# AWS Command Line Interface

## Guía del usuario

# AWS Command Line Interface: Guía del usuario

# Table of Contents

# ¿Qué es la AWS Command Line Interface?

La AWS CLI es una herramienta de código abierto basada en el AWS SDK for Python (Boto) que proporciona comandos para interactuar con los servicios de AWS. Con una configuración mínima, puede comenzar a utilizar toda la funcionalidad que ofrece la Consola de administración de AWS con el programa de terminal que desee.

- Shells de Linux: Utilice programas de shell comunes, tales como `Bash`, `Zsh` y `tsch`, para ejecutar comandos en Linux, macOS, or Unix.
- Línea de comandos de Windows – En Microsoft Windows, ejecute comandos en PowerShell o en el procesador de comandos de Windows.
- De forma remota: Ejecute comandos en instancias Amazon EC2 a través de una terminal remota, como PuTTY o SSH, o con Amazon EC2 Systems Manager.

La AWS CLI proporciona acceso directo a las API públicas de los servicios de AWS. Explore las capacidades de un servicio con la AWS CLI y desarrolle scripts de shell para administrar sus recursos. También puede utilizar lo que ha aprendido para desarrollar programas en otros lenguajes con el SDK de AWS.

Además de los comandos equivalentes de la API de bajo nivel, la AWS CLI también ofrece personalizaciones para varios servicios. Las personalizaciones son comandos de mayor nivel que simplifican el uso de un servicio con una API compleja. Por ejemplo, el conjunto de comandos de `aws s3` proporciona una sintaxis familiar para administrar archivos en Amazon S3.

### Example Cargar un archivo en Amazon S3

`aws s3 cp` proporciona un comando de copia tipo shell y automáticamente lleva a cabo una carga multiparte para transferir archivos grandes de forma rápida y robusta.

```
~$ aws s3 cp myvideo.mp4 s3://mybucket/
```

Realizar la misma tarea con los comandos de bajo nivel (disponibles en `aws s3api`) requeriría mucho más esfuerzo.

Según su caso de uso, es posible que desee utilizar el SDK de AWS, un conjunto de herramientas o las Herramientas de AWS para Windows PowerShell.

- Herramientas de AWS para Windows PowerShell
- AWS SDK for Java
- AWS SDK para .NET
- AWS SDK for JavaScript
- AWS SDK parar Ruby
- AWS SDK for Python (Boto)
- AWS SDK para PHP
- AWS SDK for Go
- AWS Toolkit for Eclipse

- AWS Toolkit for Visual Studio
- AWS Mobile SDK for iOS
- SDK de AWS Mobile para Android

Puede ver (y adaptar) el código fuente de la AWS CLI en GitHub en el repositorio aws-cli. Únase a la comunidad de usuarios de GitHub para hacernos llegar sus comentarios, solicitar características y realizar aportaciones.

# Uso de los ejemplos de esta guía

Los ejemplos que aparecen en esta guía se formatean con las siguientes convenciones:

- Prompt: los comandos se muestran como un símbolo de dólar ("$"). No incluya el símbolo al escribir comandos.
- Directory: cuando los comandos se deben ejecutar desde un directorio específico, el nombre del directorio se muestra antes del símbolo de comando.
- User Input: el texto del comando que se debe introducir en la línea de comando se formatea como **`entrada del usuario`**.
- Replaceable Text: el texto variable, incluidos los nombres de los recursos que usted elija o los ID generados por los servicios de AWS que deba incluir en los comandos, se formatea como *`texto reemplazable`*. En comandos de varias líneas o comandos en los que se requiere una entrada específica desde el teclado, los comandos de teclado también se pueden mostrar como texto reemplazable.
- Output: los resultados que devuelven los servicios de AWS se muestran bajo las entradas del usuario sin formato especial.

Por ejemplo, el siguiente comando incluye la entrada del usuario, el texto reemplazable, y la salida:

```
$ aws configure
AWS Access Key ID [None]: AKIAIOSFODNN7EXAMPLE
AWS Secret Access Key [None]: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
Default region name [None]: us-west-2
Default output format [None]: ENTER
```

Para usar este ejemplo, escriba **`aws configure`** en la línea de comando y pulse Intro. **`aws configure`** es el comando. Este comando es interactivo, por lo que la CLI de AWS devuelve líneas de texto que solicitan la introducción de información adicional. Introduzca cada una de sus claves de acceso y pulse Intro. A continuación, introduzca el nombre de una región en el formato que se muestra, pulse Intro y pulse de nuevo Intro para omitir la configuración del formato de salida. El comando Enter final se muestra como texto reemplazable porque no hay entrada del usuario para esa línea. De lo contrario, sería implícita.

En el siguiente ejemplo se muestra un comando sencillo no interactivo con salida desde el servicio en formato JSON:

```
$ aws ec2 create-security-group --group-name my-sg --description "My security group"
{
    "GroupId": "sg-903004f8"
}
```

Para usar este ejemplo, introduzca el texto completo del comando (el texto resaltado después de la signo) y pulse Intro. El nombre del grupo de seguridad, `my-sg`, es reemplazable. En este caso, puede usar el nombre del grupo, tal y como se muestra, pero probablemente querrá usar un nombre más descriptivo.

Note

Los argumentos que se deben reemplazar (como el ID de clave de acceso de AWS), y los que deberían reemplazarse (por ejemplo, el nombre de grupo), se muestran como *texto reemplazable*. Si debe reemplazarse un argumento, figurará en el texto que describe el ejemplo.

El documento JSON, incluidas las llaves, produce una salida. Si configura la CLI para producir la salida en formado de texto o tabla, se aplicará un formato diferente a la misma. JSON es el formato de salida predeterminado.

# Acerca de Amazon Web Services

Amazon Web Services (AWS) es una colección de servicios de infraestructura digital que los desarrolladores pueden usar cuando desarrollan sus aplicaciones. Los servicios incluyen informática, almacenamiento, base de datos y sincronización de aplicaciones (mensajería y puesta en cola). AWS usa un modelo de servicio de pago por uso. Solo se le cobrará por los servicios que usted (o su aplicación) utilice. Asimismo, para poder usar AWS como una plataforma para prototipos y experimentos, AWS ofrece una capa de uso gratuita. En esta capa, los servicios son gratuitos por debajo de determinado nivel de uso. Para obtener más información sobre los costos de AWS y la capa gratuita, consulte Test-Driving AWS in the Free Usage Tier. Para obtener una cuenta de AWS, abra la página de inicio de AWS y haga clic en Inscribirse.

# Instalación de la AWS Command Line Interface

El método de distribución principal para la AWS CLI en Linux, Windows y macOS es `pip`, un administrador de paquetes para Python que proporciona una manera sencilla de instalar, actualizar y eliminar paquetes de Python y sus dependencias.

### Versión actual de la AWS CLI

La AWS CLI se actualiza con frecuencia para incluir nuevos servicios y comandos. Para ver si tiene la versión más reciente, consulte la página de versiones en GitHub.

## Requisitos

- Python 2 versión 2.6.5+ o Python 3 versión 3.3+
- Windows, Linux, macOS, or Unix

### Note

Puede que las versiones más antiguas de Python no funcionen con todos los servicios de AWS. Si ve `InsecurePlatformWarning` o avisos de retirada de funcionalidad al instalar o utilizar la CLI de AWS, actualícese a una versión reciente.

Si ya tiene `pip` y una versión compatible de Python, puede instalar la AWS CLI con el siguiente comando:

```
$ pip install awscli --upgrade --user
```

La opción `--upgrade` indica a `pip` que actualice los requisitos ya instalados. La opción `--user` indica a `pip` que instale el programa en un subdirectorio de su directorio de usuarios para no modificar las bibliotecas que usa su sistema operativo.

Si tiene problemas al intentar instalar la AWS CLI con `pip`, puede instalar la AWS CLI en un entorno virtual (p. 13) para aislar la herramienta y sus dependencias, o utilizar una versión de Python diferente de la que usa habitualmente.

### Instaladores independientes

Para instalaciones sin conexión o automatizadas en Linux, macOS, or Unix, pruebe el instalador agregado (p. 14). El instalador agregado incluye la AWS CLI, sus dependencias y un script de shell que realiza la instalación.
En Windows, también puede usar el instalador MSI (p. 10). Ambos métodos simplifican la instalación inicial, pero son más difíciles de actualizar cuando esté disponible una nueva versión de la AWS CLI.

Una vez instalada la AWS CLI, es posible que tenga que añadir la ruta del archivo ejecutable a su variable PATH. Si desea obtener instrucciones específicas de las distintas plataformas, consulte estos temas:

- Linux – Añadir el ejecutable de la AWS CLI a su ruta de la línea de comandos (p. 8)
- Windows – Añadir el ejecutable de la AWS CLI a su ruta de la línea de comandos (p. 11)

- macOS – Añadir el ejecutable de la AWS CLI a su ruta de la línea de comandos (p. 13)

Compruebe que la AWS CLI se ha instalado correctamente ejecutando `aws --version`.

```
$ aws --version
aws-cli/1.11.84 Python/3.5.2 Linux/4.4.0-59-generic botocore/1.5.47
```

La AWS CLI se actualiza periódicamente para incluir nuevos servicios y comandos. Para actualizar a la última versión de la AWS CLI, ejecute el comando de instalación de nuevo.

```
$ pip install awscli --upgrade --user
```

Si necesita desinstalar la AWS CLI, utilice `pip uninstall`.

```
$ pip uninstall awscli
```

Si no dispone de Python ni `pip`, utilice el procedimiento para su sistema operativo:

Secciones

- Instalación de la AWS Command Line Interface en Linux (p. 5)
- Instalación de la AWS Command Line Interface en Microsoft Windows (p. 9)
- Instalación de la AWS Command Line Interface en macOS (p. 12)
- Instalación de la AWS Command Line Interface en un entorno virtual (p. 13)
- Instalación de la AWS CLI con el instalador agregado (Linux, macOS, or Unix) (p. 14)

# Instalación de la AWS Command Line Interface en Linux

Puede instalar la AWS Command Line Interface y sus dependencias en la mayoría de distribuciones de Linux con `pip`, un administrador de paquetes para Python.

> **Important**
>
> El paquete `awscli` está disponible en los repositorios de otros administradores de paquetes, como APT y yum, pero no se garantiza que sea la versión más actualizada, a menos que la obtenga de `pip` o use el instalador agregado (p. 14)

Si ya tiene pip, siga las instrucciones del tema de instalación (p. 4) principal. Ejecute `pip --version` para ver si su versión de Linux ya incluye Python y pip.

```
$ pip --version
```

Si no dispone de pip, compruebe qué versión de Python está instalada.

```
$ python --version
```

o bien

```
$ python3 --version
```

Si no dispone de Python 2 versión 2.6.5+ o Python 3 versión 3.3+, instale Python (p. 6). De lo contrario, instale pip y la AWS CLI.

# Instalación de Python en Linux

Si su distribución no incluye Python, o incluye una versión antigua, instale Python antes de instalar pip y la AWS CLI.

Para instalar Python 3 en Linux

1. Compruebe si Python ya está instalado:

```
$ python --version
```

   Note

   Si su distribución de Linux incluye Python, es posible que tenga que instalar el paquete de desarrollador de Python para obtener los encabezados y las bibliotecas necesarios para compilar extensiones e instalar la AWS CLI. Instale el paquete de desarrollador (normalmente denominado python-dev o python-devel) usando el administrador de paquetes.

2. Si no está instalado Python 2.7 o una versión posterior, instale Python con el administrador de paquetes de su distribución. El comando y el nombre del paquete varían:

   - En derivados de Debian, como por ejemplo Ubuntu, use `APT`:

```
$ sudo apt-get install python3
```

   - En Red Hat y sus derivados, use `yum`:

```
$ sudo yum install python
```

   - En SUSE y sus derivados, use `zypper`:

```
$ sudo zypper install python3
```

3. Abra un símbolo del sistema o shell y ejecute el siguiente comando para verificar que Python está instalado correctamente:

```
$ python3 --version
Python 3.5.2
```

# Instalación de la AWS Command Line Interface en Amazon Linux 2017

La AWS CLI viene preinstalada en la AMI de Amazon Linux. Consulte la versión instalada actualmente con `aws --version`.

```
$ aws --version
aws-cli/1.11.83 Python/2.7.12 Linux/4.9.20-11.31.amzn1.x86_64 botocore/1.5.46
```

Puede utilizar `sudo yum update` para obtener la última versión disponible en el repositorio yum, pero es posible que no sea la última versión. Utilice pip para obtener la última versión.

Para actualizar la AWS CLI en Amazon Linux (nodo raíz)

1. Utilice `pip install` para instalar la última versión de la AWS CLI.

```
$ sudo pip install --upgrade awscli
```

2. Verifique la nueva versión con `aws --version`.

```
$ aws --version
aws-cli/1.11.85 Python/2.7.12 Linux/4.9.20-11.31.amzn1.x86_64 botocore/1.5.48
```

Si no tiene privilegios raíz, instale la AWS CLI en modo de usuario.

Para actualizar la AWS CLI en Amazon Linux (usuario)

1. Utilice `pip install` para instalar la última versión de la AWS CLI.

```
$ sudo pip install --upgrade --user awscli
```

2. Añada la ubicación de instalación al principio de la variable `PATH`.

```
$ export PATH=/home/ec2-user/.local/bin:$PATH
```

   Añada este comando al final de `~/.bashrc` para conservar el cambio de una sesión a otra.

3. Verifique la nueva versión con `aws --version`.

```
$ aws --version
aws-cli/1.11.85 Python/2.7.12 Linux/4.9.20-11.31.amzn1.x86_64 botocore/1.5.48
```

Secciones



# Instalación de Pip

Si no dispone de `pip`, instale `pip` en el script proporcionado por Python Packaging Authority.

Para instalar pip

1. Descargue el script de instalación desde pypa.io:

```
$ curl -O https://bootstrap.pypa.io/get-pip.py
```

   El script descarga e instala la última versión de `pip` y otro paquete necesario denominado `setuptools`.

2. Ejecute el script con Python:

```
$ python get-pip.py --user
```

3. Añada la ruta del ejecutable a su variable PATH: `~/.local/bin`

   Para modificar la variable PATH (Linux, macOS, or Unix)

   1. Busque el script de perfil de su shell en su carpeta de usuario. Si no está seguro de cuál es el shell que tiene, ejecute `echo $SHELL`.

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

      - Bash: `.bash_profile`, `.profile` o `.bash_login`.
      - Zsh: `.zshrc`
      - Tcsh: `.tcshrc`, `.cshrc` o `.login`.
   2. Añada un comando de exportación al script de su perfil.

      ```
      export PATH=~/.local/bin:$PATH
      ```

      Este comando añade una ruta, `~/.local/bin` en este ejemplo, a la variable PATH actual.
   3. Cargue el perfil en su sesión actual.

      ```
      $ source ~/.bash_profile
      ```
4. Compruebe que pip se instala correctamente.

   ```
   $ pip --version
   pip 8.1.2 from ~/.local/lib/python3.4/site-packages (python 3.4)
   ```

# Instalación de la AWS CLI con Pip

Utilice `pip` para instalar la AWS CLI.

```
$ pip install awscli --upgrade --user
```

Compruebe que la AWS CLI esté instalada correctamente.

```
$ aws --version
aws-cli/1.11.84 Python/3.5.2 Linux/4.4.0-59-generic botocore/1.5.47
```

Si obtiene un error, consulte Solución de errores de la AWS CLI (p. 92).

Para actualizar a la versión más reciente, ejecute el comando de instalación de nuevo:

```
$ pip install awscli --upgrade --user
```

# Añadir el ejecutable de la AWS CLI a su ruta de la línea de comandos

Después de realizar la instalación con `pip`, es posible que deba añadir el archivo ejecutable `aws` a la variable de entorno `PATH` del sistema operativo.

Example Ubicación de instalación de la AWS CLI: Linux con pip (modo usuario)

```
~/.local/bin
```

Si no realizó la instalación en modo de usuario, el archivo ejecutable podría estar en la carpeta `bin` de la instalación de Python. Si no sabe dónde se ha instalado Python, ejecute `which python`.

```
$ which python
/usr/local/bin/python
```

La salida puede ser la ruta a un symlink, no el archivo ejecutable real. Ejecute `ls -al` para ver adónde apunta.

```
$ ls -al /usr/local/bin/python
~/.local/Python/3.6/bin/python3.6
```

Para modificar la variable PATH (Linux, macOS, or Unix)

1. Busque el script de perfil de su shell en su carpeta de usuario. Si no está seguro de cuál es el shell que tiene, ejecute `echo $SHELL`.

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

   - Bash: `.bash_profile`, `.profile` o `.bash_login`.
   - Zsh: `.zshrc`
   - Tcsh: `.tcshrc`, `.cshrc` o `.login`.
2. Añada un comando de exportación al script de su perfil.

   ```
   export PATH=~/.local/bin:$PATH
   ```

   Este comando añade una ruta, `~/.local/bin` en este ejemplo, a la variable PATH actual.
3. Cargue el perfil en su sesión actual.

   ```
   $ source ~/.bash_profile
   ```

# Instalación de la AWS Command Line Interface en Microsoft Windows

Puede instalar la AWS CLI en Windows con un instalador independiente o con `pip`, un administrador de paquetes para Python. Si ya tiene `pip`, siga las instrucciones del tema de instalación (p. 4) principal.

Secciones

# Instalador de MSI

La CLI de AWS es compatible con Microsoft Windows XP o posterior. Para los usuarios de Windows, el paquete de instalación de MSI ofrece un entorno familiar y cómodo para instalar la AWS CLI sin necesidad de instalar ningún otro requisito previo.

Cuando se hayan publicado las actualizaciones, deberá repetir el proceso de instalación para obtener la última versión de la AWS CLI. Si prefiere actualizarla con frecuencia, tenga en cuenta la posibilidad de usar pip (p. 11) para instalar actualizaciones con más facilidad.

Para instalar la AWS CLI utilizando el instalador MSI

1. Descargue el instalador MSI adecuado.

   - Descarga del instalador MSI de la AWS CLI para Windows (64 bits)
   - Descarga del instalador MSI de la AWS CLI para Windows (32 bits)

   > **Note**
   >
   > El instalador MSI para la CLI de AWS no funciona con Windows Server 2008 (versión 6.0.6002). Use pip (p. 11) para instalarla en esta versión de Windows.

2. Ejecute el instalador MSI descargado.
3. Siga las instrucciones que aparecen en pantalla.

La CLI se instala de forma predeterminada en `C:\Program Files\Amazon\AWSCLI` (64 bits) o `C:\Program Files (x86)\Amazon\AWSCLI` (32 bits). Para confirmar la instalación, use el comando `aws --version` en un símbolo del sistema (abra el menú INICIO y busque "cmd" si no sabe dónde está instalado el símbolo del sistema).

```
> aws --version
aws-cli/1.11.84 Python/3.5.2 Windows/7 botocore/1.5.47
```

No incluya el símbolo de comando (">", que aparece arriba) al escribir un comando. Estos símbolos se incluyen en los listados del programa para diferenciar los comandos introducidos de los que devuelve la CLI. En el resto de esta guía se utiliza el símbolo de comando genérico "$", excepto en los casos en los que un comando es específico de Windows.

Si Windows no puede encontrar el ejecutable, es posible que tenga que volver a abrir el símbolo del sistema o añadir el directorio de instalación a su variable de entorno PATH (p. 11) manualmente.

## Actualización de una instalación MSI

La CLI de AWS se actualiza de forma habitual. Consulte la página de Versiones en GitHub para ver si se ha publicado una versión más reciente. Para actualizar a la última versión, descargue y ejecute de nuevo el instalador MSI según se detalla anteriormente.

## Desinstalación

Para desinstalar la AWS CLI, abra el Panel de control y seleccione Programas y características. Seleccione la entrada denominada interfaz de línea de comandos de AWS y haga clic en Desinstalar para iniciar el desinstalador. Confirme que desea desinstalar la AWS CLI cuando se le solicite.

También puede iniciar el menú Programas y características desde la línea de comandos con el siguiente comando:

```
> appwiz.cpl
```

# Instalación de Python, pip y la AWS CLI en Windows

La Python Software Foundation ofrece instaladores para Windows que incluyen `pip`.

### Para instalar Python 3.6 y pip (Windows)

1. Descargue el instalador del archivo ejecutable de Python 3.6 para Windows x86-64 desde la página de descargas de Python.org.
2. Ejecute el instalador.
3. Elija Add Python 3.6 to PATH.
4. Seleccione Install Now.

El instalador instala Python en su carpeta de usuario y añade sus directorios de archivos ejecutables a su ruta de usuario.

### Para instalar la AWS CLI con pip (Windows)

1. Abra el procesador de comandos de Windows desde el menú Inicio.
2. Compruebe que tanto Python como pip estén instalados correctamente con los siguientes comandos:

```
C:\Windows\System32> python --version
Python 3.6.2
C:\Windows\System32> pip --version
pip 9.0.1 from c:\users\myname\appdata\local\programs\python\python36\lib\site-packages
 (python 3.6)
```

3. Instale la AWS CLI con `pip`:

```
C:\Windows\System32> pip install awscli
```

4. Compruebe que la AWS CLI esté instalada correctamente:

```
C:\Windows\System32> aws --version
aws-cli/1.11.84 Python/3.5.2 Windows/7 botocore/1.5.47
```

Para actualizar a la versión más reciente, ejecute el comando de instalación de nuevo:

```
C:\Windows\System32> pip install --user --upgrade awscli
```

# Añadir el ejecutable de la AWS CLI a su ruta de la línea de comandos

Después de realizar la instalación con `pip`, añada el ejecutable `aws` a su variable de entorno `PATH` del sistema operativo. Con una instalación de MSI, esto debería ocurrir automáticamente, pero es posible que deba configurarlo manualmente si el comando `aws` no funciona.

- Python 3.6 y pip: `%USERPROFILE%\AppData\Local\Programs\Python\Python36\Scripts`
- Instalador MSI (64 bits): `C:\Program Files\Amazon\AWSCLI`
- Instalador MSI (32 bits): `C:\Program Files (x86)\Amazon\AWSCLI`

Para modificar su variable PATH (Windows)

1. Pulse la tecla de Windows y escriba las **variables de entorno**.
2. Elija Editar las variables de entorno de esta cuenta.
3. Elija PATH y después elija Editar.
4. Añada rutas al campo Valor de la variable, separadas por punto y coma. Por ejemplo: *C:\existing\path*;***C:\new\path***
5. Elija Aceptar dos veces para aplicar la nueva configuración.
6. Cierre los símbolos del sistema en ejecución y vuelva a abrirlos.

# Instalación de la AWS Command Line Interface en macOS

Si tiene pip, siga las instrucciones del tema de instalación (p. 4) principal. Ejecute `pip --version` para ver si su versión de macOS ya incluye Python y pip.

```
$ pip --version
```

Secciones



## Instalación de Python, pip y la AWS CLI en macOS

Puede instalar la última versión de Python y pip y utilizarlos después para instalar la AWS CLI.

Para instalar la AWS CLI en macOS

1. Descargue e instale Python 3.6 de la página de descargas de Python.org.
2. Instale `pip` con el script proporcionado por Python Packaging Authority.

   ```
   $ curl -O https://bootstrap.pypa.io/get-pip.py
   $ python3 get-pip.py --user
   ```

3. Utilice `pip` para instalar la AWS CLI.

   ```
   $ pip3 install awscli --upgrade --user
   ```

4. Compruebe que la AWS CLI esté instalada correctamente.

   ```
   $ aws --version
   AWS CLI 1.11.84 (Python 3.6.1)
   ```

   Si el ejecutable no se encuentra, añádalo a su ruta de la línea de comandos (p. 13).

Para actualizar a la versión más reciente, ejecute el comando de instalación de nuevo:

```
$ pip3 install awscli --upgrade --user
```

# Añadir el ejecutable de la AWS CLI a su ruta de la línea de comandos

Después de realizar la instalación con `pip`, es posible que deba añadir el archivo ejecutable `aws` a la variable de entorno `PATH` del sistema operativo. La ubicación del archivo ejecutable depende de dónde se haya instalado Python.

