d5dd678b	instance	no
auto-delete	vol-01d234aa5678901a2	volume	no
auto-delete	vol-01234bfb5def6f7b8	volume	no
auto-delete	vol-01ccb23f4c5e67890	volume	no

Contoh 4: Contoh ini memperoleh semua sumber daya dengan tag 'hapus otomatis' dengan nilai 'no' dan filter lebih lanjut di pipa berikutnya untuk mengurai hanya jenis sumber daya 'instance' dan akhirnya membuat tag 'ThisInstance' untuk setiap sumber daya instance dengan nilai menjadi id instance itu sendiri

```
Get-EC2Tag -Region eu-west-1 -Filter @{"Name":"tag:auto-delete";Values="no"} |
Where-Object ResourceType -eq "instance" | ForEach-Object {New-EC2Tag -ResourceId
$_.ResourceId -Tag @{"Key":"ThisInstance";Value=$_.ResourceId}}
```

Contoh 5: Contoh ini mengambil tag untuk semua sumber daya instance serta kunci 'Nama' dan menampilkannya dalam format tabel

```
Get-EC2Tag -Filter @{"Name":"resource-
type";Values="instance"},@{"Name":"key";Values="Name"} | Select-Object ResourceId,
@{"Name":"Name-Tag";Expression={$PSItem.Value}} | Format-Table -AutoSize
```

Output:

ResourceId	Name-Tag
-----	-----
i-012e3cb4df567e1aa	jump1
i-01c23a45d6fc7a89f	repro-3

- Untuk detail API, lihat [DescribeTags](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2Volume

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2Volume`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan volume EBS yang ditentukan.

```
Get-EC2Volume -VolumeId vol-12345678
```

Output:

```
Attachments      : {}
AvailabilityZone  : us-west-2c
CreateTime       : 7/17/2015 4:35:19 PM
Encrypted        : False
Iops             : 90
KmsKeyId         :
Size            : 30
SnapshotId       : snap-12345678
State            : in-use
Tags            : {}
VolumeId         : vol-12345678
VolumeType       : standard
```

Contoh 2: Contoh ini menjelaskan volume EBS Anda yang memiliki status 'tersedia'.

```
Get-EC2Volume -Filter @{ Name="status"; Values="available" }
```

Output:

```
Attachments      : {}
AvailabilityZone  : us-west-2c
CreateTime       : 12/21/2015 2:31:29 PM
Encrypted        : False
Iops             : 60
KmsKeyId         :
Size            : 20
SnapshotId       : snap-12345678
State            : available
Tags            : {}
VolumeId         : vol-12345678
VolumeType       : gp2
...
```

Contoh 3: Contoh ini menjelaskan semua volume EBS Anda.

```
Get-EC2Volume
```

- Untuk detail API, lihat [DescribeVolumes](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2VolumeAttribute

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2VolumeAttribute`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan atribut tertentu dari volume yang ditentukan.

```
Get-EC2VolumeAttribute -VolumeId vol-12345678 -Attribute AutoEnableIO
```

Output:

AutoEnableIO	ProductCodes	VolumeId
-----	-----	-----
False	{}	vol-12345678

- Untuk detail API, lihat [DescribeVolumeAttribute](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2VolumeStatus

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2VolumeStatus`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan status volume yang ditentukan.

```
Get-EC2VolumeStatus -VolumeId vol-12345678
```

Output:

```
Actions          : {}
AvailabilityZone  : us-west-2a
Events           : {}
VolumeId         : vol-12345678
```

```
VolumeStatus      : Amazon.EC2.Model.VolumeStatusInfo
```

```
(Get-EC2VolumeStatus -VolumeId vol-12345678).VolumeStatus
```

Output:

```
Details              Status
-----
{io-enabled, io-performance}  ok
```

```
(Get-EC2VolumeStatus -VolumeId vol-12345678).VolumeStatus.Details
```

Output:

```
Name              Status
----
io-enabled         passed
io-performance     not-applicable
```

- Untuk detail API, lihat [DescribeVolumeStatus](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2Vpc

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2Vpc`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan VPC yang ditentukan.

```
Get-EC2Vpc -VpcId vpc-12345678
```

Output:

```
CidrBlock      : 10.0.0.0/16
DhcpOptionsId  : dopt-1a2b3c4d
InstanceTenancy : default
IsDefault      : False
State          : available
Tags           : {Name}
```

```
VpcId          : vpc-12345678
```

Contoh 2: Contoh ini menjelaskan VPC default (hanya ada satu per wilayah). Jika akun Anda mendukung EC2 -Classic di wilayah ini, tidak ada VPC default.

```
Get-EC2Vpc -Filter @{"Name"="isDefault"; Values="true"}
```

Output:

```
CidrBlock      : 172.31.0.0/16
DhcpOptionsId  : dopt-12345678
InstanceTenancy : default
IsDefault      : True
State          : available
Tags           : {}
VpcId          : vpc-45678901
```

Contoh 3: Contoh ini menjelaskan VPCs yang cocok dengan filter yang ditentukan (yaitu, memiliki CIDR yang cocok dengan nilai '10.0.0.0/16' dan berada dalam keadaan 'tersedia').

```
Get-EC2Vpc -Filter @{"Name"="cidr";
  Values="10.0.0.0/16"},@{"Name"="state";Values="available"}
```

Contoh 4: Contoh ini menjelaskan semua Anda VPCs.

```
Get-EC2Vpc
```

- Untuk detail API, lihat [DescribeVpcs](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2VpcAttribute

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2VpcAttribute`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan atribut `enableDnsSupport` ".

```
Get-EC2VpcAttribute -VpcId vpc-12345678 -Attribute enableDnsSupport
```

Output:

```
EnableDnsSupport
-----
True
```

Contoh 2: Contoh ini menjelaskan atribut enableDnsHostnames ".

```
Get-EC2VpcAttribute -VpcId vpc-12345678 -Attribute enableDnsHostnames
```

Output:

```
EnableDnsHostnames
-----
True
```

- Untuk detail API, lihat [DescribeVpcAttribute](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2VpcClassicLink

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2VpcClassicLink`.

Alat untuk PowerShell V4

Contoh 1: Contoh di atas mengembalikan semua VPCs dengan `ClassicLinkEnabled` negara mereka untuk wilayah

```
Get-EC2VpcClassicLink -Region eu-west-1
```

Output:

ClassicLinkEnabled	Tags	VpcId
-----	----	-----
False	{Name}	vpc-0fc1ff23f45b678eb
False	{}	vpc-01e23c4a5d6db78e9
False	{Name}	vpc-0123456b078b9d01f
False	{}	vpc-12cf3b4f
False	{Name}	vpc-0b12d3456a7e8901d

- Untuk detail API, lihat [DescribeVpcClassicLink](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2VpcClassicLinkDnsSupport

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2VpcClassicLinkDnsSupport`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan status dukungan ClassicLink DNS VPCs untuk wilayah eu-west-1

```
Get-EC2VpcClassicLinkDnsSupport -VpcId vpc-0b12d3456a7e8910d -Region eu-west-1
```

Output:

ClassicLinkDnsSupported	VpcId
False	vpc-0b12d3456a7e8910d
False	vpc-12cf3b4f

- Untuk detail API, lihat [DescribeVpcClassicLinkDnsSupport](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2VpcEndpoint

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2VpcEndpoint`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan satu atau beberapa titik akhir VPC Anda untuk wilayah eu-west-1. Kemudian pipa output ke perintah berikutnya, yang memilih `VpcEndpointId` properti dan mengembalikan array VPC ID sebagai string array

```
Get-EC2VpcEndpoint -Region eu-west-1 | Select-Object -ExpandProperty VpcEndpointId
```

Output:

```
vpce-01a2ab3f4f5cc6f7d
vpce-01d2b345a6787890b
vpce-0012e34d567890e12
vpce-0c123db4567890123
```

Contoh 2: Contoh ini menjelaskan semua titik akhir vpc untuk wilayah eu-west-1 dan memilih,, dan properti untuk menyajikannya dalam format tabel VpcEndpointId VpcId ServiceName PrivateDnsEnabled

```
Get-EC2VpcEndpoint -Region eu-west-1 | Select-Object VpcEndpointId, VpcId,
ServiceName, PrivateDnsEnabled | Format-Table -AutoSize
```

Output:

VpcEndpointId	VpcId	ServiceName	PrivateDnsEnabled
-----	-----	-----	-----
vpce-02a2ab2f2f2cc2f2d	vpc-0fc6ff46f65b039eb	com.amazonaws.eu-west-1.ssm	True
vpce-01d1b111a1114561b	vpc-0fc6ff46f65b039eb	com.amazonaws.eu-west-1.ec2	True
vpce-0011e23d45167e838	vpc-0fc6ff46f65b039eb	com.amazonaws.eu-west-1.ec2messages	True
vpce-0c123db4567890123	vpc-0fc6ff46f65b039eb	com.amazonaws.eu-west-1.ssmmessages	True

Contoh 3: Contoh ini mengeksplor dokumen kebijakan untuk VPC Endpoint vpce-01a2ab3f4f5cc6f7d ke dalam file json

```
Get-EC2VpcEndpoint -Region eu-west-1 -VpcEndpointId vpce-01a2ab3f4f5cc6f7d | Select-Object -expand PolicyDocument | Out-File vpce_policyDocument.json
```

- Untuk detail API, lihat [DescribeVpcEndpoints](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2VpcEndpointService

Contoh kode berikut menunjukkan cara menggunakan Get-EC2VpcEndpointService.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan layanan titik akhir EC2 VPC dengan filter yang diberikan, dalam hal ini com.amazonaws.eu-west-1.ecs. Selanjutnya, itu juga memperluas ServiceDetails properti dan menampilkan detailnya

```
Get-EC2VpcEndpointService -Region eu-west-1 -MaxResult 5 -Filter @{"Name"="service-name";Values="com.amazonaws.eu-west-1.ecs"} | Select-Object -ExpandProperty ServiceDetails
```

Output:

```
AcceptanceRequired      : False
AvailabilityZones       : {eu-west-1a, eu-west-1b, eu-west-1c}
BaseEndpointDnsNames    : {ecs.eu-west-1.vpce.amazonaws.com}
Owner                   : amazon
PrivateDnsName          : ecs.eu-west-1.amazonaws.com
ServiceName             : com.amazonaws.eu-west-1.ecs
ServiceType             : {Amazon.EC2.Model.ServiceTypeDetail}
VpcEndpointPolicySupported : False
```

Contoh 2: Contoh ini mengambil semua layanan EC2 VPC Endpoint dan mengembalikan “ssm” ServiceNames yang cocok

```
Get-EC2VpcEndpointService -Region eu-west-1 | Select-Object -ExpandProperty Servicenames | Where-Object { -match "ssm"}
```

Output:

```
com.amazonaws.eu-west-1.ssm
com.amazonaws.eu-west-1.ssmmessages
```

- Untuk detail API, lihat [DescribeVpcEndpointServices](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2VpnConnection

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2VpnConnection`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan koneksi VPN yang ditentukan.

```
Get-EC2VpnConnection -VpnConnectionId vpn-12345678
```

Output:

```

CustomerGatewayConfiguration : [XML document]
CustomerGatewayId            : cgw-1a2b3c4d
Options                      : Amazon.EC2.Model.VpnConnectionOptions
Routes                      : {Amazon.EC2.Model.VpnStaticRoute}
State                        : available
Tags                         : {}
Type                         : ipsec.1
VgwTelemetry                 : {Amazon.EC2.Model.VgwTelemetry,
    Amazon.EC2.Model.VgwTelemetry}
VpnConnectionId             : vpn-12345678
VpnGatewayId                 : vgw-1a2b3c4d

```

Contoh 2: Contoh ini menjelaskan koneksi VPN apa pun yang statusnya tertunda atau tersedia.

```

$filter = New-Object Amazon.EC2.Model.Filter
$filter.Name = "state"
$filter.Values = @( "pending", "available" )

Get-EC2VpnConnection -Filter $filter

```

Contoh 3: Contoh ini menjelaskan semua koneksi VPN Anda.

```
Get-EC2VpnConnection
```

- Untuk detail API, lihat [DescribeVpnConnections](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EC2VpnGateway

Contoh kode berikut menunjukkan cara menggunakan `Get-EC2VpnGateway`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan gateway pribadi virtual yang ditentukan.

```
Get-EC2VpnGateway -VpnGatewayId vgw-1a2b3c4d
```

Output:

```

AvailabilityZone :
State            : available

```

```
Tags           : {}
Type           : ipsec.1
VpcAttachments : {vpc-12345678}
VpnGatewayId   : vgw-1a2b3c4d
```

Contoh 2: Contoh ini menjelaskan gateway pribadi virtual yang statusnya tertunda atau tersedia.

```
$filter = New-Object Amazon.EC2.Model.Filter
$filter.Name = "state"
$filter.Values = @( "pending", "available" )

Get-EC2VpnGateway -Filter $filter
```

Contoh 3: Contoh ini menjelaskan semua gateway pribadi virtual Anda.

```
Get-EC2VpnGateway
```

- Untuk detail API, lihat [DescribeVpnGateways](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Grant-EC2SecurityGroupEgress

Contoh kode berikut menunjukkan cara menggunakan `Grant-EC2SecurityGroupEgress`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendefinisikan aturan jalan keluar untuk grup keamanan tertentu untuk - VPC. EC2 Aturan memberikan akses ke rentang alamat IP yang ditentukan pada port TCP 80. Sintaks yang digunakan oleh contoh ini membutuhkan PowerShell versi 3 atau lebih tinggi.

```
$ip = @{ IpProtocol="tcp"; FromPort="80"; ToPort="80"; IpRanges="203.0.113.0/24" }
Grant-EC2SecurityGroupEgress -GroupId sg-12345678 -IpPermission $ip
```

Contoh 2: Dengan PowerShell versi 2, Anda harus menggunakan `New-Object` untuk membuat objek. `IpPermission`

```
$ip = New-Object Amazon.EC2.Model.IpPermission
$ip.IpProtocol = "tcp"
$ip.FromPort = 80
$ip.ToPort = 80
$ip.IpRanges.Add("203.0.113.0/24")
```

```
Grant-EC2SecurityGroupEgress -GroupId sg-12345678 -IpPermission $ip
```

Contoh 3: Contoh ini memberikan akses ke grup keamanan sumber yang ditentukan pada port TCP 80.

```
$ug = New-Object Amazon.EC2.Model.UserIdGroupPair
$ug.GroupId = "sg-1a2b3c4d"
$ug.UserId = "123456789012"

Grant-EC2SecurityGroupEgress -GroupId sg-12345678 -IpPermission
@( @{ IpProtocol="tcp"; FromPort="80"; ToPort="80"; UserIdGroupPairs=$ug } )
```

- Untuk detail API, lihat [AuthorizeSecurityGroupEgress](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Grant-EC2SecurityGroupIngress

Contoh kode berikut menunjukkan cara menggunakan `Grant-EC2SecurityGroupIngress`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendefinisikan aturan masuk untuk grup keamanan untuk -VPC. EC2 Aturan ini memberikan akses ke alamat IP tertentu untuk SSH (port 22) dan RDC (port 3389). Perhatikan bahwa Anda harus mengidentifikasi grup keamanan untuk EC2 -VPC menggunakan ID grup keamanan bukan nama grup keamanan. Sintaks yang digunakan oleh contoh ini membutuhkan PowerShell versi 3 atau lebih tinggi.

```
$ip1 = @{ IpProtocol="tcp"; FromPort="22"; ToPort="22"; IpRanges="203.0.113.25/32" }
$ip2 = @{ IpProtocol="tcp"; FromPort="3389"; ToPort="3389";
  IpRanges="203.0.113.25/32" }

Grant-EC2SecurityGroupIngress -GroupId sg-12345678 -IpPermission @( $ip1, $ip2 )
```

Contoh 2: Dengan PowerShell versi 2, Anda harus menggunakan `New-Object` untuk membuat objek. `IpPermission`

```
$ip1 = New-Object Amazon.EC2.Model.IpPermission
$ip1.IpProtocol = "tcp"
$ip1.FromPort = 22
```

```
$ip1.ToPort = 22
$ip1.IpRanges.Add("203.0.113.25/32")

$ip2 = new-object Amazon.EC2.Model.IpPermission
$ip2.IpProtocol = "tcp"
$ip2.FromPort = 3389
$ip2.ToPort = 3389
$ip2.IpRanges.Add("203.0.113.25/32")

Grant-EC2SecurityGroupIngress -GroupId sg-12345678 -IpPermission @( $ip1, $ip2 )
```

Contoh 3: Contoh ini mendefinisikan aturan ingress untuk grup keamanan untuk -Classic. EC2 Aturan ini memberikan akses ke alamat IP tertentu untuk SSH (port 22) dan RDC (port 3389). Sintaks yang digunakan oleh contoh ini membutuhkan PowerShell versi 3 atau lebih tinggi.

```
$ip1 = @{ IpProtocol="tcp"; FromPort="22"; ToPort="22"; IpRanges="203.0.113.25/32" }
$ip2 = @{ IpProtocol="tcp"; FromPort="3389"; ToPort="3389";
  IpRanges="203.0.113.25/32" }

Grant-EC2SecurityGroupIngress -GroupName "my-security-group" -IpPermission @( $ip1,
  $ip2 )
```

Contoh 4: Dengan PowerShell versi 2, Anda harus menggunakan New-Object untuk membuat objek. IpPermission

```
$ip1 = New-Object Amazon.EC2.Model.IpPermission
$ip1.IpProtocol = "tcp"
$ip1.FromPort = 22
$ip1.ToPort = 22
$ip1.IpRanges.Add("203.0.113.25/32")

$ip2 = new-object Amazon.EC2.Model.IpPermission
$ip2.IpProtocol = "tcp"
$ip2.FromPort = 3389
$ip2.ToPort = 3389
$ip2.IpRanges.Add("203.0.113.25/32")

Grant-EC2SecurityGroupIngress -GroupName "my-security-group" -IpPermission @( $ip1,
  $ip2 )
```

Contoh 5: Contoh ini memberikan akses port TCP 8081 dari grup keamanan sumber tertentu (sg-1a2b3c4d) ke grup keamanan yang ditentukan (sg-12345678).

```
$ug = New-Object Amazon.EC2.Model.UserIdGroupPair
$ug.GroupId = "sg-1a2b3c4d"
$ug.UserId = "123456789012"

Grant-EC2SecurityGroupIngress -GroupId sg-12345678 -IpPermission
@( @{ IpProtocol="tcp"; FromPort="8081"; ToPort="8081"; UserIdGroupPairs=$ug } )
```

Contoh 6: Contoh ini menambahkan CIDR 5.5.5.5/32 ke aturan Ingress dari Grup keamanan sg-1234abcd untuk lalu lintas port TCP 22 dengan deskripsi.

```
$IpRange = New-Object -TypeName Amazon.EC2.Model.IpRange
$IpRange.CidrIp = "5.5.5.5/32"
$IpRange.Description = "SSH from Office"
$IpPermission = New-Object Amazon.EC2.Model.IpPermission
$IpPermission.IpProtocol = "tcp"
$IpPermission.ToPort = 22
$IpPermission.FromPort = 22
$IpPermission.Ipv4Ranges = $IpRange
Grant-EC2SecurityGroupIngress -GroupId sg-1234abcd -IpPermission $IpPermission
```

- Untuk detail API, lihat [AuthorizeSecurityGroupIngress](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Import-EC2Image

Contoh kode berikut menunjukkan cara menggunakan `Import-EC2Image`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengimpor image mesin virtual disk tunggal dari bucket Amazon S3 yang ditentukan ke EC2 Amazon dengan token idempotensi. Contoh ini mengharuskan Peran Layanan Impor VM dengan nama default 'vmimport' ada, dengan kebijakan yang mengizinkan akses EC2 Amazon ke bucket yang ditentukan, seperti yang dijelaskan dalam topik Prekuisisi Impor VM. Untuk menggunakan peran kustom, tentukan nama peran menggunakan **-RoleName** parameter.

```
$container = New-Object Amazon.EC2.Model.ImageDiskContainer
$container.Format="VMDK"
$container.UserBucket = New-Object Amazon.EC2.Model.UserBucket
$container.UserBucket.S3Bucket = "amzn-s3-demo-bucket"
$container.UserBucket.S3Key = "Win_2008_Server_Standard_SP2_64-bit-disk1.vmdk"
```

```
$parms = @{
    "ClientToken"="idempotencyToken"
    "Description"="Windows 2008 Standard Image Import"
    "Platform"="Windows"
    "LicenseType"="AWS"
}

Import-EC2Image -DiskContainer $container @parms
```

Output:

```
Architecture      :
Description       : Windows 2008 Standard Image
Hypervisor        :
ImageId           :
ImportTaskId      : import-ami-abcdefgh
LicenseType       : AWS
Platform          : Windows
Progress          : 2
SnapshotDetails   : {}
Status            : active
StatusMessage     : pending
```

- Untuk detail API, lihat [ImportImage](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Import-EC2KeyPair

Contoh kode berikut menunjukkan cara menggunakan `Import-EC2KeyPair`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengimpor kunci publik ke EC2. Baris pertama menyimpan isi file kunci publik (*.pub) dalam variabel. **\$publickey** Selanjutnya, contoh mengkonversi UTF8 format file kunci publik ke string Base64-encoded, dan menyimpan string dikonversi dalam variabel. **\$pkbase64** Pada baris terakhir, kunci publik yang dikonversi diimpor ke EC2. Cmdlet mengembalikan sidik jari kunci dan nama sebagai hasil.

```
$publickey=[Io.File]::ReadAllText("C:\Users\TestUser\.ssh\id_rsa.pub")
$pkbase64 =
[System.Convert]::ToBase64String([System.Text.Encoding]::UTF8.GetBytes($publickey))
```

```
Import-EC2KeyPair -KeyName Example-user-key -PublicKey $pkbase64
```

Output:

KeyFingerprint	KeyName
-----	-----
do:d0:15:8f:79:97:12:be:00:fd:df:31:z3:b1:42:z1	Example-user-key

- Untuk detail API, lihat [ImportKeyPair](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Import-EC2Snapshot

Contoh kode berikut menunjukkan cara menggunakan `Import-EC2Snapshot`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengimpor image disk VM format 'VMDK' ke snapshot Amazon EBS. Contoh ini memerlukan Peran Layanan Impor VM dengan nama default 'vmimport', dengan kebijakan yang memungkinkan EC2 Amazon mengakses bucket yang ditentukan, seperti yang dijelaskan dalam topik **VM Import Prerequisites** di <http://docs.aws.amazon.com/AWSEC2/latest/WindowsGuide/VMImportPrerequisites.html>. Untuk menggunakan peran kustom, tentukan nama peran menggunakan **-RoleName** parameter.

```
$parms = @{
    "ClientToken"="idempotencyToken"
    "Description"="Disk Image Import"
    "DiskContainer_Description" = "Data disk"
    "DiskContainer_Format" = "VMDK"
    "DiskContainer_S3Bucket" = "amzn-s3-demo-bucket"
    "DiskContainer_S3Key" = "datadiskimage.vmdk"
}

Import-EC2Snapshot @parms
```

Output:

Description	ImportTaskId	SnapshotTaskDetail
-----	-----	-----

```
Disk Image Import      import-snap-abcdefgh
Amazon.EC2.Model.SnapshotTaskDetail
```

- Untuk detail API, lihat [ImportSnapshot](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Move-EC2AddressToVpc

Contoh kode berikut menunjukkan cara menggunakan `Move-EC2AddressToVpc`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memindahkan EC2 instance dengan alamat IP publik 12.345.67.89 ke platform EC2 -VPC di wilayah AS Timur (Virginia Utara).

```
Move-EC2AddressToVpc -PublicIp 12.345.67.89 -Region us-east-1
```

Contoh 2: Contoh ini menyalurkan hasil `Get-EC2Instance` perintah ke `Move-EC2AddressToVpc` cmdlet. `Get-EC2InstancePerintah` mendapat instance yang ditentukan oleh ID instance, kemudian mengembalikan properti alamat IP publik dari instance tersebut.

```
(Get-EC2Instance -Instance i-12345678).Instances.PublicIpAddress | Move-
EC2AddressToVpc
```

- Untuk detail API, lihat [MoveAddressToVpc](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-EC2Address

Contoh kode berikut menunjukkan cara menggunakan `New-EC2Address`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengalokasikan alamat IP Elastis untuk digunakan dengan instance di VPC.

```
New-EC2Address -Domain Vpc
```

Output:

AllocationId	Domain	PublicIp
-----	-----	-----

```
eipalloc-12345678    vpc    198.51.100.2
```

Contoh 2: Contoh ini mengalokasikan alamat IP Elastis untuk digunakan dengan instance di EC2 - Classic.

```
New-EC2Address
```

Output:

```
AllocationId    Domain    PublicIp
-----
                standard  203.0.113.17
```

- Untuk detail API, lihat [AllocateAddress](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-EC2CustomerGateway

Contoh kode berikut menunjukkan cara menggunakan `New-EC2CustomerGateway`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat gateway pelanggan yang ditentukan.

```
New-EC2CustomerGateway -Type ipsec.1 -PublicIp 203.0.113.12 -BgpAsn 65534
```

Output:

```
BgpAsn          : 65534
CustomerGatewayId : cgw-1a2b3c4d
IpAddress       : 203.0.113.12
State           : available
Tags            : {}
Type            : ipsec.1
```

- Untuk detail API, lihat [CreateCustomerGateway](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-EC2DhcpOption

Contoh kode berikut menunjukkan cara menggunakan `New-EC2DhcpOption`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menciptakan kumpulan opsi DHCP yang ditentukan. Sintaks yang digunakan oleh contoh ini memerlukan PowerShell versi 3 atau yang lebih baru.

```
$options = @( @{Key="domain-name";Values=@("abc.local")}, @{Key="domain-name-servers";Values=@("10.0.0.101","10.0.0.102")})
New-EC2DhcpOption -DhcpConfiguration $options
```

Output:

DhcpConfigurations	DhcpOptionsId	Tags
-----	-----	----
{domain-name, domain-name-servers}	dopt-1a2b3c4d	{}

Contoh 2: Dengan PowerShell versi 2, Anda harus menggunakan New-Object untuk membuat setiap opsi DHCP.

```
$option1 = New-Object Amazon.EC2.Model.DhcpConfiguration
$option1.Key = "domain-name"
$option1.Values = "abc.local"

$option2 = New-Object Amazon.EC2.Model.DhcpConfiguration
$option2.Key = "domain-name-servers"
$option2.Values = @("10.0.0.101","10.0.0.102")

New-EC2DhcpOption -DhcpConfiguration @($option1, $option2)
```

Output:

DhcpConfigurations	DhcpOptionsId	Tags
-----	-----	----
{domain-name, domain-name-servers}	dopt-2a3b4c5d	{}

- Untuk detail API, lihat [CreateDhcpOptions](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-EC2FlowLog

Contoh kode berikut menunjukkan cara menggunakan New-EC2FlowLog.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat EC2 flowlog untuk subnet subnet-1d234567 ke 'subnet1-log' cloud-watch-log bernama untuk semua lalu lintas 'REJECT' menggunakan batas peran 'Admin'

```
New-EC2FlowLog -ResourceId "subnet-1d234567" -LogDestinationType cloud-watch-logs -LogGroupName subnet1-log -TrafficType "REJECT" -ResourceType Subnet -DeliverLogsPermissionArn "arn:aws:iam::98765432109:role/Admin"
```

Output:

ClientToken	FlowLogIds	Unsuccessful
-----	-----	-----
m1VN2cxP3iB4qo//VUK15EU6cF7gQL0xcqNefvjeTGw=	{f1-012fc34eed5678c9d}	{}

- Untuk detail API, lihat [CreateFlowLogs](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-EC2Host

Contoh kode berikut menunjukkan cara menggunakan New-EC2Host.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengalokasikan Host Khusus ke akun Anda untuk jenis instans dan zona ketersediaan yang diberikan

```
New-EC2Host -AutoPlacement on -AvailabilityZone eu-west-1b -InstanceType m4.xlarge -Quantity 1
```

Output:

```
h-01e23f4cd567890f3
```

- Untuk detail API, lihat [AllocateHosts](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-EC2HostReservation

Contoh kode berikut menunjukkan cara menggunakan New-EC2HostReservation.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membeli reservasi yang menawarkan hro-0c1f23456789d0ab dengan konfigurasi yang cocok dengan Host Khusus Anda h-01e23f4cd567890f1

```
New-EC2HostReservation -OfferingId hro-0c1f23456789d0ab HostIdSet
h-01e23f4cd567890f1
```

Output:

```
ClientToken      :
CurrencyCode     :
Purchase        : {hr-0123f4b5d67bedc89}
TotalHourlyPrice : 1.307
TotalUpfrontPrice : 0.000
```

- Untuk detail API, lihat [PurchaseHostReservation](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-EC2Image

Contoh kode berikut menunjukkan cara menggunakan New-EC2Image.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat AMI dengan nama dan deskripsi yang ditentukan, dari instance yang ditentukan. Amazon EC2 mencoba mematikan instance dengan bersih sebelum membuat gambar, dan memulai ulang instance setelah selesai.

```
New-EC2Image -InstanceId i-12345678 -Name "my-web-server" -Description "My web
server AMI"
```

Contoh 2: Contoh ini membuat AMI dengan nama dan deskripsi yang ditentukan, dari instance yang ditentukan. Amazon EC2 membuat gambar tanpa mematikan dan memulai ulang instance; oleh karena itu, integritas sistem file pada gambar yang dibuat tidak dapat dijamin.

```
New-EC2Image -InstanceId i-12345678 -Name "my-web-server" -Description "My web
server AMI" -NoReboot $true
```

Contoh 3: Contoh ini membuat AMI dengan tiga volume. Volume pertama didasarkan pada snapshot Amazon EBS. Volume kedua adalah volume 100 GiB Amazon EBS kosong. Volume ketiga adalah volume penyimpanan instance. Sintaks yang digunakan oleh contoh ini membutuhkan PowerShell versi 3 atau lebih tinggi.

```
$ebsBlock1 = @{SnapshotId="snap-1a2b3c4d"}
$ebsBlock2 = @{VolumeSize=100}

New-EC2Image -InstanceId i-12345678 -Name "my-web-server" -Description
  "My web server AMI" -BlockDeviceMapping @( @{DeviceName="/dev/sdf";Ebs=
  $ebsBlock1}, @{DeviceName="/dev/sdg";Ebs=$ebsBlock2}, @{DeviceName="/dev/
  sdc";VirtualName="ephemeral0"})
```

- Untuk detail API, lihat [CreateImage](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-EC2Instance

Contoh kode berikut menunjukkan cara menggunakan `New-EC2Instance`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini meluncurkan satu instance dari AMI yang ditentukan di EC2 -Classic atau VPC default.

```
New-EC2Instance -ImageId ami-12345678 -MinCount 1 -MaxCount 1 -InstanceType
m3.medium -KeyName my-key-pair -SecurityGroup my-security-group
```

Contoh 2: Contoh ini meluncurkan satu instance dari AMI yang ditentukan dalam VPC.

```
New-EC2Instance -ImageId ami-12345678 -MinCount 1 -MaxCount 1 -SubnetId
subnet-12345678 -InstanceType t2.micro -KeyName my-key-pair -SecurityGroupId
sg-12345678
```

Contoh 3: Untuk menambahkan volume EBS atau volume penyimpanan instance, tentukan pemetaan perangkat blok dan tambahkan ke perintah. Contoh ini menambahkan volume penyimpanan instance.

```
$bdm = New-Object Amazon.EC2.Model.BlockDeviceMapping
$bdm.VirtualName = "ephemeral0"
$bdm.DeviceName = "/dev/sdf"
```

```
New-EC2Instance -ImageId ami-12345678 -BlockDeviceMapping $bdm ...
```

Contoh 4: Untuk menentukan salah satu Windows saat ini AMIs, dapatkan ID AMI menggunakan `Get-EC2ImageByName`. Contoh ini meluncurkan instance dari basis AMI saat ini untuk Windows Server 2016.

```
$ami = Get-EC2ImageByName WINDOWS_2016_BASE  
  
New-EC2Instance -ImageId $ami.ImageId ...
```

Contoh 5: Meluncurkan instance ke lingkungan host khusus yang ditentukan.

```
New-EC2Instance -ImageId ami-1a2b3c4d -InstanceType m4.large -KeyName my-key-pair  
-SecurityGroupId sg-1a2b3c4d -AvailabilityZone us-west-1a -Tenancy host -HostID  
h-1a2b3c4d5e6f1a2b3
```

Contoh 6: Permintaan ini meluncurkan dua instance dan menerapkan tag dengan kunci server web dan nilai produksi ke instance. Permintaan juga menerapkan tag dengan kunci cost-center dan nilai cc123 ke volume yang dibuat (dalam hal ini, volume root untuk setiap instance).

```
$tag1 = @{ Key="webserver"; Value="production" }  
$tag2 = @{ Key="cost-center"; Value="cc123" }  
  
$tagspec1 = new-object Amazon.EC2.Model.TagSpecification  
$tagspec1.ResourceType = "instance"  
$tagspec1.Tags.Add($tag1)  
  
$tagspec2 = new-object Amazon.EC2.Model.TagSpecification  
$tagspec2.ResourceType = "volume"  
$tagspec2.Tags.Add($tag2)  
  
New-EC2Instance -ImageId "ami-1a2b3c4d" -KeyName "my-key-pair" -MaxCount 2 -  
InstanceType "t2.large" -SubnetId "subnet-1a2b3c4d" -TagSpecification $tagspec1,  
$tagspec2
```

- Untuk detail API, lihat [RunInstances](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-EC2InstanceExportTask

Contoh kode berikut menunjukkan cara menggunakan `New-EC2InstanceExportTask`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengekspor instance yang dihentikan, **i-0800b00a00EXAMPLE**, sebagai hard disk virtual (VHD) ke bucket S3. **testbucket-export-instances-2019** Lingkungan target adalah **Microsoft**, dan parameter wilayah ditambahkan karena instance ada di **us-east-1** wilayah, sedangkan AWS Wilayah default pengguna bukan us-east-1. Untuk mendapatkan status tugas ekspor, salin **ExportTaskId** nilai dari hasil perintah ini, lalu jalankan **Get-EC2ExportTask -ExportTaskId export_task_ID_from_results**.

```
New-EC2InstanceExportTask -InstanceId i-0800b00a00EXAMPLE -
ExportToS3Task_DiskImageFormat VHD -ExportToS3Task_S3Bucket "amzn-s3-demo-bucket" -
TargetEnvironment Microsoft -Region us-east-1
```

Output:

```
Description      :
ExportTaskId      : export-i-077c73108aEXAMPLE
ExportToS3Task    : Amazon.EC2.Model.ExportToS3Task
InstanceExportDetails : Amazon.EC2.Model.InstanceExportDetails
State             : active
StatusMessage     :
```

- Untuk detail API, lihat [CreateInstanceExportTask](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-EC2InternetGateway

Contoh kode berikut menunjukkan cara menggunakan **New-EC2InternetGateway**.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat gateway Internet.

```
New-EC2InternetGateway
```

Output:

Attachments	InternetGatewayId	Tags
-----	-----	----

```
{ } igw-1a2b3c4d { }
```

- Untuk detail API, lihat [CreateInternetGateway](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-EC2KeyPair

Contoh kode berikut menunjukkan cara menggunakan `New-EC2KeyPair`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat key pair dan menangkap kunci pribadi RSA yang dikodekan PEM dalam file dengan nama yang ditentukan. Saat Anda menggunakan PowerShell, pengkodean harus diatur ke `ascii` untuk menghasilkan kunci yang valid. Untuk informasi selengkapnya, lihat [Membuat, Menampilkan, dan Menghapus Pasangan EC2 Kunci Amazon \(https://docs.aws.amazon.com/cli/latest/userguide/cli-services-ec2-keypairs.html\)](https://docs.aws.amazon.com/cli/latest/userguide/cli-services-ec2-keypairs.html) di Panduan Pengguna Antarmuka Baris AWS Perintah.

```
(New-EC2KeyPair -KeyName "my-key-pair").KeyMaterial | Out-File -Encoding ascii -
FilePath C:\path\my-key-pair.pem
```

- Untuk detail API, lihat [CreateKeyPair](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-EC2NetworkAcl

Contoh kode berikut menunjukkan cara menggunakan `New-EC2NetworkAcl`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat ACL jaringan untuk VPC yang ditentukan.

```
New-EC2NetworkAcl -VpcId vpc-12345678
```

Output:

```
Associations : {}
Entries      : {Amazon.EC2.Model.NetworkAclEntry, Amazon.EC2.Model.NetworkAclEntry}
IsDefault    : False
NetworkAclId : acl-12345678
Tags         : {}
```

```
VpcId      : vpc-12345678
```

- Untuk detail API, lihat [CreateNetworkAcl](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-EC2NetworkAclEntry

Contoh kode berikut menunjukkan cara menggunakan `New-EC2NetworkAclEntry`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat entri untuk ACL jaringan tertentu. Aturan ini memungkinkan lalu lintas masuk dari mana saja (0.0.0.0/0) pada port UDP 53 (DNS) ke subnet terkait.

```
New-EC2NetworkAclEntry -NetworkAclId acl-12345678 -Egress $false -RuleNumber 100
-Protocol 17 -PortRange_From 53 -PortRange_To 53 -CidrBlock 0.0.0.0/0 -RuleAction
allow
```

- Untuk detail API, lihat [CreateNetworkAclEntry](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-EC2NetworkInterface

Contoh kode berikut menunjukkan cara menggunakan `New-EC2NetworkInterface`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menciptakan antarmuka jaringan yang ditentukan.

```
New-EC2NetworkInterface -SubnetId subnet-1a2b3c4d -Description "my network
interface" -Group sg-12345678 -PrivateIpAddress 10.0.0.17
```

Output:

```
Association      :
Attachment       :
AvailabilityZone  : us-west-2c
Description      : my network interface
Groups           : {my-security-group}
MacAddress       : 0a:72:bc:1a:cd:7f
NetworkInterfaceId : eni-12345678
OwnerId          : 123456789012
```

```

PrivateDnsName      : ip-10-0-0-17.us-west-2.compute.internal
PrivateIpAddress    : 10.0.0.17
PrivateIpAddresses  : {}
RequesterId         :
RequesterManaged   : False
SourceDestCheck     : True
Status              : pending
SubnetId            : subnet-1a2b3c4d
TagSet              : {}
VpcId               : vpc-12345678

```

- Untuk detail API, lihat [CreateNetworkInterface](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-EC2PlacementGroup

Contoh kode berikut menunjukkan cara menggunakan `New-EC2PlacementGroup`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat grup penempatan dengan nama yang ditentukan.

```
New-EC2PlacementGroup -GroupName my-placement-group -Strategy cluster
```

- Untuk detail API, lihat [CreatePlacementGroup](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-EC2Route

Contoh kode berikut menunjukkan cara menggunakan `New-EC2Route`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menciptakan rute yang ditentukan untuk tabel rute yang ditentukan. Rute cocok dengan semua lalu lintas dan mengirimkannya ke gateway Internet yang ditentukan.

```
New-EC2Route -RouteTableId rtb-1a2b3c4d -DestinationCidrBlock 0.0.0.0/0 -GatewayId igw-1a2b3c4d
```

Output:

```
True
```

- Untuk detail API, lihat [CreateRoute](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-EC2RouteTable

Contoh kode berikut menunjukkan cara menggunakan `New-EC2RouteTable`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat tabel rute untuk VPC yang ditentukan.

```
New-EC2RouteTable -VpcId vpc-12345678
```

Output:

```
Associations      : {}
PropagatingVgws   : {}
Routes            : {}
RouteTableId      : rtb-1a2b3c4d
Tags              : {}
VpcId             : vpc-12345678
```

- Untuk detail API, lihat [CreateRouteTable](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-EC2ScheduledInstance

Contoh kode berikut menunjukkan cara menggunakan `New-EC2ScheduledInstance`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini meluncurkan Instance Terjadwal yang ditentukan.

```
New-EC2ScheduledInstance -ScheduledInstanceId sci-1234-1234-1234-1234-123456789012 -
InstanceCount 1 `
-IamInstanceProfile_Name my-iam-role `
-LaunchSpecification_ImageId ami-12345678 `
-LaunchSpecification_InstanceType c4.large `
-LaunchSpecification_SubnetId subnet-12345678 `
-LaunchSpecification_SecurityGroupId sg-12345678
```

- Untuk detail API, lihat [RunScheduledInstances](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-EC2ScheduledInstancePurchase

Contoh kode berikut menunjukkan cara menggunakan `New-EC2ScheduledInstancePurchase`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membeli Instance Terjadwal.

```
$request = New-Object Amazon.EC2.Model.PurchaseRequest
$request.InstanceCount = 1
$request.PurchaseToken = "eyJ2IjoiMSIsInMiOjEsImMiOi..."
New-EC2ScheduledInstancePurchase -PurchaseRequest $request
```

Output:

```
AvailabilityZone      : us-west-2b
CreateDate            : 1/25/2016 1:43:38 PM
HourlyPrice           : 0.095
InstanceCount         : 1
InstanceType          : c4.large
NetworkPlatform       : EC2-VPC
NextSlotStartTime     : 1/31/2016 1:00:00 AM
Platform              : Linux/UNIX
PreviousSlotEndTime   :
Recurrence             : Amazon.EC2.Model.ScheduledInstanceRecurrence
ScheduledInstanceId    : sci-1234-1234-1234-1234-123456789012
SlotDurationInHours   : 32
TermEndDate           : 1/31/2017 1:00:00 AM
TermStartDate         : 1/31/2016 1:00:00 AM
TotalScheduledInstanceHours : 1696
```

- Untuk detail API, lihat [PurchaseScheduledInstances](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-EC2SecurityGroup

Contoh kode berikut menunjukkan cara menggunakan `New-EC2SecurityGroup`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat grup keamanan untuk VPC yang ditentukan.

```
New-EC2SecurityGroup -GroupName my-security-group -Description "my security group" -VpcId vpc-12345678
```

Output:

```
sg-12345678
```

Contoh 2: Contoh ini membuat grup keamanan untuk EC2 -Classic.

```
New-EC2SecurityGroup -GroupName my-security-group -Description "my security group"
```

Output:

```
sg-45678901
```

- Untuk detail API, lihat [CreateSecurityGroup](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-EC2Snapshot

Contoh kode berikut menunjukkan cara menggunakan New-EC2Snapshot.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat snapshot dari volume yang ditentukan.

```
New-EC2Snapshot -VolumeId vol-12345678 -Description "This is a test"
```

Output:

```
DataEncryptionKeyId :  
Description          : This is a test  
Encrypted            : False  
KmsKeyId             :  
OwnerAlias           :
```

```

OwnerId      : 123456789012
Progress     :
SnapshotId   : snap-12345678
StartTime    : 12/22/2015 1:28:42 AM
State        : pending
StateMessage :
Tags         : {}
VolumeId     : vol-12345678
VolumeSize   : 20

```

- Untuk detail API, lihat [CreateSnapshot](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-EC2SpotDatafeedSubscription

Contoh kode berikut menunjukkan cara menggunakan `New-EC2SpotDatafeedSubscription`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat umpan data instance Spot.

```
New-EC2SpotDatafeedSubscription -Bucket amzn-s3-demo-bucket -Prefix spotdata
```

Output:

```

Bucket      : my-s3-bucket
Fault       :
OwnerId     : 123456789012
Prefix      : spotdata
State       : Active

```

- Untuk detail API, lihat [CreateSpotDatafeedSubscription](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-EC2Subnet

Contoh kode berikut menunjukkan cara menggunakan `New-EC2Subnet`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat subnet dengan CIDR yang ditentukan.

```
New-EC2Subnet -VpcId vpc-12345678 -CidrBlock 10.0.0.0/24
```

Output:

```
AvailabilityZone      : us-west-2c
AvailableIpAddressCount : 251
CidrBlock             : 10.0.0.0/24
DefaultForAz          : False
MapPublicIpOnLaunch   : False
State                 : pending
SubnetId              : subnet-1a2b3c4d
Tag                   : {}
VpcId                 : vpc-12345678
```

- Untuk detail API, lihat [CreateSubnet](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-EC2Tag

Contoh kode berikut menunjukkan cara menggunakan New-EC2Tag.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menambahkan satu tag ke sumber daya yang ditentukan. Kunci tag adalah 'MyTag' dan nilai tag adalah 'myTagValue'. Sintaks yang digunakan oleh contoh ini membutuhkan PowerShell versi 3 atau lebih tinggi.

```
New-EC2Tag -Resource i-12345678 -Tag @{ Key="myTag"; Value="myTagValue" }
```

Contoh 2: Contoh ini memperbarui atau menambahkan tag yang ditentukan ke sumber daya yang ditentukan. Sintaks yang digunakan oleh contoh ini membutuhkan PowerShell versi 3 atau lebih tinggi.

```
New-EC2Tag -Resource i-12345678 -Tag @( @{ Key="myTag"; Value="newTagValue" },
    @{ Key="test"; Value="anotherTagValue" } )
```

Contoh 3: Dengan PowerShell versi 2, Anda harus menggunakan New-Object untuk membuat tag untuk parameter Tag.

```
$tag = New-Object Amazon.EC2.Model.Tag
```

```
$tag.Key = "myTag"
$tag.Value = "myTagValue"

New-EC2Tag -Resource i-12345678 -Tag $tag
```

- Untuk detail API, lihat [CreateTags](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-EC2Volume

Contoh kode berikut menunjukkan cara menggunakan `New-EC2Volume`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menciptakan volume yang ditentukan.

```
New-EC2Volume -Size 50 -AvailabilityZone us-west-2a -VolumeType gp2
```

Output:

```
Attachments      : {}
AvailabilityZone  : us-west-2a
CreateTime       : 12/22/2015 1:42:07 AM
Encrypted        : False
Iops             : 150
KmsKeyId         :
Size            : 50
SnapshotId       :
State            : creating
Tags             : {}
VolumeId         : vol-12345678
VolumeType       : gp2
```

Contoh 2: Permintaan contoh ini membuat volume dan menerapkan tag dengan kunci tumpukan dan nilai produksi.

```
$tag = @{ Key="stack"; Value="production" }

$tagspec = new-object Amazon.EC2.Model.TagSpecification
$tagspec.ResourceType = "volume"
$tagspec.Tags.Add($tag)
```

```
New-EC2Volume -Size 80 -AvailabilityZone "us-west-2a" -TagSpecification $tagspec
```

- Untuk detail API, lihat [CreateVolume](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-EC2Vpc

Contoh kode berikut menunjukkan cara menggunakan `New-EC2Vpc`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat VPC dengan CIDR yang ditentukan. Amazon VPC juga membuat yang berikut untuk VPC: set opsi DHCP default, tabel rute utama, dan ACL jaringan default.

```
New-EC2VPC -CidrBlock 10.0.0.0/16
```

Output:

```
CidrBlock      : 10.0.0.0/16
DhcpOptionsId  : dopt-1a2b3c4d
InstanceTenancy : default
IsDefault      : False
State          : pending
Tags           : {}
VpcId          : vpc-12345678
```

- Untuk detail API, lihat [CreateVpc](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-EC2VpcEndpoint

Contoh kode berikut menunjukkan cara menggunakan `New-EC2VpcEndpoint`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat Endpoint VPC baru untuk layanan `com.amazonaws.eu-west-1.s3` di VPC `vpc-0fc1ff23f45b678eb`

```
New-EC2VpcEndpoint -ServiceName com.amazonaws.eu-west-1.s3 -VpcId
vpc-0fc1ff23f45b678eb
```

Output:

```
ClientToken VpcEndpoint
-----
Amazon.EC2.Model.VpcEndpoint
```

- Untuk detail API, lihat [CreateVpcEndpoint](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-EC2VpnConnection

Contoh kode berikut menunjukkan cara menggunakan `New-EC2VpnConnection`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat koneksi VPN antara gateway pribadi virtual yang ditentukan dan gateway pelanggan yang ditentukan. Outputnya mencakup informasi konfigurasi yang dibutuhkan administrator jaringan Anda, dalam format XHTML.

```
New-EC2VpnConnection -Type ipsec.1 -CustomerGatewayId cgw-1a2b3c4d -VpnGatewayId
vgw-1a2b3c4d
```

Output:

```
CustomerGatewayConfiguration : [XML document]
CustomerGatewayId           : cgw-1a2b3c4d
Options                     :
Routes                     : {}
State                       : pending
Tags                       : {}
Type                       :
VgwTelemetry               : {}
VpnConnectionId            : vpn-12345678
VpnGatewayId               : vgw-1a2b3c4d
```

Contoh 2: Contoh ini membuat koneksi VPN dan menangkap konfigurasi dalam file dengan nama yang ditentukan.

```
(New-EC2VpnConnection -CustomerGatewayId cgw-1a2b3c4d -VpnGatewayId
vgw-1a2b3c4d).CustomerGatewayConfiguration | Out-File C:\path\vpn-configuration.xml
```

Contoh 3: Contoh ini membuat koneksi VPN, dengan perutean statis, antara gateway pribadi virtual yang ditentukan dan gateway pelanggan yang ditentukan.

```
New-EC2VpnConnection -Type ipsec.1 -CustomerGatewayId cgw-1a2b3c4d -VpnGatewayId
vgw-1a2b3c4d -Options_StaticRoutesOnly $true
```

- Untuk detail API, lihat [CreateVpnConnection](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-EC2VpnConnectionRoute

Contoh kode berikut menunjukkan cara menggunakan `New-EC2VpnConnectionRoute`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat rute statis yang ditentukan untuk koneksi VPN yang ditentukan.

```
New-EC2VpnConnectionRoute -VpnConnectionId vpn-12345678 -DestinationCidrBlock
11.12.0.0/16
```

- Untuk detail API, lihat [CreateVpnConnectionRoute](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-EC2VpnGateway

Contoh kode berikut menunjukkan cara menggunakan `New-EC2VpnGateway`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat gateway pribadi virtual yang ditentukan.

```
New-EC2VpnGateway -Type ipsec.1
```

Output:

```
AvailabilityZone :
State            : available
Tags             : {}
Type             : ipsec.1
VpcAttachments  : {}
VpnGatewayId     : vgw-1a2b3c4d
```

- Untuk detail API, lihat [CreateVpnGateway](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Register-EC2Address

Contoh kode berikut menunjukkan cara menggunakan `Register-EC2Address`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengaitkan alamat IP Elastis yang ditentukan dengan instance yang ditentukan dalam VPC.

```
C:\> Register-EC2Address -InstanceId i-12345678 -AllocationId eipalloc-12345678
```

Output:

```
eipassoc-12345678
```

Contoh 2: Contoh ini mengaitkan alamat IP Elastis yang ditentukan dengan instance yang ditentukan di EC2 -Classic.

```
C:\> Register-EC2Address -InstanceId i-12345678 -PublicIp 203.0.113.17
```

- Untuk detail API, lihat [AssociateAddress](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Register-EC2DhcpOption

Contoh kode berikut menunjukkan cara menggunakan `Register-EC2DhcpOption`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengaitkan opsi DHCP tertentu yang ditetapkan dengan VPC yang ditentukan.

```
Register-EC2DhcpOption -DhcpOptionsId dopt-1a2b3c4d -VpcId vpc-12345678
```

Contoh 2: Contoh ini mengaitkan opsi DHCP default yang ditetapkan dengan VPC yang ditentukan.

```
Register-EC2DhcpOption -DhcpOptionsId default -VpcId vpc-12345678
```

- Untuk detail API, lihat [AssociateDhcpOptions](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Register-EC2Image

Contoh kode berikut menunjukkan cara menggunakan `Register-EC2Image`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendaftarkan AML menggunakan file manifes yang ditentukan di Amazon S3.

```
Register-EC2Image -ImageLocation amzn-s3-demo-bucket/my-web-server-ami/  
image.manifest.xml -Name my-web-server-ami
```

- Untuk detail API, lihat [RegisterImage](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Register-EC2PrivateIpAddress

Contoh kode berikut menunjukkan cara menggunakan `Register-EC2PrivateIpAddress`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memberikan alamat IP pribadi sekunder yang ditentukan ke antarmuka jaringan yang ditentukan.

```
Register-EC2PrivateIpAddress -NetworkInterfaceId eni-1a2b3c4d -PrivateIpAddress  
10.0.0.82
```

Contoh 2: Contoh ini membuat dua alamat IP pribadi sekunder dan menetapkannya ke antarmuka jaringan yang ditentukan.

```
Register-EC2PrivateIpAddress -NetworkInterfaceId eni-1a2b3c4d -  
SecondaryPrivateIpAddressCount 2
```

- Untuk detail API, lihat [AssignPrivateIpAddresses](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Register-EC2RouteTable

Contoh kode berikut menunjukkan cara menggunakan `Register-EC2RouteTable`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengaitkan tabel rute yang ditentukan dengan subnet yang ditentukan.

```
Register-EC2RouteTable -RouteTableId rtb-1a2b3c4d -SubnetId subnet-1a2b3c4d
```

Output:

```
rtbassoc-12345678
```

- Untuk detail API, lihat [AssociateRouteTable](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-EC2Address

Contoh kode berikut menunjukkan cara menggunakan `Remove-EC2Address`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini merilis alamat IP Elastis yang ditentukan untuk instance di VPC.

```
Remove-EC2Address -AllocationId eipalloc-12345678 -Force
```

Contoh 2: Contoh ini merilis alamat IP Elastis yang ditentukan untuk instance di EC2 -Classic.

```
Remove-EC2Address -PublicIp 198.51.100.2 -Force
```

- Untuk detail API, lihat [ReleaseAddress](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-EC2CapacityReservation

Contoh kode berikut menunjukkan cara menggunakan `Remove-EC2CapacityReservation`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membatalkan reservasi kapasitas `cr-0c1f2345db6f7cdba`

```
Remove-EC2CapacityReservation -CapacityReservationId cr-0c1f2345db6f7cdba
```

Output:

```

Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-EC2CapacityReservation (CancelCapacityReservation)"
on target "cr-0c1f2345db6f7cdba".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"): y
True

```

- Untuk detail API, lihat [CancelCapacityReservation](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-EC2CustomerGateway

Contoh kode berikut menunjukkan cara menggunakan `Remove-EC2CustomerGateway`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus gateway pelanggan yang ditentukan. Anda diminta untuk konfirmasi sebelum operasi berlangsung, kecuali jika Anda juga menentukan parameter `Force`.

```
Remove-EC2CustomerGateway -CustomerGatewayId cgw-1a2b3c4d
```

Output:

```

Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2CustomerGateway (DeleteCustomerGateway)" on Target
"cgw-1a2b3c4d".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"):

```

- Untuk detail API, lihat [DeleteCustomerGateway](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-EC2DhcpOption

Contoh kode berikut menunjukkan cara menggunakan `Remove-EC2DhcpOption`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus set opsi DHCP yang ditentukan. Anda diminta untuk konfirmasi sebelum operasi berlangsung, kecuali jika Anda juga menentukan parameter Force.

```
Remove-EC2DhcpOption -DhcpOptionsId dopt-1a2b3c4d
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2DhcpOption (DeleteDhcpOptions)" on Target
"dopt-1a2b3c4d".
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is
"Y"):
```

- Untuk detail API, lihat [DeleteDhcpOptions](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-EC2FlowLog

Contoh kode berikut menunjukkan cara menggunakan Remove-EC2FlowLog.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus FlowLogId fl-01a2b3456a789c01 yang diberikan

```
Remove-EC2FlowLog -FlowLogId fl-01a2b3456a789c01
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-EC2FlowLog (DeleteFlowLogs)" on target
"fl-01a2b3456a789c01".
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is
"Y"): Y
```

- Untuk detail API, lihat [DeleteFlowLogs](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-EC2Host

Contoh kode berikut menunjukkan cara menggunakan `Remove-EC2Host`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini merilis ID host yang diberikan `h-0badafd1dcb2f3456`

```
Remove-EC2Host -HostId h-0badafd1dcb2f3456
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-EC2Host (ReleaseHosts)" on target
"h-0badafd1dcb2f3456".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"): Y

Successful                Unsuccessful
-----
{h-0badafd1dcb2f3456} {}
```

- Untuk detail API, lihat [ReleaseHosts](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-EC2Instance

Contoh kode berikut menunjukkan cara menggunakan `Remove-EC2Instance`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengakhiri instance yang ditentukan (instance mungkin berjalan atau dalam keadaan 'berhenti'). Cmdlet akan meminta konfirmasi sebelum melanjutkan; gunakan sakelar `-Force` untuk menekan prompt.

```
Remove-EC2Instance -InstanceId i-12345678
```

Output:

CurrentState	InstanceId	PreviousState
--------------	------------	---------------

```

-----
Amazon.EC2.Model.InstanceState    i-12345678    Amazon.EC2.Model.InstanceState

```

- Untuk detail API, lihat [TerminateInstances](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-EC2InternetGateway

Contoh kode berikut menunjukkan cara menggunakan `Remove-EC2InternetGateway`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus gateway Internet yang ditentukan. Anda diminta untuk konfirmasi sebelum operasi berlangsung, kecuali jika Anda juga menentukan parameter `Force`.

```
Remove-EC2InternetGateway -InternetGatewayId igw-1a2b3c4d
```

Output:

```

Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2InternetGateway (DeleteInternetGateway)" on Target
"igw-1a2b3c4d".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"):

```

- Untuk detail API, lihat [DeleteInternetGateway](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-EC2KeyPair

Contoh kode berikut menunjukkan cara menggunakan `Remove-EC2KeyPair`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus key pair yang ditentukan. Anda diminta untuk konfirmasi sebelum operasi berlangsung, kecuali jika Anda juga menentukan parameter `Force`.

```
Remove-EC2KeyPair -KeyName my-key-pair
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2KeyPair (DeleteKeyPair)" on Target "my-key-pair".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is "Y"):
```

- Untuk detail API, lihat [DeleteKeyPair](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-EC2NetworkAcl

Contoh kode berikut menunjukkan cara menggunakan `Remove-EC2NetworkAcl`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus ACL jaringan tertentu. Anda diminta untuk konfirmasi sebelum operasi berlangsung, kecuali jika Anda juga menentukan parameter `Force`.

```
Remove-EC2NetworkAcl -NetworkAclId acl-12345678
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2NetworkAcl (DeleteNetworkAcl)" on Target
"acl-12345678".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is "Y"):
```

- Untuk detail API, lihat [DeleteNetworkAcl](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-EC2NetworkAclEntry

Contoh kode berikut menunjukkan cara menggunakan `Remove-EC2NetworkAclEntry`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus aturan yang ditentukan dari ACL jaringan yang ditentukan. Anda diminta untuk konfirmasi sebelum operasi berlangsung, kecuali jika Anda juga menentukan parameter `Force`.

```
Remove-EC2NetworkAclEntry -NetworkAclId acl-12345678 -Egress $false -RuleNumber 100
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2NetworkAclEntry (DeleteNetworkAclEntry)" on Target
"acl-12345678".
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is
"Y"):
```

- Untuk detail API, lihat [DeleteNetworkAclEntry](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-EC2NetworkInterface

Contoh kode berikut menunjukkan cara menggunakan `Remove-EC2NetworkInterface`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus antarmuka jaringan yang ditentukan. Anda diminta untuk konfirmasi sebelum operasi berlangsung, kecuali jika Anda juga menentukan parameter `Force`.

```
Remove-EC2NetworkInterface -NetworkInterfaceId eni-12345678
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2NetworkInterface (DeleteNetworkInterface)" on Target
"eni-12345678".
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is
"Y"):
```

- Untuk detail API, lihat [DeleteNetworkInterface](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-EC2PlacementGroup

Contoh kode berikut menunjukkan cara menggunakan `Remove-EC2PlacementGroup`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus grup penempatan yang ditentukan. Anda diminta untuk konfirmasi sebelum operasi berlangsung, kecuali jika Anda juga menentukan parameter Force.

```
Remove-EC2PlacementGroup -GroupName my-placement-group
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2PlacementGroup (DeletePlacementGroup)" on Target
"my-placement-group".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"):
```

- Untuk detail API, lihat [DeletePlacementGroup](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-EC2Route

Contoh kode berikut menunjukkan cara menggunakan Remove-EC2Route.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus rute yang ditentukan dari tabel rute yang ditentukan. Anda diminta untuk konfirmasi sebelum operasi berlangsung, kecuali jika Anda juga menentukan parameter Force.

```
Remove-EC2Route -RouteTableId rtb-1a2b3c4d -DestinationCidrBlock 0.0.0.0/0
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2Route (DeleteRoute)" on Target "rtb-1a2b3c4d".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"):
```

- Untuk detail API, lihat [DeleteRoute](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-EC2RouteTable

Contoh kode berikut menunjukkan cara menggunakan `Remove-EC2RouteTable`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus tabel rute yang ditentukan. Anda diminta untuk konfirmasi sebelum operasi berlangsung, kecuali jika Anda juga menentukan parameter `Force`.

```
Remove-EC2RouteTable -RouteTableId rtb-1a2b3c4d
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2RouteTable (DeleteRouteTable)" on Target
"rtb-1a2b3c4d".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"):
```

- Untuk detail API, lihat [DeleteRouteTable](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-EC2SecurityGroup

Contoh kode berikut menunjukkan cara menggunakan `Remove-EC2SecurityGroup`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus grup keamanan yang ditentukan untuk EC2 -VPC. Anda diminta untuk konfirmasi sebelum operasi berlangsung, kecuali jika Anda juga menentukan parameter `Force`.

```
Remove-EC2SecurityGroup -GroupId sg-12345678
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2SecurityGroup (DeleteSecurityGroup)" on Target
"sg-12345678".
```

```
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "Y"):
```

Contoh 2: Contoh ini menghapus grup keamanan yang ditentukan untuk EC2 -Classic.

```
Remove-EC2SecurityGroup -GroupName my-security-group -Force
```

- Untuk detail API, lihat [DeleteSecurityGroup](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-EC2Snapshot

Contoh kode berikut menunjukkan cara menggunakan `Remove-EC2Snapshot`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus snapshot yang ditentukan. Anda diminta untuk konfirmasi sebelum operasi berlangsung, kecuali jika Anda juga menentukan parameter `Force`.

```
Remove-EC2Snapshot -SnapshotId snap-12345678
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-EC2Snapshot (DeleteSnapshot)" on target
"snap-12345678".
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "Y"):
```

- Untuk detail API, lihat [DeleteSnapshot](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-EC2SpotDatafeedSubscription

Contoh kode berikut menunjukkan cara menggunakan `Remove-EC2SpotDatafeedSubscription`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus umpan data instans Spot Anda. Anda diminta untuk konfirmasi sebelum operasi berlangsung, kecuali jika Anda juga menentukan parameter `Force`.

```
Remove-EC2SpotDatafeedSubscription
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2SpotDatafeedSubscription
(DeleteSpotDatafeedSubscription)" on Target "".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"):
```

- Untuk detail API, lihat [DeleteSpotDatafeedSubscription](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-EC2Subnet

Contoh kode berikut menunjukkan cara menggunakan `Remove-EC2Subnet`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus subnet yang ditentukan. Anda diminta untuk konfirmasi sebelum operasi berlangsung, kecuali jika Anda juga menentukan parameter `Force`.

```
Remove-EC2Subnet -SubnetId subnet-1a2b3c4d
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2Subnet (DeleteSubnet)" on Target "subnet-1a2b3c4d".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"):
```

- Untuk detail API, lihat [DeleteSubnet](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-EC2Tag

Contoh kode berikut menunjukkan cara menggunakan `Remove-EC2Tag`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus tag yang ditentukan dari sumber daya yang ditentukan, terlepas dari nilai tag. Sintaks yang digunakan oleh contoh ini memerlukan PowerShell versi 3 atau yang lebih baru.

```
Remove-EC2Tag -Resource i-12345678 -Tag @{ Key="myTag" } -Force
```

Contoh 2: Contoh ini menghapus tag yang ditentukan dari sumber daya yang ditentukan, tetapi hanya jika nilai tag cocok. Sintaks yang digunakan oleh contoh ini memerlukan PowerShell versi 3 atau yang lebih baru.

```
Remove-EC2Tag -Resource i-12345678 -Tag @{ Key="myTag";Value="myTagValue" } -Force
```

Contoh 3: Contoh ini menghapus tag yang ditentukan dari sumber daya yang ditentukan, terlepas dari nilai tag.

```
$tag = New-Object Amazon.EC2.Model.Tag
$tag.Key = "myTag"

Remove-EC2Tag -Resource i-12345678 -Tag $tag -Force
```

Contoh 4: Contoh ini menghapus tag yang ditentukan dari sumber daya yang ditentukan, tetapi hanya jika nilai tag cocok.

```
$tag = New-Object Amazon.EC2.Model.Tag
$tag.Key = "myTag"
$tag.Value = "myTagValue"

Remove-EC2Tag -Resource i-12345678 -Tag $tag -Force
```

- Untuk detail API, lihat [DeleteTags](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-EC2Volume

Contoh kode berikut menunjukkan cara menggunakan `Remove-EC2Volume`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini melepaskan volume yang ditentukan. Anda diminta untuk konfirmasi sebelum operasi berlangsung, kecuali jika Anda juga menentukan parameter Force.

```
Remove-EC2Volume -VolumeId vol-12345678
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-EC2Volume (DeleteVolume)" on target "vol-12345678".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is "Y"):
```

- Untuk detail API, lihat [DeleteVolume](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-EC2Vpc

Contoh kode berikut menunjukkan cara menggunakan Remove-EC2Vpc.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus VPC yang ditentukan. Anda diminta untuk konfirmasi sebelum operasi berlangsung, kecuali jika Anda juga menentukan parameter Force.

```
Remove-EC2Vpc -VpcId vpc-12345678
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2Vpc (DeleteVpc)" on Target "vpc-12345678".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is "Y"):
```

- Untuk detail API, lihat [DeleteVpc](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-EC2VpnConnection

Contoh kode berikut menunjukkan cara menggunakan Remove-EC2VpnConnection.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus koneksi VPN yang ditentukan. Anda diminta untuk konfirmasi sebelum operasi berlangsung, kecuali jika Anda juga menentukan parameter Force.

```
Remove-EC2VpnConnection -VpnConnectionId vpn-12345678
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2VpnConnection (DeleteVpnConnection)" on Target
"vpn-12345678".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"):
```

- Untuk detail API, lihat [DeleteVpnConnection](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-EC2VpnConnectionRoute

Contoh kode berikut menunjukkan cara menggunakan `Remove-EC2VpnConnectionRoute`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus rute statis yang ditentukan dari koneksi VPN yang ditentukan. Anda diminta untuk konfirmasi sebelum operasi berlangsung, kecuali jika Anda juga menentukan parameter Force.

```
Remove-EC2VpnConnectionRoute -VpnConnectionId vpn-12345678 -DestinationCidrBlock
11.12.0.0/16
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2VpnConnectionRoute (DeleteVpnConnectionRoute)" on
Target "vpn-12345678".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"):
```

- Untuk detail API, lihat [DeleteVpnConnectionRoute](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-EC2VpnGateway

Contoh kode berikut menunjukkan cara menggunakan `Remove-EC2VpnGateway`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus gateway pribadi virtual yang ditentukan. Anda diminta untuk konfirmasi sebelum operasi berlangsung, kecuali jika Anda juga menentukan parameter `Force`.

```
Remove-EC2VpnGateway -VpnGatewayId vgw-1a2b3c4d
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2VpnGateway (DeleteVpnGateway)" on Target
"vgw-1a2b3c4d".
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is
"Y"):
```

- Untuk detail API, lihat [DeleteVpnGateway](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Request-EC2SpotFleet

Contoh kode berikut menunjukkan cara menggunakan `Request-EC2SpotFleet`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat permintaan armada Spot di Availability Zone dengan harga terendah untuk jenis instans yang ditentukan. Jika akun Anda hanya mendukung EC2 -VPC, armada Spot akan meluncurkan instans di Availability Zone dengan harga terendah yang memiliki subnet default. Jika akun Anda mendukung EC2 -Classic, armada Spot meluncurkan instans di EC2 -Classic di Availability Zone dengan harga terendah. Perhatikan bahwa harga yang Anda bayar tidak akan melebihi harga Spot yang ditentukan untuk permintaan tersebut.

```
$sg = New-Object Amazon.EC2.Model.GroupIdentifier
```

```
$sg.GroupId = "sg-12345678"
$l1c = New-Object Amazon.EC2.Model.SpotFleetLaunchSpecification
$l1c.ImageId = "ami-12345678"
$l1c.InstanceType = "m3.medium"
$l1c.SecurityGroups.Add($sg)
Request-EC2SpotFleet -SpotFleetRequestConfig_SpotPrice 0.04 `
-SpotFleetRequestConfig_TargetCapacity 2 `
-SpotFleetRequestConfig_IamFleetRole arn:aws:iam::123456789012:role/my-spot-fleet-
role `
-SpotFleetRequestConfig_LaunchSpecification $l1c
```

- Untuk detail API, lihat [RequestSpotFleet](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Request-EC2SpotInstance

Contoh kode berikut menunjukkan cara menggunakan `Request-EC2SpotInstance`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini meminta instance Spot satu kali di subnet yang ditentukan. Perhatikan bahwa grup keamanan harus dibuat untuk VPC yang berisi subnet yang ditentukan, dan harus ditentukan oleh ID menggunakan antarmuka jaringan. Saat Anda menentukan antarmuka jaringan, Anda harus menyertakan ID subnet menggunakan antarmuka jaringan.

```
$n = New-Object Amazon.EC2.Model.InstanceNetworkInterfaceSpecification
$n.DeviceIndex = 0
$n.SubnetId = "subnet-12345678"
$n.Groups.Add("sg-12345678")
Request-EC2SpotInstance -InstanceCount 1 -SpotPrice 0.050 -Type one-time `
-IamInstanceProfile_Arn arn:aws:iam::123456789012:instance-profile/my-iam-role `
-LaunchSpecification_ImageId ami-12345678 `
-LaunchSpecification_InstanceType m3.medium `
-LaunchSpecification_NetworkInterface $n
```

Output:

```
ActualBlockHourlyPrice      :
AvailabilityZoneGroup       :
BlockDurationMinutes        : 0
CreateTime                  : 12/26/2015 7:44:10 AM
Fault                        :
```

```

InstanceId           :
LaunchedAvailabilityZone :
LaunchGroup          :
LaunchSpecification   : Amazon.EC2.Model.LaunchSpecification
ProductDescription    : Linux/UNIX
SpotInstanceRequestId : sir-12345678
SpotPrice             : 0.050000
State                 : open
Status                : Amazon.EC2.Model.SpotInstanceStatus
Tags                  : {}
Type                  : one-time

```

- Untuk detail API, lihat [RequestSpotInstances](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Reset-EC2ImageAttribute

Contoh kode berikut menunjukkan cara menggunakan `Reset-EC2ImageAttribute`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini me-reset atribut 'launchPermission' ke nilai defaultnya. Secara default, AMIs bersifat pribadi.

```
Reset-EC2ImageAttribute -ImageId ami-12345678 -Attribute launchPermission
```

- Untuk detail API, lihat [ResetImageAttribute](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Reset-EC2InstanceAttribute

Contoh kode berikut menunjukkan cara menggunakan `Reset-EC2InstanceAttribute`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini me-reset atribut `sriovNetSupport` untuk instance tertentu.

```
Reset-EC2InstanceAttribute -InstanceId i-12345678 -Attribute sriovNetSupport
```

Contoh 2: Contoh ini me-reset atribut 'EBSOptimized' untuk instance tertentu.

```
Reset-EC2InstanceAttribute -InstanceId i-12345678 -Attribute ebsOptimized
```

Contoh 3: Contoh ini me-reset atribut sourceDestCheck " untuk instance tertentu.

```
Reset-EC2InstanceAttribute -InstanceId i-12345678 -Attribute sourceDestCheck
```

Contoh 4: Contoh ini me-reset atribut disableApiTermination " untuk instance tertentu.

```
Reset-EC2InstanceAttribute -InstanceId i-12345678 -Attribute disableApiTermination
```

Contoh 5: Contoh ini me-reset atribut 'instanceInitiatedShutdownBehavior' untuk instance tertentu.

```
Reset-EC2InstanceAttribute -InstanceId i-12345678 -Attribute  
instanceInitiatedShutdownBehavior
```

- Untuk detail API, lihat [ResetInstanceAttribute](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Reset-EC2NetworkInterfaceAttribute

Contoh kode berikut menunjukkan cara menggunakan `Reset-EC2NetworkInterfaceAttribute`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengatur ulang pemeriksaan sumber/tujuan untuk antarmuka jaringan yang ditentukan.

```
Reset-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-1a2b3c4d -SourceDestCheck
```

- Untuk detail API, lihat [ResetNetworkInterfaceAttribute](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Reset-EC2SnapshotAttribute

Contoh kode berikut menunjukkan cara menggunakan `Reset-EC2SnapshotAttribute`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengatur ulang atribut tertentu dari snapshot yang ditentukan.

```
Reset-EC2SnapshotAttribute -SnapshotId snap-12345678 -Attribute
CreateVolumePermission
```

- Untuk detail API, lihat [ResetSnapshotAttribute](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Restart-EC2Instance

Contoh kode berikut menunjukkan cara menggunakan `Restart-EC2Instance`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini me-reboot instance yang ditentukan.

```
Restart-EC2Instance -InstanceId i-12345678
```

- Untuk detail API, lihat [RebootInstances](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Revoke-EC2SecurityGroupEgress

Contoh kode berikut menunjukkan cara menggunakan `Revoke-EC2SecurityGroupEgress`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus aturan untuk grup keamanan yang ditentukan untuk EC2 - VPC. Ini mencabut akses ke rentang alamat IP yang ditentukan pada port TCP 80. Sintaks yang digunakan oleh contoh ini membutuhkan PowerShell versi 3 atau lebih tinggi.

```
$ip = @{ IpProtocol="tcp"; FromPort="80"; ToPort="80"; IpRanges="203.0.113.0/24" }
Revoke-EC2SecurityGroupEgress -GroupId sg-12345678 -IpPermission $ip
```

Contoh 2: Dengan PowerShell versi 2, Anda harus menggunakan `New-Object` untuk membuat objek. `IpPermission`

```
$ip = New-Object Amazon.EC2.Model.IpPermission
$ip.IpProtocol = "tcp"
$ip.FromPort = 80
$ip.ToPort = 80
$ip.IpRanges.Add("203.0.113.0/24")
Revoke-EC2SecurityGroupEgress -GroupId sg-12345678 -IpPermission $ip
```

Contoh 3: Contoh ini mencabut akses ke grup keamanan sumber yang ditentukan pada port TCP 80.

```
$ug = New-Object Amazon.EC2.Model.UserIdGroupPair
$ug.GroupId = "sg-1a2b3c4d"
$ug.UserId = "123456789012"
Revoke-EC2SecurityGroupEgress -GroupId sg-12345678 -IpPermission
@( @{ IpProtocol="tcp"; FromPort="80"; ToPort="80"; UserIdGroupPairs=$ug } )
```

- Untuk detail API, lihat [RevokeSecurityGroupEgress](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Revoke-EC2SecurityGroupIngress

Contoh kode berikut menunjukkan cara menggunakan `Revoke-EC2SecurityGroupIngress`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencabut akses ke port TCP 22 dari rentang alamat yang ditentukan untuk grup keamanan yang ditentukan untuk -VPC. EC2 Perhatikan bahwa Anda harus mengidentifikasi grup keamanan untuk EC2 -VPC menggunakan ID grup keamanan bukan nama grup keamanan. Sintaks yang digunakan oleh contoh ini membutuhkan PowerShell versi 3 atau lebih tinggi.

```
$ip = @{ IpProtocol="tcp"; FromPort="22"; ToPort="22"; IpRanges="203.0.113.0/24" }
Revoke-EC2SecurityGroupIngress -GroupId sg-12345678 -IpPermission $ip
```

Contoh 2: Dengan PowerShell versi 2, Anda harus menggunakan `New-Object` untuk membuat objek. `IpPermission`

```
$ip = New-Object Amazon.EC2.Model.IpPermission
$ip.IpProtocol = "tcp"
$ip.FromPort = 22
$ip.ToPort = 22
$ip.IpRanges.Add("203.0.113.0/24")

Revoke-EC2SecurityGroupIngress -GroupId sg-12345678 -IpPermission $ip
```

Contoh 3: Contoh ini mencabut akses ke port TCP 22 dari rentang alamat yang ditentukan untuk grup keamanan yang ditentukan untuk -Classic. EC2 Sintaks yang digunakan oleh contoh ini membutuhkan PowerShell versi 3 atau lebih tinggi.

```
$ip = @{ IpProtocol="tcp"; FromPort="22"; ToPort="22"; IpRanges="203.0.113.0/24" }

Revoke-EC2SecurityGroupIngress -GroupName "my-security-group" -IpPermission $ip
```

Contoh 4: Dengan PowerShell versi 2, Anda harus menggunakan New-Object untuk membuat objek. IpPermission

```
$ip = New-Object Amazon.EC2.Model.IpPermission
$ip.IpProtocol = "tcp"
$ip.FromPort = 22
$ip.ToPort = 22
$ip.IpRanges.Add("203.0.113.0/24")

Revoke-EC2SecurityGroupIngress -GroupName "my-security-group" -IpPermission $ip
```

- Untuk detail API, lihat [RevokeSecurityGroupIngress](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Send-EC2InstanceState

Contoh kode berikut menunjukkan cara menggunakan Send-EC2InstanceState.

