ユーザーガイド

AWS CloudShell

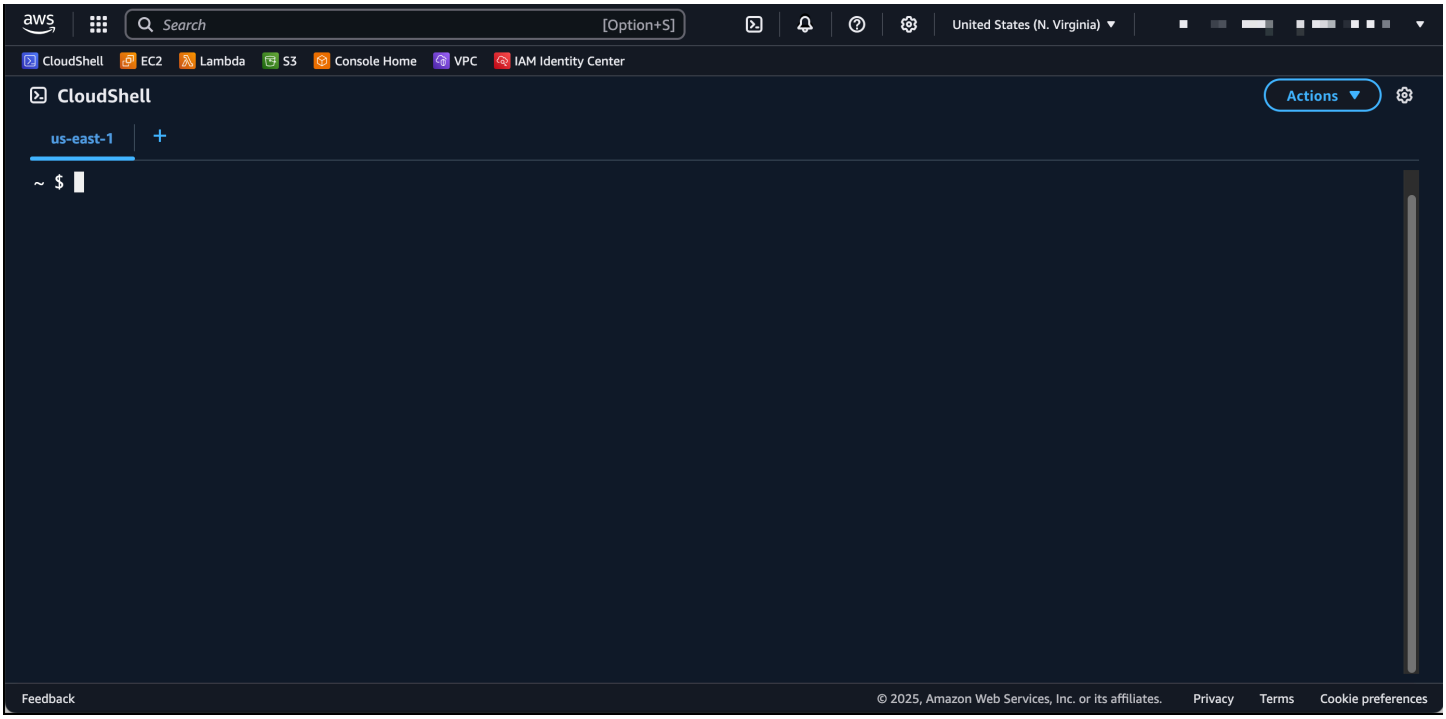
AWS CloudShell: ユーザーガイド

Table of Contents

とは AWS CloudShell

AWS CloudShell はブラウザベースの事前認証済みシェルで、 から直接起動できます AWS マネジメントコンソール。いくつかの異なる方法から CloudShell AWS マネジメントコンソール に移動できます。詳細については、「 の開始方法」を参照してください。 AWS CloudShell

コマンドは、、PowerShellBash、 などの任意のシェル AWS CLI を使用して実行できますZ shell。またこの手順は、コマンドラインツールのダウンロードもインストールも不要です。



を起動すると AWS CloudShell、Amazon Linux 2023 に基づくコンピューティング環境が作成されます。この環境内では、広範なプリインストールされた開発ツール、ファイルのアップロードおよびダウンロードのオプション、およびセッション間で保持されるファイルストレージにアクセスできます。CloudShell は、Google Chrome、Mozilla Firefox、Microsoft Edge、Apple Safari ブラウザの最新バージョンで使用できます。

(今すぐ試す: の開始方法 AWS CloudShell)

の機能 AWS CloudShell

AWS CloudShell には次の機能があります。

AWS Command Line Interface

AWS CloudShell から を起動できます AWS マネジメントコンソール。コンソールへのサインインに使用した AWS 認証情報は、新しいシェルセッションで自動的に使用できます。 AWS CloudShell ユーザーは事前認証されているため、 AWS CLI バージョン 2 を使用して を操作する AWS のサービス ときに認証情報を設定する必要はありません。 AWS CLI はシェルのコンピューティング環境にプリインストールされています。

コマンドラインインターフェイス AWS のサービス を使用した の操作の詳細については、「」を参照してくださいCloudShell で CLI から AWS サービスを管理する。

シェルおよび開発ツール

AWS CloudShell セッション用に作成されたシェルを使用すると、任意のコマンドラインシェルをシームレスに切り替えることができます。具体的には、Bash、PowerShell、Z shell 間で切り替えることができます。プリイントールされたツールとユーティリティにもアクセスできます。例として、git 、 make 、 pip 、 sudo 、 tar 、 tmux 、 vim 、 wget 、 zip などがあります。

シェル環境は、Node.js や Python のような主要なソフトウェア言語をサポートするように事前設定されています。これは、例えば、最初にランタイムのインストールを実行しなくても Node.js や Python プロジェクトを実行できるということです。PowerShell ユーザーは、.NET Core ランタイムを使用できます。

詳細については、「AWS CloudShell コンピューティング環境: 仕様とソフトウェア」を参照してください。

永続ストレージ

を使用すると AWS CloudShell、追加料金 AWS リージョン なしで各 で最大 1 GB の永続的ストレージを使用できます。永続的ストレージはホームディレクトリ (`$HOME`) にあり、ユーザーのプライベートな記憶域です。各シェルセッションが終了した後にリサイクルされるエフェメラル環境リソースとは異なり、ホームディレクトリ内のデータはセッション間で保持されます。

永続的ストレージでのデータの保持の詳細については、「永続ストレージ」を参照してください。

> **Note**
>
> CloudShell の VPC 環境には永続ストレージはありません。$HOME ディレクトリは、VPC 環境がタイムアウトするか (20 ~ 30 分間非アクティブ状態が続いた後)、環境を削除または再起動すると、削除されます。

CloudShell の VPC 環境

AWS CloudShell Virtual Private Cloud (VPC) を使用すると、VPC に CloudShell 環境を作成できます。VPC 環境ごとに、VPC を割り当て、サブネットを追加し、1 つ以上のセキュリティグループを関連付けることができます。 AWS CloudShell は VPC のネットワーク設定を継承し、VPC 内の他のリソースと同じサブネット内で AWS CloudShell を安全に使用できます。

セキュリティ

AWS CloudShell 環境とそのユーザーは、特定のセキュリティ機能によって保護されています。これには、IAM アクセス許可管理、シェルセッション制限、テキスト入力用の安全な貼り付けなどの機能が含まれます。

IAM を使用したアクセス許可管理

管理者は、IAM ポリシーを使用して AWS CloudShell ユーザーにアクセス許可を付与および拒否できます。また、ユーザーがシェル環境で実行できる特定のアクションを指定するポリシーを作成することもできます。詳細については、「IAM ポリシーによる AWS CloudShell アクセスと使用状況の管理」を参照してください。

シェルセッション管理

非アクティブなセッションと長時間実行されているセッションは自動的に停止され、リサイクルされます。詳細については、「シェルセッション」を参照してください。

テキスト入力用の安全な貼り付け

デフォルトでは、安全な貼り付けが有効になっています。このセキュリティ機能では、シェルに貼り付けようとしている複数行のテキストに悪意のあるスクリプトが含まれていないことを確認する必要があります。詳細については、「マルチテキストに安全な貼り付けを使用する」を参照してください。

カスタマイズオプション

AWS CloudShell エクスペリエンスは、好みに合わせてカスタマイズできます。例えば、画面レイアウト (複数タブ) や表示テキストサイズ、明暗インターフェースのテーマの切り替えの変更が可能です。詳細については、「AWS CloudShell エクスペリエンスのカスタマイズ」を参照してください。

独自のソフトウェアをインストールしてスクリプトでシェルを変更すれば、シェル環境を拡張することもできます。

セッションの復元

セッションの復元機能は、CloudShell ターミナルの 1 つまたは複数のブラウザタブで実行していたセッションを復元します。最近閉じたブラウザタブを更新または再度開くと、非アクティブなセッションが原因でシェルが停止するまで、この機能によりセッションを回復します。CloudShell セッションを引き続き使用するには、ターミナルウィンドウ内の任意のキーを押します。シェルセッションの詳細については、「シェルセッション」を参照してください。

セッションの復元では、最新のターミナル出力と各ターミナルタブ内の実行中のプロセスも復元されます。

> **Note**
>
> セッションの復元はモバイルアプリケーションでは利用できません。

の料金 AWS CloudShell

AWS CloudShell は、追加料金なしで利用できる AWS のサービス です。ただし、 で実行する他の AWS リソースに対しては料金が発生します AWS CloudShell。さらに、スタンダードデータ転送料金も適用されます。詳細については、「AWS CloudShell 料金表」を参照してください。

詳細については、「のサービスクォータと制限 AWS CloudShell」を参照してください。

主要な AWS CloudShell トピック

- の開始方法 AWS CloudShell
- AWS CloudShell 概念
- CloudShell で CLI から AWS サービスを管理する

- AWS CloudShell エクスペリエンスのカスタマイズ
- AWS CloudShell コンピューティング環境: 仕様とソフトウェア

の開始方法 AWS CloudShell

この入門チュートリアルでは、シェルコマンドラインインターフェイスを使用して主要なタスクを起動 AWS CloudShell および実行する方法を示します。

まず、 にサインイン AWS マネジメントコンソール し、 を選択します AWS リージョン。新しいブラウザで、CloudShell と使用するシェルタイプを起動します。

次に、ホームディレクトリに新しいフォルダを作成し、ローカルマシンからフォルダにファイルをアップロードします。コマンドラインからプログラムとして実行する前に、プリインストールされたエディタを使用してそのファイルを操作します。最後に、 AWS CLI コマンドを呼び出して Amazon S3 バケットを作成し、ファイルをオブジェクトとしてバケットに追加します。

前提条件

IAM アクセス許可

のアクセス許可を取得するには、次の AWS 管理ポリシーを IAM ID (ユーザー、ロール、グループなど) にア AWS CloudShell タッチします。

- AWSCloudShellFullAccess: ユーザーに AWS CloudShell とその機能へのフルアクセスを提供します。

このチュートリアルでは、 も操作します AWS のサービス。具体的には、S3 バケットを作成して、そのバケットにオブジェクトを追加して、Amazon S3 を操作します。IAM アイデンティティには、最低限、`s3:CreateBucket` および `s3:PutObject` アクセス許可を付与するポリシーが必要です。

詳細については、Amazon Simple Storage Service ユーザーガイドのAmazon S3 Action を参照してください。

演習ファイル

この演習では、コマンドラインインターフェイスからプログラムとして実行されるファイルをアップロードして編集することが含まれます。ローカルマシンでテキストエディタを開き、次のコードスニペットを追加します。

```
import sys
x=int(sys.argv[1])
```

```
y=int(sys.argv[2])
sum=x+y
print("The sum is",sum)
```

次に、add_prog.py という名前でファイルを保存します。

内容

ステップ 1: にサインインする AWS マネジメントコンソール

このステップでは、IAM ユーザー情報を入力して にアクセスします AWS マネジメントコンソール。コンソールにすでに入っている場合は、ステップ 2に進みます。

- にアクセスするには、IAM ユーザーのサインイン URL AWS マネジメントコンソール を使用するか、メインのサインインページに移動します。

  IAM user sign-in URL

  - ブラウザを開き、次のサインイン URL を入力します。管理者にもらったアカウントエイリアスもしくはアカウント ID を account_alias_or_id と置き換えます。

    ```
    https://account_alias_or_id.signin.aws.amazon.com/console/
    ```

  - IAM サインイン認証情報を入力し、 [サインイン] を選択します。

Main sign-in page

- https://aws.amazon.com/console/ を開きます。
- このブラウザを使用して以前にサインインしなかった場合は、メインのサインインページが表示されます。IAM ユーザー を選択し、アカウントエイリアスもしくはアカウント ID を入力して、[次へ] を選択します。
- 以前にすでに IAM ユーザーとしてサインインしている場合。ブラウザには、 AWS アカウントのアカウントエイリアスもしくはアカウント ID が記憶されている可能性があります。その場合、IAM サインイン認証情報を入力し [サインイン] を選択します。

> **Note**
>
> ルートユーザーとしてサインインすることもできます。この ID は、アカウント内のすべての AWS のサービス および リソースに完全にアクセスできます。日常的なタスクには (それが管理タスクであっても)、ルートユーザーを使用しないよう強くお勧めします。代わりに、初期の IAM ユーザーを作成するためにのみ、ルートユーザーを使用するというベストプラクティスに従います。

ステップ 2: リージョンの選択、起動 AWS CloudShell、シェルの選択

このステップでは、コンソールインターフェイスから CloudShell を起動し、使用可能な を選択し AWS リージョン、Bash、PowerShell、 などの任意のシェルに切り替えますZ shell。

1. 作業 AWS リージョン する を選択するには、「リージョンの選択」メニューに移動し、サポートされている AWS リージョンを選択して作業します。(使用可能なリージョンがハイライト表示されます)。

> **Important**
>
> リージョンを切り替えると、インターフェースが再読込みされ、選択した AWS リージョン の名前がコマンドラインテキストの上に表示されます。永続的ストレージに追加

したファイルは、同じ AWS リージョン内のみにて使用できます。リージョンを変更すると、異なるストレージおよびファイルにアクセスできます。

> **Important**
>
> Console Toolbar にある CloudShell を起動したときに、選択したリージョンで CloudShell が使用できない場合、デフォルトのリージョンは選択したリージョンに最も近いリージョンに設定されます。デフォルトのリージョンとは別のリージョンのリソースを管理する許可を付与するコマンドを実行できます。詳細については、「Working in AWS リージョン」を参照してください。
>
> Example
>
> 例
>
> 欧州 (スペイン) eu-south-2 を選択したのに CloudShell が欧州 (スペイン) eu-south-2 で利用できない場合、デフォルトのリージョンは欧州 (スペイン) eu-south-2 に最も近い欧州 (アイルランド) eu-west-1 に設定されます。
>
> デフォルトのリージョンである欧州 (アイルランド) eu-west-1 の Service Quotas を使用すると、同じ CloudShell セッションがすべてのリージョンで復元されます。デフォルトのリージョンは変更される可能性があり、CloudShell ブラウザウィンドウで通知されます。

2. から AWS マネジメントコンソール、次のいずれかのオプションを選択して CloudShell を起動できます。

   1. ナビゲーションバーで、CloudShell アイコンを選択します。
   2. [検索] ボックスに 「CloudShell」 と入力し、 [CloudShell] を選択します。
   3. 最近アクセスしたウィジェットで、[CloudShell] を選択します。
   4. コンソールの左下の Console Toolbar にある [CloudShell] を選択します。
      - = をドラッグすることで CloudShell セッションの高さを調整できます。
      - [新しいブラウザタブで開く] をクリックすることにより、CloudShell セッションを全画面に切り替えることができます。

   コマンドプロンプトが表示されたら、シェルは対話的な操作の準備ができています。

> **Note**
>
> が正常に起動または操作できない問題が発生した場合は AWS CloudShell、 でそれらの問題を特定して対処するための情報を確認してくださいトラブルシューティング AWS CloudShell。

3. 作業に使用するプリインストールシェルを選択するには、コマンドラインプロンプトでプログラム名を入力します。

   Bash

   ```
   bash
   ```

   Bash に切り替えると、コマンドプロンプトの記号が $ に更新します。

   > **Note**
   >
   > Bash は、起動時に実行されるデフォルトのシェルです AWS CloudShell。

   PowerShell

   ```
   pwsh
   ```

   PowerShell に切り替えると、コマンドプロンプトの記号が更新されて PS> になります。

   Z shell

   ```
   zsh
   ```

   Z shell に切り替えると、コマンドプロンプトの記号が % に更新します。

   シェル環境にプリインストールされているバージョンの詳細については、「AWS CloudShell コンピューティング環境」セクションの「シェルの表」を参照してください。

ステップ 3: からファイルをダウンロードする AWS CloudShell

> **Note**
>
> このオプションは、VPC 環境では使用できません。

このステップでは、ファイルのダウンロード手順について説明します。

1. ファイルをダウンロードするには、[アクション] に移動し、メニューから [ファイルのダウンロード] を選択します。

   [ファイルのダウンロード] ダイアログボックスが表示されます。
2. [ファイルのダウンロード] ダイアログボックスでダウンロードするファイルのパスを入力します。

   > **Note**
   >
   > ダウンロードするファイルを指定するときは、絶対パスもしくは相対パスを使用できます。相対パス名で指定すると、`/home/cloudshell-user/` がデフォルトで自動的にスタートに追加されます。`mydownload-file` というファイルをダウンロードしようとする場合、次のどちらも有効なパスです。
   >
   > - 絶対パス: `/home/cloudshell-user/subfolder/mydownloadfile.txt`
   > - 相対パス: `subfolder/mydownloadfile.txt`
3. [ダウンロード] を選択します。

   ファイルパスが正しい場合は、ダイアログボックスが表示されます。このダイアログボックスを使用して、デフォルトでのアプリケーションでファイルを開くことができます。または、ファイルをローカルマシン上のフォルダに保存することもできます。

> **Note**
>
> Console Toolbar で CloudShell を起動する場合、[ダウンロード] オプションは使用できません。CloudShell コンソールから、または Chrome ウェブブラウザを使用してファイルをダウンロードすることができます。

ステップ 4: にファイルをアップロードする AWS CloudShell

> **Note**
>
> このオプションは、VPC 環境では使用できません。

このステップでは、ファイルをアップロードし、ホームディレクトリ内の新しいディレクトリに移動させる方法を説明します。

1. 現在の作業ディレクトリをチェックするには、プロンプトで次のコマンドを入力します。

   ```
   pwd
   ```

   Enter を押すと、シェルは現在の作業ディレクトリ (例えば、`/home/cloudshell-user` など) を戻します。
2. ファイルをこのディレクトリにアップロードするには、[アクション] に移動し、メニューから [ファイルのアップロード] を選択します。

   [ファイルのアップロード] ダイアログボックスが表示されます。
3. Browse (参照) を選択します。
4. システムの[ファイルのアップロード] ダイアログボックスで、このチュートリアル (`add_prog.py`) 用に作成したテキストファイルを選択し、[オープン] を選びます。
5. [ファイルのアップロード] ダイアログボックスで、[アップロード] を選択します。

   プログレスバーはアップロードを追跡します。アップロードが成功すると、`add_prog.py`がホームディレクトリのルートに追加されたというメッセージがチェックされます。
6. ファイルのディレクトリを作成するには、ディレクトリ作成コマンドを入力します: `mkdir mysub_dir`
7. アップロードしたファイルをホームディレクトリのルートから新しいディレクトリに移動するには、`mv` コマンドを使用します。

   ```
   mv add_prog.py mysub_dir.
   ```
8. 作業ディレクトリを新しいディレクトリに変更するには、`cd mysub_dir`を入力します。

   コマンドプロンプトがアップロードされ、作業ディレクトリが変更されたことを示します。
9. 現在のディレクトリ `mysub_dir` の内容を表示するには、`ls` コマンドを入力します。

作業ディレクトリの内容が一覧表示されます。ここには、アップロードしたばかりのファイルも含まれます。

ステップ 5: からファイルを削除する AWS CloudShell

このステップでは、 からファイルを削除する方法について説明します AWS CloudShell。

1. からファイルを削除するには AWS CloudShell、 `rm` (削除) などの標準シェルコマンドを使用します。

   ```
   rm my-file-for-removal
   ```

2. 指定した条件を満たす複数のファイルを削除するには、`find` コマンドを実行します。

   次の例では、名前に「.pdf」という接頭辞が含まれるすべてのファイルを削除します。

   ```
   find -type f -name '*.pdf' -delete
   ```

> **Note**
>
> 特定の AWS CloudShell で の使用を停止するとします AWS リージョン。そのリージョンにある永続的ストレージ内のデータは、指定された期間を過ぎると自動的に削除されます。詳細については、「永続的ストレージ」を参照してください。

ステップ 6 : ホームディレクトリのバックアップを作成する

このステップでは、ホームディレクトリのバックアップを作成する方法について説明します。

1. バックアップファイルの作成

   ホームディレクトリの外部に一時フォルダを作成します。

   ```
   HOME_BACKUP_DIR=$(mktemp --directory)
   ```

   次のいずれかのオプションを使用して、バックアップを作成できます。

   a. tar を使用したバックアップファイルの作成

tar を使用したバックファイルの作成には、次のコマンドを入力します。

```
tar \
    --create \
    --gzip \
    --verbose \
    --file=${HOME_BACKUP_DIR}/home.tar.gz \
    [--exclude ${HOME}/.cache] \ // Optional
    ${HOME}/
echo "Home directory backed up to this file: ${HOME_BACKUP_DIR}/home.tar.gz"
```

b. zip を使用したバックアップファイルの作成

zip を使用したバックファイルの作成には、次のコマンドを入力します。

```
zip \
    --recurse-paths \
    ${HOME_BACKUP_DIR}/home.zip \
    ${HOME} \
    [--exclude ${HOME}/.cache/\*] // Optional
echo "Home directory backed up to this file: ${HOME_BACKUP_DIR}/home.zip"
```

2. CloudShell の外部へのバックアップファイルの転送

次のいずれかのオプションを使用して、バックアップファイルを CloudShell の外部に転送できます。

a. バックアップファイルをローカルマシンにダウンロード

前のステップで作成したファイルをダウンロードすることができます。CloudShell からファイルをダウンロードする方法の詳細については、「AWS CloudShellからファイルをダウンロードする」を参照してください。

ファイルのダウンロードダイアログボックス内で、ダウンロードするファイルのパス (例えば、/tmp/tmp.iA99tD9L98/home.tar.gz など) を入力します。

b. バックアップファイルを S3 に転送する

バケットを生成するには、次のコマンドを入力します。

```
aws s3 mb s3://${BUCKET_NAME}
```

AWS CLI を使用してファイルを S3 バケットにコピーします。

```
aws s3 cp ${HOME_BACKUP_DIR}/home.tar.gz s3://${BUCKET_NAME}
```

> **Note**
>
> データ転送料金が適用される場合があります。

3. 直接 S3 バケットにバックアップする

   直接 S3 バケットにバックアップを行うには、次のコマンドを入力します。

```
aws s3 cp \
    ${HOME}/ \
    s3://${BUCKET_NAME} \
    --recursive \
    [--exclude .cache/\*] // Optional
```

ステップ 7 : シェルセッションを再開する

このステップでは、シェルセッションを再開する方法について説明します。

> **Note**
>
> セキュリティ対策として、長時間キーボードもしくはポインタを使用してシェルと対話しないと、セッションは自動的に停止します。長時間実行されているセッションも自動的に停止します。詳細については、「シェルセッション」を参照してください。

1. シェルセッションを再開するには、[アクション]、[再開] を選択します。

   再起動すると、現在の のすべてのアクティブなセッションが AWS CloudShell 停止することが通知されます AWS リージョン。

2. 確認するには、[再開]を選択します。

CloudShell コンピューティング環境が停止しているというメッセージがインターフェースに表示されます。環境が停止して再開したら、新しいセッションでコマンドラインの操作を開始できます。

> **Note**
>
> 場合によっては、環境を再起動するまで数分かかる場合があります。

ステップ 8 : シェルセッションのホームディレクトリを削除する

このステップでは、シェルセッションを削除する方法について説明します。

> **Note**
>
> このオプションは、VPC 環境では使用できません。VPC 環境を再起動すると、ホームディレクトリは削除されます。

> **Warning**
>
> ホームディレクトリを削除することは、ホームディレクトリに保存されているすべてのデータが完全に削除されるという不可逆的なアクションです。ただし、次のような場合には、このオプションを考慮してもよいでしょう。
>
> - ファイルが正しく変更されていないため、 AWS CloudShell コンピューティング環境にアクセスできません。ホームディレクトリを削除すると AWS CloudShell 、デフォルト設定に戻ります。
> - からすべてのデータを AWS CloudShell すぐに削除します。 AWS リージョン AWS CloudShell で の使用を停止すると、リージョンで AWS CloudShell 再度起動しない限り、永続ストレージは保持期間の終了時に自動的に削除されます。
>
> ファイルを長期的に保存する必要がある場合は、Amazon S3 などのサービスを検討してください。

1. シェルセッションを削除するには、[アクション]、[削除] の順に選択します。

AWS CloudShell ホームディレクトリを削除すると、 AWS CloudShell 環境に現在保存されているすべてのデータが削除されます。

> **Note**
>
> このアクションは元に戻すことができません。

2. 削除されたことを確認するには、テキスト入力フィールドに削除と入力した上で、[削除] を選択します。

   AWS CloudShell は、現在の AWS リージョンでアクティブになっているすべてのセッションを停止します。新しい環境を作成するか、CloudShell の VPC 環境をセットアップできます。

3. 新しい環境を作成するには、[タブを開く] を選択します。
4. CloudShell の VPC 環境を作成するには、[VPC 環境を作成] を選択します。

   シェルセッションを手動で終了する

   コマンドラインで、コマンドを使用してexitコシェルセッションを終了し、シェルセッションを終了し、ログアウトができます。次いで、任意のキーを押して再接続すれば、引き続き AWS CloudShellを使用できます。

ステップ 9 : ファイルのコードを編集し、コマンドラインを使用して実行する

このステップでは、プリインストールされた Vim エディタを使用してファイルを操作する方法を説明します。その後、コマンドラインからそのファイルをプログラムとして実行します。

1. 前のステップでアップロードしたファイルを編集するには、次のコマンドを入力します。

   ```
   vim add_prog.py
   ```

   シェルインターフェースがアップロードされ、Vim エディタが表示されます。

2. Vim でファイルを編集するには、I キーを押します。次に、プログラムが 2 つではなく 3 つの数字を加算するように内容を編集します。

   ```
   import sys
   x=int(sys.argv[1])
   ```

```
y=int(sys.argv[2])
z=int(sys.argv[3])
sum=x+y+z
print("The sum is",sum)
```

