Panduan Pengguna

AWS Tools for PowerShell

AWS Tools for PowerShell: Panduan Pengguna

Table of Contents

Apa yang dimaksud dengan AWS Tools for PowerShell?

AWS Tools for PowerShellIni adalah satu set PowerShell modul yang dibangun di atas fungsionalitas yang diekspos olehAWS SDK for .NET. AWS Tools for PowerShellMemungkinkan 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 AWS API kueri HTTP layanan. Misalnya, cmdlet untuk PowerShell pipa AWS Tools for PowerShell pendukung—yaitu, Anda dapat menyalurkan PowerShell objek masuk dan keluar dari cmdlet.

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

AWS Tools for PowerShell mendukung set layanan yang sama dan Daerah AWS yang didukung oleh SDK. Anda dapat menginstal AWS Tools for PowerShell pada komputer yang menjalankan sistem operasi Windows, Linux, atau MacOS.

> **Note**
>
> AWS Tools for PowerShell versi 4 adalah rilis utama terbaru, dan merupakan pembaruan kesesuaian-kebelakang ke AWS Tools for 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 rekomendasikan Anda mengujinya secara menyeluruh sebelum memperbaruinya. Untuk informasi selengkapnya tentang perubahan di versi 4, lihat Migrasi dari AWS Tools for PowerShell Versi 3.3 ke Versi 4.

AWS Tools for PowerShell tersedia sebagai tiga paket yang berbeda berikut ini:

- `AWS.Tools`
- AWSPowerShell.NetCore
- AWSPowerShell

Pemeliharaan dan dukungan untuk versi utama SDK

Untuk informasi tentang pemeliharaan dan dukungan untuk versi utama SDK dan dependensi yang mendasarinya, lihat berikut di Panduan Referensi SDK dan Alat AWS:

- AWSKebijakan pemeliharaan SDK dan alat
- AWSMatriks dukungan versi SDK dan alat

`AWS.Tools` — Versi termodulasi dari AWS Tools for PowerShell

Versi ini AWS Tools for 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 dari AWS Tools for PowerShell dan berjalan di 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 layanan AWS, misalnya, `AWS.Tools.EC2`, `AWS.Tools.IAM`, `AWS.Tools.S3`, dan seterusnya.

Modul `AWS.Tools.Installer` menyediakan cmdlet yang memungkinkan Anda untuk menginstal, memperbarui, dan menghapus modul untuk masing-masing 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. PowerShellsecara 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 AWS Tools for PowerShell ini pada komputer yang menjalankan:

- PowerShell Core 6.0 atau yang lebih baru di Windows, Linux, atau macOS.
- Windows PowerShell 5.1 atau yang lebih baru pada 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`*.

AWSPowerShell. NetCore - Versi modul tunggal dari AWS Tools for PowerShell

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

Anda dapat menginstal versi AWS Tools for PowerShell ini pada komputer yang menjalankan:

- 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

Versi ini AWS Tools for 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 modul besar tunggal yang berisi dukungan untuk semua layanan AWS.

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

Cara menggunakan panduan ini

Panduan ini dibagi menjadi beberapa bagian utama berikut.

Menginstal AWS Tools for PowerShell

Bagian ini menjelaskan cara menginstal AWS Tools for PowerShell. Ini mencakup 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 AWS Tools for PowerShell, seperti menentukan kredensial dan Wilayah AWS, menemukan cmdlet untuk layanan tertentu, dan menggunakan alias untuk cmdlet.

Bekerja dengan AWS layanan di AWS Tools for PowerShell

Bagian ini berisi informasi tentang penggunaan AWS Tools for PowerShell untuk melakukan beberapa tugas AWS yang paling umum.

Menginstal AWS Tools for PowerShell

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

Topik

Menginstal AWS Tools for PowerShell pada Windows

Komputer berbasis Windows dapat menjalankan salah satu opsi AWS Tools for PowerShell paket:

- **`AWS.Tools`**- Versi termodulasi dari. AWS Tools for PowerShell Setiap AWS layanan didukung oleh modul kecil individualnya sendiri, dengan modul dukungan bersama `AWS.Tools.Common` dan`AWS.Tools.Installer`.
- AWSPowerShell. NetCore- Versi tunggal, modul besar dari. AWS Tools for PowerShell Semua AWS layanan didukung oleh modul tunggal dan besar ini.

  > **Note**
  >
  > Ketahuilah bahwa modul tunggal mungkin terlalu besar untuk digunakan dengan AWS Lambdafungsi. Sebagai gantinya, gunakan versi termodulasi yang ditunjukkan di atas.

- AWSPowerShell- Versi lama khusus Windows, tunggal, modul besar dari. AWS Tools for 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 untuk Windows PowerShell (AWSPowerShell modul) diinstal secara default pada semua Gambar Mesin Amazon (AMI) berbasis Windows.

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

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

   > **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**
>
> Cmdlet `Install-AWSToolsModule` mengunduh semua modul yang diminta dari `PSRepository` bernama `PSGallery` (https://www.powershellgallery.com/) dan menganggapnya sebagai sumber terpercaya. 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 AWS Tools for 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 AWSPowerShell. NetCore pada Windows

Anda dapat menginstal AWSPowerShell. 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 AWSPowerShell. 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.
- Menginstal dari PowerShell Galeri menggunakan `Install-Module` cmdlet, seperti yang dijelaskan dalam prosedur berikut.

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

Untuk menginstal AWSPowerShell. NetCore dari PowerShell Galeri, komputer Anda harus menjalankan PowerShell 5.0 atau lebih baru, atau berjalan PowerShellGetpada PowerShell 3 atau lebih baru. Jalankan perintah berikut.

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

Jika Anda menjalankan PowerShell sebagai administrator, perintah sebelumnya akan diinstal AWS Tools for PowerShell untuk semua pengguna di komputer. Jika Anda menjalankan PowerShell sebagai pengguna standar tanpa izin administrator, perintah yang sama akan diinstal hanya AWS Tools for 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 AWSPowerShell.NetCore -Scope CurrentUser
```

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

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

Untuk memuat AWSPowerShell. 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 AWSPowerShell di Windows PowerShell

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

- Mengunduh modul AWSPowerShelldari.zip dan mengekstraknya di salah satu direktori modul. Anda dapat menemukan direktori modul Anda dengan menampilkan nilai variabel `PSModulePath` lingkungan.
- Menginstal dari PowerShell Galeri menggunakan `Install-Module` cmdlet seperti yang dijelaskan dalam prosedur berikut.

Untuk menginstal AWSPowerShell dari PowerShell Galeri menggunakan cmdlet Install-Module

Anda dapat menginstal AWSPowerShell dari PowerShell Galeri jika Anda menjalankan PowerShell 5.0 atau lebih baru, atau telah diinstal PowerShellGetpada PowerShell 3 atau lebih baru. Anda dapat menginstal dan memperbarui AWSPowerShell dari PowerShellGaleri Microsoft dengan menjalankan perintah berikut.

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

Untuk memuat AWSPowerShell modul 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.

> **Note**
>
> Alat untuk Windows diinstal PowerShell secara default di semua Gambar Mesin Amazon (AMI) berbasis Windows.

Aktifkan Eksekusi Skrip

Untuk memuat AWS Tools for 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 PS ModulePath untuk memasukkan 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...
```

Versioning

AWS merilis versi baru secara AWS Tools for PowerShell berkala untuk mendukung AWS layanan dan fitur baru. Untuk menentukan versi Tools yang telah Anda instal, jalankan Get-AWSPowerShellVersion cmdlet.

```
PS > Get-AWSPowerShellVersion

Tools for PowerShell
Version 4.1.11.0
Copyright 2012-2021 Amazon.com, Inc. or its affiliates. All Rights Reserved.

Amazon Web Services SDK for .NET
Core Runtime Version 3.7.0.12
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.

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

Service                            Noun Prefix Module Name                        SDK

 Assembly

 Version
-------                            ----------- -----------
 -----------
Alexa For Business                 ALXB        AWS.Tools.AlexaForBusiness
 3.7.0.11
Amplify Backend                    AMPB        AWS.Tools.AmplifyBackend
 3.7.0.11
Amazon API Gateway                 AG          AWS.Tools.APIGateway
 3.7.0.11
Amazon API Gateway Management API  AGM         AWS.Tools.ApiGatewayManagementApi
 3.7.0.11
Amazon API Gateway V2              AG2         AWS.Tools.ApiGatewayV2
 3.7.0.11
Amazon Appflow                     AF          AWS.Tools.Appflow
 3.7.1.4
Amazon Route 53                    R53         AWS.Tools.Route53
 3.7.0.12
Amazon Route 53 Domains            R53D        AWS.Tools.Route53Domains
 3.7.0.11
Amazon Route 53 Resolver           R53R        AWS.Tools.Route53Resolver
 3.7.1.5
Amazon Simple Storage Service (S3) S3          AWS.Tools.S3
 3.7.0.13
...
```

Untuk menentukan versi PowerShell yang Anda jalankan, masukkan `$PSVersionTable` untuk melihat isi variabel VersionTable otomatis $PS.

```
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 AWS Tools for PowerShell pada Windows

Secara berkala, saat versi terbaru dirilis, Anda harus memperbarui versi yang Anda jalankan secara lokal. AWS Tools for 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**
>
> Cmdlet `Update-AWSToolsModule` mengunduh semua modul dari `PSRepository` bernama `PSGallery` (https://www.powershellgallery.com/) dan menganggapnya sebagai sumber terpercaya. 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 yang lebih baru dari. AWSPowerShell NetCore, hapus instalan 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 AWS Tools for PowerShell konsol khusus dari menu Start Anda.

Menginstal AWS Tools for PowerShell di Linux atau macOS

Topik ini memberikan petunjuk tentang cara menginstal AWS Tools for PowerShell di Linux atau macOS.

Gambaran Umum Pengaturan

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

- **`AWS.Tools`**— Versi termodulasi dari. AWS Tools for PowerShell Setiap AWS layanan didukung oleh modul kecil individualnya sendiri, dengan modul dukungan bersama`AWS.Tools.Common`.
- AWSPowerShell. NetCore— Versi modul tunggal dan besar dari. AWS Tools for PowerShell Semua AWS layanan didukung oleh modul tunggal dan besar ini.

> **Note**
>
> Ketahuilah bahwa modul tunggal mungkin terlalu besar untuk digunakan dengan AWS Lambdafungsi. 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 Initialize- AWSDefaultConfiguration cmdlet untuk memberikan kredensil 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 di PowerShell macOS di situs web Microsoft.

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

Anda dapat menginstal versi termodulasi AWS Tools for 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**
>
> Cmdlet `Install-AWSToolsModule` mengunduh semua modul yang diminta dari `PSRepository` bernama `PSGallery` (https://www.powershellgallery.com/) dan menganggap repositori sebagai sumber terpercaya. 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 AWSPowerShell. NetCore di Linux atau macOS

Untuk meningkatkan ke rilis yang lebih baru dari. AWSPowerShell NetCore, ikuti instruksi diMemperbarui AWS Tools for PowerShell di Linux atau macOS. Copot pemasangan versi sebelumnya. AWSPowerShell NetCore pertama.

Anda dapat menginstal AWSPowerShell. 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 AWSPowerShell. 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 AWSPowerShell. 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 AWS Tools for PowerShell untuk semua pengguna komputer. Untuk melakukan ini, jalankan perintah berikut dalam PowerShell sesi yang telah Anda mulai`sudo 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…
```

AWSPowerShell.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 AWSPowerShell. NetCore karena ukurannya yang besar. Untuk mulai berlari AWSPowerShell. 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 AWS Tools for PowerShell, Anda harus menjalankan Initialize- AWSDefaultConfiguration untuk menentukan kunci akses mana yang akan digunakan. AWS Untuk informasi selengkapnya tentang `Initialize-AWSDefaultConfiguration`, lihat Menggunakan Kredenensial AWS.

> **Note**
>
> Dalam rilis sebelumnya (sebelum 3.3.96.0) dari AWS Tools for PowerShell, cmdlet ini diberi nama. `Initialize-AWSDefaults`

Versioning

AWS merilis versi baru secara AWS Tools for PowerShell berkala untuk mendukung AWS layanan dan fitur baru. Untuk menentukan versi yang telah Anda instal, jalankan Get- AWSPowerShellVersion cmdlet. AWS Tools for PowerShell

```
PS > Get-AWSPowerShellVersion
```

```
Tools for PowerShell
Version 4.0.123.0
Copyright 2012-2019 Amazon.com, Inc. or its affiliates. All Rights Reserved.

Amazon Web Services SDK for .NET
Core Runtime Version 3.3.103.22
Copyright 2009-2015 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.

```
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 AWS Tools for PowerShell di Linux atau macOS

Secara berkala, saat versi terbaru dirilis, Anda harus memperbarui versi yang Anda jalankan secara lokal. AWS Tools for 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**
>
> Cmdlet `Update-AWSToolsModule` mengunduh semua modul dari `PSRepository` bernama `PSGallery` (https://www.powershellgallery.com/) dan menganggap repositori sebagai sumber terpercaya. 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 yang lebih baru dari. AWSPowerShell NetCore, hapus instalan 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 AWS Tools for PowerShell

Migrasi dari AWS Tools for PowerShell Versi 3.3 ke Versi 4

AWS Tools for PowerShell versi 4 adalah pembaruan kesesuaian balik ke AWS Tools for 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 rekomendasikan 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

The AWSPowerShell. NetCore dan AWSPowerShell paket “monolitik”. Ini artinya bahwa semua layanan AWS tersebut didukung dalam modul yang sama, yang membuatnya menjadi sangat besar, dan tumbuh lebih besar karena setiap layanan dan fitur AWS baru telah ditambahkan. Paket `AWS.Tools` baru dipecah menjadi modul yang lebih kecil yang memberikan fleksibilitas untuk mengunduh dan menginstal hanya yang Anda butuhkan untuk layanan AWS 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 masing-masing layanan AWS di kumpulan modul `AWS.Tools`, Anda dapat menggunakan cmdlet `Get-AWSService`.

```
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 API layanan AWS untuk Anda menggunakan AWS SDK for .NET. Kemudian AWS Tools for 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 obyek .NET yang dikembalikan oleh cmdlet.

Misalnya, cmdlet Get-S3Object memanggil operasi Amazon S3 SDK. ListObjects Operasi itu mengembalikan ListObjectsResponseobjek. Namun, secara default, `Get-S3Object` cmdlet hanya mengembalikan `S3Objects` elemen respons SDK kepada pengguna. PowerShell Pada contoh berikut, obyek adalah array dengan dua elemen.