Alat untuk PowerShell V4

Contoh 1: Contoh ini melaporkan umpan balik status untuk contoh yang ditentukan.

```
Send-EC2InstanceState -Instance i-12345678 -Status impaired -ReasonCode
unresponsive
```

- Untuk detail API, lihat [ReportInstanceState](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Set-EC2NetworkAclAssociation

Contoh kode berikut menunjukkan cara menggunakan Set-EC2NetworkAclAssociation.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengaitkan ACL jaringan tertentu dengan subnet untuk asosiasi ACL jaringan tertentu.

```
Set-EC2NetworkAclAssociation -NetworkAclId acl-12345678 -AssociationId  
aclassoc-1a2b3c4d
```

Output:

```
aclassoc-87654321
```

- Untuk detail API, lihat [ReplaceNetworkAclAssociation](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Set-EC2NetworkAclEntry

Contoh kode berikut menunjukkan cara menggunakan `Set-EC2NetworkAclEntry`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menggantikan entri yang ditentukan untuk ACL jaringan tertentu. Aturan baru memungkinkan lalu lintas masuk dari alamat yang ditentukan ke subnet terkait.

```
Set-EC2NetworkAclEntry -NetworkAclId acl-12345678 -Egress $false -RuleNumber 100  
-Protocol 17 -PortRange_From 53 -PortRange_To 53 -CidrBlock 203.0.113.12/24 -  
RuleAction allow
```

- Untuk detail API, lihat [ReplaceNetworkAclEntry](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Set-EC2Route

Contoh kode berikut menunjukkan cara menggunakan `Set-EC2Route`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menggantikan rute yang ditentukan untuk tabel rute yang ditentukan. Rute baru mengirimkan lalu lintas yang ditentukan ke gateway pribadi virtual yang ditentukan.

```
Set-EC2Route -RouteTableId rtb-1a2b3c4d -DestinationCidrBlock 10.0.0.0/24 -GatewayId  
vgw-1a2b3c4d
```

- Untuk detail API, lihat [ReplaceRoute](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Set-EC2RouteTableAssociation

Contoh kode berikut menunjukkan cara menggunakan `Set-EC2RouteTableAssociation`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengaitkan tabel rute yang ditentukan dengan subnet untuk asosiasi tabel rute yang ditentukan.

```
Set-EC2RouteTableAssociation -RouteTableId rtb-1a2b3c4d -AssociationId
rtbassoc-12345678
```

Output:

```
rtbassoc-87654321
```

- Untuk detail API, lihat [ReplaceRouteTableAssociation](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Start-EC2Instance

Contoh kode berikut menunjukkan cara menggunakan `Start-EC2Instance`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memulai instance yang ditentukan.

```
Start-EC2Instance -InstanceId i-12345678
```

Output:

CurrentState	InstanceId	PreviousState
-----	-----	-----
Amazon.EC2.Model.InstanceState	i-12345678	Amazon.EC2.Model.InstanceState

Contoh 2: Contoh ini memulai instance yang ditentukan.

```
@("i-12345678", "i-76543210") | Start-EC2Instance
```

Contoh 3: Contoh ini memulai kumpulan instance yang saat ini dihentikan. Objek Instance yang dikembalikan oleh `Get-EC2Instance` disalurkan ke `Start-EC2Instance`. Sintaks yang digunakan oleh contoh ini membutuhkan PowerShell versi 3 atau lebih tinggi.

```
(Get-EC2Instance -Filter @{ Name="instance-state-name"; Values="stopped"}).Instances
| Start-EC2Instance
```

Contoh 4: Dengan PowerShell versi 2, Anda harus menggunakan `New-Object` untuk membuat filter untuk parameter `Filter`.

```
$filter = New-Object Amazon.EC2.Model.Filter
$filter.Name = "instance-state-name"
$filter.Values = "stopped"

(Get-EC2Instance -Filter $filter).Instances | Start-EC2Instance
```

- Untuk detail API, lihat [StartInstances](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Start-EC2InstanceMonitoring

Contoh kode berikut menunjukkan cara menggunakan `Start-EC2InstanceMonitoring`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memungkinkan pemantauan terperinci untuk contoh yang ditentukan.

```
Start-EC2InstanceMonitoring -InstanceId i-12345678
```

Output:

InstanceId	Monitoring
-----	-----
i-12345678	Amazon.EC2.Model.Monitoring

- Untuk detail API, lihat [MonitorInstances](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Stop-EC2ImportTask

Contoh kode berikut menunjukkan cara menggunakan `Stop-EC2ImportTask`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membatalkan tugas impor yang ditentukan (baik snapshot atau impor gambar). Jika diperlukan, alasan dapat menyediakan menggunakan **-CancelReason** parameter.

```
Stop-EC2ImportTask -ImportTaskId import-ami-abcdefgh
```

- Untuk detail API, lihat [CancelImportTask](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Stop-EC2Instance

Contoh kode berikut menunjukkan cara menggunakan `Stop-EC2Instance`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghentikan instance yang ditentukan.

```
Stop-EC2Instance -InstanceId i-12345678
```

Output:

CurrentState	InstanceId	PreviousState
-----	-----	-----
Amazon.EC2.Model.InstanceState	i-12345678	Amazon.EC2.Model.InstanceState

- Untuk detail API, lihat [StopInstances](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Stop-EC2InstanceMonitoring

Contoh kode berikut menunjukkan cara menggunakan `Stop-EC2InstanceMonitoring`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menonaktifkan pemantauan terperinci untuk contoh yang ditentukan.

```
Stop-EC2InstanceMonitoring -InstanceId i-12345678
```

Output:

InstanceId	Monitoring
------------	------------

```
-----
i-12345678    Amazon.EC2.Model.Monitoring
```

- Untuk detail API, lihat [UnmonitorInstances](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Stop-EC2SpotFleetRequest

Contoh kode berikut menunjukkan cara menggunakan `Stop-EC2SpotFleetRequest`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membatalkan permintaan armada Spot yang ditentukan dan mengakhiri instance Spot terkait.

```
Stop-EC2SpotFleetRequest -SpotFleetRequestId sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE -TerminateInstance $true
```

Contoh 2: Contoh ini membatalkan permintaan armada Spot yang ditentukan tanpa menghentikan instans Spot terkait.

```
Stop-EC2SpotFleetRequest -SpotFleetRequestId sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE -TerminateInstance $false
```

- Untuk detail API, lihat [CancelSpotFleetRequests](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Stop-EC2SpotInstanceRequest

Contoh kode berikut menunjukkan cara menggunakan `Stop-EC2SpotInstanceRequest`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membatalkan permintaan instance Spot yang ditentukan.

```
Stop-EC2SpotInstanceRequest -SpotInstanceRequestId sir-12345678
```

Output:

```
SpotInstanceRequestId    State
```

```
-----  
sir-12345678      cancelled
```

- Untuk detail API, lihat [CancelSpotInstanceRequests](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Unregister-EC2Address

Contoh kode berikut menunjukkan cara menggunakan `Unregister-EC2Address`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memisahkan alamat IP Elastis yang ditentukan dari instance yang ditentukan dalam VPC.

```
Unregister-EC2Address -AssociationId eipassoc-12345678
```

Contoh 2: Contoh ini memisahkan alamat IP Elastis yang ditentukan dari instance yang ditentukan di EC2 -Classic.

```
Unregister-EC2Address -PublicIp 203.0.113.17
```

- Untuk detail API, lihat [DisassociateAddress](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Unregister-EC2Image

Contoh kode berikut menunjukkan cara menggunakan `Unregister-EC2Image`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membatalkan pendaftaran AMI yang ditentukan.

```
Unregister-EC2Image -ImageId ami-12345678
```

- Untuk detail API, lihat [DeregisterImage](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Unregister-EC2PrivateIpAddress

Contoh kode berikut menunjukkan cara menggunakan `Unregister-EC2PrivateIpAddress`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membatalkan penetapan alamat IP pribadi yang ditentukan dari antarmuka jaringan yang ditentukan.

```
Unregister-EC2PrivateIpAddress -NetworkInterfaceId eni-1a2b3c4d -PrivateIpAddress 10.0.0.82
```

- Untuk detail API, lihat [UnassignPrivateIpAddresses](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Unregister-EC2RouteTable

Contoh kode berikut menunjukkan cara menggunakan `Unregister-EC2RouteTable`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus asosiasi tertentu antara tabel rute dan subnet.

```
Unregister-EC2RouteTable -AssociationId rtbassoc-1a2b3c4d
```

- Untuk detail API, lihat [DisassociateRouteTable](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-EC2SecurityGroupRuleIngressDescription

Contoh kode berikut menunjukkan cara menggunakan `Update-EC2SecurityGroupRuleIngressDescription`.

Alat untuk PowerShell V4

Contoh 1: Memperbarui deskripsi aturan grup keamanan ingress (inbound) yang ada.

```
$existingInboundRule = Get-EC2SecurityGroupRule -SecurityGroupRuleId "sgr-1234567890"
$ruleWithUpdatedDescription = [Amazon.EC2.Model.SecurityGroupRuleDescription]@{
    "SecurityGroupRuleId" = $existingInboundRule.SecurityGroupRuleId
    "Description" = "Updated rule description"
}
```

```
Update-EC2SecurityGroupRuleIngressDescription -GroupId $existingInboundRule.GroupId
-SecurityGroupRuleDescription $ruleWithUpdatedDescription
```

Contoh 2: Menghapus deskripsi aturan grup keamanan ingress (inbound) yang ada (dengan menghilangkan parameter dalam permintaan).

```
$existingInboundRule = Get-EC2SecurityGroupRule -SecurityGroupRuleId
"sgr-1234567890"
$ruleWithoutDescription = [Amazon.EC2.Model.SecurityGroupRuleDescription]@{
    "SecurityGroupId" = $existingInboundRule.SecurityGroupId
}

Update-EC2SecurityGroupRuleIngressDescription -GroupId $existingInboundRule.GroupId
-SecurityGroupRuleDescription $ruleWithoutDescription
```

- Untuk detail API, lihat [UpdateSecurityGroupRuleDescriptionsIngress](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Contoh Amazon ECR menggunakan Alat untuk V4 PowerShell

Contoh kode berikut menunjukkan cara melakukan tindakan dan menerapkan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan Amazon ECR.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

Get-ECRLoginCommand

Contoh kode berikut menunjukkan cara menggunakan `Get-ECRLoginCommand`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan informasi login yang PObject berisi yang dapat digunakan untuk mengautentikasi ke registri Amazon ECR mana pun yang dapat diakses oleh kepala sekolah IAM Anda. Kredensi dan titik akhir wilayah yang diperlukan untuk panggilan untuk mendapatkan token otorisasi diperoleh dari default shell (diatur oleh atau cmdlet). **Set-AWSCredential/Set-DefaultAWSRegion Initialize-AWSDefaultConfiguration** Anda dapat menggunakan properti Command dengan Invoke-Expression untuk masuk ke registri yang ditentukan atau menggunakan kredensi yang dikembalikan di alat lain yang memerlukan login.

```
Get-ECRLoginCommand
```

Output:

```
Username      : AWS
Password      : eyJwYXlsb2Fk...kRBVEffS0VZIn0=
ProxyEndpoint : https://123456789012.dkr.ecr.us-west-2.amazonaws.com
Endpoint      : https://123456789012.dkr.ecr.us-west-2.amazonaws.com
ExpiresAt     : 9/26/2017 6:08:23 AM
Command       : docker login --username AWS --password
eyJwYXlsb2Fk...kRBVEffS0VZIn0= https://123456789012.dkr.ecr.us-west-2.amazonaws.com
```

Contoh 2: Mengambil informasi login yang PObject berisi yang Anda gunakan sebagai input ke perintah docker login. Anda dapat menentukan URI registri Amazon ECR apa pun untuk diautentikasi selama prinsipal IAM Anda memiliki akses ke registri tersebut.

```
(Get-ECRLoginCommand).Password | docker login --username AWS --password-stdin
012345678910.dkr.ecr.us-east-1.amazonaws.com
```

- Untuk detail API, lihat [Dapatkan- ECRLogin Perintah](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Contoh Amazon ECS menggunakan Alat untuk PowerShell V4

Contoh kode berikut menunjukkan cara melakukan tindakan dan menerapkan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan Amazon ECS.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

Get-ECSClusterDetail

Contoh kode berikut menunjukkan cara menggunakan `Get-ECSClusterDetail`.

Alat untuk PowerShell V4

Contoh 1: Cmdlet ini menjelaskan satu atau lebih cluster ECS Anda.

```
Get-ECSClusterDetail -Cluster "LAB-ECS-CL" -Include SETTINGS | Select-Object *
```

Output:

```
LoggedAt      : 12/27/2019 9:27:41 PM
Clusters      : {LAB-ECS-CL}
Failures      : {}
ResponseMetadata : Amazon.Runtime.ResponseMetadata
ContentLength  : 396
HttpStatusCode : OK
```

- Untuk detail API, lihat [DescribeClusters](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-ECSClusterList

Contoh kode berikut menunjukkan cara menggunakan `Get-ECSClusterList`.

Alat untuk PowerShell V4

Contoh 1: Cmdlet ini mengembalikan daftar cluster ECS yang ada.

```
Get-ECSClusterList
```

Output:

```
arn:aws:ecs:us-west-2:012345678912:cluster/LAB-ECS-CL  
arn:aws:ecs:us-west-2:012345678912:cluster/LAB-ECS
```

- Untuk detail API, lihat [ListClusters](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-ECSClusterService

Contoh kode berikut menunjukkan cara menggunakan `Get-ECSClusterService`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan semua layanan yang berjalan di klaster default Anda.

```
Get-ECSClusterService
```

Contoh 2: Contoh ini mencantumkan semua layanan yang berjalan di cluster yang ditentukan.

```
Get-ECSClusterService -Cluster myCluster
```

- Untuk detail API, lihat [ListServices](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-ECSService

Contoh kode berikut menunjukkan cara menggunakan `Get-ECSService`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menunjukkan cara mengambil detail layanan tertentu dari klaster default Anda.

```
Get-ECSService -Service my-http-service
```

Contoh 2: Contoh ini menunjukkan cara mengambil rincian layanan tertentu yang berjalan di cluster bernama.

```
Get-ECSService -Cluster myCluster -Service my-http-service
```

- Untuk detail API, lihat [DescribeServices](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-ECSCluster

Contoh kode berikut menunjukkan cara menggunakan `New-ECSCluster`.

Alat untuk PowerShell V4

Contoh 1: Cmdlet ini membuat cluster Amazon ECS baru.

```
New-ECSCluster -ClusterName "LAB-ECS-CL" -Setting @{"Name"="containerInsights";
Value="enabled"}
```

Output:

```
ActiveServicesCount      : 0
Attachments              : {}
AttachmentsStatus       :
CapacityProviders       : {}
ClusterArn               : arn:aws:ecs:us-west-2:012345678912:cluster/LAB-
ECS-CL
ClusterName              : LAB-ECS-CL
DefaultCapacityProviderStrategy : {}
PendingTasksCount       : 0
RegisteredContainerInstancesCount : 0
RunningTasksCount       : 0
Settings                 : {containerInsights}
Statistics               : {}
Status                   : ACTIVE
Tags                     : {}
```

- Untuk detail API, lihat [CreateCluster](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-ECSService

Contoh kode berikut menunjukkan cara menggunakan `New-ECSService`.

Alat untuk PowerShell V4

Contoh 1: Perintah contoh ini membuat layanan di cluster default Anda yang disebut `ecs-simple-service`. Layanan ini menggunakan definisi tugas `ecs-demo` dan mempertahankan 10 instantiasi tugas itu.

```
New-ECSService -ServiceName ecs-simple-service -TaskDefinition ecs-demo -
DesiredCount 10
```

Contoh 2: Perintah contoh ini membuat layanan di belakang penyeimbang beban di cluster default Anda yang disebut `ecs-simple-service`. Layanan ini menggunakan definisi tugas `ecs-demo` dan mempertahankan 10 instantiasi tugas itu.

```
$lb = @{
    LoadBalancerName = "EC2Contai-EcsElast-S06278JGSJCM"
    ContainerName = "simple-demo"
    ContainerPort = 80
}
New-ECSService -ServiceName ecs-simple-service -TaskDefinition ecs-demo -
DesiredCount 10 -LoadBalancer $lb
```

- Untuk detail API, lihat [CreateService](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-ECSCluster

Contoh kode berikut menunjukkan cara menggunakan `Remove-ECSCluster`.

Alat untuk PowerShell V4

Contoh 1: Cmdlet ini menghapus cluster ECS yang ditentukan. Anda harus membatalkan pendaftaran semua instance kontainer dari cluster ini sebelum Anda dapat menghapusnya.

```
Remove-ECSCluster -Cluster "LAB-ECS"
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-ECSCluster (DeleteCluster)" on target "LAB-ECS".
```

```
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "Y"): Y
```

- Untuk detail API, lihat [DeleteCluster](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-ECSService

Contoh kode berikut menunjukkan cara menggunakan `Remove-ECSService`.

Alat untuk PowerShell V4

Contoh 1: Menghapus layanan bernama 'my-http-service' di cluster default. Layanan harus memiliki hitungan yang diinginkan dan menjalankan hitungan 0 sebelum Anda dapat menghapusnya. Anda diminta untuk konfirmasi sebelum perintah dilanjutkan. Untuk melewati prompt konfirmasi tambahkan sakelar `-Force`.

```
Remove-ECSService -Service my-http-service
```

Contoh 2: Menghapus layanan bernama 'my-http-service' di cluster bernama.

```
Remove-ECSService -Cluster myCluster -Service my-http-service
```

- Untuk detail API, lihat [DeleteService](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-ECSClusterSetting

Contoh kode berikut menunjukkan cara menggunakan `Update-ECSClusterSetting`.

Alat untuk PowerShell V4

Contoh 1: Cmdlet ini memodifikasi pengaturan yang akan digunakan untuk cluster ECS.

```
Update-ECSClusterSetting -Cluster "LAB-ECS-CL" -Setting @{Name="containerInsights"; Value="disabled"}
```

Output:

```
ActiveServicesCount      : 0
Attachments              : {}
```

```

AttachmentsStatus      :
CapacityProviders      : {}
ClusterArn             : arn:aws:ecs:us-west-2:012345678912:cluster/LAB-
ECS-CL
ClusterName            : LAB-ECS-CL
DefaultCapacityProviderStrategy : {}
PendingTasksCount      : 0
RegisteredContainerInstancesCount : 0
RunningTasksCount      : 0
Settings               : {containerInsights}
Statistics              : {}
Status                 : ACTIVE
Tags                   : {}

```

- Untuk detail API, lihat [UpdateClusterSettings](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-ECSService

Contoh kode berikut menunjukkan cara menggunakan `Update-ECSService`.

Alat untuk PowerShell V4

Contoh 1: Perintah contoh ini memperbarui layanan `my-http-service` untuk menggunakan definisi tugas `amazon-ecs-sample`.

```
Update-ECSService -Service my-http-service -TaskDefinition amazon-ecs-sample
```

Contoh 2: Perintah contoh ini memperbarui jumlah yang diinginkan dari layanan `my-http-service` ke 10.

```
Update-ECSService -Service my-http-service -DesiredCount 10
```

- Untuk detail API, lihat [UpdateService](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Amazon EFS contoh menggunakan Alat untuk PowerShell V4

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan Amazon EFS.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

Edit-EFSMountTargetSecurityGroup

Contoh kode berikut menunjukkan cara menggunakan `Edit-EFSMountTargetSecurityGroup`.

Alat untuk PowerShell V4

Contoh 1: Memperbarui grup keamanan yang berlaku untuk target pemasangan yang ditentukan. Hingga 5 dapat ditentukan, dalam format "sg-xxxxxxx".

```
Edit-EFSMountTargetSecurityGroup -MountTargetId fsmt-1a2b3c4d -SecurityGroup sg-group1,sg-group3
```

- Untuk detail API, lihat [ModifyMountTargetSecurityGroups](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EFSFileSystem

Contoh kode berikut menunjukkan cara menggunakan `Get-EFSFileSystem`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan koleksi semua sistem file yang dimiliki oleh akun penelepon di wilayah tersebut.

```
Get-EFSFileSystem
```

Output:

```

CreationTime      : 5/26/2015 4:02:38 PM
CreationToken     : 1a2bff54-85e0-4747-bd95-7bc172c4f555
FileSystemId      : fs-1a2b3c4d
LifeCycleState    : available
Name              :
NumberOfMountTargets : 0
OwnerId           : 123456789012
SizeInBytes       : Amazon.ElasticFileSystem.Model.FileSystemSize

CreationTime      : 5/26/2015 4:06:23 PM
CreationToken     : 2b4daa14-85e0-4747-bd95-7bc172c4f555
FileSystemId      : fs-4d3c2b1a
...

```

Contoh 2: Mengembalikan rincian sistem file yang ditentukan.

```
Get-EFSFileSystem -FileSystemId fs-1a2b3c4d
```

Contoh 3: Mengembalikan rincian sistem file menggunakan token penciptaan idempotensi yang ditentukan pada saat sistem file dibuat.

```
Get-EFSFileSystem -CreationToken 1a2bff54-85e0-4747-bd95-7bc172c4f555
```

- Untuk detail API, lihat [DescribeFileSystems](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EFSMountTarget

Contoh kode berikut menunjukkan cara menggunakan `Get-EFSMountTarget`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan koleksi target mount yang terkait dengan sistem file yang ditentukan.

```
Get-EFSMountTarget -FileSystemId fs-1a2b3c4d
```

Output:

```

FileSystemId      : fs-1a2b3c4d
IpAddress         : 10.0.0.131

```

```

LifeCycleState      : available
MountTargetId       : fsmt-1a2b3c4d
NetworkInterfaceId  : eni-1a2b3c4d
OwnerId              : 123456789012
SubnetId             : subnet-1a2b3c4d

```

- Untuk detail API, lihat [DescribeMountTargets](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EFSMountTargetSecurityGroup

Contoh kode berikut menunjukkan cara menggunakan `Get-EFSMountTargetSecurityGroup`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan id grup keamanan yang saat ini ditetapkan ke antarmuka jaringan yang terkait dengan target pemasangan.

```
Get-EFSMountTargetSecurityGroup -MountTargetId fsmt-1a2b3c4d
```

Output:

```
sg-1a2b3c4d
```

- Untuk detail API, lihat [DescribeMountTargetSecurityGroups](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EFSTag

Contoh kode berikut menunjukkan cara menggunakan `Get-EFSTag`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan koleksi tag saat ini terkait dengan sistem file tertentu.

```
Get-EFSTag -FileSystemId fs-1a2b3c4d
```

Output:

Key	Value
-----	-------

```

---      -----
Name      My File System
tagkey1    tagvalue1
tagkey2    tagvalue2

```

- Untuk detail API, lihat [DescribeTags](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-EFSFileSystem

Contoh kode berikut menunjukkan cara menggunakan `New-EFSFileSystem`.

Alat untuk PowerShell V4

Contoh 1: Membuat sistem file baru yang kosong. Token yang digunakan untuk memastikan pembuatan idempoten akan dihasilkan secara otomatis dan dapat diakses dari **CreationToken** anggota objek yang dikembalikan.

```
New-EFSFileSystem
```

Output:

```

CreationTime      : 5/26/2015 4:02:38 PM
CreationToken      : 1a2bff54-85e0-4747-bd95-7bc172c4f555
FileSystemId       : fs-1a2b3c4d
LifecycleState     : creating
Name               :
NumberOfMountTargets : 0
OwnerId            : 123456789012
SizeInBytes        : Amazon.ElasticFileSystem.Model.FileSystemSize

```

Contoh 2: Membuat sistem file baru yang kosong menggunakan token khusus untuk memastikan pembuatan idempoten.

```
New-EFSFileSystem -CreationToken "MyUniqueToken"
```

- Untuk detail API, lihat [CreateFileSystem](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-EFSMountTarget

Contoh kode berikut menunjukkan cara menggunakan `New-EFSMountTarget`.

Alat untuk PowerShell V4

Contoh 1: Membuat target mount baru untuk sistem file. Subnet yang ditentukan akan digunakan untuk menentukan Virtual Private Cloud (VPC) tempat target mount akan dibuat dan alamat IP yang akan ditetapkan secara otomatis (dari kisaran alamat subnet). Alamat IP yang ditetapkan dapat digunakan untuk kemudian me-mount sistem file ini pada EC2 instance Amazon. Karena tidak ada grup keamanan yang ditentukan, antarmuka jaringan yang dibuat untuk target dikaitkan dengan grup keamanan default untuk VPC subnet.

```
New-EFSMountTarget -FileSystemId fs-1a2b3c4d -SubnetId subnet-1a2b3c4d
```

Output:

```
FileSystemId      : fs-1a2b3c4d
IpAddress         : 10.0.0.131
LifeCycleState    : creating
MountTargetId     : fsmt-1a2b3c4d
NetworkInterfaceId : eni-1a2b3c4d
OwnerId           : 123456789012
SubnetId          : subnet-1a2b3c4d
```

Contoh 2: Membuat target mount baru untuk sistem file yang ditentukan dengan alamat IP yang ditetapkan secara otomatis. Antarmuka jaringan yang dibuat untuk target mount dikaitkan dengan grup keamanan yang ditentukan (hingga 5, dalam format “sg-xxxxxxx”, dapat ditentukan).

```
New-EFSMountTarget -FileSystemId fs-1a2b3c4d -SubnetId subnet-1a2b3c4d -
SecurityGroup sg-group1,sg-group2,sg-group3
```

Contoh 3: Membuat target mount baru untuk sistem file yang ditentukan dengan alamat IP yang ditentukan.

```
New-EFSMountTarget -FileSystemId fs-1a2b3c4d -SubnetId subnet-1a2b3c4d -IpAddress
10.0.0.131
```

- Untuk detail API, lihat [CreateMountTarget](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-EFSTag

Contoh kode berikut menunjukkan cara menggunakan New-EFSTag.

Alat untuk PowerShell V4

Contoh 1: Menerapkan koleksi tag ke sistem file yang ditentukan. Jika tag dengan kunci yang ditentukan sudah ada pada sistem file, nilai tag diperbarui.

```
New-EFSTag -FileSystemId fs-1a2b3c4d -Tag  
@{Key="tagkey1";Value="tagvalue1"},@{Key="tagkey2";Value="tagvalue2"}
```

Contoh 2: Menetapkan tag nama untuk sistem file yang ditentukan. Nilai ini dikembalikan bersama dengan rincian sistem file lainnya ketika Get-EFSFileSystem cmdlet digunakan.

```
New-EFSTag -FileSystemId fs-1a2b3c4d -Tag @{Key="Name";Value="My File System"}
```

- Untuk detail API, lihat [CreateTags](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-EFSFileSystem

Contoh kode berikut menunjukkan cara menggunakan Remove-EFSFileSystem.

Alat untuk PowerShell V4

Contoh 1: Menghapus sistem file tertentu yang tidak lagi digunakan (jika sistem file memiliki target mount, mereka harus dihapus terlebih dahulu). Anda diminta untuk konfirmasi sebelum cmdlet berlangsung - untuk menekan konfirmasi, gunakan sakelar. **-Force**

```
Remove-EFSFileSystem -FileSystemId fs-1a2b3c4d
```

- Untuk detail API, lihat [DeleteFileSystem](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-EFSMountTarget

Contoh kode berikut menunjukkan cara menggunakan Remove-EFSMountTarget.

Alat untuk PowerShell V4

Contoh 1: Menghapus target mount yang ditentukan. Anda diminta untuk konfirmasi sebelum operasi berlangsung. Untuk menekan prompt gunakan **-Force** sakelar. Perhatikan bahwa operasi ini secara paksa merusak setiap mount sistem file melalui target - Anda mungkin ingin mempertimbangkan untuk melepas sistem file sebelum menjalankan perintah ini, jika memungkinkan.

```
Remove-EFSMountTarget -MountTargetId fsmt-1a2b3c4d
```

- Untuk detail API, lihat [DeleteMountTarget](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-EFSTag

Contoh kode berikut menunjukkan cara menggunakan `Remove-EFSTag`.

Alat untuk PowerShell V4

Contoh 1: Menghapus koleksi satu atau lebih tag dari sistem file. Anda diminta untuk konfirmasi sebelum cmdlet berlangsung - untuk menekan konfirmasi, gunakan sakelar. **-Force**

```
Remove-EFSTag -FileSystemId fs-1a2b3c4d -TagKey "tagkey1","tagkey2"
```

- Untuk detail API, lihat [DeleteTags](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Contoh Amazon EKS menggunakan Alat untuk PowerShell V4

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan Amazon EKS.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

Add-EKSResourceTag

Contoh kode berikut menunjukkan cara menggunakan `Add-EKSResourceTag`.

Alat untuk PowerShell V4

Contoh 1: Cmdlet ini mengaitkan tag yang ditentukan ke sumber daya dengan ResourceArn yang ditentukan.

```
Add-EKSResourceTag -ResourceArn "arn:aws:eks:us-west-2:012345678912:cluster/PROD" -
Tag @{Name = "EKSPRODCLUSTER"}
```

- Untuk detail API, lihat [TagResource](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EKSCluster

Contoh kode berikut menunjukkan cara menggunakan `Get-EKSCluster`.

Alat untuk PowerShell V4

Contoh 1: Cmdlet ini menampilkan informasi deskriptif tentang kluster Amazon EKS.

```
Get-EKSCluster -Name "PROD"
```

Output:

```
Arn                : arn:aws:eks:us-west-2:012345678912:cluster/PROD
CertificateAuthority : Amazon.EKS.Model.Certificate
ClientRequestToken  :
CreatedAt           : 12/25/2019 6:46:17 AM
Endpoint            : https://669608765450FBBE54D1D78A3D71B72C.gr8.us-
west-2.eks.amazonaws.com
Identity            : Amazon.EKS.Model.Identity
Logging             : Amazon.EKS.Model.Logging
Name                : PROD
PlatformVersion     : eks.7
ResourcesVpcConfig  : Amazon.EKS.Model.VpcConfigResponse
RoleArn             : arn:aws:iam::012345678912:role/eks-iam-role
Status              : ACTIVE
Tags                : {}
Version             : 1.14
```

- Untuk detail API, lihat [DescribeCluster](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EKSClusterList

Contoh kode berikut menunjukkan cara menggunakan `Get-EKSClusterList`.

Alat untuk PowerShell V4

Contoh 1: Cmdlet ini mencantumkan kluster Amazon EKS di Region yang Akun AWS ditentukan.

```
Get-EKSClusterList
```

Output:

```
PROD
```

- Untuk detail API, lihat [ListClusters](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EKSFargateProfile

Contoh kode berikut menunjukkan cara menggunakan `Get-EKSFargateProfile`.

Alat untuk PowerShell V4

Contoh 1: Cmdlet ini mengembalikan informasi deskriptif tentang profil AWS Fargate.

```
Get-EKSFargateProfile -FargateProfileName "EKSFargate" -ClusterName "TEST"
```

Output:

```
ClusterName      : TEST
CreatedAt        : 12/26/2019 12:34:47 PM
FargateProfileArn : arn:aws:eks:us-east-2:012345678912:fargateprofile/TEST/
EKSFargate/42b7a119-e16b-a279-ce97-bdf303adec92
FargateProfileName : EKSFargate
PodExecutionRoleArn : arn:aws:iam::012345678912:role/
AmazonEKSFargatePodExecutionRole
Selectors        : {Amazon.EKS.Model.FargateProfileSelector}
Status           : ACTIVE
Subnets         : {subnet-0cd976f08d5fbfaae, subnet-02f6ff500ff2067a0}
Tags             : {}
```

- Untuk detail API, lihat [DescribeFargateProfile](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EKSFargateProfileList

Contoh kode berikut menunjukkan cara menggunakan `Get-EKSFargateProfileList`.

Alat untuk PowerShell V4

Contoh 1: Cmdlet ini mencantumkan profil AWS Fargate yang terkait dengan cluster tertentu di Region yang ditentukan. Akun AWS

```
Get-EKSFargateProfileList -ClusterName "TEST"
```

Output:

```
EKSFargate
EKSFargateProfile
```

- Untuk detail API, lihat [ListFargateProfiles](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EKSNodegroup

Contoh kode berikut menunjukkan cara menggunakan `Get-EKSNodegroup`.

Alat untuk PowerShell V4

Contoh 1: Cmdlet ini mengembalikan informasi deskriptif tentang grup node Amazon EKS.

```
Get-EKSNodegroup -NodegroupName "ProdEKSNodeGroup" -ClusterName "PROD"
```

Output:

```
AmiType       : AL2_x86_64
ClusterName   : PROD
CreatedAt     : 12/25/2019 10:16:45 AM
DiskSize      : 40
Health        : Amazon.EKS.Model.NodegroupHealth
InstanceTypes : {t3.large}
Labels        : {}
ModifiedAt    : 12/25/2019 10:16:45 AM
NodegroupArn  : arn:aws:eks:us-west-2:012345678912:nodegroup/PROD/
               ProdEKSNodeGroup/7eb79e47-82b6-04d9-e984-95110db6fa85
```

```

NodegroupName : ProdEKSNodeGroup
NodeRole       : arn:aws:iam::012345678912:role/NodeInstanceRole
ReleaseVersion : 1.14.7-20190927
RemoteAccess   :
Resources      :
ScalingConfig  : Amazon.EKS.Model.NodegroupScalingConfig
Status        : CREATING
Subnets       : {subnet-0d1a9fff35efa7691, subnet-0a3f4928edbc224d4}
Tags           : {}
Version        : 1.14

```

- Untuk detail API, lihat [DescribeNodegroup](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EKSNodegroupList

Contoh kode berikut menunjukkan cara menggunakan `Get-EKSNodegroupList`.

Alat untuk PowerShell V4

Contoh 1: Cmdlet ini mencantumkan grup node Amazon EKS yang terkait dengan cluster yang ditentukan di Region yang ditentukan. Akun AWS

```
Get-EKSNodegroupList -ClusterName PROD
```

Output:

```
ProdEKSNodeGroup
```

- Untuk detail API, lihat [ListNodegroups](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EKSResourceTag

Contoh kode berikut menunjukkan cara menggunakan `Get-EKSResourceTag`.

Alat untuk PowerShell V4

Contoh 1: Cmdlet ini mencantumkan tag untuk sumber daya Amazon EKS.

```
Get-EKSResourceTag -ResourceArn "arn:aws:eks:us-west-2:012345678912:cluster/PROD"
```

Output:

Key	Value
---	-----
Name	EKSPRODCLUSTER

- Untuk detail API, lihat [ListTagsForResource](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EKSUpdate

Contoh kode berikut menunjukkan cara menggunakan `Get-EKSUpdate`.

Alat untuk PowerShell V4

Contoh 1: Cmdlet ini menampilkan informasi deskriptif tentang pembaruan terhadap kluster Amazon EKS atau grup node terkelola terkait.

```
Get-EKSUpdate -Name "PROD" -UpdateId "ee708232-7d2e-4ed7-9270-d0b5176f0726"
```

Output:

```
CreatedAt : 12/25/2019 5:03:07 PM
Errors    : {}
Id        : ee708232-7d2e-4ed7-9270-d0b5176f0726
Params    : {Amazon.EKS.Model.UpdateParam}
Status    : Successful
Type      : LoggingUpdate
```

- Untuk detail API, lihat [DescribeUpdate](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-EKSUpdateList

Contoh kode berikut menunjukkan cara menggunakan `Get-EKSUpdateList`.

Alat untuk PowerShell V4

Contoh 1: Cmdlet ini mencantumkan pembaruan yang terkait dengan kluster Amazon EKS atau grup node terkelola di Akun AWS Wilayah yang ditentukan.

```
Get-EKSUpdateList -Name "PROD"
```

Output:

```
ee708232-7d2e-4ed7-9270-d0b5176f0726
```

- Untuk detail API, lihat [ListUpdates](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-EKSCluster

Contoh kode berikut menunjukkan cara menggunakan `New-EKSCluster`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat cluster baru yang disebut 'prod'.

```
New-EKSCluster -Name prod -ResourcesVpcConfig
@{SubnetIds=@("subnet-0a1b2c3d", "subnet-3a2b1c0d");SecurityGroupIds="sg-6979fe18"}
-RoleArn "arn:aws:iam::012345678901:role/eks-service-role"
```

Output:

```
Arn                : arn:aws:eks:us-west-2:012345678901:cluster/prod
CertificateAuthority : Amazon.EKS.Model.Certificate
ClientRequestToken  :
CreatedAt           : 12/10/2018 9:25:31 PM
Endpoint            :
Name                : prod
PlatformVersion     : eks.3
ResourcesVpcConfig  : Amazon.EKS.Model.VpcConfigResponse
RoleArn             : arn:aws:iam::012345678901:role/eks-service-role
Status              : CREATING
Version             : 1.10
```

- Untuk detail API, lihat [CreateCluster](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-EKSFargateProfile

Contoh kode berikut menunjukkan cara menggunakan `New-EKSFargateProfile`.

Alat untuk PowerShell V4

Contoh 1: Cmdlet ini membuat profil AWS Fargate untuk klaster Amazon EKS Anda. Anda harus memiliki setidaknya satu profil Fargate dalam sebuah cluster untuk dapat menjadwalkan pod pada infrastruktur Fargate.

```
New-EKSFargateProfile -FargateProfileName EKSFargateProfile -ClusterName TEST -
Subnet "subnet-02f6ff500ff2067a0", "subnet-0cd976f08d5fbfaae" -PodExecutionRoleArn
arn:aws:iam::012345678912:role/AmazonEKSFargatePodExecutionRole -Selector
@{Namespace="default"}
```

Output:

```
ClusterName      : TEST
CreatedAt        : 12/26/2019 12:38:21 PM
FargateProfileArn : arn:aws:eks:us-east-2:012345678912:fargateprofile/TEST/
EKSFargateProfile/20b7a11b-8292-41c1-bc56-ffa5e60f6224
FargateProfileName : EKSFargateProfile
PodExecutionRoleArn : arn:aws:iam::012345678912:role/
AmazonEKSFargatePodExecutionRole
Selectors        : {Amazon.EKS.Model.FargateProfileSelector}
Status           : CREATING
Subnets         : {subnet-0cd976f08d5fbfaae, subnet-02f6ff500ff2067a0}
Tags             : {}
```

- Untuk detail API, lihat [CreateFargateProfile](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-EKSNodeGroup

Contoh kode berikut menunjukkan cara menggunakan New-EKSNodeGroup.

Alat untuk PowerShell V4

Contoh 1: Cmdlet ini membuat grup node pekerja terkelola untuk klaster Amazon EKS. Anda hanya dapat membuat grup node untuk klaster Anda yang sama dengan versi Kubernetes terkini untuk klaster tersebut. Semua grup node dibuat dengan versi rilis AMI terbaru untuk masing-masing versi minor Kubernetes dari cluster.

```
New-EKSNodeGroup -NodeGroupName "ProdEKSNodeGroup" -AmiType "AL2_x86_64"
-DiskSize 40 -ClusterName "PROD" -ScalingConfig_DesiredSize 2 -
```

```
ScalingConfig_MinSize 2 -ScalingConfig_MaxSize 5 -InstanceType t3.large
-NodeRole "arn:aws:iam::012345678912:role/NodeInstanceRole" -Subnet
"subnet-0d1a9fff35efa7691", "subnet-0a3f4928edbc224d4"
```

Output:

```
AmiType      : AL2_x86_64
ClusterName  : PROD
CreatedAt    : 12/25/2019 10:16:45 AM
DiskSize     : 40
Health       : Amazon.EKS.Model.NodegroupHealth
InstanceTypes : {t3.large}
Labels       : {}
ModifiedAt   : 12/25/2019 10:16:45 AM
NodegroupArn : arn:aws:eks:us-west-2:012345678912:nodegroup/PROD/
ProdEKSNODEGROUP/7eb79e47-82b6-04d9-e984-95110db6fa85
NodegroupName : ProdEKSNODEGROUP
NodeRole      : arn:aws:iam::012345678912:role/NodeInstanceRole
ReleaseVersion : 1.14.7-20190927
RemoteAccess  :
Resources     :
ScalingConfig : Amazon.EKS.Model.NodegroupScalingConfig
Status        : CREATING
Subnets      : {subnet-0d1a9fff35efa7691, subnet-0a3f4928edbc224d4}
Tags          : {}
Version       : 1.14
```

- Untuk detail API, lihat [CreateNodegroup](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-EKSCluster

Contoh kode berikut menunjukkan cara menggunakan `Remove-EKSCluster`.

Alat untuk PowerShell V4

Contoh 1: Cmdlet ini menghapus bidang kontrol cluster Amazon EKS.

```
Remove-EKSCluster -Name "DEV-KUBE-CL"
```

Output:

```
Confirm
```

```

Are you sure you want to perform this action?
Performing the operation "Remove-EKSCluster (DeleteCluster)" on target "DEV-KUBE-CL".
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "Y"): Y

Arn                : arn:aws:eks:us-west-2:012345678912:cluster/DEV-KUBE-CL
CertificateAuthority : Amazon.EKS.Model.Certificate
ClientRequestToken  :
CreatedAt           : 12/25/2019 9:33:25 AM
Endpoint            : https://02E6D31E3E4F8C15D7BE7F58D527776A.y14.us-west-2.eks.amazonaws.com
Identity            : Amazon.EKS.Model.Identity
Logging             : Amazon.EKS.Model.Logging
Name                : DEV-KUBE-CL
PlatformVersion     : eks.7
ResourcesVpcConfig  : Amazon.EKS.Model.VpcConfigResponse
RoleArn             : arn:aws:iam::012345678912:role/eks-iam-role
Status              : DELETING
Tags                : {}
Version             : 1.14

```

- Untuk detail API, lihat [DeleteCluster](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-EKSFargateProfile

Contoh kode berikut menunjukkan cara menggunakan `Remove-EKSFargateProfile`.

Alat untuk PowerShell V4

Contoh 1: Cmdlet ini menghapus profil AWS Fargate. Saat Anda menghapus profil Fargate, semua pod yang berjalan di Fargate yang dibuat dengan profil akan dihapus.

```
Remove-EKSFargateProfile -FargateProfileName "EKSFargate" -ClusterName "TEST"
```

Output:

```

Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-EKSFargateProfile (DeleteFargateProfile)" on target "EKSFargate".

```

```
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "Y"): Y
```

```
ClusterName      : TEST
CreatedAt        : 12/26/2019 12:34:47 PM
FargateProfileArn : arn:aws:eks:us-east-2:012345678912:fargateprofile/TEST/
EKSFargate/42b7a119-e16b-a279-ce97-bdf303adec92
FargateProfileName : EKSFargate
PodExecutionRoleArn : arn:aws:iam::012345678912:role/
AmazonEKSFargatePodExecutionRole
Selectors        : {Amazon.EKS.Model.FargateProfileSelector}
Status           : DELETING
Subnets         : {subnet-0cd976f08d5fbfaae, subnet-02f6ff500ff2067a0}
Tags             : {}
```

- Untuk detail API, lihat [DeleteFargateProfile](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-EKSNodegroup

Contoh kode berikut menunjukkan cara menggunakan `Remove-EKSNodegroup`.

Alat untuk PowerShell V4

Contoh 1: Cmdlet ini menghapus grup node Amazon EKS untuk sebuah cluster.

```
Remove-EKSNodegroup -NodegroupName "ProdEKSNodeGroup" -ClusterName "PROD"
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-EKSNodegroup (DeleteNodegroup)" on target
"ProdEKSNodeGroup".
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is
"Y"): Y

AmiType          : AL2_x86_64
ClusterName      : PROD
CreatedAt        : 12/25/2019 10:16:45 AM
DiskSize         : 40
Health           : Amazon.EKS.Model.NodegroupHealth
```

```

InstanceTypes : {t3.large}
Labels        : {}
ModifiedAt    : 12/25/2019 11:01:16 AM
NodegroupArn  : arn:aws:eks:us-west-2:012345678912:nodegroup/PROD/
ProdEKSNodeGroup/7eb79e47-82b6-04d9-e984-95110db6fa85
NodegroupName : ProdEKSNodeGroup
NodeRole      : arn:aws:iam::012345678912:role/NodeInstanceRole
ReleaseVersion : 1.14.7-20190927
RemoteAccess  :
Resources     : Amazon.EKS.Model.NodegroupResources
ScalingConfig : Amazon.EKS.Model.NodegroupScalingConfig
Status        : DELETING
Subnets      : {subnet-0d1a9fff35efa7691, subnet-0a3f4928edbc224d4}
Tags          : {}
Version       : 1.14

```

- Untuk detail API, lihat [DeleteNodegroup](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-EKSResourceTag

Contoh kode berikut menunjukkan cara menggunakan `Remove-EKSResourceTag`.

Alat untuk PowerShell V4

Contoh 1: Cmdlet ini menghapus tag tertentu dari sumber daya EKS.

```

Remove-EKSResourceTag -ResourceArn "arn:aws:eks:us-west-2:012345678912:cluster/PROD"
-TagKey "Name"

```

Output:

```

Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-EKSResourceTag (UntagResource)" on target
"arn:aws:eks:us-west-2:012345678912:cluster/PROD".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"): Y

```

- Untuk detail API, lihat [UntagResource](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-EKSClusterConfig

Contoh kode berikut menunjukkan cara menggunakan `Update-EKSClusterConfig`.

Alat untuk PowerShell V4

Contoh 1: Memperbarui konfigurasi kluster Amazon EKS. Cluster Anda terus berfungsi selama pembaruan.

```
Update-EKSClusterConfig -Name "PROD" -Logging_ClusterLogging
@{Types="api","audit","authenticator","controllerManager","scheduler",Enabled="True"}
```

Output:

```
CreatedAt : 12/25/2019 5:03:07 PM
Errors    : {}
Id        : ee708232-7d2e-4ed7-9270-d0b5176f0726
Params    : {Amazon.EKS.Model.UpdateParam}
Status    : InProgress
Type      : LoggingUpdate
```

- Untuk detail API, lihat [UpdateClusterConfig](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-EKSClusterVersion

Contoh kode berikut menunjukkan cara menggunakan `Update-EKSClusterVersion`.

Alat untuk PowerShell V4

Contoh 1: Cmdlet ini memperbarui kluster Amazon EKS ke versi Kubernetes yang ditentukan. Cluster Anda terus berfungsi selama pembaruan.

```
Update-EKSClusterVersion -Name "PROD-KUBE-CL" -Version 1.14
```

Output:

```
CreatedAt : 12/26/2019 9:50:37 AM
Errors    : {}
Id        : ef186eff-3b3a-4c25-bcfc-3dcdf9e898a8
Params    : {Amazon.EKS.Model.UpdateParam, Amazon.EKS.Model.UpdateParam}
```

Status : InProgress
Type : VersionUpdate

- Untuk detail API, lihat [UpdateClusterVersion](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Elastic Load Balancing - Contoh versi 1 menggunakan Alat untuk V4 PowerShell

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan Elastic Load Balancing - Versi 1.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

Add-ELBLoadBalancerToSubnet

Contoh kode berikut menunjukkan cara menggunakan Add-ELBLoadBalancerToSubnet.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menambahkan subnet yang ditentukan ke set subnet yang dikonfigurasi untuk penyeimbang beban yang ditentukan. Outputnya mencakup daftar lengkap subnet.

```
Add-ELBLoadBalancerToSubnet -LoadBalancerName my-load-balancer -Subnet  
subnet-12345678
```

Output:

```
subnet-12345678  
subnet-87654321
```

- Untuk detail API, lihat [AttachLoadBalancerToSubnets](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Add-ELBResourceTag

Contoh kode berikut menunjukkan cara menggunakan `Add-ELBResourceTag`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menambahkan tag yang ditentukan ke penyeimbang beban yang ditentukan. Sintaks yang digunakan oleh contoh ini memerlukan PowerShell versi 3 atau yang lebih baru.

```
Add-ELBResourceTag -LoadBalancerName my-load-balancer -Tag  
@{ Key="project";Value="lima" },@{ Key="department";Value="digital-media" }
```

Contoh 2: Dengan PowerShell versi 2, Anda harus menggunakan `New-Object` untuk membuat tag untuk parameter `Tag`.

```
$tag = New-Object Amazon.ElasticLoadBalancing.Model.Tag  
$tag.Key = "project"  
$tag.Value = "lima"  
Add-ELBResourceTag -LoadBalancerName my-load-balancer -Tag $tag
```

- Untuk detail API, lihat [AddTags](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Disable-ELBAvailabilityZoneForLoadBalancer

Contoh kode berikut menunjukkan cara menggunakan `Disable-ELBAvailabilityZoneForLoadBalancer`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus Availability Zone yang ditentukan dari load balancer yang ditentukan. Outputnya mencakup Availability Zone yang tersisa.

```
Disable-ELBAvailabilityZoneForLoadBalancer -LoadBalancerName my-load-balancer -  
AvailabilityZone us-west-2a
```

Output:

```
us-west-2b
```

- Untuk detail API, lihat [DisableAvailabilityZonesForLoadBalancer](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Dismount-ELBLoadBalancerFromSubnet

Contoh kode berikut menunjukkan cara menggunakan `Dismount-ELBLoadBalancerFromSubnet`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus subnet yang ditentukan dari kumpulan subnet yang dikonfigurasi untuk penyeimbang beban yang ditentukan. Outputnya termasuk subnet yang tersisa.

```
Dismount-ELBLoadBalancerFromSubnet -LoadBalancerName my-load-balancer -Subnet  
subnet-12345678
```

Output:

```
subnet-87654321
```

- Untuk detail API, lihat [DetachLoadBalancerFromSubnets](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Edit-ELBLoadBalancerAttribute

Contoh kode berikut menunjukkan cara menggunakan `Edit-ELBLoadBalancerAttribute`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memungkinkan penyeimbangan beban lintas zona untuk penyeimbang beban yang ditentukan.

```
Edit-ELBLoadBalancerAttribute -LoadBalancerName my-load-balancer -  
CrossZoneLoadBalancing_Enabled $true
```

Contoh 2: Contoh ini menonaktifkan pengurutan koneksi untuk penyeimbang beban yang ditentukan.

```
Edit-ELBLoadBalancerAttribute -LoadBalancerName my-load-balancer -
ConnectionDraining_Enabled $false
```

Contoh 3: Contoh ini memungkinkan pencatatan akses untuk penyeimbang beban yang ditentukan.

```
Edit-ELBLoadBalancerAttribute -LoadBalancerName my-load-balancer `
>> -AccessLog_Enabled $true `
>> -AccessLog_S3BucketName amzn-s3-demo-logging-bucket `
>> -AccessLog_S3BucketPrefix my-app/prod `
>> -AccessLog_EmitInterval 60
```

- Untuk detail API, lihat [ModifyLoadBalancerAttributes](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Enable-ELBAvailabilityZoneForLoadBalancer

Contoh kode berikut menunjukkan cara menggunakan `Enable-ELBAvailabilityZoneForLoadBalancer`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menambahkan Availability Zone yang ditentukan ke load balancer yang ditentukan. Outputnya mencakup daftar lengkap Availability Zones.

```
Enable-ELBAvailabilityZoneForLoadBalancer -LoadBalancerName my-load-balancer -
AvailabilityZone us-west-2a
```

Output:

```
us-west-2a
us-west-2b
```

- Untuk detail API, lihat [EnableAvailabilityZonesForLoadBalancer](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-ELBInstanceHealth

Contoh kode berikut menunjukkan cara menggunakan `Get-ELBInstanceHealth`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan keadaan instance yang terdaftar dengan penyeimbang beban yang ditentukan.

```
Get-ELBInstanceHealth -LoadBalancerName my-load-balancer
```

Output:

Description State	InstanceId	ReasonCode
-----	-----	-----
N/A	i-87654321	N/A
InService		
Instance has failed at lea... i-12345678		Instance
OutOfService		

Contoh 2: Contoh ini menjelaskan keadaan instance tertentu yang terdaftar dengan penyeimbang beban yang ditentukan.

```
Get-ELBInstanceHealth -LoadBalancerName my-load-balancer -Instance i-12345678
```

Contoh 3: Contoh ini menampilkan deskripsi lengkap dari keadaan contoh yang ditentukan.

```
(Get-ELBInstanceHealth -LoadBalancerName my-load-balancer -Instance  
i-12345678).Description
```

Output:

```
Instance has failed at least the UnhealthyThreshold number of health checks  
consecutively.
```

- Untuk detail API, lihat [DescribeInstanceHealth](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-ELBLoadBalancer

Contoh kode berikut menunjukkan cara menggunakan `Get-ELBLoadBalancer`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan nama penyeimbang beban Anda.

```
Get-ELBLoadBalancer | format-table -property LoadBalancerName
```

Output:

```
LoadBalancerName
-----
my-load-balancer
my-other-load-balancer
my-internal-load-balancer
```

Contoh 2: Contoh ini menjelaskan penyeimbang beban yang ditentukan.

```
Get-ELBLoadBalancer -LoadBalancerName my-load-balancer
```

Output:

```
AvailabilityZones      : {us-west-2a, us-west-2b}
BackendServerDescriptions :
  {Amazon.ElasticLoadBalancing.Model.BackendServerDescription}
CanonicalHostedZoneName  : my-load-balancer-1234567890.us-west-2.elb.amazonaws.com
CanonicalHostedZoneNameID : Z3DZXE0EXAMPLE
CreatedTime              : 4/11/2015 12:12:45 PM
DNSName                   : my-load-balancer-1234567890.us-west-2.elb.amazonaws.com
HealthCheck              : Amazon.ElasticLoadBalancing.Model.HealthCheck
Instances                 : {i-207d9717, i-afefb49b}
ListenerDescriptions     : {Amazon.ElasticLoadBalancing.Model.ListenerDescription}
LoadBalancerName         : my-load-balancer
Policies                  : Amazon.ElasticLoadBalancing.Model.Policies
Scheme                   : internet-facing
SecurityGroups            : {sg-a61988c3}
SourceSecurityGroup       : Amazon.ElasticLoadBalancing.Model.SourceSecurityGroup
Subnets                  : {subnet-15aaab61}
VPCId                    : vpc-a01106c2
```

Contoh 3: Contoh ini menjelaskan semua penyeimbang beban Anda di wilayah saat ini AWS .

```
Get-ELBLoadBalancer
```

Contoh 4: Contoh ini menjelaskan semua penyeimbang beban Anda di semua yang tersedia. Wilayah AWS

```
Get-AWSRegion | % { Get-ELBLoadBalancer -Region $_ }
```

- Untuk detail API, lihat [DescribeLoadBalancers](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-ELBLoadBalancerAttribute

Contoh kode berikut menunjukkan cara menggunakan `Get-ELBLoadBalancerAttribute`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan atribut untuk penyeimbang beban yang ditentukan.

```
Get-ELBLoadBalancerAttribute -LoadBalancerName my-load-balancer
```

Output:

```
AccessLog           : Amazon.ElasticLoadBalancing.Model.AccessLog
AdditionalAttributes : {}
ConnectionDraining  : Amazon.ElasticLoadBalancing.Model.ConnectionDraining
ConnectionSettings  : Amazon.ElasticLoadBalancing.Model.ConnectionSettings
CrossZoneLoadBalancing : Amazon.ElasticLoadBalancing.Model.CrossZoneLoadBalancing
```

- Untuk detail API, lihat [DescribeLoadBalancerAttributes](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-ELBLoadBalancerPolicy

Contoh kode berikut menunjukkan cara menggunakan `Get-ELBLoadBalancerPolicy`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan kebijakan yang terkait dengan penyeimbang beban yang ditentukan.

```
Get-ELBLoadBalancerPolicy -LoadBalancerName my-load-balancer
```

Output:

PolicyAttributeDescriptions PolicyTypeName	PolicyName
-----	-----
{ProxyProtocol}	my-ProxyProtocol-policy
ProxyProtocolPolicyType	
{CookieName}	my-app-cookie-policy
AppCookieStickinessPolicyType	

Contoh 2: Contoh ini menjelaskan atribut kebijakan yang ditentukan.

```
(Get-ELBLoadBalancerPolicy -LoadBalancerName my-load-balancer -PolicyName my-ProxyProtocol-policy).PolicyAttributeDescriptions
```

Output:

AttributeName	AttributeValue
-----	-----
ProxyProtocol	true

Contoh 3: Contoh ini menjelaskan kebijakan yang telah ditentukan sebelumnya, termasuk kebijakan sampel. Nama-nama kebijakan sampel memiliki awalan ELBSample -.

```
Get-ELBLoadBalancerPolicy
```

Output:

PolicyAttributeDescriptions PolicyTypeName	PolicyName
-----	-----
{Protocol-SSLv2, Protocol-TLSv1, Pro... SSLNegotiationPolicyType	ELBSecurityPolicy-2015-05
{Protocol-SSLv2, Protocol-TLSv1, Pro... SSLNegotiationPolicyType	ELBSecurityPolicy-2015-03
{Protocol-SSLv2, Protocol-TLSv1, Pro... SSLNegotiationPolicyType	ELBSecurityPolicy-2015-02
{Protocol-SSLv2, Protocol-TLSv1, Pro... SSLNegotiationPolicyType	ELBSecurityPolicy-2014-10

```
{Protocol-SSLv2, Protocol-TLSv1, Pro... ELBSecurityPolicy-2014-01
  SSLNegotiationPolicyType
{Protocol-SSLv2, Protocol-TLSv1, Pro... ELBSecurityPolicy-2011-08
  SSLNegotiationPolicyType
{Protocol-SSLv2, Protocol-TLSv1, Pro... ELBSample-ELBDefaultCipherPolicy
  SSLNegotiationPolicyType
{Protocol-SSLv2, Protocol-TLSv1, Pro... ELBSample-OpenSSLDefaultCipherPolicy
  SSLNegotiationPolicyType
```

- Untuk detail API, lihat [DescribeLoadBalancerPolicies](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-ELBLoadBalancerPolicyType

Contoh kode berikut menunjukkan cara menggunakan `Get-ELBLoadBalancerPolicyType`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan tipe kebijakan yang didukung oleh Elastic Load Balancing.

```
Get-ELBLoadBalancerPolicyType
```

Output:

Description	PolicyAttributeTypeDescriptions
PolicyTypeName	
-----	-----

Stickiness policy with session lifet...	{CookieExpirationPeriod}
LBCookieStickinessPolicyType	
Policy that controls authentication ...	{PublicKeyPolicyName}
BackendServerAuthenticationPolicyType	
Listener policy that defines the cip...	{Protocol-SSLv2, Protocol-TLSv1, Pro...
SSLNegotiationPolicyType	
Policy containing a list of public k...	{PublicKey}
PublicKeyPolicyType	
Stickiness policy with session lifet...	{CookieName}
AppCookieStickinessPolicyType	
Policy that controls whether to incl...	{ProxyProtocol}
ProxyProtocolPolicyType	

Contoh 2: Contoh ini menjelaskan jenis kebijakan yang ditentukan.

```
Get-ELBLoadBalancerPolicyType -PolicyTypeName ProxyProtocolPolicyType
```

Output:

Description	PolicyAttributeTypeDescriptions
PolicyTypeName	
-----	-----

Policy that controls whether to incl...	{ProxyProtocol}
ProxyProtocolPolicyType	

Contoh 3: Contoh ini menampilkan deskripsi lengkap dari jenis kebijakan yang ditentukan.

```
(Get-ELBLoadBalancerPolicyType -PolicyTypeName).Description
```

Output:

```
Policy that controls whether to include the IP address and port of the originating
request for TCP messages.
This policy operates on TCP/SSL listeners only
```

- Untuk detail API, lihat [DescribeLoadBalancerPolicyTypes](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-ELBResourceTag

Contoh kode berikut menunjukkan cara menggunakan `Get-ELBResourceTag`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan tag untuk penyeimbang beban yang ditentukan.

```
Get-ELBResourceTag -LoadBalancerName @("my-load-balancer","my-internal-load-
balancer")
```

Output:

LoadBalancerName	Tags
-----	----
my-load-balancer	{project, department}

```
my-internal-load-balancer {project, department}
```

Contoh 2: Contoh ini menjelaskan tag untuk penyeimbang beban yang ditentukan.

```
(Get-ELBResourceTag -LoadBalancerName my-load-balancer).Tags
```

Output:

Key	Value
---	-----
project	lima
department	digital-media

- Untuk detail API, lihat [DescribeTags](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Join-ELBSecurityGroupToLoadBalancer

Contoh kode berikut menunjukkan cara menggunakan `Join-ELBSecurityGroupToLoadBalancer`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menggantikan grup keamanan saat ini untuk penyeimbang beban yang ditentukan dengan grup keamanan yang ditentukan.

```
Join-ELBSecurityGroupToLoadBalancer -LoadBalancerName my-load-balancer -
SecurityGroup sg-87654321
```

Output:

```
sg-87654321
```

Contoh 2: Untuk menjaga grup keamanan saat ini dan menentukan grup keamanan tambahan, tentukan grup keamanan yang ada dan yang baru.

```
Join-ELBSecurityGroupToLoadBalancer -LoadBalancerName my-load-balancer -
SecurityGroup @("sg-12345678", "sg-87654321")
```

Output:

```
sg-12345678  
sg-87654321
```

- Untuk detail API, lihat [ApplySecurityGroupsToLoadBalancer](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-ELBAppCookieStickinessPolicy

Contoh kode berikut menunjukkan cara menggunakan `New-ELBAppCookieStickinessPolicy`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat kebijakan lengket yang mengikuti masa pakai sesi lengket dari cookie yang dihasilkan aplikasi tertentu.

```
New-ELBAppCookieStickinessPolicy -LoadBalancerName my-load-balancer -PolicyName my-app-cookie-policy -CookieName my-app-cookie
```

- Untuk detail API, lihat [CreateAppCookieStickinessPolicy](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-ELBLBCookieStickinessPolicy

Contoh kode berikut menunjukkan cara menggunakan `New-ELBLBCookieStickinessPolicy`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat kebijakan lengket dengan masa pakai sesi lengket yang dikendalikan oleh periode kedaluwarsa yang ditentukan (dalam hitungan detik).

```
New-ELBLBCookieStickinessPolicy -LoadBalancerName my-load-balancer -PolicyName my-duration-cookie-policy -CookieExpirationPeriod 60
```

Contoh 2: Contoh ini membuat kebijakan lengket dengan masa pakai sesi lengket yang dikendalikan oleh masa pakai browser (agen pengguna).

```
New-ELBLBCookieStickinessPolicy -LoadBalancerName my-load-balancer -PolicyName my-duration-cookie-policy
```

- Untuk detail API, lihat [CreateLbCookieStickinessPolicy](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-ELBLoadBalancer

Contoh kode berikut menunjukkan cara menggunakan New-ELBLoadBalancer.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat penyeimbang beban dengan pendengar HTTP di VPC.

```
$httpListener = New-Object Amazon.ElasticLoadBalancing.Model.Listener
$httpListener.Protocol = "http"
$httpListener.LoadBalancerPort = 80
$httpListener.InstanceProtocol = "http"
$httpListener.InstancePort = 80
New-ELBLoadBalancer -LoadBalancerName my-vpc-load-balancer -SecurityGroup sg-
a61988c3 -Subnet subnet-15aaab61 -Listener $httpListener

my-vpc-load-balancer-1234567890.us-west-2.elb.amazonaws.com
```

Contoh 2: Contoh ini membuat penyeimbang beban dengan pendengar HTTP di EC2 -Classic.

```
New-ELBLoadBalancer -LoadBalancerName my-classic-load-balancer -AvailabilityZone us-
west-2a -Listener $httpListener
```

Output:

```
my-classic-load-balancer-123456789.us-west-2.elb.amazonaws.com
```

Contoh 3: Contoh ini membuat penyeimbang beban dengan pendengar HTTPS.

```
$httpsListener = New-Object Amazon.ElasticLoadBalancing.Model.Listener
$httpsListener.Protocol = "https"
$httpsListener.LoadBalancerPort = 443
$httpsListener.InstanceProtocol = "http"
$httpsListener.InstancePort = 80
$httpsListener.SSLCertificateId="arn:aws:iam::123456789012:server-certificate/my-
server-cert"
New-ELBLoadBalancer -LoadBalancerName my-load-balancer -AvailabilityZone us-west-2a
-Listener $httpsListener
```

```
my-load-balancer-123456789.us-west-2.elb.amazonaws.com
```

- Untuk detail API, lihat [CreateLoadBalancer](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-ELBLoadBalancerListener

Contoh kode berikut menunjukkan cara menggunakan `New-ELBLoadBalancerListener`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menambahkan pendengar HTTPS ke penyeimbang beban yang ditentukan.

```
$httpsListener = New-Object Amazon.ElasticLoadBalancing.Model.Listener
$httpsListener.Protocol = "https"
$httpsListener.LoadBalancerPort = 443
$httpsListener.InstanceProtocol = "https"
$httpsListener.InstancePort = 443
$httpsListener.SSLCertificateId="arn:aws:iam::123456789012:server-certificate/my-
server-cert"
New-ELBLoadBalancerListener -LoadBalancerName my-load-balancer -Listener
$httpsListener
```

- Untuk detail API, lihat [CreateLoadBalancerListeners](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-ELBLoadBalancerPolicy

Contoh kode berikut menunjukkan cara menggunakan `New-ELBLoadBalancerPolicy`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat kebijakan protokol proxy baru untuk penyeimbang beban tertentu.

```
$attribute = New-Object Amazon.ElasticLoadBalancing.Model.PolicyAttribute -Property
@{
    AttributeName="ProxyProtocol"
    AttributeValue="True"
}
```

```
New-ELBLoadBalancerPolicy -LoadBalancerName my-load-balancer -PolicyName my-ProxyProtocol-policy -PolicyTypeName ProxyProtocolPolicyType -PolicyAttribute $attribute
```

- Untuk detail API, lihat [CreateLoadBalancerPolicy](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Register-ELBInstanceWithLoadBalancer

Contoh kode berikut menunjukkan cara menggunakan `Register-ELBInstanceWithLoadBalancer`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendaftarkan EC2 instance tertentu dengan penyeimbang beban yang ditentukan.

```
Register-ELBInstanceWithLoadBalancer -LoadBalancerName my-load-balancer -Instance i-12345678
```

Output:

```
InstanceId
-----
i-12345678
i-87654321
```

- Untuk detail API, lihat [RegisterInstancesWithLoadBalancer](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-ELBInstanceFromLoadBalancer

Contoh kode berikut menunjukkan cara menggunakan `Remove-ELBInstanceFromLoadBalancer`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus EC2 instance tertentu dari penyeimbang beban yang ditentukan. Anda diminta untuk konfirmasi sebelum operasi berlangsung, kecuali jika Anda juga menentukan parameter `Force`.

```
Remove-ELBInstanceFromLoadBalancer -LoadBalancerName my-load-balancer -Instance
i-12345678
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-ELBInstanceFromLoadBalancer
(DeregisterInstancesFromLoadBalancer)" on Target
"Amazon.ElasticLoadBalancing.Model.Instance".
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is
"Y"):

InstanceId
-----
i-87654321
```

- Untuk detail API, lihat [DeregisterInstancesFromLoadBalancer](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-ELBLoadBalancer

Contoh kode berikut menunjukkan cara menggunakan `Remove-ELBLoadBalancer`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus penyeimbang beban yang ditentukan. Anda diminta untuk konfirmasi sebelum operasi berlangsung, kecuali jika Anda juga menentukan parameter `Force`.

```
Remove-ELBLoadBalancer -LoadBalancerName my-load-balancer
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-ELBLoadBalancer (DeleteLoadBalancer)" on Target "my-
load-balancer".
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is
"Y"):
```

- Untuk detail API, lihat [DeleteLoadBalancer](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-ELBLoadBalancerListener

Contoh kode berikut menunjukkan cara menggunakan `Remove-ELBLoadBalancerListener`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus pendengar pada port 80 untuk penyeimbang beban yang ditentukan. Anda diminta untuk konfirmasi sebelum operasi berlangsung, kecuali jika Anda juga menentukan parameter `Force`.

```
Remove-ELBLoadBalancerListener -LoadBalancerName my-load-balancer -LoadBalancerPort 80
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-ELBLoadBalancerListener (DeleteLoadBalancerListeners)"
on Target "80".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"):
```

- Untuk detail API, lihat [DeleteLoadBalancerListeners](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-ELBLoadBalancerPolicy

Contoh kode berikut menunjukkan cara menggunakan `Remove-ELBLoadBalancerPolicy`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus kebijakan yang ditentukan dari penyeimbang beban yang ditentukan. Anda diminta untuk konfirmasi sebelum operasi berlangsung, kecuali jika Anda juga menentukan parameter `Force`.

```
Remove-ELBLoadBalancerPolicy -LoadBalancerName my-load-balancer -PolicyName my-
duration-cookie-policy
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-ELBLoadBalancerPolicy (DeleteLoadBalancerPolicy)" on
Target "my-duration-cookie-policy".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"):
```

- Untuk detail API, lihat [DeleteLoadBalancerPolicy](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-ELBResourceTag

Contoh kode berikut menunjukkan cara menggunakan `Remove-ELBResourceTag`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus tag yang ditentukan dari penyeimbang beban yang ditentukan. Anda diminta untuk konfirmasi sebelum operasi berlangsung, kecuali jika Anda juga menentukan parameter `Force`. Sintaks yang digunakan oleh contoh ini memerlukan PowerShell versi 3 atau yang lebih baru.

```
Remove-ELBResourceTag -LoadBalancerName my-load-balancer -Tag @{ Key="project" }
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-ELBResourceTag (RemoveTags)" on target
"Amazon.ElasticLoadBalancing.Model.TagKeyOnly".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"):
```

Contoh 2: Dengan Powershell versi 2, Anda harus menggunakan `New-Object` untuk membuat tag untuk parameter `Tag`.

```
$tag = New-Object Amazon.ElasticLoadBalancing.Model.TagKeyOnly
$tag.Key = "project"
Remove-ELBResourceTag -Tag $tag -Force
```

- Untuk detail API, lihat [RemoveTags](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Set-ELBHealthCheck

Contoh kode berikut menunjukkan cara menggunakan `Set-ELBHealthCheck`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengonfigurasi pengaturan pemeriksaan kesehatan untuk penyeimbang beban yang ditentukan.

```
Set-ELBHealthCheck -LoadBalancerName my-load-balancer `
>> -HealthCheck_HealthyThreshold 2 `
>> -HealthCheck_UnhealthyThreshold 2 `
>> -HealthCheck_Target "HTTP:80/ping" `
>> -HealthCheck_Interval 30 `
>> -HealthCheck_Timeout 3
```

Output:

```
HealthyThreshold    : 2
Interval            : 30
Target              : HTTP:80/ping
Timeout             : 3
UnhealthyThreshold  : 2
```

- Untuk detail API, lihat [ConfigureHealthCheck](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Set-ELBLoadBalancerListenerSSLCertificate

Contoh kode berikut menunjukkan cara menggunakan `Set-ELBLoadBalancerListenerSSLCertificate`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menggantikan sertifikat yang mengakhiri koneksi SSL untuk pendengar yang ditentukan.

```
Set-ELBLoadBalancerListenerSSLCertificate -LoadBalancerName my-load-balancer `
```

```
>> -LoadBalancerPort 443 `
>> -SSLCertificateId "arn:aws:iam::123456789012:server-certificate/new-server-cert"
```

- Untuk detail API, lihat [SetLoadBalancerListenerSslCertificate](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Set-ELBLoadBalancerPolicyForBackendServer

Contoh kode berikut menunjukkan cara menggunakan `Set-ELBLoadBalancerPolicyForBackendServer`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menggantikan kebijakan untuk port yang ditentukan dengan kebijakan yang ditentukan.

```
Set-ELBLoadBalancerPolicyForBackendServer -LoadBalancerName my-load-balancer -
InstancePort 80 -PolicyName my-ProxyProtocol-policy
```

Contoh 2: Contoh ini menghapus semua kebijakan yang terkait dengan port yang ditentukan.

```
Set-ELBLoadBalancerPolicyForBackendServer -LoadBalancerName my-load-balancer -
InstancePort 80
```

- Untuk detail API, lihat [SetLoadBalancerPoliciesForBackendServer](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Set-ELBLoadBalancerPolicyOfListener

Contoh kode berikut menunjukkan cara menggunakan `Set-ELBLoadBalancerPolicyOfListener`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menggantikan kebijakan untuk listener tertentu dengan kebijakan yang ditentukan.

```
Set-ELBLoadBalancerPolicyOfListener -LoadBalancerName my-load-balancer -
LoadBalancerPort 443 -PolicyName my-SSLNegotiation-policy
```

Contoh 2: Contoh ini menghapus semua kebijakan yang terkait dengan listener tertentu.

```
Set-ELBLoadBalancerPolicyOfListener -LoadBalancerName my-load-balancer -  
LoadBalancerPort 443
```

- Untuk detail API, lihat [SetLoadBalancerPoliciesOfListener](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Elastic Load Balancing - Contoh versi 2 menggunakan Alat untuk V4 PowerShell

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan Elastic Load Balancing - Versi 2.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

Add-ELB2ListenerCertificate

Contoh kode berikut menunjukkan cara menggunakan Add-ELB2ListenerCertificate.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menambahkan sertifikat tambahan ke Listener yang ditentukan.

```
Add-ELB2ListenerCertificate -ListenerArn 'arn:aws:elasticloadbalancing:us-  
east-1:123456789012:listener/app/test-alb/3651b4394dd9a24f/3873f123b98f7618' -
```

```
Certificate @{CertificateArn = 'arn:aws:acm:us-east-1:123456789012:certificate/19478bd5-491d-47d4-b1d7-5217feba1d97'}
```

Output:

```
CertificateArn
IsDefault
-----
-----
arn:aws:acm:us-east-1:123456789012:certificate/19478bd5-491d-47d4-b1d7-5217feba1d97
False
```

- Untuk detail API, lihat [AddListenerCertificates](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Add-ELB2Tag

Contoh kode berikut menunjukkan cara menggunakan `Add-ELB2Tag`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menambahkan Tag baru ke **AWS.Tools.ElasticLoadBalancingV2** sumber daya tertentu.

```
Add-ELB2Tag -ResourceArn 'arn:aws:elasticloadbalancing:us-east-1:123456789012:loadbalancer/app/test-alb/3651b4394dd9a24f' -Tag @{Key = 'productVersion'; Value = '1.0.0'}
```

- Untuk detail API, lihat [AddTags](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Edit-ELB2Listener

Contoh kode berikut menunjukkan cara menggunakan `Edit-ELB2Listener`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengubah tindakan default listener yang ditentukan menjadi respons tetap.

```
$newDefaultAction = [Amazon.ElasticLoadBalancingV2.Model.Action]{@{
```

```

    "FixedResponseConfig" = @{
        "ContentType" = "text/plain"
        "MessageBody" = "Hello World"
        "StatusCode" = "200"
    }
    "Type" = [Amazon.ElasticLoadBalancingV2.ActionTypeEnum]::FixedResponse
}

Edit-ELB2Listener -ListenerArn 'arn:aws:elasticloadbalancing:us-east-1:123456789012:listener/app/testALB/3e2f03b558e19676/d19f2f14974db685' -Port 8080 -DefaultAction $newDefaultAction

```

Output:

```

Certificates      : {}
DefaultActions   : {Amazon.ElasticLoadBalancingV2.Model.Action}
ListenerArn      : arn:aws:elasticloadbalancing:us-east-1:123456789012:listener/app/testALB/3e2f03b558e19676/d19f2f14974db685
LoadBalancerArn  : arn:aws:elasticloadbalancing:us-east-1:123456789012:loadbalancer/app/testALB/3e2f03b558e19676
Port             : 8080
Protocol         : HTTP
SslPolicy        :

```

- Untuk detail API, lihat [ModifyListener](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Edit-ELB2LoadBalancerAttribute

Contoh kode berikut menunjukkan cara menggunakan `Edit-ELB2LoadBalancerAttribute`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memodifikasi Atribut penyeimbang beban yang ditentukan.

```

Edit-ELB2LoadBalancerAttribute -LoadBalancerArn 'arn:aws:elasticloadbalancing:us-east-1:123456789012:loadbalancer/app/test-alb/3651b4394dd9a24f' -Attribute @{Key = 'deletion_protection.enabled'; Value = 'true'}

```

Output:

Key	Value
deletion_protection.enabled	true

```

---
deletion_protection.enabled      true
access_logs.s3.enabled          false
access_logs.s3.bucket
access_logs.s3.prefix
idle_timeout.timeout_seconds    60
routing.http2.enabled           true
routing.http.drop_invalid_header_fields.enabled false

```

- Untuk detail API, lihat [ModifyLoadBalancerAttributes](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Edit-ELB2Rule

Contoh kode berikut menunjukkan cara menggunakan `Edit-ELB2Rule`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memodifikasi konfigurasi aturan Listener yang ditentukan.

```

$newRuleCondition = [Amazon.ElasticLoadBalancingV2.Model.RuleCondition]@{
    "PathPatternConfig" = @{
        "Values" = "/login1", "/login2", "/login3"
    }
    "Field" = "path-pattern"
}

Edit-ELB2Rule -RuleArn 'arn:aws:elasticloadbalancing:us-
east-1:123456789012:listener-rule/app/testALB/3e2f03b558e19676/1c84f02aec143e80/
f4f51dfaa033a8cc' -Condition $newRuleCondition

```

Output:

```

Actions      : {Amazon.ElasticLoadBalancingV2.Model.Action}
Conditions   : {Amazon.ElasticLoadBalancingV2.Model.RuleCondition}
IsDefault    : False
Priority      : 10
RuleArn      : arn:aws:elasticloadbalancing:us-east-1:123456789012:listener-rule/app/
testALB/3e2f03b558e19676/1c84f02aec143e80/f4f51dfaa033a8cc

```

- Untuk detail API, lihat [ModifyRule](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Edit-ELB2TargetGroup

Contoh kode berikut menunjukkan cara menggunakan `Edit-ELB2TargetGroup`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memodifikasi properti Grup Target yang ditentukan.

```
Edit-ELB2TargetGroup -TargetGroupArn 'arn:aws:elasticloadbalancing:us-east-1:123456789012:targetgroup/test-tg/a4e04b3688be1970' -HealthCheckIntervalSecond 60 -HealthCheckPath '/index.html' -HealthCheckPort 8080
```

Output:

```
HealthCheckEnabled      : True
HealthCheckIntervalSeconds : 60
HealthCheckPath         : /index.html
HealthCheckPort         : 8080
HealthCheckProtocol     : HTTP
HealthCheckTimeoutSeconds : 5
HealthyThresholdCount   : 5
LoadBalancerArns       : {}
Matcher                 : Amazon.ElasticLoadBalancingV2.Model.Matcher
Port                    : 80
Protocol                : HTTP
TargetGroupArn          : arn:aws:elasticloadbalancing:us-east-1:123456789012:targetgroup/test-tg/a4e04b3688be1970
TargetGroupName         : test-tg
TargetType               : instance
UnhealthyThresholdCount : 2
VpcId                    : vpc-2cfd7000
```

- Untuk detail API, lihat [ModifyTargetGroup](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Edit-ELB2TargetGroupAttribute

Contoh kode berikut menunjukkan cara menggunakan `Edit-ELB2TargetGroupAttribute`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memodifikasi atribut `deregistration_delay` dari Grup Target yang ditentukan.

```
Edit-ELB2TargetGroupAttribute -TargetGroupArn 'arn:aws:elasticloadbalancing:us-east-1:123456789012:targetgroup/test-tg/a4e04b3688be1970' -Attribute @{Key = 'deregistration_delay.timeout_seconds'; Value = 600}
```

Output:

Key	Value
---	----
stickiness.enabled	false
deregistration_delay.timeout_seconds	600
stickiness.type	lb_cookie
stickiness.lb_cookie.duration_seconds	86400
slow_start.duration_seconds	0
load_balancing.algorithm.type	round_robin

- Untuk detail API, lihat [ModifyTargetGroupAttributes](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-ELB2AccountLimit

Contoh kode berikut menunjukkan cara menggunakan `Get-ELB2AccountLimit`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini mencantumkan batas ELB2 akun untuk wilayah tertentu.

```
Get-ELB2AccountLimit
```

Output:

Max	Name
---	----
3000	target-groups
1000	targets-per-application-load-balancer
50	listeners-per-application-load-balancer
100	rules-per-application-load-balancer
50	network-load-balancers
3000	targets-per-network-load-balancer
500	targets-per-availability-zone-per-network-load-balancer
50	listeners-per-network-load-balancer
5	condition-values-per-alb-rule

```

5    condition-wildcards-per-alb-rule
100  target-groups-per-application-load-balancer
5    target-groups-per-action-on-application-load-balancer
1    target-groups-per-action-on-network-load-balancer
50   application-load-balancers

```

- Untuk detail API, lihat [DescribeAccountLimits](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-ELB2Listener

Contoh kode berikut menunjukkan cara menggunakan `Get-ELB2Listener`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan pendengar dari ALB/NLB yang ditentukan.

```

Get-ELB2Listener -LoadBalancerArn 'arn:aws:elasticloadbalancing:us-
east-1:123456789012:loadbalancer/app/test-alb/3651b4394dd9a24f'

```

Output:

```

Certificates      : {}
DefaultActions    : {Amazon.ElasticLoadBalancingV2.Model.Action}
ListenerArn       : arn:aws:elasticloadbalancing:us-east-1:123456789012:listener/app/
test-alb/3651b4394dd9a24f/1dac07c21187d41e
LoadBalancerArn   : arn:aws:elasticloadbalancing:us-east-1:123456789012:loadbalancer/
app/test-alb/3651b4394dd9a24f
Port              : 80
Protocol          : HTTP
SslPolicy         :

Certificates      : {Amazon.ElasticLoadBalancingV2.Model.Certificate}
DefaultActions    : {Amazon.ElasticLoadBalancingV2.Model.Action}
ListenerArn       : arn:aws:elasticloadbalancing:us-east-1:123456789012:listener/app/
test-alb/3651b4394dd9a24f/66e10e3aaf5b6d9b
LoadBalancerArn   : arn:aws:elasticloadbalancing:us-east-1:123456789012:loadbalancer/
app/test-alb/3651b4394dd9a24f
Port              : 443
Protocol          : HTTPS
SslPolicy         : ELBSecurityPolicy-2016-08

```

- Untuk detail API, lihat [DescribeListeners](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-ELB2ListenerCertificate

Contoh kode berikut menunjukkan cara menggunakan `Get-ELB2ListenerCertificate`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan sertifikat untuk pendengar yang ditentukan.

```
Get-ELB2ListenerCertificate -ListenerArn 'arn:aws:elasticloadbalancing:us-east-1:123456789012:listener/app/test-alb/3651b4394dd9a24f/66e10e3aaf5b6d9b'
```

Output:

```
CertificateArn
IsDefault
-----
-----
arn:aws:acm:us-east-1:123456789012:certificate/5fc7c092-68bf-4862-969c-22fd48b6e17c
True
```

- Untuk detail API, lihat [DescribeListenerCertificates](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-ELB2LoadBalancer

Contoh kode berikut menunjukkan cara menggunakan `Get-ELB2LoadBalancer`.

