aws

ユーザーガイド

AWS Transfer Family

AWS Transfer Family: ユーザーガイド

Table of Contents

とは AWS Transfer Family

AWS Transfer Family は、 AWS ストレージサービスとの間でファイルを転送できる安全な転送サービスです。Transfer Family は AWS クラウド platform の一部です。 は、SFTP、AS2、FTPS、および FTP 経由で Amazon S3 または Amazon EFS との間で直接ファイルを転送するためのフルマネージドサポート AWS Transfer Family を提供します。認証、アクセス、ファイアウォールの既存のクライアント側設定を維持することで、ファイル転送ワークフローをシームレスに移行、自動化、モニタリングできるため、顧客、パートナー、内部チーム、またはアプリケーションに変更を加える必要はありません。

詳細については、「 の開始方法 AWS」を参照して、Amazon Web Services でクラウドアプリケーションの構築を開始してください。

AWS Transfer Family は、以下の AWS ストレージサービスとの間でのデータ転送をサポートします。

- Amazon Simple Storage Service (Amazon S3) ストレージ。Amazon S3 の詳細については、「Amazon Simple Storage Service の開始方法」を参照してください。
- Amazon Elastic File System （ Amazon EFS ） Network File System （ NFS ）ファイルシステム。Amazon EFS については、「Amazon Elastic File System とは？」を参照してください。

AWS Transfer Family は、次のプロトコルを介したデータ転送をサポートします。

- Secure File Transfer Protocol (SFTP): バージョン 3

  公式の IETF ドキュメントは、SSH File Transfer Protocol draft-ietf-secsh-filexfer-02.txt です。
- File Transfer Protocol Secure (FTPS)
- File Transfer Protocol (FTP)
- 適用性ステートメント 2 (AS2)

> **Note**
>
> FTP および FTPS データ接続の場合、Transfer Family がデータチャネルを確立するために使用するポート範囲は 8192～8200 です。

ファイル転送プロトコルは、金融サービス、医療、広告、リテールなどのさまざまな業界にわたってデータ交換ワークフローに使用されます。Transfer Family は、ファイル転送ワークフローの への移行を簡素化します AWS。

以下は、Amazon S3 で Transfer Family を使う場合の一般的な使用例である：

- ベンダーやパートナーなどのサードパーティーからのアップロード AWS 用の のデータレイク。
- 顧客によるサブスクリプション型のデータ分散。
- 組織内の内部転送。

以下は、Amazon EFS で Transfer Family を使用する一般的なユースケースである：

- データのディストリビューション
- サプライチェーン
- コンテンツ管理
- ウェブサーバーアプリケーション

以下は、AS2 で Transfer Family を使用する場合の一般的な使用例である：

- プロトコルにデータ保護とセキュリティ機能が組み込まれていることが前提となる、コンプライアンス要件のあるワークフロー
- サプライチェーンロジスティクス
- 支払いワークフロー
- B usiness-to-business (B2B) トランザクション
- 企業資源管理 (ERP) システムおよび顧客関係管理 (CRM) システムとの統合

Transfer Family を使用すると、サーバーインフラストラクチャを実行する AWS ことなく、 でファイル転送プロトコル対応サーバーにアクセスできます。このサービスを使用して、エンドユーザーのクライアントと設定をそのまま維持 AWS しながら、ファイル転送ベースのワークフローを に移行できます。最初にホスト名をサーバーエンドポイントに関連付けてから、ユーザーを追加して、適切なアクセスレベルをプロビジョニングします。これにより、ユーザーの転送リクエストが Transfer Family サーバーエンドポイントから直接提供されます。