```
PS > Get-S3Object -BucketName mybucket

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

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

Dalam AWS Tools for PowerShell versi 4, Anda dapat menyebutkan `-Select *` untuk mengembalikan obyek jawaban .NET lengkap oleh panggilan SDK API.

```
PS > Get-S3Object -BucketName mybucket -Select *
  IsTruncated    : False
  NextMarker     :
  S3Objects      : {file1.txt, file2.txt}
  Name           : mybucket
  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 mybucket -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 mybucket -Select S3Objects.Key |
>> Write-S3ObjectTagSet -Select ^Key -BucketName mybucket -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

AWS Tools for PowerShell versi sebelumnya memungkinkan Anda untuk menggunakan parameter `-MaxItems` untuk menyebutkan jumlah maksimum obyek yang dikembalikan dalam output akhir.

Perilaku ini dihapus dari `AWS.Tools`.

Perilaku ini tidak digunakan lagi di. AWSPowerShell NetCore dan AWSPowerShell, 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, kirim output ke cmdlet `Select-Object` dan sebutkan parameter `-First` *n*, di mana *n* adalah jumlah maksimum item yang disertakan dalam output akhir.

```
PS > Get-S3ObjectV2 -BucketName BUCKET_NAME -Select S3Objects.Key | select -first 2
file1.txt
file2.txt
```

Tidak semua layanan AWS mendukung `-MaxItems` dengan cara yang sama, sehingga akan menghilangkan ketidakkonsistenan dan hasil yang tidak diinginkan 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"', '}')
```

AWS Tools for PowerShell mengubah semua string ke `byte[]` menggunakan pengkodean UTF-8.

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='myBucket'; Key='file1.txt'; PartNumber=1 } | Get-
S3ObjectMetadata
```

> **Note**
>
> Beberapa properti mendukung ini di AWS Tools for PowerShell versi sebelumnya. Versi 4 membuatnya lebih konsisten dengan memungkinkannya untuk semua parameter.

Parameter Umum Statis

Untuk meningkatkan konsistensi dalam AWS Tools for PowerShell versi 4.0, semua parameter adalah statik.

Di AWS Tools for PowerShell versi sebelumnya, beberapa parameter umum seperti `AccessKey`, `SecretKey`, `ProfileName`, atau `Region`, adalah dinamis, sementara semua parameter lainnya 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 API Amazon EC2 akan gagal karena layanan AWS masih membutuhkan 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 lagi digunakan dalam AWS Tools for PowerShell rilis sebelumnya dan dihapus dari 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 AWSWilayah 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

Konfigurasikan 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 Tools for PowerShell, lihat Autentikasi dan akses di AWS SDK dan Tools Reference Guide.

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 kredensil 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 konfigurasikan Pusat Identitas IAM
2. Konfigurasikan Alat PowerShell untuk menggunakan IAM Identity Center.
3. Memulai sesi portal AWS akses

Aktifkan dan konfigurasikan 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 autentikasi IAM Identity Center di AWS SDK and Tools Reference Guide. Secara khusus, ikuti instruksi yang diperlukan di bawah Saya tidak memiliki akses yang ditetapkan melalui Pusat Identitas IAM.

Konfigurasikan 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, lihatGunakan 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 AWS Tools for PowerShell sesuai untuk sistem operasi dan lingkungan Anda, termasuk cmdlet umum. Untuk informasi tentang cara melakukan ini, lihat Menginstal AWS Tools for 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 AWS SDK 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 AWS SDK dan Panduan Referensi Alat.

   Alat untuk PowerShell menggunakan penyedia token SSO profil untuk memperoleh kredensil sebelum mengirim permintaan ke. AWS`sso_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 kredensil. 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 AWS SDK dan Panduan Referensi 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 AWS SDK dan Alat.

- Untuk mempelajari lebih lanjut tentang praktik terbaik, lihat Praktik terbaik keamanan di IAM di Panduan Pengguna IAM.
- Untuk membuat AWS kredensil jangka pendek, lihat Kredensial Keamanan Sementara di Panduan Pengguna IAM.
- Untuk mempelajari tentang penyedia kredensi lainnya, lihat Penyedia kredensi terstandarisasi di Panduan Referensi AWS SDK 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. Konfigurasikan 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

Konfigurasikan 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 mengonfigurasi Alat PowerShell untuk menggunakan Pusat Identitas IAM melalui AWS CLI ada di Langkah 2 dalam topik otentikasi Pusat Identitas IAM di AWS SDK 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 `config`File 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 AWS SDK dan Panduan Referensi Alat. Alat untuk PowerShell menggunakan penyedia token SSO profil untuk memperoleh kredensil sebelum mengirim permintaan ke. AWS`sso_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.

`config`File 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.SSOOIDC`

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 .SSOOIDC 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 kredensil. 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 AWS SDK dan Panduan Referensi 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 kredensil.
> 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 SSO, SSOOIDC

# 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.SSO
Import-Module AWS.Tools.SSOOIDC

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

# Now we can invoke cmdlets using the SSO 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 AWS SDK dan Panduan Referensi Alat.

Informasi tambahan

- Untuk opsi lebih lanjut tentang otentikasi Alat untuk PowerShell, seperti penggunaan profil dan variabel lingkungan, lihat bagian konfigurasi di AWS SDK dan Panduan Referensi 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 AWS SDK dan Alat.
- Untuk mempelajari lebih lanjut tentang praktik terbaik, lihat Praktik terbaik keamanan di IAM di Panduan Pengguna IAM.
- Untuk membuat AWS kredensil jangka pendek, lihat Kredenal Keamanan Sementara di Panduan Pengguna IAM.

- Untuk mempelajari tentang penyedia kredensi lainnya, lihat Penyedia kredensi terstandarisasi di Panduan Referensi AWS SDK dan Alat.

Tentukan AWS Wilayah

Ada dua cara untuk menentukan AWS Wilayah yang akan digunakan saat menjalankan AWS Tools for 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 AWS SDK dan Panduan Referensi Alat.

Konfigurasikan identitas federasi dengan AWS Tools for PowerShell

Untuk mengizinkan pengguna di organisasi Anda mengakses sumber daya AWS, Anda harus mengkonfigurasi metode autentikasi standar dan dapat diulang untuk tujuan keamanan, kemampuan audit, kepatuhan, dan kemampuan untuk mendukung pemisahan peran dan akun. Meskipun umum untuk menyediakan pengguna dengan kemampuan untuk mengakses API AWS, tanpa akses API federasi, Anda juga harus membuat pengguna (IAM) AWS Identity and Access Management, yang mengenyampingkan tujuan menggunakan federasi. Topik ini menjelaskan dukungan SAML (Security Assertion Markup Language) di AWS Tools for PowerShell yang memudahkan solusi akses federasi Anda.

Dukungan SAML di AWS Tools for PowerShell memungkinkan Anda memberikan pengguna Anda akses federasi ke layanan AWS. SAML adalah format standar terbuka berbasis XML untuk mentransmisikan autentikasi 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 SAML di AWS Tools for PowerShell kompatibel dengan SAML 2.0.

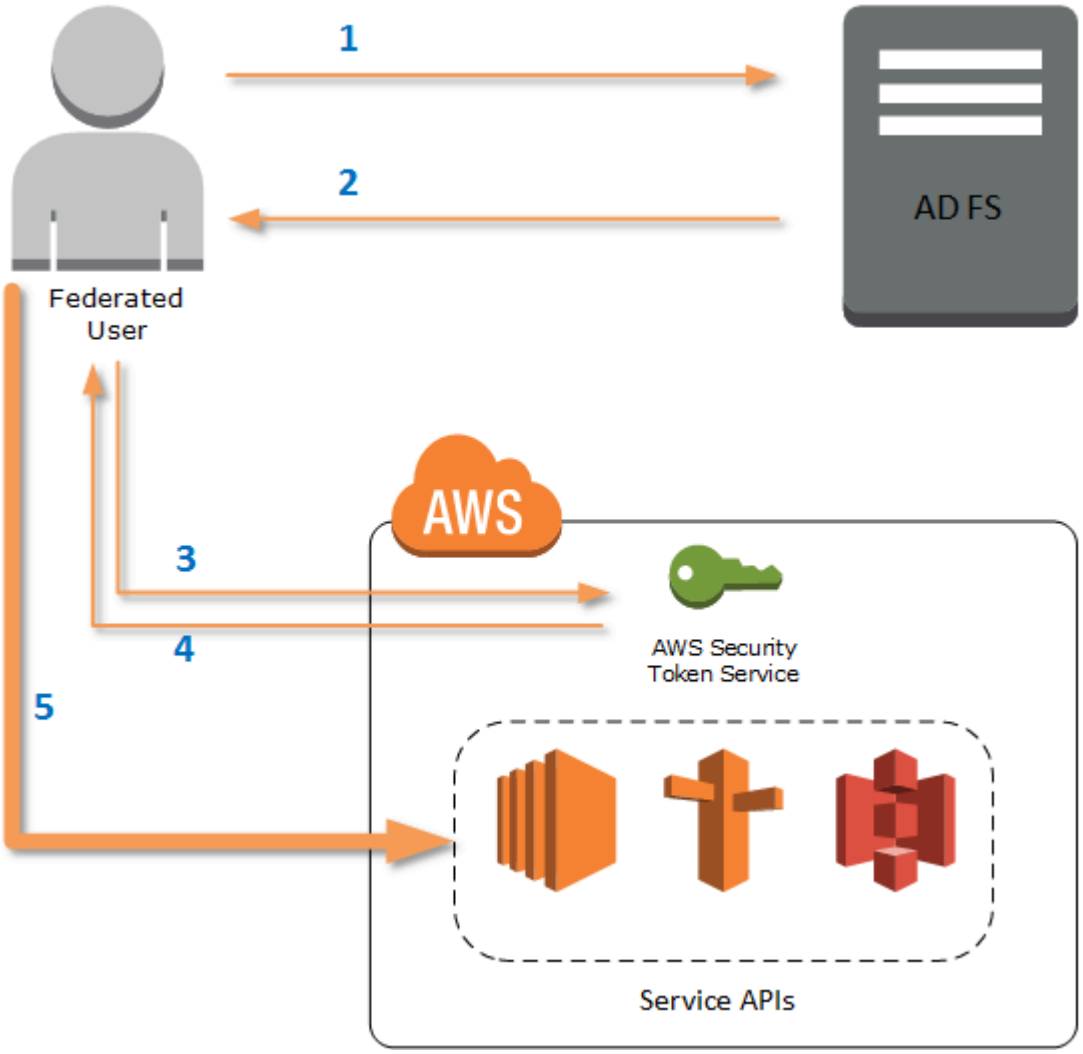
Prasyarat

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 melakukannya khususnya untuk Layanan Federasi Direktori Aktif, lihat Tentang Federasi SAML 2.0 di Panduan Pengguna IAM, dan posting blog, Mengaktifkan Federasi untuk AWS Menggunakan Direktori Aktif Windows, AD FS, dan SAML 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 AWS Tools for PowerShell diinstal pada stasiun kerja lokal Anda.

Bagaimana Pengguna Federasi Identitas-Mendapat Akses Federasi ke API Layanan AWS

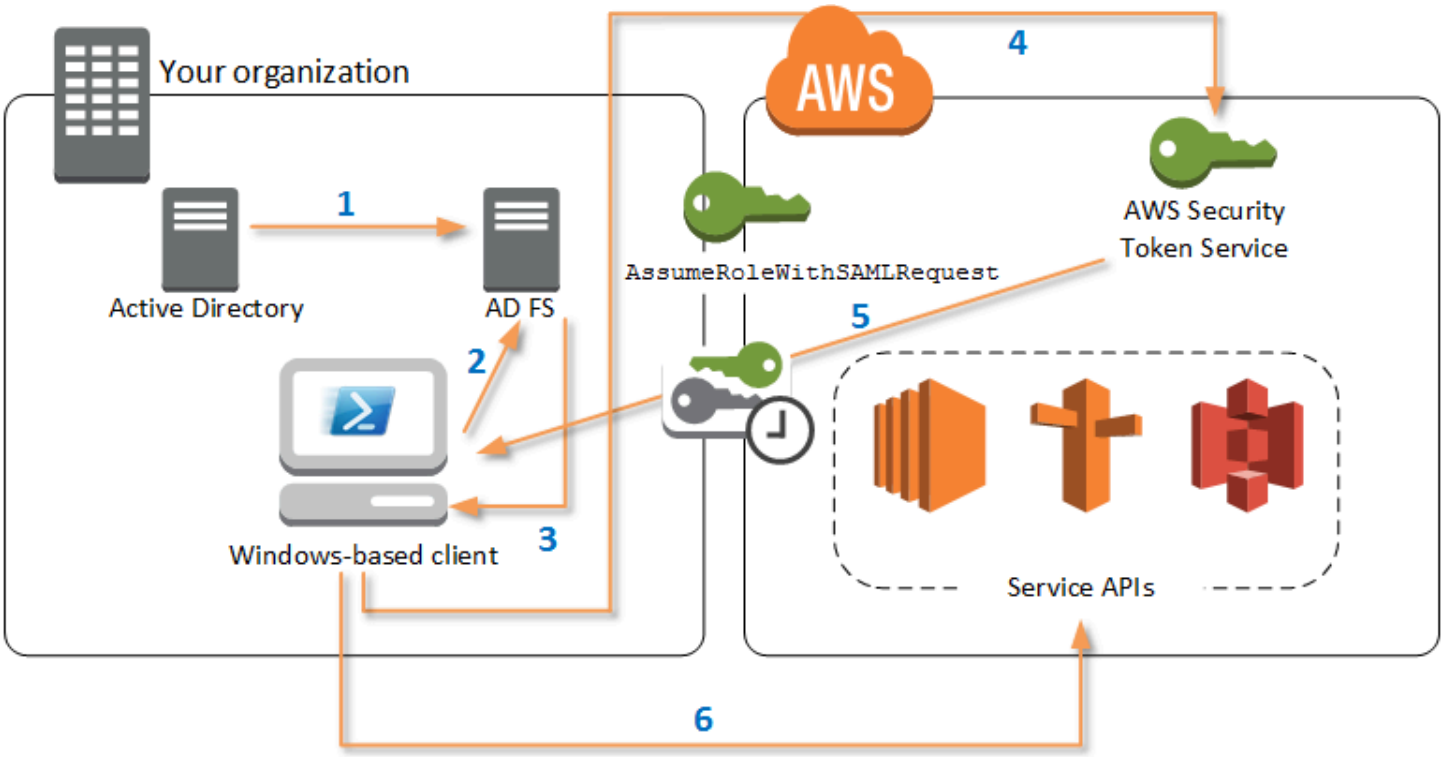
Proses berikut menjelaskan, pada tingkat tinggi, bagaimana pengguna Direktori Aktif (AD) digabungkan oleh AD FS untuk mendapatkan akses ke sumber daya AWS.



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 SAML ke (STS) AWS Security Token Service sebagai bagian dari permintaan federasi SAML.
4. STS mengembalikan jawaban SAML yang berisi kredensial sementara AWS untuk peran yang dapat dikerjakan pengguna.
5. Pengguna mengakses API layanan AWS dengan memasukkan kredensial sementara tersebut dalam permintaan yang dibuat oleh AWS Tools for PowerShell.

Cara Kerja Support SAML di AWS Tools for PowerShell

Bagian ini menjelaskan cara cmdlet AWS Tools for PowerShell mengaktifkan konfigurasi identitas berbasis SAML untuk pengguna.



1. AWS Tools for PowerShell mengotentikasi terhadap AD FS dengan menggunakan kredensial pengguna Windows saat ini, atau secara interaktif, ketika pengguna mencoba untuk menjalankan cmdlet yang memerlukan pemanggilan kredensial ke AWS.
2. AD FS mengotentikasi pengguna.
3. AD FS menghasilkan jawaban autentikasi SAML 2.0 yang mencakup pernyataan; tujuan pernyataan adalah untuk mengidentifikasi dan memberikan informasi tentang pengguna. AWS Tools for PowerShell mengekstrak daftar peran resmi pengguna dari pernyataan SAML.
4. AWS Tools for PowerShell meneruskan permintaan SAML, termasuk Amazon Resource Name (ARN) peran yang diminta, ke STS dengan membuat 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 API layanan AWS yang diizinkan diakses oleh peran pengguna. AWS Tools for PowerShell secara otomatis menerapkan

kredensial ini untuk setiap panggilan API AWS, dan memperbaharui mereka secara otomatis setelah masa berlakunya berakhir.

> **Note**
>
> Ketika kredensial berakhir, dan mandat baru diperlukan, AWS Tools for 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 tanpa tergabung domain, AWS Tools for PowerShell meminta pengguna untuk memasukkan kredensialnya sebelum dapat diautentikasi kembali.

Cara Menggunakan Cmdlet Konfigurasi PowerShell SAFL

AWS Tools for PowerShell mencakup dua cmdlet baru yang menyediakan dukungan SAML.

- `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 kredensial AWS, Anda memberikan nama yang mudah untuk profil peran. Anda dapat menggunakan nama mudah yang sama dengan cmdlet `Set-AWSCredential`, atau sebagai nilai parameter `-ProfileName` untuk setiap cmdlet yang membuka API layanan AWS.

Membuka sesi AWS Tools for PowerShell baru. Jika Anda menjalankan PowerShell 3.0 atau yang lebih baru, AWS Tools for 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, konfigurasikan 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 ID akun placeholder dan nama host AD FS dengan ID akun Anda sendiri 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 > $epName = 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 $epName
   ```

