ユーザーガイド

AWS CloudShell

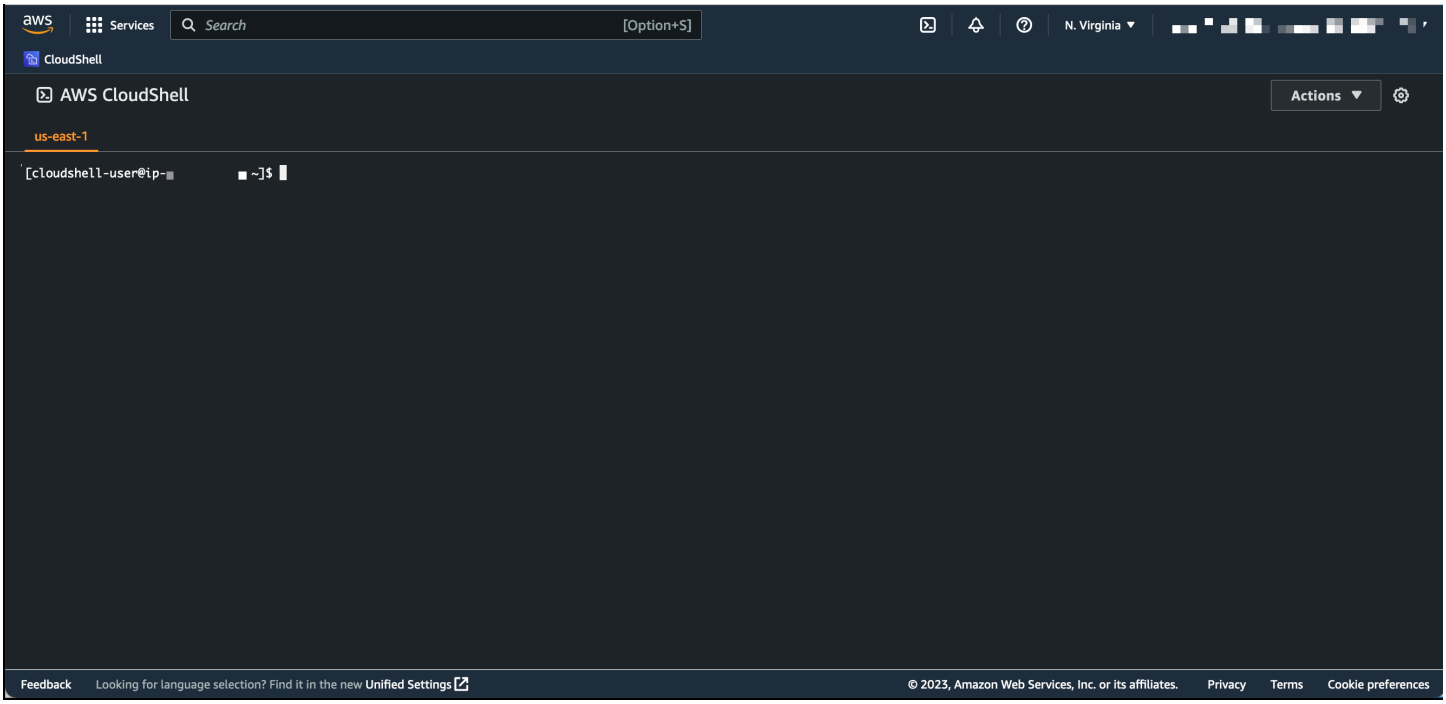
AWS CloudShell: ユーザーガイド

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 Management Console。には AWS Management Console 、さまざまな方法 CloudShell があります。詳細については、AWS CloudShellの開始方法を参照してください。

コマンドは、、Bash PowerShell、または 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。

(今すぐ試す: AWS CloudShell の開始方法)

AWS CloudShell features

このトピックでは、 コンソール CloudShell から を起動し、好みのコマンドラインシェルをシームレスに切り替えて、好み CloudShell に合わせてカスタマイズする方法について説明します。さらに、各 で最大 1 GB の永続ストレージを使用できます。また AWS リージョン、特定のセキュリティ機能で CloudShell 環境を保護する方法も可能です。

AWS Command Line Interface

AWS CloudShell から を起動できます AWS Management Console。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 ユーザーは事前に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AWS での AWS CloudShell サービスの使用。

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、 を切り替えることができますZ shell。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。つまり、例えば、最初にランタイム installations. PowerShell users を実行することなく、 Node.jsおよび Pythonプロジェクトを実行できます.NET Core。

ローカルリポジトリで作成またはアップロードされたファイルは AWS CloudShell 、 によって管理されるリモートリポジトリにプッシュする前にコミットできます AWS CodeCommit。

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

永続的ストレージ

では 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、「永続的ストレージ」を参照してください。

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、希望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 環境のカスタマイズ」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タートアップシェルスクリプトを変更すれば、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

セッション復元機能は、 CloudShell ターミナルの単一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

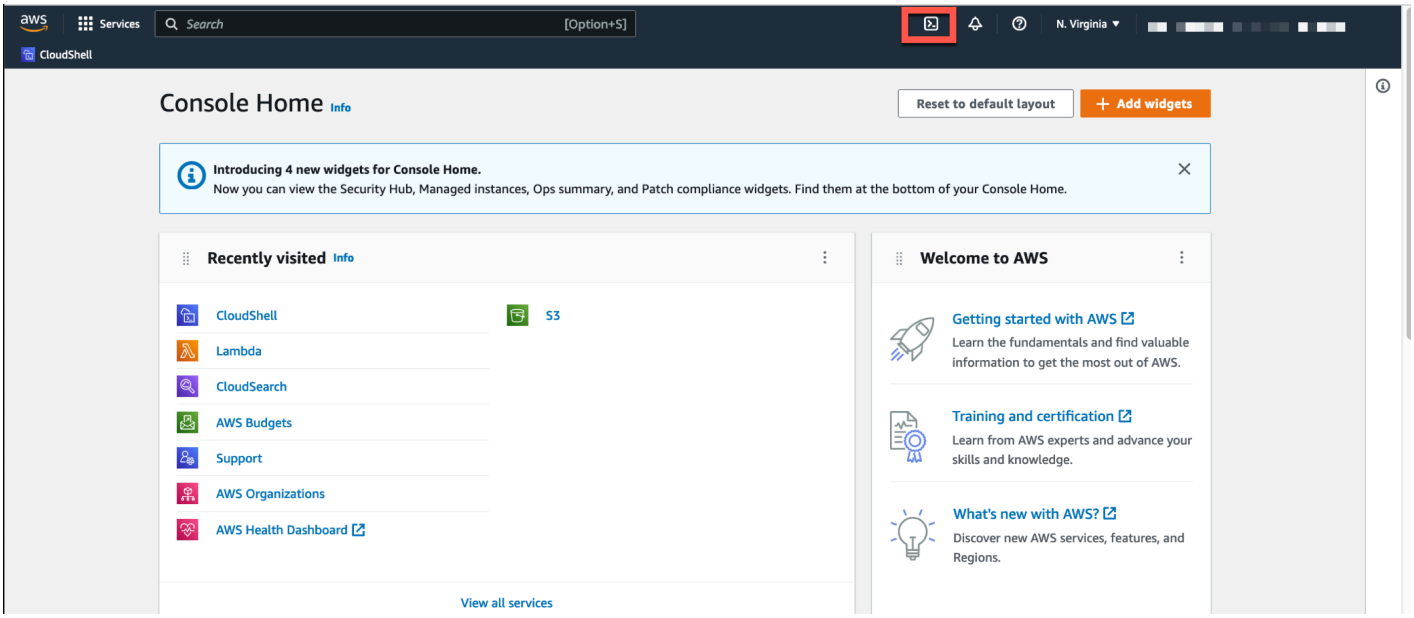
AWS CloudShell は追加料金なしで利用できる AWS のサービス です。ただし、 で実行する他の AWS リソースに対して料金が発生します AWS CloudShell。さらに、スタンダードデータ転送料金も適用されます。詳細については、「AWS CloudShell 料金」を参照してください。

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

の使用を開始する方法 AWS CloudShell

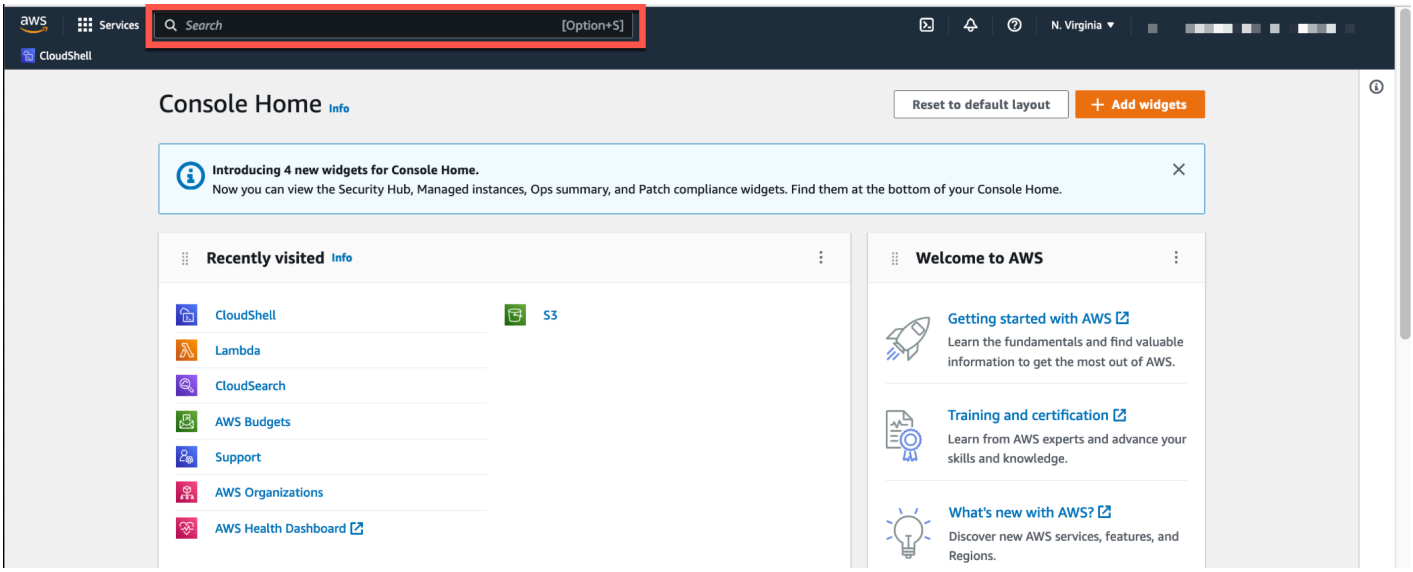
シェルの使用を開始するには、 にサインイン AWS Management Console し、次のいずれかのオプションを選択します。

- ナビゲーションバーで、 CloudShell アイコンを選択します。



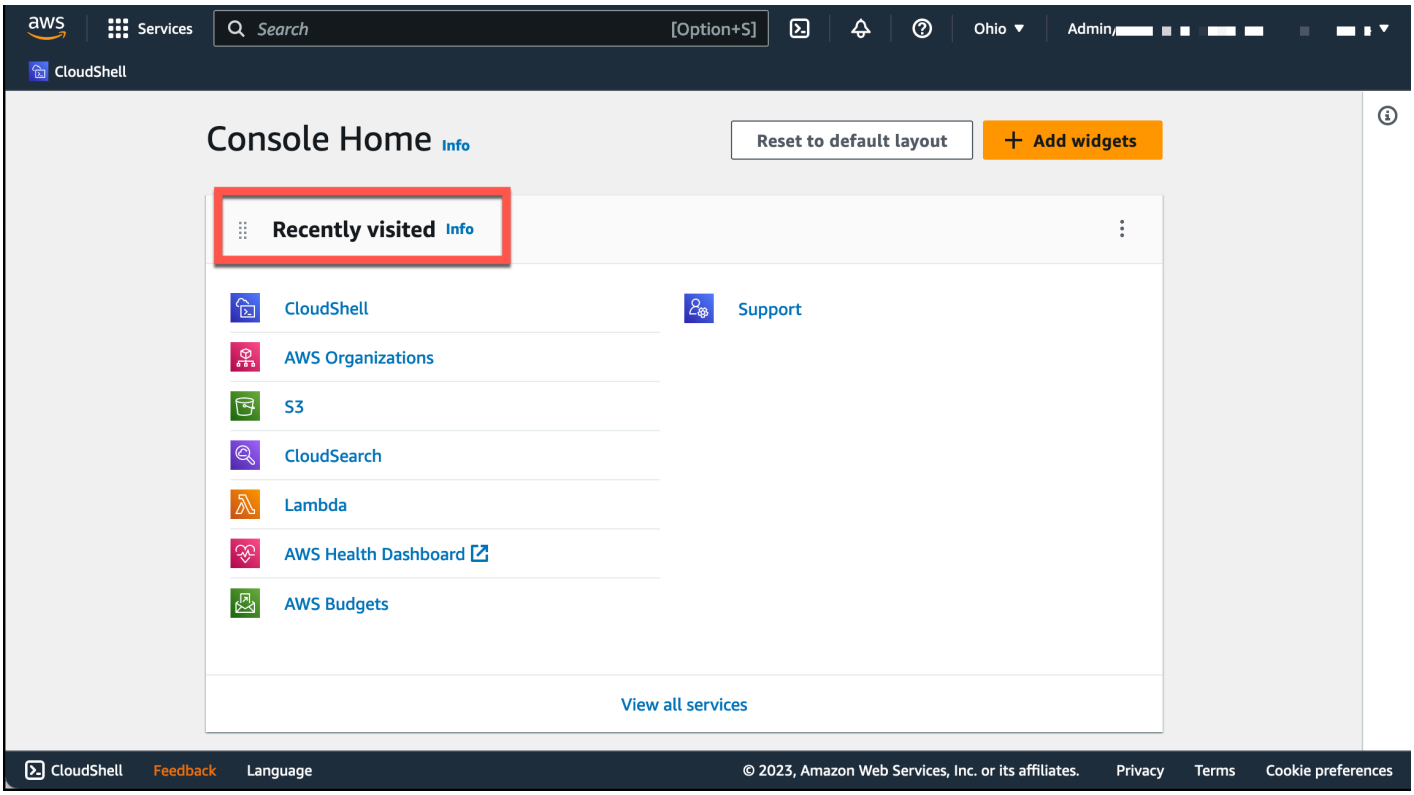
- 検索ボックスにCloudShell「」と入力し、 を選択しますCloudShell。

