使用者指南

AWS Tools for PowerShell

AWS Tools for PowerShell: 使用者指南

Table of Contents

AWS Tools for PowerShell 是什麼？

AWS Tools for PowerShell 是一組 PowerShell 模組，建立在 AWS SDK for .NET 公開的功能之上。AWS Tools for PowerShell 可讓您從 PowerShell 命令列對 AWS 資源執行指令碼作業。

這些 cmdlet 會提供慣常的 PowerShell 方式來指定參數與處理結果，即使它們的實作方式使用各種 AWS 服務 HTTP 查詢 API。舉例來說，AWS Tools for PowerShell 的 cmdlet 支援 PowerShell 管道傳輸，意即您可以採用管道傳輸，將 PowerShell 物件傳入或傳出 cmdlet。

AWS Tools for PowerShell 讓您以彈性靈活的方式處理憑證，包括對 AWS Identity and Access Management (IAM) 基礎設施的支援。您可以使用這些工具，搭配 IAM 使用者憑證、暫時安全字符和 IAM 角色。

AWS Tools for PowerShell 支援開發套件所支援的相同服務與 AWS 區域組合。您可以在執行 Windows、Linux 或 macOS 作業系統的電腦上安裝 AWS Tools for PowerShell。

> **Note**
>
> AWS Tools for PowerShell 第 4 版是最新主要版本，而且是與 AWS Tools for PowerShell 3.3 版回溯相容的更新。此版本新增了大幅改善的項目，同時保持現有的 Cmdlet 行為。在升級至新版本後，現有指令碼應該能繼續運作，但建議您在升級前先進行徹底的測試。如需第 4 版變更的詳細資訊，請參閱從 AWS Tools for PowerShell 3.3 版遷移至第 4 版。

AWS Tools for PowerShell 以下列三種不同套裝服務提供：

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

開發套件主要版本的維護與支援

如需開發套件主要版本及其基礎相依性之維護與支援的相關資訊，請參閱《AWS 開發套件及工具參考指南》中的以下內容：

- AWS 開發套件及工具維護政策
- AWS 開發套件及工具版本支援對照表

`AWS.Tools` - AWS Tools for PowerShell 的模組化版本

在實際執行環境中執行 PowerShell 的任何電腦，都建議使用此版的 AWS Tools for PowerShell。因為此版本已模組化，因此您必須僅下載並載入您要使用之服務的模組。這樣可以縮減下載時間、記憶體使用量，並能在大多數情況下啟用 `AWS.Tools` Cmdlet 的自動匯入功能，而不需先手動呼叫 `Import-Module`。

這是最新版的 AWS Tools for PowerShell，可在所有支援的作業系統上執行，包括 Windows、Linux 和 macOS。此套件提供一個安裝模組 `AWS.Tools.Installer`、一個通用模組 `AWS.Tools.Common`，以及一個適用於每項 AWS 服務的模組，例如 `AWS.Tools.EC2`、`AWS.Tools.IAM`、`AWS.Tools.S3` 等等。

`AWS.Tools.Installer` 模組提供多個 Cmdlet，可讓您安裝、更新及移除每項 AWS 服務的模組。此模組中的 Cmdlet 會自動確保您擁有支援您要使用之模組所需的所有相依模組。

`AWS.Tools.Common` 模組提供適用於非服務專用的組態和驗證 cmdlet。若要針對 AWS 服務使用 cmdlet，您只需執行命令即可。PowerShell 會自動匯入 `AWS.Tools.Common` 模組和您想要執行其 cmdlet 的 AWS 服務模組。如果您使用 `AWS.Tools.Installer` 模組來安裝服務模組，就會自動安裝此模組。

您可將此版本的 AWS Tools for PowerShell 安裝在執行下列應用程式的電腦上：

- Windows、Linux 或 macOS 版的 PowerShell Core 6.0 或更新版本。
- Windows 版的 Windows PowerShell 5.1 或更新版本及 .NET Framework 4.7.2 或更新版本。

在本指南中，必須明確只指出此版本時，我們會以模組名稱指稱該版本：*`AWS.Tools`*。

AWSPowerShell.NetCore - AWS Tools for PowerShell 的單一模組版本

此版本是由包含所有 AWS 服務支援的單一、大型模組所構成。您必須先手動匯入此模組，才能使用它。

您可將此版本的 AWS Tools for PowerShell 安裝在執行下列應用程式的電腦上：

- Windows、Linux 或 macOS 版的 PowerShell Core 6.0 或更新版本。
- Windows 版的 Windows PowerShell 3.0 或更新版本及 .NET Framework 4.7.2 或更新版本。

在本指南中，必須明確只指出此版本時，我們會以模組名稱指稱該版本：AWSPowerShell.NetCore。

AWSPowerShell -適用於 Windows PowerShell 的單一模組版本

這個版本的 AWS Tools for PowerShell 與 Windows PowerShell 2.0 版到 5.1 版相容，也只能安裝在執行這些版本的 Windows 電腦上。它與 PowerShell Core 6.0 或更新版本，或任何其他作業系統 (Linux 或 macOS) 都不相容。此版本是由包含所有 AWS 服務支援的單一、大型模組所構成。

在本指南中，必須明確只指出此版本時，我們會以模組名稱指稱該版本：AWSPowerShell。

本指南的使用方式

本指南分為以下幾個主要章節：

安裝 AWS Tools for PowerShell

本節說明如何安裝 AWS Tools for PowerShell。其包括如何註冊 AWS (如果還沒有帳戶)，以及如何建立可用來執行 cmdlet 的 IAM 使用者。

開始使用 AWS Tools for Windows PowerShell。

本節會介紹使用 AWS Tools for PowerShell 的基礎原理，例如：指定憑證與 AWS 區域、尋找特定服務的 cmdlet，以及使用 cmdlet 的別名。

在 AWS Tools for PowerShell 中使用 AWS 服務

本節包含使用 AWS Tools for PowerShell 執行一些最常見 AWS 任務的資訊。

安裝 AWS Tools for PowerShell

若要成功安裝和使用 AWS Tools for PowerShell Cmdlet，請參閱下列主題中的步驟。

主題

AWS Tools for PowerShell 在視窗上安裝

以 Windows 為基礎的電腦可以執行下列任何 AWS Tools for PowerShell 套件選項：

- **`AWS.Tools`**-的模組化版本。 AWS Tools for PowerShell每個 AWS 服務都由自己的個人小模塊支持，並具有共享的支持模塊`AWS.Tools.Common`和`AWS.Tools.Installer`.
- AWSPowerShell。 NetCore-單一、大型模組版本的 AWS Tools for PowerShell. 這個單一的大型模組支援所有 AWS 服務。

  > **Note**
  >
  > 請注意，單一模組可能太大，無法與 AWS Lambda 功能一起使用。請改用上面的模組化版本。

- AWSPowerShell - AWS Tools for PowerShell的舊版 Windows 專用、單一、大型模組版本。這個單一的大型模組支援所有 AWS 服務。

您選擇的套件取決於您正在執行的 Windows 版本。

> **Note**
>
> 默認情況下，Windows 工具 PowerShell （AWSPowerShell 模塊）安裝在所有基於 Windows 的 Amazon 機器映像（AMI）上。

設定 AWS Tools for PowerShell 包含下列高階工作，在本主題中詳細說明。

1. 安 AWS Tools for PowerShell 裝適合您環境的套件選項。
2. 執行 `Get-ExecutionPolicy` Cmdlet 來驗證指令碼執行已啟用。
3. 將 AWS Tools for PowerShell 模組匯入您的 PowerShell 工作階段。

必要條件

較新版本的 PowerShell，包括 PowerShell 核心，可作為下載從 Microsoft 在微軟的網站 PowerShell上安裝各種版本。

在 Windows 上安裝 **AWS.Tools**

您可以在執行視窗 PowerShell 5.1 或 PowerShell 核心 6.0 或更新版本的電腦 AWS Tools for PowerShell 上安裝模組化版本。如需有關如何安裝 PowerShell核心的詳細資訊，請參閱 PowerShell在 Microsoft 網站上安裝各種版本的。

有三種方式可以安裝 `AWS.Tools`：

- 使用 `AWS.Tools.Installer` 模組中的 Cmdlet。該模塊簡化了其他模`AWS.Tools`塊的安裝和更新。 `AWS.Tools.Installer`需要`PowerShellGet`，並自動下載並安裝它的更新版本。`AWS.Tools.Installer`自動保持您的模塊版本同步。當您安裝或更新至某個模組的較新版本時，中的指令程式`AWS.Tools.Installer`會自動將所有其他`AWS.Tools`模組更新為相同版本。

  此方法會在下列程序中說明。
- 從 `AWS.Tools.zip` 下載模組，然後解壓縮至其中一個模組資料夾。您可以顯示 `PSModulePath` 環境變數的值來探索自己的模組資料夾。
- 使用`Install-Module`指令程式從程式 PowerShell 庫安裝每個服務模組。

若要使用該`AWS.Tools.Installer`模組`AWS.Tools`在視窗上安裝

1. 啟動 PowerShell 工作階段。

   > **Note**
   >
   > 我們建議您不要 PowerShell 以具有提升權限的系統管理員身分執行，除非手頭的工作需要。這是因為可能會有潛在的安全風險，且不符合最低權限原則。

2. 若要安裝模組化 `AWS.Tools` 套件，請執行以下命令。

```
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
```

若您收到通知，顯示儲存庫「不受信任」，表示系統正在詢問您是否無論如何都要進行安裝。輸入**y**以允 PowerShell 許安裝模組。若要在不信任儲存庫的情況下避免出現提示及安裝模組，您可以執行命令搭配 `-Force` 參數。

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

3. 您現在可以使用`Install-AWSToolsModule`指令程式，針對要使用的每個 AWS 服務安裝模組。例如，以下命令會安裝 Amazon EC2 和 Amazon S3 模組。此命令也會安裝指定模組運作所需的任何相依模組。例如，當您安裝第一個 `AWS.Tools` 服務模組時，它也會安裝 `AWS.Tools.Common`。這是所有 AWS 服務模組所需的共用模組。它也會移除舊版模組，並將其他模組更新至相同的更新版本。

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

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

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

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

> **Note**
>
> `Install-AWSToolsModule` Cmdlet 會從名為 `PSGallery` 的 `PSRepository` (https://www.powershellgallery.com/) 下載所有要求的模組，並視為是信任的來源。如需此 `PSRepository` 的詳細資訊，請使用 `Get-PSRepository -Name PSGallery` 命令。

根據預設，前一個命令會將模組安裝至 `%USERPROFILE%\Documents\WindowsPowerShell\Modules` 資料夾中。若要為電腦的所有使用者安裝，您必須在以系統管理員身分啟動的 PowerShell 工作階段中執行下列命令。 AWS Tools for PowerShell 例如，以下命令會將 IAM 模組安裝至所有使用者可存取的 `%ProgramFiles%\WindowsPowerShell\Modules` 資料夾。

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

要安裝其他模塊，請使用相應的模塊名稱運行類似的命令，如PowerShell 圖庫中所找到的。

安裝 AWSPowerShell。 NetCore 在視窗上

您可以安裝 AWSPowerShell. NetCore 在執行 Windows PowerShell 版本 3 到 5.1 或 PowerShell 核心 6.0 或更新版本的電腦上。如需有關如何安裝 PowerShell 核心的詳細資訊，請參閱在 Microsoft PowerShell 網站 PowerShell上安裝各種版本的。

您可以安裝 AWSPowerShell。 NetCore 以兩種方式之一

- 從下載模組AWSPowerShell。 NetCore.zip 並將其解壓縮到其中一個模塊目錄中。您可以顯示 `PSModulePath` 環境變數的值來探索自己的模組目錄。
- 使用`Install-Module`指令程式從程式 PowerShell 庫安裝，如下列程序所述。

若要安裝 AWSPowerShell。 NetCore 從 PowerShell 圖庫中使用安裝模組指令程式

若要安裝 AWSPowerShell. NetCore 在 PowerShell 圖庫中，您的計算機必須運行 PowerShell 5.0 或更高版本，或者PowerShellGet在 PowerShell 3 或更高版本上運行。執行下列命令。

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

如果您以系統管理員 PowerShell 身分執行，則先前的指令會 AWS Tools for PowerShell 安裝電腦上的所有使用者。如果您以沒有管理員權限的標準使用者身分執行 PowerShell ，則該指令只會安裝 AWS Tools for PowerShell 給目前的使用者。

若在使用者具備管理員許可時，僅要為目前的使用者進行安裝，請搭配 `-Scope CurrentUser` 參數執行命令，如下。

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

雖然 PowerShell 3.0 及更新版本通常 PowerShell 會在您第一次在模組中執行指令程式時，將模組載入到您的工作階段中 AWSPowerShell，. NetCore 模組太大，無法支援此功能。您必須明確載入 AWSPowerShell. NetCore 核心模塊通過運行以下命令進入您的 PowerShell 會話。

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

若要載入 AWSPowerShell. NetCore 模塊自動進入 PowerShell 會話，將該命令添加到您的 PowerShell 配置文件中。如需有關編輯設定 PowerShell 檔的詳細資訊，請參閱文件中的關於設定 PowerShell 檔。

AWSPowerShell 在視窗上安裝 PowerShell

您可以使用下列 AWS Tools for Windows PowerShell 兩種方式之一來安裝：

- 從 AWSPowerShell.zip 下載模塊並將其解壓縮到其中一個模塊目錄中。您可以顯示 `PSModulePath` 環境變數的值來探索自己的模組目錄。
- 使用`Install-Module`指令程式從程式 PowerShell 庫安裝，如下列程序所述。

使用安裝 PowerShell 模組指令程式 AWSPowerShell 從程式庫安裝

如果您正在運行 PowerShell 5.0 或更高版本，或者已安裝PowerShellGet在 PowerShell 3 或更高版本上，則可以 AWSPowerShell 從 PowerShell 圖庫安裝。您可以通過運行以下命令 AWSPowerShell 從微軟的PowerShell圖庫安裝和更新。

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

若要將 AWSPowerShell 模組自動載入 PowerShell 工作階段，請將先前的`import-module`指令程式新增至您的 PowerShell 設定檔。如需有關編輯設定 PowerShell 檔的詳細資訊，請參閱文件中的關於設定 PowerShell 檔。

> **Note**
>
> 默認情況下，Windows PowerShell 工具安裝在所有基於 Windows 的 Amazon 機器映像（AMI）上。

啟用指令碼執行

若要載入 AWS Tools for PowerShell 模組，您必須啟用 PowerShell 指令碼執行。欲啟用指令碼執行，請執行 `Set-ExecutionPolicy` cmdlet 來設定 `RemoteSigned` 政策。如需詳細資訊，請參閱 Microsoft Technet 網站上的 About Execution Policies。

> **Note**
>
> 此需求僅適用於執行 Windows 的電腦。`ExecutionPolicy` 安全限制不會出現在其他作業系統上。

啟用指令碼執行

1. 需要管理員權限才能設定執行政策。如果您未以具有管理員權限的使用者身分登入，請以管理員身分開啟 PowerShell 工作階段。選擇 Start (開始)，然後選擇 All Programs (所有程式)。選擇 [附件]，然後選擇 [視窗] PowerShell。在 Windows 上按一下滑鼠右鍵 PowerShell，然後在內容功能表上選擇「以系統管理員
2. 在命令提示中，輸入以下內容。

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

> **Note**
>
> 在 64 位元系統上，您必須針對 32 位元版本的 PowerShell Windows PowerShell (x86) 個別執行此動作。

如果您沒有正確設定執行原則，每當您嘗試執行指令碼 (例如設定檔) 時，都會 PowerShell 顯示下列錯誤。

```
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
```

Windows PowerShell 安裝程式的工具會自動更新 PS，ModulePath以包含包含AWSPowerShell模組的目錄位置。

由於PSModulePath包含 AWS 模組目錄的位置，所以Get-Module -ListAvailable指令程式會顯示該模組。

```
PS > Get-Module -ListAvailable

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

版本控制

AWS AWS Tools for PowerShell 定期發行新版本，以支援新的 AWS 服務和功能。若要判斷已安裝之工具的版本，請執行 G et-AWSPowerShellVersion 指令程式。

```
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]
```

您也可以將-ListServiceVersionInfo參數新增至 G et-AWSPowerShellVersion 命令，以查看目前版本工具所支援的 AWS 服務清單。若您使用模組化的 AWS.Tools.* 選項，則只會顯示您目前已匯入的模組。

```
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
...
```

若要判斷您正在執行 PowerShell 的版本，請輸入`$PSVersionTable`以檢視 $PS VersionTable 自動變數的內容。

```
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
```

AWS Tools for PowerShell 在視窗上更新

隨著更新版本的發行 AWS Tools for PowerShell ，您應該定期更新您在本機執行的版本。

更新模塊化模塊 **AWS.Tools**

若要將`AWS.Tools`模組更新為最新版本，請執行下列命令：

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

此命令會更新所有目前安裝的 `AWS.Tools` 模組，並且會在更新成功後移除其他安裝的版本。

> **Note**
>
> `Update-AWSToolsModule` Cmdlet 會從名為 `PSGallery` 的 `PSRepository` (https://www.powershellgallery.com/) 下載所有模組，並視為是信任的來源。如需此 `PSRepository` 的詳細資訊，請使用 `Get-PSRepository -Name PSGallery` 命令。

更新 PowerShell 核心工具

執行指`Get-AWSPowerShellVersion`令程式以判斷您正在執行的版本，並將其與 [PowerShell 圖庫] 網站上提供 PowerShell 的 Windows 工具版本進行比較。我們建議您每兩到三週檢查一次。只有在您更新至具有該 Support 的版本之後，才能支援新指令和 AWS 服務。

在您安裝的較新版本之前 AWSPowerShell。 NetCore，解除安裝現有的模組。在解除安裝現有套件之前，請先關閉所有開啟的 PowerShell 工作 執行下列命令以解除安裝套件。

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

解除安裝套件後，請執行以下命令來安裝更新後的模組。

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

安裝之後，`Import-Module AWSPowerShell.NetCore`請執行命令，將更新的指令程式載入您的 PowerShell 工作階段。

更新視窗的工具 PowerShell

執行指`Get-AWSPowerShellVersion`令程式以判斷您正在執行的版本，並將其與 [PowerShell 圖庫] 網站上提供 PowerShell 的 Windows 工具版本進行比較。我們建議您每兩到三週檢查一次。只有在您更新至具有該 Support 的版本之後，才能支援新指令和 AWS 服務。

- 若您使用 `Install-Module` Cmdlet 進行安裝，請執行下列命令。

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

- 若您使用下載的 ZIP 檔案安裝：

  1. 從適用於 PowerShell網站的工具下載最新版本。將所下載檔案名稱中的套件版本號碼，與您在執行 `Get-AWSPowerShellVersion` Cmdlet 時取得的版本號碼進行比較。
  2. 如果下載版本的數字高於您已安裝的版本，請關閉所有 Windows PowerShell 主控台工具。
  3. 安裝較新版本的視窗工具 PowerShell。

安裝之後，`Import-Module AWSPowerShell`請執行以將更新的指令程式載入您的 PowerShell 工作階段。或者從 [開始] 功能表執行自訂 AWS Tools for PowerShell 主控台。

在 Mac 或蘋果系統 AWS Tools for PowerShell 上安裝

本主題提供有關如何在 Linux 或 macOS AWS Tools for PowerShell 上安裝的指示。

設定概觀

若要在 Linux 或 macOS 電腦 AWS Tools for PowerShell 上安裝，您可以從兩個套件選項中進行選擇：

- **`AWS.Tools`**— 的模組化版本。 AWS Tools for PowerShell每個 AWS 服務都由自己的個人小模塊支持，並具有共享的支持模塊`AWS.Tools.Common`。
- AWSPowerShell。 NetCore— 單一、大型模組版本的 AWS Tools for PowerShell. 此單一大型模組支援所有 AWS 服務。

> **Note**
>
> 請注意，單一模組可能太大，無法與 AWS Lambda 功能一起使用。請改用上面的模組化版本。

在執行 Linux 或 macOS 的電腦上設定任何一個都會涉及下列任務，如本主題中稍後部分所說明：

1. 在支援的系統上安裝 PowerShell 核心 6.0 或更新版本。
2. 安裝 PowerShell 核心後， PowerShell 通過在系統外殼`pwsh`中運行開始。
3. 安裝`AWS.Tools`或 AWSPowerShell。 NetCore。
4. 執行適當的`Import-Module`指令程式，將模組匯入您的 PowerShell工作階段。
5. 執行初始化AWSDefaultConfiguration指令程式以提供您的 AWS 認證。

必要條件

若要執行 AWS Tools for PowerShell Core，您的電腦必須執行 PowerShell Core 6.0 或更新版本。

- 如需支援的 Linux 平台版本清單，以及如需如何在 Linux 電腦 PowerShell 上安裝最新版本的詳細資訊，請參閱在 Microsoft 的網站 PowerShell 上安裝 Linux。有些 Linux 類型作業系統 (例如 Arch、Kali 和 Raspbian) 不受支援，但擁有不同程度的社群支援。
- 如需有關支援的 macOS 版本，以及如何 PowerShell 在 macOS 上安裝最新版本的詳細資訊，請參閱在微軟網站 PowerShell 上安裝 macOS。

在 Linux 或 macOS 上安裝 **AWS.Tools**

您可以在執行 PowerShell Core 6.0 或更新版本的電腦 AWS Tools for PowerShell 上安裝模組化版本。如需有關如何安裝 PowerShell 核心的詳細資訊，請參閱在 Microsoft PowerShell 網站 PowerShell 上安裝各種版本的。

有三種方式可以安裝 AWS.Tools：

- 使用 AWS.Tools.Installer 模組中的 Cmdlet。該模塊簡化了其他模AWS.Tools塊的安裝和更新。 AWS.Tools.Installer需要PowerShellGet，並自動下載並安裝它的更新版本。AWS.Tools.Installer自動保持您的模塊版本同步。當您安裝或更新至某個模組的較新版本時，中的指令程式AWS.Tools.Installer會自動將所有其他AWS.Tools模組更新為相同版本。

  此方法會在下列程序中說明。
- 從 AWS.Tools.zip 下載模組，然後解壓縮至其中一個模組目錄。您可以列印 $Env:PSModulePath 變數的值來探索您的模組目錄。
- 使用Install-Module指令程式從程式 PowerShell 庫安裝每個服務模組。

若要使用該**AWS.Tools.Installer**模組**AWS.Tools**在 Linux 或 macOS 上安裝

1. 執行下列命令以啟動 PowerShell 核心工作階段。

   ```
   $ pwsh
   ```

   > **Note**
   >
   > 我們建議您不要 PowerShell 以具有提升權限的系統管理員身分執行，除非手頭的工作需要。這是因為可能會有潛在的安全風險，且不符合最低權限原則。

2. 若要使用 AWS.Tools.Installer 安裝模組化 AWS.Tools 套件，請執行以下命令。

   ```
   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
```

若您收到通知，顯示儲存庫「不受信任」，表示系統正在詢問您是否無論如何都要進行安裝。輸入**y**以允 PowerShell 許安裝模組。若要在不信任儲存庫的情況下避免出現提示及安裝模組，您可以執行以下命令。

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

3. 您現在可以安裝每個您希望使用服務的模組。例如，以下命令會安裝 Amazon EC2 和 Amazon S3 模組。此命令也會安裝指定模組運作所需的任何相依模組。例如，當您安裝第一個 AWS.Tools 服務模組時，它也會安裝 AWS.Tools.Common。這是所有 AWS 服務模組所需的共用模組。它也會移除舊版模組，並將其他模組更新至相同的更新版本。

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

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

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

> **Note**
>
> Install-AWSToolsModule Cmdlet 會從名為 PSGallery 的 PSRepository (https://www.powershellgallery.com/) 下載所有要求的模組，並將儲存庫視為信任的來源。如需此 PSRepository 的詳細資訊，請使用 Get-PSRepository -Name PSGallery 命令。

上一個命令會將模組安裝到系統上的預設目錄中。實際的目錄取決於您的作業系統發行版本和版本，以及 PowerShell 您安裝的版本。例如，如果您在類似 RHEL 的系統上安裝了 PowerShell 7，則預設模組很可能位於 `/opt/microsoft/powershell/7/Modules` (或`$PSHOME/Modules`) 中，而使用者模組很可能位於中。`~/.local/share/powershell/Modules`如需詳細資訊，請參閱 PowerShell 在 Linux 上安裝 Microsoft PowerShell 網站。若要查看安裝模組的位置，請執行下列命令：

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

要安裝其他模塊，請使用相應的模塊名稱運行類似的命令，如PowerShell 圖庫中所示。

安裝 AWSPowerShell。 NetCore 在 Linux 或 macOS 系統上

若要升級到更新版本的 AWSPowerShell. NetCore，請遵循中的指示AWS Tools for PowerShell 在 Linux 或 macOS 系統上更新。解除安裝舊版的 AWSPowerShell. NetCore 第一。

您可以安裝 AWSPowerShell。 NetCore 以兩種方式之一：

- 從 `AWSPowerShell.NetCore.zip` 下載模組，然後解壓縮至其中一個模組目錄。您可以列印 `$Env:PSModulePath` 變數的值來探索您的模組目錄。
- 使用`Install-Module`指令程式從程式 PowerShell 庫安裝，如下列程序所述。

要安裝 AWSPowerShell。 NetCore 在 Linux 或 macOS 上使用安裝模組指令程式

執行下列命令以啟動 PowerShell 核心工作階段。