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 permntaan 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 kredensial AWS, Anda dapat menggunakan profil peran yang didefinisikan dalam file kredensial bersama AWS. Memberikan nama profil peran untuk `Set-AWSCredential` (atau sebagai nilai untuk parameter `ProfileName` mana pun dalam AWS Tools for PowerShell) untuk mendapatkan kredensial AWS sementara secara otomatis untuk peran yang dijelaskan di 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.

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

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

Contoh ini mengatur peran default untuk sesi AWS Tools for PowerShell 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 berisi semua buket Amazon S3 yang tersedia di akun AWS peran yang terkait dengan profil `SAMLDemoProfile`. Contoh tersebut menunjukkan bahwa meskipun Anda mungkin telah menggunakan profil lain sebelumnya di sesi AWS Tools for PowerShell Anda, Anda dapat mengubah profil dengan menentukan nilai yang berbeda untuk parameter `-ProfileName` 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                                          mybucket1
4/15/2015 12:46:50 AM                                         mybucket2
4/15/2015 6:15:53 AM                                          mybucket3
1/12/2015 11:20:16 PM                                         mybucket4
```

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 kedaluarsa setelah 1 jam (batas yang diberlakukan oleh STS), AWS Tools for 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, AWS Tools for PowerShell menampilkan prompt PowerShell kredenal 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 instans Amazon EC2 di Wilayah Asia Pacific (Sydney) yang terkait dengan akun yang digunakan oleh profil `ADFS-Production`. Ini adalah perintah yang berguna untuk mengembalikan semua instans Amazon EC2 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 cara daftar layanan yang didukung oleh AWS Tools for PowerShell, cara menampilkan kumpulan cmdlet yang disediakan oleh AWS Tools for PowerShell untuk mendukung layanan tersebut, dan cara menemukan nama cmdlet alternatif (juga disebut alias) untuk mengakses layanan tersebut.

Penemuan Cmdlet

Semua AWS operasi layanan (atau API) didokumentasikan dalam Panduan Referensi API untuk masing-masing layanan. Misalnya, lihat bagian Referensi API IAM. Ada, dalam banyak kasus, one-to-one korespondensi antara API AWS layanan dan AWS PowerShell cmdlet. Untuk mendapatkan nama cmdlet yang sesuai dengan nama API layanan AWS, jalankan cmdlet AWS `Get-AWSCmdletName` dengan parameter `-ApiOperation` dan nama API layanan AWS. Misalnya, untuk mendapatkan semua kemungkinan nama cmdlet yang didasarkan pada setiap API layanan `DescribeInstances` AWS 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 tahu nama-nama API dan layanan, Anda dapat menyertakan parameter `-Service` bersama dengan awalan kata benda cmdlet atau bagian dari nama layanan AWS. Sebagai contoh, awalan kata benda cmdlet untuk Amazon EC2 adalah `EC2`. Untuk mendapatkan nama cmdlet yang sesuai dengan API `DescribeInstances` dalam layanan Amazon EC2, 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 layanan AWS atau layanan AWS yang diinginkan, Anda dapat menggunakan parameter `-ApiOperation`, bersama dengan pola untuk mencocokkan, dan parameter `-MatchWithRegex`. 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-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
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 tahu nama layanan AWS tetapi tidak tahu API layanan AWS, masukkan parameter `-MatchWithRegex` dan parameter `-Service` untuk lingkup pencarian ke layanan tunggal. Misalnya, untuk mendapatkan semua nama cmdlet yang berisi `SecurityGroup` hanya di layanan Amazon EC2, 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 tahu nama perintah AWS Command Line Interface (AWS CLI), Anda dapat menggunakan parameter `-AwsCliCommand` dan nama perintah AWS CLI yang diinginkan untuk mendapatkan nama cmdlet yang didasarkan pada API yang sama. Misalnya, untuk mendapatkan semua nama cmdlet yang sesuai dengan panggilan perintah `authorize-security-group-ingress` AWS CLI di layanan Amazon EC2, 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
```

Cmdlet `Get-AWSCmdletName` hanya membutuhkan nama perintah AWS CLI untuk mengidentifikasi layanan dan perintah 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 bekerja dengan wilayah AWS, 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-CWEResourceTag                                 3.3.563.1
 AWSPowerShell.NetCore
Cmdlet          Disable-CWEEventSource                             3.3.563.1
 AWSPowerShell.NetCore
Cmdlet          Disable-CWERule                                    3.3.563.1
 AWSPowerShell.NetCore
Cmdlet          Enable-CWEEventSource                              3.3.563.1
 AWSPowerShell.NetCore
Cmdlet          Enable-CWERule                                     3.3.563.1
 AWSPowerShell.NetCore
Cmdlet          Get-CWEEventBus                                    3.3.563.1
 AWSPowerShell.NetCore
Cmdlet          Get-CWEEventBusList                                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-CWERule                                       3.3.563.1
 AWSPowerShell.NetCore
Cmdlet          Write-CWETarget                                     3.3.563.1
 AWSPowerShell.NetCore
```

Penamaan dan Alias Cmdlet

Cmdlet dalam AWS Tools for PowerShell untuk masing-masing layanan didasarkan pada metode yang disediakan oleh SDK AWS untuk layanan. Namun, karena konvensi penamaan wajib, nama cmdlet mungkin berbeda dari nama panggilan API atau metode yang menjadi dasarnya. PowerShell Misalnya, cmdlet `Get-EC2Instance` didasarkan pada metode `DescribeInstances`Amazon EC2.

Dalam beberapa kasus, nama cmdlet mungkin mirip dengan nama metode, tetapi keduanya mungkin sebenarnya melakukan fungsi yang berbeda. Misalnya, metode `GetObject`Amazon 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 AWS Tools for PowerShell, jalankan perintah cmdlet `Read-S3Object`:

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

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

> **Note**
>
> Bantuan cmdlet untuk cmdlet AWS menyediakan nama SDK API AWS yang menjadi dasar cmdlet.
> Untuk informasi selengkapnya tentang PowerShell kata kerja standar dan artinya, lihat Kata Kerja yang Disetujui untuk PowerShell Perintah.

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

> **Important**
>
> Cmdlet AWS tidak mendukung switch `-WhatIf`.

Alias

Penyiapan AWS Tools for PowerShell akan memasang 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 berada di direktori pemasangan AWS Tools for PowerShell. 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 alias AWS Tools for PowerShell, 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           Get-ASACommunications                           3.3.343.0
 AWSPowerShell
Alias           Get-ASAServices                                 3.3.343.0
 AWSPowerShell
Alias           Get-ASASeverityLevels                           3.3.343.0
 AWSPowerShell
Alias           Get-ASATrustedAdvisorCheckRefreshStatuses       3.3.343.0
 AWSPowerShell
Alias           Get-ASATrustedAdvisorChecks                     3.3.343.0
 AWSPowerShell
Alias           Get-ASATrustedAdvisorCheckSummaries             3.3.343.0
 AWSPowerShell
Alias           Get-ASLifecycleHooks                            3.3.343.0
 AWSPowerShell
Alias           Get-ASLifecycleHookTypes                        3.3.343.0
 AWSPowerShell
Alias           Get-AWSCredentials                              3.3.343.0
 AWSPowerShell
Alias           Get-CDApplications                              3.3.343.0
 AWSPowerShell
Alias           Get-CDDeployments                               3.3.343.0
 AWSPowerShell
Alias           Get-CFCloudFrontOriginAccessIdentities          3.3.343.0
 AWSPowerShell
Alias           Get-CFDistributions                             3.3.343.0
 AWSPowerShell
Alias           Get-CFGConfigRules                              3.3.343.0
 AWSPowerShell
Alias           Get-CFGConfigurationRecorders                   3.3.343.0
 AWSPowerShell
Alias           Get-CFGDeliveryChannels                         3.3.343.0
 AWSPowerShell
Alias           Get-CFInvalidations                             3.3.343.0
 AWSPowerShell
Alias           Get-CFNAccountLimits                            3.3.343.0
 AWSPowerShell
Alias           Get-CFNStackEvents                              3.3.343.0
 AWSPowerShell

...
```

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 dan $AWSHistory

Untuk panggilan layanan AWS yang mengembalikan pengumpulan, obyek yang dikumpulkan dihitung ke alur. Hasil obyek yang berisi bidang tambahan di luar pengumpulan dan yang tidak membagi bidang kontrol mendapatkan bidang ini ditambahkan sebagai properti Catatan untuk panggilan. Properti Catatan ini dicatat dalam variabel sesi `$AWSHistory`, jika Anda perlu untuk mengakses data ini. Variabel `$AWSHistory` dijelaskan di bagian berikut.

> **Note**
>
> Dalam versi Tools for Windows PowerShell sebelum v1.1, objek pengumpulan itu sendiri dipancarkan, yang memerlukan penggunaan foreach {$_.getenumerator ()} untuk melanjutkan pipelining.

Contoh

Contoh berikut mengembalikan daftar WIlayah AWS dan citra mesin Amazon EC2 (AMIS) Anda di setiap WIlayah.

```
PS > Get-AWSRegion | % { Echo $_.Name; Get-EC2Image -Owner self -Region $_ }
```

Contoh berikut memfilter stopword semua instans Amazon EC2 di wilayah default saat ini.

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

Karena pengumpulan menghitung alur, output dari cmdlet tertentu mungkin berupa `$null`, satu objek, atau kumpulan obyek. Jika berupa kumpulan obyek, Anda dapat menggunakan properti `.Count` untuk menentukan ukuran kumpulan tersebut. Namun, properti `.Count` tidak ada ketika hanya satu objek yang dipancarkan. Jika skrip Anda perlu menentukan, secara konsisten, berapa banyak objek yang dipancarkan, Anda dapat memeriksa properti `EmittedObjectsCount` dari nilai perintah terakhir di `$AWSHistory`.

$AWSHistory

Untuk mendukung pipelining yang lebih baik, output dari cmdlet AWS tidak dibentuk kembali untuk menyertakan respons layanan dan instans hasil sebagai properti Catatan pada objek pengumpulan yang dipancarkan. Sebaliknya, untuk panggilan yang memancarkan pengumpulan tunggal sebagai output, pengumpulan tersebut dihitung ke PowerShell alur. Ini berarti bahwa respon dan data hasil SDK AWS tidak dapat tersedia dalam alur, karena tidak ada objek pengumpulan 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 ke dan diterima dari panggilan layanan AWS yang mendasari yang dilakukan oleh cmdlet.

Dimulai dengan versi 1.1, data ini dan banyak lagi saat ini tersedia dalam variabel shell baru bernama `$AWSHistory`. Variabel ini menyimpan catatan invokasi cmdlet AWS dan respons layanan yang diterima untuk setiap invokasi. Secara opsional, riwayat ini dapat dikonfigurasi juga untuk mencatat permintaan layanan yang dibuat setiap cmdlet. Tambahan data yang berguna, seperti waktu eksekusi keseluruhan cmdlet, juga dapat diperoleh dari setiap entri. Demi alasan keamanan, permintaan dan tanggapan yang berisi data sensitif tidak direkam secara default. Namun, riwayat dapat dikonfigurasi untuk menimpa perilaku ini jika diperlukan. Untuk informasi selengkapnya, lihat`Set-AWSHistoryConfiguration` cmdlet yang ditunjukkan di bawah ini.

Setiap entri di daftar `$AWSHistory.Commands` adalah tipe `AWSCmdletHistory`. Tipe ini memiliki anggota yang berguna sebagai berikut:

CmdletName

Nama cmdlet.

CmdletStart

DateTime saat cmdlet dijalankan.

CmdletEnd

DateTime saat cmdlet menyelesaikan semua proses.

Permintaan

Jika perekaman permintaan diaktifkan, daftar permintaan layanan terakhir.

