AWS Command Line Interface

Guia do usuário

AWS Command Line Interface: Guia do usuário

Table of Contents

O que é o AWS Command Line Interface?

A AWS CLI é uma ferramenta de código aberto que permite interagir com os serviços da AWS usando comandos no shell da linha de comando. Com o mínimo de configuração, você pode começar a usar uma funcionalidade equivalente à fornecida pelo Console de gerenciamento da AWS baseado em navegador no prompt de comando do seu programa de terminal preferencial.

- Shells do Linux – Use programas comuns de shell, como `bash`, `zsh` e `tsch`, para executar comandos em Linux, macOS, or Unix.
- Linha de comando do Windows – No Microsoft Windows, execute comandos no PowerShell ou no prompt de comando do Windows.
- Remotamente – Execute comandos em instâncias do Amazon EC2 por meio de um terminal remoto, como PuTTY ou SSH, ou com o gerenciador de sistemas do Amazon EC2.

Toda a administração, gerenciamento e funções de acesso da AWSIaaS (Infraestrutura como um serviço) no Console de gerenciamento da AWS estão disponíveis na API e CLI da AWS. Os novos recursos e serviços da AWS IaaS fornecem funcionalidade completa do Console de gerenciamento da AWS por meio da API e da CLI no lançamento ou dentro de 180 dias após o lançamento.

A AWS CLI fornece acesso direto a APIs públicas de serviços da AWS. Você pode explorar os recursos de um serviço com a AWS CLI e desenvolver scripts de shell para gerenciar seus recursos. Ou você pode utilizar o que aprender para desenvolver programas em outras linguagens com o SDKs da AWS.

Além dos comandos de nível inferior equivalentes a API, vários serviços da AWS fornecem personalizações para a AWS CLI. As personalizações podem incluir comandos de nível mais elevado que simplificam o uso de um serviço com uma API complexa. Por exemplo, o conjunto de comandos `aws s3` fornece uma sintaxe familiar para o gerenciamento de arquivos no Amazon S3.

Example Para fazer upload de um arquivo ao Amazon S3

`aws s3 cp` fornece um comando de cópia do tipo shell, e automaticamente realiza um multipart upload para transferir arquivos grandes de maneira rápida e resiliente.

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

Executar a mesma tarefa com os comandos de nível inferior (disponíveis em `aws s3api`) levaria muito mais esforço.

Dependendo do seu caso de uso, você pode usar o SDK da AWS, um kit de ferramentas ou o AWS Tools para Windows PowerShell.

- AWS Tools para Windows PowerShell
- AWS SDK for Java
- AWS SDK para .NET

  AWS SDK for JavaScript
- AWS SDK para Ruby
- AWS SDK for Python (Boto)
- AWS SDK para PHP

- AWS SDK para Go
- AWS Toolkit for Eclipse
- AWS Toolkit for Visual Studio
- AWS Mobile SDK for iOS
- AWS Mobile SDK para Android

Você pode ver — e bifurcar — o código-fonte para a AWS CLI no GitHub no repositório aws-cli. Faça parte da comunidade de usuários no GitHub para fornecer feedback, solicitar recursos e enviar suas próprias contribuições!

Usar os exemplos neste guia

Os exemplos neste guia estão formatados com as seguintes convenções:

- Prompt – O prompt de comando é exibido como um sinal de dólar seguido por um espaço ("$ "). Não inclua prompt quando você digitar comandos.
- Diretório – Quando comandos devem ser executados de um diretório específico, o nome do diretório é mostrado antes do símbolo do comando.
- Entrada do usuário – Texto de comando inserido na linha de comando é formatado como **`user input`**.
- Texto de substituição – O texto variável, incluindo nomes de recursos que você escolher ou IDs gerados pelos serviços da AWS, que você deve incluir em comandos é formatado como *`texto de substituição`*. Em vários comandos ou linha de comandos, onde a entrada de teclado específico é necessária, os comandos do teclado substituíveis também podem ser mostrados como texto.
- Saída – A saída retornada pelos serviços da AWS é mostrada abaixo de entrada do usuário formatada como `computer output`.

Por exemplo, o seguinte comando inclui entradas do usuário, substituição de texto e saída:

```
$ 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 esse exemplo, digite **`aws configure`** na linha de comando e pressione ENTER. **`aws configure`** é o comando. Este comando é interativo, de modo que o AWS CLI resulta em linhas de textos, solicitando que você fornaça informações adicionais. Insira cada uma de suas chaves de acesso e pressione ENTER. Em seguida, insira um nome de região no formato mostrado, pressione ENTER e pressione ENTER uma última vez para ignorar o formato de saída. O comando final ENTER é mostrado como texto substituível porque não há entradas do usuário para essa linha. Caso contrário, ele seria implícito.

O exemplo a seguir mostra um comando não interativo simples com a saída de serviço em formato JSON:

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

Para usar este exemplo, insira o texto completo do comando (o texto destacado após o prompt) e pressione ENTER. O nome do grupo de segurança *`my-sg`* é substituível. Neste caso, você pode usar o nome do grupo, conforme mostrado, mas use um nome mais descritivo.

Note

Argumentos que devem ser substituídos (como ID da chave de acesso da AWS) e aqueles que devem ser substituídos (como nome do grupo) aparecem como *`texto de substituição`*. Se um argumento deve ser substituído, ele será mencionado no texto que descreve o exemplo.

O documento JSON, incluindo as chaves, é saída. Se configurar o CLI para resultar em texto ou formato de tabela, a saída será formatada de forma diferente. JSON é o formato de saída padrão.

Sobre a Amazon Web Services

A Amazon Web Services (AWS) é um conjunto de serviços de infraestrutura digital que os desenvolvedores podem utilizar ao desenvolver seus aplicativos. Os serviços incluem computação, armazenamento, banco de dados e sincronização de aplicativos (sistema de mensagens e filas). A AWS usa um modelo de serviço de pagamento por utilização. Você será cobrado apenas pelos serviços que você ou seus aplicativos usarem. Além disso, para tornar AWS mais acessível como plataforma para criação de protótipos e experimentação, a AWS oferece um nível de uso gratuito. Neste nível, os serviços são gratuitos abaixo de um determinado nível de uso. Para obter mais informações sobre os custos da AWS e o nível gratuito, consulte Test-Driving AWS in the Free Usage Tier. Para obter uma conta da AWS, abra a Página inicial da AWS e clique em Cadastrar-se.

Instalação do AWS Command Line Interface

Formas de instalar a AWS CLI

Requisitos

- Python 2 versão 2.6.5+ ou Python 3 versão 3.3+
- Windows, Linux, macOS, or Unix

> **Note**
>
> As versões mais antigas do Python podem não funcionar com todos os serviços da AWS. Se você vir `InsecurePlatformWarning` ou notificações de substituição quando instalar ou usar a &CLI;, atualize para uma versão mais recente.

Instalar a AWS CLI usando o *pip*

O método de distribuição principal para a AWS CLI; no Linux, Windows e macOS é `pip`, um gerenciador de pacote para Python que fornece uma maneira fácil de instalar, atualizar e remover pacotes Python e suas dependências.

Versão AWS CLI atual

O AWS CLI é atualizado com frequência para se tornar compatível com novos serviços e comandos. Para verificar se a versão mais recente está instalada, consulte a página de versões no GitHub.

Se você já tem `pip` e uma versão compatível do Python, é possível instalar a AWS CLI com o seguinte comando:

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

A opção `--upgrade` informa ao `pip` para atualizar os requisitos que já estão instalados. A opção `--user` informa ao `pip` para instalar o programa em um subdiretório do seu diretório de usuário para evitar a modificação de bibliotecas usadas pelo seu sistema operacional.

Instalar a AWS CLI em um ambiente virtual

Caso haja problemas ao tentar instalar a AWS CLI com o `pip`, é possível instalá-la em um ambiente virtual (p. 15) para isolar a ferramenta e suas dependências ou usar uma versão diferente do Python que você normalmente usa.

Instalar a AWS CLI usando um instalador

Para instalações automatizadas ou off-line fornecidas no Linux, macOS, or Unix, experimente o pacote do instalador (p. 16). O pacote de instalador inclui o AWS CLI, suas dependências e um script de shell que executa a instalação para você.

No Windows, também é possível usar o instalador MSI (p. 10). Esses dois métodos simplificam a instalação inicial, sendo mais difícil atualizar quando uma nova versão do AWS CLI é lançada.

Etapas a serem realizadas após a instalação

Após instalar a AWS CLI, talvez seja preciso adicionar o caminho para o arquivo executável à sua variável PATH. Para instruções específicas da plataforma, consulte os seguintes tópicos:

- Linux – Adicione o executável AWS CLI em seu caminho de linha de comando (p. 7)
- Windows – Adicione o executável AWS CLI em seu caminho de linha de comando (p. 12)
- macOS – Adicione o executável AWS CLI em seu caminho de linha de comando (p. 14)

Verifique se a AWS CLI foi instalada corretamente executando `aws --version`.

```
$ aws --version
aws-cli/1.16.67 Python/3.7.1 Linux/4.14.77-81.59-amzn2.x86_64 botocore/1.12.57
```

O AWS CLI é atualizado regularmente para ser compatível com novos serviços e comandos. Para atualizar para a versão mais recente do AWS CLI, execute o comando de instalação novamente. Para obter detalhes sobre a versão mais recente do AWS CLI, consulte as notas de release do AWS CLI.

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

Se precisar desinstalar a AWS CLI, use `pip uninstall`.

```
$ pip uninstall awscli
```

Se não tem Python nem `pip`, use o procedimento para o seu sistema operacional:

Instruções detalhadas para cada ambiente

- Instalar o AWS Command Line Interface no Linux (p. 5)
- Instalar o AWS Command Line Interface no Microsoft Windows (p. 10)
- Instalar o AWS Command Line Interface em macOS (p. 12)
- Instalar o AWS Command Line Interface em um ambiente virtual (p. 15)
- Instalar a AWS CLI usando o Bundled Installer (Linux, macOS, or Unix) (p. 16)

Instalar o AWS Command Line Interface no Linux

Instale o AWS Command Line Interface e suas dependências na maioria das distribuições do Linux com o `pip`, um gerenciador de pacote para Python.

Important

O pacote `awscli` está disponível em repositórios para outros gerenciadores de pacote, como APT e yum, mas não é garantido que você obtenha a versão mais recente, a menos que obtenha de `pip` ou usando o pacote de instalador (p. 16).

Caso já tenha pip, siga as instruções no tópico de instalação (p. 4) principal. Execute `pip --version` para ver se a sua versão do Linux já inclui Python e pip.

```
$ pip --version
```

Se não houver pip, verifique qual versão do Python está instalada.

```
$ python --version
```

ou

```
$ python3 --version
```

Se você ainda não tem o Python 2 versão 2.6.5+ ou Python 3 versão 3.3+, você deverá primeiro instalar o Python (p. 8). Se você já tiver o Python instalado, prossiga para instalar o pip e a AWS CLI.

Seções

Instalar o Pip

Se você ainda não tem o `pip` instalado, você pode instalá-lo com o script fornecido pelo Python Packaging Authority.

Para instalar o pip

1. Use o comando `curl` a seguir para fazer download do script de instalação:

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

2. Execute o script com Python para fazer download e instalar a versão mais recente do `pip` e de outros pacotes de suporte necessários.

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

   Quando você inclui a chave `--user`, o script instala o `pip` no caminho `~/.local/bin`.

3. Certifique-se de que o caminho com pip faça parte da sua variável PATH.

   a. Encontre o script de perfil do shell em sua pasta de usuário. Se você não tiver certeza de qual shell você tem, execute `echo $SHELL`.

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

- Bash – `.bash_profile`, `.profile` ou `.bash_login`.
- Zsh – `.zshrc`
- Tcsh – `.tcshrc`, `.cshrc` ou `.login`.

b. Adicione um comando de exportação ao final do script de perfil semelhante ao exemplo a seguir:

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

Esse comando adiciona o caminho `~/.local/bin` neste exemplo na frente da variável PATH atual.

c. Recarregue o perfil em sua sessão atual para colocar essas alterações em vigor.

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

4. Agora você pode fazer o teste para verificar se o pip está instalado corretamente.

```
$ pip --version
pip 18.1 from ~/.local/lib/python3.7/site-packages (python 3.7)
```

Instalar a AWS CLI com Pip

Use o `pip` para instalar a AWS CLI.

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

Quando você usa a chave `--user`, o `pip` instala a AWS CLI como `~/.local/bin`.

Verifique se o AWS CLI está instalado corretamente.

```
$ aws --version
aws-cli/1.16.67 Python/3.7.1 Linux/4.14.77-81.59-amzn2.x86_64 botocore/1.12.57
```

Se houver um erro, consulte Solução de problemas de erros do AWS CLI (p. 97).

Para atualizar para a versão mais recente, execute o comando de instalação novamente:

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

Adicione o executável AWS CLI em seu caminho de linha de comando

Após a instalação com `pip`, talvez seja necessário adicionar o arquivo executável `aws` à variável de ambiente `PATH` do SO.

Você pode verificar em qual pasta o pip instalou a AWS CLI executando este comando:

```
$ which aws
```

```
/home/username/.local/bin/aws
```

Você pode fazer referência a isso como `~/.local/bin/` porque `/home/username` corresponde a `~` no Linux.

Se você omitiu a chave `--user` e, portanto, não a instalou no modo usuário, o arquivo executável poderá estar na pasta `bin` de sua instalação do Python. Se você não souber onde o Python está instalado, execute este comando:

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

A saída pode ser o caminho para um symlink, e não o arquivo executável real. Execute `ls -al` para saber para onde ele aponta.

```
$ ls -al /usr/local/bin/python
/usr/local/bin/python -> ~/.local/Python/3.7/bin/python3.7
```

Se esta for a mesma pasta que você adicionou ao caminho na etapa 3 em Instalar o Pip (p. 6), então terá concluído. Caso contrário, execute as mesmas etapas 3a a 3c novamente, adicionando essa pasta adicional ao caminho.

Instalar o Python no Linux

Se a sua distribuição não veio com Python, ou veio com uma versão mais antiga, instale o Python antes de instalar pip e o AWS CLI.

Para instalar Python 3 no Linux

1. Verifique se o Python já está instalado:

   ```
   $ python --version
   ```

   Note

   Se a sua distribuição do Linux acompanha Python, pode ser necessário instalar o pacote do desenvolvedor Python para ter acesso aos cabeçalhos e bibliotecas necessários para compilar extensões e instalar o AWS CLI. Instale o pacote do desenvolvedor (geralmente chamado `python-dev` ou `python-devel`) usando o gerenciador de pacote.

2. Se Python 2.7 ou posterior não estiver instalado, instale Python com o gerenciador de pacote de distribuição. O comando e o nome do pacote varia de:

   - No derivados do Debian, como Ubuntu, use `APT`:

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

   - No Red Hat e derivados, use `yum`:

     ```
     $ sudo yum install python
     ```

   - Na SUSE e derivados, use `zyppero` :

     ```
     $ sudo zypper install python3
     ```

3. Abra um prompt de comando ou shell e execute o comando a seguir para verificar se o Python foi instalado corretamente:

```
$ python --version
Python 2.7.15
```

Como instalar a AWS Command Line Interface no Amazon Linux

A AWS CLI vem pré-instalada no Amazon Linux e no Amazon Linux 2. Verifique a versão atualmente instalada usando o comando a seguir.

```
$ aws --version
aws-cli/1.16.67 Python/3.7.1 Linux/4.14.77-81.59.amzn2.x86_64 botocore/1.12.57
```

Você pode usar `sudo yum update` para obter a versão mais recente disponível no repositório yum, mas talvez essa não seja a versão mais recente. Em vez disso, recomendamos que você use o `pip` para obter a versão mais recente.

Pré-requisitos

Verifique se o Python e pip já estão instalados. Para obter mais informações, consulte Instalar o AWS Command Line Interface no Linux (p. 5).

Como atualizar a AWS CLI no Amazon Linux (raiz)

1. Use o `pip install` para instalar a versão mais recente da AWS CLI.

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

2. Verifique a nova versão com `aws --version`.

```
$ aws --version
aws-cli/1.16.67 Python/3.7.1 Linux/4.14.77-81.59.amzn2.x86_64 botocore/1.12.57
```

Se você não tiver privilégios de raiz, instale a AWS CLI no modo de usuário.

Como atualizar a AWS CLI no Amazon Linux (usuário)

1. Use o `pip install` para instalar a versão mais recente da AWS CLI.

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

2. Adicione o local de instalação ao início da variável `PATH`.

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

   Adicione esse comando ao fim de `~/.bashrc` para manter a alteração entre sessões.

3. Verifique a nova versão com `aws --version`.

```
$ aws --version
```

```
aws-cli/1.16.67 Python/3.7.1 Linux/4.14.77-81.59.amzn2.x86_64 botocore/1.12.57
```

Instalar o AWS Command Line Interface no Microsoft Windows

Você pode instalar a AWS CLI no Windows com um instalador independente ou o `pip`, um gerenciador de pacote para Python. Caso já tenha `pip`, siga as instruções no tópico de instalação (p. 4) principal.

Seções

Instalador MSI

O AWS CLI é compatível com o Microsoft Windows XP ou posterior. Para usuários do Windows, o pacote de instalação MSI oferece uma maneira familiar e conveniente para instalar o AWS CLI sem necessitar de qualquer outro pré-requisitos.

Quando as atualizações são lançadas, repita o processo de instalação para instalar a versão mais recente do AWS CLI. Se preferir atualizar frequentemente, considere usar o pip (p. 11) para facilitar as atualizações.

Instalar o AWS CLI usando o instalador MSI

1. Faça o download do instalador MSI apropriado.

   - Faça download do instalador MSI da AWS CLI para Windows (64 bits)
   - Faça download do instalador MSI da AWS CLI para Windows (32 bits)
   - Fazer download do arquivo de configuração da AWS CLI (inclui os instaladores MSI de 32 e 64 bits e instala automaticamente a versão correta)

   Note

   O instalador MSI para a AWS CLI não funciona com o Windows Server 2008 (versão 6.0.6002). Use pip (p. 11) para instalar com esta versão do Windows.
2. Execute o instalador MSI obtido por download ou o arquivo de configuração.
3. Siga as instruções da tela.

A CLI instala para `C:\Program Files\Amazon\AWSCLI` (versão de 64 bits) ou `C:\Program Files (x86)\Amazon\AWSCLI` (versão de 32 bits) por padrão. Para confirmar a instalação, use o comando `aws --version` em um prompt de comando (você pode abrir o menu Start (Iniciar) e pesquisar `cmd` para iniciar um prompt de comando).