```
$ pwsh
```

> **Note**
>
> 我們建議您不要以更高 PowerShell 的系統管理員權限執行`sudo pwsh`來執行 PowerShell 。這是因為可能會有潛在的安全風險，且不符合最低權限原則。

若要安裝 AWSPowerShell. NetCore 來自 PowerShell 圖庫的單模塊包，運行以下命令。

```
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
```

若您收到通知，顯示儲存庫「不受信任」，表示系統正在詢問您是否無論如何都要進行安裝。輸入**y**以允 PowerShell 許安裝模組。若要在不信任儲存庫的情況下避免出現提示，您可以執行以下命令。

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

您不必以 root 身份運行此命令，除非您想為計算機 AWS Tools for PowerShell 的所有用戶安裝。若要這麼做，請在您開始的 PowerShell 工作階段中執行下列命令`sudo pwsh`。

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

指令碼執行

`Set-ExecutionPolicy` 命令不適用於非 Windows 系統。您可以執行`Get-ExecutionPolicy`，顯示在非 Windows 系統上執行的 PowerShell Core 中的預設執行原則設定為`Unrestricted`。如需詳細資訊，請參閱 Microsoft Technet 網站上的 About Execution Policies。

由於`PSModulePath`包含 AWS 模組目錄的位置，所以`Get-Module -ListAvailable`指令程式會顯示您安裝的模組。

`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
```

設定 PowerShell 主控台以使用 AWS Tools for PowerShell Core (AWSPowerShell. NetCore 只有)

PowerShell 每當您在模組中執行指令程式時，Core 通常都會自動載入模組。但是這不起作用 AWSPowerShell。 NetCore 因為它的大尺寸。開始跑步 AWSPowerShell。 NetCore 指令程式時，您必須先執行命`Import-Module AWSPowerShell.NetCore`令。`AWS.Tools` 模組中的 Cmdlet 不需要此操作。

初始化您的 PowerShell 會話

當您 PowerShell 在安裝完 Linux 或基於 Mac 的系統上啟動時 AWS Tools for PowerShell，您必須執行初始化-AWSDefaultConfiguration 來指定要使用的 AWS 存取金鑰。如需 `Initialize-AWSDefaultConfiguration` 的相關資訊，請參閱 使用 AWS 憑證。

> **Note**
>
> 在舊版 (3.3.96.0 之前) 的版本中 AWS Tools for PowerShell，此指令程式已命名為。`Initialize-AWSDefaults`

版本控制

AWS AWS Tools for PowerShell 定期發行新版本以支援新 AWS 服務和功能。若要判斷已安裝 AWS Tools for PowerShell 的版本，請執行 G et-AWSPowerShellVersion 指令程式。

```
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]
```

若要查看目前版本工具中支援的 AWS 服務清單，請將`-ListServiceVersionInfo`參數新增至 Get-AWSPowerShellVersion 指令程式。

若要判斷您正在執行 PowerShell 的版本，請輸入`$PSVersionTable`以檢視`$PSVersionTable`自動變數的內容。

```
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
```

AWS Tools for PowerShell 在 Linux 或 macOS 系統上更新

隨著更新版本的發行 AWS Tools for PowerShell ，您應該定期更新您在本機執行的版本。

更新模塊化模塊 **AWS.Tools**

若要將`AWS.Tools`模組更新為最新版本，請執行下列命令：

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

此命令會更新所有目前安裝的 AWS.Tools 模組，而對於已成功更新的模組，則會移除舊版。

> **Note**
>
> Update-AWSToolsModule Cmdlet 會從名為 PSGallery 的 PSRepository (https://www.powershellgallery.com/) 下載所有模組，並視為是信任的來源。如需此 PSRepository 的詳細資訊，請使用 Get-PSRepository -Name PSGallery 命令。

更新 PowerShell 核心工具

執行指Get-AWSPowerShellVersion令程式以判斷您正在執行的版本，並將其與 [PowerShell 圖庫] 網站上提供 PowerShell 的 Windows 工具版本進行比較。我們建議您每兩到三週檢查一次。只有在您更新至具有該 Support 的版本之後，才能支援新指令和 AWS 服務。

在您安裝較新版本的 AWSPowerShell. NetCore，解除安裝現有模組。在解除安裝現有套件之前，請先關閉所有開啟的 PowerShell 工作 執行下列命令以解除安裝套件。

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

解除安裝套件後，請執行以下命令來安裝更新後的模組。

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

安裝之後，執行命令Import-Module AWSPowerShell.NetCore，將更新的指令程式載入您的 PowerShell 工作階段。

相關資訊

- 開始使用 AWS Tools for Windows PowerShell。
- 在 AWS Tools for PowerShell 中使用 AWS 服務

從 AWS Tools for PowerShell 3.3 版遷移至第 4 版

AWS Tools for PowerShell 第 4 版是與 AWS Tools for PowerShell 3.3 版回溯相容的更新。此版本新增了大幅改善的項目，同時保持現有的 Cmdlet 行為。

在升級至新版本後，現有指令碼應該能繼續運作，但建議您在升級生產環境前先進行徹底的測試。

本節說明這些變更並解說這些變更可能會對指令碼有何影響。

新全面模組化 **AWS.Tools** 版本

的 AWSPowerShell. NetCore 和 AWSPowerShell 軟件包是「整體」。這表示所有 AWS 服務都是由同一個模組提供支援，模組會變得很大，而且會隨著新增每項 AWS 服務和功能而變得更大。全新 AWS.Tools 套件會分為多個較小的模組，能讓您彈性選擇下載及安裝您使用之 AWS 服務所需的項目。此套件包含一個所有其他模組需要的共用 AWS.Tools.Common 模組，以及一個可視需要簡化模組安裝、更新及移除作業的 AWS.Tools.Installer 模組。

這也會在第一次呼叫時啟用 Cmdlet 的自動匯入功能，而不必先呼叫 Import-module。不過，若要在呼叫指令程式之前與相關聯的 .NET 物件互動，您仍然必須呼叫Import-Module以告 PowerShell 知相關的 .NET 類型。

例如，以下命令句有 Amazon.EC2.Model.Filter 的參照。這種類型的參照無法觸發自動匯入功能，因此您必須先呼叫 Import-Module，否則命令則會失敗。

```
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
```

全新 **Get-AWSService** Cmdlet

您可以使用 Get-AWSService Cmdlet，協助您探索 AWS.Tools 模組集合中每項 AWS 服務的模組名稱。

```
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
  ...
```

控制 Cmdlet 傳回之物件的全新 -Select 參數

第 4 版中的大多數 Cmdlet 可支援全新的 `-Select` 參數。每個 Cmdlet 都會使用 AWS SDK for .NET 呼叫 AWS 服務 API。然後AWS Tools for PowerShell客戶端將響應轉換為一個對象，您可以在 PowerShell 腳本中使用該對象，並將其管道轉換為其他命令。有時，final PowerShell 對像在原始響應中具有比您需要的更多字段或屬性，而其他時候您可能希望對象包含默認情況下不存在的響應的字段或屬性。`-Select` 參數可讓您指定 Cmdlet 所傳回之 .NET 物件中要包含的項目。

例如，Get-S3Object 指令程式會叫用 Amazon S3 開發套件作業。ListObjects該操作返回一個ListObjectsResponse對象。不過，根據預設，指`Get-S3Object`令程式只會傳回 SDK 回應的`S3Objects`元素給 PowerShell 使用者。在以下範例中，該物件是具有兩個元素的陣列。

```
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
```

在 AWS Tools for PowerShell 第 4 版中，您可以指定 `-Select *` 傳回開發套件 API 呼叫所傳回的完整 .NET 回應物件。

```
PS > Get-S3Object -BucketName mybucket -Select *
  IsTruncated    : False
  NextMarker     :
```

```
  S3Objects      : {file1.txt, file2.txt}
  Name           : mybucket
  Prefix         :
  MaxKeys        : 1000
  CommonPrefixes : {}
  Delimiter      :
```

您也可以指定您想要的特定巢狀屬性的路徑。以下範例只會傳回 S3Objects 陣列中每個元素的 Key 屬性。

```
PS > Get-S3Object -BucketName mybucket -Select S3Objects.Key
file1.txt
file2.txt
```

在某些情況下，傳回 Cmdlet 參數可能非常有用。您可以使用 -Select ^ParameterName 來達成此操作。此功能取代了 -PassThru 參數，該參數仍可取得但已遭取代。

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

每個 Cmdlet 的參考主題會識別是否支援 -Select 參數。

以更一致的方式限制輸出的項目數

舊版 AWS Tools for PowerShell 可讓您使用 -MaxItems 參數指定最終輸出中傳回的項目數上限。

AWS.Tools 已移除此行為。

此行為已在中取代 AWSPowerShell。 NetCore 和 AWSPowerShell，並將在 future 的發行版本中從這些版本中移除。

如果基礎服務 API 支援 MaxItems 參數，則仍可依 API 指定取得及運作，但不會再有限制 Cmdlet 輸出中傳回之項目數的新增行為。

若要限制最終輸出中傳回的項目數，請將輸出輸送至 Select-Object Cmdlet 並指定 -First *n* 參數，其中 *n* 是要在最終輸出中包含的項目數上限。

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

```
file1.txt
file2.txt
```

以往並非所有 AWS 服務皆以同樣的方式支援 `-MaxItems`，因此這移除了該不一致的情況，以及有時會發生的非預期結果。此外結合新 `-Select` 參數的 `-MaxItems` 有時可能會導致令人混淆的結果。

更易於使用的串流參數

`Stream` 或 `byte[]` 類型的參數現在可以接受 `string`、`string[]` 或 `FileInfo` 值。

例如，您可以使用以下任一範例。

```
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 會使用 UTF-8 編碼將所有字串轉換為 `byte[]`。

依屬性名稱擴充管道

為了提升使用者體驗的一致性，您現在可以指定「任何」參數的屬性名稱，藉此傳遞管道輸入。

在以下範例中，我們建立了一個自訂物件，其屬性具有符合目標 Cmdlet 參數名稱的名稱。當 Cmdlet 執行時，它會自動使用這些屬性做為其參數。

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

> **Note**
>
> 在舊版 AWS Tools for PowerShell 中，有些屬性可支援此體驗。第 4 版可讓您指定「所有」參數，讓此體驗更加一致。

靜態常見參數

為提升 AWS Tools for PowerShell 4.0 版的一致性，所有參數皆為靜態。

在舊版 AWS Tools for PowerShell 中，某些常見參數 (例如 AccessKey、SecretKey、ProfileName 或 Region) 為動態，而所有其他參數則為靜態。這可能會產生問題，因為在動態參數之前 PowerShell 綁定靜態參數。例如，假設您執行以下命令。

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

早期版本的us-west-2將值 PowerShell 繫結至-RegionName靜態參數，而不是動-Region態參數。這可能會讓使用者產生混淆。

AWS.Tools 會宣告並強制執行必要參數

AWS.Tools.* 模組現在會宣告並強制執行必要的 Cmdlet 參數。當AWS服務宣告需要 API 的參數時，如果您未指定對應的 cmdlet 參數，則 PowerShell 會提示您輸入對應的 cmdlet 參數。這僅適用於 AWS.Tools。為了確保回溯相容性，這不適用於 AWSPowerShell。 NetCore 或 AWSPowerShell。

所有參數皆可為 Null

您現在可以將 $null 指派給值類型參數 (數字和日期)。此變更應該不會影響現有的指令碼。這可讓您略過必要參數的提示。必要參數只會在 AWS.Tools 中強制執行。

如果您使用第 4 版執行以下範例，則會有效略過用戶端驗證，因為您會為每個必要參數提供「值」。不過，Amazon EC2 API 服務呼叫會失敗，因為 AWS 服務仍需要該資訊。

```
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
```

移除先前取代的功能

舊版 AWS Tools for PowerShell 已取代以下功能，而第 4 版也取代了這些功能：

- 移除 `Stop-EC2Instance` Cmdlet 中的 `-Terminate` 參數。請改用 `Remove-EC2Instance`。
- 已從清除AWSCredential 指令程式中移除`-ProfileName`參數。請改用 `Remove-AWSCredentialProfile`。
- 移除 `Import-EC2Instance` 和 `Import-EC2Volume` Cmdlet。

開始使用 AWS Tools for Windows PowerShell。

本節中的部分主題說明建立工具之後，使用 Tools for Windows PowerShell 的基本原理。例如，這些主題會說明如何指定 Tools for Windows PowerShell 與 AWS 互動時應使用的憑證和 AWS 區域。

本節中的其他主題提供了一些進階方法的資訊，您可以使用這些方法來設定工具、環境和專案。

主題

- 設定工具驗證 AWS
- 指定 AWS 區域
- 使用 AWS Tools for PowerShell 設定聯合身分
- Cmdlet 探索和別名
- 管道傳輸和 $AWSHistory
- 憑證和設定檔解析
- 有關使用者和角色的其他資訊
- 使用舊版憑證

設定工具驗證 AWS

在開發 AWS 時，您必須建立程式碼的驗證方式。 AWS 服務您可以透過不同的方式設定 AWS 資源的程式設計存取，具體取決於環境和您可用的 AWS 存取權限。

要查看工具的各種身份驗證方法 PowerShell，請參閱 AWS SDK 和工具參考指南中的身份驗證和訪問。

本主題假設新使用者正在本機進行開發，並未提供雇主進行驗證的方法，而且將用 AWS IAM Identity Center 來取得臨時認證。如果您的環境不適用這些假設，則本主題中的有些資訊可能不適用您的環境，或者有些資訊可能已經提供給您。

設定此環境需要幾個步驟，總結如下：

1. 啟用和設定 IAM Identity Center
2. 設定 PowerShell 要使用 IAM 身分中心的工具。
3. 啟動 AWS 存取入口網站會話

啟用和設定 IAM Identity Center

若要使用 AWS IAM Identity Center，必須先啟用並設定它。要查看有關如何執行此操作的詳細信息 PowerShell，請查看 AWS SDK 和工具參考指南中 IAM 身份中心身份驗證主題中的步驟 1。具體來說，請遵循我沒有透過 IAM Identity Center 建立存取權限下的任何必要說明。

設定 PowerShell 要使用 IAM 身分中心的工具。

> **Note**
>
> 從工具的 4.1.538 版開始 PowerShell，設定 SSO 認證和啟動 AWS 存取入口網站工作階段的建議方法是使用`Initialize-AWSSSOConfiguration`和`Invoke-AWSSSOLogin`指令程式，如本主題所述。如果您無法存取 PowerShell (或更新版本) 的工具版本，或無法使用這些指令程式，您仍然可以使用 AWS CLI. 要了解如何操作，請參閱使用入口 AWS CLI 網站登入。

下列程序會使用 SSO 資訊來更新共用 AWS `config`檔案，這些資訊可 PowerShell 用來取得暫時認證的工具。由於此程序，也會啟動 AWS 存取入口網站工作階段。如果共用`config`檔案已有 SSO 資訊，而您只想知道如何使用的工具啟動存取入口網站工作階段 PowerShell，請參閱本主題的下一節啟動 AWS 存取入口網站會話。

1. 如果您尚未這麼做，請開啟 PowerShell 並安裝適合您作業系統和環境的，包括一般指令程式。 AWS Tools for PowerShell 如需如何進行該服務的詳細資訊，請參閱安裝 AWS Tools for PowerShell。

   例如，如果 PowerShell 在 Windows 上安裝「工具」的模組化版本，您很可能會執行類似下列的命令：

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

2. 執行下列命令。將範例屬性值取代為 IAM 身分中心組態中的值。有關這些屬性以及如何找到它們的詳細資訊，請參閱 AWS SDK 和工具參考指南中的 IAM 身分中心憑證提供者設定。

   ```
   $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
```

或者，您也可以單獨使用指令程式`Initialize-AWSSSOConfiguration`，以及「工具」來提 PowerShell 示您輸入內容值。

特定屬性值的考量：

- 如果您只是按照指示啟用和設定 IAM 身分中心，則的值`-RoleName`可能是`PowerUserAccess`。但是，如果您建立了專為 PowerShell 工作設定的 IAM 身分中心權限，請改用該權限。
- 請務必使用已設定 IAM 身分中心的 AWS 區域 位置。

3. 此時，共用 AWS `config`檔案包含一個名為的設定檔，其中包`my-sso-profile`含一組可從的「工具」參考的組態值 PowerShell。若要尋找此檔案的位置，請參閱 AWS SDK 和工具參考指南中的共用檔案位置。

   在傳送要求之前， PowerShell 使用設定檔的 SSO 權杖提供者取得認證的工具 AWS。這個`sso_role_name`值是連接到 IAM 身分中心權限集的 IAM 角色，應該允許存取應用程式中 AWS 服務 使用的角色。

   下列範例顯示使用上述指令建立的設定檔。在您的實際配置文件中，某些屬性值及其順序可能會有所不同。設定檔的`sso-session`屬性是指名為的區段`my-sso-session`，其中包含啟動 AWS 存取入口網站工作階段的設定。

   ```
   [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. 如果您已經擁有使用中的 AWS 存取入口網站工作階段，用於通 PowerShell 知您已登入的工具。

如果不是這種情況，則 PowerShell 嘗試在默認 Web 瀏覽器中自動打開 SSO 授權頁面的工具。遵循瀏覽器中的提示，其中可能包括 SSO 授權碼、使用者名稱和密碼，以及存取 AWS IAM Identity Center 帳戶和權限集的權限。

PowerShell 通知您 SSO 登入成功的工具。

啟動 AWS 存取入口網站會話

在執行存取的命令之前 AWS 服務，您需要使用中存 AWS 取入口網站工作階段， PowerShell 以便使用 IAM 身分中心驗證來解析登入資料的工具。若要登入 AWS 存取入口網站，請在中執行下列命令 PowerShell，其中`-ProfileName my-sso-profile`是當您遵循本主題前一節中的程序時，在共用`config`檔案中建立的設定檔名稱。

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

如果您已經擁有使用中的 AWS 存取入口網站工作階段，用於通 PowerShell 知您已登入的工具。

如果不是這種情況，則 PowerShell 嘗試在默認 Web 瀏覽器中自動打開 SSO 授權頁面的工具。遵循瀏覽器中的提示，其中可能包括 SSO 授權碼、使用者名稱和密碼，以及存取 AWS IAM Identity Center 帳戶和權限集的權限。

PowerShell 通知您 SSO 登入成功的工具。

若要測試您是否已有作用中的工作階段，請依需要在安裝或匯入`AWS.Tools.SecurityToken`模組之後執行下列命令。

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

對`Get-STSCallerIdentity`指令程式的回應會報告共用`config`檔案中設定的 IAM 身分中心帳戶和權限集。

範例

以下是如何將 IAM 身分中心與的工具搭配使用的範例 PowerShell。假設如下：

- 您已啟用 IAM Identity Center，並依照本主題先前所述進行設定。SSO 內容位於本主題稍早設定的設定`my-sso-profile`檔中。

- 當您透過Initialize-AWSSSOConfiguration或Invoke-AWSSSOLogin指令程式登入時，使用者至少擁有 Amazon S3 的唯讀許可。
- 部分 S3 儲存貯體可供該使用者檢視。

視需要安裝或匯入AWS.Tools.S3模組，然後使用下列 PowerShell 命令顯示 S3 儲存貯體的清單。

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

其他資訊

- 有關「工具」身份驗證的更多選項 PowerShell，例如使用配置文件和環境變量，請參閱 AWS SDK 和工具參考指南中的配置一章。
- 某些指令需要指定「 AWS 區域」。有許多方法可以這樣做，包括 -Region cmdlet 選項、[default]設定檔和AWS_REGION環境變數。如需詳細資訊，請參閱本指南指定 AWS 區域中的以及 AWS SDK 和工具參考指南中的AWS 區域。
- 如需了解有關最佳實務的資訊，請參閱 IAM 使用者指南中的 IAM 安全最佳實務。
- 若要建立短期 AWS 登入資料，請參閱 IAM 使用者指南中的臨時安全登入資料。
- 若要瞭解其他憑證提供者，請參閱 AWS SDK 和工具參考指南中的標準化憑證提供者。

主題

- 使用入口 AWS CLI 網站登入

使用入口 AWS CLI 網站登入

從工具的 4.1.538 版開始 PowerShell，設定 SSO 認證和啟動 AWS 存取入口網站工作階段的建議方法是使用Initialize-AWSSSOConfiguration和Invoke-AWSSSOLogin指令程式，如中所述。設定工具驗證 AWS如果您無法存取 PowerShell (或更新版本) 的工具版本，或無法使用這些指令程式，您仍然可以使用 AWS CLI.