Tanggapan

Daftar tanggapan layanan terakhir yang diterima.

LastServiceResponse

Helper akan mengembalikan tangggapan layanan terkini.

LastServiceRequest

Helper akan mengembalikan permintaan layanan terkini, jika tersedia.

Perhatikan bahwa variabel `$AWSHistory` tidak dibuat hingga cmdlet AWS yang membuat panggilan layanan digunakan. Ini mengevaluasi ke $null sampai saat itu.

> **Note**
>
> Versi sebelumnya dari Tools for Windows PowerShell memancarkan data yang terkait dengan jawaban layanan sebagai`Note` properti pada objek yang dikembalikan. Ini sekarang ditemukan pada entri jawaban yang direkam untuk setiap invokasi dalam daftar.

Set-AWSHistoryConfiguration

Invokasi cmdlet dapat menahan nol atau lebih entri permintaan dan jawaban layanan. Untuk membatasi dampak memori, daftar `$AWSHistory` menyimpan catatan hanya lima eksekusi cmdlet terakhir secara default; dan untuk masing-masing, lima jawaban layanan terakhir (dan jika diaktifkan, lima permintaan layanan terakhir). Anda dapat mengubah batas default ini dengan menjalankan cmdlet `Set-AWSHistoryConfiguration`. Hal ini memungkinkan Anda untuk mengontrol ukuran daftar, dan apakah permintaan layanan juga dicatat atau tidak:

```
PS > Set-AWSHistoryConfiguration -MaxCmdletHistory <value> -MaxServiceCallHistory
 <value> -RecordServiceRequests -IncludeSensitiveData
```

Semua parameter bersifat opsional.

Parameter `MaxCmdletHistory` menetapkan jumlah maksimum cmdlet yang dapat dilacak setiap saat. Nilai 0 mematikan perekaman kegiatan cmdlet AWS. Parameter `MaxServiceCallHistory` menetapkan jumlah maksimum jawaban (dan/atau permintaan) layanan yang dilacak untuk setiap cmdlet. Parameter `RecordServiceRequests`, jika ditentukan, mengaktifkan pelacakan permintaan layanan untuk setiap cmdlet. `IncludeSensitiveData`Parameter, jika ditentukan, mengaktifkan pelacakan jawaban layanan dan permintaan (jika dilacak) yang berisi data sensitif untuk setiap cmdlet.

Jika dijalankan tanpa parameter, `Set-AWSHistoryConfiguration` hanya mematikan setiap rekaman permintaan sebelumnya, tidak merubah ukuran daftar saat ini.

Untuk menghapus semua entri dalam daftar riwayat saat ini, jalankan cmdlet `Clear-AWSHistory`.

Contoh **$AWSHistory**

Menghitung rincian cmdlet AWS yang sedang disimpan dalam daftar ke alur.

```
PS > $AWSHistory.Commands
```

Mengakses rincian yang cmdlet AWS terakhir yang dijalankan:

```
PS > $AWSHistory.LastCommand
```

Mengakses rincian jawaban layanan terakhir yang diterima oleh cmdlet AWS terakhir yang dijalankan. Jika cmdlet AWS membagi output, cmdlet tersebut dapat membuat beberapa panggilan layanan untuk mendapatkan semua data atau jumlah maksimum data (ditentukan oleh parameter pada cmdlet).

```
PS > $AWSHistory.LastServiceResponse
```

Mengakses rincian permintaan terakhir yang dibuat (sekali lagi, cmdlet dapat membuat lebih dari satu permintaan jika membaginya atas nama pengguna). Hasil $null kecuali pelacakan permintaan layanan diaktifkan.

```
PS > $AWSHistory.LastServiceRequest
```

Page-to-Completion Otomatis untuk Operasi yang Mengembalikan Banyak Pembagian

Untuk API layanan yang memaksakan obyek maksimum default mengembalikan jumlah untuk panggilan tertentu atau yang mendukung set hasil yang dapat dihitung, semua cmdlet "page-to-completion" secara default. Setiap cmdlet membuat panggilan sebanyak yang diperlukan atas nama Anda untuk mengembalikan data lengkap ke alur.

Pada contoh berikut, yang menggunakan `Get-S3Object`, variabel `$c` berisi `S3Object` instans untuk setiap kunci dalam `test` bucket, kemungkinan merupakan data lengkap yang sangat besar.

```
PS > $c = Get-S3Object -BucketName test
```

Jika Anda ingin mempertahankan kontrol jumlah data yang dikembalikan, Anda dapat menggunakan parameter pada masing-masing cmdlet (misalnya, `MaxKey` pada `Get-S3Object`) atau Anda dapat secara eksplisit menangani pembagian sendiri dengan menggunakan kombinasi parameter pembagian pada cmdlet, dan data yang ditempatkan di variabel `$AWSHistory` untuk mendapatkan data token berikutnya dari layanan. Contoh berikut menggunakan MaxKeys parameter untuk membatasi jumlah`S3Object` instans yang dikembalikan ke paling banyak 500 pertama yang ditemukan dalam bucket.

```
PS > $c = Get-S3Object -BucketName test -MaxKey 500
```

Untuk mengetahui apakah lebih banyak data tersedia tetapi tidak dikembalikan, gunakan entri variabel sesi `$AWSHistory` yang mencatat panggilan layanan yang dibuat oleh cmdlet.

Jika ekspresi berikut mengevaluasi ke $true, Anda dapat menemukan penanda `next` untuk rangkaian hasil berikutnya menggunakan `$AWSHistory.LastServiceResponse.NextMarker`:

```
$AWSHistory.LastServiceResponse -ne $null &&
 $AWSHistory.LastServiceResponse.IsTruncated
```

Untuk mengontrol pembagian secara manual dengan `Get-S3Object`, gunakan kombinasi dari parameter `MaxKey` dan `Marker` untuk cmdlet dan catatan `IsTruncated`/`NextMarker` pada jawaban yang direkam terakhir. Pada contoh berikut, variabel `$c` berisi hingga maksimum 500 instans `S3Object` untuk 500 objek berikutnya yang ditemukan dalam bucket setelah dimulainya penanda prefiks kunci tertentu.

```
PS > $c = Get-S3Object -BucketName test -MaxKey 500 -Marker
 $AWSHistory.LastServiceResponse.NextMarker
```

Resolusi kredensialdan profil

Urutan Pencarian Kredensial

Ketika Anda menjalankan perintah, AWS Tools for PowerShell mencari kredensial dengan 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 menyebutkan nama profil, perintah mencari profil tertentu di penyimpanan SDK AWS dan, jika tidak ada, maka akan mencari profil yang disebutkan dari file kredensial bersama AWS 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. Profil `default` di penyimpanan SDK AWS.

   b. Profil `default` di file kredensial bersama AWS.

   c. Profil `AWS PS Default` di penyimpanan SDK AWS.

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

   Untuk informasi lebih lanjut tentang cara menggunakan peran IAM untuk instans Amazon EC2, lihat AWS SDK for .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 aktifAWS, 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 diberikanKonfigurasikan 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 AWS SDK dan Alat dan Memulai di Panduan Pengguna. AWS IAM Identity Center

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 instans Amazon EC2 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 AWSlayanan, temukan dan pilih EC2 (misalnya), lalu pilih kasus penggunaan EC2 (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 hal di akun Anda.

5. Pilih Selanjutnya. 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 Create role (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 dalam Panduan Referensi AWSSDK dan Alat.
> Untuk praktik keamanan terbaik, gunakanAWS IAM Identity Center, seperti yang dijelaskan dalamKonfigurasikan otentikasi alat.

Peringatan penting dan panduan untuk kredensional

Peringatan untuk kredensional

- 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 kredensional Anda.
- Ketahuilah bahwa setiap kredensional yang disimpan dalam AWS `credentials` file bersama, disimpan dalam teks biasa.

Panduan tambahan untuk mengelola kredensional dengan aman

Untuk diskusi umum tentang cara mengelola AWS kredensional dengan aman, lihat kredenal AWSkeamanan dalam praktik dan kasus penggunaan terbaik Keamanan Referensi Umum AWSdan 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 instans Amazon EC2.

Topik

Menggunakan Kredenensial AWS

Setiap perintah AWS Tools for PowerShell harus menyertakan satu set kredensial AWS, yang digunakan untuk menandatangani permintaan layanan web yang sesuai secara kriptografi. 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 kredensi jangka pendek dan jangka panjang, lihat Cara lain untuk mengautentikasi dalam Panduan Referensi AWSSDK dan Alat.
> Untuk praktik keamanan terbaik, gunakanAWS IAM Identity Center, seperti yang dijelaskan dalamKonfigurasikan otentikasi alat.

Sebagai praktik terbaik, agar kredensial Anda tidak terlihat, jangan menempatkan kredensial literal dalam perintah. Sebaiknya, 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 AWS Tools for PowerShell terkait. Untuk diskusi umum tentang cara mengelola AWS kredensional dengan aman, lihat Praktik Terbaik untuk Mengelola Kunci AWS Akses di. Referensi Umum Amazon Web Services

> **Note**
>
> Anda memerlukan akun AWS untuk mendapatkan kredensial dan menggunakan AWS Tools for PowerShell. Untuk membuat AWS akun, lihat Memulai: Apakah Anda AWS pengguna pertama kali? dalam Panduan AWS Account Management Referensi.

Topik

Lokasi Penyimpanan Kredensial

AWS Tools for PowerShell dapat menggunakan salah satu dari dua penyimpanan kredensial:

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

  AWS SDK for .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`

  SDK AWS dan AWS Command Line Interface dapat juga menggunakan file kredensial. Jika Anda menjalankan skrip di luar konteks pengguna AWS, pastikan bahwa 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 merujuk set kredensial yang berbeda dengan AWS Tools for PowerShell. Anda dapat menggunakan cmdlet AWS Tools for PowerShell untuk mengelola profil Anda di penyimpanan SDK AWS. Anda juga dapat mengelola profil di penyimpanan SDK AWS dengan menggunakan Toolkit for Visual Studio atau secara pemrograman dengan menggunakan AWS SDK for .NET. Untuk petunjuk tentang cara mengelola profil di file kredensial, lihat Praktik Terbaik untuk Mengelola AWS Access Key.

Menambahkan Profil Baru

Untuk menambahkan profil baru ke penyimpanan SDK AWS, 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 SDK AWS harus dikelola 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 disarankan adalah dengan mengidentifikasi profil bukannya menggabungkan kredensial literal ke baris perintah Anda. AWS Tools for PowerShell menempatkan profil menggunakan urutan pencarian yang dijelaskan di Urutan Pencarian Kredensial.

Pada Windows, kredensial AWS yang disimpan dalam penyimpanan SDK AWS dienkripsi dengan identitas pengguna Windows untuk 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 parameter `-StoredCredentials` di rilis AWS Tools for PowerShell sebelumnya. Untuk kompatibilitas balik, `-StoredCredentials` masih didukung.

Profil Default (Disarankan)

Semua SDK dan alat pengelolaan AWS 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 menetapkannya sebagai profil default.

Meskipun menjalankan `Initialize-AWSDefaultConfiguration` memungkinkan Anda menentukan profil default untuk setiap PowerShell sesi, cmdlet memuat kredensi 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 instans Amazon EC2 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 kredensial yang dihasilkan dari menjalankan `Initialize-AWSDefaultConfiguration` pada instans EC2 tidak secara langsung menyimpan kredensial, tetapi sebaliknya menunjuk ke metadata instans (yang menyediakan kredensial sementara yang secara otomatis berputar). 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**
>
> Kredensial default termasuk dalam penyimpanan SDK AWS di bawah nama profil `default`. Perintah tersebut akan menimpa profil yang ada dengan nama itu.

Jika instans EC2 Anda diluncurkan dengan profil instans, PowerShell secara otomatis mendapatkan AWS kredensi dan informasi Wilayah dari profil instans. Anda tidak perlu menjalankan `Initialize-AWSDefaultConfiguration`. Menjalankan `Initialize-AWSDefaultConfiguration` cmdlet pada instans EC2 yang diluncurkan dengan profil instans 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 kredensial bersama AWS 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 skrip AWS Tools for PowerShell Anda berjalan dengan kredensial akun yang benar, Anda harus menambahkan parameter `-ProfileLocation` 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, AWS Tools for PowerShell mencari kredensial dengan 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 menyebutkan nama profil, perintah mencari profil tertentu di penyimpanan SDK AWS dan, jika tidak ada, maka akan mencari profil yang disebutkan dari file kredensial bersama AWS 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. Profil `default` di penyimpanan SDK AWS.
   b. Profil `default` di file kredensial bersama AWS.
   c. Profil `AWS PS Default` di penyimpanan SDK AWS.
6. Jika perintah berjalan pada instans Amazon EC2 yang dikonfigurasi untuk menggunakan IAM role, kredensial sementara instans EC2 yang diakses dari profil instans.

   Untuk informasi lebih lanjut tentang cara menggunakan peran IAM untuk instans Amazon EC2, lihat AWS SDK for .NET.

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

Penanganan Kredensial di AWS Tools for PowerShell Core

Cmdlet di AWS Tools for PowerShell Core menerima akses dan secret key atau nama-nama profil kredensial AWS ketika berjalan, begitu juga dengan AWS Tools for Windows PowerShell. Ketika berjalan di Windows, kedua modul memiliki akses ke file penyimpanan kredensial AWS SDK

for .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 dituju AWS Tools for PowerShell untuk mencari profil kredensial, dan juga file di mana AWS Tools for PowerShell menyimpan profil kredensialnya. Untuk informasi lebih lanjut tentang file penyimpanan kredensial AWS SDK for .NET, lihat Mengkonfigurasi Kredensial AWS. PowerShellModul Tools for Windows saat ini tidak mendukung kredensi penulisan ke file atau lokasi lain.

Kedua modul dapat membaca profil dari file kredensial berbagi AWS yang digunakan oleh SDK AWS dan 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.

Penyimpanan kredensial SDK memegang kredensial Anda dalam bentuk terenkripsi dengan menggunakan API kriptografi Windows . API ini tidak tersedia pada platform lain, jadi modul AWS Tools for PowerShell Core menggunakan file kredensial bersama AWS secara eksklusif, dan mendukung penulisan profil kredensial baru untuk file kredensial bersama.

Contoh skrip berikut yang menggunakan **`Set-AWSCredential`** cmdlet menunjukkan opsi untuk menangani profil kredensi di Windows dengan atau. AWSPowerShellAWSPowerShell 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-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 AWS Tools for PowerShell

Alat untuk Windows PowerShell mendukung penggunaan file kredensial AWS bersama, mirip dengan SDK AWS CLI dan lainnyaAWS. 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 dalam Panduan Referensi AWSSDK dan Alat.
> Untuk praktik keamanan terbaik, gunakanAWS IAM Identity Center, seperti yang dijelaskan dalamKonfigurasikan otentikasi alat.