Example Ubicación de instalación de la AWS CLI - macOS con Python 3.6 y pip (modo usuario)

```
~/Library/Python/3.6/bin
```

Si no sabe dónde se ha instalado Python, ejecute `which python`.

```
$ which python
/usr/local/bin/python
```

La salida puede ser la ruta a un symlink, no el archivo ejecutable real. Ejecute `ls -al` para ver adónde apunta.

```
$ ls -al /usr/local/bin/python
~/Library/Python/3.6/bin/python3.6
```

`pip` instala los archivos ejecutables en la misma carpeta que contiene el archivo ejecutable de Python. Añada esta carpeta a la variable PATH.

Para modificar la variable PATH (Linux, macOS, or Unix)

1. Busque el script de perfil de su shell en su carpeta de usuario. Si no está seguro de cuál es el shell que tiene, ejecute `echo $SHELL`.

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

   - Bash: `.bash_profile`, `.profile` o `.bash_login`.
   - Zsh: `.zshrc`
   - Tcsh: `.tcshrc`, `.cshrc` o `.login`.
2. Añada un comando de exportación al script de su perfil.

   ```
   export PATH=~/.local/bin:$PATH
   ```

   Este comando añade una ruta, `~/.local/bin` en este ejemplo, a la variable PATH actual.
3. Cargue el perfil en su sesión actual.

   ```
   $ source ~/.bash_profile
   ```

# Instalación de la AWS Command Line Interface en un entorno virtual

Puede evitar conflictos de versiones con otros paquetes pip instalando la AWS CLI en un entorno virtual.

Para instalar la AWS CLI en un entorno virtual

1. Instale `virtualenv` con pip.

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

2. Cree un entorno virtual.

```
$ virtualenv ~/cli-ve
```

Puede usar la opción `-p` para utilizar un Python ejecutable diferente al predeterminado.

```
$ virtualenv -p /usr/bin/python3.4 ~/cli-ve
```

3. Active el entorno virtual.

Linux, macOS, or Unix

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

Windows

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

4. Instalar la AWS CLI.

```
(cli-ve)~$ pip install --upgrade awscli
```

5. Compruebe que la AWS CLI esté instalada correctamente.

```
$ aws --version
aws-cli/1.11.84 Python/3.5.2 Linux/4.4.0-59-generic botocore/1.5.47
```

Puede utilizar el comando `deactivate` para salir del entorno virtual. Siempre que inicie una nueva sesión, vuelva a ejecutar el comando de activación.

Para actualizar a la versión más reciente, ejecute el comando de instalación de nuevo:

```
(cli-ve)~$ pip install --upgrade awscli
```

# Instalación de la AWS CLI con el instalador agregado (Linux, macOS, or Unix)

En Linux, macOS, or Unix, también puede usar el instalador agregado para instalar la AWS CLI. Este incluye todas las dependencias y se puede usar sin conexión.

> **Important**
>
> El instalador agregado no admite la instalación en rutas que contienen espacios.

Secciones

# Requisitos previos

- Linux, macOS, or Unix
- Python 2 versión 2.6.5+ o Python 3 versión 3.3+

Compruebe su instalación de Python:

```
$ python --version
```

Si aún no se ha instalado Python en el equipo o quiere instalar otra versión distinta, siga el procedimiento detallado en Instalación de la AWS Command Line Interface en Linux (p. 5).

# Instalación de la AWS CLI con el instalador agregado

Siga estos pasos desde la línea de comandos para instalar la AWS CLI con el instalador agregado.

Para instalar la AWS CLI con el instalador agregado

1. Descargue el instalador agregado de la AWS CLI.

   ```
   $ curl "https://s3.amazonaws.com/aws-cli/awscli-bundle.zip" -o "awscli-bundle.zip"
   ```

2. Descomprima el paquete.

   ```
   $ unzip awscli-bundle.zip
   ```

   Note

   Si no tiene `unzip`, use el administrador de paquetes integrado de la distribución de Linux para instalarlo.

3. Ejecute el archivo de instalación.

   ```
   $ sudo ./awscli-bundle/install -i /usr/local/aws -b /usr/local/bin/aws
   ```

   Note

   El script de instalación se ejecuta en la versión de Python predeterminada del sistema. Si tiene instalada una versión alternativa de Python y quiere usarla para instalar la CLI de AWS, ejecute el script de instalación de esa versión mediante una ruta absoluta al ejecutable de Python. Por ejemplo:

   ```
   $ sudo /usr/local/bin/python2.7 awscli-bundle/install -i /usr/local/aws -b /usr/
   local/bin/aws
   ```

El instalador instala la AWS CLI en `/usr/local/aws` y crea el symlink `aws` en el directorio `/usr/local/bin`. Si se usa la opción `-b` para crear un symlink, no es necesario especificar el directorio de instalación

en la variable `$PATH` del usuario. Con esto, todos los usuarios deberían poder llamar a la AWS CLI al escribir `aws` desde cualquier directorio.

Para ver una explicación de las opciones `-i` y `-b`, use la opción `-h`:

```
$ ./awscli-bundle/install -h
```

# Instalación de la AWS CLI sin sudo (Linux, macOS, or Unix)

Si no tiene permisos de sudo o quiere instalar la AWS CLI solo para el usuario actual, puede utilizar una versión modificada de los comandos anteriores:

```
$ curl "https://s3.amazonaws.com/aws-cli/awscli-bundle.zip" -o "awscli-bundle.zip"
$ unzip awscli-bundle.zip
$ ./awscli-bundle/install -b ~/bin/aws
```

Esto instala la AWS CLI en la ubicación predeterminada (`~/.local/lib/aws`) y crea un vínculo simbólico (symlink) en `~/bin/aws`. Asegúrese de que `~/bin` está en la variable de entorno `PATH` para que el symlink funcione:

```
$ echo $PATH | grep ~/bin     // See if $PATH contains ~/bin (output will be empty if it
 doesn't)
$ export PATH=~/bin:$PATH     // Add ~/bin to $PATH if necessary
```

> **Tip**
>
> Para asegurarse de que la configuración de $PATH se conserva entre las sesiones, añada la línea `export` al perfil de shell (~/.profile, ~/.bash_profile, etc).

# Desinstalación

El instalador agregado no añade recursos fuera del directorio de instalación, salvo el symlink opcional. Por tanto, para desinstalar, solo es necesario eliminar esos dos elementos.

```
$ sudo rm -rf /usr/local/aws
$ sudo rm /usr/local/bin/aws
```

# Configuración de la AWS CLI

En esta sección se explica cómo configurar los ajustes que usa la AWS Command Line Interface al interactuar con AWS, como las credenciales de seguridad y la región predeterminada.

> Note
>
> La AWS CLI firma solicitudes en su nombre e incluye una fecha en la firma. Asegúrese de que la fecha y la hora se establezcan correctamente; si no, la fecha de la firma podría no coincidir con la fecha de la solicitud, y AWS rechazaría la solicitud.

Secciones



## Configuración rápida

Para el uso general, el comando `aws configure` es la forma más rápida de configurar la instalación de la AWS CLI.

```
$ aws configure
AWS Access Key ID [None]: AKIAIOSFODNN7EXAMPLE
AWS Secret Access Key [None]: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
Default region name [None]: us-west-2
Default output format [None]: json
```

La AWS CLI le solicitará cuatro datos. El ID de clave de acceso de AWS y la clave de acceso secreta de AWS son las credenciales de su cuenta.

### Para obtener acceso al ID de la clave de acceso y a la clave de acceso secreta de un usuario de IAM

Las claves de acceso constan de un ID de clave de acceso y una clave de acceso secreta, que se utilizan para firmar mediante programación las solicitudes que realiza a AWS. Si no tiene claves de acceso, puede crearlas desde la Consola de administración de AWS. Le recomendamos que utilice las claves de acceso de IAM en lugar de las claves de acceso de la cuenta raíz de AWS. IAM le permite controlar de forma segura el acceso a los servicios de AWS y a los recursos de su cuenta de AWS.

El único momento en que puede ver o descargar las claves de acceso secretas es cuando crea las claves. No puede recuperarlas más adelante. Sin embargo, puede crear nuevas claves de acceso en cualquier momento. También debe tener permisos para realizar las acciones de IAM requeridas. Para obtener más información, consulte Granting IAM User Permission to Manage Password Policy and Credentials en la Guía del usuario de IAM.

1. Abra la consola de IAM.
2. En el panel de navegación, seleccione Users.
3. Seleccione su nombre de usuario de IAM (no la casilla de verificación).
4. Elija la pestaña Security credentials y, a continuación, seleccione Create access key.
5. Para ver la nueva clave de acceso, elija Show. Sus credenciales tendrán el aspecto siguiente:

   - ID de clave de acceso: AKIAIOSFODNN7EXAMPLE
   - Clave de acceso secreta: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
6. Para descargar el par de claves, elija Download .csv file. Almacene las claves en un lugar seguro.

   Mantenga las claves en secreto para proteger su cuenta y no las envíe nunca por correo electrónico. No las comparta fuera de su organización, aunque reciba una petición que parezca provenir de AWS o Amazon.com. Nadie que represente legítimamente a Amazon le pedirá nunca su clave secreta.

Temas relacionados

- ¿Qué es IAM? en la Guía del usuario de IAM
- AWS Security Credentials en AWS General Reference

La región predeterminada es el nombre de la región a la que desea realizar llamadas de forma predeterminada. Suele ser la región más cercana a usted, pero puede ser cualquier región. Por ejemplo, escriba `us-west-2` para utilizar EE.UU. Oeste (Oregón).

> Note
>
> Debe especificar una región de AWS al usar la AWS CLI. Para obtener una lista de los servicios y las regiones disponibles, consulte Regiones y puntos de conexión. Los designadores de región que la AWS CLI utiliza son los mismos nombres que aparecen en las URL y los puntos de enlace de servicio de Consola de administración de AWS.

El formato de salida predeterminado puede ser `json`, `text` o `table`. Si no especifica ningún formato de salida, se usa `json`.

Si tiene varios perfiles, puede configurar perfiles adicionales con nombre usando la opción `--profile`.

```
$ aws configure --profile user2
AWS Access Key ID [None]: AKIAI44QH8DHBEXAMPLE
AWS Secret Access Key [None]: je7MtGbClwBF/2Zp9Utk/h3yCo8nvbEXAMPLEKEY
Default region name [None]: us-east-1
Default output format [None]: text
```

Para actualizar cualquiera de sus ajustes, solo tiene que ejecutar `aws configure` de nuevo e introducir nuevos valores, según proceda. En las secciones siguientes aparece más información sobre los archivos que crea `aws configure`, los ajustes adicionales y los perfiles con nombre.

# Ajustes de configuración y precedencia

La AWS CLI usa una cadena de provisión para buscar las credenciales de AWS en diversos lugares, entre otros, las variables de entorno del sistema o del usuario y los archivos de configuración de AWS.

La AWS CLI buscará credenciales y ajustes de configuración en el siguiente orden:

1. Opciones de línea de comandos: la región, el formato de salida y el perfil se pueden especificar como opciones de comandos para anular la configuración predeterminada.

2. Variables de entorno (p. 21): `AWS_ACCESS_KEY_ID`, `AWS_SECRET_ACCESS_KEY` y `AWS_SESSION_TOKEN`.
3. Archivo de credenciales de AWS: ubicado en `~/.aws/credentials` en Linux, macOS, or Unix, o en `C:\Users\`*`USERNAME`*` \.aws\credentials` en Windows. Este archivo puede contener varios perfiles con nombre, además de un perfil predeterminado.
4. Archivo de configuración de la CLI: normalmente se encuentra en `~/.aws/config` o en Linux, macOS, or Unix, o en `C:\Users\`*`USERNAME`*` \.aws\config` en Windows. Este archivo puede contener un perfil predeterminado, perfiles con nombre y parámetros de configuración específicos de la CLI para cada uno.
5. Credenciales de contenedor: las proporciona Amazon Elastic Container Service en las instancias de contenedor al asignar un rol a su tarea.
6. Credenciales de perfil de instancia: estas credenciales pueden utilizarse en instancias EC2 con un rol de instancia asignado, y se entregan a través del servicio de metadatos de Amazon EC2.

# Archivos de configuración y credenciales

La CLI almacena las credenciales especificadas con `aws configure` en un archivo local denominado `credentials` en una carpeta con el nombre `.aws` en su directorio de inicio. La ubicación del directorio de inicio varía, pero se puede hacer referencia a ella usando las variables de entorno `%UserProfile%` en Windows y `$HOME` o ~ (símbolo tilde) en sistemas tipo Unix.

Por ejemplo, en la siguiente lista aparecen los contenidos de la carpeta `.aws`:

Linux, macOS, or Unix

```
$ ls  ~/.aws
```

Windows

```
> dir "%UserProfile%\.aws"
```

Para separar los credenciales de opciones menos sensibles, la región y el formato de salida se almacenan en un archivo independiente denominado `config`, que está en la misma carpeta.

La ubicación de archivo predeterminada para el archivo de configuración puede sobrescribirse si se configura la variable de entorno AWS_CONFIG_FILE en otra ruta local. Para obtener más información, consulte Variables de entorno (p. 21).

### Almacenamiento de credenciales en la configuración

La AWS CLI también lee credenciales desde archivo de configuración. Si desea conservar todos los ajustes de su perfil en un único archivo, puede hacerlo. Si siempre hay credenciales en ambas ubicaciones para un perfil (digamos que ha utilizado `aws configure` para actualizar las claves del perfil), las claves que tienen prioridad son las del archivo de credenciales.
Si usa uno de los SDK además de la AWS CLI, puede que aparezcan advertencias adicionales si las credenciales no están almacenadas en su propio archivo.

Los archivos generados por la CLI para el perfil configurado en la sección anterior tienen el siguiente aspecto:

**`~/.aws/credentials`**

```
[default]
aws_access_key_id=AKIAIOSFODNN7EXAMPLE
```

```
aws_secret_access_key=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
```

**~/.aws/config**

```
[default]
region=us-west-2
output=json
```

Se admiten los siguientes ajustes.

aws_access_key_id: clave de acceso de AWS.

aws_secret_access_key: clave secreta de AWS.

aws_session_token: token de sesión de AWS. Un token de sesión solo es necesario si utiliza credenciales de seguridad temporales.

region: región de AWS.

output: formato de salida (json, texto, o tabla)

# Perfiles con nombre

La AWS CLI permite el almacenamiento de perfiles con nombre en los archivos de configuración y credenciales. Puede configurar perfiles adicionales usando `aws configure` con la opción `--profile` o añadiendo entradas a los archivos de configuración y credenciales.

El siguiente ejemplo muestra un archivo de credenciales con dos perfiles:

**~/.aws/credentials**

```
[default]
aws_access_key_id=AKIAIOSFODNN7EXAMPLE
aws_secret_access_key=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY

[user2]
aws_access_key_id=AKIAI44QH8DHBEXAMPLE
aws_secret_access_key=je7MtGbClwBF/2Zp9Utk/h3yCo8nvbEXAMPLEKEY
```

Cada perfil usa credenciales diferentes (puede que sean de dos usuarios de IAM diferentes) y también puede emplear diferentes regiones y formatos de salida.

**~/.aws/config**

```
[default]
region=us-west-2
output=json

[profile user2]
region=us-east-1
output=text
```

> **Important**
>
> El archivo de credenciales de AWS usa un formato de denominación diferente que el archivo de configuración de la CLI para perfiles con nombre. No incluya el prefijo "perfil" al configurar un perfil con nombre en el archivo de credenciales de AWS.

## Uso de perfiles con la AWS CLI

Para usar un perfil con nombre, añada la opción `--profile` a su comando. El siguiente ejemplo muestra instancias en ejecución con el perfil `user2` de la sección anterior.

```
$ aws ec2 describe-instances --profile user2
```

Si va a usar un perfil con nombre para varios comandos, puede evitar especificar el perfil en todos los comandos si configura la variable de entorno AWS_PROFILE en la línea de comandos:

Linux, macOS, or Unix

```
$ export AWS_PROFILE=user2
```

Windows

```
> set AWS_PROFILE=user2
```

La configuración de la variable de entorno cambia el perfil predeterminado hasta que finalice la sesión del shell, o hasta que otorgue a la variable a un valor diferente. Encontrará más información sobre las variables en la siguiente sección.

# Variables de entorno

Las variables de entorno sobrescriben los archivos de configuración y credenciales y pueden ser útiles para crear scripts o configurar temporalmente un perfil con nombre como predeterminado.

La AWS CLI admite las siguientes variables de entorno.

- `AWS_ACCESS_KEY_ID`: clave de acceso de AWS.
- `AWS_SECRET_ACCESS_KEY`: clave secreta de AWS. Las variables de acceso y clave secreta sobrescriben las credenciales almacenadas en los archivos de credenciales y configuración.
- `AWS_SESSION_TOKEN`: especifique un token de sesión si utiliza credenciales de seguridad temporales.
- `AWS_DEFAULT_REGION`: región de AWS. Esta variable sobrescribe la región predeterminada del perfil en uso, si está establecida.
- `AWS_DEFAULT_OUTPUT`: cambie el formato de salida de la AWS CLI a `json`, `text` o `table`.
- `AWS_PROFILE`: nombre del perfil de la CLI que se ha de usar. Puede ser el nombre de un perfil almacenado en un archivo de credenciales o configuración, o `default` para usar el perfil predeterminado.
- `AWS_CA_BUNDLE`: especifique la ruta a un paquete de certificados que desea utilizar para la validación de certificados HTTPS.
- `AWS_SHARED_CREDENTIALS_FILE`: cambie la ubicación del archivo que la AWS CLI utiliza para almacenar claves de acceso.
- `AWS_CONFIG_FILE`: cambie la ubicación del archivo que la AWS CLI utiliza para almacenar perfiles de configuración.

El siguiente ejemplo muestra cómo configurar las variables de entorno para el usuario predeterminado que aparece en secciones anteriores de esta guía.

Linux, macOS, or Unix

```
$ export AWS_ACCESS_KEY_ID=AKIAIOSFODNN7EXAMPLE
$ export AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
$ export AWS_DEFAULT_REGION=us-west-2
```

Windows

```
> set AWS_ACCESS_KEY_ID=AKIAIOSFODNN7EXAMPLE
> set AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
> set AWS_DEFAULT_REGION=us-west-2
```

# Opciones de la línea de comandos

La AWS CLI usa opciones largas de línea de comandos de estilo GNU precedidas de dos guiones. Las opciones de la línea de comandos se pueden utilizar para sobrescribir las opciones de configuración predeterminadas en una única operación, pero no se pueden usar para especificar credenciales.

Se puede especificar los siguientes ajustes en la línea de comandos.

--profile: nombre del perfil que se ha de usar, o "default" para usar el perfil predeterminado.

--region: región de AWS a la que llamar.

--output: formato de salida.

--endpoint-url: punto de enlace al que realizar la llamada. El punto de enlace puede ser la dirección de un proxy o una URL de punto de enlace para la región de AWS en uso. No es necesario especificar un punto de enlace para el uso normal, ya que la AWS CLI determina el punto de enlace al que se debe llamar en función de la región en uso.

Las opciones anteriores sobrescriben la configuración de perfil correspondiente para una única operación. Cada una toma una cadena de argumento con un espacio o signo igual ("=") que separa el argumento del nombre de la opción. Las comillas no son necesarias en torno al argumento a menos que la cadena del argumento contenga un espacio.

> Tip
>
> Puede usar la opción --profile con `aws configure` para configurar perfiles adicionales
>
> ```
> $ aws configure --profile profilename
> ```

Entre los usos habituales de las opciones de la línea de comandos se incluyen la comprobación de sus recursos en varias regiones y el cambio del formato de salida para obtener una mayor legibilidad o facilidad de uso al generar scripts. Por ejemplo, si no está seguro de en qué región se ejecuta su instancia, puede ejecutar el comando describe-instances en cada región hasta que la encuentre:

```
$ aws ec2 describe-instances --output table --region us-east-1
------------------
|DescribeInstances|
+-----------------+
$ aws ec2 describe-instances --output table --region us-west-1
------------------
|DescribeInstances|
+-----------------+
$ aws ec2 describe-instances --output table --region us-west-2
-----------------------------------------------------------------------------
|                            DescribeInstances                              |
```

```
+-------------------------------------------------------------------------+
||                              Reservations                               ||
|+-----------------------------------+-----------------------------------+|
||  OwnerId                          |  012345678901                     ||
||  ReservationId                    |  r-abcdefgh                       ||
|+-----------------------------------+-----------------------------------+|
|||                               Instances                               |||
||+-------------------------+-----------------------------------------------+||
|||  AmiLaunchIndex         |  0                                            |||
|||  Architecture           |  x86_64                                       |||
...
```

Los tipos de parámetros de opción de la línea de comandos (cadena, booleanos, etc.) se tratan en detalle en la sección Especificación de valores de parámetros para la AWS Command Line Interface (p. 38), que aparece más adelante en esta guía.

# Metadatos de instancia

Para usar la CLI desde una instancia EC2, cree un rol que tenga acceso a los recursos necesarios y asigne el rol a la instancia cuando se inicie. Inicie la instancia y compruebe si la AWS CLI ya está instalada (viene preinstalada en Amazon Linux).

Instale la AWS CLI si fuera necesario y configurar una región predeterminada para evitar tener que especificarla en cada comando. Puede definir la región usando `aws configure` sin introducir credenciales pulsando Intro dos veces para omitir los dos primeros comandos:

```
$ aws configure
AWS Access Key ID [None]: ENTER
AWS Secret Access Key [None]: ENTER
Default region name [None]: us-west-2
Default output format [None]: json
```

La AWS CLI se encargará de leer las credenciales desde los metadatos de la instancia. Para obtener más información, consulte Permitir el acceso a los recursos de AWS a aplicaciones que se ejecutan en instancias EC2 de Amazon en la Guía del usuario de IAM.

# Uso de un proxy HTTP

Si necesita acceder a AWS a través de servidores proxy, debe configurar las variables de entorno `HTTP_PROXY` y `HTTPS_PROXY` con direcciones IP para sus servidores proxy.

Linux, macOS, or Unix

```
$ export HTTP_PROXY=http://a.b.c.d:n
$ export HTTPS_PROXY=http://w.x.y.z:m
```

Windows

```
> set HTTP_PROXY=http://a.b.c.d:n
> set HTTPS_PROXY=http://w.x.y.z:m
```

En estos ejemplos, `http://a.b.c.d:n` y `http://w.x.y.z:m` son los puertos y direcciones IP para los servidores proxy HTTP y HTTPS.

## Autenticación en un proxy

La AWS CLI admite la autenticación básica de HTTP. Especifique un nombre de usuario y una contraseña en la URL del proxy como se indica a continuación:

Linux, macOS, or Unix

```
$ export HTTP_PROXY=http://username:password@a.b.c.d:n
$ export HTTPS_PROXY=http://username:password@w.x.y.z:m
```

Windows

```
> set HTTP_PROXY=http://username:password@a.b.c.d:n
> set HTTPS_PROXY=http://username:password@w.x.y.z:m
```

> **Note**
>
> La AWS CLI no admite servidores proxy NTLM. Si utiliza un proxy NTLM o Kerberos, puede conectarse a través de un proxy de autenticación como Cntlm.

## Uso de un proxy en instancias EC2

Si configura un proxy en una instancia EC2 lanzada con una función de IAM, también debe configurar la variable de entorno `NO_PROXY` con la dirección IP 169.254.169.254, de tal forma que la AWS CLI pueda acceder a Instance Meta Data Service (IMDS).