```
C:\> aws --version
aws-cli/1.16.67 Python/3.7.1 Windows/10 botocore/1.12.57
```

Não inclua o símbolo de prompt ("C:\>" acima) ao digitar um comando. Essas listas são incluídas no programa para diferenciar comandos que tipo de saída retornado pela CLI. O restante desse guia usa o símbolo de prompt genérico "$", exceto nos casos em que um comando é específico do Windows.

Se o Windows não conseguir localizar o programa, talvez seja necessário fechar e reabrir o prompt de comando para atualizar o caminho ou adicionar o diretório de instalação à variável de ambiente PATH (p. 12) manualmente.

Atualizar uma instalação MSI

A AWS CLI é atualizado regularmente. Confira a página de Liberações no GitHub para ver quando a versão mais recente foi lançada. Para atualizar para a versão mais recente, faça o download e execute o instalador MSI novamente conforme detalhado acima.

Desinstalar

Para desinstalar a AWS CLI, abra o Control Panel (Painel de controle) e selecione Programs and Features (Programas e recursos). Selecione a entrada denominada AWS Command Line Interface e clique em Uninstall (Desinstalar) para executar o desinstalador. Confirme se você deseja desinstalar o AWS CLI quando solicitado.

Você também pode executar o programa Programs and Features (Programas e recursos) a partir da linha de comando com o seguinte comando:

```
C:\> appwiz.cpl
```

Instalar Python, pip, e o AWS CLI no Windows

O Python Software Foundation fornece instaladores para Windows que incluem `pip`.

Para instalar o Python3 e pip (Windows)

1. Faça download do instalador do Python3 Windows x86-64 na página de downloads do Python.org.
2. Execute o instalador.
3. Escolha Add Python 3 to PATH (Adicionar Python 3 ao PATH).
4. Escolha Instalar agora.

O instalador instala o Python em sua pasta de usuário e adiciona suas pastas do programa ao caminho do usuário.

Para instalar o AWS CLI com pip (Windows)

1. Abra o Windows Command Prompt (prompt de comando do Windows) no menu Start (Iniciar).
2. Verifique se o Python e pip são instalados corretamente com os seguintes comandos:

   ```
   C:\Windows\System32> python --version
   Python 3.7.1
   C:\Windows\System32> pip --version
   pip 18.1 from c:\program files\python37\lib\site-packages\pip (python 3.7)
   ```

3. Instale a AWS CLI usando o `pip`:

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

4. Verifique se o AWS CLI está instalado corretamente:

   ```
   C:\Windows\System32> aws --version
   aws-cli/1.16.67 Python/3.7.1 Windows/10 botocore/1.12.57
   ```

Para atualizar para a versão mais recente, execute o comando de instalação novamente:

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

Adicione o executável AWS CLI em seu caminho de linha de comando

Após a instalação com o `pip`, adicione o programa `aws` à variável de ambiente `PATH` do seu sistema operacional. Com uma instalação MSI, isso deve acontecer automaticamente, mas pode ser necessário definir manualmente se o comando `aws` não estiver funcionando depois que você fizer a instalação dele.

Você pode encontrar onde o programa aws está instalado executando o seguinte comando:

```
C:\> where aws
C:\Program Files\Python37\Scripts\aws
```

Os caminhos típicos incluem:

- Python 3 e pip – `C:\Program Files\Python37\Scripts\`
- Python 3 e pip – opção do usuário – `%USERPROFILE%\AppData\Local\Programs\Python\Python37\Scripts`
- Instalador MSI (64 bits) – `C:\Program Files\Amazon\AWSCLI`
- Instalador MSI (32 bits) – `C:\Program Files (x86)\Amazon\AWSCLI`

> **Note**
>
> Os nomes de pasta que incluem os números de versão podem variar.

Para modificar sua variável PATH (Windows)

1. Pressione a tecla Windows e digite **`environment variables`**.
2. Escolha Edit environment variables for your account.
3. Selecione PATH e, em seguida, Editar.
4. Adicione caminhos ao campo Variable value, separados por ponto e vírgula. Por exemplo: *`C:\existing\path`*`;`***`C:\new\path`**
5. Selecione OK duas vezes para aplicar as novas configurações.
6. Feche todas as solicitações de comando em execução e abra novamente.

Instalar o AWS Command Line Interface em macOS

A maneira recomendada de instalar a AWS CLI no macOS é usar o instalador incluído. O pacote de instalador inclui todas as dependências do pacote e pode ser usado off-line.

> **Important**
>
> O pacote de instalador fornecido não é compatível com a instalação em caminhos com espaços.

Seções

Pré-requisitos

- Python 2 versão 2.6.5+ ou Python 3 versão 3.3+

Verifique a instalação do Python:

```
$ python --version
```

Se o Python ainda não foi instalado no computador, ou se deseja instalar uma versão diferente do Python, siga as instruções Instalar o AWS Command Line Interface no Linux (p. 5).

Instalar o AWS CLI usando o Bundled Installer

Siga estas etapas a partir da linha de comando para instalar o AWS CLI usando o pacote de instalador.

Instalar o AWS CLI usando o pacote de instalador

1. Faça o download do Bundled Installer da AWS CLI.

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

2. Descompacte o pacote.

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

   > **Note**
   >
   > Se você não tiver `unzip`, use a distribuição do Linux incluída no gerenciador de pacote para fazer a instalação.

3. Execute o programa de instalação.

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

   > **Note**
   >
   > Por padrão, o script de instalação é executado sob a versão padrão do sistema do Python. Se uma versão diferente do Python estiver instalada, mas deseja usá-la para instalar a CLI da AWS, execute o script de instalação especificando essa versão incluindo o caminho absoluto para o programa Python. Por exemplo:
   >
   > ```
   > $ sudo /usr/local/bin/python3.6 awscli-bundle/install -i /usr/local/aws -b /usr/
   > local/bin/aws
   > ```

Esse comando instala a AWS CLI em `/usr/local/aws` e cria o symlink `aws` no diretório `/usr/local/bin`. Usar a opção `-b` para a criação de um symlink elimina a necessidade de especificar o diretório de instalação na variável `$PATH` do usuário. Isso permite que todos os usuários chamem a AWS CLI ao digitar `aws` a partir de qualquer diretório.

Para ver uma explicação das opções `-i` e `-b`, use a opção `-h`:

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

Instalar a AWS CLI no macOS usando o pip

Você também pode usar o pip diretamente para instalar a AWS CLI. Se você não tem o pip, siga as instruções no tópico de instalação (p. 4) principal. Execute `pip --version` para ver se a sua versão do macOS já inclui Python e pip.

```
$ pip --version
```

Instalar o AWS CLI no macOS

1. Faça o download e instale o Python 3.6 na página de download em Python.org.
2. Faça download e execute o script de instalação `pip` fornecido pela Python Packaging Authority.

   ```
   $ curl -O https://bootstrap.pypa.io/get-pip.py
   $ python3 get-pip.py --user
   ```
3. Use o `pip` recém-instalado para instalar a AWS CLI.

   ```
   $ pip install awscli --upgrade --user
   ```
4. Verifique se o AWS CLI está instalado corretamente.

   ```
   $ aws --version
   AWS CLI 1.16.67 (Python 3.7.1)
   ```

   Se o programa não for encontrado, adicione-o ao caminho da linha de comando (p. 14).

Para atualizar para a versão mais recente, execute o comando de instalação novamente:

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

Adicione o executável AWS CLI em seu caminho de linha de comando

Após a instalação com `pip`, talvez seja necessário adicionar o programa `aws` à variável de ambiente `PATH` do SO. A localização do programa depende de onde o Python está instalado.

Example Local de instalação da AWS CLI – macOS com Python 3.7 e pip (modo usuário)

```
~/Library/Python/3.7/bin
```

Se você não souber onde o Python está instalado, execute `which python`.

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

A saída pode ser o caminho para um symlink, e não o programa real. Execute `ls -al` para saber para onde ele aponta.

```
$ ls -al /usr/local/bin/python
~/Library/Python/3.7/bin/python3.7
```

O `pip` instala programas na mesma pasta que contém o programa Python. Adicione esta pasta a sua variável `PATH`.

Para modificar sua variável PATH (Linux, macOS, or Unix)

1. Encontre o script de perfil do shell em sua pasta de usuário. Se você não tiver certeza de qual shell você tem, execute `echo $SHELL`.

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

   - Bash – `.bash_profile`, `.profile` ou `.bash_login`.
   - Zsh – `.zshrc`
   - Tcsh – `.tcshrc`, `.cshrc` ou `.login`.
2. Adicione um comando de exportação ao script de perfil.

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

   Este comando adiciona um caminho, `~/.local/bin` neste exemplo, para a variável PATH atual.
3. Carregue o perfil em sua sessão atual.

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

Instalar o AWS Command Line Interface em um ambiente virtual

Instale o AWS CLI em um ambiente virtual para evitar conflitos de versão de requisito com outros pacotes pip.

Instalar o AWS CLI em um ambiente virtual.

1. Instale `virtualenv` usando o `pip`.

   ```
   $ pip install --user virtualenv
   ```
2. Crie um ambiente virtual e atribua um nome a ele.

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

   Como alternativa, você pode usar a opção `-p` para especificar uma versão do Python que não seja a padrão.

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

3. Ative seu novo ambiente virtual.

   Linux, macOS, or Unix

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

   Windows

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

4. Instale a AWS CLI em seu ambiente virtual.

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

5. Verifique se o AWS CLI está instalado corretamente.

   ```
   $ aws --version
   aws-cli/1.16.67 Python/3.7.1 Linux/4.14.77-81.59-amzn2.x86_64 botocore/1.12.57
   ```

Use o comando `deactivate` para sair do ambiente virtual. Sempre que você iniciar uma nova sessão, deverá ativar o ambiente novamente.

Para atualizar para a versão mais recente, execute o comando de instalação novamente:

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

Instalar a AWS CLI usando o Bundled Installer (Linux, macOS, or Unix)

No Linux, macOS, or Unix, é possível usar o Bundled Installer para instalar a AWS CLI. O pacote de instalador inclui todas as dependências do pacote e pode ser usado off-line.

> **Important**
>
> O pacote de instalador fornecido não é compatível com a instalação em caminhos com espaços.

Seções

Pré-requisitos

- Linux, macOS, or Unix
- Python 2 versão 2.6.5+ ou Python 3 versão 3.3+

Verifique a instalação do Python:

```
$ python --version
```

Se o Python ainda não foi instalado no computador, ou se deseja instalar uma versão diferente do Python, siga as instruções Instalar o AWS Command Line Interface no Linux (p. 5).

Instalar o AWS CLI usando o Bundled Installer

Siga estas etapas a partir da linha de comando para instalar o AWS CLI usando o pacote de instalador.

Instalar o AWS CLI usando o pacote de instalador

1. Faça o download do Bundled Installer da AWS CLI.

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

2. Descompacte o pacote.

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

   Note

   Se você não tiver `unzip`, use a distribuição do Linux incluída no gerenciador de pacote para fazer a instalação.

3. Execute a instalação executável.

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

   Note

   Por padrão, o script de instalação é executado sob a versão padrão do sistema do Python. Se uma versão diferente do Python estiver instalada, mas deseja usá-la para instalar o AWS CLI, execute o script de instalação com essa versão por caminho absoluto para o Python executável. Por exemplo:

   ```
   $ sudo /usr/local/bin/python3.7 awscli-bundle/install -i /usr/local/aws -b /usr/
   local/bin/aws
   ```

O instalador instala a AWS CLI no `/usr/local/aws` e cria o symlink `aws` no diretório `/usr/local/bin`. Usar a opção `-b` para a criação de um symlink elimina a necessidade de especificar o diretório de instalação na variável `$PATH` do usuário. Isso permite que todos os usuários chamem a AWS CLI ao digitar `aws` a partir de qualquer diretório.

Para ver uma explicação das opções `-i` e `-b`, use a opção `-h`:

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

Instalar o AWS CLI Without Sudo (Linux, macOS, or Unix)

Caso não tenha permissões sudo ou deseje instalar o AWS CLI apenas para o usuário atual, use uma versão modificada dos comandos acima:

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

Isso instalará a AWS CLI no local padrão (`~/.local/lib/aws`) e criará um link simbólico (symlink) em `~/bin/aws`. Verifique se o `~/bin` está na sua variável de ambiente `PATH` para que o 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 garantir que suas configurações $ PATH sejam mantidas entre sessões, adicione a linha `export` em seu perfil de shell (~ /.profile, ~ /.bash_profile, etc).

Desinstalar

O pacote de instalador não retira nada do diretório de instalação, exceto o opcional symlink. Portanto, a desinstalação é feita da mesma maneira destes dois itens.

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

Configurar o AWS CLI

Esta seção explica como definir as configurações que a AWS CLI usa para interagir com a AWS, inclusive suas credenciais de segurança, o formato de saída padrão e a região padrão.

> Note
>
> A AWS exige que todas as solicitações recebidas sejam assinadas criptograficamente. A AWS CLI faz isso para você. A "assinatura"" inclui uma data/time stamp. Portanto, você deve garantir que a data e a hora do seu computador estejam definidas corretamente. Se não fizer isso, e a data/hora na assinatura estiver muito longe da data/hora reconhecida pelo serviço da AWS, a AWS rejeitará a solicitação.

Seções

Configuração Rápida

Para uso geral, o comando `aws configure` é a maneira mais rápida de configurar sua instalação da 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
```

Quando você digita esse comando, a AWS CLI solicita quatro informações e armazena-as em um perfil (uma coleção de configurações) denominado `default`. Esse perfil é usado sempre que você executar um comando da AWS CLI que não especifique explicitamente um perfil a ser usado.

Chave/credenciais de acesso

O `AWS Access Key ID` e a `AWS Secret Access Key` são credenciais da AWS. Elas estão associadas a um usuário ou função do IAM que determina quais permissões você tem. Para obter um tutorial sobre como criar um usuário com o serviço do IAM, consulte Criar seu primeiro usuário administrativo e grupo do IAM no Guia do usuário do IAM.

Para obter o ID de chave de acesso e a chave de acesso secreta para um usuário do IAM

As chaves de acesso consistem em um ID de chave de acesso e uma chave de acesso secreta, que são usados para assinar as solicitações programáticas que você faz à AWS. Se você não tiver chaves de acesso, poderá criá-las no Console de gerenciamento da AWS. Recomendamos que você use as chaves de acesso do IAM em vez de usar as chaves de acesso de Usuário raiz da conta da AWS. O IAM permite que você controle com segurança o acesso aos serviços e recursos da AWS em sua conta da AWS.

A única ocasião em que você poderá visualizar ou fazer download das chaves de acesso secretas é quando você criar as chaves. Não será possível recuperá-las posteriormente. No entanto, você pode criar novas chaves de acesso a qualquer momento. Você também deve ter permissões para realizar as ações do IAM necessárias. Para obter mais informações sobre permissões e políticas, consulte Permissões necessárias para acessar os recursos do IAM no Guia do usuário do IAM.

1. Abra o console do IAM.
2. No painel de navegação do console, escolha Users.
3. Selecione seu nome de usuário do IAM (não a caixa de seleção).
4. Escolha a guia Credenciais de segurança e escolha Criar chave de acesso.
5. Para ver a nova chave de acesso, escolha Mostrar. Suas credenciais terão a seguinte aparência:
   - ID de chave de acesso: AKIAIOSFODNN7EXAMPLE
   - Chave de acesso secreta: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
6. Para baixar o par de chaves, escolha Baixar arquivo .csv. Armazene as chaves em um lugar seguro.

   Mantenha a confidencialidade das chaves para proteger sua conta da AWS e nunca envie-as por e-mail. Não compartilhe as chaves fora da sua organização, mesmo se uma pesquisa parecer vir da AWS ou da Amazon.com. Alguém que legitimamente represente a Amazon jamais pedirá a você sua chave secreta.

Tópicos relacionados

- O que é o IAM? no Guia do usuário do IAM
- Credenciais de segurança da AWS no AWS General Reference

Região

O `Default region name` identifica a região dos servidores para os quais você deseja enviar suas solicitações por padrão. Normalmente, é a região mais próxima para você, mas pode ser qualquer região. Por exemplo, você pode usar digitar `us-west-2` para usar Oeste dos EUA (Oregon). Essa é a região para a qual todas as solicitações posteriores são enviadas, a menos que você especifique o contrário em um comando individual.

> Note
>
> Você deve especificar uma região da AWS ao usar a AWS CLI, explicitamente ou definindo uma região padrão. Para obter uma lista das regiões disponíveis, consulte Regiões e endpoints Os designadores de região usados pela AWS CLI têm os mesmos nomes que você vê nos URLs nos endpoints de serviço do Console de gerenciamento da AWS.

Output Format

O `Default output format` especifica como os resultados são formatados. O valor pode ser qualquer um dos valores na lista a seguir. Se você não especificar um formato de saída, `json` será usado como padrão.

- **`json`**: a saída é formatada como uma string JSON.
- **`text`**: a saída é formatada como várias linhas de valores de string separadas por tabulação, o que pode ser útil se você quiser transmitir a saída para um processador de texto, como `grep`, `sed` ou `awk`.
- **`table`**: a saída é formatada como uma tabela usando os caracteres +|- para formar as bordas da célula. Geralmente, a informação é apresentada em um formato "amigável", que é muito mais fácil de ler do que outros, mas não tão útil programaticamente.

Configuração rápida e diversos perfis

Se você usar o comando como mostrado acima, o resultado será um único perfil com o nome `default`. Você também pode criar configurações adicionais especificando o nome de um perfil usando a opção `--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
```

Em seguida, quando você executar um comando, você pode omitir a opção `--profile` e usar as configurações armazenadas no perfil `default`:

```
$ aws s3 ls
```

Ou você pode especificar um `--profile` *`profilename`* e usar as configurações armazenadas com esse nome:

```
$ aws s3 ls --profile myuser
```

Para atualizar qualquer uma das suas configurações, basta executar `aws configure` novamente (com ou sem o parâmetro `--profile` dependendo do perfil que deseja atualizar) e inserir novos valores como apropriado. As próximas seções contêm mais informações sobre os arquivos que `aws configure` cria, configurações adicionais e perfis nomeados.

Definições de Configuração e Precedência

A AWS CLI usa um conjunto de provedores de credenciais para procurar credenciais da AWS. Cada provedor de credenciais procura em um local diferente, como as variáveis de ambiente do sistema ou do usuário, arquivos de configuração locais da AWS ou explicitamente declarado na linha de comando como um parâmetro. A AWS CLI procura por credenciais e definições de configuração chamando os provedores na seguinte ordem, interrompendo quando encontra um conjunto de credenciais para usar:

1. Opções de linha de comando (p. 25) – Você pode especificar `--region`, `--output` e `--profile` como parâmetros na linha de comando.
2. Variáveis de ambiente (p. 24) – Você pode armazenar valores nas variáveis de ambiente: `AWS_ACCESS_KEY_ID`, `AWS_SECRET_ACCESS_KEY` e `AWS_SESSION_TOKEN`. Se eles estiverem presentes, eles serão usados.
3. O arquivo de credenciais da CLI (p. 22) – Esse é um dos arquivos que é atualizado quando você executa o comando `aws configure`. O arquivo está localizado em `~/.aws/credentials`, Linux, macOS, or Unix ou `C:\Users\`*`USERNAME`*`\.aws\credentials` no Windows. Esse arquivo pode conter os detalhes da credencial para o perfil `default` e quaisquer perfis nomeados.

4. O arquivo de configuração da CLI (p. 22) – Esse é outro dos arquivos que é atualizado quando você executa o comando `aws configure`. O arquivo está localizado em `~/.aws/config`, Linux, macOS, or Unix ou `C:\Users\USERNAME\.aws\config` no Windows. Esse arquivo contém as definições de configuração para o perfil padrão e quaisquer perfis nomeados.
5. Credenciais de contêiner – Você pode associar uma função do IAM a cada uma das suas definições de tarefa do Amazon Elastic Container Service. As credenciais temporárias para essa função estão disponíveis para os contêineres dessa tarefa. Para obter mais informações, consulte Funções do IAM para tarefas no Amazon Elastic Container Service Developer Guide.
6. Credenciais do perfil de instância – Você pode associar uma função do IAM a cada uma das suas instâncias do Amazon Elastic Compute Cloud (Amazon EC2). As credenciais temporárias para essa função estão disponíveis para o código em execução na instância. As credenciais são fornecidas por meio do serviço de metadados do Amazon EC2. Para obter mais informações, consulte Funções do IAM para o Amazon EC2 no Guia do usuário do Amazon EC2 para instâncias do Linux e Uso de perfis de instância no Guia do usuário do IAM.

Arquivos de configuração e credencial

A CLI armazena credenciais especificadas com `aws configure` em um arquivo local chamado `credentials` em uma pasta chamada `.aws` em seu diretório inicial. As outras opções de configuração que você especifica com `aws configure` são armazenadas em um arquivo local chamado `config`, também armazenado na pasta `.aws` no seu diretório inicial.

O local do diretório inicial varia de acordo com o sistema operacional, mas é acessado usando as variáveis de ambiente `%UserProfile%` no Windows e `$HOME` ou `~` (til) em sistemas baseados em Unix.

Por exemplo, os seguintes comandos relacionam o conteúdo da pasta `.aws`:

Linux, macOS, or Unix

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

Windows

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

A AWS CLI usa dois arquivos para manter as informações confidenciais de credenciais (em `~/.aws/credentials`) separadas das opções de configuração menos confidenciais (em `~/.aws/config`).

Você pode especificar um local não padrão para o arquivo `config` definindo a variável de ambiente `AWS_CONFIG_FILE` para outro caminho local. Para mais detalhes, consulte Variáveis de ambiente (p. 24).

Armazenar Credenciais na Config

A AWS CLI também pode ler credenciais a partir do arquivo `config`. Se desejar, é possível manter todas as suas configurações de perfil em um único arquivo. Se houver credenciais em dois locais para um perfil (por exemplo, que você usou `aws configure` para atualizar as chaves do perfil), as chaves no arquivo de credenciais terá precedência.
Se você usa um dos SDKs, além da AWS CLI, você pode observar avisos adicionais se as credenciais não são armazenados em seu próprio arquivo.

Os arquivos gerados pelo CLI para o perfil configurado na seção anterior é semelhante a:

`~/.aws/credentials`