  このステップでは、 CloudShell セッションを全画面表示に開きます。

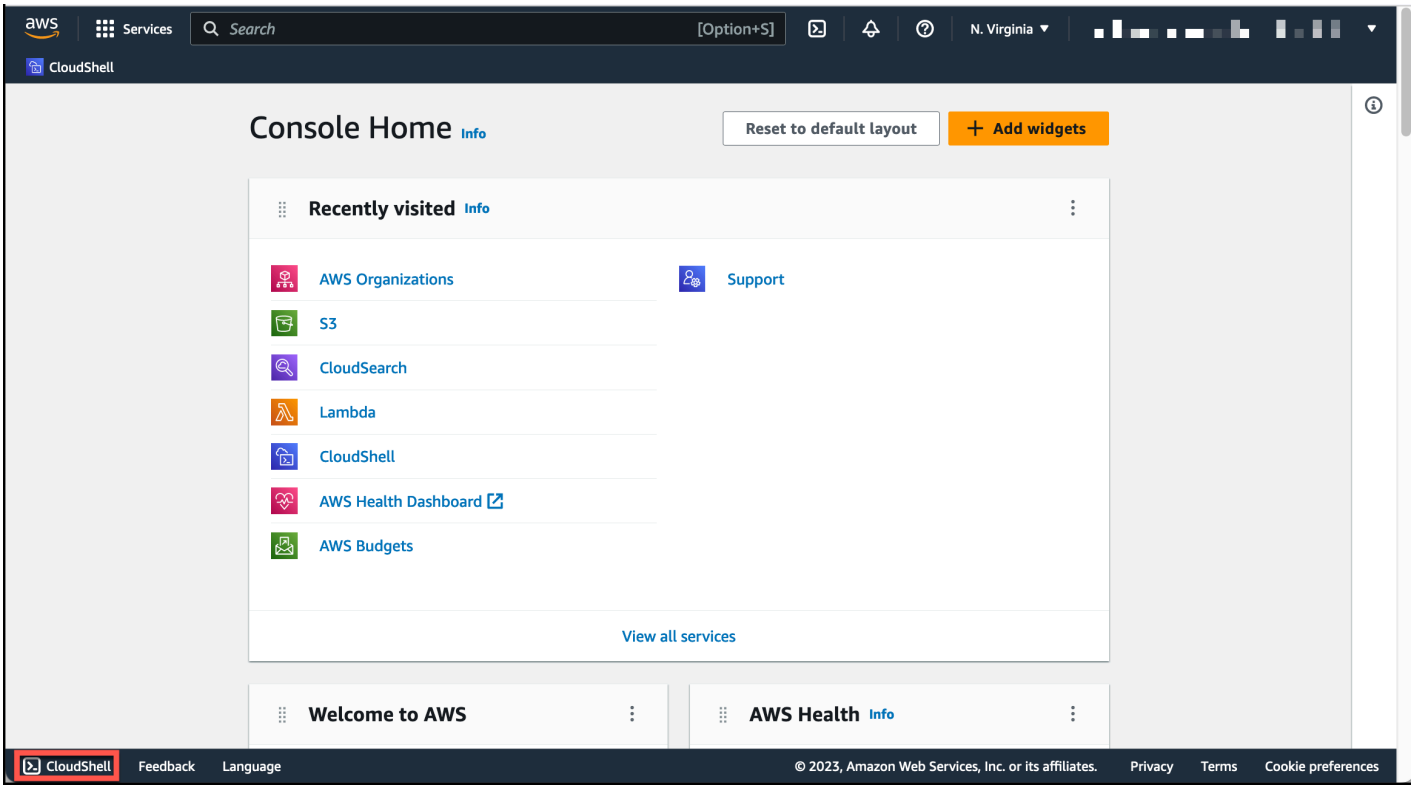


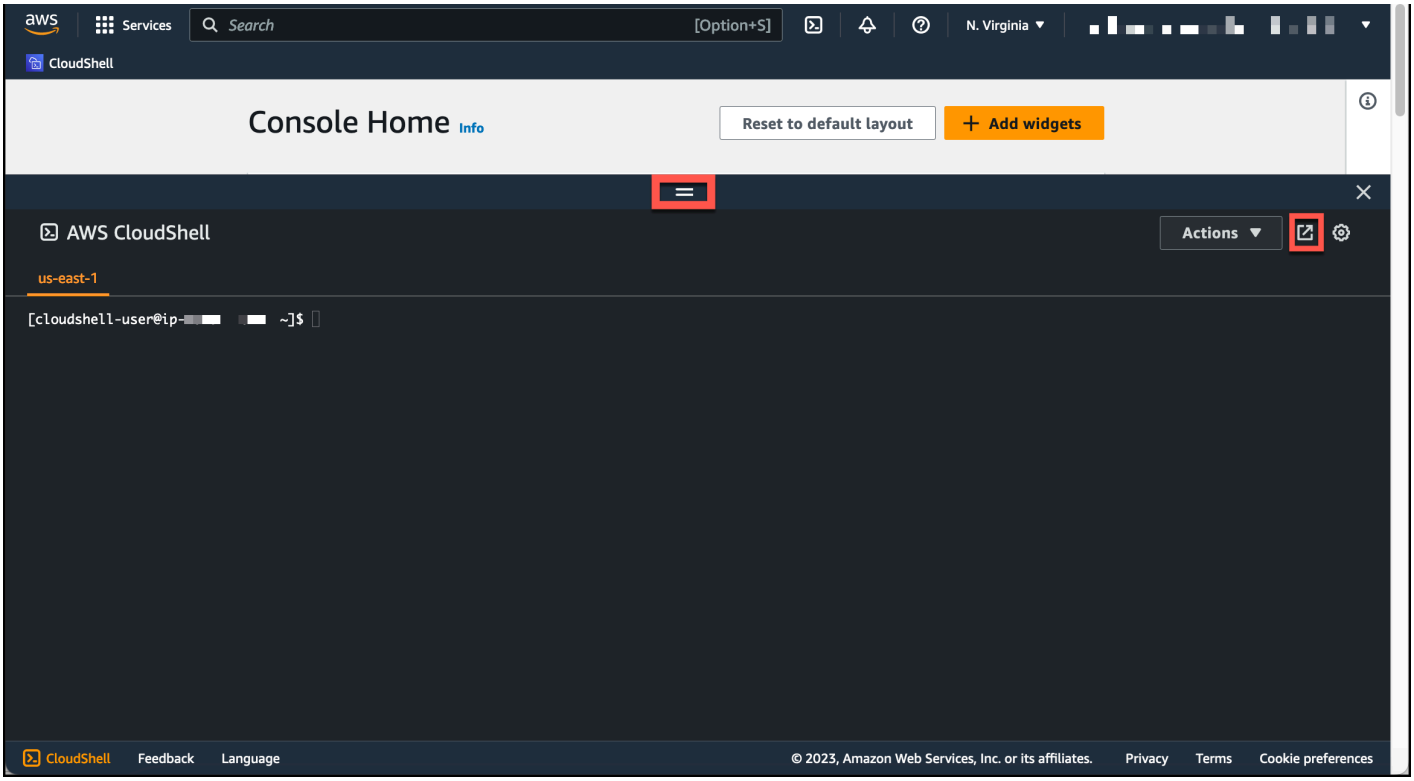
- 最近訪問したウィジェットで、 を選択しますCloudShell。

  このステップでは、 CloudShell セッションを全画面表示に開きます。



- コンソールの左下にある CloudShell Console Toolbar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 をクリックして、 CloudShell セッションを全画面表示に切り替えることもできます。

にサインイン AWS Management Console し、 で主要なタスクを実行する方法については AWS CloudShell、「 の開始方法 AWS CloudShell」を参照してください。

主な AWS CloudShell トピック

- AWS CloudShell の開始方法
- AWS CloudShell の使用
- AWS での AWS CloudShell サービスの使用
- AWS CloudShell 環境のカスタマイズ
- AWS CloudShell コンピューティング環境: 仕様およびソフトウェア

AWS CloudShell FAQs

以下は、 に関するいくつかの一般的な質問に対する回答です AWS CloudShell。

セキュリティに関するよくある質問については、「AWS CloudShell セキュリティFAQs」を参照してください。

- の使用を開始するにはどうすればよいですか AWS CloudShell？
- へのアクセスには何が必要ですか AWS CloudShell？
- AWS CloudShell の とは Console Toolbar
- AWS CloudShell で を起動するにはどうすればよいですかConsole Toolbar？
- AWS CloudShell 環境を作成および管理するにはどうすればよいですか？
- AWS リージョン どの で AWS CloudShell 利用できますか？
- CloudShell で を起動するときに、選択したリージョンで AWS CloudShell が利用できない場合、どの AWS リージョン が割り当てられますConsole Toolbarか？
- AWS CloudShellで使用できるシェルの種類は?
- ではどのウェブブラウザを使用できます AWS CloudShellか？
- AWS CloudShell で を起動するときに、どのウェブブラウザを使用できますかConsole Toolbar？
- AWS CloudShell で Console Toolbar を起動したときにファイルをダウンロードできますか?
- シェル環境にプリインストールされているソフトウェアは何ですか。
- シェル環境では利用できないソフトウェアをインストールできますか。

- AWS CloudShell内でユーザーが実行できるアクションを制限できますか?
- AWS リージョン を使用している を変更する場合、ホームディレクトリからデータを移動するにはどうすればよいですか AWS CloudShell？
- ユーザーがアクティブでないことにより、 AWS CloudShell がタイムアウトになる制限時間を延長することはできますか?
- の AWS CloudShell には、ホームページ AWS Console Mobile Application からアクセスできますか？
- AWS CloudShell で を起動するにはどうすればよいですか AWS Console Mobile Application？
- で を使用する場合、IOS キーボードと Android キーボード AWS CloudShell で修飾キーを使用できますか AWS Console Mobile Application？
- AWS CloudShell タブ表示を の複数のタブに分割できますか AWS Console Mobile Application？
- モバイルデバイス AWS CloudShell のコンソールツールバーで にアクセスできますか？

の使用を開始するにはどうすればよいですか AWS CloudShell？

の使用を開始するには、 から数ステップ AWS CloudShell で を起動します AWS Management Console。これを行うには、https://console.aws.amazon.com/console/home で AWS アカウント または IAM 認証情報を使用してコンソールにサインインします。

詳細については、「AWS CloudShellの開始方法」を参照してください。

へのアクセスには何が必要ですか AWS CloudShell？

AWS CloudShell から にアクセスするため AWS Management Console、有効な アカウントのエイリアスまたは ID、ユーザー名、パスワードを提供できる IAM ユーザーである必要があります。

コンソール AWS CloudShell で を起動するには、アタッチされたポリシーが提供する IAM アクセス許可が必要です。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。

AWS CloudShell の とは Console Toolbar

の左下にある CloudShell アイコン AWS Management Console。

AWS CloudShell で を起動するにはどうすればよいですかConsole Toolbar？

コンソールの左下にあるCloudShellアイコンConsole Toolbarを選択すると、 AWS CloudShell で を起動できます。

AWS リージョン どの で AWS CloudShell 利用できますか？

サポートされている AWS リージョン および関連するサービスエンドポイントのリストについては、「」のAWS CloudShell 「」ページを参照してくださいAmazon Web Services 全般のリファレンス。

CloudShell で を起動するときに、選択したリージョンで AWS CloudShell が利用できない場合、どの AWS リージョン が割り当てられますConsole Toolbarか？

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割り当てられます。詳細については、「リージョンの選択」、「」の起動 AWS CloudShell、「シェルの選択」を参照してください。

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、「 での作業 AWS リージョン」を参照してください。

AWS CloudShellで使用できるシェルの種類は?

では AWS CloudShell、Bash shell、 PowerShell、または を使用してコマンドを実行できますZ shell。シェルを切り替えるには、コマンドプロンプトで次の形式を使用して、使用するシェル名を入力します。

- `bash`: Bash shell を使用します
- `pwsh`: を使用する PowerShell
- `zsh`: Z shell を使用します

ではどのウェブブラウザを使用できます AWS CloudShellか？

AWS CloudShell は、Google Chrome、Mozilla Firefox、Microsoft Edge、Apple Safari ブラウザの最新バージョンをサポートしています。

AWS CloudShell 環境を作成および管理するにはどうすればよいですか？

AWS CloudShell 環境は、リージョンごとに IAM ユーザー ID ごとに作成および管理されます。を実行するUserIdと、 を確認できますaws sts get-caller-identity。環境は、その特定のリージョンの IAM ユーザー ID によって所有されます。IAM UserIdまたはリージョンを変更すると、別の AWS CloudShell 環境にアクセスできます。

AWS CloudShell で を起動するときに、どのウェブブラウザを使用できますかConsole Toolbar？

Google Chrome、Microsoft Edge、Mozilla Firefox、Apple Safari ブラウザの最新バージョンConsole Toolbarを使用して、 CloudShell で を起動できます。

で を起動するときにファイルをダウンロードできますか AWS CloudShellConsole Toolbar？

はい。 CloudShell で を起動するときにファイルをダウンロードできますConsole Toolbar。ファイルは、最新バージョンの Google Chrome および Microsoft Edge ブラウザを使用してダウンロードできます。

現在、Mozilla Firefox と Apple Safari ブラウザを使用してファイルをダウンロードすることはできません。

シェル環境にプリインストールされているソフトウェアは何ですか。

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 (Bash、 PowerShell、) をシームレスに切り替えることができますZ shell。Make 、pip 、sudo 、tar 、tmux 、Vim 、Wget 、Zip などのプリインストールされたツールやユーティリティにもアクセスできます。

シェル環境は、最も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。例えば、最初にランタイム installations. PowerShell users を実行することなく、 Node.jsおよび Pythonプロジェクトを実行できます。ユーザーは .NET Coreランタイムを使用できます。

シェルを使用して作成したか、またはシェルインターフェースでアップロードされたファイルは、git のプリインストールされたバージョンを使用して、バージョン管理されたリポジトリに追加できます。

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。

シェル環境では利用できないソフトウェアをインストールできますか。

はい。 AWS CloudShell ユーザーには sudo権限があり、コマンドラインからソフトウェアをインストールできます。詳細については、「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」を参照してください。

AWS CloudShell内でユーザーが実行できるアクションを制限できますか?

はい、ユーザーが AWS CloudShellで実行できるアクションを制御できます。例えば、 へのアクセスをユーザーに許可しても AWS CloudShell 、シェル環境内でのファイルのアップロードやダウンロードは禁止できます。あるいは、ユーザーが AWS CloudShellにアクセスすることを完全にできないように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。

AWS リージョン を使用している を変更する場合、ホームディレクトリからデータを移動するにはどうすればよいですか AWS CloudShell ?

AWS CloudShell データを 1 つのリージョンから AWS リージョン 別のリージョンに移動するには、まず 1 つのリージョンのホームディレクトリの内容をローカルマシンにダウンロードし、次にそこから別のリージョンのホームディレクトリにアップロードします。詳細については、「チュートリアル: ローカルマシンと AWS CloudShell の間で複数のファイルをコピーする」を参照してください。

ユーザーがアクティブでないことにより、 AWS CloudShell がタイムアウトになる制限時間を延長することはできますか?

キーボードまたはポインタ AWS CloudShell を使用して を操作しない場合、シェルセッションは約 20 ~ 30 分後に自動的に終了します。実行中のプロセスは、操作数としてカウントされません。CloudShell はタスクベースのアクティビティに集中するように設計されているため、現時点ではこのタイムアウト制限を増やす計画はありません。

より柔軟なタイムアウト AWS のサービス を持つ 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、クラウドベースの IDE、、AWS Cloud9または Amazon EC2 インスタンスの起動と接続を使用することをお勧めします。

の AWS CloudShell には、ホームページ AWS Console Mobile Application からアクセスできますか？

はい。コンソールモバイルアプリケーションにログイン AWS CloudShell AWS Console Mobile Application することで、 で にアクセスできます。詳細については、AWS Console Mobile Application ユーザーガイド を参照してください。

AWS CloudShell で を起動するにはどうすればよいですか AWS Console Mobile Application？

は AWS CloudShell 、次のいずれかの方法で起動できます。

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

で を使用する場合、iOS および Android キーボード AWS CloudShell で修飾キーを使用できますか AWS Console Mobile Application？

はい。iOS と Android のキーボードで修飾キーを使用できます。詳細については、「AWS コンソールモバイルアプリケーションユーザーガイド」を参照してください。

AWS CloudShell タブ表示を の複数のタブに分割できますか AWS Console Mobile Application？

いいえ。現在、モバイルアプリケーションで複数の AWS CloudShell タブを実行することはできません。

モバイルデバイス AWS CloudShell の Console Toolbar で にアクセスできますか？

いいえ。現在、モバイルデバイス AWS CloudShell の Console Toolbar では にアクセスできません。

AWS CloudShell の開始方法

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

まず、AWS Management Consoleにサインインして、を選択しますAWS リージョン。次に、CloudShell 新しいブラウザウィンドウが開き、使用するシェルタイプが表示されます。

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その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、あらかじめインストールされている工ディタを使用してそのファイルを編集します。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

AWS以下の管理ポリシーを IAM ID (ユーザー、ロール、グループなど) AWS CloudShell にアタッチすることで、のアクセス権限を取得できます。

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

このチュートリアルでは、AWS のサービスとの対話も行います。具体的には、S3 バケットを作成し、そのバケットにオブジェクトを追加することで Amazon S3 を操作します。IAM ID には、`s3:CreateBucket`と`s3:PutObject`少なくともおよびのアクセス権限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。

目次

ステップ 1: AWS Management Console にサインインします。

このステップでは、IAM ユーザー情報を入力して、AWS Management Consoleにアクセスします。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。

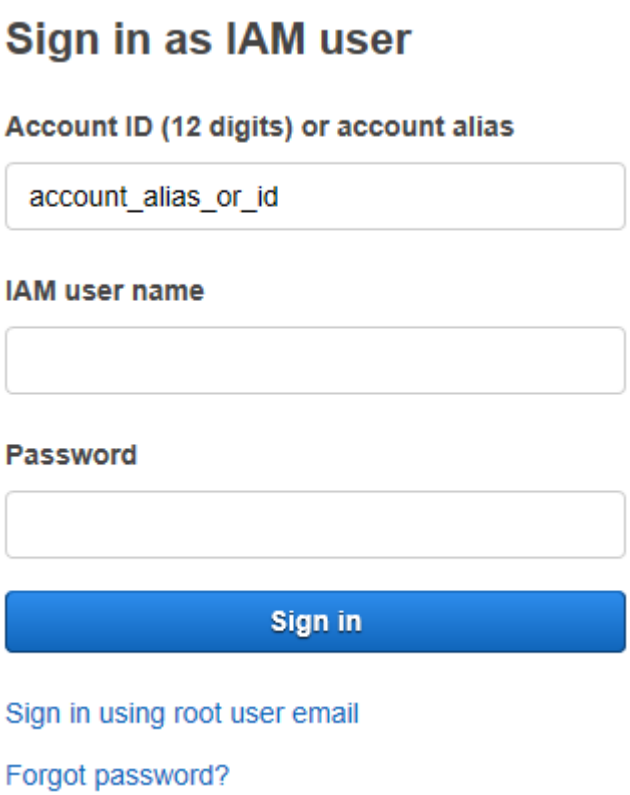
- IAM ユーザーのサインイン URL を使用するか、メインのサインインページに移動して、AWS Management Consoleにアクセスできます。

  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。管理者が提供したアカウントエイリアスまたはアカウント ID account_alias_or_id に置き換えます。

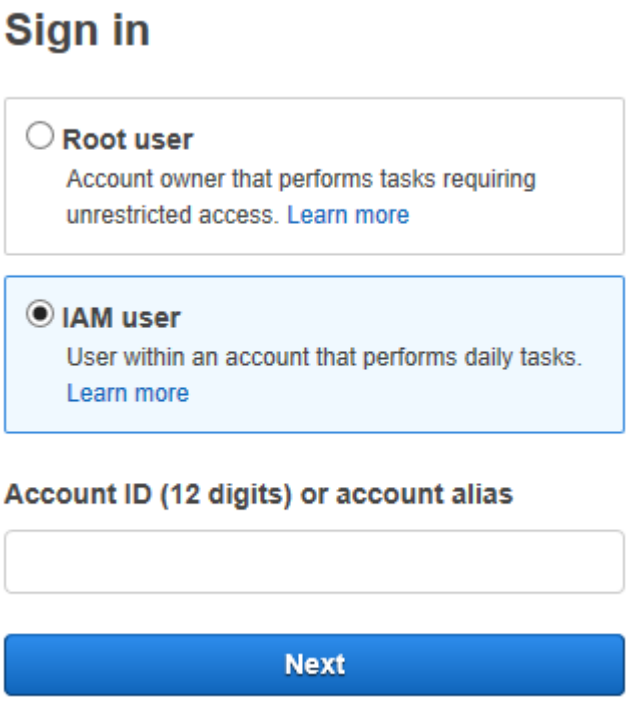
    ```
    https://account_alias_or_id.signin.aws.amazon.com/console/
    ```