Transfer Family には次の利点があります。

- ニーズに合わせてリアルタイムでスケーリングが可能なフルマネージドサービス。

- アプリケーションを変更したり、FTP インフラストラクチャを運用する必要はありません。
- 耐久性の高い Amazon S3 ストレージ内のデータでは、ネイティブ を使用して AWS のサービス処理、分析、レポート、監査、アーカイブ機能を実行できます。
- Amazon EFS をデータストアとして使用すると、 AWS クラウド のサービスやオンプレミスリソースで使用できるように、フルマネージド型の Elastic File System を利用できます。Amazon EFS は、アプリケーションを中断することなく、ファイルの追加や削除に伴って自動的に伸縮し、ペタバイト規模までオンデマンドで拡張できるように構築されています。これにより、拡張に対応するために容量をプロビジョニングおよび管理する必要がなくなります。
- フルマネージドサーバーレスファイル転送ワークフローサービスで、 AWS Transfer Familyを使用してアップロードされたファイルを処理する際の設定、実行、自動化、モニタリングを容易にします。
- 前払い料金はなく、サービスの使用料金のみを支払います。

以下のセクションでは、Transfer Family のさまざまな機能の説明、入門チュートリアル、さまざまなプロトコル対応サーバーを設定するための詳細手順、各種の ID プロバイダーを使用する方法、およびサービスの API リファレンスについて説明します。

Transfer Family の使用を開始するには、以下を参照してください。

- AWS Transfer Family の仕組み
- 前提条件
- AWS Transfer Family サーバーエンドポイントの開始方法

AWS Transfer Family の仕組み

AWS Transfer Family は、以下のプロトコルを使用して、Amazon Simple Storage Service (Amazon S3) ストレージまたは Amazon Elastic File System (Amazon EFS) ファイルシステムとの間でファイルを転送するために使用できるフルマネージド AWS サービスです。

- Secure File Transfer Protocol (SFTP): バージョン 3

  公式の IETF ドキュメントは、SSH File Transfer Protocol draft-ietf-secsh-filexfer-02.txt です。
- File Transfer Protocol Secure (FTPS)
- File Transfer Protocol (FTP)
- 適用性ステートメント 2 (AS2)

AWS Transfer Family は最大 3 つのアベイラビリティーゾーンをサポートし、自動スケーリング、接続および転送リクエスト用の冗長フリートによってバックアップされます。レイテンシーベースのルーティングを使用して冗長性を高め、ネットワークレイテンシーを最小限に抑える方法の例については、ブログ記事「SFTP サーバーの AWS 転送によるネットワークレイテンシーの最小化」を参照してください。

Transfer Family Managed File Transfer Workflows (MFTW) は、フルマネージドサーバーレスファイル転送ワークフローサービスで、 AWS Transfer Familyを使用してアップロードされたファイルを処理する際の設定、実行、自動化、モニタリングを容易にします。お客様は MFTW を使用して、Transfer Family を使用して転送されるデータのコピー、タグ付け、スキャン、フィルタリング、圧縮/解凍、暗号化/復号化など、さまざまな処理ステップを自動化できます。これにより、追跡と監査可能性についてエンドツーエンドの可視性が提供されます。詳細については、「AWS Transfer Family マネージドワークフロー」を参照してください。

AWS Transfer Family は、標準のファイル転送プロトコルクライアントをサポートします。一般的に使用されるクライアントには次のものがあります。

- OpenSSH – macOS と Linux のコマンドラインユーティリティです。
- WinSCP – Windows 専用のグラフィカルクライアントです。
- Cyberduck – Linux、macOS、および Microsoft Windows のグラフィカルクライアントです。
- FileZilla – Linux、Macintosh、および Windows のグラフィカルクライアント。

AWS では、以下の Transfer Family ワークショップを提供しています。

- マネージド SFTP/FTPS エンドポイントに 、ユーザー管理 AWS Transfer Family に Amazon Cognito と DynamoDB を活用するファイル転送ソリューションを構築します。このワークショップの詳細については、「こちら」をご覧ください。
- AS2 を有効にして Transfer Family エンドポイントを構築し、Transfer Family AS2 コネクタを構築します。このワークショップの詳細は、「」で確認できます。
- 既存のアプリケーションを変更したり、サーバーインフラストラクチャを管理 AWS したりすることなく、 でスケーラブルで安全なファイル転送アーキテクチャを構築する方法に関する規範的なガイダンスと実践的なラボを提供するソリューションを構築します。このワークショップの詳細については、「こちら」をご覧ください。