> **Note**
>
> テキストをエディタに貼り付けて、安全な貼り付け機能を有効にすると、警告が表示されます。コピーされたマルチテキストには、悪意のあるスクリプトが含まれている可能性があります。安全な貼り付け機能を使用すると、貼り付け前にテキスト全体が検証できます。テキストが安全であることが満足したら、Paste (貼り付ける)を選択します。

3. プログラムを編集したら、Esc をクリックして Vim コマンドモードに入力します。次に、 `:wq` コマンドを入力してファイルを保存し、エディタを終了します。

> **Note**
>
> Vim コマンドモードを初めて使用する場合、はじめはコマンドモードと挿入モードの切り替えが難しいと感じるかもしれません。コマンドモードは、ファイルを保存してアプリケーションを終了するときに使用されます。挿入モードは、新しいテキストを挿入するときに使用されます。挿入モードと入力するには、 I を押し、コマンドモードと入力して、 Esc を押します。Vim および で使用できるその他のツールの詳細については AWS CloudShell、「」を参照してください開発ツールおよびシェルユーティリティ。

4. メインコマンドラインインターフェースで、次のプログラムを実行し、入力用に次の 3 つの数値を指定します。構文は次のとおりです。

   ```
   python3 add_prog.py 4 5 6
   ```

   コマンドラインにプログラムの出力が表示されます: `The sum is 15`

ステップ 10: AWS CLI を使用して Amazon S3 バケットのオブジェクトとしてファイルを追加する

このステップでは、Amazon S3 バケットを作成し、PutObject メソッドを使用して、コードファイルをオブジェクトとしてバケットに追加します。

> **Note**
>
> このチュートリアルでは、 AWS CLI で AWS CloudShell を使用して他の AWS サービスとやり取りする方法を示します。この方法を使用すれば、追加のリソースをダウンロードもしくはインストールする必要はありません。さらに、ユーザーはシェル内で既に認証されているので、呼び出しを行う前に認証情報を設定する必要はありません。

1. 指定された にバケットを作成するには AWS リージョン、次のコマンドを入力します。

```
aws s3api create-bucket --bucket insert-unique-bucket-name-here --region us-east-1
```

> **Note**
>
> us-east-1 リージョン外にバケットを作成しようとする場合、LocationConstraint パラメータ付きの create-bucket-configuration を追加してリージョンを指定します。構文の例を次に示します。
>
> ```
> $ aws s3api create-bucket --bucket my-bucket --region eu-west-1 --create-
> bucket-configuration LocationConstraint=eu-west-1
> ```

コールが成功すると、コマンドラインに次の出力に似たサービスからのレスポンスが表示されます。

```
{
    "Location": "/insert-unique-bucket-name-here"
}
```

> **Note**
>
> バケット名の命名規則に従わない場合、以下のようなエラーが表示されます: CreateBucket オペレーションの呼び出し時にエラー (InvalidBucketName) が発生しました。指定されたバケットは有効ではありません。

2. ファイルをアップロードし、作成したばかりのバケットにオブジェクトとしてファイルを追加するには、PutObject メソッドを呼び出します。

```
aws s3api put-object --bucket insert-unique-bucket-name-here --key add_prog --body
 add_prog.py
```

オブジェクトが Amazon S3 バケットにアップロードされたら、コマンドラインに次の出力に似たサービスからのレスポンスが表示されます。

```
{"ETag": "\"ab123c1:w:wad4a567d8bfd9a1234ebeea56\""}
```

ETag は、格納されたオブジェクトのハッシュです。このハッシュを使用して、Amazon S3 にアップロードされたオブジェクトの整合性を確認できます。

関連トピック

- CloudShell で CLI から AWS サービスを管理する
- ローカルマシンと CloudShell の間で複数のファイルをコピーする
- AWS CloudShell 概念
- AWS CloudShell エクスペリエンスのカスタマイズ

AWS CloudShell のチュートリアル

以下のチュートリアルでは、AWS CloudShell の使用時にさまざまな機能および統合を試し、テストする方法を示します。

チュートリアルの概要	詳細
複数のファイルのコピー	the section called “チュートリアル: 複数のファイルをコピーする”
署名付き URL の作成	???
AWS CloudShell 内で Docker コンテナを構築して Amazon ECR にプッシュする	???
AWS CDK を使用して Lambda 関数をデプロイする	???

ローカルマシンと CloudShell の間で複数のファイルをコピーする

このチュートリアルでは、ローカルマシンと CloudShell の間で複数のファイルをコピーする方法を示します。

AWS CloudShell インターフェイスを使用して、ローカルマシンとシェル環境の間で一度に 1 つのファイルをアップロードまたはダウンロードできます。CloudShell とローカルマシン間で複数のファイルを同時にコピーするには、次のいずれかのオプションを使用します。

- Amazon S3: ローカルマシンと CloudShell 間でファイルをコピーするときは、S3 バケットを仲介として使用します。
- Zip ファイル: CloudShell インターフェイスを使用してアップロードまたはダウンロードできる 1 つの zip フォルダに複数のファイルを圧縮します。

> **Note**
>
> CloudShell は着信インターネットトラフィックを許可しないため、現時点では `scp` または `rsync` などのコマンドを使用してローカルマシンと CloudShell コンピューティング環境の間で複数のファイルをコピーすることはできません。

Amazon S3 を使用した複数のファイルのアップロードとダウンロード

このステップでは、Amazon S3 を使用して複数のファイルをアップロードおよびダウンロードする方法について説明します。

前提条件

バケットとオブジェクトを操作するには、次の Amazon S3 API アクションを実行するアクセス許可を付与する IAM ポリシーが必要です。

- `s3:CreateBucket`
- `s3:PutObject`
- `s3:GetObject`
- `s3:ListBucket`

Amazon S3 のアクション一覧については、Amazon Simple Storage Service API リファレンスの「アクション」を参照してください。

Amazon S3 AWS CloudShell を使用して複数のファイルを にアップロードする Amazon S3

このステップでは、Amazon S3 を使用して複数のファイルをアップロードする方法について説明します。

1. で AWS CloudShell、次の`s3`コマンドを実行して S3 バケットを作成します。

   ```
   aws s3api create-bucket --bucket your-bucket-name --region us-east-1
   ```

   コールが成功すると、コマンドラインに S3 サービスからのレスポンスが表示されます。

   ```
   {
       "Location": "/your-bucket-name"
   ```

```
}
```

2. ローカルマシンからバケットにディレクトリ内のファイルをアップロードします。ファイルをアップロードするために、次のいずれかのオプションを選択します。

   - AWS マネジメントコンソール: ドラッグアンドドロップを使用してフォルダとファイルを S3 バケットにアップロードするには
   - AWS CLI: ローカルマシンにインストールされているバージョンのツールで、コマンドラインを使用してファイルとフォルダをバケットにアップロードします。

   Using the console

   - https://s3.console.aws.amazon.com/s3/ で Amazon S3 コンソールを開きます。

     (を使用している場合は AWS CloudShell、 コンソールに既にログインしている必要があります) 。
   - 左のナビゲーションペインで、 [バケット] を選択し、次にフォルダやファイルのアップロード先のバケット名を選択します。 [バケットの作成] からも、選択したバケットを作成することができます。
   - アップロードしたいファイルやフォルダを選択するには、 [アップロード] を選択します。次に、選択したファイルやフォルダーを宛先バケット内のオブジェクトを一覧表示するコンソールウィンドウ内にドラッグアンドドロップします。または、 [ファイルの追加] もしくは [フォルダーの追加] を選択します。

     選択したファイルは、[Upload (アップロード)] ページに一覧表示されます。
   - チェックボックスを選択して、追加するファイルを指定します。
   - [アップロード] を選択して、選択したファイルをバケットに追加します。

   > **Note**
   >
   > コンソールを使用する際の設定オプションの全範囲の詳細については、Amazon Simple Storage Service コンソールユーザーガイドの「S3 バケットにファイルとフォルダをアップロードする方法」を参照してください。

Using AWS CLI

> **Note**
>
> このオプションでは、 AWS CLI ツールをローカルマシンにインストールし、 AWS サービスの呼び出し用に認証情報を設定する必要があります。詳細については、「AWS Command Line Interface ユーザーガイド」を参照してください。

- AWS CLI ツールを起動し、次のaws s3コマンドを実行して、指定されたバケットをローカルマシンの現在のディレクトリの内容と同期します。

```
aws s3 sync folder-path s3://your-bucket-name
```

同期が成功すると、バケットに追加されたすべてのオブジェクトについてアップロードメッセージが表示されます。

3. CloudShell コマンドラインに戻り、次のコマンドを入力して、シェル環境のディレクトリを S3 バケットの内容と同期させます。

```
aws s3 sync  s3://your-bucket-name folder-path
```

> **Note**
>
> パターンマッチングを実行して特定のファイルやオブジェクトを除外または含めるには、`--exclude "<value>"` や `--include "<value>"` のパラメータを sync コマンドに追加します。
> 詳細については、 [AWS CLI コマンドリファレンス] の [除外フィルタと包含フィルタの使用] を参照してください。

同期が成功すると、バケットからディレクトリにダウンロードされたすべてのファイルについて、ダウンロードメッセージが表示されます。

> **Note**
>
> 新しいファイルおよび更新されたファイルをソースディレクトリから送信先に再帰的にコピーします。

Amazon S3 AWS CloudShell を使用して から複数のファイルをダウンロードする

このステップでは、Amazon S3 を使用して複数のファイルをダウンロードする方法について説明します。

1. AWS CloudShell コマンドラインを使用して、次のaws s3コマンドを入力して、S3 バケットをシェル環境の現在のディレクトリの内容と同期します。

```
aws s3 sync folder-path s3://your-bucket-name
```

> **Note**
>
> パターンマッチングを実行して特定のファイルやオブジェクトを除外または含めるには、`--exclude "<value>"` や `--include "<value>"` のパラメータを sync コマンドに追加します。
> 詳細については、 [AWS CLI コマンドリファレンス] の [除外フィルタと包含フィルタの使用] を参照してください。

   同期が成功すると、バケットに追加されたすべてのオブジェクトについてアップロードメッセージが表示されます。

2. バケットの内容をローカルマシンにダウンロードします。Amazon S3 コンソールは複数のオブジェクトのダウンロードをサポートしていないので、ローカルマシンにインストールされる AWS CLI ツールを使用する必要があります。

   AWS CLI ツールのコマンドラインから、次のコマンドを実行します。

```
aws s3 sync s3://your-bucket-name folder-path
```

   同期が成功すると、宛先ディレクトリに更新または追加された各ファイルのダウンロードメッセージがコマンドラインに表示されます。

> **Note**
>
> このオプションでは、 AWS CLI ツールをローカルマシンにインストールし、 AWS サービスの呼び出し用に認証情報を設定する必要があります。詳細については、「AWS Command Line Interface ユーザーガイド」を参照してください。

zip フォルダを使用した複数のファイルのアップロードとダウンロード

このステップでは、圧縮フォルダを使用して複数のファイルをアップロードおよびダウンロードする方法について説明します。

zip/unzip ユーティリティを使用すると、単一のファイルとして扱うことができるアーカイブ内の複数のファイルを圧縮できます。ユーティリティは CloudShell コンピューティング環境に事前にインストールされています。

プリインストールツールの詳細については、「開発ツールおよびシェルユーティリティ」を参照してください。

zip フォルダ AWS CloudShell を使用して複数のファイルを にアップロードする

このステップでは、圧縮フォルダを使用して複数のファイルをアップロードする方法について説明します。

1. ローカルマシンで、アップロードするファイルを zip フォルダに追加します。
2. CloudShell を起動させ、[アクション]、[ファイルのアップロード] を選択します。
3. [ファイルのアップロード] ダイアログボックスで[ファイルを選択] を選択し、作成した zip フォルダを選択します。
4. [ファイルのアップロード] ダイアログボックスで [アップロード] を選択し、選択したファイルをシェル環境に追加します。
5. CloudShell コマンドラインで次のコマンドを実行して、zip アーカイブの内容を指定されたディレクトリに解凍します。

```
unzip zipped-files.zip -d my-unzipped-folder
```

zip フォルダ AWS CloudShell を使用して から複数のファイルをダウンロードする

このステップでは、圧縮フォルダを使用して複数のファイルをダウンロードする方法について説明します。

1. CloudShell コマンドラインで次のコマンドを実行して、現在のディレクトリ内のすべてのファイルを zip フォルダに追加します。

   ```
   zip -r zipped-archive.zip *
   ```

2. [Actions] の [ダウンロード ファイル] を選択します。
3. [ファイルのダウンロード] ダイアログボックスで、zip フォルダのパス (例えば、`/home/cloudshell-user/zip-folder/zipped-archive.zip`など) を入力し、[ダウンロード] を選択します。

   パスが正しい場合は、ブラウザのダイアログで zip フォルダを開くか、ローカルマシンに保存するかを選択できます。
4. ローカルマシンで、ダウンロードした zip フォルダの内容を解凍できるようになりました。

CloudShell を使用して Amazon S3 オブジェクトの署名付き URL を作成する

このチュートリアルでは、Amazon S3 オブジェクトを他のユーザーと共有するために、署名付き URL を作成する方法を示します。オブジェクトの所有者は、共有時に独自のセキュリティ認証情報を指定するため、署名済み URL を受信したユーザーは誰でも期間限定でオブジェクトにアクセスできます。

前提条件

- AWSCloudShellFullAccess ポリシーで提供されるアクセス許可を持つ IAM ユーザー。
- 署名付き URL を作成するのに必要な IAM アクセス許可については、Amazon Simple Storage Service ユーザーガイドの「他のユーザーとのオブジェクトの共有」を参照してください。

ステップ 1: Amazon S3 バケットへのアクセスを許可する IAM ロールを作成する

このステップでは、Amazon S3 バケットへのアクセスを許可する IAM ロールを作成する方法について説明します。

1. 共有できる IAM の詳細を取得するには、`get-caller-identity` から AWS CloudShellコマンドを呼び出します。

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

   コールが正常に終了すると、コマンドラインに次のようなレスポンスが表示されます。

   ```
   {
       "Account": "123456789012",
       "UserId": "AROAXXOZUUOTTWDCVIDZ2:redirect_session",
       "Arn": "arn:aws:sts::531421766567:assumed-role/Feder08/redirect_session"
   }
   ```

2. 前のステップで取得したユーザー情報を取得し、ある CloudFormation テンプレート追加。このテンプレートにより IAM ロールが作成されます。このロールは、共有リソースの最小特権を共同作業者に付与します。

   ```
   Resources:
     CollaboratorRole:
       Type: AWS::IAM::Role
       Properties:
         AssumeRolePolicyDocument:
           Version: 2012-10-17
           Statement:
             - Effect: Allow
               Principal:
                 AWS: "arn:aws:iam::531421766567:role/Feder08"
               Action: "sts:AssumeRole"
         Description: Role used by my collaborators
         MaxSessionDuration: 7200
     CollaboratorPolicy:
       Type: AWS::IAM::Policy
       Properties:
         PolicyDocument:
           Version: 2012-10-17
   ```

```
        Statement:
          - Effect: Allow
            Action:
              - 's3:*'
            Resource: 'arn:aws:s3:::<YOUR_BUCKET_FOR_FILE_TRANSFER>'
            Condition:
              StringEquals:
                s3:prefix:
                 - "myfolder/*"
      PolicyName: S3ReadSpecificFolder
      Roles:
        - !Ref CollaboratorRole
Outputs:
  CollaboratorRoleArn:
    Description: Arn for the Collaborator's Role
    Value: !GetAtt CollaboratorRole.Arn
```

3. CloudFormation テンプレートを という名前のファイルに保存しますtemplate.yaml。
4. テンプレートを使用してスタックをデプロイし、deploy コマンドを呼び出して IAM ロールを作成します。

```
aws cloudformation deploy --template-file ./template.yaml --stack-name
 CollaboratorRole --capabilities CAPABILITY_IAM
```

署名付き URL の生成

このステップでは、署名付き URL を生成する方法について説明します。

1. でエディタを使用して AWS CloudShell、次のコードを追加します。このコードは、フェデレーティッドユーザーが AWS マネジメントコンソールに直接アクセスできるように URL を作成します。

```
import urllib, json, sys
import requests
import boto3
import os

def main():
  sts_client = boto3.client('sts')
  assume_role_response = sts_client.assume_role(
      RoleArn=os.environ.get(ROLE_ARN),
```

```
        RoleSessionName="collaborator-session"
    )
    credentials = assume_role_response['Credentials']
    url_credentials = {}
    url_credentials['sessionId'] = credentials.get('AccessKeyId')
    url_credentials['sessionKey'] = credentials.get('SecretAccessKey')
    url_credentials['sessionToken'] = credentials.get('SessionToken')
    json_string_with_temp_credentials = json.dumps(url_credentials)
    print(f"json string {json_string_with_temp_credentials}")

    request_parameters = f"?
Action=getSigninToken&Session={urllib.parse.quote(json_string_with_temp_credentials)}"
    request_url = "https://signin.aws.amazon.com/federation" + request_parameters
    r = requests.get(request_url)
    signin_token = json.loads(r.text)
    request_parameters = "?Action=login"
    request_parameters += "&Issuer=Example.org"
    request_parameters += "&Destination=" + urllib.parse.quote("https://us-
west-2.console.aws.amazon.com/cloudshell")
    request_parameters += "&SigninToken=" + signin_token["SigninToken"]
    request_url = "https://signin.aws.amazon.com/federation" + request_parameters

    # Send final URL to stdout
    print (request_url)

if __name__ == "__main__":
    main()
```

2. share.py という名前のファイルにコードを保存します。
3. コマンドラインで以下を実行し、IAM ロールの Amazon リソースネーム (ARN) を CloudFormationから取得します。次に、Python スクリプトでこれを使用して、一時的なセキュリティ認証情報を取得します。

```
ROLE_ARN=$(aws cloudformation describe-stacks --stack-name CollaboratorRole --query
 "Stacks[*].Outputs[?OutputKey=='CollaboratorRoleArn'].OutputValue" --output text)
 python3 ./share.py
```

このスクリプトは、共同作業者がクリックして AWS CloudShell の に移動できる URL を返します AWS マネジメントコンソール。共同作業者は、次の 3,600 秒 (1 時間) だけ Amazon S3 バケット内の myfolder/ フォルダを完全に制御できます。認証情報は 1 時間後に無効になります。この期間が過ぎると、 共同作業者はバケットにアクセスできなくなります。

CloudShell 内で Docker コンテナを構築して Amazon ECR リポジトリにプッシュする

このチュートリアルでは、 で Docker コンテナを定義して構築 AWS CloudShell し、Amazon ECR リポジトリにプッシュする方法について説明します。

前提条件

- コンテナを作成して Amazon ECR リポジトリにプッシュするためのアクセス許可が必要です。Amazon ECR のリポジトリの詳細については、「Amazon ECR ユーザーガイド」の「Amazon ECR プライベートリポジトリ」を参照してください。Amazon ECR にイメージをプッシュするために必要なアクセス許可の詳細については、「Amazon ECR ユーザーガイド」の「イメージをプッシュするために必要な IAM アクセス許可」を参照してください。

チュートリアルの手順

次のチュートリアルでは、CloudShell インターフェイスを使用して Docker コンテナを構築し、Amazon ECR リポジトリにプッシュする方法について説明します。

1. ホームディレクトリに新しいフォルダを作成します。

```
mkdir ~/docker-cli-tutorial
```

2. 作成したフォルダに移動します。

```
cd ~/docker-cli-tutorial
```

3. 空の Dockerfile を作成します。

```
touch Dockerfile
```

4. テキストエディタ (nano Dockerfile など) を使用し、ファイルを開いて次の内容を貼り付けます。

```
# Dockerfile

# Base this container on the latest Amazon Linux version
FROM public.ecr.aws/amazonlinux/amazonlinux:latest
```

```
# Install the cowsay binary
RUN dnf install --assumeyes cowsay

# Default entrypoint binary
ENTRYPOINT [ "cowsay" ]

# Default argument for the cowsay entrypoint
CMD [ "Hello, World!" ]
```

5. これで、Dockerfile を構築する準備が整いました。docker build を実行してコンテナを構築します。コンテナには、将来のコマンドで使用できるように、入力しやすい名前をタグ付けします。

```
docker build --tag test-container .
```

末尾のピリオド (.) を必ず含めます。
AWS CloudShell内で実行する Docker ビルドコマンドの画像。

6. これで、コンテナをテストし、 AWS CloudShellで正しく動作することを確認できます。

```
docker container run test-container
```

内の docker コンテナ実行コマンドの画像 AWS CloudShell

7. Docker コンテナが正常に動作することを確認したら、これを Amazon ECR リポジトリにプッシュする必要があります。既存の Amazon ECR リポジトリがある場合は、このステップをスキップできます。

次のコマンドを実行して、このチュートリアル用の Amazon ECR リポジトリを作成します。

```
ECR_REPO_NAME=docker-tutorial-repo
aws ecr create-repository --repository-name ${ECR_REPO_NAME}
```

内に Amazon ECR リポジトリを作成するために使用されるコマンドの画像 AWS CloudShell

8. Amazon ECR リポジトリを作成したら、これに Docker コンテナをプッシュできます。

次のコマンドを実行して、Docker の Amazon ECR サインイン認証情報を取得します。

```
AWS_ACCOUNT_ID=$(aws sts get-caller-identity --query "Account" --output text)
ECR_URL=${AWS_ACCOUNT_ID}.dkr.ecr.${AWS_REGION}.amazonaws.com
```

```
aws ecr get-login-password | docker login --username AWS --password-stdin
 ${ECR_URL}
```

Docker の Amazon ECR サインイン認証情報の取得に使用するコマンドの画像。

> **Note**
>
> AWS_REGION 環境変数を CloudShell に設定していないか、他の AWS リージョンのリソースとやり取りする場合は、次のコマンドを実行します。
>
> ```
> AWS_REGION=<your-desired-region>
> ```

9. イメージにターゲットの Amazon ECR リポジトリのタグを付け、そのリポジトリにイメージをプッシュします。

```
docker tag test-container ${ECR_URL}/${ECR_REPO_NAME}
docker push ${ECR_URL}/${ECR_REPO_NAME}
```

イメージにターゲット Amazon ECR リポジトリのタグを付けるために使用するコマンドの画像。

このチュートリアルを実行しようとしてエラーや問題が発生した場合は、このガイドの「トラブルシューティング」セクションを参照してください。

クリーンアップ

これで、Docker コンテナを Amazon ECR リポジトリに正常にデプロイしました。このチュートリアルで作成したファイルを AWS CloudShell 環境から削除するには、次のコマンドを実行します。

- ```
  cd ~
  rm -rf ~/docker-cli-tutorial
  ```
- Amazon ECR リポジトリを削除します。

```
aws ecr delete-repository --force --repository-name ${ECR_REPO_NAME}
```
```

CloudShell で を使用して Lambda 関数 AWS CDK をデプロイする

このチュートリアルでは、CloudShell で AWS Cloud Development Kit (AWS CDK) を使用して Lambda 関数をアカウントにデプロイする方法を示します。

前提条件

- AWS CDK用にアカウントをブートストラップします。でのブートストラップの詳細については AWS CDK、「 AWS CDK v2 デベロッパーガイド」の「ブートストラップ」を参照してください。アカウントをブートストラップしていない場合は、CloudShell で `cdk bootstrap` を実行できます。

- リソースをアカウントにデプロイするための適切なアクセス許可があることを確認します。管理者アクセス許可が推奨されます。

チュートリアルの手順

次のチュートリアルでは、CloudShell の を使用して Docker コンテナベースの Lambda 関数 AWS CDK をデプロイする方法について説明します。

1. ホームディレクトリに新しいフォルダを作成します。

   ```
   mkdir ~/docker-cdk-tutorial
   ```

2. 作成したフォルダに移動します。

   ```
   cd ~/docker-cdk-tutorial
   ```

3. AWS CDK 依存関係をローカルにインストールします。

   ```
   npm install aws-cdk aws-cdk-lib
   ```

   AWS CDK 依存関係のインストールに使用されるコマンドの画像。

4. 作成したフォルダにスケルトン AWS CDK プロジェクトを作成します。

   ```
   touch cdk.json
   mkdir lib
   touch lib/docker-tutorial.js lib/Dockerfile lib/hello.js
   ```

5. テキストエディタ (nano cdk.json など) を使用し、ファイルを開いて次の内容を貼り付けます。

```
{
  "app": "node lib/docker-tutorial.js"
}
```

6. lib/docker-tutorial.js ファイルを開いて次の内容を貼り付けます。

```
// this file defines the CDK constructs we want to deploy

const { App, Stack } = require('aws-cdk-lib');
const { DockerImageFunction, DockerImageCode } = require('aws-cdk-lib/aws-lambda');
const path = require('path');

// create an application
const app = new App();

// define stack
class DockerTutorialStack extends Stack {
  constructor(scope, id, props) {
    super(scope, id, props);

    // define lambda that uses a Docker container
    const dockerfileDir = path.join(__dirname);
    new DockerImageFunction(this, 'DockerTutorialFunction', {
      code: DockerImageCode.fromImageAsset(dockerfileDir),
      functionName: 'DockerTutorialFunction',
    });
  }
}

// instantiate stack
new DockerTutorialStack(app, 'DockerTutorialStack');
```

7. lib/Dockerfile を開いて次の内容を貼り付けます。

```
# Use a NodeJS 20.x runtime
FROM public.ecr.aws/lambda/nodejs:20

# Copy the function code to the LAMBDA_TASK_ROOT directory
# This environment variable is provided by the lambda base image
COPY hello.js ${LAMBDA_TASK_ROOT}
```

```
# Set the CMD to the function handler
CMD [ "hello.handler" ]
```

8. lib/hello.js ファイルを開いて次の内容を貼り付けます。

```
// define the handler
exports.handler = async (event) => {
  // simply return a friendly success response
  const response = {
    statusCode: 200,
    body: JSON.stringify('Hello, World!'),
  };
  return response;
};
```

9. AWS CDK CLI を使用してプロジェクトを合成し、リソースをデプロイします。アカウントをブートストラップする必要があります。

```
npx cdk synth
npx cdk deploy --require-approval never
```

CLI AWS CDK を使用してプロジェクトを合成し、リソースをデプロイするコマンドの画像。

10. Lambda 関数を呼び出して確認および検証します。

```
aws lambda invoke --function-name DockerTutorialFunction out.json
jq . out.json
```

Lambda 関数の呼び出しに使用するコマンドの画像。

これで、 AWS CDKを使用して Docker コンテナベースの Lambda 関数を正常にデプロイしました。詳細については AWS CDK、「 AWS CDK v2 デベロッパーガイド」を参照してください。このチュートリアルを実行しようとしてエラーや問題が発生した場合は、このガイドの「トラブルシューティング」セクションを参照してください。

クリーンアップ

これで、 AWS CDKを使用して Docker コンテナベースの Lambda 関数を正常にデプロイしました。AWS CDK プロジェクト内で次のコマンドを実行して、関連付けられたリソースを削除します。削除を確認するプロンプトが表示されます。