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

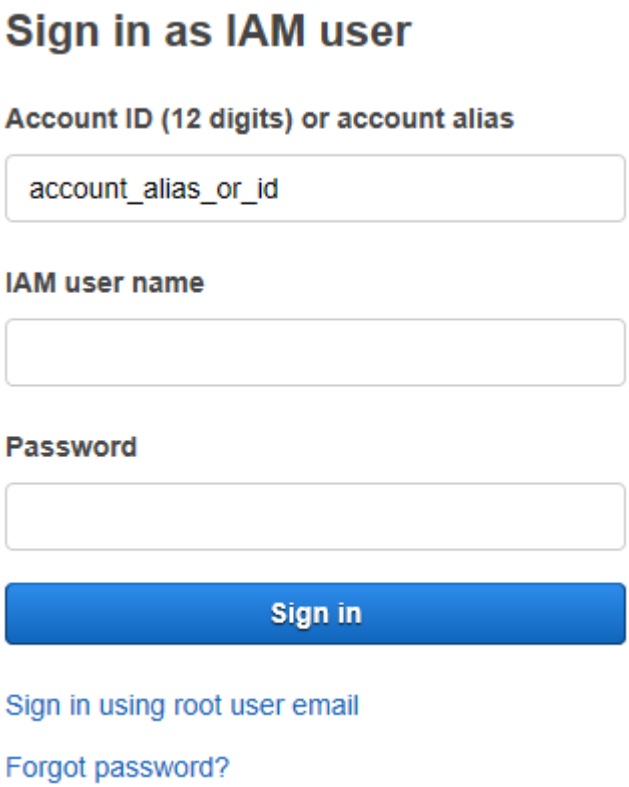


Main sign-in page

- https://aws.amazon.com/console/ を開きます。
- 以前にこのブラウザを使用してサインインしたことがない場合は、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。お使いのブラウザは、のアカウントエイリアスまたはアカウント ID AWS アカウント を記憶している可能性があります。その場合は、IAM サインイン認証情報を入力して [Sign in] を選択します。



> **Note**
>
> root 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、AWS のサービスアカウント内のすべての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: リージョンを選択し、起動してAWS CloudShell、シェルを選択します。

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

1. AWS リージョン作業する地域を選択するには、「地域を選択」 メニューに移動し、AWS作業対象としてサポートされている地域を選択します。(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更新され、AWS リージョン選択した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。永続ストレージに追加したファイルは、このストレージ内でのみ使用できます。AWS リージョンリージョンを変更すると、別のストレージやファイルにアクセスできるようになります。

> **Important**
>
> CloudShell コンソールの左下にあるを起動したときに選択したリージョンで利用できない場合、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。 CloudShell Console Toolbarデフォルトリージョンとは別のリージョンのリソースを管理する権限を付与するコマンドを実行できます。詳細については、「作業中」を参照してくださいAWS リージョン。

Example

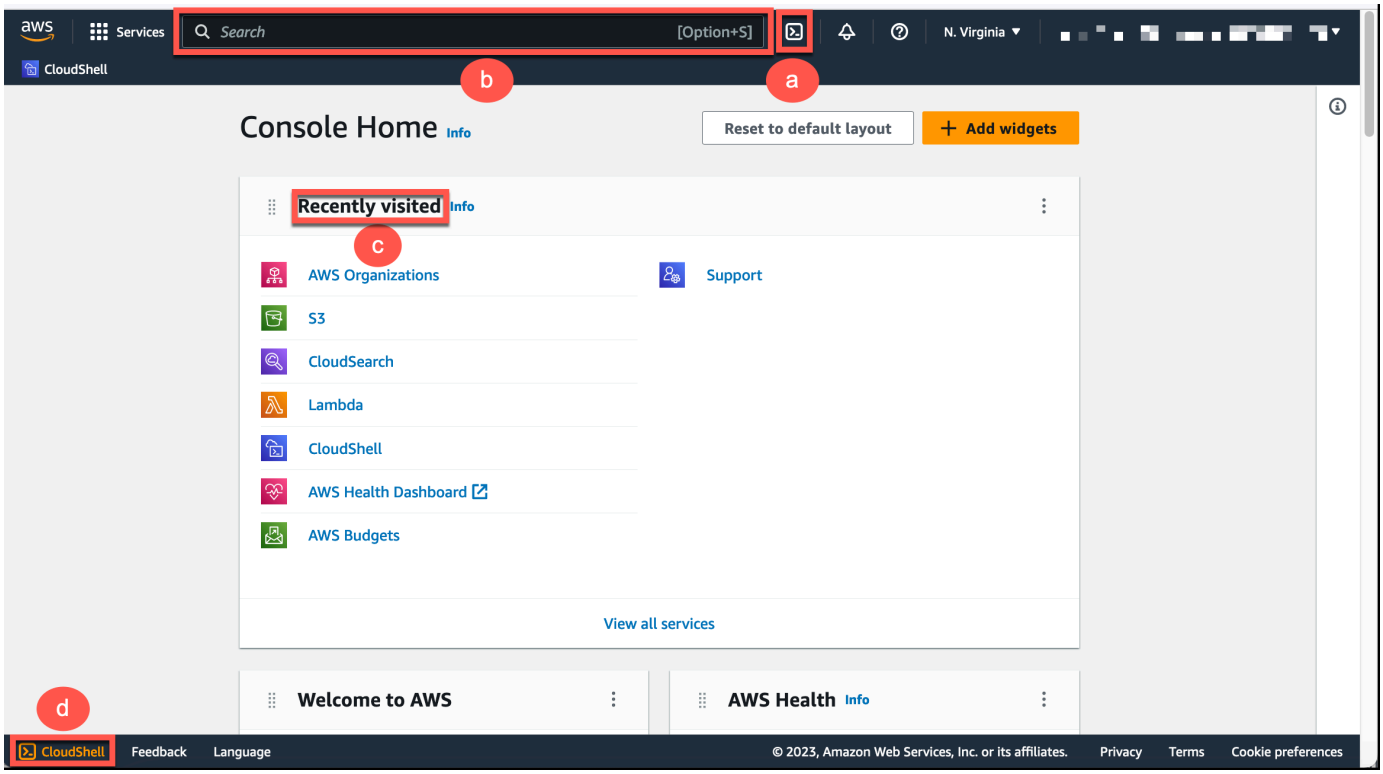
例

ヨーロッパ (スペイン) eu-south-2 を選択してもヨーロッパ (スペイン) では利用できない場合eu-south-2、デフォルトの地域はヨーロッパ (スペイン) eu-west-1 に最も近いヨーロッパ (アイルランド) eu-south-2 に設定されます。 CloudShell

デフォルトのリージョンのヨーロッパ (アイルランド) eu-west-1 のサービスクォータを使用すると、 CloudShell すべてのリージョンで同じセッションが復元されます。デフォルトのリージョンは変更される可能性があり、 CloudShell その場合はブラウザウィンドウに通知されます。

2. からAWS Management Console、 CloudShell 以下のオプションのいずれかを選択して起動できます。

   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、を選択しますCloudShell。
   3. 「最近訪れた」ウィジェットで、を選択しますCloudShell。
   4. コンソールの左下にあるを選択しますCloudShell。Console Toolbar
      - CloudShell セッションの高さを調整するには、ドラッグします=。
      - CloudShell セッションを全画面に切り替えるには、「新しいブラウザタブで開く」 アイコンをクリックします。



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**
>
> AWS CloudShell を正常に起動または操作できない問題が発生した場合、AWS CloudShell のトラブルシューティング でこ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。

3. プリインストールされているシェルを選択して使用するには、コマンドラインプロンプトにそのプログラム名を入力します。

   Bash

   ```
   bash
   ```

   に切り替えるとBash、$コマンドプロンプトのシンボルがに更新されます。

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

PowerShell

pwsh

に切り替えると PowerShell、コマンドプロンプトのシンボルがに更新されますPS>。

Z shell

zsh

に切り替えるとZ shell、コマンドプロンプトのシンボルがに更新されます%。

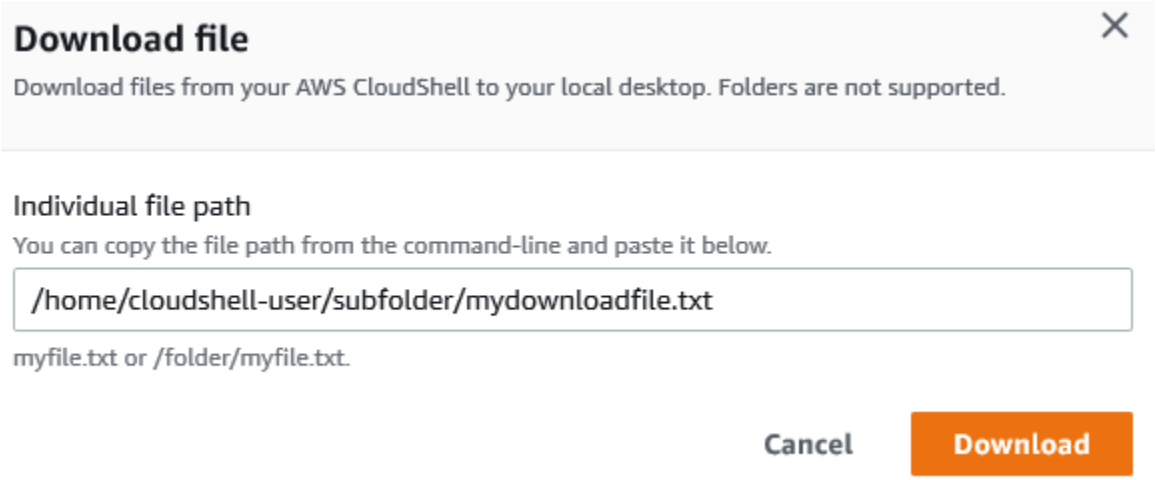
シェル環境にプリインストールされているバージョンについては、AWS CloudShell コンピュート環境セクションのシェルテーブルを参照してください。

ステップ 3: ファイルをダウンロードする AWS CloudShell

このステップでは、ファイルをダウンロードする手順を順を追って説明します。

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. ダウンロードファイルダイアログボックス、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。というわけで、というファイルをダウンロードするにはmydownload-file、以下の両方が有効なパスです。
>
> - 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ーに保存することもできます。

> **Note**
>
> CloudShell を起動しても、ダウンロードオプションは使用できませんConsole Toolbar。CloudShell コンソールまたは 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。ファイルのダウンロード方法について詳しくは、「ステップ 3: ファイルをダウンロードする」をご覧くださいAWS CloudShell。

ステップ 4: ファイルをにアップロードする AWS CloudShell

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の新しいディレクトリに移動する方法について説明します。

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 キーを押すと、シェルは現在の作業ディレクトリ (例:/home/cloudshell-user) を返します。

2. このディレクトリにファイルを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 [ファイルをアップロード] を選択します。

   [ファイルをアップロード] 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「ファイルアップロード」ダイアログで、このチュートリアル用に作成したテキストファイル (add_prog.py) を選択し、「開く」を選択します。

5. アップロードファイル 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。

   ワーキングディレクトリの内容が一覧表示されます。これには、アップロードしたばかりのファイルが含まれます。

ステップ 5: 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. ファイルを削除するにはAWS CloudShell、rm (remove) 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」というサフィックスを含むファイルをすべて削除します。

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

> **Note**
>
> AWS CloudShell特定の場所での使用をやめたとします。AWS リージョンその後、そのリージョンの永続ストレージにあるデータは、指定した期間が経過すると自動的に削除されます。詳細については、「永続ストレージ」を参照してください。

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

1. バックアップファイルを作成する

   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)
   ```

   以下のオプションのいずれかを使用してバックアップを作成できます。

   a. tar を使用してバックアップファイルを作成します。

      tar を使用してバックアップファイルを作成するには、以下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 を使用してバックアップファイルを作成します。

zip を使用してバックアップファイルを作成するには、以下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. バックアップファイルを外部に転送します。 CloudShell

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

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します。

前のステップで作成したファイルをダウンロードできます。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 CloudShell、「ファイルのダウンロード元」を参照してください AWS CloudShell。

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

b. バックアップファイルを S3 に転送します。

バケットを生成するには、以下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. S3 バケットへの直接Backup

   S3 バケットに直接バックアップするには、以下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

> **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。

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

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

2. 確認するには、[再起動] を選択します。

   インターフェースに、 CloudShell コンピューティング環境が停止中であることを示すメッセージが表示されます。環境を停止して再起動したら、新しいセッションでコマンドラインでの作業を開始できます。

> **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

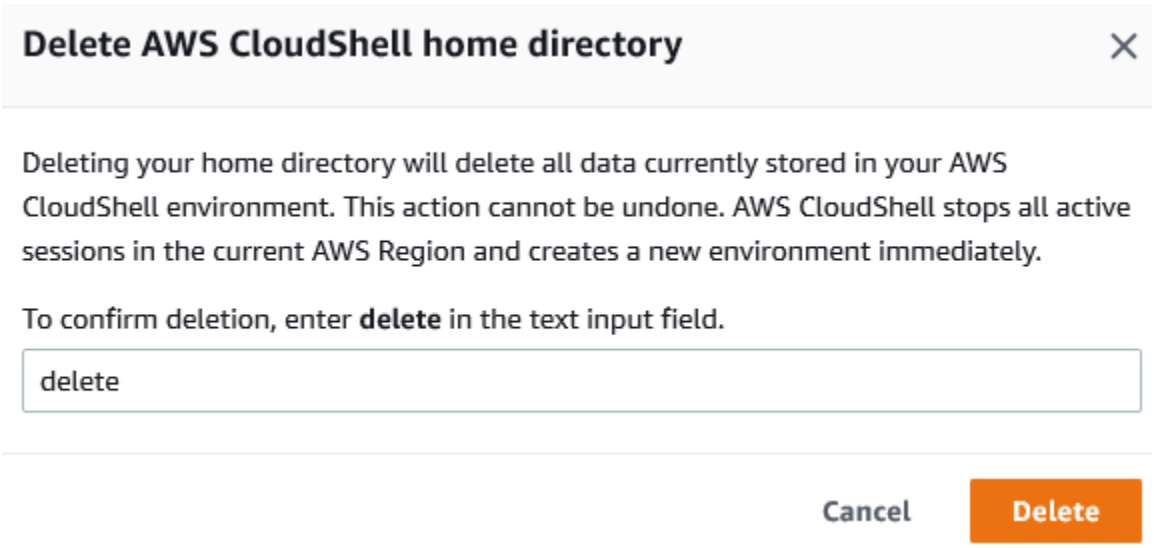
> **Warning**
>
> ホームディレクトリの削除は元に戻せないアクションで、ホームディレクトリに保存されているデータはすべて完全に削除されます。ただし、次のような場合には、このオプションを考慮してもよいでしょう。
>
> - ファイルを誤って変更したため、コンピューティング環境にアクセスできません。AWS CloudShellホームディレクトリを削除すると、AWS CloudShell がデフォルト設定に戻ります。
> - AWS CloudShell からすべてのデータをすぐに削除してください。AWS CloudShellリージョンでの使用を停止すると、AWSAWS CloudShellそのリージョンで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。
>
> ファイルの長期保存が必要な場合は、Amazon S3 やなどのサービスを検討してください CodeCommit。

1. シェルセッションを削除するには、[アクション]、[AWS CloudShell ホームディレクトリの削除] を選択します。

   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ることが通知されます。

   > **Note**
   >
   > このアクションは元に戻すことができません。

2. 削除を確認するには、テキスト入力フィールドに delete と入力し、[Delete] を選択します。



AWS CloudShell AWS リージョン は現在のアクティブなセッションをすべて停止し、すぐに新しい環境を作成します。

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します。

このステップでは、Vimプリインストールされているエディターを使用してファイルを操作する方法を示します。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更新され、エディターが表示されます。Vim
2. でファイルを編集するにはVim、Iキーを押します。次に、プログラムが 2 つの数字ではなく 3 つの数字を足す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。セーフペースト機能を使えば、ペーストする前にテキスト全体を検証できます。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、EscVimを押してコマンドモードに入ります。次に、`:wq`コマンドを入力してファイルを保存し、エディターを終了します。

> **Note**
>
> コマンドモードを初めて使用する場合、Vim最初はコマンドモードと挿入モードを切り替えるの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。挿入モードに入るにはを押しI、コマンドモードに入るにはを押しますEsc。Vimの詳細や、で使用できるその他のツールについては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. メインのコマンドラインインターフェイスで、次のプログラムを実行し、入力用に 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、ファイルを Amazon S3 バケットにオブジェクトとして追加します

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そのバケットに追加します。

> **Note**
>
> ほとんどの場合、CodeCommitソフトウェアファイルをバージョン管理されたリポジトリにコミットするなどのサービスを使用できます。このチュートリアルでは、AWS CLI in を使用して他の AWS AWS CloudShell サービスと連携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. 指定した場所にバケットを作成するにはAWS リージョン、以下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
> ```

呼び出しが成功すると、コマンドラインに次の出力のようなサービスからの応答が表示されます。

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

> **Note**
>
> バケットの命名規則に従わないと、次の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が発生しました (InvalidBucketName): 指定されたバケットは無効です。

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追加するには、メソッドを呼び出します。PutObject

```
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ると、コマンドラインには次の出力のようなサービスからの応答が表示されます。

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

ETagは保存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

- AWS での AWS CloudShell サービスの使用
- チュートリアル: ローカルマシンと AWS CloudShell の間で複数のファイルをコピーする
- チュートリアル: CodeCommit での使用AWS CloudShell
- AWS CloudShell の使用
- AWS CloudShell 環境のカスタマイズ

AWS CloudShell のチュートリアル

以下のチュートリアルでは、使用時にさまざまな機能や統合を試したりテストしたりできます。AWS CloudShell

トピック

チュートリアル: ローカルマシンと AWS CloudShell の間で複数のファイルをコピーする

CloudShell この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: ローカルマシンと間でファイルをコピーするときは、S3 CloudShell 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ので、現時点では、`scp`rsync CloudShell またはなどのコマンドを使用してローカルマシンと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

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`

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

1. AWS CloudShell 内で、`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。次のいずれかのオプションを選択して、ファイルをアップロードします。

   - AWS Management Console: drag-and-drop バケットにファイルとフォルダをアップロードするために使用します。
   - 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/ にある Amazon S3 コンソールを開きます。

(AWS CloudShell を使用する場合、コンソールに既にログインしている必要があります。)

- 左側のナビゲーションペインで、[バケット] を選択してから、フォルダまたはファイルのアップロード先のバケットの名前を選択します。Create bucket を選択して、お好みのバケットを作成することもできます。
- アップロードするファイルと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、ターゲット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選択内容をドラッグアンドドロップするか、[ファイルを追加] または [フォルダを追加] 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。
- 選択したファイルをバケットに追加するには、[アップロード] を選択します。