Transfer Family に関連するブログ記事

次の表は、Transfer Familyのお客様に役立つ情報を含むブログ投稿の一覧です。テーブルは時系列の逆順になっているため、最新の投稿はテーブルの先頭に表示されます。

ブログ投稿のタイトルとリンク	日付
AWS Transfer Family SFTP コネクタと PGP 暗号化を使用した、安全で準拠のマネージドファイル転送の設計	2024 年 5 月 16 日
AWS Transfer Family および Amazon S3 での ID プロバイダーとしての Amazon Cognito Amazon S3の使用	2024 年 5 月 14 日
Transfer Family が安全で準拠のマネージドファイル転送ソリューションの構築にどのように役立つか	2024 年 1 月 3 日
を使用してマルウェアの脅威を検出する AWS Transfer Family	2023 年 7 月 20 日
による SAP ワークロードの拡張 AWS Transfer Family	2023 年 7 月 13 日
PGP と を使用してファイルを暗号化および復号する AWS Transfer Family	2023 年 6 月 21 日
Azure Active Directory と AWS Transfer Family を使用した への認証 AWS Lambda	2022 年 12 月 15 日
AWS Transfer Family マネージドワークフローを使用してファイル配信通知をカスタマイズする	2022 年 10 月 14 日
AWS Transfer Family ワークフローを使ったクラウドネイティブなファイル転送プラットフォームの構築	2022 年 1 月 5 日

ブログ投稿のタイトルとリンク	日付
A AWS Transfer Family と によるユーザーセルフサービスキー管理の AWS Lambda有効化。	2021 年 12 月 17 日
AWS Transfer Family と Amazon S3 によるデータアクセスコントロールの強化	2021 年 10 月 5 日
AWS Global Accelerator および AWS Transfer Family サービスを使用したインターネット向けファイル転送のスループットの向上	2021 年 6 月 7 日
AWS Web Application Firewall と Amazon API Gateway AWS Transfer Family による保護	2021 年 5 月 5 日
AWS Web Application Firewall と Amazon API Gateway AWS Transfer Family による保護	2021 年 1 月 15 日
AWS Transfer Family Amazon Elastic File System のサポート	2021 年 1 月 7 日
AWS Transfer Family を使用して のパスワード認証を有効にする AWS Secrets Manager	2020 年 11 月 5 日
AWS Transfer Family と を使用してデータアクセスを一元化する AWS Storage Gateway	2020年6月22日
サーバーレスアプリケーションで AWS Lambda に Amazon EFS を使用する	2020 年 6 月 18 日
IP 許可リストを使用して AWS Transfer Family サーバーを保護する	2020 年 4 月 8 日
SFTP サーバーの AWS 転送によるネットワークレイテンシーの最小化	2020 年 2 月 19 日
SFTP サーバーの への移行をリフトアンドシフトする AWS	2020 年 2 月 12 日

ブログ投稿のタイトルとリンク	日付
chroot ディレクトリと論理ディレクトリを使用して AWS SFTP 構造を簡素化する	2019 年 9 月 26 日
での ID プロバイダーとしての Okta の使用 AWS Transfer Family	2019 年 5 月 30 日

前提条件

以下のセクションでは、 AWS Transfer Family サービスを使用するために必要な前提条件について説明します。少なくとも、Amazon Simple Storage Service (Amazon S3) バケットを作成し、 AWS Identity and Access Management (IAM) ロールを介してそのバケットへのアクセスを提供する必要があります。また、ロールが信頼関係を確立することも必要です。この信頼関係により、Transfer Family は、バケットにアクセスするためのロールを引き受け、ユーザーからのファイル転送リクエストに対応できるようになります。

トピック

- サポートされている AWS リージョン、エンドポイント、クォータ
- にサインアップする AWS
- で使用するストレージを設定する AWS Transfer Family
- IAM ポリシーとロールを作成する

サポートされている AWS リージョン、エンドポイント、クォータ

AWS サービスにプログラムで接続するには、 エンドポイントを使用します。例えば、米国東部 (オハイオ) リージョン (us-east-2) のお客様のエンドポイントは ですtransfer.us-east-2.amazonaws.com。サービスクォータ (制限とも呼ばれます) は、 AWS アカウントのサービスリソースまたはオペレーションの最大数です。このガイドでは、 AS2 クォータおよび でクォータを確認できますSFTPコネクタのクォータ。