Alat untuk PowerShell V4

Contoh 1: Sampel ini menampilkan semua penyeimbang beban untuk wilayah tertentu.

```
Get-ELB2LoadBalancer
```

Output:

```
AvailabilityZones      : {us-east-1c}
CanonicalHostedZoneId : Z26RNL4JYFT0TI
```

```

CreatedTime      : 6/22/18 11:21:50 AM
DNSName          : test-elb1234567890-238d34ad8d94bc2e.elb.us-
east-1.amazonaws.com
IpAddressType    : ipv4
LoadBalancerArn  : arn:aws:elasticloadbalancing:us-
east-1:123456789012:loadbalancer/net/test-elb1234567890/238d34ad8d94bc2e
LoadBalancerName : test-elb1234567890
Scheme           : internet-facing
SecurityGroups   : {}
State            : Amazon.ElasticLoadBalancingV2.Model.LoadBalancerState
Type             : network
VpcId            : vpc-2cf00000

```

- Untuk detail API, lihat [DescribeLoadBalancers](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-ELB2LoadBalancerAttribute

Contoh kode berikut menunjukkan cara menggunakan `Get-ELB2LoadBalancerAttribute`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini menjelaskan atribut Load balancer yang diberikan.

```

Get-ELB2LoadBalancerAttribute -LoadBalancerArn 'arn:aws:elasticloadbalancing:us-
east-1:123456789012:loadbalancer/net/test-elb/238d34ad8d94bc2e'

```

Output:

Key	Value
---	----
access_logs.s3.enabled	false
load_balancing.cross_zone.enabled	true
access_logs.s3.prefix	
deletion_protection.enabled	false
access_logs.s3.bucket	

- Untuk detail API, lihat [DescribeLoadBalancerAttributes](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-ELB2Rule

Contoh kode berikut menunjukkan cara menggunakan `Get-ELB2Rule`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan aturan listener untuk ARN Listener yang ditentukan.

```
Get-ELB2Rule -ListenerArn 'arn:aws:elasticloadbalancing:us-east-1:123456789012:listener/app/test-alb/3651b4394dd9a24f/66e10e3aaf5b6d9b'
```

Output:

```
Actions      : {Amazon.ElasticLoadBalancingV2.Model.Action}
Conditions   : {Amazon.ElasticLoadBalancingV2.Model.RuleCondition}
IsDefault    : False
Priority      : 1
RuleArn      : arn:aws:elasticloadbalancing:us-east-1:123456789012:listener-rule/app/test-alb/3651b4394dd9a24f/66e10e3aaf5b6d9b/2286fff5055e0f79

Actions      : {Amazon.ElasticLoadBalancingV2.Model.Action}
Conditions   : {Amazon.ElasticLoadBalancingV2.Model.RuleCondition}
IsDefault    : False
Priority      : 2
RuleArn      : arn:aws:elasticloadbalancing:us-east-1:123456789012:listener-rule/app/test-alb/3651b4394dd9a24f/66e10e3aaf5b6d9b/14e7b036567623ba

Actions      : {Amazon.ElasticLoadBalancingV2.Model.Action}
Conditions   : {}
IsDefault    : True
Priority      : default
RuleArn      : arn:aws:elasticloadbalancing:us-east-1:123456789012:listener-rule/app/test-alb/3651b4394dd9a24f/66e10e3aaf5b6d9b/853948cf3aa9b2bf
```

- Untuk detail API, lihat [DescribeRules](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-ELB2SSLPolicy

Contoh kode berikut menunjukkan cara menggunakan `Get-ELB2SSLPolicy`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan semua kebijakan pendengar yang tersedia untuk ElasticLoadBalancing V2.

```
Get-ELB2SSLPolicy
```

Output:

Ciphers	Name	SslProtocols
{ECDHE-ECDSA-AES128-GCM-SHA256, ECDHE-RSA-AES128-GCM-SHA256, ECDHE-ECDSA-AES128-SHA256, ECDHE-RSA-AES128-SHA256}	ELBSecurityPolicy-2016-08	{TLSv1, TLSv1.1, TLSv1.2}
{ECDHE-ECDSA-AES128-GCM-SHA256, ECDHE-RSA-AES128-GCM-SHA256, ECDHE-ECDSA-AES128-SHA256, ECDHE-RSA-AES128-SHA256}	ELBSecurityPolicy-TLS-1-2-2017-01	{TLSv1.2}
{ECDHE-ECDSA-AES128-GCM-SHA256, ECDHE-RSA-AES128-GCM-SHA256, ECDHE-ECDSA-AES128-SHA256, ECDHE-RSA-AES128-SHA256}	ELBSecurityPolicy-TLS-1-1-2017-01	{TLSv1.1, TLSv1.2}
{ECDHE-ECDSA-AES128-GCM-SHA256, ECDHE-RSA-AES128-GCM-SHA256, ECDHE-ECDSA-AES128-SHA256, ECDHE-RSA-AES128-SHA256}	ELBSecurityPolicy-TLS-1-2-Ext-2018-06	{TLSv1.2}
{ECDHE-ECDSA-AES128-GCM-SHA256, ECDHE-RSA-AES128-GCM-SHA256, ECDHE-ECDSA-AES128-SHA256, ECDHE-RSA-AES128-SHA256}	ELBSecurityPolicy-FS-2018-06	{TLSv1, TLSv1.1, TLSv1.2}
{ECDHE-ECDSA-AES128-GCM-SHA256, ECDHE-RSA-AES128-GCM-SHA256, ECDHE-ECDSA-AES128-SHA256, ECDHE-RSA-AES128-SHA256}	ELBSecurityPolicy-2015-05	{TLSv1, TLSv1.1, TLSv1.2}
{ECDHE-ECDSA-AES128-GCM-SHA256, ECDHE-RSA-AES128-GCM-SHA256, ECDHE-ECDSA-AES128-SHA256, ECDHE-RSA-AES128-SHA256}	ELBSecurityPolicy-TLS-1-0-2015-04	{TLSv1, TLSv1.1, TLSv1.2}
{ECDHE-ECDSA-AES128-GCM-SHA256, ECDHE-RSA-AES128-GCM-SHA256, ECDHE-ECDSA-AES128-SHA256, ECDHE-RSA-AES128-SHA256}	ELBSecurityPolicy-FS-1-2-Res-2019-08	{TLSv1.2}
{ECDHE-ECDSA-AES128-GCM-SHA256, ECDHE-RSA-AES128-GCM-SHA256, ECDHE-ECDSA-AES128-SHA256, ECDHE-RSA-AES128-SHA256}	ELBSecurityPolicy-FS-1-1-2019-08	{TLSv1.1, TLSv1.2}
{ECDHE-ECDSA-AES128-GCM-SHA256, ECDHE-RSA-AES128-GCM-SHA256, ECDHE-ECDSA-AES128-SHA256, ECDHE-RSA-AES128-SHA256}	ELBSecurityPolicy-FS-1-2-2019-08	{TLSv1.2}

- Untuk detail API, lihat [DescribeSslPolicies](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-ELB2Tag

Contoh kode berikut menunjukkan cara menggunakan `Get-ELB2Tag`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan Tag untuk sumber daya yang ditentukan.

```
Get-ELB2Tag -ResourceArn 'arn:aws:elasticloadbalancing:us-east-1:123456789012:loadbalancer/app/test-alb/3651b4394dd9a24f'
```

Output:

```
ResourceArn
            Tags
-----
-----
arn:aws:elasticloadbalancing:us-east-1:123456789012:loadbalancer/app/test-alb/3651b4394dd9a24f {stage, internalName, version}
```

- Untuk detail API, lihat [DescribeTags](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-ELB2TargetGroup

Contoh kode berikut menunjukkan cara menggunakan `Get-ELB2TargetGroup`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan Grup Target yang ditentukan.

```
Get-ELB2TargetGroup -TargetGroupArn 'arn:aws:elasticloadbalancing:us-east-1:123456789012:targetgroup/test-tg/a4e04b3688be1970'
```

Output:

```
HealthCheckEnabled      : True
HealthCheckIntervalSeconds : 30
HealthCheckPath          : /
HealthCheckPort          : traffic-port
HealthCheckProtocol      : HTTP
```

```

HealthCheckTimeoutSeconds : 5
HealthyThresholdCount      : 5
LoadBalancerArns           : {arn:aws:elasticloadbalancing:us-
east-1:123456789012:loadbalancer/app/test-alb/3651b4394dd9a24f}
Matcher                    : Amazon.ElasticLoadBalancingV2.Model.Matcher
Port                        : 80
Protocol                   : HTTP
TargetGroupArn             : arn:aws:elasticloadbalancing:us-
east-1:123456789012:targetgroup/test-tg/a4e04b3688be1970
TargetGroupName            : test-tg
TargetType                 : instance
UnhealthyThresholdCount    : 2
VpcId                     : vpc-2cfd7000

```

- Untuk detail API, lihat [DescribeTargetGroups](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-ELB2TargetGroupAttribute

Contoh kode berikut menunjukkan cara menggunakan `Get-ELB2TargetGroupAttribute`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan atribut dari Grup Target yang ditentukan.

```

Get-ELB2TargetGroupAttribute -TargetGroupArn 'arn:aws:elasticloadbalancing:us-
east-1:123456789012:targetgroup/test-tg/a4e04b3688be1970'

```

Output:

Key	Value
---	----
stickiness.enabled	false
deregistration_delay.timeout_seconds	300
stickiness.type	lb_cookie
stickiness.lb_cookie.duration_seconds	86400
slow_start.duration_seconds	0
load_balancing.algorithm.type	round_robin

- Untuk detail API, lihat [DescribeTargetGroupAttributes](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-ELB2TargetHealth

Contoh kode berikut menunjukkan cara menggunakan `Get-ELB2TargetHealth`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan status kesehatan Target yang ada di Grup Target yang ditentukan.

```
Get-ELB2TargetHealth -TargetGroupArn 'arn:aws:elasticloadbalancing:us-east-1:123456789012:targetgroup/test-tg/a4e04b3688be1970'
```

Output:

HealthCheckPort	Target	TargetHealth
80	Amazon.ElasticLoadBalancingV2.Model.TargetDescription	
	Amazon.ElasticLoadBalancingV2.Model.TargetHealth	

- Untuk detail API, lihat [DescribeTargetHealth](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-ELB2Listener

Contoh kode berikut menunjukkan cara menggunakan `New-ELB2Listener`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat pendengar ALB baru dengan tindakan default 'Teruskan' untuk mengirim lalu lintas ke Grup Target tertentu.

```
$defaultAction = [Amazon.ElasticLoadBalancingV2.Model.Action]@{
    ForwardConfig = @{
        TargetGroups = @(
            @{ TargetGroupArn = "arn:aws:elasticloadbalancing:us-east-1:123456789012:targetgroup/testAlbTG/3d61c2f20aa5bccb" }
        )
        TargetGroupStickinessConfig = @{
            DurationSeconds = 900
            Enabled = $true
        }
    }
```

```

    }
    Type = "Forward"
}

New-ELB2Listener -LoadBalancerArn 'arn:aws:elasticloadbalancing:us-east-1:123456789012:loadbalancer/app/testALB/3e2f03b558e19676' -Port 8001 -Protocol "HTTP" -DefaultAction $defaultAction

```

Output:

```

Certificates      : {}
DefaultActions    : {Amazon.ElasticLoadBalancingV2.Model.Action}
ListenerArn       : arn:aws:elasticloadbalancing:us-east-1:123456789012:listener/app/testALB/3e2f03b558e19676/1c84f02aec143e80
LoadBalancerArn   : arn:aws:elasticloadbalancing:us-east-1:123456789012:loadbalancer/app/testALB/3e2f03b558e19676
Port              : 8001
Protocol          : HTTP
SslPolicy         :

```

- Untuk detail API, lihat [CreateListener](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-ELB2LoadBalancer

Contoh kode berikut menunjukkan cara menggunakan `New-ELB2LoadBalancer`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menciptakan penyeimbang beban Aplikasi yang menghadap internet baru dengan dua subnet.

```

New-ELB2LoadBalancer -Type application -Scheme internet-facing -IpAddressType ipv4 -Name 'New-Test-ALB' -SecurityGroup 'sg-07c3414abb8811cbd' -subnet 'subnet-c37a67a6', 'subnet-fc02eea0'

```

Output:

```

AvailabilityZones : {us-east-1b, us-east-1a}
CanonicalHostedZoneId : Z35SXD0TRQ7X7K
CreatedTime       : 12/28/19 2:58:03 PM
DNSName           : New-Test-ALB-1391502222.us-east-1.elb.amazonaws.com

```

```

IpAddressType      : ipv4
LoadBalancerArn    : arn:aws:elasticloadbalancing:us-
east-1:123456789012:loadbalancer/app/New-Test-ALB/dab2e4d90eb51493
LoadBalancerName   : New-Test-ALB
Scheme             : internet-facing
SecurityGroups     : {sg-07c3414abb8811cbd}
State              : Amazon.ElasticLoadBalancingV2.Model.LoadBalancerState
Type               : application
VpcId              : vpc-2cfd7000

```

- Untuk detail API, lihat [CreateLoadBalancer](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-ELB2Rule

Contoh kode berikut menunjukkan cara menggunakan `New-ELB2Rule`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat aturan Listener baru dengan tindakan respons tetap berdasarkan nilai header pelanggan untuk Listener yang ditentukan.

```

$newRuleAction = [Amazon.ElasticLoadBalancingV2.Model.Action]@{
    "FixedResponseConfig" = @{
        "ContentType" = "text/plain"
        "MessageBody" = "Hello World"
        "StatusCode" = "200"
    }
    "Type" = [Amazon.ElasticLoadBalancingV2.ActionTypeEnum]::FixedResponse
}

$newRuleCondition = [Amazon.ElasticLoadBalancingV2.Model.RuleCondition]@{
    "httpHeaderConfig" = @{
        "HttpHeaderName" = "customHeader"
        "Values" = "header2","header1"
    }
    "Field" = "http-header"
}

New-ELB2Rule -ListenerArn 'arn:aws:elasticloadbalancing:us-
east-1:123456789012:listener/app/testALB/3e2f03b558e19676/1c84f02aec143e80' -Action
$newRuleAction -Condition $newRuleCondition -Priority 10

```

Output:

```

Actions      : {Amazon.ElasticLoadBalancingV2.Model.Action}
Conditions   : {Amazon.ElasticLoadBalancingV2.Model.RuleCondition}
IsDefault    : False
Priority      : 10
RuleArn      : arn:aws:elasticloadbalancing:us-east-1:123456789012:listener-rule/app/
testALB/3e2f03b558e19676/1c84f02aec143e80/f4f51dfaa033a8cc

```

- Untuk detail API, lihat [CreateRule](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-ELB2TargetGroup

Contoh kode berikut menunjukkan cara menggunakan `New-ELB2TargetGroup`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat grup Target baru dengan parameter yang disediakan.

```

New-ELB2TargetGroup -HealthCheckEnabled 1 -HealthCheckIntervalSeconds 30 -
HealthCheckPath '/index.html' -HealthCheckPort 80 -HealthCheckTimeoutSecond 5 -
HealthyThresholdCount 2 -UnhealthyThresholdCount 5 -Port 80 -Protocol 'HTTP' -
TargetType instance -VpcId 'vpc-2cfd7000' -Name 'NewTargetGroup'

```

Output:

```

HealthCheckEnabled      : True
HealthCheckIntervalSeconds : 30
HealthCheckPath         : /index.html
HealthCheckPort         : 80
HealthCheckProtocol     : HTTP
HealthCheckTimeoutSeconds : 5
HealthyThresholdCount    : 2
LoadBalancerArns        : {}
Matcher                 : Amazon.ElasticLoadBalancingV2.Model.Matcher
Port                    : 80
Protocol                : HTTP
TargetGroupArn          : arn:aws:elasticloadbalancing:us-
east-1:123456789012:targetgroup/NewTargetGroup/534e484681d801bf
TargetGroupName         : NewTargetGroup
TargetType              : instance
UnhealthyThresholdCount : 5

```

```
VpcId : vpc-2cfd7000
```

- Untuk detail API, lihat [CreateTargetGroup](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Register-ELB2Target

Contoh kode berikut menunjukkan cara menggunakan `Register-ELB2Target`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendaftarkan instance 'i-0672a4c4c4cdeae3111' dengan kelompok target yang ditentukan.

```
Register-ELB2Target -TargetGroupArn 'arn:aws:elasticloadbalancing:us-east-1:123456789012:targetgroup/test-tg/a4e04b3688be1970' -Target @{Port = 80; Id = 'i-0672a4c4c4cdeae3111'}
```

- Untuk detail API, lihat [RegisterTargets](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-ELB2Listener

Contoh kode berikut menunjukkan cara menggunakan `Remove-ELB2Listener`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus Listener yang ditentukan.

```
Remove-ELB2Listener -ListenerArn 'arn:aws:elasticloadbalancing:us-east-1:123456789012:listener/app/test-alb/3651b4394dd9a24f/66e10e3aaf5b6d9b'
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-ELB2Listener (DeleteListener)" on target
"arn:aws:elasticloadbalancing:us-east-1:123456789012:listener/app/test-alb/3651b4394dd9a24f/66e10e3aaf5b6d9b".
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "Y"): y
```

Contoh 2: Contoh ini menghapus listener tertentu dari Load balancer.

```
Remove-ELB2Listener -ListenerArn 'arn:aws:elasticloadbalancing:us-east-1:123456789012:listener/app/test-alb/3651b4394dd9a24f/3873f123b98f7618'
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-ELB2Listener (DeleteListener)" on target
"arn:aws:elasticloadbalancing:us-east-1:123456789012:listener/app/test-alb/3651b4394dd9a24f/3873f123b98f7618".
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "Y"): y
```

- Untuk detail API, lihat [DeleteListener](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-ELB2ListenerCertificate

Contoh kode berikut menunjukkan cara menggunakan `Remove-ELB2ListenerCertificate`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus sertifikat tertentu dari kelompok Target yang ditentukan.

```
Remove-ELB2ListenerCertificate -Certificate @{CertificateArn = 'arn:aws:acm:us-east-1:123456789012:certificate/19478bd5-491d-47d4-b1d7-5217feba1d97'} -ListenerArn
'arn:aws:elasticloadbalancing:us-east-1:123456789012:listener/app/test-alb/3651b4394dd9a24f/3873f123b98f7618'
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-ELB2ListenerCertificate
(RemoveListenerCertificates)" on target "arn:aws:elasticloadbalancing:us-east-1:123456789012:listener/app/test-alb/3651b4394dd9a24f/3873f123b98f7618".
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "Y"): y
```

- Untuk detail API, lihat [RemoveListenerCertificates](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-ELB2LoadBalancer

Contoh kode berikut menunjukkan cara menggunakan `Remove-ELB2LoadBalancer`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus penyeimbang Load yang ditentukan.

```
Remove-ELB2LoadBalancer -LoadBalancerArn 'arn:aws:elasticloadbalancing:us-east-1:123456789012:loadbalancer/app/test-alb/3651b4394dd9a24f'
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-ELB2LoadBalancer (DeleteLoadBalancer)" on target
"arn:aws:elasticloadbalancing:us-east-1:123456789012:loadbalancer/app/test-
alb/3651b4394dd9a24f".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"): y
```

- Untuk detail API, lihat [DeleteLoadBalancer](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-ELB2Rule

Contoh kode berikut menunjukkan cara menggunakan `Remove-ELB2Rule`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus aturan yang ditentukan dari Listener

```
Remove-ELB2Rule -RuleArn 'arn:aws:elasticloadbalancing:us-east-1:123456789012:listener-rule/app/test-alb/3651b4394dd9a24f/3873f123b98f7618/4b25eb10a42e33ab'
```

Output:

```
Confirm
Are you sure you want to perform this action?
```

```
Performing the operation "Remove-ELB2Rule (DeleteRule)" on target
"arn:aws:elasticloadbalancing:us-east-1:123456789012:listener-rule/app/test-
alb/3651b4394dd9a24f/3873f123b98f7618/4b25eb10a42e33ab".
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is
"Y"): y
```

- Untuk detail API, lihat [DeleteRule](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-ELB2Tag

Contoh kode berikut menunjukkan cara menggunakan `Remove-ELB2Tag`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus tag untuk kunci yang ditentukan.

```
Remove-ELB2Tag -ResourceArn 'arn:aws:elasticloadbalancing:us-
east-1:123456789012:loadbalancer/app/test-alb/3651b4394dd9a24f' -TagKey
'productVersion'
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-ELB2Tag (RemoveTags)" on target
"arn:aws:elasticloadbalancing:us-east-1:123456789012:loadbalancer/app/test-
alb/3651b4394dd9a24f".
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is
"Y"): y
```

- Untuk detail API, lihat [RemoveTags](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-ELB2TargetGroup

Contoh kode berikut menunjukkan cara menggunakan `Remove-ELB2TargetGroup`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus Grup Target yang ditentukan.

```
Remove-ELB2TargetGroup -TargetGroupArn 'arn:aws:elasticloadbalancing:us-east-1:123456789012:targetgroup/testsssss/4e0b6076bc6483a7'
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-ELB2TargetGroup (DeleteTargetGroup)" on
target "arn:aws:elasticloadbalancing:us-east-1:123456789012:targetgroup/
testsssss/4e0b6076bc6483a7".
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is
"Y"): y
```

- Untuk detail API, lihat [DeleteTargetGroup](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Set-ELB2IpAddressType

Contoh kode berikut menunjukkan cara menggunakan Set-ELB2IpAddressType.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengubah jenis alamat IP Load balancer dari 'IPv4' to DualStack '.

```
Set-ELB2IpAddressType -LoadBalancerArn 'arn:aws:elasticloadbalancing:us-east-1:123456789012:loadbalancer/app/test-alb/3651b4394dd9a24f' -IpAddressType
dualstack
```

Output:

```
Value
-----
dualstack
```

- Untuk detail API, lihat [SetIpAddressType](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Set-ELB2RulePriority

Contoh kode berikut menunjukkan cara menggunakan Set-ELB2RulePriority.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengubah prioritas aturan pendengar yang ditentukan.

```
Set-ELB2RulePriority -RulePriority -RulePriority @{Priority = 11; RuleArn =  
'arn:aws:elasticloadbalancing:us-east-1:123456789012:listener-rule/app/test-  
alb/3651b4394dd9a24f/a4eb199fa5046f80/dbf4c6dcef3ec6f8'}
```

Output:

```
Actions      : {Amazon.ElasticLoadBalancingV2.Model.Action}  
Conditions   : {Amazon.ElasticLoadBalancingV2.Model.RuleCondition}  
IsDefault    : False  
Priority      : 11  
RuleArn      : arn:aws:elasticloadbalancing:us-east-1:123456789012:listener-rule/app/  
test-alb/3651b4394dd9a24f/a4eb199fa5046f80/dbf4c6dcef3ec6f8
```

- Untuk detail API, lihat [SetRulePriorities](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Set-ELB2SecurityGroup

Contoh kode berikut menunjukkan cara menggunakan `Set-ELB2SecurityGroup`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menambahkan grup keamanan 'sg-07c3414abb8811cbd' ke penyeimbang Load yang ditentukan.

```
Set-ELB2SecurityGroup -LoadBalancerArn 'arn:aws:elasticloadbalancing:us-  
east-1:123456789012:loadbalancer/app/test-alb/3651b4394dd9a24f' -SecurityGroup  
'sg-07c3414abb8811cbd'
```

Output:

```
sg-07c3414abb8811cbd
```

- Untuk detail API, lihat [SetSecurityGroups](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Set-ELB2Subnet

Contoh kode berikut menunjukkan cara menggunakan `Set-ELB2Subnet`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memodifikasi subnet dari Load balancer yang ditentukan.

```
Set-ELB2Subnet -LoadBalancerArn 'arn:aws:elasticloadbalancing:us-east-1:123456789012:loadbalancer/app/test-alb/3651b4394dd9a24f' -Subnet 'subnet-7d8a0a51', 'subnet-c37a67a6'
```

Output:

LoadBalancerAddresses	SubnetId	ZoneName
-----	-----	-----
{}	subnet-7d8a0a51	us-east-1c
{}	subnet-c37a67a6	us-east-1b

- Untuk detail API, lihat [SetSubnets](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Unregister-ELB2Target

Contoh kode berikut menunjukkan cara menggunakan `Unregister-ELB2Target`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membatalkan pendaftaran instance 'i-0672a4c4c4cdeae3111' dari grup Target yang ditentukan.

```
$targetDescription = New-Object Amazon.ElasticLoadBalancingV2.Model.TargetDescription
$targetDescription.Id = 'i-0672a4c4c4cdeae3111'
Unregister-ELB2Target -Target $targetDescription -TargetGroupArn 'arn:aws:elasticloadbalancing:us-east-1:123456789012:targetgroup/test-tg/a4e04b3688be1970'
```

- Untuk detail API, lihat [DeregisterTargets](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

FSx Contoh Amazon menggunakan Alat untuk PowerShell V4

Contoh kode berikut menunjukkan cara melakukan tindakan dan menerapkan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan Amazon FSx.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

Add-FSXResourceTag

Contoh kode berikut menunjukkan cara menggunakanAdd-FSXResourceTag.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menambahkan tag ke sumber daya yang diberikan.

```
Add-FSXResourceTag -ResourceARN "arn:aws:fsx:eu-west-1:123456789012:file-system/fs-01cd23bc4bdf5678a" -Tag @{Key="Users";Value="Test"} -PassThru
```

Output:

```
arn:aws:fsx:eu-west-1:123456789012:file-system/fs-01cd23bc4bdf5678a
```

- Untuk detail API, lihat [TagResource](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-FSXBackup

Contoh kode berikut menunjukkan cara menggunakanGet-FSXBackup.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil cadangan yang dibuat sejak kemarin untuk id sistem file yang diberikan.

```
Get-FSXBackup -Filter @{Name="file-system-id";Values=$fsx.FileSystemId} | Where-Object CreationTime -gt (Get-Date).AddDays(-1)
```

Output:

```

BackupId      : backup-01dac234e56782bcc
CreationTime   : 6/14/2019 3:35:14 AM
FailureDetails :
FileSystem     : Amazon.FSx.Model.FileSystem
KmsKeyId       : arn:aws:kms:eu-west-1:123456789012:key/f1af23c4-1b23-1bde-a1f1-
e1234c5af123
Lifecycle     : AVAILABLE
ProgressPercent : 100
ResourceARN    : arn:aws:fsx:eu-west-1:123456789012:backup/backup-01dac234e56782bcc
Tags          : {}
Type          : AUTOMATIC

```

- Untuk detail API, lihat [DescribeBackups](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-FSXFileSystem

Contoh kode berikut menunjukkan cara menggunakan `Get-FSXFileSystem`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan deskripsi `FileSystemId` yang diberikan.

```
Get-FSXFileSystem -FileSystemId fs-01cd23bc4bdf5678a
```

Output:

```

CreationTime      : 1/17/2019 9:55:30 AM
DNSName           : fs-01cd23bc4bdf5678a.ktmsad.local
FailureDetails    :
FileSystemId      : fs-01cd23bc4bdf5678a
FileSystemType    : WINDOWS
KmsKeyId          : arn:aws:kms:eu-west-1:123456789012:key/f1af23c4-5b67-8bde-
a9f0-e1234c5af678
Lifecycle        : AVAILABLE
LustreConfiguration :
NetworkInterfaceIds : {eni-07d1dda1322b7e209}
OwnerId          : 123456789012
ResourceARN       : arn:aws:fsx:eu-west-1:123456789012:file-system/
fs-01cd23bc4bdf5678a
StorageCapacity   : 300

```

```
SubnetIds      : {subnet-7d123456}
Tags           : {FSx-Service}
VpcId         : vpc-41cf2b3f
WindowsConfiguration : Amazon.FSx.Model.WindowsFileSystemConfiguration
```

- Untuk detail API, lihat [DescribeFileSystems](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-FSXResourceTagList

Contoh kode berikut menunjukkan cara menggunakan `Get-FSXResourceTagList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan tag untuk sumber daya yang disediakan arn.

```
Get-FSXResourceTagList -ResourceARN $fsx.ResourceARN
```

Output:

Key	Value
---	----
FSx-Service	Windows
Users	Dev

- Untuk detail API, lihat [ListTagsForResource](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-FSXBackup

Contoh kode berikut menunjukkan cara menggunakan `New-FSXBackup`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat cadangan dari sistem file yang diberikan.

```
New-FSXBackup -FileSystemId fs-0b1fac2345623456ba
```

Output:

```

BackupId      : backup-0b1fac2345623456ba
CreationTime   : 6/14/2019 5:37:17 PM
FailureDetails :
FileSystem     : Amazon.FSx.Model.FileSystem
KmsKeyId       : arn:aws:kms:eu-west-1:123456789012:key/f1af23c4-1b23-1bde-a1f3-
e1234c5af678
Lifecycle     : CREATING
ProgressPercent : 0
ResourceARN    : arn:aws:fsx:eu-west-1:123456789012:backup/
backup-0b1fac2345623456ba
Tags          : {}
Type          : USER_INITIATED

```

- Untuk detail API, lihat [CreateBackup](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-FSXFileSystem

Contoh kode berikut menunjukkan cara menggunakan `New-FSXFileSystem`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat sistem file Windows 300GB baru, memungkinkan akses dari subnet yang ditentukan, yang mendukung throughput hingga 8 megabyte per detik. Sistem file baru secara otomatis bergabung ke Microsoft Active Directory yang ditentukan.

```

New-FSXFileSystem -FileSystemType WINDOWS -StorageCapacity
300 -SubnetId subnet-1a2b3c4d5e6f -WindowsConfiguration
@{ThroughputCapacity=8;ActiveDirectoryId='d-1a2b3c4d'}

```

Output:

```

CreationTime      : 12/10/2018 6:06:59 PM
DNSName           : fs-abcdef01234567890.example.com
FailureDetails    :
FileSystemId      : fs-abcdef01234567890
FileSystemType    : WINDOWS
KmsKeyId          : arn:aws:kms:us-west-2:123456789012:key/a1234567-252c-45e9-
afaa-123456789abc
Lifecycle        : CREATING
LustreConfiguration :
NetworkInterfaceIds : {}

```

```

OwnerId           : 123456789012
ResourceARN       : arn:aws:fsx:us-west-2:123456789012:file-system/fs-
abcdef01234567890
StorageCapacity   : 300
SubnetIds         : {subnet-1a2b3c4d5e6f}
Tags              : {}
VpcId             : vpc-1a2b3c4d5e6f
WindowsConfiguration : Amazon.FSx.Model.WindowsFileSystemConfiguration

```

- Untuk detail API, lihat [CreateFileSystem](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-FSXFileSystemFromBackup

Contoh kode berikut menunjukkan cara menggunakan `New-FSXFileSystemFromBackup`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat sistem FSx file Amazon baru dari cadangan Amazon FSx untuk Windows File Server yang ada.

```

New-FSXFileSystemFromBackup -BackupId $backupID -Tag @{Key="tag:Name";Value="from-
manual-backup"} -SubnetId $SubnetID -SecurityGroupId $SG_ID -WindowsConfiguration
@{ThroughputCapacity=8;ActiveDirectoryId=$DirectoryID}

```

Output:

```

CreationTime       : 8/8/2019 12:59:58 PM
DNSName            : fs-012ff34e56789120.ktmsad.local
FailureDetails     :
FileSystemId       : fs-012ff34e56789120
FileSystemType     : WINDOWS
KmsKeyId           : arn:aws:kms:eu-west-1:123456789012:key/f1af23c4-5b67-1bde-
a2f3-e4567c8a9321
Lifecycle          : CREATING
LustreConfiguration :
NetworkInterfaceIds : {}
OwnerId           : 933303704102
ResourceARN       : arn:aws:fsx:eu-west-1:123456789012:file-system/
fs-012ff34e56789120
StorageCapacity   : 300
SubnetIds         : {subnet-fa1ae23c}
Tags              : {tag:Name}

```

```
VpcId           : vpc-12cf3b4f
WindowsConfiguration : Amazon.FSx.Model.WindowsFileSystemConfiguration
```

- Untuk detail API, lihat [CreateFileSystemFromBackup](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-FSXBackup

Contoh kode berikut menunjukkan cara menggunakan `Remove-FSXBackup`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus backup-id yang diberikan.

```
Remove-FSXBackup -BackupId $backupID
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-FSXBackup (DeleteBackup)" on target
"backup-0bbca1e2345678e12".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"): Y

BackupId           Lifecycle
-----
backup-0bbca1e2345678e12 DELETED
```

- Untuk detail API, lihat [DeleteBackup](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-FSXFileSystem

Contoh kode berikut menunjukkan cara menggunakan `Remove-FSXFileSystem`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus ID sistem file FSX yang diberikan.

```
Remove-FSXFileSystem -FileSystemId fs-012ff34e567890120
```

Output:

```

Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-FSXFileSystem (DeleteFileSystem)" on target
"fs-012ff34e567890120".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"): Y

FileSystemId          Lifecycle WindowsResponse
-----
fs-012ff34e567890120 DELETING  Amazon.FSx.Model.DeleteFileSystemWindowsResponse

```

- Untuk detail API, lihat [DeleteFileSystem](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-FSXResourceTag

Contoh kode berikut menunjukkan cara menggunakan `Remove-FSXResourceTag`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus tag sumber daya untuk ARN sumber daya sistem file FSX yang diberikan.

```
Remove-FSXResourceTag -ResourceARN $FSX.ResourceARN -TagKey Users
```

Output:

```

Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-FSXResourceTag (UntagResource)" on target
"arn:aws:fsx:eu-west-1:933303704102:file-system/fs-07cd45bc6bdf2674a".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"): Y

```

- Untuk detail API, lihat [UntagResource](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-FSXFileSystem

Contoh kode berikut menunjukkan cara menggunakan `Update-FSXFileSystem`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memperbarui hari penyimpanan cadangan otomatis sistem file FSX melalui `UpdateFileSystemWindowsConfiguration`.

```
$UpdateFSXWinConfig = [Amazon.FSx.Model.UpdateFileSystemWindowsConfiguration]::new()
$UpdateFSXWinConfig.AutomaticBackupRetentionDays = 35
Update-FSXFileSystem -FileSystemId $FSX.FileSystemId -WindowsConfiguration
$UpdateFSXWinConfig
```

Output:

```
CreationTime      : 1/17/2019 9:55:30 AM
DNSName           : fs-01cd23bc4bdf5678a.ktmsad.local
FailureDetails    :
FileSystemId       : fs-01cd23bc4bdf5678a
FileSystemType     : WINDOWS
KmsKeyId           : arn:aws:kms:eu-west-1:123456789012:key/f1af23c4-1b23-1bde-
a1f2-e1234c5af678
Lifecycle         : AVAILABLE
LustreConfiguration :
NetworkInterfaceIds : {eni-01cd23bc4bdf5678a}
OwnerId           : 933303704102
ResourceARN        : arn:aws:fsx:eu-west-1:933303704102:file-system/
fs-07cd45bc6bdf2674a
StorageCapacity    : 300
SubnetIds          : {subnet-1d234567}
Tags               : {FSx-Service}
VpcId              : vpc-23cf4b5f
WindowsConfiguration : Amazon.FSx.Model.WindowsFileSystemConfiguration
```

- Untuk detail API, lihat [UpdateFileSystem](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

AWS Glue contoh menggunakan Alat untuk PowerShell V4

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan AWS Glue.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

New-GLUEJob

Contoh kode berikut menunjukkan cara menggunakan New-GLUEJob.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menciptakan pekerjaan baru di AWS Glue. Nilai nama perintah selalu **glueetl**. AWS Glue mendukung menjalankan skrip pekerjaan yang ditulis dengan Python atau Scala. Dalam contoh ini, skrip pekerjaan (MyTestGlueJob.py) ditulis dengan Python. Parameter Python ditentukan dalam **\$DefArgs** variabel, dan kemudian diteruskan ke PowerShell perintah dalam **DefaultArguments** parameter, yang menerima hashtable. Parameter dalam **\$JobParams** variabel berasal dari CreateJob API, didokumentasikan dalam topik Jobs (<https://docs.aws.amazon.com/glue/latest/dg/aws-glue-api-jobs-job.html>) dari referensi AWS Glue API.

```
$Command = New-Object Amazon.Glue.Model.JobCommand
$Command.Name = 'glueetl'
$Command.ScriptLocation = 's3://amzn-s3-demo-source-bucket/admin/MyTestGlueJob.py'
$Command

$Source = "source_test_table"
$Target = "target_test_table"
$Connections = $Source, $Target

$DefArgs = @{
    '--TempDir' = 's3://amzn-s3-demo-bucket/admin'
    '--job-bookmark-option' = 'job-bookmark-disable'
    '--job-language' = 'python'
}
$DefArgs

$ExecutionProp = New-Object Amazon.Glue.Model.ExecutionProperty
$ExecutionProp.MaxConcurrentRuns = 1
$ExecutionProp
```

```
$JobParams = @{
    "AllocatedCapacity"    = "5"
    "Command"              = $Command
    "Connections_Connection" = $Connections
    "DefaultArguments"     = $DefArgs
    "Description"          = "This is a test"
    "ExecutionProperty"    = $ExecutionProp
    "MaxRetries"           = "1"
    "Name"                 = "MyOregonTestGlueJob"
    "Role"                 = "Amazon-GlueServiceRoleForSSM"
    "Timeout"              = "20"
}
```

```
New-GlueJob @JobParams
```

- Untuk detail API, lihat [CreateJob](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

AWS Health contoh menggunakan Alat untuk PowerShell V4

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan AWS Health.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

Get-HLTHEvent

Contoh kode berikut menunjukkan cara menggunakan `Get-HLTHEvent`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini mengembalikan peristiwa dari Dashboard AWS Personal Health. Pengguna menambahkan parameter `-Region` untuk melihat peristiwa yang tersedia untuk layanan di Wilayah AS Timur (Virginia N.), tetapi parameter `-Filter_Region` memfilter untuk peristiwa yang dicatat di Wilayah UE (London) dan AS Barat (Oregon) (`eu-west-2` dan `us-west-2`). `StartTime` Parameter `-Filter_` memfilter untuk rentang waktu dimana peristiwa dapat dimulai, sedangkan `EndTime` parameter `-Filter_` memfilter untuk rentang waktu dimana peristiwa dapat berakhir. Hasilnya adalah acara pemeliharaan terjadwal untuk RDS yang dimulai dalam rentang `-Filter_` yang ditentukan, dan berakhir dalam `StartTime` rentang `-Filter_` terjadwal. `EndTime`

```
Get-HLTEvent -Region us-east-1 -Filter_Region "eu-west-2","us-west-2" -
Filter_StartTime @{from="3/14/2019 6:30:00AM";to="3/15/2019 5:00:00PM"} -
Filter_EndTime @{from="3/21/2019 7:00:00AM";to="3/21/2019 5:00:00PM"}
```

Output:

```
Arn          : arn:aws:health:us-west-2::event/RDS/
AWS_RDS_HARDWARE_MAINTENANCE_SCHEDULED/
AWS_RDS_HARDWARE_MAINTENANCE_SCHEDULED_USW2_20190314_20190321
AvailabilityZone :
EndTime        : 3/21/2019 2:00:00 PM
EventTypeCategory : scheduledChange
EventTypeCode   : AWS_RDS_HARDWARE_MAINTENANCE_SCHEDULED
LastUpdatedTime : 2/28/2019 2:26:07 PM
Region         : us-west-2
Service        : RDS
StartTime      : 3/14/2019 2:00:00 PM
StatusCode     : open
```

- Untuk detail API, lihat [DescribeEvents](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Contoh IAM menggunakan Alat untuk V4 PowerShell

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan IAM.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

Add-IAMClientIDToOpenIDConnectProvider

Contoh kode berikut menunjukkan cara menggunakan `Add-IAMClientIDToOpenIDConnectProvider`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini menambahkan ID klien (atau audiens) **my-application-ID** ke penyedia OIDC yang ada bernama **server.example.com**

```
Add-IAMClientIDToOpenIDConnectProvider -ClientID "my-application-ID"
-OpenIDConnectProviderARN "arn:aws:iam::123456789012:oidc-provider/
server.example.com"
```

- Untuk detail API, lihat [AddClientIdToOpenIdConnectProvider](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Add-IAMRoleTag

Contoh kode berikut menunjukkan cara menggunakan `Add-IAMRoleTag`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menambahkan tag ke Peran dalam Layanan Manajemen Identitas

```
Add-IAMRoleTag -RoleName AdminRoleaccess -Tag @{ Key = 'abac'; Value = 'testing'}
```

- Untuk detail API, lihat [TagRole](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Add-IAMRoleToInstanceProfile

Contoh kode berikut menunjukkan cara menggunakan `Add-IAMRoleToInstanceProfile`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini menambahkan peran bernama **S3Access** ke profil instance yang ada bernama **webserver**. Untuk membuat profil instance, gunakan **New-IAMInstanceProfile** perintah. Setelah Anda membuat profil instance dan mengaitkannya dengan peran menggunakan perintah ini, Anda dapat melampirkannya ke sebuah EC2 instance. Untuk melakukan itu, gunakan **New-EC2Instance** cmdlet dengan **InstanceProfile-Name** parameter **InstanceProfile_Arn** atau untuk meluncurkan instance baru.

```
Add-IAMRoleToInstanceProfile -RoleName "S3Access" -InstanceProfileName "webserver"
```

- Untuk detail API, lihat [AddRoleToInstanceProfile](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Add-IAMUserTag

Contoh kode berikut menunjukkan cara menggunakan **Add-IAMUserTag**.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menambahkan tag ke Pengguna di Layanan Manajemen Identitas

```
Add-IAMUserTag -UserName joe -Tag @{ Key = 'abac'; Value = 'testing' }
```

- Untuk detail API, lihat [TagUser](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Add-IAMUserToGroup

Contoh kode berikut menunjukkan cara menggunakan **Add-IAMUserToGroup**.

Alat untuk PowerShell V4

Contoh 1: Perintah ini menambahkan nama pengguna **Bob** ke grup bernama **Admins**.

```
Add-IAMUserToGroup -UserName "Bob" -GroupName "Admins"
```

- Untuk detail API, lihat [AddUserToGroup](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Disable-IAMMFADevice

Contoh kode berikut menunjukkan cara menggunakan `Disable-IAMMFADevice`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini menonaktifkan perangkat MFA perangkat keras yang terkait dengan **Bob** pengguna yang memiliki nomor seri. **123456789012**

```
Disable-IAMMFADevice -UserName "Bob" -SerialNumber "123456789012"
```

Contoh 2: Perintah ini menonaktifkan perangkat MFA virtual yang terkait dengan **David** pengguna yang memiliki ARN. **arn:aws:iam::210987654321:mfa/David** Perhatikan bahwa perangkat MFA virtual tidak dihapus dari akun. Perangkat virtual masih ada dan muncul di output **Get-IAMVirtualMFADevice** perintah. Sebelum Anda dapat membuat perangkat MFA virtual baru untuk pengguna yang sama, Anda harus menghapus yang lama dengan menggunakan perintah. **Remove-IAMVirtualMFADevice**

```
Disable-IAMMFADevice -UserName "David" -SerialNumber "arn:aws:iam::210987654321:mfa/David"
```

- Untuk detail API, lihat [DeactivateMfaDevice](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Edit-IAMPassword

Contoh kode berikut menunjukkan cara menggunakan `Edit-IAMPassword`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini mengubah kata sandi untuk pengguna yang menjalankan perintah. Perintah ini hanya dapat dipanggil oleh pengguna IAM. Jika perintah ini dipanggil ketika Anda masuk dengan kredensial AWS akun (root), perintah mengembalikan kesalahan. **InvalidUserType**

```
Edit-IAMPassword -OldPassword "MyOldP@ssw0rd" -NewPassword "MyNewP@ssw0rd"
```

- Untuk detail API, lihat [ChangePassword](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Enable-IAMMFADevice

Contoh kode berikut menunjukkan cara menggunakan `Enable-IAMMFADevice`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini memungkinkan perangkat MFA perangkat keras dengan nomor seri **987654321098** dan mengaitkan perangkat dengan pengguna. **Bob** Ini termasuk dua kode pertama secara berurutan dari perangkat.

```
Enable-IAMMFADevice -UserName "Bob" -SerialNumber "987654321098" -
AuthenticationCode1 "12345678" -AuthenticationCode2 "87654321"
```

Contoh 2: Contoh ini membuat dan mengaktifkan perangkat MFA virtual. Perintah pertama membuat perangkat virtual dan mengembalikan representasi objek perangkat dalam variabel `$MFADevice`. Anda dapat menggunakan `QRCodePng` properti `.Base32StringSeed` atau untuk mengkonfigurasi aplikasi perangkat lunak pengguna. Perintah terakhir menetapkan perangkat kepada pengguna **David**, mengidentifikasi perangkat dengan nomor serinya. Perintah ini juga menyinkronkan perangkat AWS dengan memasukkan dua kode pertama secara berurutan dari perangkat MFA virtual.

```
$MFADevice = New-IAMVirtualMFADevice -VirtualMFADeviceName "MyMFADevice"
# see example for New-IAMVirtualMFADevice to see how to configure the software
program with PNG or base32 seed code
Enable-IAMMFADevice -UserName "David" -SerialNumber $MFADevice.SerialNumber
-AuthenticationCode1 "24681357" -AuthenticationCode2
"13572468"
```

- Untuk detail API, lihat [EnableMfaDevice](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-IAMAccessKey

Contoh kode berikut menunjukkan cara menggunakan `Get-IAMAccessKey`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini mencantumkan kunci akses untuk pengguna IAM bernama **Bob**. Perhatikan bahwa Anda tidak dapat mencantumkan kunci akses rahasia untuk pengguna IAM. Jika kunci akses rahasia hilang, Anda harus membuat kunci akses baru dengan **New-IAMAccessKey** cmdlet.

```
Get-IAMAccessKey -UserName "Bob"
```

Output:

AccessKeyId	CreateDate	Status	UserName
-----	-----	-----	-----
AKIAIOSFODNN7EXAMPLE	12/3/2014 10:53:41 AM	Active	Bob
AKIAI44QH8DHBEXAMPLE	6/6/2013 8:42:26 PM	Inactive	Bob

- Untuk detail API, lihat [ListAccessKeys](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-IAMAccessKeyLastUsed

Contoh kode berikut menunjukkan cara menggunakan `Get-IAMAccessKeyLastUsed`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan nama pengguna yang dimiliki dan informasi penggunaan terakhir untuk kunci akses yang disediakan.

```
Get-IAMAccessKeyLastUsed -AccessKeyId ABCDEXAMPLE
```

- Untuk detail API, lihat [GetAccessKeyLastUsed](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-IAMAccountAlias

Contoh kode berikut menunjukkan cara menggunakan `Get-IAMAccountAlias`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini mengembalikan alias akun untuk Akun AWS

```
Get-IAMAccountAlias
```

Output:

```
ExampleCo
```

- Untuk detail API, lihat [ListAccountAliases](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-IAMAccountAuthorizationDetail

Contoh kode berikut menunjukkan cara menggunakan `Get-IAMAccountAuthorizationDetail`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan rincian otorisasi tentang identitas di AWS akun, dan menampilkan daftar elemen objek yang dikembalikan, termasuk pengguna, grup, dan peran. Misalnya, **UserDetailList** properti menampilkan detail tentang pengguna. Informasi serupa tersedia di **RoleDetailList** dan **GroupDetailList** properti.

```
$Details=Get-IAMAccountAuthorizationDetail
$Details
```

Output:

```
GroupDetailList : {Administrators, Developers, Testers, Backup}
IsTruncated     : False
Marker          :
RoleDetailList  : {TestRole1, AdminRole, TesterRole, clirole...}
UserDetailList  : {Administrator, Bob, BackupToS3, }
```

```
$Details.UserDetailList
```

Output:

```
Arn          : arn:aws:iam::123456789012:user/Administrator
CreateDate   : 10/16/2014 9:03:09 AM
GroupList    : {Administrators}
Path         : /
UserId       : AIDACKCEVSQ6CEXAMPLE1
UserName     : Administrator
UserPolicyList : {}

Arn          : arn:aws:iam::123456789012:user/Bob
CreateDate   : 4/6/2015 12:54:42 PM
GroupList    : {Developers}
Path         : /
UserId       : AIDACKCEVSQ6CEXAMPLE2
```

```

UserName      : bab
UserPolicyList : {}

Arn           : arn:aws:iam::123456789012:user/BackupToS3
CreateDate    : 1/27/2015 10:15:08 AM
GroupList     : {Backup}
Path          : /
UserId        : AIDACKCEVSQ6CEXAMPLE3
UserName      : BackupToS3
UserPolicyList : {BackupServicePermissionsToS3Buckets}

```

- Untuk detail API, lihat [GetAccountAuthorizationDetails](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-IAMAccountPasswordPolicy

Contoh kode berikut menunjukkan cara menggunakan `Get-IAMAccountPasswordPolicy`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan detail tentang kebijakan kata sandi untuk akun saat ini. Jika tidak ada kebijakan kata sandi yang ditentukan untuk akun, perintah mengembalikan **NoSuchEntity** kesalahan.

```
Get-IAMAccountPasswordPolicy
```

Output:

```

AllowUsersToChangePassword : True
ExpirePasswords             : True
HardExpiry                  : False
MaxPasswordAge              : 90
MinimumPasswordLength       : 8
PasswordReusePrevention     : 20
RequireLowercaseCharacters  : True
RequireNumbers               : True
RequireSymbols               : False
RequireUppercaseCharacters  : True

```

- Untuk detail API, lihat [GetAccountPasswordPolicy](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-IAccountSummary

Contoh kode berikut menunjukkan cara menggunakan `Get-IAccountSummary`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan informasi tentang penggunaan entitas IAM saat ini dan kuota entitas IAM saat ini di Akun AWS

```
Get-IAccountSummary
```

Output:

Key	Value
Users	7
GroupPolicySizeQuota	5120
PolicyVersionsInUseQuota	10000
ServerCertificatesQuota	20
AccountSigningCertificatesPresent	0
AccountAccessKeysPresent	0
Groups	3
UsersQuota	5000
RolePolicySizeQuota	10240
UserPolicySizeQuota	2048
GroupsPerUserQuota	10
AssumeRolePolicySizeQuota	2048
AttachedPoliciesPerGroupQuota	2
Roles	9
VersionsPerPolicyQuota	5
GroupsQuota	100
PolicySizeQuota	5120
Policies	5
RolesQuota	250
ServerCertificates	0
AttachedPoliciesPerRoleQuota	2
MFADevicesInUse	2
PoliciesQuota	1000
AccountMFAEnabled	1
Providers	2
InstanceProfilesQuota	100
MFADevices	4
AccessKeysPerUserQuota	2

AttachedPoliciesPerUserQuota	2
SigningCertificatesPerUserQuota	2
PolicyVersionsInUse	4
InstanceProfiles	1
...	

- Untuk detail API, lihat [GetAccountSummary](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-IAMAttachedGroupPolicyList

Contoh kode berikut menunjukkan cara menggunakan `Get-IAMAttachedGroupPolicyList`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini mengembalikan nama dan kebijakan ARNs terkelola yang dilampirkan ke grup IAM yang disebutkan **Admins** di AWS akun. Untuk melihat daftar kebijakan sebaris yang disematkan dalam grup, gunakan **Get-IAMGroupPolicyList** perintah.

```
Get-IAMAttachedGroupPolicyList -GroupName "Admins"
```

Output:

PolicyArn	PolicyName
-----	-----
arn:aws:iam::aws:policy/SecurityAudit	SecurityAudit
arn:aws:iam::aws:policy/AdministratorAccess	AdministratorAccess

- Untuk detail API, lihat [ListAttachedGroupPolicies](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-IAMAttachedRolePolicyList

Contoh kode berikut menunjukkan cara menggunakan `Get-IAMAttachedRolePolicyList`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini mengembalikan nama dan kebijakan ARNs terkelola yang dilampirkan pada peran IAM yang disebutkan **SecurityAuditRole** di AWS akun. Untuk melihat daftar kebijakan sebaris yang disematkan dalam peran, gunakan **Get-IAMRolePolicyList** perintah.

```
Get-IAMAttachedRolePolicyList -RoleName "SecurityAuditRole"
```

Output:

PolicyArn	PolicyName
-----	-----
arn:aws:iam::aws:policy/SecurityAudit	SecurityAudit

- Untuk detail API, lihat [ListAttachedRolePolicies](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-IAMAttachedUserPolicyList

Contoh kode berikut menunjukkan cara menggunakan `Get-IAMAttachedUserPolicyList`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini mengembalikan nama dan kebijakan ARNs terkelola untuk pengguna IAM yang disebutkan **Bob** di AWS akun. Untuk melihat daftar kebijakan inline yang disematkan di pengguna IAM, gunakan perintah. **Get-IAMUserPolicyList**

```
Get-IAMAttachedUserPolicyList -UserName "Bob"
```

Output:

PolicyArn	PolicyName
-----	-----
arn:aws:iam::aws:policy/TesterPolicy	TesterPolicy

- Untuk detail API, lihat [ListAttachedUserPolicies](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-IAMContextKeysForCustomPolicy

Contoh kode berikut menunjukkan cara menggunakan `Get-IAMContextKeysForCustomPolicy`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil semua kunci konteks yang ada dalam kebijakan yang disediakan JSON. Untuk memberikan beberapa kebijakan yang dapat Anda berikan sebagai daftar nilai yang dipisahkan koma.

```
$policy1 = '{"Version":"2012-10-17","Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/","Condition":{"DateGreaterThan":
{"aws:CurrentTime":"2015-08-16T12:00:00Z"}}}}'
$policy2 = '{"Version":"2012-10-17","Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/"},"}'
Get-IAMContextKeysForCustomPolicy -PolicyInputList $policy1,$policy2
```

- Untuk detail API, lihat [GetContextKeysForCustomPolicy](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-IAMContextKeysForPrincipalPolicy

Contoh kode berikut menunjukkan cara menggunakan `Get-IAMContextKeysForPrincipalPolicy`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil semua kunci konteks yang ada dalam json kebijakan yang disediakan dan kebijakan yang dilampirkan ke entitas IAM (pengguna/peran, dll.). Untuk `-PolicyInputList` Anda dapat memberikan beberapa daftar nilai sebagai nilai yang dipisahkan koma.

```
$policy1 = '{"Version":"2012-10-17","Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/","Condition":{"DateGreaterThan":
{"aws:CurrentTime":"2015-08-16T12:00:00Z"}}}}'
$policy2 = '{"Version":"2012-10-17","Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/"},"}'
Get-IAMContextKeysForPrincipalPolicy -PolicyInputList $policy1,$policy2 -
PolicySourceArn arn:aws:iam::852640994763:user/TestUser
```

- Untuk detail API, lihat [GetContextKeysForPrincipalPolicy](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-IAMCredentialReport

Contoh kode berikut menunjukkan cara menggunakan `Get-IAMCredentialReport`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuka laporan yang dikembalikan dan mengeluarkannya ke pipeline sebagai array baris teks. Baris pertama adalah header dengan nama kolom yang dipisahkan koma. Setiap baris berturut-turut adalah baris detail untuk satu pengguna, dengan setiap bidang dipisahkan dengan koma. Sebelum Anda dapat melihat laporan, Anda harus membuatnya dengan **Request-IAMCredentialReport** cmdlet. Untuk mengambil laporan sebagai string tunggal, gunakan **-Raw** sebagai pengganti. **-AsTextArray** Alias juga **-SplitLines** diterima untuk **-AsTextArray** sakelar. Untuk daftar lengkap kolom dalam output, lihat referensi API layanan. Perhatikan bahwa jika Anda tidak menggunakan **-AsTextArray** atau **-SplitLines**, maka Anda harus mengekstrak teks dari **.Content** properti menggunakan **StreamReader** kelas.NET.

```
Request-IAMCredentialReport
```

Output:

Description	State
-----	----
No report exists. Starting a new report generation task	STARTED

```
Get-IAMCredentialReport -AsTextArray
```

Output:

```
user,arn,user_creation_time,password_enabled,password_last_used,password_last_changed,password
root_account,arn:aws:iam::123456789012:root,2014-10-15T16:31:25+00:00,not_supported,2015-04-20T16:06:00+00:00,A,false,N/A,false,N/A,false,N/A
Administrator,arn:aws:iam::123456789012:user/Administrator,2014-10-16T16:03:09+00:00,true,2015-04-20T15:18:32+00:00,2014-10-16T16:06:00+00:00,A,false,true,2014-12-03T18:53:41+00:00,true,2015-03-25T20:38:14+00:00,false,N/A,false,N/A
Bill,arn:aws:iam::123456789012:user/Bill,2015-04-15T18:27:44+00:00,false,N/A,N/A,N/A,A,false,false,N/A,false,N/A,false,2015-04-20T20:00:12+00:00,false,N/A
```

- Untuk detail API, lihat [GetCredentialReport](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-IAMEntitiesForPolicy

Contoh kode berikut menunjukkan cara menggunakan `Get-IAMEntitiesForPolicy`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menampilkan daftar grup, peran, dan pengguna IAM yang memiliki kebijakan yang **arn:aws:iam::123456789012:policy/TestPolicy** dilampirkan.

```
Get-IAMEntitiesForPolicy -PolicyArn "arn:aws:iam::123456789012:policy/TestPolicy"
```

Output:

```
IsTruncated   : False
Marker        :
PolicyGroups  : {}
PolicyRoles   : {testRole}
PolicyUsers   : {Bob, Theresa}
```

- Untuk detail API, lihat [ListEntitiesForPolicy](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-IAMGroup

Contoh kode berikut menunjukkan cara menggunakan `Get-IAMGroup`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan detail tentang grup **IAMTesters**, termasuk kumpulan semua pengguna IAM yang termasuk dalam grup.

```
$results = Get-IAMGroup -GroupName "Testers"
$results
```

Output:

Group Users	IsTruncated	Marker
-----	-----	-----
Amazon.IdentityManagement.Model.Group {Theresa, David}	False	

```
$results.Group
```

Output:

```

Arn      : arn:aws:iam::123456789012:group/Testers
CreateDate : 12/10/2014 3:39:11 PM
GroupId   : 3RHNZZGQJ7QHMAEXAMPLE1
GroupName : Testers
Path      : /

```

```
$results.Users
```

Output:

```

Arn      : arn:aws:iam::123456789012:user/Theresa
CreateDate : 12/10/2014 3:39:27 PM
PasswordLastUsed : 1/1/0001 12:00:00 AM
Path      : /
UserId    : 40SVDDJJTF4XEEXAMPLE2
UserName  : Theresa

Arn      : arn:aws:iam::123456789012:user/David
CreateDate : 12/10/2014 3:39:27 PM
PasswordLastUsed : 3/19/2015 8:44:04 AM
Path      : /
UserId    : Y4FKWQCXTA52QEXAMPLE3
UserName  : David

```

- Untuk detail API, lihat [GetGroup](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-IAMGroupForUser

Contoh kode berikut menunjukkan cara menggunakan `Get-IAMGroupForUser`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan daftar grup IAM yang **David** dimiliki pengguna IAM.

```
Get-IAMGroupForUser -UserName David
```

Output:

```
Arn      : arn:aws:iam::123456789012:group/Administrators
CreateDate : 10/20/2014 10:06:24 AM
GroupId   : 6WCH4TRY3KIHIEEXAMPLE1
GroupName : Administrators
Path      : /

Arn      : arn:aws:iam::123456789012:group/Testers
CreateDate : 12/10/2014 3:39:11 PM
GroupId   : RHNZZGQJ7QHMAEXAMPLE2
GroupName : Testers
Path      : /

Arn      : arn:aws:iam::123456789012:group/Developers
CreateDate : 12/10/2014 3:38:55 PM
GroupId   : ZU2E0WMK6WBZ0EXAMPLE3
GroupName : Developers
Path      : /
```

- Untuk detail API, lihat [ListGroupsForUser](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-IAMGroupList

Contoh kode berikut menunjukkan cara menggunakan `Get-IAMGroupList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan koleksi semua grup IAM didefinisikan dalam saat ini Akun AWS.

```
Get-IAMGroupList
```

Output:

```

Arn      : arn:aws:iam::123456789012:group/Administrators
CreateDate : 10/20/2014 10:06:24 AM
GroupId   : 6WCH4TRY3KIHEXAMPLE1
GroupName : Administrators
Path      : /

Arn      : arn:aws:iam::123456789012:group/Developers
CreateDate : 12/10/2014 3:38:55 PM
GroupId   : ZU2E0WMK6WBZ0EXAMPLE2
GroupName : Developers
Path      : /

Arn      : arn:aws:iam::123456789012:group/Testers
CreateDate : 12/10/2014 3:39:11 PM
GroupId   : RHNZZGQJ7QHMAEXAMPLE3
GroupName : Testers
Path      : /

```

- Untuk detail API, lihat [ListGroups](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-IAMGroupPolicy

Contoh kode berikut menunjukkan cara menggunakan `Get-IAMGroupPolicy`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan rincian tentang kebijakan inline tertanam yang dinamai **PowerUserAccess-Testers** untuk grup **Testers**. **PolicyDocument** Properti ini dikodekan URL. Ini diterjemahkan dalam contoh ini dengan **UrlDecode** metode.NET.

```

$results = Get-IAMGroupPolicy -GroupName Testers -PolicyName PowerUserAccess-Testers
$results

```

Output:

```

GroupName      PolicyDocument                                     PolicyName
-----
Testers        %7B%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C%0A%20...
PowerUserAccess-Testers

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")

```

```
[System.Web.HttpUtility]::UrlDecode($results.PolicyDocument)
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "NotAction": "iam:*",
      "Resource": "*"
    }
  ]
}
```

- Untuk detail API, lihat [GetGroupPolicy](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-IAMGroupPolicyList

Contoh kode berikut menunjukkan cara menggunakan `Get-IAMGroupPolicyList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan daftar kebijakan inline yang disematkan dalam grup **Testers**. Untuk mendapatkan kebijakan terkelola yang dilampirkan ke grup, gunakan perintah **Get-IAMAttachedGroupPolicyList**.

```
Get-IAMGroupPolicyList -GroupName Testers
```

Output:

```
Deny-Assume-S3-Role-In-Production
PowerUserAccess-Testers
```

- Untuk detail API, lihat [ListGroupPolicies](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-IAMInstanceProfile

Contoh kode berikut menunjukkan cara menggunakan `Get-IAMInstanceProfile`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan rincian profil contoh bernama **ec2instanceroles** yang didefinisikan dalam AWS akun saat ini.

```
Get-IAMInstanceProfile -InstanceProfileName ec2instancerole
```

Output:

```
Arn           : arn:aws:iam::123456789012:instance-profile/ec2instancerole
CreateDate    : 2/17/2015 2:49:04 PM
InstanceId    : HH36PTZQJUR32EXAMPLE1
InstanceProfileName : ec2instancerole
Path          : /
Roles         : {ec2instancerole}
```

- Untuk detail API, lihat [GetInstanceProfile](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-IAMInstanceProfileForRole

Contoh kode berikut menunjukkan cara menggunakan `Get-IAMInstanceProfileForRole`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan rincian profil instance yang terkait dengan peran **ec2instancerole**.

```
Get-IAMInstanceProfileForRole -RoleName ec2instancerole
```

Output:

```
Arn           : arn:aws:iam::123456789012:instance-profile/
ec2instancerole
CreateDate    : 2/17/2015 2:49:04 PM
InstanceId    : HH36PTZQJUR32EXAMPLE1
InstanceProfileName : ec2instancerole
Path          : /
Roles         : {ec2instancerole}
```

- Untuk detail API, lihat [ListInstanceProfilesForRole](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-IAMInstanceProfileList

Contoh kode berikut menunjukkan cara menggunakan `Get-IAMInstanceProfileList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan koleksi profil instance didefinisikan dalam saat ini Akun AWS.

```
Get-IAMInstanceProfileList
```

Output:

```
Arn                : arn:aws:iam::123456789012:instance-profile/ec2instancerole
CreateDate         : 2/17/2015 2:49:04 PM
InstanceProfileId  : HH36PTZQJUR32EXAMPLE1
InstanceProfileName : ec2instancerole
Path               : /
Roles              : {ec2instancerole}
```

- Untuk detail API, lihat [ListInstanceProfiles](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-IAMLoginProfile

Contoh kode berikut menunjukkan cara menggunakan `Get-IAMLoginProfile`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan tanggal pembuatan kata sandi dan apakah reset kata sandi diperlukan untuk pengguna **David** IAM.

```
Get-IAMLoginProfile -UserName David
```

Output:

CreateDate	PasswordResetRequired	UserName
-----	-----	-----
12/10/2014 3:39:44 PM	False	David

- Untuk detail API, lihat [GetLoginProfile](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-IAMMFADevice

Contoh kode berikut menunjukkan cara menggunakan `Get-IAMMFADevice`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan rincian tentang perangkat MFA yang ditetapkan untuk pengguna IAM. **David** Dalam contoh ini Anda dapat mengatakan bahwa itu adalah perangkat virtual karena **SerialNumber** adalah ARN bukan nomor seri aktual perangkat fisik.

```
Get-IAMMFADevice -UserName David
```

Output:

EnableDate	SerialNumber	UserName
-----	-----	-----
4/8/2015 9:41:10 AM	arn:aws:iam::123456789012:mfa/David	David

- Untuk detail API, lihat [ListMfaDevices](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-IAMOpenIDConnectProvider

Contoh kode berikut menunjukkan cara menggunakan `Get-IAMOpenIDConnectProvider`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan rincian tentang penyedia OpenID Connect yang ARN adalah. **arn:aws:iam::123456789012:oidc-provider/accounts.google.com** **ClientIDList** Properti adalah koleksi yang berisi semua Klien yang IDs ditentukan untuk penyedia ini.

```
Get-IAMOpenIDConnectProvider -OpenIDConnectProviderArn
arn:aws:iam::123456789012:oidc-provider/oidc.example.com
```

Output:

ClientIDList Url	CreateDate	ThumbprintList
-----	-----	-----

{MyOIDCApp}	2/3/2015 3:00:30 PM	
{12345abcdefghijk67890lmnopqrst98765uvwxy}		oidc.example.com

- Untuk detail API, lihat [GetOpenIdConnectProvider](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-IAMOpenIDConnectProviderList

Contoh kode berikut menunjukkan cara menggunakan `Get-IAMOpenIDConnectProviderList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan daftar ARNS dari semua penyedia OpenID Connect yang didefinisikan dalam Akun AWS saat ini.

```
Get-IAMOpenIDConnectProviderList
```

Output:

```
Arn
---
arn:aws:iam::123456789012:oidc-provider/server.example.com
arn:aws:iam::123456789012:oidc-provider/another.provider.com
```

- Untuk detail API, lihat [ListOpenIdConnectProviders](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-IAMPolicy

Contoh kode berikut menunjukkan cara menggunakan `Get-IAMPolicy`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan rincian tentang kebijakan terkelola yang ARN.

arn:aws:iam::123456789012:policy/MySamplePolicy

```
Get-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
```

Output:

```
Arn           : arn:aws:iam::aws:policy/MySamplePolicy
AttachmentCount : 0
CreateDate    : 2/6/2015 10:40:08 AM
```

```

DefaultVersionId : v1
Description      :
IsAttachable    : True
Path            : /
PolicyId        : Z27SI6FQMGNQ2EXAMPLE1
PolicyName      : MySamplePolicy
UpdateDate      : 2/6/2015 10:40:08 AM

```

- Untuk detail API, lihat [GetPolicy](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-IAMPolicyList

Contoh kode berikut menunjukkan cara menggunakan `Get-IAMPolicyList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan kumpulan dari tiga kebijakan terkelola pertama yang tersedia di AWS akun saat ini. Karena tidak **-scope** ditentukan, defaultnya **all** dan mencakup kebijakan terkelola dan yang AWS dikelola pelanggan.

```
Get-IAMPolicyList -MaxItem 3
```

Output:

```

Arn          : arn:aws:iam::aws:policy/AWSDirectConnectReadOnlyAccess
AttachmentCount : 0
CreateDate    : 2/6/2015 10:40:08 AM
DefaultVersionId : v1
Description   :
IsAttachable  : True
Path         : /
PolicyId      : Z27SI6FQMGNQ2EXAMPLE1
PolicyName    : AWSDirectConnectReadOnlyAccess
UpdateDate    : 2/6/2015 10:40:08 AM

Arn          : arn:aws:iam::aws:policy/AmazonGlacierReadOnlyAccess
AttachmentCount : 0
CreateDate    : 2/6/2015 10:40:27 AM
DefaultVersionId : v1
Description   :
IsAttachable  : True
Path         : /

```

```

PolicyId      : NJKMU274MET4EEXAMPLE2
PolicyName    : AmazonGlacierReadOnlyAccess
UpdateDate   : 2/6/2015 10:40:27 AM

Arn          : arn:aws:iam::aws:policy/AWSMarketplaceFullAccess
AttachmentCount : 0
CreateDate   : 2/11/2015 9:21:45 AM
DefaultVersionId : v1
Description   :
IsAttachable  : True
Path         : /
PolicyId     : 5ULJS02FYVPYGEXAMPLE3
PolicyName    : AWSMarketplaceFullAccess
UpdateDate   : 2/11/2015 9:21:45 AM

```

Contoh 2: Contoh ini mengembalikan kumpulan dari dua kebijakan terkelola pelanggan pertama yang tersedia di AWS akun saat ini. Ini digunakan **-Scope local** untuk membatasi output hanya pada kebijakan yang dikelola pelanggan.

```
Get-IAMPolicyList -Scope local -MaxItem 2
```

Output:

```

Arn          : arn:aws:iam::123456789012:policy/MyLocalPolicy
AttachmentCount : 0
CreateDate   : 2/12/2015 9:39:09 AM
DefaultVersionId : v2
Description   :
IsAttachable  : True
Path         : /
PolicyId     : SQVCBLC4VA0UCEEXAMPLE4
PolicyName    : MyLocalPolicy
UpdateDate   : 2/12/2015 9:39:53 AM

Arn          : arn:aws:iam::123456789012:policy/policyforec2instancerole
AttachmentCount : 1
CreateDate   : 2/17/2015 2:51:38 PM
DefaultVersionId : v11
Description   :
IsAttachable  : True
Path         : /
PolicyId     : X5JPBLJH2Z2S0EXAMPLE5
PolicyName    : policyforec2instancerole

```

UpdateDate : 2/18/2015 8:52:31 AM

- Untuk detail API, lihat [ListPolicies](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-IAMPolicyVersion

Contoh kode berikut menunjukkan cara menggunakan `Get-IAMPolicyVersion`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan dokumen kebijakan untuk **v2** versi kebijakan yang ARN-nya. **arn:aws:iam::123456789012:policy/MyManagedPolicy** Dokumen kebijakan dalam **Document** properti adalah URL yang dikodekan dan diterjemahkan dalam contoh ini dengan metode.NET. **UrlDecode**

```
$results = Get-IAMPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/
MyManagedPolicy -VersionId v2
$results
```

Output:

```
CreateDate          Document
-----
IsDefaultVersion    VersionId
-----
2/12/2015 9:39:53 AM %7B%0A%20%20%22Version%22%3A%20%222012-10...    True
v2

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
$policy = [System.Web.HttpUtility]::UrlDecode($results.Document)
$policy
{
  "Version": "2012-10-17",
  "Statement":
  {
    "Effect": "Allow",
    "Action": "*",
    "Resource": "*"
  }
}
```

- Untuk detail API, lihat [GetPolicyVersion](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-IAMPolicyVersionList

Contoh kode berikut menunjukkan cara menggunakan `Get-IAMPolicyVersionList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan daftar versi kebijakan yang tersedia yang ARN-nya. **arn:aws:iam::123456789012:policy/MyManagedPolicy** Untuk mendapatkan dokumen kebijakan untuk versi tertentu, gunakan **Get-IAMPolicyVersion** perintah dan tentukan **VersionId** yang Anda inginkan.

```
Get-IAMPolicyVersionList -PolicyArn arn:aws:iam::123456789012:policy/MyManagedPolicy
```

Output:

CreateDate VersionId ----- -----	Document -----	IsDefaultVersion -----
2/12/2015 9:39:53 AM v2		True
2/12/2015 9:39:09 AM v1		False

- Untuk detail API, lihat [ListPolicyVersions](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-IAMRole

Contoh kode berikut menunjukkan cara menggunakan `Get-IAMRole`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan rincian **lambda_exec_role**. Ini termasuk dokumen kebijakan kepercayaan yang menentukan siapa yang dapat mengambil peran ini. Dokumen kebijakan adalah URL yang dikodekan dan dapat diterjemahkan menggunakan metode.NET. **UrlDecode** Dalam contoh ini, kebijakan asli menghapus semua spasi putih sebelum diunggah ke kebijakan. Untuk melihat dokumen kebijakan izin yang menentukan apa yang dapat dilakukan oleh seseorang yang mengasumsikan peran tersebut, gunakan kebijakan **Get-IAMRolePolicy** for inline, dan **Get-IAMPolicyVersion** untuk kebijakan terkelola terlampir.

```
$results = Get-IamRole -RoleName lambda_exec_role
$results | Format-List
```

Output:

```
Arn : arn:aws:iam::123456789012:role/lambda_exec_role
AssumeRolePolicyDocument : %7B%22Version%22%3A%22012-10-17%22%2C%22Statement%22%3A%5B%7B%22Sid%22%3A%22%22%2C%22Effect%22%3A%22Allow%22%2C%22Principal%22%3A%7B%22Service%22%3A%22lambda.amazonaws.com%22%7D%2C%22Action%22%3A%22sts%3AAssumeRole%22%7D%5D%7D
CreateDate : 4/2/2015 9:16:11 AM
Path : /
RoleId : 2YBIKAIBHNKB4EXAMPLE1
RoleName : lambda_exec_role
```

```
$policy = [System.Web.HttpUtility]::UrlDecode($results.AssumeRolePolicyDocument)
$policy
```

Output:

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

- Untuk detail API, lihat [GetRole](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-IAMRoleList

Contoh kode berikut menunjukkan cara menggunakan `Get-IAMRoleList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil daftar semua peran IAM di Akun AWS

```
Get-IAMRoleList
```

- Untuk detail API, lihat [ListRoles](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-IAMRolePolicy

Contoh kode berikut menunjukkan cara menggunakan `Get-IAMRolePolicy`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan dokumen kebijakan izin untuk kebijakan bernama **oneClick_lambda_exec_role_policy** yang disematkan dalam peran IAM. **lambda_exec_role** Dokumen kebijakan yang dihasilkan adalah URL yang dikodekan. Ini diterjemahkan dalam contoh ini dengan **UrlDecode** metode.NET.

```
$results = Get-IAMRolePolicy -RoleName lambda_exec_role -PolicyName
oneClick_lambda_exec_role_policy
$results
```

Output:

PolicyDocument	PolicyName
<pre> UserName ----- ----- %7B%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C%... oneClick_lambda_exec_role_policy lambda_exec_role </pre>	

```
[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.PolicyDocument)
```

Output:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "logs:*"
      ],
      "Resource": "arn:aws:logs:*:*:*"
    },
    {
      "Effect": "Allow",
      "Action": [
```

```

        "s3:GetObject",
        "s3:PutObject"
    ],
    "Resource": [
        "arn:aws:s3:::*"
    ]
}
]
}

```

- Untuk detail API, lihat [GetRolePolicy](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-IAMRolePolicyList

Contoh kode berikut menunjukkan cara menggunakan `Get-IAMRolePolicyList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan daftar nama kebijakan inline yang disematkan dalam peran IAM. **lambda_exec_role** Untuk melihat detail kebijakan inline, gunakan perintah **Get-IAMRolePolicy**.