透過設定 PowerShell 要使用 IAM 身分中心的工具 AWS CLI。

如果您尚未這麼做，請務必在繼續之前啟用和設定 IAM 身分中心。

有關如何設定 PowerShell 要使用 IAM 身分中心工具的資訊，請參閱 AWS SDK 和工具參考指南中 IAM 身分中心身分中心主題的步驟 2。 AWS CLI 完成此組態之後，您的系統應該包含下列元素：

- 您可以在 AWS CLI執行應用程式之前啟動 AWS 存取入口網站工作階段。
- 包含設定 AWSconfig檔的共用檔案，其中包含可從的「工具」參考的一組組態值 PowerShell。[default]若要尋找此檔案的位置，請參閱 AWS SDK 和工具參考指南中的共用檔案位置。在傳送要求之前， PowerShell 使用設定檔的 SSO 權杖提供者取得認證的工具 AWS。這個sso_role_name值是連接到 IAM 身分中心權限集的 IAM 角色，應該允許存取應用程式中 AWS 服務 使用的角色。

  下列範例config檔案顯示使用 [default] SSO 權杖提供者設定的設定檔。設定檔的 sso_session 設定是指已命名的 sso-session 區段。此sso-session區段包含用來啟動 AWS 存取入口網站工作階段的設定。

```
[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**
>
> 您的工作 PowerShell 階段必須安裝並匯入下列模組，以便 SSO 解析能夠運作：
>
> - AWS.Tools.SSO
> - AWS.Tools.SSOOIDC
>
> 如果您使用的是較舊版本的工具， PowerShell 並且沒有這些模塊，則會出現類似以下內容的錯誤：「找不到彙編 AWSSDK .SSOIDC...」。

啟動 AWS 存取入口網站會話

在執行存取的命令之前 AWS 服務，您需要使用中存 AWS 取入口網站工作階段，以便 Windows 適用的工具 PowerShell 可以使用 IAM 身分中心驗證來解析登入資料。根據您設定的工作階段長度，您的存

取最終會過期，而 Windows 的工具 PowerShell 會遇到驗證錯誤。若要登入 AWS 存取入口網站，請在中執行下列命令 AWS CLI。

```
aws sso login
```

由於您使用的是[default]設定檔，因此您不需要使用此--profile選項呼叫命令。如果您的 SSO 權杖提供者組態使用具名的設定檔，則aws sso login --profile *named-profile*改為此命令。有關命名配置文件的更多信息，請參閱 AWS SDK 和工具參考指南中的配置文件部分。

若要測試您是否已經有作用中的工作階段，請執行下列 AWS CLI 命令 (同樣考量具名的設定檔)：

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

對此命令的回應，應報告共用 config 檔案中設定的 IAM Identity Center 帳戶和許可集合。

> **Note**
>
> 如果您已經擁有作用中的 AWS 存取入口網站工作階段並執行aws sso login，則不需要提供認證。
>
> 登入程序可能會提示您允許 AWS CLI 存取您的資料。由於 AWS CLI 是建置在適用於 Python 的 SDK 之上，因此權限訊息可能包含botocore名稱的變體。

範例

以下是如何將 IAM 身分中心與的工具搭配使用的範例 PowerShell。假設如下：

- 您已啟用 IAM Identity Center，並依照本主題先前所述進行設定。SSO 屬性位於 [default] 設定檔中。
- 當您透過使用登入時aws sso login，該使用者至少擁有 Amazon S3 的唯讀許可。 AWS CLI
- 部分 S3 儲存貯體可供該使用者檢視。

使用下列 PowerShell 命令來顯示 S3 儲存貯體的清單：

```
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
```

如上所述，由於您正在使用[default]設定檔，因此不需要使用此-ProfileName選項呼叫Get-S3Bucket指令程式。如果您的 SSO 權杖提供者組態使用已命名的設定檔，則命令為 Get-S3Bucket -ProfileName *named-profile*。有關命名配置文件的更多信息，請參閱 AWS SDK 和工具參考指南中的配置文件部分。

其他資訊

- 有關「工具」身份驗證的更多選項 PowerShell，例如使用配置文件和環境變量，請參閱 AWS SDK 和工具參考指南中的配置一章。
- 某些指令需要指定「 AWS 區域」。有許多方法可以這樣做，包括 -Region cmdlet 選項、[default]設定檔和AWS_REGION環境變數。如需詳細資訊，請參閱本指南指定 AWS 區域中的以及 AWS SDK 和工具參考指南中的AWS 區域。
- 如需了解有關最佳實務的資訊，請參閱 IAM 使用者指南中的 IAM 安全最佳實務。
- 若要建立短期 AWS 登入資料，請參閱 IAM 使用者指南中的臨時安全登入資料。
- 若要瞭解其他憑證提供者，請參閱 AWS SDK 和工具參考指南中的標準化憑證提供者。

指定 AWS 區域

有兩種方法可以指定執行指 AWS Tools for PowerShell 令時要使用的「 AWS 區域」：

- 在個別命令上使用 -Region 通用參數。
- 使用 Set-DefaultAWSRegion 命令來設定所有命令的預設區域。

如果 Windows PowerShell 的工具無法找出要使用的區域，許多 AWS 指令程式會失敗。例外情況包括適用於 Amazon S3、Amazon SES 的指令程式，以及 AWS Identity and Access Management自動預設為全域端點的指令程式。

為單一指令指定區域的步 AWS 驟

將 `-Region` 參數新增到您的命令，如下。

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

為目前階段作業中的所有 AWS CLI 命令設定預設區域

在 PowerShell 命令提示字元中，輸入下列命令。

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

> **Note**
>
> 此設定僅會存在於目前的工作階段。若要將設定套用至所有 PowerShell工作階段，請將此命令新增至您的設定 PowerShell 檔，如同您為`Import-Module`指令所做的一樣。

檢視所有 AWS CLI 命令的目前預設區域

在 PowerShell 命令提示字元中，輸入下列命令。

```
PS > Get-DefaultAWSRegion

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

若要清除所有 AWS CLI 命令的目前預設區域

在 PowerShell 命令提示字元中，輸入下列命令。

```
PS > Clear-DefaultAWSRegion
```

檢視所有可用 AWS 區域的清單

在 PowerShell 命令提示字元中，輸入下列命令。範例輸出的第三個欄位會識別出目前工作階段的預設區域。

```
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**
>
> 系統可能會支援部分區域，但不會在 Get-AWSRegion Cmdlet 的輸出中包含該區域。例如，這有時對還不是全域的區域是如此。如果新增 -Region 參數至命令仍無法指定區域，請改為嘗試指定自訂端點中的區域，如下節所示。

指定自訂或非標準端點

以下列範例格式將一-EndpointUrl般參數新增至 Windows 適用的工具 PowerShell 命令，將自訂端點指定為 URL。

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

以下範例是採用 Get-EC2Instance cmdlet。在本範例中，自訂端點位於 us-west-2，亦稱美國西部 (奧勒岡)，但您可以使用任何其他支援的 AWS 區域 (包括 Get-AWSRegion 未列舉的區域)。

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

其他資訊

如需有關AWS 區 AWS域的其他資訊，請參閱 AWS SDK 和工具參考指南中的區域。

使用 AWS Tools for PowerShell 設定聯合身分

為了讓您組織中的使用者可以存取 AWS 資源，您必須基於安全性、稽核性、合規性以及可支援角色和帳戶區隔的功能，來設定標準的可重複身分驗證方法。儘管讓使用者能夠存取 AWS API 很常見，但若不使用聯合 API 存取，您還必須建立 AWS Identity and Access Management (IAM) 使用者，而這違背了使用聯合的目的。本主題說明 AWS Tools for PowerShell 中的 SAML (安全性聲明標記語言) 支援，可簡化您的聯合存取解決方案。

AWS Tools for PowerShell 中的 SAML 支援可讓您為使用者提供 AWS 服務的聯合身分存取。SAML 是一種 XML 開放標準格式，可在服務之間傳輸使用者身分驗證和授權資料；特別是在身分提供者 (如 Active Directory Federation Services) 和服務供應商 (如 AWS) 之間。如需有關 SAML 及其運作方式的詳細資訊，請參閱 Wikipedia 上的 SAML，或「結構化資訊標準促進組織」(OASIS) 網站上的 SAML 技術規格。AWS Tools for PowerShell 中的 SAML 支援與 SAML 2.0 相容。

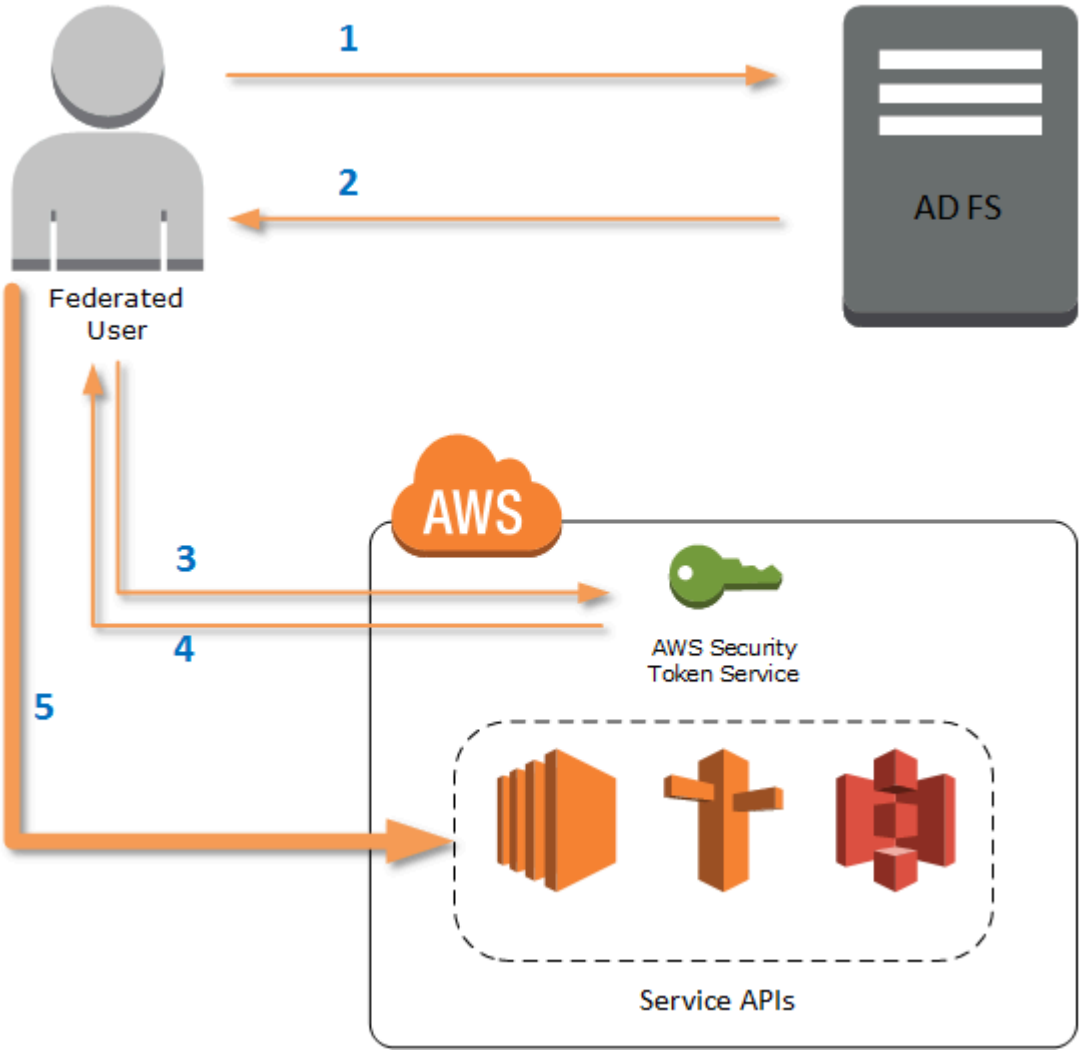
先決條件

您必須具備以下項目，才能首次嘗試使用 SAML 支援。

- 與您的 AWS 帳戶正確整合的聯合身分解決方案，可以僅使用您的組織登入資料進行主控台存取。如需如何專門針對 Active Directory Federation Services 執行此操作的詳細資訊，請參閱 IAM 使用者指南中的關於 SAML 2.0 聯合身分，以及部落格文章：使用 Windows Active Directory、AD FS 和 SAML 2.0 啟用聯合身分到 AWS。雖然部落格文章說明的是 AD FS 2.0，但 AD FS 3.0 的步驟也類似。
- 本機工作站上安裝的是 AWS Tools for PowerShell 3.1.31.0 版或更新版本。

聯合身分使用者如何取得 AWS 服務 API 的存取權

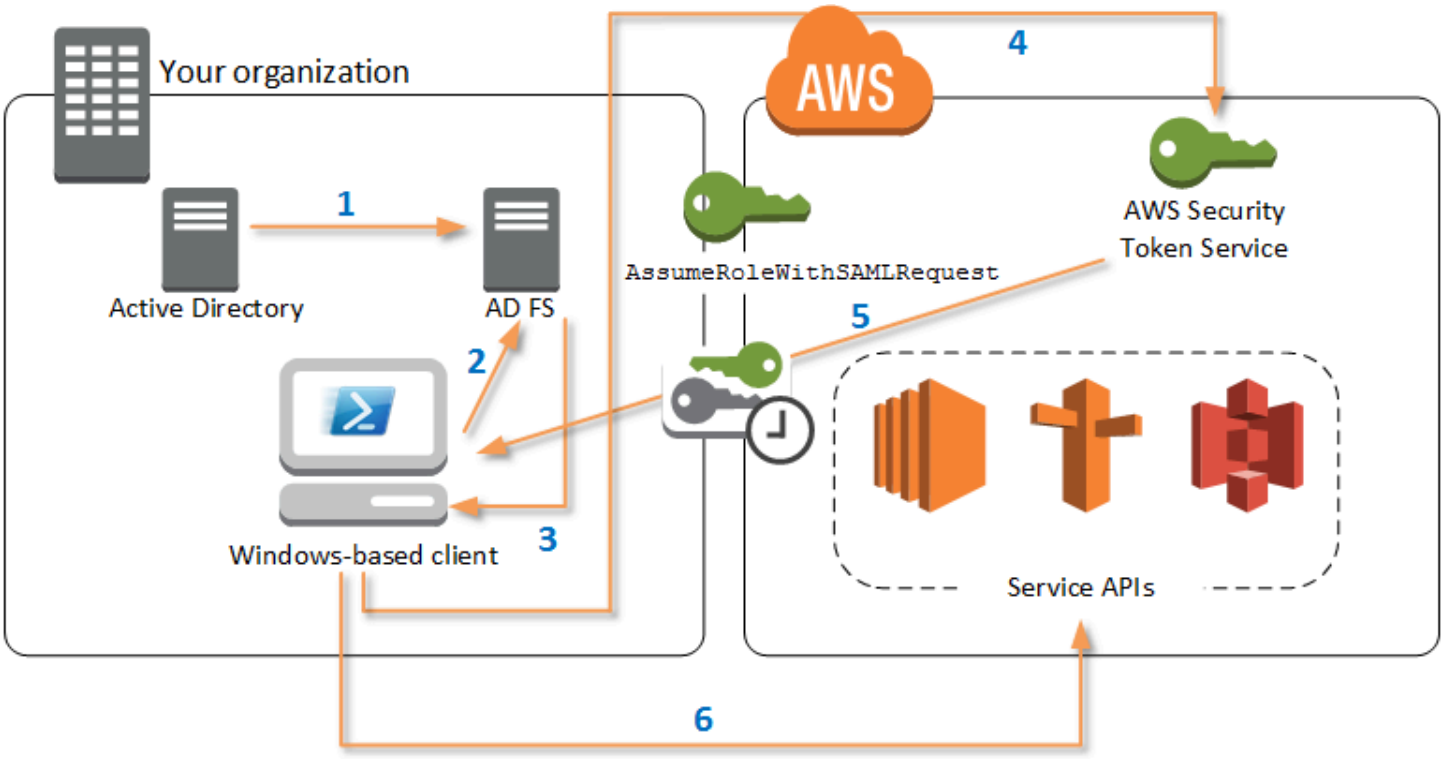
以下程序大致描述 Active Directory (AD) 使用者如何由 AD FS 聯合以獲得 AWS 資源的存取權。



1. 聯合身分使用者電腦上的用戶端會驗證 AD FS。
2. 如果驗證成功，AD FS 會向使用者傳送 SAML 聲明。
3. 使用者的用戶端會將 SAML 聲明傳送至 AWS Security Token Service (STS) 做為 SAML 同盟請求的一部分。
4. STS 會傳回一個 SAML 回應，其中包含使用者可擔任角色的 AWS 暫時登入資料。
5. 使用者透過在 AWS Tools for PowerShell 提出的請求加入這些暫時登入資料，來存取 AWS 服務 API。

SAML 支援在 AWS Tools for PowerShell 中的運作方式

本節說明 AWS Tools for PowerShell cmdlet 如何啟用使用者的 SAML 聯合身分組態。



1. AWS Tools for PowerShell 會使用 Windows 使用者目前的登入資料來對 AD FS 驗證身分，或在使用者嘗試執行需要登入資料以呼叫 AWS 的 cmdlet 時以互動方式進行。
2. AD FS 驗證使用者。
3. AD FS 會產生包含聲明的 SAML 2.0 驗證回應，此聲明的目的在於識別並提供使用者資訊。AWS Tools for PowerShell 會從 SAML 聲明中擷取使用者授權角色的清單。
4. AWS Tools for PowerShell 會發起 `AssumeRoleWithSAMLRequest` API 呼叫，將 SAML 請求 (包括所請求角色的 Amazon Resource Name (ARN)) 轉送到 STS。
5. 如果 SAML 請求是有效的，STS 會傳回回應，其中包含 AWS、`AccessKeyId`、`SecretAccessKey` 和 `SessionToken`。這些登入資料可持續 3,600 秒 (1 小時)。
6. 使用者現在擁有有效的登入資料，可使用使用者角色獲授權存取的任何 AWS 服務 API。AWS Tools for PowerShell 會在任何後續 AWS API 呼叫中自動套入這些登入資料，並在它們到期時自動予以更新。

> **Note**
>
> 當憑證過期且需要新的登入資料時，AWS Tools for PowerShell 會使用 AD FS 自動重新驗證，並取得下個小時的新登入資料。對於已加入網域的使用者帳戶，此程序會以無提示的方

式進行。針對未加入網域的帳戶，AWS Tools for PowerShell 會提示使用者先輸入他們的登入資料，再重新驗證。

如何使用 PowerShell SAML 組態 Cmdlet

AWS Tools for PowerShell 包含兩個新的 cmdlet，可提供 SAML 支援。

- `Set-AWSSamlEndpoint` 會設定您的 AD FS 端點、指派易記的名稱給端點，以及選擇性地說明端點的身分驗證類型。
- `Set-AWSSamlRoleProfile` 會建立或編輯您想使其與 AD FS 端點建立關聯的使用者帳戶描述檔，透過指定您提供給 `Set-AWSSamlEndpoint` cmdlet 的易記名稱來識別。每個角色描述檔對應到使用者獲權執行的單一角色。

  如同使用 AWS 登入資料描述檔，您指派易記名稱給角色描述檔。您可以使用相同的易用名稱搭配 `Set-AWSCredential` cmdlet，或搭配呼叫 AWS 服務 API 之任何 cmdlet 的 `-ProfileName` 參數值。

開啟新的 AWS Tools for PowerShell 工作階段。如果您執行的是 PowerShell 3.0 或更新版本，則當您執行其任一 cmdlet 時，都會自動匯入 AWS Tools for PowerShell 模組。如果您執行的是 PowerShell 2.0，則必須執行 ``Import-Module`` cmdlet，以手動方式匯入模組，如下列範例所示。

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

如何執行 **Set-AWSSamlEndpoint** 和 **Set-AWSSamlRoleProfile** Cmdlet

1. 首先，設定 AD FS 系統的端點設定。最簡單的方式是將端點存放在變數中，如本步驟所示。請務必將預留位置帳戶 ID 和 AD FS 主機名稱更換為您自己的帳戶 ID 和 AD FS 主機名稱。在 `Endpoint` 參數中指定 AD FS 主機名稱。

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

2. 若要建立端點設定，請執行 `Set-AWSSamlEndpoint` cmdlet，指定正確的 `AuthenticationType` 參數值。有效值包括 `Basic`、`Digest`、`Kerberos`、`Negotiate` 及 `NTLM`。如果您未指定此參數，預設值是 `Kerberos`。

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

Cmdlet 會使用 `-StoreAs` 參數傳回您指派的易記名稱，因此您可以在下一行中執行 `Set-AWSSamlRoleProfile` 時使用它。

3. 現在，請執行 `Set-AWSSamlRoleProfile` cmdlet 以向 AD FS 身分提供者驗證身分，並取得讓使用者獲得授權能夠 (在 SAML 聲明中) 執行的一組角色。

   `Set-AWSSamlRoleProfile` cmdlet 使用傳回的一組角色，來提示使用者選取角色以關聯至指定的描述檔，或驗證參數中提供的角色資料是否存在 (如果不存在，系統會提示使用者進行選擇)。如果使用者只獲得一個角色的授權，cmdlet 會自動將該角色關聯至描述檔，而不會提示使用者。不需要提供登入資料即可設定已加入網域的描述檔。

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

   或者，對於未加入網域的帳戶，您可以提供 Active Directory 登入資料，然後選取使用者有權存取的 AWS 角色，如以下行所示。如果您有不同的 Active Directory 使用者帳戶，這可用來區分您組織內的角色 (例如，系統管理函數)。

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

4. 在任何一種情況下，`Set-AWSSamlRoleProfile` cmdlet 都會提示您選擇應在描述檔中存放哪個角色。下列範例顯示兩個可用的角色：`ADFS-Dev` 和 `ADFS-Production`。IAM 角色與 AD FS 系統管理員的 AD 登入資料相關聯。

   ```
   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"):
   ```

   或者，您可以輸入 `RoleARN`、`PrincipalARN` 和選用的 `NetworkCredential` 參數，在沒有提示的情況下指定角色。如果身分驗證傳回的聲明中未列出指定的角色，系統會提示使用者從可用的角色中選擇。