サポートされている AWS リージョン、エンドポイント、サービスクォータの詳細については、「」のAWS Transfer Family 「 エンドポイントとクォータ」を参照してくださいAmazon Web Services 全般のリファレンス。

にサインアップする AWS

アマゾン ウェブ サービス (AWS) にサインアップすると AWS、 AWS アカウントは を含む のすべてのサービスに自動的にサインアップされます AWS Transfer Family。料金は、使用するサービスの料金のみが請求されます。

AWS アカウントがすでにある場合は、次のタスクに進んでください。 AWS アカウントをお持ちでない場合は、以下の手順に従ってアカウントを作成してください。

がない場合は AWS アカウント、次の手順を実行して作成します。

にサインアップするには AWS アカウント

1. https://portal.aws.amazon.com/billing/signup を開きます。
2. オンラインの手順に従います。

   サインアップ手順の一環として、通話呼び出しを受け取り、電話キーパッドで検証コードを入力するように求められます。

   にサインアップすると AWS アカウント、 AWS アカウントのルートユーザーが作成されます。ルートユーザーには、アカウントのすべての AWS のサービス とリソースへのアクセス権があります。セキュリティのベストプラクティスとして、ユーザーに管理アクセスを割り当て、ルートユーザーのみを使用してルートユーザーアクセスが必要なタスクを実行してください。

料金と、Transfer Family の使用コストの見積もりを取得するために使用する方法については、「 のAWS Transfer Family 料金 AWS Pricing Calculator 」を参照してください。

AWS リージョンの可用性については、「」のAWS Transfer Family 「 エンドポイントとクォータ」を参照してくださいAWS 全般のリファレンス。

で使用するストレージを設定する AWS Transfer Family

このトピックでは、 で使用できるストレージオプションについて説明します AWS Transfer Family。Transfer Family サーバーのストレージとして Amazon S3 または Amazon EFS を使用できます。

目次

- サポートされている Amazon EFS コマンド

Amazon S3 バケットを設定する

AWS Transfer Family は Amazon S3 バケットにアクセスしてユーザーの転送リクエストを処理するため、ファイル転送プロトコル対応サーバーの設定の一環として Amazon S3 バケットを提供する必要があります。既存のバケットを使用するか、新しいバケットを作成することができます。

> **Note**
>
> 同じ AWS リージョンにあるサーバーと Amazon S3 バケットを使用する必要はありませんが、ベストプラクティスとしてそうすることをお勧めします。

ユーザーを設定する際に、各ユーザーに IAM ロールを割り当てます。このロールによって、Amazon S3 バケットへのユーザーのアクセスレベルが決まります。

新しいバケットの作成方法については、Amazon Simple Storage Service ユーザーガイドの「S3 バケットを作成する方法」を参照してください。

> **Note**
>
> Amazon S3 Object オブジェクトロックを使用して、オブジェクトが固定期間または無期限に上書きされることを防止できます。これは、他のサービスの場合と同じように Transfer Family と連携します。オブジェクトが存在して保護されている場合、そのファイルへの書き込みや削除は許可されません。Amazon S3 オブジェクトロックの詳細については、Amazon Simple Storage Service ユーザーガイドの「Amazon S3 オブジェクトロックの使用」を参照してください。

Amazon S3 アクセスポイント

AWS Transfer Family はAmazon S3 の機能である Amazon S3 アクセスポイント をサポートしています。これにより、共有データセットへのきめ細かなアクセスを簡単に管理できます。 Amazon S3 S3 Access Point のエイリアスは、S3 バケット名であればどこでも使用できます。Amazon S3 バケット内の共有データにアクセスするためのさまざまな権限を持つユーザーのために、Amazon S3 に何百ものアクセス ポイントを作成できます。