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

Menggunakan IAM role dengan AWS Tools for PowerShell

File kredensial bersama AWS memungkinkan jenis akses tambahan. Misalnya, Anda dapat mengakses sumber daya AWS Anda dengan menggunakan IAM role bukan kredensial jangka panjang pengguna IAM. Untuk melakukannya, Anda harus memiliki profil standar yang memiliki izin untuk melaksanakan peran tersebut. Saat Anda memberi tahu AWS Tools for PowerShell untuk menggunakan profil yang menentukan peran, AWS Tools for PowerShell mencari profil yang diidentifikasi oleh parameter `SourceProfile`. 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
ExternalId	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 ExternalId sebagai parameter saat mengasumsikan peran yang ditugaska n. Untuk informasi selengkapnya, lihat Cara

Nama Parameter	Deskripsi
	Menggunakan ID Eksternal Saat Memberikan Akses ke Sumber Daya AWS Anda ke Pihak Ketiga dalam Panduan Pengguna IAM.
MfaSerial	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.
RoleArn	ARN peran untuk melaksanakan kredensial peran. Untuk informasi selengkapnya tentang pembuatan dan penggunaan peran, lihat IAM role dalam Panduan Pengguna IAM.
SourceProfile	Nama profil sumber yang akan digunakan dengan melaksanakan kredensial peran. Kredensial yang ditemukan di profil ini digunakan untuk melaksanakan peran yang ditentukan oleh parameter `RoleArn`.

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 didefinisikan dalam contoh sebelumnya untuk mengakses cmdlet Get-S3Bucket. AWS Tools for PowerShell mencari kredensialnya di `my_source_profile`, menggunakan kredensialnya untuk memanggil `AssumeRole` atas nama pengguna, dan kemudian menggunakan kredensial peran sementara untuk memanggil `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

Dasar
 	
Ini adalah kredensial jangka panjang untuk pengguna IAM
	`-AccessKey`
	
	`-SecretKey`

Sesi:
 	
Ini adalah kredensial jangka pendek untuk IAM role yang Anda ambil secara manual, seperti dengan langsung memanggil cmdlet Use-STSRole.
	`-AccessKey`
	
	`-SecretKey`
	
	`-SessionToken`

Tipe kredensial	Parameter yang harus Anda gunakan

Peran:
 	
Ini adalah kredensial jangka pendek untuk IAM role yang diambil oleh AWS Tools for PowerShell untuk Anda.
	`-SourceProfile`
	
	`-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 bersama AWS di *`(user's home directory)`* `\.aws\credentials` .
Jalur ke file dalam format file kredensial bersama AWS	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 AWSCredentialProfile cmdlet Remove- baru. Clear-AWSCredential tidak digunakan lagi, tetapi masih tersedia untuk kompatibilitas mundur.

Catatan Penting

Hanya Inisialisasi- AWSDefaultConfiguration, 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`.

Bekerja dengan AWS layanan di AWS Tools for PowerShell

Bagian ini memberikan contoh penggunaan AWS Tools for PowerShell untuk mengakses layanan AWS. Contoh ini membantu menunjukkan cara menggunakan cmdlet untuk melakukan tugas AWS aktual. Contoh-contoh ini bergantung pada cmdlet yang disediakan oleh Tools for PowerShell . Untuk melihat cmdlet apa yang tersedia, lihat Referensi AWS Tools for PowerShellCmdlet.

PowerShell Pengkodean Penggabungan File

Beberapa cmdlet di AWS Tools for PowerShell mengedit file yang ada atau catatan yang Anda miliki di AWS. Contohnya adalah`Edit-R53ResourceRecordSet`, yang memanggil ChangeResourceRecordSetsAPI 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 AWS Tools for PowerShell lebih berguna di PowerShell lingkungan asli, objek yang dikembalikan oleh AWS Tools for 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. `Buckets`Koleksi 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 AWS Tools for 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 instans Amazon EC2 termasuk cara:

- Mengambil daftar Amazon Machine Image (AM).
- Membuat pasangan kunci untuk otentikasi SSH.
- Membuat dan mengkonfigurasi grup keamanan Amazon EC2.
- 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 AWS Tools for 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 Tools for Windows PowerShell
- Amazon EC2 dan Tools for Windows PowerShell
- AWS Lambda dan AWS Tools for PowerShell
- Amazon SQS, Amazon SNS dan Tools for Windows PowerShell
- CloudWatch dari AWS Tools for Windows PowerShell

- Menggunakan ClientConfig parameter dalam cmdlet

Amazon S3 dan Tools for Windows PowerShell

Di bagian ini, kami membuat situs web statisAWS 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 PowerShell sesi Anda. 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 Tools for Windows untuk melakukan tugas ini. Untuk mengubah nama objek di S3, kami sarankan Anda menyalin objek ke obyek dengan nama baru, dengan menjalankan cmdlet Copy-S3Object, dan kemudian menghapus objek asli dengan menjalankan cmdlet Remove-S3Object.

Lihat juga

- Bekerja dengan AWS layanan di AWS Tools for 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 AWS Tools for PowerShell
- Masukan Bucket (Referensi Layanan Amazon S3)
- AWSKawasan PowerShell 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      : T7hlDOeLqA5Q2XfTe8j2q3SLoP3/5XwhUU3RyJBGHU/LnC+CIWLeGgP0MY24xAlI
ResponseStream :
Headers        : {x-amz-id-2, x-amz-request-id, Content-Length, Date...}
Metadata       : {}
ResponseXml    :
```

Lihat Juga

- Bekerja dengan AWS layanan di AWS Tools for PowerShell
- Masukan Situs Web Bucket (Referens API Amazon S3)
- Masukan ACL Bucket (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
```

Pilihan Canned ACL

Nilai-nilai untuk menentukan canned ACL dengan Tools for Windows PowerShell adalah sama dengan yang digunakan oleh AWS SDK for .NET. Perhatikan, bagaimanapun, ini berbeda dari nilai-nilai yang digunakan oleh tindakan`Put Object` Amazon S3. Tools for Windows PowerShell mendukung canned ACL berikut:

- 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, cmdlet `Write-S3Object` 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 menjelajahinya dengan menggunakan peramban. URL untuk situs web statis yang ditempatkan di Amazon S3 mengikuti format standar.

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

Sebagai contoh:

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

Lihat Juga

- Bekerja dengan AWS layanan di AWS Tools for PowerShell
- Masukan Obyek (Referensi API Amazon S3)
- Canned ACL (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    :
```

Dalam versi 1.1 dan yang lebih baru dari AWS Tools for PowerShell, Anda dapat menambahkan parameter `-DeleteBucketContent` ke `Remove-S3Bucket`, yang terlebih dulu menghapus semua objek dan versi objek dalam bucket tertentu sebelum mencoba untuk menghapus bucket itu sendiri. Bergantung pada jumlah objek atau versi objek dalam bucket, operasi ini dapat memerlukan waktu lama. Dalam versi Tools for Windows PowerShell yang lebih lama dari versi 1.1, bucket harus dalam keadaan kosong terlebih dulu sebelum `Remove-S3Bucket` bisa menghapusnya.

> **Note**
>
> Kecuali Anda menambahkan parameter `-Force`, AWS Tools for PowerShell meminta konfirmasi Anda sebelum cmdlet berjalan.

Lihat Juga

- Bekerja dengan AWS layanan di AWS Tools for PowerShell
- Hapus Obyek (Referensi API Amazon S3)
- Hapus Obyek (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 mybucket -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 mybucket -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 mybucket -key myobject.txt -content $x
```

Amazon EC2 dan Tools for Windows PowerShell

Anda dapat melakukan tugas-tugas umum yang berhubungan dengan Amazon EC2 menggunakan AWS Tools for PowerShell.

Contoh perintah yang ditampilkan di sini mengasumsikan bahwa Anda telah menetapkan kredensial default dan wilayah default untuk sesi PowerShell Anda. Oleh karena itu, kami tidak menyertakan kredensial atau wilayah saat kami membuka cmdlet. Untuk informasi selengkapnya, lihat Memulai dengan AWS Tools for Windows PowerShell.

Topik

- Membuat Pasangan Kunci
- Membuat Grup Keamanan Menggunakan Windows PowerShell
- Temukan Amazon Machine Image Menggunakan Windows PowerShell
- Luncurkan Instans Amazon EC2 Menggunakan Windows PowerShell

Membuat Pasangan Kunci

Contoh `New-EC2KeyPair` berikut ini menciptakan pasangan kunci dan toko dalam `$myPSKeyPair` variabel PowerShell

```
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----
                   MIIEogIBAAKCAQEAkK+ANYUS9c7niNjYfaCn6KYj/D0I6djnFoQE...
                   Mz6btoxPcE7EMeH1wySUp8nouAS9xbl9l7+VkD74bN9KmNcPa/Mu...
                   Zyn4vVe0Q5il/MpkrRogHqOB0rigeTeV5Yc3lvO0RFFPu0Kz4kcm...
                   w3Jg8dKsWn0plOpX7V3sRC02KgJIbejQUvBFGi5OQK9bm4tXBIeC...
                   daxKIAQMtDUdmBDrhR1/YMv8itFe5DiLLbq7Ga+FDcS85NstBa3h...
                   iuskGkcvgWkcFQkLmRHRoDpPb+OdFsZtjHZDpMVFmA9tT8EdbkEF...
                   3SrNeqZPsxJJIxOodb3CxLJpg75JU5kyWnb0+sDNVHoJiZCULCr0...
                   GGlLfEgB95KjGIk7zEv2Q7K6s+DHclrDeMZWa7KFNRZuCuX7jssC...
                   xO98abxMr3o3TNU6p1ZYRJEQ0oJr0W+kc+/8SWb8NIwfLtwhmJEy...
                   1BX9X8WFX/A8VLHrT1elrKmLkNECgYEAwltkV1pOJAFhz9p7ZFEv...
                   vvVsPaF0Ev9bk9pqhx269PB5Ox2KokwCagDMMaYvasWobuLmNu/1...
                   lmwRx7KTeQ7W1J3OLgxHA1QNMkip9c4Tb3q9vVc3t/fPf8vwfJ8C...
                   63g6N6rk2FkHZX1E62BgbewUd3eZOS05Ip4VUdvtGcuc8/qa+e5C...
                   KXgyt9nl64pMv+VaXfXkZhdLAdY0Khc9TGB9++VMSG5TrD15YJId...
                   gYALEI7m1jJKpHWAEs0hiemw5VmKyIZpzGstSJsFStERlAjiETDH...
                   YAtnI4J8dRyP9I7BOVOn3wNfIjk85gi1/0Oc+j8S65giLAfndWGR...
                   9R9wIkm5BMUcSRRcDy0yuwKBgEbkOnGGSD0ah4HkvrUkepIbUDTD...
                   AnEBM1cXI5UT7BfKInpUihZi59QhgdK/hkOSmWhlZGWikJ5VizBf...
                   drkBr/vTKVRMTi3lVFB7KkIV1xJxC5E/BZ+YdZEpWoCZAoGAC/Cd...
                   TTld5N6opgOXAcQJwzqoGa9ZMwc5Q9f4bfRc67emkw0ZAAwSsvWR...
                   x3O2duuy7/smTwWwskEWRK5IrUxoMv/VVYaqdzcOajwieNrblr7c...
                   -----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 privat yang kembali 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
```

(Alat `openssl` tidak disertakan dengan AWS Tools for PowerShell atau AWS SDK for .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, menggunakan cmdlet `Remove-EC2KeyPair`. 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 dalam sesi PowerShell saat ini dan masih berisi informasi pasangan kunci. File `myPSKeyPair.pem` juga ada. Namun, kunci privat tidak berlaku lagi karena kunci publik untuk pasangan kunci tidak lagi disimpan di AWS.

Membuat Grup Keamanan Menggunakan Windows PowerShell

Anda dapat menggunakan AWS Tools for PowerShell untuk membuat dan mengkonfigurasi grup keamanan. Ketika Anda membuat grup keamanan, Anda menentukan apakah itu untuk EC2-Classic atau EC2-VPC. 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

Anda memerlukan alamat IP publik komputer Anda, dalam notasi CIDR. Anda bisa mendapatkan alamat IP publik dari komputer lokal menggunakan layanan. Misalnya, kami 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-Classic

> **Warning**
>
> Kami pensiun EC2-Classic pada 15 Agustus 2022. Kami menyarankan Anda memindahkan EC2-Classic ke VPC. Untuk informasi selengkapnya, lihatMemindahkan dari EC2-Classic ke

VPCdi dalamPanduan Pengguna Amazon EC2 untuk Instans LinuxatauPanduan Pengguna Amazon EC2 untuk Instans Windows. Lihat juga posting blogJaringan EC2-Classic Pensiun - Inilah Cara Mempersiapkan.