Linux, macOS, or Unix

```
$ export NO_PROXY=169.254.169.254
```

Windows

```
> set NO_PROXY=169.254.169.254
```

# Asumir un rol

Si desea que la AWS Command Line Interface use un rol, cree un perfil para dicho rol en el archivo `~/.aws/config`. En el siguiente ejemplo se muestra un perfil de rol con el nombre `marketingadmin`, que es asumido por el perfil predeterminado.

```
[profile marketingadmin]
role_arn = arn:aws:iam::123456789012:role/marketingadmin
source_profile = default
```

En este caso, el perfil predeterminado es un usuario de IAM con credenciales y permisos para asumir un rol denominado marketingadmin. Para acceder al rol, debe crear un perfil con nombre. En lugar de configurar este perfil con credenciales, debe especificar el ARN de la función y el nombre del perfil que tiene acceso a él.

Secciones

- Configuración y uso de un rol (p. 25)

# Configuración y uso de un rol

Cuando ejecuta comandos usando la función del rol, la AWS CLI emplea las credenciales del perfil de origen para llamar a AWS Security Token Service y asumir el rol especificado. El perfil de origen debe tener permiso para llamar a `sts:assume-role` con respecto al rol, y el rol debe tener una relación de confianza con el perfil de origen para permitir que lo asuma.

Puede crear un nuevo rol en IAM con los permisos que quiera que asuman los usuarios siguiendo el procedimiento de Creación de un rol para delegar permisos a un usuario de IAM en la Guía del usuario de AWS Identity and Access Management. Si el rol y el usuario de IAM de destino se encuentran en la misma cuenta, puede introducir su propio ID de cuenta al configurar la relación de confianza del rol.

Después de crear el rol, modifique la relación de confianza para permitir que el usuario de IAM lo asuma. En el siguiente ejemplo se muestra una relación de confianza que permite la asunción de un rol por parte de un usuario de IAM denominado `jonsmith`:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:user/jonsmith"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

A continuación, facilite al usuario de IAM permiso para asumir el rol. En el siguiente ejemplo se muestra una política de AWS Identity and Access Management que permite a un usuario de IAM asumir el rol `marketingadmin`:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "sts:AssumeRole",
      "Resource": "arn:aws:iam::123456789012:role/marketingadmin"
    }
  ]
}
```

El usuario no necesita tener permisos adicionales para ejecutar comandos desde el perfil del rol. Si desea que sus usuarios puedan acceder a los recursos de AWS sin usar el rol, aplique políticas adicionales insertadas o administradas para esos recursos.

Con el perfil de rol, los permisos de rol, la relación de confianza y los permisos de usuario aplicados, puede asumir el rol en la línea de comandos usando la opción `profile`, por ejemplo:

```
$ aws s3 ls --profile marketingadmin
```

Para usar el rol para varias llamadas, puede configurar la variable de entorno AWS_PROFILE para la sesión actual desde la línea de comandos:

Linux, macOS, or Unix

```
$ export AWS_PROFILE=marketingadmin
```

Windows

```
> set AWS_PROFILE=marketingadmin
```

Para obtener más información sobre cómo configurar usuarios y roles de IAM, consulte Usuarios y grupos y Roles en la Guía del usuario de AWS Identity and Access Management.

# Uso de la autenticación multifactor

Para aumentar la seguridad, puede pedir a los usuarios que proporcionen una clave única generada a partir de un dispositivo de autenticación multifactor o una aplicación móvil a la hora de realizar una llamada usando el perfil de rol.

En primer lugar, modifique la relación de confianza del rol para que requiera una autenticación multifactor:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": { "AWS": "arn:aws:iam::123456789012:user/jonsmith" },
      "Action": "sts:AssumeRole",
      "Condition": { "Bool": { "aws:MultiFactorAuthPresent": true } }
    }
  ]
}
```

A continuación, añada una línea al perfil del rol que especifique el ARN del dispositivo de MFA del usuario:

```
[profile marketingadmin]
role_arn = arn:aws:iam::123456789012:role/marketingadmin
source_profile = default
mfa_serial = arn:aws:iam::123456789012:mfa/jonsmith
```

La configuración `mfa_serial` puede ser un ARN, tal como se muestra, o el número de serie de un token de MFA de hardware.

# Roles entre cuentas

Puede habilitar que los usuarios de IAM asuman roles que pertenezcan a diferentes cuentas configurando el rol como rol entre cuentas. Al crear un rol, elija una de las opciones de Role for Cross-Account Access para establecer el tipo de rol y, de forma opcional, seleccione Require MFA. La opción Require MFA configura la condición correspondiente en la relación de confianza tal y como se describe en Uso de la autenticación multifactor (p. 26).

Si usa un identificador externo para ofrecer más control sobre quién puede asumir un rol entre cuentas, agregue un parámetro `external_id` al perfil del rol:

```
[profile crossaccountrole]
role_arn = arn:aws:iam::234567890123:role/xaccount
source_profile = default
mfa_serial = arn:aws:iam::123456789012:mfa/jonsmith
external_id = 123456
```

## Borrado de las credenciales almacenadas en memoria caché

Al asumir un rol, la AWS CLI almacena en caché localmente las credenciales temporales hasta que caducan. Si las credenciales temporales de su rol se revocan, puede eliminar la caché para forzar que la AWS CLI recupere nuevas credenciales.

Linux, macOS, or Unix

```
$ rm -r ~/.aws/cli/cache
```

Windows

```
> del /s /q %UserProfile%\.aws\cli\cache
```

# Finalización de comandos

En los sistemas tipo Unix, la AWS CLI incluye una característica de finalización de comandos que le permite utilizar la tecla de tabulación para completar un comando introducido parcialmente. Esta característica no se instala automáticamente, por lo que es necesario configurarla manualmente.

La configuración de la finalización de comandos requiere dos datos: el nombre del shell que esté usando y la ubicación del script aws_completer.

Finalización en Amazon Linux

La finalización de comandos se configura de forma predeterminada en las instancias que ejecuten Amazon Linux.

Secciones



## Identificación de su shell

Si no está seguro de qué shell utiliza, puede identificarlo con uno de los siguientes comandos:

echo $ SHELL: muestra el directorio de instalación del shell. Este suele coincidir con el shell en uso, a menos que iniciara otro shell tras iniciar sesión.

```
$ echo $SHELL
/bin/bash
```

ps: muestra los procesos en ejecución para el usuario actual. El shell ha de ser uno de ellos.

```
$ ps
  PID TTY          TIME CMD
 2148 pts/1    00:00:00 bash
 8756 pts/1    00:00:00 ps
```

# Localización del AWS Completer

La ubicación puede variar en función del método de instalación utilizado.

Package Manager: los programas como pip, yum, brew y apt-get suelen instalar el AWS completer (o un symlink al mismo) en una ubicación de ruta estándar. En este caso, `which` localizará el completer por usted.

```
$ which aws_completer
/usr/local/bin/aws_completer
```

Instalador agregado: si ha usado el instalador agregado según las instrucciones de la sección anterior, el AWS completer se encontrará en la subcarpeta bin del directorio de instalación.

```
$ ls /usr/local/aws/bin
activate
activate.csh
activate.fish
activate_this.py
aws
aws.cmd
aws_completer
...
```

Si no funciona nada más, puede utilizar `find` para buscar el AWS completer en todo el sistema de archivos.

```
$ find / -name aws_completer
/usr/local/aws/bin/aws_completer
```

# Habilitar la finalización de comandos

Ejecute un comando para habilitar la finalización de comandos. El comando que use para activar la finalización dependerá del shell que esté usando. Puede añadir el comando al archivo RC del shell para que se ejecute cada vez que abra un nuevo shell.

- bash: use el comando integrado `complete`.

  ```
  $ complete -C '/usr/local/bin/aws_completer' aws
  ```

  Añada el comando a `~/.bashrc` para que se ejecute cada vez que abra un nuevo shell. `~/.bash_profile` debe obtener `~/.bashrc` para asegurarse de que el comando se ejecute también en los shells de inicio de sesión.
- tcsh: la finalización en tcsh requiere un tipo de palabra y un patrón para definir el comportamiento de finalización.

  ```
  > complete aws 'p/*/`aws_completer`/'
  ```

Añada el comando a `~/.tschrc` para que se ejecute cada vez que abra un nuevo shell.

- zsh: origen `bin/aws_zsh_completer.sh`.

```
% source /usr/local/bin/aws_zsh_completer.sh
```

La AWS CLI usa la finalización automática de compatibilidad bash (`bashcompinit`) para zsh. Si desea obtener más información, consulte la parte superior de `aws_zsh_completer.sh`.

Añada el comando a `~/.zshrc` para que se ejecute cada vez que abra un nuevo shell.

# Comprobación de la finalización de comandos

Tras habilitar la finalización de comandos, escriba parcialmente un comando y pulse la tecla de tabulación para ver los comandos disponibles.

```
$ aws sTAB
s3              ses             sqs             sts             swf
s3api           sns             storagegateway  support
```

# Implementación de un entorno de desarrollo en Amazon EC2 con la AWS Command Line Interface

En este tutorial, se describe cómo configurar un entorno de desarrollo en Amazon EC2 utilizando la AWS CLI. Incluye una breve versión de las instrucciones de instalación y configuración, y puede ejecutarse de principio a fin en Windows, Linux, macOS, or Unix.

Pasos



## Instalar la AWS CLI

Puede instalar la AWS CLI con un instalador (Windows) o utilizando `pip`, un administrador de paquetes para Python.

### Windows

1. Descargue el instalador MSI.
   - Descarga del instalador MSI de la AWS CLI para Windows (64 bits)
   - Descarga del instalador MSI de la AWS CLI para Windows (32 bits)
2. Ejecute el instalador MSI descargado.
3. Siga las instrucciones que aparecen en pantalla.

### Linux, macOS, or Unix

Estos pasos requieren que tenga activa una instalación de Python 2 versión 2.6.5+ o Python 3 versión 3.3+. Si tiene algún problema al seguir estos pasos, consulte las instrucciones de instalación completas de la AWS Command Line Interface Guía del usuario.

1. Descargue y ejecute el script de instalación desde el sitio web de pip:

```
$ curl "https://bootstrap.pypa.io/get-pip.py" -o "get-pip.py"
$ python get-pip.py --user
```

2. Instalación de la AWS CLI con `pip`:

```
$ pip install awscli --user
```

# Configuración de la AWS CLI

Ejecute `aws configure` en la línea de comandos para configurar sus credenciales y ajustes.

```
$ aws configure
AWS Access Key ID [None]: AKIAIOSFODNN7EXAMPLE
AWS Secret Access Key [None]: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
Default region name [None]: us-west-2
Default output format [None]: json
```

La AWS CLI le pedirá la siguiente información:

- El ID de clave de acceso de AWS y la clave de acceso secreta de AWS: estas son las credenciales de su cuenta. Si no tiene claves, consulte ¿Cómo obtengo credenciales de seguridad? en la Referencia general de Amazon Web Services.
- Nombre de región predeterminado: este es el nombre de la región a la que desea realizar llamadas de forma predeterminada.

  Note

  Utilice `us-west-2` para este tutorial (la AMI que usaremos es específica de esta región). Posteriormente, podrá cambiar la región predeterminada ejecutando `aws configure` de nuevo.
- Formato de salida predeterminado: este formato puede ser json, texto o tabla. Si no especifica un formato de salida, se usará json.

Ejecute un comando para comprobar que sus credenciales están configuradas correctamente y que puede conectarse a AWS.

```
$ aws ec2 describe-regions --output table
----------------------------------------------------------
|                    DescribeRegions                     |
+--------------------------------------------------------+
||                        Regions                       ||
|+-----------------------------------+------------------+|
||             Endpoint              |   RegionName     ||
|+-----------------------------------+------------------+|
||  ec2.ap-south-1.amazonaws.com     |  ap-south-1      ||
||  ec2.eu-west-2.amazonaws.com      |  eu-west-2       ||
||  ec2.eu-west-1.amazonaws.com      |  eu-west-1       ||
||  ec2.ap-northeast-2.amazonaws.com |  ap-northeast-2  ||
||  ec2.ap-northeast-1.amazonaws.com |  ap-northeast-1  ||
||  ec2.sa-east-1.amazonaws.com      |  sa-east-1       ||
||  ec2.ca-central-1.amazonaws.com   |  ca-central-1    ||
||  ec2.ap-southeast-1.amazonaws.com |  ap-southeast-1  ||
||  ec2.ap-southeast-2.amazonaws.com |  ap-southeast-2  ||
||  ec2.eu-central-1.amazonaws.com   |  eu-central-1    ||
||  ec2.us-east-1.amazonaws.com      |  us-east-1       ||
||  ec2.us-east-2.amazonaws.com      |  us-east-2       ||
||  ec2.us-west-1.amazonaws.com      |  us-west-1       ||
||  ec2.us-west-2.amazonaws.com      |  us-west-2       ||
|+-----------------------------------+------------------+|
```

# Creación de un grupo de seguridad, un par de claves y un rol para la instancia EC2

El siguiente paso consiste en configurar requisitos previos para lanzar una instancia EC2 a la que se pueda acceder mediante SSH. Para obtener más información sobre las características de Amazon EC2, vaya a Guía del usuario de Amazon EC2 para instancias de Linux

Para crear un grupo de seguridad, un par de claves y una función

1. En primer lugar, cree un nuevo grupo de seguridad y añada una regla que admita tráfico entrante a través del puerto 22 para SSH. Tenga en cuenta el ID del grupo de seguridad para un uso posterior.

```
$ aws ec2 create-security-group --group-name devenv-sg --description "security group
 for development environment in EC2"
{
    "GroupId": "sg-b018ced5"
}
$ aws ec2 authorize-security-group-ingress --group-name devenv-sg --protocol tcp --port
 22 --cidr 0.0.0.0/0
```

2. Sustituya el intervalo de CIDR anterior por el intervalo desde el que se conectará para una mayor seguridad. Puede utilizar el comando `aws ec2 describe-security-groups` para admirar su trabajo.
3. A continuación, cree un par de claves, que le permitirá conectarse a la instancia.

```
$ aws ec2 create-key-pair --key-name devenv-key --query 'KeyMaterial' --output text >
 devenv-key.pem
```

Este comando guarda el contenido de la clave en un archivo denominado `devenv-key.pem`.

Windows

En Procesador de comandos de Windows, entrecomille las consultas con comillas dobles en lugar de comillas simples.

4. En Linux, también deberá cambiar el modo del archivo para que solo usted tenga acceso al archivo de clave.

```
$ chmod 400 devenv-key.pem
```

# Lanzamiento de la instancia y conexión a ella

Finalmente, estará listo para lanzar una instancia y conectarse a ella.

Para lanzar la instancia y conectarse a ella

1. Ejecute el siguiente comando, sustituyendo el ID del grupo de seguridad del paso anterior.

```
$ aws ec2 run-instances --image-id ami-6e1a0117 --security-group-ids sg-
b018ced5 --count 1 --instance-type t2.micro --key-name devenv-key --query
 'Instances[0].InstanceId'
"i-0787e4282810ef9cf"
```

El ID de imagen *ami-6e1a0117* especifica la Imagen de máquina de Amazon (AMI) que Amazon EC2 utiliza para arrancar la instancia. Puede encontrar ID de imagen para otras regiones y sistemas operativos en el asistente de lanzamiento de instancias de la consola de administración de Amazon EC2.

> **Note**
>
> Los tipos de instancia T2 requieren una VPC. Si no dispone de una VPC predeterminada, puede especificar una subred en una VPC personalizada con la opción `--subnet-id`. Si no dispone de ninguna VPC, elija un tipo de instancia diferente, como `t1.micro`.

2. La instancia tardará unos instantes en lanzarse. Una vez que la instancia se esté ejecutando, el siguiente comando recuperará la dirección IP pública que utilizará para conectarse a la instancia.

```
$ aws ec2 describe-instances --instance-ids "i-0787e4282810ef9cf" --query
 'Reservations[0].Instances[0].PublicIpAddress'
"54.183.22.255"
```

3. Para conectarse a la instancia, utilice la dirección IP pública y la clave privada con el programa de terminal que desee. En Linux, macOS, or Unix, podrá hacer esto desde la línea de comandos con el siguiente comando:

```
$ ssh -i devenv-key.pem ubuntu@54.183.22.255
```

   Si aparece un error como Permiso denegado (clave pública) cuando intente conectarse a su instancia, compruebe que lo siguiente sea correcto:

   - Clave: la clave especificada con la opción `-i` debe estar en la ruta indicada y debe ser la clave privada, no la clave pública. Los permisos de la clave deben limitarse al propietario.
   - Nombre de usuario: el nombre de usuario debe coincidir con el usuario asociado al par de claves de la instancia. Para las instancias de Ubuntu, será `ubuntu`. Para Amazon Linux, será `ec2-user`.
   - Instancia: la dirección IP pública o el nombre de DNS de la instancia. Compruebe que la dirección sea pública y que el puerto 22 esté abierto a su máquina local en el grupo de seguridad de la instancia.

   También puede utilizar la opción `-v` para ver más información relacionada con el error.

   > **SSH en Windows**
   >
   > En Windows, puede utilizar la aplicación de terminal PuTTY disponible aquí. Obtenga `putty.exe` y `puttygen.exe` en la página de descargas.
   > Utilice `puttygen.exe` para convertir su clave privada en un archivo `.ppk` que requiere PuTTY. Lance `putty.exe`, introduzca la dirección IP pública de la instancia en el campo Nombre de host y ajuste el tipo de conexión en SSH.
   > En el panel Categoría, vaya a Conexión > SSH > Autorización y haga clic en Buscar para seleccionar el archivo `.ppk`. A continuación, haga clic en Abrir para conectarse.

4. El terminal le pedirá que acepte la clave pública del servidor. Escriba `yes` y haga clic en Intro para completar la conexión.

Habrá configurado un grupo de seguridad, habrá creado un par de claves, habrá lanzado una instancia EC2 y se habrá conectado a ella sin necesidad de salir de la línea de comandos.

# Uso de la AWS Command Line Interface

En esta sección se presentan las características comunes y los patrones de llamada que se usan en la AWS Command Line Interface.

> Note
>
> La CLI de AWS realiza llamadas API a los servicios a través de HTTPS. Las conexiones de salida en el puerto TCP 443 deben estar habilitadas para realizar llamadas.

Temas



## Obtener ayuda con la AWS Command Line Interface

Para obtener ayuda al utilizar la AWS CLI, solo tiene que añadir `help` al final de un comando. Por ejemplo, el siguiente comando muestra la ayuda de las opciones generales de la AWS CLI generales y los comandos de nivel superior.

```
$ aws help
```

El siguiente comando muestra los subcomandos de Amazon EC2.

```
$ aws ec2 help
```

En el siguiente ejemplo se muestra la ayuda detallada de la operación `DescribeInstances` de EC2, que incluye descripciones de los parámetros de entrada, filtros y salida. Consulte la sección de ejemplos de la ayuda si no está seguro de la sintaxis de un comando.

```
$ aws ec2 describe-instances help
```

La ayuda de cada comando se divide en seis secciones:

Name: el nombre del comando.

```
NAME
```

```
       describe-instances -
```

Description: una descripción de la operación de la API que el comando invoca, extraída de la documentación de la API para el servicio del comando.

```
DESCRIPTION
       Describes one or more of your instances.

       If you specify one or more instance IDs, Amazon EC2 returns information
       for those instances. If you do not specify  instance  IDs,  Amazon  EC2
       returns  information  for  all  relevant  instances.  If you specify an
       instance ID that is not valid, an error is returned. If you specify  an
       instance  that  you  do  not  own,  it  is not included in the returned
       results.
...
```

Synopsis: muestra una lista del comando y sus opciones. Si una opción aparece entre corchetes, significa que es opcional, que tiene un valor predeterminado o que existe una opción alternativa que puede utilizarse en su lugar.

```
SYNOPSIS
            describe-instances
          [--dry-run | --no-dry-run]
          [--instance-ids <value>]
          [--filters <value>]
          [--cli-input-json <value>]
          [--starting-token <value>]
          [--page-size <value>]
          [--max-items <value>]
          [--generate-cli-skeleton]
```

`describe-instances` tiene un comportamiento predeterminado que describe todas las instancias en la cuenta y la región actuales. Si lo desea, puede especificar una lista de `instance-ids` para describir una o varias instancias. `dry-run` es un indicador booleano opcional que no adopta un valor. Para utilizar un indicador, especifique el valor que se muestra, en este caso `--dry-run` o en `--no-dry-run`. Del mismo modo, `--generate-cli-skeleton` no adopta un valor. Si existen condiciones para usar una opción, se deberían describir en la sección `OPTIONS` o mostrarse en los ejemplos.

Options: descripción de cada una de las opciones que aparecen en el resumen.

```
OPTIONS
       --dry-run | --no-dry-run (boolean)
          Checks whether you have the required  permissions  for  the  action,
          without actually making the request, and provides an error response.
          If you have the required permissions, the error response is  DryRun-
          Operation . Otherwise, it is UnauthorizedOperation .

       --instance-ids (list)
          One or more instance IDs.

          Default: Describes all your instances.
...
```

Examples: ejemplos que ilustran el uso del comando y sus opciones. Si no se proporciona un ejemplo de un comando o un caso de uso que necesite, solicítelo a través del enlace de comentarios de esta página o de la referencia de comandos de la CLI de AWS en la página de ayuda del comando.

```
    EXAMPLES
    To describe an Amazon EC2 instance
```

```
    Command:

    aws ec2 describe-instances --instance-ids i-5203422c

    To describe all instances with the instance type m1.small

    Command:

    aws ec2 describe-instances --filters "Name=instance-type,Values=m1.small"

    To describe all instances with a Owner tag

    Command:

    aws ec2 describe-instances --filters "Name=tag-key,Values=Owner"
...
```

Output: descripciones de cada uno de los campos y tipos de datos que se devuelven en la respuesta de AWS.

Para `describe-instances`, la salida es una lista de objetos de reserva, cada uno de los cuales contiene varios campos y objetos con información acerca de la instancia o las instancias asociadas al mismo. Esta información procede de la documentación de la API para el tipo de datos de reserva utilizado por Amazon EC2.

```
OUTPUT
       Reservations -> (list)
          One or more reservations.

          (structure)
              Describes a reservation.

              ReservationId -> (string)
                 The ID of the reservation.

              OwnerId -> (string)
                 The ID of the AWS account that owns the reservation.

              RequesterId -> (string)
                 The ID of the requester that launched the instances  on  your
                 behalf (for example, AWS Management Console or Auto Scaling).

              Groups -> (list)
                 One or more security groups.

                 (structure)
                     Describes a security group.

                     GroupName -> (string)
                        The name of the security group.

                     GroupId -> (string)
                        The ID of the security group.

              Instances -> (list)
                 One or more instances.

                 (structure)
                     Describes an instance.

                     InstanceId -> (string)
                        The ID of the instance.
```

```
                    ImageId -> (string)
                       The ID of the AMI used to launch the instance.

                    State -> (structure)
                       The current state of the instance.

                       Code -> (integer)
                           The  low  byte represents the state. The high byte
                           is an opaque internal value and should be ignored.
...
```

Cuando la CLI de AWS proporciona la salida en formato JSON, esta se convierte en una matriz de objetos de reserva como a continuación:

```
{
    "Reservations": [
        {
            "OwnerId": "012345678901",
            "ReservationId": "r-4c58f8a0",
            "Groups": [],
            "RequesterId": "012345678901",
            "Instances": [
                {
                    "Monitoring": {
                        "State": "disabled"
                    },
                    "PublicDnsName": "ec2-52-74-16-12.us-west-2.compute.amazonaws.com",
                    "State": {
                        "Code": 16,
                        "Name": "running"
                    },
...
```

Cada objeto de reserva tiene campos que describen la reserva y una matriz de objetos de instancia, cada uno de ellos con sus propios campos (p. ej. `PublicDnsName`) y objetos (por ejemplo `State`) que los describen.