例えば、アクセスポイントを使用して、同じ共有データセットへのアクセスを3つの異なるチームに許可できます。1つのチームはS3からデータを読み取ることができ、2つ目のチームはS3にデータを書き込むことができ、3つ目のチームはS3からデータを読み取り、書き込み、削除することができます。上記で言及されているような細かいアクセス制御を実装するには、異なるチームに非対称なアクセスを与えるポリシーを含むS3アクセスポイントを作成できます。Transfer Family サーバーで S3 アクセス ポイントを使用すると、何百ものユースケースにまたがる複雑な S3 バケット ポリシーを作成しなくても、きめ細かいアクセス制御を実現できます。Transfer Family サーバーで S3 アクセスポイントを使用する方法の詳細については、「 によるデータアクセスコントロールの強化 AWS Transfer Family 」およびAmazon S3ブログ記事」を参照してください。

> **Note**
>
> AWS Transfer Family は現在、Amazon S3 マルチリージョンアクセスポイントをサポートしていません。

Amazon S3 HeadObject の動作

> **Note**
>
> Transfer Family サーバーを作成または更新すると、Amazon S3 ディレクトリのパフォーマンスを最適化できるため、`HeadObject`呼び出しがなくなります。

Amazon S3 では、バケットとオブジェクトが主要なリソースであり、オブジェクトはバケットに格納されます。Amazon S3 は階層ファイル システムを模倣できますが、一般的なファイル システムとは異なる動作をする場合があります。例えば、ディレクトリは Amazon S3 のファーストクラスの概念ではなく、オブジェクトキーに基づいています。オブジェクトのキーをスラッシュ文字 (/) で分割し、最後の要素をファイル名として扱い、同じプレフィックスを持つファイル名を同じパスの下にグループ化することで、ディレクトリパスを AWS Transfer Family 格納します。`mkdir` または Amazon S3 コンソールを使用して空のディレクトリを作成すると、フォルダーのパスを表すゼロバイトのオブジェクトが作成されます。これらのオブジェクトのキーは末尾のスラッシュで終わります。これらのゼロバイトのオブジェクトについては、Amazon S3 ユーザーガイドの「フォルダーを使用した Amazon S3 コンソールでのオブジェクトの整理」で説明されています。

`ls` コマンドを実行し、一部の結果が Amazon S3 のゼロバイトオブジェクト (これらのオブジェクトにはスラッシュ文字で終わるキーがあります) である場合、Transfer Family はこれらのオブジェクトごとに`HeadObject`リクエストを発行します (詳細についてはHeadObject、Amazon Simple Storage

Service API リファレンスの「」を参照してください)。これにより、Transfer Family で Amazon S3 をストレージとして使用する場合、次の問題が発生する可能性があります。

ファイルの書き込みとリストのみの権限を付与します。

場合によっては、Amazon S3 オブジェクトへの書き込みアクセスのみを提供することもできます。例えば、バケット内のオブジェクトを書き込む (またはアップロードする) アクセスと一覧表示するアクセスを許可し、オブジェクトを読み取る (ダウンロードする) アクセスを許可したい場合があります。ファイル転送クライアントを使用して `ls`および `mkdir` コマンドを実行するには、Amazon S3 `ListObjects`および アクセス`PutObject`許可が必要です。ただし、Transfer Family がファイルの書き込みまたは一覧表示のいずれかを`HeadObject`呼び出す必要がある場合、この呼び出しには アクセス許可が必要なため、呼び出しは失敗し、アクセス拒否 というエラーが表示されます。`GetObject`

> **Note**
>
> Transfer Family サーバーを作成または更新すると、Amazon S3 ディレクトリのパフォーマンスを最適化できるため、`HeadObject`呼び出しがなくなります。

この場合、スラッシュ () で終わるオブジェクトにのみアクセス`GetObject`許可を追加する AWS Identity and Access Management (IAM) ポリシー条件を追加することで、アクセスを許可できます`/`。この条件は、ファイルに対する`GetObject`呼び出し (読み取れないように) を防ぎますが、ユーザーはフォルダを一覧表示してトラバースできます。次のポリシー例では、Amazon S3 バケットへの書き込みおよびリストアクセスのみを提供します。このポリシーを使用するには、 をバケットの名前*DOC-EXAMPLE-BUCKET*に置き換えます。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowListing",
            "Effect": "Allow",
            "Action": "s3:ListBucket",
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET"
        },
        {
            "Sid": "AllowReadWrite",
            "Effect": "Allow",
            "Action": [
```