Contoh berikut menggunakan cmdlet `New-EC2SecurityGroup` untuk menciptakan grup keamanan untuk `EC2-Classic`.

```
PS > New-EC2SecurityGroup -GroupName myPSSecurityGroup -GroupDescription "EC2-Classic
 from PowerShell"

sg-0a346530123456789
```

Untuk melihat konfigurasi awal grup keamanan, gunakan cmdlet `Get-EC2SecurityGroup`.

```
PS > Get-EC2SecurityGroup -GroupNames myPSSecurityGroup

Description         : EC2-Classic from PowerShell
GroupId             : sg-0a346530123456789
GroupName           : myPSSecurityGroup
IpPermissions       : {}
IpPermissionsEgress : {Amazon.EC2.Model.IpPermission}
OwnerId             : 123456789012
Tags                : {}
VpcId               : vpc-9668ddef
```

Untuk mengkonfigurasi grup keamanan untuk mengizinkan lalu lintas masuk pada TCP port 22 (SSH) dan TCP port 3389, gunakan cmdlet `Grant-EC2SecurityGroupIngress`. Sebagai contoh, contoh skrip berikut menunjukkan bagaimana Anda dapat mengaktifkan lalu lintas SSH dari alamat IP tunggal, `203.0.113.25/32`.

```
$cidrBlocks = New-Object 'collections.generic.list[string]'
$cidrBlocks.add("203.0.113.25/32")
$ipPermissions = New-Object Amazon.EC2.Model.IpPermission
$ipPermissions.IpProtocol = "tcp"
$ipPermissions.FromPort = 22
$ipPermissions.ToPort = 22
ipPermissions.IpRanges = $cidrBlocks
Grant-EC2SecurityGroupIngress -GroupName myPSSecurityGroup -IpPermissions
 $ipPermissions
```

Untuk memverifikasi grup keamanan telah diperbarui, jalankan cmdlet `Get-EC2SecurityGroup` lagi. Perhatikan bahwa Anda tidak dapat menentukan aturan keluar untuk `EC2-Classic`.

```
PS > Get-EC2SecurityGroup -GroupNames myPSSecurityGroup

OwnerId             : 123456789012
GroupName           : myPSSecurityGroup
GroupId             : sg-0a346530123456789
Description         : EC2-Classic from PowerShell
IpPermissions       : {Amazon.EC2.Model.IpPermission}
IpPermissionsEgress : {}
VpcId               :
Tags                : {}
```

Untuk melihat aturan grup keamanan, gunakan properti `IpPermissions`.

```
PS > (Get-EC2SecurityGroup -GroupNames myPSSecurityGroup).IpPermissions

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

Membuat Grup Keamanan untuk EC2-VPC

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 merujuk grup keamanan untuk EC2-VPC dengan 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 Amazon Machine Image Menggunakan Windows PowerShell

Ketika Anda meluncurkan sebuah instans Amazon EC2, Anda menentukan Amazon Machine Image (AMI) untuk berfungsi sebagai template untuk instans tersebut. Namun, ID untuk AWS Windows AMIS sering berubah karena AWS menyediakan AMI baru dengan pembaruan terbaru dan peningkatan keamanan. Anda dapat menggunakan Get-EC2Image dan Get-EC2Image untuk menemukan Windows AMI saat ini dan mendapatkan ID mereka.

Topik

- Get-EC2Image()
- Get-EC2ImageByName

Get-EC2Image()

cmdlet `Get-EC2Image` mengambil daftar AMI yang dapat Anda gunakan.

Gunakan parameter `-Owner` dengan nilai array `amazon, self` sehingga `Get-EC2Image` mengambil hanya AMI yang milik Amazon atau 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 menampilkan hanya Windows AMI.

```
$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 AMI yang dikembalikan oleh cmdlet; output sebenarnya dari perintah sebelumnya memberikan informasi bagi banyak AMI.

```
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

Cmdlet `Get-EC2ImageByName` memungkinkan Anda untuk mem-filter daftar AWS Windows AMI 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, xvdcb, xvdcc…}
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 Instans Amazon EC2 Menggunakan Windows PowerShell

Untuk meluncurkan instans Amazon EC2, Anda memerlukan pasangan kunci 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 Amazon Machine Image 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 EC2-Classic

> **Warning**
>
> Kami pensiun EC2-Classic pada 15 Agustus 2022. Kami menyarankan Anda memindahkan EC2-Classic ke VPC. Untuk informasi selengkapnya, lihatMemindahkan dari EC2-Classic ke VPCdi dalamPanduan Pengguna Amazon EC2 untuk Instans LinuxatauPanduan Pengguna Amazon EC2 untuk Mesin Windows. Lihat juga posting blogJaringan EC2-Classic Pensiun - Inilah Cara Mempersiapkan.

Perintah berikut membuat dan meluncurkan satu instans `t1.micro`.

```
PS > New-EC2Instance -ImageId ami-c49c0dac `
    -MinCount 1 `
    -MaxCount 1 `
    -KeyName myPSKeyPair `
    -SecurityGroups myPSSecurityGroup `
    -InstanceType t1.micro

ReservationId   : r-b70a0ef1
OwnerId         : 123456789012
RequesterId     :
Groups          : {myPSSecurityGroup}
GroupName       : {myPSSecurityGroup}
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 mem-filter 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-5caa4371")
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          : t1.micro
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              :
Tags                  : {}
VirtualizationType    : hvm
VpcId                 :
```

Meluncurkan Instans di VPC

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 mem-filter 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 AWS Tools for PowerShell

Dengan menggunakan AWSLambdaPSCoremodul, Anda dapat mengembangkanAWS Lambda fungsi-fungsi diPowerShell Core 6.0 menggunakan waktu aktif .NET Core 2.1. PowerShellpengembang dapat mengelolaAWS sumber daya dan menulis skrip otomatisasi diPowerShell lingkungan dengan menggunakan Lambda. PowerShelldukungan di Lambda memungkinkan Anda menjalankanPowerShell skrip atau fungsi dalam menanggapi setiap peristiwa Lambda, seperti acara Amazon S3, seperti acara Amazon S3 atau acaraCloudWatch terjadwal Amazon S3. AWSLambdaPSCoreModul adalahAWS modul terpisah untukPowerShell; ini bukan bagian dariAWS Tools for PowerShell, juga tidak menginstalAWSLambdaPSCore modul menginstalAWS Tools for PowerShell.

Setelah Anda menginstalAWSLambdaPSCore modul, Anda dapat menggunakanPowerShell cmdlet yang tersedia—atau mengembangkannya fungsi tanpa server untuk Anda sendiri. PowerShellModulAWS Lambda Tools for mencakup template proyek untuk aplikasi tanpa serverPowerShell berbasis, dan alat untuk mempublikasikan proyek keAWS.

AWSLambdaPSCoreDukungan 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 menggunakanAWSLambdaPSCore modul. Untuk detail selengkapnya tentang langkah-langkah ini, lihat Menyiapkan LingkunganPowerShell Pengembangan dalam PanduanAWS Lambda Pengembang.

- Instal versi yang benar dariPowerShell — Dukungan Lambda untukPowerShell didasarkan pada rilisPowerShell Core 6.0 lintas platform. Anda dapat mengembangkan fungsiPowerShell Lambda di Windows, Linux, atau Mac. Jika Anda belumPowerShell menginstal setidaknya rilis ini, petunjuk tersedia di situs webPowerShell dokumentasi Microsoft.
- Instal .NET Core 2.1 SDK — KarenaPowerShell Core dibangun di atas .NET Core, dukungan Lambda untukPowerShell menggunakan waktu aktif .NET Core 2.1 Lambda yang sama baik untuk fungsi .NET Core maupunPowerShell Lambda. Lambda yangPowerShell menerbitkan cmdlet menggunakan .NET Core 2.1 SDK untuk membuat paket deployment Lambda. .NET Core 2.1 SDK tersedia dari Pusat Unduhan Microsoft. Pastikan untuk menginstal SDK, bukan Runtime.

PasangAWSLambdaPSCore Modul

Setelah menyelesaikan prasyarat, Anda siap untuk menginstalAWSLambdaPSCore modul. Jalankan perintah berikut dalam sesiPowerShell Core.

```
PS> Install-Module AWSLambdaPSCore -Scope CurrentUser
```

Anda siap untuk mulai mengembangkan fungsi Lambda diPowerShell. Untuk informasi selengkapnya tentang cara memulai, lihat Model Pemrograman untuk Mengotorisasi Fungsi LambdaPowerShell dalam dalam PanduanAWS Lambda Pengembang.

Lihat Juga

- Mengumumkan Support Lambda untukPowerShell Core di BlogAWS Developer
- AWSLambdaPSCoremodul di situsPowerShell Gallery
- Menyiapkan LingkunganPowerShell Pengembangan
- AWSAlat Lambda untuk Powershell aktifGitHub
- AWSKonsol Lambda

Amazon SQS, Amazon SNS dan Tools for 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
- Memberikan izin pengguna IAM atau izin akun AWS untuk mempublikasikan topik SNS dan membaca pesan dari antrean 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 antrean 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 antrean 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" :
 "llE17A2+XOuJZnw3TlgcXz4C4KPLXZxbxoEMIirelhl3u/oxkWmz5+9tJKFMns1ZOqQvKxk
+ExfEZcD5yWt6biVuBb8pyRmZ1bO3hUENl3ayv2WQiQT1vpLpM7VEQN5m+hLIiPFcs
                        vyuGkJReV7lOJWPHnCN
+qTE2lId2RPkFOeGtLGawTsSPTWEvJdDbLlf7E0zZ0q1niXTUtpsZ8Swx01X3QO6u9i9qBFt0ekJFZNJp6Avu05hIklb4yo
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          :
 AQEB2vvk1e5cOKFjeIWJticabkc664yuDEjhucnIOqdVUmie7bX7GiJbl7F0enABUgaI2XjEcNPxixhVc/
wfsAJZLNHnl8SlbQa0R/kD+Saqa4OIvfj8x3M4Oh1yM1cVKpYmhAzsYrAwAD5g5FvxNBD6zs
                           +HmXdkax2Wd+9AxrHlQZV5ur1MoByKWWbDbsqoYJTJquCclOgWIak/sBx/
daBRMTiVQ4GHsrQWMVHtNC14q7Jy/0L2dkmb4dzJfJq0VbFSX1G+u/lrSLpgae+Dfux646y8yFiPFzY4ua4mCF/
SVUn63Spy
                           sHN12776axknhg3j9K/Xwj54DixdsegnrKoLx+ctI
+0jzAetBR66Q1VhIoJAq7s0a2MseyOeM/Jjucg6Sr9VUnTWVhV8ErXmotoiEg==
```

CloudWatch dari AWS Tools for Windows PowerShell

Bagian ini menunjukkan contoh cara menggunakan Tools for Windows PowerShell untuk menerbitkan data metrik khusus ke CloudWatch.

Contoh ini mengasumsikan bahwa Anda telah menetapkan kredensial default dan wilayah default untuk sesi PowerShell Anda.

Mempublikasikan Metrik Khusus ke Dasbor CloudWatch Anda

Kode PowerShell berikut menginisialisasi objek `MetricDatum` CloudWatch dan mempostingnya ke layanan. Anda dapat melihat hasil dari 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-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 AWS Tools for PowerShell
- AmazonCloudWatchClient.PutMetricData (Referensi .NET SDK)
- MetricDatum (Referensi API Layanan)
- Konsol Amazon CloudWatch

Menggunakan ClientConfig parameter dalam cmdlet

`ClientConfig`Parameter dapat digunakan untuk menentukan pengaturan konfigurasi tertentu saat Anda terhubung ke layanan. Sebagian besar properti yang mungkin dari parameter ini didefinisikan dalam `Amazon.Runtime.ClientConfig`kelas, yang diwariskan ke dalam API untukAWS layanan. Untuk contoh warisan sederhana, lihat `Amazon.Keyspaces.AmazonKeyspacesConfig`kelas. Selain itu, beberapa layanan mendefinisikan properti tambahan yang sesuai hanya 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 splatting untuk meneruskan kumpulan nilai parameter ke perintah sebagai satu unit. Metode ini ditunjukkan dalam contoh berikut. Contoh mengasumsikan bahwa`AWS.Tools.S3` modul telah diinstal dan diimpor, dan bahwa Anda memiliki profil`[default]` kredenial dengan izin yang sesuai.

Mendefinisikan**`ClientConfig`** objek

```
$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 splatting

```
$params=@{
    ClientConfig=@{
        ForcePathStyle=$true
        Timeout=[TimeSpan]::FromMilliseconds(150000)
    }
    BucketName="<BUCKET_NAME>"
}

Get-S3Object @params
```

Menggunakan properti yang tidak terdefinisi

Saat menggunakan PowerShell splatting, jika Anda menentukan`ClientConfig` properti yang tidak ada,AWS Tools for PowerShell tidak mendeteksi kesalahan sampai 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 serupa dengan yang 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.
```

MenentukanWilayah AWS

Anda dapat menggunakan`ClientConfig` parameterWilayah AWS untuk mengatur perintah. Wilayah diatur melalui`RegionEndpoint` properti. AWS Tools for PowerShellMenghitung Wilayah untuk digunakan sesuai dengan prioritas berikut:

1. `-Region`Parameternya
2. Wilayah dilewatkan dalam`ClientConfig` parameter
3. PowerShell Sesi
4. AWS`config`File yang dibagikan
5. Variabel lingkungan
6. Metadata instans Amazon EC2, jika diaktifkan.

Keamanan untuk AWS Produk atau Layanan ini

Keamanan cloud di Amazon Web Services (AWS) merupakan prioritas tertinggi. Sebagai pelanggan AWS, Anda akan mendapatkan manfaat dari pusat data dan arsitektur jaringan yang dibangun untuk memenuhi persyaratan organisasi yang paling sensitif terhadap keamanan. Keamanan adalah tanggung jawab bersama antara AWS dan Anda. Model Tanggung Jawab Bersama menggambarkan ini sebagai Keamanan dari Cloud dan Keamanan dalam Cloud.

Keamanan Cloud – AWS bertanggung jawab untuk melindungi infrastruktur yang menjalankan semua layanan yang ditawarkan di AWS Cloud dan menyediakan layanan yang dapat Anda gunakan dengan aman. Tanggung Jawab keamanan kami merupakan prioritas tertinggi di AWS, dan keefektifan keamanan kami diuji dan diverifikasi secara berkala oleh auditor pihak ketiga sebagai bagian dari Program Kepatuhan AWS.

Keamanan di Cloud – Tanggung jawab Anda ditentukan oleh layanan AWS yang sedang Anda gunakan, serta faktor lain termasuk sensitivitas data Anda, persyaratan organisasi Anda, dan undang-undang dan hukum yang berlaku.

AWSProduk atau layanan ini mengikuti model tanggung jawab bersama melalui layanan Amazon Web Services (AWS) tertentu yang didukungnya. Untuk informasi keamanan layanan AWS, lihat halaman dokumentasi keamanan layanan AWS dan layanan AWS yang berada dalam cakupan upaya kepatuhan AWS 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

Perlindungan data dalam AWS produk atau layanan ini

Model tanggung jawab AWS bersama model berlaku untuk perlindungan data dalam AWS produk atau layanan ini. Sebagaimana diuraikan dalam model ini, AWS bertanggung jawab untuk memberikan perlindungan terhadap infrastruktur global yang menjalankan semua AWS Cloud. Anda harus bertanggung jawab untuk memelihara kendali terhadap konten yang di-hosting pada

infrastruktur ini. Anda juga bertanggung jawab atas konfigurasi keamanan dan tugas manajemen untuk berbagai layanan Layanan AWS yang Anda gunakan. Untuk informasi selengkapnya tentang privasi data, silakan lihat Pertanyaan Umum Privasi Data. Untuk informasi tentang perlindungan data di Eropa, silakan lihat postingan blog Model Tanggung Jawab Bersama AWS dan GDPR di Blog Keamanan AWS.