```
[default]
```

```
aws_access_key_id=AKIAIOSFODNN7EXAMPLE
aws_secret_access_key=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
```

~/.aws/config

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

> **Note**
>
> Os exemplos anteriores mostram os arquivos com um único perfil padrão. Para obter exemplos dos arquivos com vários perfis nomeados, consulte Perfis nomeados (p. 23).

As seguintes configurações são compatíveis:

`aws_access_key_id` – A chave de acesso da AWS.

`aws_secret_access_key` – A chave secreta da AWS.

`aws_session_token` – O token da sessão da AWS. Um token de sessão só é necessário se você estiver usando credenciais de segurança temporárias.

`region` – A região padrão da AWS para enviar solicitações a partir desse perfil.

`output` (p. 20) – O formato de saída padrão para este perfil.

Perfis nomeados

A AWS CLI oferece suporte usando qualquer um dos vários perfis nomeados armazenados nos `credentials` arquivos `config`. Você pode configurar perfis adicionais usando `aws configure` com a opção `--profile` ou adicionando entradas aos arquivos `config` e `credentials`.

O exemplo a seguir mostra um arquivo `credentials` com dois perfis. O primeiro é usado quando você executa um comando da CLI sem perfil e o segundo é usado quando você executa um comando da CLI com o parâmetro `--profile user1`.

~/.aws/credentials

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

[user1]
aws_access_key_id=AKIAI44QH8DHBEXAMPLE
aws_secret_access_key=je7MtGbClwBF/2Zp9Utk/h3yCo8nvbEXAMPLEKEY
```

Cada perfil usa credenciais diferentes — talvez provenientes de usuários diferentes do IAM — e também pode usar regiões e formatos de saída diferentes:

~/.aws/config

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

[profile user1]
```

```
region=us-east-1
output=text
```

> **Important**
>
> O arquivo `credentials` usa um formato de nomeação diferente do que o arquivo `config` da CLI para perfis nomeados. Inclua o prefixo "`profile`" somente ao configurar um perfil nomeado no arquivo `config`. Não use o `profile` ao configurar o arquivo `credentials`.

Usar perfis com o AWS CLI

Para usar um perfil designado, adicione a opção `--profile` *`profile-name`* para o comando. O exemplo a seguir lista todas as suas instâncias do Amazon EC2 usando o perfil `user1` dos arquivos de exemplo anteriores:

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

Se você deseja usar um perfil designado para vários comandos, evite especificar o perfil ao configurar o comando em cada variável de ambiente `AWS_PROFILE` na linha de comando:

Linux, macOS, or Unix

```
$ export AWS_PROFILE=user2
```

Windows

```
C:\> set AWS_PROFILE=user2
```

Configurar a variável de ambiente altera o perfil padrão até o final do seu shell sessão ou até que você defina a variável a um valor diferente. Mais informações sobre variáveis na próxima seção.

Variáveis de ambiente

Variáveis de ambiente fornecem outra maneira de especificar opções de configuração e credenciais e podem ser úteis para criação de scripts ou configuração temporária de um perfil nomeado como o padrão.

> **Important**
>
> A configuração de uma das seguintes variáveis de ambiente substitui todas as outras formas de especificar essa opção exceto especificar a opção como um parâmetro de linha de comando.

A AWS CLI é compatível com as seguintes variáveis de ambiente:

- `AWS_ACCESS_KEY_ID` – Especifica uma chave de acesso da AWS associada a um usuário ou função do IAM.
- `AWS_SECRET_ACCESS_KEY` – Especifica a chave secreta associada à chave de acesso. Essa é essencialmente a "senha" para a chave de acesso.
- `AWS_SESSION_TOKEN` – Especifica o valor do token de sessão necessário se você estiver usando credenciais de segurança temporárias. Para obter mais informações, consulte Seção de saída do comando assume-role no AWS CLI Command Reference.
- `AWS_DEFAULT_REGION` – Especifica a região da AWS (p. 20) para a qual enviar a solicitação.
- `AWS_DEFAULT_OUTPUT` – Especifica o formato de saída a ser usado.

- `AWS_PROFILE` – Especifica o nome do perfil da CLI (p. 23) com as credenciais e as opções a serem usadas. Esse pode ser o nome de um perfil armazenado em um arquivo `credentials` ou `config` ou o valor `default` para usar o perfil padrão.
- `AWS_CA_BUNDLE` – Especifica o caminho para um pacote de certificado a ser usado para a validação de certificado HTTPS.
- `AWS_SHARED_CREDENTIALS_FILE` – Especifica a localização do arquivo que a AWS CLI usa para armazenar as chaves de acesso (o padrão é `~/.aws/credentials`).
- `AWS_CONFIG_FILE` – Especifica a localização do arquivo que a AWS CLI usa para armazenar os perfis de configuração (o padrão é `~/.aws/config`).

O exemplo a seguir mostra como você pode configurar variáveis de ambiente para o usuário padrão. Esses valores substituirão quaisquer valores encontrados em um perfil nomeado ou metadados de instância. Depois de definidos, esses valores podem ser substituídos especificando-se um parâmetro na linha de comando da CLI ou alterando ou removendo a variável de ambiente.

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

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

Opções de linha de comando

Você pode usar as opções de linha de comando a seguir para substituir as definições de configuração padrão para um único comando. Você não pode usar opções de linha de comando para especificar diretamente credenciais, embora seja possível especificar qual perfil usar.

--profile

Especifica o perfil nomeado (p. 23) para usar esse comando. Para configurar perfis nomeados adicionais, você pode usar o comando `aws configure` com a opção `--profile`.

```
$ aws configure --profile <profilename>
```

--região

Especifica para qual região da AWS enviar a solicitação da AWS desse comando. Para obter uma lista de todas as regiões que você pode especificar, consulte AWS Regiões e endpoints no Referência geral do Amazon Web Services.

--resultado

Especifica o formato de saída a ser usado para este comando. Você pode especificar qualquer um dos seguintes valores:

- **`json`**: a saída é formatada como uma string JSON.
- **`text`**: a saída é formatada como várias linhas de valores de string separadas por tabulação, o que pode ser útil se você quiser transmitir a saída para um processador de texto, como `grep`, `sed` ou `awk`.

- **`table`**: a saída é formatada como uma tabela usando os caracteres +|- para formar as bordas da célula. Geralmente, a informação é apresentada em um formato "amigável", que é muito mais fácil de ler do que outros, mas não tão útil programaticamente.

--endpoint-url

O URL para o qual a solicitação é enviada. Para a maioria dos comandos, a AWS CLI determina automaticamente o URL com base no serviço selecionado e na região da AWS especificada. No entanto, alguns comandos exigem que você especifique um URL específico para a conta. Você também pode configurar alguns serviços da AWS para hospedar um endpoint diretamente dentro de sua VPC privada, que talvez precise ser especificada.

Para obter uma lista dos endpoints de serviços padrão disponíveis em cada região, consulte Regiões e endpoints da AWS no Referência geral do Amazon Web Services.

Quando você fornece uma ou mais dessas opções como parâmetros de linha de comando, ela substitui a configuração padrão ou qualquer configuração de perfil correspondente para esse único comando. Cada opção obtém um argumento de string com um espaço ou sinal de igual (=) separando o argumento do nome da opção. Se o valor do argumento contiver um espaço, você deverá colocar o argumento entre aspas.

Os usos comuns das opções de linha de comando incluem a verificação de seus recursos em várias regiões da AWS e a alteração do formato de saída para legibilidade ou a facilidade de uso ao criar scripts. Por exemplo, se você não tem certeza em relação à região em que a sua instância está sendo executada, você pode executar o comando describe-instances em cada região até encontrá-la, da maneira a seguir.

```
$ 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                                       |||
...
```

Os tipos de argumento (string, booleano, etc.) para cada opção de linha de comando são descritos em detalhes em Especificar valores de parâmetro (p. 44).

Metadados da instância

Quando você executar a AWS CLI de dentro de uma instância do Amazon EC2, você pode simplificar o fornecimento de credenciais aos seus comandos. Cada instância do Amazon EC2 contém metadados que a AWS CLI pode consultar diretamente por credenciais temporárias. Para fornecer isso, crie uma função do IAM que tenha acesso aos recursos necessários e anexe essa função à instância do Amazon EC2 ao iniciá-la.

Execute a instância e verificar se o AWS CLI já está instalado (já vem pré-instalado no Amazon Linux). Instale a AWS CLI, se necessário. Você ainda deve configurar uma região padrão para não precisar especificar em cada comando.

Você pode definir a região e o formato de saída padrão executando `aws configure` sem especificar credenciais pressionando inserir duas vezes para ignorar as duas primeiras solicitações:

```
$ 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
```

Quando uma função do IAM é anexada à instância, a AWS CLI recupera de forma automática e segura as credenciais dos metadados da instância. Para obter mais informações, consulte Como conceder acesso aos recursos da AWS para aplicativos que são executados em instâncias do Amazon Amazon EC2 em Guia do usuário do IAM.

Usar um proxy HTTP

Para acessar a AWS por meio de servidores de proxy, é possível configurar as variáveis do ambiente `HTTP_PROXY` e `HTTPS_PROXY` com os endereços IP e números de porta usados pelos 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

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

Nesses exemplos, `http://a.b.c.d:n` e `http://w.x.y.z:m` são os endereços IP e os números de porta para o HTTP e HTTPS proxies.

Autenticar para um proxy

O AWS CLI é compatível com a autenticação básica HTTP. Especifique o nome de usuário e uma senha no URL de proxy da seguinte maneira:

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

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

> **Note**
>
> O AWS CLI não é compatível com proxies NTLM. Se você usa um proxy NTLM ou Kerberos, talvez seja possível se conectar por meio de um proxy de autenticação, como Cntlm.

Usar um proxy em instâncias do EC2

Se você configurar um proxy em uma instância do Amazon EC2 iniciada com uma função anexada do IAM, isente o endereço usado para acessar os metadados da instância. Para fazer isso, defina a variável de ambiente `NO_PROXY` como o endereço IP do serviço de metadados da instância 169.254.169.254.

Linux, macOS, or Unix

```
$ export NO_PROXY=169.254.169.254
```

Windows

```
C:\> set NO_PROXY=169.254.169.254
```

Assumir uma função do IAM

Uma função do IAM é uma ferramenta de autorização que permite que um usuário do IAM obtenha permissões adicionais (ou diferentes) ou obtenha permissões para executar ações em uma conta diferente da AWS.

Você pode configurar a AWS Command Line Interface para usar uma função do IAM definindo um perfil para a função no arquivo `~/.aws/config`. O exemplo a seguir mostra um perfil de função denominado `marketingadmin` que é assumido quando você executa comandos que especificam o perfil `marketingadmin`.

```
[profile marketingadmin]
role_arn = arn:aws:iam::123456789012:role/marketingadmin
source_profile = user1
```

A função deve estar vinculada a um perfil nomeado separado que contenha credenciais de usuário do IAM com permissão para assumir a função. No exemplo anterior, o perfil `marketingadmin` é vinculado usando o campo `source-profile` para o perfil `user1`. Quando você especifica que um comando da AWS CLI deve usar o perfil `marketingadmin`, a CLI automaticamente procura as credenciais para o perfil `user1` vinculado e as utiliza para solicitar credenciais temporárias para a função especificada do IAM. Essas credenciais temporárias são usadas para executar o comando da CLI. A função especificada deve ter políticas de permissão do IAM anexadas que permitam a execução do comando da CLI.

Seções

Configurar e usar uma função

Ao executar comandos usando um perfil que especifica uma função do IAM, a AWS CLI usa as credenciais do perfil de origem para chamar AWS Security Token Service (AWS STS) e solicitar credenciais temporárias para a função especificada. O usuário no perfil de origem deve ter permissão para chamar `sts:assume-role` para a função no perfil especificado. A função deve ter uma relação de confiança que permita que o usuário no perfil de origem assuma a função. Geralmente, o processo de recuperar e depois usar credenciais temporárias para uma função é chamado de assumir a função.

Você pode criar uma nova função no IAM com as permissões que você deseja que os usuários assumam. Para isso, siga o procedimento em Como criar uma função para conceder permissões a um usuário do IAM no Guia do usuário do AWS Identity and Access Management. Se a função e o usuário do IAM do perfil de origem estão na mesma conta, é possível inserir seu próprio ID de conta ao configurar a relação de confiança da função.

Depois de criar a função, modifique a relação de confiança para permitir que o usuário do IAM (ou os usuários da conta da AWS) a assumam. O exemplo a seguir mostra uma relação de confiança que permite que uma função seja assumida por qualquer usuário do IAM na conta 123456789012, se o administrador dessa conta explicitamente conceder a permissão `sts:assumerole` para o usuário:

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

A política de confiança não concede permissões. O administrador da conta deve delegar a permissão para assumir a função para usuários individuais, anexando uma política com as permissões apropriadas. O exemplo a seguir permite que o usuário do IAM anexado assuma apenas a função `marketingadmin`:

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

O usuário do IAM não precisa ter permissões adicionais para executar os comandos da CLI usando o perfil da função. Em vez disso, as permissões necessárias para executar o comando vêm daquelas anexadas à função. No entanto, se você quiser que seus usuários possam acessar os recursos da AWS sem usar uma função, deverá anexar políticas adicionais gerenciadas ou em linha ao usuário do IAM que conceder permissões para esses recursos.

Agora que você tem o perfil de função, permissões de função, relação de confiança de função e permissões de usuário configuradas corretamente, é possível usar a função na linha de comando invocando a opção `--profile`. Por exemplo, o comando a seguir chama o comando do Amazon S3 `ls` usando as permissões anexadas à função `marketingadmin` conforme definido no exemplo no início desse tópico:

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

Se você quiser usar a função para várias chamadas, poderá definir a variável de ambiente `AWS_PROFILE` para a sessão atual na linha de comando. Enquanto essa variável de ambiente é definida, você não precisa especificar a opção `--profile` em cada comando.

Linux, macOS, or Unix

```
$ export AWS_PROFILE=marketingadmin
```

Windows

```
C:\> set AWS_PROFILE=marketingadmin
```

Para obter mais informações sobre como configurar usuários e funções do IAM, consulte Usuários e grupos e Funções no Guia do usuário do IAM.

Uso de autenticação multifator

Para segurança adicional, você pode exigir que os usuários forneçam uma chave única gerada a partir de um dispositivo de autenticação multifator, um dispositivo U2F ou um aplicativo móvel quando eles tentarem fazer uma chamada usando o perfil de função.

Primeiro, modifique a relação de confiança na função do IAM para exigir autenticação multifator. Por exemplo, veja a linha `Condition` no exemplo a seguir:

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

Em seguida, adicione uma linha à função que especifica o perfil do usuário ARN do dispositivo de MFA:

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

A configuração `mfa_serial` pode precisar de um ARN, conforme mostrado, ou o número de série de um token MFA de hardware.

Funções entre contas

Você pode permitir que os usuários do IAM assumam funções que pertencem a diferentes contas ao configurar a função como uma função de conta cruzada. Durante a criação da função, defina o tipo de função como Outra conta da AWS, conforme descrito em Criação de uma função para delegar permissões a um usuário do IAM e, opcionalmente, selecione Solicitar MFA. A opção Solicitar MFA configura a condição apropriada na relação de confiança, conforme descrito em Uso de autenticação multifator (p. 30).

Se usar um ID externo para fornecer mais controle sobre quem pode assumir uma função em todas as contas, você também deverá adicionar o parâmetro `external_id` ao perfil da função. Normalmente, você usa isso somente quando a outra conta é controlada por alguém de fora da sua empresa ou organização.

```
[profile crossaccountrole]
role_arn = arn:aws:iam::234567890123:role/SomeRole
```

```
source_profile = default
mfa_serial = arn:aws:iam::123456789012:mfa/saanvi
external_id = 123456
```

Limpeza de credenciais em cache

Quando você assume uma função, a AWS CLI armazena em cache as credenciais temporárias localmente até que elas expirem. Se as credenciais temporárias da sua função forem revogadas, você poderá excluir o cache para forçar a AWS CLI a recuperar novas credenciais.

Linux, macOS, or Unix

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

Windows

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

Conclusão de comando

Em sistemas semelhantes ao Unix, a AWS CLI inclui um recurso de conclusão de comando que permite que você use a tecla TAB para concluir um comando digitado parcialmente. Na maioria dos sistemas, esse recurso não é instalado automaticamente, portanto, você precisa configurá-lo manualmente.

Para configurar a conclusão do comando, você deve ter duas informações: o nome do shell que você está usando e a localização do script `aws_completer`.

Amazon Linux

A conclusão do comando é automaticamente configurada e ativada por padrão nas instâncias do Amazon EC2 que executam o Amazon Linux.

Seções

Identifique o Shell

Se você não tiver certeza de qual shell está usando, poderá identificá-lo com um dos seguintes comandos:

echo $ SHELL – Mostra o diretório de instalação do shell. Isso geralmente corresponde ao shell em uso, a menos que você tenha iniciado um shell diferente após fazer o login.

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

ps – Mostra os processos em execução para o usuário atual. O shell será um deles.