Usuarios de Windows

Añada al resultado del comando de ayuda una barra vertical y `more` para ver las páginas del archivo de ayuda una a una. Pulse la barra de espacio o AvPág para avanzar por el documento y q para salir.

```
> aws ec2 describe-instances help | more
```

# Documentación de la AWS CLI

La AWS Command Line Interface Reference proporciona el contenido de todos los archivos de ayuda de comandos de la AWS CLI, compilados y presentados online para facilitar la navegación y la consulta en móviles, tabletas y equipos de escritorio.

A veces, los archivos de ayuda contienen enlaces que no se pueden seguir o cuyo contenido no se puede visualizar desde la vista de la línea de comandos pero sí en la referencia online de la AWS CLI.

# Documentación de la API

Todos los subcomandos de la AWS CLI corresponden a las llamadas realizadas a la API pública de un servicio. Cada servicio con una API pública, a su vez, tiene una serie de documentación de referencia de API que puede encontrarse en la página de inicio del servicio en el sitio web Documentación de AWS.

El contenido de una referencia de API varía en función de cómo se construye la API y del protocolo utilizado. Normalmente, una referencia de API contiene información detallada sobre las acciones compatibles con la API, los datos enviados a y desde el servicio, y posibles situaciones de error.

Secciones de documentación de la API

- Actions: información detallada de parámetros (incluidas las restricciones de longitud o contenido) y errores específicos de una acción. Las acciones corresponden a subcomandos en la AWS CLI.
- Data Types: puede contener información adicional sobre los datos de objetos devueltos por un subcomando.
- Common Parameters: información detallada sobre los parámetros que utilizan todas las acciones de un servicio.
- Common Errors: información detallada sobre los errores que devuelven todas las acciones de un servicio.

El nombre y la disponibilidad de cada sección puede variar en función del servicio.

CLI específicas de los servicios

Algunos servicios tienen una CLI distinta que existía antes de que se creara una única AWS CLI que funciona con todos ellos. Las CLI específicas de los servicios tienen documentación aparte, a la que se accede con un enlace a la página de documentación del servicio. La documentación de las CLI específicas de los servicios no se aplica a la AWS CLI.

# Estructura de comandos en la AWS Command Line Interface

La AWS CLI usa una estructura multiparte en la línea de comandos. Comienza con la llamada base a `aws`. La siguiente parte especifica un comando de nivel superior, que a menudo representa un servicio de AWS compatible en la AWS CLI. Cada servicio de AWS tiene subcomandos adicionales que especifican la operación que se ha de llevar a cabo. Las opciones generales de la CLI, o los parámetros específicos para una operación, se pueden especificar en la línea de comandos en cualquier orden. Si se especifica un parámetro exclusivo varias veces, solo se aplica el último valor.

```
$ aws <command> <subcommand> [options and parameters]
```

Los parámetros pueden tomar varios tipos de valores de entrada, como números, cadenas, listas, mapas y estructuras JSON.

# Especificación de valores de parámetros para la AWS Command Line Interface

Muchos parámetros son simples valores numéricos o de cadena, como el nombre del par de claves `my-key-pair` en el siguiente ejemplo:

```
$ aws ec2 create-key-pair --key-name my-key-pair
```

Las cadenas sin caracteres de espacio pueden entrecomillarse o no entrecomillarse. Sin embargo, las cadenas que incluyen uno o varios caracteres de espacio deben entrecomillarse. Utilice comillas simples (')

en Linux, macOS, or Unix y Windows PowerShell o utilice comillas dobles (") en el símbolo del sistema de Windows, tal y como se muestra en los siguientes ejemplos.

Windows PowerShell, Linux, macOS, or Unix

```
$ aws ec2 create-key-pair --key-name 'my key pair'
```

Procesador de comandos de Windows

```
> aws ec2 create-key-pair --key-name "my key pair"
```

También puede utilizar un signo de igual en lugar de un espacio. Normalmente, esto solo es necesario si el valor del parámetro comienza con un guion:

```
$ aws ec2 delete-key-pair --key-name=-mykey
```

Temas



# Tipos de parámetros comunes

En esta sección, se describen algunos de los tipos de parámetros comunes y el formato al que los servicios esperan que estos se ajusten. Si tiene problemas con el formato de un parámetro para un comando específico, consulte el manual escribiendo **`help`** después del nombre del comando, por ejemplo:

```
$ aws ec2 describe-spot-price-history help
```

La ayuda para cada subcomando describe su función, opciones, resultado y ejemplos. La sección de opciones incluye el nombre y la descripción de cada una de las opciones con el tipo de parámetro de la opción entre paréntesis.

Cadena: los parámetros de cadena pueden contener caracteres alfanuméricos, símbolos y espacios en blanco del conjunto de caracteres ASCII. Las cadenas que contienen espacios en blanco deben entrecomillarse. No se recomienda utilizar símbolos y espacios en blanco diferentes al carácter de espacio estándar, ya que esto podría causar problemas al utilizar la AWS CLI.

Algunos parámetros de cadena pueden aceptar datos binarios de un archivo. Consulte Archivos binarios (p. 43) para ver un ejemplo.

Marca temporal: las marcas temporales tienen un formato conforme a la norma ISO 8601. A veces se les denomina parámetros de tipo "DateTime" o "Date".

```
$ aws ec2 describe-spot-price-history --start-time 2014-10-13T19:00:00Z
```

Los formatos aceptados son:

- AAAA-MM-DDThh: mm: ss.sssTZD (UTC), por ejemplo, 2014-10-01T20:30:00.000Z
- AAAA-MM-DDThh: mm: ss.sssTZD (con compensación), por ejemplo, 2014-10-01T12:30:00.000-08:00

- AAAA-MM-DD, por ejemplo, 2014-10-01
- Tiempo Unix en segundos, por ejemplo, 1412195400

Lista: una o varias cadenas separadas por espacios.

```
$ aws ec2 describe-spot-price-history --instance-types m1.xlarge m1.medium
```

Valor booleano: marca binaria que activa o desactiva una opción. Por ejemplo, `ec2 describe-spot-price-history` cuenta con un parámetro de simulacro booleano que, cuando se especifica, valida el comando frente al servicio sin ejecutar realmente una consulta.

```
$ aws ec2 describe-spot-price-history --dry-run
```

El resultado indica si el comando tenía el formato correcto o no. Este comando también incluye una versión de no simulacro del parámetro que puede utilizarse para indicar de forma explícita que el comando debe ejecutarse con normalidad, aunque no es necesario incluirlo, ya que este es el comportamiento predeterminado.

Entero: un número entero sin firma.

```
$ aws ec2 describe-spot-price-history --max-items 5
```

Blob: objeto binario. Los parámetros blob toman una ruta hacia un archivo local que contiene datos binarios. La ruta no debe contener ningún identificador de protocolo como `http://` o `file://`.

El parámetro `--body` para `aws s3api put-object` es un blob:

```
$  aws s3api put-object --bucket my-bucket --key testimage.png --body /tmp/image.png
```

Mapa: una secuencia de pares clave-valor especificada en JSON o sintaxis abreviada (p. 53). En el siguiente ejemplo, se lee un elemento de una tabla de DynamoDB denominada my-table con un parámetro de mapa, `--key`. El parámetro especifica la clave principal denominada id con un valor numérico de 1 en una estructura JSON anidada.

```
$ aws dynamodb get-item --table-name my-table --key '{"id": {"N":"1"}}'
{
    "Item": {
        "name": {
            "S": "John"
        },
        "id": {
            "N": "1"
        }
    }
}
```

En la siguiente sección, se explican los argumentos de JSON con más detalle.

# Uso de JSON para parámetros

JSON resulta útil para especificar parámetros de línea de comandos complejos. Por ejemplo, el siguiente comando mostrará todas las instancias EC2 que tienen un tipo de instancia de `m1.small` o `m1.medium` que también se encuentran en la zona de disponibilidad `us-west-2c`.

```
$ aws ec2 describe-instances --filters "Name=instance-type,Values=t2.micro,m1.medium"
 "Name=availability-zone,Values=us-west-2c"
```

En el siguiente ejemplo, se especifica la lista equivalente de filtros en una matriz JSON. Los corchetes se utilizan para crear una matriz de objetos JSON separados por comas. Cada objeto es una lista de pares clave-valor separados por comas ("Name" y "Values" son claves en esta instancia).

Tenga en cuenta que el valor situado a la derecha de la clave "Values" es una matriz en sí. Esto es necesario, aunque la matriz solo contenga una cadena de valor.

```
[
  {
    "Name": "instance-type",
    "Values": ["t2.micro", "m1.medium"]
  },
  {
    "Name": "availability-zone",
    "Values": ["us-west-2c"]
  }
]
```

Por el contrario, los paréntesis de los extremos solo son necesarios si se especifica más de un filtro. La versión de un solo filtro del comando anterior, con formato JSON, tiene este aspecto:

```
$ aws ec2 describe-instances --filters '{"Name": "instance-type", "Values": ["t2.micro",
 "m1.medium"]}'
```

Algunas operaciones necesitan que los datos estén en formato JSON. Por ejemplo, para pasar parámetros al parámetro `--block-device-mappings` en el comando `ec2 run-instances`, la información del dispositivo de bloque debe estar en formato JSON.

En este ejemplo, se muestra el JSON para especificar la asignación de un único dispositivo Elastic Block Store de 20 GiB a `/dev/sdb` en la instancia que se va a lanzar.

```
{
  "DeviceName": "/dev/sdb",
  "Ebs": {
    "VolumeSize": 20,
    "DeleteOnTermination": false,
    "VolumeType": "standard"
  }
}
```

Para adjuntar varios dispositivos, indique los objetos en una matriz como en el siguiente ejemplo.

```
[
  {
    "DeviceName": "/dev/sdb",
    "Ebs": {
      "VolumeSize": 20,
      "DeleteOnTermination": false,
      "VolumeType": "standard"
    }
  },
  {
    "DeviceName": "/dev/sdc",
    "Ebs": {
      "VolumeSize": 10,
      "DeleteOnTermination": true,
```

```
      "VolumeType": "standard"
    }
  }
]
```

Puede introducir el JSON directamente en la línea de comandos (consulte Entrecomillado de cadenas (p. 42)) o guardarlo en un archivo al que se haga referencia desde la línea de comandos (consulte Carga de parámetros desde un archivo (p. 42)).

Al pasar grandes bloques de datos, podría resultarle más sencillo guardar el JSON en un archivo y hacer referencia a él desde la línea de comandos. Los datos JSON de un archivo son más fáciles de leer, editar y compartir con otros. Esta técnica se describe en la siguiente sección.

Para obtener más información sobre JSON, consulte Wikipedia - JSON y RFC4627 - The application/json Media Type for JSON.

# Entrecomillado de cadenas

El modo en que introduzca parámetros con formato JSON en la línea de comandos varía en función de su sistema operativo. Linux, macOS, or Unix y Windows PowerShell utilizan comillas simples (') para entrecomillar la estructura de datos JSON, como en el siguiente ejemplo:

```
$ aws ec2 run-instances --image-id ami-05355a6c --block-device-mappings '[{"DeviceName":"/
dev/sdb","Ebs":{"VolumeSize":20,"DeleteOnTermination":false,"VolumeType":"standard"}}]'
```

Por el contrario, el símbolo del sistema de Windows utiliza comillas dobles (") para entrecomillar la estructura de datos JSON. Además, se requiere un carácter de escape de barra inversa (\) para cada comilla doble (") dentro de la propia estructura de datos JSON, como en el siguiente ejemplo:

```
> aws ec2 run-instances --image-id ami-05355a6c --block-device-mappings "[{\"DeviceName
\":\"/dev/sdb\",\"Ebs\":{\"VolumeSize\":20,\"DeleteOnTermination\":false,\"VolumeType\":
\"standard\"}}]"
```

Windows PowerShell requiere comillas simples (') para entrecomillar la estructura de datos JSON, así como una barra inversa (\) para aplicar escape a cada comilla doble (") dentro de la estructura JSON, como en el siguiente ejemplo:

```
> aws ec2 run-instances --image-id ami-05355a6c --block-device-mappings '[{\"DeviceName
\":\"/dev/sdb\",\"Ebs\":{\"VolumeSize\":20,\"DeleteOnTermination\":false,\"VolumeType\":
\"standard\"}}]'
```

Si el valor de un parámetro es un documento JSON en sí, aplique escape a las comillas del documento JSON incluido. Por ejemplo, el parámetro `attribute` para `aws sqs create-queue` puede tomar una clave `RedrivePolicy`. El valor de `RedrivePolicy` es un documento JSON, al que se debe aplicar escape:

```
$ aws sqs create-queue --queue-name my-queue --
attributes '{ "RedrivePolicy":"{\"deadLetterTargetArn\":\"arn:aws:sqs:us-
west-2:0123456789012:deadletter\", \"maxReceiveCount\":\"5\"}"}'
```

# Carga de parámetros desde un archivo

Para evitar la necesidad de aplicar escape a cadenas JSON en la línea de comandos, cargue el JSON desde un archivo. Cargue los parámetros desde un archivo local proporcionando la ruta al archivo con el prefijo `file://`, como en los siguientes ejemplos.

Linux, macOS, or Unix

```
// Read from a file in the current directory
$ aws ec2 describe-instances --filters file://filter.json

// Read from a file in /tmp
$ aws ec2 describe-instances --filters file:///tmp/filter.json
```

Windows

```
// Read from a file in C:\temp
> aws ec2 describe-instances --filters file://C:\temp\filter.json
```

La opción de prefijo `file://` admite ampliaciones tipo Unix, incluidas '~ /', './', y '../'. En Windows, la expresión '~/' se amplía a su directorio de usuarios, guardado en la variable de entorno %USERPROFILE %. Por ejemplo, en Windows 7 normalmente tendría un directorio de usuarios en `C:\Users\`*`User Name`*`\`.

A los documentos JSON que se proporcionan como el valor de una clave de parámetro se les debe aplicar escape:

```
$ aws sqs create-queue --queue-name my-queue --attributes file://attributes.json
```

attributes.json

```
{
  "RedrivePolicy":"{\"deadLetterTargetArn\":\"arn:aws:sqs:us-
west-2:0123456789012:deadletter\", \"maxReceiveCount\":\"5\"}"
}
```

## Archivos binarios

Para los comandos que toman datos binarios como un parámetro, especifique que los datos son contenido binario utilizando el prefijo `fileb://`. Los comandos que aceptan datos binarios son:

- Parámetro **`aws ec2 run-instances`** – `--user-data`.
- Parámetro **`aws s3api put-object`** – `--sse-customer-key`.
- Parámetro **`aws kms decrypt`** – `--ciphertext-blob`.

En el siguiente ejemplo, se genera una clave AES binaria de 256 bits utilizando una herramienta de línea de comandos de Linux y después se proporciona dicha clave a Amazon S3 para cifrar un archivo cargado en el lado del servidor:

```
$ dd if=/dev/urandom bs=1 count=32 > sse.key
32+0 records in
32+0 records out
32 bytes (32 B) copied, 0.000164441 s, 195 kB/s
$ aws s3api put-object --bucket my-bucket --key test.txt --body test.txt --sse-customer-key
 fileb://sse.key --sse-customer-algorithm AES256
{
    "SSECustomerKeyMD5": "iVg8oWa8sy714+FjtesrJg==",
    "SSECustomerAlgorithm": "AES256",
    "ETag": "\"a6118e84b76cf98bf04bbe14b6045c6c\""
}
```

## Archivos remotos

La AWS CLI también permite cargar parámetros desde un archivo ubicado en Internet con una URL `http://` o `https://`. En el siguiente ejemplo, se hace referencia a un archivo de un bucket Amazon S3. Esto le permite acceder a archivos de parámetros desde cualquier equipo, pero requiere que el archivo esté guardado en una ubicación de acceso público.

```
$ aws ec2 run-instances --image-id ami-a13d6891 --block-device-mappings http://my-bucket.s3.amazonaws.com/filename.json
```

En los ejemplos anteriores, el archivo `filename.json` contiene los siguientes datos JSON.

```
[
  {
    "DeviceName": "/dev/sdb",
    "Ebs": {
      "VolumeSize": 20,
      "DeleteOnTermination": false,
      "VolumeType": "standard"
    }
  }
]
```

Por ver otro ejemplo que haga referencia a un archivo que contenga parámetros con formato JSON más complejos, consulte Definición de una política de IAM para un usuario de IAM (p. 75).

# Parámetros Generate CLI Skeleton y CLI Input JSON

La mayoría de comandos de la CLI de AWS permiten usar los parámetros `--generate-cli-skeleton` y `--cli-input-json`, que puede usar para almacenar parámetros en JSON y leerlos desde un archivo en lugar de escribirlos en la línea de comandos.

Generate CLI Skeleton produce un resultado JSON en el que se describen todos los parámetros que se pueden especificar para la operación.

Para utilizar --generate-cli-skeleton con aws ec2 run-instances

1. Ejecute el comando run-instances con la opción `--generate-cli-skeleton` para ver el esqueleto JSON.

```
$ aws ec2 run-instances --generate-cli-skeleton
{
    "DryRun": true,
    "ImageId": "",
    "MinCount": 0,
    "MaxCount": 0,
    "KeyName": "",
    "SecurityGroups": [
        ""
    ],
    "SecurityGroupIds": [
        ""
    ],
    "UserData": "",
```

```
    "InstanceType": "",
    "Placement": {
        "AvailabilityZone": "",
        "GroupName": "",
        "Tenancy": ""
    },
    "KernelId": "",
    "RamdiskId": "",
    "BlockDeviceMappings": [
        {
            "VirtualName": "",
            "DeviceName": "",
            "Ebs": {
                "SnapshotId": "",
                "VolumeSize": 0,
                "DeleteOnTermination": true,
                "VolumeType": "",
                "Iops": 0,
                "Encrypted": true
            },
            "NoDevice": ""
        }
    ],
    "Monitoring": {
        "Enabled": true
    },
    "SubnetId": "",
    "DisableApiTermination": true,
    "InstanceInitiatedShutdownBehavior": "",
    "PrivateIpAddress": "",
    "ClientToken": "",
    "AdditionalInfo": "",
    "NetworkInterfaces": [
        {
            "NetworkInterfaceId": "",
            "DeviceIndex": 0,
            "SubnetId": "",
            "Description": "",
            "PrivateIpAddress": "",
            "Groups": [
                ""
            ],
            "DeleteOnTermination": true,
            "PrivateIpAddresses": [
                {
                    "PrivateIpAddress": "",
                    "Primary": true
                }
            ],
            "SecondaryPrivateIpAddressCount": 0,
            "AssociatePublicIpAddress": true
        }
    ],
    "IamInstanceProfile": {
        "Arn": "",
        "Name": ""
    },
    "EbsOptimized": true
}
```

2. Exporte el resultado a un archivo para guardar el esqueleto en el disco local:

```
$ aws ec2 run-instances --generate-cli-skeleton > ec2runinst.json
```

3. Abra el esqueleto en un editor de texto y elimine los parámetros que no vaya a usar:

```
{
    "DryRun": true,
    "ImageId": "",
    "KeyName": "",
    "SecurityGroups": [
        ""
    ],
    "InstanceType": "",
    "Monitoring": {
        "Enabled": true
    }
}
```

Deje el parámetro `DryRun` como true para usar la función de simulacro de EC2, que le permite probar su configuración sin necesidad de crear recursos.

4. Rellene los valores para el tipo de instancia, el nombre de la clave, el grupo de seguridad y la AMI en la región predeterminada. En este ejemplo, `ami-dfc39aef` es una imagen Amazon Linux de 64 bits en la región `us-west-2`.

```
{
    "DryRun": true,
    "ImageId": "ami-dfc39aef",
    "KeyName": "mykey",
    "SecurityGroups": [
        "my-sg"
    ],
    "InstanceType": "t2.micro",
    "Monitoring": {
        "Enabled": true
    }
}
```

5. Pase la configuración JSON al parámetro `--cli-input-json` con el prefijo `file://`:

```
$ aws ec2 run-instances --cli-input-json file://ec2runinst.json
A client error (DryRunOperation) occurred when calling the RunInstances operation:
 Request would have succeeded, but DryRun flag is set.
```

El simulacro indica que el error JSON está formado correctamente y que los valores de los parámetros son válidos. Si en los resultados apareciera cualquier otro problema, corríjalo y repita el paso anterior hasta que el se muestre el error de simulacro.

6. Especifique el parámetro DryRun como false para deshabilitar la función de simulacro.

```
{
    "DryRun": false,
    "ImageId": "ami-dfc39aef",
    "KeyName": "mykey",
    "SecurityGroups": [
        "my-sg"
    ],
    "InstanceType": "t2.micro",
    "Monitoring": {
        "Enabled": true
    }
}
```

7. Ejecute el comando `run-instances` de nuevo para iniciar una instancia:

```
$ aws ec2 run-instances --cli-input-json file://ec2runinst.json
```

```
{
    "OwnerId": "123456789012",
    "ReservationId": "r-d94a2b1",
    "Groups": [],
    "Instances": [
...
```

# Control de la salida de comandos desde la AWS Command Line Interface

En esta sección se describen las diferentes maneras en las que puede controlar la salida desde la AWS CLI.

Temas



## Cómo elegir el formato de salida

La AWS CLI admite tres formatos de salida diferentes:

- JSON (json)
- Texto delimitado por tabulaciones (texto)
- Tabla con formato ASCII (tabla)

Tal como se explica en el tema de configuración (p. 17), el formato de salida se puede especificar de tres formas diferentes:

- Con la opción `output` en el archivo de configuración. El siguiente ejemplo establece la salida como `text`:

```
[default]
output=text
```

- Usando de la variable de entorno `AWS_DEFAULT_OUTPUT`. Por ejemplo:

```
$ export AWS_DEFAULT_OUTPUT="table"
```

- Usando la opción `--output` de la línea de comandos. Por ejemplo:

```
$ aws swf list-domains --registration-status REGISTERED --output text
```

> **Note**
>
> Si el formato de salida se especifica de varias maneras, se aplican las normas de precedencia de la AWS CLI (p. 18) habituales. Por ejemplo, si se usa la variable de entorno `AWS_DEFAULT_OUTPUT`, se anula cualquier valor establecido en el archivo de configuración con

`output`, y un valor transferido a un comando de la AWS CLI con `--output` anula cualquier valor establecido en el entorno o en el archivo de configuración.

JSON es la mejor opción para administrar la salida de forma programada con varios lenguajes o `jq` (un procesador JSON en la línea de comandos). El formato de tabla es fácil de leer para las personas, el formato de texto funciona bien con las herramientas de procesamiento de texto tradicionales de Unix, como por ejemplo `sed`, `grep` y `awk`, además de scripts de Windows PowerShell.