```
Get-IAMRolePolicyList -RoleName lambda_exec_role
```

Output:

```
oneClick_lambda_exec_role_policy
```

- Untuk detail API, lihat [ListRolePolicies](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-IAMRoleTagList

Contoh kode berikut menunjukkan cara menggunakan `Get-IAMRoleTagList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil tag yang terkait dengan peran..

```
Get-IAMRoleTagList -RoleName MyRoleName
```

- Untuk detail API, lihat [ListRoleTags](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-IAMSAMLProvider

Contoh kode berikut menunjukkan cara menggunakan `Get-IAMSAMLProvider`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil detail tentang penyedia SAMP 2.0 yang ARM adalah `arn:aws:iam: :123456789012: Saml-provider/Samladfs`. Responsnya mencakup dokumen metadata yang Anda dapatkan dari penyedia identitas untuk membuat entitas penyedia AWS SAMP serta tanggal pembuatan dan kedaluwarsa.

```
Get-IAMSAMLProvider -SAMLProviderArn arn:aws:iam::123456789012:saml-provider/
SAMLADFS
```

Output:

CreateDate ValidUntil	SAMLMetadataDocument
----- -----	-----
12/23/2014 12:16:55 PM 12/23/2114 12:16:54 PM	<EntityDescriptor ID="_12345678-1234-5678-9012-example1...

- Untuk detail API, lihat [GetSamlProvider](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-IAMSAMLProviderList

Contoh kode berikut menunjukkan cara menggunakan `Get-IAMSAMLProviderList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil daftar penyedia SAMP 2.0 yang dibuat saat ini. Akun AWS Ini mengembalikan ARN, tanggal pembuatan, dan tanggal kedaluwarsa untuk setiap penyedia SAFL.

```
Get-IAMSAMLProviderList
```

Output:

Arn ValidUntil	CreateDate
-------------------	------------

```

---
-----
arn:aws:iam::123456789012:saml-provider/SAMLADFS    12/23/2014 12:16:55 PM
12/23/2114 12:16:54 PM

```

- Untuk detail API, lihat [Daftar SAMLProviders](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-IAMServerCertificate

Contoh kode berikut menunjukkan cara menggunakan `Get-IAMServerCertificate`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil rincian tentang sertifikat server bernama **MyServerCertificate**. Anda dapat menemukan rincian sertifikat di **CertificateBody** dan **ServerCertificateMetadata** properti.

```

$result = Get-IAMServerCertificate -ServerCertificateName MyServerCertificate
$result | format-list

```

Output:

```

CertificateBody      : -----BEGIN CERTIFICATE-----

MIICiTCCAfICCQD6m7oRw0uX0jANBgkqhkiG9w0BAQUFADCBiDELMakGA1UEBhMC

VVMxCzAJBgNVBAGTAldBMRAwDgYDVQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6

b24xFDASBgNVBAstC01BTSBDb25zb2x1MRIwEAYDVQQDEw1UZXR0Q21sYWMxHzAd

BgkqhkiG9w0BCQEWEG5vb25lQGFTYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN

MTIwNDI0MjA0NTIxWjCBiDELMakGA1UEBhMCVVMxCzAJBgNVBAGTAldBMRAwDgYD

VQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAstC01BTSBDb25z

b2x1MRIwEAYDVQQDEw1UZXR0Q21sYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFT

YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMak0dn

+a4GmWIWJ

21uUSfwfEvySwTc2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEI03IyNoH/

f0wYK8m9T

```

```

                                rDHudUZg3qX4waLG5M43q7Wgc/
MbQITx0USQv7c7ugFFDzQGBzZswY6786m86gpE

    Ibb30hjZnzcvcQAaRHhd1QWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
                                nUhVVxYUntneD9+h8Mg9q6q
+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb

    FFBjvSfpJI1J00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp3780D8uTs7fLvJx79LjSTb
                                NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
                                -----END CERTIFICATE-----

CertificateChain      :
ServerCertificateMetadata :
    Amazon.IdentityManagement.Model.ServerCertificateMetadata

```

```
$result.ServerCertificateMetadata
```

Output:

```

Arn          : arn:aws:iam::123456789012:server-certificate/0rg1/0rg2/
MyServerCertificate
Expiration    : 1/14/2018 9:52:36 AM
Path          : /0rg1/0rg2/
ServerCertificateId : ASCAJIFEXAMPLE17HQZYW
ServerCertificateName : MyServerCertificate
UploadDate    : 4/21/2015 11:14:16 AM

```

- Untuk detail API, lihat [GetServerCertificate](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-IAMServerCertificateList

Contoh kode berikut menunjukkan cara menggunakan `Get-IAMServerCertificateList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil daftar sertifikat server yang telah diunggah ke saat ini. Akun AWS

```
Get-IAMServerCertificateList
```

Output:

```

Arn                : arn:aws:iam::123456789012:server-certificate/0rg1/0rg2/
MyServerCertificate
Expiration         : 1/14/2018 9:52:36 AM
Path               : /0rg1/0rg2/
ServerCertificateId : ASCAJIFEXAMPLE17HQZYW
ServerCertificateName : MyServerCertificate
UploadDate        : 4/21/2015 11:14:16 AM

```

- Untuk detail API, lihat [ListServerCertificates](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-IAMServiceLastAccessedDetail

Contoh kode berikut menunjukkan cara menggunakan `Get-IAMServiceLastAccessedDetail`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memberikan rincian layanan yang terakhir diakses oleh entitas IAM (pengguna, grup, peran, atau kebijakan) yang terkait dalam panggilan Permintaan.

```
Request-IAMServiceLastAccessedDetail -Arn arn:aws:iam::123456789012:user/TestUser
```

Output:

```
f0b7a819-eab0-929b-dc26-ca598911cb9f
```

```
Get-IAMServiceLastAccessedDetail -JobId f0b7a819-eab0-929b-dc26-ca598911cb9f
```

- Untuk detail API, lihat [GetServiceLastAccessedDetails](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-IAMServiceLastAccessedDetailWithEntity

Contoh kode berikut menunjukkan cara menggunakan `Get-IAMServiceLastAccessedDetailWithEntity`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memberikan stempel waktu terakhir yang diakses untuk layanan dalam permintaan oleh entitas IAM masing-masing.

```
$results = Get-IAMServiceLastAccessedDetailWithEntity -JobId f0b7a819-eab0-929b-
dc26-ca598911cb9f -ServiceNamespace ec2
$results
```

Output:

```
EntityDetailsList : {Amazon.IdentityManagement.Model.EntityDetails}
Error              :
IsTruncated        : False
JobCompletionDate  : 12/29/19 11:19:31 AM
JobCreationDate    : 12/29/19 11:19:31 AM
JobStatus          : COMPLETED
Marker             :
```

```
$results.EntityDetailsList
```

Output:

```
EntityInfo                                LastAuthenticated
-----                                -
Amazon.IdentityManagement.Model.EntityInfo 11/16/19 3:47:00 PM
```

```
$results.EntityInfo
```

Output:

```
Arn  : arn:aws:iam::123456789012:user/TestUser
Id   : AIDA4NBK5CXF5TZHU1234
Name : TestUser
Path : /
Type : USER
```

- Untuk detail API, lihat [GetServiceLastAccessedDetailsWithEntities](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-IAMSigningCertificate

Contoh kode berikut menunjukkan cara menggunakan `Get-IAMSigningCertificate`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil rincian tentang sertifikat penandatanganan yang terkait dengan nama **Bob** pengguna.

```
Get-IAMSigningCertificate -UserName Bob
```

Output:

```
CertificateBody : -----BEGIN CERTIFICATE-----
                  MIICiTCcAfICCQD6m7oRw0uX0jANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
                  VVMxCzAJBgNVBAGTAldBMRAwDgYDVQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6
                  b24xFDASBgNVBA5TC0lBTSBDb25zb2x1MRIwEAYDVQQDEw1UZXRhbnQ2LsYWMxH3Ad
                  BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvi5jb20wHhcNMTEwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAGTAldBMRAwDgYD
                  VQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBA5TC0lBTSBDb25z
                  b2x1MRIwEAYDVQQDEw1UZXRhbnQ2LsYWMxH3AdBgkqhkiG9w0BCQEWEG5vb25lQGFt
                  YXpvi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
                  21uUSfwfEvySWtC2XADZ4nB+BLyGVIk60CpiwsZ3G93vUEI03IyNoH/f0wYK8m9T
                  rDHudUZg3qX4waLG5M43q7Wgc/MbQITx0USQv7c7ugFFDzQGBzZswY6786m86gpE
                  Ibb30hjZnzcvcQAaRHhd1QWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEA4n
                  UhVVxYUntneD9+h8Mg9q6q+auNKyExzyLwax1Aoo7TJHidbtS4J5iNmZgXL0Fkb
                  FFBjvSfpJI1J00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp3780D8uTs7fLvJx79LjSTb
                  NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
                  -----END CERTIFICATE-----

CertificateId   : Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU
Status          : Active
UploadDate      : 4/20/2015 1:26:01 PM
UserName        : Bob
```

- Untuk detail API, lihat [ListSigningCertificates](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-IAMUser

Contoh kode berikut menunjukkan cara menggunakan `Get-IAMUser`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil rincian tentang nama **David** pengguna.

```
Get-IAMUser -UserName David
```

Output:

```
Arn          : arn:aws:iam::123456789012:user/David
CreateDate   : 12/10/2014 3:39:27 PM
PasswordLastUsed : 3/19/2015 8:44:04 AM
Path         : /
UserId       : Y4FKWQCXTA52QEXAMPLE1
UserName     : David
```

Contoh 2: Contoh ini mengambil detail tentang pengguna IAM yang saat ini masuk.

```
Get-IAMUser
```

Output:

```
Arn          : arn:aws:iam::123456789012:user/Bob
CreateDate   : 10/16/2014 9:03:09 AM
PasswordLastUsed : 3/4/2015 12:12:33 PM
Path         : /
UserId       : 7K3GJEANSKZF2EXAMPLE2
UserName     : Bob
```

- Untuk detail API, lihat [GetUser](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-IAMUserList

Contoh kode berikut menunjukkan cara menggunakan `Get-IAMUserList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil koleksi pengguna saat ini Akun AWS.

```
Get-IAMUserList
```

Output:

```
Arn          : arn:aws:iam::123456789012:user/Administrator
CreateDate   : 10/16/2014 9:03:09 AM
```

```

PasswordLastUsed : 3/4/2015 12:12:33 PM
Path              : /
UserId            : 7K3GJEANSKZF2EXAMPLE1
UserName          : Administrator

Arn               : arn:aws:iam::123456789012:user/Bob
CreateDate        : 4/6/2015 12:54:42 PM
PasswordLastUsed  : 1/1/0001 12:00:00 AM
Path              : /
UserId            : L3EWNNDOM3YUEXAMPLE2
UserName          : bab

Arn               : arn:aws:iam::123456789012:user/David
CreateDate        : 12/10/2014 3:39:27 PM
PasswordLastUsed  : 3/19/2015 8:44:04 AM
Path              : /
UserId            : Y4FKWQCXTA52QEXAMPLE3
UserName          : David

```

- Untuk detail API, lihat [ListUsers](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-IAMUserPolicy

Contoh kode berikut menunjukkan cara menggunakan `Get-IAMUserPolicy`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil rincian kebijakan inline bernama **Davids_IAM_Admin_Policy** yang disematkan dalam nama pengguna IAM. **David** Dokumen kebijakan adalah URL yang dikodekan.

```
$results = Get-IAMUserPolicy -PolicyName Davids_IAM_Admin_Policy -UserName David
$results
```

Output:

PolicyDocument	PolicyName
UserName ----- ----- %7B%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C%... David	----- Davids_IAM_Admin_Policy

```
[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.PolicyDocument)
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iam:*"
      ],
      "Resource": [
        "*"
      ]
    }
  ]
}
```

- Untuk detail API, lihat [GetUserPolicy](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-IAMUserPolicyList

Contoh kode berikut menunjukkan cara menggunakan `Get-IAMUserPolicyList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil daftar nama kebijakan inline yang disematkan dalam nama pengguna IAM. **David**

```
Get-IAMUserPolicyList -UserName David
```

Output:

```
Davids_IAM_Admin_Policy
```

- Untuk detail API, lihat [ListUserPolicies](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-IAMUserTagList

Contoh kode berikut menunjukkan cara menggunakan `Get-IAMUserTagList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil tag yang terkait dengan pengguna.

```
Get-IAMUserTagList -UserName joe
```

- Untuk detail API, lihat [ListUserTags](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-IAMVirtualMFADevice

Contoh kode berikut menunjukkan cara menggunakan `Get-IAMVirtualMFADevice`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil koleksi perangkat MFA virtual yang ditetapkan untuk pengguna di AWS akun. **User** Properti masing-masing adalah objek dengan detail pengguna IAM tempat perangkat ditugaskan.

```
Get-IAMVirtualMFADevice -AssignmentStatus Assigned
```

Output:

```
Base32StringSeed :
EnableDate       : 4/13/2015 12:03:42 PM
QRCodePNG        :
SerialNumber     : arn:aws:iam::123456789012:mfa/David
User             : Amazon.IdentityManagement.Model.User

Base32StringSeed :
EnableDate       : 4/13/2015 12:06:41 PM
QRCodePNG        :
SerialNumber     : arn:aws:iam::123456789012:mfa/root-account-mfa-device
User             : Amazon.IdentityManagement.Model.User
```

- Untuk detail API, lihat [ListVirtualMfaDevices](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-IAMAccessKey

Contoh kode berikut menunjukkan cara menggunakan `New-IAMAccessKey`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat kunci akses baru dan secret access key pair dan menentukannya ke pengguna **David**. Pastikan Anda menyimpan **AccessKeyId** dan **SecretAccessKey** nilai ke file karena ini adalah satu-satunya waktu Anda dapat memperoleh file **SecretAccessKey**. Anda tidak dapat mengambilnya di lain waktu. Jika Anda kehilangan kunci rahasia, Anda harus membuat access key pair baru.

```
New-IAMAccessKey -UserName David
```

Output:

```
AccessKeyId      : AKIAIOSFODNN7EXAMPLE
CreateDate       : 4/13/2015 1:00:42 PM
SecretAccessKey  : wJalrXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY
Status          : Active
UserName        : David
```

- Untuk detail API, lihat [CreateAccessKey](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-IAMAccountAlias

Contoh kode berikut menunjukkan cara menggunakan `New-IAMAccountAlias`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengubah alias akun untuk AWS **mycompanyaws** akun Anda. Alamat halaman login pengguna diarahkan ke `panyaws.signin.aws.amazon.com/console`. `https://mycom` URL asli menggunakan nomor ID akun Anda, bukan alias (`https://<accountidnumber>.signin.aws.amazon.com/console`) terus berfungsi. Namun, semua berbasis alias yang didefinisikan sebelumnya URLs berhenti bekerja.

```
New-IAMAccountAlias -AccountAlias mycompanyaws
```

- Untuk detail API, lihat [CreateAccountAlias](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-IAMGroup

Contoh kode berikut menunjukkan cara menggunakan `New-IAMGroup`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat grup IAM baru bernama **Developers**.

```
New-IAMGroup -GroupName Developers
```

Output:

```
Arn          : arn:aws:iam::123456789012:group/Developers
CreateDate   : 4/14/2015 11:21:31 AM
GroupId      : QNEJ5PM4NFSQCEXAMPLE1
GroupName    : Developers
Path         : /
```

- Untuk detail API, lihat [CreateGroup](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-IAMInstanceProfile

Contoh kode berikut menunjukkan cara menggunakan **New-IAMInstanceProfile**.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat profil instans IAM baru bernama **ProfileForDevEC2Instance**. Anda harus menjalankan **Add-IAMRoleToInstanceProfile** perintah secara terpisah untuk mengaitkan profil instance dengan peran IAM yang ada yang memberikan izin ke instance. Terakhir, lampirkan profil instance ke EC2 instance saat Anda meluncurkannya. Untuk melakukan itu, gunakan **New-EC2Instance** cmdlet dengan parameter **InstanceProfile_Arn** or **InstanceProfile_Name**.

```
New-IAMInstanceProfile -InstanceProfileName ProfileForDevEC2Instance
```

Output:

```
Arn          : arn:aws:iam::123456789012:instance-profile/
ProfileForDevEC2Instance
CreateDate    : 4/14/2015 11:31:39 AM
InstanceId    : DYMFXL556EY46EXAMPLE1
InstanceProfileName : ProfileForDevEC2Instance
Path         : /
```

Roles : {}

- Untuk detail API, lihat [CreateInstanceProfile](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-IAMLoginProfile

Contoh kode berikut menunjukkan cara menggunakan `New-IAMLoginProfile`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat kata sandi (sementara) untuk pengguna IAM bernama Bob, dan menetapkan tanda yang mengharuskan pengguna untuk mengubah kata sandi saat **Bob** masuk berikutnya.

```
New-IAMLoginProfile -UserName Bob -Password P@ssw0rd -PasswordResetRequired $true
```

Output:

CreateDate	PasswordResetRequired	UserName
-----	-----	-----
4/14/2015 12:26:30 PM	True	Bob

- Untuk detail API, lihat [CreateLoginProfile](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-IAMOpenIDConnectProvider

Contoh kode berikut menunjukkan cara menggunakan `New-IAMOpenIDConnectProvider`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat penyedia IAM OIDC yang terkait dengan layanan penyedia kompatibel OIDC yang ditemukan di URL **https://example.oidcprovider.com** dan ID klien. **my-testapp-1** Penyedia OIDC memasok sidik jari. Untuk mengautentikasi sidik jari, ikuti langkah-langkah di <http://docs.aws.amazon.com/IAM/latest/UserGuide/identity-providers-oidc-obtain-thumbprint.html>.

```
New-IAMOpenIDConnectProvider -Url https://example.oidcprovider.com -ClientIDList my-testapp-1 -ThumbprintList 990F419EXAMPLEECF12DDEDA5EXAMPLE52F20D9E
```

Output:

```
arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com
```

- Untuk detail API, lihat [CreateOpenIdConnectProvider](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-IAMPolicy

Contoh kode berikut menunjukkan cara menggunakan `New-IAMPolicy`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat kebijakan IAM baru di AWS akun saat ini bernama **MySamplePolicy**. File **MySamplePolicy.json** menyediakan konten kebijakan. Perhatikan bahwa Anda harus menggunakan parameter **-Raw** switch untuk berhasil memproses file kebijakan JSON.

```
New-IAMPolicy -PolicyName MySamplePolicy -PolicyDocument (Get-Content -Raw  
MySamplePolicy.json)
```

Output:

```
Arn                : arn:aws:iam::123456789012:policy/MySamplePolicy  
AttachmentCount    : 0  
CreateDate         : 4/14/2015 2:45:59 PM  
DefaultVersionId   : v1  
Description        :  
IsAttachable       : True  
Path               : /  
PolicyId           : LD4KP6HVFE7WGEXAMPLE1  
PolicyName         : MySamplePolicy  
UpdateDate         : 4/14/2015 2:45:59 PM
```

- Untuk detail API, lihat [CreatePolicy](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-IAMPolicyVersion

Contoh kode berikut menunjukkan cara menggunakan `New-IAMPolicyVersion`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat versi “v2” baru dari kebijakan IAM yang ARN-nya **arn:aws:iam::123456789012:policy/MyPolicy** dan menjadikannya versi default. **NewPolicyVersion.json** menyediakan konten kebijakan. Perhatikan bahwa Anda harus menggunakan parameter **-Raw** switch untuk berhasil memproses file kebijakan JSON.

```
New-IAMPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/MyPolicy -
PolicyDocument (Get-content -Raw NewPolicyVersion.json) -SetAsDefault $true
```

Output:

CreateDate VersionId	Document	IsDefaultVersion
-----	-----	-----
4/15/2015 10:54:54 AM v2		True

- Untuk detail API, lihat [CreatePolicyVersion](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-IAMRole

Contoh kode berikut menunjukkan cara menggunakan **New-IAMRole**.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat peran baru bernama **MyNewRole** dan melampirkan padanya kebijakan yang ditemukan dalam file **NewRoleTrustPolicy.json**. Perhatikan bahwa Anda harus menggunakan parameter **-Raw** switch untuk berhasil memproses file kebijakan JSON. Dokumen kebijakan yang ditampilkan dalam output adalah URL yang dikodekan. Ini diterjemahkan dalam contoh ini dengan **UrlDecode** metode.NET.

```
$results = New-IAMRole -AssumeRolePolicyDocument (Get-Content -raw
NewRoleTrustPolicy.json) -RoleName MyNewRole
$results
```

Output:

```

Arn : arn:aws:iam::123456789012:role/MyNewRole
AssumeRolePolicyDocument : %7B%0D%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C%0D%0A%20%20%22Statement%22%3A%20%5B%0D%0A%20%20%20%20%7B%0D%0A%20%20%20%20%20%22Sid%22%3A%20%22%22%2C%0D%0A%20%20%20%20%20%20%20%22Effect%22%3A%20%22Allow%22%2C%0D%0A%20%20%20%20%20%20%20%22Principal%22%3A%20%7B%0D%0A%20%20%20%20%20%20%20%22AWS%22%3A%20%22arn%3Aaws%3Aiam%3A%3A123456789012%3ADavid%22%0D%0A%20%20%20%20%20%20%20%7D%2C%0D%0A%20%20%20%20%20%20%20%22Action%22%3A%20%22sts%3AAssumeRole%22%0D%0A%20%20%20%20%7D%0D%0A%20%20%5D%0D%0A%7D
CreateDate : 4/15/2015 11:04:23 AM
Path : /
RoleId : V5PAJI2KPN4EAEXAMPLE1
RoleName : MyNewRole

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.AssumeRolePolicyDocument)
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:David"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}

```

- Untuk detail API, lihat [CreateRole](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-IAMSAMLProvider

Contoh kode berikut menunjukkan cara menggunakan `New-IAMSAMLProvider`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat entitas penyedia SAMP baru di IAM. Ini dinamai **MySAMLProvider** dan dijelaskan oleh dokumen metadata SAMP yang ditemukan dalam file **SAMLMetaData.xml**, yang diunduh secara terpisah dari situs web penyedia layanan SAMP.

```
New-IAMSAMLProvider -Name MySAMLProvider -SAMLMetadataDocument (Get-Content -Raw SAMLMetaData.xml)
```

Output:

```
arn:aws:iam::123456789012:saml-provider/MySAMLProvider
```

- Untuk detail API, lihat [Membuat SAMLProvider](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-IAMServiceLinkedRole

Contoh kode berikut menunjukkan cara menggunakan **New-IAMServiceLinkedRole**.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat peran servicelinked untuk layanan penskalaan otomatis.

```
New-IAMServiceLinkedRole -AWSServiceName autoscaling.amazonaws.com -CustomSuffix RoleNameEndsWithThis -Description "My service-linked role to support autoscaling"
```

- Untuk detail API, lihat [CreateServiceLinkedRole](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-IAMUser

Contoh kode berikut menunjukkan cara menggunakan **New-IAMUser**.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat pengguna IAM bernama **Bob**. Jika Bob perlu masuk ke AWS konsol, maka Anda harus menjalankan perintah secara terpisah **New-IAMLoginProfile** untuk membuat profil masuk dengan kata sandi. Jika Bob perlu menjalankan AWS PowerShell atau

perintah CLI lintas platform atau AWS melakukan panggilan API, maka Anda harus menjalankan **New-IAMAccessKey** perintah secara terpisah untuk membuat kunci akses.

```
New-IAMUser -UserName Bob
```

Output:

```
Arn          : arn:aws:iam::123456789012:user/Bob
CreateDate   : 4/22/2015 12:02:11 PM
PasswordLastUsed : 1/1/0001 12:00:00 AM
Path         : /
UserId       : AIDAJWGEFDMEMEXAMPLE1
UserName     : Bob
```

- Untuk detail API, lihat [CreateUser](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-IAMVirtualMFADevice

Contoh kode berikut menunjukkan cara menggunakan **New-IAMVirtualMFADevice**.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat perangkat MFA virtual baru. Baris 2 dan 3 mengekstrak **Base32StringSeed** nilai yang dibutuhkan program perangkat lunak MFA virtual untuk membuat akun (sebagai alternatif dari kode QR). Setelah Anda mengkonfigurasi program dengan nilai, dapatkan dua kode otentikasi berurutan dari program. Terakhir, gunakan perintah terakhir untuk menautkan perangkat MFA virtual ke pengguna IAM **Bob** dan menyinkronkan akun dengan dua kode otentikasi.

```
$Device = New-IAMVirtualMFADevice -VirtualMFADeviceName BobsMFADevice
$SR = New-Object System.IO.StreamReader($Device.Base32StringSeed)
$base32stringseed = $SR.ReadToEnd()
$base32stringseed
CZWZMCQNW4DEXAMPLE3VOUGXJFZYSUW7EXAMPLECR4NJFD65GX2SLUDW2EXAMPLE
```

Output:

```
-- Pause here to enter base-32 string seed code into virtual MFA program to register
account. --
```

```
Enable-IAMMFADevice -SerialNumber $Device.SerialNumber -UserName Bob -
AuthenticationCode1 123456 -AuthenticationCode2 789012
```

Contoh 2: Contoh ini membuat perangkat MFA virtual baru. Baris 2 dan 3 mengekstrak **QRCodePNG** nilai dan menuliskannya ke file. Gambar ini dapat dipindai oleh program perangkat lunak MFA virtual untuk membuat akun (sebagai alternatif untuk memasukkan nilai StringSeed Base32 secara manual). Setelah Anda membuat akun di program MFA virtual Anda, dapatkan dua kode otentikasi berurutan dan masukkan dalam perintah terakhir untuk menautkan perangkat MFA virtual ke pengguna IAM dan menyinkronkan akun. **Bob**

```
$Device = New-IAMVirtualMFADevice -VirtualMFADeviceName BobsMFADevice
$BR = New-Object System.IO.BinaryReader($Device.QRCodePNG)
$BR.ReadBytes($BR.BaseStream.Length) | Set-Content -Encoding Byte -Path QRCode.png
```

Output:

```
-- Pause here to scan PNG with virtual MFA program to register account. --

Enable-IAMMFADevice -SerialNumber $Device.SerialNumber -UserName Bob -
AuthenticationCode1 123456 -AuthenticationCode2 789012
```

- Untuk detail API, lihat [CreateVirtualMfaDevice](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Publish-IAMServerCertificate

Contoh kode berikut menunjukkan cara menggunakan `Publish-IAMServerCertificate`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengunggah sertifikat server baru ke akun IAM. File yang berisi badan sertifikat, kunci pribadi, dan (opsional) rantai sertifikat semuanya harus dikodekan PEM. Perhatikan bahwa parameter memerlukan konten sebenarnya dari file daripada nama file. Anda harus menggunakan parameter **-Raw** sakelar untuk berhasil memproses konten file.

```
Publish-IAMServerCertificate -ServerCertificateName MyTestCert -CertificateBody
(Get-Content -Raw server.crt) -PrivateKey (Get-Content -Raw server.key)
```

Output:

```

Arn                : arn:aws:iam::123456789012:server-certificate/MyTestCert
Expiration         : 1/14/2018 9:52:36 AM
Path               : /
ServerCertificateId : ASCAJIEXAMPLE7J7HQZYW
ServerCertificateName : MyTestCert
UploadDate         : 4/21/2015 11:14:16 AM

```

- Untuk detail API, lihat [UploadServerCertificate](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Publish-IAMSigningCertificate

Contoh kode berikut menunjukkan cara menggunakan `Publish-IAMSigningCertificate`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengunggah sertifikat penandatanganan X.509 baru dan mengaitkannya dengan nama pengguna IAM. **Bob** File yang berisi badan sertifikat dikodekan PEM.

CertificateBody Parameter memerlukan konten sebenarnya dari file sertifikat daripada nama file. Anda harus menggunakan parameter **-Raw** sakelar untuk berhasil memproses file.

```

Publish-IAMSigningCertificate -UserName Bob -CertificateBody (Get-Content -Raw
SampleSigningCert.pem)

```

Output:

```

CertificateBody : -----BEGIN CERTIFICATE-----
MIICiTCCAFICQD6m7oRw0uX0jANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
VVMxCzAJBgNVBAGTAldBMRAwDgYDVQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6
b24xFDASBgNVBAstC01BTSBDb25zb2x1MRIwEAYDVQQDEw1UZXRhbnQ21sYWMxH2Ad
BgkqhkiG9w0BCQEWEG5vb251QGFTYXpvcjE5b20wHhcNMTEwNDI1MjA0NTIxWhcN
MTIwNDI1MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAGTAldBMRAwDgYD
VQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAstC01BTSBDb25z
b2x1MRIwEAYDVQQDEw1UZXRhbnQ21sYWMxH2AdBgkqhkiG9w0BCQEWEG5vb251QGFT
YXpvcjE5b20wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMak0dn+a4GmWIWJ
21uUSfwfEvySwTc2XADZ4nB+BLyGVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
rDHudUZg3qX4waLG5M43q7Wgc/MbQITx0USQv7c7ugFFDzQGBzZswY6786m86gpE
Ibb30hjZnzcvcQAaRHhd1QWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
nUHVvYUntneD9+h8Mg9q6q+auNKyExzyLwax1Aoo7TJHidbtS4J5iNmZgXL0Fkb
FFBjvSfpJl1J00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp3780D8uTs7fLvJx79LjStB
NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=

```

```

-----END CERTIFICATE-----
CertificateId    : Y3EK7RMEXAMPLESV33FCEXAMPLEHMJLU
Status          : Active
UploadDate      : 4/20/2015 1:26:01 PM
UserName        : Bob

```

- Untuk detail API, lihat [UploadSigningCertificate](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Register-IAMGroupPolicy

Contoh kode berikut menunjukkan cara menggunakan `Register-IAMGroupPolicy`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini melampirkan kebijakan terkelola pelanggan yang diberi nama **TesterPolicy** ke grup IAM. **Testers** Pengguna dalam grup tersebut langsung terpengaruh oleh izin yang ditentukan dalam versi default kebijakan tersebut.

```
Register-IAMGroupPolicy -GroupName Testers -PolicyArn
arn:aws:iam::123456789012:policy/TesterPolicy
```

Contoh 2: Contoh ini melampirkan kebijakan AWS terkelola bernama **AdministratorAccess** ke grup IAM. **Admins** Pengguna dalam grup tersebut langsung terpengaruh oleh izin yang ditentukan dalam versi terbaru kebijakan tersebut.

```
Register-IAMGroupPolicy -GroupName Admins -PolicyArn arn:aws:iam::aws:policy/
AdministratorAccess
```

- Untuk detail API, lihat [AttachGroupPolicy](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Register-IAMRolePolicy

Contoh kode berikut menunjukkan cara menggunakan `Register-IAMRolePolicy`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini melampirkan kebijakan AWS terkelola yang diberi nama **SecurityAudit** ke peran IAM. **CoSecurityAuditors** Pengguna yang menganggap peran tersebut langsung terpengaruh oleh izin yang ditentukan dalam versi terbaru kebijakan tersebut.

```
Register-IAMRolePolicy -RoleName CoSecurityAuditors -PolicyArn  
arn:aws:iam::aws:policy/SecurityAudit
```

- Untuk detail API, lihat [AttachRolePolicy](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Register-IAMUserPolicy

Contoh kode berikut menunjukkan cara menggunakan `Register-IAMUserPolicy`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini melampirkan kebijakan AWS terkelola bernama **AmazonCognitoPowerUser** ke pengguna IAM. **Bob** Pengguna langsung terpengaruh oleh izin yang ditentukan dalam versi terbaru kebijakan tersebut.

```
Register-IAMUserPolicy -UserName Bob -PolicyArn arn:aws:iam::aws:policy/  
AmazonCognitoPowerUser
```

- Untuk detail API, lihat [AttachUserPolicy](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-IAMAccessKey

Contoh kode berikut menunjukkan cara menggunakan `Remove-IAMAccessKey`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus AWS access key pair dengan ID kunci **AKIAIOSFODNN7EXAMPLE** dari nama **Bob** pengguna.

```
Remove-IAMAccessKey -AccessKeyId AKIAIOSFODNN7EXAMPLE -UserName Bob -Force
```

- Untuk detail API, lihat [DeleteAccessKey](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-IAMAccountAlias

Contoh kode berikut menunjukkan cara menggunakan `Remove-IAMAccountAlias`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus alias akun dari akun Anda Akun AWS. Halaman login pengguna dengan alias di <https://mycompanyaws.signin.aws.amazon.com/console> tidak lagi berfungsi. Sebagai gantinya, Anda harus menggunakan URL asli dengan nomor Akun AWS ID Anda di <https://signin.aws.amazon.com/console>. <accountidnumber>

```
Remove-IAMAccountAlias -AccountAlias mycompanyaws
```

- Untuk detail API, lihat [DeleteAccountAlias](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-IAMAccountPasswordPolicy

Contoh kode berikut menunjukkan cara menggunakan `Remove-IAMAccountPasswordPolicy`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus kebijakan kata sandi untuk Akun AWS dan mengatur ulang semua nilai ke default aslinya. Jika kebijakan kata sandi saat ini tidak ada, pesan galat berikut akan muncul: Kebijakan akun dengan nama PasswordPolicy tidak dapat ditemukan.

```
Remove-IAMAccountPasswordPolicy
```

- Untuk detail API, lihat [DeleteAccountPasswordPolicy](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-IAMClientIDFromOpenIDConnectProvider

Contoh kode berikut menunjukkan cara menggunakan `Remove-IAMClientIDFromOpenIDConnectProvider`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus ID klien **My-TestApp-3** dari daftar klien yang IDs terkait dengan penyedia IAM OIDC yang ARN-nya. **arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com**

```
Remove-IAMClientIDFromOpenIDConnectProvider -ClientID My-TestApp-3  
-OpenIDConnectProviderArn arn:aws:iam::123456789012:oidc-provider/  
example.oidcprovider.com
```

- Untuk detail API, lihat [RemoveClientIdFromOpenIdConnectProvider](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-IAMGroup

Contoh kode berikut menunjukkan cara menggunakan `Remove-IAMGroup`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus grup IAM bernama. **MyTestGroup** Perintah pertama menghapus setiap pengguna IAM yang merupakan anggota grup, dan perintah kedua menghapus grup IAM. Kedua perintah bekerja tanpa ada petunjuk untuk konfirmasi.

```
(Get-IAMGroup -GroupName MyTestGroup).Users | Remove-IAMUserFromGroup -GroupName  
MyTestGroup -Force  
Remove-IAMGroup -GroupName MyTestGroup -Force
```

- Untuk detail API, lihat [DeleteGroup](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-IAMGroupPolicy

Contoh kode berikut menunjukkan cara menggunakan `Remove-IAMGroupPolicy`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus kebijakan inline bernama **TesterPolicy** dari grup IAM. **Testers** Pengguna dalam grup tersebut segera kehilangan izin yang ditentukan dalam kebijakan tersebut.

```
Remove-IAMGroupPolicy -GroupName Testers -PolicyName TestPolicy
```

- Untuk detail API, lihat [DeleteGroupPolicy](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-IAMInstanceProfile

Contoh kode berikut menunjukkan cara menggunakan `Remove-IAMInstanceProfile`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus profil EC2 instance bernama **MyAppInstanceProfile**. Perintah pertama melepaskan peran apa pun dari profil instance, dan kemudian perintah kedua menghapus profil instance.

```
(Get-IAMInstanceProfile -InstanceProfileName MyAppInstanceProfile).Roles | Remove-IAMRoleFromInstanceProfile -InstanceProfileName MyAppInstanceProfile
Remove-IAMInstanceProfile -InstanceProfileName MyAppInstanceProfile
```

- Untuk detail API, lihat [DeleteInstanceProfile](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-IAMLoginProfile

Contoh kode berikut menunjukkan cara menggunakan `Remove-IAMLoginProfile`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus profil login dari pengguna IAM bernama **Bob**. Ini mencegah pengguna masuk ke konsol. AWS itu tidak mencegah pengguna menjalankan AWS CLI, PowerShell, atau panggilan API apa pun menggunakan kunci AWS akses yang mungkin masih dilampirkan ke akun pengguna.

```
Remove-IAMLoginProfile -UserName Bob
```

- Untuk detail API, lihat [DeleteLoginProfile](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-IAMOpenIDConnectProvider

Contoh kode berikut menunjukkan cara menggunakan `Remove-IAMOpenIDConnectProvider`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus penyedia IAM OIDC yang terhubung ke penyedia **example.oidcprovider.com**. Pastikan Anda memperbarui atau menghapus peran apa pun yang mereferensikan penyedia ini dalam **Principal** elemen kebijakan kepercayaan peran.

```
Remove-IAMOpenIDConnectProvider -OpenIDConnectProviderArn
arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com
```

- Untuk detail API, lihat [DeleteOpenIdConnectProvider](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-IAMPolicy

Contoh kode berikut menunjukkan cara menggunakan `Remove-IAMPolicy`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus kebijakan **arn:aws:iam::123456789012:policy/MySamplePolicy** ARN-nya. Sebelum Anda dapat menghapus kebijakan, Anda harus terlebih dahulu menghapus semua versi kecuali default dengan menjalankan **Remove-IAMPolicyVersion**. Anda juga harus melepaskan kebijakan dari setiap pengguna, grup, atau peran IAM.

```
Remove-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
```

Contoh 2: Contoh ini menghapus kebijakan dengan terlebih dahulu menghapus semua versi kebijakan non-default, melepaskannya dari semua entitas IAM terlampir, dan akhirnya menghapus kebijakan itu sendiri. Baris pertama mengambil objek kebijakan. Baris kedua mengambil semua versi kebijakan yang tidak ditandai sebagai versi default ke dalam koleksi dan kemudian menghapus setiap kebijakan dalam koleksi. Baris ketiga mengambil semua pengguna, grup, dan peran IAM yang dilampirkan kebijakan tersebut. Baris empat hingga enam melepaskan kebijakan dari setiap entitas terlampir. Baris terakhir menggunakan perintah ini untuk menghapus kebijakan terkelola serta versi default yang tersisa. Contohnya termasuk parameter **-Force** sakelar pada baris apa pun yang membutuhkannya untuk menekan permintaan konfirmasi.

```
$pol = Get-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
Get-IAMPolicyVersions -PolicyArn $pol.Arn | where {-not $_.IsDefaultVersion} |
  Remove-IAMPolicyVersion -PolicyArn $pol.Arn -force
$attached = Get-IAMEntitiesForPolicy -PolicyArn $pol.Arn
$attached.PolicyGroups | Unregister-IAMGroupPolicy -PolicyArn $pol.arn
$attached.PolicyRoles | Unregister-IAMRolePolicy -PolicyArn $pol.arn
$attached.PolicyUsers | Unregister-IAMUserPolicy -PolicyArn $pol.arn
Remove-IAMPolicy $pol.Arn -Force
```

- Untuk detail API, lihat [DeletePolicy](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-IAMPolicyVersion

Contoh kode berikut menunjukkan cara menggunakan `Remove-IAMPolicyVersion`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus versi yang diidentifikasi sebagai **v2** dari kebijakan yang **arn:aws:iam::123456789012:policy/MySamplePolicy** ARN-nya.

```
Remove-IAMPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy -VersionID v2
```

Contoh 2: Contoh ini menghapus kebijakan dengan terlebih dahulu menghapus semua versi kebijakan non-default dan kemudian menghapus kebijakan itu sendiri. Baris pertama mengambil objek kebijakan. Baris kedua mengambil semua versi kebijakan yang tidak ditandai sebagai default ke dalam koleksi dan kemudian menggunakan perintah ini untuk menghapus setiap kebijakan dalam koleksi. Baris terakhir menghapus kebijakan itu sendiri serta versi default yang tersisa. Perhatikan bahwa agar berhasil menghapus kebijakan terkelola, Anda juga harus melepaskan kebijakan dari pengguna, grup, atau peran apa pun dengan menggunakan perintah **Unregister-IAMUserPolicy**, **Unregister-IAMGroupPolicy**, dan **Unregister-IAMRolePolicy** perintah. Lihat contoh untuk **Remove-IAMPolicy** cmdlet.

```
$pol = Get-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
Get-IAMPolicyVersions -PolicyArn $pol.Arn | where {-not $_.IsDefaultVersion} |
  Remove-IAMPolicyVersion -PolicyArn $pol.Arn -force
Remove-IAMPolicy -PolicyArn $pol.Arn -force
```

- Untuk detail API, lihat [DeletePolicyVersion](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-IAMRole

Contoh kode berikut menunjukkan cara menggunakan `Remove-IAMRole`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus peran yang dinamai **MyNewRole** dari akun IAM saat ini. Sebelum Anda dapat menghapus peran, Anda harus terlebih dahulu menggunakan **Unregister-IAMRolePolicy** perintah untuk melepaskan kebijakan terkelola apa pun. Kebijakan sebaris dihapus dengan peran tersebut.

```
Remove-IAMRole -RoleName MyNewRole
```

Contoh 2: Contoh ini melepaskan kebijakan terkelola dari peran bernama **MyNewRole** dan kemudian menghapus peran tersebut. Baris pertama mengambil kebijakan terkelola yang melekat pada peran sebagai koleksi dan kemudian melepaskan setiap kebijakan dalam koleksi dari peran tersebut. Baris kedua menghapus peran itu sendiri. Kebijakan inline dihapus bersama dengan peran.

```
Get-IAMAttachedRolePolicyList -RoleName MyNewRole | Unregister-IAMRolePolicy -  
RoleName MyNewRole  
Remove-IAMRole -RoleName MyNewRole
```

- Untuk detail API, lihat [DeleteRole](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-IAMRoleFromInstanceProfile

Contoh kode berikut menunjukkan cara menggunakan `Remove-IAMRoleFromInstanceProfile`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus peran bernama **MyNewRole** dari profil EC2 instance bernama **MyNewRole**. Profil instance yang dibuat di konsol IAM selalu memiliki nama yang sama dengan peran, seperti dalam contoh ini. Jika Anda membuatnya di API atau CLI, maka mereka dapat memiliki nama yang berbeda.

```
Remove-IAMRoleFromInstanceProfile -InstanceProfileName MyNewRole -RoleName MyNewRole  
-Force
```

- Untuk detail API, lihat [RemoveRoleFromInstanceProfile](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-IAMRolePermissionsBoundary

Contoh kode berikut menunjukkan cara menggunakan `Remove-IAMRolePermissionsBoundary`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menunjukkan cara menghapus batas izin yang dilampirkan ke peran IAM.

```
Remove-IAMRolePermissionsBoundary -RoleName MyRoleName
```

- Untuk detail API, lihat [DeleteRolePermissionsBoundary](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-IAMRolePolicy

Contoh kode berikut menunjukkan cara menggunakan `Remove-IAMRolePolicy`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus kebijakan inline **S3AccessPolicy** yang disematkan dalam peran IAM. **S3BackupRole**

```
Remove-IAMRolePolicy -PolicyName S3AccessPolicy -RoleName S3BackupRole
```

- Untuk detail API, lihat [DeleteRolePolicy](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-IAMRoleTag

Contoh kode berikut menunjukkan cara menggunakan `Remove-IAMRoleTag`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus tag dari peran bernama "MyRoleName" dengan kunci tag sebagai "abac". Untuk menghapus beberapa tag, berikan daftar kunci tag yang dipisahkan koma.

```
Remove-IAMRoleTag -RoleName MyRoleName -TagKey "abac","xyzw"
```

- Untuk detail API, lihat [UntagRole](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-IAMSAMLProvider

Contoh kode berikut menunjukkan cara menggunakan `Remove-IAMSAMLProvider`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus penyedia IAM SALL 2.0 yang ARN-nya.

arn:aws:iam::123456789012:saml-provider/SAMLADFSPProvider

```
Remove-IAMSAMLProvider -SAMLProviderArn arn:aws:iam::123456789012:saml-provider/  
SAMLADFSPProvider
```

- Untuk detail API, lihat [Menghapus SAMLProvider](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-IAMServerCertificate

Contoh kode berikut menunjukkan cara menggunakan `Remove-IAMServerCertificate`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus sertifikat server bernama **MyServerCert**.

```
Remove-IAMServerCertificate -ServerCertificateName MyServerCert
```

- Untuk detail API, lihat [DeleteServerCertificate](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-IAMServiceLinkedRole

Contoh kode berikut menunjukkan cara menggunakan `Remove-IAMServiceLinkedRole`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus peran terkait layanan. Harap dicatat bahwa jika layanan masih menggunakan peran ini, maka perintah ini mengakibatkan kegagalan.

```
Remove-IAMServiceLinkedRole -RoleName  
AWSServiceRoleForAutoScaling_RoleNameEndsWithThis
```

- Untuk detail API, lihat [DeleteServiceLinkedRole](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-IAMSigningCertificate

Contoh kode berikut menunjukkan cara menggunakan `Remove-IAMSigningCertificate`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus sertifikat penandatanganan dengan ID **Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU** dari pengguna IAM bernama. **Bob**

```
Remove-IAMSigningCertificate -UserName Bob -CertificateId
Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU
```

- Untuk detail API, lihat [DeleteSigningCertificate](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-IAMUser

Contoh kode berikut menunjukkan cara menggunakan `Remove-IAMUser`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus nama pengguna IAM. **Bob**

```
Remove-IAMUser -UserName Bob
```

Contoh 2: Contoh ini menghapus nama pengguna IAM **Theresa** bersama dengan elemen apa pun yang harus dihapus terlebih dahulu.

```
$name = "Theresa"

# find any groups and remove user from them
$groups = Get-IAMGroupForUser -UserName $name
foreach ($group in $groups) { Remove-IAMUserFromGroup -GroupName $group.GroupName -
  UserName $name -Force }

# find any inline policies and delete them
$inlinepols = Get-IAMUserPolicies -UserName $name
foreach ($pol in $inlinepols) { Remove-IAMUserPolicy -PolicyName $pol -UserName
  $name -Force}

# find any managed polices and detach them
$managedpols = Get-IAMAttachedUserPolicies -UserName $name
foreach ($pol in $managedpols) { Unregister-IAMUserPolicy -PolicyArn $pol.PolicyArn
  -UserName $name }
```

```
# find any signing certificates and delete them
$certs = Get-IAMSigningCertificate -UserName $name
foreach ($cert in $certs) { Remove-IAMSigningCertificate -CertificateId
    $cert.CertificateId -UserName $name -Force }

# find any access keys and delete them
$keys = Get-IAMAccessKey -UserName $name
foreach ($key in $keys) { Remove-IAMAccessKey -AccessKeyId $key.AccessKeyId -
    UserName $name -Force }

# delete the user's login profile, if one exists - note: need to use try/catch to
# suppress not found error
try { $prof = Get-IAMLoginProfile -UserName $name -ea 0 } catch { out-null }
if ($prof) { Remove-IAMLoginProfile -UserName $name -Force }

# find any MFA device, detach it, and if virtual, delete it.
$mfa = Get-IAMMFADevice -UserName $name
if ($mfa) {
    Disable-IAMMFADevice -SerialNumber $mfa.SerialNumber -UserName $name
    if ($mfa.SerialNumber -like "arn:*") { Remove-IAMVirtualMFADevice -SerialNumber
        $mfa.SerialNumber }
}

# finally, remove the user
Remove-IAMUser -UserName $name -Force
```

- Untuk detail API, lihat [DeleteUser](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-IAMUserFromGroup

Contoh kode berikut menunjukkan cara menggunakan `Remove-IAMUserFromGroup`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus pengguna IAM **Bob** dari grup **Testers**.

```
Remove-IAMUserFromGroup -GroupName Testers -UserName Bob
```

Contoh 2: Contoh ini menemukan grup yang pengguna IAM **Theresa** adalah anggota, dan kemudian dihapus **Theresa** dari grup tersebut.

```
$groups = Get-IAMGroupForUser -UserName Theresa
```

```
foreach ($group in $groups) { Remove-IAMUserFromGroup -GroupName $group.GroupName -
  UserName Theresa -Force }
```

Contoh 3: Contoh ini menunjukkan cara alternatif untuk menghapus pengguna IAM **Bob** dari **Testers** grup.

```
Get-IAMGroupForUser -UserName Bob | Remove-IAMUserFromGroup -UserName Bob -GroupName
  Testers -Force
```

- Untuk detail API, lihat [RemoveUserFromGroup](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-IAMUserPermissionsBoundary

Contoh kode berikut menunjukkan cara menggunakan `Remove-IAMUserPermissionsBoundary`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menunjukkan cara menghapus batas izin yang dilampirkan ke pengguna IAM.

```
Remove-IAMUserPermissionsBoundary -UserName joe
```

- Untuk detail API, lihat [DeleteUserPermissionsBoundary](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-IAMUserPolicy

Contoh kode berikut menunjukkan cara menggunakan `Remove-IAMUserPolicy`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus kebijakan inline bernama **AccessToEC2Policy** yang disematkan dalam nama pengguna IAM. **Bob**

```
Remove-IAMUserPolicy -PolicyName AccessToEC2Policy -UserName Bob
```

Contoh 2: Contoh ini menemukan semua kebijakan inline yang disematkan dalam nama pengguna IAM **Theresa** dan kemudian menghapusnya.

```
$inlinepols = Get-IAMUserPolicies -UserName Theresa
foreach ($pol in $inlinepols) { Remove-IAMUserPolicy -PolicyName $pol -UserName
Theresa -Force}
```

- Untuk detail API, lihat [DeleteUserPolicy](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-IAMUserTag

Contoh kode berikut menunjukkan cara menggunakan `Remove-IAMUserTag`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus tag dari pengguna bernama “joe” dengan kunci tag sebagai “abac” dan “xyzw”. Untuk menghapus beberapa tag, berikan daftar kunci tag yang dipisahkan koma.

```
Remove-IAMUserTag -UserName joe -TagKey "abac","xyzw"
```

- Untuk detail API, lihat [UntagUser](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-IAMVirtualMFADevice

Contoh kode berikut menunjukkan cara menggunakan `Remove-IAMVirtualMFADevice`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus perangkat MFA virtual IAM yang ARN-nya.

arn:aws:iam::123456789012:mfa/bob

```
Remove-IAMVirtualMFADevice -SerialNumber arn:aws:iam::123456789012:mfa/bob
```

Contoh 2: Contoh ini memeriksa untuk melihat apakah pengguna IAM Theresa memiliki perangkat MFA yang ditetapkan. Jika ditemukan, perangkat dinonaktifkan untuk pengguna IAM. Jika perangkat virtual, maka itu juga dihapus.

```
$mfa = Get-IAMMFADevice -UserName Theresa
if ($mfa) {
    Disable-IAMMFADevice -SerialNumber $mfa.SerialNumber -UserName $name
    if ($mfa.SerialNumber -like "arn:*") { Remove-IAMVirtualMFADevice -SerialNumber
$mfa.SerialNumber }
```

```
}
```

- Untuk detail API, lihat [DeleteVirtualMfaDevice](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Request-IAMCredentialReport

Contoh kode berikut menunjukkan cara menggunakan `Request-IAMCredentialReport`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini meminta pembuatan laporan baru, yang dapat dilakukan setiap empat jam. Jika laporan terakhir masih terbaru, bidang Negara berbunyi **COMPLETE**. Gunakan **Get-IAMCredentialReport** untuk melihat laporan yang sudah selesai.

```
Request-IAMCredentialReport
```

Output:

Description	State
-----	----
No report exists. Starting a new report generation task	STARTED

- Untuk detail API, lihat [GenerateCredentialReport](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Request-IAMServiceLastAccessedDetail

Contoh kode berikut menunjukkan cara menggunakan `Request-IAMServiceLastAccessedDetail`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini setara dengan cmdlet API. `GenerateServiceLastAccessedDetails` Ini menyediakan dengan id pekerjaan yang dapat digunakan di `Get-IAMServiceLastAccessedDetail` dan Dapatkan- `IAMService LastAccessedDetailWithEntity`

```
Request-IAMServiceLastAccessedDetail -Arn arn:aws:iam::123456789012:user/TestUser
```

- Untuk detail API, lihat [GenerateServiceLastAccessedDetails](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Set-IAMDefaultPolicyVersion

Contoh kode berikut menunjukkan cara menggunakan `Set-IAMDefaultPolicyVersion`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menetapkan **v2** versi kebijakan yang ARN-nya **arn:aws:iam::123456789012:policy/MyPolicy** sebagai versi aktif default.

```
Set-IAMDefaultPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/MyPolicy -VersionId v2
```

- Untuk detail API, lihat [SetDefaultPolicyVersion](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Set-IAMRolePermissionsBoundary

Contoh kode berikut menunjukkan cara menggunakan `Set-IAMRolePermissionsBoundary`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menunjukkan cara mengatur batas Izin untuk Peran IAM. Anda dapat menetapkan Kebijakan AWS terkelola atau Kebijakan khusus sebagai batas izin.

```
Set-IAMRolePermissionsBoundary -RoleName MyRoleName -PermissionsBoundary arn:aws:iam::123456789012:policy/intern-boundary
```

- Untuk detail API, lihat [PutRolePermissionsBoundary](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Set-IAMUserPermissionsBoundary

Contoh kode berikut menunjukkan cara menggunakan `Set-IAMUserPermissionsBoundary`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menunjukkan cara mengatur batas Izin untuk pengguna. Anda dapat menetapkan Kebijakan AWS terkelola atau Kebijakan khusus sebagai batas izin.

```
Set-IAMUserPermissionsBoundary -UserName joe -PermissionsBoundary  
arn:aws:iam::123456789012:policy/intern-boundary
```

- Untuk detail API, lihat [PutUserPermissionsBoundary](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Sync-IAMMFADevice

Contoh kode berikut menunjukkan cara menggunakan Sync-IAMMFADevice.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menyinkronkan perangkat MFA yang terkait dengan pengguna IAM **Bob** dan ARN-nya dengan program autentikator yang **arn:aws:iam::123456789012:mfa/bob** menyediakan dua kode otentikasi.

```
Sync-IAMMFADevice -SerialNumber arn:aws:iam::123456789012:mfa/theresa -  
AuthenticationCode1 123456 -AuthenticationCode2 987654 -UserName Bob
```

Contoh 2: Contoh ini menyinkronkan perangkat IAM MFA yang dikaitkan dengan pengguna **Theresa** IAM dengan perangkat fisik yang memiliki nomor seri **ABCD12345678** dan yang menyediakan dua kode otentikasi.

```
Sync-IAMMFADevice -SerialNumber ABCD12345678 -AuthenticationCode1 123456 -  
AuthenticationCode2 987654 -UserName Theresa
```

- Untuk detail API, lihat [ResyncMfaDevice](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Unregister-IAMGroupPolicy

Contoh kode berikut menunjukkan cara menggunakan Unregister-IAMGroupPolicy.

Alat untuk PowerShell V4

Contoh 1: Contoh ini melepaskan kebijakan grup terkelola yang **arn:aws:iam::123456789012:policy/TesterAccessPolicy** ARNnya berasal dari grup bernama. **Testers**

```
Unregister-IAMGroupPolicy -GroupName Testers -PolicyArn  
arn:aws:iam::123456789012:policy/TesterAccessPolicy
```

Contoh 2: Contoh ini menemukan semua kebijakan terkelola yang dilampirkan pada grup bernama **Testers** dan memisahnya dari grup.

```
Get-IAMAttachedGroupPolicies -GroupName Testers | Unregister-IAMGroupPolicy -  
Groupname Testers
```

- Untuk detail API, lihat [DetachGroupPolicy](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Unregister-IAMRolePolicy

Contoh kode berikut menunjukkan cara menggunakan `Unregister-IAMRolePolicy`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini melepaskan kebijakan grup terkelola yang **arn:aws:iam::123456789012:policy/FederatedTesterAccessPolicy** ARNnya berasal dari peran bernama. **FedTesterRole**

```
Unregister-IAMRolePolicy -RoleName FedTesterRole -PolicyArn  
arn:aws:iam::123456789012:policy/FederatedTesterAccessPolicy
```

Contoh 2: Contoh ini menemukan semua kebijakan terkelola yang dilampirkan pada peran yang diberi nama **FedTesterRole** dan memisahnya dari peran.

```
Get-IAMAttachedRolePolicyList -RoleName FedTesterRole | Unregister-IAMRolePolicy -  
Rolename FedTesterRole
```

- Untuk detail API, lihat [DetachRolePolicy](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Unregister-IAMUserPolicy

Contoh kode berikut menunjukkan cara menggunakan `Unregister-IAMUserPolicy`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini melepaskan kebijakan terkelola yang ARNnya berasal dari **arn:aws:iam::123456789012:policy/TesterPolicy** nama pengguna IAM. **Bob**

```
Unregister-IAMUserPolicy -UserName Bob -PolicyArn arn:aws:iam::123456789012:policy/TesterPolicy
```

Contoh 2: Contoh ini menemukan semua kebijakan terkelola yang dilampirkan ke pengguna IAM bernama **Theresa** dan melepaskan kebijakan tersebut dari pengguna.

```
Get-IAMAttachedUserPolicyList -UserName Theresa | Unregister-IAMUserPolicy -Username Theresa
```

- Untuk detail API, lihat [DetachUserPolicy](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-IAMAccessKey

Contoh kode berikut menunjukkan cara menggunakan `Update-IAMAccessKey`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengubah status kunci akses **AKIAIOSFODNN7EXAMPLE** untuk pengguna IAM bernama **Bob**. **Inactive**

```
Update-IAMAccessKey -UserName Bob -AccessKeyId AKIAIOSFODNN7EXAMPLE -Status Inactive
```

- Untuk detail API, lihat [UpdateAccessKey](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-IAMAccountPasswordPolicy

Contoh kode berikut menunjukkan cara menggunakan `Update-IAMAccountPasswordPolicy`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memperbarui kebijakan kata sandi untuk akun dengan pengaturan yang ditentukan. Perhatikan bahwa parameter apa pun yang tidak termasuk dalam perintah tidak dibiarkan tidak dimodifikasi. Sebaliknya, mereka diatur ulang ke nilai default.

```
Update-IAMAccountPasswordPolicy -AllowUsersToChangePasswords $true -HardExpiry $false -MaxPasswordAge 90 -MinimumPasswordLength 8 -PasswordReusePrevention 20 -RequireLowercaseCharacters $true -RequireNumbers $true -RequireSymbols $true -RequireUppercaseCharacters $true
```

- Untuk detail API, lihat [UpdateAccountPasswordPolicy](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-IAMAssumeRolePolicy

Contoh kode berikut menunjukkan cara menggunakan `Update-IAMAssumeRolePolicy`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memperbarui peran IAM yang diberi nama **ClientRole** dengan kebijakan kepercayaan baru, yang isinya berasal dari file **ClientRolePolicy.json**. Perhatikan bahwa Anda harus menggunakan parameter **-Raw** switch untuk berhasil memproses isi file JSON.

```
Update-IAMAssumeRolePolicy -RoleName ClientRole -PolicyDocument (Get-Content -raw ClientRolePolicy.json)
```

- Untuk detail API, lihat [UpdateAssumeRolePolicy](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-IAMGroup

Contoh kode berikut menunjukkan cara menggunakan `Update-IAMGroup`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengganti nama grup IAM menjadi. **Testers AppTesters**

```
Update-IAMGroup -GroupName Testers -NewGroupName AppTesters
```

Contoh 2: Contoh ini mengubah jalur grup IAM **AppTesters** menjadi **/Org1/Org2/**. Ini mengubah ARN untuk grup menjadi. **arn:aws:iam::123456789012:group/Org1/Org2/AppTesters**

```
Update-IAMGroup -GroupName AppTesters -NewPath /Org1/Org2/
```

- Untuk detail API, lihat [UpdateGroup](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-IAMLoginProfile

Contoh kode berikut menunjukkan cara menggunakan `Update-IAMLoginProfile`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menetapkan kata sandi sementara baru untuk pengguna **IAMBob**, dan mengharuskan pengguna untuk mengubah kata sandi saat pengguna masuk berikutnya.

```
Update-IAMLoginProfile -UserName Bob -Password "P@ssw0rd1234" -PasswordResetRequired $true
```

- Untuk detail API, lihat [UpdateLoginProfile](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-IAMOpenIDConnectProviderThumbprint

Contoh kode berikut menunjukkan cara menggunakan `Update-IAMOpenIDConnectProviderThumbprint`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memperbarui daftar cap jempol sertifikat untuk penyedia OIDC yang ARN-nya menggunakan sidik jari baru. **arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com** Penyedia OIDC membagikan nilai baru ketika sertifikat yang terkait dengan penyedia berubah.

```
Update-IAMOpenIDConnectProviderThumbprint -OpenIDConnectProviderArn arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com -ThumbprintList 7359755EXAMPLEabc3060bce3EXAMPLEec4542a3
```

- Untuk detail API, lihat [UpdateOpenIdConnectProviderThumbprint](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-IAMRole

Contoh kode berikut menunjukkan cara menggunakan `Update-IAMRole`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memperbarui deskripsi peran dan nilai durasi sesi maksimum (dalam detik) yang sesi peran dapat diminta.

```
Update-IAMRole -RoleName MyRoleName -Description "My testing role" -  
MaxSessionDuration 43200
```

- Untuk detail API, lihat [UpdateRole](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-IAMRoleDescription

Contoh kode berikut menunjukkan cara menggunakan `Update-IAMRoleDescription`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memperbarui deskripsi peran IAM di akun Anda.

```
Update-IAMRoleDescription -RoleName MyRoleName -Description "My testing role"
```

- Untuk detail API, lihat [UpdateRoleDescription](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-IAMSAMLProvider

Contoh kode berikut menunjukkan cara menggunakan `Update-IAMSAMLProvider`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memperbarui penyedia SAMP di IAM yang ARN-nya **arn:aws:iam::123456789012:saml-provider/SAMLADFS** dengan dokumen metadata SAMP baru dari file. **SAMLMetaData.xml** Perhatikan bahwa Anda harus menggunakan parameter **-Raw** switch untuk berhasil memproses isi file JSON.

```
Update-IAMSAMLProvider -SAMLProviderArn arn:aws:iam::123456789012:saml-provider/  
SAMLADFS -SAMLMetadataDocument (Get-Content -Raw SAMLMetaData.xml)
```

- Untuk detail API, lihat [UpdateSamlProvider](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-IAMServerCertificate

Contoh kode berikut menunjukkan cara menggunakan `Update-IAMServerCertificate`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengganti nama sertifikat yang dinamai **MyServerCertificate** menjadi **MyRenamedServerCertificate**.

```
Update-IAMServerCertificate -ServerCertificateName MyServerCertificate -  
NewServerCertificateName MyRenamedServerCertificate
```

Contoh 2: Contoh ini memindahkan sertifikat yang diberi nama **MyServerCertificate** ke path `/Org1/Org 2/`. Ini mengubah ARN untuk sumber daya menjadi.

**arn:aws:iam::123456789012:server-certificate/Org1/Org2/
MyServerCertificate**

```
Update-IAMServerCertificate -ServerCertificateName MyServerCertificate -NewPath /  
Org1/Org2/
```

- Untuk detail API, lihat [UpdateServerCertificate](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-IAMSigningCertificate

Contoh kode berikut menunjukkan cara menggunakan `Update-IAMSigningCertificate`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memperbarui sertifikat yang terkait dengan nama pengguna IAM **Bob** dan yang ID sertifikatnya **Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU** untuk menandainya sebagai tidak aktif.

```
Update-IAMSigningCertificate -CertificateId Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU -  
UserName Bob -Status Inactive
```

- Untuk detail API, lihat [UpdateSigningCertificate](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-IAMUser

Contoh kode berikut menunjukkan cara menggunakan `Update-IAMUser`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengganti nama pengguna IAM menjadi **Bob Robert**

```
Update-IAMUser -UserName Bob -NewUserName Robert
```

Contoh 2: Contoh ini mengubah jalur Pengguna IAM **Bob** ke **/Org1/Org2/**, yang secara efektif mengubah ARN untuk pengguna. **arn:aws:iam::123456789012:user/Org1/Org2/bob**

```
Update-IAMUser -UserName Bob -NewPath /Org1/Org2/
```

- Untuk detail API, lihat [UpdateUser](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Write-IAMGroupPolicy

Contoh kode berikut menunjukkan cara menggunakan `Write-IAMGroupPolicy`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat kebijakan inline bernama **AppTesterPolicy** dan menyematkannya dalam grup IAM. **AppTesters** Jika kebijakan inline dengan nama yang sama sudah ada, maka itu akan ditimpa. Konten kebijakan JSON datang file **apptesterpolicy.json**. Perhatikan bahwa Anda harus menggunakan **-Raw** parameter untuk berhasil memproses konten file JSON.

```
Write-IAMGroupPolicy -GroupName AppTesters -PolicyName AppTesterPolicy -  
PolicyDocument (Get-Content -Raw apptesterpolicy.json)
```

- Untuk detail API, lihat [PutGroupPolicy](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Write-IAMRolePolicy

Contoh kode berikut menunjukkan cara menggunakan `Write-IAMRolePolicy`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat kebijakan inline bernama **FedTesterRolePolicy** dan menyimpannya dalam peran IAM. **FedTesterRole** Jika kebijakan inline dengan nama yang sama sudah ada, maka itu akan ditimpa. Konten kebijakan JSON berasal dari file **FedTesterPolicy.json**. Perhatikan bahwa Anda harus menggunakan **-Raw** parameter untuk berhasil memproses konten file JSON.

```
Write-IAMRolePolicy -RoleName FedTesterRole -PolicyName FedTesterRolePolicy -  
PolicyDocument (Get-Content -Raw FedTesterPolicy.json)
```

- Untuk detail API, lihat [PutRolePolicy](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Write-IAMUserPolicy

Contoh kode berikut menunjukkan cara menggunakan **Write-IAMUserPolicy**.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat kebijakan inline bernama **EC2AccessPolicy** dan menyimpannya di pengguna IAM. **Bob** Jika kebijakan inline dengan nama yang sama sudah ada, maka itu akan ditimpa. Konten kebijakan JSON berasal dari file **EC2AccessPolicy.json**. Perhatikan bahwa Anda harus menggunakan **-Raw** parameter untuk berhasil memproses konten file JSON.

```
Write-IAMUserPolicy -UserName Bob -PolicyName EC2AccessPolicy -PolicyDocument (Get-  
Content -Raw EC2AccessPolicy.json)
```

- Untuk detail API, lihat [PutUserPolicy](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Contoh Kinesis menggunakan Alat untuk V4 PowerShell

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan Kinesis.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

Get-KINRecord

Contoh kode berikut menunjukkan cara menggunakan `Get-KINRecord`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menunjukkan cara mengembalikan dan mengekstrak data dari serangkaian satu atau lebih catatan. Iterator `supplierd` untuk `Get-KINRecord` menentukan posisi awal dari catatan untuk kembali yang dalam contoh ini ditangkap ke dalam variabel, `$records`. Setiap catatan individu kemudian dapat diakses dengan mengindeks koleksi `$records`. Dengan asumsi data dalam catatan adalah teks yang dikodekan UTF-8, perintah terakhir menunjukkan bagaimana Anda dapat mengekstrak data dari objek dan mengembalikannya sebagai teks ke konsol. `MemoryStream`

```
$records
$records = Get-KINRecord -ShardIterator "AAAAAAAAAAGIc....9VnbiRNpP"
```

Output:

MillisBehindLatest	NextShardIterator	Records
0	AAAAAAAAAAERNIq...uDn11HuUs	{Key1, Key2}

```
$records.Records[0]
```

Output:

ApproximateArrivalTimestamp	Data	PartitionKey	SequenceNumber
3/7/2016 5:14:33 PM	System.IO.MemoryStream	Key1	
4955986459776...	931586		

```
[Text.Encoding]::UTF8.GetString($records.Records[0].Data.ToArray())
```

Output:

```
test data from string
```

- Untuk detail API, lihat [GetRecords](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-KINShardIterator

Contoh kode berikut menunjukkan cara menggunakan `Get-KINShardIterator`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan iterator shard untuk shard tertentu dan posisi awal. Rincian pengidentifikasi pecahan dan nomor urut dapat diperoleh dari output `Get-KINStream` cmdlet, dengan mereferensikan koleksi Shards dari objek aliran yang dikembalikan. Iterator yang dikembalikan dapat digunakan dengan `Get-KINRecord` cmdlet untuk menarik catatan data dalam pecahan.

```
Get-KINShardIterator -StreamName "mystream" -ShardId "shardId-000000000000" -  
ShardIteratorType AT_SEQUENCE_NUMBER -StartingSequenceNumber "495598645..."
```

Output:

```
AAAAAAAAAAGIc....9VnbiRNaP
```

- Untuk detail API, lihat [GetShardIterator](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-KINStream

Contoh kode berikut menunjukkan cara menggunakan `Get-KINStream`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan rincian aliran tertentu.

```
Get-KINStream -StreamName "mystream"
```

Output:

```
HasMoreShards      : False
RetentionPeriodHours : 24
Shards             : {}
StreamARN          : arn:aws:kinesis:us-west-2:123456789012:stream/mystream
StreamName         : mystream
StreamStatus       : ACTIVE
```

- Untuk detail API, lihat [DescribeStream](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-KINStream

Contoh kode berikut menunjukkan cara menggunakan `New-KINStream`.

Alat untuk PowerShell V4

Contoh 1: Membuat aliran baru. Secara default cmdlet ini tidak mengembalikan output sehingga `PassThru` sakelar - ditambahkan untuk mengembalikan nilai yang diberikan ke `StreamName` parameter - untuk penggunaan selanjutnya.

```
$streamName = New-KINStream -StreamName "mystream" -ShardCount 1 -PassThru
```

- Untuk detail API, lihat [CreateStream](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-KINStream

Contoh kode berikut menunjukkan cara menggunakan `Remove-KINStream`.

Alat untuk PowerShell V4

Contoh 1: Menghapus aliran yang ditentukan. Anda diminta untuk konfirmasi sebelum perintah dijalankan. Untuk menekan konfirmasi yang diminta, gunakan sakelar `-Force`.

```
Remove-KINStream -StreamName "mystream"
```

- Untuk detail API, lihat [DeleteStream](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Write-KINRecord

Contoh kode berikut menunjukkan cara menggunakan Write-KINRecord.

Alat untuk PowerShell V4

Contoh 1: Menulis catatan yang berisi string yang dipasang ke parameter -Text.

```
Write-KINRecord -Text "test data from string" -StreamName "mystream" -PartitionKey "Key1"
```

Contoh 2: Menulis catatan yang berisi data yang terkandung dalam file yang ditentukan. File diperlakukan sebagai urutan byte sehingga jika berisi teks, itu harus ditulis dengan pengkodean yang diperlukan sebelum menggunakannya dengan cmdlet ini.