```
$ ps
  PID TTY          TIME CMD
```

```
 2148 pts/1    00:00:00 bash
 8756 pts/1    00:00:00 ps
```

Localize o Completer da AWS

A localização pode variar, dependendo do método de instalação usado.

Package Manager – programas como `pip`, `yum`, `brew` e `apt-get` normalmente instalam o completer da AWS (ou um symlink) para um local de caminho padrão. Neste caso, o comando `which` pode localizar o completer para você.

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

Bundled Installer – se você usou o instalador de acordo com as instruções fornecidas na seção anterior, o completer da AWS estará localizado na subpasta `bin` do diretório de instalação.

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

Se tudo falhar, use `find` para pesquisar o completer da AWS em todo o seu sistema de arquivos.

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

Habilitar conclusão de comando

Execute um comando para habilitar a conclusão de comando. O comando que usado para habilitar a conclusão depende do shell usado. Você pode adicionar o comando ao arquivo RC do shell para executá-lo cada vez que você abrir um novo shell.

- bash – Usa o comando interno `complete`.

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

  Adicione o comando ao `~/.bashrc` para executá-lo cada vez que você abrir um novo shell. Seu `~/.bash_profile` deve se basear em `~/.bashrc` para garantir que o comando seja executado também em shells de login.
- tcsh – A conclusão do tcsh leva um tipo de palavra e padrão para definir o comportamento de conclusão.

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

  Adicione o comando ao `~/.tschrc` para executá-lo cada vez que você abrir um novo shell.
- zsh – `bin/aws_zsh_completer.sh` de origem.

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

A AWS CLI usa a conclusão automática de compatibilidade do bash (`bashcompinit`) para suporte ao zsh. Para mais detalhes, consulte o topo do `aws_zsh_completer.sh`.

Adicione o comando ao `~/.zshrc` para executá-lo cada vez que você abrir um novo shell.

Conclusão do comando de teste

Após ativar a conclusão de comando, digite um comando parcial e pressione "Tab" para ver os comandos disponíveis.

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

Tutorial: como usar a AWS Command Line Interface para implantar um ambiente de desenvolvimento do Amazon EC2

Este tutorial descreve como usar a AWS CLI para configurar um ambiente de desenvolvimento no Amazon EC2. Ele inclui uma versão simplificada das instruções de instalação e configuração. Ele pode ser executado do início ao fim no Windows, Linux, macOS, or Unix.

Etapas

Instalar a AWS CLI

Instale a AWS CLI com um instalador (Windows) ou use `pip`, um gerenciador de pacote para Python.

Windows

1. Faça o download do instalador MSI.
   - Faça download do instalador MSI da AWS CLI para Windows (64 bits)
   - Faça download do instalador MSI da AWS CLI para Windows (32 bits)
2. Execute o download do instalador MSI.
3. Siga as instruções da tela.

Linux, macOS, or Unix

Para estas etapas, é necessário ter a versão 2.6.5+ do Python 2 ou a versão 3.3+ do Python 3 instalada. Se houver qualquer problema nas etapas seguintes, consulte as instruções completas de instalação no Guia do usuário do AWS Command Line Interface.

1. Se você instalou o pip, pule para a etapa 2. Se você ainda não tiver o pip instalado, faça o download e execute o script de instalação do site pip:

```
$ curl "https://bootstrap.pypa.io/get-pip.py" -o "get-pip.py"
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
```

```
100 1622k  100 1622k    0     0  14.9M      0 --:--:-- --:--:-- --:--:-- 14.9M
$ python3 get-pip.py --user
Collecting pip
  Using cached https://files.pythonhosted.org/packages/c2/
d7/90f34cb0d83a6c5631cf71dfe64cc1054598c843a92b400e55675cc2ac37/pip-18.1-py2.py3-none-
any.whl
Collecting setuptools
  Using cached https://files.pythonhosted.org/packages/e7/16/
da8cb8046149d50940c6110310983abb359bbb8cbc3539e6bef95c29428a/setuptools-40.6.2-py2.py3-
none-any.whl
Collecting wheel
  Using cached https://files.pythonhosted.org/packages/
ff/47/1dfa4795e24fd6f93d5d58602dd716c3f101cfd5a77cd9acbe519b44a0a9/wheel-0.32.3-
py2.py3-none-any.whl
Installing collected packages: pip, setuptools, wheel
  The script wheel is installed in '/home/myusername/.local/bin' which is not on PATH.
  Consider adding this directory to PATH or, if you prefer to suppress this warning,
 use --no-warn-script-location.
```

A opção `--user` especifica que você deseja instalar o pip no diretório inicial local do usuário. Isso é necessário se você não tiver permissões root/admin. Se você não especificar o `--user`, o pip tentará instalar em uma pasta do sistema para todos os usuários.

2. Certifique-se de que a pasta de instalação do pip está em sua variável de ambiente PATH. O instalador normalmente informa se não estiver, como mostrado no exemplo anterior. Para resolver isso, adicione uma instrução como o seguinte no final do script RC do shell, que é apropriado se `pip` foi instalado na pasta `.local/bin` em seu diretório inicial.

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

3. Agora você pode usar `pip` para instalar a AWS CLI.

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

É recomendável que você use novamente a chave `--user` para instalar a AWS CLI em sua pasta inicial local que não exige permissões de root/admin.

Configurar o AWS CLI

Primeiro, configure a AWS CLI com suas credenciais e configurações padrão.

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

A AWS CLI solicitará as seguintes informações:

- ID da chave de acesso e chave de acesso secreta do AWS – Essas são as credenciais da conta ou do usuário. Se você não tiver chaves, consulte Chaves de acesso (ID da chave de acesso e Chave de acesso secreta) no Referência geral do Amazon Web Services.
- Nome da região padrão – Isso especifica o nome da região para a qual você deseja que a CLI envie suas solicitações por padrão.
- Formato de saída padrão – Isso especifica o formato de saída que você deseja que a CLI use por padrão. O valor pode ser `json`, `text` ou `table`. Se você não especificar um formato de saída, será usado `json`.

Agora experimente um simples comando para verificar se suas credenciais estão configuradas corretamente e se você pode se conectar ao AWS.

```
$ aws ec2 describe-regions --output table
-------------------------------------------------------------
|                     DescribeRegions                       |
+-----------------------------------------------------------+
||                         Regions                         ||
|+-----------------------------------+-------------------+|
||             Endpoint              |   RegionName      ||
|+-----------------------------------+-------------------+|
||  ec2.ap-south-1.amazonaws.com     |  ap-south-1       ||
||  ec2.eu-west-3.amazonaws.com      |  eu-west-3        ||
||  ec2.eu-west-2.amazonaws.com      |  eu-west-2        ||
||  ec2.eu-west-1.amazonaws.com      |  eu-west-1        ||
||  ec2.ap-northeast-3.amazonaws.com |  ap-northeast-3   ||
||  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        ||
|+-----------------------------------+-------------------+|
```

Crie um security group, e um par de chaves para a instância do EC2

A próxima etapa é configurar os pré-requisitos para a execução de uma instância do Amazon EC2 que pode ser acessada com um emulador de terminal conectado usando o protocolo Secure Shell (SSH). Para obter mais informações sobre o Amazon EC2 e seus recursos, consulte Guia do usuário do Amazon EC2 para instâncias do Linux.

O Amazon EC2 requer os seguintes pré-requisitos para se comunicar com uma instância do EC2:

- Grupo de segurança – um grupo de segurança determina qual tráfego de rede pode entrar e sair da sua instância. Você pode pensar neles como um firewall virtual. Um grupo de segurança contém as regras que controlam o tráfego de rede de entrada e de saída.
- Par de chaves – A criptografia de chave pública usa uma chave pública para criptografar uma parte dos dados, como uma senha e, em seguida, o destinatário usa a chave privada para descriptografar os dados. O Amazon EC2 usa o par de chaves especificado para criptografar as credenciais usadas para acessar a instância.

Para criar um grupo de segurança e um par de chaves

1. Primeiro, crie um novo grupo de segurança para a VPC na qual você vai iniciar a instância. Se você estiver usando a VPC padrão da região, pode omitir o parâmetro `--vpc-id`; caso contrário, especifique o ID da VPC na qual você vai iniciar a instância. A saída mostra o identificador do seu novo grupo de segurança.

   ```
   $ aws ec2 create-security-group --group-name devenv-sg --vpc-id vpc-xxxxxxxx --
   description "Security group for development environment"
   ```

```
{
    "GroupId": "sg-b018ced5"
}
```

2. Em seguida, crie uma regra para o grupo de segurança que permite o tráfego de rede de entrada na porta 22 no intervalo de endereços de rede CIDR que você usará para se conectar à instância.

   Important

   Este exemplo mostra o intervalo `0.0.0.0/0` CIDR que permite o tráfego de entrada de qualquer lugar na Internet. Recomendamos que você substitua isso pelo intervalo CIDR público de rede (conforme mostrado pelo AWS) do qual você vai se conectar à sua instância.

```
$ aws ec2 authorize-security-group-ingress --group-name devenv-sg --protocol tcp --port
 22 --cidr 0.0.0.0/0
```

   Anote o ID do grupo de segurança. Você precisará dele mais tarde ao ativar sua instância.

3. Em seguida, crie o par de chaves criptográficas SSH que você usará para se conectar à instância. Este comando de exemplo mostra como salvar o conteúdo da chave em um arquivo chamado `devenv-key.pem`.

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

   O parâmetro `--query "KeyMaterial"` extrai apenas a parte da saída que você precisa no arquivo .pem.

   Aspas duplas para o parâmetro **`--query`**

   Todos os exemplos deste tópico que incluem o parâmetro `--query` usam aspas duplas. Embora o Linux permita que você use aspas simples para o parâmetro `--query`, o prompt de comando do Windows exige que você use aspas duplas em vez de aspas simples.

4. No Linux, altere o acesso da novo arquivo de chaves para que somente você tenha acesso a ele.

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

Executar e conectar-se à instância

Agora você está tudo pronto para iniciar uma instância e se conectar a ela.

Iniciar e conectar-se à instância

1. Execute o comando a seguir, usando o ID do security group que você criou na etapa anterior. O parâmetro `--image-id` especifica a imagem de máquina da Amazon (AMI) que o Amazon EC2 usa para inicializar a instância. É possível localizar um ID de imagem para a sua região e sistema operacional usando o console do Amazon EC2. Se você estiver usando a sub-rede padrão para uma VPC padrão, pode omitir o parâmetro `--subnet-id`; caso contrário, especifique o ID da sub-rede na qual você vai iniciar a instância.

   Note

   Este exemplo mostra o comando dividido em várias linhas com o caractere de continuação de linha '\' do Linux. É possível enviar tudo como uma única linha. Para a linha de comando do Windows, substitua '\' por '^'.

```
$ aws ec2 run-instances --image-id ami-xxxxxxxx \
```

```
                                --subnet-id subnet-xxxxxxxx \
                                --security-group-ids sg-b018ced5 \
                                --count 1 \
                                --instance-type t2.micro \
                                --key-name devenv-key \
                                --query "Instances[0].InstanceId"
"i-0787e4282810ef9cf"
```

2. A inicialização leva alguns instantes. Depois que a instância estiver em execução, você poderá recuperar o endereço IP público dela que você precisará para se conectar a ela com o comando a seguir:

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

3. Para se conectar à instância, use o endereço IP público e o arquivo .pem de chave privada com o programa de terminal de sua preferência. Na Linux, macOS, or Unix, é possível fazer isso a partir da linha usando o seguinte comando:

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

   Caso ocorra um erro como Permissão negada (publickey) durante a tentativa de se conectar à sua instância, verifique se os itens a seguir estão corretos:

   - Chave – A chave especificada deve estar no caminho indicado e deve ser privada, não pública. As permissões na chave devem ser restritas ao proprietário.
   - Usuário – O nome de usuário deve ser igual ao nome de usuário padrão associado à AMI usada para iniciar a instância. Para uma AMI do Ubuntu, use `ubuntu`. Para uma AMI do Amazon Linux, use `ec2-user`.
   - Instância – O endereço IP público ou nome DNS da instância. Verificar se o endereço é público e se a porta 22 está aberta para a máquina local na instância do grupo de segurança.

   Também é possível usar a opção `-v` para visualizar informações adicionais relacionadas ao erro.

   SSH no Windows

   No Windows, use o aplicativo de terminal PuTTY disponível aqui. Tenha acesso ao `putty.exe` e ao `puttygen.exe` a partir da página de downloads.
   Use `puttygen.exe` para converter sua chave privada para um arquivo `.ppk` exigido por PuTTY. Execute `putty.exe`, insira o endereço IP público da instância no campo Nome do host e defina o tipo de conexão para SSH.
   No painel Categoria, navegue até Conexão > SSH > Auth, clique em Procurar para selecionar o arquivo `.ppk` e, em seguida, clique em Abrir para se conectar.

4. Será necessário aceitar a chave pública do servidor. Digite `yes` e pressione Enter para concluir a conexão.

Você configurou um grupo de segurança, criou um par de chaves, iniciou uma instância do EC2 e conectou-se a ela sem sair da linha de comando.

Como usar AWS Command Line Interface

Esta seção apresenta muitos dos recursos comuns e das opções disponíveis na AWS Command Line Interface.

> **Note**
>
> Por padrão, a AWS CLI envia solicitações para serviços do AWS usando HTTPS na porta TCP 443. Para usar a AWS CLI com êxito, você deve ser capaz de fazer as conexões de saída na porta TCP 443.

Tópicos

Receber ajuda com a AWS Command Line Interface

Você pode obter ajuda com qualquer comando ao usar a AWS CLI. Para fazer isso, basta digitar `help` no final de um nome do comando. Por exemplo, o comando a seguir exibe a ajuda para as opções gerais da AWS CLI e os comandos de nível superior disponíveis.

```
$ aws help
```

O comando a seguir exibe os comandos específicos do Amazon EC2 disponíveis.

```
$ aws ec2 help
```

O exemplo a seguir exibe ajuda detalhada para a operação `DescribeInstances` do Amazon EC2. A ajuda inclui descrições de seus parâmetros de entrada, filtros disponíveis, e o que é incluído como saída. Ela também inclui exemplos que mostram como digitar as variações comuns do comando.

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

A ajuda para cada comando é dividida em seis seções:

Nome

O nome do comando.

```
NAME
       describe-instances -
```

Descrição

Uma descrição da operação da API que o comando invoca.

```
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.
...
```

Resumo

A sintaxe básica para usar o comando e suas opções. Se uma opção estiver entre colchetes, significa que ela é opcional, tem um valor padrão ou tem uma alternativa opção que podem ser usadas em vez.

```
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]
```

Por exemplo, `describe-instances` tem um comportamento padrão que descreve todas as instâncias na conta e região atuais. Opcionalmente, você pode especificar uma lista de `instance-ids` para descrever uma ou mais instâncias.O `dry-run` é um sinalizador booleano opcional que não têm um valor. Para usar um sinalizador booleano, especifique um valor apresentado, nesse caso `--dry-run` ou `--no-dry-run`. Da mesma forma, `--generate-cli-skeleton` não tem um valor. Se houver condições no uso de uma opção, elas serão descritas na seção `OPTIONS` ou mostradas nos exemplos.

Opções

A descrição de cada uma das opções mostradas na sinopse.

```
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.
...
```

Exemplos

Os exemplos que mostram o uso do comando e suas opções. Se nenhum exemplo estiver disponível para um comando ou caso de uso que você precisa, solicite um usando o link de comentários nesta página ou na referência de comandos da AWS CLI na página de ajuda para o 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"
...
```

Resultado

As descrições de cada um dos campos e os tipos de dados incluídos na resposta da AWS.

Para `describe-instances`, a saída é uma lista de objetos de reserva, cada um com vários campos e objetos que contêm informações sobre as instâncias associadas a ele. Essas informações são provenientes da documentação da API para o tipo de dados de reserva usada pelo 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.
...
```

Quando a saída é renderizada em JSON pela AWS CLI, ela se torna uma variedade de objetos de reserva, semelhante ao seguinte exemplo:

```
{
    "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 contém campos que descrevem a reserva e uma série de objetos de instância, cada um com seus próprios campos (por exemplo, `PublicDnsName`) e objetos (por exemplo, `State`) que os descrevem.

Usuários do Windows

Você pode usar pipe (|) na saída do comando de ajuda do comando `more` para visualizar o arquivo de ajuda uma página por vez. Pressione a barra de espaço ou PgDn para visualizar mais detalhes do documento ou **`q`** se quiser sair.

```
C:\> aws ec2 describe-instances help | more
```

Documentação do AWS CLI

O AWS CLI Command Reference também contém o conteúdo da ajuda para todos os comandos da AWS CLI. As descrições são apresentadas para facilitar a navegação e a visualização em dispositivos móveis, tablets e telas de desktop.

Note

Os arquivos de ajuda contêm links que não podem ser visualizados a partir da linha de comando nem é possível navegar até eles a partir dela. Você pode visualizar e interagir com esses links usando a referência da AWS CLI online.

Documentação de API

Todos os comandos na AWS CLI correspondem a solicitações feitas para uma API pública de serviço do AWS. Cada serviço com uma API pública tem um Guia de Referência da API que pode ser encontrado na página inicial do serviço no site de documentação do AWS. O conteúdo de um Guia de Referência da API varia dependendo de como a API é construída e de qual protocolo é usado. Normalmente, um Guia de Referência da API contém informações detalhadas sobre as ações compatíveis com a API, os dados enviados para e do serviço e quaisquer condições de erro que o serviço pode relatar.

Seções da documentação de API

- Ações – Informações detalhadas sobre cada ação e seus parâmetros (incluindo restrições de tamanho ou conteúdo e valores padrão). Ele lista os erros que podem ocorrer nesta ação. Cada ação corresponde a um subcomando na AWS CLI.
- Tipos de dados – Informações detalhadas sobre estruturas que um comando pode exigir como um parâmetro ou retornar em resposta a uma solicitação.
- Parâmetros comuns – Informações detalhadas sobre os parâmetros que são compartilhados por toda a ação para o serviço.
- Erros comuns – Informações detalhadas sobre erros que podem ser retornados por todas as ações de um serviço.

O nome e a disponibilidade de cada seção podem variar de acordo com o serviço.

CLIs específicas para o serviço

Alguns serviços têm uma CLI separado desde antes da criação de uma única AWS CLI que funciona com todos os serviços. Essas CLIs específicas do serviço têm documentação separada vinculada a partir da página de documentação do serviço. A documentação para CLIs específicas do serviço não se aplica à AWS CLI.

Estrutura de comando na AWS Command Line Interface

A AWS CLI usa uma estrutura de várias partes na linha de comando que deve ser especificada nesta ordem:

1. A chamada básica para o programa `aws`
2. O comando de nível superior que normalmente corresponde a um serviço do AWS compatível com a AWS CLI.
3. O subcomando que especifica a operação a ser realizada.
4. As opções gerais da CLI ou os parâmetros necessários para a operação. Esses podem ser especificados em qualquer ordem, desde que siga as três primeiras partes. Se um parâmetro exclusivo é especificado várias vezes, apenas o último valor se aplica.

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

Parâmetros pode levar vários tipos de valores de entrada, como números, sequências de caracteres, listas, mapas e estruturas de JSON.

Especificar valores de parâmetro para a AWS Command Line Interface

Muitos parâmetros são sequências ou valores numéricos simples, como o nome do par de chaves `my-key-pair` no exemplo a seguir:

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

Sequências sem nenhum caractere de espaço podem ser citados ou não citadas. No entanto, sequências que incluem um ou mais caracteres de espaço devem ser citadas. Use aspas simples (') no Linux, macOS, or Unix e no Windows PowerShell e aspas duplas (") no prompt de comando do Windows, como mostrado nos exemplos a seguir.

Windows PowerShell, Linux, macOS, or Unix

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

Processador de comando do Windows

```
C:\> aws ec2 create-key-pair --key-name "my key pair"
```

Opcionalmente, você pode separar o nome de parâmetro do valor com um sinal de igual (=), em vez de um espaço. Isso geralmente é necessário apenas se o valor do parâmetro começa com um hífen:

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

Tópicos

Tipos comuns de parâmetros

Esta seção descreve alguns dos tipos de parâmetros comuns e o formato típico necessário. Caso haja problemas com a formatação de um parâmetro para um comando específico, consulte a ajuda digitando **`help`** após o nome do comando, por exemplo:

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

A ajuda para cada subcomando descreve sua função, opções e exemplos de saída. A seção de opções inclui o nome e a descrição de cada opção com o tipo de parâmetro da opção entre parênteses.

String – Parâmetros de string podem conter caracteres alfanuméricos, símbolos e espaço em branco a partir do conjunto de caracteres ASCII. Sequências com espaço em branco devem ser incluídas entre aspas. Não é recomendado o uso de símbolos e espaço em branco diferente do caractere de espaço padrão, pois pode causar problemas ao usar a AWS CLI.