# Cómo filtrar la salida con la opción `--query`

La AWS CLI ofrece capacidades de filtrado de salida integradas con la opción `--query`. Para demostrar cómo funciona, en primer lugar empezaremos con el valor de salida JSON predeterminado, que describe dos volúmenes EBS (Elastic Block Storage) adjuntos a instancias EC2 independientes.

```
$ aws ec2 describe-volumes
{
    "Volumes": [
        {
            "AvailabilityZone": "us-west-2a",
            "Attachments": [
                {
                    "AttachTime": "2013-09-17T00:55:03.000Z",
                    "InstanceId": "i-a071c394",
                    "VolumeId": "vol-e11a5288",
                    "State": "attached",
                    "DeleteOnTermination": true,
                    "Device": "/dev/sda1"
                }
            ],
            "VolumeType": "standard",
            "VolumeId": "vol-e11a5288",
            "State": "in-use",
            "SnapshotId": "snap-f23ec1c8",
            "CreateTime": "2013-09-17T00:55:03.000Z",
            "Size": 30
        },
        {
            "AvailabilityZone": "us-west-2a",
            "Attachments": [
                {
                    "AttachTime": "2013-09-18T20:26:16.000Z",
                    "InstanceId": "i-4b41a37c",
                    "VolumeId": "vol-2e410a47",
                    "State": "attached",
                    "DeleteOnTermination": true,
                    "Device": "/dev/sda1"
                }
            ],
            "VolumeType": "standard",
            "VolumeId": "vol-2e410a47",
            "State": "in-use",
            "SnapshotId": "snap-708e8348",
            "CreateTime": "2013-09-18T20:26:15.000Z",
            "Size": 8
        }
    ]
}
```

En primer lugar, podemos mostrar solo el primer volumen de la lista `Volumes` con el siguiente comando.

```
$ aws ec2 describe-volumes --query 'Volumes[0]'
{
```

```
    "AvailabilityZone": "us-west-2a",
    "Attachments": [
        {
            "AttachTime": "2013-09-17T00:55:03.000Z",
            "InstanceId": "i-a071c394",
            "VolumeId": "vol-e11a5288",
            "State": "attached",
            "DeleteOnTermination": true,
            "Device": "/dev/sda1"
        }
    ],
    "VolumeType": "standard",
    "VolumeId": "vol-e11a5288",
    "State": "in-use",
    "SnapshotId": "snap-f23ec1c8",
    "CreateTime": "2013-09-17T00:55:03.000Z",
    "Size": 30
}
```

Ahora, utilizaremos la notación comodín `[*]` para iterar por toda la lista y filtrar tres elementos: `VolumeId`, `AvailabilityZone` y `Size`. Tenga en cuenta que la notación diccionario requiere que proporcione un alias para cada clave, como por ejemplo: {Alias1:Key1,Alias2:Key2}. Un diccionario es inherentemente desordenado, por lo que la ordenación de los alias de clave dentro de una estructura podría no resultar coherente en algunos casos.

```
$ aws ec2 describe-volumes --query 'Volumes[*].{ID:VolumeId,AZ:AvailabilityZone,Size:Size}'
[
    {
        "AZ": "us-west-2a",
        "ID": "vol-e11a5288",
        "Size": 30
    },
    {
        "AZ": "us-west-2a",
        "ID": "vol-2e410a47",
        "Size": 8
    }
]
```

En la notación diccionario, también puede utilizar claves encadenadas como, por ejemplo, `key1.key2[0].key3`, para filtrar elementos profundamente anidados en la estructura. El ejemplo siguiente ilustra este caso con la clave `Attachments[0].InstanceId`, con el simple alias `InstanceId`.

```
$ aws ec2 describe-volumes --query 'Volumes[*].
{ID:VolumeId,InstanceId:Attachments[0].InstanceId,AZ:AvailabilityZone,Size:Size}'
[
    {
        "InstanceId": "i-a071c394",
        "AZ": "us-west-2a",
        "ID": "vol-e11a5288",
        "Size": 30
    },
    {
        "InstanceId": "i-4b41a37c",
        "AZ": "us-west-2a",
        "ID": "vol-2e410a47",
        "Size": 8
    }
]
```

También puede filtrar varios elementos con la notación en lista: `[key1, key2]`. Así se formatearán todos los atributos filtrados en una única lista ordenada por objeto, independientemente del tipo.

```
$ aws ec2 describe-volumes --query 'Volumes[*].[VolumeId, Attachments[0].InstanceId,
 AvailabilityZone, Size]'
[
    [
        "vol-e11a5288",
        "i-a071c394",
        "us-west-2a",
        30
    ],
    [
        "vol-2e410a47",
        "i-4b41a37c",
        "us-west-2a",
        8
    ]
]
```

Para filtrar los resultados por el valor de un campo específico, use el operador JMESPath "?" . El siguiente ejemplo de consulta devuelve únicamente volúmenes de la zona de disponibilidad us-west-2a:

```
$ aws ec2 describe-volumes --query 'Volumes[?AvailabilityZone==`us-west-2a`]'
```

> **Note**
>
> Al especificar un valor literal como "us-west-2" en una expresión de consulta JMESPath, debe rodear el valor de acentos graves (') para que se pueda leer correctamente.

En combinación con las tres formatos de salida que se explicarán con más detalle en las siguientes secciones, la opción `--query` es una potente herramienta que se puede utilizar para personalizar el contenido y el estilo de las salidas. Para obtener más ejemplos y todas las especificaciones de JMESPath, la biblioteca de procesamiento JSON subyacente, visite http://jmespath.org/specification.html.

# Formato de salida JSON

JSON es el formato de salida predeterminado de la AWS CLI. La mayoría de lenguajes pueden decodificar fácilmente cadenas JSON con funciones integradas o con bibliotecas disponibles públicamente. Como se muestra en el tema anterior, además de los ejemplos de salida, la opción `--query` proporciona métodos eficaces de filtrar y formatear las salidas de la AWS CLI con formato JSON. Si necesita funciones más avanzadas, que podrían no ser posibles con `--query`, puede recurrir a `jq`, un procesador JSON de línea de comandos. Puede descargarlo y consultar el tutorial oficial en: http://stedolan.github.io/jq/.

# Formato de salida de texto

El formato de texto organiza la salida de la AWS CLI en líneas delimitadas por tabulaciones. Se combina bien con herramientas de texto tradicionales de Unix como `grep`, `sed` y `awk`, además de Windows PowerShell.

El formato de salida de texto sigue la estructura básica que se muestran a continuación. Las columnas se ordenan alfabéticamente según los nombres de las claves correspondientes del objeto JSON subyacente.

```
IDENTIFIER  sorted-column1 sorted-column2
IDENTIFIER2 sorted-column1 sorted-column2
```

A continuación figura un ejemplo de una salida de texto.

```
$ aws ec2 describe-volumes --output text
```

```
VOLUMES us-west-2a      2013-09-17T00:55:03.000Z        30      snap-f23ec1c8   in-use
 vol-e11a5288    standard
ATTACHMENTS     2013-09-17T00:55:03.000Z        True    /dev/sda1       i-a071c394
 attached        vol-e11a5288
VOLUMES us-west-2a      2013-09-18T20:26:15.000Z        8       snap-708e8348   in-use
 vol-2e410a47    standard
ATTACHMENTS     2013-09-18T20:26:16.000Z        True    /dev/sda1       i-4b41a37c
 attached        vol-2e410a47
```

Recomendamos encarecidamente que la salida de texto se use junto con la opción *`--query`* para garantizar un comportamiento coherente. Esto se debe a que el formato de texto ordena alfabéticamente las columnas de salida, y los recursos similares no siempre tienen el mismo conjunto de claves. Por ejemplo, una representación JSON de una instancia EC2 de Linux puede tener elementos que no están presentes en la representación JSON de una instancia de Windows, o viceversa. Además, los recursos pueden tener elementos de valor de clave añadidos o eliminados en futuras actualizaciones, alterando el orden de las columnas. Aquí es donde `--query` incrementa la funcionalidad de la salida de texto para permitir un control total sobre el formato de salida. En el ejemplo siguiente, el comando selecciona previamente qué elementos mostrar y define el orden de las columnas con la notación en lista `[key1, key2, ...]`. Esto ofrece a los usuarios la plena confianza de que los valores de clave correctos se mostrarán siempre en la columna prevista. Por último, tenga en cuenta que la AWS CLI devuelve el valor "None" para las claves que no existen.

```
$ aws ec2 describe-volumes --query 'Volumes[*].[VolumeId, Attachments[0].InstanceId,
 AvailabilityZone, Size, FakeKey]' --output text
vol-e11a5288    i-a071c394      us-west-2a      30      None
vol-2e410a47    i-4b41a37c      us-west-2a      8       None
```

A continuación se muestra un ejemplo de cómo `grep` y `awk` se pueden utilizar junto con una salida de texto a partir del comando `aws ec2 describe-instances`. El primer comando muestra la zona de disponibilidad, el estado y el ID de instancia de cada instancia en la salida de texto. El segundo comando devuelve únicamente los ID de instancia de todas las instancias en ejecución en la zona de disponibilidad `us-west-2a`.

```
$ aws ec2 describe-instances --query 'Reservations[*].Instances[*].
[Placement.AvailabilityZone, State.Name, InstanceId]' --output text
us-west-2a      running i-4b41a37c
us-west-2a      stopped i-a071c394
us-west-2b      stopped i-97a217a0
us-west-2a      running i-3045b007
us-west-2a      running i-6fc67758

$ aws ec2 describe-instances --query 'Reservations[*].Instances[*].
[Placement.AvailabilityZone, State.Name, InstanceId]' --output text | grep us-west-2a |
 grep running | awk '{print $3}'
i-4b41a37c
i-3045b007
i-6fc67758
```

En el siguiente comando se muestra un ejemplo similar para todas las instancias detenidas y da un paso más para automatizar los tipos de instancias cambiantes para cada instancia detenida.

```
$ aws ec2 describe-instances --query 'Reservations[*].Instances[*].[State.Name,
 InstanceId]' --output text |
> grep stopped |
> awk '{print $2}' |
> while read line;
> do aws ec2 modify-instance-attribute --instance-id $line --instance-type '{"Value":
 "m1.medium"}';
> done
```

La salida de texto también resulta útil en Windows PowerShell. Dado que la salida de texto de la AWS CLI está delimitada por tabulaciones, es fácil dividirla en una matriz de PowerShell utilizando el delimitador `` `t ``. El siguiente comando muestra el valor de la tercera columna (`InstanceId`) si la primera columna (`AvailabilityZone`) coincide con `us-west-2a`.

```
> aws ec2 describe-instances --query 'Reservations[*].Instances[*].
[Placement.AvailabilityZone, State.Name, InstanceId]' --output text |
%{if ($_.split("`t")[0] -match "us-west-2a") { $_.split("`t")[2]; } }
i-4b41a37c
i-a071c394
i-3045b007
i-6fc67758
```

# Formato de salida de tabla

El formato `table` produce representaciones legibles para los humanos de la salida de la AWS CLI. A continuación se muestra un ejemplo:

```
$ aws ec2 describe-volumes --output table
-------------------------------------------------------------------------------------------------------------
|                                                DescribeVolumes
                        |
+-----------------------------------------------------------------------------------------------------------
+
||                                                   Volumes
                         ||
|+------------------+---------------------------+-------+----------------+---------
+----------------+--------------+|
|| AvailabilityZone |        CreateTime         | Size  |  SnapshotId    |  State  |
 VolumeId     | VolumeType   ||
|+------------------+---------------------------+-------+----------------+---------
+----------------+--------------+|
||  us-west-2a      |  2013-09-17T00:55:03.000Z |  30   |  snap-f23ec1c8 |  in-use |  vol-
e11a5288  |  standard    ||
|+------------------+---------------------------+-------+----------------+---------
+----------------+--------------+|
|||                                                Attachments
                        |||
||+--------------------------+-----------------------+-------------+--------------
+------------+----------------+||
|||        AttachTime         |  DeleteOnTermination   |   Device    | InstanceId   |
 State    |   VolumeId     |||
||+--------------------------+-----------------------+-------------+--------------
+------------+----------------+||
|||  2013-09-17T00:55:03.000Z |  True                  |  /dev/sda1  |  i-a071c394  |
 attached  |  vol-e11a5288  |||
||+--------------------------+-----------------------+-------------+--------------
+------------+----------------+||
||                                                   Volumes
                         ||
|+------------------+---------------------------+-------+----------------+---------
+----------------+--------------+|
|| AvailabilityZone |        CreateTime         | Size  |  SnapshotId    |  State  |
 VolumeId     | VolumeType   ||
|+------------------+---------------------------+-------+----------------+---------
+----------------+--------------+|
||  us-west-2a      |  2013-09-18T20:26:15.000Z |  8    |  snap-708e8348 |  in-use |
 vol-2e410a47  |  standard    ||
|+------------------+---------------------------+-------+----------------+---------
+----------------+--------------+|
|||                                                Attachments
                        |||
```

```
||+--------------------------+------------------------+-------------+--------------
+------------+----------------+||
|||        AttachTime         |  DeleteOnTermination   |   Device    | InstanceId   |
 State    |   VolumeId     |||
||+--------------------------+------------------------+-------------+--------------
+------------+----------------+||
|||  2013-09-18T20:26:16.000Z |  True                  |  /dev/sda1  |  i-4b41a37c  |
 attached  |  vol-2e410a47  |||
||+--------------------------+------------------------+-------------+--------------
+------------+----------------+||
```

La opción `--query` se puede utilizar con el formato de tabla para mostrar un conjunto de elementos preseleccionados a partir de la salida en bruto. Tenga en cuenta las diferencias de la salida en las notaciones diccionario y lista: los nombres de columna están ordenados alfabéticamente en el primer ejemplo, y las columnas sin nombre se ordenan según lo definido por el usuario en el segundo ejemplo.

```
$ aws ec2 describe-volumes --query 'Volumes[*].
{ID:VolumeId,InstanceId:Attachments[0].InstanceId,AZ:AvailabilityZone,Size:Size}' --output
 table
-------------------------------------------------------
|                   DescribeVolumes                   |
+------------+----------------+--------------+-------+
|     AZ     |      ID        | InstanceId   | Size  |
+------------+----------------+--------------+-------+
|  us-west-2a|  vol-e11a5288  |  i-a071c394  |  30   |
|  us-west-2a|  vol-2e410a47  |  i-4b41a37c  |  8    |
+------------+----------------+--------------+-------+

$ aws ec2 describe-volumes --query 'Volumes[*].
[VolumeId,Attachments[0].InstanceId,AvailabilityZone,Size]' --output table
-----------------------------------------------------
|                  DescribeVolumes                  |
+--------------+--------------+--------------+-----+
|  vol-e11a5288|  i-a071c394  |  us-west-2a  |  30 |
|  vol-2e410a47|  i-4b41a37c  |  us-west-2a  |  8  |
+--------------+--------------+--------------+-----+
```

# Uso sintaxis abreviada con la AWS Command Line Interface

Aunque la AWS Command Line Interface puede aceptar parámetros de opciones no escalares en formato JSON, resultaría tedioso escribir listas o estructuras extensas en JSON en la línea de comandos. Para solucionar este problema, la AWS CLI admite una sintaxis abreviada que permite una representación de los parámetros de opciones más sencilla que si se usara el formato JSON íntegro.

## Parámetros estructurales

La sintaxis abreviada de la AWS CLI facilita a los usuarios la introducción de parámetros planos (con estructuras no anidadas). El formato consiste en una lista de pares clave-valor separados por comas:

Linux, macOS, or Unix

```
--option key1=value1,key2=value2,key3=value3
```

Windows PowerShell

```
--option "key1=value1,key2=value2,key3=value3"
```

Esto equivale al siguiente ejemplo con formato JSON:

```
--option '{"key1":"value1","key2":"value2","key3":"value3"}'
```

No debe haber espacios en blanco entre cada uno de los pares clave-valor separados por comas. A continuación se muestra un ejemplo del comando DynamoDB `update-table` con la opción `--provisioned-throughput` en modo abreviado.

```
$ aws dynamodb update-table --provisioned-
throughput ReadCapacityUnits=15,WriteCapacityUnits=10 --table-name MyDDBTable
```

Esto equivale al siguiente ejemplo con formato JSON:

```
$ aws dynamodb update-table --provisioned-
throughput '{"ReadCapacityUnits":15,"WriteCapacityUnits":10}' --table-name MyDDBTable
```

# Parámetros de lista

Los parámetros de entrada en una lista se pueden especificar de dos formas: en JSON y en sintaxis abreviada. La sintaxis abreviada de la AWS CLI ha sido diseñada para que su inserción en listas con números, cadenas o estructuras no anidadas resulte más sencilla. A continuación se muestra el formato básico, en el que los valores de la lista están separados por un único espacio.

```
--option value1 value2 value3
```

Esto equivale al siguiente ejemplo con formato JSON.

```
--option '[value1,value2,value3]'
```

Como ya hemos mencionado anteriormente, puede especificar una lista de números, una lista de cadenas o una lista de estructuras no anidadas en formato abreviado. A continuación se muestra un ejemplo del comando `stop-instances` para Amazon EC2, en el que el parámetro de entrada (lista de cadenas) para la opción `--instance-ids` se especifica en forma abreviada.

```
$ aws ec2 stop-instances --instance-ids i-1486157a i-1286157c i-ec3a7e87
```

Esto equivale al siguiente ejemplo con formato JSON.

```
$ aws ec2 stop-instances --instance-ids '["i-1486157a","i-1286157c","i-ec3a7e87"]'
```

A continuación se muestra un ejemplo de los comandos `create-tags` de Amazon EC2, que toma una lista de estructuras no anidadas para la opción `--tags`. La opción `--resources` especifica el identificador de la instancia que debe etiquetarse.

```
$ aws ec2 create-tags --resources i-1286157c --tags Key=My1stTag,Value=Value1
 Key=My2ndTag,Value=Value2 Key=My3rdTag,Value=Value3
```

Esto equivale al siguiente ejemplo con formato JSON. El parámetro JSON está escrito en varias líneas para facilitar su lectura.

```
$ aws ec2 create-tags --resources i-1286157c --tags '[
  {"Key": "My1stTag", "Value": "Value1"},
  {"Key": "My2ndTag", "Value": "Value2"},
  {"Key": "My3rdTag", "Value": "Value3"}
]'
```

# Uso de las opciones de paginación de la AWS Command Line Interface

Para los comandos que pueden devolver una amplia lista de elementos, la AWS CLI añade tres opciones que puede utilizar para modificar el comportamiento de la paginación de la CLI cuando llame a la API de un servicio para rellenar la lista.

De forma predeterminada, la CLI utiliza un tamaño de página de 1000 y recupera todos los elementos disponibles. Por ejemplo, si ejecuta `aws s3api list-objects` en un bucket Amazon S3 que contenga 3500 objetos, la CLI hace cuatro llamadas a Amazon S3, gestionando la lógica de paginación específica del servicio en segundo plano.

Si observa algún problema al ejecutar los comandos de la lista en un gran número de recursos, el tamaño de página predeterminado puede ser demasiado grande y ocasionar que se agote el tiempo de espera de las llamadas a los servicios de AWS. Puede utilizar la opción `--page-size` para especificar un menor tamaño de página y solucionar este problema. La CLI recuperará aun así la lista completa, pero realizará un mayor número de llamadas en segundo plano, recuperando un menor número de elementos con cada llamada:

```
$ aws s3api list-objects --bucket my-bucket --page-size 100
{
    "Contents": [
...
```

Para recuperar menos elementos, utilice la opción `--max-items`. La CLI gestionará la paginación de la misma forma, pero solo imprimirá el número de elementos que especifique:

```
$ aws s3api list-objects --bucket my-bucket --max-items 100
{
    "NextToken": "eyJNYXJrZXIiOiBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQiOiAxfQ==",
    "Contents": [
...
```

Si el número de elementos impresos (`--max-items`) es inferior al número total de elementos, se incluirá un `NextToken` que podrá pasar a un comando posterior para recuperar el siguiente conjunto de elementos:

```
$ aws s3api list-objects --bucket my-bucket --max-items 100 --starting-token
 eyJNYXJrZXIiOiBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQiOiAxfQ==
{
    "NextToken": "eyJNYXJrZXIiOiBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQiOiAyfQ==",
    "Contents": [
...
```

Puede que un servicio no devuelva los elementos en el mismo orden cada vez que lo llame. Si especifica un siguiente token en mitad de una página, puede que obtenga unos resultados diferentes a los previstos. Para evitarlo, utilice el mismo número para `--page-size` y `--max-items`; de esta forma se sincronizará

la paginación de la CLI con la del servicio. También puede recuperar toda la lista y realizar las operaciones de análisis necesarias localmente.

# Trabajar con Amazon Web Services

En esta sección se proporcionan ejemplos de cómo utilizar la AWS Command Line Interface para acceder a los servicios de AWS. Estos ejemplos están pensados para demostrar cómo utilizar la AWS CLI para realizar tareas administrativas.

Para obtener una referencia completa de todos los comandos disponibles para cada servicio, consulte la AWS Command Line Interface Reference o use la función de ayuda integrada de la línea de comandos. Para obtener más información, consulte Obtener ayuda con la AWS Command Line Interface (p. 34).

Temas



# Uso de Amazon DynamoDB con la AWS Command Line Interface

La AWS Command Line Interface (AWS CLI) proporciona compatibilidad con Amazon DynamoDB. Puede usar la AWS CLI para operaciones ad-hoc, como crear una tabla. También puede usarla para incluir operaciones de DynamoDB en scripts de utilidades.

El formato de la línea de comandos se compone de un nombre de API de Amazon DynamoDB seguido de los parámetros de dicha API. La AWS CLI admite la sintaxis abreviada de los valores de los parámetros, además de JSON.

Por ejemplo, el comando siguiente creará una tabla llamada `MusicCollection`.

> **Note**
>
> Para facilitar la legibilidad, los comandos largos de esta sección se dividen en líneas separadas. El carácter de barra inversa permite copiar y pegar (o escribir) varias líneas en un terminal Linux. Si el shell que utiliza no usa la barra inversa para los caracteres de escape, reemplácela por otro carácter de escape o quite las barras inversas e incluya el comando completo en una sola línea.

```
$ aws dynamodb create-table \
    --table-name MusicCollection \
    --attribute-definitions \
        AttributeName=Artist,AttributeType=S AttributeName=SongTitle,AttributeType=S \
    --key-schema AttributeName=Artist,KeyType=HASH AttributeName=SongTitle,KeyType=RANGE \
    --provisioned-throughput ReadCapacityUnits=1,WriteCapacityUnits=1
```

Los comandos siguientes añadirán nuevos elementos a la tabla. En este ejemplo se usa una combinación de sintaxis abreviada y JSON.

```
$ aws dynamodb put-item \
    --table-name MusicCollection \
    --item '{
```

```
        "Artist": {"S": "No One You Know"},
        "SongTitle": {"S": "Call Me Today"} ,
        "AlbumTitle": {"S": "Somewhat Famous"} }' \
    --return-consumed-capacity TOTAL
{
    "ConsumedCapacity": {
        "CapacityUnits": 1.0,
        "TableName": "MusicCollection"
    }
}
$ aws dynamodb put-item \
    --table-name MusicCollection \
    --item '{
        "Artist": {"S": "Acme Band"},
        "SongTitle": {"S": "Happy Day"} ,
        "AlbumTitle": {"S": "Songs About Life"} }' \
    --return-consumed-capacity TOTAL
{
    "ConsumedCapacity": {
        "CapacityUnits": 1.0,
        "TableName": "MusicCollection"
    }
}
```

Puede ser difícil crear código JSON válido en la línea de comandos. Sin embargo, la AWS CLI puede leer archivos JSON. Por ejemplo, considere el fragmento de código JSON siguiente, que se almacena en un archivo llamado `expression-attributes.json`:

Example expression-attributes.json

```
{
  ":v1": {"S": "No One You Know"},
  ":v2": {"S": "Call Me Today"}
}
```

Ahora puede emitir una solicitud de `Query` con la AWS CLI. En este ejemplo, el contenido del archivo `expression-attributes.json` se usa para el parámetro `--expression-attribute-values`:

```
$ aws dynamodb query --table-name MusicCollection \
    --key-condition-expression "Artist = :v1 AND SongTitle = :v2" \
    --expression-attribute-values file://expression-attributes.json
{
    "Count": 1,
    "Items": [
        {
            "AlbumTitle": {
                "S": "Somewhat Famous"
            },
            "SongTitle": {
                "S": "Call Me Today"
            },
            "Artist": {
                "S": "No One You Know"
            }
        }
    ],
    "ScannedCount": 1,
    "ConsumedCapacity": null
}
```

Para obtener más documentación sobre el uso de la AWS CLI con DynamoDB, vaya a http://docs.aws.amazon.com/cli/latest/reference/dynamodb/index.html.

Además de con DynamoDB, también puede usar la AWS CLI con DynamoDB Local. DynamoDB Local es un pequeño servidor y base de datos del lado del cliente que imita el servicio de DynamoDB y que permite escribir aplicaciones que usan la API de DynamoDB sin manipular sus tablas o datos. En lugar de eso, todas las acciones de la API se redirigen a DynamoDB Local. Cuando la aplicación crea una tabla o modifica datos, estos cambios se escriben en una base de datos local, lo que le permite ahorrar en rendimiento aprovisionado, almacenamiento de datos y tarifas de transferencia de datos.