```
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. 您可以在單一命令中新增 StoreAllRoles 參數，為所有角色建立描述檔，如以下程式碼所示。請注意，角色名稱將用於描述檔名稱。

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

如何使用角色描述檔來執行需要 AWS 登入資料的 Cmdlet

若要執行需要 AWS 登入資料的 cmdlet，您可以使用 AWS 共用登入資料檔案中定義的角色描述檔。將角色描述檔的名稱提供給 Set-AWSCredential (或做為 AWS Tools for PowerShell 中任何 ProfileName 參數的值)，以自動取得描述檔中所述角色的暫時 AWS 登入資料。

雖然您一次只能使用一個角色描述檔，但可以在 shell 工作階段中切換描述檔。當您執行 Set-AWSCredential cmdlet 其本身時，此 cmdlet 不會驗證並取得登入資料；cmdlet 會記錄您想要使用指定的角色描述檔。除非您執行需要 AWS 登入資料的 cmdlet，否則無需身分驗證或請求登入資料。

您現在可以使用從 SAMLDemoProfile 描述檔取得的暫時 AWS 登入資料，來搭配 AWS 服務 API 使用。以下章節示範如何使用角色描述檔的範例。

範例 1：使用 **Set-AWSCredential** 設定預設角色

此範例使用 Set-AWSCredential 設定 AWS Tools for PowerShell 工作階段的預設角色。接著，您可以執行需要登入資料的 cmdlet，並獲得指定角色的授權。此範例列出美國西部 (奧勒岡) 區域的所有 Amazon Elastic Compute Cloud 執行個體，與您透過 Set-AWSCredential cmdlet 指定的描述檔相關聯。

```
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}
```

範例 2：在 PowerShell 工作階段期間變更角色描述檔

此範例列出與 SAMLDemoProfile 描述檔相關聯角色的 AWS 帳戶中，所有可用的 Amazon S3 儲存貯體。此範例顯示雖然您可能稍早在 AWS Tools for PowerShell 工作階段中已使用另一個描述檔，但仍可透過使用支援的 cmdlet 指定不同的 -ProfileName 參數值，來變更描述檔。這對於從 PowerShell 命令列管理 Amazon S3 的管理員來說，是常見的任務。

```
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
```

請注意，Get-S3Bucket cmdlet 指定透過執行 Set-AWSSamlRoleProfile cmdlet 而建立之描述檔的名稱。如果您稍早在工作階段中已設定角色描述檔 (例如透過執行 Set-AWSCredential cmdlet)，但想為 Get-S3Bucket cmdlet 使用不同的角色描述檔，這個命令會很有用。描述檔經理會讓臨時登入資料可用於 Get-S3Bucket cmdlet。

雖然登入資料在 1 小時 (STS 強制執行的限制) 後過期，但 AWS Tools for PowerShell 會在工具偵測到目前登入資料過期時請求新的 SAML 聲明，以自動重新整理登入資料。

對於已加入網域的使用者，此程序不會中斷，因為目前使用者的 Windows 身分已用於身分驗證。針對未加入網域的使用者帳戶，AWS Tools for PowerShell 會顯示 PowerShell 登入資料提示，請求使用者提供密碼。使用者提供的登入資料，將用於重新驗證使用者並取得新的聲明。

範例 3：在區域中取得執行個體

以下範例列出亞太區域 (雪梨) 區域的所有 Amazon EC2 執行個體，與 ADFS-Production 描述檔使用的帳戶相關聯。這個實用命令很適合用來傳回一個區域中的所有 Amazon EC2 執行個體。

```
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
```

其他閱讀資料

如需如何實作聯合 API 存取的詳細資訊，請參閱如何使用 SAML 2.0 為聯合 API/CLI 存取實作一般解決方案。

如有支援問題或意見，請前往 AWS 開發人員論壇的 PowerShell Scripting (PowerShell 指令碼) 或 .NET Development (.NET 開發)。

Cmdlet 探索和別名

本節介紹如何列出 AWS Tools for PowerShell 支援的服務，如何顯示 AWS Tools for PowerShell 所提供支援這些服務的 Cmdlet 集，以及如何尋找用於存取這些服務的 Cmdlet 替代名稱 (也稱為別名)。

Cmdlet 探索

所有 AWS 服務操作 (或 API) 都記錄在每個服務的 API 參考指南中。舉例來說，請參閱 IAM API 參考。在大多數情況下，AWS 服務 API 和 AWS PowerShell Cmdlet 間都會有一個一對一的對應。欲取得與 AWS 服務 API 名稱對應的 cmdlet 名稱，請執行 AWS `Get-AWSCmdletName` cmdlet 並搭配 `-ApiOperation` 參數和該 AWS 服務 API 名稱。例如，若要根據任何可用 `DescribeInstances` AWS 服務 API 取得所有可能的 Cmdlet 名稱，請執行以下命令：

```
PS > Get-AWSCmdletName -ApiOperation DescribeInstances

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

```
Get-GMLInstance   DescribeInstances   Amazon GameLift Service        GML
```

-ApiOperation 參數是預設參數，因此您可以省略參數名稱。以下範例相當於先前的範例：

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

若您同時知道 API 和服務的名稱，您可以包含 -Service 參數並加上 Cmdlet 名詞字首或該 AWS 服務名稱的一部分。例如，適用於 Amazon EC2 的 Cmdlet 名詞字首為 EC2。若要取得對應到 Amazon EC2 服務中 DescribeInstances API 的 Cmdlet 名稱，請執行以下其中一個命令。他們都會產生相同的輸出：

```
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
```

這些命令的參數值有區分大小寫。

若您不知道所需 AWS 服務 API 或 AWS 服務的名稱，您可以使用包含要比對模式的 -ApiOperation 參數與 -MatchWithRegex 參數。例如，若要取得包含 SecurityGroup 的所有可用 Cmdlet 名稱，執行以下命令：

```
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
```

若您知道 AWS 服務名稱，但不知道 AWS 服務 API 的名稱，請同時包含 `-MatchWithRegex` 參數和 `-Service` 參數，將搜尋範圍限縮至單一服務。例如，若只要取得 Amazon EC2 服務中內含 `SecurityGroup` 的所有 Cmdlet 名稱，請執行以下命令

```
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
```

若您知道 AWS Command Line Interface (AWS CLI) 命令的名稱，您可以使用 `-AwsCliCommand` 參數和所需 AWS CLI 命令名稱來取得以相同 API 為基礎的 Cmdlet 名稱。例如，若要取得與 Amazon EC2 服務中 `authorize-security-group-ingress` AWS CLI 命令呼叫相對應的 Cmdlet 名稱，請執行下列命令：

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

CmdletName                    ServiceOperation              ServiceName
  CmdletNounPrefix
----------                    ----------------              -----------
  ----------------
Grant-EC2SecurityGroupIngress AuthorizeSecurityGroupIngress Amazon Elastic Compute
 Cloud EC2
```

`Get-AWSCmdletName` Cmdlet 只需要足夠的 AWS CLI 命令名稱，就能辨識服務和 AWS API。

若要取得 Tools for PowerShell Core 中所有 Cmdlet 的清單，請執行 PowerShell `Get-Command` Cmdlet，如以下範例所示。

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

您可以搭配 `-Module AWSPowerShell` 執行相同的命令，來查看 AWS Tools for Windows PowerShell 中的 Cmdlet。

`Get-Command` Cmdlet 產生的 Cmdlet 清單會以字母順序排序。請注意，根據預設，清單會根據 PowerShell 動詞排序，而非 PowerShell 名詞。

若要改為根據服務來排序結果，請執行以下命令：

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

若要篩選由 `Get-Command` Cmdlet 傳回的 Cmdlet，請將輸出輸送到 PowerShell `Select-String` Cmdlet。例如，若要檢視使用 AWS 區域的 Cmdlet 集，請執行以下命令：

```
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
```

您亦可篩選 cmdlet 名詞的服務字首，藉此尋找特定服務的 cmdlet。若要查看可用服務字首的清單，請執行 `Get-AWSPowerShellVersion -ListServiceVersionInfo`。以下範例會傳回支援 Amazon CloudWatch Events 服務的 Cmdlet。

```
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
```

Cmdlet 命名和別名

每個服務 AWS Tools for PowerShell 中的 Cmdlet 都是以該服務 AWS 開發套件所提供的方法為基礎。但是，由於 PowerShell 的強制命名慣例，Cmdlet 的名稱可能會和 API 呼叫名稱，或其根據的方法名稱不同。例如，`Get-EC2Instance` cmdlet 是根據 Amazon EC2 `DescribeInstances` 方法。

在某些情況下，cmdlet 名稱可能與方法名稱類似，但實際執行的功能則不同。例如，Amazon S3 `GetObject` 方法會擷取 Amazon S3 物件。然而，`Get-S3Object` cmdlet 會回傳 Amazon S3 物件的資訊，而非物件本身。

```
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
```

若要使用 AWS Tools for PowerShell 取得 S3 物件，請執行 `Read-S3Object` Cmdlet：

```
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**
>
> AWS cmdlet 的 cmdlet 協助會提供 cmdlet 做為根據的 AWS 開發套件 API 名稱。
> 如需標準 PowerShell 動詞及其含意的詳細資訊，請參閱核准的 PowerShell 命令動詞。

所有使用 `Remove` 動詞 (以及當您新增 `-Terminate` 參數時的 `Stop-EC2Instance` cmdlet) 的 AWS cmdlet，在繼續前會出現確認提示。若要略過確認，可在命令裡加入 `-Force` 參數。

> **Important**
>
> AWS Cmdlet 不支援 `-WhatIf` 切換。

別名

AWS Tools for PowerShell 的設定會安裝一個別名檔案，其中包含許多 AWS Cmdlet 的別名。這些別名可能較 cmdlet 名稱更為直觀。例如，在部分別名中，服務名稱和 AWS 開發套件方法名稱取代了 PowerShell 動詞和名詞。`EC2-DescribeInstances` 別名就是一個範例。

其他別名即使未遵從標準 PowerShell 慣例，但其使用的動詞可能更詳細描述實際操作。例如，別名檔案會將別名 `Get-S3Content` 對應至 cmdlet `Read-S3Object`。

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

別名檔案位於 AWS Tools for PowerShell 安裝目錄。若要將別名載入到您的環境，請對該檔案使用 dot-source命令。以下是 Windows 的範例。

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

針對 Linux 或 macOS shell，它看起來會如下：

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

若要顯示所有 AWS Tools for PowerShell 別名，請執行以下命令。這個命令會使用 PowerShell `Where-Object` Cmdlet 的 ? 別名，以及 `Source` 屬性來只篩選來自 `AWSPowerShell.NetCore` 模組的別名。

```
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

...
```

若要在這個檔案中加入您自己的別名，您可能需要將 PowerShell 的 `$MaximumAliasCount` 偏好變數值提高為大於 5500 的值。預設值為 4096；最多可以提高到 32768。若要進行這項動作，請執行下列命令。

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

若要確認變更成功，可輸入變數名稱以顯示其目前的值。