> **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**
>
> このオプションの場合、ローカルマシンに AWS CLI ツールをインストールし、AWS のサービスを呼び出せるように認証情報を設定しておく必要があります。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、指定したバケットをローカルマシン上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 コマンドラインで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します。

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

> **Note**
>
> また、`--exclude "<value>"--include "<value>"sync`コマンドにパラメータを追加してパターンマッチングを実行して、特定のファイルまたはオブジェクトを除外または含めることもできます。
> 詳細については、AWS CLIコマンドリファレンスの「除外フィルターと包含フィルターの使用」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 を使用して AWS CloudShell から複数のファイルをダウンロードする

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、シェル環境内の現在のディレクトリの内容に S3 バケットを同期します。

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

> **Note**
>
> また、`--exclude "<value>"--include "<value>"sync`コマンドにパラメータを追加してパターンマッチングを実行して、特定のファイルまたはオブジェクトを除外または含めることもできます。
> 詳細については、AWS CLIコマンドリファレンスの「除外フィルターと包含フィルターの使用」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**
   >
   > このオプションの場合、ローカルマシンに AWS CLI ツールをインストールし、AWS のサービスを呼び出せるように認証情報を設定しておく必要があります。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

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。 CloudShell ユーティリティは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 フォルダを使用して AWS CloudShell に複数のファイルをアップロードする

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. を起動し CloudShell、[アクション]、[ファイルのアップロード] を選択します。
3. [ファイルをアップロードする] ダイアログボックスで、[ファイルを選択] を選択してから、先ほど作成した zip フォルダを選択します。
4. [ファイルをアップロードする] ダイアログボックスで、[アップロード] を選択して、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 フォルダを使用して AWS CloudShell から複数のファイルをダウンロードする

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 (`/home/cloudshell-user/zip-folder/zipped-archive.zip`たとえば) 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。

チュートリアル: CodeCommit での使用AWS CloudShell

CodeCommit は、プライベート Git リポジトリをホストする、安全で高度にスケーラブルなマネージド型のソース管理サービスです。を使用するとAWS CloudShell、 CodeCommit git-remote-codecommitユーティリティを使用してコマンドラインで操作できます。このユーティリティは、AWS CloudShellコンピューティング環境を提供し、 CodeCommit リポジトリからコードをプッ

シュおよびプルするための簡単な方法を提供します。このユーティリティは Git を拡張することによって行われます。詳細については、AWS CodeCommit ユーザーガイドを参照してください。

このチュートリアルでは、 CodeCommit リポジトリを作成し、AWS CloudShellコンピューティング環境のクローンを作成する方法について説明します。また、AWS Cloud で管理されているリモートリポジトリにプッシュする前に、クローンしたリポジトリにファイルをステージングしてコミットする方法も説明します。

前提条件

IAM ユーザーがAWS CloudShell を使用するために必要なアクセス許可については、入門チュートリアルの「前提条件」セクションを参照してください。また、使用するには IAM 権限も必要です CodeCommit。

さらに、開始する前に次の項目が格納されていることを確認します。

- Git コマンドとバージョン管理の概念に関する基本事項の理解
- シェルのホームディレクトリ内のファイルで、ローカルおよびリモートリポジトリにコミットできます。このチュートリアルでは、`my-git-file` と表記します。

ステップ 1: CodeCommit リポジトリを作成してそのクローンを作成する

1. CloudShell コマンドラインインターフェイスで、`codecommit`次のコマンドを入力して、CodeCommit というリポジトリを作成しますMyDemoRepo。

   ```
   aws codecommit create-repository --repository-name MyDemoRepo --repository-
   description "My demonstration repository"
   ```

   リポジトリが正常に作成されると、コマンドラインにサービスのレスポンスが表示されます。

   ```
   {
       "repositoryMetadata": {
           "accountId": "111122223333",
           "repositoryId": "0dcd29a8-941a-1111-1111-11111111111a",
           "repositoryName": "MyDemoRepo",
           "repositoryDescription": "My demonstration repository",
           "lastModifiedDate": "2020-11-23T20:38:23.068000+00:00",
           "creationDate": "2020-11-23T20:38:23.068000+00:00",
           "cloneUrlHttp": "https://git-codecommit.eu-west-1.amazonaws.com/v1/repos/
   MyDemoRepo",
   ```

```
        "cloneUrlSsh": "ssh://git-codecommit.eu-west-1.amazonaws.com/v1/repos/
MyDemoRepo",
        "Arn": "arn:aws:codecommit:eu-west-1:111111111111:MyDemoRepo"
    }
 )
```

2. コマンドラインを使用して、ローカルリポジトリ用の新しいディレクトリを作成し、それを作業ディレクトリにします。

```
mkdir my-shell-repo
cd my-shell-repo
```

3. リモートリポジトリのクローンを作成するには、git cloneコマンドを使用します。(ここではgit-remote-codecommit、HTTPS (GRC) の URL スタイルを使用します)。

```
git clone codecommit::eu-west-1://MyDemoRepo
```

リポジトリのクローンが正常に作成されると、コマンドラインにサービスのレスポンスが表示されます。

```
Cloning into 'MyDemoRepo'...
warning: You appear to have cloned an empty repository.
```

4. リポジトリのクローンを作成するには、cdコマンドを使用します。

```
cd MyDemoRepo
```

ステップ 2: CodeCommit リポジトリにプッシュする前にファイルをステージングしてコミットする

1. MyDemoRepo の Vim エディタ、または AWS CloudShell のファイルアップロード機能を使用して、my-git-file というファイルをフォルダに追加します。両方の使い方については、「入門チュートリアル」を参照してください。
2. 作成したファイルをリポジトリにステージングするには、gitadd コマンドを実行します。

```
git add my-git-file
```

3. ファイルがステージングされ、コミットできる状態になったことを確認するには、gitstatus コマンドを実行します。

```
git status
```

my-git-file は、新しいファイルとしてリストされ、コミットする準備ができていることを示す緑色のテキストで表示されます。

4. ステージングされたファイルのこのバージョンをリポジトリにコミットします。

```
git commit -m "first commit to repo"
```

> **Note**
>
> コミットを完了するための設定情報の入力を指示される場合は、次の形式を使用します。
>
> ```
> $ git config --global user.name "Jane Doe"
> $ git config --global user.email janedoe@example.com
> ```

5. ローカルリポジトリに加えた変更とリモートリポジトリを同期するには、変更内容をアップストリームブランチにプッシュします。

```
git push
```

チュートリアル: AWS 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ポリシーによって提供されるアクセス権限を持つ IAM ユーザー。

- 署名済み URL を作成するために必要な IAM 権限については、Amazon Simple Storage Service ユーザーガイドの「オブジェクトを他のユーザーと共有する」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

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

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

   コールが成功すると、コマンドラインに次のような応答が表示されます。

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

2. AWS CloudFormation前のステップで取得したユーザー情報をテンプレートに追加します。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. AWS CloudFormationtemplate.yamlテンプレートをという名前のファイルに保存します。
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

1. AWS CloudShell でエディタを使用し、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 Management Console 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. コマンドラインから以下の実行で IAM ロールの Amazon リソースネーム (ARN) を取得します AWS CloudFormation。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
```

スクリプトで返された URL をクリックすることで共同作業者は AWS Management Console 内で AWS CloudShell に移動できます。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。

チュートリアル:内部に Docker AWS CloudShell 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、Docker AWS CloudShell コンテナを定義してビルドし、Amazon ECR リポジトリにプッシュする方法を示します。

前提条件

- Amazon ECR リポジトリを作成してプッシュするために必要な権限が必要です。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 でイメージをプッシュするために必要な権限の詳細については、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 権限」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の概要を説明します。

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. 作成したフォルダーに移動します。

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

3. 空の Docker ファイルを作成します。

```
touch Dockerfile
```

4. たとえば、テキストエディターを使用してファイルを開きnano Dockerfile、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 をビルドする準備ができました。を実行してコンテナをビルドします。docker buildfuture コマンドで使用できるように、 easy-to-type コンテナに名前をタグ付けします。

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

末尾には必ずピリオド (.) を入れてください。

6. これで、コンテナをテストして、AWS CloudShellで正しく実行されていることを確認できます。

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

7. 機能する Docker コンテナができたので、それを 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。

以下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}
```

8. Amazon ECR リポジトリを作成したら、Docker コンテナをそのリポジトリにプッシュできます。

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}
```

9. イメージにターゲット Amazon ECR リポジトリをタグ付けし、そのリポジトリにプッシュします。

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

このチュートリアルを完了しようとしてエラーが発生したり、問題が発生したりした場合は、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できました。AWS CloudShell このチュートリアルで作成したファイルを環境から削除するには、以下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}
```

チュートリアル:を使用して Lambda 関数をデプロイする AWS CDK

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

前提条件

- で使用できるようにアカウントをブートストラップします。AWS CDKでのブートストラップについてはAWS CDK、『v2 開発者ガイド』の「ブートストラップ」を参照してください。AWS CDKアカウントをまだブートストラップしていない場合は、を実行できます。`cdk bootstrap` CloudShell

- アカウントにリソースをデプロイするための適切な権限があることを確認してください。管理者権限が推奨されます。

チュートリアルの手順

次のチュートリアルでは、を使用して Docker コンテナベースの Lambda 関数をデプロイする方法の概要を説明します。AWS CDK

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. 作成したフォルダーに移動します。

```
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
```

4. AWS CDK作成したフォルダーにスケルトン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. たとえば、テキストエディターを使用してファイルを開きnano cdk.json、次の内容をそのファイルに貼り付けます。

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

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

```
// 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. を開きlib/Dockerfile、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. ファイルを開き、次の内容を貼り付けます。lib/hello.js

```
// 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 CDKCLI を使用してプロジェクトを統合し、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
```

10. Lambda 関数を呼び出して、確認と検証を行います。

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

これで、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できました。AWS CDK詳細については、『v2 AWS CDK 開発者ガイド』を参照してください。AWS CDKこのチュートリアルを完了しようとしてエラーや問題が発生した場合は、本ガイドの「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、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できました。AWS CDKAWS CDKプロジェクト内で以下のコマンドを実行して、関連するリソースを削除します。削除の確認を求めるプロンプトが表示されます。

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

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

AWS CloudShell の使用

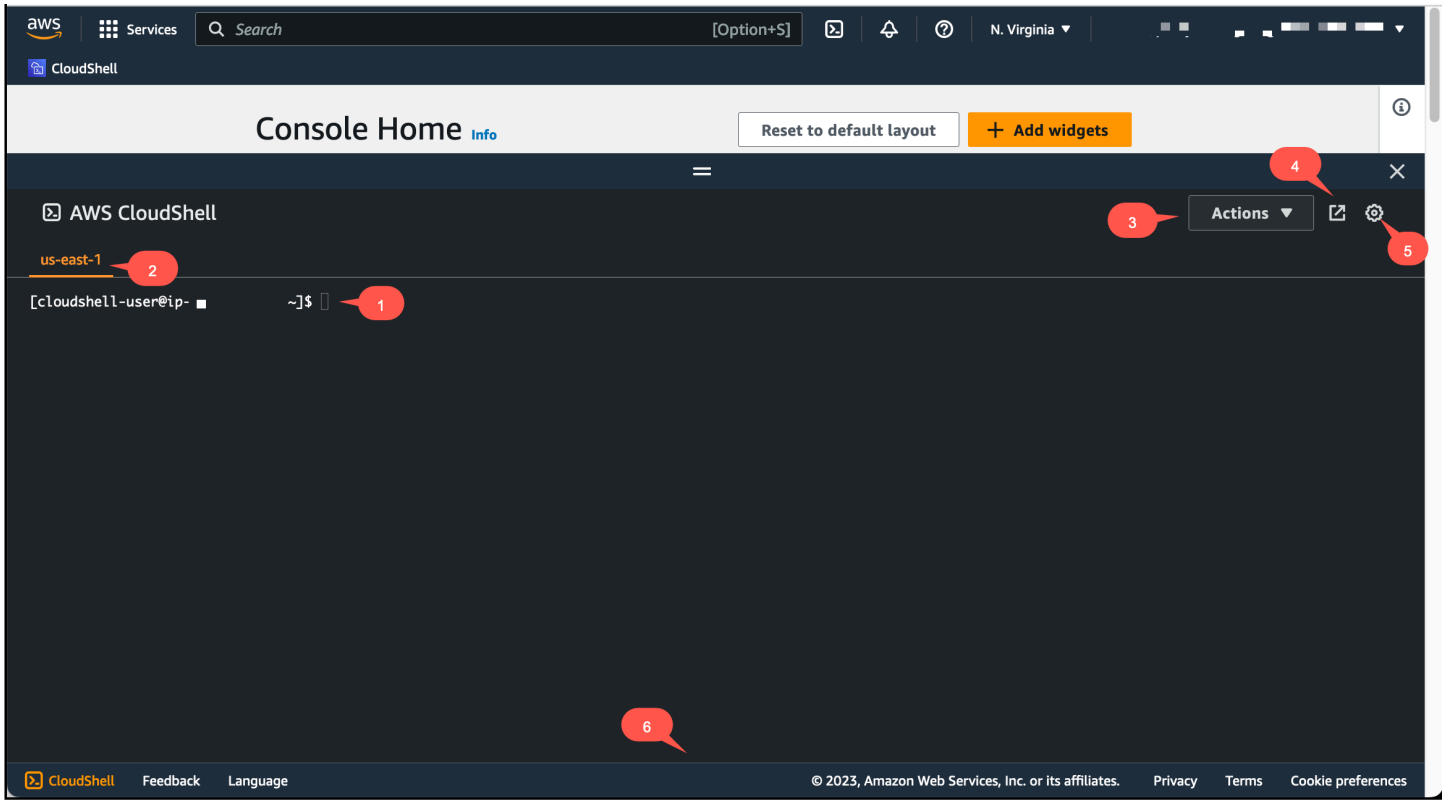
このセクションでは、AWS CloudShellサポートされているアプリケーションを操作して特定のアクションを実行する方法について説明します。

トピック

AWS CloudShell インターフェイスのナビゲーション

CloudShell AWS Management Consoleインターフェイス機能はとから操作できますConsole Toolbar。

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



1. 任意のシェルを使用して、コマンドを実行するための AWS CloudShell コマンドラインインターフェース。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. ターミナルタブは、現在 AWS CloudShell が実行されている AWS リージョン を使用しています。
3. [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**
>
> CloudShell を起動すると、ダウンロードオプションは使用できませんConsole Toolbar。

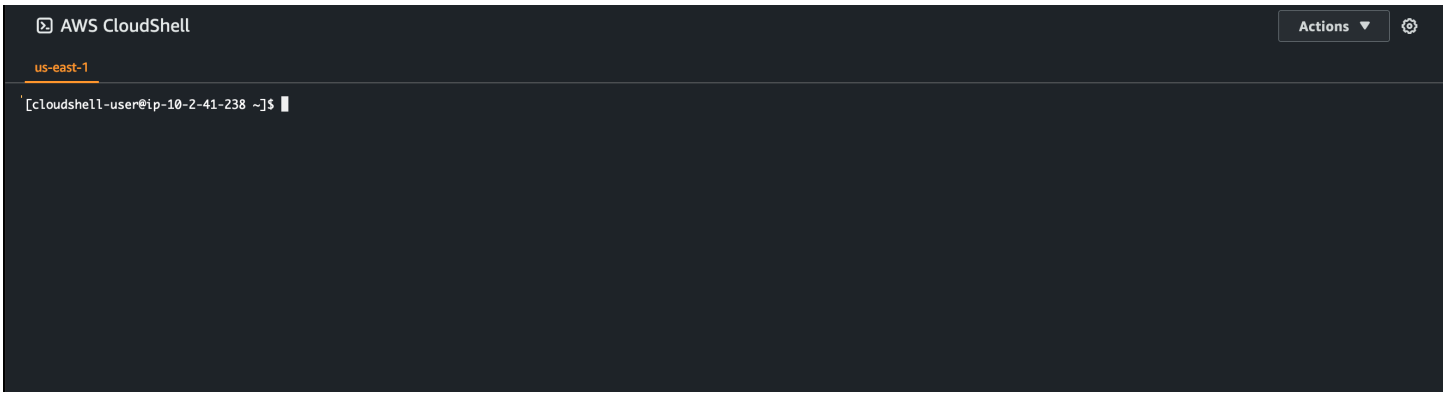
4. [新しいブラウザで開く] タブでは、 CloudShell セッションに全画面でアクセスできます。
5. シェル環境のカスタマイズに使用できる [Preferences (設定)] オプション。
6. 下部のバーには、以下のオプションがあります。
   - CloudShell CloudShellアイコンから起動します。
   - [フィードバック] アイコンでフィードバックを送信します。送信するフィードバックの種類を選択し、コメントを追加して、[送信] を選択します。
     - フィードバックを送信するには CloudShell、以下のいずれかのオプションを選択します。
    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

     > **Note**
     >
     > CloudShell を起動しても、フィードバックオプションは使用できませんConsole Toolbar。

   - 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。

AWS リージョン での作業

実行を行っている現在の AWS リージョン は、コマンドラインインターフェースの上部に表示されています。



リージョンセレクタから 特定のリージョンを選択することで、使用する AWS リージョン 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 AWS リージョン で最大 1 GB の永続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、「永続的ストレージ」を参照してください。

AWS CLI のデフォルト AWS リージョン を指定する

環境変数を使用すると、AWS CLI を使用して AWS のサービス にアクセスするうえで必要な設定オプションおよび認証情報を指定できます。シェルセッションのデフォルト AWS リージョン を指定する環境変数は、AWS Management Console で特定のリージョンから AWS CloudShell を起動したとき、またはリージョンセレクタでオプションを選択したときに設定されます。

環境変数は、aws configure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。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。その代わりに、 AWS CLI コマンドで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すると、環境変数は現在のセッションの間だけ値を保存します。または、 PowerShell 変数をプロファイルに追加して、future PowerShell すべてのセッションで変数を設定できます。環境変数の保存について詳しくは、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 の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コマンドは、コマンドラインオプション --region を使用してデフォルトでのリージョンを上書きできます。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して、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についての詳細は、「 AWS CloudShell の開始方法」を参照してください。

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: 各 AWS リージョン に 1 GB の永続的ストレージが利用できます。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。

Docker の使用

AWS CloudShellインストールや設定なしで Docker を完全にサポートします。Docker コンテナは内部で定義、ビルド、実行できます。AWS CloudShellDocker コンテナに基づく Lambda 関数などの Docker ベースのリソースは、AWS CDKツールキットを介してデプロイできます。また、Docker CLI を使用して 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ともできます。これら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:を使用して Lambda 関数をデプロイする AWS CDK
- チュートリアル:内部に Docker AWS CloudShell コンテナを構築して Amazon ECR リポジトリにプッシュする

Docker を以下と併用することには、特定の制限と制限があります。AWS CloudShell

- Docker の環境内のスペースは限られています。個別のイメージが大きかったり、既存の Docker イメージが多すぎたりすると、問題が発生して追加のイメージをプル、ビルド、または実行できなくなる可能性があります。Docker について詳しくは、『Docker ドキュメンテーションガイド』を参照してください。
- Docker は特定の地域でのみサポートされています。Docker でどのリージョンがサポートされているかについては、「Docker リージョン」を参照してください。
- で Docker を使用しているとき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」セクションを参照して、問題を解決する方法を確認してください。

のアクセシビリティ機能の使用AWS CloudShell

このトピックでは、以下のユーザー補助機能の使用方法について説明します。CloudShell。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間を移動できます。また、外観をカスタマイズすることもできます。CloudShell(フォントサイズやインターフェーステーマなど)。

でのキーボードナビゲーションCloudShell

ページ上のフォーカス可能な要素間を移動するには、を押します。Tab。

CloudShell端末アクセシビリティ機能

を使用できます。Tabキーは以下のモードで使えます。

- ターミナルモード (デフォルト)— このモードでは、端末は以下をキャプチャしますTabキー入力。端末にフォーカスが移ったら、を押します。Tab端末の機能のみにアクセスするには。
- ナビゲーションモード— このモードでは、端末はあなたをキャプチャしませんTabキー入力。を押します。Tabページ上のフォーカス可能な要素間を移動します。

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、を押しますCtrl+M。元の状態に戻すと、タブ:ナビゲーションヘッダーに表示され、次のものを使用できます。Tabキーを押してページ内を移動します。

ターミナルモードに戻るには、を押します。Ctrl+M。または、選択してくださいX[隣]タブ:ナビゲーション。

> **Note**
>
> 現在、CloudShell端末のアクセシビリティ機能はモバイルデバイスでは使用できません。

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

の外観はカスタマイズできます。CloudShell好みのビジュアルに合わせることができます。

- フォントサイズ— 以下から選択最小、小さい、ミディアム、ラージ、および最大ターミナルのフォントサイズ。フォントサイズの変更に関する詳細は、を参照してください。the section called “フォントサイズを変更する”。
- テーマ— 次の中から選択ライトそしてダークインターフェーステーマ。インターフェーステーマの変更に関する詳細は、を参照してください。the section called “インターフェイステーマの変更”。

AWS での AWS CloudShell サービスの使用

AWS CloudShell の主な利点は、それを使用してコマンドラインインターフェイスから AWS の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。AWS CloudShell を起動すると、次の AWS コマンドラインツールが既にインストールされた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

そして、既に AWS にサイインしているので、サービスを使用する前に認証情報をローカルに設定する必要はありません。AWS Management Console へのサイインに使用した認証情報は、AWS CloudShell に転送されます。

AWS CLI で使用されるデフォルト AWS リージョンを変更したい場合、`AWS_REGION` 環境変数に割り当てられている値を変更します。(詳細については、「AWS CLI のデフォルト AWS リージョン を指定する」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で選択した AWS サービスとやり取りを開始する方法を示します。

選択した AWS サービス用の AWS CLI コマンドラインの例

次の例は、AWS CLI バージョン 2 から使用できるコマンドで処理できる多数の AWS のサービスの一部のみを表しています。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
- AWS Cloud9
- Amazon EC2
- S3 グレイシャー

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拡張性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、AWS アカウント内で `MusicCollection` という NoSQL 形式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。

AWS Cloud9

AWS Cloud9 は、クラウドベースの統合開発環境 (IDE) であり、これを使用するとブラウザのウィンドウでコードを書き、実行し、デバッグできます。環境には、コードエディタ、デバッガーおよびターミナルが付属しています。

次の `create-environment-ec2` コマンドは、指定した設定の AWS Cloud9 EC2 開発環境を作成します。Amazon EC2 インスタンスを起動し、インスタンスから環境に接続します。