Para obtener más información sobre DynamoDB Local y cómo usarlo con la AWS CLI, consulte las secciones siguientes de Guía para desarrolladores de Amazon DynamoDB:

- DynamoDB Local
- Usar la CLI de AWS con DynamoDB Local

# Uso de Amazon EC2 a través de la AWS Command Line Interface

Puede acceder a las características de Amazon EC2 con la AWS CLI. Para mostrar los comandos de la AWS CLI para Amazon EC2, utilice el siguiente comando.

```
$ aws ec2 help
```

Antes de ejecutar los comandos, defina sus credenciales predeterminadas. Para obtener más información, consulte Configuración de la AWS CLI (p. 17).

Para ver ejemplos de tareas comunes de Amazon EC2, consulte los siguientes temas.

Temas



## Uso de pares de claves

Puede utilizar la AWS CLI para crear, mostrar y eliminar sus pares de claves. Debe especificar un par de claves al lanzar y conectarse a una instancia Amazon EC2.

> **Note**
>
> Antes de probar los comandos de ejemplo, defina sus credenciales predeterminadas.

Temas



### Creación de un par de claves

Para crear un par de claves denominado `MyKeyPair`, utilice el comando create-key-pair y utilice la opción `--query` y la opción `--output text` para transferir su clave privada directamente a un archivo.

```
$ aws ec2 create-key-pair --key-name MyKeyPair --query 'KeyMaterial' --output text >
 MyKeyPair.pem
```

Tenga en cuenta que para Windows PowerShell, el redireccionamiento de `> file` adopta la codificación UTF-8 de forma predeterminada, que no se puede utilizar en algunos clientes SSH. Por lo tanto, debe indicar de forma explícita la codificación ASCII en el comando `out-file`.

```
> aws ec2 create-key-pair --key-name MyKeyPair --query 'KeyMaterial' --output text | out-
file -encoding ascii -filepath MyKeyPair.pem
```

El archivo `MyKeyPair.pem` resultante tiene este aspecto:

```
-----BEGIN RSA PRIVATE KEY-----
EXAMPLEKEYKCAQEAy7WZhaDsrA1W3mRlQtvhwyORRX8gnxgDAfRt/gx42kWXsT4rXE/b5CpSgie/
vBoU7jLxx92pNHoFnByP+Dc21eyyz6CvjTmWA0JwfWiW5/akH7iO5dSrvC7dQkW2duV5QuUdE0QW
Z/aNxMniGQE6XAgfwlnXVBwrerrQo+ZWQeqiUwwMkuEbLeJFLhMCvYURpUMSC1oehm449ilx9X1F
G50TCFeOzfl8dqqCP6GzbPaIjiU19xX/azOR9V+tpUOzEL+wmXnZt3/nHPQ5xvD2OJH67km6SuPW
oPzev/D8V+x4+bHthfSjR9Y7DvQFjfBVwHXigBdtZcU2/wei8D/HYwIDAQABAoIBAGZ1kaEvnrqu
/uler7vgIn5m7lN5LKw4hJLAIW6tUT/fzvtcHK0SkbQCQXuriHmQ2MQyJX/0kn2NfjLV/ufGxbL1
mb5qwMGUnEpJaZD6QSSs3kICLwWUYUiGfc0uiSbmJoap/GTLU0W5Mfcv36PaBUNy5p53V6G7hXb2
bahyWyJNfjLe4M86yd2YK3V2CmK+X/BOsShnJ36+hjrXPPWmV3N9zEmCdJjA+K15DYmhm/tJWSD9
81oGk9TopEp7CkIfatEATyyZiVqoRq6k64iuM9JkA3OzdXzMQexXVJ1TLZVEH0E7bhlY9d8O1ozR
oQs/FiZNAx2iijCWyv0lpjE73+kCgYEA9mZtyhkHkFDpwrSM1APaL8oNAbbjwEy7Z5Mqfql+lIp1
YkriL0DbLXlvRAH+yHPRit2hHOjtUNZh4Axv+cpg09qbUI3+43eEy24B7G/Uh+GTfbjsXsOxQx/x
p9otyVwc7hsQ5TA5PZb+mvkJ5OBEKzet9XcKwONBYELGhnEPe7cCgYEA06Vgov6YHleHui9kHuws
ayav0elc5zkxjF9nfHFJRry21R1trw2Vdpn+9g481URrpzWVOEihvm+xTtmaZlSp//lkq75XDwnU
WA8gkn6O3QE3fq2yN98BURsAKdJfJ5RL1HvGQvTe10HLYYXpJnEkHv+Unl2ajLivWUt5pbBrKbUC
gYBjbO+OZk0sCcpZ29sbzjYjpIddErySIyRX5gV2uNQwAjLdp9PfN295yQ+BxMBXiIycWVQiw0bH
oMo7yykABY7Ozd5wQewBQ4AdSlWSX4nGDtsiFxWiI5sKuAAeOCbTosy1s8w8fxoJ5Tz1sdoxNeGs
Arq6Wv/G16zQuAE9zK9vvwKBgF+09VI/1wJBirsDGz9whVWfFPrTkJNvJZzYt69qezxlsjgFKshy
WBhd4xHZtmCqpBPlAymEjr/TOlbxyARmXMnIOWIAnNXMGB4KGSyl1mzSVAoQ+fqR+cJ3d0dyPl1j
jjb0Ed/NY8frlNDxAVHE8BSkdsx2f6ELEyBKJSRr9snRAoGAMrTwYneXzvTskF/S5Fyu0iOegLDa
NWUH38v/nDCgEpIXD5Hn3qAEcju1IjmbwlvtW+nY2jVhv7UGd8MjwUTNGItdb6nsYqM2asrnF3qS
VRkAKKKYeGjkpUfVTrW0YFjXkfcrR/V+QFL5OndHAKJXjW7a4ejJLncTzmZSpYzwApc=
-----END RSA PRIVATE KEY-----
```

Su clave privada no se almacena en AWS y solo se puede recuperar en el momento de crearla.

Si está usando un cliente SSH en un equipo Linux para conectarse a su instancia, utilice el siguiente comando para establecer los permisos de su archivo de clave privada de modo que solo usted pueda leerlo.

```
$ chmod 400 MyKeyPair.pem
```

## Visualización de su par de claves

Se genera una huella digital a partir del par de claves. Puede utilizarla para verificar que la clave privada que tiene en su equipo local coincide con la clave pública almacenada en AWS. La huella digital es un valor hash SHA1 tomado a partir de una copia de la clave privada con codificación DER. Este valor se almacena en AWS y se puede ver en la consola de administración de EC2 o llamando a `aws ec2 describe-key-pairs`. Por ejemplo, puede ver la huella digital de `MyKeyPair` utilizando el siguiente comando:

```
$ aws ec2 describe-key-pairs --key-name MyKeyPair
{
    "KeyPairs": [
        {
```

```
            "KeyName": "MyKeyPair",
            "KeyFingerprint": "1f:51:ae:28:bf:89:e9:d8:1f:25:5d:37:2d:7d:b8:ca:9f:f5:f1:6f"
        }
    ]
}
```

Para obtener más información sobre las claves y las huellas, consulte la página Pares de claves de EC2 de Amazon en la Guía del usuario de Amazon EC2.

## Eliminación del par de claves

Para eliminar `MyKeyPair`, utilice el comando delete-key-pair del modo siguiente:

```
$ aws ec2 delete-key-pair --key-name MyKeyPair
```

# Uso de grupos de seguridad

Puede crear un grupo de seguridad para usarlo en EC2-Classic o en EC2-VPC. Para obtener más información acerca de EC2-Classic y EC2-VPC, consulte Plataformas compatibles en la Guía del usuario de Amazon EC2 para instancias de Linux.

Puede utilizar la AWS CLI para crear grupos de seguridad, añadirles reglas y eliminarlos.

> **Note**
>
> Antes de probar los comandos de ejemplo, defina sus credenciales predeterminadas.

Temas



## Creación de un grupo de seguridad

Para crear un grupo de seguridad denominado `my-sg`, utilice el comando create-security-group.

### EC2-VPC

El siguiente comando crea un grupo de seguridad denominado `my-sg` para la VPC especificada:

```
$ aws ec2 create-security-group --group-name my-sg --description "My security group" --vpc-
id vpc-1a2b3c4d
{
    "GroupId": "sg-903004f8"
}
```

Para ver la información inicial de `my-sg`, utilice el comando describe-security-groups del modo siguiente: Tenga en cuenta que no puede hacer referencia a un grupo de seguridad de EC2-VPC por nombre.

```
$ aws ec2 describe-security-groups --group-ids sg-903004f8
{
    "SecurityGroups": [
        {
```

```
            "IpPermissionsEgress": [
                {
                    "IpProtocol": "-1",
                    "IpRanges": [
                        {
                            "CidrIp": "0.0.0.0/0"
                        }
                    ],
                    "UserIdGroupPairs": []
                }
            ],
            "Description": "My security group"
            "IpPermissions": [],
            "GroupName": "my-sg",
            "VpcId": "vpc-1a2b3c4d",
            "OwnerId": "123456789012",
            "GroupId": "sg-903004f8"
        }
    ]
}
```

### EC2-Classic

El siguiente comando crea un grupo de seguridad para EC2-Classic:

```
$ aws ec2 create-security-group --group-name my-sg --description "My security group"
{
    "GroupId": "sg-903004f8"
}
```

Para ver la información inicial de `my-sg`, utilice el comando describe-security-groups del modo siguiente:

```
$ aws ec2 describe-security-groups --group-names my-sg
{
    "SecurityGroups": [
        {
            "IpPermissionsEgress": [],
            "Description": "My security group"
            "IpPermissions": [],
            "GroupName": "my-sg",
            "OwnerId": "123456789012",
            "GroupId": "sg-903004f8"
        }
    ]
}
```

## Añadir reglas al grupo de seguridad

Si lanza una instancia de Windows, debe añadir una regla para permitir el tráfico entrante en el puerto TCP 3389 (RDP). Si lanza una instancia de Linux, debe añadir una regla para permitir el tráfico entrante en el puerto TCP 22 (SSH). Utilice el comando authorize-security-group-ingress para añadir una regla a su grupo de seguridad. Uno de los parámetros obligatorios para este comando es la dirección IP pública de su equipo en notación CIDR.

> **Note**
>
> Puede obtener la dirección IP pública de su equipo local usando un servicio. Por ejemplo, brindamos el servicio a continuación: http://checkip.amazonaws.com/. Para buscar otro servicio que le brinde su dirección IP, utilice la frase de búsqueda "what is my IP address" (cuál es mi

dirección IP). Si se conecta a través de un ISP o protegido por su firewall sin una dirección IP estática, deberá encontrar el rango de direcciones IP utilizadas por los equipos cliente.

## EC2-VPC

El siguiente comando añade una regla de RDP al grupo de seguridad con el identificador sg-903004f8:

```
$ aws ec2 authorize-security-group-ingress --group-id sg-903004f8 --protocol tcp --port
 3389 --cidr 203.0.113.0/24
```

El siguiente comando añade una regla de SSH al grupo de seguridad con el identificador sg-903004f8:

```
$ aws ec2 authorize-security-group-ingress --group-id sg-903004f8 --protocol tcp --port 22
 --cidr 203.0.113.0/24
```

Para ver los cambios realizados en `my-sg`, utilice el comando describe-security-groups del modo siguiente:

```
$ aws ec2 describe-security-groups --group-ids sg-903004f8
{
    "SecurityGroups": [
        {
            "IpPermissionsEgress": [
                {
                    "IpProtocol": "-1",
                    "IpRanges": [
                        {
                            "CidrIp": "0.0.0.0/0"
                        }
                    ],
                    "UserIdGroupPairs": []
                }
            ],
            "Description": "My security group"
            "IpPermissions": [
                {
                    "ToPort": 22,
                    "IpProtocol": "tcp",
                    "IpRanges": [
                        {
                            "CidrIp": "203.0.113.0/24"
                        }
                    ]
                    "UserIdGroupPairs": [],
                    "FromPort": 22
                }
            ],
            "GroupName": "my-sg",
            "OwnerId": "123456789012",
            "GroupId": "sg-903004f8"
        }
    ]
}
```

## EC2-Classic

El siguiente comando añade una regla de RDP al grupo de seguridad `my-sg`:

```
$ aws ec2 authorize-security-group-ingress --group-name my-sg --protocol tcp --port 3389 --
cidr 203.0.113.0/24
```

El siguiente comando añade una regla de SSH al grupo de seguridad para `my-sg`:

```
$ aws ec2 authorize-security-group-ingress --group-name my-sg --protocol tcp --port 22 --
cidr 203.0.113.0/24
```

Para ver los cambios realizados en `my-sg`, utilice el comando describe-security-groups del modo siguiente:

```
$ aws ec2 describe-security-groups --group-names my-sg
{
    "SecurityGroups": [
        {
            "IpPermissionsEgress": [],
            "Description": "My security group"
            "IpPermissions": [
                {
                    "ToPort": 22,
                    "IpProtocol": "tcp",
                    "IpRanges": [
                        {
                            "CidrIp": "203.0.113.0/24"
                        }
                    ]
                    "UserIdGroupPairs": [],
                    "FromPort": 22
                }
            ],
            "GroupName": "my-sg",
            "OwnerId": "123456789012",
            "GroupId": "sg-903004f8"
        }
    ]
}
```

## Eliminación de un grupo de seguridad

Para eliminar un grupo de seguridad, utilice el comando delete-security-group. Tenga en cuenta que no puede eliminar un grupo de seguridad que esté conectado a un entorno.

### EC2-VPC

El siguiente comando elimina el grupo de seguridad con el identificador sg-903004f8:

```
$ aws ec2 delete-security-group --group-id sg-903004f8
```

### EC2-Classic

El siguiente comando elimina el grupo de seguridad denominado `my-sg`:

```
$ aws ec2 delete-security-group --group-name my-sg
```

# Uso de instancias Amazon EC2

Puede utilizar la AWS CLI para lanzar, enumerar y finalizar sus instancias. Necesitará un par de claves y un grupo de seguridad; para obtener información acerca de cómo crearlos mediante la AWS CLI, consulte Uso de pares de claves (p. 59) y Uso de grupos de seguridad (p. 61). También necesitará seleccionar una imagen de máquina de Amazon (AMI) y anotar su identificador. Para obtener más

información, consulte Finding a Suitable AMI en la Guía del usuario de Amazon EC2 para instancias de Linux.

Si lanza una instancia que no figura en la capa de uso gratuita, se le facturará en cuanto la lance y se le cobrará el tiempo en que la instancia esté funcionando, aunque permanezca inactiva.

> **Note**
>
> Antes de probar el comando del ejemplo, defina sus credenciales predeterminadas.

Temas



# Lanzamiento de una instancia

Para lanzar una sola instancia Amazon EC2 utilizando la AMI que ha seleccionado, use el comando run-instances. Según las plataformas que su cuenta admita, puede lanzar la instancia en EC2-Classic o EC2-VPC.

Al principio, la instancia tiene el estado `pending`, pero cambia al estado `running` en unos minutos.

## EC2-VPC

El siguiente comando lanza una instancia `t1.micro` en la subred especificada:

```
$ aws ec2 run-instances --image-id
          ami-xxxxxxxx --count 1 --instance-type t2.micro --key-name MyKeyPair --security-
group-ids sg-xxxxxxxx --subnet-id subnet-xxxxxxxx
{
    "OwnerId": "123456789012",
    "ReservationId": "r-5875ca20",
    "Groups": [
        {
            "GroupName": "my-sg",
            "GroupId": "sg-903004f8"
        }
    ],
    "Instances": [
        {
            "Monitoring": {
                "State": "disabled"
            },
            "PublicDnsName": null,
            "Platform": "windows",
            "State": {
                "Code": 0,
                "Name": "pending"
            },
            "EbsOptimized": false,
            "LaunchTime": "2013-07-19T02:42:39.000Z",
            "PrivateIpAddress": "10.0.1.114",
            "ProductCodes": [],
            "VpcId": "vpc-1a2b3c4d",
```

```
            "InstanceId": "i-5203422c",
            "ImageId": "ami-173d747e",
            "PrivateDnsName": ip-10-0-1-114.ec2.internal,
            "KeyName": "MyKeyPair",
            "SecurityGroups": [
                {
                    "GroupName": "my-sg",
                    "GroupId": "sg-903004f8"
                }
            ],
            "ClientToken": null,
            "SubnetId": "subnet-6e7f829e",
            "InstanceType": "t2.micro",
            "NetworkInterfaces": [
                {
                    "Status": "in-use",
                    "SourceDestCheck": true,
                    "VpcId": "vpc-1a2b3c4d",
                    "Description": "Primary network interface",
                    "NetworkInterfaceId": "eni-a7edb1c9",
                    "PrivateIpAddresses": [
                        {
                            "PrivateDnsName": "ip-10-0-1-114.ec2.internal",
                            "Primary": true,
                            "PrivateIpAddress": "10.0.1.114"
                        }
                    ],
                    "PrivateDnsName": "ip-10-0-1-114.ec2.internal",
                    "Attachment": {
                        "Status": "attached",
                        "DeviceIndex": 0,
                        "DeleteOnTermination": true,
                        "AttachmentId": "eni-attach-52193138",
                        "AttachTime": "2013-07-19T02:42:39.000Z"
                    },
                    "Groups": [
                        {
                            "GroupName": "my-sg",
                            "GroupId": "sg-903004f8"
                        }
                    ],
                    "SubnetId": "subnet-6e7f829e",
                    "OwnerId": "123456789012",
                    "PrivateIpAddress": "10.0.1.114"
                }
            ],
            "SourceDestCheck": true,
            "Placement": {
                "Tenancy": "default",
                "GroupName": null,
                "AvailabilityZone": "us-west-2b"
            },
            "Hypervisor": "xen",
            "BlockDeviceMappings": [
                {
                    "DeviceName": "/dev/sda1",
                    "Ebs": {
                        "Status": "attached",
                        "DeleteOnTermination": true,
                        "VolumeId": "vol-877166c8",
                        "AttachTime": "2013-07-19T02:42:39.000Z"
                    }
                }
            ],
            "Architecture": "x86_64",
            "StateReason": {
```

```
                "Message": "pending",
                "Code": "pending"
            },
            "RootDeviceName": "/dev/sda1",
            "VirtualizationType": "hvm",
            "RootDeviceType": "ebs",
            "Tags": [
                {
                    "Value": "MyInstance",
                    "Key": "Name"
                }
            ],
            "AmiLaunchIndex": 0
        }
    ]
}
```

## EC2-Classic

El siguiente comando lanza una instancia `t1.micro` en EC2-Classic:

```
$ aws ec2 run-instances --image-id ami-xxxxxxxx --count 1 --instance-type t1.micro --key-
name MyKeyPair --security-groups my-sg
{
    "OwnerId": "123456789012",
    "ReservationId": "r-5875ca20",
    "Groups": [
        {
            "GroupName": "my-sg",
            "GroupId": "sg-903004f8"
        }
    ],
    "Instances": [
        {
            "Monitoring": {
                "State": "disabled"
            },
            "PublicDnsName": null,
            "Platform": "windows",
            "State": {
                "Code": 0,
                "Name": "pending"
            },
            "EbsOptimized": false,
            "LaunchTime": "2013-07-19T02:42:39.000Z",
            "ProductCodes": [],
            "InstanceId": "i-5203422c",
            "ImageId": "ami-173d747e",
            "PrivateDnsName": null,
            "KeyName": "MyKeyPair",
            "SecurityGroups": [
                {
                    "GroupName": "my-sg",
                    "GroupId": "sg-903004f8"
                }
            ],
            "ClientToken": null,
            "InstanceType": "t1.micro",
            "NetworkInterfaces": [],
            "Placement": {
                "Tenancy": "default",
                "GroupName": null,
                "AvailabilityZone": "us-west-2b"
            },
```

```
            "Hypervisor": "xen",
            "BlockDeviceMappings": [
                {
                    "DeviceName": "/dev/sda1",
                    "Ebs": {
                        "Status": "attached",
                        "DeleteOnTermination": true,
                        "VolumeId": "vol-877166c8",
                        "AttachTime": "2013-07-19T02:42:39.000Z"
                    }
                }
            ],
            "Architecture": "x86_64",
            "StateReason": {
                "Message": "pending",
                "Code": "pending"
            },
            "RootDeviceName": "/dev/sda1",
            "VirtualizationType": "hvm",
            "RootDeviceType": "ebs",
            "Tags": [
                {
                    "Value": "MyInstance",
                    "Key": "Name"
                }
            ],
            "AmiLaunchIndex": 0
        }
    ]
}
```

## Añadir una asignación de dispositivos de bloque a la instancia

Cada instancia que lance tiene un volumen de dispositivo raíz asociado. Puede utilizar la asignación de dispositivos de bloque para especificar los volúmenes de EBS adicionales o volúmenes de almacén de instancias para adjuntar a una instancia a la hora de lanzarla.

Para añadir una asignación de dispositivos de bloque a la instancia, especifique la opción `--block-device-mappings` cuando utilice `run-instances`.

En el siguiente ejemplo se añade un volumen de Amazon EBS estándar, asignado a `/dev/sdf`, cuyo tamaño es de 20 GB.

```
--block-device-mappings "[{\"DeviceName\":\"/dev/sdf\",\"Ebs\":{\"VolumeSize\":20,
\"DeleteOnTermination\":false}}]"
```

En el siguiente ejemplo se añade un volumen de Amazon EBS, asignado a `/dev/sdf`, a partir de una snapshot. Al especificar una instantánea no es necesario especificar un tamaño de volumen, pero si lo hace, debe ser igual o mayor que el tamaño de la instantánea.

```
--block-device-mappings "[{\"DeviceName\":\"/dev/sdf\",\"Ebs\":{\"SnapshotId\":
\"snap-xxxxxxxx\"}}]"
```

En el siguiente ejemplo se añaden dos volúmenes de almacén de instancias. Tenga en cuenta que el número de volúmenes de almacén de instancias disponibles para su instancia depende del tipo de instancia.

```
--block-device-mappings "[{\"DeviceName\":\"/dev/sdf\",\"VirtualName\":\"ephemeral0\"},
{\"DeviceName\":\"/dev/sdg\",\"VirtualName\":\"ephemeral1\"}]"
```

En el siguiente ejemplo se omite una asignación de un dispositivo especificado por la AMI utilizado para lanzar la instancia (`/dev/sdj`):

```
--block-device-mappings "[{\"DeviceName\":\"/dev/sdj\",\"NoDevice\":\"\"}]"
```

Para obtener más información, consulte Asignación de dispositivos de bloques en la Guía del usuario de Amazon EC2 para instancias de Linux.

## Añadir una etiqueta Name a la instancia

Para añadir la etiqueta `Name=MyInstance` a la instancia, utilice el comando create-tags del modo siguiente:

```
$ aws ec2 create-tags --resources i-xxxxxxxx --tags Key=Name,Value=MyInstance
```

Para obtener más información, consulte Tagging Your Resources en la Guía del usuario de Amazon EC2 para instancias de Linux.

## Conexión a la instancia

Puede conectarse a su instancia mientras se esté ejecutando y utilizarla como si fuera un equipo que tiene delante. Para obtener más información, consulte Connect to Your Amazon EC2 Instance en la Guía del usuario de Amazon EC2 para instancias de Linux.

## Mostrar las instancias

Puede utilizar la AWS CLI para mostrar las instancias y ver información de las mismas. Puede incluir todas sus instancias en la lista o filtrar los resultados en función de las instancias que le interesen.

> **Note**
>
> Antes de probar los comandos de ejemplo, defina sus credenciales predeterminadas.

En los siguientes ejemplos se muestra cómo utilizar el comando describe-instances.