- ```
  npx cdk destroy DockerTutorialStack
  ```
- このチュートリアルで作成したファイルとリソースを AWS CloudShell 環境から削除するには、次のコマンドを実行します。

  ```
  cd ~
  rm -rf ~/docker-cli-tutorial
  ```
  ```

AWS CloudShell 概念

このセクションでは、サポートされているアプリケーションとやり取り AWS CloudShell し、特定のアクションを実行する方法について説明します。

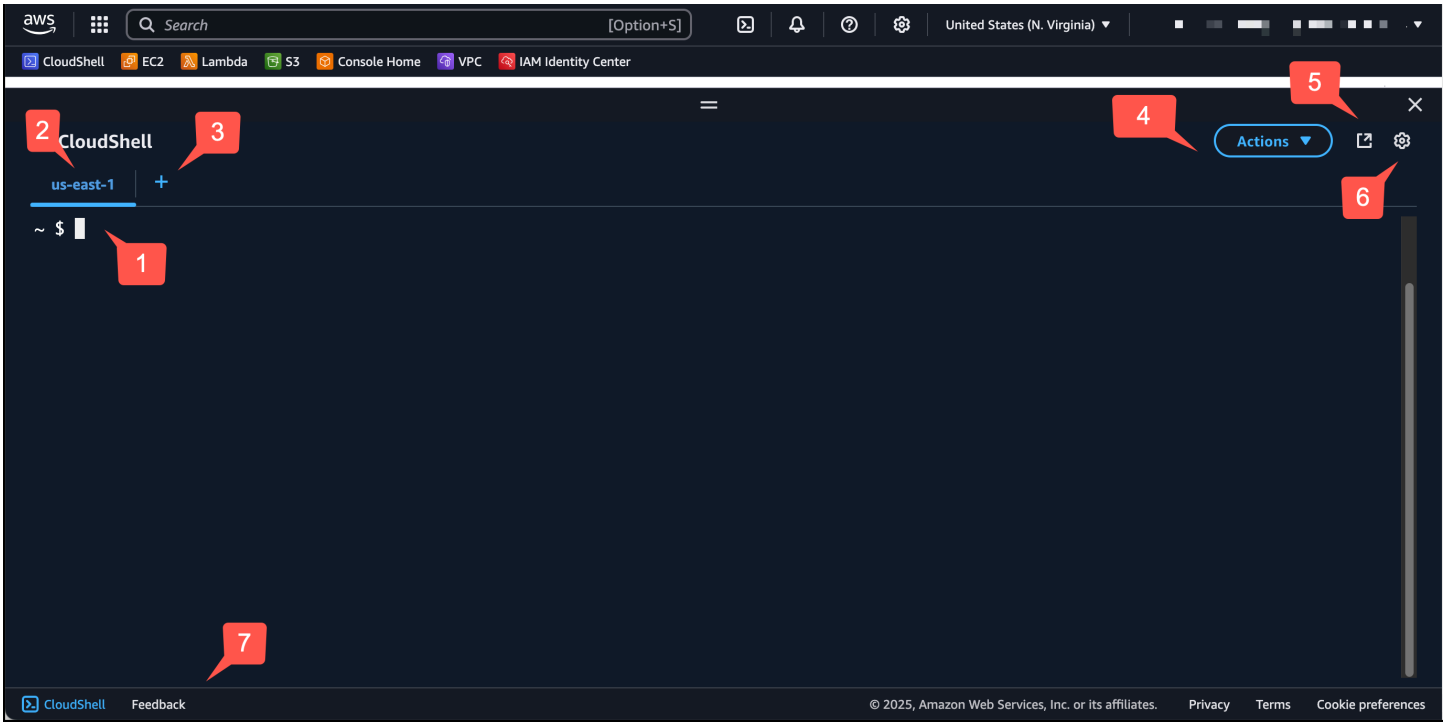
トピック

- AWS CloudShell インターフェイスの操作
- での作業 AWS リージョン
- ファイルおよびストレージの操作
- コンソールモバイルアプリケーションで CloudShell にアクセスする
- Docker の使用

AWS CloudShell インターフェイスの操作

CloudShell インターフェイス機能は、 AWS マネジメントコンソール および からナビゲートできますConsole Toolbar。

次のスクリーンショットは、いくつかの主要な AWS CloudShell インターフェイス機能を示しています。



1. AWS CloudShell 任意のシェルを使用してコマンドを実行するために使用するコマンドラインインターフェイス。現在のシェルの種類は、コマンドプロンプトで示されます。
2. ターミナルタブ AWS CloudShell 。現在実行中 AWS リージョン の を使用します。
3. [+] アイコンは、環境を作成、再起動、削除するオプションを含むドロップダウンメニューです。
4. [アクション] メニューには、画面レイアウトの変更、ファイルのダウンロードとアップロード、AWS CloudShellの再起動、AWS CloudShell ホームディレクトリの削除のためのオプションがあります。

> **Note**
>
> Console Toolbar で CloudShell を起動する場合、[ダウンロード] オプションは使用できません。

5. [新しいブラウザで開くタブ] では、CloudShell セッションに全画面表示でアクセスすることができます。
6. シェル環境のカスタマイズに使用できる [Preferences (設定)] オプション。
7. 下部のバーには、以下のオプションがあります。
   - [CloudShell] アイコンで CloudShell を起動します。
   - [フィードバック] アイコンでフィードバックを送信します。送信するフィードバックの種類を選択し、コメントを追加して、[送信] を選択します。
     - CloudShell のフィードバックを送信するには、以下のいずれかのオプションを選択します。
       - コンソールから CloudShell を起動し、[フィードバック] を選択します。コメントを追加し、[送信] を選択します。
       - コンソールの左下にある Console Toolbar で [CloudShell] を選択し、[新しいブラウザタブで開く] アイコン、 [フィードバック] を選択します。コメントを追加し、[送信] を選択します。

> **Note**
>
> Console Toolbar で CloudShell を起動する場合、[フィードバック] オプションは使用できません。

- 当社のプライバシーポリシーと利用規約を確認し、Cookie の設定をカスタマイズしてください。

での作業 AWS リージョン

で実行 AWS リージョン している現在の がタブとして表示されます。

リージョンセレクタを使用して特定のリージョンを選択することで、 AWS リージョン 作業する を選択できます。リージョンを変更すると、シェルセッションが選択されたリージョンで実行中の異なるコンピューティング環境に接続するため、インターフェースが更新されます。

> **Important**
>
> - 各 で最大 1 GB の永続的ストレージを使用できます AWS リージョン。永続的ストレージは、ホームディレクトリに保存されます ($HOME)。つまりこれは、ホームディレクトリに保存されている個人用ファイル、ディレクトリ、プログラムまたはスクリプトが 1 つの AWS リージョンに保存されることを意味します。さらに、ホームディレクトリに配置され、別のリージョンに格納されているものとは異なります。
>
>   永続的ストレージ内のファイルの長期保存もリージョンごとに管理されます。詳細については、「永続ストレージ」を参照してください。
> - 永続的ストレージは AWS CloudShell VPC 環境では使用できません。

AWS リージョン のデフォルトを指定する AWS CLI

環境変数を使用して、 AWS のサービス を使用するために必要な設定オプションと認証情報を指定できます AWS CLI。シェルセッション AWS リージョン のデフォルトを指定する環境変数は、 の特定のリージョンから起動 AWS CloudShell するとき、 AWS マネジメントコンソール またはリージョンセレクタでオプションを選択するときに、 に設定されます。

環境変数は、 によって更新される AWS CLI 認証情報ファイルよりも優先されますaws configure。そのため、環境変数で指定したリージョンを変更する `aws configure` コマンドを実行することはできません。代わりに、 AWS CLI コマンドのデフォルトのリージョンを変更するには、`AWS_REGION`環境変数に値を割り当てます。以下の例では、`us-east-1` を現在のリージョンに置き換えてください。

Bash or Zsh

```
$ export AWS_REGION=us-east-1
```

環境変数を設定することで、シェルセッションの終了時、または変数に別の値を設定するまで、使用する値が変更されます。シェルのスタートアップスクリプトで変数を設定することで、以降のセッションでその変数を永続的なものにすることができます。

PowerShell

```
PS C:\> $Env:AWS_REGION="us-east-1"
```

PowerShell プロンプトで環境変数を設定した場合、環境変数は現在のセッションの期間だけ値を保存します。または、PowerShell プロファイルに変数を追加すると、以降のすべての PowerShell セッションにその変数が設定されます。環境変数の保存についての詳細は、PowerShell ドキュメントを参照してください。

デフォルトのリージョンを変更したことを確認するには、 `aws configure list` コマンドを実行して現在の AWS CLI 設定データを表示します。

> **Note**
>
> 特定の AWS CLI コマンドでは、コマンドラインオプション を使用してデフォルトのリージョンを上書きできます`--region`。詳細については、 AWS Command Line Interface ユーザーガイドの「コマンドラインオプション」を参照してください。

ファイルおよびストレージの操作

AWS CloudShellの インターフェイスを使用すると、シェル環境にファイルをアップロードしたり、シェル環境からファイルをダウンロードしたりできます。ファイルのダウンロードとアップロードの詳細については、「 の開始方法」を参照してください AWS CloudShell。

セッション終了後に追加したファイルを使用できるようにするには、永続的ストレージおよび一時ストレージの違いを知っておく必要があります。

- 永続的ストレージ: それぞれ 1 GB の永続的ストレージがあります AWS リージョン。永続的ストレージは、ホームディレクトリにあります。

- 一時ストレージ: 一時ストレージはセッションの終了時にリサイクルされます。一時ストレージは、ホームディレクトリの外部のディレクトリにあります。

> **Important**
>
> 今後のシェルセッション用に確保し、使用したいファイルは、ホームディレクトリに残してください。例えば、`mv` コマンドを実行してファイルをホームディレクトリの外に移動したとします。その後、そのファイルは現在のシェルセッションが終了するとリサイクルされます。

コンソールモバイルアプリケーションで CloudShell にアクセスする

の CloudShell には、ホーム画面 AWS Console Mobile Application からアクセスできます。ホーム画面から、CloudShell およびその他の AWS サービスに関する情報を表示できます。詳細については、「AWS Console Mobile Applicationを使い始める」を参照してください。で CloudShell を起動するには AWS Console Mobile Application、次のいずれかのオプションを選択します。

- ナビゲーションバーの下部にある CloudShell アイコンを選択します。
- サービスメニューの CloudShell を選択します。

X を選択すると、いつでも CloudShell を終了できます。

コンソールモバイルアプリケーションで CloudShell にアクセスする方法の詳細については、「アクセス AWS CloudShell」を参照してください。

> **Note**
>
> 現在、 AWS Console Mobile Applicationで VPC 環境を作成または起動することはできません。

Docker の使用

AWS CloudShell は、インストールや設定なしで Docker を完全にサポートします。内部で Docker コンテナを定義、構築、実行できます AWS CloudShell。Toolkit を介して Docker コンテナに基づく

Lambda 関数などの Docker ベースのリソースをデプロイ AWS CDK したり、Docker コンテナを構築して Docker CLI を介して Amazon ECR リポジトリにプッシュしたりできます。これらの両方のデプロイを実行する方法の詳細な手順については、以下のチュートリアルを参照してください。

- チュートリアル: を使用した Lambda 関数のデプロイ AWS CDK
- チュートリアル: 内に Docker コンテナを構築し AWS CloudShell 、Amazon ECR リポジトリにプッシュする

AWS CloudShellで Docker を使用する場合、特定の成約と制限があります。

- 環境における Docker のスペースは限られています。個々のイメージが大きい場合、または既存の Docker イメージが多すぎる場合、追加のイメージのプル、構築、または実行を妨げるような問題が発生する可能性があります。Docker の詳細については、Docker ドキュメントのガイドを参照してください。
- Docker は、AWS GovCloud (米国) リージョンを除く、すべての AWS リージョンで使用できます。Docker が利用可能なリージョンのリストについては、「 でサポートされている AWS リージョン AWS CloudShell」を参照してください。
- で Docker を使用する際に問題が発生した場合は AWS CloudShell、このガイドの「トラブルシューティング」セクションで、これらの問題を解決する方法について説明します。

のアクセシビリティ機能 AWS CloudShell

このトピックでは、CloudShell のアクセシビリティ機能の使用方法について説明します。キーボードを使用して、ページ上のフォーカス可能な要素の間を移動することができます。フォントサイズやインターフェーステーマなど、CloudShell の外観をカスタマイズすることもできます。

CloudShell のキーボードナビゲーション

ページ上のフォーカス可能な要素間を移動するには、Tab を押します。

CloudShell ターミナルのアクセシビリティ機能

Tab キーを使用して、以下のモードを使用できます。

- ターミナルモード (デフォルト) — このモードでは、Tab ターミナルはキーエントリーをキャプチャします。ターミナルにフォーカスが移ったら、Tab を押してターミナルの機能のみにアクセスします。
- ナビゲーションモード — このモードでは、ターミナルは Tab キー入力をキャプチャしません。Tab を押すと、ページ上のフォーカス可能な要素の間を移動できます。

ターミナルモードとナビゲーションモードを切り替えるには、Ctrl+M を押します。切り替え後、ヘッダーに [タブ: ナビゲーション] が表示されるので、Tab キーを使用してページ内を移動できます。

ターミナルモードに戻るには、Ctrl+M を押します。または、[タブ:ナビゲーション]の横にある X を選択します。

> **Note**
>
> 現在、CloudShell ターミナルのアクセシビリティ機能はモバイルデバイスでは利用できません。

CloudShell でのフォントサイズとインターフェーステーマの選択

CloudShell の外観は、好みの見た目に合わせてカスタマイズできます。

- フォントサイズ — ターミナルのフォントサイズを [最小]、[小]、[中]、[大]、[最大] から選択します。フォントサイズの変更の詳細については、「the section called “フォントサイズを変更する”」を参照してください。
- テーマ — インターフェーステーマを「明るい」と「暗い」から選択します。インターフェーステーマの変更の詳細については、「the section called “インターフェイステーマの変更”」を参照してください。

CloudShell で CLI から AWS サービスを管理する

の主な利点 AWS CloudShell は、これを使用してコマンドラインインターフェイスから AWS サービスを管理できることです。つまり、ツールをダウンロードしてインストールしたり、ローカルで認証情報を事前に設定する必要はありません。を起動すると AWS CloudShell、次の AWS コマンドラインツールが既にインストールされているコンピューティング環境が作成されます。

- AWS CLI
- AWS Elastic Beanstalk CLI
- Amazon ECS CLI
- AWS SAM

また、すでにサインインしているため AWS、 サービスを使用する前に認証情報をローカルで設定する必要はありません。 AWS マネジメントコンソール へのサインに使用した認証情報は、 AWS CloudShellに転送されます。

使用するデフォルトの AWS リージョンを変更する場合は AWS CLI、`AWS_REGION`環境変数に割り当てられた値を変更できます。(詳細については「AWS リージョン のデフォルトを指定する AWS CLI」 を参照してください)。

このトピックの残りの部分では、 AWS CloudShell を使用してコマンドラインから選択した AWS サービスとやり取りする方法を示します。

AWS CLI 選択した AWS サービスのコマンドラインの例

次の例は、 AWS CLI バージョン 2 から利用可能なコマンドを使用して操作できる多数の AWS サービスの一部のみを示しています。詳細なリストについては、AWS CLI コマンドリファレンスを参照してください。

- DynamoDB
- Amazon EC2
- Amazon Glacier

DynamoDB

DynamoDB は、高速で予測可能なパフォーマンスとシームレスなスケーラビリティを特長とするフルマネージド NoSQL データベースサービスです。このサービスの NoSQL モードの実装は、キーバリューおよびドキュメントデータ構造をサポートしています。

次のcreate-tableコマンドは、MusicCollection AWS アカウントに という名前の NoSQL スタイルのテーブルを作成します。

```
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=5,WriteCapacityUnits=5 \
    --tags Key=Owner,Value=blueTeam
```

詳細については、AWS Command Line Interface ユーザーガイドの「AWS CLIでの Amazon DynamoDB の使用」を参照してください。

Amazon EC2

Amazon Elastic Compute Cloud (Amazon EC2) は、クラウド内で安心で再サイズを変更できるコンピューティング性能を提供するウェブサービスです。ウェブスケールのクラウドコンピューティングを簡単かつアクセスしやすく利用できるように設計されています。

次の run-instances コマンドは、VPC の特定のサブネット内で t2 マイクロインスタンスを起動します。

```
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
```

詳細については、AWS Command Line Interface ユーザーガイドの「AWS CLIでの Amazon EC2 の使用」を参照してください。

Amazon Glacier

Amazon Glacier および Amazon Glacier Deep Archive は、データのアーカイブおよび長期バックアップを行うための、安全で耐久性が高く、非常に低コストの Amazon S3 クラウドストレージクラスです。

次の `create-vault` コマンドは、アーカイブを保存するためのコンテナであるボールトを作成します。

```
aws glacier create-vault --vault-name my-vault --account-id -
```

詳細については、AWS Command Line Interface ユーザーガイドの「AWS CLIでの Amazon Glacier の使用」を参照してください。

AWS Elastic Beanstalk CLI

AWS Elastic Beanstalk CLI には、ローカルリポジトリからの環境の作成、更新、モニタリングを簡素化するためのコマンドラインインターフェイスが用意されています。このコンテキストでは、環境とは、アプリケーションバージョンを実行する AWS リソースのコレクションを指します。

次の `create` コマンドは、カスタム Amazon Virtual Private Cloud (VPC) に新しい環境を作成します。

```
$ eb create dev-vpc --vpc.id vpc-0ce8dd99 --vpc.elbsubnets subnet-
b356d7c6,subnet-02f74b0c --vpc.ec2subnets subnet-0bb7f0cd,subnet-3b6697c1 --
vpc.securitygroup sg-70cff265
```

詳細については、AWS Elastic Beanstalk デベロッパーガイドの「EB CLI コマンドレファレンス」を参照してください。

Amazon ECS CLI

Amazon Elastic Container Service (Amazon ECS) コマンドラインインターフェイス (CLI) には、いくつもの高レベルコマンドが用意されています。これらは、ローカル開発環境からのクラスターおよびタスクの作成、更新、モニタリングプロセスを簡素化する設計がされています。(Amazon ECS クラスターは、タスクまたはサービスの論理グループです。)

次の `configure` コマンドは、Amazon ECS CLI を設定して、`ecs-cli-demo` というクラスター設定を作成します。このクラスター構成では、`us-east-1 region` 内の `ecs-cli-demo` クラスターのデフォルト起動タイプとして `FARGATE` を使用します。

```
ecs-cli configure --region us-east-1 --cluster ecs-cli-demo --default-launch-type
 FARGATE --config-name ecs-cli-demo
```

詳細については、Amazon Elastic Container Service デベロッパーガイドの「Amazon ECS コマンドラインリファレンス」を参照してください。

AWS SAM CLI

AWS SAM CLI は、 AWS Serverless Application Model テンプレートとアプリケーションコードで動作するコマンドラインツールです。これを使用して複数のタスクが実行できます。これには、Lambda 関数のローカル呼び出し、サーバーレスアプリケーションのデプロイパッケージの作成、サーバーレスアプリケーションの AWS クラウドへのデプロイが含まれます。

次の `init` コマンド は、パラメータとして渡された必須パラメータを使用して、新しい SAM プロジェクトを初期化します。

```
sam init --runtime python3.9 --dependency-manager pip --app-template hello-world --name
 sam-app
```

詳細については、AWS Serverless Application Model デベロッパーガイドの「AWS SAM CLI コマンドレファレンス」を参照してください。

CloudShell での Kiro CLI の使用

Kiro CLI は、Kiro とやり取りできるコマンドラインインターフェイスです。詳細については、Kiro ユーザーガイドの「Kiro CLI の主な機能」を参照してください。

CloudShell の Kiro CLI を使用すると、自然言語の会話でやり取りしたり、質問をしたり、ターミナルから Kiro からレスポンスを受信したりできます。関連するシェルコマンドを取得できるため、検索や構文の記憶の必要性が減り、ターミナルに入力するときにコマンド候補を受け取ることができます。

CloudShell に Kiro CLI 機能が表示されない場合は、管理者に連絡して IAM アクセス許可を付与してください。詳細については、Kiro ユーザーガイドの「Kiro Developer のアイデンティティベースのポリシーの例」を参照してください。

この章では、CloudShell で Kiro CLI 機能を使用する方法について説明します。

一部の Kiro CLI 機能には認証が必要です。詳細については、「 Kiro ユーザーガイド」の「認証」を参照してください。

CloudShell での Kiro チャットコマンドの使用

`kiro-cli` コマンドを使用すると、Kiro から質問したり、ターミナルからレスポンスを受信したりできます。Kiro との会話を開始するには、CloudShell ターミナルで `kiro-cli` コマンドを実行します。詳細については、「 Kiro ユーザーガイド」の「 CLI での Kiro とのチャット」を参照してください。

CloudShell での Kiro translate コマンドの使用

`kiro-cli translate` コマンドを使用すると、自然言語で指示を記述できます。Kiro で翻訳するには、CloudShell ターミナルで `kiro-cli translate` コマンドを実行します。詳細については、Kiro ユーザーガイドの「自然言語から bash への翻訳」を参照してください。

CloudShell での CLI コマンドの完了

CloudShell で CLI を完了すると、ターミナルに入力する際のコマンドとオプションの提案が提供されます。詳細については、Kiro ユーザーガイドの「コマンドライン補完の生成」を参照してください。

CloudShell での Kiro インライン提案の使用

CloudShell の Kiro インライン提案では、ターミナルに入力するときにコマンド提案が提供されます。詳細については、Kiro ユーザーガイドのコマンドラインで Kiro inline を参照してください。

CloudShell で Kiro インライン提案を使用するには

1. から AWS マネジメントコンソール、CloudShell を選択します。
2. CloudShell ターミナルで、Z シェルに切り替え、入力を開始します。Z シェルに切り替えるには、ターミナルで「`zsh`」と入力し、Enter キーを押します。

   > **Note**
   >
   > 現在、Kiro インラインは Z シェルでのみサポートされています。

   コマンドの入力を開始すると、Kiro は現在の入力と以前のコマンドに基づいて提案を行います。インライン提案は自動的に有効になります。

インライン提案を無効にするには、次のコマンドを実行します。

```
kiro-cli inline disable
```

インライン提案を有効にするには、次のコマンドを実行します。

```
kiro-cli inline enable
```

CloudShell での Kiro CLI のアイデンティティベースのポリシー

CloudShell で Kiro CLI を使用するには、必要な IAM アクセス許可があることを確認してください。詳細については、Kiro ユーザーガイドの「Kiro Developer のアイデンティティベースのポリシーの例」を参照してください。

AWS サービスコンソールから CloudShell でコマンドを実行する

CloudShell ターミナルでコマンドを実行するには、 AWS マネジメントコンソールの Amazon ElastiCache および Amazon DocumentDB (MongoDB 互換) コンソールを使用します。

他の AWS Service コンソールから CloudShell でコマンドを実行するには、ロールに割り当てられた IAM ポリシーに `cloudshell:approveCommand` アクセス許可が含まれている必要があります。

[コンソールツールバー] で CloudShell が開き、CloudShell に [コマンドを実行] ポップアップが表示されます。[コマンドを実行] ポップアップで、コマンドがコマンドボックスに表示されます。

CloudShell ターミナルでコマンドを実行するには、次のいずれかの手順を選択します。

1. CloudShell で VPC 環境を作成していない場合は、[新しい環境名] ボックスに名前を入力します。

   リソースの VPC の詳細に基づく VPC 環境の詳細を表示できます。

   a. Create and run を選択します。

      このステップでは、新しい CloudShell VPC 環境を作成し、CloudShell ターミナルで コマンドを実行します。

2. CloudShell VPC 環境を既に作成している場合は、CloudShell 環境名を表示できます。

   > **Note**
   >
   > CloudShell VPC 環境がすでにある場合、新しい VPC 環境を作成することはできません。

   a. [Run] (実行) を選択します。

      このステップでは、選択した CloudShell VPC 環境の CloudShell ターミナルでコマンドを実行します。

      > **Note**
      >
      > 作成された VPC 環境を表示するアクセス許可がない場合は、管理者に連絡して `cloudshell:describeEnvironments` アクセス許可の追加を依頼してください。詳

細については、「IAM ポリシーを使用した AWS CloudShell アクセスと使用の管理」を参照してください。

CloudShell ターミナルでコマンドを引き続き実行できます。

AWS CloudShell エクスペリエンスのカスタマイズ

AWS CloudShell エクスペリエンスの以下の側面をカスタマイズできます。

- タブのレイアウト: コマンドラインインターフェイスを複数の列と行に分割します。
- フォントサイズ: コマンドラインテキストの文字サイズを調整します。
- カラーテーマ: 明るいテーマと暗いテーマを切り替えます。
- 安全な貼り付け: 複数行のテキストを貼り付ける前に確認を求める機能のオンとオフを切り替えます。
- tmux からセッションの復元: tmux を使用すると、非アクティブになるまでセッションが復元されます。

独自のソフトウェアをインストールしてスクリプトでシェルを変更すれば、シェル環境を拡張することもできます。

コマンドライン表示を複数のタブに分割する

コマンドラインインターフェースを複数のペインに分割して、複数のコマンドを実行します。

> **Note**
>
> 複数のタブを開いたら、選択したペイン内の任意の場所をクリックして、作業するタブを選択することができます。リージョン名の横にある x の記号を選択すると、タブを閉じることができます。

- [アクション] を選択し、[タブのレイアウト] から次のいずれかのオプションを選択します。
  - [新しいタブ] : 現在アクティブなタブの隣に新しいタブを追加します。
  - [行方向の分割] : 現在アクティブなタブの下の行に新しいタブを追加します。
  - 列方向の分割: 現在アクティブなタブの隣の列に新しいタブを追加します。

  各タブを完全に表示する十分なスペースがない場合は、スクロールするとタブ全体を見ることができます。ペインを分割する分割バーを選択し、ポインタを使用してドラッグし、ペインサイズを増減させることもできます。