Alguns parâmetros de sequência de caracteres pode aceitar dados binários a partir de um arquivo. Para ver um exemplo, consulte Arquivos binários (p. 48).

Marca de tempo – As marcas de tempo são formatadas de acordo com a norma ISO 8601. Esses são, às vezes, chamado de parâmetros de digitação "DateTime" ou "Data".

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

Os formatos aceitos incluem:

- *AAAA*-*MM*-*DD*T*hh*:*mm*:*ss.sss*TZD *(UTC)*, por exemplo, 2014-10-01T20:30:00.000Z
- *AAAA*-*MM*-*DD*T*hh*:*mm*:*ss.sss*TZD *(com deslocamento)*, por exemplo, 2014-10-01T12:30:00.000-08:00
- *AAAA*-*MM*-*DD*, por exemplo, 2014-10-01
- Tempo do Unix em segundos, por exemplo, 1412195400. Isso às vezes é conhecido como Unix Epoch Time e representa o número de segundos desde a meia-noite, 1º de janeiro de 1970 UTC.

Lista – Uma ou mais strings separadas por espaços. Se qualquer um dos itens de string contiver espaços, você deverá colocar aspas em torno desse item.

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

Booleano – Sinalizador binário que liga/desliga uma opção. Por exemplo, `ec2 describe-spot-price-history` tem um booleano parâmetro dry-run que, quando especificado, valida o comando contra o serviço sem executar uma consulta.

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

A saída indica se o comando foi bem formado ou não. Este comando também inclui uma versão no-dry-run do parâmetro que pode ser usado para indicar explicitamente que o comando deve ser executado normalmente. A inclusão não é necessária, já que este é o comportamento padrão.

Inteiro – Um número inteiro não assinado.

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

Blob – Objeto binário. Os parâmetros de Blob levam um caminho para um arquivo local que contém dados binários. O caminho não deve conter qualquer identificador de protocolo, como `http://` ou `file://`. O caminho especificado é interpretado como sendo relativo ao diretório de trabalho atual.

Por exemplo, o parâmetro `--body` para `aws s3api put-object` é um blob:

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

Mapear – Um conjunto de pares de chave/valor especificado no JSON ou usando a sintaxe abreviada (p. 59) da CLI. O exemplo de JSON a seguir lê um item de uma tabela do DynamoDB chamada my-table com um parâmetro de mapa, `--key`. O parâmetro especifica a chave primária chamada id com um valor numérico de 1 em uma estrutura JSON aninhada.

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

```
    }
}
```

A próxima seção aborda os argumentos JSON mais detalhadamente.

Usar JSON para parâmetros

JSON é útil para especificar parâmetros da linha de comando complexos. Por exemplo, o seguinte comando listará todas as instâncias do EC2 que tenham um tipo de instância `m1.small` ou `m1.medium` que também estão na zona de disponibilidade `us-west-2c`.

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

O exemplo a seguir especifica a lista de filtros equivalente como uma matriz JSON. Os colchetes são usados para criar uma variedade de objetos JSON separados por vírgulas. Cada objeto é uma lista separada por vírgula de pares chave-valor ("Nome" e "Valores" são chaves nessa instância).

Observe que o valor à direita da chave "Valores" também é uma matriz. É necessário, mesmo se a série tiver apenas uma sequência de valor.

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

Os colchetes mais externos, por outro lado, são necessários apenas se mais de um filtro é especificado. A versão de um único filtro do comando acima, formatada em JSON, é semelhante a:

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

Algumas operações exigem dados para ser formatadas como JSON. Por exemplo, para passar parâmetros para o parâmetro `--block-device-mappings` no comando `ec2 run-instances`, é necessário formatar as informações de dispositivo de bloco como JSON.

Este exemplo mostra o JSON para especificar um único dispositivo Elastic Block Store de 20 GiB a ser mapeado em `/dev/sdb` na instância de execução.

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

Para anexar vários dispositivos, liste os objetos em uma matriz, como no exemplo a seguir.

```
[
  {
```

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

Você pode inserir o JSON diretamente na linha de comando (consulte Colocar sequências entre aspas (p. 47)) ou salvá-lo em um arquivo referenciado na linha de comando (consulte Carregar parâmetros a partir de um arquivo (p. 48)).

Ao fornecer grandes blocos de dados, é mais fácil salvar o JSON em um arquivo e fazer referência a ele a partir da linha de comando. É mais fácil de ler, editar e compartilhar dados JSON em um arquivo. Essa técnica é descrita em uma seção posterior.

Para obter mais informações sobre JSON, consulte JSON.org, entrada JSON na Wikipedia e RFC4627 - O tipo de mídia json/aplicativo para JSON.

Colocar sequências entre aspas

A maneira como insere os parâmetros formatados pelo JSON na linha de comando difere dependendo de seu sistema operacional.

Linux, macOS ou Unix

Use aspas simples (') para estabelecer a estrutura de dados JSON, como no exemplo a seguir:

```
$ aws ec2 run-instances --image-id ami-12345678 --
block-device-mappings '[{"DeviceName":"/dev/sdb","Ebs":
{"VolumeSize":20,"DeleteOnTermination":false,"VolumeType":"standard"}}]'
```

Windows PowerShell

Para o Windows PowerShell, é necessário uma única cotação (') para estabelecer a estrutura de dados JSON, bem como uma barra invertida (\) para escapar cada aspas duplas (") dentro da estrutura JSON, como no exemplo a seguir:

```
PS C:\> aws ec2 run-instances --image-id ami-12345678 --block-device-
mappings '[{\"DeviceName\":\"/dev/sdb\",\"Ebs\":{\"VolumeSize\":20,
\"DeleteOnTermination\":false,\"VolumeType\":\"standard\"}}]'
```

Prompt de comando do Windows

O prompt de comando do Windows exige aspas duplas (") para estabelecer a estrutura de dados JSON. Você deve usar o caractere de escape (precedido com um caractere de barra invertida (\)) para cada aspas duplas (") dentro da estrutura de dados JSON, como no exemplo a seguir:

```
C:\> aws ec2 run-instances --image-id ami-12345678 --block-device-
mappings "[{\"DeviceName\":\"/dev/sdb\",\"Ebs\":{\"VolumeSize\":20,
\"DeleteOnTermination\":false,\"VolumeType\":\"standard\"}}]"
```

Somente as aspas duplas mais externas não são de escape.

Se o valor de um parâmetro é em si um documento JSON, escape as aspas no documento JSON incorporado. Por exemplo, o parâmetro `attribute` para `aws sqs create-queue` pode levar uma chave `RedrivePolicy`. O parâmetro `--attributes` usa um documento JSON, que, por sua vez, contém `RedrivePolicy` que também leva um documento JSON como seu valor. O JSON interno incorporado no JSON externo deve ser recuado:

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

Carregar parâmetros a partir de um arquivo

Para evitar a necessidade de escapar sequências de JSON na linha de comando, carregue o JSON a partir de um arquivo. Carregue parâmetros de um arquivo local fornecendo o caminho para o arquivo usando o prefixo `file://`, como nos exemplos a seguir. Os caminhos do arquivo são interpretados como sendo relativos ao diretório de trabalho atual.

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
```

Windows

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

A opção de prefixo `file://` é compatível com as expansões de estilo Unix, incluindo '`~/`', '`./`' e '`../`'. No Windows, a expressão '`~/`' se expande para o seu diretório de usuário, armazenado na variável de ambiente `%USERPROFILE%`. Por exemplo, no Windows 10 haveria normalmente um diretório de usuário em `C:\Users\User Name\`.

Os documentos JSON que são incorporados como o valor de outro documento JSON ainda deverão ser recuados:

```
$ 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\"}"
}
```

Arquivos binários

Para comandos que incluem dados binários como um parâmetro, especifique que os dados são conteúdo binário usando o prefixo `fileb://`. Comandos que aceitam dados binários incluem:

- **`aws ec2 run-instances`** – parâmetro `--user-data`.

- **`aws s3api put-object`** – parâmetro `--sse-customer-key`.
- **`aws kms decrypt`** – parâmetro `--ciphertext-blob`.

O exemplo a seguir gera uma chave binária AES de 256 bits usando uma ferramenta de linha de comando do Linux e, em seguida, fornece para o Amazon S3 para criptografar o arquivo enviado no lado do 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\""
}
```

Arquivos remotos

A AWS CLI também é compatível com parâmetros de carregamento de um arquivo hospedado na Internet com uma URL `http://` ou `https://`. O exemplo a seguir faz referência a um arquivo armazenado em um bucket do Amazon S3. Isso permite que você acesse arquivos de parâmetros de qualquer computador, mas requer que o container seja acessível publicamente.

```
$ aws ec2 run-instances --image-id ami-12345678 --block-device-mappings http://my-
bucket.s3.amazonaws.com/filename.json
```

O exemplo anterior assume que o arquivo `filename.json` contém os seguintes dados JSON:

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

Para outro exemplo que se refere a um arquivo que contém parâmetros mais complexos formatados pelo JSON, consulte Conexão de uma política gerenciada do IAM a um usuário do IAM (p. 81).

Gerar parciais de CLI e parâmetros de entrada JSON de CLI

A maioria dos comandos da AWS CLI é compatível com a capacidade de aceitar todas as entradas de parâmetros a partir de um arquivo usando o parâmetro `--cli-input-json`.

Esses mesmos comandos fornecem de uma forma útil o `--generate-cli-skeleton` para gerar um arquivo com todos os parâmetros que você pode editar e preencher. Em seguida, você pode executar o comando com o parâmetro `--cli-input-json` e apontar para o arquivo preenchido.

O parâmetro `--generate-cli-skeleton` faz com que o comando não seja executado, mas, em vez disso, gere e exibe um modelo de parâmetro que você pode personalizar e usar como entrada em um comando posterior. O modelo gerado inclui todos os parâmetros compatíveis com o comando.

Por exemplo, se você executar o seguinte comando, ele gerará o modelo de parâmetro para o comando Amazon EC2 run-instances:

```
$ 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
}
```

Para gerar e usar um arquivo de esqueleto de parâmetro

1. Execute o comando com o parâmetro `--generate-cli-skeleton` e direcione a saída para um arquivo para salvá-lo:

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

2. Abra o arquivo de esqueleto do parâmetro em seu editor de texto e remova os parâmetros que não são mais necessários. Por exemplo, você pode simplificá-lo da seguinte forma:

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

   Neste exemplo, vamos deixar o parâmetro `DryRun` definido como verdadeiro para usar o recurso de simulação do EC2, que permite que você teste com segurança o comando sem realmente criar ou modificar qualquer recurso.

3. Preencha os valores restantes com valores apropriados para seu cenário. Neste exemplo, fornecemos o tipo de instância, nome da chave, grupo de segurança e identificador da AMI a ser usado. Este exemplo assume a região padrão. A AMI `ami-dfc39aef` é uma imagem de Amazon Linux de 64 bits na região `us-west-2`. Se você usar uma região diferente, você deverá encontrar o ID da AMI correta para usar.

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

4. Execute o comando com os parâmetros concluídos, transmitindo o arquivo JSON para o parâmetro `--cli-input-json` usando o prefixo `file://`. A AWS CLI interpreta o caminho a ser associado ao diretório de trabalho atual de forma que o exemplo a seguir que exibe somente o nome do arquivo sem nenhum caminho seja procurado diretamente no diretório de trabalho atual.

```
$ 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.
```

   O erro indica que a simulação JSON está formada corretamente e os valores de parâmetro são válidos. Se quaisquer outros problemas forem relatados na saída, faça a correção e repita a etapa acima até que a mensagem "Solicitação bem-sucedida" seja exibida.
5. Agora você pode definir o parâmetro `DryRun` como `false` para desativar a simulação:

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

6. Agora, quando você executar o comando, `run-instances` realmente executará uma instância do EC2 e exibirá os detalhes gerados pela execução bem-sucedida:

```
$ aws ec2 run-instances --cli-input-json file://ec2runinst.json
{
    "OwnerId": "123456789012",
    "ReservationId": "r-d94a2b1",
    "Groups": [],
    "Instances": [
...
```

Controlar a saída de comando da AWS Command Line Interface

Esta seção descreve as diferentes maneiras de controlar a saída do AWS CLI.

Tópicos

Como selecionar o formato de saída

O AWS CLI é compatível com três diferentes formatos de saída:

- JSON (`json`)
- Texto delimitado por tabulação (`text`)
- Tabela no formato ASCII (`table`)

Como explicado no tópico sobre configuração (p. 19), o formato de saída pode ser especificado de três diferentes maneiras:

- Usando a opção `output` em um perfil nomeado no arquivo `config`. O exemplo a seguir define o formato de saída padrão como `text`:

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

- Usando a variável de ambiente `AWS_DEFAULT_OUTPUT`. A saída a seguir define o formato como `table` para os comandos nesta sessão de linha de comando até que a variável seja alterada ou a sessão seja encerrada. Usar essa variável de ambiente substitui qualquer valor definido no arquivo `config`:

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

- Usando a opção `--output` na linha de comando. O exemplo a seguir define a saída de apenas este único comando como `json`. Usar essa opção no comando substitui qualquer variável de ambiente definida atualmente ou o valor no arquivo `config`.

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

As regras de precedência da AWS CLI (p. 21) se aplicam. Por exemplo, usar a variável de ambiente `AWS_DEFAULT_OUTPUT` substitui qualquer valor definido no arquivo `config`, e um valor passado para um comando da AWS CLI com `--output` substitui qualquer valor definido na variável de ambiente ou no arquivo `config`.

A opção `json` é melhor para lidar com a saída de forma programática por meio de várias linguagens ou `jq` (um processador JSON de linha de comando).

O formato `table` é de fácil leitura.

O formato `text` funciona bem com ferramentas de processamento de texto Unix tradicionais, como `sed`, `grep` e `awk`, bem como em scripts do Windows PowerShell.

Como filtrar a saída com a opção `--query`

A AWS CLI fornece recursos de filtragem de saída integrada com base em JSON com a opção `--query`. Para mostrar como ele funciona, vamos começar com o padrão JSON saída, que descreve dois volumes de Elastic Block Storage (EBS) conectados a instâncias separadas do EC2.

```
$ 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
        }
    ]
}
```

Nós podemos optar por exibir somente o primeiro volume na lista de `Volumes` com o seguinte 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
}
```

No exemplo a seguir, nós usamos a notação curinga `[*]` para iterar sobre todos os volumes na lista e também filtrar três elementos: `VolumeId`, `AvailabilityZone` e `Size` para cada um deles. Observe que a notação do dicionário exige que você forneça um alias para cada chave JSON, por exemplo: `{Alias1:JSONKey1,Alias2:JSONKey2}`. Um dicionário é inerentemente não ordenado, de modo que a ordem de principais aliases dentro de uma estrutura pode não ser consistente.

```
$ 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
    }
]
```

Na notação de dicionário, também é possível usar chaves encadeadas como `key1.key2[0].key3` para filtrar elementos altamente aninhados na estrutura. O exemplo a seguir demonstra isso com a chave `Attachments[0].InstanceId`, com o alias simples `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
    }
]
```

Você também pode filtrar vários elementos usando a notação da lista: `[key1, key2]`. Isso formata todos os atributos filtrados em uma única lista ordenada por objeto, independentemente do 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 resultados pelo valor de um campo específico, use o operador JMESPath "?". O exemplo a seguir saídas de consulta apenas volumes no us-west-2a zona de disponibilidade:

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

> **Note**
>
> Ao especificar um valor literal, como "us-west-2" acima em uma expressão de consulta JMESPath, você deve delimitar o valor em acentos graves (`) para ser lido corretamente.

Combinado com três formatos de saída que serão explicados em mais detalhes nas seções a seguir, a opção `--query` é uma ferramenta poderosa que você pode usar para personalizar o conteúdo e estilo de saídas. Para mais exemplos e a especificação completa de JMESPath, a biblioteca de processamento de JSON subjacente, visite http://jmespath.org/specification.html.

Formato de saída JSON

JSON é o formato de saída padrão do AWS CLI. A maioria dos idiomas pode facilmente decodificar strings JSON usando funções internas ou com bibliotecas disponíveis publicamente. Como mostrado no tópico anterior, juntamente com exemplos de saída, a opção `--query` fornece ótimas maneiras de filtrar e formatar a saída formatada JSON da AWS CLI. Se precisar de mais recursos avançados que podem não ser possíveis com `--query`, verifique `jq`, uma linha de comando JSON. Faça o download e localize o tutorial em: oficial http://stedolan.github.io/jq/.

Formato de saída de texto

O formato de texto organiza a saída da AWS CLI em linhas delimitados por tabulação. Isso funciona bem com ferramentas de processamento de texto Unix tradicionais, como `grep`, `sed` e `awk`, bem como o processamento de texto executado pelo Windows PowerShell.

O formato de saída de texto segue a estrutura básica mostrada abaixo. As colunas são classificados alfabeticamente por nomes de chaves correspondentes subjacentes do objeto JSON.

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

Veja a seguir um exemplo de uma saída 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
```

> **Important**
>
> Recomendamos que, se você especificar uma saída *`text`*, que você também sempre use a opção *`--query`* para garantir um comportamento consistente. Isso ocorre porque o formato de texto ordena alfabeticamente as colunas de saída pelo nome da chave do objeto JSON subjacente e recursos similares podem não têm os mesmos nomes de chave. Por exemplo, a representação JSON de uma instância do EC2 baseada em Linux pode ter elementos que não estão presentes na representação JSON de uma instância baseada em Windows ou vice-versa. Além disso, os recursos podem ter elementos de valores de chave adicionados ou removidos em futuras atualizações, alterando a ordem das colunas. Este é o local em que `--query` expande a funcionalidade da saída de texto para fornecer a você total controle sobre o formato de saída. No exemplo a seguir, o comando especifica quais elementos devem ser exibidos e define a ordem das colunas com a notação da lista `[key1, key2, ...]`. Isso oferece a você a segurança de que os valores de chave corretos sempre serão exibido na coluna esperada. Por fim, observe como a AWS CLI resulta em `None` como valores para chaves que não existem.

```
$ 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
```

O exemplo a seguir mostra como você pode usar `grep` e `awk` com a saída `text` do comando `aws ec2 describe-instances`. O primeiro comando exibe a Zona de disponibilidade, estado atual e ID de instância de cada instância na saída de texto. O segundo comando processa essa saída para exibir somente os IDs de todas as instâncias em execução na Zona de disponibilidade `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
```

O exemplo a seguir vai mais longe e mostra não apenas como filtrar a saída, mas como usar essa saída para automatizar a alteração dos tipos de instância para cada instância interrompida.

```
$ 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
```

A saída de texto também pode ser útil no Windows PowerShell. Como as colunas na saída `text` são delimitadas por tabulação, elas são facilmente divididas em uma matriz no delimitador do PowerSell `` `t ``. O comando a seguir exibe o valor da terceira coluna (`InstanceId`) se a primeira coluna (`AvailabilityZone`) corresponder à string: `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
```

Tabela Formato de Saída

O formato `table` produz representações legíveis da saída da AWS CLI complexa em um formato tabular:

```
$ 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  |||
||+---------------------------+-----------------------+-------------+--------------
+------------+----------------+||
```

Você pode combinar a opção `--query` com o formato de tabela para exibir um conjunto de elementos pré-selecionado na saída bruta. Observe as diferenças das saída entre as notações de lista e dicionário: nomes de coluna são alfabeticamente ordenados no primeiro exemplo e colunas sem nome são ordenados conforme definido pelo usuário no segundo exemplo.