```
aws cloud9 create-environment-ec2 --name my-demo-env --description "My demonstration
 development environment." --instance-type t2.micro --subnet-id subnet-1fab8aEX --
automatic-stop-time-minutes 60 --owner-arn arn:aws:iam::123456789012:user/MyDemoUser
```

詳細については、「AWS Cloud9 コマンドラインリファレンス」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。

S3 Glacier

S3 Glacier および S3 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 S3 Glacier の使用」を参照してください。

AWS Elastic Beanstalk CLI

AWS Elastic Beanstalk CLI はコマンドラインインターフェイスを提供し、これはローカルリポジトリから環境の作成、更新、モニタリングを簡素化するために作られたものです。この文脈では、環境とは、アプリケーションバージョンを実行している AWS リソースのコレクションです。

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 テンプレートおよび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。Lambda 関数をローカルで呼び出すサーバーレスアプリケーションのデプロイパッケージを作成するサーバーレスアプリケーションを AWS Cloud にデプロイするなどがあります。

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 python3.7 --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。

AWS CloudShell 環境のカスタマイズ

AWS CloudShell 環境について以下の側面をカストマイズできます。

- 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: ライトテーマとダークテーマを切り替えます。
- [セーフペースト]: 貼り付け前に複数行のテキストを確認する必要がある機能をオンまたはオフに切り替えます。
- Tmux からセッションへの復元: tmux を使用すると、セッションが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タートアップシェルスクリプトを変更すれば、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イ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、選択したペインの任意の場所をクリックして、作業したいタブを選択できます。タブを選択するとタブを閉じることができます。x地域名の横にある記号。

- [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。
  - 行に分割: 現在アクティブなタブの下にある行に新しいタブを追加します。
  - 列に分割: 現在アクティブな列の横にある列に新しいタブを追加します。

  各タブを完全に表示するのに十分なスペースがない場合は、スクロールしてタブ全体を表示してください。ペインを区切る分割バーを選択し、ポインターを使用してドラッグしてペインのサイズを拡大または縮小することもできます。

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イスに表示されるテキストのサイズを拡大または縮小します。

1. を変更するにはAWS CloudShell端末設定は、[設定]、環境設定。
2. テキストサイズを選択してください。選択肢は以下のとおりです。最小、小さい、ミディアム、ラージ、および最大。

インターフェイステーマの変更

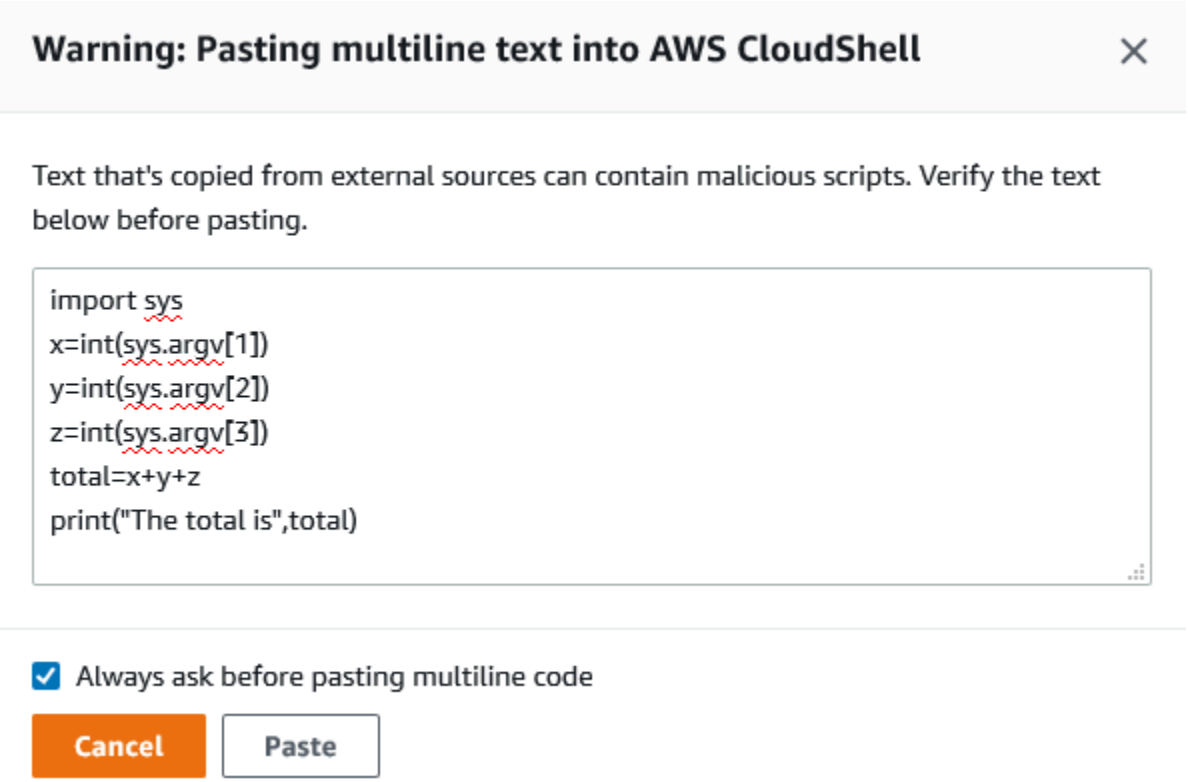
コマンドラインインターフェイスのライトテーマとダークテーマを切り替えます。

1. 変更するにはAWS CloudShellテーマは、[設定]、環境設定。
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

セーフペーストは、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ように求めるセキュリティ機能です。サードパーティ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引き起こす隠しコードが含まれている可能性があります。

Safe Paste ダイアログには、クリップボードにコピーしたテキスト全体が表示されます。セキュリティ上のリスクがないことが確認できたら、を選択してください。ペースト。



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。この機能のオンとオフは、を選択して切り替えることができます。[設定]、セーフペーストを有効にするそしてセーフペーストを無効にする。

使用するtmuxセッションリストアへ

AWS CloudShelltmux を使用して、1 つまたは複数のブラウザタブのセッションを復元します。ブラウザのタブを更新すると、セッションが非アクティブになるまでセッションが再開されます。詳細については、を参照してください。セッション復元。

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。 AWS のお客様は、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から利点を得られます。セキュリティは、 AWS とユーザー間で共有される責任です。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 — クラウドで提供され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 AWS 、安全に使用できるサービスを提供します。当社のセキュリティ責任は における最優先事項であり 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人によって定期的にテストおよび検証されています。

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、およびお客様のデータの機密性、組織の要件、適用可能な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、サポートしている特定の 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 、 AWS コンプライアンスプログラム によるコンプライアンスの取り組みの対象となる の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 AWS CloudShell を達成するために 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 の Identity and Access Management CloudShell
- 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
- での設定と脆弱性の分析 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、 でのデータ保護に適用されます AWS CloudShell。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を担います AWS クラウド。このインフラストラクチャでホストされているコンテンツに対する管理を維持する責任はユーザーにあります。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、「データプライバシーのよくある質問」を参照してください。欧州でのデータ保護の詳細については、「AWS セキュリティブログ」に投稿された「AWS 責任共有モデルおよび GDPR」のブログ記事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 Identity Center 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。こうすると、それぞれのジョブを遂行するために必要なアクセス許可のみを各ユーザーに付与できます。また、以下の方法でデータを保護することをお勧めします。

- 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、できれば TLS 1.3 が必要です。
- 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。
- AWS 暗号化ソリューションを、 内のすべてのデフォルトのセキュリティコントロールとともに使用します AWS のサービス。
- Amazon Macie などの高度なマネージド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 140-2 検証済みの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) 140-2」を参照してください。

お客様の E メールアドレスなどの機密情報やセンシティブ情報は、タグや名前フィールドなどの自由形式のフィールドに配置し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 で AWS CLIまたは他の AWS のサービス を使用する場合も同様です。 AWS SDKs 名前に使用する自由記述のテキストフィールドやタグに入力したデータは、課金や診断ログに使用される場合があります。外部サーバーへの URL を提供する場合は、そのサーバーへのリクエストを検証するための認証情報を URL に含めないように強くお勧めします。

データ暗号化

データ暗号化とは、保管時 (に保存中 AWS CloudShell) および転送時 (AWS CloudShell とサービスエンドポイント間を移動中) のデータを保護することです。

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、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、カスタマーマスターキー (CMKs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。これは、 AWS CloudShell 環境に保存されているカスタマーデータの暗号化に使用される暗号化キーです。 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 の Identity and Access Management CloudShell

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) し、誰に CloudShell リソースの使用を承認する (アクセス許可を付与する) かを制御します。IAM は、追加料金なしで AWS のサービス 使用できる です。

トピック

- 対象者
- 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 と IAM の CloudShell 連携方法
- AWS のアイデンティティベースのポリシーの例 CloudShell

- AWS CloudShell 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

対象者

AWS Identity and Access Management (IAM) の使用方法は、 で行う作業によって異なります CloudShell。

サービスユーザー – CloudShell サービスを使用してジョブを実行する場合は、管理者から必要な認証情報とアクセス許可が与えられます。さらに多くの CloudShell 機能を使用して作業を行う場合は、追加のアクセス許可が必要になることがあります。アクセスの管理方法を理解しておくと、管理者に適切な許可をリクエストするうえで役立ちます。 CloudShell 機能にアクセスできない場合は、「AWS CloudShell 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」を参照してください。

サービス管理者 – 社内の CloudShell リソースを担当している場合は、通常、 へのフルアクセスがあります CloudShell。サービスのユーザーがどの CloudShell 機能やリソースにアクセスするかを決めるのは管理者の仕事です。その後、IAM 管理者にリクエストを送信して、サービスユーザーの権限を変更する必要があります。このページの情報を点検して、IAM の基本概念を理解してください。お客様の会社で で IAM を利用する方法の詳細については CloudShell、「」を参照してくださいAWS と IAM の CloudShell 連携方法。

IAM 管理者 - 管理者は、 CloudShell へのアクセスを管理するポリシーの書き込み方法の詳細について確認する場合があります。IAM で使用できる CloudShell アイデンティティベースのポリシーの例を表示するには、「」を参照してくださいAWS のアイデンティティベースのポリシーの例 CloudShell。

アイデンティティを使用した認証

認証は、ID 認証情報 AWS を使用して にサインインする方法です。として、IAM ユーザーとして 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によって認証 (にサインイン AWS) される必要があります。

ID ソース (AWS IAM Identity Center) から提供された認証情報を使用して、フェデレーティッド ID AWS として にサインインできます。IAM Identity Center) ユーザー、会社のシングルサインオン認証、Google または Facebook の認証情報は、フェデレーティッド ID の例です。フェデレーションアイデンティティとしてサインインする場合、IAM ロールを使用して、前もって管理者により ID フェデレーションが設定されています。フェデレーションを使用して にアクセスすると、間接的 AWS にロールを引き受けます。

ユーザーのタイプに応じて、 AWS Management Console または AWS アクセスポータルにサインインできます。へのサインインの詳細については AWS、「 AWS サインイン ユーザーガイド」の「 にサインインする方法 AWS アカウント」を参照してください。

AWS プログラムで にアクセスする場合、 は Software Development Kit (SDK) とコマンドラインインターフェイス (CLI) AWS を提供し、 認証情報を使用してリクエストに暗号で署名します。 AWS ツールを使用しない場合は、リクエストを自分で署名する必要があります。推奨される方法を使用してリクエストを自分で署名する方法の詳細については、「IAM ユーザーガイド」のAWS 「 API リクエストの署名」を参照してください。

使用する認証方法を問わず、セキュリティ情報の提供を追加でリクエストされる場合もあります。例えば、 では、多要素認証 (MFA) を使用してアカウントのセキュリティを高めることを AWS 推奨しています。詳細については、「AWS IAM Identity Center ユーザーガイド」の「多要素認証 (MFA)」および「IAM ユーザーガイド」の「AWSでの多要素認証 (MFA) の使用」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、アカウント内のすべての 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 1 つのサインインアイデンティティから始めます。このアイデンティティは AWS アカウント ルートユーザーと呼ばれ、アカウントの作成に使用した E メールアドレスとパスワードでサインインすることでアクセスできます。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。ルートユーザーの認証情報を保護し、それらを使用してルートユーザーのみが実行できるタスクを実行してください。ルートユーザーとしてサインインする必要があるタスクの完全なリストについては、「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。

フェデレーション ID

ベストプラクティスとして、管理者アクセスを必要とするユーザーを含む人間のユーザーに、一時的な AWS のサービス 認証情報を使用して にアクセスする ID プロバイダーとのフェデレーションの使用を要求します。

フェデレーティッド ID とは、エンタープライズユーザーディレクトリ、ウェブ ID プロバイダー、AWS Directory Service、、Identity Center ディレクトリのユーザー、または ID ソースから提供された認証情報 AWS のサービス を使用して にアクセスするすべてのユーザーです。フェデレーティッド ID が にアクセスすると AWS アカウント、ロールを引き受け、ロールは一時的な認証情報を提供します。

アクセスを一元管理する場合は、 AWS IAM Identity Centerを使用することをお勧めします。IAM Identity Center でユーザーとグループを作成することも、すべての とアプリケーションで使用する

独自の ID ソースのユーザー AWS アカウント とグループのセットに接続して同期することもできます。IAM アイデンティティセンターの詳細については、[AWS IAM Identity Center 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、単一のユーザーまたはアプリケーションに対して特定のアクセス許可 AWS アカウント を持つ 内のアイデンティティです。可能であれば、パスワードやアクセスキーなどの長期的な認証情報を保有する IAM ユーザーを作成する代わりに、一時的な認証情報を使用することをお勧めします。ただし、IAM ユーザーでの長期的な認証情報が必要な特定のユースケースがある場合は、アクセスキーをローテーション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「長期的な認証情報を必要とするユースケースのためにアクセスキーを定期的にローテーションする」を参照してください。

IAM グループは、IAM ユーザーの集団を指定するアイデンティティです。グループとしてサインインすることはできません。グループを使用して、複数のユーザーに対して一度に権限を指定できます。多数のユーザーグループがある場合、グループを使用することで権限の管理が容易になります。例えば、IAMAdmins という名前のグループを設定して、そのグループに IAM リソースを管理する権限を与えることができます。

ユーザーは、ロールとは異なります。ユーザーは 1 人の人または 1 つのアプリケーションに一意に関連付けられますが、ロールはそれを必要とする任意の人が引き受けるようになっています。ユーザーには永続的な長期の認証情報がありますが、ロールでは一時的な認証情報が提供されます。詳細については、「IAM ユーザーガイド」の「IAM ユーザー (ロールではなく) の作成が適している場合」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 AWS アカウント を持つ 内のアイデンティティです。これは IAM ユーザーに似ていますが、特定のユーザーには関連付けられていません。ロールを切り替える AWS Management Console ことで、 で IAM ロールを一時的に引き受けることができます。ロールを引き受けるには、 または AWS API オペレーションを AWS CLI 呼び出すか、カスタム URL を使用します。ロールを使用する方法の詳細については、「IAM ユーザーガイド」の「IAM ロールの使用」を参照してください。

一時的な認証情報を持った IAM ロールは、以下の状況で役立ちます。

- フェデレーションユーザーアクセス – フェデレーションアイデンティティに権限を割り当てるには、ロールを作成してそのロールの権限を定義します。フェデレーションアイデンティティが認証されると、そのアイデンティティはロールに関連付けられ、ロールで定義されている権限が付与さ

れます。フェデレーションの詳細については、「IAM ユーザーガイド」の「サードパーティーアイデンティティプロバイダー向けロールの作成」を参照してください。IAM アイデンティティセンターを使用する場合、権限セットを設定します。アイデンティティが認証後にアクセスできるものを制御するため、IAM Identity Center は、権限セットを IAM のロールに関連付けます。権限セットの詳細については、「AWS IAM Identity Center ユーザーガイド」の「権限セット」を参照してください。

- 一時的な IAM ユーザー権限 - IAM ユーザーまたはロールは、特定のタスクに対して複数の異なる権限を一時的に IAM ロールで引き受けることができます。
- クロスアカウントアクセス - IAM ロールを使用して、自分のアカウントのリソースにアクセスすることを、別のアカウントの人物 (信頼済みプリンシパル) に許可できます。クロスアカウントアクセス権を付与する主な方法は、ロールを使用することです。ただし、一部の では AWS のサービス、(ロールをプロキシとして使用する代わりに) リソースにポリシーを直接アタッチできます。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違いについては、「IAM ユーザーガイド」の「IAM ロールとリソースベースのポリシーとの相違点」を参照してください。
- クロスサービスアクセス — 一部の では、他の の機能 AWS のサービス を使用します AWS のサービス。例えば、あるサービスで呼び出しを行うと、通常そのサービスによって Amazon EC2 でアプリケーションが実行されたり、Amazon S3 にオブジェクトが保存されたりします。サービスでは、呼び出し元プリンシパルの権限、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使用してこれを行う場合があります。
  - 転送アクセスセッション (FAS) – IAM ユーザーまたはロールを使用して でアクションを実行する場合 AWS、プリンシパルと見なされます。一部のサービスを使用する際に、アクションを実行することで、別のサービスの別のアクションがトリガーされることがあります。FAS は、を呼び出すプリンシパルのアクセス許可 AWS のサービス を使用し AWS のサービス、ダウンストリームサービスにリクエストを行うリクエストと組み合わせて使用します。FAS リクエストは、他の AWS のサービス またはリソースとのやり取りを完了する必要があるリクエストをサービスが受信した場合にのみ行われます。この場合、両方のアクションを実行するためのアクセス許可が必要です。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。
  - サービスロール - 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 のサービスにアクセス許可を委任するロールの作成」を参照してください。
  - サービスにリンクされたロール – 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権限を表示できますが、編集することはできません。

- Amazon EC2 で実行されているアプリケーション – IAM ロールを使用して、EC2 インスタンスで実行され、 AWS CLI または AWS API リクエストを作成しているアプリケーションの一時的な認証情報を管理できます。これは、EC2 インスタンス内でのアクセスキーの保存に推奨されます。EC2 インスタンスに AWS ロールを割り当て、そのすべてのアプリケーションで使用できるようにするには、インスタンスにアタッチされたインスタンスプロファイルを作成します。インスタンスプロファイルにはロールが含まれ、EC2 インスタンスで実行されるプログラムは一時的な認証情報を取得できます。詳細については、「IAM ユーザーガイド」の「Amazon EC2 インスタンスで実行されるアプリケーションに IAM ロールを使用して権限を付与する」を参照してください。