```
Write-KINRecord -FilePath "C:\TestData.txt" -StreamName "mystream" -PartitionKey "Key2"
```

- Untuk detail API, lihat [PutRecord](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Contoh Lambda menggunakan Alat untuk V4 PowerShell

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan Lambda.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

Add-LMResourceTag

Contoh kode berikut menunjukkan cara menggunakan Add-LMResourceTag.

Alat untuk PowerShell V4

Contoh 1: Menambahkan tiga tag (Washington, Oregon dan California) dan nilai terkaitnya ke fungsi tertentu yang diidentifikasi oleh ARN-nya.

```
Add-LMResourceTag -Resource "arn:aws:lambda:us-west-2:123456789012:function:MyFunction" -Tag @{ "Washington" = "Olympia"; "Oregon" = "Salem"; "California" = "Sacramento" }
```

- Untuk detail API, lihat [TagResource](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-LMAccountSetting

Contoh kode berikut menunjukkan cara menggunakan `Get-LMAccountSetting`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini ditampilkan untuk membandingkan Batas Akun dan Penggunaan Akun

```
Get-LMAccountSetting | Select-Object
@{Name="TotalCodeSizeLimit";Expression={$_.AccountLimit.TotalCodeSize}},
@{Name="TotalCodeSizeUsed";Expression={$_.AccountUsage.TotalCodeSize}}
```

Output:

TotalCodeSizeLimit	TotalCodeSizeUsed
80530636800	15078795

- Untuk detail API, lihat [GetAccountSettings](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-LMAlias

Contoh kode berikut menunjukkan cara menggunakan `Get-LMAlias`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil bobot Routing Config untuk Alias Fungsi Lambda tertentu.

```
Get-LMAlias -FunctionName "MylambdaFunction123" -Name "newlabel1" -Select
RoutingConfig
```

Output:

```
AdditionalVersionWeights
-----
{[1, 0.6]}
```

- Untuk detail API, lihat [GetAlias](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-LMFunctionConcurrency

Contoh kode berikut menunjukkan cara menggunakan `Get-LMFunctionConcurrency`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan konkurensi Cadangan untuk Fungsi Lambda

```
Get-LMFunctionConcurrency -FunctionName "MylambdaFunction123" -Select *
```

Output:

```
ReservedConcurrentExecutions
-----
100
```

- Untuk detail API, lihat [GetFunctionConcurrency](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-LMFunctionConfiguration

Contoh kode berikut menunjukkan cara menggunakan `Get-LMFunctionConfiguration`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan konfigurasi spesifik versi dari Fungsi Lambda.

```
Get-LMFunctionConfiguration -FunctionName "MylambdaFunction123" -Qualifier
"PowershellAlias"
```

Output:

```
CodeSha256 : uW0W0R7z+f0VyLuUg7+/D08hkMFsq0SF4seuyUZJ/R8=
```

```

CodeSize                : 1426
DeadLetterConfig         : Amazon.Lambda.Model.DeadLetterConfig
Description              : Version 3 to test Aliases
Environment              : Amazon.Lambda.Model.EnvironmentResponse
FunctionArn              : arn:aws:lambda:us-
east-1:123456789012:function:MyLambdaFunction123
                        :PowershellAlias
FunctionName             : MyLambdaFunction123
Handler                  : lambda_function.launch_instance
KMSKeyArn                :
LastModified             : 2019-12-25T09:52:59.872+0000
LastUpdateStatus         : Successful
LastUpdateStatusReason   :
LastUpdateStatusReasonCode :
Layers                   : {}
MasterArn                :
MemorySize               : 128
RevisionId               : 5d7de38b-87f2-4260-8f8a-e87280e10c33
Role                     : arn:aws:iam::123456789012:role/service-role/lambda
Runtime                  : python3.8
State                    : Active
StateReason              :
StateReasonCode          :
Timeout                  : 600
TracingConfig             : Amazon.Lambda.Model.TracingConfigResponse
Version                  : 4
VpcConfig                : Amazon.Lambda.Model.VpcConfigDetail

```

- Untuk detail API, lihat [GetFunctionConfiguration](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-LMFunctionList

Contoh kode berikut menunjukkan cara menggunakan `Get-LMFunctionList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menampilkan semua fungsi Lambda dengan ukuran kode yang diurutkan

```

Get-LMFunctionList | Sort-Object -Property CodeSize | Select-Object FunctionName,
RunTime, Timeout, CodeSize

```

Output:

FunctionName	Runtime	Timeout
CodeSize		
-----	-----	-----
test	python2.7	3
243		
MylambdaFunction123	python3.8	600
659		
myfuncpython1	python3.8	303
675		

- Untuk detail API, lihat [ListFunctions](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-LMPolicy

Contoh kode berikut menunjukkan cara menggunakan `Get-LMPolicy`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menampilkan kebijakan Fungsi dari fungsi Lambda

```
Get-LMPolicy -FunctionName test -Select Policy
```

Output:

```
{
  "Version": "2012-10-17",
  "Id": "default",
  "Statement": [
    {
      "Sid": "xxxx",
      "Effect": "Allow",
      "Principal": {
        "Service": "sns.amazonaws.com"
      },
      "Action": "lambda:InvokeFunction",
      "Resource": "arn:aws:lambda:us-east-1:123456789102:function:test"
    }
  ]
}
```

- Untuk detail API, lihat [GetPolicy](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-LMProvisionedConcurrencyConfig

Contoh kode berikut menunjukkan cara menggunakan `Get-LMProvisionedConcurrencyConfig`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan Konfigurasi Konkurensi yang disediakan untuk Alias yang ditentukan dari Fungsi Lambda.

```
C:\>Get-LMProvisionedConcurrencyConfig -FunctionName "MylambdaFunction123" -
Qualifier "NewAlias1"
```

Output:

```
AllocatedProvisionedConcurrentExecutions : 0
AvailableProvisionedConcurrentExecutions : 0
LastModified                             : 2020-01-15T03:21:26+0000
RequestedProvisionedConcurrentExecutions : 70
Status                                    : IN_PROGRESS
StatusReason                             :
```

- Untuk detail API, lihat [GetProvisionedConcurrencyConfig](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-LMProvisionedConcurrencyConfigList

Contoh kode berikut menunjukkan cara menggunakan `Get-LMProvisionedConcurrencyConfigList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil daftar konfigurasi konkurensi yang disediakan untuk fungsi Lambda.

```
Get-LMProvisionedConcurrencyConfigList -FunctionName "MylambdaFunction123"
```

- Untuk detail API, lihat [ListProvisionedConcurrencyConfigs](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-LMResourceTag

Contoh kode berikut menunjukkan cara menggunakan `Get-LMResourceTag`.

Alat untuk PowerShell V4

Contoh 1: Mengambil tag dan nilainya saat ini ditetapkan pada fungsi yang ditentukan.

```
Get-LMResourceTag -Resource "arn:aws:lambda:us-west-2:123456789012:function:MyFunction"
```

Output:

Key	Value
---	-----
California	Sacramento
Oregon	Salem
Washington	Olympia

- Untuk detail API, lihat [ListTags](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-LMVersionsByFunction

Contoh kode berikut menunjukkan cara menggunakan `Get-LMVersionsByFunction`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan daftar konfigurasi spesifik versi untuk setiap versi Fungsi Lambda.

```
Get-LMVersionsByFunction -FunctionName "MylambdaFunction123"
```

Output:

FunctionName	Runtime	MemorySize	Timeout	CodeSize	LastModified
RoleName					
-----	-----	-----	-----	-----	-----

MylambdaFunction123	python3.8	128	600	659	
2020-01-10T03:20:56.390+0000	lambda				
MylambdaFunction123	python3.8	128	5	1426	
2019-12-25T09:19:02.238+0000	lambda				
MylambdaFunction123	python3.8	128	5	1426	
2019-12-25T09:39:36.779+0000	lambda				

```
MyLambdaFunction123 python3.8      128      600      1426
2019-12-25T09:52:59.872+0000 lambda
```

- Untuk detail API, lihat [ListVersionsByFunction](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-LMAlias

Contoh kode berikut menunjukkan cara menggunakan `New-LMAlias`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat Alias Lambda Baru untuk versi tertentu dan konfigurasi perutean untuk menentukan persentase permintaan pemanggilan yang diterimanya.

```
New-LMAlias -FunctionName "MyLambdaFunction123" -
RoutingConfig_AdditionalVersionWeight @{Name="1";Value="0.6"} -Description "Alias for
version 4" -FunctionVersion 4 -Name "PowershellAlias"
```

- Untuk detail API, lihat [CreateAlias](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Publish-LMFunction

Contoh kode berikut menunjukkan cara menggunakan `Publish-LMFunction`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat fungsi C# (dotnetcore1.0 runtime) baru bernama `MyFunction` AWS Lambda, menyediakan binari yang dikompilasi untuk fungsi dari file zip pada sistem file lokal (jalur relatif atau absolut dapat digunakan). Fungsi C# Lambda menentukan handler untuk fungsi menggunakan penunjukan: `:Namespace.AssemblyName.ClassName::MethodName`. Anda harus mengganti nama assembly (tanpa akhiran `.dll`), namespace, nama kelas dan bagian nama metode dari spesifikasi handler dengan tepat. Fungsi baru akan memiliki variabel lingkungan `'envvar1'` dan `'envvar2'` yang diatur dari nilai yang disediakan.

```
Publish-LMFunction -Description "My C# Lambda Function" `
-FunctionName MyFunction `
-ZipFilename .\MyFunctionBinaries.zip `
-Handler "AssemblyName::Namespace.ClassName::MethodName" `
-Role "arn:aws:iam::123456789012:role/LambdaFullExecRole" `
```

```
-Runtime dotnetcore1.0 `
-Environment_Variable @{ "envvar1"="value";"envvar2"="value" }
```

Output:

```
CodeSha256      : /NgBMd...gq71I=
CodeSize        : 214784
DeadLetterConfig :
Description      : My C# Lambda Function
Environment      : Amazon.Lambda.Model.EnvironmentResponse
FunctionArn      : arn:aws:lambda:us-west-2:123456789012:function:ToUpper
FunctionName     : MyFunction
Handler         : AssemblyName::Namespace.ClassName::MethodName
KMSKeyArn        :
LastModified     : 2016-12-29T23:50:14.207+0000
MemorySize       : 128
Role            : arn:aws:iam::123456789012:role/LambdaFullExecRole
Runtime         : dotnetcore1.0
Timeout         : 3
Version         : $LATEST
VpcConfig        :
```

Contoh 2: Contoh ini mirip dengan yang sebelumnya kecuali binari fungsi pertama kali diunggah ke bucket Amazon S3 (yang harus berada di wilayah yang sama dengan fungsi Lambda yang dimaksud) dan objek S3 yang dihasilkan kemudian direferensikan saat membuat fungsi.

```
Write-S3Object -BucketName amzn-s3-demo-bucket -Key MyFunctionBinaries.zip -File .
\MyFunctionBinaries.zip
Publish-LMFunction -Description "My C# Lambda Function" `
    -FunctionName MyFunction `
    -BucketName amzn-s3-demo-bucket `
    -Key MyFunctionBinaries.zip `
    -Handler "AssemblyName::Namespace.ClassName::MethodName" `
    -Role "arn:aws:iam::123456789012:role/LambdaFullExecRole" `
    -Runtime dotnetcore1.0 `
    -Environment_Variable @{ "envvar1"="value";"envvar2"="value" }
```

- Untuk detail API, lihat [CreateFunction](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Publish-LMVersion

Contoh kode berikut menunjukkan cara menggunakan Publish-LMVersion.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat versi untuk snapshot Kode Fungsi Lambda yang ada

```
Publish-LMVersion -FunctionName "MylambdaFunction123" -Description "Publishing Existing Snapshot of function code as a new version through Powershell"
```

- Untuk detail API, lihat [PublishVersion](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-LMAlias

Contoh kode berikut menunjukkan cara menggunakan `Remove-LMAlias`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus fungsi Lambda Alias yang disebutkan dalam perintah.

```
Remove-LMAlias -FunctionName "MylambdaFunction123" -Name "NewAlias"
```

- Untuk detail API, lihat [DeleteAlias](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-LMFunction

Contoh kode berikut menunjukkan cara menggunakan `Remove-LMFunction`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus versi tertentu dari fungsi Lambda

```
Remove-LMFunction -FunctionName "MylambdaFunction123" -Qualifier '3'
```

- Untuk detail API, lihat [DeleteFunction](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-LMFunctionConcurrency

Contoh kode berikut menunjukkan cara menggunakan `Remove-LMFunctionConcurrency`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus Function Concurrency dari Fungsi Lambda.

```
Remove-LMFunctionConcurrency -FunctionName "MylambdaFunction123"
```

- Untuk detail API, lihat [DeleteFunctionConcurrency](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-LMPermission

Contoh kode berikut menunjukkan cara menggunakan `Remove-LMPermission`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus kebijakan fungsi untuk yang ditentukan `StatementId` dari Fungsi Lambda.

```
$policy = Get-LMPolicy -FunctionName "MylambdaFunction123" -Select Policy |  
ConvertFrom-Json | Select-Object -ExpandProperty Statement  
Remove-LMPermission -FunctionName "MylambdaFunction123" -StatementId $policy[0].Sid
```

- Untuk detail API, lihat [RemovePermission](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-LMProvisionedConcurrencyConfig

Contoh kode berikut menunjukkan cara menggunakan `Remove-LMProvisionedConcurrencyConfig`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus Konfigurasi Konkurensi yang Disediakan untuk Alias tertentu.

```
Remove-LMProvisionedConcurrencyConfig -FunctionName "MylambdaFunction123" -Qualifier  
"NewAlias1"
```

- Untuk detail API, lihat [DeleteProvisionedConcurrencyConfig](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-LMResourceTag

Contoh kode berikut menunjukkan cara menggunakan `Remove-LMResourceTag`.

Alat untuk PowerShell V4

Contoh 1: Menghapus tag yang disediakan dari fungsi. Cmdlet akan meminta konfirmasi sebelum melanjutkan kecuali sakelar -Force ditentukan. Satu panggilan dilakukan ke layanan untuk menghapus tag.

```
Remove-LMResourceTag -Resource "arn:aws:lambda:us-west-2:123456789012:function:MyFunction" -TagKey "Washington","Oregon","California"
```

Contoh 2: Menghapus tag yang disediakan dari fungsi. Cmdlet akan meminta konfirmasi sebelum melanjutkan kecuali sakelar -Force ditentukan. Setelah panggilan ke layanan dilakukan per tag yang disediakan.

```
"Washington","Oregon","California" | Remove-LMResourceTag -Resource "arn:aws:lambda:us-west-2:123456789012:function:MyFunction"
```

- Untuk detail API, lihat [UntagResource](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-LMAlias

Contoh kode berikut menunjukkan cara menggunakan Update-LMAlias.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memperbarui Konfigurasi Alias fungsi Lambda yang ada. Ini memperbarui RoutingConfiguration nilai untuk menggeser 60% (0,6) lalu lintas ke versi 1

```
Update-LMAlias -FunctionName "MylambdaFunction123" -Description " Alias for version 2" -FunctionVersion 2 -Name "newlabel1" -RoutingConfig_AdditionalVersionWeight @{"Name="1";Value="0.6"}
```

- Untuk detail API, lihat [UpdateAlias](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-LMFunctionCode

Contoh kode berikut menunjukkan cara menggunakan Update-LMFunctionCode.

Alat untuk PowerShell V4

Contoh 1: Memperbarui fungsi bernama 'MyFunction' dengan konten baru yang terkandung dalam file zip yang ditentukan. Untuk fungsi C# .NET Core Lambda, file zip harus berisi rakitan yang dikompilasi.

```
Update-LMFunctionCode -FunctionName MyFunction -ZipFilename .\UpdatedCode.zip
```

Contoh 2: Contoh ini mirip dengan yang sebelumnya tetapi menggunakan objek Amazon S3 yang berisi kode yang diperbarui untuk memperbarui fungsi.

```
Update-LMFunctionCode -FunctionName MyFunction -BucketName amzn-s3-demo-bucket -Key UpdatedCode.zip
```

- Untuk detail API, lihat [UpdateFunctionCode](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-LMFunctionConfiguration

Contoh kode berikut menunjukkan cara menggunakan `Update-LMFunctionConfiguration`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memperbarui Konfigurasi Fungsi Lambda yang ada

```
Update-LMFunctionConfiguration -FunctionName "MylambdaFunction123" -Handler "lambda_function.launch_instance" -Timeout 600 -Environment_Variable @{ "envvar1"="value";"envvar2"="value" } -Role arn:aws:iam::123456789101:role/service-role/lambda -DeadLetterConfig_TargetArn arn:aws:sns:us-east-1:123456789101:MyfirstTopic
```

- Untuk detail API, lihat [UpdateFunctionConfiguration](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Write-LMFunctionConcurrency

Contoh kode berikut menunjukkan cara menggunakan `Write-LMFunctionConcurrency`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menerapkan pengaturan konkurensi untuk Fungsi secara keseluruhan.

```
Write-LMFunctionConcurrency -FunctionName "MyLambdaFunction123" -  
ReservedConcurrentExecution 100
```

- Untuk detail API, lihat [PutFunctionConcurrency](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Write-LMProvisionedConcurrencyConfig

Contoh kode berikut menunjukkan cara menggunakan `Write-LMProvisionedConcurrencyConfig`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menambahkan konfigurasi konkurensi yang disediakan ke Alias Fungsi

```
Write-LMProvisionedConcurrencyConfig -FunctionName "MyLambdaFunction123" -  
ProvisionedConcurrentExecution 20 -Qualifier "NewAlias1"
```

- Untuk detail API, lihat [PutProvisionedConcurrencyConfig](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Contoh Amazon ML menggunakan Alat untuk PowerShell V4

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan Amazon ML.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

Get-MLBatchPrediction

Contoh kode berikut menunjukkan cara menggunakan `Get-MLBatchPrediction`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan metadata rinci untuk prediksi batch dengan ID id.

```
Get-MLBatchPrediction -BatchPredictionId ID
```

- Untuk detail API, lihat [GetBatchPrediction](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-MLBatchPredictionList

Contoh kode berikut menunjukkan cara menggunakan `Get-MLBatchPredictionList`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan daftar semua BatchPredictions dan catatan data terkait yang cocok dengan kriteria pencarian yang diberikan dalam permintaan.

```
Get-MLBatchPredictionList
```

Contoh 2: Mengembalikan daftar semua BatchPredictions dengan status SELESAI.

```
Get-MLBatchPredictionList -FilterVariable Status -EQ COMPLETED
```

- Untuk detail API, lihat [DescribeBatchPredictions](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-MLDataSource

Contoh kode berikut menunjukkan cara menggunakan `Get-MLDataSource`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan metadata, status, dan informasi file data untuk DataSource dengan ID id

```
Get-MLDataSource -DataSourceId ID
```

- Untuk detail API, lihat [GetDataSource](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-MLDataSourceList

Contoh kode berikut menunjukkan cara menggunakan `Get-MLDataSourceList`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan daftar semua DataSources dan catatan data terkait mereka.

```
Get-MLDataSourceList
```

Contoh 2: Mengembalikan daftar semua DataSources dengan status SELESAI.

```
Get-MLDataDourceList -FilterVariable Status -EQ COMPLETED
```

- Untuk detail API, lihat [DescribeDataSources](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-MLEvaluation

Contoh kode berikut menunjukkan cara menggunakan `Get-MLEvaluation`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan metadata dan status untuk Evaluasi dengan ID id.

```
Get-MLEvaluation -EvaluationId ID
```

- Untuk detail API, lihat [GetEvaluation](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-MLEvaluationList

Contoh kode berikut menunjukkan cara menggunakan `Get-MLEvaluationList`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan daftar semua sumber Evaluasi

```
Get-MLEvaluationList
```

Contoh 2: Mengembalikan daftar semua Evaluations dengan status COMPLETED.

```
Get-MLEvaluationList -FilterVariable Status -EQ COMPLETED
```

- Untuk detail API, lihat [DescribeEvaluations](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-MLModel

Contoh kode berikut menunjukkan cara menggunakan `Get-MLModel`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan metadata detail, status, skema, dan informasi file data untuk ID MLModel with id.

```
Get-MLModel -ModelId ID
```

- Untuk detail API, lihat [Dapatkan MLModel](#) Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-MLModelList

Contoh kode berikut menunjukkan cara menggunakan `Get-MLModelList`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan daftar semua Model dan catatan data yang terkait.

```
Get-MLModelList
```

Contoh 2: Mengembalikan daftar semua Model dengan status SELESAI.

```
Get-MLModelList -FilterVariable Status -EQ COMPLETED
```

- Untuk detail API, lihat [Menjelaskan MLModels](#) dalam Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-MLPrediction

Contoh kode berikut menunjukkan cara menggunakan `Get-MLPrediction`.

Alat untuk PowerShell V4

Contoh 1: Kirim catatan ke URL titik akhir prediksi waktu nyata untuk Model dengan ID id.

```
Get-MLPrediction -ModelId ID -PredictEndpoint URL -Record @{"A" = "B"; "C" = "D";}
```

- Untuk detail API, lihat [Memprediksi](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-MLBatchPrediction

Contoh kode berikut menunjukkan cara menggunakan `New-MLBatchPrediction`.

Alat untuk PowerShell V4

Contoh 1: Buat permintaan prediksi batch baru untuk model dengan ID id dan letakkan output di lokasi S3 yang ditentukan.

```
New-MLBatchPrediction -ModelId ID -Name NAME -OutputURI s3://...
```

- Untuk detail API, lihat [CreateBatchPrediction](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-MLDataSourceFromS3

Contoh kode berikut menunjukkan cara menggunakan `New-MLDataSourceFromS3`.

Alat untuk PowerShell V4

Contoh 1: Buat sumber data dengan data untuk lokasi S3, dengan nama NAME dan skema SCHEMA.

```
New-MLDataSourceFromS3 -Name NAME -ComputeStatistics $true -DataSpec_DataLocationS3  
"s3://BUCKET/KEY" -DataSchema SCHEMA
```

- Untuk detail API, lihat [CreateDataSourceFromS3](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-MLEvaluation

Contoh kode berikut menunjukkan cara menggunakan `New-MLEvaluation`.

Alat untuk PowerShell V4

Contoh 1: Buat evaluasi untuk id sumber data dan id model yang diberikan

```
New-MLEvaluation -Name NAME -DataSourceId DSID -ModelId MID
```

- Untuk detail API, lihat [CreateEvaluation](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-MLModel

Contoh kode berikut menunjukkan cara menggunakan `New-MLModel`.

Alat untuk PowerShell V4

Contoh 1: Buat model baru dengan data pelatihan.

```
New-MLModel -Name NAME -ModelType BINARY -Parameter @{...} -TrainingDataSourceId ID
```

- Untuk detail API, lihat [Membuat MLModel](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-MLRealtimeEndpoint

Contoh kode berikut menunjukkan cara menggunakan `New-MLRealtimeEndpoint`.

Alat untuk PowerShell V4

Contoh 1: Buat titik akhir prediksi realtime baru untuk id model yang diberikan.

```
New-MLRealtimeEndpoint -ModelId ID
```

- Untuk detail API, lihat [CreateRealtimeEndpoint](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Contoh Macie menggunakan Alat untuk V4 PowerShell

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan Macie.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

Get-MAC2FindingList

Contoh kode berikut menunjukkan cara menggunakan Get-MAC2FindingList.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan daftar FindingIds untuk Temuan yang berisi deteksi data sensitif dengan tipe "CREDIT_CARD_NUMBER" atau "US_SOCIAL_SECURITY_NUMBER"

```
$criterionAddProperties = New-Object
    Amazon.Macie2.Model.CriterionAdditionalProperties

$criterionAddProperties.Eq = @(
    "CREDIT_CARD_NUMBER"
    "US_SOCIAL_SECURITY_NUMBER"
)

$FindingCriterion = @{
    'classificationDetails.result.sensitiveData.detections.type' =
        [Amazon.Macie2.Model.CriterionAdditionalProperties]$criterionAddProperties
}

Get-MAC2FindingList -FindingCriteria_Criterion $FindingCriterion -MaxResult 5
```

- Untuk detail API, lihat [ListFindings](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

AWS OpsWorks contoh menggunakan Alat untuk PowerShell V4

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan AWS OpsWorks.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

New-OPSDeployment

Contoh kode berikut menunjukkan cara menggunakan New-OPSDeployment.

Alat untuk PowerShell V4

Contoh 1: Perintah ini membuat penerapan aplikasi baru pada semua instance berbasis Linux dalam lapisan di Stacks. AWS OpsWorks Bahkan jika Anda menentukan ID lapisan, Anda harus menentukan ID tumpukan juga. Perintah memungkinkan penerapan memulai ulang instance jika diperlukan.

```
New-OPSDeployment -StackID "724z93zz-zz78-4zzz-8z9z-1290123zzz1z" -LayerId  
"511b99c5-ec78-4caa-8a9d-1440116ffd1b" -AppId "0f7a109c-bf68-4336-8cb9-  
d37fe0b8c61d" -Command_Name deploy -Command_Arg @{Name="allow_reboot";Value="true"}
```

Contoh 2: Perintah ini menyebarkan **appsetup** resep dari **phpapp** buku masak, dan **secbaseline** resep dari buku masak. **testcookbook** Target penerapan adalah satu contoh, tetapi ID tumpukan dan ID lapisan juga diperlukan. **allow_reboot** Atribut parameter Command_Arg disetel ke **true**, yang memungkinkan penerapan memulai ulang instance jika diperlukan.

```
$commandArgs = '{ "Name":"execute_recipes", "Args":{"recipes":
["phpapp::appsetup","testcookbook::secbaseline"]} } }'
New-OPSDeployment -StackID "724z93zz-zz78-4zzz-8z9z-1290123zzz1z"
-LayerId "511b99c5-ec78-4caa-8a9d-1440116ffd1b" -InstanceId
"d89a6118-0007-4ccf-a51e-59f844127021" -Command_Name $commandArgs -Command_Arg
@{Name="allow_reboot";Value="true"
```

- Untuk detail API, lihat [CreateDeployment](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Daftar Harga AWS contoh menggunakan Alat untuk PowerShell V4

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan Daftar Harga AWS.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

Get-PLSAttributeValue

Contoh kode berikut menunjukkan cara menggunakan `Get-PLSAttributeValue`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan nilai untuk atribut 'VolumeType' untuk Amazon EC2 di wilayah us-east-1.

```
Get-PLSAttributeValue -ServiceCode AmazonEC2 -AttributeName "volumeType" -region us-
east-1
```

Output:

```
Value
-----
Cold HDD
General Purpose
Magnetic
Provisioned IOPS
Throughput Optimized HDD
```

- Untuk detail API, lihat [GetAttributeValues](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-PLSProduct

Contoh kode berikut menunjukkan cara menggunakan `Get-PLSProduct`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan detail semua produk untuk Amazon EC2.

```
Get-PLSProduct -ServiceCode AmazonEC2 -Region us-east-1
```

Output:

```
{"product":{"productFamily":"Compute Instance","attributes":
{"enhancedNetworkingSupported":"Yes","memory":"30.5
GiB","dedicatedEbsThroughput":"800 Mbps","vcpu":"4","locationType":"AWS
Region","storage":"EBS only","instanceFamily":"Memory
optimized","operatingSystem":"SUSE","physicalProcessor":"Intel Xeon E5-2686 v4
(Broadwell)","clockSpeed":"2.3 GHz","ecu":"Variable","networkPerformance":"Up
to 10 Gigabit","servicename":"Amazon Elastic Compute
Cloud","instanceType":"r4.xlarge","tenancy":"Shared","usagetype":"USW2-
BoxUsage:r4.xlarge","normalizationSizeFactor":"8","processorFeatures":"Intel AVX,
Intel AVX2, Intel Turbo","servicecode":"AmazonEC2","licenseModel":"No License
required","currentGeneration":"Yes","preInstalledSw":"NA","location":"US West
(Oregon)","processorArchitecture":"64-bit","operation":"RunInstances:000g"},...
```

Contoh 2: Mengembalikan data untuk Amazon EC2 di wilayah us-east-1 yang difilter berdasarkan jenis volume 'Tujuan Umum' yang didukung SSD.

```
Get-PLSProduct -ServiceCode AmazonEC2 -Filter
@{Type="TERM_MATCH";Field="volumeType";Value="General
```

```
Purpose"},@{Type="TERM_MATCH";Field="storageMedia";Value="SSD-backed"} -Region us-east-1
```

Output:

```
{"product":{"productFamily":"Storage","attributes":{"storageMedia":"SSD-backed","maxThroughputvolume":"160 MB/sec","volumeType":"General Purpose","maxIopsvolume":"10000"},...
```

- Untuk detail API, lihat [GetProducts](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-PLSService

Contoh kode berikut menunjukkan cara menggunakan `Get-PLSService`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan metadata untuk semua kode layanan yang tersedia di wilayah us-east-1.

```
Get-PLSService -Region us-east-1
```

Output:

AttributeNames	ServiceCode
-----	-----
{productFamily, servicecode, groupDescription, termType...}	AWSBudgets
{productFamily, servicecode, termType, usagetype...}	AWSCloudTrail
{productFamily, servicecode, termType, usagetype...}	AWSCodeCommit
{productFamily, servicecode, termType, usagetype...}	AWSCodeDeploy
{productFamily, servicecode, termType, usagetype...}	AWSCodePipeline
{productFamily, servicecode, termType, usagetype...}	AWSConfig
...	

Contoh 2: Mengembalikan metadata untuk EC2 layanan Amazon di wilayah us-east-1.

```
Get-PLSService -ServiceCode AmazonEC2 -Region us-east-1
```

Output:

AttributeNames	ServiceCode
-----	-----
{volumeType, maxIopsvolume, instanceCapacity10xlarge, locationType...}	AmazonEC2

- Untuk detail API, lihat [DescribeServices](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Contoh Resource Groups menggunakan Alat untuk PowerShell V4

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan Resource Groups.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

Add-RGResourceTag

Contoh kode berikut menunjukkan cara menggunakan Add-RGResourceTag.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menambahkan kunci tag 'Instance' dengan nilai 'kotak kerja' ke grup sumber daya yang diberikan arn

```
Add-RGResourceTag -Tag @{Instances="workboxes"} -Arn arn:aws:resource-groups:eu-west-1:123456789012:group/workboxes
```

Output:

Arn	Tags
---	----

```
arn:aws:resource-groups:eu-west-1:123456789012:group/workboxes {[Instances,
workboxes]}
```

- Untuk detail API, lihat [Tag](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Find-RGResource

Contoh kode berikut menunjukkan cara menggunakan `Find-RGResource`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat tipe sumber daya `ResourceQuery` untuk Instance dengan filter tag dan menemukan sumber daya.

```
$query = [Amazon.ResourceGroups.Model.ResourceQuery]::new()
$query.Type = [Amazon.ResourceGroups.QueryType]::TAG_FILTERS_1_0
$query.Query = ConvertTo-Json -Compress -Depth 4 -InputObject @{
    ResourceTypeFilters = @('AWS::EC2::Instance')
    TagFilters = @( @{
        Key = 'auto'
        Values = @('no')
    })
}

Find-RGResource -ResourceQuery $query | Select-Object -ExpandProperty
ResourceIdentifiers
```

Output:

ResourceArn	ResourceType
-----	-----
arn:aws:ec2:eu-west-1:123456789012:instance/i-0123445b6cb7bd67b	AWS::EC2::Instance

- Untuk detail API, lihat [SearchResources](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-RGGroup

Contoh kode berikut menunjukkan cara menggunakan `Get-RGGroup`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil grup sumber daya sesuai nama grup

```
Get-RGGroup -GroupName auto-no
```

Output:

Description	GroupArn	Name
-----	-----	----
	arn:aws:resource-groups:eu-west-1:123456789012:group/auto-no	auto-no

- Untuk detail API, lihat [GetGroup](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-RGGroupList

Contoh kode berikut menunjukkan cara menggunakan `Get-RGGroupList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan grup sumber daya yang sudah dibuat.

```
Get-RGGroupList
```

Output:

GroupArn	GroupName
-----	-----
arn:aws:resource-groups:eu-west-1:123456789012:group/auto-no	auto-no
arn:aws:resource-groups:eu-west-1:123456789012:group/auto-yes	auto-yes
arn:aws:resource-groups:eu-west-1:123456789012:group/build600	build600

- Untuk detail API, lihat [ListGroups](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-RGGroupQuery

Contoh kode berikut menunjukkan cara menggunakan `Get-RGGroupQuery`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil kueri sumber daya untuk grup sumber daya yang diberikan

```
Get-RGGroupQuery -GroupName auto-no | Select-Object -ExpandProperty ResourceQuery
```

Output:

```

Query
-----
Type
-----
{"ResourceTypeFilters":["AWS::EC2::Instance"],"TagFilters":[{"Key":"auto","Values":["no"]}]} TAG_FILTERS_1_0

```

- Untuk detail API, lihat [GetGroupQuery](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-RGGroupResourceList

Contoh kode berikut menunjukkan cara menggunakan `Get-RGGroupResourceList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan sumber daya grup berdasarkan disaring berdasarkan jenis sumber daya

```

Get-RGGroupResourceList -Filter @{"Name="resource-type";Values="AWS::EC2::Instance"}
-GroupName auto-yes | Select-Object -ExpandProperty ResourceIdentifiers

```

Output:

```

ResourceArn
-----
arn:aws:ec2:eu-west-1:123456789012:instance/i-0123bc45b567890e1 AWS::EC2::Instance
arn:aws:ec2:eu-west-1:123456789012:instance/i-0a1caf2345f67d8dc AWS::EC2::Instance
arn:aws:ec2:eu-west-1:123456789012:instance/i-012e3cb4df567e8aa AWS::EC2::Instance
arn:aws:ec2:eu-west-1:123456789012:instance/i-0fd12dd3456789012 AWS::EC2::Instance

```

- Untuk detail API, lihat [ListGroupResources](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-RGResourceTag

Contoh kode berikut menunjukkan cara menggunakan `Get-RGResourceTag`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan tag untuk kelompok sumber daya yang diberikan arn

```
Get-RGResourceTag -Arn arn:aws:resource-groups:eu-west-1:123456789012:group/workboxes
```

Output:

Key	Value
---	-----
Instances	workboxes

- Untuk detail API, lihat [GetTags](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-RGGroup

Contoh kode berikut menunjukkan cara menggunakan New-RGGroup.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat grup AWS sumber daya Resource Groups berbasis tag baru bernama TestPowerShellGroup. Grup ini menyertakan EC2 instance Amazon di wilayah saat ini yang ditandai dengan kunci tag “Nama”, dan nilai tag “test2”. Perintah mengembalikan query dan jenis grup, dan hasil operasi.

```
$ResourceQuery = New-Object -TypeName Amazon.ResourceGroups.Model.ResourceQuery
$ResourceQuery.Type = "TAG_FILTERS_1_0"
$ResourceQuery.Query = '{"ResourceTypeFilters":["AWS::EC2::Instance"],"TagFilters":
[{"Key":"Name","Values":["test2"]}]}
$ResourceQuery

New-RGGroup -Name TestPowerShellGroup -ResourceQuery $ResourceQuery -Description
"Test resource group."
```

Output:

Query	Type
-------	------

```

-----
      -----
{"ResourceTypeFilters":["AWS::EC2::Instance"],"TagFilters":[{"Key":"Name","Values":
["test2"]}]} TAG_FILTERS_1_0

LoggedAt      : 11/20/2018 2:40:59 PM
Group         : Amazon.ResourceGroups.Model.Group
ResourceQuery : Amazon.ResourceGroups.Model.ResourceQuery
Tags          : {}
ResponseMetadata : Amazon.Runtime.ResponseMetadata
ContentLength  : 338
HttpStatusCode : OK

```

- Untuk detail API, lihat [CreateGroup](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-RGGroup

Contoh kode berikut menunjukkan cara menggunakan `Remove-RGGroup`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus grup sumber daya bernama

```
Remove-RGGroup -GroupName non-tag-cfn-elbv2
```

Output:

```

Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-RGGroup (DeleteGroup)" on target "non-tag-cfn-
elbv2".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"): Y

Description GroupArn
Name
-----
-----
arn:aws:resource-groups:eu-west-1:123456789012:group/non-tag-cfn-elbv2
non-tag-cfn-elbv2

```

- Untuk detail API, lihat [DeleteGroup](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-RGResourceTag

Contoh kode berikut menunjukkan cara menggunakan `Remove-RGResourceTag`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus tag yang disebutkan dari grup sumber daya

```
Remove-RGResourceTag -Arn arn:aws:resource-groups:eu-west-1:123456789012:group/workboxes -Key Instances
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-RGResourceTag (Untag)" on target "arn:aws:resource-groups:eu-west-1:933303704102:group/workboxes".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is "Y"): Y

Arn                                                    Keys
---                                                    ----
arn:aws:resource-groups:eu-west-1:123456789012:group/workboxes {Instances}
```

- Untuk detail API, lihat [Untag](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-RGGroup

Contoh kode berikut menunjukkan cara menggunakan `Update-RGGroup`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memperbarui deskripsi grup

```
Update-RGGroup -GroupName auto-yes -Description "Instances auto-remove"
```

Output:

```
Description      GroupArn
-----
Name
-----
----
```

```
Instances to be cleaned arn:aws:resource-groups:eu-west-1:123456789012:group/auto-yes auto-yes
```

- Untuk detail API, lihat [UpdateGroup](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-RGGroupQuery

Contoh kode berikut menunjukkan cara menggunakan `Update-RGGroupQuery`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat objek query dan memperbarui query untuk grup.

```
$query = [Amazon.ResourceGroups.Model.ResourceQuery]::new()
$query.Type = [Amazon.ResourceGroups.QueryType]::TAG_FILTERS_1_0
$query.Query = @{
    ResourceTypeFilters = @('AWS::EC2::Instance')
    TagFilters = @( @{
        Key='Environment'
        Values='Build600.11'
    })
} | ConvertTo-Json -Compress -Depth 4

Update-RGGroupQuery -GroupName build600 -ResourceQuery $query
```

Output:

```
GroupName ResourceQuery
-----
build600  Amazon.ResourceGroups.Model.ResourceQuery
```

- Untuk detail API, lihat [UpdateGroupQuery](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Contoh API Penandaan Resource Groups menggunakan Alat untuk V4 PowerShell

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan Resource Groups Tagging API.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

Add-RGTResourceTag

Contoh kode berikut menunjukkan cara menggunakan Add-RGTResourceTag.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menambahkan kunci tag “stage” dan “version” dengan nilai “beta” dan “preprod_test” ke bucket Amazon S3 dan tabel Amazon DynamoDB. Satu panggilan dilakukan ke layanan untuk menerapkan tag.

```
$arn1 = "arn:aws:s3:::amzn-s3-demo-bucket"
$arn2 = "arn:aws:dynamodb:us-west-2:123456789012:table/mytable"

Add-RGTResourceTag -ResourceARNList $arn1,$arn2 -Tag @{ "stage"="beta";
"version"="preprod_test" }
```

Contoh 2: Contoh ini menambahkan tag dan nilai yang ditentukan ke bucket Amazon S3 dan tabel Amazon DynamoDB. Dua panggilan dilakukan ke layanan, satu untuk setiap sumber daya ARN disalurkan ke cmdlet.

```
$arn1 = "arn:aws:s3:::amzn-s3-demo-bucket"
$arn2 = "arn:aws:dynamodb:us-west-2:123456789012:table/mytable"

$arn1,$arn2 | Add-RGTResourceTag -Tag @{ "stage"="beta"; "version"="preprod_test" }
```

- Untuk detail API, lihat [TagResources](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-RGTResource

Contoh kode berikut menunjukkan cara menggunakan `Get-RGTResource`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan semua sumber daya yang ditandai di wilayah dan kunci tag yang terkait dengan sumber daya. Jika tidak ada parameter `-Region` yang diberikan ke cmdlet, ia akan mencoba menyimpulkan wilayah dari shell atau metadata instance. EC2

```
Get-RGTResource
```

Output:

ResourceARN	Tags
-----	----
arn:aws:dynamodb:us-west-2:123456789012:table/mytable	{stage, version}
arn:aws:s3:::mybucket	{stage, version,
othertag}	

Contoh 2: Mengembalikan semua sumber daya yang ditandai dari jenis yang ditentukan di suatu wilayah. String untuk setiap nama layanan dan jenis sumber daya sama dengan yang disematkan di Amazon Resource Name (ARN) sumber daya.

```
Get-RGTResource -ResourceType "s3"
```

Output:

ResourceARN	Tags
-----	----
arn:aws:s3:::mybucket	{stage, version,
othertag}	

Contoh 3: Mengembalikan semua sumber daya yang ditandai dari jenis tertentu di wilayah. Perhatikan bahwa ketika jenis sumber daya disalurkan ke cmdlet, satu panggilan ke layanan dibuat untuk setiap jenis sumber daya yang disediakan.

```
"dynamodb", "s3" | Get-RGTResource
```

Output:

ResourceARN	Tags
-----	----
arn:aws:dynamodb:us-west-2:123456789012:table/mytable	{stage, version}
arn:aws:s3::mybucket	{stage, version,
othertag}	

Contoh 4: Mengembalikan semua sumber daya yang ditandai yang cocok dengan filter yang ditentukan.

```
Get-RGTRResource -TagFilter @{ Key="stage" }
```

Output:

ResourceARN	Tags
-----	----
arn:aws:s3::mybucket	{stage, version,
othertag}	

Contoh 5: Mengembalikan semua sumber daya yang ditandai yang cocok dengan filter dan jenis sumber daya yang ditentukan.

```
Get-RGTRResource -TagFilter @{ Key="stage" } -ResourceType "dynamodb"
```

Output:

ResourceARN	Tags
-----	----
arn:aws:dynamodb:us-west-2:123456789012:table/mytable	{stage, version}

Contoh 6: Mengembalikan semua sumber daya yang ditandai yang cocok dengan filter yang ditentukan.

```
Get-RGTRResource -TagFilter @{ Key="stage"; Values=@("beta","gamma") }
```

Output:

ResourceARN	Tags
-----	----
arn:aws:dynamodb:us-west-2:123456789012:table/mytable	{stage, version}

- Untuk detail API, lihat [GetResources](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-RGTTagKey

Contoh kode berikut menunjukkan cara menggunakan `Get-RGTTagKey`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan semua kunci tag di wilayah tertentu. Jika parameter `-Region` tidak ditentukan, cmdlet akan mencoba menyimpulkan wilayah dari wilayah shell default atau metadata instance. EC2 Perhatikan bahwa kunci tag tidak dikembalikan dalam urutan tertentu.

```
Get-RGTTagKey -region us-west-2
```

Output:

```
version  
stage
```

- Untuk detail API, lihat [GetTagKeys](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-RGTTagValue

Contoh kode berikut menunjukkan cara menggunakan `Get-RGTTagValue`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan nilai untuk tag tertentu di wilayah. Jika parameter `-Region` tidak ditentukan, cmdlet akan mencoba menyimpulkan wilayah dari wilayah shell default atau metadata instance. EC2

```
Get-RGTTagValue -Key "stage" -Region us-west-2
```

Output:

```
beta
```

- Untuk detail API, lihat [GetTagValues](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-RGTResourceTag

Contoh kode berikut menunjukkan cara menggunakan `Remove-RGTResourceTag`.

Alat untuk PowerShell V4

Contoh 1: Menghapus kunci tag “stage” dan “version”, dan nilai terkait, dari bucket Amazon S3 dan tabel Amazon DynamoDB. Satu panggilan dilakukan ke layanan untuk menghapus tag. Sebelum tag dihapus, cmdlet akan meminta konfirmasi. Untuk melewati konfirmasi tambahkan parameter `-Force`.

```
$arn1 = "arn:aws:s3:::amzn-s3-demo-bucket"
$arn2 = "arn:aws:dynamodb:us-west-2:123456789012:table/mytable"

Remove-RGTResourceTag -ResourceARNList $arn1,$arn2 -TagKey "stage","version"
```

Contoh 2: Menghapus kunci tag “stage” dan “version”, dan nilai terkait, dari bucket Amazon S3 dan tabel Amazon DynamoDB. Dua panggilan dilakukan ke layanan, satu untuk setiap sumber daya ARN disalurkan ke cmdlet. Sebelum setiap panggilan, cmdlet akan meminta konfirmasi. Untuk melewati konfirmasi tambahkan parameter `-Force`.

```
$arn1 = "arn:aws:s3:::amzn-s3-demo-bucket"
$arn2 = "arn:aws:dynamodb:us-west-2:123456789012:table/mytable"

$arn1,$arn2 | Remove-RGTResourceTag -TagKey "stage","version"
```

- Untuk detail API, lihat [UntagResources](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Contoh rute 53 menggunakan Alat untuk PowerShell V4

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan Route 53.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

Edit-R53ResourceRecordSet

Contoh kode berikut menunjukkan cara menggunakan `Edit-R53ResourceRecordSet`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat catatan A untuk `www.example.com` dan mengubah catatan A untuk `test.example.com` dari `192.0.2.3` menjadi `192.0.2.1`. Perhatikan bahwa nilai untuk perubahan catatan tipe TXT harus dalam tanda kutip ganda. Lihat dokumentasi Amazon Route 53 untuk detail selengkapnya. Anda dapat menggunakan `Get-R53Change` cmdlet untuk melakukan polling untuk menentukan kapan perubahan selesai.

```
$change1 = New-Object Amazon.Route53.Model.Change
$change1.Action = "CREATE"
$change1.ResourceRecordSet = New-Object Amazon.Route53.Model.ResourceRecordSet
$change1.ResourceRecordSet.Name = "www.example.com"
$change1.ResourceRecordSet.Type = "TXT"
$change1.ResourceRecordSet.TTL = 600
$change1.ResourceRecordSet.ResourceRecords.Add(@{Value="item 1 item 2 item 3"})

$change2 = New-Object Amazon.Route53.Model.Change
$change2.Action = "DELETE"
$change2.ResourceRecordSet = New-Object Amazon.Route53.Model.ResourceRecordSet
$change2.ResourceRecordSet.Name = "test.example.com"
$change2.ResourceRecordSet.Type = "A"
$change2.ResourceRecordSet.TTL = 600
$change2.ResourceRecordSet.ResourceRecords.Add(@{Value="192.0.2.3"})

$change3 = New-Object Amazon.Route53.Model.Change
$change3.Action = "CREATE"
$change3.ResourceRecordSet = New-Object Amazon.Route53.Model.ResourceRecordSet
$change3.ResourceRecordSet.Name = "test.example.com"
$change3.ResourceRecordSet.Type = "A"
$change3.ResourceRecordSet.TTL = 600
$change3.ResourceRecordSet.ResourceRecords.Add(@{Value="192.0.2.1"})
```

```
$params = @{
    HostedZoneId="Z1PA6795UKMFR9"
    ChangeBatch_Comment="This change batch creates a TXT record for www.example.com.
    and changes the A record for test.example.com. from 192.0.2.3 to 192.0.2.1."
    ChangeBatch_Change=$change1,$change2,$change3
}

Edit-R53ResourceRecordSet @params
```

Contoh 2: Contoh ini menunjukkan cara membuat kumpulan catatan sumber daya alias. 'Z222222222' adalah ID dari zona yang dihosting Amazon Route 53 tempat Anda membuat kumpulan catatan sumber daya alias. 'example.com' adalah puncak zona tempat Anda ingin membuat alias dan 'www.example.com' adalah subdomain yang Anda juga ingin membuat alias. 'Z1111111111' adalah contoh ID zona yang dihosting untuk penyeimbang beban dan 'example-load-balancer-1111111111.us-east-1.elb.amazonaws.com' adalah contoh nama domain penyeimbang beban yang digunakan Amazon Route 53 untuk menanggapi kueri untuk example.com dan www.example.com. Lihat dokumentasi Amazon Route 53 untuk detail selengkapnya. Anda dapat menggunakan Get-R53Change cmdlet untuk melakukan polling untuk menentukan kapan perubahan selesai.

```
$change1 = New-Object Amazon.Route53.Model.Change
$change1.Action = "CREATE"
$change1.ResourceRecordSet = New-Object Amazon.Route53.Model.ResourceRecordSet
$change1.ResourceRecordSet.Name = "example.com"
$change1.ResourceRecordSet.Type = "A"
$change1.ResourceRecordSet.AliasTarget = New-Object Amazon.Route53.Model.AliasTarget
$change1.ResourceRecordSet.AliasTarget.HostedZoneId = "Z111111111111"
$change1.ResourceRecordSet.AliasTarget.DNSName = "example-load-
balancer-1111111111.us-east-1.elb.amazonaws.com."
$change1.ResourceRecordSet.AliasTarget.EvaluateTargetHealth = $true

$change2 = New-Object Amazon.Route53.Model.Change
$change2.Action = "CREATE"
$change2.ResourceRecordSet = New-Object Amazon.Route53.Model.ResourceRecordSet
$change1.ResourceRecordSet.Name = "www.example.com"
$change1.ResourceRecordSet.Type = "A"
$change1.ResourceRecordSet.AliasTarget = New-Object Amazon.Route53.Model.AliasTarget
$change1.ResourceRecordSet.AliasTarget.HostedZoneId = "Z111111111111"
$change1.ResourceRecordSet.AliasTarget.DNSName = "example-load-
balancer-1111111111.us-east-1.elb.amazonaws.com."
$change1.ResourceRecordSet.AliasTarget.EvaluateTargetHealth = $false
```

```
$params = @{
    HostedZoneId="Z222222222"
    ChangeBatch_Comment="This change batch creates two alias resource record sets, one
    for the zone apex, example.com, and one for www.example.com, that both point to
    example-load-balancer-111111111.us-east-1.elb.amazonaws.com."
    ChangeBatch_Change=$change1,$change2
}

Edit-R53ResourceRecordSet @params
```

Contoh 3: Contoh ini membuat dua catatan A untuk `www.example.com`. Seperempat dari waktu ($1/(1+3)$), Amazon Route 53 menanggapi kueri untuk `www.example.com` dengan dua nilai untuk kumpulan catatan sumber daya pertama (`192.0.2.9` dan `192.0.2.10`). Tiga perempat waktu ($3/(1+3)$) Amazon Route 53 menanggapi kueri untuk `www.example.com` dengan dua nilai untuk kumpulan catatan sumber daya kedua (`192.0.2.11` dan `192.0.2.12`). Lihat dokumentasi Amazon Route 53 untuk detail selengkapnya. Anda dapat menggunakan `Get-R53Change` cmdlet untuk melakukan polling untuk menentukan kapan perubahan selesai.

```
$change1 = New-Object Amazon.Route53.Model.Change
$change1.Action = "CREATE"
$change1.ResourceRecordSet = New-Object Amazon.Route53.Model.ResourceRecordSet
$change1.ResourceRecordSet.Name = "www.example.com"
$change1.ResourceRecordSet.Type = "A"
$change1.ResourceRecordSet.SetIdentifier = "Rack 2, Positions 4 and 5"
$change1.ResourceRecordSet.Weight = 1
$change1.ResourceRecordSet.TTL = 600
$change1.ResourceRecordSet.ResourceRecords.Add(@{Value="192.0.2.9"})
$change1.ResourceRecordSet.ResourceRecords.Add(@{Value="192.0.2.10"})

$change2 = New-Object Amazon.Route53.Model.Change
$change2.Action = "CREATE"
$change2.ResourceRecordSet = New-Object Amazon.Route53.Model.ResourceRecordSet
$change2.ResourceRecordSet.Name = "www.example.com"
$change2.ResourceRecordSet.Type = "A"
$change2.ResourceRecordSet.SetIdentifier = "Rack 5, Positions 1 and 2"
$change2.ResourceRecordSet.Weight = 3
$change2.ResourceRecordSet.TTL = 600
$change2.ResourceRecordSet.ResourceRecords.Add(@{Value="192.0.2.11"})
$change2.ResourceRecordSet.ResourceRecords.Add(@{Value="192.0.2.12"})

$params = @{
    HostedZoneId="Z1PA6795UKMFR9"
```

```

    ChangeBatch_Comment="This change creates two weighted resource record sets, each
    of which has two values."
    ChangeBatch_Change=$change1,$change2
}

Edit-R53ResourceRecordSet @params

```

Contoh 4: Contoh ini menunjukkan cara membuat kumpulan catatan sumber daya alias tertimbang dengan asumsi bahwa example.com adalah domain yang ingin Anda buat kumpulan catatan sumber daya alias tertimbang. SetIdentifier membedakan dua set catatan sumber daya alias tertimbang satu sama lain. Elemen ini diperlukan karena elemen Nama dan Jenis memiliki nilai yang sama untuk kedua kumpulan catatan sumber daya. Z1111111111111 dan Z3333333333333333 adalah contoh zona host untuk penyeimbang beban ELB yang ditentukan oleh nilai. IDs DNSName example-load-balancer-222222222.us-east-1.elb.amazonaws.com dan example-load-balancer -4444444444.us-east-1.elb.amazonaws.com adalah contoh domain Elastic Load Balancing dari mana Amazon Route 53 menanggapi kueri untuk example.com. Lihat dokumentasi Amazon Route 53 untuk detail selengkapnya. Anda dapat menggunakan Get-R53Change cmdlet untuk melakukan polling untuk menentukan kapan perubahan selesai.

```

$change1 = New-Object Amazon.Route53.Model.Change
$change1.Action = "CREATE"
$change1.ResourceRecordSet = New-Object Amazon.Route53.Model.ResourceRecordSet
$change1.ResourceRecordSet.Name = "example.com"
$change1.ResourceRecordSet.Type = "A"
$change1.ResourceRecordSet.SetIdentifier = "1"
$change1.ResourceRecordSet.Weight = 3
$change1.ResourceRecordSet.AliasTarget = New-Object Amazon.Route53.Model.AliasTarget
$change1.ResourceRecordSet.AliasTarget.HostedZoneId = "Z1111111111111"
$change1.ResourceRecordSet.AliasTarget.DNSName = "example-load-
balancer-222222222.us-east-1.elb.amazonaws.com."
$change1.ResourceRecordSet.AliasTarget.EvaluateTargetHealth = $true

$change2 = New-Object Amazon.Route53.Model.Change
$change2.Action = "CREATE"
$change2.ResourceRecordSet = New-Object Amazon.Route53.Model.ResourceRecordSet
$change2.ResourceRecordSet.Name = "example.com"
$change2.ResourceRecordSet.Type = "A"
$change2.ResourceRecordSet.SetIdentifier = "2"
$change2.ResourceRecordSet.Weight = 1
$change2.ResourceRecordSet.AliasTarget = New-Object Amazon.Route53.Model.AliasTarget
$change2.ResourceRecordSet.AliasTarget.HostedZoneId = "Z3333333333333333"

```

```

$change2.ResourceRecordSet.AliasTarget.DNSName = "example-load-
balancer-4444444444.us-east-1.elb.amazonaws.com."
$change2.ResourceRecordSet.AliasTarget.EvaluateTargetHealth = $false

$params = @{
    HostedZoneId="Z5555555555"
    ChangeBatch_Comment="This change batch creates two weighted alias resource
record sets. Amazon Route 53 responds to queries for example.com with the first ELB
domain 3/4ths of the times and the second one 1/4th of the time."
    ChangeBatch_Change=$change1,$change2
}

Edit-R53ResourceRecordSet @params

```

Contoh 5: Contoh ini membuat dua set catatan sumber daya alias latensi, satu untuk penyeimbang beban ELB di wilayah AS Barat (Oregon) (us-west-2), dan satu lagi untuk penyeimbang beban di wilayah Asia Pasifik (Singapura) (ap-southeast-1). Lihat dokumentasi Amazon Route 53 untuk detail selengkapnya. Anda dapat menggunakan Get-R53Change cmdlet untuk melakukan polling untuk menentukan kapan perubahan selesai.

```

$change1 = New-Object Amazon.Route53.Model.Change
$change1.Action = "CREATE"
$change1.ResourceRecordSet = New-Object Amazon.Route53.Model.ResourceRecordSet
$change1.ResourceRecordSet.Name = "example.com"
$change1.ResourceRecordSet.Type = "A"
$change1.ResourceRecordSet.SetIdentifier = "Oregon load balancer 1"
$change1.ResourceRecordSet.Region = us-west-2
$change1.ResourceRecordSet.AliasTarget = New-Object Amazon.Route53.Model.AliasTarget
$change1.ResourceRecordSet.AliasTarget.HostedZoneId = "Z11111111111111"
$change1.ResourceRecordSet.AliasTarget.DNSName = "example-load-
balancer-2222222222.us-west-2.elb.amazonaws.com"
$change1.ResourceRecordSet.AliasTarget.EvaluateTargetHealth = $true

$change2 = New-Object Amazon.Route53.Model.Change
$change2.Action = "CREATE"
$change2.ResourceRecordSet = New-Object Amazon.Route53.Model.ResourceRecordSet
$change2.ResourceRecordSet.Name = "example.com"
$change2.ResourceRecordSet.Type = "A"
$change2.ResourceRecordSet.SetIdentifier = "Singapore load balancer 1"
$change2.ResourceRecordSet.Region = ap-southeast-1
$change2.ResourceRecordSet.AliasTarget = New-Object Amazon.Route53.Model.AliasTarget
$change2.ResourceRecordSet.AliasTarget.HostedZoneId = "Z22222222222222"

```

```
$change2.ResourceRecordSet.AliasTarget.DNSName = "example-load-
balancer-1111111111.ap-southeast-1.elb.amazonaws.com"
$change2.ResourceRecordSet.AliasTarget.EvaluateTargetHealth = $true

$params = @{
    HostedZoneId="Z5555555555"
    ChangeBatch_Comment="This change batch creates two latency resource record
sets, one for the US West (Oregon) region and one for the Asia Pacific (Singapore)
region."
    ChangeBatch_Change=$change1,$change2
}

Edit-R53ResourceRecordSet @params
```

- Untuk detail API, lihat [ChangeResourceRecordSets](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-R53AccountLimit

Contoh kode berikut menunjukkan cara menggunakan `Get-R53AccountLimit`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan jumlah maksimum zona yang dihosting yang dapat dibuat menggunakan akun saat ini.

```
Get-R53AccountLimit -Type MAX_HOSTED_ZONES_BY_OWNER
```

Output:

```
15
```

- Untuk detail API, lihat [GetAccountLimit](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-R53CheckerIpRanges

Contoh kode berikut menunjukkan cara menggunakan `Get-R53CheckerIpRanges`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan CIDRs untuk memeriksa kesehatan Route53

Get-R53CheckerIpRanges**Output:**

```
15.177.2.0/23
15.177.6.0/23
15.177.10.0/23
15.177.14.0/23
15.177.18.0/23
15.177.22.0/23
15.177.26.0/23
15.177.30.0/23
15.177.34.0/23
15.177.38.0/23
15.177.42.0/23
15.177.46.0/23
15.177.50.0/23
15.177.54.0/23
15.177.58.0/23
15.177.62.0/23
54.183.255.128/26
54.228.16.0/26
54.232.40.64/26
54.241.32.64/26
54.243.31.192/26
54.244.52.192/26
54.245.168.0/26
54.248.220.0/26
54.250.253.192/26
54.251.31.128/26
54.252.79.128/26
54.252.254.192/26
54.255.254.192/26
107.23.255.0/26
176.34.159.192/26
177.71.207.128/26
```

- Untuk detail API, lihat [GetCheckerIpRanges](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-R53HostedZone

Contoh kode berikut menunjukkan cara menggunakan `Get-R53HostedZone`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan rincian zona yang dihosting dengan ID PJJN98 FT9 Z1D633.

```
Get-R53HostedZone -Id Z1D633PJJN98FT9
```

- Untuk detail API, lihat [GetHostedZone](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-R53HostedZoneCount

Contoh kode berikut menunjukkan cara menggunakan `Get-R53HostedZoneCount`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan jumlah total zona host publik dan pribadi untuk saat ini Akun AWS.

```
Get-R53HostedZoneCount
```

- Untuk detail API, lihat [GetHostedZoneCount](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-R53HostedZoneLimit

Contoh kode berikut menunjukkan cara menggunakan `Get-R53HostedZoneLimit`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan batas jumlah maksimum catatan yang dapat dibuat di zona host yang ditentukan.

```
Get-R53HostedZoneLimit -HostedZoneId Z3MEQ8T7HAAAAF -Type MAX_RRSETS_BY_ZONE
```

Output:

```
5
```

- Untuk detail API, lihat [GetHostedZoneLimit](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-R53HostedZoneList

Contoh kode berikut menunjukkan cara menggunakan `Get-R53HostedZoneList`.

Alat untuk PowerShell V4

Contoh 1: Mengeluarkan semua zona host publik dan pribadi Anda.

```
Get-R53HostedZoneList
```

Contoh 2: Mengeluarkan semua zona yang dihosting yang terkait dengan kumpulan delegasi yang dapat digunakan kembali yang memiliki ID X2CISAMPLE NZ8

```
Get-R53HostedZoneList -DelegationSetId NZ8X2CISAMPLE
```

- Untuk detail API, lihat [ListHostedZones](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-R53HostedZonesByName

Contoh kode berikut menunjukkan cara menggunakan `Get-R53HostedZonesByName`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan semua zona host publik dan pribadi Anda dalam urutan ASCII berdasarkan nama domain.

```
Get-R53HostedZonesByName
```

Contoh 2: Mengembalikan zona host publik dan pribadi Anda, dalam urutan ASCII berdasarkan nama domain, dimulai dari nama DNS yang ditentukan.

```
Get-R53HostedZonesByName -DnsName example2.com
```

- Untuk detail API, lihat [ListHostedZonesByName](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-R53QueryLoggingConfigList

Contoh kode berikut menunjukkan cara menggunakan `Get-R53QueryLoggingConfigList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan semua konfigurasi untuk pencatatan kueri DNS yang terkait dengan arus. Akun AWS

```
Get-R53QueryLoggingConfigList
```

Output:

Id	HostedZoneId	CloudWatchLogsLogGroupArn
--	-----	-----
59b0fa33-4fea-4471-a88c-926476aaa40d	Z385PDS6EAAAZR	arn:aws:logs:us-east-1:111111111112:log-group:/aws/route53/example1.com:*
ee528e95-4e03-4fdc-9d28-9e24ddaaa063	Z94SJHBV1AAAAZ	arn:aws:logs:us-east-1:111111111112:log-group:/aws/route53/example2.com:*
e38dddda-ceb6-45c1-8cb7-f0ae56aaaa2b	Z3MEQ8T7AAA1BF	arn:aws:logs:us-east-1:111111111112:log-group:/aws/route53/example3.com:*

- Untuk detail API, lihat [ListQueryLoggingConfigs](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-R53ReusableDelegationSet

Contoh kode berikut menunjukkan cara menggunakan `Get-R53ReusableDelegationSet`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil informasi tentang set delegasi tertentu termasuk empat server nama yang ditugaskan ke set delegasi.

```
Get-R53ReusableDelegationSet -Id N23DS9X4AYEAAA
```

Output:

Id	CallerReference	NameServers
--	-----	-----

```
/delegationset/N23DS9X4AYEAAA testcaller {ns-545.awsdns-04.net,
ns-1264.awsdns-30.org, ns-2004.awsdns-58.co.uk, ns-240.awsdns-30.com}
```

- Untuk detail API, lihat [GetReusableDelegationSet](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-R53HostedZone

Contoh kode berikut menunjukkan cara menggunakan `New-R53HostedZone`.

Alat untuk PowerShell V4

Contoh 1: Membuat zona host baru bernama 'example.com', terkait dengan kumpulan delegasi yang dapat digunakan kembali. Perhatikan bahwa Anda harus memberikan nilai untuk `CallerReference` parameter sehingga permintaan yang perlu dicoba lagi jika perlu tanpa risiko mengeksekusi operasi dua kali. Karena zona yang dihosting sedang dibuat dalam VPC, zona ini secara otomatis bersifat pribadi dan Anda tidak boleh mengatur parameter - `HostedZoneConfig_PrivateZone`.

```
$params = @{
    Name="example.com"
    CallerReference="myUniqueIdentifier"
    HostedZoneConfig_Comment="This is my first hosted zone"
    DelegationSetId="NZ8X2CISAMPLE"
    VPC_VPCId="vpc-1a2b3c4d"
    VPC_VPCRegion="us-east-1"
}

New-R53HostedZone @params
```

- Untuk detail API, lihat [CreateHostedZone](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-R53QueryLoggingConfig

Contoh kode berikut menunjukkan cara menggunakan `New-R53QueryLoggingConfig`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat konfigurasi pencatatan kueri DNS Route53 baru untuk zona host yang ditentukan. Amazon Route53 akan mempublikasikan log kueri DNS ke grup log Cloudwatch yang ditentukan.

```
New-R53QueryLoggingConfig -HostedZoneId Z3MEQ8T7HAAAAF -CloudWatchLogsLogGroupArn
arn:aws:logs:us-east-1:111111111111:log-group:/aws/route53/example.com:*
```

Output:

QueryLoggingConfig	Location
-----	-----
Amazon.Route53.Model.QueryLoggingConfig	https://route53.amazonaws.com/2013-04-01/ queryloggingconfig/ee5aaa95-4e03-4fdc-9d28-9e24ddaaaaa3

- Untuk detail API, lihat [CreateQueryLoggingConfig](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-R53ReusableDelegationSet

Contoh kode berikut menunjukkan cara menggunakan `New-R53ReusableDelegationSet`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat kumpulan delegasi yang dapat digunakan kembali dari 4 server nama yang dapat dilanjutkan oleh beberapa zona yang dihosting.

```
New-R53ReusableDelegationSet -CallerReference testcallerreference
```

Output:

DelegationSet	Location
-----	-----
Amazon.Route53.Model.DelegationSet	https://route53.amazonaws.com/2013-04-01/ delegationset/N23DS9XAAAAAXM

- Untuk detail API, lihat [CreateReusableDelegationSet](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Register-R53VPCWithHostedZone

Contoh kode berikut menunjukkan cara menggunakan `Register-R53VPCWithHostedZone`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengaitkan VPC yang ditentukan dengan zona host pribadi.

```
Register-R53VPCWithHostedZone -HostedZoneId Z3MEQ8T7HAAAAF -VPC_VPCId vpc-f1b9aaaa -
VPC_VPCRegion us-east-1
```

Output:

Id	Status	SubmittedAt	Comment
--	-----	-----	-----
/change/C3SCAAA633Z6DX	PENDING	01/28/2020 19:32:02	

- Untuk detail API, lihat [Associate VPCWith HostedZone](#) in Alat AWS untuk PowerShell Cmdlet Reference (V4).

Remove-R53HostedZone

Contoh kode berikut menunjukkan cara menggunakan Remove-R53HostedZone.

Alat untuk PowerShell V4

Contoh 1: Menghapus zona yang dihosting dengan ID yang ditentukan. Anda akan diminta konfirmasi sebelum perintah dilanjutkan kecuali Anda menambahkan parameter sakelar -Force.

```
Remove-R53HostedZone -Id Z1PA6795UKMFR9
```

- Untuk detail API, lihat [DeleteHostedZone](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-R53QueryLoggingConfig

Contoh kode berikut menunjukkan cara menggunakan Remove-R53QueryLoggingConfig.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus konfigurasi yang ditentukan untuk pencatatan kueri DNS.

```
Remove-R53QueryLoggingConfig -Id ee528e95-4e03-4fdc-9d28-9e24daaa20063
```

- Untuk detail API, lihat [DeleteQueryLoggingConfig](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-R53ReusableDelegationSet

Contoh kode berikut menunjukkan cara menggunakan `Remove-R53ReusableDelegationSet`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus set delegasi yang dapat digunakan kembali yang ditentukan.

```
Remove-R53ReusableDelegationSet -Id N23DS9X4AYAAAM
```

- Untuk detail API, lihat [DeleteReusableDelegationSet](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Unregister-R53VPCFromHostedZone

Contoh kode berikut menunjukkan cara menggunakan `Unregister-R53VPCFromHostedZone`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memisahkan VPC yang ditentukan dari zona host pribadi.

```
Unregister-R53VPCFromHostedZone -HostedZoneId Z3MEQ8T7HAAAAF -VPC_VPCId vpc-f1b9aaaa  
-VPC_VPCRegion us-east-1
```

Output:

Id	Status	SubmittedAt	Comment
--	-----	-----	-----
/change/C2XFCAAAA9HKZG	PENDING	01/28/2020 10:35:55	

- Untuk detail API, lihat [Memisahkan VPCFrom HostedZone](#) dalam Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-R53HostedZoneComment

Contoh kode berikut menunjukkan cara menggunakan `Update-R53HostedZoneComment`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini memperbarui komentar untuk zona host yang ditentukan.

```
Update-R53HostedZoneComment -Id Z385PDS6AAAAAR -Comment "This is my first hosted zone"
```

Output:

```
Id                : /hostedzone/Z385PDS6AAAAAR
Name              : example.com.
CallerReference   : C5B55555-7147-EF04-8341-69131E805C89
Config           : Amazon.Route53.Model.HostedZoneConfig
ResourceRecordSetCount : 9
LinkedService     :
```

- Untuk detail API, lihat [UpdateHostedZoneComment](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Contoh Amazon S3 menggunakan Alat untuk V4 PowerShell

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan Amazon S3.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

Copy-S3object

Contoh kode berikut menunjukkan cara menggunakan `Copy-S3object`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini menyalin objek "sample.txt" dari bucket "test-files" ke bucket yang sama tetapi dengan kunci baru "sample-copy.txt".

```
Copy-S3Object -BucketName amzn-s3-demo-bucket -Key sample.txt -DestinationKey  
sample-copy.txt
```

Contoh 2: Perintah ini menyalin objek "sample.txt" dari bucket "test-files" ke bucket "backup files" dengan kunci "sample-copy.txt".

```
Copy-S3Object -BucketName amzn-s3-demo-source-bucket -Key sample.txt -DestinationKey  
sample-copy.txt -DestinationBucket amzn-s3-demo-destination-bucket
```

Contoh 3: Perintah ini mengunduh objek "sample.txt" dari bucket "test-files" ke file lokal dengan nama "local-sample.txt".

```
Copy-S3Object -BucketName amzn-s3-demo-bucket -Key sample.txt -LocalFile local-  
sample.txt
```

Contoh 4: Mengunduh objek tunggal ke file yang ditentukan. File yang diunduh akan ditemukan di c:\downloads\data\archive.zip

```
Copy-S3Object -BucketName amzn-s3-demo-bucket -Key data/archive.zip -LocalFolder c:  
\downloads
```

Contoh 5: Download semua objek yang cocok dengan key prefix yang ditentukan ke folder lokal. Hirarki kunci relatif akan dipertahankan sebagai subfolder di lokasi unduhan keseluruhan.

```
Copy-S3Object -BucketName amzn-s3-demo-bucket -KeyPrefix data -LocalFolder c:  
\downloads
```

- Untuk detail API, lihat [CopyObject](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-S3ACL

Contoh kode berikut menunjukkan cara menggunakan `Get-S3ACL`.

Alat untuk PowerShell V4

Contoh 1: Perintah mendapatkan rincian pemilik objek objek S3.

```
Get-S3ACL -BucketName 'amzn-s3-demo-bucket' -key 'initialize.ps1' -Select
AccessControlList.Owner
```

Output:

```
DisplayName Id
----- --
testusername 9988776a6554433d22f1100112e334acb45566778899009e9887bd7f66c5f544
```

- Untuk detail API, lihat [GetACL](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-S3Bucket

Contoh kode berikut menunjukkan cara menggunakan `Get-S3Bucket`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini mengembalikan semua bucket S3.

```
Get-S3Bucket
```

Contoh 2: Perintah ini mengembalikan bucket bernama “test-files”

```
Get-S3Bucket -BucketName amzn-s3-demo-bucket
```

- Untuk detail API, lihat [ListBuckets](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-S3BucketAccelerateConfiguration

Contoh kode berikut menunjukkan cara menggunakan `Get-S3BucketAccelerateConfiguration`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini mengembalikan nilai Diaktifkan, jika pengaturan akselerasi transfer diaktifkan untuk bucket yang ditentukan.

```
Get-S3BucketAccelerateConfiguration -BucketName 'amzn-s3-demo-bucket'
```

Output:

```
Value
-----
Enabled
```

- Untuk detail API, lihat [GetBucketAccelerateConfiguration](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-S3BucketAnalyticsConfiguration

Contoh kode berikut menunjukkan cara menggunakan `Get-S3BucketAnalyticsConfiguration`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini mengembalikan detail filter analitik dengan nama 'testfilter' di bucket S3 yang diberikan.

```
Get-S3BucketAnalyticsConfiguration -BucketName 'amzn-s3-demo-bucket' -AnalyticsId 'testfilter'
```

- Untuk detail API, lihat [GetBucketAnalyticsConfiguration](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-S3BucketAnalyticsConfigurationList

Contoh kode berikut menunjukkan cara menggunakan `Get-S3BucketAnalyticsConfigurationList`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini mengembalikan 100 konfigurasi analitik pertama dari bucket S3 yang diberikan.

```
Get-S3BucketAnalyticsConfigurationList -BucketName 'amzn-s3-demo-bucket'
```

- Untuk detail API, lihat [ListBucketAnalyticsConfigurations](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-S3BucketEncryption

Contoh kode berikut menunjukkan cara menggunakan `Get-S3BucketEncryption`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini mengembalikan semua aturan enkripsi sisi server yang terkait dengan bucket yang diberikan.

```
Get-S3BucketEncryption -BucketName 'amzn-s3-demo-bucket'
```

- Untuk detail API, lihat [GetBucketEncryption](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-S3BucketInventoryConfiguration

Contoh kode berikut menunjukkan cara menggunakan `Get-S3BucketInventoryConfiguration`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini mengembalikan detail inventaris bernama 'testinventory' untuk bucket S3 yang diberikan.

```
Get-S3BucketInventoryConfiguration -BucketName 'amzn-s3-demo-bucket' -InventoryId  
'testinventory'
```

- Untuk detail API, lihat [GetBucketInventoryConfiguration](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-S3BucketInventoryConfigurationList

Contoh kode berikut menunjukkan cara menggunakan `Get-S3BucketInventoryConfigurationList`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini mengembalikan 100 konfigurasi inventaris pertama dari bucket S3 yang diberikan.

```
Get-S3BucketInventoryConfigurationList -BucketName 'amzn-s3-demo-bucket'
```

- Untuk detail API, lihat [ListBucketInventoryConfigurations](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-S3BucketLocation

Contoh kode berikut menunjukkan cara menggunakan `Get-S3BucketLocation`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini mengembalikan batasan lokasi untuk bucket 's3testbucket', jika ada kendala.

```
Get-S3BucketLocation -BucketName 'amzn-s3-demo-bucket'
```

Output:

```
Value
-----
ap-south-1
```

- Untuk detail API, lihat [GetBucketLocation](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-S3BucketLogging

Contoh kode berikut menunjukkan cara menggunakan `Get-S3BucketLogging`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini mengembalikan status logging untuk bucket yang ditentukan.

```
Get-S3BucketLogging -BucketName 'amzn-s3-demo-bucket'
```

Output:

TargetBucketName	Grants	TargetPrefix
-----	-----	-----
testbucket1	{}	testprefix

- Untuk detail API, lihat [GetBucketLogging](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-S3BucketMetricsConfiguration

Contoh kode berikut menunjukkan cara menggunakan `Get-S3BucketMetricsConfiguration`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini mengembalikan detail tentang filter metrik bernama 'testfilter' untuk bucket S3 yang diberikan.

```
Get-S3BucketMetricsConfiguration -BucketName 'amzn-s3-demo-bucket' -MetricsId  
'testfilter'
```

- Untuk detail API, lihat [GetBucketMetricsConfiguration](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-S3BucketNotification

Contoh kode berikut menunjukkan cara menggunakan `Get-S3BucketNotification`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil konfigurasi notifikasi dari bucket yang diberikan

```
Get-S3BucketNotification -BucketName amzn-s3-demo-bucket | select -ExpandProperty  
TopicConfigurations
```

Output:

Id	Topic
mimo	arn:aws:sns:eu-west-1:123456789012:topic-1

- Untuk detail API, lihat [GetBucketNotification](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-S3BucketPolicy

Contoh kode berikut menunjukkan cara menggunakan `Get-S3BucketPolicy`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini menampilkan kebijakan bucket yang terkait dengan bucket S3 yang diberikan.

```
Get-S3BucketPolicy -BucketName 'amzn-s3-demo-bucket'
```

- Untuk detail API, lihat [GetBucketPolicy](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-S3BucketPolicyStatus

Contoh kode berikut menunjukkan cara menggunakan `Get-S3BucketPolicyStatus`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini mengembalikan status kebijakan untuk bucket S3 yang diberikan, yang menunjukkan apakah bucket bersifat publik.