フォントサイズを変更する

コマンドラインインターフェースに表示されるテキストのサイズを増減させます。

1. AWS CloudShell ターミナル設定を変更するには、「設定、設定」を参照してください。
2. テキストサイズを選択します。オプションには [最小]、[小]、[中]、[大]、[最大] があります。

インターフェイステーマの変更

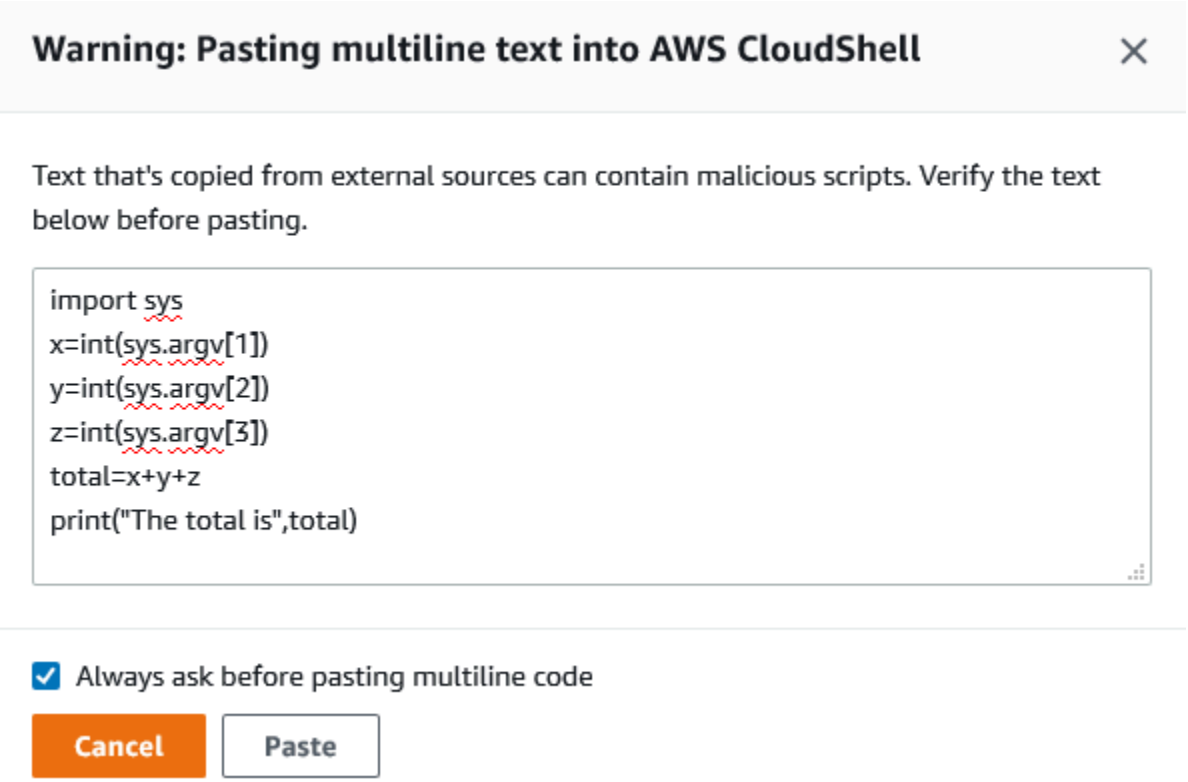
コマンドラインインターフェースでは、薄い色のテーマと濃い色のテーマを切り替えます。

1. AWS CloudShell テーマを変更するには、「設定、設定」を参照してください。
2. [Light (薄い色)] または [Dark (濃い色)] を選択します。

マルチテキストに安全な貼り付けを使用する

安全な貼り付けは、シェルに貼り付けようとしている複数の行のテキストに悪意のあるスクリプトが含まれていないことを確認するよう指示するセキュリティ機能です。サードパーティーのサイトからコピーされたテキストには、シェル環境で予期しない動作をトリガーする隠しコードが含まれている可能性があります。

安全な貼り付けダイアログには、クリップボードにコピーした完全なテキストが表示されます。セキュリティリスクがないことに十分に確認できたら、[貼り付ける] を選択します。



スクリプトで潜在的なセキュリティリスクを検出するには、安全な貼り付けを有効にすることをお勧めします。[プリファレンス] 、 [安全な貼り付けを有効にする] および [安全な貼り付けを無効にする] を選択し、この機能のオンとオフを切り替えることができます。

セッションの復元に tmux を使用する

AWS CloudShell は tmux を使用して、単一または複数のブラウザタブ間でセッションを復元します。ブラウザタブを更新すると、非アクティブになるまでセッションを回復します。詳細については、「セッションの復元」を参照してください。

Amazon VPC AWS CloudShell での の使用

AWS CloudShell Virtual Private Cloud (VPC) を使用すると、VPC に CloudShell 環境を作成できます。VPC 環境ごとに、VPC を割り当て、サブネットを追加し、最大 5 つのセキュリティグループを関連付けることができます。 は VPC のネットワーク設定を AWS CloudShell 継承し、VPC 内の他のリソースと同じサブネット内で AWS CloudShell を安全に使用して接続できるようにします。

Amazon VPC を使用すると、定義した論理的に分離された仮想ネットワークで AWS リソースを起動できます。仮想ネットワークは、お客様自身のデータセンターで運用されていた従来のネットワークによく似ていますが、 AWSのスケーラブルなインフラストラクチャを使用できるというメリットがあります。VPC の詳細については、「Amazon Virtual Private Cloud」を参照してください。

運用上の制約

AWS CloudShell VPC 環境には次の制約があります。

- IAM プリンシパルごとに作成できる VPC 環境の数は最大 2 つです。
- VPC 環境に割り当てることができるセキュリティグループの数は最大 5 つです。
- VPC 環境では、[アクション] メニューで CloudShell のアップロードおよびダウンロードオプションは使用できません。

> **Note**
>
> ファイルのアップロードまたはダウンロードは、他の CLI ツールを介して、インターネットの進入/退出にアクセスできる VPC 環境から行うことができます。

- VPC 環境は永続ストレージをサポートしていません。ストレージはエフェメラルです。アクティブな環境セッションが終了すると、データとホームディレクトリは削除されます。
- AWS CloudShell 環境は、プライベート VPC サブネットにある場合にのみインターネットに接続できます。

> **Note**
>
> デフォルトでは、パブリック IP アドレスは CloudShell の VPC 環境に割り当てられません。すべてのトラフィックをインターネットゲートウェイにルーティングするようにルーティングテーブルが設定されているパブリックサブネットに作成した VPC 環境は、パブリックインターネットにアクセスできません。しかし、ネットワークアドレス変換 (NAT)

が設定されているプライベートサブネットは、パブリックインターネットにアクセスできます。このようなプライベートサブネットに作成した VPC 環境は、パブリックインターネットにアクセスできます。

- デフォルトでは、 AWS CloudShell 環境は VPC 内のローカル DNS ゾーンを使用するように設定されません。
- アカウントにマネージド CloudShell 環境を提供するには、基盤となるコンピューティングホストに対して以下のサービスへのネットワークアクセスをプロビジョニング AWS できます。
  - Amazon S3
  - VPC エンドポイント
    - com.amazonaws.<リージョン>.ssmmessages
    - com.amazonaws.<リージョン>.logs
    - com.amazonaws.<リージョン>.kms
    - com.amazonaws.<リージョン>.execute-api
    - com.amazonaws.<リージョン>.ecs-telemetry
    - com.amazonaws.<リージョン>.ecs-agent
    - com.amazonaws.<リージョン>.ecs
    - com.amazonaws.<リージョン>.ecr.dkr
    - com.amazonaws.<リージョン>.ecr.api
    - com.amazonaws.<リージョン>.codecatalyst.packages
    - com.amazonaws.<リージョン>.codecatalyst.git
    - aws.api.global.codecatalyst

  VPC 設定を変更して、これらのエンドポイントへのアクセスを制限することはできません。

  CloudShell VPC は、すべての AWS リージョンと GovCloud リージョンで利用できます。CloudShell VPC が利用可能なリージョンのリストについては、「 でサポートされている AWS リージョン AWS CloudShell」を参照してください。

CloudShell の VPC 環境を作成する

このトピックでは、CloudShell の VPC 環境を作成する手順について説明します。

前提条件

VPC 環境を作成するために必要な IAM アクセス許可を管理者から取得する必要があります。CloudShell の VPC 環境を作成するためのアクセス許可の有効化の詳細については、「the section called “CloudShell の VPC 環境を作成および使用するために必要な IAM アクセス許可”」を参照してください。

CloudShell の VPC 環境を作成するには

1. CloudShell コンソールページで、[+] アイコンを選択し、ドロップダウンメニューから [VPC 環境を作成] を選択します。
2. [VPC 環境を作成] ページで、VPC 環境の名前を [名前] ボックスに入力します。
3. [仮想プライベートクラウド (VPC)] ドロップダウンリストから、VPC を選択します。
4. [サブネット] ドロップダウンリストから、サブネットを選択します。
5. [セキュリティグループ] ドロップダウンリストから、VPC 環境に割り当てるセキュリティグループを 1 つ以上選択します。

   > **Note**
   >
   > 最大 5 つのセキュリティグループを選択できます。

6. [作成] を選択して VPC 環境を作成します。
7. (オプション) [アクション]、[詳細を表示] の順に選択し、新しく作成した VPC 環境の詳細を確認します。VPC 環境の IP アドレスがコマンドラインプロンプトに表示されます。

VPC 環境の使用方法については、「開始方法」を参照してください。

CloudShell の VPC 環境を作成および使用するために必要な IAM アクセス許可

CloudShell の VPC 環境を作成して使用するには、IAM 管理者が VPC 固有の Amazon EC2 アクセス許可へのアクセスを有効にする必要があります。このセクションでは、VPC 環境の作成と使用に必要な Amazon EC2 アクセス許可を一覧表示します。

VPC 環境を作成するには、ロールに割り当てる IAM ポリシーに、以下の Amazon EC2 アクセス許可を含める必要があります。

- `ec2:DescribeVpcs`
- `ec2:DescribeSubnets`

- ec2:DescribeSecurityGroups
- ec2:DescribeDhcpOptions
- ec2:DescribeNetworkInterfaces

- ec2:CreateTags
- ec2:CreateNetworkInterface
- ec2:CreateNetworkInterfacePermission

以下を含めることをお勧めします。

- ec2:DeleteNetworkInterface

> **Note**
>
> このアクセス許可は必須ではありませんが、CloudShell で作成した ENI リソース (CloudShell の VPC 環境用に作成した ENI には ManagedByCloudShell キーのタグが付きます) をクリーンアップするために必要です。このアクセス許可が有効になっていない場合は、CloudShell の各 VPC 環境の使用後に ENI リソースを手動でクリーンアップする必要があります。

CloudShell へのフルアクセス (VPC へのアクセスを含む) を許可する IAM ポリシー

次の例は、CloudShell へのフルアクセス (VPC へのアクセスを含む) を許可する方法を示しています。

JSON

```
{
 "Version":"2012-10-17",
 "Statement": [
  {
   "Sid": "AllowCloudShellOperations",
   "Effect": "Allow",
   "Action": [
```

```
    "cloudshell:*"
   ],
   "Resource": "*"
  },
  {
   "Sid": "AllowDescribeVPC",
   "Effect": "Allow",
   "Action": [
    "ec2:DescribeSubnets",
    "ec2:DescribeSecurityGroups",
    "ec2:DescribeVpcs"
   ],
   "Resource": "*"
  },
  {
   "Sid": "AllowInspectVPCConfigurationViaCloudShell",
   "Effect": "Allow",
   "Action": [
    "ec2:DescribeDhcpOptions",
    "ec2:DescribeNetworkInterfaces"
   ],
   "Resource": "*",
   "Condition": {
    "ForAnyValue:StringEquals": {
     "aws:CalledVia": "cloudshell.amazonaws.com"
    }
   }
  },
  {
   "Sid": "AllowCreateTagWithCloudShellKeyViaCloudShell",
   "Effect": "Allow",
   "Action": [
    "ec2:CreateTags"
   ],
   "Resource": "arn:aws:ec2:*:*:network-interface/*",
   "Condition": {
    "StringEquals": {
     "ec2:CreateAction": "CreateNetworkInterface"
    },
    "ForAnyValue:StringEquals": {
     "aws:TagKeys": "ManagedByCloudShell",
     "aws:CalledVia": "cloudshell.amazonaws.com"
    }
   }
```

```
    },
    {
     "Sid": "AllowCreateNetworkInterfaceWithSubnetsAndSGViaCloudShell",
     "Effect": "Allow",
     "Action": [
      "ec2:CreateNetworkInterface"
     ],
     "Resource": [
      "arn:aws:ec2:*:*:subnet/*",
      "arn:aws:ec2:*:*:security-group/*"
     ],
     "Condition": {
      "ForAnyValue:StringEquals": {
       "aws:CalledVia": "cloudshell.amazonaws.com"
      }
     }
    },
    {
     "Sid": "AllowCreateNetworkInterfaceWithCloudShellTagViaCloudShell",
     "Effect": "Allow",
     "Action": [
      "ec2:CreateNetworkInterface"
     ],
     "Resource": "arn:aws:ec2:*:*:network-interface/*",
     "Condition": {
      "ForAnyValue:StringEquals": {
       "aws:TagKeys": "ManagedByCloudShell",
       "aws:CalledVia": "cloudshell.amazonaws.com"
      }
     }
    },
    {
     "Sid": "AllowCreateNetworkInterfacePermissionWithCloudShellTagViaCloudShell",
     "Effect": "Allow",
     "Action": [
      "ec2:CreateNetworkInterfacePermission"
     ],
     "Resource": "arn:aws:ec2:*:*:network-interface/*",
     "Condition": {
      "StringEquals": {
       "aws:ResourceTag/ManagedByCloudShell": ""
      },
      "ForAnyValue:StringEquals": {
       "aws:CalledVia": "cloudshell.amazonaws.com"
```

```
    }
   }
  },
  {
   "Sid": "AllowDeleteNetworkInterfaceWithCloudShellTagViaCloudShell",
   "Effect": "Allow",
   "Action": [
    "ec2:DeleteNetworkInterface"
   ],
   "Resource": "arn:aws:ec2:*:*:network-interface/*",
   "Condition": {
    "StringEquals": {
     "aws:ResourceTag/ManagedByCloudShell": ""
    },
    "ForAnyValue:StringEquals": {
     "aws:CalledVia": "cloudshell.amazonaws.com"
    }
   }
  }
 ]
}
```

VPC 環境での IAM 条件キーの使用

VPC 設定で CloudShell 固有の条件キーを使用して、VPC 環境に追加のアクセス許可コントロールを提供できます。また、VPC 環境で使用できるサブネットとセキュリティグループ、および使用できないサブネットとセキュリティグループを指定することもできます。

CloudShell は IAM ポリシーで以下の条件キーをサポートしています。

- `CloudShell:VpcIds` – 1 つ以上の VPC を許可または拒否する
- `CloudShell:SubnetIds` – 1 つ以上のサブネットを許可または拒否する
- `CloudShell:SecurityGroupIds` – 1 つ以上のセキュリティグループを許可または拒否する

> **Note**
>
> ユーザーに CloudShell のパブリック環境へのアクセス権がある場合、ユーザーのアクセス許可を変更して `cloudshell:createEnvironment` アクションに制限を追加しても、ユーザーは依然として既存のパブリック環境にアクセスできます。ただし、この制限を追加して

IAM ポリシーを変更し、既存のパブリック環境へのユーザーアクセスを無効にしたい場合は、まず、IAM ポリシーを更新して、この制限を含める必要があります。次に、アカウントのすべての CloudShell ユーザーに、CloudShell ウェブユーザーインターフェイスを使用して既存のパブリック環境を手動で確実に削除してもらいます ([アクション] → [CloudShell 環境を削除])。

VPC 設定の条件キーを使用したポリシーの例

以下の例は、VPC 設定で条件キーを使用する方法を示しています。必要な制限を含むポリシーステートメントを作成したら、このポリシーステートメントをターゲットのユーザーまたはロールに追加します。

ユーザーに VPC 環境の作成のみを許可し、パブリック環境の作成を拒否する

ユーザーに VPC 環境の作成のみを許可するには、次の例に示すように [拒否] アクセス許可を使用します。

```
{
  "Statement": [
    {
      "Sid": "DenyCloudShellNonVpcEnvironments",
      "Action": [
        "cloudshell:CreateEnvironment"
      ],
      "Effect": "Deny",
      "Resource": "*",
      "Condition": {
        "Null": {
          "cloudshell:VpcIds": "true"
        }
      }
    }
  ]
}
```

特定の VPC、サブネット、セキュリティグループへのアクセスをユーザーに拒否する

特定の VPC へのアクセスをユーザーに拒否するには、`StringEquals` を使用して `cloudshell:VpcIds` 条件の値を確認します。次の例では、`vpc-1` および `vpc-2` へのアクセスをユーザーに拒否します。

特定の VPC へのアクセスをユーザーに拒否するには、StringEquals を使用して cloudshell:SubnetIds 条件の値を確認します。次の例では、subnet-1 および subnet-2 へのアクセスをユーザーに拒否します。

特定の VPC へのアクセスをユーザーに拒否するには、StringEquals を使用して cloudshell:SecurityGroupIds 条件の値を確認します。次の例では、sg-1 および sg-2 へのアクセスをユーザーに拒否します。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "EnforceOutOfSecurityGroups",
      "Action": [
        "cloudshell:CreateEnvironment"
      ],
      "Effect": "Deny",
      "Resource": "*",
      "Condition": {
        "ForAnyValue:StringEquals": {
          "cloudshell:SecurityGroupIds": [
            "sg-1",
            "sg-2"
          ]
        }
      }
    }
  ]
}
```

特定の VPC 設定で環境を作成することをユーザーに許可する

特定の VPC にアクセスすることをユーザーに許可するには、StringEquals を使用して cloudshell:VpcIds 条件の値を確認します。次の例では、vpc-1 および vpc-2 にアクセスすることをユーザーに許可します。

特定の VPC にアクセスすることをユーザーに許可するには、StringEquals を使用して cloudshell:SubnetIds 条件の値を確認します。次の例では、subnet-1 および subnet-2 にアクセスすることをユーザーに許可します。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "EnforceStayInSpecificSubnets",
      "Action": [
        "cloudshell:CreateEnvironment"
      ],
      "Effect": "Allow",
      "Resource": "*",
      "Condition": {
        "ForAllValues:StringEquals": {
          "cloudshell:SubnetIds": [
            "subnet-1",
            "subnet-2"
          ]
        }
      }
    }
  ]
}
```

特定の VPC にアクセスすることをユーザーに許可するには、StringEquals を使用して cloudshell:SecurityGroupIds 条件の値を確認します。次の例では、sg-1 および sg-2 にアクセスすることをユーザーに許可します。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "EnforceStayInSpecificSecurityGroup",
      "Action": [
```