```
                "s3:PutObject",
                "s3:GetObject",
                "s3:GetObjectVersion"
            ],
            "Resource": [
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
            ]
        },
        {
            "Sid": "DenyIfNotFolder",
            "Effect": "Deny",
            "Action": [
                "s3:GetObject",
                "s3:GetObjectVersion"
            ],
            "NotResource": [
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*/"
            ]
        }
    ]
}
```

> **Note**
>
> このポリシーでは、ユーザーがファイルを追加することはできません。つまり、このポリシーが割り当てられているユーザーは、ファイルを開いてコンテンツを追加したり、ファイルを変更したりすることはできません。さらに、HeadObject ファイルをアップロードする前に呼び出しが必要なユースケースでは、このポリシーは機能しません。

大量のゼロバイトオブジェクトが遅延の問題を引き起こします。

Amazon S3 バケットにこれらのゼロバイトオブジェクトが多数含まれている場合、Transfer Family は多くのHeadObject呼び出しを発行し、処理の遅延が発生する可能性があります。この問題の推奨される解決策は、Optimized Directory を有効にしてレイテンシーを減らすことです。

例えば、ホームディレクトリに移動し、10,000 個のサブディレクトリがあるとします。つまり、Amazon S3 バケットには 10,000 個のフォルダがあります。このシナリオでは、 `ls` (list) コマンドを実行すると、リストオペレーションに 6～8 分かかります。ただし、ディレクトリを最適化する場合、このオペレーションには数秒しかかかりません。このオプションは、サーバーの作成または更

新手順中に、「追加の詳細の設定」画面で設定します。これらの手順については、SFTP、FTPS、または FTP サーバーエンドポイントの設定「」トピックで詳しく説明します。

> **Note**
>
> GUI クライアントは制御できない`ls`コマンドを発行する可能性があるため、可能であればこの設定を有効にすることが重要です。

ディレクトリを最適化しない、または最適化できない場合は、この問題の代替解決策として、ゼロバイトのオブジェクトをすべて削除します。次の点に注意してください。

- 空のディレクトリは存在しなくなります。ディレクトリは、その名前がオブジェクトのキーに含まれている場合にのみ存在します。
- 誰かが `mkdir` を呼び出して、物事を何度も壊すことを防ぐことはできません。ディレクトリの作成を防止するポリシーを作成することで、これを軽減できます。
- 一部のシナリオでは、これらの 0 バイトのオブジェクトが使用されます。たとえば、[/inboxes/customer1000] のような構造があり、受信トレイ ディレクトリは毎日クリーンアップされます。

最後に、可能な解決策の 1 つは、ポリシー条件を通じて表示されるオブジェクトの数を制限して、`HeadObject`呼び出しの数を減らすことです。これを実行可能な解決策とするためには、すべてのサブディレクトリの限られたセットしか表示できない可能性があることを受け入れる必要があります。

Amazon EFS ファイルシステムを設定する

AWS Transfer Family は Amazon Elastic File System (Amazon EFS) にアクセスして、ユーザーの転送リクエストを処理します。したがって、ファイル転送プロトコル対応サーバーの設定の一部として、Amazon EFS ファイルシステムを提供する必要があります。既存のファイルシステムを使用するか、または新しいファイルシステムを作成できます。

次の点に注意してください。

- Transfer Family サーバーと Amazon EFS ファイルシステムを使用する場合、サーバーとファイルシステムは同じ にある必要があります AWS リージョン。
- サーバーとファイルシステムは同じアカウントになくてもかまいません。サーバーとファイルシステムが同じアカウントに属していない場合、ファイルシステムポリシーでユーザーロールに明示的なアクセス許可を付与する必要があります。