```
Get-S3BucketPolicyStatus -BucketName 'amzn-s3-demo-bucket'
```

- Untuk detail API, lihat [GetBucketPolicyStatus](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-S3BucketReplication

Contoh kode berikut menunjukkan cara menggunakan `Get-S3BucketReplication`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan informasi konfigurasi replikasi yang disetel pada bucket bernama 'mybucket'.

```
Get-S3BucketReplication -BucketName amzn-s3-demo-bucket
```

- Untuk detail API, lihat [GetBucketReplication](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-S3BucketRequestPayment

Contoh kode berikut menunjukkan cara menggunakan `Get-S3BucketRequestPayment`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan konfigurasi permintaan pembayaran untuk bucket bernama 'mybucket'. Secara default, pemilik bucket membayar unduhan dari bucket.

```
Get-S3BucketRequestPayment -BucketName amzn-s3-demo-bucket
```

- Untuk detail API, lihat [GetBucketRequestPayment](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-S3BucketTagging

Contoh kode berikut menunjukkan cara menggunakan `Get-S3BucketTagging`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini mengembalikan semua tag yang terkait dengan bucket yang diberikan.

```
Get-S3BucketTagging -BucketName 'amzn-s3-demo-bucket'
```

- Untuk detail API, lihat [GetBucketTagging](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-S3BucketVersioning

Contoh kode berikut menunjukkan cara menggunakan `Get-S3BucketVersioning`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini mengembalikan status pembuatan versi sehubungan dengan bucket yang diberikan.

```
Get-S3BucketVersioning -BucketName 'amzn-s3-demo-bucket'
```

- Untuk detail API, lihat [GetBucketVersioning](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-S3BucketWebsite

Contoh kode berikut menunjukkan cara menggunakan `Get-S3BucketWebsite`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini mengembalikan detail konfigurasi situs web statis dari bucket S3 yang diberikan.

```
Get-S3BucketWebsite -BucketName 'amzn-s3-demo-bucket'
```

- Untuk detail API, lihat [GetBucketWebsite](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-S3CORSConfiguration

Contoh kode berikut menunjukkan cara menggunakan `Get-S3CORSConfiguration`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini mengembalikan objek yang berisi semua aturan konfigurasi CORS yang sesuai dengan Bucket S3 yang diberikan.

```
Get-S3CORSConfiguration -BucketName 'amzn-s3-demo-bucket' -Select  
Configuration.Rules
```

Output:

```
AllowedMethods : {PUT, POST, DELETE}  
AllowedOrigins : {http://www.example1.com}  
Id             :  
ExposeHeaders  : {}  
MaxAgeSeconds  : 0  
AllowedHeaders : {*}  
  
AllowedMethods : {PUT, POST, DELETE}  
AllowedOrigins : {http://www.example2.com}  
Id             :  
ExposeHeaders  : {}  
MaxAgeSeconds  : 0  
AllowedHeaders : {*}  
  
AllowedMethods : {GET}  
AllowedOrigins : {*}  
Id             :  
ExposeHeaders  : {}  
MaxAgeSeconds  : 0
```

```
AllowedHeaders : {}
```

- Untuk detail API, lihat [Dapatkan CORSConfiguration](#) Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-S3LifecycleConfiguration

Contoh kode berikut menunjukkan cara menggunakan `Get-S3LifecycleConfiguration`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil konfigurasi siklus hidup untuk bucket.

```
Get-S3LifecycleConfiguration -BucketName amzn-s3-demo-bucket
```

Output:

```
Rules
-----
{Remove-in-150-days, Archive-to-Glacier-in-30-days}
```

- Untuk detail API, lihat [GetLifecycleConfiguration](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-S3Object

Contoh kode berikut menunjukkan cara menggunakan `Get-S3Object`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini mengambil informasi tentang semua item di bucket “test-files”.

```
Get-S3Object -BucketName amzn-s3-demo-bucket
```

Contoh 2: Perintah ini mengambil informasi tentang item "sample.txt" dari bucket “test-files”.

```
Get-S3Object -BucketName amzn-s3-demo-bucket -Key sample.txt
```

Contoh 3: Perintah ini mengambil informasi tentang semua item dengan awalan “sample” dari bucket “test-files”.

```
Get-S3Object -BucketName amzn-s3-demo-bucket -KeyPrefix sample
```

- Untuk detail API, lihat [ListObjects](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-S3ObjectLockConfiguration

Contoh kode berikut menunjukkan cara menggunakan `Get-S3ObjectLockConfiguration`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini mengembalikan nilai 'Diaktifkan' jika konfigurasi kunci Objek diaktifkan untuk bucket S3 yang diberikan.

```
Get-S3ObjectLockConfiguration -BucketName 'amzn-s3-demo-bucket' -Select  
ObjectLockConfiguration.ObjectLockEnabled
```

Output:

```
Value  
-----  
Enabled
```

- Untuk detail API, lihat [GetObjectLockConfiguration](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-S3ObjectMetadata

Contoh kode berikut menunjukkan cara menggunakan `Get-S3ObjectMetadata`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini mengembalikan metadata objek dengan kunci 'ListTrusts.txt' di bucket S3 yang diberikan.

```
Get-S3ObjectMetadata -BucketName 'amzn-s3-demo-bucket' -Key 'ListTrusts.txt'
```

Output:

```
Headers : Amazon.S3.Model.HeadersCollection
```

```

Metadata                : Amazon.S3.Model.MetadataCollection
DeleteMarker             :
AcceptRanges             : bytes
ContentRange             :
Expiration               :
RestoreExpiration        :
RestoreInProgress        : False
LastModified             : 01/01/2020 08:02:05
ETag                    : "d000011112a222e333e3bb4ee5d43d21"
MissingMeta              : 0
VersionId               : null
Expires                 : 01/01/0001 00:00:00
WebsiteRedirectLocation  :
ServerSideEncryptionMethod : AES256
ServerSideEncryptionCustomerMethod :
ServerSideEncryptionKeyManagementServiceKeyId :
ReplicationStatus        :
PartsCount               :
ObjectLockLegalHoldStatus :
ObjectLockMode           :
ObjectLockRetainUntilDate : 01/01/0001 00:00:00
StorageClass             :
RequestCharged           :

```

- Untuk detail API, lihat [GetObjectMetadata](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-S3ObjectRetention

Contoh kode berikut menunjukkan cara menggunakan `Get-S3ObjectRetention`.

Alat untuk PowerShell V4

Contoh 1: Perintah mengembalikan mode dan tanggal sampai objek akan dipertahankan.

```
Get-S3ObjectRetention -BucketName 'amzn-s3-demo-bucket' -Key 'testfile.txt'
```

- Untuk detail API, lihat [GetObjectRetention](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-S3ObjectTagSet

Contoh kode berikut menunjukkan cara menggunakan `Get-S3ObjectTagSet`.

Alat untuk PowerShell V4

Contoh 1: Sampel mengembalikan tag yang terkait dengan objek yang ada pada bucket S3 yang diberikan.

```
Get-S3ObjectTagSet -Key 'testfile.txt' -BucketName 'amzn-s3-demo-bucket'
```

Output:

```
Key  Value
---  -
test value
```

- Untuk detail API, lihat [GetObjectTagging](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-S3PreSignedURL

Contoh kode berikut menunjukkan cara menggunakan `Get-S3PreSignedURL`.

Alat untuk PowerShell V4

Contoh 1: Perintah mengembalikan URL yang telah ditandatangani sebelumnya untuk kunci tertentu dan tanggal kedaluwarsa.

```
Get-S3PreSignedURL -BucketName 'amzn-s3-demo-bucket' -Key 'testkey' -Expires  
'2023-11-16'
```

Contoh 2: Perintah mengembalikan URL yang telah ditandatangani sebelumnya untuk Bucket Direktori dengan kunci tertentu dan tanggal kedaluwarsa.

```
[Amazon.AWSConfigsS3]::UseSignatureVersion4 = $true  
Get-S3PreSignedURL -BucketName amzn-s3-demo-bucket--usw2-az1--x-s3 -Key  
'testkey' -Expire '2023-11-17'
```

- Untuk detail API, lihat [GetPreSignedURL](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-S3PublicAccessBlock

Contoh kode berikut menunjukkan cara menggunakan `Get-S3PublicAccessBlock`.

Alat untuk PowerShell V4

Contoh 1: Perintah mengembalikan konfigurasi blok akses publik dari bucket S3 yang diberikan.

```
Get-S3PublicAccessBlock -BucketName 'amzn-s3-demo-bucket'
```

- Untuk detail API, lihat [GetPublicAccessBlock](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-S3Version

Contoh kode berikut menunjukkan cara menggunakan `Get-S3Version`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini mengembalikan metadata tentang semua versi objek dalam bucket S3 yang diberikan.

```
Get-S3Version -BucketName 'amzn-s3-demo-bucket'
```

Output:

```
IsTruncated      : False
KeyMarker        :
VersionIdMarker  :
NextKeyMarker    :
NextVersionIdMarker :
Versions         : {EC2.txt, EC2MicrosoftWindowsGuide.txt, ListDirectories.json,
  ListTrusts.json}
Name             : s3testbucket
Prefix           :
MaxKeys          : 1000
CommonPrefixes   : {}
Delimiter        :
```

- Untuk detail API, lihat [ListVersions](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-S3Bucket

Contoh kode berikut menunjukkan cara menggunakan `New-S3Bucket`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini membuat bucket pribadi baru bernama “sample-bucket”.

```
New-S3Bucket -BucketName amzn-s3-demo-bucket
```

Contoh 2: Perintah ini membuat bucket baru bernama “sample-bucket” dengan izin baca-tulis.

```
New-S3Bucket -BucketName amzn-s3-demo-bucket -PublicReadWrite
```

Contoh 3: Perintah ini membuat bucket baru bernama “sample-bucket” dengan izin hanya-baca.

```
New-S3Bucket -BucketName amzn-s3-demo-bucket -PublicReadOnly
```

Contoh 4: Perintah ini membuat bucket Direktori baru bernama “samplebucket--use1-az5--x-s3” with. PutBucketConfiguration

```
$bucketConfiguration = @{
    BucketInfo = @{
        DataRedundancy = 'SingleAvailabilityZone'
        Type = 'Directory'
    }
    Location = @{
        Name = 'usw2-az1'
        Type = 'AvailabilityZone'
    }
}
New-S3Bucket -BucketName amzn-s3-demo-bucket--usw2-az1--x-s3 -BucketConfiguration
$bucketConfiguration -Region us-west-2
```

- Untuk detail API, lihat [PutBucket](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Read-S3Object

Contoh kode berikut menunjukkan cara menggunakan Read-S3Object.

Alat untuk PowerShell V4

Contoh 1: Perintah ini mengambil item "sample.txt" dari bucket “test-files” dan menyimpannya ke file bernama "local-sample.txt" di lokasi saat ini. File "local-sample.txt" tidak harus ada sebelum perintah ini dipanggil.

```
Read-S3Object -BucketName amzn-s3-demo-bucket -Key sample.txt -File local-sample.txt
```

Contoh 2: Perintah ini mengambil direktori virtual “DIR” dari bucket “test-files” dan menyimpannya ke folder bernama “Local-dir” di lokasi saat ini. Folder “Local-dir” tidak harus ada sebelum perintah ini dipanggil.

```
Read-S3Object -BucketName amzn-s3-demo-bucket -KeyPrefix DIR -Folder Local-DIR
```

Contoh 3: Mengunduh semua objek dengan kunci yang diakhiri dengan '.json' dari ember dengan 'konfigurasi' dalam nama ember ke file di folder yang ditentukan. Kunci objek digunakan untuk mengatur nama file.

```
Get-S3Bucket | ? { $_.BucketName -like '*config*' } | Get-S3Object | ? { $_.Key -like '*.json' } | Read-S3Object -Folder C:\ConfigObjects
```

- Untuk detail API, lihat [GetObject](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-S3Bucket

Contoh kode berikut menunjukkan cara menggunakan `Remove-S3Bucket`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini menghapus semua objek dan versi objek dari bucket 'test-files' dan kemudian menghapus bucket. Perintah akan meminta konfirmasi sebelum melanjutkan. Tambahkan sakelar `-Force` untuk menekan konfirmasi. Perhatikan bahwa ember yang tidak kosong tidak dapat dihapus.

```
Remove-S3Bucket -BucketName amzn-s3-demo-bucket -DeleteBucketContent
```

- Untuk detail API, lihat [DeleteBucket](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-S3BucketAnalyticsConfiguration

Contoh kode berikut menunjukkan cara menggunakan `Remove-S3BucketAnalyticsConfiguration`.

Alat untuk PowerShell V4

Contoh 1: Perintah menghapus filter analitik dengan nama 'testfilter' di bucket S3 yang diberikan.

```
Remove-S3BucketAnalyticsConfiguration -BucketName 'amzn-s3-demo-bucket' -AnalyticsId 'testfilter'
```

- Untuk detail API, lihat [DeleteBucketAnalyticsConfiguration](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-S3BucketEncryption

Contoh kode berikut menunjukkan cara menggunakan `Remove-S3BucketEncryption`.

Alat untuk PowerShell V4

Contoh 1: Ini menonaktifkan enkripsi yang diaktifkan untuk bucket S3 yang disediakan.

```
Remove-S3BucketEncryption -BucketName 'amzn-s3-demo-bucket'
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-S3BucketEncryption (DeleteBucketEncryption)" on
target "s3casetestbucket".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"): Y
```

- Untuk detail API, lihat [DeleteBucketEncryption](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-S3BucketInventoryConfiguration

Contoh kode berikut menunjukkan cara menggunakan `Remove-S3BucketInventoryConfiguration`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini menghapus inventori bernama 'testInventoryName' yang sesuai dengan bucket S3 yang diberikan.

```
Remove-S3BucketInventoryConfiguration -BucketName 'amzn-s3-demo-bucket' -InventoryId  
'testInventoryName'
```

Output:

```
Confirm  
Are you sure you want to perform this action?  
Performing the operation "Remove-S3BucketInventoryConfiguration  
(DeleteBucketInventoryConfiguration)" on target "s3testbucket".  
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is  
"Y"): Y
```

- Untuk detail API, lihat [DeleteBucketInventoryConfiguration](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-S3BucketMetricsConfiguration

Contoh kode berikut menunjukkan cara menggunakan `Remove-S3BucketMetricsConfiguration`.

Alat untuk PowerShell V4

Contoh 1: Perintah menghapus filter metrik dengan nama 'testmetrics' di bucket S3 yang diberikan.

```
Remove-S3BucketMetricsConfiguration -BucketName 'amzn-s3-demo-bucket' -MetricsId  
'testmetrics'
```

- Untuk detail API, lihat [DeleteBucketMetricsConfiguration](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-S3BucketPolicy

Contoh kode berikut menunjukkan cara menggunakan `Remove-S3BucketPolicy`.

Alat untuk PowerShell V4

Contoh 1: Perintah menghapus kebijakan bucket yang terkait dengan bucket S3 yang diberikan.

```
Remove-S3BucketPolicy -BucketName 'amzn-s3-demo-bucket'
```

- Untuk detail API, lihat [DeleteBucketPolicy](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-S3BucketReplication

Contoh kode berikut menunjukkan cara menggunakan `Remove-S3BucketReplication`.

Alat untuk PowerShell V4

Contoh 1: Menghapus konfigurasi replikasi yang terkait dengan bucket bernama 'mybucket'. Perhatikan bahwa operasi ini memerlukan izin untuk `DeleteReplicationConfiguration` tindakan s3:. Anda akan diminta konfirmasi sebelum operasi berlangsung - untuk menekan konfirmasi, gunakan sakelar `-Force`.

```
Remove-S3BucketReplication -BucketName amzn-s3-demo-bucket
```

- Untuk detail API, lihat [DeleteBucketReplication](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-S3BucketTagging

Contoh kode berikut menunjukkan cara menggunakan `Remove-S3BucketTagging`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini menghapus semua tag yang terkait dengan bucket S3 yang diberikan.

```
Remove-S3BucketTagging -BucketName 'amzn-s3-demo-bucket'
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-S3BucketTagging (DeleteBucketTagging)" on target
"s3testbucket".
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is
"Y"): Y
```

- Untuk detail API, lihat [DeleteBucketTagging](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-S3BucketWebsite

Contoh kode berikut menunjukkan cara menggunakan `Remove-S3BucketWebsite`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini menonaktifkan properti hosting situs web statis dari bucket S3 yang diberikan.

```
Remove-S3BucketWebsite -BucketName 'amzn-s3-demo-bucket'
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-S3BucketWebsite (DeleteBucketWebsite)" on target
"s3testbucket".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"): Y
```

- Untuk detail API, lihat [DeleteBucketWebsite](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-S3CORSConfiguration

Contoh kode berikut menunjukkan cara menggunakan `Remove-S3CORSConfiguration`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini menghapus konfigurasi CORS untuk bucket S3 yang diberikan.

```
Remove-S3CORSConfiguration -BucketName 'amzn-s3-demo-bucket'
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-S3CORSConfiguration (DeleteCORSConfiguration)" on
target "s3testbucket".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"): Y
```

- Untuk detail API, lihat [Menghapus CORSConfiguration](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-S3LifecycleConfiguration

Contoh kode berikut menunjukkan cara menggunakan `Remove-S3LifecycleConfiguration`.

Alat untuk PowerShell V4

Contoh 1: Perintah menghapus semua aturan siklus hidup untuk bucket S3 yang diberikan.

```
Remove-S3LifecycleConfiguration -BucketName 'amzn-s3-demo-bucket'
```

- Untuk detail API, lihat [DeleteLifecycleConfiguration](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-S3MultipartUpload

Contoh kode berikut menunjukkan cara menggunakan `Remove-S3MultipartUpload`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini membatalkan unggahan multipart yang dibuat lebih awal dari 5 hari yang lalu.

```
Remove-S3MultipartUpload -BucketName amzn-s3-demo-bucket -DaysBefore 5
```

Contoh 2: Perintah ini membatalkan unggahan multipart yang dibuat lebih awal dari 2 Januari 2014.

```
Remove-S3MultipartUpload -BucketName amzn-s3-demo-bucket -InitiatedDate "Thursday,  
January 02, 2014"
```

Contoh 3: Perintah ini membatalkan unggahan multipart yang dibuat lebih awal dari 2 Januari 2014, 10:45:37.

```
Remove-S3MultipartUpload -BucketName amzn-s3-demo-bucket -InitiatedDate "2014/01/02  
10:45:37"
```

- Untuk detail API, lihat [AbortMultipartUpload](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-S3Object

Contoh kode berikut menunjukkan cara menggunakan `Remove-S3Object`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini menghapus objek "sample.txt" dari bucket "test-files". Anda diminta untuk konfirmasi sebelum perintah dijalankan; untuk menekan prompt gunakan sakelar `-Force`.

```
Remove-S3Object -BucketName amzn-s3-demo-bucket -Key sample.txt
```

Contoh 2: Perintah ini menghapus versi objek "sample.txt" yang ditentukan dari bucket "test-files", dengan asumsi bucket telah dikonfigurasi untuk mengaktifkan versi objek.

```
Remove-S3Object -BucketName amzn-s3-demo-bucket -Key sample.txt -VersionId  
HLbxnx6V9omT6AQYVpks8mmFKQcejpqt
```

Contoh 3: Perintah ini menghapus objek "sample1.txt", "sample2.txt" dan "sample3.txt" dari bucket "test-files" sebagai operasi batch tunggal. Respons layanan akan mencantumkan semua kunci yang diproses, terlepas dari status keberhasilan atau kesalahan penghapusan. Untuk mendapatkan hanya kesalahan untuk kunci yang tidak dapat diproses oleh layanan tambahkan `ReportErrorsOnly` parameter - (parameter ini juga dapat ditentukan dengan alias `-Quiet`).

```
Remove-S3Object -BucketName amzn-s3-demo-bucket -KeyCollection @( "sample1.txt",  
"sample2.txt", "sample3.txt" )
```

Contoh 4: Contoh ini menggunakan ekspresi sebaris dengan `KeyCollection` parameter - untuk mendapatkan kunci objek yang akan dihapus. `Get-S3Object` mengembalikan koleksi contoh `Amazon.S3.Model.S3Object`, yang masing-masing memiliki anggota `Key` dari jenis string mengidentifikasi objek.

```
Remove-S3Object -bucketname "amzn-s3-demo-bucket" -KeyCollection (Get-S3Object  
"test-files" -KeyPrefix "prefix/subprefix" | select -ExpandProperty Key)
```

Contoh 5: Contoh ini memperoleh semua objek yang memiliki key prefix "prefix/subprefix" di bucket dan menghapusnya. Perhatikan bahwa objek yang masuk diproses satu per satu. Untuk

koleksi besar, pertimbangkan untuk meneruskan koleksi ke parameter cmdlet's - InputObject (alias -S3ObjectCollection) untuk memungkinkan penghapusan terjadi sebagai batch dengan satu panggilan ke layanan.

```
Get-S3Object -BucketName "amzn-s3-demo-bucket" -KeyPrefix "prefix/subprefix" |
Remove-S3Object -Force
```

Contoh 6: Contoh ini menyalurkan kumpulan ObjectVersion instance Amazon.S3.Model.S3 yang mewakili penanda hapus ke cmdlet untuk dihapus. Perhatikan bahwa objek yang masuk diproses satu per satu. Untuk koleksi besar, pertimbangkan untuk meneruskan koleksi ke parameter cmdlet's - InputObject (alias -S3ObjectCollection) untuk memungkinkan penghapusan terjadi sebagai batch dengan satu panggilan ke layanan.

```
(Get-S3Version -BucketName "amzn-s3-demo-bucket").Versions | Where
{$_ .IsDeleteMarker -eq "True"} | Remove-S3Object -Force
```

Contoh 7: Skrip ini menunjukkan bagaimana melakukan penghapusan batch dari satu set objek (dalam hal ini menghapus penanda) dengan membangun array objek yang akan digunakan dengan - KeyAndVersionCollection parameter.

```
$keyVersions = @()
$markers = (Get-S3Version -BucketName $BucketName).Versions | Where
{$_ .IsDeleteMarker -eq "True"}
foreach ($marker in $markers) { $keyVersions += @{ Key = $marker.Key; VersionId =
$marker.VersionId } }
Remove-S3Object -BucketName $BucketName -KeyAndVersionCollection $keyVersions -Force
```

- Untuk detail API, lihat [DeleteObjects](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-S3ObjectTagSet

Contoh kode berikut menunjukkan cara menggunakan Remove-S3ObjectTagSet.

Alat untuk PowerShell V4

Contoh 1: Perintah ini menghapus semua tag yang terkait dengan objek dengan kunci 'testfile.txt' di Bucket S3 yang diberikan.

```
Remove-S3ObjectTagSet -Key 'testfile.txt' -BucketName 'amzn-s3-demo-bucket' -Select  
'^Key'
```

Output:

```
Confirm  
Are you sure you want to perform this action?  
Performing the operation "Remove-S3ObjectTagSet (DeleteObjectTagging)" on target  
"testfile.txt".  
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is  
"Y"): Y  
testfile.txt
```

- Untuk detail API, lihat [DeleteObjectTagging](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-S3PublicAccessBlock

Contoh kode berikut menunjukkan cara menggunakan `Remove-S3PublicAccessBlock`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini mematikan setelan blokir akses publik untuk bucket yang diberikan.

```
Remove-S3PublicAccessBlock -BucketName 'amzn-s3-demo-bucket' -Force -Select  
'^BucketName'
```

Output:

```
s3testbucket
```

- Untuk detail API, lihat [DeletePublicAccessBlock](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Set-S3BucketEncryption

Contoh kode berikut menunjukkan cara menggunakan `Set-S3BucketEncryption`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini mengaktifkan enkripsi sisi AES256 server default dengan Amazon S3 Managed Keys (SSE-S3) pada bucket yang diberikan.

```
$Encryptionconfig = @{ServerSideEncryptionByDefault =  
    @{ServerSideEncryptionAlgorithm = "AES256"}}  
Set-S3BucketEncryption -BucketName 'amzn-s3-demo-bucket' -  
ServerSideEncryptionConfiguration_ServerSideEncryptionRule $Encryptionconfig
```

- Untuk detail API, lihat [PutBucketEncryption](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Test-S3Bucket

Contoh kode berikut menunjukkan cara menggunakan `Test-S3Bucket`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini mengembalikan True jika bucket ada, False sebaliknya. Perintah mengembalikan True bahkan jika bucket bukan milik pengguna.

```
Test-S3Bucket -BucketName amzn-s3-demo-bucket
```

- Untuk detail API, lihat [Test-S3Bucket](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Write-S3BucketAccelerateConfiguration

Contoh kode berikut menunjukkan cara menggunakan `Write-S3BucketAccelerateConfiguration`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini memungkinkan percepatan transfer untuk bucket S3 yang diberikan.

```
$statusVal = New-Object Amazon.S3.BucketAccelerateStatus('Enabled')  
Write-S3BucketAccelerateConfiguration -BucketName 'amzn-s3-demo-bucket' -  
AccelerateConfiguration_Status $statusVal
```

- Untuk detail API, lihat [PutBucketAccelerateConfiguration](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Write-S3BucketNotification

Contoh kode berikut menunjukkan cara menggunakan Write-S3BucketNotification.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengonfigurasi konfigurasi topik SNS untuk acara S3 ObjectRemovedDelete dan mengaktifkan notifikasi untuk bucket s3 yang diberikan

```
$topic = [Amazon.S3.Model.TopicConfiguration] @{
    Id = "delete-event"
    Topic = "arn:aws:sns:eu-west-1:123456789012:topic-1"
    Event = [Amazon.S3.EventType]::ObjectRemovedDelete
}

Write-S3BucketNotification -BucketName amzn-s3-demo-bucket -TopicConfiguration
$topic
```

Contoh 2: Contoh ini memungkinkan pemberitahuan ObjectCreatedAll untuk bucket yang diberikan mengirimnya ke fungsi Lambda.

```
$lambdaConfig = [Amazon.S3.Model.LambdaFunctionConfiguration] @{
    Events = "s3:ObjectCreated:*"
    FunctionArn = "arn:aws:lambda:eu-west-1:123456789012:function:rdplock"
    Id = "ObjectCreated-Lambda"
    Filter = @{
        S3KeyFilter = @{
            FilterRules = @(
                @{Name="Prefix";Value="dada"}
                @{Name="Suffix";Value=".pem"}
            )
        }
    }
}

Write-S3BucketNotification -BucketName amzn-s3-demo-bucket -
LambdaFunctionConfiguration $lambdaConfig
```

Contoh 3: Contoh ini membuat 2 konfigurasi Lambda yang berbeda berdasarkan akhiran kunci yang berbeda dan dikonfigurasi keduanya dalam satu perintah.

```
#Lambda Config 1
```

```

$firstLambdaConfig = [Amazon.S3.Model.LambdaFunctionConfiguration] @{
    Events = "s3:ObjectCreated:*"
    FunctionArn = "arn:aws:lambda:eu-west-1:123456789012:function:verifynet"
    Id = "ObjectCreated-dada-ps1"
    Filter = @{
        S3KeyFilter = @{
            FilterRules = @(
                @{Name="Prefix";Value="dada"}
                @{Name="Suffix";Value=".ps1"}
            )
        }
    }
}

#Lambda Config 2

$secondLambdaConfig = [Amazon.S3.Model.LambdaFunctionConfiguration] @{
    Events = [Amazon.S3.EventType]::ObjectCreatedAll
    FunctionArn = "arn:aws:lambda:eu-west-1:123456789012:function:verifyssm"
    Id = "ObjectCreated-dada-json"
    Filter = @{
        S3KeyFilter = @{
            FilterRules = @(
                @{Name="Prefix";Value="dada"}
                @{Name="Suffix";Value=".json"}
            )
        }
    }
}

Write-S3BucketNotification -BucketName amzn-s3-demo-bucket -
LambdaFunctionConfiguration $firstLambdaConfig,$secondLambdaConfig

```

- Untuk detail API, lihat [PutBucketNotification](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Write-S3BucketReplication

Contoh kode berikut menunjukkan cara menggunakan Write-S3BucketReplication.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menetapkan konfigurasi replikasi dengan satu aturan yang memungkinkan replikasi ke bucket 'exampletargetbucket' setiap objek baru yang dibuat dengan awalan nama kunci "" di bucket 'examplebucket'. TaxDocs

```
$rule1 = New-Object Amazon.S3.Model.ReplicationRule
$rule1.ID = "Rule-1"
$rule1.Status = "Enabled"
$rule1.Prefix = "TaxDocs"
$rule1.Destination = @{ BucketArn = "arn:aws:s3:::amzn-s3-demo-destination-bucket" }

$params = @{
    BucketName = "amzn-s3-demo-bucket"
    Configuration_Role = "arn:aws:iam::35667example:role/
CrossRegionReplicationRoleForS3"
    Configuration_Rule = $rule1
}

Write-S3BucketReplication @params
```

Contoh 2: Contoh ini menetapkan konfigurasi replikasi dengan beberapa aturan yang memungkinkan replikasi ke bucket 'exampletargetbucket' setiap objek baru yang dibuat dengan awalan nama kunci "" atau ". TaxDocs OtherDocs Awalan kunci tidak boleh tumpang tindih.

```
$rule1 = New-Object Amazon.S3.Model.ReplicationRule
$rule1.ID = "Rule-1"
$rule1.Status = "Enabled"
$rule1.Prefix = "TaxDocs"
$rule1.Destination = @{ BucketArn = "arn:aws:s3:::amzn-s3-demo-destination-bucket" }

$rule2 = New-Object Amazon.S3.Model.ReplicationRule
$rule2.ID = "Rule-2"
$rule2.Status = "Enabled"
$rule2.Prefix = "OtherDocs"
$rule2.Destination = @{ BucketArn = "arn:aws:s3:::amzn-s3-demo-destination-bucket" }

$params = @{
    BucketName = "amzn-s3-demo-bucket"
    Configuration_Role = "arn:aws:iam::35667example:role/
CrossRegionReplicationRoleForS3"
    Configuration_Rule = $rule1,$rule2
```

```
}

Write-S3BucketReplication @params
```

Contoh 3: Contoh ini memperbarui konfigurasi replikasi pada bucket yang ditentukan untuk menonaktifkan aturan yang mengontrol replikasi objek dengan awalan nama kunci "TaxDocs" ke bucket 'exampletargetbucket'.

```
$rule1 = New-Object Amazon.S3.Model.ReplicationRule
$rule1.ID = "Rule-1"
$rule1.Status = "Disabled"
$rule1.Prefix = "TaxDocs"
$rule1.Destination = @{ BucketArn = "arn:aws:s3:::amzn-s3-demo-destination-bucket" }

$params = @{
    BucketName = "amzn-s3-demo-bucket"
    Configuration_Role = "arn:aws:iam::35667example:role/
CrossRegionReplicationRoleForS3"
    Configuration_Rule = $rule1
}

Write-S3BucketReplication @params
```

- Untuk detail API, lihat [PutBucketReplication](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Write-S3BucketRequestPayment

Contoh kode berikut menunjukkan cara menggunakan Write-S3BucketRequestPayment.

Alat untuk PowerShell V4

Contoh 1: Memperbarui konfigurasi pembayaran permintaan untuk bucket bernama 'mybucket' sehingga orang yang meminta unduhan dari bucket akan dikenakan biaya untuk unduhan. Secara default, pemilik bucket membayar unduhan. Untuk mengatur permintaan pembayaran kembali ke default gunakan 'BucketOwner' untuk parameter RequestPaymentConfiguration _Payer.

```
Write-S3BucketRequestPayment -BucketName amzn-s3-demo-bucket -
RequestPaymentConfiguration_Payer Requester
```

- Untuk detail API, lihat [PutBucketRequestPayment](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Write-S3BucketTagging

Contoh kode berikut menunjukkan cara menggunakan `Write-S3BucketTagging`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini menerapkan dua tag ke bucket bernama **cloudtrail-test-2018**: tag dengan kunci Stage dan nilai Test, dan tag dengan kunci Environment dan nilai Alpha. Untuk memverifikasi bahwa tag telah ditambahkan ke bucket, jalankan **Get-S3BucketTagging -BucketName bucket_name**. Hasilnya harus menunjukkan tag yang Anda terapkan ke bucket di perintah pertama. Perhatikan bahwa **Write-S3BucketTagging** menimpa seluruh set tag yang ada di bucket. Untuk menambah atau menghapus tag individual, jalankan cmdlet Resource Groups dan Tagging API, dan. **Add-RGTResourceTag Remove-RGTResourceTag** Atau, gunakan Editor Tag di AWS Management Console untuk mengelola tag bucket S3.

```
Write-S3BucketTagging -BucketName amzn-s3-demo-bucket -TagSet @( @{ Key="Stage"; Value="Test" }, @{ Key="Environment"; Value="Alpha" } )
```

Contoh 2: Perintah ini menyalurkan ember yang diberi nama **cloudtrail-test-2018** ke dalam **Write-S3BucketTagging** cmdlet. Ini berlaku tag Tahap: Produksi dan Departemen: Keuangan ke ember. Perhatikan bahwa **Write-S3BucketTagging** menimpa seluruh set tag yang ada di bucket.

```
Get-S3Bucket -BucketName amzn-s3-demo-bucket | Write-S3BucketTagging -TagSet @( @{ Key="Stage"; Value="Production" }, @{ Key="Department"; Value="Finance" } )
```

- Untuk detail API, lihat [PutBucketTagging](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Write-S3BucketVersioning

Contoh kode berikut menunjukkan cara menggunakan `Write-S3BucketVersioning`.

Alat untuk PowerShell V4

Contoh 1: Perintah memungkinkan pembuatan versi untuk bucket S3 yang diberikan.

```
Write-S3BucketVersioning -BucketName 'amzn-s3-demo-bucket' -VersioningConfig_Status
Enabled
```

- Untuk detail API, lihat [PutBucketVersioning](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Write-S3BucketWebsite

Contoh kode berikut menunjukkan cara menggunakan `Write-S3BucketWebsite`.

Alat untuk PowerShell V4

Contoh 1: Perintah memungkinkan hosting situs web untuk bucket yang diberikan dengan dokumen indeks sebagai 'index.html' dan dokumen kesalahan sebagai 'error.html'.

```
Write-S3BucketWebsite -BucketName 'amzn-s3-demo-bucket'
-WebsiteConfiguration_IndexDocumentSuffix 'index.html' -
WebsiteConfiguration_ErrorDocument 'error.html'
```

- Untuk detail API, lihat [PutBucketWebsite](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Write-S3LifecycleConfiguration

Contoh kode berikut menunjukkan cara menggunakan `Write-S3LifecycleConfiguration`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menulis/ menggantikan konfigurasi yang disediakan dalam `$NewRule`. Konfigurasi ini memastikan untuk membatasi objek lingkup dengan awalan dan nilai tag yang diberikan.

```
$NewRule = [Amazon.S3.Model.LifecycleRule] @{
    Expiration = @{
        Days= 50
    }
    Id = "Test-From-Write-cmdlet-1"
    Filter= @{
        LifecycleFilterPredicate = [Amazon.S3.Model.LifecycleAndOperator]@{
            Operands= @(
                [Amazon.S3.Model.LifecyclePrefixPredicate] @{
                    "Prefix" = "py"
                }
            )
        }
    }
}
```

```

    },
    [Amazon.S3.Model.LifecycleTagPredicate] @{
        "Tag"= @{
            "Key" = "non-use"
            "Value" = "yes"
        }
    }
)
}
}
"Status"= 'Enabled'
NoncurrentVersionExpiration = @{
    NoncurrentDays = 75
}
}

Write-S3LifecycleConfiguration -BucketName amzn-s3-demo-bucket -Configuration_Rule
$NewRule

```

Contoh 2: Contoh ini menetapkan beberapa aturan dengan penyaringan. \$ ArchiveRule menetapkan objek untuk diarsipkan dalam 30 hari ke Glacier dan 120 ke. DeepArchive \$ ExpireRule kedaluwarsa versi saat ini dan sebelumnya dalam 150 hari untuk objek dengan awalan 'py' dan tag:key 'archieved' disetel ke 'ya'.

```

$ExpireRule = [Amazon.S3.Model.LifecycleRule] @{
    Expiration = @{
        Days= 150
    }
    Id = "Remove-in-150-days"
    Filter= @{
        LifecycleFilterPredicate = [Amazon.S3.Model.LifecycleAndOperator]@{
            Operands= @(
                [Amazon.S3.Model.LifecyclePrefixPredicate] @{
                    "Prefix" = "py"
                },
                [Amazon.S3.Model.LifecycleTagPredicate] @{
                    "Tag"= @{
                        "Key" = "archieved"
                        "Value" = "yes"
                    }
                }
            )
        }
    }
}

```

```

    }
    Status= 'Enabled'
    NoncurrentVersionExpiration = @{
        NoncurrentDays = 150
    }
}

$ArchiveRule = [Amazon.S3.Model.LifecycleRule] @{
    Expiration = $null
    Id = "Archive-to-Glacier-in-30-days"
    Filter= @{
        LifecycleFilterPredicate = [Amazon.S3.Model.LifecycleAndOperator]@{
            Operands= @(
                [Amazon.S3.Model.LifecyclePrefixPredicate] @{
                    "Prefix" = "py"
                },
                [Amazon.S3.Model.LifecycleTagPredicate] @{
                    "Tag"= @{
                        "Key" = "reviewed"
                        "Value" = "yes"
                    }
                }
            )
        }
    }
    Status = 'Enabled'
    NoncurrentVersionExpiration = @{
        NoncurrentDays = 75
    }
    Transitions = @(
        @{
            Days = 30
            "StorageClass"= 'Glacier'
        },
        @{
            Days = 120
            "StorageClass"= [Amazon.S3.S3StorageClass]::DeepArchive
        }
    )
}

Write-S3LifecycleConfiguration -BucketName amzn-s3-demo-bucket -Configuration_Rule
$ExpireRule,$ArchiveRule

```

- Untuk detail API, lihat [PutLifecycleConfiguration](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Write-S3Object

Contoh kode berikut menunjukkan cara menggunakan `Write-S3Object`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini mengunggah file tunggal "local-sample.txt" ke Amazon S3, membuat objek dengan kunci "sample.txt" di bucket "test-files".

```
Write-S3Object -BucketName amzn-s3-demo-bucket -Key "sample.txt" -File .\local-sample.txt
```

Contoh 2: Perintah ini mengunggah file tunggal "sample.txt" ke Amazon S3, membuat objek dengan kunci "sample.txt" di bucket "test-files". Jika parameter `-Key` tidak disediakan, nama file digunakan sebagai kunci objek S3.

```
Write-S3Object -BucketName amzn-s3-demo-bucket -File .\sample.txt
```

Contoh 3: Perintah ini mengunggah file tunggal "local-sample.txt" ke Amazon S3, membuat objek dengan kunci prefix/to/sample ".txt" di bucket "test-files".

```
Write-S3Object -BucketName amzn-s3-demo-bucket -Key "prefix/to/sample.txt" -File .\local-sample.txt
```

Contoh 4: Perintah ini mengunggah semua file di subdirektori "Scripts" ke bucket "test-files" dan menerapkan common key prefix "" untuk setiap objek. SampleScripts Setiap file yang diunggah akan memiliki kunci "SampleScripts/filename" di mana 'nama filen' bervariasi.

```
Write-S3Object -BucketName amzn-s3-demo-bucket -Folder .\Scripts -KeyPrefix SampleScripts\
```

Contoh 5: Perintah ini mengunggah semua file*.ps1 di direktur lokal "Scripts" ke bucket "test-files" dan menerapkan common key prefix "" ke setiap objek. SampleScripts Setiap file yang diunggah akan memiliki kunci "SampleScripts/filename.ps1" di mana 'nama filen' bervariasi.

```
Write-S3Object -BucketName amzn-s3-demo-bucket -Folder .\Scripts -KeyPrefix
SampleScripts\ -SearchPattern *.ps1
```

Contoh 6: Perintah ini membuat objek S3 baru yang berisi string konten tertentu dengan kunci 'sample.txt'.

```
Write-S3Object -BucketName amzn-s3-demo-bucket -Key "sample.txt" -Content "object
contents"
```

Contoh 7: Perintah ini mengunggah file yang ditentukan (nama file digunakan sebagai kunci) dan menerapkan tag yang ditentukan ke objek baru.

```
Write-S3Object -BucketName amzn-s3-demo-bucket -File "sample.txt" -TagSet
@{Key="key1";Value="value1"},@{Key="key2";Value="value2"}
```

Contoh 8: Perintah ini secara rekursif mengunggah folder yang ditentukan dan menerapkan tag yang ditentukan ke semua objek baru.

```
Write-S3Object -BucketName amzn-s3-demo-bucket -Folder . -KeyPrefix "TaggedFiles" -
Recurse -TagSet @{Key="key1";Value="value1"},@{Key="key2";Value="value2"}
```

- Untuk detail API, lihat [PutObject](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Write-S3ObjectRetention

Contoh kode berikut menunjukkan cara menggunakan Write-S3ObjectRetention.

Alat untuk PowerShell V4

Contoh 1: Perintah mengaktifkan mode retensi tata kelola hingga tanggal '31 Des 2019 00:00:00' untuk objek 'testfile.txt' di bucket S3 yang diberikan.

```
Write-S3ObjectRetention -BucketName 'amzn-s3-demo-bucket' -Key 'testfile.txt' -
Retention_Mode GOVERNANCE -Retention_RetainUntilDate "2019-12-31T00:00:00"
```

- Untuk detail API, lihat [PutObjectRetention](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Contoh S3 Glacier menggunakan Alat untuk V4 PowerShell

Contoh kode berikut menunjukkan cara melakukan tindakan dan menerapkan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan S3 Glacier.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

Get-GLCJob

Contoh kode berikut menunjukkan cara menggunakan Get-GLCJob.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan rincian pekerjaan yang ditentukan. Ketika pekerjaan berhasil diselesaikan, cmdlet Read- GCJob Output dapat digunakan untuk mengambil konten pekerjaan (arsip atau daftar inventaris) ke sistem file lokal.

```
Get-GLCJob -VaultName myvault -JobId "op1x...JSbthM"
```

Output:

```
Action : ArchiveRetrieval
ArchiveId : o909j...X-TpIhQJw
ArchiveSHA256TreeHash : 79f3ea754c02f58...dc57bf4395b
ArchiveSizeInBytes : 38034480
Completed : False
CompletionDate : 1/1/0001 12:00:00 AM
CreationDate : 12/13/2018 11:00:14 AM
InventoryRetrievalParameters :
InventorySizeInBytes : 0
```

```

JobDescription      :
JobId               : op1x...JSbthM
JobOutputPath       :
OutputLocation      :
RetrievalByteRange  : 0-38034479
SelectParameters    :
SHA256TreeHash      : 79f3ea754c02f58...dc57bf4395b
SNSTopic            :
StatusCode          : InProgress
StatusMessage       :
Tier                : Standard
VaultARN            : arn:aws:glacier:us-west-2:012345678912:vaults/test

```

- Untuk detail API, lihat [DescribeJob](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-GLCVault

Contoh kode berikut menunjukkan cara menggunakan `New-GLCVault`.

Alat untuk PowerShell V4

Contoh 1: Membuat vault baru untuk akun pengguna. Karena tidak ada nilai yang diberikan ke `AccountId` parameter -, cmdlet menggunakan default “-” yang menunjukkan akun saat ini.

```
New-GLCVault -VaultName myvault
```

Output:

```
/01234567812/vaults/myvault
```

- Untuk detail API, lihat [CreateVault](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Read-GLCJobOutput

Contoh kode berikut menunjukkan cara menggunakan `Read-GLCJobOutput`.

Alat untuk PowerShell V4

Contoh 1: Mengunduh konten arsip yang dijadwalkan untuk pengambilan dalam pekerjaan yang ditentukan dan menyimpan konten ke dalam file pada disk. Unduhan memvalidasi checksum

untuk Anda, jika tersedia. Jika diinginkan seluruh respons termasuk checksum dapat dikembalikan dengan menentukan **-Select '*'**.

```
Read-GLCJobOutput -VaultName myvault -JobId "HSWjArc...Zq2XLiW" -FilePath "c:\temp\blue.bin"
```

- Untuk detail API, lihat [GetJobOutput](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Start-GLCJob

Contoh kode berikut menunjukkan cara menggunakan **Start-GLCJob**.

Alat untuk PowerShell V4

Contoh 1: Memulai pekerjaan untuk mengambil arsip dari brankas tertentu yang dimiliki oleh pengguna. Status pekerjaan dapat diperiksa menggunakan **Get-GLCJob** cmdlet. Ketika pekerjaan berhasil diselesaikan, cmdlet **Read-GLCJob Output** dapat digunakan untuk mengambil isi arsip ke sistem file lokal.

```
Start-GLCJob -VaultName myvault -JobType "archive-retrieval" -JobDescription "archive retrieval" -ArchiveId "o909j...TX-TpIhQJw"
```

Output:

JobId	JobOutputPath	Location
-----	-----	-----
op1x...JSbthM		/012345678912/vaults/test/jobs/op1xe...I4HqCHkSJSbthM

- Untuk detail API, lihat [InitiateJob](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Write-GLCArchive

Contoh kode berikut menunjukkan cara menggunakan **Write-GLCArchive**.

Alat untuk PowerShell V4

Contoh 1: Mengunggah satu file ke brankas yang ditentukan, mengembalikan ID arsip dan checksum yang dihitung.

```
Write-GLCArchive -VaultName myvault -FilePath c:\temp\blue.bin
```

Output:

FilePath	ArchiveId	Checksum
-----	-----	-----
C:\temp\blue.bin	o909jUUs...TTX-TpIhQJw	79f3e...f4395b

Contoh 2: Mengunggah konten hierarki folder ke brankas yang ditentukan di akun pengguna. Untuk setiap file yang diunggah, cmdlet memancarkan nama file, ID arsip yang sesuai, dan checksum arsip yang dihitung.

```
Write-GLCArchive -VaultName myvault -FolderPath . -Recurse
```

Output:

FilePath	ArchiveId	Checksum
-----	-----	-----
C:\temp\blue.bin	o909jUUs...TTX-TpIhQJw	79f3e...f4395b
C:\temp\green.bin	qXAf0dSG...czo729UHXrw	d50a1...9184b9
C:\temp\lum.bin	39aNifP3...q9nb8nZkFIg	28886...5c3e27
C:\temp\red.bin	vp7E6rU...Ejk_HhjAxKA	e05f7...4e34f5
C:\temp\Folder1\file1.txt	_eRINlip...5Sxy7dD2BaA	d0d2a...c8a3ba
C:\temp\Folder2\file2.iso	-Ix3jlm...iXiDh-Xf0PA	7469e...3e86f1

- Untuk detail API, lihat [UploadArchive](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Contoh Security Hub menggunakan Alat untuk PowerShell V4

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan Security Hub.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

Get-SHUBFinding

Contoh kode berikut menunjukkan cara menggunakan `Get-SHUBFinding`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini mengambil temuan Security Hub dari Amazon EC2; layanan.

```
$filter = New-Object -TypeName Amazon.SecurityHub.Model.AwsSecurityFindingFilters
$filter.ResourceType = New-Object -TypeName Amazon.SecurityHub.Model.StringFilter -
Property @{
    Comparison = 'PREFIX'
    Value = 'AwsEc2'
}
Get-SHUBFinding -Filter $filter
```

Contoh 2: Perintah ini mengambil temuan Security Hub dari ID AWS akun 123456789012.

```
$filter = New-Object -TypeName Amazon.SecurityHub.Model.AwsSecurityFindingFilters
$filter.AwsAccountId = New-Object -TypeName Amazon.SecurityHub.Model.StringFilter -
Property @{
    Comparison = 'EQUALS'
    Value = '123456789012'
}
Get-SHUBFinding -Filter $filter
```

Contoh 3: Perintah ini mengambil temuan Security Hub yang dihasilkan untuk standar “pci-dss”.

```
$filter = New-Object -TypeName Amazon.SecurityHub.Model.AwsSecurityFindingFilters
$filter.GeneratorId = New-Object -TypeName Amazon.SecurityHub.Model.StringFilter -
Property @{
    Comparison = 'PREFIX'
    Value = 'pci-dss'
}
Get-SHUBFinding -Filter $filter
```

Contoh 4: Perintah ini mengambil temuan tingkat keparahan kritis Security Hub yang memiliki status alur kerja NOTIFIED.

```
$filter = New-Object -TypeName Amazon.SecurityHub.Model.AwsSecurityFindingFilters
```

```
$filter.SeverityLabel = New-Object -TypeName Amazon.SecurityHub.Model.StringFilter -
Property @{
    Comparison = 'EQUALS'
    Value = 'CRITICAL'
}
$filter.WorkflowStatus = New-Object -TypeName Amazon.SecurityHub.Model.StringFilter
-Property @{
    Comparison = 'EQUALS'
    Value = 'NOTIFIED'
}
Get-SHUBFinding -Filter $filter
```

- Untuk detail API, lihat [GetFindings](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Amazon SES contoh menggunakan Alat untuk PowerShell V4

Contoh kode berikut menunjukkan cara melakukan tindakan dan menerapkan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan Amazon SES.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

Get-SESIentity

Contoh kode berikut menunjukkan cara menggunakan `Get-SESIentity`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini mengembalikan daftar yang berisi semua identitas (alamat email dan domain) untuk AWS Akun tertentu, terlepas dari status verifikasi.

```
Get-SESIIdentity
```

- Untuk detail API, lihat [ListIdentities](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SESSendQuota

Contoh kode berikut menunjukkan cara menggunakan `Get-SESSendQuota`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini mengembalikan batas pengiriman pengguna saat ini.

```
Get-SESSendQuota
```

- Untuk detail API, lihat [GetSendQuota](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SESSendStatistic

Contoh kode berikut menunjukkan cara menggunakan `Get-SESSendStatistic`.

Alat untuk PowerShell V4

Contoh 1: Perintah ini mengembalikan statistik pengiriman pengguna. Hasilnya adalah daftar titik data, yang mewakili dua minggu terakhir aktivitas pengiriman. Setiap titik data dalam daftar berisi statistik untuk interval 15 menit.

```
Get-SESSendStatistic
```

- Untuk detail API, lihat [GetSendStatistics](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Amazon SES API v2 contoh menggunakan Alat untuk PowerShell V4

Contoh kode berikut menunjukkan cara melakukan tindakan dan menerapkan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan Amazon SES API v2.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

Send-SES2Email

Contoh kode berikut menunjukkan cara menggunakan Send-SES2Email.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menunjukkan cara mengirim pesan email standar.

```
Send-SES2Email -FromEmailAddress "sender@example.com" -Destination_ToAddress  
"recipient@example.com" -Subject_Data "Email Subject" -Text_Data "Email Body"
```

- Untuk detail API, lihat [SendEmail](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Contoh Amazon SNS menggunakan Alat untuk V4 PowerShell

Contoh kode berikut menunjukkan cara melakukan tindakan dan menerapkan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan Amazon SNS.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

Publish-SNSMessage

Contoh kode berikut menunjukkan cara menggunakan Publish-SNSMessage.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menunjukkan penerbitan pesan dengan satu baris yang MessageAttribute dideklarasikan.

```
Publish-SNSMessage -TopicArn "arn:aws:sns:us-west-2:123456789012:my-topic" -Message  
"Hello" -MessageAttribute  
@{'City'=[Amazon.SimpleNotificationService.Model.MessageAttributeValue]@{'DataType='String';  
StringValue ='AnyCity'}}
```

Contoh 2: Contoh ini menunjukkan penerbitan pesan dengan beberapa MessageAttributes dideklarasikan sebelumnya.

```
$cityAttributeValue = New-Object  
Amazon.SimpleNotificationService.Model.MessageAttributeValue  
$cityAttributeValue.DataType = "String"  
$cityAttributeValue.StringValue = "AnyCity"  
  
$populationAttributeValue = New-Object  
Amazon.SimpleNotificationService.Model.MessageAttributeValue  
$populationAttributeValue.DataType = "Number"  
$populationAttributeValue.StringValue = "1250800"  
  
$messageAttributes = New-Object System.Collections.Hashtable  
$messageAttributes.Add("City", $cityAttributeValue)  
$messageAttributes.Add("Population", $populationAttributeValue)  
  
Publish-SNSMessage -TopicArn "arn:aws:sns:us-west-2:123456789012:my-topic" -Message  
"Hello" -MessageAttribute $messageAttributes
```

- Untuk detail API, lihat [Menerbitkan](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Contoh Amazon SQS menggunakan Alat untuk V4 PowerShell

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan Amazon SQS.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

Add-SQSPermission

Contoh kode berikut menunjukkan cara menggunakan Add-SQSPermission.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memungkinkan yang ditentukan Akun AWS untuk mengirim pesan dari antrian yang ditentukan.

```
Add-SQSPermission -Action SendMessage -AWSAccountId 80398EXAMPLE -Label  
SendMessagesFromMyQueue -QueueUrl https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/  
MyQueue
```

- Untuk detail API, lihat [AddPermission](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Clear-SQSQueue

Contoh kode berikut menunjukkan cara menggunakan Clear-SQSQueue.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus semua pesan dari antrian yang ditentukan.

```
Clear-SQSQueue -QueueUrl https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyQueue
```

- Untuk detail API, lihat [PurgeQueue](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Edit-SQSMessageVisibility

Contoh kode berikut menunjukkan cara menggunakan `Edit-SQSMessageVisibility`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengubah batas waktu visibilitas untuk pesan dengan pegangan tanda terima yang ditentukan dalam antrian yang ditentukan menjadi 10 jam (10 jam * 60 menit * 60 detik = 36000 detik).

```
Edit-SQSMessageVisibility -QueueUrl https://sqs.us-east-1.amazonaws.com/8039EXAMPLE/MyQueue -ReceiptHandle AQEBgGDh...J/Iqww== -VisibilityTimeout 36000
```

- Untuk detail API, lihat [ChangeMessageVisibility](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Edit-SQSMessageVisibilityBatch

Contoh kode berikut menunjukkan cara menggunakan `Edit-SQSMessageVisibilityBatch`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengubah batas waktu visibilitas untuk 2 pesan dengan tanda terima yang ditentukan dalam antrian yang ditentukan. Batas waktu visibilitas pesan pertama diubah menjadi 10 jam (10 jam * 60 menit * 60 detik = 36000 detik). Batas waktu visibilitas pesan kedua diubah menjadi 5 jam (5 jam * 60 menit * 60 detik = 18000 detik).

```
$changeVisibilityRequest1 = New-Object
    Amazon.SQS.Model.ChangeMessageVisibilityBatchRequestEntry
$changeVisibilityRequest1.Id = "Request1"
$changeVisibilityRequest1.ReceiptHandle = "AQEBd329...v6gl8Q=="
$changeVisibilityRequest1.VisibilityTimeout = 36000

$changeVisibilityRequest2 = New-Object
    Amazon.SQS.Model.ChangeMessageVisibilityBatchRequestEntry
$changeVisibilityRequest2.Id = "Request2"
```

```
$changeVisibilityRequest2.ReceiptHandle = "AQEBgGDh...J/Iqww=="
$changeVisibilityRequest2.VisibilityTimeout = 18000

Edit-SQSMessageVisibilityBatch -QueueUrl https://sqs.us-
east-1.amazonaws.com/80398EXAMPLE/MyQueue -Entry $changeVisibilityRequest1,
$changeVisibilityRequest2
```

Output:

```
Failed      Successful
-----
{}          {Request2, Request1}
```

- Untuk detail API, lihat [ChangeMessageVisibilityBatch](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SQSDeadLetterSourceQueue

Contoh kode berikut menunjukkan cara menggunakan `Get-SQSDeadLetterSourceQueue`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan antrian apa pun yang bergantung pada antrian yang ditentukan sebagai antrian surat mati mereka. URLs

```
Get-SQSDeadLetterSourceQueue -QueueUrl https://sqs.us-
east-1.amazonaws.com/80398EXAMPLE/MyDeadLetterQueue
```

Output:

```
https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyQueue
https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyOtherQueue
```

- Untuk detail API, lihat [ListDeadLetterSourceQueues](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SQSQueue

Contoh kode berikut menunjukkan cara menggunakan `Get-SQSQueue`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan semua antrian.

```
Get-SQSQueue
```

Output:

```
https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyQueue  
https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/AnotherQueue  
https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/DeadLetterQueue  
https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyOtherQueue  
https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyDeadLetterQueue
```

Contoh 2: Contoh ini mencantumkan antrian apa pun yang dimulai dengan nama yang ditentukan.

```
Get-SQSQueue -QueueNamePrefix My
```

Output:

```
https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyQueue  
https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyOtherQueue  
https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyDeadLetterQueue
```

- Untuk detail API, lihat [ListQueues](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SQSQueueAttribute

Contoh kode berikut menunjukkan cara menggunakan `Get-SQSQueueAttribute`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan semua atribut untuk antrian yang ditentukan.

```
Get-SQSQueueAttribute -AttributeName All -QueueUrl https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyQueue
```

Output:

```

VisibilityTimeout           : 30
DelaySeconds                : 0
MaximumMessageSize         : 262144
MessageRetentionPeriod     : 345600
ApproximateNumberOfMessages : 0
ApproximateNumberOfMessagesNotVisible : 0
ApproximateNumberOfMessagesDelayed : 0
CreatedTimestamp           : 2/11/2015 5:53:35 PM
LastModifiedTimestamp      : 12/29/2015 2:23:17 PM
QueueARN                   : arn:aws:sqs:us-east-1:80398EXAMPLE:MyQueue
Policy                     :
  {"Version":"2008-10-17","Id":"arn:aws:sqs:us-east-1:80398EXAMPLE:MyQueue/
SQSDefaultPolicy","Statement":[{"Sid":"Sid14
                                495134224EX","Effect":"Allow","Principal":
{"AWS":"*"},"Action":"SQS:SendMessage","Resource":"arn:aws:sqs:us-east-1:80
                                398EXAMPLE:MyQueue","Condition":
{"ArnEquals":{"aws:SourceArn":"arn:aws:sns:us-east-1:80398EXAMPLE:MyTopic"}}},
{"Sid":
  "SendMessageFromMyQueue","Effect":"Allow","Principal":
{"AWS":"80398EXAMPLE"},"Action":"SQS:SendMessage","Resource":
                                arn:aws:sqs:us-
east-1:80398EXAMPLE:MyQueue"}]}
Attributes                  : {[QueueArn, arn:aws:sqs:us-
east-1:80398EXAMPLE:MyQueue], [ApproximateNumberOfMessages, 0],
                                [ApproximateNumberOfMessagesNotVisible, 0],
                                [ApproximateNumberOfMessagesDelayed, 0]...}

```

Contoh 2: Contoh ini mencantumkan secara terpisah hanya atribut yang ditentukan untuk antrian yang ditentukan.

```

Get-SQSQueueAttribute -AttributeName MaximumMessageSize, VisibilityTimeout -QueueUrl
https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyQueue

```

Output:

```

VisibilityTimeout           : 30
DelaySeconds                : 0
MaximumMessageSize         : 262144
MessageRetentionPeriod     : 345600
ApproximateNumberOfMessages : 0
ApproximateNumberOfMessagesNotVisible : 0

```

```

ApproximateNumberOfMessagesDelayed      : 0
CreatedTimestamp                        : 2/11/2015 5:53:35 PM
LastModifiedTimestamp                   : 12/29/2015 2:23:17 PM
QueueARN                               : arn:aws:sqs:us-east-1:80398EXAMPLE:MyQueue
Policy                                 :
  {"Version":"2008-10-17","Id":"arn:aws:sqs:us-east-1:80398EXAMPLE:MyQueue/
SQSDefaultPolicy","Statement":[{"Sid":"Sid14
                                495134224EX","Effect":"Allow","Principal":
{"AWS":"*"},"Action":"SQS:SendMessage","Resource":"arn:aws:sqs:us-east-1:80
                                398EXAMPLE:MyQueue","Condition":
{"ArnEquals":{"aws:SourceArn":"arn:aws:sns:us-east-1:80398EXAMPLE:MyTopic"}}},
{"Sid":
  "SendMessageFromMyQueue","Effect":"Allow","Principal":
{"AWS":"80398EXAMPLE"},"Action":"SQS:SendMessage","Resource":"
                                arn:aws:sqs:us-
east-1:80398EXAMPLE:MyQueue"}]}
Attributes                             : {[MaximumMessageSize, 262144],
[VisibilityTimeout, 30]}

```

- Untuk detail API, lihat [GetQueueAttributes](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SQSQueueUrl

Contoh kode berikut menunjukkan cara menggunakan `Get-SQSQueueUrl`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan URL antrian dengan nama yang ditentukan.

```
Get-SQSQueueUrl -QueueName MyQueue
```

Output:

```
https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyQueue
```

- Untuk detail API, lihat [GetQueueUrl](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-SQSQueue

Contoh kode berikut menunjukkan cara menggunakan `New-SQSQueue`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat antrian dengan nama yang ditentukan.

```
New-SQSQueue -QueueName MyQueue
```

Output:

```
https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyQueue
```

- Untuk detail API, lihat [CreateQueue](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Receive-SQSMessage

Contoh kode berikut menunjukkan cara menggunakan `Receive-SQSMessage`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan informasi hingga 10 pesan berikutnya yang akan diterima untuk antrian yang ditentukan. Informasi akan berisi nilai untuk atribut pesan yang ditentukan, jika ada.

```
Receive-SQSMessage -AttributeName SenderId, SentTimestamp -MessageAttributeName
  StudentName, StudentGrade -MessageCount 10 -QueueUrl https://sqs.us-
  east-1.amazonaws.com/80398EXAMPLE/MyQueue
```

Output:

```
Attributes           : {[SenderId, AIDAIAZKMSNQ7TEEXAMPLE], [SentTimestamp,
  1451495923744]}
Body                  : Information about John Doe's grade.
MD5ofBody             : ea572796e3c231f974fe75d89EXAMPLE
MD5ofMessageAttributes : 48c1ee811f0fe7c4e88fbe0f5EXAMPLE
MessageAttributes     : {[StudentGrade, Amazon.SQS.Model.MessageAttributeValue],
  [StudentName, Amazon.SQS.Model.MessageAttributeValue]}
MessageId             : 53828c4b-631b-469b-8833-c093cEXAMPLE
ReceiptHandle         : AQEBpfGp...20Q5cg==
```

- Untuk detail API, lihat [ReceiveMessage](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-SQSMessage

Contoh kode berikut menunjukkan cara menggunakan `Remove-SQSMessage`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus pesan dengan pegangan tanda terima yang ditentukan dari antrian yang ditentukan.

```
Remove-SQSMessage -QueueUrl https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyQueue
-ReceiptHandle AQEBd329...v6gl8Q==
```

- Untuk detail API, lihat [DeleteMessage](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-SQSMessageBatch

Contoh kode berikut menunjukkan cara menggunakan `Remove-SQSMessageBatch`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus 2 pesan dengan tanda terima yang ditentukan dari antrian yang ditentukan.

```
$deleteMessageRequest1 = New-Object Amazon.SQS.Model.DeleteMessageBatchRequestEntry
$deleteMessageRequest1.Id = "Request1"
$deleteMessageRequest1.ReceiptHandle = "AQEBX2g4...wtJSQg=="

$deleteMessageRequest2 = New-Object Amazon.SQS.Model.DeleteMessageBatchRequestEntry
$deleteMessageRequest2.Id = "Request2"
$deleteMessageRequest2.ReceiptHandle = "AQEBq0VY...KTsLYg=="

Remove-SQSMessageBatch -QueueUrl https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/
MyQueue -Entry $deleteMessageRequest1, $deleteMessageRequest2
```

Output:

Failed	Successful
-----	-----
{}	{Request1, Request2}

- Untuk detail API, lihat [DeleteMessageBatch](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-SQSPermission

Contoh kode berikut menunjukkan cara menggunakan `Remove-SQSPermission`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus pengaturan izin dengan label yang ditentukan dari antrian yang ditentukan.

```
Remove-SQSPermission -Label SendMessagesFromMyQueue -QueueUrl https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyQueue
```

- Untuk detail API, lihat [RemovePermission](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-SQSQueue

Contoh kode berikut menunjukkan cara menggunakan `Remove-SQSQueue`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus antrian yang ditentukan.

```
Remove-SQSQueue -QueueUrl https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyQueue
```

- Untuk detail API, lihat [DeleteQueue](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Send-SQSMessage

Contoh kode berikut menunjukkan cara menggunakan `Send-SQSMessage`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengirimkan pesan dengan atribut dan badan pesan yang ditentukan ke antrian yang ditentukan dengan pengiriman pesan tertunda selama 10 detik.

```
$cityAttributeValue = New-Object Amazon.SQS.Model.MessageAttributeValue
```

```

$cityAttributeValue.DataType = "String"
$cityAttributeValue.StringValue = "AnyCity"

$populationAttributeValue = New-Object Amazon.SQS.Model.MessageAttributeValue
$populationAttributeValue.DataType = "Number"
$populationAttributeValue.StringValue = "1250800"

$messageAttributes = New-Object System.Collections.Hashtable
$messageAttributes.Add("City", $cityAttributeValue)
$messageAttributes.Add("Population", $populationAttributeValue)

Send-SQSMessage -DelayInSeconds 10 -MessageAttributes $messageAttributes -
MessageBody "Information about the largest city in Any Region." -QueueUrl https://
sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyQueue

```

Output:

MD50fMessageAttributes	MD50fMessageBody	MessageId
-----	-----	-----
1d3e51347bc042efbdf6dda31EXAMPLE c739-4d0c-818b-1820eEXAMPLE	51b0a3256d59467f973009b73EXAMPLE	c35fed8f-

- Untuk detail API, lihat [SendMessage](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Send-SQSMessageBatch

Contoh kode berikut menunjukkan cara menggunakan `Send-SQSMessageBatch`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengirimkan 2 pesan dengan atribut tertentu dan badan pesan ke antrian yang ditentukan. Pengiriman ditunda selama 15 detik untuk pesan pertama dan 10 detik untuk pesan kedua.

```

$student1NameAttributeValue = New-Object Amazon.SQS.Model.MessageAttributeValue
$student1NameAttributeValue.DataType = "String"
$student1NameAttributeValue.StringValue = "John Doe"

$student1GradeAttributeValue = New-Object Amazon.SQS.Model.MessageAttributeValue

```

```

$student1GradeAttributeValue.DataType = "Number"
$student1GradeAttributeValue.StringValue = "89"

$student2NameAttributeValue = New-Object Amazon.SQS.Model.MessageAttributeValue
$student2NameAttributeValue.DataType = "String"
$student2NameAttributeValue.StringValue = "Jane Doe"

$student2GradeAttributeValue = New-Object Amazon.SQS.Model.MessageAttributeValue
$student2GradeAttributeValue.DataType = "Number"
$student2GradeAttributeValue.StringValue = "93"

$message1 = New-Object Amazon.SQS.Model.SendMessageBatchRequestEntry
$message1.DelaySeconds = 15
$message1.Id = "FirstMessage"
$message1.MessageAttributes.Add("StudentName", $student1NameAttributeValue)
$message1.MessageAttributes.Add("StudentGrade", $student1GradeAttributeValue)
$message1.MessageBody = "Information about John Doe's grade."

$message2 = New-Object Amazon.SQS.Model.SendMessageBatchRequestEntry
$message2.DelaySeconds = 10
$message2.Id = "SecondMessage"
$message2.MessageAttributes.Add("StudentName", $student2NameAttributeValue)
$message2.MessageAttributes.Add("StudentGrade", $student2GradeAttributeValue)
$message2.MessageBody = "Information about Jane Doe's grade."

Send-SQSMessageBatch -QueueUrl https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/
MyQueue -Entry $message1, $message2

```

Output:

```

Failed      Successful
-----
{}          {FirstMessage, SecondMessage}

```

- Untuk detail API, lihat [SendMessageBatch](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Set-SQSQueueAttribute

Contoh kode berikut menunjukkan cara menggunakan `Set-SQSQueueAttribute`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menunjukkan cara menyetel kebijakan berlangganan antrian ke topik SNS. Ketika pesan dipublikasikan ke topik, pesan dikirim ke antrian berlangganan.

```
# create the queue and topic to be associated
$qurl = New-SQSQueue -QueueName "myQueue"
$topicarn = New-SNSTopic -Name "myTopic"

# get the queue ARN to inject into the policy; it will be returned
# in the output's QueueARN member but we need to put it into a variable
# so text expansion in the policy string takes effect
$qarn = (Get-SQSQueueAttribute -QueueUrl $qurl -AttributeName "QueueArn").QueueARN

# construct the policy and inject arns
$policy = @"
{
  "Version": "2008-10-17",
  "Id": "$qarn/SQSPOLICY",
  "Statement": [
    {
      "Sid": "1",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "SQS:SendMessage",
      "Resource": "$qarn",
      "Condition": {
        "ArnEquals": {
          "aws:SourceArn": "$topicarn"
        }
      }
    }
  ]
}
"@

# set the policy
Set-SQSQueueAttribute -QueueUrl $qurl -Attribute @{ Policy=$policy }
```

Contoh 2: Contoh ini menetapkan atribut tertentu untuk antrian tertentu.

```
Set-SQSQueueAttribute -Attribute @{"DelaySeconds" = "10"; "MaximumMessageSize" =
  "131072"} -QueueUrl https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyQueue
```

- Untuk detail API, lihat [SetQueueAttributes](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

AWS STS contoh menggunakan Alat untuk PowerShell V4

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan AWS STS.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

Convert-STSAuthorizationMessage

Contoh kode berikut menunjukkan cara menggunakan `Convert-STSAuthorizationMessage`.

Alat untuk PowerShell V4

Contoh 1: Mendekode informasi tambahan yang terkandung dalam konten pesan disandikan yang disediakan yang dikembalikan sebagai tanggapan atas permintaan. Informasi tambahan dikodekan karena rincian status otorisasi dapat merupakan informasi istimewa yang tidak boleh dilihat oleh pengguna yang meminta tindakan.

```
Convert-STSAuthorizationMessage -EncodedMessage "...encoded message..."
```

- Untuk detail API, lihat [DecodeAuthorizationMessage](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-STSFederationToken

Contoh kode berikut menunjukkan cara menggunakan `Get-STSFederationToken`.

Alat untuk PowerShell V4

Contoh 1: Meminta token federasi yang valid selama satu jam menggunakan “Bob” sebagai nama pengguna federasi. Nama ini dapat digunakan untuk mereferensikan nama pengguna federasi dalam kebijakan berbasis sumber daya (seperti kebijakan bucket Amazon S3). Kebijakan IAM yang disediakan, dalam format JSON, digunakan untuk menutupi izin yang tersedia untuk pengguna IAM. Kebijakan yang diberikan tidak dapat memberikan izin lebih dari yang diberikan kepada pengguna yang meminta, dengan izin akhir untuk pengguna federasi menjadi set yang paling ketat berdasarkan persimpangan kebijakan yang disahkan dan kebijakan pengguna IAM.

```
Get-STSFederationToken -Name "Bob" -Policy "...JSON policy..." -DurationInSeconds
3600
```

- Untuk detail API, lihat [GetFederationToken](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-STSSessionToken

Contoh kode berikut menunjukkan cara menggunakan `Get-STSSessionToken`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan sebuah **Amazon.Runtime.AWSCredentials** instance yang berisi kredensial sementara yang valid untuk jangka waktu tertentu. Kredensial yang digunakan untuk meminta kredensial sementara disimpulkan dari default shell saat ini. Untuk menentukan kredensial lainnya, gunakan parameter `-ProfileName` atau `-AccessKey SecretKey` /-.

```
Get-STSSessionToken
```

Output:

AccessKeyId	Expiration
SecretAccessKey	SessionToken
-----	-----
-----	-----
EXAMPLEACCESSKEYID	2/16/2015 9:12:28 PM
examplesecretaccesskey...	SamPleToken.....

Contoh 2: Mengembalikan sebuah **Amazon.Runtime.AWSCredentials** instance yang berisi kredensi sementara yang valid selama satu jam. Kredensi yang digunakan untuk membuat permintaan diperoleh dari profil yang ditentukan.

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile
```

Output:

AccessKeyId	Expiration
SecretAccessKey	SessionToken
-----	-----
-----	-----
EXAMPLEACCESSKEYID	2/16/2015 9:12:28 PM
examplesecretaccesskey...	SamPleTokenN.....

Contoh 3: Mengembalikan **Amazon.Runtime.AWSCredentials** instance yang berisi kredensial sementara yang valid selama satu jam menggunakan nomor identifikasi perangkat MFA yang terkait dengan akun yang kredensialnya ditentukan dalam profil 'myprofile' dan nilai yang diberikan oleh perangkat.

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile -SerialNumber  
YourMFADeviceSerialNumber -TokenCode 123456
```

Output:

AccessKeyId	Expiration
SecretAccessKey	SessionToken
-----	-----
-----	-----
EXAMPLEACCESSKEYID	2/16/2015 9:12:28 PM
examplesecretaccesskey...	SamPleTokenN.....

- Untuk detail API, lihat [GetSessionToken](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Use-STSRole

Contoh kode berikut menunjukkan cara menggunakan `Use-STSRole`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan satu set kredensial sementara (kunci akses, kunci rahasia, dan token sesi) yang dapat digunakan selama satu jam untuk mengakses AWS sumber daya yang biasanya tidak dapat diakses oleh pengguna yang meminta. Kredensial yang dikembalikan memiliki izin yang diizinkan oleh kebijakan akses dari peran yang diasumsikan dan kebijakan yang diberikan (Anda tidak dapat menggunakan kebijakan yang disediakan untuk memberikan izin melebihi yang ditentukan oleh kebijakan akses peran yang diasumsikan).

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo" -  
Policy "...JSON policy..." -DurationInSeconds 3600
```

Contoh 2: Mengembalikan satu set kredensi sementara, berlaku selama satu jam, yang memiliki izin yang sama yang ditentukan dalam kebijakan akses peran yang diasumsikan.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo" -  
DurationInSeconds 3600
```

Contoh 3: Mengembalikan satu set kredensi sementara yang memasok nomor seri dan token yang dihasilkan dari MFA yang terkait dengan kredensial pengguna yang digunakan untuk mengeksekusi cmdlet.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo" -  
DurationInSeconds 3600 -SerialNumber "GAHT12345678" -TokenCode "123456"
```

Contoh 4: Mengembalikan satu set kredensi sementara yang telah mengambil peran yang ditentukan dalam akun pelanggan. Untuk setiap peran yang dapat diasumsikan oleh pihak ketiga, akun pelanggan harus membuat peran menggunakan pengidentifikasi yang harus diteruskan dalam ExternalId parameter - setiap kali peran diasumsikan.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo" -  
DurationInSeconds 3600 -ExternalId "ABC123"
```

- Untuk detail API, lihat [AssumeRole](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Use-STSWebIdentityRole

Contoh kode berikut menunjukkan cara menggunakan `Use-STSWebIdentityRole`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan kumpulan kredensial sementara, berlaku selama satu jam, untuk pengguna yang telah diautentikasi dengan penyedia identitas Login with Amazon. Kredensi mengasumsikan kebijakan akses yang terkait dengan peran yang diidentifikasi oleh peran ARN. Secara opsional, Anda dapat meneruskan kebijakan JSON ke parameter `-Policy` yang selanjutnya menyempurnakan izin akses (Anda tidak dapat memberikan izin lebih dari yang tersedia dalam izin yang terkait dengan peran). Nilai yang diberikan ke `-WebIdentityToken` adalah pengidentifikasi pengguna unik yang dikembalikan oleh penyedia identitas.

```
Use-STSWebIdentityRole -DurationInSeconds 3600 -ProviderId "www.amazon.com"  
-RoleSessionName "app1" -RoleArn "arn:aws:iam::123456789012:role/  
FederatedWebIdentityRole" -WebIdentityToken "Atza...DVI0r1"
```

- Untuk detail API, lihat [AssumeRoleWithWebIdentity](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Dukungan contoh menggunakan Alat untuk PowerShell V4

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan Dukungan.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

Add-ASACommunicationToCase

Contoh kode berikut menunjukkan cara menggunakan `Add-ASACommunicationToCase`.

Alat untuk PowerShell V4

Contoh 1: Menambahkan badan komunikasi email ke kasus yang ditentukan.

```
Add-ASACommunicationToCase -CaseId "case-12345678910-2013-c4c1d2bf33c5cf47" -  
CommunicationBody "Some text about the case"
```

Contoh 2: Menambahkan badan komunikasi email ke kasus yang ditentukan ditambah satu atau lebih alamat email yang terkandung dalam baris CC email.