```
        "cloudshell:CreateEnvironment"
      ],
      "Effect": "Allow",
      "Resource": "*",
      "Condition": {
        "ForAllValues:StringEquals": {
          "cloudshell:SecurityGroupIds": [
            "sg-1",
            "sg-2"
          ]
        }
      }
    }
  ]
}
```

のセキュリティ AWS CloudShell

クラウドセキュリティは Amazon Web Services (AWS) の最優先事項です。お客様は AWS 、セキュリティを最も重視する組織の要件を満たすように構築されたデータセンターとネットワークアーキテクチャを活用できます。セキュリティは、 AWS お客様とお客様の間の責任共有です。責任共有モデルでは、これをクラウドのセキュリティおよびクラウド内のセキュリティとして説明しています。

クラウドのセキュリティ – AWS クラウドで提供されているすべてのサービスを実行するインフラストラクチャ AWS を保護し、安全に使用できるサービスを提供します。における当社のセキュリティ責任は最優先事項であり AWS、当社のセキュリティの有効性は、AWS コンプライアンスプログラムの一環として、サードパーティーの監査者によって定期的にテストおよび検証されます。

クラウド内のセキュリティ – お客様の責任は、使用している AWS サービス、データの機密性、組織の要件、適用される法律や規制などのその他の要因によって決まります。

AWS CloudShell は、サポートする特定の AWS サービスを通じて責任共有モデルに従います。AWS サービスセキュリティ情報については、AWS 「サービスセキュリティドキュメント」ページとAWS 、コンプライアンスプログラムによる AWS コンプライアンスの取り組みの対象となるサービスを参照してください。

以下のトピックでは、セキュリティおよびコンプライアンスの目的を達成する AWS CloudShell ように を設定する方法を示します。

トピック

- でのデータ保護 AWS CloudShell
- AWS CloudShell の Identity and Access Management
- でのログ記録とモニタリング AWS CloudShell
- のコンプライアンス検証 AWS CloudShell
- の耐障害性 AWS CloudShell
- のインフラストラクチャセキュリティ AWS CloudShell
- のセキュリティのベストプラクティス AWS CloudShell
- AWS CloudShell セキュリティFAQs

でのデータ保護 AWS CloudShell

AWS 責任共有モデル は、 AWS CloudShellでのデータ保護に適用されます。このモデルで説明されているように、 AWS はすべての を実行するグローバルインフラストラクチャを保護する責任があります AWS クラウド。ユーザーは、このインフラストラクチャでホストされるコンテンツに対する管理を維持する責任があります。また、使用する「 AWS のサービス 」のセキュリティ設定と管理タスクもユーザーの責任となります。データプライバシーの詳細については、「Data Privacy FAQChina」を参照してください。 欧州におけるデータ保護に関する情報については、General Data Protection Regulation (GDPR) Center を参照してください。

データ保護の目的で、認証情報を保護し AWS アカウント 、 AWS IAM アイデンティティセンターまたは AWS Identity and Access Management (IAM) を使用して個々のユーザーを設定することをお勧めします。この方法により、それぞれのジョブを遂行するために必要な権限のみが各ユーザーに付与されます。また、次の方法でデータを保護することもお勧めします:

- 各アカウントで多要素認証 (MFA) を使用します。
- SSL/TLS を使用して AWS リソースと通信します。TLS 1.2 は必須ですが、TLS 1.3 を推奨します。
- で API とユーザーアクティビティのログ記録を設定します AWS CloudTrail。CloudTrail 証跡を使用して AWS アクティビティをキャプチャする方法については、「 AWS CloudTrail ユーザーガイド」のCloudTrail 証跡の使用」を参照してください。
- AWS 暗号化ソリューションと、 内のすべてのデフォルトのセキュリティコントロールを使用します AWS のサービス。
- Amazon Macie などの高度な管理されたセキュリティサービスを使用します。これらは、Amazon S3 に保存されている機密データの検出と保護を支援します。
- コマンドラインインターフェイスまたは API AWS を介して にアクセスするときに FIPS 140-3 検証済み暗号化モジュールが必要な場合は、FIPS エンドポイントを使用します。利用可能な FIPS エンドポイントの詳細については、「連邦情報処理規格 (FIPS) 140-3」を参照してください。

お客様の E メールアドレスなどの極秘または機密情報を、タグ、または [名前] フィールドなどの自由形式のテキストフィールドに含めないことを強くお勧めします。これは、コンソール AWS CloudShell 、API、または SDK を使用して AWS CLIまたは他の AWS のサービス を操作する場合も同様です。 AWS SDKs タグ、または名前に使用される自由記述のテキストフィールドに入力したデータは、請求または診断ログに使用される場合があります。外部サーバーに URL を提供する場合、そのサーバーへのリクエストを検証できるように、認証情報を URL に含めないことを強くお勧めします。

データ暗号化

データ暗号化とは、保管中のデータを保護すること AWS CloudShell 、および転送中のデータが AWS CloudShell とサービスエンドポイント間を移動することを指します。

を使用した保管時の暗号化 AWS KMS

保存時の暗号化とは、保存中にデータを暗号化することで、不正なアクセスからデータを保護することです。を使用する場合 AWS CloudShell、 AWS リージョンあたり 1 GB の永続的ストレージを無料で使用できます。永続的ストレージはホームディレクトリ (`$HOME`) にあり、ユーザーのプライベートな記憶域です。各シェルセッションが終了した後にリサイクルされる エフェメラル環境リソースとは異なり、ホームディレクトリ内のデータは保持されます。

に保存されているデータの暗号化 AWS CloudShell は、 AWS Key Management Service () が提供する暗号化キーを使用して実装されますAWS KMS。これは、 AWS CloudShell 環境に保存されている顧客データの暗号化に使用される AWS KMS keys暗号化キーを作成および制御するためのマネージド AWS サービスです。 は、顧客に代わってデータを暗号化するための暗号化キー AWS CloudShell を生成および管理します。

転送中の暗号化

転送中の暗号化とは、通信エンドポイント間の移動中にデータが傍受されるのを防ぐことです。

デフォルトでは、クライアントのウェブブラウザコンピュータとクラウドベースの間のすべてのデータ通信 AWS CloudShell は、HTTPS/TLS 接続を介してすべてを送信することで暗号化されます。

コミュニケーションのために、HTTPS/TLS の使用を有効にするために必要な操作はありません。

AWS CloudShell の Identity and Access Management

AWS Identity and Access Management (IAM) は、管理者が AWS リソースへのアクセスを安全に制御 AWS のサービス するのに役立つ です。IAM 管理者は、誰を認証 (サインインを許可) できるかと、誰に CloudShell リソースの使用を認可 (アクセス許可を付与) できるかを制御します。IAM は、追加料金なしで使用できる AWS のサービス です。

トピック

- アイデンティティを使用した認証
- ポリシーを使用したアクセスの管理
- AWS CloudShell と IAM の連携方法
- AWS CloudShell のアイデンティティベースのポリシー例
- AWS CloudShell のアイデンティティとアクセスのトラブルシューティング
- IAM ポリシーによる AWS CloudShell アクセスと使用状況の管理

オーディエンス

AWS Identity and Access Management (IAM) の使用方法は、ロールによって異なります。

- サービスユーザー - 機能にアクセスできない場合は、管理者にアクセス許可をリクエストします (「AWS CloudShell のアイデンティティとアクセスのトラブルシューティング」を参照)。
- サービス管理者 - ユーザーアクセスを決定し、アクセス許可リクエストを送信します (「AWS CloudShell と IAM の連携方法」を参照)
- IAM 管理者 - アクセスを管理するためのポリシーを作成します (「AWS CloudShell のアイデンティティベースのポリシー例」を参照)

アイデンティティを使用した認証

認証とは、ID 認証情報 AWS を使用して にサインインする方法です。、IAM ユーザー AWS アカウントのルートユーザー、または IAM ロールを引き受けることで認証される必要があります。

AWS IAM アイデンティティセンター (IAM Identity Center)、シングルサインオン認証、Google/Facebook 認証情報などの ID ソースからの認証情報を使用して、フェデレーティッド ID としてサインインできます。サインインの詳細については、「AWS サインイン ユーザーガイド」の「AWS アカウントにサインインする方法」を参照してください。

プログラムによるアクセスの場合、 は SDK と CLI AWS を提供してリクエストを暗号化して署名します。詳細については、「IAM ユーザーガイド」の「API リクエストに対するAWS 署名バージョン 4」を参照してください。

AWS アカウント ルートユーザー

を作成するときは AWS アカウント、まず、すべての AWS のサービス および リソースへの完全なアクセス権を持つ AWS アカウント root ユーザーと呼ばれる 1 つのサインインアイデンティティか

ら始めます。日常的なタスクには、ルートユーザーを使用しないことを強くお勧めします。ルートユーザー認証情報を必要とするタスクについては、「IAM ユーザーガイド」の「ルートユーザー認証情報が必要なタスク」を参照してください。

フェデレーテッドアイデンティティ

ベストプラクティスとして、人間のユーザーが一時的な認証情報 AWS のサービス を使用して にアクセスするには、ID プロバイダーとのフェデレーションを使用する必要があります。

フェデレーティッド ID は、エンタープライズディレクトリ、ウェブ ID プロバイダー、または ID Directory Service ソースからの認証情報 AWS のサービス を使用して にアクセスするユーザーです。フェデレーテッドアイデンティティは、一時的な認証情報を提供するロールを引き受けます。

アクセスを一元管理する場合は、 AWS IAM アイデンティティセンターをお勧めします。詳細については、「AWS IAM アイデンティティセンター ユーザーガイド」の「IAM アイデンティティセンターとは」を参照してください。

IAM ユーザーとグループ

IAM ユーザーは、特定の個人やアプリケーションに対する特定のアクセス許可を持つアイデンティティです。長期認証情報を持つ IAM ユーザーの代わりに一時的な認証情報を使用することをお勧めします。詳細については、IAM ユーザーガイドの「ID プロバイダーとのフェデレーションを使用して にアクセスする必要がある AWS」を参照してください。

IAM グループは、IAM ユーザーの集合を指定し、大量のユーザーに対するアクセス許可の管理を容易にします。詳細については、「IAM ユーザーガイド」の「IAM ユーザーに関するユースケース」を参照してください。

IAM ロール

IAM ロールは、特定のアクセス許可を持つアイデンティであり、一時的な認証情報を提供します。ユーザーから IAM ロール (コンソール) に切り替えるか、 または API オペレーションを呼び出すことで、ロールを引き受けることができます。 AWS CLI AWS 詳細については、「IAM ユーザーガイド」の「ロールを引き受けるための各種方法」を参照してください。

IAM ロールは、フェデレーションユーザーアクセス、一時的な IAM ユーザーのアクセス許可、クロスアカウントアクセス、クロスサービスアクセス、および Amazon EC2 で実行するアプリケーションに役立ちます。詳細については、IAM ユーザーガイド の IAM でのクロスアカウントリソースアクセス を参照してください。

ポリシーを使用したアクセスの管理

でアクセスを制御する AWS には、ポリシーを作成し、ID AWS またはリソースにアタッチします。ポリシーは、ID またはリソースに関連付けられている場合のアクセス許可を定義します。 は、プリンシパルがリクエストを行うときにこれらのポリシー AWS を評価します。ほとんどのポリシーは JSON ドキュメント AWS として に保存されます。JSON ポリシードキュメントの詳細については、「IAM ユーザーガイド」の「JSON ポリシー概要」を参照してください。

管理者は、ポリシーを使用して、どのプリンシパルがどのリソースに対して、どのような条件でアクションを実行できるかを定義することで、誰が何にアクセスできるかを指定します。

デフォルトでは、ユーザーやロールにアクセス許可はありません。IAM 管理者は IAM ポリシーを作成してロールに追加し、このロールをユーザーが引き受けられるようにします。IAM ポリシーは、オペレーションの実行方法を問わず、アクセス許可を定義します。

アイデンティティベースのポリシー

アイデンティティベースのポリシーは、アイデンティティ (ユーザー、グループ、またはロール) にアタッチできる JSON アクセス許可ポリシードキュメントです。これらのポリシーは、アイデンティティがどのリソースに対してどのような条件下でどのようなアクションを実行できるかを制御します。アイデンティティベースポリシーの作成方法については、IAM ユーザーガイド の カスタマー管理ポリシーでカスタム IAM アクセス許可を定義する を参照してください。

アイデンティティベースのポリシーは、インラインポリシー (単一の ID に直接埋め込む) または管理ポリシー (複数の ID にアタッチされたスタンドアロンポリシー) にすることができます。管理ポリシーとインラインポリシーのいずれかを選択する方法については、「IAM ユーザーガイド」の「管理ポリシーとインラインポリシーのいずれかを選択する」を参照してください。

リソースベースのポリシー

リソースベースのポリシーは、リソースに添付する JSON ポリシードキュメントです。例としては、IAM ロール信頼ポリシーや Amazon S3 バケットポリシーなどがあります。リソースベースのポリシーをサポートするサービスでは、サービス管理者はポリシーを使用して特定のリソースへのアクセスを制御できます。リソースベースのポリシーでは、プリンシパルを指定する必要があります。

リソースベースのポリシーは、そのサービス内にあるインラインポリシーです。リソースベースのポリシーでは、IAM の AWS マネージドポリシーを使用できません。

その他のポリシータイプ

AWS は、より一般的なポリシータイプによって付与されるアクセス許可の最大数を設定できる追加のポリシータイプをサポートしています。

- アクセス許可の境界 – アイデンティティベースのポリシーで IAM エンティティに付与することのできるアクセス許可の数の上限を設定します。詳細については、「IAM ユーザーガイド」の「IAM エンティティのアクセス許可境界」を参照してください。
- サービスコントロールポリシー (SCP) - AWS Organizations内の組織または組織単位の最大のアクセス許可を指定します。詳細については、「AWS Organizations ユーザーガイド」の「サービスコントロールポリシー」を参照してください。
- リソースコントロールポリシー (RCP) – は、アカウント内のリソースで利用できる最大数のアクセス許可を定義します。詳細については、「AWS Organizations ユーザーガイド」の「リソースコントロールポリシー (RCP)」を参照してください。
- セッションポリシー – ロールまたはフェデレーションユーザーの一時セッションを作成する際にパラメータとして渡される高度なポリシーです。詳細については、「IAM ユーザーガイド」の「セッションポリシー」を参照してください。

複数のポリシータイプ

1 つのリクエストに複数のタイプのポリシーが適用されると、結果として作成されるアクセス許可を理解するのがさらに難しくなります。が複数のポリシータイプが関与する場合にリクエストを許可するかどうか AWS を決定する方法については、「IAM ユーザーガイド」の「ポリシー評価ロジック」を参照してください。

AWS CloudShell と IAM の連携方法

IAM を使用して CloudShell へのアクセスを管理する前に、CloudShell で利用できる IAM の機能を確認します。

AWS CloudShell で利用できる IAM の機能

IAM 機能	CloudShell のサポート
アイデンティティベースのポリシー	あり
リソースベースのポリシー	なし

IAM 機能	CloudShell のサポート
ポリシーアクション	あり
ポリシーリソース	はい
ポリシー条件キー (サービス固有)	はい
ACL	なし
ABAC (ポリシー内のタグ)	いいえ
一時的な認証情報	あり
転送アクセスセッション (FAS)	いいえ
サービスロール	いいえ
サービスリンクロール	いいえ

CloudShell およびその他の AWS のサービスがほとんどの IAM 機能と連携する方法の概要については、「IAM ユーザーガイド」のAWS 「IAM と連携する のサービス」を参照してください。

CloudShell のアイデンティティベースのポリシー

アイデンティティベースのポリシーのサポート: あり

アイデンティティベースポリシーは、IAM ユーザー、ユーザーグループ、ロールなど、アイデンティティにアタッチできる JSON 許可ポリシードキュメントです。これらのポリシーは、ユーザーとロールが実行できるアクション、リソース、および条件をコントロールします。アイデンティティベースポリシーの作成方法については、「IAM ユーザーガイド」の「カスタマー管理ポリシーでカスタム IAM アクセス許可を定義する」を参照してください。

IAM アイデンティティベースのポリシーでは、許可または拒否するアクションとリソース、およびアクションを許可または拒否する条件を指定できます。JSON ポリシーで使用できるすべての要素について学ぶには、「IAM ユーザーガイド」の「IAM JSON ポリシーの要素のリファレンス」を参照してください。

CloudShell のアイデンティティベースのポリシー例

CloudShell のアイデンティティベースのポリシー例については、「AWS CloudShell のアイデンティティベースのポリシー例」を参照してください。

CloudShell 内のリソースベースのポリシー

リソースベースのポリシーのサポート: なし

リソースベースのポリシーは、リソースに添付する JSON ポリシードキュメントです。リソースベースのポリシーには例として、IAM ロールの信頼ポリシー や Amazon S3 バケットポリシー があげられます。リソースベースのポリシーをサポートするサービスでは、サービス管理者はポリシーを使用して特定のリソースへのアクセスをコントロールできます。ポリシーがアタッチされているリソースの場合、指定されたプリンシパルがそのリソースに対して実行できるアクションと条件は、ポリシーによって定義されます。リソースベースのポリシーで、プリンシパルを指定する必要があります。プリンシパルには、アカウント、ユーザー、ロール、フェデレーティッドユーザー、または を含めることができます AWS のサービス。

クロスアカウントアクセスを有効にするには、全体のアカウント、または別のアカウントの IAM エンティティを、リソースベースのポリシーのプリンシパルとして指定します。詳細については、IAM ユーザーガイドのIAM でのクロスアカウントリソースアクセスを参照してください。

CloudShell のポリシーアクション

ポリシーアクションのサポート: あり

管理者は JSON AWS ポリシーを使用して、誰が何にアクセスできるかを指定できます。つまり、どのプリンシパルがどのリソースに対してどのような条件下でアクションを実行できるかということです。

JSON ポリシーの `Action` 要素にはポリシー内のアクセスを許可または拒否するために使用できるアクションが記述されます。このアクションは関連付けられたオペレーションを実行するためのアクセス許可を付与するポリシーで使用されます。

CloudShell アクションのリストを確認するには、「サービス認可リファレンス」の「AWS CloudShell で定義されるアクション」を参照してください。一部のアクションには複数の API が含まれている場合があります。

CloudShell のポリシーアクションは、アクションの前に次のプレフィックスを使用します。

```
cloudshell
```

単一のステートメントで複数のアクションを指定するには、アクションをカンマで区切ります。

```
"Action": [
      "cloudshell:action1",
      "cloudshell:action2"
         ]
```

CloudShell のアイデンティティベースのポリシー例については、「AWS CloudShell のアイデンティティベースのポリシー例」を参照してください。

CloudShell のポリシーリソース

ポリシーリソースのサポート: あり

管理者は JSON AWS ポリシーを使用して、誰が何にアクセスできるかを指定できます。つまり、どのプリンシパルがどのリソースに対してどのような条件下でアクションを実行できるかということです。

`Resource` JSON ポリシー要素はアクションが適用されるオブジェクトを指定します。ベストプラクティスとして、Amazon リソースネーム (ARN) を使用してリソースを指定します。リソースレベルのアクセス許可をサポートしないアクションの場合は、ステートメントがすべてのリソースに適用されることを示すために、ワイルドカード (*) を使用します。

```
"Resource": "*"
```

CloudShell リソースのタイプとその ARN のリストを確認するには、「サービス認可リファレンス」の「AWS CloudShell で定義されるリソース」を参照してください。どのアクションで各リソースの ARN を指定できるかについては、「AWS CloudShell で定義されるアクション」を参照してください。

CloudShell のアイデンティティベースのポリシー例については、「AWS CloudShell のアイデンティティベースのポリシー例」を参照してください。

CloudShell のポリシー条件キー

サービス固有のポリシー条件キーのサポート: あり

管理者は JSON AWS ポリシーを使用して、誰が何にアクセスできるかを指定できます。つまり、どのプリンシパルがどのリソースに対してどのような条件下でアクションを実行できるかということです。

Condition 要素は、定義された基準に基づいてステートメントが実行される時期を指定します。イコールや未満などの条件演算子を使用して条件式を作成して、ポリシーの条件とリクエスト内の値を一致させることができます。すべての AWS グローバル条件キーを確認するには、「IAM ユーザーガイド」のAWS 「グローバル条件コンテキストキー」を参照してください。

CloudShell の条件キーのリストを確認するには、「サービス認可リファレンス」の「AWS CloudShell の条件キー」を参照してください。どのアクションやリソースで条件キーを使用できるかについては、「AWS CloudShell で定義されるアクション」を参照してください。

CloudShell のアイデンティティベースのポリシー例については、「AWS CloudShell のアイデンティティベースのポリシー例」を参照してください。

CloudShell の ACL

ACL のサポート: なし

アクセスコントロールリスト (ACL) は、どのプリンシパル (アカウントメンバー、ユーザー、またはロール) がリソースにアクセスするためのアクセス許可を持つかを制御します。ACL はリソースベースのポリシーに似ていますが、JSON ポリシードキュメント形式は使用しません。

CloudShell による ABAC

ABAC (ポリシー内のタグ) のサポート: なし

属性ベースのアクセス制御 (ABAC) は、タグと呼ばれる属性に基づいてアクセス許可を定義する認可戦略です。IAM エンティティと AWS リソースにタグをアタッチし、プリンシパルのタグがリソースのタグと一致するときにオペレーションを許可するように ABAC ポリシーを設計できます。

タグに基づいてアクセスを管理するには、aws:ResourceTag/*key-name*、aws:RequestTag/*key-name*、または aws:TagKeys の条件キーを使用して、ポリシーの条件要素でタグ情報を提供します。

サービスがすべてのリソースタイプに対して 3 つの条件キーすべてをサポートする場合、そのサービスの値はありです。サービスが一部のリソースタイプに対してのみ 3 つの条件キーのすべてをサポートする場合、値は「部分的」になります。

ABAC の詳細については、「IAM ユーザーガイド」の「ABAC 認可でアクセス許可を定義する」を参照してください。ABAC をセットアップする手順を説明するチュートリアルについては、「IAM ユーザーガイド」の「属性ベースのアクセスコントロール (ABAC) を使用する」を参照してください。

CloudShell での一時的な認証情報の使用

一時的な認証情報のサポート: あり

一時的な認証情報は、 AWS リソースへの短期的なアクセスを提供し、フェデレーションまたはスイッチロールの使用時に自動的に作成されます。長期的なアクセスキーを使用する代わりに、一時的な認証情報を動的に生成 AWS することをお勧めします。詳細については、「IAM ユーザーガイド」の「IAM の一時的な認証情報」および「AWS のサービス と IAM との連携」を参照してください。

ロールを切り替えると、別の環境を使用することになります。同じ AWS CloudShell 環境内でロールを切り替えることはできません。

CloudShell の転送アクセスセッション

転送アクセスセッション (FAS) のサポート: なし

転送アクセスセッション (FAS) は、 を呼び出すプリンシパルのアクセス許可と AWS のサービス、ダウンストリームサービス AWS のサービス へのリクエストをリクエストする を使用します。FAS リクエストを行う際のポリシーの詳細については、「転送アクセスセッション」を参照してください。

CloudShell のサービスロール

サービスロールのサポート: なし

サービスロールとは、サービスがユーザーに代わってアクションを実行するために引き受ける IAM ロールです。IAM 管理者は、IAM 内からサービスロールを作成、変更、削除できます。詳細については、IAM ユーザーガイドの AWS のサービスに許可を委任するロールを作成するを参照してください。

> **Warning**
>
> サービスロールのアクセス許可を変更すると、CloudShell の機能が破損する可能性があります。CloudShell が指示する場合以外は、サービスロールを編集しないでください。

CloudShell のサービスリンクロール

サービスにリンクされたロールのサポート: なし

サービスにリンクされたロールは、 にリンクされたサービスロールの一種です AWS のサービス。サービスは、ユーザーに代わってアクションを実行するロールを引き受けることができます。サービ

スにリンクされたロールは に表示され AWS アカウント 、サービスによって所有されます。IAM 管理者は、サービスリンクロールのアクセス許可を表示できますが、編集することはできません。

AWS CloudShell のアイデンティティベースのポリシー例

デフォルトでは、ユーザーおよびロールには、CloudShell リソースを作成または変更するアクセス許可がありません。IAM 管理者は、リソースで必要なアクションを実行するための権限をユーザーに付与する IAM ポリシーを作成できます。

これらのサンプルの JSON ポリシードキュメントを使用して IAM アイデンティティベースのポリシーを作成する方法については、「IAM ユーザーガイド」の「IAM ポリシーを作成する (コンソール)」を参照してください。

CloudShell が定義するアクションとリソースタイプ (リソースタイプごとの ARN の形式を含む) の詳細については、「サービス認可リファレンス」の「AWS CloudShell のアクション、リソース、および条件キー」を参照してください。

トピック

- ポリシーに関するベストプラクティス
- CloudShell コンソールの使用
- 自分の権限の表示をユーザーに許可する

ポリシーに関するベストプラクティス

アイデンティティベースのポリシーは、お客様のアカウント内で誰かが CloudShell リソースを作成、アクセス、または削除できるかどうかを決定します。これらのアクションでは、 AWS アカウントに費用が発生する場合があります。アイデンティティベースポリシーを作成したり編集したりする際には、以下のガイドラインと推奨事項に従ってください:

- AWS 管理ポリシーを開始し、最小特権のアクセス許可に移行する – ユーザーとワークロードにアクセス許可の付与を開始するには、多くの一般的なユースケースにアクセス許可を付与するAWS 管理ポリシーを使用します。これらは で使用できます AWS アカウント。ユースケースに固有の AWS カスタマー管理ポリシーを定義することで、アクセス許可をさらに減らすことをお勧めします。詳細については、IAM ユーザーガイド の AWS マネージドポリシー または ジョブ機能のAWS マネージドポリシー を参照してください。
- 最小特権を適用する – IAM ポリシーでアクセス許可を設定する場合は、タスクの実行に必要な許可のみを付与します。これを行うには、特定の条件下で特定のリソースに対して実行できるアクションを定義します。これは、最小特権アクセス許可とも呼ばれています。IAM を使用して許可

を適用する方法の詳細については、IAM ユーザーガイド の IAM でのポリシーとアクセス許可 を参照してください。

- IAM ポリシーで条件を使用してアクセスをさらに制限する - ポリシーに条件を追加して、アクションやリソースへのアクセスを制限できます。たとえば、ポリシー条件を記述して、すべてのリクエストを SSL を使用して送信するように指定できます。条件を使用して、サービスアクションが などの特定の を通じて使用されている場合に AWS のサービス、サービスアクションへのアクセスを許可することもできます CloudFormation。詳細については、IAM ユーザーガイド の IAM JSON ポリシー要素:条件 を参照してください。
- IAM アクセスアナライザー を使用して IAM ポリシーを検証し、安全で機能的な権限を確保する - IAM アクセスアナライザー は、新規および既存のポリシーを検証して、ポリシーが IAM ポリシー言語 (JSON) および IAM のベストプラクティスに準拠するようにします。IAM アクセスアナライザーは 100 を超えるポリシーチェックと実用的な推奨事項を提供し、安全で機能的なポリシーの作成をサポートします。詳細については、IAM ユーザーガイド の IAM Access Analyzer でポリシーを検証する を参照してください。
- 多要素認証 (MFA) を要求する – で IAM ユーザーまたはルートユーザーを必要とするシナリオがある場合は AWS アカウント、MFA をオンにしてセキュリティを強化します。API オペレーションが呼び出されるときに MFA を必須にするには、ポリシーに MFA 条件を追加します。詳細については、IAM ユーザーガイド の MFA を使用した安全な API アクセス を参照してください。

IAM でのベストプラクティスの詳細については、IAM ユーザーガイド の IAM でのセキュリティのベストプラクティス を参照してください。

CloudShell コンソールの使用

AWS CloudShell コンソールにアクセスするには、最小限のアクセス許可のセットが必要です。これらのアクセス許可により、 の CloudShell リソースの詳細を一覧表示および表示できます AWS アカウント。最小限必要な許可よりも制限が厳しいアイデンティティベースのポリシーを作成すると、そのポリシーを持つエンティティ (ユーザーまたはロール) に対してコンソールが意図したとおりに機能しません。

AWS CLI または AWS API のみを呼び出すユーザーには、最小限のコンソールアクセス許可を付与する必要はありません。代わりに、実行しようとしている API オペレーションに一致するアクションのみへのアクセスが許可されます。

ユーザーとロールが引き続き CloudShell コンソールを使用できるようにするには、エンティティに CloudShell *ConsoleAccess*または *ReadOnly* AWS 管理ポリシーもアタッチします。詳細については、「IAM ユーザーガイド」の「ユーザーへのアクセス許可の追加」を参照してください。

自分の権限の表示をユーザーに許可する

この例では、ユーザーアイデンティティにアタッチされたインラインおよびマネージドポリシーの表示を IAM ユーザーに許可するポリシーの作成方法を示します。このポリシーには、コンソールで、または AWS CLI または AWS API を使用してプログラムでこのアクションを実行するアクセス許可が含まれています。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

AWS CloudShell のアイデンティティとアクセスのトラブルシューティング

以下の情報は、CloudShell と IAM の使用時に発生する可能性がある一般的な問題の診断や修復に役立ちます。

トピック

CloudShell でアクションを実行する権限がありません

アクションを実行する権限がないというエラーが表示された場合は、そのアクションを実行できるようにポリシーを更新する必要があります。

次のエラー例は、mateojackson IAM ユーザーがコンソールを使用して、ある *my-example-widget* リソースに関する詳細情報を表示しようとしたことを想定して、その際に必要な awes:*GetWidget* アクセス許可を持っていない場合に発生するものです。

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

この場合、awes:*GetWidget* アクションを使用して *my-example-widget* リソースへのアクセスを許可するように、mateojackson ユーザーのポリシーを更新する必要があります。

サポートが必要な場合は、 AWS 管理者にお問い合わせください。サインイン認証情報を提供した担当者が管理者です。

iam:PassRole を実行する権限がありません

iam:PassRole アクションを実行する権限がないというエラーが表示された場合は、ポリシーを更新して CloudShell にロールを渡せるようにする必要があります。

一部の AWS のサービス では、新しいサービスロールまたはサービスにリンクされたロールを作成する代わりに、そのサービスに既存のロールを渡すことができます。そのためには、サービスにロールを渡すアクセス許可が必要です。

次の例に示すエラーは、marymajor という名前の IAM ユーザーがコンソールを使用して CloudShell でアクションを実行しようとした場合に発生します。ただし、このアクションをサービ

スが実行するには、サービスロールから付与されたアクセス許可が必要です。Mary には、ロールをサービスに渡すアクセス許可がありません。

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

この場合、Mary のポリシーを更新してメアリーに `iam:PassRole` アクションの実行を許可する必要があります。

サポートが必要な場合は、 AWS 管理者にお問い合わせください。サインイン資格情報を提供した担当者が管理者です。

自分の 以外のユーザーに CloudShell リソース AWS アカウント へのアクセスを許可したい

他のアカウントのユーザーや組織外の人が、リソースにアクセスするために使用できるロールを作成できます。ロールの引き受けを委託するユーザーを指定できます。リソースベースのポリシーまたはアクセスコントロールリスト (ACL) をサポートするサービスの場合、それらのポリシーを使用して、リソースへのアクセスを付与できます。

詳細については、以下を参照してください:

- CloudShell でこれらの機能がサポートされているかどうかを確認するには、「AWS CloudShell と IAM の連携方法」を参照してください。
- 所有 AWS アカウント している のリソースへのアクセスを提供する方法については、「IAM ユーザーガイド」の「所有 AWS アカウント している別の の IAM ユーザーへのアクセスを提供する」を参照してください。
- リソースへのアクセスをサードパーティーに提供する方法については AWS アカウント、IAM ユーザーガイドの「サードパーティーが所有する へのアクセスを提供する AWS アカウント」を参照してください。
- ID フェデレーションを介してアクセスを提供する方法については、IAM ユーザーガイド の 外部で認証されたユーザー (ID フェデレーション) へのアクセスの許可 を参照してください。
- クロスアカウントアクセスにおけるロールとリソースベースのポリシーの使用方法の違いについては、IAM ユーザーガイド の IAM でのクロスアカウントのリソースへのアクセス を参照してください。

IAM ポリシーによる AWS CloudShell アクセスと使用状況の管理

が提供できるアクセス管理リソースを使用すると AWS Identity and Access Management、管理者は IAM ユーザーにアクセス許可を付与できます。これにより、これらのユーザーは環境の機能にアクセスして AWS CloudShell 使用できます。管理者は、ユーザーがシェル環境で実行できるアクションをきめ細かく指定するポリシーを作成することもできます。

管理者がユーザーにアクセス権を付与する最も簡単な方法は、 AWS マネージドポリシーを使用することです。AWS マネージドポリシーは、 AWSで作成および管理されるスタンドアロンポリシーです。の次の AWS 管理ポリシーを IAM ID にアタッチ AWS CloudShell できます。

- AWS CloudShellFullAccess: AWS CloudShell のすべての機能にフルアクセスできるアクセス許可を付与します。

AWS CloudShellFullAccess ポリシーでは、ワイルドカード (*) 文字を使用して、IAM アイデンティティ (ユーザー、ロール、またはグループ) に CloudShell および機能へのフルアクセスを許可します。このポリシーの詳細については、「AWS マネージドポリシーユーザーガイド」の「AWS CloudShellFullAccess」を参照してください。

> **Note**
>
> 以下の AWS 管理ポリシーを持つ IAM ID は、CloudShell を起動することもできます。ただし、これらのポリシーは広範な許可を付与します。そのため、 IAM ユーザーのジョブロールに必須な場合のみ、これらのポリシーを許可することを推奨します。
>
> - 管理者: IAM ユーザーにフルアクセスを提供し、 のすべてのサービスとリソースにアクセス許可を委任できるようにします AWS。
> - 開発者パワーユーザー: IAM ユーザーがアプリケーション開発タスクを実行し、 AWS アプリケーション開発をサポートするリソースとサービスを作成および設定できるようにします。
>
> マネージドポリシーをアタッチする方法の詳細については、IAM ユーザーガイドのIAM アイデンティ許可の追加 (コンソール)を参照してください。

カスタムポリシー AWS CloudShell を使用して で許可されるアクションを管理する

IAM ユーザーが CloudShell で実行できるアクションを管理するには、CloudShellPolicy マネージドポリシーをテンプレートとして使用するカスタムポリシーを作成します。または、関連する IAM アイデンティティ (ユーザー、グループ、もしくはロール) に埋め込まれているインラインポリシーを編集します。

例えば、IAM ユーザーに CloudShell へのアクセスを許可する一方で、 AWS マネジメントコンソールへのログインに使用する CloudShell 環境の認証情報の転送を禁止できます。

> **Important**
>
> AWS CloudShell から を起動するには AWS マネジメントコンソール、IAM ユーザーに次のアクションに対するアクセス許可が必要です。
>
> - `CreateEnvironment`
> - `CreateSession`
> - `GetEnvironmentStatus`
> - `StartEnvironment`
>
> これらのアクションのひとつがアタッチされたポリシーによって明示的に許可されていない場合、CloudShell の起動時に IAM アクセス許可エラーが返されます。

AWS CloudShell アクセス許可

名前	付与されたアクセス許可の説明	CloudShell の起動に必要か?
`cloudshell:CreateEnvironment`	CloudShell 環境を作成し、CloudShell セッションの開始時にレイアウトを取得して、バックエンドのウェブアプリケー	はい

名前	付与されたアクセス許可の説明	CloudShell の起動に必要か?
	ションから現在のレイアウトを保存します。このアクセス許可は、「the section called “CloudShell の IAM ポリシーの例”」で説明しているように * を Resource の値としてのみ期待します。	
cloudshell:CreateSession	から CloudShell 環境に接続します AWS マネジメントコンソール。	はい
cloudshell:GetEnvironmentStatus	CloudShell 環境のステータスを読み取ります。	はい
cloudshell:DeleteEnvironment	CloudShell 環境を削除します。	いいえ
cloudshell:GetFileDownloadUrls	CloudShell ウェブインターフェースを使用して CloudShell 経由でファイルをダウンロードする際に使用する、事前署名された Amazon S3 URL を生成します。これは VPC 環境では使用できません。	いいえ

名前	付与されたアクセス許可の説明	CloudShell の起動に必要か?
cloudshell:GetFileUploadUrls	CloudShell ウェブインターフェースを使用して CloudShell 経由でファイルをアップロードする際に使用する、事前署名された Amazon S3 URL を生成します。これは VPC 環境では使用できません。	いいえ
cloudshell:DescribeEnvironments	環境について説明します。	いいえ
cloudshell:PutCredentials	への ログインに使用される認証情報を CloudShell に転送 AWS マネジメントコンソール します。	いいえ
cloudshell:StartEnvironment	停止している CloudShell 環境を起動します。	はい
cloudshell:StopEnvironment	実行中の CloudShell 環境を停止します。	いいえ
cloudshell:ApproveCommand	他の AWS Service コンソールから CloudShell に送信されたコマンドを承認します。	いいえ

CloudShell の IAM ポリシーの例

次の例は、CloudShell へのアクセス可能なユーザーを制限するためのポリシーの作成方法を示しています。またこの例は、シェル環境で実行可能なアクションも示しています。

次のポリシーでは、CloudShell とその機能へのアクセスの完全拒否を強制的に実行します。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
        "Sid": "DenyCloudShell",
        "Effect": "Deny",
        "Action": [
            "cloudshell:*"
        ],
        "Resource": "*"
    }]
}
```