IAM ロールと IAM ユーザーのどちらを使用するかについては、「IAM ユーザーガイド」の「IAM ロールの作成が適している場合 (ユーザーではなく)」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、 AWS ID またはリソースにアタッチします。ポリシーは のオブジェクト AWS であり、アイデンティティまたはリソースに関連付けられると、これらのアクセス許可を定義します。 は、プリンシパル (ユーザー、ルートユーザー、またはロールセッション) がリクエストを行うと、これらのポリシー AWS を評価します。ポリシーでの権限により、リクエストが許可されるか拒否されるかが決まります。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構造と内容の詳細については、「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 AWS JSON 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。

デフォルトでは、ユーザーやロールに権限はありません。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。その後、管理者はロールに IAM ポリシーを追加し、ユーザーはロールを引き継ぐことができます。

IAM ポリシーは、オペレーションの実行方法を問わず、アクションの権限を定義します。例えば、`iam:GetRole` アクションを許可するポリシーがあるとします。このポリシーを持つユーザーは、 AWS Management Console、、 AWS CLIまたは AWS API からロール情報を取得できます。

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ポリシーは、IAM ユーザー、ユーザーのグループ、ロールなど、アイデンティティにアタッチできる JSON 権限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、および条件を制御します。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 ポリシーの作成」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、さらに インラインポリシー または マネージドポリシー に分類できます。インラインポリシーは、単一のユーザー、グループ、またはロールに直接埋め込まれています。管理ポリシーは、 の複数のユーザー、グループ、ロールにアタッチできるスタンドアロンポリシーです AWS アカウント。管理ポリシーには、 AWS 管理ポリシーとカスタマー管理ポリシーが含まれます。管理ポリシーまたは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比較」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。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制御できます。ポリシーが添付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は、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。

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 IAM の AWS マネージドポリシーを使用することはできません。

アクセスコントロールリスト (ACL)

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権限を持つかをコントロールします。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。

Amazon S3、、および Amazon VPC は AWS WAF、ACLs。ACL の詳細については、「Amazon Simple Storage Service デベロッパーガイド」の「アクセスコントロールリスト (ACL) の概要」を参照してください。

その他のポリシータイプ

AWS は、追加の一般的でないポリシータイプをサポートします。これらのポリシータイプでは、より一般的なポリシータイプで付与された最大の権限を設定できます。

- 権限の境界 - 権限の境界は、アイデンティティベースのポリシーによって IAM エンティティ (IAM ユーザーまたはロール) に付与できる権限の上限を設定する高度な機能です。エンティティに権限の境界を設定できます。結果として得られる権限は、エンティティのアイデンティティベースポリシーとその権限の境界の共通部分になります。`Principal` フィールドでユーザーまたはロールを指定するリソースベースのポリシーでは、権限の境界は制限されません。これらのポリシーのいずれかを明示的に拒否した場合、許可は無効になります。権限の境界の詳細については、「IAM ユーザーガイド」の「IAM エンティティの権限の境界」を参照してください。
- サービスコントロールポリシー (SCPs) – SCPs は、 の組織または組織単位 (OU) に最大アクセス許可を指定する JSON ポリシーです AWS Organizations。 AWS Organizations は、 AWS アカウント ビジネスが所有する複数の をグループ化して一元管理するためのサービスです。組織内のすべての機能を有効にすると、サービスコントロールポリシー (SCP) を一部またはすべてのアカウントに適用できます。SCP は、各 を含むメンバーアカウントのエンティティのアクセス許可を制限します AWS アカウントのルートユーザー。組織と SCP の詳細については、「AWS Organizations ユーザーガイド」の「SCP の仕組み」を参照してください。
- セッションポリシー - セッションポリシーは、ロールまたはフェデレーションユーザーの一時的なセッションをプログラムで作成する際にパラメータとして渡す高度なポリシーです。結果としてセッションの権限される範囲は、ユーザーまたはロールのアイデンティティベースポリシーとセッションポリシーの共通部分になります。また、リソースベースのポリシーから権限が派生する場合もあります。これらのポリシーのいずれかを明示的に拒否した場合、許可は無効になります。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権限を理解するのがさらに難しくなります。複数のポリシータイプが関連する場合に が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「ポリシーの評価ロジック」を参照してください。

AWS と IAM の CloudShell 連携方法

IAM を使用して へのアクセスを管理する前に CloudShell、 で使用できる IAM の機能について学びます CloudShell。

AWS で使用できる IAM の機能 CloudShell

IAM 機能	CloudShell 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	いいえ
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	はい
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を把握するには、「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。

のアイデンティティベースのポリシー CloudShell

アイデンティティベースポリシーをサポートする	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 権限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、および条件を制御します。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 ポリシーの作成」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。プリンシパルは、それが添付されているユーザーまたはロールに適用されるため、アイデンティティベースのポリシーでは指定できません。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。

のアイデンティティベースのポリシーの例 CloudShell

CloudShell アイデンティティベースのポリシーの例を表示するには、「」を参照してくださいAWS のアイデンティティベースのポリシーの例 CloudShell。

内のリソースベースのポリシー CloudShell

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制御できます。ポリシーが添付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は、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、アカウント全体、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。リソースベースのポリシーにクロスアカウントのプリンシパルを追加しても、信頼関係は半分しか確立されない点に注意してください。プリンシパルとリソースが異なる にある場合 AWS アカウント、信頼されたアカウントの IAM 管理者は、リソースへのアクセス許可をプリンシパルエンティティ (ユーザーまたはロール) に付与する必要もあります。IAM 管理者は、アイデンティティベースのポリシーをエンティティにアタッチすることで権限を付与します。ただし、リソースベースのポリシーで、同じアカウントのプリンシパルへのアクセス権が付与されている場合は、アイデンティティベースのポリシーを追加する必要はありません。詳細については、「IAM ユーザーガイド」の「IAM ロールとリソースベースのポリシーとの相違点」を参照してください。

のポリシーアクション CloudShell

ポリシーアクションに対するサポート	はい

管理者は AWS JSON 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んなリソースにどんな条件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。ポリシーアクションの名前は通常、関連する AWS API オペレーションと同じです。一致する API オペレーションのない権限のみのアクションなど、いくつかの例外があります。また、ポリシーに複数アクションが必要なオペレーションもあります。これらの追加アクションは、依存アクションと呼ばれます。

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、「サービス認証リファレンス」の「AWS で定義されるアクション CloudShell」を参照してください。一部のアクションには、複数の API が含まれる場合があります。

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 CloudShell 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 アイデンティティベースのポリシーの例を表示するには、「」を参照してくださいAWS のアイデンティティベースのポリシーの例 CloudShell。

のポリシーリソース CloudShell

ポリシーリソースに対するサポート	はい

管理者は AWS JSON 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 ポリシーの要素は、オブジェクトあるいはアクションが適用されるオブジェクトを指定します。ステートメントには、Resource または NotResource 要素を含める必要があります。ベストプラクティスとしては、Amazon リソースネーム (ARN) を使用してリソースを指定します。これは、リソースレベルの権限と呼ばれる特定のリソースタイプをサポートするアクションに対して実行できます。

オペレーションのリスト化など、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 リソースタイプとその ARNs」の「AWS で定義されるリソース CloudShell」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 で定義されるアクション CloudShell」を参照してください。

CloudShell アイデンティティベースのポリシーの例を表示するには、「」を参照してくださいAWS のアイデンティティベースのポリシーの例 CloudShell。

のポリシー条件キー CloudShell

サービス固有のポリシー条件キーのサポート	はい

管理者は AWS JSON 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んなリソースにどんな条件でアクションを実行できるかということです。

Condition 要素 (または Condition ブロック) を使用すると、ステートメントが有効な条件を指定できます。Condition 要素はオプションです。equal や less than などの条件演算子を使用して条件式を作成することによって、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。

1 つのステートメントに複数の Condition 要素を指定する場合、または 1 つの Condition 要素に複数のキーを指定する場合、 AWS は AND 論理演算子を使用してそれらを評価します。1 つの条件

キーに複数の値を指定すると、 は論理OR演算を使用して条件 AWS を評価します。ステートメントの権限が付与される前にすべての条件が満たされる必要があります。

条件を指定する際にプレースホルダー変数も使用できます。例えば IAM ユーザーに、IAM ユーザー名がタグ付けされている場合のみリソースにアクセスできる権限を付与することができます。詳細については、「IAM ユーザーガイド」の「IAM ポリシーの要素: 変数およびタグ」を参照してください。

AWS は、 グローバル条件キーとサービス固有の条件キーをサポートします。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 条件キーのリストを確認するには、「サービス認証リファレンス」の「AWS の条件キー CloudShell」を参照してください。条件キーを使用できるアクションとリソースについては、「AWS で定義されるアクション CloudShell」を参照してください。

CloudShell アイデンティティベースのポリシーの例を表示するには、「」を参照してくださいAWS のアイデンティティベースのポリシーの例 CloudShell。

ACLs CloudShell

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権限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。

での ABAC CloudShell

ABAC (ポリシー内のタグ) のサポート	いいえ

属性ベースのアクセス制御 (ABAC) は、属性に基づいて権限を定義する認可戦略です。では AWS、これらの属性はタグ と呼ばれます。タグは、IAM エンティティ (ユーザーまたはロール) および多くの AWS リソースにアタッチできます。エンティティとリソースのタグ付けは、ABAC の最初の手順です。次に、プリンシパルのタグがアクセスを試行するリソースのタグと一致したときにオペレーションを許可するよう、ABAC ポリシーを設計します。

ABAC は、急成長する環境やポリシー管理が煩雑になる状況で役立ちます。

タグに基づいてアクセスを制御するには、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 Yes です。サービスが一部のリソースタイプに対してのみ 3 つの条件キーすべてをサポートする場合、値は Partial です。

ABAC の詳細については、「IAM ユーザーガイド」の [ABAC とは?」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制御 (ABAC) を使用する」を参照してください。

での一時的な認証情報の使用 CloudShell

一時的な認証情報のサポート	はい

一部の AWS のサービス は、一時的な認証情報を使用してサインインすると機能しません。一時的な認証情報 AWS のサービス を使用する などの詳細については、IAM ユーザーガイドの「IAM AWS のサービス と連携する 」を参照してください。

ユーザー名とパスワード以外の AWS Management Console 方法で にサインインする場合は、一時的な認証情報を使用しています。例えば、会社の Single Sign-On (SSO) リンク AWS を使用して にアクセスすると、そのプロセスは自動的に一時的な認証情報を作成します。また、ユーザーとしてコンソールにサインインしてからロールを切り替える場合も、一時的な認証情報が自動的に作成されます。ロールの切り替えに関する詳細については、「IAM ユーザーガイド」の「ロールへの切り替え (コンソール)」を参照してください。

一時的な認証情報は、 AWS CLI または AWS API を使用して手動で作成できます。その後、これらの一時的な認証情報を使用して、長期的なアクセスキーを使用する代わりに、動的に一時的な認証情報を生成する AWS. AWS recommends にアクセスできます。詳細については、「IAM の一時的セキュリティ認証情報」を参照してください。

ロールを切り替えると、別の環境が使用されます。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。

の転送アクセスセッション CloudShell

転送アクセスセッション (FAS) をサポート	いいえ

IAM ユーザーまたはロールを使用して でアクションを実行する場合 AWS、ユーザーはプリンシパルと見なされます。一部のサービスを使用する際に、アクションを実行することで、別のサービスの別のアクションがトリガーされることがあります。FAS は、 を呼び出すプリンシパルのアクセス許可を使用し AWS のサービス、ダウンストリームサービスにリクエストを行う AWS のサービス リクエストと組み合わせて使用します。FAS リクエストは、他の AWS のサービス またはリソースとのやり取りを完了する必要があるリクエストをサービスが受信した場合にのみ行われます。この場合、両方のアクションを実行するためのアクセス許可が必要です。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	いいえ

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 のサービスにアクセス許可を委任するロールの作成」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 機能が破損する可能性があります。が指示する場合以外 CloudShell は、サービスロールを編集しないでください。

のサービスにリンクされたロール CloudShell

サービスにリンクされたロールのサポート	いいえ

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権限を表示できますが、編集することはできません。

AWS のアイデンティティベースのポリシーの例 CloudShell

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権限はありません。また、、 AWS Command Line Interface （ AWS CLI ） AWS Management Console、または AWS API を使用してタスクを実行することはできません。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。その後、管理者がロールに IAM ポリシーを追加すると、ユーザーはロールを引き受けることができます。

これらサンプルの 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 ポリシーの作成」を参照してください。

各リソースタイプの ARN の形式など CloudShell、 で定義されるアクションとリソースタイプの詳細については、「サービス認証リファレンス」の「AWS のアクション、リソース、および条件キー CloudShell」を参照してください。 ARNs

トピック

- ポリシーのベストプラクティス
- CloudShell コンソールを使用する
- 自分の許可の表示をユーザーに許可する

ポリシーのベストプラクティス

ID ベースのポリシーは、ユーザーのアカウントで誰かが CloudShell リソースを作成、アクセス、または削除できるどうかを決定します。これらのアクションを実行すると、 AWS アカウントに料金が発生する可能性があります。アイデンティティベースの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を付与するには、多くの一般的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して、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「AWS ジョブ機能の管理ポリシー」を参照してください。
- 最小特権を適用する – IAM ポリシーで権限を設定するときは、タスクの実行に必要な権限のみを付与します。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、最小特権とも呼ばれています。IAM を使用して権限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 でのポリシーと権限」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。例えば、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、 などの特定の を通じてサービスアクションを使用する場合 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 AWS CloudFormation。詳細については、「IAM ユーザーガイド」の [IAM JSON policy elements: Condition] (IAM JSON ポリシー要素：条件) を参照してください。
- IAM Access Analyzer 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 Access Analyzer 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 ポリシーの検証」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、セキュリティを強化するために MFA を有効にします。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 保護 API アクセスの設定」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 コンソールを使用する

AWS CloudShell コンソールにアクセスするには、一連の最小限のアクセス許可が必要です。これらのアクセス許可により、 の CloudShell リソースの詳細をリスト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「ユーザーへの権限の追加」を参照してください。

自分の許可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添付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を作成する方法を示します。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 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

次の情報は、 と IAM の使用に伴って発生する可能性がある一般的な問題の診断 CloudShell や修復に役立ちます。

トピック

- でアクションを実行する権限がない CloudShell
- iam を実行する権限がありません。PassRole
- 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

でアクションを実行する権限がない CloudShell

「I am not authorized to perform an action in Amazon Bedrock」というエラーが表示された場合、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。

次の例は、mateojackson という IAM ユーザーがコンソールを使用して架空の *my-example-widget* リソースに関する詳細を表示しようとしたとき、架空の awes:*GetWidget* アクセス許可がない場合に発生するエラーを示しています。