### Example 1: Mostrar las instancias con el tipo de instancia especificado

El siguiente comando muestra sus instancias de `m1.small`.

```
$ aws ec2 describe-instances --filters "Name=instance-type,Values=m1.small"
```

### Example 2: Mostrar las instancias lanzadas usando las imágenes especificadas

El siguiente comando muestra las instancias que tiene y que se iniciaron desde las siguientes AMI: `ami-x0123456`, `ami-y0123456`y `ami-z0123456`.

```
$ aws ec2 describe-instances --filters "Name=image-id,Values=ami-x0123456,ami-y0123456,ami-z0123456"
```

## Finalización de la instancia

Cuando se finaliza una instancia, se elimina totalmente: no es posible volver a conectarse a ella. En cuanto el estado de la instancia cambie a `shutting-down` o a `terminated`, dejará de incurrir en costos por ella.

Cuando haya terminado con la instancia, utilice el comando terminate-instances del modo siguiente:

```
$ aws ec2 terminate-instances --instance-ids i-5203422c
{
    "TerminatingInstances": [
        {
            "InstanceId": "i-5203422c",
            "CurrentState": {
                "Code": 32,
                "Name": "shutting-down"
            },
            "PreviousState": {
                "Code": 16,
                "Name": "running"
            }
        }
    ]
}
```

Para obtener más información, consulte Terminate Your Instance en la Guía del usuario de Amazon EC2 para instancias de Linux.

# Uso de Amazon Glacier con la AWS Command Line Interface

Para cargar un archivo de gran tamaño en Amazon Glacier, puede dividirlo en partes más pequeñas y cargarlas desde la línea de comandos. Este tema describe el proceso para crear un almacén y dividir un archivo, además de para configurar y ejecutar una carga multiparte en Amazon Glacier con la AWS CLI.

> Note
>
> En este tutorial se usan varias herramientas de línea de comandos que suelen venir preinstaladas en los sistemas operativos tipo Unix, incluidos Linux y OS X. Los usuarios de Windows pueden utilizar las mismas herramientas si instalan Cygwin y ejecutan los comandos desde la terminal de Cygwin. Si hay disponibles utilidades y comandos nativos de Windows que realizan las mismas funciones, se indican en cada caso.

Temas



## Creación de un almacén Amazon Glacier

Cree un almacén con el comando `aws glacier create-vault`. El comando siguiente crea un almacén llamado `myvault`.

```
$ aws glacier create-vault --account-id - --vault-name myvault
{
    "location": "/123456789012/vaults/myvault"
}
```

Note

Todos los comandos de glacier requieren un parámetro de ID de cuenta. Use un guion para especificar la cuenta actual.

# Preparación de un archivo para cargarlo

Cree un archivo para la carga de prueba. Los comandos siguientes crean un archivo que contiene exactamente 3 MiB (3 x 1024 x 1024 bytes) de datos aleatorios.

Linux, macOS, or Unix

```
$ dd if=/dev/urandom of=largefile bs=3145728 count=1
1+0 records in
1+0 records out
3145728 bytes (3.1 MB) copied, 0.205813 s, 15.3 MB/s
```

`dd` es una utilidad que copia un número de bytes de un archivo de entrada en un archivo de salida. En el ejemplo anterior se usa el archivo de dispositivo `/dev/urandom` como origen de los datos aleatorios. `fsutil` realiza una función similar en Windows:

Windows

```
C:\temp>fsutil file createnew largefile 3145728
File C:\temp\largefile is created
```

A continuación, divida el archivo en fragmentos de 1 MiB (1048576 bytes).

```
$ split --bytes=1048576 --verbose largefile chunk
creating file `chunkaa'
creating file `chunkab'
creating file `chunkac'
```

Note

HJ-Split es un divisor de archivos gratuito para Windows y muchas otras plataformas.

# Inicio de una carga multiparte y carga de archivos

Cree una carga multiparte en Amazon Glacier con el comando `aws glacier initiate-multipart-upload`.

```
$ aws glacier initiate-multipart-upload --account-id - --archive-description "multipart
 upload test" --part-size 1048576 --vault-name myvault
{
    "uploadId": "19gaRezEXAMPLES6Ry5YYdqthHOC_kGRCT03L9yetr220UmPtBYKk-
OssZtLqyFu7sY1_lR7vgFuJV6NtcV5zpsJ",
    "location": "/123456789012/vaults/myvault/multipart-
uploads/19gaRezEXAMPLES6Ry5YYdqthHOC_kGRCT03L9yetr220UmPtBYKk-
OssZtLqyFu7sY1_lR7vgFuJV6NtcV5zpsJ"
}
```

Amazon Glacier requiere especificar el tamaño de cada parte en bytes (1 MiB en este ejemplo), el nombre del almacén y el ID de cuenta para configurar la carga multiparte. La AWS CLI genera un ID de carga al completarse la operación. Guarde el ID de carga en una variable de shell para su uso posterior.

Linux, macOS, or Unix

```
$ UPLOADID="19gaRezEXAMPLES6Ry5YYdqthHOC_kGRCT03L9yetr220UmPtBYKk-
OssZtLqyFu7sY1_lR7vgFuJV6NtcV5zpsJ"
```

Windows

```
C:\temp> set UPLOADID="19gaRezEXAMPLES6Ry5YYdqthHOC_kGRCT03L9yetr220UmPtBYKk-
OssZtLqyFu7sY1_lR7vgFuJV6NtcV5zpsJ"
```

A continuación, use el comando `aws glacier upload-multipart-part` para cargar cada parte.

```
$ aws glacier upload-multipart-part --upload-id $UPLOADID --body chunkaa --range 'bytes
 0-1048575/*' --account-id - --vault-name myvault
{
    "checksum": "e1f2a7cd6e047fa606fe2f0280350f69b9f8cfa602097a9a026360a7edc1f553"
}
$ aws glacier upload-multipart-part --upload-id $UPLOADID --body chunkab --range 'bytes
 1048576-2097151/*' --account-id - --vault-name myvault
{
    "checksum": "e1f2a7cd6e047fa606fe2f0280350f69b9f8cfa602097a9a026360a7edc1f553"
}
$ aws glacier upload-multipart-part --upload-id $UPLOADID --body chunkac --range 'bytes
 2097152-3145727/*' --account-id - --vault-name myvault
{
    "checksum": "e1f2a7cd6e047fa606fe2f0280350f69b9f8cfa602097a9a026360a7edc1f553"
}
```

> **Note**
>
> En el ejemplo anterior se usa el signo de dólar ("$") para desreferenciar la variable de shell `UPLOADID`. En la línea de comandos de Windows, use dos signos de porcentaje (es decir, `%UPLOADID%`).

Debe especificar el rango de bytes de cada parte al cargarla, para que Amazon Glacier pueda volver a ensamblarlas en el orden adecuado. Cada parte tiene 1048576 bytes, por lo que la primera ocupa los bytes del 0 al 1048575, la segunda del 1048576 al 2097151 y la tercera del 2097152 al 3145727.

# Completar la carga

Amazon Glacier requiere un algoritmo hash en árbol del archivo original, para poder confirmar que todas las partes cargadas llegan a AWS intactas. Para calcular un algoritmo hash en árbol, divida el archivo en partes de 1 MiB y calcule un hash SHA-256 binario de cada fragmento. A continuación, divida la lista de hashes en pares, combine los dos hashes binarios de cada par y tome los hashes de los resultados. Repita este proceso hasta que solo quede un hash. Si hay un número impar de hashes en cualquier nivel, páselo al nivel siguiente sin modificarlo.

La clave para calcular un algoritmo hash en árbol correctamente al usar utilidades de línea de comandos es almacenar cada hash en formato binario y convertir a hexadecimal solamente en el último paso. Si la versión hexadecimal de cualquier hash del árbol se combina o se convierte a algoritmo hash, se obtendrá un resultado incorrecto.

> **Note**
>
> Los usuarios de Windows pueden utilizar el comando `type` en lugar de `cat`. OpenSSL está disponible para Windows en OpenSSL.org.

### Para calcular un algoritmo hash en árbol

1. Divida el archivo original en partes de 1 MiB, si no lo ha hecho aún.

```
$ split --bytes=1048576 --verbose largefile chunk
creating file `chunkaa'
creating file `chunkab'
creating file `chunkac'
```

2. Calcule y almacene el hash SHA-256 binario de cada fragmento.

```
$ openssl dgst -sha256 -binary chunkaa > hash1
$ openssl dgst -sha256 -binary chunkab > hash2
$ openssl dgst -sha256 -binary chunkac > hash3
```

3. Combine los dos primeros hashes y tome el hash binario del resultado.

```
$ cat hash1 hash2 > hash12
$ openssl dgst -sha256 -binary hash12 > hash12hash
```

4. Combine el hash principal de los fragmentos aa y ab con el hash del fragmento ac y cree un hash del resultado, esta vez con formato hexadecimal. Almacene el resultado en una variable de shell.

```
$ cat hash12hash hash3 > hash123
$ openssl dgst -sha256 hash123
SHA256(hash123)= 9628195fcdbcbbe76cdde932d4646fa7de5f219fb39823836d81f0cc0e18aa67
$ TREEHASH=9628195fcdbcbbe76cdde932d4646fa7de5f219fb39823836d81f0cc0e18aa67
```

Por último, complete la carga con el comando `aws glacier complete-multipart-upload`, que toma el tamaño del archivo original en bytes, el valor final del algoritmo hash en árbol en formato hexadecimal, su ID de cuenta y el nombre del almacén.

```
$ aws glacier complete-multipart-upload --checksum $TREEHASH --archive-size 3145728 --
upload-id $UPLOADID --account-id - --vault-name myvault
{
    "archiveId": "d3AbWhE0YE1m6f_fI1jPG82F8xzbMEEZmrAlLGAAONJAzo5QdP-
N83MKqd96Unspoa5H5lItWX-sK8-QS0ZhwsyGiu9-R-kwWUyS1dSBlmgPPWkEbeFfqDSav053rU7FvVLHfRc6hg",
    "checksum": "9628195fcdbcbbe76cdde932d4646fa7de5f219fb39823836d81f0cc0e18aa67",
    "location": "/123456789012/vaults/myvault/archives/
d3AbWhE0YE1m6f_fI1jPG82F8xzbMEEZmrAlLGAAONJAzo5QdP-N83MKqd96Unspoa5H5lItWX-sK8-
QS0ZhwsyGiu9-R-kwWUyS1dSBlmgPPWkEbeFfqDSav053rU7FvVLHfRc6hg"
}
```

También puede comprobar el estado del almacén con `aws glacier describe-vault`:

```
$ aws glacier describe-vault --account-id - --vault-name myvault
{
    "SizeInBytes": 3178496,
    "VaultARN": "arn:aws:glacier:us-west-2:123456789012:vaults/myvault",
    "LastInventoryDate": "2015-04-07T00:26:19.028Z",
    "NumberOfArchives": 1,
    "CreationDate": "2015-04-06T21:23:45.708Z",
    "VaultName": "myvault"
}
```

> **Note**
>
> El estado del almacén se actualiza una vez al día. Consulte Working with Vaults para obtener más información.

Ahora puede eliminar sin problemas la parte y los archivos hash que ha creado:

```
$ rm chunk* hash*
```

Para obtener más información sobre las cargas multiparte, consulte las secciones para cargar archivos grandes por partes y calcular sumas de comprobación en la Guía para desarrolladores de Amazon Glacier.

# AWS Identity and Access Management desde la AWS Command Line Interface

En esta sección se describen algunas tareas comunes relacionadas con AWS Identity and Access Management (IAM) y cómo realizarlas usando la AWS Command Line Interface.

Los comandos que se muestran aquí suponen que ya ha establecido credenciales predeterminadas y una región predeterminada.

Temas



## Creación de nuevos usuarios y grupos de IAM

En esta sección se describe cómo crear un nuevo grupo de IAM y un nuevo usuario de IAM y cómo agregar el usuario al grupo.

Para crear un grupo de IAM y añadir un usuario de IAM:

1. En primer lugar, use el comando `create-group` para crear el grupo.

```
$ aws iam create-group --group-name MyIamGroup
{
    "Group": {
        "GroupName": "MyIamGroup",
        "CreateDate": "2012-12-20T03:03:52.834Z",
        "GroupId": "AKIAI44QH8DHBEXAMPLE",
        "Arn": "arn:aws:iam::123456789012:group/MyIamGroup",
        "Path": "/"
    }
}
```

2. A continuación, use el comando `create-user` para crear el usuario.

```
$ aws iam create-user --user-name MyUser
{
    "User": {
        "UserName": "MyUser",
        "Path": "/",
        "CreateDate": "2012-12-20T03:13:02.581Z",
        "UserId": "AKIAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::123456789012:user/MyUser"
    }
}
```

3. Por último, use el comando `add-user-to-group` para añadir el usuario al grupo.

```
$ aws iam add-user-to-group --user-name MyUser --group-name MyIamGroup
```

4. Para verificar que el grupo `MyIamGroup` contiene al usuario `MyUser`, use el comando `get-group`.

```
$ aws iam get-group --group-name MyIamGroup
{
    "Group": {
        "GroupName": "MyIamGroup",
        "CreateDate": "2012-12-20T03:03:52Z",
        "GroupId": "AKIAI44QH8DHBEXAMPLE",
        "Arn": "arn:aws:iam::123456789012:group/MyIamGroup",
        "Path": "/"
    },
    "Users": [
        {
            "UserName": "MyUser",
            "Path": "/",
            "CreateDate": "2012-12-20T03:13:02Z",
            "UserId": "AKIAIOSFODNN7EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:user/MyUser"
        }
    ],
    "IsTruncated": "false"
}
```

También puede ver los usuarios y grupos IAM con la Consola de administración de AWS.

# Definición de una política de IAM para un usuario de IAM

Los siguientes comandos muestran cómo asignar una política de IAM a un usuario de IAM. La política especificada aquí ofrece al usuario acceso avanzado ("Power User Access"). Esta política es idéntica a la plantilla de la política Power User Access proporcionada en la consola de IAM. En este ejemplo, la política se guarda en un archivo, `MyPolicyFile.json`:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "NotAction": "iam:*",
      "Resource": "*"
    }
  ]
}
```

Para especificar la política, use el comando `put-user-policy`.

```
$ aws iam put-user-policy --user-name MyUser --policy-name MyPowerUserRole --policy-
document file://C:\Temp\MyPolicyFile.json
```

Compruebe que la política se ha asignado al usuario con el comando `list-user-policies`.

```
$ aws iam list-user-policies --user-name MyUser
```

```
{
    "PolicyNames": [
        "MyPowerUserRole"
    ],
    "IsTruncated": "false"
}
```

## Recursos adicionales

Para obtener más información, consulte Recursos de aprendizaje sobre permisos y políticas. Este tema incluye vínculos a información general sobre permisos y políticas, además de vínculos a ejemplos de políticas para acceder a Amazon S3, Amazon EC2 y otros servicios.

# Establecimiento de una contraseña inicial para un usuario de IAM

El siguiente ejemplo ilustra cómo utilizar el comando `create-login-profile` para configurar una contraseña inicial para un usuario de IAM.

```
$ aws iam create-login-profile --user-name MyUser --password My!User1Login8P@ssword
{
    "LoginProfile": {
        "UserName": "MyUser",
        "CreateDate": "2013-01-02T21:10:54.339Z",
        "MustChangePassword": "false"
    }
}
```

Use el comando `update-login-profile` para actualizar la contraseña para un usuario de IAM.

# Creación de credenciales de seguridad para un usuario de IAM

En el siguiente ejemplo se utiliza el comando `create-access-key` para crear credenciales de seguridad para un usuario de IAM. Un conjunto de credenciales de seguridad se compone de un ID de clave de acceso y una clave secreta. Tenga en cuenta que un usuario de IAM no puede tener más de dos conjuntos de credenciales en cualquier momento dado. Si intenta crear un tercer conjunto, el comando `create-access-key` devolverá un error "LimitExceeded".

```
$ aws iam create-access-key --user-name MyUser
{
    "AccessKey": {
        "SecretAccessKey": "je7MtGbClwBF/2Zp9Utk/h3yCo8nvbEXAMPLEKEY",
        "Status": "Active",
        "CreateDate": "2013-01-02T22:44:12.897Z",
        "UserName": "MyUser",
        "AccessKeyId": "AKIAI44QH8DHBEXAMPLE"
    }
}
```

Use el comando `delete-access-key` para eliminar un conjunto de credenciales de un usuario de IAM. Especifique las credenciales que quiera eliminar mediante el ID de clave de acceso.

```
$ aws iam delete-access-key --user-name MyUser --access-key-id AKIAI44QH8DHBEXAMPLE
```

# Uso de Amazon S3 con la AWS Command Line Interface

La AWS CLI ofrece dos niveles de comandos para acceder a Amazon S3.

- El primer nivel, denominado s3, se compone de comandos de alto nivel para las operaciones más utilizadas, como crear, manipular y eliminar objetos y buckets.
- El segundo nivel, denominado s3api, expone todas las operaciones de Amazon S3, incluida la modificación la lista de control de acceso (ACL) de un bucket, el uso compartido de recursos entre orígenes (CORS) o el registro de políticas. Le permite realizar operaciones avanzadas que podrían no ser posibles solamente con los comandos de alto nivel.

Para obtener una lista de todos los comandos disponibles en cada nivel, use el argumento `help` con los comandos `aws s3` o `aws s3api`:

```
$ aws s3 help
```

o bien

```
$ aws s3api help
```

> **Note**
>
> La AWS CLI admite la copia, la transferencia y la sincronización de Amazon S3 a Amazon S3. Estas operaciones usan la operación de servicio COPY facilitada por Amazon S3: sus archivos se conservan en la nube, no se descargan en el equipo cliente y realizan una copia de seguridad en Amazon S3.
> Cuando este tipo de operaciones se pueden llevar a cabo totalmente en la nube, solo se usa el ancho de banda necesario para la solicitud y la respuesta HTTP.

Para ver ejemplos del uso de Amazon S3, consulte los siguientes temas en esta sección.

Temas



## Uso de comandos S3 de alto nivel con la AWS Command Line Interface

En esta sección se describe cómo administrar los buckets y los objetos de Amazon S3 a través de comandos `aws s3` de alto nivel.

### Administración de buckets

Los comandos `aws s3` de alto nivel sirven para realizar operaciones bucket de uso común, como, por ejemplo, la creación, eliminación y muestra de buckets.

#### Creación de buckets

Utilice el comando `aws s3 mb` para crear un nuevo bucket. Los nombres de los buckets deben ser únicos y compatibles con DNS. Los nombres de los buckets pueden contener minúsculas, números, guiones y

puntos. Los nombres de los buckets solo pueden empezar y terminar con una letra o número, y no pueden contener un punto junto a un guion u otro punto.

```
$ aws s3 mb s3://bucket-name
```

### Eliminación de buckets

Para eliminar un bucket, use el comando `aws s3 rb`.

```
$ aws s3 rb s3://bucket-name
```

De forma predeterminada, el bucket debe estar vacío para que la operación se realice correctamente. Para eliminar un bucket que no esté vacío, debe incluir la opción `--force`.

```
$ aws s3 rb s3://bucket-name --force
```

En primer lugar, esto eliminará todos los objetos y subcarpetas del bucket y, a continuación, eliminará el bucket.

> Note
>
> Si utiliza un bucket versionado que contenga objetos eliminados previamente, pero que se conservan, este comando no le permitirá eliminar el bucket.

### Visualización de buckets

Para mostrar todos los buckets o su contenido, use el comando `aws s3 ls`. A continuación se muestran algunos ejemplos de uso comunes.

El siguiente comando muestra todos los buckets.

```
$ aws s3 ls
2013-07-11 17:08:50 my-bucket
2013-07-24 14:55:44 my-bucket2
```

El siguiente comando muestra todos los objetos y carpetas (prefijos) de un bucket.

```
$ aws s3 ls s3://bucket-name
                           PRE path/
2013-09-04 19:05:48          3 MyFile1.txt
```

El siguiente comando muestra los objetos en *nombre-del-bucket*`/path` (es decir, los objetos que estén en *nombre-del-bucket* filtrados por el prefijo `path/`).

```
$ aws s3 ls s3://bucket-name/path/
2013-09-06 18:59:32          3 MyFile2.txt
```

## Administración de objetos

Los comandos `aws s3` de alto nivel también facilitan la administración de los objetos de Amazon S3. Entre los comandos de objeto se incluyen `aws s3 cp`, `aws s3 ls`, `aws s3 mv`, `aws s3 rm`, y `sync`. Los comandos `cp`, `ls`, `mv` y `rm` funcionan de forma similar a sus homólogos Unix y le permiten trabajar perfectamente en sus directorios locales y los buckets de Amazon S3. El comando `sync` sincroniza el contenido de un bucket y un directorio, o de dos buckets.

> **Note**
>
> Todos los comandos de alto nivel que implican cargar objetos en un bucket de Amazon S3 (`aws s3 cp`, `aws s3 mv` y `aws s3 sync`) realizarán automáticamente una carga multiparte cuando el objeto es grande.
> Los errores de carga no pueden reanudarse cuando se usan estos comandos. Si la carga multiparte falla debido al agotamiento del tiempo de espera o se cancela manualmente pulsando CTRL+C, la AWS CLI limpia los archivos creados e interrumpe la carga. Este proceso puede tardar varios minutos.
> Si el proceso se interrumpe mediante un comando de interrupción o un error del sistema, la carga multiparte en curso permanece en Amazon S3 y debe limpiarse manualmente en la Consola de administración de AWS o con el comando s3api abort-multipart-upload.

Los comandos `cp`, `mv` y `sync` incluyen una opción `--grants` que se puede usar para conceder permisos sobre el objeto a usuarios o grupos específicos. Puede configurar la opción `--grants` para una lista de permisos mediante la sintaxis siguiente:

```
--grants Permission=Grantee_Type=Grantee_ID
         [Permission=Grantee_Type=Grantee_ID ...]
```

Cada valor contiene los siguientes elementos:

- *`Permission`*: especifica los permisos otorgados y puede configurarse como `read`, `readacl`, `writeacl` o `full`.
- *`Grantee_Type`*: especifica cómo debe identificarse al beneficiario del permiso, y se puede configurar como `uri`, `emailaddress` o `id`.
- *`Grantee_ID`*: especifica el beneficiario del permiso según *`Grantee_Type`*.
  - `uri`: URI del grupo. Para obtener más información, consulte la sección ¿Quién es el beneficiario?
  - `emailaddress`: dirección de correo electrónico de la cuenta.
  - `id`: ID canónico de la cuenta.

Para obtener más información sobre el control del acceso de Amazon S3, consulte el apartado Control de acceso.

En el siguiente ejemplo se copia un objeto en un bucket. Concede permisos `read` sobre el objeto a todos los usuarios y permisos `full` (`read`, `readacl` y `writeacl`) a la cuenta asociada con `user@example.com`.

```
$ aws s3 cp file.txt s3://my-bucket/ --grants read=uri=http://acs.amazonaws.com/groups/
global/AllUsers full=emailaddress=user@example.com
```

Para especificar una clase de almacenamiento no predeterminada (`REDUCED_REDUNDANCY` o `STANDARD_IA`) para los objetos que se cargan en Amazon S3, utilice la opción `--storage-class`:

```
$ aws s3 cp file.txt s3://my-bucket/ --storage-class REDUCED_REDUNDANCY
```

El comando `sync` tiene la forma siguiente. Las combinaciones de origen y destino posibles son:

- Sistema de archivos local a Amazon S3
- Amazon S3 a sistema de archivos local
- Amazon S3 a Amazon S3

```
$ aws s3 sync <source> <target> [--options]
```

En el siguiente ejemplo se sincroniza el contenido de una carpeta de Amazon S3 llamada path en mi-bucket con el directorio activo actual. `s3 sync` actualiza los archivos que tengan un tamaño o tiempo de modificación diferente que los archivos con el mismo nombre en el destino. La salida muestra las operaciones específica realizadas durante la sincronización. Tenga en cuenta que la operación sincroniza repetidamente el subdirectorio MySubdirectory y su contenido con s3://my-bucket/path/MySubdirectory.