```
$ 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  |
+--------------+--------------+--------------+-----+
```

Usar sintaxe simplificada com o AWS Command Line Interface

A AWS Command Line Interface pode aceitar muitos de seus parâmetros de opção no formato JSON. No entanto, pode ser entediante digitar grandes estruturas ou listas de JSON na linha de comando. Para tornar isso mais fácil, a AWS CLI também oferece suporte a uma sintaxe abreviada mais simples que permite a representação de seus parâmetros de opção em vez de utilizar o formato JSON completo.

Parâmetros de estrutura

A sintaxe abreviada no AWS CLI facilita a inserção de parâmetros simples de entrada pelos usuários (estruturas não aninhadas). O formato é uma lista de pares de chave/valor separados por vírgula:

Linux, macOS, or Unix

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

Windows PowerShell

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

Ambos são equivalentes ao exemplo a seguir formatado em JSON:

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

Não pode haver nenhum espaço em branco entre cada pares de chave/valor separado por vírgula. Aqui está um exemplo do comando `update-table` do DynamoDB com a opção `--provisioned-throughput` especificada no formato simplificado.

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

Isso é equivalente ao exemplo a seguir formatado em JSON:

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

Parâmetros de lista

Os parâmetros de entrada em um formulário de lista podem ser especificados de duas formas: JSON ou abreviada. A sintaxe abreviada da AWS CLI é projetada para facilitar a inserção de listas com número,

sequência de caracteres, estruturas aninhados ou não. O formato básico é mostrada aqui, onde os valores na lista são separados por um único espaço.

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

Isso é equivalente ao exemplo a seguir formatado em JSON.

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

Como mencionado anteriormente, é possível especificar uma lista de números, uma lista de strings ou uma lista de estruturas de dados não aninhados em formato abreviado. Veja a seguir um exemplo do comando `stop-instances` do Amazon EC2, em que o parâmetro de entrada (lista de strings) para a opção `--instance-ids` é especificado no formato simplificado:

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

Isso é equivalente ao exemplo a seguir formatado em JSON:

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

A seguir está um exemplo do comando `create-tags` do Amazon EC2, que leva uma lista de estruturas não aninhadas para a opção `--tags`. A opção `--resources` especifica o ID da instância a ser marcada.

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

Isso é equivalente ao exemplo a seguir formatado em JSON. O parâmetro JSON é escrito em várias linhas para melhor leitura.

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

Usar as opções de paginação da AWS Command Line Interface

Para comandos que podem retornar uma grande lista de itens, a AWS CLI adiciona três opções que você pode usar para controlar o número de itens incluídos na saída quando a CLI acessa uma API do serviço para preencher a lista.

Por padrão, o CLI usa um tamanho de página de 1.000 e recupera todos os itens disponíveis. Por exemplo, se você executar `aws s3api list-objects` em um bucket Amazon S3 que contém 3.500 objetos, a CLI fará quatro chamadas para Amazon S3, lidando com a lógica específica de paginação para você em segundo plano e retornando todos os 3.500 objetos na saída final.

Caso haja problemas ao executar os comandos da lista em um grande número de recursos, o tamanho da página padrão de 1.000 poderá ser muito alto. Isso pode fazer com que as chamadas para os serviços do AWS excedam o tempo máximo permitido e gerar um erro de "expiração". Você pode usar a opção `--page-size` para especificar que a CLI solicite um número menor de itens de cada chamada para o

serviço do AWS. A CLI ainda recuperará a lista completa, mas executará um número maior de chamadas de API de serviço em segundo plano e recuperará um número menor de itens em cada chamada: Isso fornece às chamadas individuais mais chances de sucesso sem um tempo limite. Alterar o tamanho da página não afeta a saída, afeta somente o número de chamadas de API que precisam ser feitas para gerar a saída.

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

Para recuperar menos itens por vez na saída da CLI, use a opção `--max-items`. A CLI ainda lida com paginação conforme descrito acima, mas imprime apenas o número de itens que você especificar de cada vez:

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

Se o número de saída de itens (`--max-items`) for menor do que o número total de itens retornados pelas chamadas de API subjacentes, a saída incluirá um `NextToken` que pode ser passado para um comando subsequente para recuperar o próximo conjunto de itens. O exemplo a seguir mostra como usar o valor `NextToken` retornado pelo exemplo anterior e permite que você recupere a segunda centena de itens:

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

O serviço do AWS especificado pode não retornar itens na mesma ordem cada vez que você chamá-lo. Se você especificar valores diferentes para `--page-size` e `--max-items`, poderá obter resultados inesperados com itens ausentes ou duplicados. Para evitar que isso aconteça, use o mesmo número para `--page-size` e `--max-items` para sincronizar a paginação da CLI com a do serviço subjacente. Também é possível recuperar a lista completa e executar você mesmo quaisquer operações de paginação necessárias localmente.

Uso da AWS CLI para trabalhar com os Serviços do AWS

Esta seção fornece exemplos que mostram como usar a AWS CLI para acessar vários Serviços do AWS.

Para ter uma referência completa para todos os comandos disponíveis para cada serviço, consulte a AWS CLI Command Reference ou use a ajuda da linha de comando. Para obter mais informações, consulte Receber ajuda com a AWS Command Line Interface (p. 39).

Tópicos

Uso do Amazon DynamoDB com a AWS Command Line Interface

A AWS Command Line Interface (AWS CLI) fornece suporte para todos os serviços de banco de dados do AWS, incluindo Amazon DynamoDB. Use o AWS CLI para operações ad hoc, como a criação de uma tabela. Também é possível usá-lo para operações DynamoDB incorporadas em scripts.

Para listar os comandos da AWS CLI do DynamoDB, use o comando a seguir:

```
aws dynamodb help
```

Antes de executar quaisquer comandos, defina suas credenciais padrão. Para obter mais informações, consulte Configurar o AWS CLI (p. 19).

O formato de linha de comando consiste em um Amazon DynamoDB nome da API, seguido os parâmetros para essa API. A AWS CLI oferece suporte à sintaxe abreviada (p. 59) da CLI para os valores de parâmetro, assim como JSON completo.

Por exemplo, o seguinte comando cria uma tabela chamada `MusicCollection`.

> Note
>
> Para facilitar a leitura, comandos longos nesta seção são divididos em linhas separadas. O caractere de barra invertida é o caractere de continuação de linha da linha de comando do Linux e permite que você copie e cole (ou digite) várias linhas em um prompt do Linux. Se você estiver usando um shell que não use a barra invertida para continuação de linha, substitua a barra

invertida pelo caractere de continuação de linha do shell ou remova as barras invertidas e coloque todo o comando em uma única linha.

```
$ 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
```

Em seguida, adicione novas linhas à tabela com comandos semelhantes aos mostrados no exemplo a seguir. Esses exemplos usam uma combinação de sintaxe abreviada e 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"
    }
}
```

Pode ser difícil compor um JSON válido em um comando com uma única linha. Para tornar isso mais fácil, a AWS CLI pode ler arquivos JSON. Por exemplo, considere o seguinte trecho de código JSON, que é armazenado em um arquivo chamado `expression-attributes.json`:

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

Você pode usar esse arquivo para emitir uma solicitação `query` usando a AWS CLI. Neste exemplo a seguir, o conteúdo do arquivo `expression-attributes.json` é usado para o parâmetro `--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 obter mais informações sobre como usar a AWS CLI com o DynamoDB, DynamoDB no AWS CLI Command Reference.

Além do DynamoDB, é possível usar a AWS CLI com DynamoDB local. O DynamoDB Local é um banco de dados e servidor pequeno no lado do cliente que copia o serviço DynamoDB. O DynamoDB Local permite criar aplicativos que usam a API DynamoDB sem de fato manipular tabelas ou dados no serviço web do DynamoDB. Em vez disso, todas as ações da API são roteadas para um banco de dados local. Isso economiza a taxa de transferência provisionada, o armazenamento de dados e as taxas de transferência de dados.

Para mais informações sobre o DynamoDB Local e como usá-lo com o AWS CLI, consulte as seguintes seções do Guia do desenvolvedor do Amazon DynamoDB:

- DynamoDB Local
- Usar o AWS CLI com o DynamoDB Local

Uso do Amazon EC2 com a AWS Command Line Interface

Acesse os recursos do Amazon EC2 usando a AWS CLI. Para listar os comandos da AWS CLI do Amazon EC2 use o comando a seguir:

```
aws ec2 help
```

Antes de executar quaisquer comandos, defina suas credenciais padrão. Para obter mais informações, consulte Configurar o AWS CLI (p. 19).

Este tópico mostra exemplos de comandos da CLI que executam tarefas comuns para o Amazon EC2.

Tópicos

Criar, exibir e excluir pares de chaves do EC2

Use a AWS CLI para criar, exibir e excluir seus pares de chaves para Amazon EC2. Você usa pares de chaves ao se conectar a uma instância do Amazon EC2. Você deve fornecer o par de chaves para o EC2

quando você cria a instância e, em seguida, usar esse par de chaves para autenticar ao se conectar à instância.

> **Note**
>
> Os exemplos mostrados a seguir assumem que você já configurou suas credenciais padrão (p. 64).

Tópicos

Criação de um par de chaves

Para criar um par de chaves, use o comando create-key-pair `--query`com a opção e a opção `--output text` para canalizar sua chave privada diretamente em um arquivo.

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

Observe que para o Windows PowerShell, o padrão de redirecionamento `> file` assume a codificação UTF-8, que não pode ser usada com alguns clientes SSH. Portanto, você deve converter a saída redirecionando-a para o comando `out-file` e definir explicitamente a codificação para `ascii`:

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

O arquivo resultante `MyKeyPair.pem` é semelhante a:

```
-----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-----
```

Sua chave privada não é armazenada no AWS e só pode ser recuperada quando ela é criada. Não será possível recuperá-la posteriormente. Ao invés disso, se você perder a chave privada, deverá criar um novo par de chaves.

Se você estiver se conectando à sua instância a partir de um computador com Linux, recomendamos que você use o seguinte comando para definir as permissões do arquivo de chave privada, de maneira que apenas você possa lê-lo.

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

Exibir seu par de chaves

A 'impressão digital' é gerada a partir do par de chaves, e você pode usá-la para verificar se a chave privada que você tem em sua máquina local corresponde à chave pública armazenada no AWS. A impressão digital é um hash SHA1 de uma cópia codificada DER da chave privada. Esse valor é capturado quando o par de chaves é criado e é armazenado no AWS juntamente com a chave pública. A impressão digital pode ser visualizada no console de gerenciamento do Amazon EC2 ou executando o comando da CLI `aws ec2 describe-key-pairs`. O exemplo a seguir exibe a impressão digital para `MyKeyPair`:

```
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 mais informações sobre chaves e impressões digitais, consulte a página Pares de chave do Amazon EC2 no Guia do usuário do Amazon EC2 para instâncias do Linux.

Excluir o par de chaves

Para excluir um par de chaves, execute o comando a seguir, substituindo *MyKeyPair* pelo nome do par que você deseja excluir:

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

Criar, configurar e excluir grupos de segurança para Amazon EC2

Você pode criar um grupo de segurança para suas instâncias do EC2 que essencialmente opera como um firewall, com regras que determinam qual tráfego de rede pode entrar e sair. Você pode criar grupos de segurança a serem usados em uma VPC ou na rede simples compartilhada pelo EC2-Classic. Para mais informações sobre as diferenças entre o EC2-Classic e o EC2-VPC, consulte Plataformas compatíveis no Guia do usuário do Amazon EC2 para instâncias do Linux.

Você pode usar a AWS CLI para criar um novo grupo de segurança, adicionar regras a grupos de segurança existentes e excluir grupos de segurança.

> **Note**
>
> Os exemplos mostrados a seguir assumem que você já configurou suas credenciais padrão (p. 64).

Tópicos

- Excluir o grupo de segurança (p. 70)

Criar um grupo de segurança

Você pode criar grupos de segurança associado a VPCs ou para o EC2-Classic.

EC2-VPC

O exemplo a seguir mostra como criar um grupo de segurança para a VPC especificada:

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

Para visualizar as informações iniciais para um grupo de segurança, execute o comando describe-security-groups. Observe que você pode fazer referência a um grupo de segurança do EC2-VPC somente por seu `vpc-id`, e não apenas por seu nome.

```
$ 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

O exemplo a seguir mostra como criar um grupo de segurança para o EC2-Classic:

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

Para visualizar as informações iniciais para `my-sg`, execute o comando describe-security-groups. Para um grupo de segurança do EC2-Classic, você pode fazer referência a ele por seu nome:

```
aws ec2 describe-security-groups --group-names my-sg
```

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

Como adicionar regras ao grupo de segurança

Quando você executa uma instância do Amazon EC2, você deve ativar regras no grupo de segurança para permitir o tráfego de rede de entrada como meio de se conectar à imagem. Por exemplo, se estiver iniciando uma instância do Windows, normalmente você adiciona uma regra para permitir que o tráfego de entrada na porta TCP 3389 ofereça suporte ao Protocolo de Desktop Remoto (RDP). Se estiver iniciando uma instância do Linux, geralmente você adiciona uma regra para permitir que o tráfego de entrada na porta 22 ofereça suporte às conexões SSH. Use o comando `authorize-security-group-ingress` para adicionar uma regra a um grupo de segurança. Um dos parâmetros obrigatórios deste comando é o endereço IP público de seu computador ou rede (na forma de um intervalo de endereços) ao qual seu computador está conectado na notação CIDR.

> Note
>
> Fornecemos o seguinte serviço: https://checkip.amazonaws.com/ para permitir que você determine seu endereço IP público. Para encontrar outros serviços que podem ajudar você a identificar o endereço IP, use o navegador para pesquisar "qual é meu endereço IP". Se você se conectar por meio de um ISP ou protegido por um firewall usando um endereço IP dinâmico, seu endereço poderá mudar periodicamente. Nesse caso, você deve descobrir o intervalo de endereços IP usados por computadores cliente.

EC2-VPC

O exemplo a seguir mostra como adicionar uma regra para RDP (porta TCP 3389) em um grupo de segurança do EC2-VPC com o ID `sg-903004f8`. Este exemplo assume que o computador cliente tem um endereço em algum lugar no intervalo CIDR 203.0.113.0/24.

Você pode começar, confirmando que o endereço público seja exibido como incluído no intervalo CIDR 203.0.113.0/24:

```
$ curl https://checkip.amazonaws.com
203.0.113.57
```

Com essas informações confirmadas, você pode adicionar o intervalo ao seu grupo de segurança, executando `authorize-security-group-ingress`:

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

O comando a seguir adiciona uma regra para habilitar o SSH para instâncias no mesmo grupo de segurança:

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

Para visualizar as alterações no grupo de segurança, execute o comando describe-security-groups:

```
$ 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

O comando a seguir adiciona uma regra para RDP ao grupo de segurança EC2-Classic chamado *my-sg*:

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

O comando a seguir adiciona outra regra para SSH para o mesmo grupo de segurança:

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

Para visualizar as alterações em seu grupo de segurança, execute o comando describe-security-groups:

```
$ 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"
        }
    ]
}
```

Excluir o grupo de segurança

Para excluir um grupo de segurança, execute o comando delete-security-group. Não é possível excluir um grupo de segurança se ele estiver anexado no momento a um ambiente.

EC2-VPC

O comando a seguir exclui um grupo de segurança do EC2-VPC:

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

EC2-Classic

O comando a seguir exclui o grupo de segurança do EC2-Classic chamado `my-sg`:

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

Inicializar, listar e encerrar instâncias do EC2

Use a AWS CLI para iniciar, listar e encerrar instâncias do Amazon EC2. Você precisará de um par de chaves (p. 64) e de um grupo de segurança (p. 66). Será necessário selecionar uma Imagem de máquina da Amazon (AMI) e anotar o ID de AMI. Para mais informações, consulte Como localizar uma AMI adequada no Guia do usuário do Amazon EC2 para instâncias do Linux.

> **Important**
>
> Se você executar uma instância que não esteja no nível gratuito do AWS, será faturado depois do início da instância e cobrado pelo tempo que executou a instância, mesmo se ela permanecer ociosa.

> **Note**
>
> Os exemplos mostrados a seguir assumem que você já configurou suas credenciais padrão (p. 64).

Tópicos

- Como iniciar uma instância (p. 71)

Como iniciar uma instância

Para iniciar uma instância do Amazon EC2 usando a AMI selecionada, use o comando run-instances. Você pode executar a instância em uma VPC ou, se sua conta oferecer suporte, no EC2-Classic.

Inicialmente, a instância será exibida no estado `pending`, mas mudará para o estado `running` depois de alguns minutos.

EC2-VPC

O exemplo a seguir mostra como iniciar uma instância `t2.micro` na sub-rede especificada de uma VPC. Substitua os valores dos parâmetros *em vermelho e itálico* por seus próprios.

```
$ aws ec2 run-instances --image-id ami-xxxxxxxx --count 1 --instance-type t2.micro --key-
name MyKeyPair --security-group-ids sg-903004f8 --subnet-id subnet-6e7f829e
{
    "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

Se a conta for compatível com ele, você poderá usar o comando a seguir para iniciar uma instância `t1.micro` no EC2-Classic. Substitua os valores dos parâmetros *em vermelho e itálico* por seus próprios.

```
$ aws ec2 run-instances --image-id ami-173d747e --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
        }
    ]
}
```

Adição de um dispositivo de blocos adicional em sua instância

Cada instância lançada tem um volume do dispositivo root associados. Use o mapeamento de dispositivos de blocos para especificar mais volumes do EBS ou volumes de armazenamento de instâncias para anexar a uma instância quando ela for executada.

Para adicionar um dispositivo de blocos em sua instância, especifique a opção `--block-device-mappings` ao usar `run-instances`.

O parâmetro de exemplo a seguir provisiona um volume de Amazon EBS padrão que é de 20 GB e o mapeia para sua instância usando o identificador `/dev/sdf`.

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

O exemplo a seguir adiciona um volume do Amazon EBS, mapeado para `/dev/sdf`, com base em um snapshot existente. Um snapshot representa uma imagem que é carregada no volume para você. Ao especificar um snapshot, não é necessário especificar um tamanho de volume; ele será grande o suficiente para armazenar sua imagem. No entanto, se você especificar um tamanho, ele deverá ser maior que ou igual ao tamanho do snapshot.

```
--block-device-mappings "[{\"DeviceName\":\"/dev/sdf\",\"Ebs\":{\"SnapshotId\":\"snap-
a1b2c3d4\"}}]"
```

O exemplo a seguir adiciona dois volumes à sua instância. Note que o número de volumes disponíveis para sua instância depende do seu tipo de instância.

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

O exemplo a seguir cria o mapeamento (`/dev/sdj`), mas não provisiona um volume para a instância:

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

Para obter mais informações, consulte Mapeamento de dispositivo de bloco no Guia do usuário do Amazon EC2 para instâncias do Linux.

Adição de uma tag à sua instância

Uma tag é um rótulo que você atribui a um recurso da AWS. Permite que você adicione metadados aos seus recursos que você pode usar para diversas finalidades. Para mais informações, consulte Como colocar tags nos seus recursos em Guia do usuário do Amazon EC2 para instâncias do Linux.

O exemplo a seguir mostra como adicionar uma tag com o nome da chave "`Name` e o valor "`MyInstance`" à instância especificada, usando o comando create-tags:

```
$ aws ec2 create-tags --resources i-5203422c --tags Key=Name,Value=MyInstance
```

Conectar à sua instância

Durante a execução da instância, é possível se conectar a ela e usá-la da mesma forma que você usaria um computador. Para obter mais informações, consulte o tópico Conecte-se à sua instância do Amazon EC2 no Guia do usuário do Amazon EC2 para instâncias do Linux.

Listar suas instâncias

Use o AWS CLI para listar suas instâncias e visualizar informações sobre elas. Liste todas as suas instâncias, ou filtre os resultados de acordo com as instâncias de interesse.

Os exemplos a seguir mostram como usar o comando describe-instances.