```
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。サインイン資格情報を提供した担当者が管理者です。

iam を実行する権限がありません。PassRole

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 にロールを渡すことができ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、既存のロールをそのサービスに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。

以下の例のエラーは、marymajor という 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する場合に発生します。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権限が必要です。Mary には、ロールをサービスに渡す権限がありません。

```
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 Mary 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:

- がこれらの機能 CloudShell をサポートしているかどうかを確認するには、「」を参照してくださいAWS と IAM の CloudShell 連携方法。
- 所有 AWS アカウント する 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 する別の の IAM ユーザーへのアクセスを許可する」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の 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「第三者 AWS アカウント が所有する へのアクセス権を付与する」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) へのアクセス権の提供」を参照してください。
- クロスアカウントアクセスでのロールとリソースベースのポリシーの使用の違いの詳細については、「IAM ユーザーガイド」の「IAM ロールとリソースベースのポリシーとの相違点」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

AWS Identity and Access Management (IAM) によって提供できるアクセス管理リソースを使用すると、管理者は IAM ユーザーにアクセス許可を付与できます。こうすれば、ユーザーは AWS CloudShell にアクセスし、環境の機能を使用できます。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 管理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 マネージドポリシーを IAM ID にアタッチ AWS CloudShell できます。

- AWS CloudShellFullAccess: すべての機能へのフルアクセス権のある AWS CloudShell を使用するためのアクセス許可を付与します。

この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 および 機能への CloudShell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、「 マネージドポリシーユーザーガイドAWS CloudShellFullAccess」の「」を参照してください。 AWS

> **Note**
>
> 以下の AWS マネージドポリシーを持つ IAM ID は、 を起動することもできます CloudShell。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およびリソースにアクセス許可を委任できるようにします AWS。
> - デベロッパーパワーユーザー : IAM ユーザーがアプリケーション開発タスクを実行し、AWS 対応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 で実行できるアクションを管理するには CloudShell、 CloudShellPolicy 管理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 へのアクセスを許可しても CloudShell、 へのログインに使用される CloudShell 環境認証情報は転送しないようにすることができます AWS Management Console。

> **Important**
>
> AWS CloudShell から を起動するには AWS Management Console、IAM ユーザーに次のアクションの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いずれか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、 を起動しようとすると IAM アクセス許可エラーが返されます CloudShell。

AWS CloudShell アクセス許可

名前	付与されたアクセス許可の説明	を起動するのに必要です CloudShell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、バックエンドのウェブアプリケーショ	はい

名前	付与されたアクセス許可の説明	を起動するのに必要ですCloudShellか？
	ンから現在のレイアウトを保存します。このアクセス許可では、「」で説明Resourceされているように、 の値*として のみを想定していますthe section called “の IAM ポリシーの例 CloudShell”。	
cloudshell:CreateSession	から CloudShell 環境に接続します AWS Management Console。	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イス CloudShell を使用して からファイルをダウンロードするために使用される署名付き Amazon S3 URLs を生成します。	いいえ
cloudshell:GetFileUploadUrls	CloudShell ウェブインターフェイス CloudShell を使用して からファイルをアップロードするために使用される、署名付き Amazon S3 URLs を生成します。	いいえ

名前	付与されたアクセス許可の説明	を起動するのに必要ですCloudShellか？
cloudshell:PutCredentials	へのログインに使用される認証情報 AWS Management Console をに転送します CloudShell。	いいえ
cloudshell:StartEnvironment	停止した CloudShell 環境を開始します。	はい
cloudshell:StopEnvironment	実行中の CloudShell 環境を停止します。	いいえ

の IAM ポリシーの例 CloudShell

次の例は、 にアクセスできる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 CloudShell。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。

次のポリシーは、 CloudShell とその機能へのアクセスを完全に拒否します。

```
{
    "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 にアクセスすることを許可します CloudShell が、ファイルのアップロードとダウンロード用の署名URLs の生成はブロックします。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。

```
{
```

```
    "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 ユーザーが にアクセスすることを許可します CloudShell。ただし、このポリシーは、ログインに使用した認証情報が CloudShell 環境に転送 AWS Management Console されることを防ぎます。このポリシーを持つ IAM ユーザーは、 内で認証情報を手動で設定する必要があります CloudShell。

```
{
    "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 ユーザーが AWS CloudShell 環境を作成することを許可します。

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Sid": "CloudShellUser",
        "Effect": "Allow",
        "Action": [
            "cloudshell:CreateEnvironment",
            "cloudshell:CreateSession",
            "cloudshell:GetEnvironmentStatus",
            "cloudshell:StartEnvironment"
        ],
        "Resource": "*"
     }]
}
```

にアクセスするためのアクセス許可 AWS のサービス

CloudShell は、 へのサインインに使用した IAM 認証情報を使用します AWS Management Console。

> **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 Management Console、 アクセス`cloudshell:PutCredentials`許可が必要です。

のこの事前認証機能は CloudShell 、 の使用に役立ちます AWS CLI。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。

でのログ記録とモニタリング AWS CloudShell

このトピックでは、 を使用して AWS CloudShell アクティビティとパフォーマンスをログに記録して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 CloudTrail

AWS CloudShell は と統合されています。これは AWS CloudTrail、ユーザー、ロール、または AWS のサービス で実行されたアクションを記録するサービスです AWS CloudShell。 は、 のすべての API コールをイベント AWS CloudShell として CloudTrail 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は、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 [Event history (イベント履歴)] で最新のイベントを表示できます。によって収集された情報を使用して CloudTrail、リクエストに関するさまざまな情報を検出できます。例えば、AWS に対して行われたリクエスト、リクエスト元の IP アドレス、リクエスト者、リクエスト日時 CloudShellを確認できます。

AWS CloudShell の CloudTrail

次の表は、ログファイルに保存 CloudTrail されている AWS CloudShell イベントの一覧です。

> **Note**
>
> AWS CloudShell 以下を含む イベント：
>
> - * は、変更なし (読み取り専用) API コールであることを示します。
> - という単語は、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Environmentに関連しています。
> - という単語は、ターミナルのすべてのブラウザタブをLayout復元します CloudShell。

CloudShell のイベント CloudTrail

イベント名	説明
createEnvironment	CloudShell 環境の作成時に発生します。
createSession	CloudShell 環境が から接続されたときに発生します AWS Management Console。
deleteEnvironment	CloudShell 環境が削除されたときに発生します。
deleteSession	現在のブラウザ CloudShell タブで実行されているタブのセッションが削除されたときに発生します。
getEnvironmentStatus*	CloudShell 環境のステータスが取得されたときに発生します。
getFileDownloadUrls*	CloudShell ウェブインターフェイス CloudShell を使用して からファイルをダウンロードするために使用される署名付き Amazon S3 URLs が生成されたときに発生します。
getFileUploadUrls*	CloudShell ウェブインターフェイス CloudShell を使用して からファイルをアップロードするために使用される署名付き Amazon S3 URLs が生成されたときに発生します。
getLayout*	セッション開始時の CloudShell レイアウトが取得されたときに発生します。
putCredentials	への ログインに使用される認証情報 AWS Management Console CloudShell が転送されたときに発生します。

イベント名	説明
redeemCode*	環境で更新トークンを取得するワークフローが開始されたときに CloudShell発生します。後でこのトークンを putCredentials コマンドで使用して CloudShell 環境にアクセスできます。
sendHeartBeat	CloudShell 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 環境が開始され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	バックエンドのウェブアプリケーションからの現在のレイアウトが保存されたときに発生します。

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

EventBridge AWS CloudShell アクションの ルール

EventBridge ルールでは、 がルールに一致するイベント EventBridge 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アクションに基づいて実行するターゲット AWS CloudShell アクションを指定するルールを定義できます。

例えば、 put-rule コマンドを使用して で EventBridge ルールを作成できます AWS CLI。put-rule コールには、少なくとも EventPattern または が含まれている必要があります ScheduleExpression。を持つルール EventPatterns は、一致するイベントが観測されたときにトリガーされます。 EventPattern AWS CloudShell イベントの :

```
{ "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「 のイベントとイベントパターン EventBridge」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、さまざまな 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 System and Organization Controls (SOC) レポートは、 が AWS 主要なコンプライアンス管理と目標を達成した方法を示す、独立したサードパーティーによる審査報告書です。

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

PCI

Payment Card Industry Data Security Standard (PCI DSS) は、PCI セキュリティ標準委員会によって管理される専有情報セキュリティ標準であり、 American Express、Discover Financial Services、JCB International、 MasterCard Worldwide、および Visa Inc によって設立されています。

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 には、ISO/IEC 27001:2013、27017:2015、27018:2019、27701:2019、22301:2019、9001:2015、および CSA TAK CCM v4.0 への準拠に関する認定があります。

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、認証、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 の暫定認証を取得して 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承認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	3PAO 評価	該当なし	該当なし	該当なし	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 の対象となるエンティティとそのビジネスアソシエイトが、保護対象の医療情報 (PHI) を安全に処理、保存、転送できるようにします。さらに、2013 年 7 月現在、 は、そのようなお客様向けに標準化された事業提携契約 (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。例えば、IAM ポリシーを作成するときは、Amazon リソースネーム (ARNsを使用して AWS CloudTrail ログを読み込みます。

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 要件に合致していることは、フィンランドのトランスポート通信局である Traficom が設定した、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サービスのリストについては、 AWS「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「コンプライアンスプログラム」を参照してください。

サードパーティーの監査レポートは、 を使用して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、企業のコンプライアンス目的、適用法規によって決まります。 では、コンプライアンスに役立つ以下のリソース AWS を提供しています。

- セキュリティおよび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 にデプロイするための手順を説明します 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、お客様の業界や場所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、リソース設定が社内プラクティス、業界ガイドライン、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 — 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および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、および高度の冗長ネットワークで接続されている複数の物理的に独立および隔離された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が高く、フォールトトレラントで、スケーラブルです。

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に対応できるように特定の機能 AWS CloudShell をサポートしています。

- 作成して に追加するファイルをコミットします AWS CodeCommit。これは、クラウド内のアセット (ドキュメント、ソースコード、バイナリファイルなど) をプライベートで保存および管理するために使用できる Amazon Web Services によってホストされるバージョン管理サービスです。これらのアセットは、ドキュメント、ソースコード、およびバイナリファイルで構成しています。詳細については、「チュートリアル: CodeCommit での使用AWS CloudShell」を参照してください。
- 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 バケットにオブジェクトとして追加します。例については、入門チュートリアル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 グローバル AWS 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 AWS 環境を設計するには、「セキュリティの柱 AWS Well-Architected Framework」の「インフラストラクチャ保護」を参照してください。

が AWS 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。クライアントは以下をサポートする必要があります:

- Transport Layer Security (TLS)。TLS 1.2、できれば TLS 1.3 が必要です。
- DHE (Ephemeral Diffie-Hellman) や ECDHE (Elliptic Curve Ephemeral Diffie-Hellman) などの Perfect Forward Secrecy (PFS) を使用した暗号スイート。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。

また、リクエストには、アクセスキー ID と、IAM プリンシパルに関連付けられているシークレットアクセスキーを使用して署名する必要があります。または、AWS Security Token Service (AWS STS) を使用して、一時的なセキュリティ認証情報を生成し、リクエストに署名することもできます。

> **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。

での設定と脆弱性の分析 AWS CloudShell

コンピューティング環境にインストールしたソフトウェアにパッチが適用され、最新であることを確認するのは AWS CloudShell ユーザーの責任です。

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ユーザー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、役に立つ情報としてお考えください。

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、ユーザーが自分のロールに必要なアクション (ファイルのダウンロードとアップロードなど) のみを実行 AWS CloudShell 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。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。
- サードパーティーアプリケーションを AWS CloudShellのコンピューティング環境にインストールした場合、共有セキュリティ責任モデルについて理解を深めます。
- 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。詳細については、「スクリプトでシェルを修正する」を参照してください。
- コードをバージョン管理システムに安全に保存します。例: AWS CodeCommit。

AWS CloudShell セキュリティFAQs

この AWS のサービスに関するよくある質問への回答。

- シェルセッションを起動して開始するときに使用される AWS プロセス CloudShell とテクノロジーは何ですか？
- へのネットワークアクセスを制限することは可能です CloudShellか？
- CloudShell 環境をカスタマイズできますか？
- 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?
- 自分の $HOME ディレクトリを暗号化することはできますか?
- 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?

シェルセッションを起動して開始するときに使用される AWS プロセス CloudShell とテクノロジーは何ですか？

にサインインするときは AWS Management Console、IAM ユーザーの認証情報を入力します。また、コンソールインターフェイス CloudShell から 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 への呼び出しで使用されます。その後、 AWS

Systems Manager セッションがコンピューティング環境用に作成され、そのセッションにコマンド CloudShell が送信されます。

セキュリティに関するよくある質問リストに戻る

へのネットワークアクセスを制限することは可能です CloudShellか？

ネットワークプロバイダーに接続 CloudShell することで、 へのネットワークアクセスを制限できる場合があります。または、IAM アクセス許可の使用、IAM アクセス許可の明示的な拒否 CloudShell または提供の拒否、および暗黙的な拒否の IAM 機能を使用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。

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ー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?

$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。

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?

$HOME ディレクトリ内のデータは、Amazon S3 Encryption を使用して既に暗号化されています。

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

AWS CloudShell コンピューティング環境: 仕様およびソフトウェア

を起動するとAWS CloudShell、シェルエクスペリエンスをホストするための 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。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。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

どの AWS CloudShell コンピューティング環境にも、次の CPU および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、「永続的ストレージ」を参照してください。

CloudShell ネットワーク要件

WebSockets

CloudShell WebSocket プロトコルによって異なります。これにより、 CloudShell AWSユーザーのウェブブラウザとクラウド内のサービスとの間で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は、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容易になると考えられます。 WebSocket 通常、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、 WebSockets プロキシサーバーが正しく動作しなくなることがあります。この問題が発生すると、 CloudShell `Failed to open sessions : Timed out while opening the session`インターフェイスに次のエラーが報告されます。

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、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 WebSockets。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 を許可リストに追加して詳細な権限を定義したい場合は、AWS Systems Manager WebSocket セッションが入力を送受信するための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 StreamUrl が使用するこの形式はです`wss://ssmmessages.`**`region`**`.amazonaws.com/v1/data-channel/`**`session-id`**`?stream=(input|output)`。
> リージョンは、米国東部 (オハイオ) リージョンの `us-east-2` のように、AWS Systems Manager でサポートされている AWS リージョンのリージョン識別子を表します。
> 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 ManagerAPI StartSessionリファレンスのオペレーションをSession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ているので、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`) 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	[Version information]
Bash	Bash シェルは、AWS CloudShell のデフォルトのシェルアプリケーションです。	`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サポートを提供し、Microsoft の.NET PowerShell コマンド言語ランタイムをベースに構築されています。 PowerShell .NET オブジェクトを受け入れたり返したりする、`cmdlets`という軽量コマンドを使用します。	`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	[Version information]
AWS CDK ツールキット CLI	AWS CDK ツールキット、CLI コマンド、 `cdk` は、 AWS CDK アプリを操作する主要なツールです。アプリを実行し、定義したアプリケーショ	`cdk --version`

名前	説明	[Version information]
	ンモデルを調べ、 AWS CDK によって生成された AWS CloudFormation テンプレートを作成してデプロイします。 詳細については、「AWS CDK ツールキット」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のコマンドラインインターフェイスです。詳細については、「AWS での AWS CloudShell サービスの使用」を参照してください。 up-to-date AWS CLIほとんどのバージョン 2 を使用していることを確認する方法については、を参照してくださいAWS CLI をホームディレクトリにインストールする。	`aws --version`

名前	説明	[Version information]
EB CLI	EB CLI は AWS Elastic Beanstalk のコマンドラインインターフェイスで、ローカルリポジトリからの環境作成、更新、およびモニタリングを簡素化するインタラクティブなコマンドを提供します。 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	[Version information]
AWS SAM CLI	AWS SAM CLI は、AWS Serverless Application Model テンプレートおよび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。Lambda 関数をローカルで呼び出すサーバーレスアプリケーションのデプロイパッケージを作成するサーバーレスアプリケーションを AWS Cloud にデプロイするなどがあります。 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	[Version information]
AWS Tools for PowerShell	AWS Tools for PowerShellこれらは、 PowerShell AWS SDK for .NETによって公開されている機能に基づいて構築されたモジュールです。を使用するとAWS Tools for PowerShell、AWS PowerShell コマンドラインからリソースに対する操作をスクリプト化できます。 AWS CloudShell は、 AWS Tools for PowerShell のモジュール化バージョン (AWS.Tools) をプレインストールします。 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-Module -ListAvailable -Name AWS.Tools.Common'`