Untuk tujuan perlindungan data, sebaiknya Anda melindungi kredensial Akun AWS dan menyiapkan akun pengguna individu dengan AWS IAM Identity Center atau AWS Identity and Access Management (IAM). Dengan cara seperti itu, setiap pengguna hanya diberi izin yang diperlukan untuk memenuhi tanggung jawab tugas mereka. Kami juga merekomendasikan agar Anda mengamankan data Anda dengan cara-cara berikut:

- Gunakan autentikasi multi-faktor (MFA) pada setiap akun.
- Gunakan SSL/TLS untuk melakukan komunikasi dengan sumber daya AWS. Kami mensyaratkan TLS 1.2 dan merekomendasikan TLS 1.3.
- Siapkan API dan log aktivitas pengguna dengan AWS CloudTrail.
- Gunakan solusi enkripsi AWS, bersama dengan semua kontrol keamanan default dalam layanan Layanan AWS.
- Gunakan layanan keamanan terkelola lanjutan seperti Amazon Macie, yang membantu menemukan dan mengamankan data pribadi yang disimpan di Amazon S3.
- Jika Anda memerlukan modul kriptografi tervalidasi FIPS 140-2 ketika mengakses AWS melalui antarmuka baris perintah atau API, gunakan titik akhir FIPS. Untuk informasi selengkapnya tentang titik akhir FIPS yang tersedia, silakan lihat Standar Pemrosesan Informasi Federal (FIPS) 140-2.

Kami sangat merekomendasikan agar Anda tidak pernah memasukkan informasi identifikasi yang sensitif, seperti nomor rekening pelanggan Anda, ke dalam tag atau bidang isian bebas seperti bidang Nama. Ini termasuk saat Anda bekerja dengan AWS produk atau layanan ini atau lainnya Layanan AWS menggunakan konsol, APIAWS CLI, atau AWS SDK. Data apa pun yang Anda masukkan ke dalam tag atau bidang isian bebas yang digunakan untuk nama dapat digunakan untuk log penagihan atau log diagnostik. Saat Anda memberikan URL ke server eksternal, sebaiknya jangan menyertakan informasi kredensial di URL untuk memvalidasi permintaan Anda ke server tersebut.

Enkripsi data

Fitur utama dari setiap layanan aman adalah bahwa informasi dienkripsi ketika tidak aktif digunakan.

Enkripsi saat Istirahat

AWS Tools for PowerShell sendiri tidak menyimpan data konsumen selain kredensialnya yang dibutuhkan untuk berinteraksi dengan layanan AWS atas nama pengguna.

Jika Anda menggunakan AWS Tools for PowerShell untuk membuka sebuah layanan AWS yang mentransmisikan data pelanggan ke komputer lokal Anda untuk penyimpanan, kemudian lihat bagian Keamanan & Kepatuhan dalam Panduan Pengguna layanan tersebut untuk informasi tentang bagaimana data tersebut disimpan, dilindungi, dan dienkripsi.

Enkripsi dalam Transit

Secara default, semua data yang dikirim dari komputer klien yang menjalankan titik akhir layanan AWS Tools for PowerShell dan AWS 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

(IAM) AWS Identity and Access Management adalah Layanan AWS yang membantu seorang administrator dalam mengendalikan akses ke sumber daya AWS secara aman. Administrator IAM mengontrol siapa yang dapat terautentikasi (masuk) dan berwenang (memiliki izin) untuk menggunakan sumber daya AWS. IAM adalah sebuah layanan Layanan AWS yang dapat Anda gunakan tanpa dikenakan biaya tambahan.

Topik

- Audiens
- Mengautentikasi dengan identitas
- Mengelola kebijakan menggunakan akses
- Bagaimana Layanan AWS bekerja dengan IAM
- Pemecahan masalah identitas dan akses AWS

Audiens

Cara menggunakan AWS Identity and Access Management (IAM) berbeda, tergantung pada pekerjaan yang Anda lakukan di 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 fitur AWS untuk melakukan pekerjaan, Anda mungkin memerlukan izin tambahan. Memahami cara mengelola akses dapat membantu Anda meminta izin yang tepat dari administrator Anda. Jika Anda tidak dapat mengakses fiturAWS, lihat Pemecahan masalah identitas dan akses AWS atau panduan pengguna yang Layanan AWS Anda gunakan.

Administrator layanan – Jika Anda bertanggung jawab atas sumber daya AWS 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 Anda 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 IAMAWS, 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

Autentikasi merupakan cara Anda untuk masuk ke AWS dengan menggunakan kredensial identitas Anda. Anda harus terautentikasi (masuk keAWS) sebagai Pengguna root akun AWS, sebagai pengguna IAM, atau dengan mengambil peran IAM.

Anda dapat masuk ke AWS sebagai identitas terfederasi dengan menggunakan kredensial yang disediakan melalui sumber identitas. AWS IAM Identity Center Para pengguna (Pusat Identitas IAM), otentikasi sign-on tunggal perusahaan Anda, dan kredensial Google atau Facebook Anda merupakan contoh identitas terfederasi. Saat Anda masuk sebagai identitas terfederasi, administrator Anda sebelumnya menyiapkan federasi identitas dengan menggunakan peran IAM. Ketika Anda mengakses AWS dengan menggunakan federasi, Anda secara tidak langsung mengambil suatu peran.

Tergantung pada jenis pengguna Anda, Anda dapat masuk ke AWS Management Console atau portal akses AWS. Untuk informasi selengkapnya tentang masuk ke AWS, silakan lihat Cara masuk ke Akun AWS Anda di Panduan Pengguna AWS Sign-In.

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 peralatan AWS, maka Anda harus menandatangani sendiri permintaan tersebut. Untuk informasi selengkapnya tentang penggunaan metode yang disarankan untuk menandatangani permintaan sendiri, silakan lihat Menandatangani permintaan API AWS di Panduan Pengguna IAM.

Terlepas dari metode autentikasi yang Anda gunakan, Anda mungkin juga diminta untuk menyediakan informasi keamanan tambahan. Sebagai contoh, AWS menyarankan supaya Anda menggunakan autentikasi multi-faktor (MFA) untuk meningkatkan keamanan akun Anda. Untuk mempelajari selengkapnya, silakan lihat Autentikasi multi-faktor di Panduan Pengguna AWS IAM Identity Center dan Menggunakan autentikasi multi-faktor (MFA) di AWS di Panduan Pengguna IAM.

Pengguna root Akun AWS

Ketika Anda membuat Akun AWS, Anda memulai dengan satu identitas masuk yang memiliki akses ke semua Layanan AWS dan sumber daya di akun tersebut. Identitas ini disebut pengguna root Akun AWS dan diakses dengan cara masuk ke alamat email dan kata sandi yang Anda gunakan untuk membuat akun. Kami sangat menyarankan 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, silakan lihat Tugas yang memerlukan kredensial pengguna root di Panduan Pengguna IAM.

Identitas terfederasi

Praktik terbaiknya berupa, mewajibkan pengguna manusia, termasuk pengguna yang memerlukan akses administrator, untuk menggunakan federasi dengan penyedia identitas untuk mengakses Layanan AWS dengan menggunakan kredensial temporer.

Identitas terfederasi adalah pengguna dari direktori pengguna perusahaan Anda, penyedia identitas web, dikenal sebagai AWS Directory Service, direktori Pusat Identitas, atau pengguna mana pun yang mengakses Layanan AWS dengan menggunakan kredensial yang disediakan melalui sumber identitas. Ketika identitas terfederasi mengakses Akun AWS, identitas tersebut mengambil peran, dan peran memberikan kredensial temporer.

Untuk pengelolaan akses terpusat, kami sarankan Anda menggunakanAWS 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 Akun AWS Anda dan aplikasi Anda. Untuk informasi tentang Pusat Identitas IAM, silakan lihat Apakah Pusat Identitas IAM itu? di User Guide AWS IAM Identity Center.

Pengguna dan Grup IAM

Pengguna IAM adalah identitas dalam Akun AWS Anda yang memiliki izin khusus untuk satu orang atau aplikasi. Apabila memungkinkan, kami menyarankan untuk mengandalkan pada kredensial temporer alih-alih membuat pengguna IAM yang memiliki kredensyal jangka panjang seperti kata sandi dan kunci akses. Namun, jika Anda memiliki kasus penggunaan khusus yang memerlukan kredensial jangka panjang dengan pengguna IAM, kami menyarankan Anda memutar kunci akses. Untuk informasi selengkapnya, silakan lihat Memutar kunci akses secara teratur untuk kasus penggunaan yang memerlukan kredensial jangka panjang di Panduan Pengguna IAM.

Grup IAM adalah identitas yang menerangkan secara spesifik kumpulan pengguna IAM. Anda tidak dapat masuk sebagai kelompok. Anda dapat menggunakan grup untuk menerangkan secara spesifik izin untuk beberapa pengguna sekaligus. Grup membuat izin lebih mudah dikelola untuk sekelompok besar pengguna. Sebagai contoh, Anda dapat memiliki grup yang diberi nama AdminIAM 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 tersebut dimaksudkan untuk dapat digunakan oleh siapa pun yang membutuhkannya. Pengguna memiliki kredensial jangka panjang permanen, tetapi peran memberikan kredensial temporer. Untuk mempelajari selengkapnya, silakan lihat Kapan harus membuat pengguna IAM (alih-alih peran) di Panduan Pengguna IAM.

Peran IAM

Peran IAM merupakan identitas dalam Akun AWS Anda yang memiliki izin khusus. Peran ini mirip dengan pengguna IAM, tetapi tidak terkait dengan orang tertentu. Anda dapat menggunakan peran IAM untuk sementara dalam AWS Management Console dengan berganti peran. Anda dapat mengambil peran dengan cara memanggil operasi API AWS CLI atau AWS atau menggunakan URL kustom. Untuk informasi selengkapnya tentang cara menggunakan peran, silakan lihat menggunakan peran IAM di Panduan Pengguna IAM.

IAM role dengan kredensial temporer berguna dalam situasi berikut:

- Akses pengguna gabungan – Untuk menetapkan izin ke sebuah identitas terfederasi, Anda harus membuat sebuah peran dan menentukan izin untuk peran tersebut. Ketika identitas gabungan terfederasi mengautentikasi, identitas tersebut terhubung dengan peran dan diberikan izin yang

ditentukan oleh peran. Untuk informasi tentang peran-peran untuk federasi, silakan lihat Membuat sebuah peran untuk Penyedia Identitas pihak ketiga di Panduan Pengguna IAM. Jika Anda menggunakan Pusat Identitas IAM, Anda mengonfigurasi serangkain izin. Untuk mengontrol apa yang dapat diakses identitas Anda setelah identitas tersebut diautentikasi, Pusat Identitas IAM mengkorelasikan izin yang diatur ke peran dalam IAM. Untuk informasi tentang rangkaian izin, silakan lihat Rangkaian izin di Panduan Pengguna AWS IAM Identity Center.

- Izin pengguna IAM sementara – Pengguna atau peran IAM dapat mengambil peran IAM untuk sementara mengambil izin berbeda untuk tugas tertentu.
- Akses lintas akun – Anda dapat menggunakan peran IAM untuk mengizinkan seseorang (pengguna utama tepercaya) di akun berbeda untuk mengakses sumber daya yang ada di akun Anda. Peran adalah cara utama untuk memberikan akses lintas akun. Namun, pada beberapa Layanan AWS, Anda dapat melampirkan kebijakan secara langsung ke sumber daya (alih-alih menggunakan suatu peran sebagai proksi). Untuk mempelajari perbedaan antara kebijakan peran dan kebijakan berbasis sumber daya untuk akses lintas akun, silakan lihat Bagaimana peran IAM role berbeda dari kebijakan berbasis sumber daya dalam Panduan Pengguna IAM.
- Akses lintas layanan – Sebagian Layanan AWS menggunakan fitur di Layanan AWS lainnya. Sebagai contoh, ketika Anda melakukan panggilan dalam suatu layanan, lazim pada layanan tersebut menjalankan aplikasi di Amazon EC2 atau menyimpan objek di Amazon S3. Suatu layanan mungkin melakukan hal tersebut menggunakan izin pengguna utama panggilan, menggunakan peran layanan, atau peran tertaut layanan.
  - Sesi akses maju (FAS) – Ketika Anda menggunakan pengguna IAM atau peran IAM untuk melakukan tindakan-tindakan di AWS, Anda akan dianggap sebagai seorang pengguna utama. Ketika Anda menggunakan beberapa layanan, Anda mungkin melakukan sebuah tindakan yang kemudian dilanjutkan oleh tindakan lain pada layanan yang berbeda. FAS menggunakan izin dari pengguna utama untuk memanggil Layanan AWS, yang dikombinasikan dengan Layanan AWS yang diminta untuk membuat pengajuan ke layanan hilir. Permintaan FAS hanya diajukan ketika sebuah layanan menerima pengajuan yang memerlukan interaksi dengan Layanan AWS lain atau sumber daya lain untuk diselesaikan. Dalam hal ini, Anda harus memiliki izin untuk melakukan kedua tindakan tersebut. Untuk detail kebijakan ketika mengajukan permintaan FAS, silakan lihat Meneruskan sesi akses.
  - Peran layanan – Sebuah peran layanan adalah sebuah peran IAM yang dijalankan oleh suatu layanan untuk melakukan tindakan atas nama Anda. Administrator IAM dapat membuat, memodifikasi, dan menghapus peran layanan dari dalam IAM. Untuk informasi selengkapnya, silakan lihat Membuat sebuah peran untuk mendelegasikan izin ke Layanan AWS dalam Panduan pengguna IAM.

  - Peran tertaut layanan – Peran tertaut layanan adalah tipe peran layanan yang tertaut dengan Layanan AWS. Layanan tersebut dapat menjalankan peran untuk melakukan sebuah tindakan atas nama Anda. Peran tertaut layanan akan muncul di Akun AWS Anda dan dimiliki oleh layanan tersebut. Administrator IAM dapat melihat, tetapi tidak dapat mengedit izin untuk peran tertaut layanan.
- Aplikasi yang berjalan di Amazon EC2 – Anda dapat menggunakan peran IAM untuk mengelola kredensial temporer untuk aplikasi yang berjalan di instans EC2 dan mengajukan permintaan AWS CLI atau API AWS. Cara ini lebih baik daripada menyimpan kunci akses dalam instans EC2. Untuk memberikan peran AWS ke instans EC2 dan menjadikannya terdapat di semua aplikasinya, Anda dapat membuat profil instans yang dilampirkan ke instans tersebut. Profil instans berisi peran dan memungkinkan program yang berjalan di instans EC2 untuk mendapatkan kredensial temporer. Untuk informasi selengkapnya, silakan lihat Menggunakan peran IAM untuk memberikan izin ke aplikasi yang berjalan di instans Amazon EC2 di Panduan Pengguna IAM.