```
PS > $MaximumAliasCount
32768
```

管道傳輸和 $AWSHistory

針對傳回集合的 AWS 服務呼叫，集合內的物件會列舉到管道。包含集合以外欄位以及不是分頁控制欄位的結果物件，會將這些欄位新增為呼叫的 Note 屬性。如果您需要存取這些資料，這些 Note 屬性記錄在新的 `$AWSHistory` 工作階段變數中。下一節會說明 `$AWSHistory` 變數。

> **Note**
>
> 在 v1.1 之前的 Tools for Windows PowerShell 版本中，會發出集合物件本身，這需要使用 foreach {$_.getenumerator()} 來繼續管道輸送。

範例

下列範例會傳回 AWS 區域和每個區域中您 Amazon EC2 機器映像 (AMI) 的清單。

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

下列範例會停止目前預設區域中的所有 Amazon EC2 執行個體。

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

由於集合會列舉到管道，所以指定 cmdlet 的輸出可能會是 $null、單一物件或集合。如果是集合，您可以使用 .Count 屬性來判斷集合的大小。不過，僅發出單一物件時，.Count 屬性不存在。如果您的指令碼需要以一致的方法判斷發出多少物件，您可在 $AWSHistory 中檢查最後一個命令值的 EmittedObjectsCount 屬性。

$AWSHistory

為提供管道傳輸更好的支援，從 AWS cmdlet 的輸出不會重新改造為包含服務回應和結果執行個體，做為發出集合物件的 Note 屬性。而是改為對於發出單一集合做為輸出的這些呼叫，現在將集合列舉到 PowerShell 管道。這表示 AWS 開發套件回應和結果資料無法存在於管道中，因為沒有可連接的包含集合物件。

雖然大多數使用者可能不需要此資料，但它可用於診斷目的，因為您可以查看 cmdlet 所進行基礎 AWS 服務呼叫已傳送和已接收的確實內容。

從版本 1.1 起，名為 $AWSHistory 的新 shell 變數現在提供此項資料及更多資料。此變數維護 AWS cmdlet 叫用的記錄以及每個叫用收到的服務回應。或者，這個歷史記錄也可以設定為記錄每個 cmdlet 記錄所進行的服務請求。另外也可以從每個項目取得其他有用資料，例如 cmdlet 整體執行時間。基於安全理由，預設情況下不會記錄包含敏感資料的請求和回應。但是，如果需要的話，可以設定歷史記錄以覆寫此行為。如需詳細資訊，請參閱下方顯示的 Set-AWSHistoryConfiguration Cmdlet。

$AWSHistory.Commands 清單中的每個項目都是 AWSCmdletHistory 類型。此類型有下列有用的成員：

CmdletName

Cmdlet 的名稱。

CmdletStart

Cmdlet 執行的日期時間。

CmdletEnd

Cmdlet 完成所有處理的日期時間。

請求

如果啟用請求記錄，此為最後服務請求清單。

回應

收到的最後服務回應清單。

LastServiceResponse

傳回最新服務回應的 Helper。

LastServiceRequest

傳回最新服務回應的小幫手 (如有)。

請注意，直到使用進行服務呼叫的 AWS cmdlet，$AWSHistory 變數才會建立。在該時間之前，它會判斷值為 $null。

> **Note**
>
> 舊版 Tools for Windows PowerShell 將服務回應的相關資料發出為傳回物件的 Note 屬性。這些項目現在可在清單中為每個叫用記錄的回應項目中找到。

Set-AWSHistoryConfiguration

Cmdlet 呼叫可以保留零或多個服務請求和回應項目。為了降低對記憶體的影響，預設情況下 $AWSHistory 清單僅保留最後五個 cmdlet 執行的記錄，以及對於每個記錄的最後五個服務回應 (若啟用，則也包含最後五個服務請求)。您可以執行 Set-AWSHistoryConfiguration cmdlet 來變更這些預設限制。這可讓您同時控制清單大小，以及是否記錄服務請求：

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

所有參數都是選用的。

MaxCmdletHistory 參數設定隨時可追蹤的 cmdlet 數量上限。0 值會關閉記錄 AWS cmdlet 活動。MaxServiceCallHistory 參數設定針對每個 cmdlet 追蹤的服務回應 (和/或請求) 數量上

限。RecordServiceRequests 參數 (如果指定) 會開啟追蹤每個 cmdlet 的服務請求。如果指定 IncludeSensitiveData 參數，則會開啟追蹤包含每個 Cmdlet 之敏感資料的服務回應和請求 (如果已追蹤)。

如果不使用參數執行，Set-AWSHistoryConfiguration 會單純關閉任何之前的請求記錄，將目前清單大小保持不變。

若要清除目前歷史記錄清單中的所有項目，請執行 Clear-AWSHistory cmdlet。

$AWSHistory 範例

列舉管道清單中保留的 AWS cmdlet 詳細資訊。

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

存取最後一個執行的 AWS cmdlet 詳細資訊：

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

存取最後一個執行 AWS cmdlet 所收到最後服務回應的詳細資訊。如果 AWS cmdlet 是分頁輸出，它可能會進行多個服務呼叫以取得所有資料或最大資料量 (取決於 cmdlet 的參數)。

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

存取最後發出請求的詳細資訊 (同樣的，如果代表使用者進行分頁，cmdlet 可能會進行多次請求)。除非服務請求追蹤已啟用，否則會產生 $null。

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

對於傳回多個頁面的作業，自動為分頁到完成

對於特定呼叫施行預設最大物件傳回計數或支援可分頁結果集的服務 API，所有 cmdlet 預設為「分頁到完成」(page-to-completion)。每個 cmdlet 會代表您進行所需數量的呼叫，以傳回完整資料集至管道。

在使用 Get-S3Object 的下列範例中，$c 變數包含儲存貯體 test 中每個金鑰的 S3Object 執行個體，可能會是非常大的資料集。

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

如果您想要保有對傳回資料量的控制權，您可以使用個別 cmdlet 上的參數 (例如 Get-S3Object 上的 MaxKey)，或者您也可以使用 cmdlet 的分頁參數組合以及 $AWSHistory 變數中存放的資料來取得服務的後續字符資料，以明確地自行處理分頁。以下範例使用 MaxKeys 參數，限制傳回的 S3Object 執行個體數量為不超過儲存貯體中找到的前 500 個。

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

若要知道是否有更多可用資料但不傳回這些資料，請使用記錄 cmdlet 所進行服務呼叫的 $AWSHistory 工作階段變數項目。

如果以下表達式判斷值為 $true，您可以使用 $AWSHistory.LastServiceResponse.NextMarker 找到下一個結果集的 next 標記。

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

若要使用 Get-S3Object 手動控制分頁，請搭配使用 cmdlet 的 MaxKey 與 Marker 參數，以及最後一個記錄回應的 IsTruncated/NextMarker 備註。在下列範例中，變數 $c 包含在指定的金鑰前綴標記開始之後，在儲存貯體中找到的下 500 個物件的最多 500 個 S3Object 執行個體。

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

憑證和設定檔解析

憑證搜尋順序

當您執行命令時，AWS Tools for PowerShell 會依下列順序搜尋憑證。當它找到可用的憑證時就會停止。

1. 在命令列中內嵌為參數的文字憑證。

   強烈建議您使用描述檔，不要在命令列中放置文字憑證。
2. 指定的描述檔名稱或描述檔位置。
   - 如果您僅指定描述檔名稱，則命令會在 AWS 開發套件存放區中尋找指定的描述檔，如果該檔案不存在，則在預設位置的 AWS 共用憑證檔案中尋找指定的描述檔。

- 如果您只指定描述檔位置，命令會在該憑證檔案中尋找 `default` 描述檔。
- 如果同時指定名稱和位置，命令就會在該憑證檔案中尋找指定的描述檔。

如果找不到指定的描述檔或位置，命令會擲出例外狀況。只有在您未指定描述檔或位置時，搜尋才會繼續執行下列步驟。

3. `-Credential` 參數指定的憑證。
4. 工作階段描述檔 (如果存在)。
5. 依以下順序使用預設描述檔：
   a. AWS 開發套件存放區中的 `default` 描述檔。
   b. `default` 共用憑證檔案中的 AWS 描述檔。
   c. AWS 開發套件存放區中的 `AWS PS Default` 描述檔。
6. 如果命令是在設定使用 IAM 角色的 Amazon EC2 執行個體上執行，就會從執行個體描述檔存取 EC2 執行個體的暫時憑證。

   如需使用適用於 Amazon EC2 執行個體 的 IAM 角色詳細資訊，請參閱 AWS SDK for .NET。

如果此搜尋無法找到指定的憑證，命令會擲出例外狀況。

有關使用者和角色的其他資訊

若要在 AWS 上執行 Tools for PowerShell 命令，您需要有適合您工作的特定使用者、許可集合和服務角色組合。

您建立的特定使用者、許可集合和服務角色，以及使用這些角色的方式，將視您的需求而定。以下提供一些額外的資訊，幫助您了解可能使用它們的原因，以及建立的方法。

使用者和許可集合

雖然可以使用具有長期憑證的 IAM 使用者帳戶來存取 AWS 服務，但這不再是最佳實務，應該避免使用。即使在開發期間，最佳實務是在 AWS IAM Identity Center 中建立使用者和許可集合，並使用身分來源提供的臨時憑證。

若是開發環境，您可以使用您在 設定工具身分驗證 中建立或提供給您的使用者。如果您有適當的 AWS Management Console 許可，您也可以為該使用者建立具有最低權限的不同許可集合，或建立開發專案專用的新使用者，並提供具有最少權限的許可集合。您選擇的行動方式 (如有的話) 取決於您的情況。

如需有關這些使用者和許可集合，以及如何建立它們的詳細資訊，請參閱 AWS SDK 和工具參考指南中的身分驗證和存取和 AWS IAM Identity Center 使用者指南中的入門。

服務角色

您可以設定 AWS 服務角色，以代表使用者存取 AWS 服務。如果有多人將遠端執行您的應用程式，則此類型的存取是適當的；例如，在您為此目的建立的 Amazon EC2 執行個體上。

建立服務角色的程序會根據情況而有所不同，但基本上如下所示。

1. 登入 AWS Management Console，並開啟位於 https://console.aws.amazon.com/iam/ 的 IAM 主控台。
2. 選擇 Roles (角色)，然後選擇 Create role (建立角色)。
3. 選擇 AWS 服務，尋找並選取 EC2 (範例)，然後選擇 EC2 使用案例 (範例)。
4. 選擇下一步，然後針對您的應用程式將使用的 AWS 服務選取適當的政策。

   > **Warning**
   >
   > 請勿選擇 AdministratorAccess 政策，因為該政策會啟用您帳戶中幾乎所有內容的讀取和寫入許可。

5. 選擇 Next (下一步)。輸入角色名稱、說明，以及您想要的任何標籤。

   您可以在 IAM 使用者指南的使用 AWS 資源標籤控制存取權中找到標籤的相關資訊。
6. 選擇建立角色。

您可以在 IAM 使用者指南中的 IAM 身分 (使用者、使用者群組和角色) 中找到有關 IAM 角色的進階資訊。在 IAM 角色主題中尋找有關角色的詳細資訊。

使用舊版憑證

本節中的主題提供有關在不使用 AWS IAM Identity Center 的情況下使用長期或短期憑證資訊。

> **Warning**
>
> 為避免安全風險，在開發專用軟體或使用真實資料時，請勿使用 IAM 使用者進行身分驗證。相反地，搭配使用聯合功能和身分提供者，例如 AWS IAM Identity Center。

> **Note**
>
> 這些主題中的資訊適用於您需要手動取得及管理短期或長期憑證的情況。有關短期和長期憑證的其他資訊，請參閱 AWS SDK 和工具參考指南中的其他驗證方法。
> 如需最佳安全實務，請依照 設定工具身分驗證 中所述使用 AWS IAM Identity Center。

憑證的重要警告和指引

憑證警告

- 請勿使用您帳戶的根憑證存取 AWS 資源。這些登入資料可讓未管制的帳戶存取和很難撤銷這些帳戶。
- 請勿在命令或指令碼中放置常值存取金鑰或憑證資訊。如果這樣做，則會造成意外暴露您的憑證的風險。
- 請注意，共享 AWS `credentials` 檔案中儲存的任何憑證均以純文字形式儲存。

安全管理憑證的其他指引

如需如何安全地管理 AWS 憑證的一般討論，請參閱 AWS 一般參考 中的 AWS 安全憑證，以及《IAM 使用者指南》https://docs.aws.amazon.com/IAM/latest/UserGuide/中的安全最佳實務和使用案例。除了這些討論之外，請考慮下列事項：

- 建立其他使用者 (例如 IAM Identity Center 中的使用者)，並使用其憑證，而不是使用您的 AWS 根使用者憑證。如有必要，其他使用者的憑證可以被撤銷，或本質上是臨時的。此外，您可以將政策套用至每個使用者，以便僅存取特定資源和動作，從而採取最低權限許可的立場。

- 使用適用於 Amazon Elastic Container Service (Amazon ECS) 任務的任務 IAM 角色。

- 使用在 Amazon EC2 執行個體上執行的應用程式的 IAM 角色。

主題

- 使用 AWS 憑證
- AWS Tools for PowerShell 的共用憑證

使用 AWS 憑證

每個 AWS Tools for PowerShell 命令都必須包含一組 AWS 憑證，用於以密碼演算法登入對應的 Web 服務請求。您可以依命令、工作階段或針對所有工作階段指定憑證。

> **Warning**
>
> 為避免安全風險，在開發專用軟體或使用真實資料時，請勿使用 IAM 使用者進行身分驗證。相反地，搭配使用聯合功能和身分提供者，例如 AWS IAM Identity Center。

> **Note**
>
> 本主題中的資訊適用於您需要手動取得及管理短期或長期憑證的情況。有關短期和長期憑證的其他資訊，請參閱 AWS SDK 和工具參考指南中的其他驗證方法。
> 如需最佳安全實務，請依照 設定工具身分驗證 中所述使用 AWS IAM Identity Center。

做為避免公開憑證的最佳實務，請勿在命令中輸入文字憑證。請為您要使用的每一組憑證建立一個描述檔，然後將描述檔存放在兩個憑證存放區的其中一個。在命令中依名稱指定正確的描述檔，AWS Tools for PowerShell 就會擷取相關聯的憑證。如需有關如何安全地管理 AWS 的一般討論，請參閱Amazon Web Services 一般參考 中的管理 AWS 存取金鑰的最佳實務。

> **Note**
>
> 您需要 AWS 帳戶才能取得憑證並使用 AWS Tools for PowerShell。若要建立 AWS 帳戶，請參閱 AWS Account Management 參考指南中的開始使用：您是第一次使用 AWS 嗎？。

主題

憑證存放區位置

AWS Tools for PowerShell 可以使用兩個憑證存放區的任一個：

- AWS 開發套件存放區會加密您的憑證，並將它們存放在您的主資料夾中。在 Windows 中，此存放區位於：`C:\Users\`*`username`*`\AppData\Local\AWSToolkit\RegisteredAccounts.json`。

  AWS SDK for .NET 和 Toolkit for Visual Studio 也可以使用 AWS 開發套件存放區。
- 共用憑證檔案也位在您的主資料夾中，但以純文字形式存放憑證。

  根據預設，憑證檔案存放在這裡：
  - 在 Windows 上： `C:\Users\`*`username`*`\.aws\credentials`
  - 在 Mac/Linux 上：`~/.aws/credentials`

  AWS 開發套件和 AWS Command Line Interface 也可以使用憑證檔案。如果您在 AWS 使用者內容之外執行指令碼，請務必將包含您憑證的檔案複製到所有使用者帳戶 (本機系統和使用者) 都能存取憑證的位置。

管理描述檔

描述檔能讓您使用 AWS Tools for PowerShell 參考不同的憑證集。您可以使用 AWS Tools for PowerShell cmdlet 來管理 AWS 開發套件存放區中的描述檔。您也可以使用 Toolkit for Visual Studio，或以程式設計方式使用 AWS SDK for .NET，來管理 AWS 開發套件存放區中的描述檔。如需如何管理憑證檔案中描述檔的指示，請參閱管理 AWS 存取金鑰的最佳實務。

新增新的描述檔

若要將新的描述檔新增到 AWS 開發套件存放區，請執行 `Set-AWSCredential` 命令。它會將您的存取金鑰和秘密金鑰存放在您指定之描述檔名稱下的預設憑證檔案中。

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

- `-AccessKey`– 存取金鑰 ID。
- `-SecretKey`– 私密金鑰。
- `-StoreAs`– 描述檔名稱，必須是唯一的。若要指定預設描述檔，請使用名稱 `default`。

更新描述檔

AWS 開發套件存放區必須手動維護。如果您稍後變更服務上的憑證 (例如使用 IAM 主控台)，使用本機存放的憑證執行命令會失敗，並出現下列錯誤訊息：

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

您可以透過重複描述檔的 `Set-AWSCredential` 命令並將它傳遞到新的存取金鑰和私密金鑰，來更新描述檔。

列出描述檔

您可以使用以下命令檢查目前的名稱清單。在此範例中，名為 Shirley 的使用者可以存取三個描述檔，這些描述檔都儲存在共用憑證檔案 (`~/.aws/credentials`) 中。

```
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
```

移除描述檔

若要移除您不再需要的描述檔，請使用以下命令。

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

`-ProfileName` 參數會指定您要刪除的描述檔。

已移除的命令 Clear-AWSCredential 仍適用於回溯相容性，但最好使用 `Remove-AWSCredentialProfile`。

指定憑證

有幾種方式可以指定憑證。最好是識別描述檔，而不是將文字憑證合併到命令列中。AWS Tools for PowerShell 會使用憑證搜尋順序中描述的搜尋順序尋找描述檔。

在 Windows 上，存放在 AWS 開發套件存放區中的 AWS 憑證會使用登入的 Windows 使用者身分加密。它們無法使用其他帳戶解密，也無法在非最初建立帳戶的裝置上使用。若要使用其他使用者的憑證執行任務 (例如，有排程任務要執行的使用者帳戶)，請如上節所述設定憑證描述檔，以便以該使用者身

分登入電腦時可以使用其憑證。以任務執行使用者身分登入，完成憑證設定步驟，並建立適用於該使用者的描述檔。登出後，再使用您自己的憑證登入，以設定排程任務。

> **Note**
>
> 使用 `-ProfileName` 通用參數來指定描述檔。此參數等同於較舊 AWS Tools for PowerShell 版本的 `-StoredCredentials` 參數。如需回溯相容性，`-StoredCredentials` 仍然受支援。

預設描述檔 (建議)

如果憑證存放在名為 AWS 的描述檔中，則所有 `default` 開發套件和管理工具皆可自動在本機電腦上找到您的憑證。例如，如果您在本機電腦上有名為 `default` 的描述檔，就不必執行 `Initialize-AWSDefaultConfiguration` cmdlet 或 `Set-AWSCredential` cmdlet。這些工具會自動使用存放在該描述檔中的存取金鑰和秘密金鑰資料。若要使用您預設區域 (`Get-DefaultAWSRegion` 的結果) 以外的 AWS 區域，您可以執行 `Set-DefaultAWSRegion` 並指定區域。

如果您的描述檔並未命名為 `default`，但您想將其做為目前工作階段使用的預設描述檔，則執行 `Set-AWSCredential` 以將其設定為預設描述檔。

雖然執行 `Initialize-AWSDefaultConfiguration` 可讓您針對每個 PowerShell 工作階段指定預設描述檔，cmdlet 仍從您的自訂名稱描述檔載入憑證，並以命名描述檔覆寫 `default` 描述檔。

我們建議您不執行 `Initialize-AWSDefaultConfiguration`，除非您在未經執行個體描述檔啟動的 Amazon EC2 執行個體上執行 PowerShell 工作階段，且您想要手動設定憑證描述檔。請注意，此藍本中的憑證描述檔不包含憑證。在 EC2 執行個體上執行 `Initialize-AWSDefaultConfiguration` 所產生的憑證描述檔，不會直接存放憑證，而是指向執行個體中繼資料 (提供自動輪換的暫時憑證)。但是，它卻會存放執行個體的區域。另一個可能需要執行 `Initialize-AWSDefaultConfiguration` 的情況是，您希望避免執行個體正在運作的區域，對其他區域執行呼叫。執行該命令會永久覆寫存放在執行個體中繼資料的區域。

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

> **Note**
>
> 預設憑證放在 `default` 描述檔名稱下方的 AWS 開發套件存放區。命令會以該名稱覆寫任何現有的描述檔。

如果您的 EC2 執行個體經執行個體描述檔執行，PowerShell 即會自動從執行個體描述檔取得 AWS 憑證和區域資訊。你不需要執行 `Initialize-AWSDefaultConfiguration`。不需要從經執行個體描述檔啟動的 EC2 執行個體上執行 `Initialize-AWSDefaultConfiguration` cmdlet，因為它預設會使用 PowerShell 已經使用的相同執行個體描述檔資料。

工作階段描述檔

使用 `Set-AWSCredential` 指定特定工作階段的預設描述檔。此描述檔會覆寫工作階段期間內的任何預設描述檔。如果您想要在自己的工作階段中使用自訂名稱的描述檔，而不是目前的 `default` 描述檔，建議您使用此方法。

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

> **Note**
>
> 在 1.1 版以前的 Tools for Windows PowerShell 版本中，`Set-AWSCredential` cmdlet 無法正常運作，而且會覆寫 "MyProfileName" 指定的描述檔。建議使用較新版本 Tools for Windows PowerShell。

命令描述檔

在個別的命令中，您可以新增 `-ProfileName` 參數以指定僅適用該一個命令的描述檔。此描述檔會覆寫任何預設或工作階段描述檔，如下列範例所示。

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

> **Note**
>
> 當您指定預設或工作階段描述檔時，您也可以新增 `-Region` 參數來覆寫預設或工作階段區域。如需更多詳細資訊，請參閱 指定 AWS 區域。以下範例指定預設的描述檔和區域。
>
> ```
> PS > Initialize-AWSDefaultConfiguration -ProfileName MyProfileName -Region us-
> west-2
> ```

根據預設，AWS 共用憑證檔案假設位在使用者的主資料夾 (Windows 為 `C:\Users\username\.aws`，Linux 則為 `~/.aws`)。若要指定不同位置的憑證檔案，請加上 `-ProfileLocation` 參數並指定憑證檔案路徑。以下範例指定特定命令的非預設憑證檔案。

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

> **Note**
>
> 如果您要在通常不會登入 AWS 的時間執行 PowerShell 指令碼，例如，在非正常上班時間以排定的任務來執行 PowerShell 指令碼，當您指定要使用的描述檔時請新增 `-ProfileLocation` 參數，並將值設定為存放您憑證之檔案的路徑。為了確定您的 AWS Tools for PowerShell 指令碼以正確的帳戶憑證執行，當您的指令碼在不使用 `-ProfileLocation` 帳戶的內容或程序中執行時，您應該新增 AWS 參數。您也可以將您的憑證檔案複製到本機系統或指令碼用於執行任務的其他帳戶可存取的位置。

憑證搜尋順序

當您執行命令時，AWS Tools for PowerShell 會依下列順序搜尋憑證。當它找到可用的憑證時就會停止。

1. 在命令列中內嵌為參數的文字憑證。

   強烈建議您使用描述檔，不要在命令列中放置文字憑證。

2. 指定的描述檔名稱或描述檔位置。
   - 如果您僅指定描述檔名稱，則命令會在 AWS 開發套件存放區中尋找指定的描述檔，如果該檔案不存在，則在預設位置的 AWS 共用憑證檔案中尋找指定的描述檔。
   - 如果您只指定描述檔位置，命令會在該憑證檔案中尋找 `default` 描述檔。
   - 如果同時指定名稱和位置，命令就會在該憑證檔案中尋找指定的描述檔。

   如果找不到指定的描述檔或位置，命令會擲出例外狀況。只有在您未指定描述檔或位置時，搜尋才會繼續執行下列步驟。

3. `-Credential` 參數指定的憑證。
4. 工作階段描述檔 (如果存在)。
5. 依以下順序使用預設描述檔：

a. AWS 開發套件存放區中的 `default` 描述檔。

b. `default` 共用憑證檔案中的 AWS 描述檔。

c. AWS 開發套件存放區中的 `AWS PS Default` 描述檔。

6. 如果命令是在設定使用 IAM 角色的 Amazon EC2 執行個體上執行，就會從執行個體描述檔存取 EC2 執行個體的暫時憑證。

   如需使用適用於 Amazon EC2 執行個體 的 IAM 角色詳細資訊，請參閱 AWS SDK for .NET。

如果此搜尋無法找到指定的憑證，命令會擲出例外狀況。

在 AWS Tools for PowerShell Core 中處理憑證

AWS Tools for PowerShell Core 中的 Cmdlet 在執行時會接受 AWS 存取金鑰和秘密金鑰或憑證描述檔的名稱，類似 AWS Tools for Windows PowerShell。在 Windows 上執行時，這兩個模組都能存取 AWS SDK for .NET 憑證存放區檔案 (存放在每個使用者的 `AppData\Local\AWSToolkit\RegisteredAccounts.json` 檔案)。

此檔案以加密格式存放您的金鑰，而且無法用於不同的電腦。這是 AWS Tools for PowerShell 在憑證描述檔中搜尋的第一個檔案，也是 AWS Tools for PowerShell 存放憑證描述檔的檔案。如需 AWS SDK for .NET 憑證存放檔案的詳細資訊，請參閱設定 AWS 憑證。Tools for Windows PowerShell 模組目前不支援將憑證寫入其他檔案或位置。

兩個模組都能從其他 AWS 開發套件和 AWS 所用的 AWS CLI 共用憑證檔案中讀取描述檔。在 Windows 上，這個檔案的預設位置是 `C:\Users\<userid>\.aws\credentials`。在非 Windows 平台上，這個檔案存放在 `~/.aws/credentials`。`-ProfileLocation` 參數可用於指向非預設檔案名稱或檔案位置。

開發套件憑證存放區使用 Windows 加密 API，以加密形式存放您的憑證。這些 API 不可用於其他平台，因此 AWS Tools for PowerShell Core 模組獨自使用 AWS 共用憑證檔案，並支援將新的憑證描述檔寫入共用憑證檔案。

以下使用 `Set-AWSCredential` cmdlet 的範例指令碼，示範在 Windows 上使用 AWSPowerShell 或 AWSPowerShell.NetCore 模組來處理憑證描述檔的選項。

```
# 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
```

以下範例說明 Linux 或 macOS 作業系統上的 AWSPowerShell.NetCore 模組行為。

```
# 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
```

AWS Tools for PowerShell 的共用憑證

Tools for Windows PowerShell 支援使用 AWS 共用憑證檔案，類似於 AWS CLI 和其他 AWS 開發套件。Tools for Windows PowerShell 現在支援讀取和寫入 `basic`、`session` 和 `assume role` 憑證描述檔到 .NET 憑證檔案和 AWS 共用憑證檔案。此功能透過新的 `Amazon.Runtime.CredentialManagement` 命名空間來啟用。

> **Warning**
>
> 為避免安全風險，在開發專用軟體或使用真實資料時，請勿使用 IAM 使用者進行身分驗證。相反地，搭配使用聯合功能和身分提供者，例如 AWS IAM Identity Center。

> **Note**
>
> 本主題中的資訊適用於您需要手動取得及管理短期或長期憑證的情況。有關短期和長期憑證的其他資訊，請參閱 AWS SDK 和工具參考指南中的其他驗證方法。
> 如需最佳安全實務，請依照 設定工具身分驗證 中所述使用 AWS IAM Identity Center。

新的描述檔類型以及對 AWS 共用憑證檔案的存取，由已新增到憑證相關 cmdlet 的下列參數支援：Initialize-AWSDefaultConfiguration、New-AWSCredential 和 Set-AWSCredential。在服務 cmdlet 中，您可以新增通用參數 `-ProfileName` 來參考您的描述檔。

搭配 AWS Tools for PowerShell 使用 IAM 角色

AWS 共用憑證檔案可啟用其他類型的存取。例如，您可以使用 IAM 角色來存取您的 AWS 資源，而不是 IAM 使用者的長期憑證。若要執行此作業，您的標準描述檔必須擁有可擔任此角色的許可。當您通知 AWS Tools for PowerShell 使用可指定角色的描述檔時，AWS Tools for PowerShell 會尋找 `SourceProfile` 參數所識別的描述檔。這些憑證可用來請求 `RoleArn` 參數所指定角色的暫時憑證。當第三方擔任角色時，您可以選擇性地要求使用 Multi-Factor Authentication (MFA) 裝置或 `ExternalId` 代碼。

參數名稱	描述
ExternalId	使用者定義的外部 ID，以用於擔任角色 (如果角色要求)。通常只有在您將帳戶的存取權委派給第三方時才需要這麼做。假設指派的角色時，第三方必須包含 ExternalId 做為參數。如需詳細資訊，請參閱《IAM 使用者指南》中的將 AWS 資源的存取權授予第三方時如何使用外部 ID。
MfaSerial	MFA 序號，以用於擔任角色 (如果角色要求)。如需詳細資訊，請參閱《IAM 使用者指南》中的在 AWS 中使用多重要素驗證 (MFA)。

參數名稱	描述
RoleArn	採用角色憑證時擔任的角色 ARN。如需建立和使用角色的詳細資訊，請參閱《IAM 使用者指南》中的 IAM 角色。
SourceProfile	採用角色憑證所用的來源描述檔名稱。在此描述檔中找到的憑證會用來擔任 RoleArn 參數指定的角色。

設定擔任角色的描述檔

以下範例示範如何設定可直接擔任 IAM 角色的來源描述檔。

第一個命令會建立角色描述檔參考的來源描述檔。第二個命令會建立要擔任哪個角色的角色描述檔。第三個命令會顯示角色描述檔的憑證。

```
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
```

若要搭配 Tools for Windows PowerShell 服務 cmdlet 使用此角色描述檔，請將 -ProfileName 通用參數新增至命令以參考角色描述檔。下列範例會使用先前範例中定義的角色描述檔來存取 Get-S3Bucket cmdlet。AWS Tools for PowerShell 會在 my_source_profile 中查詢憑證、使用這些憑證代使用者呼叫 AssumeRole，然後使用這些暫時角色憑證來呼叫 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
```