次のポリシーでは、IAM ユーザーが CloudShell にアクセスすることを許可しますが、ファイルのアップロードとダウンロード用の署名済み URL を生成することはブロックします。ユーザーは、例えば wget のようなクライアントを使用して、環境に向けておよび環境からファイルを転送することができます。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
        "Sid": "AllowUsingCloudshell",
        "Effect": "Allow",
        "Action": [
            "cloudshell:*"
        ],
        "Resource": "*"
    },
    {
        "Sid": "DenyUploadDownload",
        "Effect": "Deny",
        "Action": [
            "cloudshell:GetFileDownloadUrls",
            "cloudshell:GetFileUploadUrls"
        ],
        "Resource": "*"
```

```
    }]
}
```

次のポリシーでは、IAM ユーザーに CloudShell へのアクセスを許可します。ただし、このポリシーは、 へのログインに使用した認証情報が CloudShell AWS マネジメントコンソール 環境に転送されないようにします。このポリシーを持つ IAM ユーザーは、CloudShell 内で認証情報を手動で設定する必要があります。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
        "Sid": "AllowUsingCloudshell",
        "Effect": "Allow",
        "Action": [
            "cloudshell:*"
        ],
        "Resource": "*"
    },
    {
        "Sid": "DenyCredentialForwarding",
        "Effect": "Deny",
        "Action": [
            "cloudshell:PutCredentials"
        ],
        "Resource": "*"
    }]
}
```

次のポリシーでは、IAM ユーザーに AWS CloudShell 環境の作成を許可します。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
        "Sid": "CloudShellUser",
```

```
        "Effect": "Allow",
        "Action": [
            "cloudshell:CreateEnvironment",
            "cloudshell:CreateSession",
            "cloudshell:GetEnvironmentStatus",
            "cloudshell:StartEnvironment"
        ],
        "Resource": "*"
    }]
}
```

CloudShell の VPC 環境を作成および使用するために必要な IAM アクセス許可

CloudShell の VPC 環境を作成して使用するには、IAM 管理者が VPC 固有の Amazon EC2 アクセス許可へのアクセスを有効にする必要があります。このセクションでは、VPC 環境の作成と使用に必要な Amazon EC2 アクセス許可を一覧表示します。

VPC 環境を作成するには、ロールに割り当てる IAM ポリシーに、以下の Amazon EC2 アクセス許可を含める必要があります。

- `ec2:DescribeVpcs`
- `ec2:DescribeSubnets`
- `ec2:DescribeSecurityGroups`
- `ec2:DescribeDhcpOptions`
- `ec2:DescribeNetworkInterfaces`

- `ec2:CreateTags`
- `ec2:CreateNetworkInterface`
- `ec2:CreateNetworkInterfacePermission`

以下も含めることをお勧めします。

- ec2:DeleteNetworkInterface

> **Note**
>
> このアクセス許可は必須ではありませんが、CloudShell で作成した ENI リソース (CloudShell の VPC 環境用に作成した ENI には ManagedByCloudShell キーのタグが付きます) をクリーンアップするために必要です。このアクセス許可が有効になっていない場合は、CloudShell の各 VPC 環境の使用後に ENI リソースを手動でクリーンアップする必要があります。

CloudShell へのフルアクセス (VPC へのアクセスを含む) を許可する IAM ポリシー

次の例は、CloudShell へのフルアクセス (VPC へのアクセスを含む) を許可する方法を示しています。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AllowCloudShellOperations",
      "Effect": "Allow",
      "Action": [
        "cloudshell:*"
      ],
      "Resource": "*"
    },
    {
      "Sid": "AllowDescribeVPC",
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeDhcpOptions",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DescribeSubnets",
        "ec2:DescribeSecurityGroups",
        "ec2:DescribeVpcs"
      ],
      "Resource": "*"
    },
    {
      "Sid": "AllowCreateTagWithCloudShellKey",
```

```
      "Effect": "Allow",
      "Action": [
        "ec2:CreateTags"
      ],
      "Resource": "arn:aws:ec2:*:*:network-interface/*",
      "Condition": {
        "StringEquals": {
          "ec2:CreateAction": "CreateNetworkInterface"
        },
        "ForAnyValue:StringEquals": {
          "aws:TagKeys": "ManagedByCloudShell"
        }
      }
    },
    {
      "Sid": "AllowCreateNetworkInterfaceWithSubnetsAndSG",
      "Effect": "Allow",
      "Action": [
        "ec2:CreateNetworkInterface"
      ],
      "Resource": [
        "arn:aws:ec2:*:*:subnet/*",
        "arn:aws:ec2:*:*:security-group/*"
      ]
    },
    {
      "Sid": "AllowCreateNetworkInterfaceWithCloudShellTag",
      "Effect": "Allow",
      "Action": [
        "ec2:CreateNetworkInterface"
      ],
      "Resource": "arn:aws:ec2:*:*:network-interface/*",
      "Condition": {
        "ForAnyValue:StringEquals": {
          "aws:TagKeys": "ManagedByCloudShell"
        }
      }
    },
    {
      "Sid": "AllowCreateNetworkInterfacePermissionWithCloudShellTag",
      "Effect": "Allow",
      "Action": [
        "ec2:CreateNetworkInterfacePermission"
      ],
```

```
      "Resource": "arn:aws:ec2:*:*:network-interface/*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/ManagedByCloudShell": ""
        }
      }
    },
    {
      "Sid": "AllowDeleteNetworkInterfaceWithCloudShellTag",
      "Effect": "Allow",
      "Action": [
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "arn:aws:ec2:*:*:network-interface/*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/ManagedByCloudShell": ""
        }
      }
    }
  ]
}
```

VPC 環境での IAM 条件キーの使用

VPC 設定で CloudShell 固有の条件キーを使用して、VPC 環境に追加のアクセス許可コントロールを提供できます。また、VPC 環境で使用できるサブネットとセキュリティグループ、および使用できないサブネットとセキュリティグループを指定することもできます。

CloudShell は IAM ポリシーで以下の条件キーをサポートしています。

- `CloudShell:VpcIds` – 1 つ以上の VPC を許可または拒否する
- `CloudShell:SubnetIds` – 1 つ以上のサブネットを許可または拒否する
- `CloudShell:SecurityGroupIds` – 1 つ以上のセキュリティグループを許可または拒否する

> **Note**
>
> ユーザーに CloudShell のパブリック環境へのアクセス権がある場合、ユーザーのアクセス許可を変更して `cloudshell:createEnvironment` アクションに制限を追加しても、ユーザーは依然として既存のパブリック環境にアクセスできます。ただし、この制限を追加して

IAM ポリシーを変更し、既存のパブリック環境へのユーザーアクセスを無効にしたい場合は、まず、IAM ポリシーを更新して、この制限を含める必要があります。次に、アカウントのすべての CloudShell ユーザーに、CloudShell ウェブユーザーインターフェイスを使用して既存のパブリック環境を手動で確実に削除してもらいます ([アクション] → [CloudShell 環境を削除])。

VPC 設定の条件キーを使用したポリシーの例

以下の例は、VPC 設定で条件キーを使用する方法を示しています。必要な制限を含むポリシーステートメントを作成したら、このポリシーステートメントをターゲットのユーザーまたはロールに追加します。

ユーザーに VPC 環境の作成のみを許可し、パブリック環境の作成を拒否する

ユーザーに VPC 環境の作成のみを許可するには、次の例に示すように [拒否] アクセス許可を使用します。

```
{
  "Statement": [
    {
      "Sid": "DenyCloudShellNonVpcEnvironments",
      "Action": [
        "cloudshell:CreateEnvironment"
      ],
      "Effect": "Deny",
      "Resource": "*",
      "Condition": {
        "Null": {
          "cloudshell:VpcIds": "true"
        }
      }
    }
  ]
}
```

特定の VPC、サブネット、セキュリティグループへのアクセスをユーザーに拒否する

特定の VPC へのアクセスをユーザーに拒否するには、`StringEquals` を使用して `cloudshell:VpcIds` 条件の値を確認します。次の例では、`vpc-1` および `vpc-2` へのアクセスをユーザーに拒否します。

特定の VPC へのアクセスをユーザーに拒否するには、StringEquals を使用して cloudshell:SubnetIds 条件の値を確認します。次の例では、subnet-1 および subnet-2 へのアクセスをユーザーに拒否します。

特定の VPC へのアクセスをユーザーに拒否するには、StringEquals を使用して cloudshell:SecurityGroupIds 条件の値を確認します。次の例では、sg-1 および sg-2 へのアクセスをユーザーに拒否します。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "EnforceOutOfSecurityGroups",
      "Action": [
        "cloudshell:CreateEnvironment"
      ],
      "Effect": "Deny",
      "Resource": "*",
      "Condition": {
        "ForAnyValue:StringEquals": {
          "cloudshell:SecurityGroupIds": [
            "sg-1",
            "sg-2"
          ]
        }
      }
    }
  ]
}
```

特定の VPC 設定で環境を作成することをユーザーに許可する

特定の VPC にアクセスすることをユーザーに許可するには、StringEquals を使用して cloudshell:VpcIds 条件の値を確認します。次の例では、vpc-1 および vpc-2 にアクセスすることをユーザーに許可します。

特定の VPC にアクセスすることをユーザーに許可するには、StringEquals を使用して cloudshell:SubnetIds 条件の値を確認します。次の例では、subnet-1 および subnet-2 にアクセスすることをユーザーに許可します。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "EnforceStayInSpecificSubnets",
      "Action": [
        "cloudshell:CreateEnvironment"
      ],
      "Effect": "Allow",
      "Resource": "*",
      "Condition": {
        "ForAllValues:StringEquals": {
          "cloudshell:SubnetIds": [
            "subnet-1",
            "subnet-2"
          ]
        }
      }
    }
  ]
}
```

特定の VPC にアクセスすることをユーザーに許可するには、StringEquals を使用して cloudshell:SecurityGroupIds 条件の値を確認します。次の例では、sg-1 および sg-2 にアクセスすることをユーザーに許可します。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "EnforceStayInSpecificSecurityGroup",
      "Action": [
```

```
        "cloudshell:CreateEnvironment"
      ],
      "Effect": "Allow",
      "Resource": "*",
      "Condition": {
        "ForAllValues:StringEquals": {
          "cloudshell:SecurityGroupIds": [
            "sg-1",
            "sg-2"
          ]
        }
      }
    }
  ]
}
```

にアクセスするためのアクセス許可 AWS のサービス

CloudShell は、 AWS マネジメントコンソールへのサインインに使用された IAM 認証情報を使用します。

> **Note**
>
> へのサインインに使用した IAM 認証情報を使用するには AWS マネジメントコンソール、 アクセス`cloudshell:PutCredentials`許可が必要です。

CloudShell のこの事前認証機能は、 AWS CLIを使用するうえで便利です。ただし、IAM ユーザーには、コマンドラインから呼び出 AWS のサービス される に対する明示的なアクセス許可が必要です。

例えば、IAM ユーザーが Amazon S3 バケットを作成し、ファイルをオブジェクトとしてそこにアップロードする必要があるとします。これらのアクションを明示的に許可するポリシーを作成することができます。IAM コンソールには、JSON 形式のポリシードキュメントを作成する手順を説明するインタラクティブなビジュアルエディタが用意されています。ポリシーを作成した後、関連する IAM アイデンティティ (ユーザー、グループ、もしくはロール) にアタッチできます。

マネージドポリシーをアタッチする方法の詳細については、IAM ユーザーガイドのIAM アイデンティティ許可の追加 (コンソール)を参照してください。

CloudShell の Amazon Q CLI 機能へのアクセス許可

CloudShell でインライン提案、チャット、翻訳などの Amazon Q CLI 機能を使用するには、必要な IAM アクセス許可があることを確認してください。CloudShell で Amazon Q CLI 機能にアクセスできない場合は、管理者に連絡して必要な IAM アクセス許可を得てください。詳細については、「Amazon Q Developer ユーザーガイド」の「Amazon Q Developer のアイデンティティベースのポリシー例」を参照してください。

でのログ記録とモニタリング AWS CloudShell

このトピックでは、CloudTrail を使用して AWS CloudShell アクティビティとパフォーマンスを記録およびモニタリングする方法について説明します。

CloudTrail によるアクティビティのモニタリング

AWS CloudShell は、ユーザー AWS CloudTrail、ロール、または によって実行されたアクションを記録するサービスである と統合 AWS のサービス されています AWS CloudShell。CloudTrail は、のすべての API コールをイベント AWS CloudShell としてキャプチャします。キャプチャされた呼び出しには、 AWS CloudShell コンソールからの呼び出しと AWS CloudShell API へのコード呼び出しが含まれます。

証跡を作成する場合、Amazon Simple Storage Service (Amazon S3) バケットへの CloudTrail イベントの継続的な配信を有効にすることができます。これには、 のイベントが含まれます AWS CloudShell。

証跡を設定しない場合でも、CloudTrail コンソールの [イベント履歴] で最新のイベントを表示できます。CloudTrail によって収集された情報を使用して、リクエストに関する多くの情報を見つけることができます。例えば、AWS CloudShell に対して作成されたリクエスト、リクエストの作成元の IP アドレス、リクエストの作成者、リクエストの作成日時を確認できます。

AWS CloudShell CloudTrail の

次の表に、CloudTrail ログファイルに保存されている AWS CloudShell イベントを示します。

> **Note**
>
> AWS CloudShell イベントには以下が含まれます。
>
> - * は、非ミューテーション (読み取り専用) API コールであることを示します。

- 単語 Environment は、シェルエクスペリエンスをホストするコンピューティング環境のライフサイクルに関連しています。
- 単語 Layout は、CloudShell ターミナルのすべてのブラウザタブを復元します。

CloudTrail の CloudShell イベント

イベント名	説明
createEnvironment	CloudShell 環境を作成したときに発生します。
createSession	CloudShell 環境が から接続されている場合に発生します AWS マネジメントコンソール。
deleteEnvironment	CloudShell 環境を削除したときに発生します。
deleteSession	現在のブラウザタブで実行中である、CloudShell タブ内のセッションを削除したときに発生します。
getEnvironmentStatus*	CloudShell 環境のステータスを取得したときに発生します。
getFileDownloadUrls*	CloudShell ウェブインターフェースを使用して CloudShell 経由でファイルをダウンロードする際に使用する、事前署名した Amazon S3 URL を生成したときに発生します。
getFileUploadUrls*	CloudShell ウェブインターフェースを使用して CloudShell 経由でファイルをアップロードする際に使用する、事前署名した Amazon S3 URL を生成したときに発生します。
cloudshell:DescribeEnvironments	環境について説明します。
getLayout*	セッション開始時に CloudShell レイアウトを取得したときに発生します。

イベント名	説明
putCredentials	への ログインに使用される認証情報が AWS マネジメントコンソール CloudShell に転送されたときに発生します。
redeemCode*	CloudShell 環境で更新トークンを取得するためのワークフローを開始したときに発生します。後で putCredentials コマンドでこのトークンを使用して CloudShell 環境にアクセスできます。
sendHeartBeat	セッションがアクティブであることを確認するために発生します。
startEnvironment	CloudShell 環境を起動したときに発生します。
stopEnvironment	実行中の CloudShell 環境が停止したときに発生します。
updateLayout	バックエンドでウェブアプリケーションの現在のレイアウトを保存したときに発生します。

「Layout」という単語を含むイベントは、 CloudShell ターミナルのすべてのブラウザタブを復元します。

AWS CloudShell アクションの EventBridge ルール

EventBridge ルールでは、EventBridge がルールに一致するイベントを受信したときに実行するターゲットアクションを指定します。CloudTrail ログファイルにイベントとして記録される AWS CloudShell アクションに基づいて実行できるターゲットアクションを指定するルール定義できます。

例えば、put-rule コマンドを使用して、AWS CLIで EventBridge ルールを作成できます。put-rule コールには、少なくとも EventPattern または ScheduleExpression を含める必要があります。EventPattern を含むルールは、一致するイベントが確認されときにトリガーされます。 AWS CloudShell イベントの EventPattern:

```
{ "source": [ "aws.cloudshell" ], "detail-type": [ "AWS API Call via CloudTrail" ],
 "detail": { "eventSource": [ "cloudshell.amazonaws.com" ] } }
```

詳細については、Amazon EventBridge ユーザーガイドの「EventBridge のイベントとイベントパターン」を参照してください。

のコンプライアンス検証 AWS CloudShell

サードパーティーの監査者は、複数の AWS コンプライアンスプログラムの一環として AWS サービスのセキュリティとコンプライアンスを評価します。

AWS CloudShell は、以下のコンプライアンスプログラムの対象です。

SOC

AWS システムおよび組織統制 (SOC) レポートは、 が主要なコンプライアンス統制と目的 AWS を達成する方法を示す独立したサードパーティー審査レポートです。

サービス	SDK	SOC 1、2、3
AWS CloudShell	CloudShell	✓

PCI

Payment Card Industry Data Security Standard (PCI DSS) は、PCI Security Standards Council によって管理される機密情報のセキュリティ標準であり、American Express、Discover Financial Services、JCB International、MasterCard Worldwide、Visa Inc により創設されました。

サービス	SDK	PCI
AWS CloudShell	CloudShell	✓

ISO および CSA STAR 認証およびサービス

AWS は、ISO/IEC 27001:2013、27017:2015、27018:2019、27701:2019、22301:2019、9001:2015、および CSA STAR CCM v4.0 への準拠の認定を受けています。

サービス	SDK	ISO および CSA STAR 認証およびサービス
AWS CloudShell	CloudShell	✓

FedRamp

Federal Risk and Authorization Management Program (FedRAMP) は米国政府全体のプログラムであり、クラウドの製品やサービスに対するセキュリティ評価、認可、および継続的なモニタリングに関する標準アプローチを提供しています。

サービス	SDK	FedRAMP Moderate (East/West)	FedRAMP High (GovCloud)
AWS CloudShell	CloudShell	✓	✓

DoD CC SRG

米国防総省 (DoD) クラウドコンピューティングセキュリティ要求事項ガイド (SRG) には、クラウドサービスプロバイダー (CSP) が DoD の暫定認可を取得して DoD ユーザーへのサービス提供を可能にする、標準化された評価と認可プロセスが規定されています。

DoD CC SRG の評価および承認を受けているサービスのステータスは、次のとおりです。

- 第三者評価機関 (3PAO) の評価: このサービスは現在、第三者評価機関による評価を受けています。
- 共同承認委員会 (JAB) による審査: このサービスは、現在、JAB による審査を受けているところです。
- アメリカ国防情報システム局 (DISA) による審査: このサービスは、現在、DISA による審査を受けているところです。

サービス	SDK	DoD CC SRG IL2 (East/West)	DoD CC SRG IL2 (GovCloud)	DoD CC SRG IL4 (GovCloud)	DoD CC SRG IL5 (GovCloud)	DoD CC SRG IL6 (AWS シークレットリージョン)
AWS CloudShell	CloudShell	✓	✓	✓	✓	該当なし

HIPAA BAA

1996 年の医療保険の相互運用性と説明責任に関する法令 (HIPAA) は、患者の同意や認識なく機密性の高い患者の健康情報が開示されないようにするための国家基準の作成を義務付けた連邦法です。

AWS は、HIPAA の対象となる対象エンティティとそのビジネスアソシエイトが、保護された医療情報 (PHI) を安全に処理、保存、送信できるようにします。さらに、2013 年 7 月現在、 はそのようなお客様に標準化されたビジネスアソシエイト補遺 (BAA) AWS を提供しています。

サービス	SDK	HIPAA BAA
AWS CloudShell	CloudShell	✓

IRAP

オーストラリア政府のお客様は、情報セキュリティ登録評価プログラム (IRAP) を使用して、適切な制御が行われていることを検証し、オーストラリアサイバーセキュリティセンター (ACSC) が作成したオーストラリア政府情報セキュリティマニュアル (ISM) の要件に対応する適切な責任モデルを決定することができます。

サービス	名前空間*	IRAP による保護
AWS CloudShell	該当なし	✓

*名前空間は、 AWS 環境全体のサービスを識別するのに役立ちます。たとえば、IAM ポリシーを作成するときは、Amazon リソースネーム (ARNs) と read AWS CloudTrail ログを使用します。

MTCS

The Multi-Tier Cloud Security (MTCS) は、ISO 27001/02 情報セキュリティ管理システム (ISMS) 規格に基づく、シンガポールで運用されているセキュリティ管理規格 (SPRING SS 584) です。

サービス	SDK	米国東部 (オハイオ)	米国東部 (バージニア北部)	米国西部 (オレゴン)	米国西部 (北カリフォルニア)	シンガポール	ソウル
AWS CloudShell	CloudShell	✓	✓	✓	該当なし	該当なし	該当なし

C5

Cloud Computing Compliance Controls Catalog (C5) は、ドイツ連邦情報セキュリティ局 (BSI) がドイツで導入したドイツ政府支援の証明スキームで、ドイツ政府の「クラウドプロバイダーに対するセキュリティに関するレコメンデーション」内でクラウドサービスを使用するときに、組織が一般的なサイバー攻撃に対する運用上のセキュリティを実証できるようにします。

サービス	SDK	C5
AWS CloudShell	CloudShell	✓

ENS High

ENS (国家セキュリティスキーム) 認証は、財務省・公共省および CCN (国立暗号センター) によって開発されました。これは、情報を適切に保護するために必要な基本原則と最低限の要件で構成されています。

サービス	SDK	ENS High
AWS CloudShell	CloudShell	✓

FINMA

スイス金融市場監督局 (FINMA) は、スイスの独立した金融市場規制機関です。 AWSが FINMA の要件に準拠していることは、スイスの金融サービス規制当局や顧客からクラウドサービスプロバイダーへの高まる期待に応えようとする当社の継続的な取り組みの表れです。

サービス	SDK	FINMA
AWS CloudShell	CloudShell	✓

PiTuKri

AWS PiTuKri 要件との整合性は、フィンランド運輸通信局である Traficom によって設定されたクラウドサービスプロバイダーに対する高い期待を満たすという当社の継続的なコミットメントを示しています。

サービス	SDK	PiTuKri
AWS CloudShell	CloudShell	✓

特定のコンプライアンスプログラムの対象となる AWS サービスのリストについては、「コンプライアンスプログラムによる AWS 対象範囲内のサービスコンプライアンスプログラム」を参照してください。一般的な情報については、AWS 「 Compliance ProgramsAssurance」を参照してください。

を使用して、サードパーティーの監査レポートをダウンロードできます AWS Artifact。詳細については、Downloading Reports in AWS および を参照してください。

を使用する際のお客様のコンプライアンス責任 AWS CloudShell は、お客様のデータの機密性、貴社のコンプライアンス目的、適用される法律および規制によって決まります。 は、コンプライアンスに役立つ以下のリソース AWS を提供します。

- セキュリティとコンプライアンスのクイックスタートガイド – これらのデプロイガイドでは、アーキテクチャ上の考慮事項について説明し、セキュリティとコンプライアンスに重点を置いたベースライン環境をデプロイする手順について説明します AWS。
- 「Architecting for HIPAA Security and Compliance」ホワイトペーパー – このホワイトペーパーでは、企業が AWS を使用して HIPAA 準拠のアプリケーションを作成する方法について説明します。

- AWS コンプライアンスリソース – このワークブックとガイドのコレクションは、お客様の業界や地域に適用される場合があります。
- 「 デベロッパーガイド」の「ルールによるリソースの評価」 – この AWS Config サービスは、リソース設定が内部プラクティス、業界ガイドライン、および規制にどの程度準拠しているかを評価します。 AWS Config
- AWS Security Hub CSPM – この AWS サービスは、 内のセキュリティ状態を包括的に把握 AWS し、セキュリティ業界標準とベストプラクティスへの準拠を確認するのに役立ちます。

の耐障害性 AWS CloudShell

AWS グローバルインフラストラクチャは、 AWS リージョンとアベイラビリティーゾーンを中心に構築されています。 AWS リージョンは、低レイテンシー、高スループット、および高度に冗長なネットワークで接続された、物理的に分離された複数のアベイラビリティーゾーンを提供します。アベイラビリティーゾーンでは、ゾーン間で中断することなく自動的にフェイルオーバーするアプリケーションとデータベースを設計および運用することができます。アベイラビリティーゾーンは、従来の単一または複数のデータセンターインフラストラクチャよりも可用性、フォールトトレランス、および拡張性が優れています。

AWS リージョンとアベイラビリティーゾーンの詳細については、AWS 「 グローバルインフラストラクチャ」を参照してください。

グローバル AWS インフラストラクチャに加えて、 はデータの耐障害性とバックアップのニーズをサポートするために、次の機能 AWS CloudShell をサポートしています。

- AWS CLI 呼び出しを使用して、 のホームディレクトリ内のファイルを指定 AWS CloudShell し、Amazon S3 バケット内のオブジェクトとして追加します。例については、「AWS CloudShell の開始方法」を参照してください。

のインフラストラクチャセキュリティ AWS CloudShell

マネージドサービスである AWS CloudShell は、 AWS グローバルネットワークセキュリティで保護されています。 AWS セキュリティサービスと がインフラストラクチャ AWS を保護する方法については、AWS 「 クラウドセキュリティ」を参照してください。インフラストラクチャセキュリティのベストプラクティスを使用して環境を AWS 設計するには、「Security Pillar AWS Well-Architected Framework」の「Infrastructure Protection」を参照してください。

AWS が公開した API コールを使用して、ネットワーク AWS CloudShell 経由で にアクセスします。クライアントは次をサポートする必要があります。

- Transport Layer Security (TLS)。TLS 1.2 が必須で、TLS 1.3 をお勧めします。
- DHE (楕円ディフィー・ヘルマン鍵共有) や ECDHE (楕円曲線ディフィー・ヘルマン鍵共有) などの完全前方秘匿性 (PFS) による暗号スイート。これらのモードは Java 7 以降など、ほとんどの最新システムでサポートされています。

> **Note**
>
> デフォルトでは、コンピューティング環境のシステムパッケージのセキュリティパッチ AWS CloudShell を自動的にインストールします。

のセキュリティのベストプラクティス AWS CloudShell

以下のベストプラクティスは一般的なガイドラインであり、完全なセキュリティソリューションに相当するものではありません。これらのベストプラクティスは、お客様の環境に適切ではないか、十分ではない場合があるため、絶対的な解決策ではなく、役立つ情報として扱うことをお勧めします。

のセキュリティのベストプラクティス AWS CloudShell

- IAM アクセス許可とポリシーを使用して、 へのアクセスを制御し AWS CloudShell 、ユーザーがロールに必要なアクション (ファイルのダウンロードやアップロードなど) のみを実行できるようにします。詳細については、「IAM ポリシーによる AWS CloudShell アクセスと使用状況の管理」を参照してください。
- ユーザー、ロール、セッション名などの機密データを IAM エンティティに含めないでください。
- 安全な貼り付け機能を有効にして、外部ソースからコピーしたテキスト内の潜在的なセキュリティリスクを捉えます。デフォルトでは、安全な貼り付けが有効になっています。複数行テキストに安全な貼り付けを使用する方法の詳細については、「複数行テキストに安全な貼り付けを使用する」を参照してください。
- AWS CloudShellのコンピューティング環境にサードパーティアプリケーションをインストールする場合は、セキュリティ責任共有モデルをよく理解します。
- ユーザーのシェル環境に影響を与えるシェルスクリプトを編集する前に、ロールバックメカニズムを準備します。デフォルトのシェル環境を変更する方法の詳細については、「スクリプトを使用してシェルを変更する」を参照してください。

- コードをバージョン管理システムで安全に保存します。

AWS CloudShell セキュリティFAQs

CloudShell のセキュリティに関するよくある質問への回答を以下に示します。

CloudShell を起動してシェルセッションを開始するときに、どのような AWS プロセスとテクノロジーが使用されますか?

サインインするときは AWS マネジメントコンソール、IAM ユーザー認証情報を入力します。また、コンソールインターフェースから CloudShell を起動すると、これらの認証情報はサービスのコンピューティング環境を作成する CloudShell API の呼び出しに使用されます。その後、コンピューティング環境の AWS Systems Manager セッションが作成され、CloudShell はそのセッションにコマンドを送信します。

セキュリティに関するよくある質問リストに戻る

CloudShell へのネットワークアクセスを制限することはできますか?

パブリック環境では、ネットワークアクセスを制限することはできません。ネットワークアクセスを制限する場合は、VPC 環境の作成のみを許可し、パブリック環境の作成を拒否するアクセス許可を有効にする必要があります。

詳細については、「ユーザーに VPC 環境の作成のみを許可し、パブリック環境の作成を拒否する」を参照してください。

CloudShell の VPC 環境の場合、ネットワーク設定は VPC から継承します。VPC で CloudShell を使用すると、CloudShell の VPC 環境のネットワークアクセスを制御できます。

セキュリティに関するよくある質問リストに戻る

CloudShell 環境をカスタマイズすることはできますか?

CloudShell 環境用のユーティリティやその他のサードパーティソフトウェアをダウンロードして、インストールできます。$HOME ディレクトリにインストールされたソフトウェアのみがセッション間で保持されます。

AWS 責任分担モデルで定義されているように、インストールするアプリケーションの必要な設定と管理に対する責任があります。

セキュリティに関するよくある質問リストに戻る

私の $HOME ディレクトリは実際には AWS クラウドのどこに保存されていますか?

パブリック環境の場合、$HOME にデータを保存するためのインフラストラクチャは、Amazon S3 によって提供されます。

VPC 環境の場合、$HOME ディレクトリは、VPC 環境がタイムアウトするか (20 ~ 30 分間非アクティブ状態が続いた後)、環境を削除または再起動すると、削除されます。

セキュリティに関するよくある質問リストに戻る

自分の $HOME ディレクトリを暗号化することはできますか?

いいえ、$HOME ディレクトリを独自のキーで暗号化することはできません。ただし、CloudShell が $HOME ディレクトリコンテンツを Amazon S3 に保管中に暗号化します。

セキュリティに関するよくある質問リストに戻る

自分の $HOME ディレクトリでウイルススキャンを実行することはできますか?

現時点では、ご自身の $HOME ディレクトリのウイルススキャンを実行することはできません。この機能のサポートは確認中です。

セキュリティに関するよくある質問リストに戻る

CloudShell のデータの進入や退出は制限はできますか?

進入や退出を制限するには、CloudShell の VPC 環境を使用することをお勧めします。VPC 環境の $HOME ディレクトリは、VPC 環境がタイムアウトするか (20 ~ 30 分間非アクティブ状態が続いた後)、環境を削除または再起動すると、削除されます。VPC 環境では、[アクション] メニューのアップロードやダウンロードのオプションは使用できません。

セキュリティに関するよくある質問リストに戻る

CloudShell ユーザーをコンテナ内のルートアクセスに制限できますか?

No. は、コンピューティングコンテナ内のルートアクセスを設計上 AWS CloudShell 提供します。CloudShell のコンテナは、コードのパッケージ化と運用上の利便性として機能します。これらはセキュリティの境界ではありません。

CloudShell は AWS Identity and Access Management (IAM) を通じてアクセスコントロールを管理します。各 CloudShell セッションは、ユーザーのアクセス許可の範囲内で、定期的にローテーションされた一時的な IAM 認証情報を受け取ります。これらの認証情報はセキュリティ境界であり、コンテナ自体ではありません。

コンテナと基盤となるインスタンスは同じ IAM 認証情報スコープを共有するため、コンテナの境界を超えてアクセスしても追加のアクセス AWS 許可は提供されません。

セキュリティに関するよくある質問リストに戻る

AWS CloudShell コンピューティング環境: 仕様とソフトウェア

を起動すると AWS CloudShell、Amazon Linux 2023 に基づくコンピューティング環境が作成され、シェルエクスペリエンスがホストされます。環境は、コンピューティングリソース (vCPU およびメモリ) に設定され、コマンドラインインターフェイスからアクセスできるプリインストールされた幅広い機能を提供しています。コンピューティング環境にインストールしたすべてのソフトウェアにパッチが適用されており、最新の状態であることを確認します。ソフトウェアをインストールし、シェルスクリプトを変更して、デフォルト環境を構成することもできます。

コンピューティング環境のリソース

各 AWS CloudShell コンピューティング環境には、次の CPU リソースとメモリリソースが割り当てられます。

- 1 vCPU (仮想 CPU)
- 2-GiB RAM

また、環境は次のストレージ構成でプロビジョニングされます。

- 1-GB の永続的ストレージ (セッション終了後もストレージは保持されます)

詳細については、「永続ストレージ」を参照してください。

CloudShell ネットワーク要件

WebSockets

CloudShell は WebSocket プロトコルに依存しています。これにより、ユーザーのウェブブラウザと AWS クラウド内の CloudShell サービス間の双方向のインタラクティブ通信が可能になります。プライベートネットワークでブラウザを使用している場合、プロキシサーバーとファイアウォールによってインターネットへの安全なアクセスが促進されていると考えられます。通常、WebSocket 通信は、問題なくプロキシサーバーを通過できます。しかし、場合によっては、プロキシサーバーが WebSockets の正常な動作を妨げることがあります。この問題が発生した場合、CloudShell インターフェースは次のエラーを報告します (`Failed to open sessions : Timed out while opening the session`)。

このエラーが繰り返し発生する場合は、プロキシサーバーのドキュメントを参照して、WebSockets を許可するように設定されていることを確認します。または、ネットワークのシステム管理者に問い合わせてください。

> **Note**
>
> 特定の URLs を許可リストに登録してきめ細かなアクセス許可を定義する場合は、 AWS Systems Manager セッションが入出力を送信および受信するための WebSocket 接続を開くために使用する URL の一部を追加できます。(AWS CloudShell コマンドはその Systems Manager セッションに送信されます)。
> Systems Manager が使用するこの StreamURL の形式は `wss://ssmmessages.`**`region`**`.amazonaws.com/v1/data-channel/`**`session-id`**`?stream=(input|output)` です。
> リージョンは、米国東部 (オハイオ) リージョンなど AWS Systems Manager、 でサポートされている リージョン`us-east-2`のリージョン識別子 AWS を表します。
> セッション ID は特定の Systems Manager セッションが正常に開始された後に作成されるため、URL 許可リストを更新するときしか `wss://ssmmessages.region.amazonaws.com` を指定できません。詳細については、「AWS Systems Manager API リファレンス」の「StartSession」オペレーションを参照してください。

プリインストールされたソフトウェア

> **Note**
>
> AWS CloudShell 開発環境は最新のソフトウェアへのアクセスを提供するために定期的に更新されるため、このドキュメントでは特定のバージョン番号は提供していません。代わりに、インストールされているバージョンをチェックする方法を記述します。インストールされているバージョンを確認するには、プログラム名の後に `--version` オプション (例えば、`git --version` など) を入力します。

シェル

プレインストールされたシェル

名前	説明	バージョン情報
Bash	Bash シェルはデフォルトのシェルアプリケーションです AWS CloudShell。	`bash --version`
PowerShell (pwsh)	コマンドラインインターフェイスとスクリプト言語のサポートを提供する PowerShell は、マイクロソフトの .NET コマンド言語ランタイムの上に構築されています。PowerShell は、.NET オブジェクトを受信して返す `cmdlets` と呼ばれる軽量コマンドを使用しています。	`pwsh --version`
Zシェル (zsh)	Z シェル、別名 `zsh` は、テーマおよびプラグインのカスタマイズサポートを強化した Bourne シェルの拡張バージョンです。	`zsh --version`

AWS コマンドラインインターフェイス (CLI)

CLI

名前	説明	バージョン情報
AWS CDK ツールキット CLI	Toolkit AWS CDK 、CLI コマンド、 は`cdk`、 AWS CDK アプリを操作する主要なツールです。アプリケーションを実行し、定義したアプリ	`cdk --version`

名前	説明	バージョン情報
	ケーションモデルを調査し、によって生成された AWS CloudFormation テンプレートを生成してデプロイします AWS CDK。 詳細については、「AWS CDK Toolkit」を参照してください。	
AWS CLI	AWS CLI は、コマンドラインから複数の AWS サービスを管理し、スクリプトを使用して自動化するために使用できるコマンドラインインターフェイスです。詳細については、「CloudShell で CLI から AWS サービスを管理する」を参照してください。 最新バージョンである AWS CLI バージョン 2 を確実に使用する方法については、「ホームディレクトリ AWS CLI への のインストール」を参照してください。	`aws --version`

名前	説明	バージョン情報
EB CLI	AWS Elastic Beanstalk CLI には、ローカルリポジトリからの環境の作成、更新、モニタリングを簡素化するコマンドラインインターフェイスが用意されています。 詳細については、AWS Elastic Beanstalk デベロッパーガイドの「Elastic Beanstalk コマンドラインインターフェイス (EB CLI) の使用」を参照してください。	`eb --version`
Amazon ECS CLI	Amazon Elastic Container Service (Amazon ECS) コマンドラインインターフェイス (CLI) は、クラスターとタスクの作成、更新、モニタリングを簡素化するための高レベルのコマンドを提供します。 詳細については、Amazon Elastic Container Service デベロッパーガイドの「Amazon ECS コマンドラインインターフェイスの使用」を参照してください。	`ecs-cli --version`

名前	説明	バージョン情報
AWS SAM CLI	AWS SAM CLI は、 AWS Serverless Application Model テンプレートとアプリケーションコードで動作するコマンドラインツールです。いくつものタスクを実行できます。これには、Lambda 関数のローカル呼び出し、サーバーレスアプリケーションのデプロイパッケージの作成、サーバーレスアプリケーションの AWS クラウドへのデプロイが含まれます。 詳細については、AWS Serverless Application Model デベロッパーガイドの「AWS SAM CLI コマンドレファレンス」を参照してください。	`sam --version`

名前	説明	バージョン情報
AWS Tools for PowerShell	AWS Tools for PowerShell は、 によって公開されている機能に基づいて構築された PowerShell モジュールです SDK for .NET。を使用すると AWS Tools for PowerShell、PowerShell コマンドラインから AWS リソースに対するオペレーションをスクリプト化できます。 AWS CloudShell は、 のモジュール化されたバージョン (AWS.Tools) をプリインストールします AWS Tools for PowerShell。 詳細については、「AWS Tools for PowerShell ユーザーガイド」の「AWS Tools for PowerShell の使用」を参照してください。	`pwsh --Command 'Get-AWSPowerShellVersion'`

ランタイムおよび AWS SDK: Node.js および Python 3

ランタイムおよび AWS SDK

名前	説明	バージョン情報
Node.js (npm 付き)	Node.js は、非同期プログラミング手法を簡単に適用できるように設計された JavaScript ランタイムです。詳細については、「Node.js の公式サイトのドキュメント」を参照してください。	• Node.js: `node --version` • npm: `npm --version`

名前	説明	バージョン情報
	npm は JavaScript モジュールのオンラインレジストリへのアクセスを提供するパッケージマネージャーです。詳細については、「公式 npm サイトのドキュメント」を参照してください。	
SDK for JavaScript in Node.js	Software Development Kit (SDK)を使用すると、Amazon S3、Amazon EC2、DynamoDB、および Amazon SWF などの AWS のサービスに JavaScript オブジェクトを提供することで、コーディングを簡素化できます。詳細については、AWS SDK for JavaScript デベロッパーガイドを参照してください。	`npm -g ls --depth 0 2>/dev/null | grep aws-sdk`

名前	説明	バージョン情報
Python	Python 3 はシェル環境で使用可能になりました。Python 3 は現在、プログラミング言語のデフォルトバージョンと見なされています (Python 2 のサポートは 2020 年 1 月に終了しました)。詳細については、「Python 公式サイトのドキュメント」を参照してください。 また、Python のパッケージインストーラである pip がプリインストールされています。このコマンドラインプログラムを使用して、Python パッケージインデックスなどのオンラインインデックスから Python パッケージをインストールできます。詳細については、Python Packaging Authority が提供するドキュメントを参照してください。	• Python 3: `python3 --version` • pip: `pip3 --version`

名前	説明	バージョン情報
SDK for Python (Boto3)	Boto は、Python 開発者が Amazon EC2 や Amazon S3 などの作成、設定 AWS のサービス、管理に使用するソフトウェア開発キット (SDK) です。 Amazon S3 SDK は、easy-to-useオブジェクト指向 API と、 への低レベルのアクセスを提供します AWS のサービス。 詳細については、Boto3 ドキュメントを参照してください。	`pip3 list | grep boto3`

開発ツールおよびシェルユーティリティ

開発ツールおよびシェルユーティリティ

名前	説明	バージョン情報
bash-completion	bash-completion は、Tab キーを押して部分的に入力されたコマンドまたは引数の残りの自動入力を可能にするシェル機能の集まりです。`/usr/share/bash-completion/completions` で bash-completion がサポートするパッケージを見つけることができます。 パッケージのコマンドの自動入力を設定するには、プログラムファイルをソース	`dnf info bash-completion`

名前	説明	バージョン情報
	にする必要があります。例えば、Git コマンドのオートコンプリートを設定するには、AWS CloudShell セッションが開始するたびにこの機能を使用.bashrcできるように、次の行を に追加します。 `source /usr/share/bash-completion/completions/git` カスタム補完スクリプトを使用したい場合、それらを永続的なホームディレクトリ ($HOME) に追加して、.bashrc 内で直接ソースとします。 詳細については、GitHub でプロジェクトの README ページを参照してください。	
cqlsh-expansion	`cqlsh-expansion` は、Apache Cassandra との完全な互換性を維持し、Amazon Keyspaces 用に事前設定された cqlsh とヘルパーを含むツールキットです。詳細については、「Amazon Keyspaces (Apache Cassandra 向け) 開発者ガイド」の「cqlsh を使用して Amazon Keyspaces に接続する」を参照してください。	`cqlsh-expansion --version`

名前	説明	バージョン情報
Docker	Docker は、アプリケーションを開発、出荷、実行するためのオープンプラットフォームです。Docker を使用すると、アプリケーションをインフラストラクチャから分離できるため、ソフトウェアを迅速に配信できます。これにより、内部に Dockerfiles を構築し AWS CloudShell、CDK を使用して Docker アセットを構築できます。Docker でサポートされている AWS リージョンについては、「サポートされている AWS リージョン AWS CloudShell」を参照してください。環境内における Docker のスペースは限られていることに注意してください。個々のイメージが大きい場合や、既存の Docker イメージが多すぎる場合は、問題が発生する可能性があります。Docker の詳細については、Docker ドキュメントのガイドを参照してください。	`docker --version`

名前	説明	バージョン情報
Git	Git は、ブランチワークフローおよびコンテンツのステージングを介して、最新のソフトウェア開発プラクティスをサポートする分散バージョン管理システムです。詳細については、Git の公式サイトのドキュメントページを参照してください。	`git --version`
iputils	iputils パッケージには Linux ネットワーク用のユーティリティが含まれています。提供されるユーティリティの詳細については、「GitHub の iputils リポジトリ」を参照してください。	iputils ツールの例: `arping -V`
jq	jq ユーティリティは JSON 形式のデータを解析して、コマンドラインフィルタによって変更された出力を生成します。詳細については、GitHub でホストされている jq マニュアルを参照してください。	`jq --version`
kubectl	kubectl は、Kubernetes API を使用して Kubernetes クラスターのコントロールプレーンと通信するためのコマンドラインツールです。	`kubectl --version`

名前	説明	バージョン情報
make	make ユーティリティは makefiles を使用して、一連のタスクを自動化し、コードのコンパイルを整理します。詳細については、GNU Make のドキュメントを参照してください。	`make --version`
man	man コマンドは、コマンドラインユーティリティおよびツールのマニュアルページを提供します。例えば、`man ls` はディレクトリの内容を一覧表示する `ls` コマンドのマニュアルページを返します。詳細については、マンページの「Wikipedia エントリ」を参照してください。	`man --version`
nano	nano は、テキストベースのインターフェース用の小さくて使いやすいエディターです。詳細については、「GNU nano ドキュメント」を参照してください。	`nano --version`

名前	説明	バージョン情報
OpenJDK 21	Amazon Corretto 21 は、OpenJDK 21 の長期サポート (LTS) ディストリビューションです。Amazon Corretto は、Open Java Development Kit (OpenJDK) の、マルチプラットフォーム対応の本番稼働可能な、無償ディストリビューションです。詳細については、「Corretto 21 ユーザーガイド」の「Amazon Corretto 21 とは」を参照してください。	`java -version`
procps	procps は、現在実行中のプロセスをモニタリングおよび停止するために使用できるシステム管理ユーティリティです。詳細については、procps で実行できるプログラムをリストする README ファイルを参照してください。	`ps --version`
psql	PostgreSQL は、複雑なデータオペレーションを安全に管理およびスケーリングするための堅牢な機能を提供しながら、標準の SQL 機能を使用する強力なオープンソースデータベースシステムです。詳細については「PostgreSQL とは」を参照してください。	`psql --version`

名前	説明	バージョン情報
SSH クライアント	SSH クライアントは、リモートコンピュータとの暗号化通信にセキュアシェルプロトコルを使用します。OpenSSH は、プリインストールされている SSH クライアントです。詳細については、OpenBSD によって維持される OpenSSH サイトを参照してください。	`ssh -V`
sudo	sudo ユーティリティを使用すると、ユーザーは別のユーザー (通常はスーパーユーザー) のセキュリティ許可でプログラムを実行できます。Sudo は、システム管理者としてアプリケーションをインストールする必要がある場合に便利です。詳細については、「Sudo マニュアル」を参照してください。	`sudo --version`
tar	tar は、複数のファイルを単一のアーカイブファイル (tarball と呼ばれることが多い) にグループ化するために使用できるコマンドラインユーティリティです。詳細については、GNU tar ドキュメントを参照してください。	`tar --version`

名前	説明	バージョン情報
tmux	tmux は、複数のWindowsで異なるプログラムを同時に実行するために使用できるターミナルマルチプレクサです。詳細については、tmux の簡潔な紹介を提供するブログを参照してください。	`tmux -V`
vim	vim は、テキストベースのインターフェースを介して対話的な操作を可能にするカスタマイズ可能なエディタです。詳細については、vim.org で提供されるドキュメントリソースを参照してください。	`vim --version`
wget	wget は、コマンドラインでエンドポイントによって指定された ウェブ サーバーからコンテンツを取得するために使用されるコンピュータプログラムです。詳細については、GNU Wgetドキュメントを参照してください。	`wget --version`

名前	説明	バージョン情報

zip/enzip	zip/unzip ユーティリティは、データを失うことなくロスレスデータ圧縮を実現するアーカイブファイル形式を使用します。zip コマンドを呼び出して、単一のアーカイブ内のファイルをグループ化して圧縮します。unzip を使用して、アーカイブから指定したディレクトリにファイルを抽出します。	`unzip --version`
		
		`zip --version`

ホームディレクトリ AWS CLI への のインストール

CloudShell 環境にプリインストールされている他のソフトウェアと同様に、 AWS CLI ツールは、スケジュールされたアップグレードとセキュリティパッチで自動的に更新されます。のup-to-dateであることを確認する場合は AWS CLI、シェルのホームディレクトリにツールを手動でインストールすることを選択できます。

> **Important**
>
> CloudShell セッションを次回開始するときに使用できるように、 のコピーをホームディレクトリ AWS CLI に手動でインストールする必要があります。このインストールが必要なのは、$HOME の外部のディレクトリに追加されたファイルが、シェルセッションが終了すると削除されるためです。また、この のコピーをインストールした後は AWS CLI、自動的には更新されません。つまり、アップデートおよびセキュリティパッチを管理するのはユーザーの責任です。
> 責任 AWS 共有モデルの詳細については、「」を参照してくださいでのデータ保護 AWS CloudShell。

をインストールするには AWS CLI

1. CloudShell コマンドラインで、 `curl` コマンドを使用して、 AWS CLI インストールされた の圧縮コピーをシェルに転送します。

```
curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
```

2. zip フォルダを解凍します。

```
unzip awscliv2.zip
```

3. 指定したフォルダにツールを追加するには、 AWS CLI インストーラを実行します。

```
sudo ./aws/install --install-dir /home/cloudshell-user/usr/local/aws-cli --bin-
dir /home/cloudshell-user/usr/local/bin
```

正常にインストールされると、コマンドラインに次のメッセージが表示されます。

```
You can now run: /home/cloudshell-user/usr/local/bin/aws --version
```

4. また、独自の便宜のために、aws コマンド実行時にツールのインストールへのパスを指定する必要がないように、PATH 環境変数を更新することもお勧めします。

```
export PATH=/home/cloudshell-user/usr/local/bin:$PATH
```

> **Note**
>
> この変更を に元に戻すとPATH、指定されたパスを持たないawsコマンドは、 AWS CLI デフォルトで のプリインストール済みバージョンを使用します。

シェル環境へのサードパーティーソフトウェアのインストール

> **Note**
>
> サードパーティーアプリケーションを AWS CloudShellコンピューティング環境にインストールする前に、共有セキュリティ責任モデルを確認することをお勧めします。

デフォルトでは、すべての AWS CloudShell ユーザーに sudo アクセス許可があります。したがって、シェルのコンピューティング環境でまだ利用できないソフトウェアをインストールするために sudo コマンドを使用できます。例えば、DNF パッケージ管理ユーティリティで sudo を使用して

cowsay をインストールできます。これにより、次のメッセージ付きの牛の ASCII アート画像が生成されます。

```
sudo dnf install cowsay
```

次に、echo "Welcome to AWS CloudShell" | cowsay を入力して、新しくインストールしたプログラムを起動できます。

> **Important**
>
> dnf などのパッケージ管理ユーティリティは、プログラムをディレクトリ (/usr/bin など) にインストールします。各プログラムは、シェルセッションが終了するとリサイクルされます。つまり、セッションごとに追加のソフトウェアがインストールされ、使用されることを意味します。

スクリプトでシェルを修正する

デフォルトのシェル環境を変更する場合は、シェル環境が起動するたびに実行されるシェルスクリプトを編集できます。デフォルトの bash シェルが起動するたびに .bashrc スクリプトが実行されます。

> **Warning**
>
> .bashrc ファイルを誤って修正した場合、その後シェル環境にアクセスできないことがあります。編集する前にファイルのコピーを作成することをお勧めします。.bashrc の編集時にシェルを 2 つ開くことでリスクを軽減することもできます。一方のシェルでアクセスできなくなった場合でも、他のシェルにログインし、変更をロールバックできます。
> .bashrc やその他のファイルを誤って変更した後にアクセスが失われた場合は、ホームディレクトリを削除してデフォルト設定 AWS CloudShell に戻すことができます。

この手順では、シェル環境で自動的に Z シェルの実行に切り替わるように .bashrc スクリプトを変更します。

1. テキストエディタ (例:Vim) を使用して、.bashrc を開きます。

   ```
   vim .bashrc
   ```

2. エディタインターフェースで、I キーを押して編集を開始し、次に以下を追加します。

```
zsh
```

3. .bashrc ファイルを終了して保存するには、Esc を押して Vim コマンドモードを入力後、以下を入力します。

```
:wq
```

4. source コマンドを使用して .bashrc ファイルを再ロードする:

```
source .bashrc
```

コマンドラインインターフェイスが再び使用可能になると、プロンプトシンボルが % に変化して、Z シェルを使用していることを示します。

AWS CloudShell AL2 から AL2023 への移行

AWS CloudShellは、Amazon Linux 2 (AL2) に基づいていましたが、今後 Amazon Linux 2023 (AL2023) に移行します。AL2023 の詳細については、「Amazon Linux 2023 ユーザーガイド」の「Amazon Linux 2023 (AL2023) とは」を参照してください。

AL2023 では、CloudShell が提供するすべてのツールを使用して既存の CloudShell 環境に引き続きアクセスできます。利用可能なツールの詳細については、「プリインストールされたソフトウェア」を参照してください。

AL2023 では、Node .js 18 や Python 3.9 などの新しいバージョンのパッケージを含む、いくつかの改良が開発ツールに加えられています。

> **Note**
>
> AL2023 では、Python 2 は CloudShell 環境に標準装備されません。

AL2 および AL2023 間の主な相違点の詳細については、「Amazon Linux 2023 ユーザーガイド」の「Amazon Linux 2 と Amazon Linux 2023 の比較」を参照してください。

ご不明な点がある場合は、サポート までお問い合わせください。また、AWS re:Post で回答を検索し、質問を投稿することもできます。「」と入力すると AWS re:Post、 へのサインインが必要になる場合があります AWS。

AWS CloudShell 移行FAQs

以下は、 を使用した AL2 から AL2023 への移行に関する一般的な質問に対する回答です AWS CloudShell。

- AL2023 への移行は、AL2 で実行されている Amazon EC2 インスタンスなど、他の AWS リソースに影響しますか?
- AL2023 への移行に伴って変更されるパッケージにはどのようなものがありますか?
- 移行をオプトアウトすることはできますか?
- 自分の AWS CloudShell 環境のバックアップを作成できますか?

AL2023 への移行は、AL2 で実行されている Amazon EC2 インスタンスなど、他の AWS リソースに影響しますか?

AWS CloudShell 環境以外のサービスやリソースは、この移行の影響を受けません。これには、内部で作成またはアクセスした可能性のあるリソースが含まれます AWS CloudShell。例えば、AL2 で実行される Amazon EC2 インスタンスを作成した場合、これは AL2023 に移行されません。

AL2023 への移行に伴って変更されたパッケージにはどのようなものがありますか?

AWS CloudShell 現在、 環境にはプリインストールされたソフトウェアが含まれています。プリインストールされたソフトウェアの完全なリストについては、「プリインストールされたソフトウェア」を参照してください。 AWS CloudShell は、Python 2 を除き、これらのパッケージを引き続き配信します。AL2 と AL2023 によって提供されるパッケージの完全な違いについては、「AL2 と AL2023 の比較」を参照してください。AL2023 への移行後に特定のパッケージとバージョンの要件が満たされなくなった場合は、 AWS サポートに連絡してリクエストを送信することをお勧めします。

移行をオプトアウトすることはできますか?

いいえ、移行をオプトアウトすることはできません。 AWS CloudShell 環境は AWSによって管理されるため、すべての環境が AL2023 にアップグレードされています。

AWS CloudShell 環境のバックアップを作成できますか?

AWS CloudShell はユーザーのホームディレクトリを引き続き保持します。詳細については、「AWS CloudShellの Service Quotas と制限」を参照してください。ホームフォルダにファイルまたは設定が保存されていて、そのバックアップを作成する場合は、「ステップ 6: ホームディレクトリのバックアップを作成する」を実行します。

トラブルシューティング AWS CloudShell

の使用中に AWS CloudShell、CloudShell を起動したり、シェルコマンドラインインターフェイスを使用して主要なタスクを実行したりするなどの問題が発生する可能性があります。この章では、可能性のある一般的ないくつかの問題のトラブルシューティング方法を紹介します。

CloudShell に関するさまざまな質問への回答については、AWS CloudShell よくある質問をご覧ください。また、AWS CloudShell ディスカッションフォーラムで回答を検索したり、質問を投稿したりすることもできます。このフォーラムに入るには、 AWSにサインインする必要がある場合があります。当社に直接お問い合わせいただくこともできます。

エラーのトラブルシューティング

以下に挙げるインデックスに関するエラーが発生した場合、解決のために次の解決方法を使用することができますす。

トピック

- 拒否されるアクセス
- アクセス権限の不足
- AWS CloudShell コマンドラインにアクセスできない
- 外部 IP アドレスに ping できません
- ターミナルの準備中に問題が発生しました
- PowerShell で矢印キーが正しく機能しません
- サポートされていないウェブソケットが原因で CloudShell セッションを開始できない
- AWSPowerShell.NetCore モジュールをインポートできない。
- の使用時に Docker が実行されない AWS CloudShell
- Docker のディスク容量が不足している
- docker push がタイムアウトし、再試行し続ける
- VPC 環境から AWS CloudShell VPC 内のリソースにアクセスできない
- が VPC 環境 AWS CloudShell に使用する ENI がクリーンアップされていない
- VPC 環境のみのCreateEnvironmentアクセス許可を持つユーザーは、パブリック AWS CloudShell 環境にもアクセスできます。

拒否されるアクセス

問題: AWS マネジメントコンソールから CloudShell を起動しようとすると、次のメッセージが表示されます。「Unable to start the environment. To retry, refresh the browser or restart by selecting Actions, Restart AWS CloudShell」。IAM 管理者からのアクセス許可を要求し、ブラウザを更新したり CloudShell を再起動したりした後でも、アクセスが拒否されます。

解決策:AWS サポート部 にお問い合わせください。

(先頭に戻ります)

アクセス権限の不足

問題: AWS マネジメントコンソールから CloudShell を起動しようとすると、次のメッセージが表示されます。「Unable to start the environment. 必要なアクセス許可がありません。IAM 管理者に へのアクセスを許可するように依頼します AWS CloudShell」。アクセスを拒否され、必要なアクセス許可がないと通知されます。

原因: アクセスに使用している IAM ID に必要な IAM アクセス許可 AWS CloudShell がありません。

解決策: IAM 管理者に必要な権限の付与を申請してください。これを行うには、アタッチされた AWS 管理ポリシー (AWSCloudShellFullAccess) または埋め込みインラインポリシーを追加します。詳細については、「IAM ポリシーによる AWS CloudShell アクセスと使用状況の管理」を参照してください。

(先頭に戻ります)

AWS CloudShell コマンドラインにアクセスできない

問題: コンピューティング環境が使用するファイルを変更した後は、コマンドラインにアクセスできません AWS CloudShell。

解決策: `.bashrc`やその他のファイルを誤って変更した後にアクセスできなくなった場合は、ホームディレクトリを削除してデフォルト設定 AWS CloudShell に戻すことができます。

(先頭に戻ります)

外部 IP アドレスに ping できません

問題:コマンドライン (`ping amazon.com` など) から ping コマンドを実行すると、次のメッセージが表示されます。

```
ping: socket: Operation not permitted
```

原因: ping ユーティリティは、インターネット制御メッセージプロトコル (ICMP) を使用して、エコー要求パケットをターゲットホストに送信します。ターゲットからのエコーレスポンスを待ちます。ICMP プロトコルは で有効になっていないため AWS CloudShell、ping ユーティリティはシェルのコンピューティング環境で動作しません。

解決策: ICMP は でサポートされていないため AWS CloudShell、次のコマンドを実行して Netcat をインストールできます。Netcat は、TCP または UDP を使用してネットワーク接続に対して読み書きするためのコンピュータネットワークユーティリティです。

```
sudo yum install nc
nc -zv www.amazon.com 443
```

(先頭に戻ります)

ターミナルの準備中に問題が発生しました

問題: Microsoft Edge ブラウザ AWS CloudShell を使用して にアクセスしようとすると、シェルセッションを開始できず、ブラウザにエラーメッセージが表示されます。

原因: AWS CloudShell 以前のバージョンの Microsoft Edge と互換性がありません。サポートされているブラウザの最新の 4 つのメジャーバージョン AWS CloudShell を使用して にアクセスできます。

解決策: マイクロソフトのサイトから Edge ブラウザの最新版をインストールします。

(先頭に戻ります)

PowerShell で矢印キーが正しく機能しません

問題:通常の操作では、矢印キーを使用してコマンドラインインターフェースを操作したり、コマンド履歴を前後にスキャンすることができますできます。しかし、PowerShell の一部のバージョンでは、 AWS CloudShellで矢印キーを押すと、文字が正しく出力されない場合があります。

原因:矢印キーが誤って文字を出力する状況は、Linux上で実行されている PowerShell 7.2.x バージョンの問題としてすでに知られています。

解決策:矢印キーの動作を変更するエスケープシーケンスを削除するには、PowerShell プロファイルファイルを編集し、`$PSStyle` 変数を `PlainText` に設定します。

1. AWS CloudShell コマンドラインで、次のコマンドを入力してプロファイルファイルを開きます。

```
vim ~/.config/powershell/Microsoft.PowerShell_profile.ps1
```

> **Note**
>
> すでに PowerShell を使用している場合は、次のコマンドを使用してエディターでプロファイルファイルを開くこともできます。
>
> ```
> vim $PROFILE
> ```

2. エディターで、ファイルの既存のテキストの末尾に移動し、i を押して挿入モードに入り、次のステートメントを追加します。

```
$PSStyle.OutputRendering = 'PlainText'
```

3. 編集したら、Esc を押してコマンドモードに入力します。次に、次のコマンドを入力してファイルを保存し、エディタを終了します。

```
:wq
```

> **Note**
>
> 変更は、PowerShell の次回起動時に有効になります。

(先頭に戻ります)

サポートされていないウェブソケットが原因で CloudShell セッションを開始できない

問題: 起動しようとすると AWS CloudShell、 というメッセージが繰り返し表示されます`Failed to open sessions : Timed out while opening the session`。

原因: CloudShell は WebSocket プロトコルに依存します。これにより、ウェブブラウザと 間の双方向インタラクティブ通信が可能になります AWS CloudShell。プライベートネットワークでブラウザ

を使用している場合、プロキシサーバーとファイアウォールによってインターネットへの安全なアクセスが促進されていると考えられます。通常、WebSocket 通信は、問題なくプロキシサーバーを通過できます。しかし、場合によっては、プロキシサーバーが WebSockets の正常な動作を妨げることがあります。この問題が発生すると、CloudShell はシェルセッションを開始できず、接続を試みても最終的にタイムアウトになります。

解決策:接続タイムアウトは、サポートされていない WebSockets 以外の問題が原因である可能性があります。その場合は、まず CloudShell コマンドラインインターフェースがあるブラウザウィンドウを更新してください。

更新後もタイムアウトエラーが続く場合は、プロキシサーバーのドキュメントを参照してください。また、プロキシサーバーが Web ソケットを許可するように設定されていることを確認してください。または、ネットワークのシステム管理者に問い合わせてください。

> **Note**
>
> 特定の URL を許可リストに登録して、詳細な権限を定義したいとしましょう。 AWS Systems Manager セッションが入力を送信および受信するための WebSocket 接続を開くために使用する URL の一部を追加できます。 AWS CloudShell コマンドはその Systems Manager セッションに送信されます。
> Systems Manager が使用するこの StreamUrl の形式は `wss://ssmmessages.region.amazonaws.com/v1/data-channel/session-id?stream=(input|output)` です。
> リージョンは、 がサポート AWS リージョン する のリージョン識別子を表します AWS Systems Manager。例えば、`us-east-2` は米国東部 (オハイオ) のリージョン識別子です。
> セッション ID は特定の Systems Manager セッションが正常に開始された後に作成されるため、URL 許可リストを更新するときしか `wss://ssmmessages.region.amazonaws.com` を指定できません。詳細については、「AWS Systems Manager API リファレンス」の「StartSession」オペレーションを参照してください。

(先頭に戻ります)

AWSPowerShell.NetCore モジュールをインポートできない。

問題: PowerShell で、`Import-Module -Name AWSPowerShell.NetCore` を使って AWSPowerShell.netCore モジュールをインポートすると、次のエラーメッセージが表示されます。

Import-Module:どのモジュールディレクトリでも有効なモジュールファイルが見つからなかったため、指定されたモジュール「AWSPowerShell.NetCore」はロードされませんでした。

原因: AWSPowerShell.NetCoreモジュールは、 のサービスごとの AWSツールモジュールに置き換えられます AWS CloudShell。

解決策: 明示的なインポートステートメントが不要になったか、関連する per-service AWS.Tools モジュールに変更する必要がある場合があります。

Example

Example

- ほとんどの場合、.NET タイプが使用されていない限り、明示的なインポートステートメントは必要ありません。以下は、インポートステートメントの例です。
  - `Get-S3Bucket`
  - `(Get-EC2Instance).Instances`

- .NET タイプを使用する場合は、サービスレベルモジュール (`AWS.Tools.<Service>`) をインポートします。構文の例を次に示します。

```
Import-Module -Name AWS.Tools.EC2
$InstanceTag = [Amazon.EC2.Model.Tag]::new("Environment","Dev")
```

```
Import-Module -Name AWS.Tools.S3
$LifecycleRule = [Amazon.S3.Model.LifecycleRule]::new()
```

詳細については、 AWS Tools for PowerShellの バージョン 4 の告知を参照してください。

(先頭に戻ります)

の使用時に Docker が実行されない AWS CloudShell

問題: AWS CloudShellの使用時に Docker が適切に動作しません。メッセージ「`docker: Cannot connect to the Docker daemon at unix:///var/run/docker.sock. Is the docker daemon running?`」が表示されます。

解決策: 環境を再起動してみてください。このエラーメッセージは、GovCloud リージョン AWS CloudShell で で Docker を実行したときに発生する可能性があります。サポートされている AWS

リージョンで Docker が実行されていることを確認します。Docker が利用可能なリージョンのリストについては、「 でサポートされている AWS リージョン AWS CloudShell」を参照してください。

Docker のディスク容量が不足している

問題: エラーメッセージ「`ERROR: failed to solve: failed to register layer: write [...]: no space left on device`」が表示されます。

原因: Dockerfile が使用可能なディスク容量を超えています AWS CloudShell。これは、個々のイメージが大きいか、既存の Docker イメージが多すぎることが原因である可能性があります。

解決策: `df -h` を実行してディスクの使用状況を確認します。`sudo du -sh /folder/folder1` を実行して、大きいと思われる特定のフォルダのサイズを評価するとともに、他のファイルを削除してスペースを解放することを検討します。1 つのオプションとしては、`docker rmi` を実行して未使用の Docker イメージの削除を検討します。環境内における Docker のスペースは限られていることに注意してください。Docker の詳細については、Docker ドキュメントのガイドを参照してください。

`docker push` がタイムアウトし、再試行し続ける

問題: `docker push` を実行すると、タイムアウトになり、成功しないまま再試行を続けます。

原因: これは、アクセス許可の不足、間違ったリポジトリへのプッシュ、または認証の欠如が原因である可能性があります。

解決策: この問題を解決するには、正しいリポジトリにプッシュしていることを確認します。`docker login` を実行して適切に認証します。Amazon ECR リポジトリにプッシュするために必要なすべてのアクセス許可があることを確認します。

VPC 環境から AWS CloudShell VPC 内のリソースにアクセスできない

問題: VPC 環境の使用中に AWS CloudShell VPC 内のリソースにアクセスできない。

原因: AWS CloudShell VPC 環境は VPC のネットワーク設定を継承します。

解決策: この問題を解決するには、リソースにアクセスできるように VPC が正しく設定されていることを確認します。詳細については、VPC ドキュメントの「VPC を他のネットワークに接続する」および Network Access Analyzer ドキュメントの「Network Access Analyzer」を参照してください。 AWS CloudShell VPC 環境が使用している IPv4 アドレスは、コマンドラインプロンプトの環境`ip -a`内で コマンドを実行するか、VPC コンソールページで確認できます。

が VPC 環境 AWS CloudShell に使用する ENI がクリーンアップされていない

問題: VPC 環境で AWS CloudShell が使用している ENI をクリーンアップできません。

原因: ロールの `ec2:DeleteNetworkInterface` アクセス許可が有効になっていません。

解決策: この問題を解決するには、次のサンプルスクリプトに示すように、ロールの `ec2:DeleteNetworkInterface` アクセス許可を必ず有効にします。

```
{
  "Effect": "Allow",
  "Action": [
    "ec2:DeleteNetworkInterface"
  ],
  "Condition": {
    "StringEquals": {
      "aws:ResourceTag/ManagedByCloudShell": ""
    }
  },
  "Resource": "arn:aws:ec2:*:*:network-interface/*"
}
```

VPC 環境のみの**CreateEnvironment**アクセス許可を持つユーザーは、パブリック AWS CloudShell 環境にもアクセスできます。

問題: VPC 環境のみの`CreateEnvironment`アクセス許可で制限されたユーザーは、パブリック AWS CloudShell 環境にアクセスすることもできます。

原因: `CreateEnvironment` アクセス許可を VPC 環境の作成のみに制限したときに、CloudShell のパブリック環境が既に作成済みである場合は、このパブリック環境をウェブユーザーインターフェイスを使用して削除するまで、引き続きパブリック環境にアクセスできます。ただし、以前に CloudShell を使用したことがない場合は、パブリック環境にアクセスできません。

解決策: パブリック AWS CloudShell 環境へのアクセスを制限するには、IAM 管理者はまず IAM ポリシーを制限付きで更新し、次にウェブ AWS CloudShell ユーザーインターフェイスを使用して既存のパブリック環境を手動で削除する必要があります。([アクション] → [CloudShell 環境を削除])。

でサポートされている AWS リージョン AWS CloudShell

が AWS CloudShell サポートする AWS リージョンとサービスエンドポイントの詳細については、次のリソースを参照してください。

- AWS リージョン
- サービスエンドポイント

のサービスクォータと制限 AWS CloudShell

このページでは、以下のエリアに適用される Service Quotas と制限について説明します。

永続ストレージ

を使用すると AWS CloudShell、 ごとに 1 GB の永続的ストレージ AWS リージョン を無料で利用できます。永続的ストレージはホームディレクトリ ($HOME) にあり、プライベートです。各シェルセッションが終了した後にリサイクルされるエフェメラル環境リソースとは異なり、ホームディレクトリ内のデータはセッション間で保持されます。

> **Note**
>
> CloudShell の VPC 環境には永続ストレージはありません。$HOME ディレクトリは、VPC 環境がタイムアウトするか (20～30 分間非アクティブ状態が続いた後)、環境を削除すると、削除されます。

AWS CloudShell で の使用を停止すると AWS リージョン、データは最後のセッションの終了から 120 日間、そのリージョンの永続ストレージに保持されます。アクションを実行しない限り、データは 120 日後にそのリージョンの永続的ストレージから自動的に削除されます。その AWS CloudShell で AWS リージョンを再度起動すれば削除を防止することができます。詳細については、「ステップ 2: リージョンの選択、起動 AWS CloudShell、シェルの選択」を参照してください。

> **Note**
>
> 使用シナリオ

Márcia は AWS CloudShell を使用して、米国東部 (バージニア北部) と欧州 (アイルランド) AWS リージョンの 2 つのホームディレクトリにファイルを保存しています。その後、欧州 (アイルランド) AWS CloudShell でのみ の使用を開始し、米国東部 (バージニア北部) でのシェルセッションの起動を停止しました。
米国東部 (バージニア北部) でデータを削除する期限が切れる前に、Márcia は米国東部 (バージニア北部) リージョンを再度起動 AWS CloudShell して選択することで、ホームディレクトリがリサイクルされないようにすることにしました。彼女はヨーロッパ (アイルランド) でシェルセッションを継続しているので、そのリージョンの永続的ストレージは影響を受けません。

毎月の使用状況

AWS リージョン の各 AWS アカウント には、毎月の使用クォータがあります AWS CloudShell。このクォータは、そのリージョンのすべての IAM プリンシパルが CloudShell を使用した合計時間を組み合わせたものです。リージョンの月間クォータに達した後で CloudShell にアクセスしようとすると、シェル環境を起動できない理由を説明するメッセージが表示されます。

Service Quotas コンソールを使用して引き上げをリクエストするには

Service Quotas コンソールを開くと、毎月の使用クォータの引き上げをリクエストできます。詳細については、「Service Quotas ユーザーガイド」の「クォータの引き上げのリクエスト」を参照してください。

同時シェル数

アカウント AWS リージョン ごとに最大 10 個のシェルを同時に実行できます。

Service Quotas コンソールを使用して引き上げをリクエストするには

Service Quotas コンソールを開くと、各リージョンのクォータの増加をリクエストできます。詳細については、「Service Quotas ユーザーガイド」の「クォータの引き上げのリクエスト」を参照してください。

コマンドサイズ

コマンドサイズは 65,412 文字を超過することはできません。

> **Note**
>
> 65,412 文字を超えるコマンドを実行する場合は、選択した言語でスクリプトを作成し、コマンドラインインターフェースから実行してください。コマンドラインインターフェースからアクセス可能な、幅広いプリインストール機能の詳細については、「プリインストールされたソフトウェア」を参照してください。
> スクリプトを作成してコマンドラインインターフェースから実行する方法の例については、「チュートリアル: AWS CloudShellの使用開始」を参照してください。

シェルセッション

- 非アクティブなセッション: AWS CloudShell はインタラクティブなシェル環境です。キーボードまたはポインタを使用して 20～30 分間操作しないと、シェルセッションは終了します。実行中のプロセスは、操作数としてカウントされません。

  より柔軟なタイムアウトで AWS サービスを使用して、ターミナルベースのタスクを実行する場合は、Amazon EC2 インスタンスを起動して接続することをお勧めします。
- 実行時間が長いセッション: 約 12 時間連続して実行するシェルセッションは、ユーザーがその期間に定期的に操作している場合でも、自動的に終了します。

VPC 環境

IAM プリンシパルごとに最大 2 つの VPC 環境を作成できます。

> **Note**
>
> プライベート VPC に接続してその内部のリソースにアクセスしても料金はかかりません。プライベート VPC 内のデータ転送は VPC 請求に含まれ、CloudShell を介した VPC 間のデータ転送には、現在の CloudShell と同じ料金がかかります。

ネットワークアクセスおよびデータ転送

以下の制限は、 AWS CloudShell 環境のインバウンドおよびアウトバウンドのトラフィックに適用されます。

- アウトバウンド: 公開インターネットにアクセスできます。
- インバウンド: インバウンドポートにアクセスできません。公開 IP アドレスは使用できません。

> **Warning**
>
> パブリックインターネットにアクセスすると、特定のユーザーが AWS CloudShell 環境からデータをエクスポートするリスクがあります。IAM 管理者は、IAM ツールを通じて信頼された AWS CloudShell ユーザーの許可リストを管理することをお勧めします。特定のユーザーが明示的にアクセスを拒否される方法については、「カスタムポリシー AWS CloudShell を使用して で許可されるアクションを管理する」を参照してください。

データ転送: 大きなファイルでは、ファイルのアップロードとダウンロードが遅く AWS CloudShell なる場合があります。または、シェルのコマンドラインインターフェイスを使用して Amazon S3 バケットから環境にファイルを転送することもできます。

システムファイルとページの再ロードの制限

- システムファイル: コンピューティング環境に必要なファイルを誤って変更すると、 AWS CloudShell 環境へのアクセス時または使用時に問題が発生する可能性があります。この場合、アクセスを取り戻すには、ホームディレクトリの削除が必要になることがあります。
- ページの再ロード: AWS CloudShell インターフェースを再ロードするには、オペレーティングシステムのデフォルトのショートカットキーシーケンスの代わりに、ブラウザの更新ボタンを使用してください。

AWS CloudShell ユーザーガイドのドキュメント履歴

最新の更新

以下の表は、AWS CloudShell ユーザーガイド の重要な変更点をまとめたものです。

変更	説明	日付
の Amazon Q CLI AWS CloudShell	AWS CloudShellで Amazon Q CLI 機能を使用するためのサポートを追加しました。	2024 年 10 月 2 日
特定のリージョン AWS CloudShell での の Amazon VPC サポート	特定のリージョンでの AWS CloudShell VPC 環境の作成と使用のサポートが追加されました。	2024 年 6 月 13 日
AWS CloudShell ユーザーガイドに新しいチュートリアルが追加されました	内に Docker コンテナを構築し、Amazon ECR リポジトリに AWS CloudShell プッシュする方法と、 を介して Lambda 関数をデプロイする方法を詳しく説明した 2 つの新しいチュートリアルが追加されました AWS CDK。	2023 年 12 月 27 日
特定のリージョン AWS CloudShell で でサポートされている Docker コンテナ	の Docker コンテナのサポート AWS CloudShell が特定のリージョンに追加されました。	2023 年 12 月 27 日
AWS CloudShell が Amazon Linux 2023 (AL2023) の使用に移行	AWS CloudShell は AL2023 を使用し、Amazon Linux 2 から移行しました。	2023 年 12 月 4 日
の新しい AWS リージョン AWS CloudShell	AWS CloudShell が、次の AWS リージョンで一般利用可能になりました。	2023 年 6 月 16 日

	• 米国西部 (北カリフォルニア) • アフリカ (ケープタウン) • アジアパシフィック (香港) • アジアパシフィック (大阪) • アジアパシフィック (ソウル) • アジアパシフィック (ジャカルタ) • アジアパシフィック (シンガポール) • 欧州 (パリ) • 欧州 (ストックホルム) • ヨーロッパ (ミラノ) • 中東 (バーレーン) • 中東 (アラブ首長国連邦)	
AWS CloudShell で を起動する Console Toolbar	CloudShell を選択することで、コンソールの左下にある Console Toolbar で CloudShell を起動します。	2023 年 3 月 28 日
の新しい AWS リージョン AWS CloudShell	AWS CloudShell が次の AWS リージョンで利用可能になりました。 • カナダ (中部) • 欧州 (ロンドン) • 南米 (サンパウロ)	2022 年 10 月 6 日
AWS CloudShell 米国 AWS GovCloud でサポート	AWS CloudShell が AWS GovCloud (米国) リージョンでサポートされるようになりました。	2022 年 1 月 29 日

セキュリティに関するよくある質問	セキュリティ問題に関するその他のよくある質問。	2022 年 4 月 14 日
Web ソケット	ネットワーク要件に、CloudShell による WebSocket プロトコルの使用について説明するセクションを追加しました。	2022 年 3 月 21 日
PowerShell の矢印キーのトラブルシューティング	手順に従って、押したときに文字が正しく出力されない矢印キーを修正します。	2022 年 2 月 7 日
Tab キーによる自動入力	bash-completion の使用方法を説明した新しいドキュメントは、タブキーを押すことで、部分的に型付けされたコマンドまたは引数の自動補完を可能にします。	2021 年 9 月 24 日
AWS リージョンの指定	AWS CLI コマンド AWS リージョン のデフォルト指定に関するドキュメント。	2021 年 5 月 11 日
PDF および Kindle 版での書式設定	表セル内の画像サイズとテキストの修正。	2021 年 3 月 10 日

選択した AWS リージョン AWS CloudShell での の一般提供 (GA) リリース	AWS CloudShell が、次の AWS リージョンで一般利用可能になりました。 • 米国東部(オハイオ) • 米国東部 (バージニア北部) • 米国西部 (オレゴン) • アジアパシフィック (東京) • 欧州 (アイルランド) • アジアパシフィック (ムンバイ) • アジアパシフィック (シドニー) • 欧州 (フランクフルト)	2020 年 12 月 15 日