O comando a seguir filtra a lista apenas para suas instâncias `t2.micro` e mostra apenas os valores `InstanceId` para cada correspondência.

```
$ aws ec2 describe-instances --filters "Name=instance-type,Values=t2.micro" --query
 "Reservations[].Instances[].InstanceId"
[
    "i-05e998023d9c69f9a"
]
```

O comando a seguir lista todas as suas instâncias que tem a tag `Name=MyInstance`.

```
$ aws ec2 describe-instances --filters "Name=tag:Name,Values=MyInstance"
```

O comando a seguir relaciona as instâncias que foram executadas usando qualquer uma das seguintes AMIs: `ami-x0123456`, `ami-y0123456` e `ami-z0123456`.

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

Como encerrar uma instância

O encerramento de uma instância significa excluí-la. Você não pode se reconectar com uma instância depois de tê-la encerrado. Assim que o estado da instância de mudar para `shutting-down` ou para `terminated`, não há mais custos para essa instância. Se você desejar se reconectar a ela mais tarde, use stop-instances em vez de `terminate-instances`. Para obter mais informações, consulte Encerrar sua instância no Guia do usuário do Amazon EC2 para instâncias do Linux.

Ao concluir uma instância, você pode usar o comando terminate-instances para excluí-la:

```
$ aws ec2 terminate-instances --instance-ids i-5203422c
```

```
{
    "TerminatingInstances": [
        {
            "InstanceId": "i-5203422c",
            "CurrentState": {
                "Code": 32,
                "Name": "shutting-down"
            },
            "PreviousState": {
                "Code": 16,
                "Name": "running"
            }
        }
    ]
}
```

Uso do Amazon S3 Glacier com a AWS Command Line Interface

Acesse os recursos do Amazon S3 Glacier usando a AWS CLI. Para listar os comandos da AWS CLI do Glacier use o comando a seguir:

```
aws glacier help
```

Este tópico mostra exemplos de comandos da CLI que executam tarefas comuns para o Glacier. Os exemplos demonstram como usar a CLI para fazer upload de um arquivo grande para Glacier dividindo em partes menores e fazendo o upload a partir da linha de comando.

Antes de executar quaisquer comandos, defina suas credenciais padrão. Para obter mais informações, consulte Configurar o AWS CLI (p. 19).

> **Note**
>
> Este tutorial usa várias ferramentas de linha de comando, que geralmente vêm pré-instaladas no Unix, como sistemas operacionais, incluindo Linux e OS X. Os usuários do Windows podem usar as mesmas ferramentas instalando Cygwin e executando os comandos do terminal Cygwin. Comandos originários do Windows e utilitários que executam as mesmas funções são observados onde disponível.

Tópicos

Criação de um cofre Glacier

Crie um cofre com o comando `create-vault`.

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

Note

Todos os comandos do Glacier exigem um parâmetro de ID de conta. Use o caractere de hífen (`--account-id -`) para usar a conta atual.

Preparação de um arquivo para carregamento

Crie um arquivo para o upload de teste. Os comandos a seguir criam um arquivo com o nome *largefile* com exatamente 3 MiB de dados aleatórios.

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` é um utilitário que copia um número de bytes a partir de um arquivo de entrada para um arquivo de saída. O exemplo acima usa o arquivo de dispositivo do sistema `/dev/urandom` como uma fonte de dados aleatórios. O `fsutil` executa uma função semelhante no Windows:

Windows

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

Em seguida, divida o arquivo em blocos de 1 MiB (1.048.576 bytes).

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

Note

HJ-Dividir é um divisor de arquivos gratuito para Windows e muitas outras plataformas.

Inicialização de um multipart upload e upload de arquivos

Crie um multipart upload no Glacier usando o comando `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"
}
```

Glacier exige o tamanho de cada parte em bytes (1 MiB neste exemplo), o nome do cofre e uma ID da conta para configurar o multipart upload. AWS CLI resulta em um ID de upload quando a operação é concluída. Salve o ID de upload para uma variável de shell para uso posterior:

Linux, macOS, or Unix

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

Windows

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

Em seguida, use o comando `upload-multipart-part` para fazer o upload das três partes:

```
$ 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**
>
> O exemplo acima usa o sinal de dólar ("$") para fazer referência ao conteúdo da variável de shell `UPLOADID` no Linux. Na linha de comando do Windows, use dois sinais de porcentagem (por exemplo, `%UPLOADID%`).

Especifique o intervalo de bytes de cada parte ao fazer o upload para que ele possa ser remontado na ordem adequada ao Glacier. Cada parte é 1.048.576 bytes, portanto, a primeira parte ocupa 0-1048575 bytes, a segunda 1048576-2097151 e a terceira 2097152-3145727.

Conclusão do upload

O Glacier exige uma árvore hash do arquivo original para confirmar que todas as partes carregadas atinjam a AWS intactas. Para calcular uma árvore hash, divida o arquivo em 1 MiB partes e calcular um binário SHA-256 hash de cada item. Em seguida, divida a lista de hashes em pares, combine os dois binários hashes em cada par e execute hashes dos resultados. Repita esse processo até que haja apenas um hash à esquerda. Se houver um número ímpar de hashes em qualquer nível, envie para o próximo nível sem modificá-lo.

A chave para calcular uma árvore hash corretamente ao usar os utilitários de linha de comando é armazenar cada hash em formato binário e apenas converter para hexadecimal na última etapa. A combinação ou hash de qualquer versão hexadecimal hash em árvore gerará um resultado incorreto.

> **Note**
>
> Os usuários do Windows podem usar o comando `type` em vez do `cat`. OpenSSL está disponível para Windows em OpenSSL.org.

Para calcular uma árvore hash

1. Divida o arquivo original em partes de 1 MiB, caso ainda não tenha feito isso.

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

2. Calcule e armazene o hash SHA-256 binário de cada fragmento.

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

3. Combine os primeiros dois hashes e execute o hash binário do resultado.

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

4. Combine o pai de partes de hash aa e ab com o hash de bloco ac e hash o resultado, desta vez exibindo hexadecimal. Armazene o resultado em um shell variável.

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

Por fim, preencha o upload com o comando `complete-multipart-upload`. Este comando usa o tamanho do arquivo original em bytes, o valor de hash em árvore final em hexadecimal, e a ID da conta e nome do cofre.

```
$ 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"
}
```

Também é possível verificar o status do cofre usando o comando `describe-vault`:

```
$ aws glacier describe-vault --account-id - --vault-name myvault
{
    "SizeInBytes": 3178496,
    "VaultARN": "arn:aws:glacier:us-west-2:123456789012:vaults/myvault",
    "LastInventoryDate": "2018-12-07T00:26:19.028Z",
    "NumberOfArchives": 1,
    "CreationDate": "2018-12-06T21:23:45.708Z",
    "VaultName": "myvault"
}
```

> **Note**
>
> O status do cofre é atualizado cerca de uma vez por dia. Consulte Como trabalhar com cofres para obter mais informações

Já é seguro remover o bloco e os arquivos de hash que você criou:

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

Para mais informações sobre multipart uploads, consulte Upload de arquivos grandes em partes e Como computar somas de verificação no Guia do desenvolvedor do Amazon S3 Glacier.

AWS Identity and Access Management do AWS Command Line Interface

Acesse os recursos do AWS Identity and Access Management (IAM) usando a AWS CLI. Para listar os comandos da AWS CLI do IAM, use o comando a seguir:

```
aws iam help
```

Este tópico mostra exemplos de comandos da CLI que executam tarefas comuns para o IAM.

Antes de executar quaisquer comandos, defina suas credenciais padrão. Para obter mais informações, consulte Configurar o AWS CLI (p. 19).

Tópicos

Criação de novos usuários e grupos do IAM

Este tópico descreve como usar os comandos da AWS CLI para criar um novo grupo de IAM e um novo usuário do IAM e, em seguida, adicionar o usuário ao grupo.

Antes de executar quaisquer comandos, defina suas credenciais padrão. Para obter mais informações, consulte Configurar o AWS CLI (p. 19).

Como criar um grupo do IAM e adicionar um novo usuário do IAM a ele

1. Primeiro, use o comando `create-group` para criar o grupo:

```
$ aws iam create-group --group-name MyIamGroup
{
    "Group": {
        "GroupName": "MyIamGroup",
        "CreateDate": "2018-12-14T03:03:52.834Z",
        "GroupId": "AGPAJNUJ2W4IJVEXAMPLE",
        "Arn": "arn:aws:iam::123456789012:group/MyIamGroup",
        "Path": "/"
    }
}
```

2. Em seguida, use o comando `create-user` para criar o usuário:

```
$ aws iam create-user --user-name MyUser
{
```

```
    "User": {
        "UserName": "MyUser",
        "Path": "/",
        "CreateDate": "2018-12-14T03:13:02.581Z",
        "UserId": "AIDAJY2PE5XUZ4EXAMPLE",
        "Arn": "arn:aws:iam::123456789012:user/MyUser"
    }
}
```

3. Finalmente, use o comando `add-user-to-group` para adicionar o usuário ao grupo:

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

4. Para verificar se o grupo `MyIamGroup` contém o `MyUser`, use o comando `get-group`.

```
$ aws iam get-group --group-name MyIamGroup
{
    "Group": {
        "GroupName": "MyIamGroup",
        "CreateDate": "2018-12-14T03:03:52Z",
        "GroupId": "AGPAJNUJ2W4IJVEXAMPLE",
        "Arn": "arn:aws:iam::123456789012:group/MyIamGroup",
        "Path": "/"
    },
    "Users": [
        {
            "UserName": "MyUser",
            "Path": "/",
            "CreateDate": "2018-12-14T03:13:02Z",
            "UserId": "AIDAJY2PE5XUZ4EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:user/MyUser"
        }
    ],
    "IsTruncated": "false"
}
```

Conexão de uma política gerenciada do IAM a um usuário do IAM

Este tópico descreve como usar os comandos da AWS CLI para conectar uma política do IAM a um usuário do IAM. Neste exemplo, a política fornece ao usuário "Acesso de Usuário Power".

Antes de executar quaisquer comandos, defina suas credenciais padrão. Para obter mais informações, consulte Configurar o AWS CLI (p. 19).

Para conectar uma política gerenciada do IAM a um usuário do IAM

1. Determine o ARN da política que você deseja conectar. O comando a seguir usa list-policies para localizar o ARN da política com o nome PowerUserAccess. Em seguida, ele armazena o ARN em uma variável de ambiente:

```
$ export POLICYARN=$(aws iam list-policies --query 'Policies[?
PolicyName==`PowerUserAccess`].{ARN:Arn}' --output text)       ~
$ echo $POLICYARN
arn:aws:iam::aws:policy/PowerUserAccess
```

2. Para conectar a política, use o comando `attach-user-policy` e faça referência à variável de ambiente que contém o ARN da política:

```
$ aws iam attach-user-policy --user-name MyUser --policy-arn $POLICYARN
```

3. Verifique se a política foi anexada ao usuário executando o comando `list-attached-user-policies`.

```
$ aws iam list-attached-user-policies --user-name MyUser
{
    "AttachedPolicies": [
        {
            "PolicyName": "PowerUserAccess",
            "PolicyArn": "arn:aws:iam::aws:policy/PowerUserAccess"
        }
    ]
}
```

Recursos adicionais

Para obter mais informações, consulte Recursos de gerenciamento de acesso. Este tópico fornece links para uma visão geral de permissões e políticas, bem como links que direcionam para exemplos de políticas de acesso ao Amazon S3, Amazon EC2 e outros serviços.

Defina uma senha inicial para um usuário IAM

Este tópico descreve como usar os comandos da AWS CLI para definir uma senha inicial para um usuário do IAM.

Antes de executar quaisquer comandos, defina suas credenciais padrão. Para obter mais informações, consulte Configurar o AWS CLI (p. 19).

O comando a seguir usa create-login-profile para definir uma senha inicial para o usuário especificado. Quando o usuário faz login pela primeira vez, ele precisa alterar a senha para algo que apenas ele sabe:

```
$ aws iam create-login-profile --user-name MyUser --password My!User1Login8P@ssword --
password-reset-required
{
    "LoginProfile": {
        "UserName": "MyUser",
        "CreateDate": "2018-12-14T17:27:18Z",
        "PasswordResetRequired": true
    }
}
```

Você pode usar o comando `update-login-profile` para alterar a senha para um usuário do IAM:

```
$ aws iam update-login-profile --user-name MyUser --password My!User1ADifferentP@ssword
```

Criação de uma chave de acesso para um usuário do IAM

Este tópico descreve como usar os comandos da AWS CLI para criar um conjunto de chaves de acesso para um usuário do IAM.

Antes de executar quaisquer comandos, defina suas credenciais padrão. Para obter mais informações, consulte Configurar o AWS CLI (p. 19).

Você pode usar o comando `create-access-key` para criar uma chave de acesso para um usuário do IAM. Uma chave de acesso é um conjunto de credenciais de segurança que consiste em um ID de chave de acesso e uma chave secreta. Observe que um usuário do IAM pode ter apenas duas chaves de acesso ao mesmo tempo. Se você tentar criar um terceiro conjunto, o comando retornará um erro `LimitExceeded`.

```
$ aws iam create-access-key --user-name MyUser
{
    "AccessKey": {
        "UserName": "MyUser",
        "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "Status": "Active",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
        "CreateDate": "2018-12-14T17:34:16Z"
    }
}
```

Use o comando `delete-access-key` para excluir uma chave de acesso para um usuário do IAM. Especifique qual chave de acesso deve ser excluída usando o ID de chave de acesso.

```
$ aws iam delete-access-key --user-name MyUser --access-key-id AKIAIOSFODNN7EXAMPLE
```

Uso do Amazon S3 com a AWS Command Line Interface

Acesse os recursos do Amazon Simple Storage Service (Amazon S3) usando a AWS CLI.

Antes de executar quaisquer comandos, defina suas credenciais padrão. Para obter mais informações, consulte Configurar o AWS CLI (p. 19).

A AWS CLI oferece dois níveis de comandos para acessar o Amazon S3.

- O primeiro nível, chamado s3, consiste em comandos de alto nível que simplificam a execução de tarefas comuns, como criar, manipular e excluir objetos e buckets.
- O nível s3api se comporta de forma idêntica a outros serviços do AWS, expondo o acesso direto a todas as operações Amazon S3 da API. Isso permite executar operações avançadas que podem não ser possíveis com os comandos de alto nível do próximo nível.

Para acessar uma lista de todos os comandos disponíveis em cada nível, use o argumento `help` com os comandos `aws s3api` ou `aws s3`:

```
$ aws s3 help
```

ou

```
$ aws s3api help
```

> **Note**
>
> A AWS CLI é compatível com a cópia, transferência e sincronização do Amazon S3 para Amazon S3 usando a operação COPY do lado do servidor fornecida pelo Amazon S3. Isso significa que os arquivos são mantidos na nuvem e não são transferidos para a máquina do cliente para, em seguida, fazer backup para o Amazon S3.

Quando operações como essas podem ser executadas completamente na nuvem, somente a largura necessária de banda para a solicitação e resposta HTTP é usada.

Para exemplificar o uso do Amazon S3, consulte os tópicos a seguir.

Tópicos

- Uso dos comandos (s3) de nível superior com a AWS Command Line Interface (p. 84)
- Uso dos comandos (s3api) em nível da API com a AWS Command Line Interface (p. 88)

Uso dos comandos (s3) de nível superior com a AWS Command Line Interface

Este tópico descreve como você pode gerenciar buckets e os objetos do Amazon S3 usando comandos `aws s3` de nível superior.

Antes de executar quaisquer comandos, defina suas credenciais padrão. Para obter mais informações, consulte Configurar o AWS CLI (p. 19).

Gerenciamento de Buckets

Comandos `aws s3` de nível superior geralmente são compatíveis com as operações de bucket comuns, como criação, listagem e exclusão de buckets.

Como criar um bucket

Use o comando `s3 mb` para criar um novo bucket. Os nomes de bucket devem ser globalmente exclusivos e devem ser compatíveis com o DNS. Podem conter letras minúsculas, números, hífens e pontos. Os nomes de bucket só podem iniciar e terminar com uma letra ou número e não pode conter um ponto ao lado de um hífen ou outro ponto.

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

Listagem de buckets

Use o comando `s3 ls` para listar seus buckets. Aqui estão alguns exemplos de uso comum.

O comando a seguir lista todos os buckets.

```
$ aws s3 ls
2018-12-11 17:08:50 my-bucket
2018-12-14 14:55:44 my-bucket2
```

O comando a seguir lista todos os objetos e pastas (mencionados no S3 como 'prefixos') em um bucket.

```
$ aws s3 ls s3://bucket-name
                           PRE path/
2018-12-04 19:05:48          3 MyFile1.txt
```

A saída anterior mostra que sob o prefixo `path/` existe um arquivo chamado `MyFile1.txt`.

Você pode filtrar a saída para um prefixo específico, incluindo-o no comando. O comando a seguir relaciona os objetos no *bucket-name*/`path` (em outras palavras, objetos no *bucket-name* filtrada pelo prefixo `path/`).

```
$ aws s3 ls s3://bucket-name/path/
2018-12-06 18:59:32          3 MyFile2.txt
```

Exclusão de um bucket

Para remover um bucket, use o comando `s3 rb`.

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

Por padrão, o bucket deve estar vazio para a operação ser bem-sucedida. Para remover um bucket vazio, é necessário incluir a opção `--force`. O comando de exemplo a seguir exclui todos os objetos e subpastas no bucket e, em seguida, remove o bucket.

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

> **Note**
>
> Se você estiver usando um bucket de controle de versão, que contém objetos anteriormente excluídos— mas ainda retidos— este comando não permitirá que você remova o bucket. Você deve primeiro remover todo o conteúdo.

Gerenciamento de Objetos

Os comandos `aws s3` de alto nível também tornam o gerenciamento de objetos do Amazon S3 conveniente. Os comandos do objeto incluem `s3 cp`, `s3 ls`, `s3 mv`, `s3 rm` e `s3 sync`. Os comandos `cp`, `ls`, `mv` e `rm` funcionam da mesma forma que suas equivalentes Unix e permitem que você trabalhe perfeitamente em seus diretórios locais e buckets do Amazon S3. O comando `sync` sincroniza o conteúdo de um bucket e um diretório, ou dois buckets.

> **Note**
>
> Todos os comandos de nível superior que envolvem o upload de objetos em um bucket do Amazon S3 (`s3 cp`, `s3 mv` e `s3 sync`) executam automaticamente um multipart upload quando o objeto é grande.
> Uploads que falharam não podem ser retomados ao usar esses comandos. Se o multipart upload falhar devido a um tempo de espera ou for manualmente cancelado pressionando CTRL+C, o AWS CLI limpa quaisquer arquivos criados e aborta o upload. Esse processo pode levar alguns minutos.
> Se o processo for interrompido por um comando kill ou falha do sistema, o andamento do multipart upload permanece no Amazon S3 e deve ser limpo manualmente no Console de gerenciamento da AWS ou com o comando s3api abort-multipart-upload.

Os comandos `cp`, `mv` e `sync` incluem uma opção `--grants` que pode ser usada para conceder permissões referentes a esse objeto para usuários ou grupos específicos. Defina a opção `--grants` como uma lista de permissões usando a seguinte sintaxe:

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

Cada valor contém os seguintes elementos:

- *`Permissão`* – Especifica a concessão de permissões e pode ser configurada para `read`, `readacl`, `writeacl` ou `full`.
- *`Grantee_Type`* – Especifica como o favorecido deve ser identificado e pode ser configurado para `uri`, `emailaddress` ou `id`.

- *Grantee_ID* – Especifica o favorecido com base em *Grantee_Type*.
  - uri – O URI do grupo. Para mais informações, consulte Quem é o favorecido?
  - emailaddress – O endereço de e-mail da conta.
  - id – O ID canônico da conta.