```
$ aws s3 sync . s3://my-bucket/path
upload: MySubdirectory\MyFile3.txt to s3://my-bucket/path/MySubdirectory/MyFile3.txt
upload: MyFile2.txt to s3://my-bucket/path/MyFile2.txt
upload: MyFile1.txt to s3://my-bucket/path/MyFile1.txt
```

Normalmente, `sync` solo copia archivos u objetos que estén desactualizados o que falten entre el origen y el destino. Sin embargo, puede introducir la opción `--delete` para eliminar archivos u objetos desde el destino que no se encuentran en el origen.

En el siguiente ejemplo, que amplía el anterior, se muestra cómo funciona esto.

```
// Delete local file
$ rm ./MyFile1.txt

// Attempt sync without --delete option - nothing happens
$ aws s3 sync . s3://my-bucket/path

// Sync with deletion - object is deleted from bucket
$ aws s3 sync . s3://my-bucket/path --delete
delete: s3://my-bucket/path/MyFile1.txt

// Delete object from bucket
$ aws s3 rm s3://my-bucket/path/MySubdirectory/MyFile3.txt
delete: s3://my-bucket/path/MySubdirectory/MyFile3.txt

// Sync with deletion - local file is deleted
$ aws s3 sync s3://my-bucket/path . --delete
delete: MySubdirectory\MyFile3.txt

// Sync with Infrequent Access storage class
$ aws s3 sync . s3://my-bucket/path --storage-class STANDARD_IA
```

Las opciones `--exclude` y `--include` le permiten especificar reglas para filtrar los archivos o los objetos que se han de copiar durante la operación de sincronización. De forma predeterminada, todos los elementos de un directorio específico se incluyen en la sincronización. Por lo tanto, solo se necesita `--include` cuando se especifican excepciones a la opción `--exclude` (por ejemplo, `--include` implica en la práctica "no excluir"). Las opciones se aplican en el orden especificado, tal y como se muestra en el siguiente ejemplo.

```
Local directory contains 3 files:
MyFile1.txt
MyFile2.rtf
MyFile88.txt
'''
$ aws s3 sync . s3://my-bucket/path --exclude '*.txt'
upload: MyFile2.rtf to s3://my-bucket/path/MyFile2.rtf
'''
$ aws s3 sync . s3://my-bucket/path --exclude '*.txt' --include 'MyFile*.txt'
upload: MyFile1.txt to s3://my-bucket/path/MyFile1.txt
upload: MyFile88.txt to s3://my-bucket/path/MyFile88.txt
upload: MyFile2.rtf to s3://my-bucket/path/MyFile2.rtf
'''
$ aws s3 sync . s3://my-bucket/path --exclude '*.txt' --include 'MyFile*.txt' --exclude
 'MyFile?.txt'
upload: MyFile2.rtf to s3://my-bucket/path/MyFile2.rtf
```

```
upload: MyFile88.txt to s3://my-bucket/path/MyFile88.txt
```

Las opciones `--exclude` e `--include` también pueden filtrar los archivos u objetos que han de eliminarse durante una operación de sincronización con la opción `--delete`. En este caso, la cadena del parámetro debe especificar qué archivos se deben excluir o incluir en la eliminación, en el contexto del directorio o bucket de destino. A continuación se muestra un ejemplo.

```
Assume local directory and s3://my-bucket/path currently in sync and each contains 3 files:
MyFile1.txt
MyFile2.rtf
MyFile88.txt
'''
// Delete local .txt files
$ rm *.txt

// Sync with delete, excluding files that match a pattern. MyFile88.txt is deleted, while
 remote MyFile1.txt is not.
$ aws s3 sync . s3://my-bucket/path --delete --exclude 'my-bucket/path/MyFile?.txt'
delete: s3://my-bucket/path/MyFile88.txt
'''
// Delete MyFile2.rtf
$ aws s3 rm s3://my-bucket/path/MyFile2.rtf

// Sync with delete, excluding MyFile2.rtf - local file is NOT deleted
$ aws s3 sync s3://my-bucket/path . --delete --exclude './MyFile2.rtf'
download: s3://my-bucket/path/MyFile1.txt to MyFile1.txt
'''
// Sync with delete, local copy of MyFile2.rtf is deleted
$ aws s3 sync s3://my-bucket/path . --delete
delete: MyFile2.rtf
```

El comando `sync` también admite una opción `--acl`, mediante la que es posible configurar los permisos de acceso para archivos copiados en Amazon S3. La opción admite los valores `private`, `public-read` y `public-read-write`.

```
$ aws s3 sync . s3://my-bucket/path --acl public-read
```

Tal y como se ha mencionado anteriormente, el conjunto de comandos `s3` incluye `cp`, `mv`, `ls` y `rm`, y se comportan de forma parecida a sus homólogos de Unix. A continuación se muestran algunos ejemplos.

```
// Copy MyFile.txt in current directory to s3://my-bucket/path
$ aws s3 cp MyFile.txt s3://my-bucket/path/

// Move all .jpg files in s3://my-bucket/path to ./MyDirectory
$ aws s3 mv s3://my-bucket/path ./MyDirectory --exclude '*' --include '*.jpg' --recursive

// List the contents of my-bucket
$ aws s3 ls s3://my-bucket

// List the contents of path in my-bucket
$ aws s3 ls s3://my-bucket/path/

// Delete s3://my-bucket/path/MyFile.txt
$ aws s3 rm s3://my-bucket/path/MyFile.txt

// Delete s3://my-bucket/path and all of its contents
$ aws s3 rm s3://my-bucket/path --recursive
```

Cuando la opción `--recursive` se utiliza en un directorio/carpeta con `cp`, `mv` o `rm`, el comando atraviesa el árbol de directorios, incluidos todos los subdirectorios. Estos comandos también admiten las opciones `--exclude`, `--include` y `--acl` igual que el comando `sync`.

# Uso de comandos de nivel de API (s3api) con la AWS Command Line Interface

Los comandos de nivel de API (contenidos en el conjunto de comandos `s3api`) proporcionan acceso directo a las API de Amazon S3 y habilitan algunas operaciones no expuestas en los comandos de alto nivel. En esta sección se describen los comandos de nivel de API y se ofrecen algunos ejemplos. Para obtener más ejemplos de Amazon S3, consulte la referencia de línea de comandos s3api y seleccione un comando disponible de la lista.

## ACL personalizadas

Con los comandos de alto nivel puede utilizar la opción `--acl` para aplicar listas de control de acceso (ACL) predefinidas en los objetos de Amazon S3, pero no puede configurar ACL para todo el bucket. Puede hacerlo con el comando de nivel de API `put-bucket-acl`. El siguiente ejemplo otorga control total a dos usuarios de AWS (user1@example.com y user2@example.com) y permisos de lectura a todos los usuarios.

```
$ aws s3api put-bucket-acl --bucket MyBucket --grant-full-control
 'emailaddress="user1@example.com",emailaddress="user2@example.com"' --grant-read
 'uri="http://acs.amazonaws.com/groups/global/AllUsers"'
```

Para obtener más información acerca de las ACL personalizadas, consulte PUT Bucket acl. Los comandos ACL `s3api` como, por ejemplo `put-bucket-acl`, usan la misma notación abreviada para el argumento.

## Política de registro

El comando de la API `put-bucket-logging` configura la política de registro del bucket. El siguiente ejemplo establece la política de registro para MyBucket. El usuario de AWS user@example.com tendrá control absoluto sobre los archivos de registro, y todos los usuarios tendrán acceso a ellos. Tenga en cuenta que el comando `put-bucket-acl` es necesario para conceder los permisos necesarios (escritura y lectura acp) al sistema del registro de Amazon S3.

```
$ aws s3api put-bucket-acl --bucket MyBucket --grant-write 'URI="http://acs.amazonaws.com/
groups/s3/LogDelivery"' --grant-read-acp 'URI="http://acs.amazonaws.com/groups/s3/
LogDelivery"'
$ aws s3api put-bucket-logging --bucket MyBucket --bucket-logging-status file://
logging.json
```

logging.json

```
{
  "LoggingEnabled": {
    "TargetBucket": "MyBucket",
    "TargetPrefix": "MyBucketLogs/",
    "TargetGrants": [
      {
        "Grantee": {
          "Type": "AmazonCustomerByEmail",
          "EmailAddress": "user@example.com"
        },
        "Permission": "FULL_CONTROL"
      },
      {
        "Grantee": {
          "Type": "Group",
          "URI": "http://acs.amazonaws.com/groups/global/AllUsers"
        },
```

```
        "Permission": "READ"
      }
    ]
  }
}
```

# Uso de la AWS Command Line Interface con Amazon SNS

En esta sección se describe algunas tareas comunes relacionadas con Amazon Simple Notification Service (Amazon SNS) y cómo realizarlas usando la AWS Command Line Interface.

Temas



## Creación de un tema

El siguiente comando crea un tema denominado **`my-topic`**:

```
$ aws sns create-topic --name my-topic
{
    "TopicArn": "arn:aws:sns:us-west-2:123456789012:my-topic"
}
```

Anote el `TopicArn`, que utilizará más adelante para publicar un mensaje.

## Suscripción a un tema

El siguiente comando permite suscribirse a un tema utilizando el protocolo de correo electrónico y una dirección de correo electrónico para el punto de enlace de notificación:

```
$ aws sns subscribe --topic-arn arn:aws:sns:us-west-2:123456789012:my-topic --protocol
 email --notification-endpoint emailusername@example.com
{
    "SubscriptionArn": "pending confirmation"
}
```

Se enviará un mensaje de correo electrónico a la dirección de correo electrónico indicada en el comando subscribe. El mensaje de correo electrónico tendrá este texto:

```
You have chosen to subscribe to the topic:
arn:aws:sns:us-west-2:123456789012:my-topic
To confirm this subscription, click or visit the following link (If this was in error no
 action is necessary):
Confirm subscription
```

Después de hacer clic en Confirm subscription, debería aparecer un mensaje de suscripción confirmada en su navegador, con información parecida a la siguiente:

```
Subscription confirmed!

You have subscribed emailusername@example.com to the topic:my-topic.

Your subscription's id is:
arn:aws:sns:us-west-2:123456789012:my-topic:1328f057-de93-4c15-512e-8bb2268db8c4

If it was not your intention to subscribe, click here to unsubscribe.
```

## Publicación de un tema

El siguiente comando publica un mensaje sobre un tema:

```
$ aws sns publish --topic-arn arn:aws:sns:us-west-2:123456789012:my-topic --message "Hello
 World!"
{
    "MessageId": "4e41661d-5eec-5ddf-8dab-2c867a709bab"
}
```

Se enviará un mensaje de correo electrónico con el texto "Hello World!" a emailusername@example.com

## Cancelación de la suscripción a un tema

El siguiente comando cancela la suscripción a un tema:

```
$ aws sns unsubscribe --subscription-arn arn:aws:sns:us-west-2:123456789012:my-
topic:1328f057-de93-4c15-512e-8bb2268db8c4
```

Para verificar la cancelación de la suscripción al tema, escriba lo siguiente:

```
$ aws sns list-subscriptions
```

## Eliminación de un tema

El siguiente comando elimina un tema:

```
$ aws sns delete-topic --topic-arn arn:aws:sns:us-west-2:123456789012:my-topic
```

Para verificar la eliminación del tema, escriba lo siguiente:

```
$ aws sns list-topics
```

# Uso de Amazon Simple Workflow Service con la AWS Command Line Interface

Puede acceder a las características de Amazon Simple Workflow Service (Amazon SWF) con la AWS CLI.

Para obtener una lista de comandos y aprender a trabajar con los dominios en Amazon SWF, consulte los temas siguientes.

Temas



# Lista de comandos de Amazon SWF por categoría

En esta sección se muestran los temas de referencia de los comandos Amazon SWF en la AWS CLI. Los comandos se muestran por categoría funcional.

Si desea ver una lista de comandos en orden alfabético, consulte la sección http://docs.aws.amazon.com/cli/latest/reference/swf Amazon SWF de la AWS Command Line Interface Reference o utilice el siguiente comando.

```
$ aws swf help
```

Para obtener ayuda sobre un determinado comando, añada `help` después del nombre del comando. A continuación se muestra un ejemplo.

```
$ aws swf register-domain help
```

Temas



## Comandos relacionados con actividades

Los trabajadores de actividades utilizan `poll-for-activity-task` para obtener nuevas tareas de actividad. En cuanto un trabajador recibe una tarea de actividad de Amazon SWF, realiza la tarea y responde utilizando `respond-activity-task-completed` si ha podido llevarla la cabo o `respond-activity-task-failed` si no lo ha hecho.

Estos son los comandos que realizan los trabajadores de actividad.

- poll-for-activity-task
- respond-activity-task-completed
- respond-activity-task-failed
- respond-activity-task-canceled
- record-activity-task-heartbeat

## Comandos relacionados con decisores

Los decisores utilizan `poll-for-decision-task` para obtener tareas de decisión. En cuanto un decisor recibe una tarea de decisión de Amazon SWF, examina el historial de ejecución del flujo de trabajo y decide qué se debe hacer a continuación. Llama a `respond-decision-task-completed` para completar la tarea de decisión y proporciona cero o más decisiones siguientes.

Estos son los comandos que realizan los decisores.

- poll-for-decision-task
- respond-decision-task-completed

## Comandos relacionados con ejecuciones de flujo de trabajo

Estos son los comandos que se utilizan en una ejecución de flujo de trabajo.

- request-cancel-workflow-execution
- start-workflow-execution
- signal-workflow-execution
- terminate-workflow-execution

## Comandos relacionados con administración

La consola Amazon SWF permite realizar tareas administrativas; sin embargo, puede utilizar los comandos de esta sección para automatizar funciones o crear sus propias herramientas administrativas.

### Administración de actividades

- register-activity-type
- deprecate-activity-type

### Administración de flujos de trabajo

- register-workflow-type
- deprecate-workflow-type

### Administración de dominios

- register-domain
- deprecate-domain

Para obtener más información y ejemplos de estos comandos de administración de dominios, consulte Uso de dominios de Amazon SWF desde la AWS Command Line Interface (p. 87).

### Administración de ejecución de flujos de trabajo

- request-cancel-workflow-execution
- terminate-workflow-execution

## Comandos de visibilidad

La consola Amazon SWF permite realizar acciones de visibilidad. Puede utilizar los comandos de esta sección para automatizar funciones o crear su propia consola o sus propias herramientas de administración.

### Visibilidad de las actividades

- list-activity-types

- describe-activity-type

### Visibilidad de los flujos de trabajo

- list-workflow-types
- describe-workflow-type

### Visibilidad de la ejecución de flujos de trabajo

- describe-workflow-execution
- list-open-workflow-executions
- list-closed-workflow-executions
- count-open-workflow-executions
- count-closed-workflow-executions
- get-workflow-execution-history

### Visibilidad de dominios

- list-domains
- describe-domain

Para obtener más información y ejemplos de estos comandos de visibilidad, consulte Uso de dominios de Amazon SWF desde la AWS Command Line Interface (p. 87).

### Visibilidad de listas de tareas

- count-pending-activity-tasks
- count-pending-decision-tasks

# Uso de dominios de Amazon SWF desde la AWS Command Line Interface

En esta sección se describe cómo realizar tareas de dominio de Amazon SWF comunes mediante la AWS CLI.

Temas



## Visualización de los dominios

Para obtener una lista de los dominios de Amazon SWF que haya registrado en su cuenta, puede utilizar `swf list-domains`. Solo hay un parámetro necesario: `--registration-status`, que puede definir como `REGISTERED` o `DEPRECATED`.

A continuación se muestra un ejemplo mínimo:

```
$ aws swf list-domains --registration-status REGISTERED
{
    "domainInfos": [
        {
            "status": "REGISTERED",
            "name": "ExampleDomain"
        },
        {
            "status": "REGISTERED",
            "name": "mytest"
        }
    ]
}
```

Note

Para ver un ejemplo del uso de `DEPRECATED`, consulte Descartar un dominio (p. 90). Como puede suponer, devuelve los dominios descartados que tenga.

## Configuración de un tamaño de página para limitar los resultados

Si tiene muchos dominios, puede definir el parámetro `--maximum-page-size` para limitar el número de resultados que se devuelven. Si obtiene más resultados que el número máximo especificado, recibirá un `nextPageToken` que puede enviar a la siguiente llamada para `list-domains` y así recuperar entradas adicionales.

A continuación se muestra un ejemplo de cómo utilizar `--maximum-page-size`:

```
$ aws swf list-domains --registration-status REGISTERED --maximum-page-size 1
{
    "domainInfos": [
        {
            "status": "REGISTERED",
            "name": "ExampleDomain"
        }
    ],
    "nextPageToken": "ANeXAMPLEtOKENiSpRETTYlONG=="
}
```

Note

El `nextPageToken` que se le devolverá será mucho más largo. Este valor es simplemente un ejemplo ilustrativo.

Cuando realice de nuevo la llamada, al facilitar en esta ocasión el valor de `nextPageToken` en el argumento `--next-page-token`, obtendrá otra página de resultados:

```
$ aws swf list-domains --registration-status REGISTERED --maximum-page-size 1 --next-page-
token "ANeXAMPLEtOKENiSpRETTYlONG=="
{
    "domainInfos": [
        {
            "status": "REGISTERED",
            "name": "mytest"
        }
    ]
}
```

Cuando no haya más páginas de resultados que recuperar, `nextPageToken` no aparecerá en los resultados.

## Obtener información sobre un dominio

Para obtener información detallada acerca de un determinado dominio, use `swf describe-domain`. Hay un parámetro necesario: `--name`, que lleva el nombre del dominio sobre el que desee información. Por ejemplo:

```
$ aws swf describe-domain --name ExampleDomain
{
    "domainInfo": {
        "status": "REGISTERED",
        "name": "ExampleDomain"
    },
    "configuration": {
        "workflowExecutionRetentionPeriodInDays": "1"
    }
}
```

## Registro de un dominio

Para registrar nuevos dominios, use `swf register-domain`. Existen dos parámetros obligatorios: `--name`, que lleva el nombre de dominio, y `--workflow-execution-retention-period-in-days`, que toma un número entero para especificar el número de días que deben conservarse los datos de ejecución de flujo de trabajo en este dominio, hasta un periodo máximo de 90 días (para obtener más información, consulte las Preguntas frecuentes sobre Amazon SWF). Si especifica cero (0) para este valor, el periodo de retención se ajusta automáticamente a la duración máxima. De lo contrario, los datos de ejecución de flujo de trabajo no se conservarán después de que haya transcurrido el número especificado de días.

A continuación se muestra un ejemplo de cómo registrar un nuevo dominio:

```
$ aws swf register-domain --name MyNeatNewDomain --workflow-execution-retention-period-in-days 0
```

Al registrar un dominio, no se devuelve nada (""), pero puede utilizar `swf list-domains` o `swf describe-domain` para ver el nuevo dominio. Por ejemplo:

```
$ aws swf list-domains --registration-status REGISTERED
{
    "domainInfos": [
        {
            "status": "REGISTERED",
            "name": "ExampleDomain"
        },
        {
            "status": "REGISTERED",
            "name": "MyNeatNewDomain"
        },
        {
            "status": "REGISTERED",
            "name": "mytest"
        }
    ]
}
```

A continuación se muestra un ejemplo de cómo utilizar `swf describe-domain`:

```
$ aws swf describe-domain --name MyNeatNewDomain
```

```
{
    "domainInfo": {
        "status": "REGISTERED",
        "name": "MyNeatNewDomain"
    },
    "configuration": {
        "workflowExecutionRetentionPeriodInDays": "0"
    }
}
```

## Descartar un dominio

Para descartar un dominio (todavía podrá verlo, pero no podrá crear nuevas ejecuciones de flujo de trabajo ni tipos de registro en él), use `swf deprecate-domain`. Solo tiene un parámetro necesario, `--name`, que toma el nombre del dominio que se va a descartar.

```
$ aws swf deprecate-domain --name MyNeatNewDomain
```

Al igual que ocurre con `register-domain`, no se devuelve ningún resultado. Si usa `list-domains` para ver los dominios registrados, sin embargo, verá que el dominio ya no aparece entre ellos.

```
$ aws swf list-domains --registration-status REGISTERED
{
    "domainInfos": [
        {
            "status": "REGISTERED",
            "name": "ExampleDomain"
        },
        {
            "status": "REGISTERED",
            "name": "mytest"
        }
    ]
}
```

Puede consultar los dominios descartados usando `--registration-status DEPRECATED` con `list-domains`.

```
$ aws swf list-domains --registration-status DEPRECATED
{
    "domainInfos": [
        {
            "status": "DEPRECATED",
            "name": "MyNeatNewDomain"
        }
    ]
}
```

También puede usar `describe-domain` para obtener información sobre un dominio descartado.

```
$ aws swf describe-domain --name MyNeatNewDomain
{
    "domainInfo": {
        "status": "DEPRECATED",
        "name": "MyNeatNewDomain"
    },
    "configuration": {
        "workflowExecutionRetentionPeriodInDays": "0"
    }
```

```
}
```

## Véase también

- descartar-dominio en la AWS Command Line Interface Reference
- describir-dominio en la AWS Command Line Interface Reference
- mostrar-dominios en la AWS Command Line Interface Reference
- registrar-dominio en la AWS Command Line Interface Reference

# Solución de errores de la AWS CLI

Después de instalar con `pip`, es posible que tenga que añadir el ejecutable `aws` la variable de entorno `PATH` de su sistema operativo, o cambiar su modo para que sea ejecutable.

Error: aws: no se ha encontrado el comando

Es posible que tenga que añadir el ejecutable `aws` a la variable de entorno `PATH` del sistema operativo.

- Windows – Añadir el ejecutable de la AWS CLI a su ruta de la línea de comandos (p. 11)
- macOS – Añadir el ejecutable de la AWS CLI a su ruta de la línea de comandos (p. 13)
- Linux – Añadir el ejecutable de la AWS CLI a su ruta de la línea de comandos (p. 8)

Si `aws` está en la variable `PATH` y sigue apareciendo este error, es posible que no tenga el modo de archivo correcto. Pruebe a ejecutarlo directamente.

```
$ ~/.local/bin/aws --version
```

Error: permiso denegado

Asegúrese de que el script de `aws` tiene un modo de archivo ejecutable. Por ejemplo, `755`.

Ejecute `chmod +x` para hacer que el archivo sea ejecutable.

```
$ chmod +x ~/.local/bin/aws
```

Error: AWS no ha podido validar las credenciales proporcionadas

Puede que la AWS CLI esté leyendo credenciales de otra ubicación distinta de la que esperaba. Ejecute `aws configure list` para confirmar que se utilizan las credenciales correctas.

```
$ aws configure list
      Name                    Value             Type    Location
      ----                    -----             ----    --------
   profile                <not set>             None    None
access_key     ****************XYVA shared-credentials-file
secret_key     ****************ZAGY shared-credentials-file
    region                us-west-2      config-file    ~/.aws/config
```

Si se están utilizando las credenciales correctas, puede que el reloj no esté sincronizado. En Linux, macOS, or Unix, ejecute `data` para comprobar la hora.

```
date
```

Si el reloj del sistema no está en hora, utilice `ntpd` para sincronizarlo.

```
sudo service ntpd stop
sudo ntpdate time.nist.gov
sudo service ntpd start
ntpstat
```

En Windows, utilice las opciones de fecha y hora del panel de control para configurar el reloj del sistema.

Error: se produjo un error (UnauthorizedOperation) al llamar a la operación *`CreateKeyPair`*: No tiene autorización para realizar esta operación.

Su usuario o rol de IAM necesita permisos para llamar a las acciones de la API que corresponden a los comandos que ejecuta con la CLI de AWS. La mayoría de los comandos llaman a una sola acción con un nombre que coincide con el nombre del comando; sin embargo, los comandos personalizados como `aws s3 sync` llaman a varias API. Mediante la `--debug` opción puede ver a qué API llama un comando.