```
Add-ASACommunicationToCase -CaseId "case-12345678910-2013-c4c1d2bf33c5cf47" -  
CcEmailAddress @"email1@address.com", "email2@address.com") -CommunicationBody  
"Some text about the case"
```

- Untuk detail API, lihat [AddCommunicationToCase](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-ASACase

Contoh kode berikut menunjukkan cara menggunakan Get-ASACase.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan rincian semua kasus dukungan.

```
Get-ASACase
```

Contoh 2: Mengembalikan rincian semua kasus dukungan sejak tanggal dan waktu yang ditentukan.

```
Get-ASACase -AfterTime "2013-09-10T03:06Z"
```

Contoh 3: Mengembalikan rincian 10 kasus dukungan pertama, termasuk yang telah diselesaikan.

```
Get-ASACase -MaxResult 10 -IncludeResolvedCases $true
```

Contoh 4: Mengembalikan rincian kasus dukungan tunggal yang ditentukan.

```
Get-ASACase -CaseIdList "case-12345678910-2013-c4c1d2bf33c5cf47"
```

Contoh 5: Mengembalikan rincian kasus dukungan tertentu.

```
Get-ASACase -CaseIdList @("case-12345678910-2013-c4c1d2bf33c5cf47",  
"case-18929034710-2011-c4fdeabf33c5cf47")
```

- Untuk detail API, lihat [DescribeCases](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-ASACommunication

Contoh kode berikut menunjukkan cara menggunakan `Get-ASACommunication`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan semua komunikasi untuk kasus tertentu.

```
Get-ASACommunication -CaseId "case-12345678910-2013-c4c1d2bf33c5cf47"
```

Contoh 2: Mengembalikan semua komunikasi sejak tengah malam UTC pada 1 Januari 2012 untuk kasus yang ditentukan.

```
Get-ASACommunication -CaseId "case-12345678910-2013-c4c1d2bf33c5cf47" -AfterTime  
"2012-01-10T00:00Z"
```

- Untuk detail API, lihat [DescribeCommunications](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-ASAService

Contoh kode berikut menunjukkan cara menggunakan `Get-ASAService`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan semua kode layanan, nama, dan kategori yang tersedia.

```
Get-ASAService
```

Contoh 2: Mengembalikan nama dan kategori untuk layanan dengan kode yang ditentukan.

```
Get-ASAService -ServiceCodeList "amazon-cloudfront"
```

Contoh 3: Mengembalikan nama dan kategori untuk kode layanan tertentu.

```
Get-ASAService -ServiceCodeList @("amazon-cloudfront", "amazon-cloudwatch")
```

Contoh 4: Mengembalikan nama dan kategori (dalam bahasa Jepang) untuk kode layanan yang ditentukan. Saat ini kode bahasa Inggris ("en") dan Jepang ("ja") didukung.

```
Get-ASAService -ServiceCodeList @("amazon-cloudfront", "amazon-cloudwatch") -  
Language "ja"
```

- Untuk detail API, lihat [DescribeServices](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-ASASeverityLevel

Contoh kode berikut menunjukkan cara menggunakan `Get-ASASeverityLevel`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan daftar tingkat keparahan yang dapat ditetapkan ke kasus AWS Support.

```
Get-ASASeverityLevel
```

Contoh 2: Mengembalikan daftar tingkat keparahan yang dapat ditetapkan ke kasus AWS Support. Nama-nama level dikembalikan dalam bahasa Jepang.

```
Get-ASASeverityLevel -Language "ja"
```

- Untuk detail API, lihat [DescribeSeverityLevels](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-ASATrustedAdvisorCheck

Contoh kode berikut menunjukkan cara menggunakan `Get-ASATrustedAdvisorCheck`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan koleksi cek Trusted Advisor. Anda harus menentukan parameter Bahasa yang dapat menerima "en" untuk output bahasa Inggris atau "ja" untuk output Jepang.

```
Get-ASATrustedAdvisorCheck -Language "en"
```

- Untuk detail API, lihat [DescribeTrustedAdvisorChecks](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-ASATrustedAdvisorCheckRefreshStatus

Contoh kode berikut menunjukkan cara menggunakan `Get-ASATrustedAdvisorCheckRefreshStatus`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan status permintaan penyegaran saat ini untuk pemeriksaan yang ditentukan. Permintaan- `ASATrusted AdvisorCheckRefresh` dapat digunakan untuk meminta agar informasi status cek disegarkan.

```
Get-ASATrustedAdvisorCheckRefreshStatus -CheckId @("checkid1", "checkid2")
```

- Untuk detail API, lihat [DescribeTrustedAdvisorCheckRefreshStatuses](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-ASATrustedAdvisorCheckResult

Contoh kode berikut menunjukkan cara menggunakan `Get-ASATrustedAdvisorCheckResult`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan hasil pemeriksaan Trusted Advisor. Daftar cek Trusted Advisor yang tersedia dapat diperoleh dengan menggunakan `Get-ASATrusted AdvisorChecks`. Outputnya adalah status keseluruhan pemeriksaan, stempel waktu di mana pemeriksaan terakhir dijalankan dan checkid unik untuk pemeriksaan tertentu. Untuk mendapatkan output hasil dalam bahasa Jepang, tambahkan parameter `-Language "ja"`.

```
Get-ASATrustedAdvisorCheckResult -CheckId "checkid1"
```

- Untuk detail API, lihat [DescribeTrustedAdvisorCheckResult](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-ASATrustedAdvisorCheckSummary

Contoh kode berikut menunjukkan cara menggunakan `Get-ASATrustedAdvisorCheckSummary`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan ringkasan terbaru untuk pemeriksaan Trusted Advisor yang ditentukan.

```
Get-ASATrustedAdvisorCheckSummary -CheckId "checkid1"
```

Contoh 2: Mengembalikan ringkasan terbaru untuk pemeriksaan Trusted Advisor yang ditentukan.

```
Get-ASATrustedAdvisorCheckSummary -CheckId @("checkid1", "checkid2")
```

- Untuk detail API, lihat [DescribeTrustedAdvisorCheckSummaries](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-ASACase

Contoh kode berikut menunjukkan cara menggunakan `New-ASACase`.

Alat untuk PowerShell V4

Contoh 1: Membuat kasus baru di AWS Support Center. Nilai untuk `CategoryCode` parameter - `ServiceCode` dan - dapat diperoleh dengan menggunakan `Get-ASAService` cmdlet. Nilai untuk `SeverityCode` parameter - dapat diperoleh dengan menggunakan `Get-ASASeverityLevel` cmdlet. Nilai - `IssueType` parameter dapat berupa “layanan pelanggan” atau “teknis”. Jika berhasil, nomor kasus AWS Support adalah output. Secara default kasus akan ditangani dalam bahasa Inggris, untuk menggunakan bahasa Jepang tambahkan parameter `-Language "ja"`. `CommunicationBody` Parameter `-ServiceCode`, `-CategoryCode`, `-Subjek` dan - adalah wajib.

```
New-ASACase -ServiceCode "amazon-cloudfront" -CategoryCode "APIs" -SeverityCode  
"low" -Subject "subject text" -CommunicationBody "description of the case" -  
CcEmailAddress @("email1@domain.com", "email2@domain.com") -IssueType "technical"
```

- Untuk detail API, lihat [CreateCase](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Request-ASATrustedAdvisorCheckRefresh

Contoh kode berikut menunjukkan cara menggunakan `Request-ASATrustedAdvisorCheckRefresh`.

Alat untuk PowerShell V4

Contoh 1: Meminta penyegaran untuk pemeriksaan Trusted Advisor yang ditentukan.

```
Request-ASATrustedAdvisorCheckRefresh -CheckId "checkid1"
```

- Untuk detail API, lihat [RefreshTrustedAdvisorCheck](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Resolve-ASACase

Contoh kode berikut menunjukkan cara menggunakan `Resolve-ASACase`.

Alat untuk PowerShell V4

Contoh 1: Mengembalikan keadaan awal dari kasus yang ditentukan dan keadaan saat ini setelah panggilan untuk menyelesaikannya selesai.

```
Resolve-ASACase -CaseId "case-12345678910-2013-c4c1d2bf33c5cf47"
```

- Untuk detail API, lihat [ResolveCase](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Contoh Systems Manager menggunakan Alat untuk PowerShell V4

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan Systems Manager.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

Add-SSMResourceTag

Contoh kode berikut menunjukkan cara menggunakan `Add-SSMResourceTag`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memperbarui jendela pemeliharaan dengan tag baru. Tidak ada output jika perintah berhasil. Sintaks yang digunakan oleh contoh ini memerlukan PowerShell versi 3 atau yang lebih baru.

```
$option1 = @{Key="Stack";Value=@"Production"}}
Add-SSMResourceTag -ResourceId "mw-03eb9db42890fb82d" -ResourceType
"MaintenanceWindow" -Tag $option1
```

Contoh 2: Dengan PowerShell versi 2, Anda harus menggunakan `New-Object` untuk membuat setiap tag. Tidak ada output jika perintah berhasil.

```
$tag1 = New-Object Amazon.SimpleSystemsManagement.Model.Tag
$tag1.Key = "Stack"
$tag1.Value = "Production"

Add-SSMResourceTag -ResourceId "mw-03eb9db42890fb82d" -ResourceType
"MaintenanceWindow" -Tag $tag1
```

- Untuk detail API, lihat [AddTagsToResource](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Edit-SSMDocumentPermission

Contoh kode berikut menunjukkan cara menggunakan `Edit-SSMDocumentPermission`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menambahkan izin “bagikan” ke semua akun untuk dokumen. Tidak ada output jika perintah berhasil.

```
Edit-SSMDocumentPermission -Name "RunShellScript" -PermissionType "Share" -
AccountIdsToAdd all
```

Contoh 2: Contoh ini menambahkan izin “bagikan” ke akun tertentu untuk dokumen. Tidak ada output jika perintah berhasil.

```
Edit-SSMDocumentPermission -Name "RunShellScriptNew" -PermissionType "Share" -
AccountIdsToAdd "123456789012"
```

- Untuk detail API, lihat [ModifyDocumentPermission](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMActivation

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMActivation`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memberikan rincian tentang aktivasi di akun Anda.

```
Get-SSMActivation
```

Output:

```
ActivationId      : 08e51e79-1e36-446c-8e63-9458569c1363
CreatedDate       : 3/1/2017 12:01:51 AM
DefaultInstanceName : MyWebServers
Description        :
ExpirationDate     : 3/2/2017 12:01:51 AM
Expired           : False
IamRole            : AutomationRole
RegistrationLimit   : 10
RegistrationsCount : 0
```

- Untuk detail API, lihat [DescribeActivations](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMAssociation

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMAssociation`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan hubungan antara instance dan dokumen.

```
Get-SSMAssociation -InstanceId "i-0000293ffd8c57862" -Name "AWS-UpdateSSMAgent"
```

Output:

```
Name                : AWS-UpdateSSMAgent
InstanceId           : i-0000293ffd8c57862
Date                : 2/23/2017 6:55:22 PM
Status.Name         : Pending
Status.Date         : 2/20/2015 8:31:11 AM
Status.Message      : temp_status_change
Status.AdditionalInfo : Additional-Config-Needed
```

- Untuk detail API, lihat [DescribeAssociation](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMAssociationExecution

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMAssociationExecution`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan eksekusi untuk ID asosiasi yang disediakan

```
Get-SSMAssociationExecution -AssociationId 123a45a0-c678-9012-3456-78901234db5e
```

Output:

```
AssociationId       : 123a45a0-c678-9012-3456-78901234db5e
AssociationVersion   : 2
CreatedTime         : 3/2/2019 8:53:29 AM
DetailedStatus      :
ExecutionId         : 123a45a0-c678-9012-3456-78901234db5e
LastExecutionDate   : 1/1/0001 12:00:00 AM
ResourceCountByStatus : {Success=4}
Status              : Success
```

- Untuk detail API, lihat [DescribeAssociationExecutions](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMAssociationExecutionTarget

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMAssociationExecutionTarget`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menampilkan ID sumber daya dan status eksekusi yang merupakan bagian dari target eksekusi asosiasi

```
Get-SSMAssociationExecutionTarget -AssociationId 123a45a0-
c678-9012-3456-78901234db5e -ExecutionId 123a45a0-c678-9012-3456-78901234db5e |
Select-Object ResourceId, Status
```

Output:

ResourceId	Status
-----	-----
i-0b1b2a3456f7a890b	Success
i-01c12a45d6fc7a89f	Success
i-0a1caf234f56d7dc8	Success
i-012a3fd45af6dbcf	Failed
i-0ddc1df23c4a5fb67	Success

Contoh 2: Perintah ini memeriksa eksekusi tertentu dari otomatisasi tertentu sejak kemarin, di mana dokumen perintah dikaitkan. Selanjutnya memeriksa apakah eksekusi asosiasi gagal, dan jika demikian, itu akan menampilkan rincian pemanggilan perintah untuk eksekusi bersama dengan id instance

```
$AssociationExecution= Get-SSMAssociationExecutionTarget -
AssociationId 1c234567-890f-1aca-a234-5a678d901cb0 -ExecutionId
12345ca12-3456-2345-2b45-23456789012 |
Where-Object {$_.LastExecutionDate -gt (Get-Date -Hour 00 -Minute
00).AddDays(-1)}

foreach ($execution in $AssociationExecution) {
    if($execution.Status -ne 'Success'){
        Write-Output "There was an issue executing the association
($($execution.AssociationId) on $($execution.ResourceId)"
```

```
Get-SSMCommandInvocation -CommandId $execution.OutputSource.OutputSourceId -
Detail:$true | Select-Object -ExpandProperty CommandPlugins
}
}
```

Output:

```
There was an issue executing the association 1c234567-890f-1aca-a234-5a678d901cb0 on
i-0a1caf234f56d7dc8
```

```
Name                : aws:runPowerShellScript
Output              :
                    -----ERROR-----
                    failed to run commands: exit status 1
OutputS3BucketName  :
OutputS3KeyPrefix   :
OutputS3Region      : eu-west-1
ResponseCode        : 1
ResponseFinishDateTime : 5/29/2019 11:04:49 AM
ResponseStartDateTime  : 5/29/2019 11:04:49 AM
StandardErrorUrl     :
StandardOutputUrl    :
Status              : Failed
StatusDetails       : Failed
```

- Untuk detail API, lihat [DescribeAssociationExecutionTargets](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMAssociationList

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMAssociationList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan semua asosiasi untuk sebuah instance. Sintaks yang digunakan oleh contoh ini memerlukan PowerShell versi 3 atau yang lebih baru.

```
$filter1 = @{Key="InstanceId";Value=@"i-0000293ffd8c57862"}
Get-SSMAssociationList -AssociationFilterList $filter1
```

Output:

```

AssociationId      : d8617c07-2079-4c18-9847-1655fc2698b0
DocumentVersion    :
InstanceId         : i-0000293ffd8c57862
LastExecutionDate  : 2/20/2015 8:31:11 AM
Name               : AWS-UpdateSSMAgent
Overview           : Amazon.SimpleSystemsManagement.Model.AssociationOverview
ScheduleExpression :
Targets            : {InstanceIds}

```

Contoh 2: Contoh ini mencantumkan semua asosiasi untuk dokumen konfigurasi. Sintaks yang digunakan oleh contoh ini memerlukan PowerShell versi 3 atau yang lebih baru.

```

$filter2 = @{Key="Name";Value=@"AWS-UpdateSSMAgent"}
Get-SSMAssociationList -AssociationFilterList $filter2

```

Output:

```

AssociationId      : d8617c07-2079-4c18-9847-1655fc2698b0
DocumentVersion    :
InstanceId         : i-0000293ffd8c57862
LastExecutionDate  : 2/20/2015 8:31:11 AM
Name               : AWS-UpdateSSMAgent
Overview           : Amazon.SimpleSystemsManagement.Model.AssociationOverview
ScheduleExpression :
Targets            : {InstanceIds}

```

Contoh 3: Dengan PowerShell versi 2, Anda harus menggunakan New-Object untuk membuat setiap filter.

```

$filter1 = New-Object Amazon.SimpleSystemsManagement.Model.AssociationFilter
$filter1.Key = "InstanceId"
$filter1.Value = "i-0000293ffd8c57862"

Get-SSMAssociationList -AssociationFilterList $filter1

```

Output:

```

AssociationId      : d8617c07-2079-4c18-9847-1655fc2698b0
DocumentVersion    :
InstanceId         : i-0000293ffd8c57862
LastExecutionDate  : 2/20/2015 8:31:11 AM

```

```

Name           : AWS-UpdateSSMAgent
Overview       : Amazon.SimpleSystemsManagement.Model.AssociationOverview
ScheduleExpression :
Targets        : {InstanceIds}

```

- Untuk detail API, lihat [ListAssociations](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMAssociationVersionList

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMAssociationVersionList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil semua versi asosiasi yang disediakan.

```
Get-SSMAssociationVersionList -AssociationId 123a45a0-c678-9012-3456-78901234db5e
```

Output:

```

AssociationId      : 123a45a0-c678-9012-3456-78901234db5e
AssociationName     :
AssociationVersion  : 2
ComplianceSeverity :
CreatedDate        : 3/12/2019 9:21:01 AM
DocumentVersion    :
MaxConcurrency     :
MaxErrors          :
Name               : AWS-GatherSoftwareInventory
OutputLocation     :
Parameters         : {}
ScheduleExpression :
Targets            : {InstanceIds}

AssociationId      : 123a45a0-c678-9012-3456-78901234db5e
AssociationName     : test-case-1234567890
AssociationVersion  : 1
ComplianceSeverity :
CreatedDate        : 3/2/2019 8:53:29 AM
DocumentVersion    :
MaxConcurrency     :
MaxErrors          :
Name               : AWS-GatherSoftwareInventory
OutputLocation     :

```

```
Parameters      : {}
ScheduleExpression : rate(30minutes)
Targets         : {InstanceIds}
```

- Untuk detail API, lihat [ListAssociationVersions](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMAutomationExecution

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMAutomationExecution`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menampilkan detail Eksekusi Otomasi.

```
Get-SSMAutomationExecution -AutomationExecutionId "4105a4fc-
f944-11e6-9d32-8fb2db27a909"
```

Output:

```
AutomationExecutionId      : 4105a4fc-f944-11e6-9d32-8fb2db27a909
AutomationExecutionStatus  : Failed
DocumentName               : AWS-UpdateLinuxAmi
DocumentVersion            : 1
ExecutionEndTime           : 2/22/2017 9:17:08 PM
ExecutionStartTime         : 2/22/2017 9:17:02 PM
FailureMessage             : Step launchInstance failed maximum allowed times. You
                           are not authorized to perform this operation. Encoded
                           authorization failure message:
                           B_V2QyyN7NhSZQYpmVzpEc4oSnj2GLTNYnXUHsTbqJkNMoDgubmbtthLmZyaiUYek0RIrA42-
                           fv1x-04q5Fjff6glh
                           Yb6TI5b0GQeeNrpwNvpDzm0-
                           PSR1swlAbg9fdM9BcNjyrznspUkWpuKu9EC10u6v30XU1KC9nZ7mPlWMFZNkSioQqpWWEvMw-
                           GZktsQzm67q0hUhbN0LWYhbS
                           pkfiqzY-5nw3S0obx30fhd3EJa50_-
                           GjV_a0nFXQJa70ik40bF0rEh3MtCSbrQT6--DvFy_FQ8TKvkIXadyVskeJI84X0F5WmA60f1pi5GI08i-
                           nRfZS6oDeU
                           gELBjjoFKD8s3L2aI0B6umWVxnQ0jqhQRxwJ53b54sZJ2PW3v_mtg9-
                           q0CK0ezS3xfh_y0ilaUG0AZG-xjQFuvU_JZedWpla3xi-MZsmb1AifBI
                           (Service: AmazonEC2; Status Code: 403; Error Code:
                           UnauthorizedOperation; Request ID:
                           6a002f94-ba37-43fd-99e6-39517715fce5)
```

```

Outputs          : {[createImage.ImageId,
  Amazon.Runtime.Internal.Util.AlwaysSendList`1[System.String]]}
Parameters       : {[AutomationAssumeRole,
  Amazon.Runtime.Internal.Util.AlwaysSendList`1[System.String]], [InstanceIamRole,
  Amazon.Runtime.Internal.Util.AlwaysSendList`1[System.String]], [SourceAmiId,
  Amazon.Runtime.Internal.Util.AlwaysSendList`1[System.String]]}
StepExecutions   : {launchInstance, updateOSSoftware, stopInstance,
  createImage...}

```

Contoh 2: Contoh ini mencantumkan detail langkah untuk id eksekusi otomatisasi yang diberikan

```

Get-SSMAutomationExecution -AutomationExecutionId e1d2bad3-4567-8901-
ae23-456c7c8901be | Select-Object -ExpandProperty StepExecutions | Select-Object
  StepName, Action, StepStatus, ValidNextSteps

```

Output:

StepName	Action	StepStatus	ValidNextSteps
-----	-----	-----	-----
LaunchInstance	aws:runInstances	Success	{OSCompatibilityCheck}
OSCompatibilityCheck	aws:runCommand	Success	{RunPreUpdateScript}
RunPreUpdateScript	aws:runCommand	Success	{UpdateEC2Config}
UpdateEC2Config	aws:runCommand	Cancelled	{}
UpdateSSMAgent	aws:runCommand	Pending	{}
UpdateAWSPVDriver	aws:runCommand	Pending	{}
UpdateAWSEnaNetworkDriver	aws:runCommand	Pending	{}
UpdateAWSNVMe	aws:runCommand	Pending	{}
InstallWindowsUpdates	aws:runCommand	Pending	{}
RunPostUpdateScript	aws:runCommand	Pending	{}
RunSysprepGeneralize	aws:runCommand	Pending	{}
StopInstance	aws:changeInstanceState	Pending	{}
CreateImage	aws:createImage	Pending	{}
TerminateInstance	aws:changeInstanceState	Pending	{}

- Untuk detail API, lihat [GetAutomationExecution](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMAutomationExecutionList

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMAutomationExecutionList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan semua Eksekusi Otomasi aktif dan dihentikan yang terkait dengan akun Anda.

```
Get-SSMAutomationExecutionList
```

Output:

```
AutomationExecutionId      : 4105a4fc-f944-11e6-9d32-8fb2db27a909
AutomationExecutionStatus  : Failed
DocumentName               : AWS-UpdateLinuxAmi
DocumentVersion            : 1
ExecutedBy                 : admin
ExecutionEndTime           : 2/22/2017 9:17:08 PM
ExecutionStartTime        : 2/22/2017 9:17:02 PM
LogFile                   :
Outputs                   : {[createImage.ImageId,
Amazon.Runtime.Internal.Util.AlwaysSendList`1[System.String]]}
```

Contoh 2: Contoh ini menampilkan ExecutionId, document, eksekusi start/end timestamp untuk eksekusi dengan selain 'Success' AutomationExecutionStatus

```
Get-SSMAutomationExecutionList | Where-Object AutomationExecutionStatus
-ne "Success" | Select-Object AutomationExecutionId, DocumentName,
AutomationExecutionStatus, ExecutionStartTime, ExecutionEndTime | Format-Table -
AutoSize
```

Output:

AutomationExecutionId	DocumentName	AutomationExecutionStatus	ExecutionStartTime	ExecutionEndTime
e1d2bad3-4567-8901-ae23-456c7c8901be	AWS-UpdateWindowsAmi	Cancelled	4/16/2019 5:37:04 AM	4/16/2019 5:47:29 AM
61234567-a7f8-90e1-2b34-567b8bf9012c	Fixed-UpdateAmi	Cancelled	4/16/2019 5:33:04 AM	4/16/2019 5:40:15 AM
91234d56-7e89-0ac1-2aee-34ea5d6a7c89	AWS-UpdateWindowsAmi		4/16/2019 5:22:46 AM	4/16/2019 5:27:29 AM

Failed

- Untuk detail API, lihat [DescribeAutomationExecutions](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMAutomationStepExecution

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMAutomationStepExecution`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menampilkan informasi tentang semua eksekusi langkah aktif dan dihentikan dalam alur kerja Otomasi.

```
Get-SSMAutomationStepExecution -AutomationExecutionId e1d2bad3-4567-8901-ae23-456c7c8901be | Select-Object StepName, Action, StepStatus
```

Output:

StepName	Action	StepStatus
-----	-----	-----
LaunchInstance	aws:runInstances	Success
OSCompatibilityCheck	aws:runCommand	Success
RunPreUpdateScript	aws:runCommand	Success
UpdateEC2Config	aws:runCommand	Cancelled
UpdateSSMAgent	aws:runCommand	Pending
UpdateAWSPVDriver	aws:runCommand	Pending
UpdateAWSEnaNetworkDriver	aws:runCommand	Pending
UpdateAWSNVMe	aws:runCommand	Pending
InstallWindowsUpdates	aws:runCommand	Pending
RunPostUpdateScript	aws:runCommand	Pending
RunSysprepGeneralize	aws:runCommand	Pending
StopInstance	aws:changeInstanceState	Pending
CreateImage	aws:createImage	Pending
TerminateInstance	aws:changeInstanceState	Pending

- Untuk detail API, lihat [DescribeAutomationStepExecutions](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMAvailablePatch

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMAvailablePatch`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan semua patch yang tersedia untuk Windows Server 2012 yang memiliki tingkat keparahan MSRC Critical. Sintaks yang digunakan oleh contoh ini memerlukan PowerShell versi 3 atau yang lebih baru.

```
$filter1 = @{"Key"="PRODUCT";Values=@"WindowsServer2012"}
$filter2 = @{"Key"="MSRC_SEVERITY";Values=@"Critical"}

Get-SSMAvailablePatch -Filter $filter1,$filter2
```

Output:

```
Classification : SecurityUpdates
ContentUrl      : https://support.microsoft.com/en-us/kb/2727528
Description     : A security issue has been identified that could allow an
                  unauthenticated remote attacker to compromise your system and gain control
                  over it. You can help protect your system by installing this update
                  from Microsoft. After you install this update, you may have to
                  restart your system.
Id              : 1eb507be-2040-4eeb-803d-abc55700b715
KbNumber        : KB2727528
Language        : All
MsrcNumber      : MS12-072
MsrcSeverity    : Critical
Product         : WindowsServer2012
ProductFamily   : Windows
ReleaseDate     : 11/13/2012 6:00:00 PM
Title           : Security Update for Windows Server 2012 (KB2727528)
Vendor         : Microsoft
...
```

Contoh 2: Dengan PowerShell versi 2, Anda harus menggunakan New-Object untuk membuat setiap filter.

```
$filter1 = New-Object Amazon.SimpleSystemsManagement.Model.PatchOrchestratorFilter
$filter1.Key = "PRODUCT"
$filter1.Values = "WindowsServer2012"
$filter2 = New-Object Amazon.SimpleSystemsManagement.Model.PatchOrchestratorFilter
$filter2.Key = "MSRC_SEVERITY"
$filter2.Values = "Critical"
```

```
Get-SSMAvailablePatch -Filter $filter1,$filter2
```

Contoh 3: Contoh ini mengambil semua pembaruan yang dirilis dalam 20 hari terakhir dan berlaku untuk produk yang cocok dengan 2019 WindowsServer

```
Get-SSMAvailablePatch | Where-Object ReleaseDate -ge (Get-Date).AddDays(-20) |
Where-Object Product -eq "WindowsServer2019" | Select-Object ReleaseDate, Product,
Title
```

Output:

ReleaseDate	Product	Title
-----	-----	-----
4/9/2019 5:00:12 PM	WindowsServer2019	2019-04 Security Update for Adobe Flash Player for Windows Server 2019 for x64-based Systems (KB4493478)
4/9/2019 5:00:06 PM	WindowsServer2019	2019-04 Cumulative Update for Windows Server 2019 for x64-based Systems (KB4493509)
4/2/2019 5:00:06 PM	WindowsServer2019	2019-03 Servicing Stack Update for Windows Server 2019 for x64-based Systems (KB4493510)

- Untuk detail API, lihat [DescribeAvailablePatches](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMCommand

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMCommand`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan semua perintah yang diminta.

```
Get-SSMCommand
```

Output:

```
CommandId      : 4b75a163-d39a-4d97-87c9-98ae52c6be35
Comment        : Apply association with id at update time: 4cc73e42-
d5ae-4879-84f8-57e09c0efcd0
```

```

CompletedCount      : 1
DocumentName        : AWS-RefreshAssociation
ErrorCount           : 0
ExpiresAfter         : 2/24/2017 3:19:08 AM
InstanceIds          : {i-0cb2b964d3e14fd9f}
MaxConcurrency       : 50
MaxErrors            : 0
NotificationConfig    : Amazon.SimpleSystemsManagement.Model.NotificationConfig
OutputS3BucketName   :
OutputS3KeyPrefix    :
OutputS3Region       :
Parameters           : {[associationIds,
    Amazon.Runtime.Internal.Util.AlwaysSendList`1[System.String]]}
RequestedDateTime    : 2/24/2017 3:18:08 AM
ServiceRole          :
Status               : Success
StatusDetails        : Success
TargetCount          : 1
Targets              : {}

```

Contoh 2: Contoh ini mendapatkan status perintah tertentu.

```
Get-SSMCommand -CommandId "4b75a163-d39a-4d97-87c9-98ae52c6be35"
```

Contoh 3: Contoh ini mengambil semua perintah SSM yang dipanggil setelah 2019-04-01T00:00:00 Z

```
Get-SSMCommand -Filter @{Key="InvokedAfter";Value="2019-04-01T00:00:00Z"} | Select-Object CommandId, DocumentName, Status, RequestedDateTime | Sort-Object -Property RequestedDateTime -Descending
```

Output:

CommandId RequestedDateTime ----- -----	DocumentName -----	Status -----
edb1b23e-456a-7adb-aef8-90e-012ac34f 5:45:23 AM	AWS-RunPowerShellScript	Cancelled 4/16/2019
1a2dc3fb-4567-890d-a1ad-234b5d6bc7d9 9:19:42 AM	AWS-ConfigureAWSPackage	Success 4/6/2019

12c3456c-7e90-4f12-1232-1234f5b67893	KT-Retrieve-Cloud-Type-Win	Failed	4/2/2019 4:13:07 AM
fe123b45-240c-4123-a2b3-234bdd567ecf	AWS-RunInspecChecks	Failed	4/1/2019 2:27:31 PM
1eb23aa4-567d-4123-12a3-4c1c2ab34561	AWS-RunPowerShellScript	Success	4/1/2019 1:05:55 PM
1c2f3bb4-ee12-4bc1-1a23-12345eea123e	AWS-RunInspecChecks	Failed	4/1/2019 11:13:09 AM

- Untuk detail API, lihat [ListCommands](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMCommandInvocation

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMCommandInvocation`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan semua pemanggilan perintah.

```
Get-SSMCommandInvocation -CommandId "b8eac879-0541-439d-94ec-47a80d554f44" -Detail $true
```

Output:

```
CommandId           : b8eac879-0541-439d-94ec-47a80d554f44
CommandPlugins      : {aws:runShellScript}
Comment             : IP config
DocumentName        : AWS-RunShellScript
InstanceId           : i-0cb2b964d3e14fd9f
InstanceName        :
NotificationConfig  : Amazon.SimpleSystemsManagement.Model.NotificationConfig
RequestedDateTime   : 2/22/2017 8:13:16 PM
ServiceRole         :
StandardErrorUrl     :
StandardOutputUrl    :
Status              : Success
StatusDetails       : Success
TraceOutput         :
```

Contoh 2: Contoh ini mencantumkan pemanggilan perintah id `CommandPlugins e1eb2e3c-ed4c-5123-45c1-234f5612345f`

```
Get-SSMCommandInvocation -CommandId e1eb2e3c-ed4c-5123-45c1-234f5612345f -Detail:
$true | Select-Object -ExpandProperty CommandPlugins
```

Output:

```
Name                : aws:runPowerShellScript
Output              : Completed 17.7 KiB/17.7 KiB (40.1 KiB/s) with 1 file(s)
                    remainingdownload: s3://dd-aess-r-ctmer/KUM0.png to ..\..\programdata\KUM0.png
                    kumo available

OutputS3BucketName  :
OutputS3KeyPrefix   :
OutputS3Region      : eu-west-1
ResponseCode        : 0
ResponseFinishDateTime : 4/3/2019 11:53:23 AM
ResponseStartDateTime : 4/3/2019 11:53:21 AM
StandardErrorUrl     :
StandardOutputUrl    :
Status              : Success
StatusDetails       : Success
```

- Untuk detail API, lihat [ListCommandInvocations](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMCommandInvocationDetail

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMCommandInvocationDetail`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menampilkan rincian perintah yang dijalankan pada sebuah instance.

```
Get-SSMCommandInvocationDetail -InstanceId "i-0cb2b964d3e14fd9f" -CommandId
"b8eac879-0541-439d-94ec-47a80d554f44"
```

Output:

```
CommandId           : b8eac879-0541-439d-94ec-47a80d554f44
Comment             : IP config
DocumentName        : AWS-RunShellScript
ExecutionElapsedTime : PT0.004S
```

```

ExecutionEndDateTime : 2017-02-22T20:13:16.651Z
ExecutionStartDateTime : 2017-02-22T20:13:16.651Z
InstanceId           : i-0cb2b964d3e14fd9f
PluginName           : aws:runShellScript
ResponseCode         : 0
StandardErrorContent :
StandardErrorUrl      :
StandardOutputContent :
StandardOutputUrl     :
Status               : Success
StatusDetails        : Success

```

- Untuk detail API, lihat [GetCommandInvocation](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMComplianceItemList

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMComplianceItemList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan daftar item kepatuhan untuk id dan tipe sumber daya yang diberikan, memfilter tipe kepatuhan menjadi 'Asosiasi'

```

Get-SSMComplianceItemList -ResourceId i-1a2caf345f67d0dc2 -ResourceType
ManagedInstance -Filter @{Key="ComplianceType";Values="Association"}

```

Output:

```

ComplianceType : Association
Details        : {[DocumentName, AWS-GatherSoftwareInventory], [DocumentVersion,
1]}
ExecutionSummary : Amazon.SimpleSystemsManagement.Model.ComplianceExecutionSummary
Id              : 123a45a1-c234-1234-1245-67891236db4e
ResourceId     : i-1a2caf345f67d0dc2
ResourceType   : ManagedInstance
Severity       : UNSPECIFIED
Status        : COMPLIANT
Title         :

```

- Untuk detail API, lihat [ListComplianceItems](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMComplianceSummaryList

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMComplianceSummaryList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan jumlah ringkasan sumber daya yang sesuai dan tidak sesuai untuk semua jenis kepatuhan.

```
Get-SSMComplianceSummaryList
```

Output:

```
ComplianceType CompliantSummary
NonCompliantSummary
-----
-----
FleetTotal      Amazon.SimpleSystemsManagement.Model.CompliantSummary
Amazon.SimpleSystemsManagement.Model.NonCompliantSummary
Association      Amazon.SimpleSystemsManagement.Model.CompliantSummary
Amazon.SimpleSystemsManagement.Model.NonCompliantSummary
Custom:InSpec    Amazon.SimpleSystemsManagement.Model.CompliantSummary
Amazon.SimpleSystemsManagement.Model.NonCompliantSummary
Patch            Amazon.SimpleSystemsManagement.Model.CompliantSummary
Amazon.SimpleSystemsManagement.Model.NonCompliantSummary
```

- Untuk detail API, lihat [ListComplianceSummaries](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMConnectionStatus

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMConnectionStatus`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil status koneksi Session Manager untuk sebuah instance untuk menentukan apakah terhubung dan siap menerima koneksi Session Manager.

```
Get-SSMConnectionStatus -Target i-0a1caf234f12d3dc4
```

Output:

Status	Target
-----	-----
Connected	i-0a1caf234f12d3dc4

- Untuk detail API, lihat [GetConnectionStatus](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMDefaultPatchBaseline

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMDefaultPatchBaseline`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menampilkan baseline patch default.

```
Get-SSMDefaultPatchBaseline
```

Output:

```
arn:aws:ssm:us-west-2:123456789012:patchbaseline/pb-04fb4ae6142167966
```

- Untuk detail API, lihat [GetDefaultPatchBaseline](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMDeployablePatchSnapshotForInstance

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMDeployablePatchSnapshotForInstance`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menampilkan snapshot saat ini untuk baseline patch yang digunakan oleh Instance. Perintah ini harus dijalankan dari instance menggunakan kredensi instance. Untuk memastikannya menggunakan kredensi instance, contoh meneruskan **Amazon.Runtime.InstanceProfileAWSCredentials** objek ke parameter `Credentials`.

```
$credentials = [Amazon.Runtime.InstanceProfileAWSCredentials]::new()
Get-SSMDeployablePatchSnapshotForInstance -SnapshotId "4681775b-098f-4435-
a956-0ef33373ac11" -InstanceId "i-0cb2b964d3e14fd9f" -Credentials $credentials
```

Output:

InstanceId	SnapshotDownloadUrl
i-0cb2b964d3e14fd9f	https://patch-baseline-snapshot-us-west-2.s3-us-west-2.amazonaws.com/853d0d3db0f0cafe...1692/4681775b-098f-4435...

Contoh 2: Contoh ini menunjukkan cara mendapatkan yang lengkap SnapshotDownloadUrl. Perintah ini harus dijalankan dari instance menggunakan kredensi instance. Untuk memastikannya menggunakan kredensi instance, contoh mengonfigurasi PowerShell sesi untuk menggunakan objek. **Amazon.Runtime.InstanceProfileAWSCredentials**

```
Set-AWSCredential -Credential
([Amazon.Runtime.InstanceProfileAWSCredentials]::new())
(Get-SSMDeployablePatchSnapshotForInstance -SnapshotId "4681775b-098f-4435-a956-0ef33373ac11" -InstanceId "i-0cb2b964d3e14fd9f").SnapshotDownloadUrl
```

Output:

```
https://patch-baseline-snapshot-us-west-2.s3-us-west-2.amazonaws.com/853d0d3db0f0cafe...
```

- Untuk detail API, lihat [GetDeployablePatchSnapshotForInstance](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMDocument

Contoh kode berikut menunjukkan cara menggunakan Get-SSMDocument.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan isi dokumen.

```
Get-SSMDocument -Name "RunShellScript"
```

Output:

```
Content
-----
```

```
{...
```

Contoh 2: Contoh ini menampilkan isi lengkap dokumen.

```
(Get-SSMDocument -Name "RunShellScript").Content
{
  "schemaVersion":"2.0",
  "description":"Run an updated script",
  "parameters":{
    "commands":{
      "type":"StringList",
      "description":"(Required) Specify a shell script or a command to run.",
      "minItems":1,
      "displayType":"textarea"
    }
  },
  "mainSteps":[
    {
      "action":"aws:runShellScript",
      "name":"runShellScript",
      "inputs":{
        "commands":"{{ commands }}"
      }
    },
    {
      "action":"aws:runPowerShellScript",
      "name":"runPowerShellScript",
      "inputs":{
        "commands":"{{ commands }}"
      }
    }
  ]
}
```

- Untuk detail API, lihat [GetDocument](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMDocumentDescription

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMDocumentDescription`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan informasi tentang dokumen.

```
Get-SSMDocumentDescription -Name "RunShellScript"
```

Output:

```
CreatedDate      : 2/24/2017 5:25:13 AM
DefaultVersion   : 1
Description      : Run an updated script
DocumentType     : Command
DocumentVersion  : 1
Hash             : f775e5df4904c6fa46686c4722fae9de1950dace25cd9608ff8d622046b68d9b
HashType        : Sha256
LatestVersion    : 1
Name             : RunShellScript
Owner           : 123456789012
Parameters       : {commands}
PlatformTypes    : {Linux}
SchemaVersion    : 2.0
Sha1             :
Status          : Active
```

- Untuk detail API, lihat [DescribeDocument](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMDocumentList

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMDocumentList`.

Alat untuk PowerShell V4

Contoh 1: Daftar semua dokumen konfigurasi di akun Anda.

```
Get-SSMDocumentList
```

Output:

```
DocumentType     : Command
DocumentVersion   : 1
Name             : AWS-ApplyPatchBaseline
Owner            : Amazon
PlatformTypes     : {Windows}
SchemaVersion     : 1.2
```

```

DocumentType      : Command
DocumentVersion   : 1
Name              : AWS-ConfigureAWSPackage
Owner             : Amazon
PlatformTypes     : {Windows, Linux}
SchemaVersion     : 2.0

DocumentType      : Command
DocumentVersion   : 1
Name              : AWS-ConfigureCloudWatch
Owner             : Amazon
PlatformTypes     : {Windows}
SchemaVersion     : 1.2
...

```

Contoh 2: Contoh ini mengambil semua dokumen otomatisasi dengan pencocokan nama 'Platform'

```

Get-SSMDocumentList -DocumentFilterList @{Key="DocumentType";Value="Automation"} |
Where-Object Name -Match "Platform"

```

Output:

```

DocumentFormat    : JSON
DocumentType      : Automation
DocumentVersion   : 7
Name              : KT-Get-Platform
Owner             : 987654123456
PlatformTypes     : {Windows, Linux}
SchemaVersion     : 0.3
Tags              : {}
TargetType        :
VersionName       :

```

- Untuk detail API, lihat [ListDocuments](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMDocumentPermission

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMDocumentPermission`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan semua versi untuk dokumen.

```
Get-SSMDocumentVersionList -Name "RunShellScript"
```

Output:

CreatedDate	DocumentVersion	IsDefaultVersion	Name
-----	-----	-----	----
2/24/2017 5:25:13 AM	1	True	RunShellScript

- Untuk detail API, lihat [DescribeDocumentPermission](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMDocumentVersionList

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMDocumentVersionList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan semua versi untuk dokumen.

```
Get-SSMDocumentVersionList -Name "AWS-UpdateSSMAgent"
```

Output:

```

CreatedDate      : 6/1/2021 5:19:10 PM
DocumentFormat   : JSON
DocumentVersion  : 1
IsDefaultVersion : True
Name             : AWS-UpdateSSMAgent
Status          : Active
  
```

- Untuk detail API, lihat [ListDocumentVersions](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMEffectiveInstanceAssociationList

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMEffectiveInstanceAssociationList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjelaskan asosiasi efektif untuk sebuah instance.

```
Get-SSMEffectiveInstanceAssociationList -InstanceId "i-0000293ffd8c57862" -MaxResult 5
```

Output:

AssociationId	Content
-----	-----
d8617c07-2079-4c18-9847-1655fc2698b0	{...

Contoh 2: Contoh ini menampilkan isi asosiasi efektif untuk sebuah instance.

```
(Get-SSMEffectiveInstanceAssociationList -InstanceId "i-0000293ffd8c57862" -MaxResult 5).Content
```

Output:

```
{
  "schemaVersion": "1.2",
  "description": "Update the Amazon SSM Agent to the latest version or specified version.",
  "parameters": {
    "version": {
      "default": "",
      "description": "(Optional) A specific version of the Amazon SSM Agent to install. If not specified, the agent will be updated to the latest version.",
      "type": "String"
    },
    "allowDowngrade": {
      "default": "false",
      "description": "(Optional) Allow the Amazon SSM Agent service to be downgraded to an earlier version. If set"
    }
  }
}
```

```

to false, the service can be upgraded to newer versions only (default). If set to
true, specify the earlier version.",
    "type": "String",
    "allowedValues": [
        "true",
        "false"
    ]
},
"runtimeConfig": {
    "aws:updateSsmAgent": {
        "properties": [
            {
                "agentName": "amazon-ssm-agent",
                "source": "https://s3.{Region}.amazonaws.com/amazon-ssm-{Region}/
ssm-agent-manifest.json",
                "allowDowngrade": "{{ allowDowngrade }}",
                "targetVersion": "{{ version }}"
            }
        ]
    }
}
}
}

```

- Untuk detail API, lihat [DescribeEffectiveInstanceAssociations](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMEffectivePatchesForPatchBaseline

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMEffectivePatchesForPatchBaseline`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan semua baseline patch, dengan daftar hasil maksimal 1.

```
Get-SSMEffectivePatchesForPatchBaseline -BaselineId "pb-0a2f1059b670ebd31" -
MaxResult 1
```

Output:

Patch	PatchStatus
-------	-------------

```

-----
Amazon.SimpleSystemsManagement.Model.Patch
Amazon.SimpleSystemsManagement.Model.PatchStatus

```

Contoh 2: Contoh ini menampilkan status patch untuk semua baseline patch, dengan daftar hasil maksimal 1.

```

(Get-SSMEffectivePatchesForPatchBaseline -BaselineId "pb-0a2f1059b670ebd31" -
MaxResult 1).PatchStatus

```

Output:

```

ApprovalDate      DeploymentStatus
-----
12/21/2010 6:00:00 PM APPROVED

```

- Untuk detail API, lihat [DescribeEffectivePatchesForPatchBaseline](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMInstanceAssociationsStatus

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMInstanceAssociationsStatus`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menunjukkan rincian asosiasi untuk sebuah instance.

```

Get-SSMInstanceAssociationsStatus -InstanceId "i-0000293ffd8c57862"

```

Output:

```

AssociationId      : d8617c07-2079-4c18-9847-1655fc2698b0
DetailedStatus     : Pending
DocumentVersion    : 1
ErrorCode          :
ExecutionDate      : 2/20/2015 8:31:11 AM
ExecutionSummary   : temp_status_change
InstanceId         : i-0000293ffd8c57862
Name              : AWS-UpdateSSMAgent

```

```
OutputUrl      :
Status         : Pending
```

Contoh 2: Contoh ini memeriksa status asosiasi instance untuk id instance yang diberikan dan selanjutnya, menampilkan status eksekusi asosiasi tersebut

```
Get-SSMInstanceAssociationsStatus -InstanceId i-012e3cb4df567e8aa | ForEach-Object
{Get-SSMAssociationExecution -AssociationId .AssociationId}
```

Output:

```
AssociationId      : 512a34a5-c678-1234-1234-12345678db9e
AssociationVersion  : 2
CreatedTime        : 3/2/2019 8:53:29 AM
DetailedStatus     :
ExecutionId        : 512a34a5-c678-1234-1234-12345678db9e
LastExecutionDate  : 1/1/0001 12:00:00 AM
ResourceCountByStatus : {Success=9}
Status             : Success
```

- Untuk detail API, lihat [DescribeInstanceAssociationsStatus](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMInstanceInformation

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMInstanceInformation`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menunjukkan detail dari setiap instance Anda.

```
Get-SSMInstanceInformation
```

Output:

```
ActivationId      :
AgentVersion      : 2.0.672.0
AssociationOverview :
  Amazon.SimpleSystemsManagement.Model.InstanceAggregatedAssociationOverview
AssociationStatus  : Success
```

```

ComputerName           : ip-172-31-44-222.us-west-2.compute.internal
IamRole                :
InstanceId             : i-0cb2b964d3e14fd9f
IPAddress              : 172.31.44.222
IsLatestVersion        : True
LastAssociationExecutionDate : 2/24/2017 3:18:09 AM
LastPingDateTime       : 2/24/2017 3:35:03 AM
LastSuccessfulAssociationExecutionDate : 2/24/2017 3:18:09 AM
Name                   :
PingStatus             : ConnectionLost
PlatformName           : Amazon Linux AMI
PlatformType           : Linux
PlatformVersion        : 2016.09
RegistrationDate        : 1/1/0001 12:00:00 AM
ResourceType           : EC2Instance

```

Contoh 2: Contoh ini menunjukkan cara menggunakan parameter `-Filter` untuk memfilter hasil hanya ke instance AWS Systems Manager di wilayah **us-east-1** dengan dari **AgentVersion. 2.2.800.0** Anda dapat menemukan daftar nilai kunci `-Filter` yang valid dalam topik referensi InstanceInformation API ([https://docs.aws.amazon.com/systems-manager/latest/APIReference/API_InstanceInformation.html](https://docs.aws.amazon.com/systems-manager/latest/APIReference/API_InstanceInformation.html#systemsmanager-Type-InstanceInformation) #systemsmanager -Type- -InstanceInformation). ActivationId

```

$Filters = @{
    Key="AgentVersion"
    Values="2.2.800.0"
}
Get-SSMInstanceInformation -Region us-east-1 -Filter $Filters

```

Output:

```

ActivationId           :
AgentVersion           : 2.2.800.0
AssociationOverview     :
  Amazon.SimpleSystemsManagement.Model.InstanceAggregatedAssociationOverview
AssociationStatus       : Success
ComputerName           : EXAMPLE-EXAMPLE.WORKGROUP
IamRole                :
InstanceId             : i-EXAMPLEb0792d98ce
IPAddress              : 10.0.0.01
IsLatestVersion        : False
LastAssociationExecutionDate : 8/16/2018 12:02:50 AM
LastPingDateTime       : 8/16/2018 7:40:27 PM

```

```

LastSuccessfulAssociationExecutionDate : 8/16/2018 12:02:50 AM
Name                                   :
PingStatus                           : Online
PlatformName                         : Microsoft Windows Server 2016 Datacenter
PlatformType                         : Windows
PlatformVersion                     : 10.0.14393
RegistrationDate                     : 1/1/0001 12:00:00 AM
ResourceType                         : EC2Instance

ActivationId                         :
AgentVersion                         : 2.2.800.0
AssociationOverview                   :
  Amazon.SimpleSystemsManagement.Model.InstanceAggregatedAssociationOverview
AssociationStatus                     : Success
ComputerName                         : EXAMPLE-EXAMPLE.WORKGROUP
IamRole                              :
InstanceId                           : i-EXAMPLEac7501d023
IPAddress                           : 10.0.0.02
IsLatestVersion                      : False
LastAssociationExecutionDate          : 8/16/2018 12:00:20 AM
LastPingDateTime                     : 8/16/2018 7:40:35 PM
LastSuccessfulAssociationExecutionDate : 8/16/2018 12:00:20 AM
Name                                   :
PingStatus                           : Online
PlatformName                         : Microsoft Windows Server 2016 Datacenter
PlatformType                         : Windows
PlatformVersion                     : 10.0.14393
RegistrationDate                     : 1/1/0001 12:00:00 AM
ResourceType                         : EC2Instance

```

Contoh 3: Contoh ini menunjukkan cara menggunakan `InstanceInformationFilterList` parameter - untuk memfilter hasil hanya ke instance AWS Systems Manager di wilayah **us-east-1** dengan **PlatformTypes** dari **Windows** atau **Linux**. Anda dapat menemukan daftar nilai `InstanceInformationFilterList` kunci yang valid dalam topik referensi `InstanceInformationFilter` API (https://docs.aws.amazon.com/systems-manager/latest/APIReference/API_InstanceInformationFilter.html).

```

$Filters = @{
    Key="PlatformTypes"
    ValueSet=("Windows","Linux")
}
Get-SSMInstanceInformation -Region us-east-1 -InstanceInformationFilterList $Filters

```

Output:

```

ActivationId                :
AgentVersion                 : 2.2.800.0
AssociationOverview         :
  Amazon.SimpleSystemsManagement.Model.InstanceAggregatedAssociationOverview
AssociationStatus           : Success
ComputerName                : EXAMPLE-EXAMPLE.WORKGROUP
IamRole                     :
InstanceId                  : i-EXAMPLEb0792d98ce
IPAddress                   : 10.0.0.27
IsLatestVersion             : False
LastAssociationExecutionDate : 8/16/2018 12:02:50 AM
LastPingDateTime            : 8/16/2018 7:40:27 PM
LastSuccessfulAssociationExecutionDate : 8/16/2018 12:02:50 AM
Name                        :
PingStatus                  : Online
PlatformName                : Ubuntu Server 18.04 LTS
PlatformType                : Linux
PlatformVersion             : 18.04
RegistrationDate            : 1/1/0001 12:00:00 AM
ResourceType                : EC2Instance

ActivationId                :
AgentVersion                 : 2.2.800.0
AssociationOverview         :
  Amazon.SimpleSystemsManagement.Model.InstanceAggregatedAssociationOverview
AssociationStatus           : Success
ComputerName                : EXAMPLE-EXAMPLE.WORKGROUP
IamRole                     :
InstanceId                  : i-EXAMPLEac7501d023
IPAddress                   : 10.0.0.100
IsLatestVersion             : False
LastAssociationExecutionDate : 8/16/2018 12:00:20 AM
LastPingDateTime            : 8/16/2018 7:40:35 PM
LastSuccessfulAssociationExecutionDate : 8/16/2018 12:00:20 AM
Name                        :
PingStatus                  : Online
PlatformName                : Microsoft Windows Server 2016 Datacenter
PlatformType                : Windows
PlatformVersion             : 10.0.14393
RegistrationDate            : 1/1/0001 12:00:00 AM
ResourceType                : EC2Instance

```

Contoh 4: Contoh ini mencantumkan instance dan ekspor yang dikelola ssm InstanceId PingStatus, LastPingDateTime dan PlatformName ke file csv.

```
Get-SSMInstanceInformation | Select-Object InstanceId, PingStatus, LastPingDateTime, PlatformName | Export-Csv Instance-details.csv -NoTypeInfo
```

- Untuk detail API, lihat [DescribeInstanceInformation](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMInstancePatch

Contoh kode berikut menunjukkan cara menggunakan Get-SSMInstancePatch.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan detail kepatuhan tambalan untuk sebuah instance.

```
Get-SSMInstancePatch -InstanceId "i-08ee91c0b17045407"
```

- Untuk detail API, lihat [DescribeInstancePatches](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMInstancePatchState

Contoh kode berikut menunjukkan cara menggunakan Get-SSMInstancePatchState.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan status ringkasan tambalan untuk sebuah instance.

```
Get-SSMInstancePatchState -InstanceId "i-08ee91c0b17045407"
```

Contoh 2: Contoh ini mendapatkan status ringkasan patch untuk dua instance.

```
Get-SSMInstancePatchState -InstanceId "i-08ee91c0b17045407","i-09a618aec652973a9"
```

- Untuk detail API, lihat [DescribeInstancePatchStates](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMInstancePatchStatesForPatchGroup

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMInstancePatchStatesForPatchGroup`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan status ringkasan patch per instance untuk grup patch.

```
Get-SSMInstancePatchStatesForPatchGroup -PatchGroup "Production"
```

- Untuk detail API, lihat [DescribeInstancePatchStatesForPatchGroup](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMInventory

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMInventory`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan metadata kustom untuk inventaris Anda.

```
Get-SSMInventory
```

Output:

```
Data
  Id
----
--
{[AWS:InstanceInformation,
 Amazon.SimpleSystemsManagement.Model.InventoryResultItem]} i-0cb2b964d3e14fd9f
```

- Untuk detail API, lihat [GetInventory](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMInventoryEntriesList

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMInventoryEntriesList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan semua entri inventaris kustom untuk sebuah instance.

```
Get-SSMInventoryEntriesList -InstanceId "i-0cb2b964d3e14fd9f" -TypeName
"Custom:RackInfo"
```

Output:

```
CaptureTime    : 2016-08-22T10:01:01Z
Entries        :
{Amazon.Runtime.Internal.Util.AlwaysSendDictionary`2[System.String,System.String]}
InstanceId     : i-0cb2b964d3e14fd9f
NextToken      :
SchemaVersion  : 1.0
TypeName       : Custom:RackInfo
```

Contoh 2: Contoh ini mencantumkan detailnya.

```
(Get-SSMInventoryEntriesList -InstanceId "i-0cb2b964d3e14fd9f" -TypeName
"Custom:RackInfo").Entries
```

Output:

```
Key           Value
---           -
RackLocation  Bay B/Row C/Rack D/Shelf E
```

- Untuk detail API, lihat [ListInventoryEntries](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMInventoryEntryList

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMInventoryEntryList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil entri inventaris **AWS:Network** tipe untuk instance.

```
Get-SSMInventoryEntryList -InstanceId mi-088dcb0ecea37b076 -TypeName AWS:Network |
Select-Object -ExpandProperty Entries
```

Output:

Key	Value
---	-----
DHCP Server	172.31.11.2
DNS Server	172.31.0.1
Gateway	172.31.11.2
IPv4	172.31.11.222
IPv6	fe12::3456:7da8:901a:12a3
MacAddress	1A:23:4E:5B:FB:67
Name	Amazon Elastic Network Adapter
SubnetMask	255.255.240.0

- Untuk detail API, lihat [Dapatkan-SSMInventoryEntryList](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMInventorySchema

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMInventorySchema`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan daftar nama jenis inventaris untuk akun.

```
Get-SSMInventorySchema
```

- Untuk detail API, lihat [GetInventorySchema](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMLatestEC2Image

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMLatestEC2Image`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan semua Windows AMIs terbaru.

```
PS Get-SSMLatestEC2Image -Path ami-windows-latest
```

Output:

Name	Value
----	-----
Windows_Server-2008-R2_SP1-English-64Bit-SQL_2012_SP4_Express ami-0e5ddd288daff4fab	
Windows_Server-2012-R2_RTM-Chinese_Simplified-64Bit-Base ami-0c5ea64e6bec1cb50	
Windows_Server-2012-R2_RTM-Chinese_Traditional-64Bit-Base ami-09775eff0bf8c113d	
Windows_Server-2012-R2_RTM-Dutch-64Bit-Base ami-025064b67e28cf5df	
...	

Contoh 2: Contoh ini mengambil id AMI dari image Amazon Linux tertentu untuk wilayah us-west-2.

```
PS Get-SSMLatestEC2Image -Path ami-amazon-linux-latest -ImageName amzn-ami-hvm-
x86_64-ebs -Region us-west-2
```

Output:

```
ami-09b92cd132204c704
```

Contoh 3: Contoh ini mencantumkan semua Windows terbaru yang AMIs cocok dengan ekspresi wildcard yang ditentukan.

```
Get-SSMLatestEC2Image -Path ami-windows-latest -ImageName *Windows*2019*English*
```

Output:

Name	Value
----	-----
Windows_Server-2019-English-Full-SQL_2017_Web	ami-085e9d27da5b73a42
Windows_Server-2019-English-STIG-Core	ami-0bfd85c29148c7f80
Windows_Server-2019-English-Full-SQL_2019_Web	ami-02099560d7fb11f20
Windows_Server-2019-English-Full-SQL_2016_SP2_Standard	ami-0d7ae2d81c07bd598
...	

- Untuk detail API, lihat [Dapatkan- SSMLatest EC2 Gambar](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMMaintenanceWindow

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMMaintenanceWindow`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan detail tentang jendela pemeliharaan.

```
Get-SSMMaintenanceWindow -WindowId "mw-03eb9db42890fb82d"
```

Output:

```
AllowUnassociatedTargets : False
CreatedDate               : 2/20/2017 6:14:05 PM
Cutoff                   : 1
Duration                 : 2
Enabled                  : True
ModifiedDate             : 2/20/2017 6:14:05 PM
Name                     : TestMaintWin
Schedule                 : cron(0 */30 * * * ? *)
WindowId                 : mw-03eb9db42890fb82d
```

- Untuk detail API, lihat [GetMaintenanceWindow](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMMaintenanceWindowExecution

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMMaintenanceWindowExecution`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan informasi tentang tugas yang dijalankan sebagai bagian dari eksekusi jendela pemeliharaan.

```
Get-SSMMaintenanceWindowExecution -WindowExecutionId "518d5565-5969-4cca-8f0e-da3b2a638355"
```

Output:

```
EndTime                 : 2/21/2017 4:00:35 PM
StartTime               : 2/21/2017 4:00:34 PM
Status                  : FAILED
```

```
StatusDetails      : One or more tasks in the orchestration failed.
TaskIds            : {ac0c6ae1-daa3-4a89-832e-d384503b6586}
WindowExecutionId  : 518d5565-5969-4cca-8f0e-da3b2a638355
```

- Untuk detail API, lihat [GetMaintenanceWindowExecution](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMMaintenanceWindowExecutionList

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMMaintenanceWindowExecutionList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan semua eksekusi untuk jendela pemeliharaan.

```
Get-SSMMaintenanceWindowExecutionList -WindowId "mw-03eb9db42890fb82d"
```

Output:

```
EndTime           : 2/20/2017 6:30:17 PM
StartTime         : 2/20/2017 6:30:16 PM
Status            : FAILED
StatusDetails     : One or more tasks in the orchestration failed.
WindowExecutionId : 6f3215cf-4101-4fa0-9b7b-9523269599c7
WindowId          : mw-03eb9db42890fb82d
```

Contoh 2: Contoh ini mencantumkan semua eksekusi untuk jendela pemeliharaan sebelum tanggal yang ditentukan.

```
$option1 = @{Key="ExecutedBefore";Values=@("2016-11-04T05:00:00Z")}
Get-SSMMaintenanceWindowExecutionList -WindowId "mw-03eb9db42890fb82d" -Filter
$option1
```

Contoh 3: Contoh ini mencantumkan semua eksekusi untuk jendela pemeliharaan setelah tanggal yang ditentukan.

```
$option1 = @{Key="ExecutedAfter";Values=@("2016-11-04T05:00:00Z")}
Get-SSMMaintenanceWindowExecutionList -WindowId "mw-03eb9db42890fb82d" -Filter
$option1
```

- Untuk detail API, lihat [DescribeMaintenanceWindowExecutions](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMMaintenanceWindowExecutionTask

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMMaintenanceWindowExecutionTask`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan informasi tentang tugas yang merupakan bagian dari eksekusi jendela pemeliharaan.

```
Get-SSMMaintenanceWindowExecutionTask -TaskId "ac0c6ae1-daa3-4a89-832e-d384503b6586"
-WindowExecutionId "518d5565-5969-4cca-8f0e-da3b2a638355"
```

Output:

```
EndTime           : 2/21/2017 4:00:35 PM
MaxConcurrency     : 1
MaxErrors          : 1
Priority           : 10
ServiceRole        : arn:aws:iam::123456789012:role/MaintenanceWindowsRole
StartTime          : 2/21/2017 4:00:34 PM
Status             : FAILED
StatusDetails      : The maximum error count was exceeded.
TaskArn            : AWS-RunShellScript
TaskExecutionId    : ac0c6ae1-daa3-4a89-832e-d384503b6586
TaskParameters     :
    {Amazon.Runtime.Internal.Util.AlwaysSendDictionary`2[System.String,Amazon.SimpleSystemsManag
        meterValueExpression]}
Type               : RUN_COMMAND
WindowExecutionId  : 518d5565-5969-4cca-8f0e-da3b2a638355
```

- Untuk detail API, lihat [GetMaintenanceWindowExecutionTask](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMMaintenanceWindowExecutionTaskInvocationList

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMMaintenanceWindowExecutionTaskInvocationList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan pemanggilan untuk tugas yang dijalankan sebagai bagian dari eksekusi jendela pemeliharaan.

```
Get-SSMMaintenanceWindowExecutionTaskInvocationList -TaskId "ac0c6ae1-
daa3-4a89-832e-d384503b6586" -WindowExecutionId "518d5565-5969-4cca-8f0e-
da3b2a638355"
```

Output:

```
EndTime           : 2/21/2017 4:00:34 PM
ExecutionId        :
InvocationId       : e274b6e1-fe56-4e32-bd2a-8073c6381d8b
OwnerInformation   :
Parameters         : {"documentName":"AWS-RunShellScript","instanceIds":
["i-0000293ffd8c57862"],"parameters":{"commands":["df"]},"maxConcurrency":"1",
                        "maxErrors":"1"}
StartTime          : 2/21/2017 4:00:34 PM
Status             : FAILED
StatusDetails      : The instance IDs list contains an invalid entry.
TaskExecutionId    : ac0c6ae1-daa3-4a89-832e-d384503b6586
WindowExecutionId  : 518d5565-5969-4cca-8f0e-da3b2a638355
WindowTargetId     :
```

- Untuk detail API, lihat [DescribeMaintenanceWindowExecutionTaskInvocations](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMMaintenanceWindowExecutionTaskList

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMMaintenanceWindowExecutionTaskList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan tugas yang terkait dengan eksekusi jendela pemeliharaan.

```
Get-SSMMaintenanceWindowExecutionTaskList -WindowExecutionId
"518d5565-5969-4cca-8f0e-da3b2a638355"
```

Output:

```

EndTime       : 2/21/2017 4:00:35 PM
StartTime     : 2/21/2017 4:00:34 PM
Status        : SUCCESS
TaskArn       : AWS-RunShellScript
TaskExecutionId : ac0c6ae1-daa3-4a89-832e-d384503b6586
TaskType      : RUN_COMMAND
WindowExecutionId : 518d5565-5969-4cca-8f0e-da3b2a638355

```

- Untuk detail API, lihat [DescribeMaintenanceWindowExecutionTasks](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMMaintenanceWindowList

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMMaintenanceWindowList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan semua jendela pemeliharaan di akun Anda.

```
Get-SSMMaintenanceWindowList
```

Output:

```

Cutoff       : 1
Duration     : 4
Enabled      : True
Name         : My-First-Maintenance-Window
WindowId     : mw-06d59c1a07c022145

```

- Untuk detail API, lihat [DescribeMaintenanceWindows](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMMaintenanceWindowTarget

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMMaintenanceWindowTarget`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan semua target untuk jendela pemeliharaan.

```
Get-SSMMaintenanceWindowTarget -WindowId "mw-06cf17cbefcb4bf4f"
```

Output:

```
OwnerInformation : Single instance
ResourceType     : INSTANCE
Targets          : {InstanceIds}
WindowId         : mw-06cf17cbefcb4bf4f
WindowTargetId   : 350d44e6-28cc-44e2-951f-4b2c985838f6

OwnerInformation : Two instances in a list
ResourceType     : INSTANCE
Targets          : {InstanceIds}
WindowId         : mw-06cf17cbefcb4bf4f
WindowTargetId   : e078a987-2866-47be-bedd-d9cf49177d3a
```

- Untuk detail API, lihat [DescribeMaintenanceWindowTargets](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMMaintenanceWindowTaskList

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMMaintenanceWindowTaskList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan semua tugas untuk jendela pemeliharaan.

```
Get-SSMMaintenanceWindowTaskList -WindowId "mw-06cf17cbefcb4bf4f"
```

Output:

```
LoggingInfo      :
MaxConcurrency    : 1
MaxErrors        : 1
Priority          : 10
ServiceRoleArn   : arn:aws:iam::123456789012:role/MaintenanceWindowsRole
Targets          : {InstanceIds}
TaskArn          : AWS-RunShellScript
TaskParameters   : {[commands,
  Amazon.SimpleSystemsManagement.Model.MaintenanceWindowTaskParameterValueExpression]}
```

```
Type           : RUN_COMMAND
WindowId        : mw-06cf17cbefcb4bf4f
WindowTaskId    : a23e338d-ff30-4398-8aa3-09cd052ebf17
```

- Untuk detail API, lihat [DescribeMaintenanceWindowTasks](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMParameterHistory

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMParameterHistory`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan riwayat nilai untuk parameter.

```
Get-SSMParameterHistory -Name "Welcome"
```

Output:

```
Description      :
KeyId            :
LastModifiedDate : 3/3/2017 6:55:25 PM
LastModifiedUser : arn:aws:iam::123456789012:user/admin
Name             : Welcome
Type             : String
Value           : helloWorld
```

- Untuk detail API, lihat [GetParameterHistory](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMParameterList

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMParameterList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan semua parameter.

```
Get-SSMParameterList
```

Output:

```

Description      :
KeyId            :
LastModifiedDate : 3/3/2017 6:58:23 PM
LastModifiedUser : arn:aws:iam::123456789012:user/admin
Name             : Welcome
Type             : String

```

- Untuk detail API, lihat [DescribeParameters](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMParameterValue

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMParameterValue`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan nilai untuk parameter.

```
Get-SSMParameterValue -Name "Welcome"
```

Output:

```

InvalidParameters Parameters
-----
{}                  {Welcome}

```

Contoh 2: Contoh ini mencantumkan rincian nilai.

```
(Get-SSMParameterValue -Name "Welcome").Parameters
```

Output:

```

Name    Type    Value
----
Welcome String Good day, Sunshine!

```

- Untuk detail API, lihat [GetParameters](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMPatchBaseline

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMPatchBaseline`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan semua baseline patch.

```
Get-SSMPatchBaseline
```

Output:

BaselineDescription	BaselineName	BaselineId
Default Patch Baseline Provided by AWS.		arn:aws:ssm:us-west-2:123456789012:patchbaseline/pb-04fb4ae6142167966
Baseline containing all updates approved for production systems	Production-B...	pb-045f10b4f382baeda
Baseline containing all updates approved for production systems	Production-B...	pb-0a2f1059b670ebd31

Contoh 2: Contoh ini mencantumkan semua baseline patch yang disediakan oleh AWS. Sintaks yang digunakan oleh contoh ini memerlukan PowerShell versi 3 atau yang lebih baru.

```
$filter1 = @{"Key"="OWNER";Values=@("AWS")}
```

Output:

```
Get-SSMPatchBaseline -Filter $filter1
```

Contoh 3: Contoh ini mencantumkan semua baseline patch dengan Anda sebagai pemilik. Sintaks yang digunakan oleh contoh ini memerlukan PowerShell versi 3 atau yang lebih baru.

```
$filter1 = @{"Key"="OWNER";Values=@("Self")}
```

Output:

```
Get-SSMPatchBaseline -Filter $filter1
```

Contoh 4: Dengan PowerShell versi 2, Anda harus menggunakan New-Object untuk membuat setiap tag.

```
$filter1 = New-Object Amazon.SimpleSystemsManagement.Model.PatchOrchestratorFilter
$filter1.Key = "OWNER"
$filter1.Values = "AWS"

Get-SSMPatchBaseline -Filter $filter1
```

Output:

BaselineDescription	BaselineName	BaselineId	DefaultBaseline
Default Patch Baseline Provided by AWS.	arn:aws:ssm:us-west-2:123456789012:patchbaseline/pb-04fb4ae6142167966	AWS-DefaultPatchBaseline	True

- Untuk detail API, lihat [DescribePatchBaselines](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMPatchBaselineDetail

Contoh kode berikut menunjukkan cara menggunakan Get-SSMPatchBaselineDetail.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menampilkan detail untuk baseline patch.

```
Get-SSMPatchBaselineDetail -BaselineId "pb-03da896ca3b68b639"
```

Output:

```
ApprovalRules : Amazon.SimpleSystemsManagement.Model.PatchRuleGroup
ApprovedPatches : {}
BaselineId : pb-03da896ca3b68b639
CreatedDate : 3/3/2017 5:02:19 PM
Description : Baseline containing all updates approved for production systems
GlobalFilters : Amazon.SimpleSystemsManagement.Model.PatchFilterGroup
```

```

ModifiedDate    : 3/3/2017 5:02:19 PM
Name            : Production-Baseline
PatchGroups     : {}
RejectedPatches : {}

```

- Untuk detail API, lihat [GetPatchBaseline](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMPatchBaselineForPatchGroup

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMPatchBaselineForPatchGroup`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menampilkan baseline patch untuk grup patch.

```
Get-SSMPatchBaselineForPatchGroup -PatchGroup "Production"
```

Output:

```

BaselineId      PatchGroup
-----
pb-045f10b4f382baeda Production

```

- Untuk detail API, lihat [GetPatchBaselineForPatchGroup](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMPatchGroup

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMPatchGroup`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan pendaftaran grup patch.

```
Get-SSMPatchGroup
```

Output:

```

BaselineIdentity      PatchGroup
-----

```

```
Amazon.SimpleSystemsManagement.Model.PatchBaselineIdentity Production
```

- Untuk detail API, lihat [DescribePatchGroups](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMPatchGroupState

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMPatchGroupState`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan ringkasan kepatuhan patch tingkat tinggi untuk grup patch.

```
Get-SSMPatchGroupState -PatchGroup "Production"
```

Output:

```
Instances : 4
InstancesWithFailedPatches : 1
InstancesWithInstalledOtherPatches : 4
InstancesWithInstalledPatches : 3
InstancesWithMissingPatches : 0
InstancesWithNotApplicablePatches : 0
```

- Untuk detail API, lihat [DescribePatchGroupState](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMResourceComplianceSummaryList

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMResourceComplianceSummaryList`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan jumlah ringkasan tingkat sumber daya. Ringkasan tersebut mencakup informasi tentang status yang sesuai dan tidak sesuai serta jumlah keparahan item kepatuhan terperinci untuk produk yang cocok dengan "Windows10". Karena `MaxResult` defaultnya adalah 100 jika parameter tidak ditentukan, dan nilai ini tidak valid, `MaxResult` parameter ditambahkan, dan nilainya diatur ke 50.

```
$FilterValues = @{
    "Key"="Product"
    "Type"="EQUAL"
    "Values"="Windows10"
}

Get-SSMResourceComplianceSummaryList -Filter $FilterValues -MaxResult 50
```

- Untuk detail API, lihat [ListResourceComplianceSummaries](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-SSMResourceTag

Contoh kode berikut menunjukkan cara menggunakan `Get-SSMResourceTag`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan tag untuk jendela pemeliharaan.

```
Get-SSMResourceTag -ResourceId "mw-03eb9db42890fb82d" -ResourceType
"MaintenanceWindow"
```

Output:

```
Key    Value
---    -
Stack  Production
```

- Untuk detail API, lihat [ListTagsForResource](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-SSMActivation

Contoh kode berikut menunjukkan cara menggunakan `New-SSMActivation`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat instance terkelola.

```
New-SSMActivation -DefaultInstanceName "MyWebServers" -IamRole "SSMAutomationRole" -
RegistrationLimit 10
```

Output:

ActivationCode	ActivationId
-----	-----
KWChh0xBTiwDcKE9B1KC	08e51e79-1e36-446c-8e63-9458569c1363

- Untuk detail API, lihat [CreateActivation](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-SSMAssociation

Contoh kode berikut menunjukkan cara menggunakan `New-SSMAssociation`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengaitkan dokumen konfigurasi dengan instance, menggunakan instance IDs.

```
New-SSMAssociation -InstanceId "i-0cb2b964d3e14fd9f" -Name "AWS-UpdateSSMAgent"
```

Output:

```
Name           : AWS-UpdateSSMAgent
InstanceId      : i-0000293ffd8c57862
Date           : 2/23/2017 6:55:22 PM
Status.Name     : Associated
Status.Date     : 2/20/2015 8:31:11 AM
Status.Message  : Associated with AWS-UpdateSSMAgent
Status.AdditionalInfo :
```

Contoh 2: Contoh ini mengaitkan dokumen konfigurasi dengan instance, menggunakan target.

```
$target = @{Key="instanceids";Values=@("i-0cb2b964d3e14fd9f")}
New-SSMAssociation -Name "AWS-UpdateSSMAgent" -Target $target
```

Output:

```
Name           : AWS-UpdateSSMAgent
InstanceId      :
Date           : 3/1/2017 6:22:21 PM
Status.Name     :
Status.Date     :
```

```
Status.Message      :
Status.AdditionalInfo :
```

Contoh 3: Contoh ini mengaitkan dokumen konfigurasi dengan instance, menggunakan target dan parameter.

```
$target = @{"Key"="instanceids";Values=@"i-0cb2b964d3e14fd9f"}
$params = @{
    "action"="configure"
    "mode"="ec2"
    "optionalConfigurationSource"="ssm"
    "optionalConfigurationLocation"=""
    "optionalRestart"="yes"
}
New-SSMAssociation -Name "Configure-CloudWatch" -AssociationName "CWConfiguration" -
Target $target -Parameter $params
```

Output:

```
Name                : Configure-CloudWatch
InstanceId           :
Date                : 5/17/2018 3:17:44 PM
Status.Name          :
Status.Date          :
Status.Message       :
Status.AdditionalInfo :
```

Contoh 4: Contoh ini membuat asosiasi dengan semua instance di wilayah, dengan **AWS-GatherSoftwareInventory**. Ini juga menyediakan file kustom dan lokasi registri dalam parameter untuk mengumpulkan

```
$params =
[Collections.Generic.Dictionary[String,Collections.Generic.List[String]]::new()
$params["windowsRegistry"] = '[{"Path":"HKEY_LOCAL_MACHINE\SOFTWARE\Amazon
\MachineImage","Recursive":false,"ValueNames":["AMIName"]}]'
$params["files"] = '[{"Path":"C:\Program Files","Pattern":
["*.exe"],"Recursive":true}, {"Path":"C:\ProgramData","Pattern":
["*.log"],"Recursive":true}]'
New-SSMAssociation -AssociationName new-in-mum -Name AWS-GatherSoftwareInventory
-Target @{"Key"="instanceids";Values="*"} -Parameter $params -region ap-south-1 -
ScheduleExpression "rate(720 minutes)"
```

Output:

```

Name           : AWS-GatherSoftwareInventory
InstanceId      :
Date           : 6/9/2019 8:57:56 AM
Status.Name     :
Status.Date     :
Status.Message  :
Status.AdditionalInfo :

```

- Untuk detail API, lihat [CreateAssociation](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-SSMAssociationFromBatch

Contoh kode berikut menunjukkan cara menggunakan `New-SSMAssociationFromBatch`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengaitkan dokumen konfigurasi dengan beberapa instance. Output mengembalikan daftar operasi yang berhasil dan gagal, jika berlaku.