使用憑證描述檔類型

若要設定憑證描述檔類型，請先了解哪些參數提供描述檔類型所需的資訊。

憑證類型	您必須使用的參數

基本
 	
這些是 IAM 使用者的長期憑證
	`-AccessKey`
	
	`-SecretKey`

工作階段：
 	
這些是您手動擷取的 IAM 角色短期憑證，例如直接呼叫 Use-STSRole cmdlet。
	`-AccessKey`
	
	`-SecretKey`
	
	`-SessionToken`

角色：
 	
這些是 AWS Tools for PowerShell 為您擷取的 IAM 角色短期憑證。
	`-SourceProfile`
	
	`-RoleArn`
	
	選用：`-ExternalId`
	
	選用：`-MfaSerial`

`ProfilesLocation` 通用參數

您可以使用 `-ProfileLocation` 寫入共用憑證檔案，以及指示 cmdlet 讀取憑證檔案。加入 `-ProfileLocation` 參數來控制 Tools for Windows PowerShell 是使用共用憑證檔案或是 .NET 憑證檔案。下表說明參數在 Tools for Windows PowerShell 中的運作方式。

描述檔位置值	描述檔解析行為
null (未設定) 或為空	首先，搜尋 .NET 憑證檔案中是否有指定名稱的描述檔。如果找不到描述檔，請在 AWS 搜尋 *(user's home directory)* `\.aws\credentials` 共用憑證檔案。

描述檔位置值	描述檔解析行為
採用 AWS 共用憑證檔案格式的檔案路徑	只搜尋指定的檔案中是否有指定名稱的描述檔。

將憑證儲存到憑證檔案

若要將憑證寫入並儲存在其中一個憑證檔案，請執行 `Set-AWSCredential` cmdlet。下列範例示範其做法。第一個命令會使用 `Set-AWSCredential` 和 `-ProfileLocation`，將存取金鑰和秘密金鑰新增至 `-ProfileName` 參數指定的描述檔。在第二行，執行 Get-Content cmdlet 以顯示憑證檔案的內容。

```
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
```

顯示您的憑證描述檔

執行 Get-AWSCredential cmdlet 並新增 `-ListProfileDetail` 參數以傳回憑證檔案類型和位置，以及描述檔名稱清單。

```
PS > Get-AWSCredential -ListProfileDetail

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

移除憑證描述檔

若要移除憑證描述檔，請執行新的 Remove-AWSCredentialProfile cmdlet。Clear-AWSCredential 已移除，但仍適用於回溯相容性。

重要說明

只有 Initialize-AWSDefaultConfiguration、New-AWSCredential 和 Set-AWSCredential 支援角色描述檔的參數。您無法直接在 `Get-S3Bucket -SourceProfile source_profile_name -RoleArn`

arn:aws:iam::999999999999:role/role_name 等命令中指定角色參數。這不起作用，因為服務 cmdlet 不直接支援 SourceProfile 或 RoleArn 參數。您反倒必須將這些參數存放在描述檔中，然後使用 -ProfileName 參數呼叫命令。

在 AWS Tools for PowerShell 中使用 AWS 服務

您可以善用本節提供的 AWS Tools for PowerShell 使用範例，藉此存取 AWS 服務。而這些範例有助於示範如何透過 Cmdlet 來執行實際的 AWS 任務。這些範例會依賴 Tools for PowerShell 所提供的 cmdlet。若要查看有哪些 cmdlet 可用，請參閱 AWS Tools for PowerShell cmdlet 參考。

PowerShell 檔案串連編碼

AWS Tools for PowerShell 有些 Cmdlet 會編輯您在 AWS 中具有的現有檔案或紀錄。範例為 `Edit-R53ResourceRecordSet`，這會呼叫適用於 Amazon Route 53 的 ChangeResourceRecordSets API。

當您在 PowerShell 5.1 或較舊版本中編輯或串連檔案時，PowerShell 會以 UTF-16 而非 UTF-8 輸出編碼。這可能增加不必要的字元，並產生無效的結果。十六進位編輯器可能展現不必要的字元。

為了避免將檔案輸出轉換為 UTF-16，您可以將命令輸送到 PowerShell 的 `Out-File` cmdlet 並指定 UTF-8 編碼，如以下範例所示。

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

如果您在 PowerShell 主控台中執行 AWS CLI 命令，則適用相同的行為。您可以在 PowerShell 主控台中將 AWS CLI 命令的輸出輸送至 `Out-File`。其他 cmdlet，例如 `Export-Csv` 或 `Export-Clixml`，也有一個 `Encoding` 參數。如需具有 `Encoding` 參數以及可讓您更正串連檔案輸出編碼的 cmdlet 完整清單，請執行下列命令：

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

> **Note**
>
> PowerShell 6.0 和更新版本 (包括 PowerShell Core) 會自動保留 UTF-8 編碼以供串連檔案輸出。

Powershell 工具傳回的物件

若要讓 AWS Tools for PowerShell 在原生 PowerShell 環境中更有用，AWS Tools for PowerShell cmdlet 傳回的物件是 .NET 物件，而不是一般從 AWS 開發套件的對應 API 傳回的 JSON 文字物件。

例如，`Get-S3Bucket` 會發出 `Buckets` 集合，而非 Amazon S3 JSON 回應物件。`Buckets` 集合可以放置在 PowerShell 管道中，並以適當的方式與之互動。同樣地，`Get-EC2Instance` 會發出 `Reservation` .NET 物件 集合，而非 `DescribeEC2Instances` JSON 結果物件。此行為的設計旨在讓 AWS Tools for PowerShell 體驗與 PowerShell 慣常體驗更趨一致。

如有需要，您可取得實際的服務回應。它們會儲存為傳回物件上的 `note` 屬性。而透過使用 `NextToken` 欄位支援分頁的 API 動作，也會連接為 `note` 屬性。

Amazon EC2

本節會逐步講解啟動 Amazon EC2 執行個體所需的步驟，其包括以下作業：

- 擷取 Amazon Machine Image (AMI) 的清單。
- 建立 SSH 驗證的金鑰對。
- 建立並設定 Amazon EC2 安全群組。
- 啟動執行個體，並擷取該執行個體的相關資訊。

Amazon Simple Storage Service (Amazon S3)

本節會逐步講解建立靜態網站並託管於 Amazon S3 所需的步驟，其將示範以下作業：

- 建立與刪除 Amazon S3 儲存貯體。
- 將檔案以物件形式上傳至 Amazon S3 儲存貯體。
- 刪除 Amazon S3 儲存貯體中的物件。
- 將 Amazon S3 儲存貯體指定為網站。

AWS Lambda 和 AWS Tools for PowerShell

本節提供 AWS Lambda Tools for PowerShell 模組的概觀，並說明設定模組所需的步驟。

Amazon SNS 和 Amazon SQS

本節會逐步講解將 Amazon SQS 佇列訂閱至 Amazon SNS 主題所需的步驟。其將示範以下作業：

- 建立一個 Amazon SNS 主題。

- 建立 Amazon SQS 佇列。
- 將佇列訂閱至 主題。
- 傳送訊息至主題。
- 從佇列接收訊息。

CloudWatch

您可以善用本節提供的範例，藉此了解如何將自訂資料發布至 CloudWatch。

- 發布自訂指標至 CloudWatch 儀表板。

另請參閱

- 開始使用 AWS Tools for Windows PowerShell。

主題

- Amazon S3 和 Tools for Windows PowerShell
- Amazon EC2 與 Tools for Windows PowerShell
- AWS Lambda 與 AWS Tools for PowerShell
- Amazon SQS、Amazon SNS 和 Tools for Windows PowerShell
- 來自 AWS Tools for Windows PowerShell 的 CloudWatch
- 在 cmdlets 中使用用 ClientConfig 參數

Amazon S3 和 Tools for Windows PowerShell

在本節中，我們以使用 Amazon S3 和 CloudFront 的 AWS Tools for Windows PowerShell 建立靜態網站。在過程中，我們展示這些服務的數個常見任務。本演練是根據託管靜態網站入門指南所建立，該指南描述使用 AWS 管理主控台的類似程序。

此處列出的命令假設您已經設定 PowerShell 工作階段的預設憑證與預設區域；所以，cmdlet 並不會呼叫憑證和區域。

> **Note**
>
> 目前沒有適用於重新命名儲存貯體或物件的 Amazon S3 API，因此，沒有可執行此任務的單一 Tools for Windows PowerShell cmdlet。若要重新命名 S3 中的物件，建議您執行 Copy-S3Object cmdlet 使用新名稱複製物件，然後執行 Remove-S3Object cmdlet 刪除原始物件。

另請參閱

- 在 AWS Tools for PowerShell 中使用 AWS 服務
- 託管於 Amazon S3 的靜態網站
- Amazon S3 主控台

主題

- 建立 Amazon S3 儲存貯體、驗證其區域，或者加以移除
- 設定 Amazon S3 儲存貯體做為網站並啟用記錄
- 將物件上傳至 Amazon S3 儲存貯體
- 刪除 Amazon S3 物件與儲存貯體
- 將內嵌文字內容上傳到 Amazon S3

建立 Amazon S3 儲存貯體、驗證其區域，或者加以移除

使用 `New-S3Bucket` cmdlet 來建立新的 Amazon S3 儲存貯體。以下範例會建立名為 `website-example` 的儲存貯體。儲存貯體的名稱必須在所有區域中為唯一。範例會在`us-west-1`區域中建立儲存貯體。

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

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

您可以使用 `Get-S3BucketLocation` cmdlet 驗證儲存貯體所在的區域。

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

```
Value
-----
us-west-2
```

當你完成本教學後，您就可以使用下面的程式碼刪除這個儲存貯體。但建議您保留這個儲存貯體，因為我們在後續範例會用到它。

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

請注意，儲存貯體移除程序需要較長時間來完成。如果您嘗試立即重新建立同名的儲存貯體，New-S3Bucket cmdlet 會失敗，直到舊的儲存貯體完全消失為止。

另請參閱

- 在 AWS Tools for PowerShell 中使用 AWS 服務
- Put Bucket (Amazon S3 服務參考)
- 適用於 Amazon S3 的 AWS PowerShell 區域

設定 Amazon S3 儲存貯體做為網站並啟用記錄

使用 Write-S3BucketWebsite cmdlet 來設定 Amazon S3 儲存貯體做為靜態網站。以下範例指定預設內容網頁的 index.html 名稱以及預設錯誤網頁的 error.html 名稱。請注意，這個 cmdlet 不會建立這些頁面。需將它們上傳為 Amazon S3 物件。

```
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    :
```

另請參閱

- 在 AWS Tools for PowerShell 中使用 AWS 服務

- Put Bucket 網站 (Amazon S3 API 參考)
- Put Bucket ACL (Amazon S3 API 參考)

將物件上傳至 Amazon S3 儲存貯體

使用 `Write-S3Object` cmdlet 可從本機檔案系統，將檔案以物件形式上傳至 Amazon S3 儲存貯體。以下範例會建立和上傳兩個簡單的 HTML 檔案至 Amazon S3 儲存貯體，並驗證上傳物件是否存在。`-File` 的 `Write-S3Object` 參數指定本機檔案系統中的檔案名稱。`-Key` 參數指定 Amazon S3 中對應物件的名稱。

Amazon 會從副檔名推斷物件的內容類型，在此例中是「.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
```

固定的 ACL 選項

使用 Tools for Windows PowerShell 指定固定的 ACL 之值，與 AWS SDK for .NET 所用的值相同。不過，請注意，這些值都不同於 Amazon S3 `Put Object` 動作所用的值。Tools for Windows PowerShell 支援下列固定的 ACL：

- NoACL
- private
- public-read
- public-read-write
- aws-exec-read
- authenticated-read
- bucket-owner-read
- bucket-owner-full-control
- log-delivery-write

如需有關這些固定 ACL 設定的詳細資訊，請參閱存取控制清單概觀。

分段上傳的相關注意事項

如果您使用 Amazon S3 API 來上傳大小大於 5 GB 的檔案，您需要使用分段上傳。不過，Tools for Windows PowerShell 提供的 `Write-S3Object` cmdlet 可以透明的方式處理大於 5 GB 的檔案上傳。

測試網站

此時，您可以使用瀏覽器瀏覽至網站來測試網站。Amazon S3 中託管的靜態網站 URL 遵循標準格式。

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

例如：

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

另請參閱

- 在 AWS Tools for PowerShell 中使用 AWS 服務
- Put 物件 (Amazon S3 API 參考)
- 固定的 ACL (Amazon S3 API 參考)

刪除 Amazon S3 物件與儲存貯體

本節會說明如何將您在先前章節中所建立的網站予以刪除。您僅需刪除 HTML 檔案的物件，接著刪除該網站的 Amazon S3 儲存貯體即可。

首先，執行 `Remove-S3Object` cmdlet，從 Amazon S3 儲存貯體中刪除 HTML 檔案的物件。

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

經由 Amazon S3 處理請求的方式，系統預期顯示的成品即為 `False` 回應。在此情況下，該回應並不表示發生問題。

您現在可以執行 `Remove-S3Bucket` cmdlet，刪除該網站目前空白的 Amazon S3 儲存貯體。

```
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    :
```

若您使用的是 AWS Tools for PowerShell 1.1 版和較新版本，則可以將 `-DeleteBucketContent` 參數新增至 `Remove-S3Bucket`；該參數會先刪除指定儲存貯體的所有物件和物件版本，再嘗試移除儲

存貯體本身。根據儲存貯體中的物件或物件版本數量，系統可能需要花費大量的時間才能完成此操作。若使用 Tools for Windows PowerShell 1.1 版以前的版本，則儲存貯體必須為空，`Remove-S3Bucket` 才能夠刪除它。

> **Note**
>
> 除非您新增 `-Force` 參數，否則 AWS Tools for PowerShell 會在 cmdlet 執行前提示您確認。

另請參閱

- 在 AWS Tools for PowerShell 中使用 AWS 服務
- 刪除物件 (Amazon S3 API 參考)
- DeleteBucket (Amazon S3 API 參考)

將內嵌文字內容上傳到 Amazon S3

`Write-S3Object` cmdlet 支援將內嵌文字內容上傳至 Amazon S3 的能力。使用 `-Content` 參數 (別名 `-Text`)，您可以指定應上傳到 Amazon S3 的文字型內容，而不需先將它轉換為檔案。參數接受簡單的一行字串，以及包含多行的 here 字串。

```
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 與 Tools for Windows PowerShell

您可以使用 AWS Tools for PowerShell 執行與 Amazon EC2 相關的常見任務。

此處列出的範例命令假設您已經設定 PowerShell 工作階段的預設憑證與預設區域；因此，當我們呼叫 cmdlet 時不包含憑證或區域。如需更多詳細資訊，請參閱 開始使用 AWS Tools for Windows PowerShell。。

主題

- 建立金鑰對
- 使用 Windows PowerShell 建立安全群組
- 使用 Windows PowerShell 來尋找 Amazon Machine Image
- 使用 Windows PowerShell 啟動 Amazon EC2 執行個體

建立金鑰對

以下 New-EC2KeyPair 範例會建立金鑰對，並存放在 PowerShell 變數 $myPSKeyPair 中

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

將金鑰對物件輸送到 Get-Member Cmdlet，來查看物件的結構。

```
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;}
```

將金鑰對物件輸送到 Format-List cmdlet 以檢視 KeyName、KeyFingerprint 和 KeyMaterial 成員的值。(輸出已截斷以利閱讀)。

```
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-----
```

KeyMaterial 成員用於存放金鑰對的私有金鑰。公有金鑰存放於 AWS 內。您無法從 AWS 擷取公有金鑰，但您可以透過比對私有金鑰和從 AWS 傳回之公有金鑰的 KeyFingerprint，來驗證公有金鑰。

檢視金鑰對的指紋

您可以使用 Get-EC2KeyPair cmdlet 來檢視金鑰對的指紋。

```
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
```

存放您的私有金鑰

若要將私有金鑰存放至檔案，請將 KeyFingerMaterial 成員輸送至 Out-File cmdlet。

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

將私有金鑰寫入檔案時，您必須指定 -Encoding ascii。否則，openssl 等工具將可能無法正確地讀取檔案。您可以使用如下所示的命令，驗證所產生檔案的格式是否正確：

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

(openssl 工具不包含在 AWS Tools for PowerShell 或 AWS SDK for .NET 中。)

移除您的金鑰對

您需要金鑰對，才能啟動和連線到執行個體。當您使用完金鑰對後，您可以將其移除。若要從 AWS 移除公有金鑰，請使用 Remove-EC2KeyPair cmdlet。出現提示時，按下 Enter 移除金鑰對。

```
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"):
```

變數 $myPSKeyPair 仍然存在於目前的 PowerShell 工作階段，也仍包含金鑰對資訊。myPSKeyPair.pem 檔案也存在。不過，私有金鑰不再有效，因為金鑰對的公有金鑰不再存放於 AWS 中。

使用 Windows PowerShell 建立安全群組

您可以使用 AWS Tools for PowerShell 來建立和設定安全群組。建立安全群組時，您需要指定該安全群組適用於 EC2-Classic 或 EC2-VPC。回應為安全群組的 ID。

如果您需要連接到您的執行個體，則必須設定安全群組以允許 SSH 流量 (Linux) 或 RDP 流量 (Windows)。

主題

- 先決條件

- 建立適用於 EC2-Classic 的安全群組
- 建立適用於 EC2-VPC 的安全群組

先決條件

您需要電腦的公有 IP 地址 (使用 CIDR 表示法)。您可以透過一項服務來取得本機電腦的公有 IP 地址。例如，Amazon 提供以下服務：http://checkip.amazonaws.com/ 或 https://checkip.amazonaws.com/。若要尋找其他能夠提供您的 IP 地址之服務，請搜尋字詞「what is my IP address」(我的 IP 地址為何)。如果您透過 ISP 或是從防火牆後方進行連線，不具備靜態 IP 地址，則需要找到您的用戶端電腦可以使用的 IP 地址範圍。

> **Warning**
>
> 若您指定 `0.0.0.0/0`，您便會啟用來自世界任何 IP 地址的流量。針對 SSH 和 RDP 通訊協定，您可能會將在測試環境中短暫進行此作業視為沒有問題，但用在生產環境則不安全。在生產環境中，請務必僅授權來自適當個別 IP 地址或地址範圍的存取。

建立適用於 EC2-Classic 的安全群組

> **Warning**
>
> 我們將於 2022 年 8 月 15 日淘汰 EC2-Classic。建議您從 EC2-Classic 遷移至 VPC。如需詳細資訊，請參閱 Amazon EC2 Linux 執行個體使用者指南中的從 EC2-Classic 遷移至 VPC 或 Amazon EC2 Windows 執行個體使用者指南。也請參閱部落格文章 EC2-Classic 網路正在淘汰 - 本文介紹如何準備。

以下範例會使用 `New-EC2SecurityGroup` Cmdlet 來建立 `EC2-Classic` 的安全群組。

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

sg-0a346530123456789
```

若要檢視安全群組的初始設定，請使用 `Get-EC2SecurityGroup` cmdlet。

```
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
```

若要設定安全群組以允許 TCP 連接埠 22 (SSH) 和 TCP 連接埠 3389 上的對內流量，請使用 Grant-EC2SecurityGroupIngress cmdlet。例如，以下範例指令碼會示範如何啟用來自單一 IP 地址 203.0.113.25/32 的 SSH 流量。

```
$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
```

若要驗證安全群組已更新，請再次執行 Get-EC2SecurityGroup Cmdlet。請注意，您無法指定 EC2-Classic 的傳出規則。

```
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                : {}
```

若要查看安全群組規則，請使用 IpPermissions 屬性。

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

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

建立適用於 EC2-VPC 的安全群組

以下 New-EC2SecurityGroup 範例會新增 -VpcId 參數來建立指定 VPC 的安全群組。

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

若要檢視安全群組的初始設定，請使用 Get-EC2SecurityGroup cmdlet。在預設情況下，適用於 VPC 的安全群組包含允許所有對外流量的規則。請注意，EC2-VPC 的安全群組不能按名稱引用。

```
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                : {}
```

若要定義 TCP 連接埠 22 (SSH) 及 TCP 連接埠 3389 上傳入流量的許可，請使用 New-Object Cmdlet。以下範例指令碼會定義單一 IP 地址 203.0.113.25/32 在 TCP 連接埠 22 和 3389 的許可。

```
$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 )
```

若要驗證安全群組是否已更新，請再次使用 Get-EC2SecurityGroup cmdlet。

```
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                : {}
```

若要檢視傳入規則，您可以從先前命令傳回的集合物件中擷取 IpPermissions 屬性。

```
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}
```

使用 Windows PowerShell 來尋找 Amazon Machine Image

啟動 Amazon EC2 執行個體時，您會指定 Amazon Machine Image (AMI) 做為執行個體的範本。然而，AWS Windows AMI 的 ID 經常會變更，因為 AWS 會提供新的 AMI 最新的更新並強化安全。您可使用 Get-EC2Image 和 Get-EC2ImageByName Cmdlet 來尋找目前的 Windows AMI 並取得其 ID。

主題

- Get-EC2Image
- Get-EC2ImageByName

Get-EC2Image

Get-EC2Image cmdlet 會擷取您可使用的 AMI 清單。

搭配陣列值 -Owner 使用 amazon, self 參數，讓 Get-EC2Image 僅擷取屬於 Amazon 或您的 AMI。在此內容中，「您」指的是您使用其憑證來呼叫 Cmdlet 的使用者。

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

您可使用 -Filter 參數來限定結果範圍。欲指定篩選條件，請建立類型 Amazon.EC2.Model.Filter 的物件。例如，下列篩選條件只會顯示 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
```

下列範例為 cmdlet 回傳的 AMI 之一；上述命令的輸出實際提供許多 AMI 的資訊。

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

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

Get-EC2ImageByName

Get-EC2ImageByName Cmdlet 可讓您依據您感興趣的伺服器組態類型，篩選 AWS Windows AMI 的清單。

不帶參數執行時，Cmdlet 會發出目前篩選條件的整組名稱，如下所示：