複数のアカウントを設定する方法については、「 ユーザーガイド」の「組織内の AWS アカウントの管理AWS Organizations 」を参照してください。

- ユーザーを設定する際に、各ユーザーに IAM ロールを割り当てます。このロールによって、ユーザーの Amazon EFS ファイルシステムへのアクセスレベルが決まります。
- Amazon EFS ファイルシステムのマウントの詳細については、「Amazon EFS ファイルシステムのマウント」を参照してください。

AWS Transfer Family と Amazon EFS の連携の詳細については、「Amazon Elastic File System ユーザーガイド」の「 AWS Transfer Family を使用して Amazon EFS ファイルシステム内のファイルにアクセスする」を参照してください。 Amazon Elastic File System

Amazon EFS ファイルの所有権

Amazon EFS は、Portable Operating System Interface (POSIX) ファイルアクセス許可モデルを使用して、ファイルの所有権を表します。

POSIX では、システム内のユーザーは 3 つの異なるアクセス許可クラスに分類されます。 を使用して Amazon EFS ファイルシステムに保存されているファイルへのアクセスをユーザーに許可する場合は AWS Transfer Family、ユーザーに「POSIX プロファイル」を割り当てる必要があります。このプロファイルは、Amazon EFS ファイルシステム内のファイルおよびディレクトリへのアクセス権を決定するために使用されます。

- User (u): ファイルまたはディレクトリの所有者。通常、ファイルまたはディレクトリの作成者は所有者でもあります。
- Group (g): 共有するファイルおよびディレクトリへの同一のアクセス権を必要とするユーザーのセット。
- Other (o): 所有者およびグループメンバー以外でシステムへのアクセス権を持つ他のすべてのユーザー。このアクセス許可クラスは「Public」とも呼ばれます。

POSIX アクセス許可モデルにおいて、すべてのファイルシステムオブジェクト (ファイル、ディレクトリ、シンボリックリンク、名前付きパイプ、ソケット) は、前述の 3 つのアクセス許可セットに関連付けられています。Amazon EFS オブジェクトには、Unix 形式のモードが関連付けられています。このモード値は、そのオブジェクトに対してアクションを実行するアクセス許可を定義します。

さらに、Unix 形式のシステムでは、ユーザーとグループは Amazon EFS がファイルの所有権を表すために使用する数値 ID にマッピングされます。Amazon EFS オブジェクトは、単一の所有者と単一

のグループによって所有されます。Amazon EFS; は、ユーザーがファイルシステムオブジェクトにアクセスしようとすると、マップされた数値 ID を使用してアクセス許可をチェックします。

Transfer Family用にAmazon EFSユーザーを設定します。

Amazon EFSユーザーを設定する前に、以下のいずれかを実行できます:

- Amazon EFS でユーザーを作成し、そのホームフォルダーを設定できます。詳細については、「Amazon EFS上でTransfer Familyユーザーを設定します。」を参照してください。
- ルートユーザーの追加に問題がない場合は、追加することができます Amazon EFS のルートユーザーの作成。

> **Note**
>
> Transfer Family サーバーは、POSIX アクセス許可を設定するための Amazon EFS アクセス ポイントをサポートしていません。Transfer Family ユーザーの POSIX プロファイル (前のセクションで説明) は、POSIX 権限を設定する機能を提供します。これらのアクセス許可は、UID、GID、およびセカンダリ GID に基づいて、きめ細かなアクセスを実現するためにユーザー レベルで設定されます。

Amazon EFS上でTransfer Familyユーザーを設定します。

Transfer Family は、ユーザーを指定した UID/GID およびディレクトリにマップします。UID/GID/ディレクトリが EFS にまだ存在していない場合は、ユーザーへの転送で割り当てる前に、それらを作成する必要があります。「Amazon Elastic File System ユーザーガイド」の「Network File System (NFS) レベルでユーザー、グループ、パーミッションを操作する」に、Amazon EFS ユーザーを作成する詳細が記載されています。

Transfer Family で Amazon EFS ユーザーを設定する手順

1. 「`PosixProfile`」 フィールドを使用して、Transfer Family のユーザーの EFS UID と GID をマップします。