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	[Version information]
Node.js (npm 付き)	Node.js は、 JavaScript 非同期プログラミングの手法を簡単に適用できるように設計されたランタイムです。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	[Version information]
	npm は、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。 JavaScript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。	
Node.js JavaScript 内の SDK	ソフトウェア開発キット (SDK) は、Amazon S3、Amazon EC2、DynamoDB、Amazon SWF などの AWS JavaScript サービスにオブジェクトを提供することで、コーディングを簡素化します。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	[Version information]
Python	Python 3 はシェル環境ですぐに使用できます。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	[Version information]
SDK for Python (Boto3)	Boto は Python デベロッパーが Amazon EC2 や Amazon S3 などの AWS のサービス を作成、設定、管理するために使用する Software Development Kit (SDK) です。SDK はオブジェクト指向の API だけでなく、への低レベルアクセスも提供します。 easy-to-use 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	[Version information]
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にする	`dnf info bash-completion`

名前	説明	[Version information]
	必要があります。例えば、Git コマンドの自動入力を設定するには、次の行を .bashrc に追加して、AWS CloudShell セッションの開始時にいつでもこの機能を利用できるようにします。 `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、プロジェクトの README ページを参照してください。 GitHub	

名前	説明	[Version information]
CodeCommit Git 用ユーティリティ	git-remote-codecommit は、Git CodeCommit を拡張してリポジトリからコードをプッシュしたりプルしたりする簡単な方法を提供するユーティリティです。これは、フェデレーティッドアクセス、アイデンティープロバイダー、および一時的な認証情報を使用した接続をサポートするために推奨される方法です。 詳細については、『ユーザーガイド』の「AWS CodeCommitwith git-remote-codecommit への HTTPS 接続の設定手順」を参照してください。AWS CodeCommit	`pip3 list | grep git-remote-codecommit`
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`

名前	説明	[Version information]
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。提供されているユーティリティの詳細については、の iputils リポジトリを参照してください。 GitHub	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、ホストされている jq マニュアルを参照してください。 GitHub	`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`
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`

名前	説明	[Version information]
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`
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`

名前	説明	[Version information]
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	[Version information]
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`
unzip	詳細については、「zip/unzip」を参照してください。	
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	[Version information]

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`

名前	説明	[Version information]
Docker	Docker は、アプリケーションの開発、出荷、実行の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ーから切り離すことができるため、ソフトウェアを迅速に提供できます。これにより、内部で Dockerfile 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 でどのリージョンがサポートされているかについては、「Docker リージョン」を参照してください。Docker には環境内のスペースが限られていることに注意する必要があります。個別のイメージが大きかったり、既存の Docker イメージが多すぎたりすると、問題が発生する可能性があります。Docker について詳しくは、『Docker ドキュメンテーションガイド』を参照してください。	`docker --version`

AWS CLI をホームディレクトリにインストールする

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によって自動的に更新されます。 up-to-date の最新バージョンを確実に入手したい場合は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もできます。

> **Important**
>
> AWS CLIのコピーをホームディレクトリに手動でインストールして、 CloudShell 次回セッションを開始したときに利用できるように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 AWS CLI のコピーをインストールした後は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。
> AWS 責任共有モデルの詳細については、「でのデータ保護 AWS CloudShell」を参照してください。

AWS CLI をインストールするには

1. CloudShell コマンドラインで、curl次のコマンドを使用して、AWS CLIインストールした ZIP 形式の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 インストーラーを実行します。

```
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 PATH に戻すと、指定したパスを機能としない aws コマンドは、デフォルトでプリインストールされた AWS CLI の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**
>
> AWS CloudShell のコンピューティング環境に、サードパーティーアプリケーションをインストールする前に、共有セキュリティ責任モデルをチェック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 の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。たとえば、DNF パッケージ管理ユーティリティーと併用してインストールするとcowsay、sudo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 インストールプログラムなどのPackage /usr/bin 管理ユーティリティをディレクトリに (たとえば) インストールします。これら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。
> 誤って .bashrc または他のファイルを変更した後にアクセスできなくなった場合、ホームディレクトリを削除することで、AWS CloudShell デフォルト設定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。

Amazon Linux 2 から Amazon Linux 2023 への AWS CloudShell の移行

AWS CloudShellは Amazon Linux 2 (AL2) をベースにしていましたが、Amazon Linux 2023 (AL2023) に移行しました。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、が提供するすべてのツールを使用して、 CloudShell 引き続き既存の環境にアクセスできます。 CloudShell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、開発ツールにいくつかの改良が加えられています。

> **Note**
>
> AL2023 では、Python 2 はお客様の環境に同梱されなくなりました。 CloudShell

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、AWS Support 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。AWS re:Post 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。

AWS CloudShell 移行に関するよくある質問

以下は、AWS CloudShell を使用した AL2 から AL2023 への移行に関するよくある質問に対する回答です。

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

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

ご利用の AWS CloudShell 環境以外のサービスやリソースには、この移行による影響はありません。これには、ユーザーが AWS CloudShell 内で作成またはアクセスした可能性のあるリソースも含まれます。たとえば、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?

現在の AWS CloudShell 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。プレインストールされたソフトウェアの全リストについては、「プレインストールソフトウェア」を参照してください。 AWS CloudShellPython 2 を除いて、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 への移行後には要件が満たされない特定のパッケージとバージョンをお持ちの場合は、AWS サポート部にリクエストを送信することを推奨します。

移行をオプトアウトすることはできますか

答えは「いいえ」です。 AWS CloudShell環境はによって管理されるためAWS、すべての環境が AL2023 にアップグレードされました。

自分の 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。

AWS CloudShell のトラブルシューティング

使用中にAWS CloudShell、 CloudShell シェルのコマンドラインインターフェイスを使用してキータスクを起動または実行するときなど、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。

に関するさまざまな質問への回答については CloudShell、AWS CloudShellFAQ を参照してください。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。

に関連する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

- 環境を起動できません。再試行するには、ブラウザを更新するか、[アクション]、 [AWS CloudShell の再起動] を選択して再起動してください。
- 環境を起動できません。必要なアクセス許可がありません。IAM 管理者に AWS CloudShell へのアクセス権を申請してください。
- AWS CloudShell コマンドラインにアクセスできません
- 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
- では矢印キーが正しく機能しない PowerShell
- サポートされていない Web ソケットがあると、セッションの開始に失敗します。 CloudShell
- AWSPowerShell.NetCore モジュールをインポートできない。
- を使用しているときに Docker が動作しない AWS CloudShell
- Docker のディスク容量が不足しています。
- docker pushがタイムアウトになり、再試行が繰り返されます。

環境を起動できません。再試行するには、ブラウザを更新するか、[アクション]、 [AWS CloudShell の再起動] を選択して再起動してください。

問題:AWS CloudShellから起動しようとするとAWS Management Console、IAM 管理者から必要な権限を得て、ブラウザを更新したり再起動したりしても、アクセスが拒否されます。 CloudShell

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)

環境を起動できません。必要なアクセス許可がありません。IAM 管理者に AWS CloudShell へのアクセス権を申請してください。

問題: AWS Management Console から AWS CloudShell を起動しようとすると、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: AWS CloudShell へのアクセスに使用している IAM アイデンティティに必要な IAM アクセス許可がありません。

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。そのためには、AWS添付の管理ポリシー (AWSCloudShellFullAccess) を追加するか、埋め込まれた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ません

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に、AWS CloudShell のコマンドラインにアクセスできまん。

解決策: `.bashrc` または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い場合、ホームディレクトリを削除することで AWS CloudShell をデフォルト設定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 プロトコルが AWS CloudShell で有効になっていないため、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は動作しません。

解決策:では ICMP が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を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 ブラウザを使用して AWS CloudShell 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 は、Microsoft Edge の以前のバージョンとの互換性がありません。サポートされるブラウザの最新の 4 つのメジャーバージョンで AWS CloudShell にアクセスすることができます。

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)

では矢印キーが正しく機能しない PowerShell

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。ただし、 PowerShell on の特定のバージョンでは矢印キーを押すとAWS CloudShell、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 で実行されている PowerShell 7.2.x バージョンの既知の問題です。

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、変数をに設定します。`$PSStyle` `PlainText`

1. AWS CloudShell コマンドラインで、以下のコマンドを入力してプロファイルファイルを生成します。

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

> **Note**
>
> 既に編集中の場合は PowerShell、以下のコマンドでプロファイルファイルをエディターで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**
>
> PowerShell変更内容は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 Web ソケットがあると、セッションの開始に失敗します。 CloudShell

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

原因: CloudShell WebSocket プロトコルによって異なります。これにより、Web ブラウザとの間で双方向の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は、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容易になると考えられます。 WebSocket 通常、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、プロキシサーバーが正しく動作しない場合もあります。 WebSockets この問題が発生すると、 CloudShell シェルセッションを開始できなくなり、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、 WebSocketsサポート対象外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イ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 WebSocketセッションが入力の送信と出力の受信のための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 StreamUrl が使用するこの形式はです `wss://ssmmessages.region.amazonaws.com/v1/data-channel/session-id?stream=(input|output)`。
> リージョンは 、 AWS Systems Manager でサポートされている AWS リージョン リージョン識別子を表します。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 ManagerAPI StartSessionリファレンスのオペレーションをAPI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:をインポートするとき AWSPowerShell。 NetCore PowerShell by のモジュールに`Import-Module -Name AWSPowerShell.NetCore`、次のエラーメッセージが表示されます。

インポートモジュール:指定されたモジュール '. AWSPowerShell NetCoreどのモジュールディレクトリにも有効なモジュールファイルが見つからなかったため、'はロードされませんでした。

原因: **`AWSPowerShell.NetCore`** モジュールが AWS CloudShell 内のサービスごとの AWS .Tools モジュールに置き換えられます。

解決策:明示的なインポートステートメントは不要になるか、関連するサービスごとの AWS .Tools モジュールに変更する必要がなくなる可能性があります。

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)

を使用しているときに Docker が動作しない AWS CloudShell

問題:使用時に Docker が正しく動作しない。AWS CloudShell`docker: Cannot connect to the Docker daemon at unix:///var/run/docker.sock. Is the docker daemon running?`次のエラーメッセージが表示されます。

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、Docker AWS CloudShell をサポートしていないリージョンで Docker を実行した場合に発生する可能性があります。サポートされ

ているリージョンで Docker を実行していることを確認してください。どのリージョンが Docker コンテナの使用をサポートしているかについてはAWS CloudShell、「Docker リージョン」を参照してください。

Docker のディスク容量が不足しています。

問題:次のエラーメッセージが表示される:`ERROR: failed to solve: failed to register layer: write [...]: no space left on device`

原因:Dockerfile がで利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かったり、既存の Docker イメージが多すぎたりすることが原因である可能性があります。

解決策:`df -h`実行してディスク使用量を調べてください。`sudo du -sh /folder/folder1`実行して容量が大きいと思われる特定のフォルダーのサイズを確認し、他のファイルを削除して空き容量を増やすことを検討してください。1 つの選択肢は、未使用の Docker イメージを実行して削除することを検討することです。`docker rmi`Docker には環境内のスペースが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 ドキュメンテーションガイドを参照してください。

`docker push`がタイムアウトになり、再試行が繰り返されます。

問題:`docker push`実行するとタイムアウトになり、再試行しても成功しません。

原因:権限がない、間違ったリポジトリにプッシュされている、または認証されていないこと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てください。`docker login`実行して正しく認証してください。Amazon ECR リポジトリへのプッシュに必要なすべての権限があることを確認してください。

AWS CloudShell のサポートされるブラウザ

次の表は、AWS CloudShell でサポートされるブラウザの一覧です。

ウェブブラウザのサポート

ブラウザ	バージョン
Google Chrome	最新の 3 つのメジャーバージョン
Mozilla Firefox	最新の 3 つのメジャーバージョン
Microsoft Edge	最新の 3 つのメジャーバージョン
MacOS 版 Apple Safari	最新の 2 つのメジャーバージョン

AWS CloudShell のサポート対象 AWS リージョン

このセクションでは、AWS でサポートされている AWS CloudShell リージョンとオプトインリージョンのリストについて説明します。AWSのサービスエンドポイントとクォータのリストについては CloudShell、AWS CloudShellのページを参照してください。Amazon Web Services 全般のリファレンス

以下が AWS CloudShellでサポートされる AWS リージョンです。

- 米国東部 (オハイオ)
- 米国東部 (バージニア北部)
- 米国西部 (北カリフォルニア)
- 米国西部 (オレゴン)
- アジアパシフィック (ムンバイ)
- 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 (シドニー)
- アジアパシフィック (シンガポール)
- アジアパシフィック (東京)
- カナダ (中部)
- 欧州 (フランクフルト)
- 欧州 (アイルランド)
- 欧州 (ロンドン)
- 欧州 (パリ)
- 欧州 (ストックホルム)
- 南米 (サンパウロ)

GovCloud リージョン

GovCloud CloudShellサポートされているリージョンは以下のとおりです。

- AWS GovCloud (米国東部)
- AWS GovCloud (米国西部)

オプトインリージョン

デフォルトでは、オプトインリージョンは有効ではありません。これらのリージョンを使用するには、手動で有効にする必要があります。詳細については、「AWS リージョンの管理」を参照してください。サポートされているオプトインリージョンは以下のとおりです。 CloudShell

- 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 (ジャカルタ)
- 欧州 (ミラノ)
- 中東 (バーレーン)
- 中東 (アラブ首長国連邦)

Docker でサポートされているリージョン

AWS CloudShellコンピューティング環境は、以下のリージョンの Docker コンテナのみをサポートします。

- 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 (ムンバイ)
- アジアパシフィック (シドニー)
- アジアパシフィック (シンガポール)
- アジアパシフィック (東京)
- カナダ (中部)
- 欧州 (フランクフルト)
- 欧州 (アイルランド)
- 欧州 (ロンドン)
- 欧州 (パリ)
- 南米 (サンパウロ)

AWS CloudShell のサービスクォータと制限

このページでは、次の領域に適用されるサービスクォータと制限について説明します。

- パーシスタントストレージ
- 月間使用量
- コマンドサイズ
- 同時シェル数
- シェルセッション
- ネットワークアクセスおよびデータ転送
- システムファイルとページの再ロード

永続的ストレージ

ではAWS CloudShell、それぞれに 1 GBAWS リージョン の永続的ストレージを無料でご利用いただけます。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。

AWS CloudShellでの使用を停止するとAWS リージョン、データは、最後のセッションが終了してから 120 日間だけリージョンの永続的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。その中でAWS CloudShellをもう一度起動すれば削除を防止できますAWS リージョン。詳細については、「ステップ 2: リージョンの選択、AWS の起動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) および欧州 (アイルランド) という 2AWS リージョン つのホームディレクトリにファイルを保存していました。その後、彼女はヨーロッパ (アイルランド) でのみ AWS CloudShell を使用するようになり、米国東部 (バージニア北部) で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) でのデータの削除期限前に、Márcia はAWS CloudShellを起動し、米国東部 (バージニア北部) リージョンをもう一度選択して、自分のホームディレクトリ

がリサイクルされないことを確認しました。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 CloudShellAWS リージョンにはそれぞれ毎月の使用クォータがありますAWS アカウント. AWS CloudShellそのリージョンの月間の制限に達した後で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。

> **Note**
>
> 月間使用制限を増やすには、AWS Support にお問い合わせください。

コマンドサイズ

コマンドサイズは 65412 文字を超えることはできません。

> **Note**
>
> 65412 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イスから実行します。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の詳細については、プリインストールソフトウェアを参照してください。
> スクリプトを作成し、コマンドラインインターフェイスから実行する方法の例については、「チュートリアル:はじめに」を参照してくださいAWS CloudShell。

同時シェル数

- 同時シェル数:AWS リージョンアカウントごとに同時に最大 10 個まで実行できます。

シェルセッション

- 非アクティブセッション:AWS CloudShellは対話型のシェル環境であり、20～30 分にわたってキーボードとポインタを使った操作がない場合、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。
- 実行時間が長いセッション:約 12 時間連続して実行され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。

ネットワークアクセスおよびデータ転送

AWS CloudShell環境のインバウンドトラフィックおよびアウトバウンドトラフィックの両方に適用されます。

- 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**
>
> 公開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伴います。IAM 管理者は、IAM ツールを介して、AWS CloudShell信頼されたユーザーの許可リストを管理してください。特定のユーザによるアクセスを明示的に拒否する方法については、「」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。

データ転送:大容量ファイルの場合、AWS CloudShellとの間でファイルをアップロードやダウンロードする処理が遅くなることがあります。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 環境にアクセスしてそれを使用する際に問題が起き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インターフェイ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
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。	内部で Docker コンテナを構築して Amazon ECR AWS CloudShell リポジトリにプッシュする方法と、経由で Lambda 関数をデプロイする方法を詳述する 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
Docker コンテナは特定のリージョンでサポートされています AWS CloudShell	とのDockerコンテナSupport 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現在は Amazon Linux 2023 (AL2023) を使用するように移行しました	AWS CloudShell現在 AL2023 を使用しており、Amazon Linux 2 から移行しました。	2023 年 12 月 4 日
AWS CloudShell の新しい AWS リージョン	次の AWS リージョンで AWS CloudShell が一般提供 (GA) されました。 • 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)	2023 年 6 月 16 日

	• 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)	
Console Toolbarで AWS CloudShell を起動します	CloudShell コンソールの左下にあるを選択して起動します。Console Toolbar CloudShell	2023 年 3 月 28 日
AWS CloudShell の新しい AWS リージョン	AWS CloudShell は以下の AWS リージョンで利用できるようになりました。 • 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 でサポートされています GovCloud	AWS CloudShellは 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	CloudShell WebSocketプロトコルの使用方法を説明するセクションをネットワーク要件に追加しました。	2022 年 3 月 21 日

の矢印キーのトラブルシューティング PowerShell	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 リージョン コマンドのデフォルトAWS CLI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 リージョンでの AWS CloudShell の一般提供 (GA)リリース	次の AWS リージョンで AWS CloudShell が一般提供 (GA) されました。 • 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

翻訳は機械翻訳により提供されています。提供された翻訳内容と英語版の間で齟齬、不一致または矛盾がある場合、英語版が優先します。