Untuk mempelajari apakah kita harus menggunakan peran IAM atau pengguna IAM, silakan lihat Kapan harus membuat peran IAM (alih-alih pengguna) di Panduan Pengguna IAM.

Mengelola kebijakan menggunakan akses

Anda mengendalikan akses di AWS dengan membuat kebijakan dan melampirkannya ke identitas atau sumber daya AWS. Kebijakan adalah objek di AWS yang, ketika terkait dengan identitas atau sumber daya, akan menentukan izinnya. AWS mengevaluasi kebijakan-kebijakan tersebut ketika seorang pengguna utama (pengguna, root user, atau sesi peran) mengajukan permintaan. Izin dalam kebijakan menentukan apakah permintaan diberikan atau ditolak. Sebagian besar kebijakan disimpan di AWS sebagai dokumen JSON. Untuk informasi selengkapnya tentang struktur dan isi dokumen kebijakan JSON, silakan lihat Gambaran Umum kebijakan JSON di Panduan Pengguna IAM.

Administrator dapat menggunakan kebijakan JSON AWS untuk menentukan secara spesifik siapa yang memiliki akses pada apa. Yaitu, pengguna utama manakah yang dapat melakukan tindakan pada sumber daya apa, dan dengan syarat apa.

Secara bawaan, para pengguna dan peran tidak memiliki izin. Untuk mengabulkan izin kepada pengguna untuk melakukan tindakan di sumber daya yang mereka perlukan, administrator IAM dapat membuat kebijakan IAM. Administrator kemudian akan dapat menambahkan kebijakan IAM ke peran, dan para pengguna dapat mengambil peran.

Kebijakan IAM mendefinisikan izin untuk suatu tindakan terlepas dari metode yang Anda gunakan untuk pengoperasiannya. Sebagai contoh, anggap saja Anda memiliki kebijakan yang mengizinkan

tindakan `iam:GetRole` . Pengguna dengan kebijakan tersebut dapat memperoleh informasi peran dari AWS Management Console, AWS CLI, atau APIAWS.

Kebijakan berbasis identitas

Kebijakan berbasis identitas adalah dokumen kebijakan izin JSON yang dapat Anda lampirkan ke sebuah identitas, misalnya pengguna IAM, grup pengguna IAM, atau peran IAM. Kebijakan ini mengontrol apa yang pengguna tindakan dan peran dapat kerjakan, pada sumber daya mana, dan dalam keadaan apa. Untuk mempelajari cara membuat kebijakan berbasis identitas, silakan lihat Membuat kebijakan IAM dalam Panduan Pengguna IAM.

Kebijakan berbasis identitas dapat dikategorikan lebih lanjut sebagai kebijakan inline atau kebijakan terkelola. Kebijakan inline ditanam secara langsung ke pengguna tunggal, grup, atau peran. Kebijakan terkelola adalah kebijakan yang berdiri sendiri yang dapat Anda lampirkan ke beberapa pengguna, grup, dan peran di Akun AWS Anda. Kebijakan terkelola mencakup kebijakan terkelola AWS dan kebijakan terkelola pelanggan. Untuk mempelajari cara memilih antara kebijakan terkelola atau kebijakan inline, silakan lihat Memilih antara kebijakan terkelola dan kebijakan inline diPanduan Pengguna IAM.

Kebijakan berbasis sumber daya

Kebijakan berbasis sumber daya adalah dokumen kebijakan JSON yang Anda lampirkan ke sumber daya. Contoh kebijakan-kebijakan berbasis sumber daya adalah kebijakan terpercaya 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 yang dilampiri kebijakan, kebijakan tersebut menentkan tindakan apa yang dapat dilakukan oleh pengguna utama yang ditentukan di sumber daya tersebut dan dalam kondisi apa. Anda harus menentukan pengguna utama dalam kebijakan berbasis sumber daya. Pengguna utama dapat mencakup akun, pengguna, peran, pengguna gabungan, atau Layanan AWS.

Kebijakan berbasis sumber daya merupakan kebijakan inline yang terletak di layanan tersebut. Anda tidak dapat menggunakan kebijakan terkelola AWS dari IAM dalam kebijakan berbasis sumber daya.

Daftar kontrol akses (ACL)

Daftar kontrol akses (ACL) mengendalikan pengguna utama mana (anggota akun, pengguna, atau peran) yang memiliki izin untuk mengakses sumber daya. ACL serupa dengan kebijakan berbasis sumber daya, meskipun kebijakan-kebijakan tersebut tidak menggunakan format dokumen kebijakan JSON.

Amazon S3, AWS WAF, dan Amazon VPC adalah contoh-contoh layanan yang mendukung ACL. Untuk mempelajari ACL selengkapnya, silakan lihat Gambaran umum daftar kontrol akses (ACL) di Panduan Pengembang Layanan Penyimpanan Ringkas Amazon.

Tipe-tipe kebijakan lain

AWS mendukung tipe kebijakan tambahan, yang kurang umum. Tipe-tipe kebijakan ini dapat mengatur izin maksimum yang diberikan kepada Anda oleh tipe kebijakan yang lebih umum.

- Batasan izin – Batasan izin adalah fitur lanjutan tempat Anda mengatur izin maksimum yang dapat diberikan oleh kebijakan berbasis identitas kepada 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 batas 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 menindihi izin. Untuk informasi selengkapnya tentang batasan izin, silakan lihat Batasan izin untuk entitas IAM di Panduan Pengguna IAM.
- Kebijakan kontrol layanan (SCP) – SCP adalah kebijakan JSON yang menentukan izin maksimum untuk sebuah organisasi atau unit organisasi (OU) di AWS Organizations. AWS Organizations adalah layanan untuk mengelompokkan dan secara terpusat mengelola beberapa Akun AWS yang dimiliki bisnis Anda. Jika Anda mengaktifkan semua fitur di suatu organisasi, maka Anda dapat menerapkan kebijakan kontrol layanan (SCP) ke salah satu atau ke semua akun Anda. SCP membatasi izin untuk entitas dalam akun anggota, termasuk setiap Pengguna root akun AWS. Untuk informasi selengkapnya tentang Organisasi dan SCP, silakan lihat Cara kerja SCP di Panduan Pengguna AWS Organizations.
- Kebijakan sesi – Kebijakan sesi adalah kebijakan lanjutan yang Anda berikan sebagai parameter ketika Anda membuat sesi sementara secara terprogram untuk peran atau pengguna gabungan. Izin sesi yang dihasilkan adalah perpotongan kebijakan berbasis identitas pengguna atau peran dan kebijakan sesi. Izin juga dapat berasal dari kebijakan berbasis sumber daya. Penolakan eksplisit dalam salah satu kebijakan ini menindihi izin. Untuk informasi selengkapnya, silakan lihat Kebijakan sesi di Panduan Pengguna IAM.

Berbagai tipe kebijakan

Ketika beberapa tipe kebijakan berlaku pada suatu permintaan, izin yang dihasilkan lebih rumit untuk dipahami. Untuk mempelajari cara AWS menentukan apakah akan mengizinkan permintaan ketika beberapa tipe kebijakan dilibatkan, silakan 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 AWSlayanan 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.

Pemecahan masalah identitas dan akses AWS

Gunakan informasi berikut untuk membantu Anda mendiagnosis dan memperbaiki masalah umum yang mungkin Anda temukan saat bekerja dengan AWS dan IAM.

Topik

- Saya tidak diotorisasi untuk melakukan tindakan di AWS
- Saya tidak berwenang untuk melakukan iam: PassRole
- Saya ingin mengizinkan orang di luar Akun AWS saya untuk mengakses sumber daya AWS saya

Saya tidak diotorisasi untuk melakukan tindakan di AWS

Jika Anda menerima pesan galat bahwa Anda tidak memiliki otorisasi untuk melakukan tindakan, kebijakan Anda harus diperbarui agar Anda dapat melakukan tindakan tersebut.

Contoh galat berikut terjadi ketika pengguna IAM `mateojackson` mencoba menggunakan konsol untuk melihat detail tentang suatu sumber daya *`my-example-widget`* rekaan, tetapi tidak memiliki izin `awes:`*`GetWidget`* rekaan.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 awes: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 `awes:`*`GetWidget`*.

Jika Anda membutuhkan bantuan, hubungi administrator AWS Anda. Administrator Anda adalah orang yang memberikan kredensial masuk Anda.

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 peranAWS.

Sebagian Layanan AWS mengizinkan Anda untuk memberikan peran yang sudah ada ke layanan tersebut alih-alih membuat peran layanan baru atau peran tertaut-layanan. Untuk melakukan tindakan tersebut, Anda harus memiliki izin untuk memberikan peran pada layanan tersebut.

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 membutuhkan bantuan, hubungi administrator AWS Anda. Administrator Anda adalah orang yang memberikan kredensial masuk Anda.

Saya ingin mengizinkan orang di luar Akun AWS saya untuk mengakses sumber daya AWS saya

Anda dapat membuat peran yang dapat digunakan para pengguna di akun lain atau orang-orang di luar organisasi Anda untuk mengakses sumber daya Anda. Anda dapat menentukan siapa yang dipercaya untuk mengambil peran tersebut. Untuk layanan yang mendukung kebijakan berbasis sumber daya atau daftar kontrol akses (ACL), Anda dapat menggunakan kebijakan tersebut untuk memberi akses kepada orang ke sumber daya Anda.

Untuk mempelajari selengkapnya, lihat hal berikut:

- Untuk mempelajari apakah AWS mendukung fitur-fitur ini, lihat Bagaimana Layanan AWS bekerja dengan IAM.
- Untuk mempelajari cara memberikan akses ke sumber daya di seluruh akun Akun AWS yang Anda miliki, lihat Memberikan akses ke pengguna IAM di akun Akun AWS lain yang Anda miliki dalam Panduan Pengguna IAM.

- Untuk mempelajari cara memberikan akses ke sumber daya Anda ke pihak ketiga Akun AWS, silakan lihat Menyediakan akses ke akun Akun AWS yang dimiliki oleh pihak ketiga dalam Panduan Pengguna IAM.
- Untuk mempelajari cara memberikan akses melalui federasi identitas, silakan lihat Menyediakan akses ke pengguna terautentikasi eksternal (gabungan identitas) di Panduan Pengguna IAM .
- Untuk mempelajari perbedaan antara penggunaan peran dan kebijakan berbasis sumber daya untuk akses lintas akun, lihat Perbedaan IAM role dan kebijakan berbasis sumber daya 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 AWSdi Lingkup oleh Program Kepatuhan Layanan AWS dan pilih program kepatuhan yang Anda minati. Untuk informasi umum, silakan lihat Program Kepatuhan AWS .

Anda bisa 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, serta hukum dan peraturan yang berlaku. AWS menyediakan sumber daya berikut untuk membantu kepatuhan:

- Panduan Memulai Cepat Keamanan dan Kepatuhan — Panduan penerapan ini membahas pertimbangan arsitektur dan memberikan langkah-langkah untuk menerapkan lingkungan dasar AWS yang berfokus pada keamanan dan kepatuhan.
- Arsitektur untuk Keamanan dan Kepatuhan HIPAA di Amazon Web Services — Whitepaper ini menjelaskan bagaimana perusahaan dapat menggunakan AWS untuk membuat aplikasi yang memenuhi syarat HIPAA.

> **Note**
>
> Tidak semua Layanan AWS memenuhi syarat HIPAA. Untuk informasi selengkapnya, lihat Referensi Layanan yang Memenuhi Syarat HIPAA.

- Sumber Daya Kepatuhan AWS – Kumpulan buku kerja dan panduan ini mungkin berlaku untuk industri dan lokasi Anda.

- AWSPanduan 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 di Panduan Developer AWS Config – Layanan AWS Config menilai seberapa baik konfigurasi sumber daya Anda dalam mematuhi praktik-praktik internal, pedoman industri, dan regulasi internal.
- AWS Security Hub— Ini Layanan AWS memberikan pandangan komprehensif tentang keadaan keamanan Anda di dalamnyaAWS. Security Hub menggunakan kontrol keamanan untuk mengevaluasi AWS sumber daya Anda dan untuk memeriksa kepatuhan Anda terhadap standar industri keamanan dan praktik terbaik. Untuk daftar layanan dan kontrol yang didukung, lihat Referensi kontrol Security Hub.
- AWS Audit Manager – Layanan AWS ini akan membantu Anda untuk terus-menerus mengaudit penggunaan AWS untuk menyederhanakan bagaimana Anda mengelola risiko dan kepatuhan terhadap regulasi dan standar industri.

AWSProduk atau layanan ini mengikuti model tanggung jawab bersama melalui layanan Amazon Web Services (AWS) tertentu yang didukungnya. Untuk informasi keamanan layanan AWS, lihat halaman dokumentasi keamanan layanan AWS dan layanan AWS yang berada dalam cakupan upaya kepatuhan AWS 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 for .NET Pengembang.

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 AWS Tools for PowerShellCmdlet.

Riwayat dokumen

Topik berikut menjelaskan perubahan signifikan pada dokumentasi untuk AWS Tools for 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 AWS Tools for PowerShell, lihat catatan rilis.

Perubahan	Deskripsi	Tanggal
Konfigurasikan otentikasi alat dengan AWS	Menambahkan informasi tentang dukungan untuk SSO di. AWS Tools for 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	Panduan yang diperbarui untuk menyelaraskan dengan praktik terbaik IAM. Untuk informasi selengkapnya, 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	Panduan yang diperbarui untuk menyelaraskan dengan praktik terbaik IAM. Untuk informasi selengkapnya, lihat	September 8, 2023

	Praktik terbaik keamanan di IAM.	
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 Instans Amazon EC2 Menggunakan Windows PowerShell	Menambahkan catatan tentang pensiun EC2-Classic.	26 Juli 2022
AWS Tools for 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
AWS Tools for 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

AWS Tools for PowerShell 3.3.343.0	Menambahkan informasi ke bagian Menggunakan AWS Tools for 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	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 - sakelar. DeleteObjects Memperbaiki masalah kegunaan dengan Set-AWSCredentials. Inisialis asi- AWSDefaults laporan dari mana ia memperoleh kredensi dan data wilayah. Stop-EC2Instance menerima contoh Amazon.EC2.Model.R eservation sebagai input. Daftar Umum <T> jenis parameter diganti dengan jenis array (T[]). Cmdlet yang menghapus atau menghenti kan perintah sumber daya untuk konfirmasi sebelum penghapusan. Write-S3O bject mendukung konten teks selaras untuk mengunggah ke Amazon S3.	15 Mei 2013

AWS Tools for Windows PowerShell 1.0.1.0	Lokasi instalasi PowerShell 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	21 Desember 2012

	melakukan `AWSPowerShell.psd1` 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.