```
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
```

欲限縮回傳的映像組，請使用 Names 參數來指定一個或多個篩選條件名稱。

```
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
```

使用 Windows PowerShell 啟動 Amazon EC2 執行個體

若要啟動 Amazon EC2 執行個體，您需要您在先前章節中建立的金鑰對和安全群組。您也需要 Amazon Machine Image (AMI) 的 ID。如需詳細資訊，請參閱下列文件：

- 建立金鑰對
- 使用 Windows PowerShell 建立安全群組
- 使用 Windows PowerShell 來尋找 Amazon Machine Image

> **Important**
>
> 如果您啟動的執行個體不包含在免費方案內，則需要在啟動該執行個體後開始收費，收費根據執行個體的執行時間而定，即使其保持閒置狀態仍會計費。

主題

- 在 EC2-Classic 中啟動執行個體
- 在 VPC 中啟動執行個體
- 在 VPC 中啟動 Spot 執行個體

在 EC2-Classic 中啟動執行個體

> **Warning**
>
> 我們將於 2022 年 8 月 15 日淘汰 EC2-Classic。建議您從 EC2-Classic 遷移至 VPC。如需詳細資訊，請參閱 Amazon EC2 Linux 執行個體使用者指南中的從 EC2-Classic 遷移至 VPC 或 Amazon EC2 Windows 執行個體使用者指南。也請參閱部落格文章 EC2-Classic 網路正在淘汰 - 本文介紹如何準備。

以下命令會建立和啟動單一 `t1.micro` 執行個體。

```
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       : {}
```

您的執行個體一開始會處於 pending 狀態，但在幾分鐘之後就會進入 running 狀態。若要檢視執行個體的相關資訊，請使用 Get-EC2Instance cmdlet。如果您有多個執行個體，則可以使用 Filter 參數來篩選預留 ID 的結果。首先，建立 Amazon.EC2.Model.Filter 類型的物件。接下來，請呼叫使用篩選條件的 Get-EC2Instance，然後顯示 Instances 屬性。

```
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                 :
```

在 VPC 中啟動執行個體

下列命令會在指定的私有子網路中建立單一 m1.small 執行個體。安全群組必須針對指定的子網路有效。

```
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       : {}
```

您的執行個體一開始會處於 pending 狀態，但在幾分鐘之後就會進入 running 狀態。若要檢視執行個體的相關資訊，請使用 Get-EC2Instance cmdlet。如果您有多個執行個體，則可以使用 Filter 參數來篩選預留 ID 的結果。首先，建立 Amazon.EC2.Model.Filter 類型的物件。接下來，請呼叫使用篩選條件的 Get-EC2Instance，然後顯示 Instances 屬性。

```
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
```

在 VPC 中啟動 Spot 執行個體

以下範例指令碼會請求指定子網路中的 Spot 執行個體。安全群組必須是您為 VPC 所建立，且包含指定的子網路。

```
$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 與 AWS Tools for PowerShell

使用 AWSLambdaPSCore 模組，您可以在 PowerShell Core 6.0 中使用 .NET Core 2.1 執行時間開發 AWS Lambda 函數。PowerShell 開發人員可以使用 Lambda，在 PowerShell 環境中管理 AWS 資源和寫入自動化指令碼。Lambda 中的 PowerShell 支援可讓您執行 PowerShell 指令碼或函數，來回應任何 Lambda 事件，例如 Amazon S3 事件或 Amazon CloudWatch 排定事件。AWSLambdaPSCore 模組是適用於 PowerShell 的獨立 AWS 模組；它不屬於 AWS Tools for PowerShell，也不會在安裝 AWS Tools for PowerShell 時安裝 AWSLambdaPSCore 模組。

安裝 AWSLambdaPSCore 模組後，您可以使用任何可用的 PowerShell 或您自行開發的 cmdlet 來編寫無伺服器函數。AWS Lambda Tools for PowerShell 模組包含 PowerShell 型無伺服器應用程式的專案範本，以及將專案發佈到 AWS 的工具。

所有支援 Lambda 的區域皆支援 AWSLambdaPSCore 模組。如需支援區域的詳細資訊，請參閱 AWS 區域表。

先決條件

您必須先執行以下必要步驟，才能安裝和使用 AWSLambdaPSCore 模組。如需這些步驟的詳細資訊，請參閱《AWS Lambda 開發人員指南》中的設定 PowerShell 開發環境。

- 安裝正確版本的 PowerShell – Lambda 對 PowerShell 的支援取決於跨平台 PowerShell Core 6.0 版本。您可以在 Windows、Linux、或 Mac 上開發 PowerShell Lambda 函數。如果您尚未安裝此版本 PowerShell，可在 Microsoft PowerShell 文件網站中找到指示。
- 安裝 .NET Core 2.1 開發套件 - 由於 PowerShell Core 是以 .NET Core 為基礎建置而成，在 .NET Core 和 PowerShell Lambda 函數兩方面，PowerShell 的 Lambda 支援皆使用相同的 .NET Core 2.1 Lambda 執行時間。Lambda PowerShell 發佈 cmdlet 使用 .NET Core 2.1 開發套件來建立 Lambda 部署套件。可從 Microsoft 下載中心取得 .NET Core 2.1 SDK。請務必安裝軟體開發套件，而非執行時間。

安裝 AWSLambdaPSCore 模組

完成先決條件之後，您就可以安裝 AWSLambdaPSCore 模組。在 PowerShell Core 工作階段中執行下列命令。

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

您已可以開始在 PowerShell 中開始開發 Lambda 函數。如需入門詳細資訊，請參閱《AWS Lambda 開發人員指南》中的以 PowerShell 撰寫 Lambda 函數的程式設計模型。

另請參閱

- 在 AWS 開發人員部落格上宣布 PowerShell Core 的 Lambda 支援
- PowerShell Gallery 網站上的 AWSLambdaPSCore 模組
- 設定 PowerShell 開發環境
- GitHub 上的 AWS Lambda Tools for Powershell
- AWS Lambda 主控台

Amazon SQS、Amazon SNS 和 Tools for Windows PowerShell

本節提供的範例，說明執行以下作業的方法：

- 建立 Amazon SQS 佇列和取得佇列 ARN (Amazon 資源名稱)。
- 建立 Amazon SNS 主題。
- 提供許可給 SNS 主題，以便其可以傳送訊息至佇列。
- 訂閱佇列至 SNS 主題

- 提供許可給 IAM 使用者或 AWS 帳戶，以發佈至 SNS 主題，並讀取來自 SQS 佇列的訊息。
- 將訊息發佈至主題並讀取來自佇列的訊息以驗證結果。

建立 Amazon SQS 佇列和取得佇列 ARN

下列命令會在您的預設區域中建立 SQS 佇列。輸出會顯示新佇列的 URL。

```
PS > New-SQSQueue -QueueName myQueue
https://sqs.us-west-2.amazonaws.com/123456789012/myQueue
```

下面的命令會擷取佇列的 ARN。

```
PS > Get-SQSQueueAttribute -QueueUrl https://sqs.us-west-2.amazonaws.com/123456789012/
myQueue -AttributeName QueueArn
...
QueueARN              : arn:aws:sqs:us-west-2:123456789012:myQueue
...
```

建立 Amazon SNS 主題

下列命令會在您的預設區域中建立 SNS 主題，並傳回新主題的 ARN。

```
PS > New-SNSTopic -Name myTopic
arn:aws:sns:us-west-2:123456789012:myTopic
```

提供許可給 SNS 主題

下列範例指令碼會同時建立 SQS 佇列和 SNS 主題，並授與 SNS 主題的許可，以便將訊息傳送到 SQS 佇列：

```
# 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 }
```

訂閱佇列至 SNS 主題

下列命令會訂閱 SNS 主題 myQueue 的佇列 myTopic，並傳回訂閱 ID：

```
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
```

提供許可

以下命令授予對主題 sns:Publish 執行 myTopic 動作的許可

```
PS > Add-SNSPermission `
    -TopicArn arn:aws:sns:us-west-2:123456789012:myTopic `
    -Label ps-cmdlet-topic `
    -AWSAccountIds 123456789012 `
    -ActionNames publish
```

以下命令授予對佇列 sqs:ReceiveMessage 執行 sqs:DeleteMessage 和 myQueue 動作的許可

```
PS > Add-SQSPermission `
    -QueueUrl https://sqs.us-west-2.amazonaws.com/123456789012/myQueue `
    -AWSAccountId "123456789012" `
```

```
    -Label queue-permission `
    -ActionName SendMessage, ReceiveMessage
```

驗證結果

下列命令會將訊息發佈至 SNS 主題 myTopic，測試您的新佇列和主題，並傳回 MessageId。

```
PS > Publish-SNSMessage `
    -TopicArn arn:aws:sns:us-west-2:123456789012:myTopic `
    -Message "Have A Nice Day!"
728180b6-f62b-49d5-b4d3-3824bb2e77f4
```

以下命令會從 SQS 佇列 myQueue 擷取並顯示訊息。

```
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
+qTE2lId2RPkFOeGtLGawTsSPTWEvJdDbLlf7E0zZ0q1niXTUtpsZ8Swx01X3QO6u9i9qBFt0ekJFZNJp6Avu05hIklb4yc
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==
```

來自 AWS Tools for Windows PowerShell 的 CloudWatch

您可以善用本節提供的 Tools for Windows PowerShell 使用範例，藉此透過該工具將自訂指標資料發佈至 CloudWatch。

此範例會假設您已經設定 PowerShell 工作階段的預設憑證與預設區域；

發布自訂指標至 CloudWatch 儀表板

以下 PowerShell 程式碼會初始化 CloudWatch `MetricDatum` 物件並將它發佈至服務。瀏覽至 CloudWatch 主控台，即可查看此操作的結果。

```
$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
```

注意下列事項：

- 日期與時間資訊必須採用國際標準時間 (UTC)，才能夠用來將 `$dat.Timestamp` 初始化。
- 加上引號的字串值，或是不加引號的數值皆可用來將 `$dat.Value` 初始化；此範例會顯示字串值。

另請參閱

- 在 AWS Tools for PowerShell 中使用 AWS 服務
- AmazonCloudWatchClient.PutMetricData (.NET 開發套件參考)
- MetricDatum (服務 API 參考)

- Amazon CloudWatch 主控台

在 cmdlets 中使用用 ClientConfig 參數

當您連線至服務時，此 `ClientConfig` 參數可用來指定特定組態設定。此參數大部分可能的屬性都是在 `Amazon.Runtime.ClientConfig` 類別中進行定義，而該類別會繼承至 AWS 服務的 API 中。請參閱 `Amazon.Keyspaces.AmazonKeyspacesConfig` 類別以查看簡易的繼承範例。此外，特定服務會定義其他僅適用於該服務的屬性。請參閱 `Amazon.S3.AmazonS3Config` 類別以查看其他屬性的範例，尤其是 `ForcePathStyle` 屬性。

使用 **`ClientConfig`** 參數。

若要使用 `ClientConfig` 參數，您可以在命令列上將其指定為 `ClientConfig` 物件，或使用 PowerShell splatting 將參數值集合當做一個單位傳遞給命令。這些方法如下範例所示。這些範例預設 `AWS.Tools.S3` 模組已安裝並匯入，而且您的 `[default]` 憑證設定檔具有適當權限。

定義 **`ClientConfig`** 物件

```
$s3Config = New-Object -TypeName Amazon.S3.AmazonS3Config
$s3Config.ForcePathStyle = $true
$s3Config.Timeout = [TimeSpan]::FromMilliseconds(150000)
Get-S3Object -BucketName <BUCKET_NAME> -ClientConfig $s3Config
```

使用 PowerShell splatting 新增 **`ClientConfig`** 屬性

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

Get-S3Object @params
```

使用未定義的屬性

使用 PowerShell splatting 時，如果您指定的 `ClientConfig` 屬性不存在，則 AWS Tools for PowerShell 要到執行階段開始後才會偵測錯誤，此時它會傳回例外狀況。依上述修改範例：

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

Get-S3Object @params
```

此範例會產生類似下列的例外狀況：

```
Cannot bind parameter 'ClientConfig'. Cannot create object of type
 "Amazon.S3.AmazonS3Config". The UndefinedProperty property was not found for the
 Amazon.S3.AmazonS3Config object.
```

指定 AWS 區域

您可以使用 `ClientConfig` 參數來設定命令的 AWS 區域。Region (區域) 是透過 `RegionEndpoint` 屬性進行設定。AWS Tools for PowerShell 會根據下列優先順序計算要使用的 Region (區域)：

1. `-Region` 參數
2. 在 `ClientConfig` 參數中傳遞的區域
3. PowerShell 工作階段狀態
4. 共享的 AWS `config` 檔案
5. 環境變數
6. Amazon EC2 執行個體中繼資料 (若已啟用)。

此 AWS 產品或服務的安全性

雲端安全是 Amazon Web Services (AWS) 最重視的一環。身為 AWS 客戶的您，將能從資料中心和網路架構的建置中獲益，以滿足組織最為敏感的安全要求。安全是 AWS 與您共同肩負的責任。共同責任模型 將此描述為雲端本身的安全和雲端內部的安全。

雲端安全性 – AWS 負責為執行 AWS 雲端中所有服務的基礎設施提供保護，讓您可安全使用服務。安全責任在 AWS 為最優先，而且 AWS 合規計劃會由第三方稽核員定期測試和驗證安全有效性。

雲端內部的安全 – 您的責任取決於您使用的 AWS 服務，其他因素則包括資料的敏感性、組織的要求以及適用的法律規章。

此 AWS 產品或服務透過其支援的特定 Amazon Web Services (AWS) 服務，遵循共同的責任模式。如需 AWS 服務安全資訊，請參閱 AWS 服務安全文件頁面和根據合規計劃在 AWS 合規工作範圍內的 AWS 服務。

主題

此 AWS 產品或服務中的資料保護

AWS 共同的責任模型適用於此 AWS 產品或服務中的資料保護。如此模型所述，AWS 負責保護執行所有 AWS 雲端 的全球基礎設施。您負責維護在此基礎設施上託管內容的控制權。您也必須負責您所使用 AWS 服務 的安全組態和管理任務。如需有關資料隱私權的更多相關資訊，請參閱資料隱私權常見問答集。如需有關歐洲資料保護的相關資訊，請參閱 AWS 安全性部落格上的 AWS 共同的責任模型和 GDPR 部落格文章。

基於資料保護目的，建議您使用 AWS IAM Identity Center 或 AWS Identity and Access Management (IAM) 保護 AWS 帳戶 憑證，並設定個人使用者。如此一來，每個使用者都只會獲得授與完成其任務所必須的許可。我們也建議您採用下列方式保護資料：

- 每個帳戶均要使用多重要素驗證 (MFA)。
- 使用 SSL/TLS 與 AWS 資源通訊。我們需要 TLS 1.2 並建議使用 TLS 1.3。

- 使用 AWS CloudTrail 設定 API 和使用者活動日誌記錄。
- 使用 AWS 加密解決方案，以及 AWS 服務 內的所有預設安全控制項。
- 使用進階的受管安全服務（例如 Amazon Macie），協助探索和保護儲存在 Amazon S3 的敏感資料。
- 如果您在透過命令列介面或 API 存取 AWS 時，需要 FIPS 140-2 驗證的加密模組，請使用 FIPS 端點。如需有關 FIPS 和 FIPS 端點的更多相關資訊，請參閱聯邦資訊處理標準 (FIPS) 140-2 概觀。

我們強烈建議您絕對不要將客戶的電子郵件地址等機密或敏感資訊，放在標籤或自由格式的文字欄位中，例如 Name (名稱) 欄位。這包括當您使用此 AWS 產品或服務，或使用此主控台、API、AWS CLI 或 AWS 開發套件的其他 AWS 服務 服務。您在標籤或自由格式文字欄位中輸入的任何資料都可能用於計費或診斷日誌。如果您提供外部伺服器的 URL，我們強烈建議請勿在驗證您對該伺服器請求的 URL 中包含憑證資訊。

資料加密

任何安全服務都有一項重要功能，就是當資訊處於非使用中狀態時，就會將資訊加密。

靜態加密

除了代表使用者與 AWS Tools for PowerShell 服務進行互動所需的憑證以外，AWS 本身不會存放任何客戶資料。

如果您使用 AWS Tools for PowerShell 來呼叫將客戶資料傳輸至您的本機電腦進行儲存的 AWS 服務，則請參閱該服務之《使用者指南》的〈安全與合規〉一章，取得該資料的儲存、保護及加密方式的相關資訊。

傳輸中加密

根據預設，從執行 AWS Tools for PowerShell 和 AWS 服務端點之用戶端電腦傳輸的所有資料都會經過加密，其採用的方法是透過 HTTPS/TLS 連線傳送所有內容。

您不須採取任何行動即可啟用 HTTPS/TLS。它隨時處於啟用的狀態。

身分和存取權管理

AWS Identity and Access Management (IAM) 是一種 AWS 服務，讓管理員能夠安全地控制對 AWS 資源的存取權限。IAM 管理員可以控制身分身分驗證 (已登入) 和授權 (具有許可) 以使用 AWS 資源。IAM 是一種您可以免費使用的 AWS 服務。

主題

對象

AWS Identity and Access Management (IAM) 的使用方式會不同，需視您在 AWS 中所執行的工作而定。

服務使用者 - 若您使用 AWS 服務 來執行您的任務，則您的管理員可以提供您需要的登入資料和許可。隨著您為了執行作業而使用的 AWS 功能數量變多，您可能會需要額外的許可。了解存取的管理方式可協助您向管理員請求正確的許可。如果您無法存取 AWS 中的某項功能，請參閱 對 AWS 身分與存取進行疑難排解 或您所使用 AWS 服務 的使用者指南。

服務管理員：如果您負責公司內的 AWS 資源，您可能具備 AWS 的完整存取權限。您的任務是判斷服務使用者應存取的 AWS 功能及資源。接著，您必須將請求提交給您的 IAM 管理員，來變更您服務使用者的許可。檢閱此頁面上的資訊，了解 IAM 的基本概念。若要深入了解貴公司如何搭配使用 IAM 與 AWS，請參閱您所使用 AWS 服務 的使用者指南。

IAM 管理員：如果您是 IAM 管理員，建議您掌握如何撰寫政策以管理 AWS 存取權的詳細資訊。若要檢視可在 IAM 中使用的 AWS 身分型政策範例，請參閱您所使用 AWS 服務 的使用者指南。

使用身分驗證

身分驗證是使用身分憑證登入 AWS 的方式。您必須以 AWS 帳戶根使用者、IAM 使用者身分，或擔任 IAM 角色進行 驗證（登入至 AWS）。

您可以使用透過身分來源 AWS IAM Identity Center 提供的憑證，以聯合身分登入 AWS。(IAM Identity Center) 使用者、貴公司的單一登入身分驗證和您的 Google 或 Facebook 憑證都是聯合身分的範例。您以聯合身分登入時，您的管理員先前已設定使用 IAM 角色的聯合身分。您 AWS 藉由使用聯合進行存取時，您會間接擔任角色。

根據您的使用者類型，您可以登入 AWS Management Console 或 AWS 存取入口網站。如需有關登入至 AWS 的詳細資訊，請參閱《AWS 登入 使用者指南》中的如何登入您的 AWS 帳戶。

如果您是以程式設計的方式存取 AWS，AWS 提供軟體開發套件 (SDK) 和命令列介面 (CLI)，以便使用您的憑證透過密碼編譯方式簽署您的請求。如果您不使用 AWS 工具，您必須自行簽署請求。如需使用建議的方法自行簽署請求的詳細資訊，請參閱《IAM 使用者指南》中的簽署 AWS API 請求。

無論您使用何種身分驗證方法，您可能都需要提供額外的安全性資訊。例如，AWS 建議您使用多重要素驗證 (MFA) 以提高帳戶的安全。如需更多資訊，請參閱《AWS IAM Identity Center 使用者指南》中的多重要素驗證和《IAM 使用者指南》中的在 AWS 中使用多重要素驗證 (MFA)。

AWS 帳戶 根使用者

如果是建立 AWS 帳戶，您會先有一個登入身分，可以完整存取帳戶中所有 AWS 服務 與資源。此身分稱為 AWS 帳戶 根使用者，使用建立帳戶時所使用的電子郵件地址和密碼即可登入並存取。強烈建議您不要以根使用者處理日常作業。保護您的根使用者憑證，並將其用來執行只能由根使用者執行的任務。如需這些任務的完整清單，了解需以根使用者登入的任務，請參閱《IAM 使用者指南》中的需要根使用者憑證的任務。

聯合身分

最佳實務是要求人類使用者（包括需要管理員存取權的使用者）搭配身分提供者使用聯合功能，使用暫時憑證來存取 AWS 服務。

聯合身分是來自您企業使用者目錄的使用者、Web 身分供應商、AWS Directory Service、Identity Center 目錄或透過身分來源提供的憑證來存取 AWS 服務 的任何使用者。聯合身分存取 AWS 帳戶時，會擔任角色，並由角色提供暫時憑證。

對於集中式存取權管理，我們建議您使用 AWS IAM Identity Center。您可以在 IAM Identity Center 中建立使用者和群組，也可以連線並同步到自己身分來源中的一組使用者和群組，以便在您的所有 AWS 帳戶和應用程式中使用。如需 IAM Identity Center 的相關資訊，請參閱《AWS IAM Identity Center 使用者指南》中的什麼是 IAM Identity Center？。

IAM 使用者和群組

IAM 使用者是您 AWS 帳戶 中的一種身分，具備單一人員或應用程式的特定許可。建議您盡可能依賴暫時憑證，而不是擁有建立長期憑證（例如密碼和存取金鑰）的 IAM 使用者。但是如果特定使用案例需要擁有長期憑證的 IAM 使用者，建議您輪換存取金鑰。如需詳細資訊，請參閱《IAM 使用者指南》中的為需要長期憑證的使用案例定期輪換存取金鑰。