```

$option1 = @{InstanceId="i-0cb2b964d3e14fd9f";Name=@"AWS-UpdateSSMAgent"}}
$option2 = @{InstanceId="i-0000293ffd8c57862";Name=@"AWS-UpdateSSMAgent"}}
New-SSMAssociationFromBatch -Entry $option1,$option2

```

Output:

```

Failed    Successful
-----
{}        {Amazon.SimpleSystemsManagement.Model.FailedCreateAssociation,
Amazon.SimpleSystemsManagement.Model.FailedCreateAsso...

```

Contoh 2: Contoh ini akan menunjukkan detail lengkap dari operasi yang berhasil.

```

$option1 = @{InstanceId="i-0cb2b964d3e14fd9f";Name=@"AWS-UpdateSSMAgent"}}
$option2 = @{InstanceId="i-0000293ffd8c57862";Name=@"AWS-UpdateSSMAgent"}}
(New-SSMAssociationFromBatch -Entry $option1,$option2).Successful

```

- Untuk detail API, lihat [CreateAssociationBatch](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-SSMDocument

Contoh kode berikut menunjukkan cara menggunakan `New-SSMDocument`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat dokumen di akun Anda. Dokumen harus dalam format JSON. Untuk informasi selengkapnya tentang menulis dokumen konfigurasi, lihat [Dokumen Konfigurasi di Referensi API SSM](#).

```
New-SSMDocument -Content (Get-Content -Raw "c:\temp\RunShellScript.json") -Name
"RunShellScript" -DocumentType "Command"
```

Output:

```

CreatedDate      : 3/1/2017 1:21:33 AM
DefaultVersion   : 1
Description      : Run an updated script
DocumentType     : Command
DocumentVersion  : 1
Hash             : 1d5ce820e999ff051eb4841ed887593daf77120fd76cae0d18a53cc42e4e22c1
HashType        : Sha256
LatestVersion    : 1
Name             : RunShellScript
Owner            : 809632081692
Parameters       : {commands}
PlatformTypes    : {Linux}
SchemaVersion    : 2.0
Sha1             :
Status           : Creating
```

- Untuk detail API, lihat [CreateDocument](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-SSMMaintenanceWindow

Contoh kode berikut menunjukkan cara menggunakan `New-SSMMaintenanceWindow`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat jendela pemeliharaan baru dengan nama tertentu yang berjalan pada pukul 4 sore setiap hari Selasa selama 4 jam, dengan cutoff 1 jam, dan yang memungkinkan target yang tidak terkait.

```
New-SSMMaintenanceWindow -Name "MyMaintenanceWindow" -Duration 4 -Cutoff 1 -
AllowUnassociatedTarget $true -Schedule "cron(0 16 ? * TUE *)"
```

Output:

```
mw-03eb53e1ea7383998
```

- Untuk detail API, lihat [CreateMaintenanceWindow](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-SSMPatchBaseline

Contoh kode berikut menunjukkan cara menggunakan New-SSMPatchBaseline.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat baseline patch yang menyetujui tambalan, tujuh hari setelah dirilis oleh Microsoft, untuk instance terkelola yang menjalankan Windows Server 2019 di lingkungan produksi.

```
$rule = New-Object Amazon.SimpleSystemsManagement.Model.PatchRule
$rule.ApproveAfterDays = 7

$ruleFilters = New-Object Amazon.SimpleSystemsManagement.Model.PatchFilterGroup

$patchFilter = New-Object Amazon.SimpleSystemsManagement.Model.PatchFilter
$patchFilter.Key="PRODUCT"
$patchFilter.Values="WindowsServer2019"

$severityFilter = New-Object Amazon.SimpleSystemsManagement.Model.PatchFilter
$severityFilter.Key="MSRC_SEVERITY"
$severityFilter.Values.Add("Critical")
$severityFilter.Values.Add("Important")
$severityFilter.Values.Add("Moderate")

$classificationFilter = New-Object Amazon.SimpleSystemsManagement.Model.PatchFilter
$classificationFilter.Key = "CLASSIFICATION"
$classificationFilter.Values.Add( "SecurityUpdates" )
$classificationFilter.Values.Add( "Updates" )
$classificationFilter.Values.Add( "UpdateRollups" )
$classificationFilter.Values.Add( "CriticalUpdates" )
```

```
$ruleFilters.PatchFilters.Add($severityFilter)
$ruleFilters.PatchFilters.Add($classificationFilter)
$ruleFilters.PatchFilters.Add($patchFilter)
$rule.PatchFilterGroup = $ruleFilters

New-SSMPatchBaseline -Name "Production-Baseline-Windows2019" -Description "Baseline
containing all updates approved for production systems" -ApprovalRules_PatchRule
$rule
```

Output:

```
pb-0z4z6221c4296b23z
```

- Untuk detail API, lihat [CreatePatchBaseline](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Register-SSMDefaultPatchBaseline

Contoh kode berikut menunjukkan cara menggunakan `Register-SSMDefaultPatchBaseline`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendaftarkan baseline patch sebagai baseline patch default.

```
Register-SSMDefaultPatchBaseline -BaselineId "pb-03da896ca3b68b639"
```

Output:

```
pb-03da896ca3b68b639
```

- Untuk detail API, lihat [RegisterDefaultPatchBaseline](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Register-SSMPatchBaselineForPatchGroup

Contoh kode berikut menunjukkan cara menggunakan `Register-SSMPatchBaselineForPatchGroup`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendaftarkan baseline patch untuk grup patch.

```
Register-SSMPatchBaselineForPatchGroup -BaselineId "pb-03da896ca3b68b639" -
PatchGroup "Production"
```

Output:

BaselineId	PatchGroup
-----	-----
pb-03da896ca3b68b639	Production

- Untuk detail API, lihat [RegisterPatchBaselineForPatchGroup](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Register-SSMTargetWithMaintenanceWindow

Contoh kode berikut menunjukkan cara menggunakan `Register-SSMTargetWithMaintenanceWindow`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendaftarkan instance dengan jendela pemeliharaan.

```
$option1 = @{Key="InstanceIds";Values=@"i-0000293ffd8c57862"}
Register-SSMTargetWithMaintenanceWindow -WindowId "mw-06cf17cbefcb4bf4f" -Target
$option1 -OwnerInformation "Single instance" -ResourceType "INSTANCE"
```

Output:

```
d8e47760-23ed-46a5-9f28-927337725398
```

Contoh 2: Contoh ini mendaftarkan beberapa instance dengan jendela pemeliharaan.

```
$option1 =
@{Key="InstanceIds";Values=@"i-0000293ffd8c57862","i-0cb2b964d3e14fd9f"}}
Register-SSMTargetWithMaintenanceWindow -WindowId "mw-06cf17cbefcb4bf4f" -Target
$option1 -OwnerInformation "Single instance" -ResourceType "INSTANCE"
```

Output:

```
6ab5c208-9fc4-4697-84b7-b02a6cc25f7d
```

Contoh 3: Contoh ini mendaftarkan instance dengan jendela pemeliharaan menggunakan EC2 tag.

```
$option1 = @{Key="tag:Environment";Values=@"Production"}
Register-SSMTargetWithMaintenanceWindow -WindowId "mw-06cf17cbefcb4bf4f" -Target
$option1 -OwnerInformation "Production Web Servers" -ResourceType "INSTANCE"
```

Output:

```
2994977e-aefb-4a71-beac-df620352f184
```

- Untuk detail API, lihat [RegisterTargetWithMaintenanceWindow](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Register-SSMTaskWithMaintenanceWindow

Contoh kode berikut menunjukkan cara menggunakan `Register-SSMTaskWithMaintenanceWindow`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendaftarkan tugas dengan jendela pemeliharaan menggunakan ID instance. Outputnya adalah Task ID.

```
$parameters = @{}
$parameterValues = New-Object
Amazon.SimpleSystemsManagement.Model.MaintenanceWindowTaskParameterValueExpression
$parameterValues.Values = @"Install"
$parameters.Add("Operation", $parameterValues)

Register-SSMTaskWithMaintenanceWindow -WindowId "mw-03a342e62c96d31b0"
-ServiceRoleArn "arn:aws:iam::123456789012:role/MaintenanceWindowsRole"
-MaxConcurrency 1 -MaxError 1 -TaskArn "AWS-RunShellScript" -Target
@{ Key="InstanceIds";Values="i-0000293ffd8c57862" } -TaskType "RUN_COMMAND" -
Priority 10 -TaskParameter $parameters
```

Output:

```
f34a2c47-ddfd-4c85-a88d-72366b69af1b
```

Contoh 2: Contoh ini mendaftarkan tugas dengan jendela pemeliharaan menggunakan ID target. Outputnya adalah Task ID.

```
$parameters = @{}
$parameterValues = New-Object
    Amazon.SimpleSystemsManagement.Model.MaintenanceWindowTaskParameterValueExpression
$parameterValues.Values = @("Install")
$parameters.Add("Operation", $parameterValues)

register-ssmtaskwithmaintenancewindow -WindowId "mw-03a342e62c96d31b0"
    -ServiceRoleArn "arn:aws:iam::123456789012:role/MaintenanceWindowsRole"
    -MaxConcurrency 1 -MaxError 1 -TaskArn "AWS-RunShellScript" -Target
    @{ Key="WindowTargetIds";Values="350d44e6-28cc-44e2-951f-4b2c985838f6" } -TaskType
    "RUN_COMMAND" -Priority 10 -TaskParameter $parameters
```

Output:

```
f34a2c47-ddfd-4c85-a88d-72366b69af1b
```

Contoh 3: Contoh ini membuat objek parameter untuk dokumen perintah run **AWS-RunPowerShellScript** dan membuat tugas dengan jendela pemeliharaan yang diberikan menggunakan ID target. Output yang dikembalikan adalah ID tugas.

```
$parameters =
    [Collections.Generic.Dictionary[String,Collections.Generic.List[String]]::new()
$parameters.Add("commands",@("ipconfig","dir env:\computername"))
$parameters.Add("executionTimeout",@(3600))

$props = @{
    WindowId = "mw-0123e4cce56ff78ae"
    ServiceRoleArn = "arn:aws:iam::123456789012:role/MaintenanceWindowsRole"
    MaxConcurrency = 1
    MaxError = 1
    TaskType = "RUN_COMMAND"
    TaskArn = "AWS-RunPowerShellScript"
    Target = @{Key="WindowTargetIds";Values="fe1234ea-56d7-890b-12f3-456b789bee0f"}
    Priority = 1
```

```

    RunCommand_Parameter = $parameters
    Name = "set-via-cmdlet"
}

Register-SSMTaskWithMaintenanceWindow @props

```

Output:

```
f1e2ef34-5678-12e3-456a-12334c5c6cbe
```

Contoh 4: Contoh ini mendaftarkan tugas Automasi AWS Systems Manager dengan menggunakan dokumen bernama **Create-Snapshots**.

```

$automationParameters = @{}
$automationParameters.Add( "instanceId", @("{ TARGET_ID }") )
$automationParameters.Add( "AutomationAssumeRole",
    @("{arn:aws:iam::111111111111:role/AutomationRole}") )
$automationParameters.Add( "SnapshotTimeout", @("PT20M") )
Register-SSMTaskWithMaintenanceWindow -WindowId mw-123EXAMPLE456`
    -ServiceRoleArn "arn:aws:iam::123456789012:role/MW-Role"`
    -MaxConcurrency 1 -MaxError 1 -TaskArn "CreateVolumeSnapshots"`
    -Target @{ Key="WindowTargetIds";Values="4b5acdf4-946c-4355-
bd68-4329a43a5fd1" }`
    -TaskType "AUTOMATION"`
    -Priority 4`
    -Automation_DocumentVersion '$DEFAULT' -Automation_Parameter
$automationParameters -Name "Create-Snapshots"

```

- Untuk detail API, lihat [RegisterTaskWithMaintenanceWindow](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-SSMActivation

Contoh kode berikut menunjukkan cara menggunakan `Remove-SSMActivation`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus aktivasi. Tidak ada output jika perintah berhasil.

```
Remove-SSMActivation -ActivationId "08e51e79-1e36-446c-8e63-9458569c1363"
```

- Untuk detail API, lihat [DeleteActivation](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-SSMAssociation

Contoh kode berikut menunjukkan cara menggunakan `Remove-SSMAssociation`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus asosiasi antara instance dan dokumen. Tidak ada output jika perintah berhasil.

```
Remove-SSMAssociation -InstanceId "i-0cb2b964d3e14fd9f" -Name "AWS-UpdateSSMAgent"
```

- Untuk detail API, lihat [DeleteAssociation](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-SSMDocument

Contoh kode berikut menunjukkan cara menggunakan `Remove-SSMDocument`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus dokumen. Tidak ada output jika perintah berhasil.

```
Remove-SSMDocument -Name "RunShellScript"
```

- Untuk detail API, lihat [DeleteDocument](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-SSMMaintenanceWindow

Contoh kode berikut menunjukkan cara menggunakan `Remove-SSMMaintenanceWindow`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus jendela pemeliharaan.

```
Remove-SSMMaintenanceWindow -WindowId "mw-06d59c1a07c022145"
```

Output:

```
mw-06d59c1a07c022145
```

- Untuk detail API, lihat [DeleteMaintenanceWindow](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-SSMParameter

Contoh kode berikut menunjukkan cara menggunakan `Remove-SSMParameter`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus parameter. Tidak ada output jika perintah berhasil.

```
Remove-SSMParameter -Name "helloWorld"
```

- Untuk detail API, lihat [DeleteParameter](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-SSMPatchBaseline

Contoh kode berikut menunjukkan cara menggunakan `Remove-SSMPatchBaseline`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus baseline patch.

```
Remove-SSMPatchBaseline -BaselineId "pb-045f10b4f382baeda"
```

Output:

```
pb-045f10b4f382baeda
```

- Untuk detail API, lihat [DeletePatchBaseline](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-SSMResourceTag

Contoh kode berikut menunjukkan cara menggunakan `Remove-SSMResourceTag`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus tag dari jendela pemeliharaan. Tidak ada output jika perintah berhasil.

```
Remove-SSMResourceTag -ResourceId "mw-03eb9db42890fb82d" -ResourceType
"MaintenanceWindow" -TagKey "Production"
```

- Untuk detail API, lihat [RemoveTagsFromResource](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Send-SSMCommand

Contoh kode berikut menunjukkan cara menggunakan `Send-SSMCommand`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjalankan perintah `echo` pada instance target.

```
Send-SSMCommand -DocumentName "AWS-RunPowerShellScript" -Parameter @{commands =
"echo helloWorld"} -Target @{Key="instanceids";Values=@"i-0cb2b964d3e14fd9f"}}
```

Output:

```
CommandId      : d8d190fc-32c1-4d65-a0df-ff5ff3965524
Comment       :
CompletedCount : 0
DocumentName   : AWS-RunPowerShellScript
ErrorCount     : 0
ExpiresAfter   : 3/7/2017 10:48:37 PM
InstanceIds    : {}
MaxConcurrency  : 50
MaxErrors      : 0
NotificationConfig : Amazon.SimpleSystemsManagement.Model.NotificationConfig
OutputS3BucketName :
OutputS3KeyPrefix :
OutputS3Region  :
Parameters     : {[commands,
  Amazon.Runtime.Internal.Util.AlwaysSendList`1[System.String]]}
RequestedDateTime : 3/7/2017 9:48:37 PM
ServiceRole     :
Status          : Pending
StatusDetails   : Pending
TargetCount     : 0
Targets         : {instanceids}
```

Contoh 2: Contoh ini menunjukkan cara menjalankan perintah yang menerima parameter bersarang.

```
Send-SSMCommand -DocumentName "AWS-RunRemoteScript" -Parameter
@{ sourceType="GitHub";sourceInfo='{ "owner": "me","repository": "amazon-
ssm","path": "Examples/Install-Win32OpenSSH"}'; "commandLine"=".\\Install-
Win32OpenSSH.ps1"} -InstanceId i-0cb2b964d3e14fd9f
```

- Untuk detail API, lihat [SendCommand](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Start-SSMAutomationExecution

Contoh kode berikut menunjukkan cara menggunakan `Start-SSMAutomationExecution`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menjalankan dokumen yang menentukan peran Otomasi, ID sumber AMI, dan peran EC2 instans Amazon.

```
Start-SSMAutomationExecution -DocumentName AWS-UpdateLinuxAmi -
Parameter @{ 'AutomationAssumeRole'='arn:aws:iam::123456789012:role/
SSMAutomationRole'; 'SourceAmiId'='ami-f173cc91'; 'InstanceIamRole'='EC2InstanceRole' }
```

Output:

```
3a532a4f-0382-11e7-9df7-6f11185f6dd1
```

- Untuk detail API, lihat [StartAutomationExecution](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Start-SSMSession

Contoh kode berikut menunjukkan cara menggunakan `Start-SSMSession`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memulai koneksi ke target untuk sesi Session Manager, memungkinkan penerusan port.

```
Start-SSMSession -Target 'i-064578e5e7454488f' -DocumentName 'AWS-
StartPortForwardingSession' -Parameter @{ localPortNumber = '8080'; portNumber =
'80' }
```

Output:

SessionId	StreamUrl
-----	-----
random-id0	wss://ssmmessages.amazonaws.com/v1/data-channel/random-id

- Untuk detail API, lihat [StartSession](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Stop-SSMAutomationExecution

Contoh kode berikut menunjukkan cara menggunakan `Stop-SSMAutomationExecution`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghentikan Eksekusi Otomasi. Tidak ada output jika perintah berhasil.

```
Stop-SSMAutomationExecution -AutomationExecutionId "4105a4fc-
f944-11e6-9d32-8fb2db27a909"
```

- Untuk detail API, lihat [StopAutomationExecution](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Stop-SSMCommand

Contoh kode berikut menunjukkan cara menggunakan `Stop-SSMCommand`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencoba untuk membatalkan perintah. Tidak ada output jika operasi berhasil.

```
Stop-SSMCommand -CommandId "9ded293e-e792-4440-8e3e-7b8ec5feaa38"
```

- Untuk detail API, lihat [CancelCommand](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Unregister-SSMManagedInstance

Contoh kode berikut menunjukkan cara menggunakan `Unregister-SSMManagedInstance`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membatalkan pendaftaran instance terkelola. Tidak ada output jika perintah berhasil.

```
Unregister-SSMManagedInstance -InstanceId "mi-08ab247cdf1046573"
```

- Untuk detail API, lihat [DeregisterManagedInstance](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Unregister-SSMPatchBaselineForPatchGroup

Contoh kode berikut menunjukkan cara menggunakan `Unregister-SSMPatchBaselineForPatchGroup`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membatalkan pendaftaran grup patch dari baseline patch.

```
Unregister-SSMPatchBaselineForPatchGroup -BaselineId "pb-045f10b4f382baeda" -  
PatchGroup "Production"
```

Output:

BaselineId	PatchGroup
-----	-----
pb-045f10b4f382baeda	Production

- Untuk detail API, lihat [DeregisterPatchBaselineForPatchGroup](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Unregister-SSMTargetFromMaintenanceWindow

Contoh kode berikut menunjukkan cara menggunakan `Unregister-SSMTargetFromMaintenanceWindow`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus target dari jendela pemeliharaan.

```
Unregister-SSMTargetFromMaintenanceWindow -WindowTargetId "6ab5c208-9fc4-4697-84b7-b02a6cc25f7d" -WindowId "mw-06cf17cbefcb4bf4f"
```

Output:

WindowId	WindowTargetId
-----	-----
mw-06cf17cbefcb4bf4f	6ab5c208-9fc4-4697-84b7-b02a6cc25f7d

- Untuk detail API, lihat [DeregisterTargetFromMaintenanceWindow](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Unregister-SSMTaskFromMaintenanceWindow

Contoh kode berikut menunjukkan cara menggunakan `Unregister-SSMTaskFromMaintenanceWindow`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus tugas dari jendela pemeliharaan.

```
Unregister-SSMTaskFromMaintenanceWindow -WindowTaskId "f34a2c47-ddfd-4c85-a88d-72366b69af1b" -WindowId "mw-03a342e62c96d31b0"
```

Output:

WindowId	WindowTaskId
-----	-----
mw-03a342e62c96d31b0	f34a2c47-ddfd-4c85-a88d-72366b69af1b

- Untuk detail API, lihat [DeregisterTaskFromMaintenanceWindow](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-SSMAssociation

Contoh kode berikut menunjukkan cara menggunakan `Update-SSMAssociation`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memperbarui asosiasi dengan versi dokumen baru.

```
Update-SSMAssociation -AssociationId "93285663-92df-44cb-9f26-2292d4ecc439" -
DocumentVersion "1"
```

Output:

```
Name           : AWS-UpdateSSMAgent
InstanceId      :
Date           : 3/1/2017 6:22:21 PM
Status.Name     :
Status.Date     :
Status.Message  :
Status.AdditionalInfo :
```

- Untuk detail API, lihat [UpdateAssociation](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-SSMAssociationStatus

Contoh kode berikut menunjukkan cara menggunakan `Update-SSMAssociationStatus`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memperbarui status asosiasi antara instance dan dokumen konfigurasi.

```
Update-SSMAssociationStatus -Name "AWS-UpdateSSMAgent" -InstanceId
    "i-0000293ffd8c57862" -AssociationStatus_Date "2015-02-20T08:31:11Z"
    -AssociationStatus_Name "Pending" -AssociationStatus_Message
    "temporary_status_change" -AssociationStatus_AdditionalInfo "Additional-Config-
    Needed"
```

Output:

```
Name           : AWS-UpdateSSMAgent
InstanceId      : i-0000293ffd8c57862
Date           : 2/23/2017 6:55:22 PM
Status.Name     : Pending
Status.Date     : 2/20/2015 8:31:11 AM
Status.Message  : temporary_status_change
```

```
Status.AdditionalInfo : Additional-Config-Needed
```

- Untuk detail API, lihat [UpdateAssociationStatus](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-SSMDocument

Contoh kode berikut menunjukkan cara menggunakan `Update-SSMDocument`.

Alat untuk PowerShell V4

Contoh 1: Ini membuat versi baru dokumen dengan konten terbaru dari file json yang Anda tentukan. Dokumen harus dalam format JSON. Anda dapat memperoleh versi dokumen dengan cmdlet `"Get-SSMDocumentVersionList"`.

```
Update-SSMDocument -Name RunShellScript -DocumentVersion "1" -Content (Get-Content -Raw "c:\temp\RunShellScript.json")
```

Output:

```

CreatedDate      : 3/1/2017 2:59:17 AM
DefaultVersion   : 1
Description      : Run an updated script
DocumentType     : Command
DocumentVersion  : 2
Hash             : 1d5ce820e999ff051eb4841ed887593daf77120fd76cae0d18a53cc42e4e22c1
HashType        : Sha256
LatestVersion    : 2
Name            : RunShellScript
Owner           : 809632081692
Parameters       : {commands}
PlatformTypes   : {Linux}
SchemaVersion    : 2.0
Sha1            :
Status          : Updating
  
```

- Untuk detail API, lihat [UpdateDocument](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-SSMDocumentDefaultVersion

Contoh kode berikut menunjukkan cara menggunakan `Update-SSMDocumentDefaultVersion`.

Alat untuk PowerShell V4

Contoh 1: Ini memperbarui versi default dokumen. Anda dapat memperoleh versi dokumen yang tersedia dengan cmdlet “Dapatkan- SSMDocumentVersionList”.

```
Update-SSMDocumentDefaultVersion -Name "RunShellScript" -DocumentVersion "2"
```

Output:

```
DefaultVersion Name
-----
2              RunShellScript
```

- Untuk detail API, lihat [UpdateDocumentDefaultVersion](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-SSMMaintenanceWindow

Contoh kode berikut menunjukkan cara menggunakan Update-SSMMaintenanceWindow.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memperbarui nama jendela pemeliharaan.

```
Update-SSMMaintenanceWindow -WindowId "mw-03eb9db42890fb82d" -Name "My-Renamed-MW"
```

Output:

```
AllowUnassociatedTargets : False
Cutoff                   : 1
Duration                 : 2
Enabled                  : True
Name                     : My-Renamed-MW
Schedule                 : cron(0 */30 * * * ? *)
WindowId                 : mw-03eb9db42890fb82d
```

Contoh 2: Contoh ini memungkinkan jendela pemeliharaan.

```
Update-SSMMaintenanceWindow -WindowId "mw-03eb9db42890fb82d" -Enabled $true
```

Output:

```

AllowUnassociatedTargets : False
Cutoff                   : 1
Duration                 : 2
Enabled                  : True
Name                     : My-Renamed-MW
Schedule                 : cron(0 */30 * * * ? *)
WindowId                 : mw-03eb9db42890fb82d

```

Contoh 3: Contoh ini menonaktifkan jendela pemeliharaan.

```
Update-SSMMaintenanceWindow -WindowId "mw-03eb9db42890fb82d" -Enabled $false
```

Output:

```

AllowUnassociatedTargets : False
Cutoff                   : 1
Duration                 : 2
Enabled                  : False
Name                     : My-Renamed-MW
Schedule                 : cron(0 */30 * * * ? *)
WindowId                 : mw-03eb9db42890fb82d

```

- Untuk detail API, lihat [UpdateMaintenanceWindow](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-SSMManagedInstanceRole

Contoh kode berikut menunjukkan cara menggunakan `Update-SSMManagedInstanceRole`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memperbarui peran instance terkelola. Tidak ada output jika perintah berhasil.

```
Update-SSMManagedInstanceRole -InstanceId "mi-08ab247cdf1046573" -IamRole
"AutomationRole"
```

- Untuk detail API, lihat [UpdateManagedInstanceRole](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Update-SSMPatchBaseline

Contoh kode berikut menunjukkan cara menggunakan `Update-SSMPatchBaseline`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menambahkan dua tambalan sebagai ditolak dan satu tambalan yang disetujui ke baseline patch yang ada.

```
Update-SSMPatchBaseline -BaselineId "pb-03da896ca3b68b639" -RejectedPatch
"KB2032276","MS10-048" -ApprovedPatch "KB2124261"
```

Output:

```
ApprovalRules    : Amazon.SimpleSystemsManagement.Model.PatchRuleGroup
ApprovedPatches : {KB2124261}
BaselineId       : pb-03da896ca3b68b639
CreatedDate      : 3/3/2017 5:02:19 PM
Description      : Baseline containing all updates approved for production systems
GlobalFilters    : Amazon.SimpleSystemsManagement.Model.PatchFilterGroup
ModifiedDate     : 3/3/2017 5:22:10 PM
Name             : Production-Baseline
RejectedPatches  : {KB2032276, MS10-048}
```

- Untuk detail API, lihat [UpdatePatchBaseline](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Write-SSMComplianceItem

Contoh kode berikut menunjukkan cara menggunakan `Write-SSMComplianceItem`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menulis item kepatuhan khusus untuk instance terkelola yang diberikan

```
$item = [Amazon.SimpleSystemsManagement.Model.ComplianceItemEntry]::new()
$item.Id = "07Jun2019-3"
$item.Severity="LOW"
$item.Status="COMPLIANT"
$item.Title="Fin-test-1 - custom"
```

```
Write-SSMComplianceItem -ResourceId mi-012dcb3ecea45b678 -ComplianceType
Custom:VSSCompliant2 -ResourceType ManagedInstance -Item $item -
ExecutionSummary_ExecutionTime "07-Jun-2019"
```

- Untuk detail API, lihat [PutComplianceItems](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Write-SSMInventory

Contoh kode berikut menunjukkan cara menggunakan `Write-SSMInventory`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memberikan informasi lokasi rak ke sebuah instance. Tidak ada output jika perintah berhasil.

```
$data = New-Object
[System.Collections.Generic.Dictionary[System.String,System.String]]
$data.Add("RackLocation", "Bay B/Row C/Rack D/Shelf F")

$items = New-Object
[System.Collections.Generic.List[System.Collections.Generic.Dictionary[System.String,
System.String]]]
$items.Add($data)

$customInventoryItem = New-Object Amazon.SimpleSystemsManagement.Model.InventoryItem
$customInventoryItem.CaptureTime = "2016-08-22T10:01:01Z"
$customInventoryItem.Content = $items
$customInventoryItem.TypeName = "Custom:TestRackInfo2"
$customInventoryItem.SchemaVersion = "1.0"

$inventoryItems = @($customInventoryItem)

Write-SSMInventory -InstanceId "i-0cb2b964d3e14fd9f" -Item $inventoryItems
```

- Untuk detail API, lihat [PutInventory](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Write-SSMParameter

Contoh kode berikut menunjukkan cara menggunakan `Write-SSMParameter`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat parameter. Tidak ada output jika perintah berhasil.

```
Write-SSMParameter -Name "Welcome" -Type "String" -Value "helloWorld"
```

Contoh 2: Contoh ini mengubah parameter. Tidak ada output jika perintah berhasil.

```
Write-SSMParameter -Name "Welcome" -Type "String" -Value "Good day, Sunshine!" -  
Overwrite $true
```

- Untuk detail API, lihat [PutParameter](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Contoh Amazon Translate menggunakan Alat untuk PowerShell V4

Contoh kode berikut menunjukkan cara melakukan tindakan dan menerapkan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan Amazon Translate.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

ConvertTo-TRNTargetLanguage

Contoh kode berikut menunjukkan cara menggunakan `ConvertTo-TRNTargetLanguage`.

Alat untuk PowerShell V4

Contoh 1: Mengonversi teks bahasa Inggris yang ditentukan ke bahasa Prancis. Teks yang akan dikonversi juga dapat diteruskan sebagai parameter `-Text`.

```
"Hello World" | ConvertTo-TRNTargetLanguage -SourceLanguageCode en -
TargetLanguageCode fr
```

- Untuk detail API, lihat [TranslateText](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

AWS WAFV2 contoh menggunakan Alat untuk PowerShell V4

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan AWS WAFV2.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

New-WAF2WebACL

Contoh kode berikut menunjukkan cara menggunakan New-WAF2WebACL.

Alat untuk PowerShell V4

Contoh 1: Perintah ini membuat ACL web baru bernama “waf-test”. Harap dicatat bahwa sesuai dokumentasi API layanan, 'DefaultAction' adalah properti wajib. Oleh karena itu, nilai untuk '- DefaultAction _Allow' dan/atau '- DefaultAction _Block' harus ditentukan. Karena '- DefaultAction _Allow' dan '- DefaultAction _Block' bukan properti yang diperlukan, nilai '@ {}' dapat digunakan sebagai placeholder seperti yang ditunjukkan pada contoh di atas.

```
New-WAF2WebACL -Name "waf-test" -Scope REGIONAL -Region eu-
west-1 -VisibilityConfig_CloudWatchMetricsEnabled $true -
VisibilityConfig_SampledRequestsEnabled $true -VisibilityConfig_MetricName "waf-
test" -Description "Test" -DefaultAction-Allow @{}
```

Output:

```

ARN          : arn:aws:wafv2:eu-west-1:139480602983:regional/webacl/waf-
test/19460b3f-db14-4b9a-8e23-a417e1eb007f
Description  : Test
Id           : 19460b3f-db14-4b9a-8e23-a417e1eb007f
LockToken    : 5a0cd5eb-d911-4341-b313-b429e6d6b6ab
Name         : waf-test

```

- Untuk detail API, lihat [CreateWebAcl](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

WorkSpaces contoh menggunakan Alat untuk PowerShell V4

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan Alat AWS untuk PowerShell V4 dengan WorkSpaces.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

Approve-WKSIPRule

Contoh kode berikut menunjukkan cara menggunakan Approve-WKSIPRule.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menambahkan aturan ke Grup IP yang ada

```

$Rule = @(
@{IPRule = "10.1.0.0/0"; RuleDesc = "First Rule Added"},

```

```
@{IPRule = "10.2.0.0/0"; RuleDesc = "Second Rule Added"}
)

Approve-WKSIPRule -GroupId wsipg-abcnx2fcw -UserRule $Rule
```

- Untuk detail API, lihat [AuthorizeIpRules](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Copy-WKSWorkspaceImage

Contoh kode berikut menunjukkan cara menggunakan `Copy-WKSWorkspaceImage`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menyalin ruang kerja Gambar dengan ID tertentu dari us-west-2 ke wilayah saat ini dengan nama "" CopiedImageTest

```
Copy-WKSWorkspaceImage -Name CopiedImageTest -SourceRegion us-west-2 -SourceImageId
wsi-djfoedhw6
```

Output:

```
wsi-456abaqfe
```

- Untuk detail API, lihat [CopyWorkspaceImage](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Edit-WKSClientProperty

Contoh kode berikut menunjukkan cara menggunakan `Edit-WKSClientProperty`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memungkinkan Rekoneksi untuk Klien Ruang Kerja

```
Edit-WKSClientProperty -Region us-west-2 -ClientProperties_ReconnectEnabled
"ENABLED" -ResourceId d-123414a369
```

- Untuk detail API, lihat [ModifyClientProperties](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Edit-WKSSelfServicePermission

Contoh kode berikut menunjukkan cara menggunakan `Edit-WKSSelfServicePermission`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memungkinkan izin layanan mandiri untuk Mengubah jenis komputasi dan Meningkatkan Ukuran Volume untuk Direktori yang ditentukan

```
Edit-WKSSelfservicePermission -Region us-west-2 -ResourceId  
d-123454a369 -SelfservicePermissions_ChangeComputeType ENABLED -  
SelfservicePermissions_IncreaseVolumeSize ENABLED
```

- Untuk detail API, lihat [ModifySelfservicePermissions](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Edit-WKSWorkspaceAccessProperty

Contoh kode berikut menunjukkan cara menggunakan `Edit-WKSWorkspaceAccessProperty`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memungkinkan akses Workspace di Android dan Chrome OS untuk Direktori yang ditentukan

```
Edit-WKSWorkspaceAccessProperty -Region us-west-2 -ResourceId  
d-123454a369 -WorkspaceAccessProperties_DeviceTypeAndroid ALLOW -  
WorkspaceAccessProperties_DeviceTypeChromeOs ALLOW
```

- Untuk detail API, lihat [ModifyWorkspaceAccessProperties](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Edit-WKSWorkspaceCreationProperty

Contoh kode berikut menunjukkan cara menggunakan `Edit-WKSWorkspaceCreationProperty`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini memungkinkan Akses Internet dan Mode Pemeliharaan menjadi true sebagai nilai default saat membuat Workspace

```
Edit-WKSWorkspaceCreationProperty -Region us-west-2 -ResourceId  
d-123454a369 -WorkspaceCreationProperties_EnableInternetAccess $true -  
WorkspaceCreationProperties_EnableMaintenanceMode $true
```

- Untuk detail API, lihat [ModifyWorkspaceCreationProperties](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Edit-WKSWorkspaceProperty

Contoh kode berikut menunjukkan cara menggunakan `Edit-WKSWorkspaceProperty`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengubah Properti Workspace Running Mode menjadi Auto Stop untuk Workspace yang ditentukan

```
Edit-WKSWorkspaceProperty -WorkspaceId ws-w361s100v -Region us-west-2 -  
WorkspaceProperties_RunningMode AUTO_STOP
```

- Untuk detail API, lihat [ModifyWorkspaceProperties](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Edit-WKSWorkspaceState

Contoh kode berikut menunjukkan cara menggunakan `Edit-WKSWorkspaceState`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengubah status Workspace yang ditentukan menjadi Tersedia

```
Edit-WKSWorkspaceState -WorkspaceId ws-w361s100v -Region us-west-2 -WorkspaceState  
AVAILABLE
```

- Untuk detail API, lihat [ModifyWorkspaceState](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-WKSClientProperty

Contoh kode berikut menunjukkan cara menggunakan `Get-WKSClientProperty`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan Properti Klien dari Klien Workspace untuk Direktori yang ditentukan

```
Get-WKSClientProperty -ResourceId d-223562a123
```

- Untuk detail API, lihat [DescribeClientProperties](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-WKSIpGroup

Contoh kode berikut menunjukkan cara menggunakan `Get-WKSIpGroup`.

Alat untuk PowerShell V4

Contoh 1: Sampel ini mendapatkan rincian Grup IP yang ditentukan di wilayah yang ditentukan

```
Get-WKSIpGroup -Region us-east-1 -GroupId wsipg-8m1234v45
```

Output:

```
GroupDesc GroupId      GroupName UserRules
-----
wsipg-8m1234v45 TestGroup {Amazon.WorkSpaces.Model.IpRuleItem,
Amazon.WorkSpaces.Model.IpRuleItem}
```

- Untuk detail API, lihat [DescribeIpGroups](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-WKSTag

Contoh kode berikut menunjukkan cara menggunakan `Get-WKSTag`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil tag untuk Workspace yang diberikan

```
Get-WKSTag -WorkspaceId ws-w361s234r -Region us-west-2
```

Output:

Key	Value
---	-----
auto-delete	no
purpose	Workbench

- Untuk detail API, lihat [DescribeTags](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-WKSWorkspace

Contoh kode berikut menunjukkan cara menggunakan `Get-WKSWorkspace`.

Alat untuk PowerShell V4

Contoh 1: Mengambil detail dari semua pipa Anda WorkSpaces .

```
Get-WKSWorkspace
```

Output:

```

BundleId                : wsb-1a2b3c4d
ComputerName             :
DirectoryId             : d-1a2b3c4d
ErrorCode                :
ErrorMessage             :
IpAddress                :
RootVolumeEncryptionEnabled : False
State                    : PENDING
SubnetId                 :
Username                  : myuser
UserVolumeEncryptionEnabled : False
VolumeEncryptionKey      :
WorkspaceId              : ws-1a2b3c4d
WorkspaceProperties       : Amazon.WorkSpaces.Model.WorkspaceProperties

```

Contoh 2: Perintah ini menunjukkan nilai properti anak **WorkspaceProperties** untuk ruang kerja di **us-west-2** wilayah tersebut. Untuk informasi selengkapnya tentang properti anak **WorkspaceProperties**, lihat https://docs.aws.amazon.com/workspaces/latest/api/API_WorkspaceProperties.html.

```
(Get-WKSWorkspace -Region us-west-2 -WorkspaceId ws-xdaf7hc9s).WorkspaceProperties
```

Output:

```
ComputeTypeName      : STANDARD
RootVolumeSizeGib    : 80
RunningMode           : AUTO_STOP
RunningModeAutoStopTimeoutInMinutes : 60
UserVolumeSizeGib     : 50
```

Contoh 3: Perintah ini menunjukkan nilai properti **RootVolumeSizeGib** anak **WorkspaceProperties** untuk ruang kerja di **us-west-2** wilayah tersebut. Ukuran volume root, dalam GiB, adalah 80.

```
(Get-WKSWorkspace -Region us-west-2 -WorkspaceId ws-xdaf7hc9s).WorkspaceProperties.RootVolumeSizeGib
```

Output:

```
80
```

- Untuk detail API, lihat [DescribeWorkspaces](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-WKSWorkspaceBundle

Contoh kode berikut menunjukkan cara menggunakan `Get-WKSWorkspaceBundle`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil rincian semua bundel Workspace di wilayah saat ini

```
Get-WKSWorkspaceBundle
```

Output:

```
BundleId      : wsb-sfhgqv342
```

```

ComputeType      : Amazon.WorkSpaces.Model.ComputeType
Description      : This bundle is custom
ImageId          : wsi-235aeqges
LastUpdatedTime  : 12/26/2019 06:44:07
Name             : CustomBundleTest
Owner           : 233816212345
RootStorage      : Amazon.WorkSpaces.Model.RootStorage
UserStorage      : Amazon.WorkSpaces.Model.UserStorage

```

- Untuk detail API, lihat [DescribeWorkspaceBundles](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-WKSWorkspaceDirectory

Contoh kode berikut menunjukkan cara menggunakan `Get-WKSWorkspaceDirectory`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mencantumkan detail direktori untuk direktori terdaftar

```
Get-WKSWorkspaceDirectory
```

Output:

```

Alias                : TestWorkspace
CustomerUserName     : Administrator
DirectoryId          : d-123414a369
DirectoryName        : TestDirectory.com
DirectoryType        : MicrosoftAD
DnsIpAddresses       : {172.31.43.45, 172.31.2.97}
IamRoleId            : arn:aws:iam::761234567801:role/workspaces_RoleDefault
IpGroupIds           : {}
RegistrationCode      : WSpdx+4RRT43
SelfservicePermissions : Amazon.WorkSpaces.Model.SelfservicePermissions
State                : REGISTERED
SubnetIds            : {subnet-1m3m7b43, subnet-ard11aba}
Tenancy              : SHARED
WorkspaceAccessProperties : Amazon.WorkSpaces.Model.WorkspaceAccessProperties
WorkspaceCreationProperties :
    Amazon.WorkSpaces.Model.DefaultWorkspaceCreationProperties
WorkspaceSecurityGroupId : sg-0ed2441234a123c43

```

- Untuk detail API, lihat [DescribeWorkspaceDirectories](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-WKSWorkspaceImage

Contoh kode berikut menunjukkan cara menggunakan `Get-WKSWorkspaceImage`.

Alat untuk PowerShell V4

Contoh 1: Sampel ini mengambil semua detail semua gambar di wilayah tersebut

```
Get-WKSWorkspaceImage
```

Output:

```
Description      :This image is copied from another image
ErrorCode        :
ErrorMessage     :
ImageId          : wsi-345ahdjgo
Name             : CopiedImageTest
OperatingSystem  : Amazon.WorkSpaces.Model.OperatingSystem
RequiredTenancy  : DEFAULT
State            : AVAILABLE
```

- Untuk detail API, lihat [DescribeWorkspaceImages](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-WKSWorkspaceSnapshot

Contoh kode berikut menunjukkan cara menggunakan `Get-WKSWorkspaceSnapshot`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menunjukkan stempel waktu snapshot terbaru yang dibuat untuk Workspace yang ditentukan

```
Get-WKSWorkspaceSnapshot -WorkspaceId ws-w361s100v
```

Output:

RebuildSnapshots	RestoreSnapshots
-----	-----
{Amazon.WorkSpaces.Model.Snapshot}	{Amazon.WorkSpaces.Model.Snapshot}

- Untuk detail API, lihat [DescribeWorkspaceSnapshots](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Get-WKSWorkspacesConnectionStatus

Contoh kode berikut menunjukkan cara menggunakan `Get-WKSWorkspacesConnectionStatus`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil status koneksi untuk Workspace yang ditentukan

```
Get-WKSWorkspacesConnectionStatus -WorkspaceId ws-w123s234r
```

- Untuk detail API, lihat [DescribeWorkspacesConnectionStatus](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-WKSIpGroup

Contoh kode berikut menunjukkan cara menggunakan `New-WKSIpGroup`.

Alat untuk PowerShell V4

Contoh 1: Sampel ini membuat grup Ip kosong bernama `FreshEmptyIpGroup`

```
New-WKSIpGroup -GroupName "FreshNewIPGroup"
```

Output:

```
wsipg-w45rty4ty
```

- Untuk detail API, lihat [CreateIpGroup](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-WKSTag

Contoh kode berikut menunjukkan cara menggunakan `New-WKSTag`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menambahkan tag baru ke ruang kerja bernama **aws-wsname**. Tag memiliki kunci "Nama", dan nilai kunci dari **AWS_Workspace**.

```
$tag = New-Object Amazon.WorkSpaces.Model.Tag
$tag.Key = "Name"
$tag.Value = "AWS_Workspace"
New-WKSTag -Region us-west-2 -WorkspaceId ws-wsname -Tag $tag
```

Contoh 2: Contoh ini menambahkan beberapa tag ke ruang kerja bernama **aws-wsname**. Satu tag memiliki kunci "Nama" dan nilai kunci **AWS_Workspace**; tag lainnya memiliki kunci tag "Tahap" dan nilai kunci "Uji".

```
$tag = New-Object Amazon.WorkSpaces.Model.Tag
$tag.Key = "Name"
$tag.Value = "AWS_Workspace"

$tag2 = New-Object Amazon.WorkSpaces.Model.Tag
$tag2.Key = "Stage"
$tag2.Value = "Test"
New-WKSTag -Region us-west-2 -WorkspaceId ws-wsname -Tag $tag,$tag2
```

- Untuk detail API, lihat [CreateTags](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

New-WKSWorkspace

Contoh kode berikut menunjukkan cara menggunakan **New-WKSWorkspace**.

Alat untuk PowerShell V4

Contoh 1: Buat WorkSpace untuk bundel, direktori, dan pengguna yang disediakan.

```
New-WKSWorkspace -Workspace @{ "BundleID" = "wsb-1a2b3c4d"; "DirectoryId" =
  "d-1a2b3c4d"; "UserName" = "USERNAME" }
```

Contoh 2: Contoh ini membuat beberapa WorkSpaces

```
New-WKSWorkspace -Workspace @{ "BundleID" = "wsb-1a2b3c4d"; "DirectoryId"
  = "d-1a2b3c4d"; "UserName" = "USERNAME_1" }, @{ "BundleID" = "wsb-1a2b3c4d";
  "DirectoryId" = "d-1a2b3c4d"; "UserName" = "USERNAME_2" }
```

- Untuk detail API, lihat [CreateWorkspaces](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Register-WKSIpGroup

Contoh kode berikut menunjukkan cara menggunakan `Register-WKSIpGroup`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendaftarkan Grup IP yang ditentukan dengan Direktori yang ditentukan

```
Register-WKSIpGroup -GroupId wsipg-23ahsdres -DirectoryId d-123412e123
```

- Untuk detail API, lihat [AssociateIpGroups](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Register-WKSWorkspaceDirectory

Contoh kode berikut menunjukkan cara menggunakan `Register-WKSWorkspaceDirectory`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendaftarkan direktori yang ditentukan untuk Workspaces Service

```
Register-WKSWorkspaceDirectory -DirectoryId d-123412a123 -EnableWorkDoc $false
```

- Untuk detail API, lihat [RegisterWorkspaceDirectory](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-WKSIpGroup

Contoh kode berikut menunjukkan cara menggunakan `Remove-WKSIpGroup`.

Alat untuk PowerShell V4

Contoh 1: Sampel ini menghapus Grup IP yang ditentukan

```
Remove-WKSIpGroup -GroupId wsipg-32fhgtred
```

Output:

```

Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-WKSIpGroup (DeleteIpGroup)" on target
"wsipg-32fhgtred".
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is
"Y"): Y

```

- Untuk detail API, lihat [DeleteIpGroup](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-WKSTag

Contoh kode berikut menunjukkan cara menggunakan `Remove-WKSTag`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus tag yang terkait dengan Workspace

```
Remove-WKSTag -ResourceId ws-w10b3abcd -TagKey "Type"
```

Output:

```

Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-WKSTag (DeleteTags)" on target "ws-w10b3abcd".
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is
"Y"): Y

```

- Untuk detail API, lihat [DeleteTags](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Remove-WKSWorkspace

Contoh kode berikut menunjukkan cara menggunakan `Remove-WKSWorkspace`.

Alat untuk PowerShell V4

Contoh 1: Mengakhiri beberapa WorkSpaces. penggunaan sakelar `-Force` menghentikan cmdlet agar tidak meminta konfirmasi.

```
Remove-WKSWorkspace -WorkspaceId "ws-1a2b3c4d5", "ws-6a7b8c9d0" -Force
```

Contoh 2: Mengambil koleksi semua Anda WorkSpaces dan pipa IDs ke - WorkspaceId parameter Remove-WKSWorkspace, mengakhiri semua. WorkSpaces Cmdlet akan meminta sebelum masing-masing Workspace dihentikan. Untuk menekan prompt konfirmasi tambahkan sakelar -Force.

```
Get-WKSWorkspaces | Remove-WKSWorkspace
```

Contoh 3: Contoh ini menunjukkan cara meneruskan TerminateRequest objek yang mendefinisikan WorkSpaces yang akan dihentikan. Cmdlet akan meminta konfirmasi sebelum melanjutkan, kecuali parameter sakelar -Force juga ditentukan.

```
$arrRequest = @()
$request1 = New-Object Amazon.WorkSpaces.Model.TerminateRequest
$request1.WorkspaceId = 'ws-12345678'
$arrRequest += $request1
$request2 = New-Object Amazon.WorkSpaces.Model.TerminateRequest
$request2.WorkspaceId = 'ws-abcdefgh'
$arrRequest += $request2
Remove-WKSWorkspace -Request $arrRequest
```

- Untuk detail API, lihat [TerminateWorkspaces](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Reset-WKSWorkspace

Contoh kode berikut menunjukkan cara menggunakan Reset-WKSWorkspace.

Alat untuk PowerShell V4

Contoh 1: Membangun kembali yang ditentukan. Workspace

```
Reset-WKSWorkspace -WorkspaceId "ws-1a2b3c4d"
```

Contoh 2: Mengambil koleksi semua Anda WorkSpaces dan pipa IDs ke - WorkspaceId parameter Reset-WKSWorkspace, menyebabkan yang akan WorkSpaces dibangun kembali.

```
Get-WKSWorkspaces | Reset-WKSWorkspace
```

- Untuk detail API, lihat [RebuildWorkspaces](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Restart-WKSWorkspace

Contoh kode berikut menunjukkan cara menggunakan `Restart-WKSWorkspace`.

Alat untuk PowerShell V4

Contoh 1: Reboot yang ditentukan Workspace.

```
Restart-WKSWorkspace -WorkspaceId "ws-1a2b3c4d"
```

Contoh 2: Reboot beberapa WorkSpaces.

```
Restart-WKSWorkspace -WorkspaceId "ws-1a2b3c4d","ws-5a6b7c8d"
```

Contoh 3: Mengambil koleksi semua Anda WorkSpaces dan pipa IDs ke - `WorkspaceId` parameter `Restart-WKSWorkspace`, menyebabkan restart. WorkSpaces

```
Get-WKSWorkspaces | Restart-WKSWorkspace
```

- Untuk detail API, lihat [RebootWorkspaces](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Stop-WKSWorkspace

Contoh kode berikut menunjukkan cara menggunakan `Stop-WKSWorkspace`.

Alat untuk PowerShell V4

Contoh 1: Menghentikan beberapa WorkSpaces.

```
Stop-WKSWorkspace -WorkspaceId "ws-1a2b3c4d5","ws-6a7b8c9d0"
```

Contoh 2: Mengambil koleksi semua Anda WorkSpaces dan pipa IDs ke - `WorkspaceId` parameter `Stop-WKSWorkspace` WorkSpaces menyebabkan dihentikan.

```
Get-WKSWorkspaces | Stop-WKSWorkspace
```

Contoh 3: Contoh ini menunjukkan cara melewati `StopRequest` objek yang mendefinisikan WorkSpaces yang akan dihentikan.

```
$arrRequest = @()
$request1 = New-Object Amazon.WorkSpaces.Model.StopRequest
$request1.WorkspaceId = 'ws-12345678'
$arrRequest += $request1
$request2 = New-Object Amazon.WorkSpaces.Model.StopRequest
$request2.WorkspaceId = 'ws-abcdefgh'
$arrRequest += $request2
Stop-WKSSpace -Request $arrRequest
```

- Untuk detail API, lihat [StopWorkspaces](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Unregister-WKSSpace

Contoh kode berikut menunjukkan cara menggunakan `Unregister-WKSSpace`.

Alat untuk PowerShell V4

Contoh 1: Contoh ini membatalkan registrasi Grup IP yang ditentukan dari Direktori yang ditentukan

```
Unregister-WKSSpace -GroupId wsipg-12abcdphq -DirectoryId d-123454b123
```

- Untuk detail API, lihat [DisassociateGroups](#) di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Keamanan untuk AWS Produk atau Layanan ini

Keamanan cloud di Amazon Web Services (AWS) merupakan prioritas tertinggi. Sebagai seorang pelanggan AWS, Anda mendapatkan manfaat dari pusat data dan arsitektur jaringan yang dibangun untuk memenuhi persyaratan dari organisasi yang paling sensitif terhadap keamanan. Keamanan adalah tanggung jawab bersama antara Anda AWS dan Anda. [Model Tanggung Jawab Bersama](#) menggambarkan ini sebagai Keamanan dari Cloud dan Keamanan dalam Cloud.

Security of the Cloud - AWS bertanggung jawab untuk melindungi infrastruktur yang menjalankan semua layanan yang ditawarkan di AWS Cloud dan memberi Anda layanan yang dapat Anda gunakan dengan aman. Tanggung jawab keamanan kami adalah prioritas tertinggi di AWS, dan efektivitas keamanan kami secara teratur diuji dan diverifikasi oleh auditor pihak ketiga sebagai bagian dari [Program AWS Kepatuhan](#).

Keamanan di Cloud — Tanggung jawab Anda ditentukan oleh AWS layanan yang Anda gunakan, dan faktor-faktor lain termasuk sensitivitas data Anda, persyaratan organisasi Anda, serta undang-undang dan peraturan yang berlaku.

AWS Produk atau layanan ini mengikuti [model tanggung jawab bersama](#) melalui layanan Amazon Web Services (AWS) tertentu yang didukungnya. Untuk informasi keamanan AWS layanan, lihat [halaman dokumentasi keamanan AWS layanan](#) dan [AWS layanan yang berada dalam lingkup upaya AWS kepatuhan oleh program kepatuhan](#).

Topik

- [Perlindungan data dalam AWS produk atau layanan ini](#)
- [Manajemen Identitas dan Akses](#)
- [Validasi Kepatuhan untuk AWS Produk atau Layanan ini](#)
- [Menerapkan versi TLS minimum di Alat untuk PowerShell](#)
- [Pertimbangan keamanan tambahan untuk Alat untuk PowerShell](#)

Perlindungan data dalam AWS produk atau layanan ini

[Model tanggung jawab AWS bersama model](#) berlaku untuk perlindungan data dalam AWS produk atau layanan ini. 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 kredensial dan mengatur pengguna individu dengan AWS IAM Identity Center atau AWS Identity and Access Management (IAM). Dengan cara itu, setiap pengguna hanya diberi izin yang diperlukan untuk memenuhi tanggung jawab tugasnya. Kami juga menyarankan supaya Anda mengamankan data dengan cara-cara berikut:

- Gunakan autentikasi multi-faktor (MFA) pada setiap akun.
- Gunakan SSL/TLS untuk berkomunikasi dengan sumber daya. AWS Kami mensyaratkan TLS 1.2 dan menganjurkan TLS 1.3.
- 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 produk atau layanan ini 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 data

Fitur utama dari setiap layanan aman adalah bahwa informasi dienkripsi ketika tidak aktif digunakan.

Enkripsi saat Data Tidak Berpindah

Itu sendiri Alat AWS untuk PowerShell tidak menyimpan data pelanggan selain kredensial yang dibutuhkan untuk berinteraksi dengan AWS layanan atas nama pengguna.

Jika Anda menggunakan Alat AWS untuk PowerShell untuk memanggil AWS layanan yang mentransmisikan data pelanggan ke komputer lokal Anda untuk penyimpanan, lihat bagian Keamanan & Kepatuhan dalam Panduan Pengguna layanan tersebut untuk informasi tentang bagaimana data tersebut disimpan, dilindungi, dan dienkripsi.

Enkripsi Saat Data Berpindah

Secara default, semua data yang dikirimkan dari komputer klien yang menjalankan titik akhir Alat AWS untuk PowerShell dan AWS layanan dienkripsi dengan mengirimkan semuanya melalui koneksi HTTPS/TLS.

Anda tidak perlu melakukan apapun untuk mengaktifkan penggunaan HTTPS/TLS. Hal ini selalu diaktifkan.

Manajemen Identitas dan Akses

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. AWS IAM adalah Layanan AWS yang dapat Anda gunakan tanpa biaya tambahan.

Topik

- [Audiens](#)
- [Mengautentikasi dengan identitas](#)
- [Mengelola akses menggunakan kebijakan](#)
- [Bagaimana Layanan AWS bekerja dengan IAM](#)
- [Memecahkan masalah AWS identitas dan akses](#)

Audiens

Cara Anda menggunakan AWS Identity and Access Management (IAM) berbeda, tergantung pada pekerjaan yang Anda lakukan. AWS

Pengguna layanan — Jika Anda menggunakan Layanan AWS untuk melakukan pekerjaan Anda, maka administrator Anda memberi Anda kredensi dan izin yang Anda butuhkan. Saat Anda menggunakan lebih banyak AWS fitur untuk melakukan pekerjaan Anda, Anda mungkin memerlukan izin tambahan. Memahami cara akses dikelola dapat membantu Anda meminta izin yang tepat dari administrator Anda. Jika Anda tidak dapat mengakses fitur AWS, lihat [Memecahkan masalah AWS identitas dan akses](#) atau panduan pengguna yang Layanan AWS Anda gunakan.

Administrator layanan — Jika Anda bertanggung jawab atas AWS sumber daya di perusahaan Anda, Anda mungkin memiliki akses penuh ke AWS. Tugas Anda adalah menentukan AWS fitur dan sumber daya 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. Untuk mempelajari lebih lanjut tentang bagaimana perusahaan Anda dapat menggunakan IAM AWS, lihat panduan pengguna yang Layanan AWS Anda gunakan.

Administrator IAM – Jika Anda adalah administrator IAM, Anda mungkin ingin belajar dengan lebih detail tentang cara Anda menulis kebijakan untuk mengelola akses ke AWS. Untuk melihat contoh kebijakan AWS berbasis identitas yang dapat Anda gunakan di IAM, lihat panduan pengguna yang Anda gunakan. Layanan AWS

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 kredensi 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 kredensial 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 kredensial yang disediakan melalui sumber identitas. Ketika identitas federasi mengakses Akun AWS, mereka mengambil peran, dan peran memberikan kredensi 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 secara 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 kredensi 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 instance yang dilampirkan ke instance. Profil instance berisi peran dan memungkinkan program yang berjalan pada EC2 instance untuk mendapatkan kredensi 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 AWS bekerja dengan IAM

Untuk mendapatkan tampilan tingkat tinggi tentang cara Layanan AWS bekerja dengan sebagian besar fitur IAM, lihat [AWS layanan yang bekerja dengan IAM di Panduan Pengguna IAM](#).

Untuk mempelajari cara menggunakan spesifik Layanan AWS dengan IAM, lihat bagian keamanan dari Panduan Pengguna layanan yang relevan.

Memecahkan masalah AWS identitas dan akses

Gunakan informasi berikut untuk membantu Anda mendiagnosis dan memperbaiki masalah umum yang mungkin Anda temui saat bekerja dengan AWS dan IAM.

Topik

- [Saya tidak berwenang untuk melakukan tindakan di AWS](#)
- [Saya tidak berwenang untuk melakukan iam: PassRole](#)
- [Saya ingin mengizinkan orang di luar saya Akun AWS untuk mengakses AWS sumber daya saya](#)

Saya tidak berwenang untuk melakukan tindakan di AWS

Jika Anda menerima pesan kesalahan bahwa Anda tidak memiliki otorisasi untuk melakukan tindakan, kebijakan Anda harus diperbarui agar Anda dapat melakukan tindakan tersebut.

Contoh kesalahan berikut terjadi ketika pengguna IAM `mateojackson` mencoba menggunakan konsol untuk melihat detail tentang suatu sumber daya `my-example-widget` rekaan, tetapi tidak memiliki izin `aws:GetWidget` rekaan.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
aws:GetWidget on resource: my-example-widget
```

Dalam hal ini, kebijakan untuk pengguna `mateojackson` harus diperbarui untuk mengizinkan akses ke sumber daya `my-example-widget` dengan menggunakan tindakan `aws:GetWidget`.

Jika Anda memerlukan bantuan, hubungi AWS administrator Anda. Administrator Anda adalah orang yang memberi Anda kredensial masuk.

Saya tidak berwenang untuk melakukan iam: PassRole

Jika Anda menerima kesalahan yang tidak diizinkan untuk melakukan `iam:PassRole` tindakan, kebijakan Anda harus diperbarui agar Anda dapat meneruskan peran AWS.

Beberapa Layanan AWS memungkinkan Anda untuk meneruskan peran yang ada ke layanan tersebut alih-alih membuat peran layanan baru atau peran terkait layanan. Untuk melakukannya, Anda harus memiliki izin untuk meneruskan peran ke layanan.

Contoh kesalahan berikut terjadi ketika pengguna IAM bernama `marymajor` mencoba menggunakan konsol tersebut untuk melakukan tindakan di AWS. Namun, tindakan tersebut memerlukan layanan untuk mendapatkan izin yang diberikan oleh peran layanan. Mary tidak memiliki izin untuk meneruskan peran tersebut pada layanan.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
iam:PassRole
```

Dalam kasus ini, kebijakan Mary harus diperbarui agar dia mendapatkan izin untuk melakukan tindakan `iam:PassRole` tersebut.

Jika Anda memerlukan bantuan, hubungi AWS administrator Anda. Administrator Anda adalah orang yang memberi Anda kredensial masuk.

Saya ingin mengizinkan orang di luar saya Akun AWS untuk mengakses AWS sumber daya saya

Anda dapat membuat peran yang dapat digunakan pengguna di akun lain atau orang-orang di luar organisasi Anda untuk mengakses sumber daya Anda. Anda dapat menentukan siapa saja yang dipercaya untuk mengambil peran tersebut. Untuk layanan yang mendukung kebijakan berbasis sumber daya atau daftar kontrol akses (ACLs), Anda dapat menggunakan kebijakan tersebut untuk memberi orang akses ke sumber daya Anda.

Untuk mempelajari selengkapnya, periksa referensi berikut:

- Untuk mempelajari apakah AWS mendukung fitur ini, lihat [Bagaimana Layanan AWS bekerja dengan IAM](#).
- Untuk mempelajari cara menyediakan akses ke sumber daya Anda di seluruh sumber daya Akun AWS yang Anda miliki, lihat [Menyediakan akses ke pengguna IAM di pengguna lain Akun AWS yang Anda miliki](#) di Panduan Pengguna IAM.

- Untuk mempelajari cara menyediakan akses ke sumber daya Anda kepada pihak ketiga Akun AWS, lihat [Menyediakan akses yang Akun AWS dimiliki oleh pihak ketiga](#) dalam Panduan Pengguna IAM.
- Untuk mempelajari cara memberikan akses melalui federasi identitas, lihat [Menyediakan akses ke pengguna terautentikasi eksternal \(federasi identitas\)](#) dalam Panduan Pengguna IAM.
- Untuk mempelajari perbedaan antara menggunakan peran dan kebijakan berbasis sumber daya untuk akses lintas akun, lihat [Akses sumber daya lintas akun di IAM di Panduan Pengguna IAM](#).

Validasi Kepatuhan untuk AWS Produk atau Layanan ini

Untuk mempelajari apakah an Layanan AWS berada dalam lingkup program kepatuhan tertentu, lihat [Layanan AWS di Lingkup oleh Program Kepatuhan Layanan AWS](#) dan pilih program kepatuhan yang Anda minati. Untuk informasi umum, lihat [Program AWS Kepatuhan Program AWS](#).

Anda dapat mengunduh laporan audit pihak ketiga menggunakan AWS Artifact. Untuk informasi selengkapnya, lihat [Mengunduh Laporan di AWS Artifact](#).

Tanggung jawab kepatuhan Anda saat menggunakan Layanan AWS ditentukan oleh sensitivitas data Anda, tujuan kepatuhan perusahaan Anda, dan hukum dan peraturan yang berlaku. AWS menyediakan sumber daya berikut untuk membantu kepatuhan:

- [Kepatuhan dan Tata Kelola Keamanan](#) – Panduan implementasi solusi ini membahas pertimbangan arsitektur serta memberikan langkah-langkah untuk menerapkan fitur keamanan dan kepatuhan.
- [Referensi Layanan yang Memenuhi Syarat HIPAA](#) — Daftar layanan yang memenuhi syarat HIPAA. Tidak semua memenuhi Layanan AWS syarat HIPAA.
- [AWS Sumber Daya AWS](#) — Kumpulan buku kerja dan panduan ini mungkin berlaku untuk industri dan lokasi Anda.
- [AWS Panduan Kepatuhan Pelanggan](#) - Memahami model tanggung jawab bersama melalui lensa kepatuhan. Panduan ini merangkum praktik terbaik untuk mengamankan Layanan AWS dan memetakan panduan untuk kontrol keamanan di berbagai kerangka kerja (termasuk Institut Standar dan Teknologi Nasional (NIST), Dewan Standar Keamanan Industri Kartu Pembayaran (PCI), dan Organisasi Internasional untuk Standardisasi (ISO)).
- [Mengevaluasi Sumber Daya dengan Aturan](#) dalam Panduan AWS Config Pengembang — AWS Config Layanan menilai seberapa baik konfigurasi sumber daya Anda mematuhi praktik internal, pedoman industri, dan peraturan.

- [AWS Security Hub](#)— Ini Layanan AWS memberikan pandangan komprehensif tentang keadaan keamanan Anda di dalamnya AWS. Security Hub menggunakan kontrol keamanan untuk sumber daya AWS Anda serta untuk memeriksa kepatuhan Anda terhadap standar industri keamanan dan praktik terbaik. Untuk daftar layanan dan kontrol yang didukung, lihat [Referensi kontrol Security Hub](#).
- [Amazon GuardDuty](#) — Ini Layanan AWS mendeteksi potensi ancaman terhadap beban kerja Akun AWS, kontainer, dan data Anda dengan memantau lingkungan Anda untuk aktivitas mencurigakan dan berbahaya. GuardDuty dapat membantu Anda mengatasi berbagai persyaratan kepatuhan, seperti PCI DSS, dengan memenuhi persyaratan deteksi intrusi yang diamanatkan oleh kerangka kerja kepatuhan tertentu.
- [AWS Audit Manager](#)— Ini Layanan AWS membantu Anda terus mengaudit AWS penggunaan Anda untuk menyederhanakan cara Anda mengelola risiko dan kepatuhan terhadap peraturan dan standar industri.

AWS Produk atau layanan ini mengikuti [model tanggung jawab bersama](#) melalui layanan Amazon Web Services (AWS) tertentu yang didukungnya. Untuk informasi keamanan AWS layanan, lihat [halaman dokumentasi keamanan AWS layanan](#) dan [AWS layanan yang berada dalam lingkup upaya AWS kepatuhan oleh program kepatuhan](#).

Menerapkan versi TLS minimum di Alat untuk PowerShell

Untuk meningkatkan keamanan saat berkomunikasi dengan AWS layanan, Anda harus mengonfigurasi Alat PowerShell untuk menggunakan versi TLS yang sesuai. Untuk informasi tentang cara melakukannya, lihat [Menerapkan versi TLS minimum di Panduan AWS SDK untuk .NET Pengembang](#).

Pertimbangan keamanan tambahan untuk Alat untuk PowerShell

Topik ini berisi pertimbangan keamanan selain topik keamanan yang dibahas di bagian sebelumnya.

Pencatatan informasi sensitif

Beberapa operasi alat ini mungkin mengembalikan informasi yang dapat dianggap sensitif, termasuk informasi dari variabel lingkungan. Paparan informasi ini mungkin mewakili risiko keamanan dalam skenario tertentu; misalnya, informasi dapat dimasukkan dalam log integrasi berkelanjutan dan penyebaran berkelanjutan (CI/CD). Oleh karena itu penting bahwa Anda meninjau ketika Anda

memasukkan output tersebut sebagai bagian dari log Anda, dan menekan output ketika tidak diperlukan. Untuk informasi tambahan tentang melindungi data sensitif, lihat [Perlindungan data dalam AWS produk atau layanan ini](#).

Pertimbangkan praktik terbaik berikut:

- Jangan gunakan variabel lingkungan untuk menyimpan nilai sensitif untuk sumber daya tanpa server Anda. Sebagai gantinya, mintalah kode tanpa server Anda secara terprogram mengambil rahasia dari toko rahasia (misalnya,). AWS Secrets Manager
- Tinjau konten log build Anda untuk memastikan bahwa log tersebut tidak berisi informasi sensitif. Pertimbangkan pendekatan seperti perpipaan to /dev/null atau menangkap output sebagai bash atau PowerShell variabel untuk menekan output perintah.
- Pertimbangkan akses log Anda dan cakupan akses yang tepat untuk kasus penggunaan Anda.

Referensi Cmdlet untuk Alat untuk PowerShell

Alat untuk PowerShell menyediakan cmdlet yang dapat Anda gunakan untuk mengakses AWS layanan. Untuk melihat cmdlet apa yang tersedia, lihat Referensi [Alat AWS untuk PowerShell Cmdlet](#).

Riwayat dokumen

Topik berikut menjelaskan perubahan signifikan pada dokumentasi untuk Alat AWS untuk PowerShell.

Kami juga memperbarui dokumentasi secara berkala dalam menanggapi umpan balik konsumen. Untuk mengirim umpan balik tentang suatu topik, gunakan tombol umpan balik di samping "Apakah halaman ini membantu Anda?" terletak di bagian bawah setiap halaman.

Untuk informasi tambahan tentang perubahan dan pembaruan Alat AWS untuk PowerShell, lihat [catatan rilis](#). Untuk notifikasi tentang pembaruan dokumentasi ini, Anda dapat berlangganan ke [umpan RSS](#).

Perubahan	Deskripsi	Tanggal
Apa yang baru	Menerbitkan panduan pengguna untuk versi 5 (pratinjau) dari Alat AWS untuk PowerShell.	28 Mei 2025
Observabilitas	Mengumumkan rilis GA untuk observabilitas	Februari 10, 2025
Apa yang baru	Menambahkan informasi tentang perilaku default baru untuk perlindungan integritas.	Januari 15, 2025
Apa yang baru	Menambahkan informasi tentang rilis pratinjau pertama Alat AWS untuk PowerShell versi 5.	November 18, 2024
Pipelining, output, dan iterasi	Mengganti konten tentang \$AWSHistory, yang telah usang.	Oktober 10, 2024
Observabilitas	Menambahkan informasi pratinjau tentang observabi	September 13, 2024

litas di Alat AWS untuk PowerShell, yang memungkinkan pengumpulan data telemetri.

[Menginstal Alat AWS untuk PowerShell pada Windows](#)

Menambahkan informasi tentang membuka blokir file ZIP sebelum mengekstrak konten.

Agustus 5, 2024

[Keterangan tentang EC2 - Classic](#)

Menghapus informasi tentang EC2 -Classic, yang telah pensiun.

Agustus 1, 2024

[Contoh Kode](#)

Termasuk sebuah chapter dengan contoh cmdlet.

April 17, 2024

[Pertimbangan keamanan tambahan](#)

Termasuk informasi tentang potensi pencatatan data sensitif.

16 April 2024

[Konfigurasikan otentikasi alat dengan AWS](#)

Menambahkan informasi tentang dukungan untuk SSO di. Alat AWS untuk PowerShell

Maret 15, 2024

[Referensi Cmdlet untuk Alat untuk PowerShell](#)

Menambahkan bagian dengan link ke Tools untuk referensi PowerShell cmdlet.

17 November 2023

[Termasuk lebih banyak pembaruan praktik terbaik IAM](#)

Memperbarui panduan untuk menyelaraskan dengan praktik terbaik IAM. Untuk informasi lebih lanjut, lihat [Praktik terbaik keamanan di IAM](#).

12 Oktober 2023

Menginstal di Windows	Menghapus informasi tentang menginstal Alat untuk Windows PowerShell dengan menggunakan MSI, yang telah usang.	25 September 2023
Pembaruan praktik terbaik IAM	Memperbarui panduan untuk menyelaraskan dengan praktik terbaik IAM. Untuk informasi lebih lanjut, lihat Praktik terbaik keamanan di IAM .	September 8, 2023
Pipelining dan \$ AWSHistory	Menambahkan IncludeSensitiveData parameter ke Set-AWSHistoryConfiguration cmdlet.	9 Maret 2023
Menggunakan ClientConfig parameter dalam cmdlet	Menambahkan informasi tentang dukungan untuk ClientConfig parameter.	28 Oktober 2022
Luncurkan EC2 Instans Amazon Menggunakan Windows PowerShell	Menambahkan catatan tentang pensiun EC2 -Classic.	26 Juli 2022
Alat AWS untuk PowerShell Versi 4	Menambahkan informasi tentang versi 4, termasuk instruksi pemasangan baik untuk Windows maupun Linux/macOS , dan topik migrasi yang menjelaskan perbedaan dari versi 3 dan menambahkan fitur baru.	21 November 2019

[Alat AWS untuk PowerShell](#)
[3.3.563](#)

Menambahkan informasi tentang cara menginstal dan menggunakan versi pratinjau modul `AWS.Tools.Common`. Modul baru ini memecah paket monolitik lama menjadi satu modul bersama dan satu modul per AWS layanan.

18 Oktober 2019

[Alat AWS untuk PowerShell](#)
[3.3.343.0](#)

Menambahkan informasi ke bagian [Menggunakan Alat AWS untuk PowerShell bagian](#) yang memperkenalkan AWS Lambda Tools for PowerShell for PowerShell Core developer untuk membangun AWS Lambda fungsi.

11 September 2018

[AWS Tools for Windows PowerShell 3.1.31.0](#)

Menambahkan informasi ke bagian [Memulai](#) tentang cmdlet baru yang menggunakan Security Assertion Markup Language (SAML) untuk mendukung konfigurasi identitas federasi untuk pengguna.

1 Desember 2015

[AWS Tools for Windows PowerShell 2.3.19](#)

Menambahkan informasi ke bagian [Cmdlets Discovery and Aliases](#) tentang cmdlet baru yang dapat membantu pengguna menemukan `Get-AWSCmdletName` cmdlet yang diinginkan dengan lebih mudah. AWS

5 Februari 2015

[AWS Tools for Windows](#)

[PowerShell 1.1.1.0](#)

15 Mei 2013

Output koleksi dari cmdlet selalu disebutkan ke pipeline. PowerShell Dukungan otomatis untuk panggilan layanan pageable. Variabel \$ AWSHistory shell baru mengumpulkan respons layanan dan permintaan layanan opsional. AWSRegion instance menggunakan bidang Region alih-alih SystemName untuk membantu pipelining. Remove-S3Bucket mendukung opsi - DeleteObjects switch. Memperbaiki masalah kegunaan dengan Set-AWSCredentials. Inisialisasi- AWSDefaults laporan dari mana ia memperoleh kredensi dan data wilayah. Stop-EC2Instance menerima Amazon.EC2.Model.Reservation instance sebagai input. Daftar Umum <T> jenis parameter diganti dengan jenis array (T[]). Cmdlet yang menghapus atau menghentikan sumber daya meminta konfirmasi sebelum penghapusan. Write-S3Object mendukung konten teks in-line untuk diunggah ke Amazon S3.

[AWS Tools for Windows PowerShell 1.0.1.0](#)

21 Desember 2012

Lokasi instalasi PowerShell I modul Tools for Windows telah berubah sehingga lingkungan yang menggunakan Windows PowerShell versi 3 dapat memanfaatkan pemuatan otomatis. Modul dan file pendukung sekarang dipasang pada subfolder `AWSPowerShell` di bawah `AWS ToolsPowerShell`. File dari versi sebelumnya yang ada di folder `AWS ToolsPowerShell` akan dihapus secara otomatis oleh installer. `PSModulePath` Untuk Windows PowerShell (semua versi) diperbarui dalam rilis ini untuk berisi folder induk modul (`AWS ToolsPowerShell`). Untuk sistem dengan Windows PowerShell versi 2, pintasan Start Menu diperbarui untuk mengimpor modul dari lokasi baru dan kemudian dijalankan `Initialize-AWSDefaults`. Untuk sistem dengan Windows PowerShell versi 3, pintasan Start Menu diperbarui untuk menghapus `Import-Module` perintah, hanya `Initialize-AWSDefaults` menyisakan. Jika Anda mengedit PowerShell profil Anda untuk

melakukan `AWSPowerShell.ps1` file, Anda perlu memperbaruinya untuk menunjuk ke lokasi baru file (atau, jika menggunakan PowerShell versi 3, hapus `Import-Module` pernyataan karena tidak lagi diperlukan). `Import-Module` Sebagai hasil dari perubahan ini, PowerShell modul Tools for Windows sekarang terdaftar sebagai modul yang tersedia saat mengeksekusi `Get-Module -ListAvailable`. Selain itu, untuk pengguna Windows PowerShell versi 3, eksekusi cmdlet apa pun yang diekspor oleh modul akan secara otomatis memuat modul di PowerShell shell saat ini tanpa perlu menggunakan terlebih dahulu `Import-Module`. Hal ini memungkinkan penggunaan interaktif cmdlet pada sistem dengan kebijakan eksekusi yang melarang eksekusi skrip.

[AWS Tools for Windows PowerShell 1.0.0.0](#)

Rilis awal

6 Desember 2012

Terjemahan disediakan oleh mesin penerjemah. Jika konten terjemahan yang diberikan bertentangan dengan versi bahasa Inggris aslinya, utamakan versi bahasa Inggris.