Para obter mais informações sobre o controle de acesso do Amazon S3, consulte Controle de acesso.

O exemplo a seguir copia um objeto em um bucket. Ele concede read permissões sobre o objeto para todos os usuários e full permissões (read, readacl, e writeacl) para a conta associada com 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
```

Também é possível especificar uma classe de armazenamento não padrão (REDUCED_REDUNDANCY ou STANDARD_IA) para objetos que você carregar no Amazon S3. Para fazer isso, use a opção --storage-class:

```
$ aws s3 cp file.txt s3://my-bucket/ --storage-class REDUCED_REDUNDANCY
```

O comando s3 sync usa a seguinte sintaxe. Possíveis combinações de destino de origem são:

- Local do sistema de arquivos para o Amazon S3
- Amazon S3 para sistema de arquivos local
- Amazon S3 para Amazon S3

```
$ aws s3 sync <source> <target> [--options]
```

O exemplo a seguir sincroniza o conteúdo de uma pasta Amazon S3 chamada de caminho no my-bucket com o diretório de trabalho atual. O s3 sync atualiza todos os arquivos que têm um tamanho ou um tempo diferente em relação aos arquivos com o mesmo nome no destino. A saída exibe operações específicas executadas durante a sincronização. Observe que a operação recursivamente sincroniza o subdiretório MySubdirectory e o conteúdo com 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, s3 sync apenas copia arquivos ausentes ou desatualizadas ou objetos entre a origem e o destino. No entanto, você também pode fornecer a opção --delete para remover arquivos ou objetos do destino que não estão presentes na origem.

O exemplo a seguir, que se estende do anterior, mostra como isso funciona.

```
// 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
```

Você pode usar as opções `--exclude` e `--include` para especificar as regras que filtram os arquivos ou objetos a serem copiados durante a operação de sincronização. Por padrão, todos os itens em uma pasta especificada são incluídos na sincronização. Portanto, só `--include` é necessário ao especificar exceções para a opção `--exclude` (em outras palavras, `--include` significa efetivamente "não excluir"). As opções se aplicam na ordem especificada, como mostrado no exemplo a seguir.

```
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
```

As opções `--exclude` e `--include` também podem filtrar arquivos ou objetos para serem excluídos durante uma operação `s3 sync` que inclui a opção `--delete`. Nesse caso, a sequência de parâmetro deve especificar os arquivos a serem excluídos ou incluídos, a exclusão no contexto do diretório de destino ou bucket. Por exemplo:

```
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
```

O comando `s3 sync` também aceita uma opção `--acl`, em que você pode definir as permissões de acesso para arquivos copiados para o Amazon S3. A opção `--acl` aceita os valores `private`, `public-reade public-read-write`.

```
$ aws s3 sync . s3://my-bucket/path --acl public-read
```

Como mencionado anteriormente, o conjunto de `s3` comandos inclui `cp`, `mv`, `lse rm` e eles funcionam de maneiras semelhantes a de seus parceiros Unix. Veja os seguintes exemplos.

```
// 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
```

Quando você usa a opção `--recursive` em um diretório/pasta com `cp`, `mv`ou `rm`, o comando mostra a árvore de diretórios, incluindo todos os subdiretórios. Esses comandos também aceitam as opções `--exclude`, `--include`, e `--acl` assim como o comando `sync`.

Uso dos comandos (`s3api`) em nível da API com a AWS Command Line Interface

Os comandos em nível de API (contidos no conjunto de comandos `s3api`) fornecem acesso direto às APIs do Amazon S3 e ativam algumas operações não expostas em comandos de alto nível `s3`. Esses comandos são equivalentes dos outros serviços do AWS que fornecem acesso em nível de API para a funcionalidade de serviços.

Este tópico fornece exemplos que demonstram como usar os comandos de nível inferior que são mapeados para a API do Amazon S3. Além disso, você pode encontrar exemplos adicionais para cada API do S3 na seção s3api do Guia de Referência da CLI.

Antes de executar quaisquer comandos, defina suas credenciais padrão. Para obter mais informações, consulte Configurar o AWS CLI (p. 19).

Aplicação de uma ACL personalizada

Com comandos de alto nível, é possível usar a opção `--acl` para aplicar listas de controle de acesso (ACLs) predefinidas em objetos do Amazon S3, mas você não pode usar esse comando para definir ACLs em todo o bucket. No entanto, você pode fazer isso com o comando em nível de API put-bucket-acl. O exemplo a seguir mostra como conceder o controle total a dois usuários do AWS (user1@example.com e user2@example.com) e a permissão de leitura a todos. O identificador para "todos" vem de um URI especial que você transmite como um parâmetro.

```
$ 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 obter detalhes sobre como construir as ACLs, consulte PUT Bucket acl no Amazon Simple Storage Service API Reference. Os comandos de ACL `s3api` na CLI, como `put-bucket-acl`, usam a mesma notação de argumento abreviada.

Configuração de uma política de registro

O comando de API `put-bucket-logging` configura política e registro de buckets. No exemplo a seguir, ao usuário do AWS usuario@exemple.com é concedido controle total sobre os arquivos de log, e todos os usuários terão acesso a eles. Observe que o comando `put-bucket-acl` também deve conceder ao sistema de entrega de log do Amazon S3 (especificado por um URI) as permissões necessárias para ler e gravar os logs no bucket.

```
$ aws s3api put-bucket-acl --bucket MyBucket --grant-read-acp 'URI="http://
acs.amazonaws.com/groups/s3/LogDelivery"' --grant-write 'URI="http://acs.amazonaws.com/
groups/s3/LogDelivery"'
$ aws s3api put-bucket-logging --bucket MyBucket --bucket-logging-status file://
logging.json
```

O arquivo `logging.json` no comando anterior tem o seguinte conteúdo:

```
{
  "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"
      }
    ]
  }
}
```

Como usar a AWS Command Line Interface com o Amazon SNS

Acesse os recursos do Amazon Simple Notification Service (Amazon SNS) usando a AWS CLI. Para listar os comandos da AWS CLI do Amazon SNS, use o comando a seguir:

```
aws sns help
```

Antes de executar quaisquer comandos, defina suas credenciais padrão. Para obter mais informações, consulte Configurar o AWS CLI (p. 19).

Este tópico mostra exemplos de comandos da CLI que executam tarefas comuns para o Amazon SNS.

Tópicos

Criar um tópico

Para criar um tópico, use o comando `create-topic` e especifique o nome que você deseja atribuir ao tópico:

```
$ aws sns create-topic --name my-topic
{
    "TopicArn": "arn:aws:sns:us-west-2:123456789012:my-topic"
}
```

Anote o `TopicArn` da resposta que você usará mais tarde para publicar uma mensagem.

Inscrever-se em um tópico

Para assinar um tópico, use o comando `subscribe`. O exemplo a seguir especifica o protocolo `email` e um endereço de e-mail para o `notification-endpoint`:

```
$ aws sns subscribe --topic-arn arn:aws:sns:us-west-2:123456789012:my-topic --
protocol email --notification-endpoint saanvi@example.com
{
    "SubscriptionArn": "pending confirmation"
}
```

O AWS envia imediatamente uma mensagem de confirmação para o endereço de e-mail especificado no comando `subscribe`. A mensagem de e-mail tem o seguinte 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
```

Assim que o destinatário clicar no link Confirmar assinatura, o navegador do destinatário exibirá uma mensagem de notificação com informações semelhantes à seguinte:

```
Subscription confirmed!

You have subscribed saanvi@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-8bb22EXAMPLE

If it was not your intention to subscribe, click here to unsubscribe.
```

Publicar em um tópico

Para enviar uma mensagem para todos os assinantes de um tópico, use o comando publish. O exemplo a seguir envia a mensagem "Hello World!" para todos os assinantes do tópico especificado.

```
$ aws sns publish --topic-arn arn:aws:sns:us-west-2:123456789012:my-topic --message "Hello
 World!"
{
    "MessageId": "4e41661d-5eec-5ddf-8dab-2c867EXAMPLE"
}
```

Neste exemplo, o AWS envia uma mensagem de e-mail com o texto "Hello World!" para `saanvi@example.com`.

Cancelar inscrição de um tópico

Para cancelar a assinatura de um tópico e parar de receber as mensagens publicadas nesse tópico, use o comando unsubscribe e especifique o ARN do tópico do qual você deseja cancelar a assinatura.

```
$ aws sns unsubscribe --subscription-arn arn:aws:sns:us-west-2:123456789012:my-
topic:1328f057-de93-4c15-512e-8bb22EXAMPLE
```

Para verificar se o registro foi cancelado com êxito, use o comando list-subscriptions para confirmar que o ARN não aparece mais na lista:

```
$ aws sns list-subscriptions
```

Excluir um tópico

Para excluir um tópico, execute o comando delete-topic-rule:

```
$ aws sns delete-topic --topic-arn arn:aws:sns:us-west-2:123456789012:my-topic
```

Para verificar se o AWS excluiu o tópico com êxito, use o comando list-topics para confirmar que o tópico não aparece mais na lista:

```
$ aws sns list-topics
```

Uso do Amazon Simple Workflow Service com a AWS Command Line Interface

Acesse os recursos do Amazon SWF usando a AWS CLI. Para listar os comandos da AWS CLI do Amazon SWF use o comando a seguir:

```
aws swf help
```

Antes de executar quaisquer comandos, defina suas credenciais padrão. Para obter mais informações, consulte Configurar o AWS CLI (p. 19).

Este tópico mostra exemplos de comandos da CLI que executam tarefas comuns para o Amazon SWF.

Tópicos

Lista de Comandos por Categoria Amazon SWF

Você pode usar a AWS CLI para criar, exibir e gerenciar fluxos de trabalho no Amazon SWF.

Esta seção lista os tópicos de referência para os comandos Amazon SWF da AWS CLI, agrupados por categoria funcional.

Para acessar uma lista de comandos em ordem alfabética, consulte a seção do Amazon SWF do AWS CLI Command Reference ou use o comando a seguir:

```
$ aws swf help
```

Você também pode obter ajuda para um comando individual, colocando a diretiva `help` após o nome do comando. Por exemplo:

```
$ aws swf register-domain help
```

Tópicos

Atividades Relacionadas a Comandos

Operadores de atividade usam `poll-for-activity-task` para receber novas tarefas de atividade. Depois que um operador recebe uma tarefa de atividade do Amazon SWF, ele realiza essa tarefa e responde usando `respond-activity-task-completed` ser ela for bem-sucedida ou `respond-activity-task-failed` se for malsucedida.

Os comandos a seguir são realizados por operadores de atividade.

- poll-for-activity-task
- respond-activity-task-completed
- respond-activity-task-failed
- respond-activity-task-canceled
- record-activity-task-heartbeat

Comandos relacionados a administradores

Agentes de decisão usam `poll-for-decision-task` para obter tarefas de decisão. Depois que um administrador recebe uma tarefa de administração do Amazon SWF, ele examina seu histórico de execuções de fluxo de trabalho e decide o que fazer a seguir. Ele chama `respond-decision-task-completed` para concluir a tarefa de decisão e fornece zero ou mais próxima decisões.

Os comandos a seguir são realizadas por administradores.

- poll-for-decision-task
- respond-decision-task-completed

Comandos relacionados a execuções de fluxo de trabalho

Os comandos a seguir operam em uma execução de fluxo de trabalho.

- request-cancel-workflow-execution
- start-workflow-execution
- signal-workflow-execution
- terminate-workflow-execution

Comandos relacionados à administração

Embora seja possível executar tarefas administrativas do console Amazon SWF, é possível usar os comandos nesta seção para automatizar funções ou criar suas próprias ferramentas administrativas.

Gerenciamento de atividade

- register-activity-type
- deprecate-activity-type

Gerenciamento de fluxo de trabalho

- register-workflow-type
- deprecate-workflow-type

Gerenciamento de domínio

- register-domain
- deprecate-domain

Para mais informações e exemplos desses comandos de gerenciamento de domínio, consulte Como trabalhar com domínios Amazon SWF usando a AWS Command Line Interface (p. 94).

Gerenciamento de execução de fluxo de trabalho

- request-cancel-workflow-execution
- terminate-workflow-execution

Comandos de visibilidade

Embora seja possível executar ações de visibilidade do console Amazon SWF, é possível usar os comandos nesta seção para automatizar funções ou criar seu próprio console e suas próprias ferramentas administrativas.

Visibilidade de Atividade

- list-activity-types
- describe-activity-type

Visibilidade do fluxo de trabalho

- list-workflow-types
- describe-workflow-type

Visibilidade de execução de fluxo de trabalho

- describe-workflow-execution
- list-open-workflow-executions
- list-closed-workflow-executions
- count-open-workflow-executions
- count-closed-workflow-executions
- get-workflow-execution-history

Visibilidade de domínio

- list-domains
- describe-domain

Para mais informações e exemplos desses comandos de visibilidade de domínio, consulte Como trabalhar com domínios Amazon SWF usando a AWS Command Line Interface (p. 94).

Visibilidade de lista de tarefas

- count-pending-activity-tasks
- count-pending-decision-tasks

Como trabalhar com domínios Amazon SWF usando a AWS Command Line Interface

Também é possível usar a AWS CLI para gerenciar domínios do SWF.

Tópicos

Listagem dos domínios

Para listar os domínios do Amazon SWF registrados para a sua conta da AWS, use `swf list-domains`. Você deve incluir `--registration-status` e especificar `REGISTERED` ou `DEPRECATED`.

Aqui está um pequeno exemplo:

```
$ aws swf list-domains --registration-status REGISTERED
{
```

```
    "domainInfos": [
        {
            "status": "REGISTERED",
            "name": "ExampleDomain"
        },
        {
            "status": "REGISTERED",
            "name": "mytest"
        }
    ]
}
```

Note

Para ver um exemplo de uso `DEPRECATED`, consulte Reprovar um domínio (p. 96).

Obtenção de informações sobre um domínio

Para acessar informações detalhadas sobre um domínio específico, use `swf describe-domain`. Há um parâmetro obrigatório: `--name`, que leva o nome do domínio sobe o qual você deseja informações. Por exemplo:

```
$ aws swf describe-domain --name ExampleDomain
{
    "domainInfo": {
        "status": "REGISTERED",
        "name": "ExampleDomain"
    },
    "configuration": {
        "workflowExecutionRetentionPeriodInDays": "1"
    }
}
```

Registrar um domínio

Para registrar novos domínios, use `swf register-domain`. Há dois parâmetros obrigatórios: `--name`, que usa o nome do domínio para registrar, e `--workflow-execution-retention-period-in-days`, que contém um número inteiro para especificar o número de dias para retenção dos dados de execução do fluxo de trabalho nesse domínio, em um período de no máximo 90 dias (para obter mais informações, consulte as perguntas frequentes do Amazon SWF). Se especificar zero (0) para esse valor, o período de retenção é definido automaticamente na duração máxima. Caso contrário, os dados de execução do fluxo de trabalho não serão retidos após o número especificado de dias. A seguir, o exemplo mostra como registrar um novo domínio:

```
$ aws swf register-domain --name MyNeatNewDomain --workflow-execution-retention-period-in-
days 0
```

O comando não retorna nenhuma saída, mas você pode usar `swf list-domains` ou `swf describe-domain` para ver o novo domínio. Por exemplo:

```
$ aws swf describe-domain --name MyNeatNewDomain
{
    "domainInfo": {
        "status": "REGISTERED",
        "name": "MyNeatNewDomain"
    },
    "configuration": {
        "workflowExecutionRetentionPeriodInDays": "0"
    }
```

```
}
```

Reprovar um domínio

Para reprovar um domínio (você ainda pode ver, mas não pode criar novas execuções de fluxo de trabalho ou registrar tipos), use `swf deprecate-domain`. Ele tem um único parâmetro obrigatório, `--name`, que causa a reprovação do nome do domínio.

```
$ aws swf deprecate-domain --name MyNeatNewDomain
```

Assim como ocorre com `register-domain`, nenhuma saída é retornada. Se você usar `list-domains` para visualizar os domínios registrados, no entanto, você verá que o domínio não aparece mais entre eles. Você também pode usar `--registration-status DEPRECATED`.

```
$ aws swf list-domains --registration-status DEPRECATED
{
    "domainInfos": [
        {
            "status": "DEPRECATED",
            "name": "MyNeatNewDomain"
        }
    ]
}
```

Consulte também

- deprecate-domain no AWS CLI Command Reference
- describe-domain no AWS CLI Command Reference
- list-domains no AWS CLI Command Reference
- register-domain no AWS CLI Command Reference

Solução de problemas de erros do AWS CLI

Problemas de instalação

Se você usar o `pip` para instalar o AWS CLI, talvez você precise adicionar a pasta que contém o programa `aws` à variável de ambiente `PATH` do sistema operacional ou alterar seu modo para fazer com que se torne executável.

Erro: aws: comando não encontrado

Talvez seja necessário adicionar o arquivo executável `aws` à variável de ambiente `PATH` do SO.

- Windows – Adicione o executável AWS CLI em seu caminho de linha de comando (p. 12)
- macOS – Adicione o executável AWS CLI em seu caminho de linha de comando (p. 14)
- Linux – Adicione o executável AWS CLI em seu caminho de linha de comando (p. 7)

Se `aws` estiver no seu `PATH` e você ainda vir este erro, talvez não tenha o modo de arquivo correto. Tente executá-lo diretamente.

```
$ ~/.local/bin/aws --version
```

Problemas de permissão

O programa CLI principal deve ter permissão de execução

Erro: permissão negada

Certifique-se de que o programa `aws` tem permissões de execução para o usuário de chamada. Normalmente, você pode usar `755`.

Execute `chmod +x` para adicionar permissões ao arquivo.

```
$ chmod +x ~/.local/bin/aws
```

Você deve usar credenciais válidas

Erro: a AWS não foi capaz de validar as credenciais fornecidas

A AWS CLI pode estar lendo credenciais a partir de um local diferente do que o esperado. Execute a `aws configure list` para confirmar quais credenciais são usadas. O exemplo a seguir mostra como verificar as credenciais usadas para o perfil padrão:

```
$ 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
```

O exemplo a seguir mostra como verificar as credenciais de um perfil nomeado:

```
$ aws configure list --profile saanvi
      Name                    Value             Type    Location
      ----                    -----             ----    --------
   profile                    saanvi           manual    --profile
access_key         **************** shared-credentials-file
secret_key         **************** shared-credentials-file
    region                us-west-2      config-file    ~/.aws/config
```

Se você estiver usando credenciais válidas, seu relógio poderá estar fora de sincronia. No Linux, macOS, or Unix, execute `date` para verificar a hora.

```
$ date
```

Se o relógio do sistema não estiver correto dentro de alguns minutos, use `ntpd` para sincronizar.

```
$ sudo service ntpd stop
$ sudo ntpdate time.nist.gov
$ sudo service ntpd start
$ ntpstat
```

No Windows, use as opções de data e hora no painel de controle para configurar o relógio do sistema.

O usuário do IAM deve ser capaz de executar o comando

Erro: ocorreu um erro (UnauthorizedOperation) ao chamar a operação *`CreateKeyPair`*: Você não está autorizado a executar esta operação.

Sua função ou usuário do IAM devem ter permissão para chamar as ações de API que correspondem aos comandos que você executa com a AWS CLI. A maioria dos comandos chama uma única ação com um nome que corresponde ao nome do comando; no entanto, comandos personalizados, como `aws s3 sync`, chamam várias APIs. Você pode ver quais APIs um comando chama usando a opção `--debug`.

Para obter mais informações sobre como atribuir permissões às funções e aos usuários do IAM, consulte Visão geral do gerenciamento de acesso: permissões e políticas no Guia do usuário do IAM.