IAM 群組是一種指定 IAM 使用者集合的身分。您無法以群組身分登入。您可以使用群組來一次為多名使用者指定許可。群組可讓管理大量使用者許可的過程變得更為容易。例如，您可以擁有一個名為 IAMAdmins 的群組，並給予該群組管理 IAM 資源的許可。

使用者與角色不同。使用者只會與單一人員或應用程式建立關聯，但角色的目的是在由任何需要它的人員取得。使用者擁有永久的長期憑證，但角色僅提供暫時憑證。如需進一步了解，請參閱《IAM 使用者指南》中的建立 IAM 使用者 (而非角色) 的時機。

IAM 角色

IAM 角色是您 AWS 帳戶 中的一種身分，具備特定許可。它類似 IAM 使用者，但不與特定的人員相關聯。您可以在 AWS Management Console 中透過切換角色來暫時取得 IAM 角色。您可以透過呼叫 AWS CLI 或 AWS API 操作，或是使用自訂 URL 來取得角色。如需使用角色的方法詳細資訊，請參閱《IAM 使用者指南》中的使用 IAM 角色。

使用暫時憑證的 IAM 角色在下列情況中非常有用：

- 聯合身分使用者存取 — 如需向聯合身分指派許可，請建立角色，並為角色定義許可。當聯合身分進行身分驗證時，該身分會與角色建立關聯，並取得由角色定義的許可。如需有關聯合角色的詳細資訊，請參閱《IAM 使用者指南》中的為第三方身分供應商建立角色。如果您使用 IAM Identity Center，則需要設定許可集。為控制身分驗證後可以存取的內容，IAM Identity Center 將許可集與 IAM 中的角色相關聯。如需有關許可集的資訊，請參閱《AWS IAM Identity Center 使用者指南》中的許可集。
- 暫時 IAM 使用者許可 – IAM 使用者或角色可以擔任 IAM 角色來暫時針對特定任務採用不同的許可。
- 跨帳戶存取權 – 您可以使用 IAM 角色，允許不同帳戶中的某人（信任的委託人）存取您帳戶中的資源。角色是授予跨帳戶存取權的主要方式。但是，針對某些 AWS 服務，您可以將政策直接連接到資源（而非使用角色作為代理）。如需了解使用角色和資源型政策進行跨帳戶存取之間的差異，請參閱《IAM 使用者指南》中的 IAM 角色與資源類型政策的差異。
- 跨服務存取 – 有些 AWS 服務 會使用其他 AWS 服務 中的功能。例如，當您在服務中進行呼叫時，該服務通常會在 Amazon EC2 中執行應用程式或將物件儲存在 Amazon Simple Storage Service (Amazon S3) 中。服務可能會使用呼叫主體的許可、使用服務角色或使用服務連結角色來執行此作業。
  - 轉發存取工作階段 (FAS)：當您使用 IAM 使用者或角色在 AWS 中執行動作時，系統會將您視為主體。當您使用某些服務時，您可能會執行一個動作，而該動作之後會在不同的服務中啟動另一個動作。FAS 使用主體的許可呼叫 AWS 服務，搭配請求 AWS 服務 以向下游服務發出請求。只有在服務收到需要與其他 AWS 服務 或資源互動才能完成的請求之後，才會提出 FAS 請求。在此情況下，您必須具有執行這兩個動作的許可。如需提出 FAS 請求時的政策詳細資訊，請參閱《轉發存取工作階段》https://docs.aws.amazon.com/IAM/latest/UserGuide/access_forward_access_sessions.html。

  - 服務角色：服務角色是服務擔任的 IAM 角色，可代表您執行動作。IAM 管理員可以從 IAM 內建立、修改和刪除服務角色。如需詳細資訊，請參閱《IAM 使用者指南》中的建立角色以委派許可給 AWS 服務 服務。
  - 服務連結角色 – 服務連結角色是一種連結到 AWS 服務 的服務角色類型。服務可以擔任代表您執行動作的角色。服務連結角色會顯示在您的 AWS 帳戶 中，並由該服務所擁有。IAM 管理員可以檢視，但不能編輯服務連結角色的許可。
- 在 Amazon EC2 上執行的應用程式 – 針對在 EC2 執行個體上執行並提出 AWS CLI 和 AWS API 請求的應用程式，您可以使用 IAM 角色來管理暫時憑證。這是在 EC2 執行個體內儲存存取金鑰的較好方式。如需指派 AWS 角色給 EC2 執行個體並提供其所有應用程式使用，您可以建立連接到執行個體的執行個體設定檔。執行個體設定檔包含該角色，並且可讓 EC2 執行個體上執行的程式取得暫時憑證。如需詳細資訊，請參閱《IAM 使用者指南》中的利用 IAM 角色來授予許可給 Amazon EC2 執行個體上執行的應用程式。

如需了解是否要使用 IAM 角色或 IAM 使用者，請參閱《IAM 使用者指南》中的建立 IAM 角色 (而非使用者) 的時機。

使用政策管理存取權

您可以透過建立政策並將其附加到 AWS 身分或資源，在 AWS 中控制存取。政策是 AWS 中的一個物件，當其和身分或資源建立關聯時，便可定義其許可。AWS 會在主體（使用者、根使用者或角色工作階段）發出請求時評估這些政策。政策中的許可決定是否允許或拒絕請求。大部分政策以 JSON 文件形式儲存在 AWS 中。如需 JSON 政策文件結構和內容的詳細資訊，請參閱《IAM 使用者指南》中的 JSON 政策概觀。

管理員可以使用 AWS JSON 政策來指定誰可以存取哪些內容。也就是說，哪個主體在什麼條件下可以對什麼資源執行哪些動作。

預設情況下，使用者和角色沒有許可。若要授與使用者對其所需資源執行動作的許可，IAM 管理員可以建立 IAM 政策。然後，管理員可以將 IAM 政策新增至角色，使用者便能擔任這些角色。

IAM 政策定義該動作的許可，無論您使用何種方法來執行操作。例如，假設您有一個允許 `iam:GetRole` 動作的政策。具備該政策的使用者便可以從 AWS Management Console、AWS CLI 或 AWS API 取得角色資訊。

身分型政策

身分型政策是可以附加到身分（例如 IAM 使用者、使用者群組或角色）的 JSON 許可政策文件。這些政策可控制身分在何種條件下能對哪些資源執行哪些動作。若要了解如何建立身分類型政策，請參閱《IAM 使用者指南》中的建立 IAM 政策。

身分型政策可進一步分類成內嵌政策或受管政策。內嵌政策會直接內嵌到單一使用者、群組或角色。受管政策則是獨立的政策，您可以將這些政策附加到 AWS 帳戶 中的多個使用者、群組和角色。受管政策包含 AWS 管理政策和客戶管理政策。如需瞭解如何在受管政策及內嵌政策間選擇，請參閱 IAM 使用者指南中的在受管政策和內嵌政策間選擇。

資源型政策

資源型政策是連接到資源的 JSON 政策文件。資源型政策的最常見範例是 IAM 角色信任政策和 Amazon S3 儲存貯體政策。在支援資源型政策的服務中，服務管理員可以使用它們來控制對特定資源的存取權。對於附加政策的資源，政策會定義指定的主體可以對該資源執行的動作以及在何種條件下執行的動作。您必須在資源型政策中指定主體。主體可以包括帳戶、使用者、角色、聯合身分使用者或 AWS 服務。

資源型政策是位於該服務中的內嵌政策。您無法在資源型政策中使用來自 IAM 的 AWS 受管政策。

存取控制清單 (ACL)

存取控制清單 (ACL) 可控制哪些委託人（帳戶成員、使用者或角色）擁有存取某資源的許可。ACL 類似於資源型政策，但它們不使用 JSON 政策文件格式。

Amazon Simple Storage Service (Amazon S3)、AWS WAF 和 Amazon VPC 是支援 ACL 的服務範例。若要進一步了解 ACL，請參閱《Amazon Simple Storage Service 開發人員指南》中的存取控制清單 (ACL) 概觀。

其他政策類型

AWS 支援其他較少見的政策類型。這些政策類型可設定較常見政策類型授與您的最大許可。

- 許可界限 – 許可範圍是一種進階功能，可供您設定身分型政策能授予 IAM 實體（IAM 使用者或角色）的最大許可。您可以為實體設定許可界限。所產生的許可會是實體的身分型政策和其許可界限的交集。會在 `Principal` 欄位中指定使用者或角色的資源型政策則不會受到許可界限限制。所有這類政策中的明確拒絕都會覆寫該允許。如需許可範圍的更多相關資訊，請參閱《IAM 使用者指南》中的 IAM 實體許可範圍。
- 服務控制政策 (SCP) – SCP 是 JSON 政策，可指定 AWS Organizations 中組織或組織單位 (OU) 的最大許可。AWS Organizations 服務可用來分組和集中管理您企業所擁有的多個 AWS 帳戶。若您

啟用組織中的所有功能，您可以將服務控制政策 (SCP) 套用到任何或所有帳戶。SCP 會限制成員帳戶中實體的許可，包括每個 AWS 帳戶根使用者。如需組織和 SCP 的更多相關資訊，請參閱《AWS Organizations 使用者指南》中的 SCP 運作方式。

- 工作階段政策 – 工作階段政策是一種進階政策，您可以在透過編寫程式的方式建立角色或聯合使用者的暫時工作階段時，作為參數傳遞。所產生工作階段的許可會是使用者或角色的身分型政策和工作階段政策的交集。許可也可以來自資源型政策。所有這類政策中的明確拒絕都會覆寫該允許。如需更多資訊，請參閱《IAM 使用者指南》中的工作階段政策。

多種政策類型

將多種政策類型套用到請求時，其結果形成的許可會更為複雜、更加難以理解。如需瞭解 AWS 在涉及多種政策類型時如何判斷是否允許一項請求，請參閱 IAM 使用者指南中的政策評估邏輯。

AWS 服務 搭配 IAM 的運作方式

若要取得 AWS 服務 如何搭配大部分 IAM 功能使用的概觀資訊，請參閱《IAM 使用者指南》中的可搭配 IAM 使用的 AWS服務。

若要了解如何使用特定 AWS 服務 搭配 IAM，請參閱相關服務使用者指南的安全章節。

對 AWS 身分與存取進行疑難排解

請使用以下資訊來協助您診斷和修復使用 AWS 和 IAM 時發生的常見問題。

主題

- 我未獲授權，不得在 AWS 中執行動作
- 我未獲得執行 iam:PassRole 的授權
- 我想要允許 AWS 帳戶 外的人員存取我的 AWS 資源

我未獲授權，不得在 AWS 中執行動作

如果您收到錯誤，告知您未獲授權執行動作，您的政策必須更新，允許您執行動作。

下列範例錯誤會在mateojackson IAM 使用者嘗試使用主控台檢視一個虛構 *my-example-widget* 資源的詳細資訊，但卻無虛構 awes:*GetWidget* 許可時發生。

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 awes:GetWidget on resource: my-example-widget
```

在此情況下，必須更新 mateojackson 使用者的政策，允許使用 awes:*GetWidget* 動作存取 *my-example-widget* 資源。

如需任何協助，請聯絡您的 AWS 管理員。您的管理員提供您的登入憑證。

我未獲得執行 iam:PassRole 的授權

如果您收到錯誤，告知您未獲授權執行 iam:PassRole 動作，您的政策必須更新，允許您將角色傳遞給 AWS。

有些 AWS 服務 允許您傳遞現有的角色至該服務，而無須建立新的服務角色或服務連結角色。如需執行此作業，您必須擁有將角色傳遞至該服務的許可。

名為 marymajor 的 IAM 使用者嘗試使用主控台在 AWS 中執行動作時，發生下列範例錯誤。但是，動作要求服務具備服務角色授予的許可。Mary 沒有將角色傳遞至該服務的許可。

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

在這種情況下，Mary 的政策必須更新，允許她執行 iam:PassRole 動作。

如需任何協助，請聯絡您的 AWS 管理員。您的管理員提供您的登入憑證。

我想要允許 AWS 帳戶 外的人員存取我的 AWS 資源

您可以建立一個角色，讓其他帳戶中的使用者或您的組織外部的人員存取您的資源。您可以指定要允許哪些信任對象取得該角色。針對支援基於資源的政策或存取控制清單 (ACL) 的服務，您可以使用那些政策來授予人員存取您的資源的許可。

如需進一步了解，請參閱以下內容：

- 若要了解 AWS 是否支援這些功能，請參閱 AWS 服務 搭配 IAM 的運作方式。
- 若要了解如何存取您擁有的所有 AWS 帳戶 所提供的資源，請參閱《IAM 使用者指南》中的將存取權提供給您所擁有的另一個 AWS 帳戶 中的 IAM 使用者。
- 如需了解如何將資源的存取權提供給第三方 AWS 帳戶，請參閱《IAM 使用者指南》中的將存取權提供給第三方擁有的 AWS 帳戶。
- 如需了解如何透過聯合身分提供存取權，請參閱《IAM 使用者指南》中的將存取權提供給在外部進行身分驗證的使用者 (聯合身分)。

- 若要了解使用角色和資源型政策進行跨帳戶存取之間的差異，請參閱《IAM 使用者指南》中的 IAM 角色與資源型政策的差異。

此 AWS 產品或服務的合規驗證

要瞭解 AWS 服務 是否在特定法規遵循方案範圍內，請參閱法規遵循方案範圍內的 AWS 服務，並選擇您感興趣的法規遵循方案。如需一般資訊，請參閱 AWS 法規遵循方案。

您可使用 AWS Artifact 下載第三方稽核報告。如需詳細資訊，請參閱 AWS Artifact 中的下載報告。

您使用 AWS 服務 時的法規遵循責任取決於資料的敏感度、您的公司的合規目標，以及適用的法律和法規。AWS 提供以下資源協助您處理法規遵循事宜：

- 安全與合規快速入門指南 – 這些部署指南討論在 AWS 上部署以安全及合規為重心的基準環境的架構考量和步驟。
- Amazon Web Services 的 HIPAA 安全與法規遵循架構：本白皮書說明公司可如何運用 AWS 來建立符合 HIPAA 規定的應用程式。

> **Note**
>
> 並非全部的 AWS 服務 都符合 HIPAA 資格。如需詳細資訊，請參閱 HIPAA 資格服務參照。

- AWS 合規資源：這組手冊和指南可能適用於您的產業和位置。
- AWS 客戶合規指南：透過合規的角度瞭解共同的責任模式。這份指南橫跨多個架構 (包含國家標準技術研究所 (NIST)、支付卡產業安全標準委員會 (PCI) 和國際標準組織 (ISO))，總結保護 AWS 服務的最佳實務並將指導方針對應至安全控制。
- AWS Config 開發人員指南中的使用規則評估資源：AWS Config 服務可評估您的資源組態對於內部實務、業界準則和法規的合規狀態。
- AWS Security Hub – 此 AWS 服務 可供您全面檢視 AWS 中的安全狀態。Security Hub 使用安全控制，可評估您的 AWS 資源並檢查您的法規遵循是否符合安全業界標準和最佳實務。如需支援的服務和控制清單，請參閱 Security Hub controls reference。
- AWS Audit Manager – 此 AWS 服務 可協助您持續稽核 AWS 使用情況，以簡化管理風險與法規與業界標準的法規遵循方式。

此 AWS 產品或服務透過其支援的特定 Amazon Web Services (AWS) 服務，遵循共同的責任模式。如需 AWS 服務安全資訊，請參閱 AWS 服務安全文件頁面和根據合規計劃在 AWS 合規工作範圍內的 AWS 服務。

在 Tools for PowerShell 中強制執行最低 TLS 版本

若要提高與 AWS 服務通訊時的安全性，您應該將 Tools for PowerShell 設定為使用適當的 TLS 版本。有關如何執行此操作的詳細資訊，請參閱 AWS SDK for .NET 開發人員指南中的強制執行最低 TLS 版本。

Tools for PowerShell cmdlet 參考

Tools for PowerShell 提供可用來存取 AWS 服務的 cmdlet。若要查看有哪些 cmdlet 可用，請參閱 AWS Tools for PowerShell cmdlet 參考。

文件歷史紀錄

本主題說明 AWS Tools for PowerShell文件的重大變更。

我們也會定期更新文件以回應客戶的意見回饋。若要傳送有關主題的意見回饋，請使用位於每個頁面底部的「此頁面是否有幫助？」 旁邊的意見回饋按鈕。

如需有關變更和更新的其他資訊 AWS Tools for PowerShell，請參閱版本說明。

變更	描述	日期
設定工具驗證 AWS	在中新增有關 SSO 支援的資訊 AWS Tools for PowerShell。	2024年3月15日
下列工具的指令程式參考 PowerShell	已新增區段，其中包含 PowerShell 指令程式參考之工具的連結。	2023 年 11 月 17 日
納入了更多 IAM 最佳實務更新	更新了指南以符合 IAM 最佳實務。如需更多詳細資訊，請參閱 IAM 中的安全最佳實務。	2023 年 10 月 12 日
在 Windows 上安裝	已移除有關使用 MSI 安裝 Windows 工具 PowerShell 的相關資訊，此資訊已被取代。	2023 年 9 月 25 日
IAM 最佳實務更新	更新了指南以符合 IAM 最佳實務。如需更多詳細資訊，請參閱 IAM 中的安全最佳實務。	2023 年 9 月 8 日
流水線和 $ AWSHistory	已將 `IncludeSensitiveData` 參數新增至 `Set-AWSHistoryConfiguration` Cmdlet。	2023 年 3 月 9 日
在指令程式中使用 ClientConfig 參數	已新增有關 ClientConfig 參數支援的資訊。	2022 年 10 月 28 日

使用視窗啟動 Amazon EC2 執行個體 PowerShell	新增有關淘汰 EC2-Classic 的注意事項。	2022 年 7 月 26 日
AWS Tools for PowerShell 第四版	新增第 4 版的相關資訊，包括 Windows 和 Linux/macOS 的安裝說明，以及說明與第 3 版之差異及介紹新功能的遷移主題。	2019 年 11 月 21 日
AWS Tools for PowerShell 3.3.563	新增如何安裝及使用 `AWS.Tools.Common` 模組預覽版的相關資訊。這個新模組將舊的整體套件分解為一個共用模組和每 AWS 個服務一個模組。	2019 年 10 月 18 日
AWS Tools for PowerShell 3.3.343.0	已新增資訊至使用 AWS Tools for PowerShell介紹適 PowerShell 用於 PowerShell 核心開發人員建置 AWS Lambda 功能的 AWS Lambda 工具的部分。	2018 年 9 月 11 日
AWS Tools for Windows PowerShell 3.1.31.0	在入門章節中增加資訊，說明使用「安全性聲明標記語言」(SAML) 來支援設定使用者聯合身分的新 cmdlet。	2015 年 12 月 1 日
AWS Tools for Windows PowerShell 2.3.19	已將新指令程式的相關資訊新增至 Cmdlet 探索與別名區段，以協助使用者更輕鬆地找到所需 AWS 的`Get-AWSCmdletName` 指令程式。	2015 年 2 月 5 日

AWS Tools for Windows PowerShell 1.1.1.0	指令程式的集合輸出一律列舉至管線。 PowerShell自動支援可分頁的服務呼叫。新的 $AWSHistory shell 變量收集服務響應和可選的服務請求。AWSRegion實例使用 Region 字段而不是 SystemName 幫助管線。刪除-S3 桶支持-開關選項。DeleteObjects 固定的可用性問題與 Set-AWSCredentials。初始化-從那裡獲得憑據和區域數據的AWSDefaults 報告。Stop-EC2Instance 接受 Amazon.EC2.Model.Reservation 執行個體做為輸入。泛型 List<T> 參數類型更換為陣列類型 (T[])。刪除或終止資源的 Cmdlet 會在刪除之前提示確認。Write-S3Object 支援將內嵌文字內容上傳至 Amazon S3。	2013 年 5 月 15 日

AWS Tools for Windows PowerShell 1.0.1.0	Windows 工具 PowerShell 模組的安裝位置已變更，因此使用 Windows PowerShell 版本 3 的環境可以利用自動載入的優勢。模組和支援檔案現在安裝到 `AWS ToolsPowerShell` 下方的 `AWSPowerShell` 子資料夾。安裝程式會自動移除在 `AWS ToolsPowerShell` 資料夾中，來自舊版的檔案。Windows 版本 PowerShell (所有版本) 已在此發行版本中更新，以包含模組的父資料夾 (`AWS ToolsPowerShell`)。`PSModulePath` 對於使用 Windows PowerShell 版本 2 的系統，會更新「開始」功能表捷徑，以從新位置匯入模組，然後執行`Initialize-AWSDefaults`。對於使用 Windows PowerShell 版本 3 的系統，將更新「開始」菜單快捷方式以刪除該`Import-Module` 命令，只留下`Initialize-AWSDefaults`。如果您編輯了 PowerShell設定檔來執行`AWSPowerShell.psd1` 檔案`Import-Module` 的某個檔案，則需要將其更新為指向檔案的新位置 (或者，如果使用 PowerShell 版本 3，請移除不再需要的`Import-Module` 陳述式)。由於這些變更，Windows 工具 PowerShell 模組現在會在執行時列為	2012 年 12 月 21 日

	可用模組`Get-Module -ListAvailable` 。此外，對於 Windows PowerShell 版本 3 的使用者而言，執行由模組匯出的任何指令程式都會自動將模組載入目前的命令 PowerShell 介面中，而不需要`Import-Module` 先使用。這可在具有不允許指令碼執行之執行政策的系統上，啟用 cmdlet 互動用途。	
AWS Tools for Windows PowerShell 1.0.0.0	初始版本	2012 年 12 月 6 日

本文為英文版的機器翻譯版本，如內容有任何歧義或不一致之處，概以英文版為準。