2. ユーザーがログイン時に特定のフォルダーで開始するようにしたい場合は、「`HomeDirectory`」 フィールドの下に EFS ディレクトリを指定できます。

CloudWatch ルールと Lambda 関数を使用して、プロセスを自動化できます。EFS とやり取りする Lambda 関数の例については、サーバーレスアプリケーション AWS Lambda の「 での Amazon EFS の使用」を参照してください。

さらに、Transfer Family ユーザーの論理ディレクトリを構成できます。詳細については、Amazon EFS の論理ディレクトリを設定する 論理ディレクトリを使用して Transfer Family ディレクトリ構造を簡素化する トピックのセクションを参照してください。

Amazon EFS のルートユーザーの作成

もし組織がユーザーの設定を行うためにSFTP/FTPS経由でのルートユーザーアクセスを有効にすることに問題がなければ、UIDおよびGIDが0 (ルートユーザー) であるユーザーを作成し、そのルートユーザーを使用してフォルダを作成し、残りのユーザーのためにPOSIX IDオーナーを割り当てることができます。このオプションの利点は、Amazon EFS ファイル システムをマウントする必要がないことです。

Amazon EFS サービスマネージドユーザーの追加 で説明されている手順を実行し、ユーザー ID とグループ ID の両方に 0 (ゼロ) を入力します。

サポートされている Amazon EFS コマンド

以下のコマンドは、 AWS Transfer Family Amazon EFS でサポートされています。

- `cd`
- `ls/dir`
- `pwd`
- `put`
- `get`
- `rename`
- chown: root (つまり uid=0 のユーザー) のみがファイルとディレクトリの所有権とアクセス許可を変更できます。
- chmod: ファイルとディレクトリの所有権とアクセス許可を変更できるのは root のみです。
- chgrp: root またはファイルの所有者が、ファイルのグループを自分のセカンダリグループの 1 つに変更できる場合にのみサポートされます。
- `ln -s/symlink`
- `mkdir`

- `rm/delete`
- `rmdir`
- `chmtime`

IAM ポリシーとロールを作成する

このトピックでは、 で使用できるポリシーとロールのタイプについて説明し AWS Transfer Family、ユーザーロールを作成するプロセスについて説明します。また、セッション ポリシーがどのように機能するかについて説明し、ユーザー ロールの例を示します。

AWS Transfer Family は、次のタイプのロールを使用します。

- ユーザーロール – サービスマネージドユーザーが必要な Transfer Family リソースにアクセスできるようにします。 は、Transfer Family ユーザー ARN のコンテキストでこのロールを AWS Transfer Family 引き受けます。
- 「アクセスロール」 — 転送中の Amazon S3 ファイルのみへのアクセスを提供します。インバウンド AS2 転送の場合、アクセスロールは契約の Amazon リソースネーム (ARN) を使用します。アウトバウンド AS2 転送の場合、アクセスロールはコネクタの ARN を使用します。
- 「呼び出しロール」 — Amazon API Gateway でサーバーのカスタム ID プロバイダーとして使用します。Transfer Family は、Transfer Family サーバー ARN のコンテキストでこの役割を引き受けます。
- ログ記録ロール — Amazon へのエントリのログ記録に使用されます CloudWatch。Transfer Family はこのロールを使用して、成功と失敗の詳細をファイル転送に関する情報とともに記録します。Transfer Family は、Transfer Family サーバー ARN のコンテキストでこの役割を引き受けます。アウトバウンドAS2転送では、ログロールはコネクタARNを使用する。
- 「実行ロール」 — Transfer Family ユーザーがワークフローを呼び出して起動できるようにします。Transfer Family は、Transfer Family のワークフローARNのコンテキストにおいて、この役割を引き受ける。

これらのロールに加えて、セッション ポリシーを使用することもできます。セッション ポリシーは、必要に応じてアクセスを制限するために使用されます。これらのポリシーはスタンドアロンであることに注意してください。つまり、これらのポリシーをロールに追加することはできません。代わりに、セッション ポリシーを Transfer Family ユーザーに直接追加します。

> **Note**
>
> サービスで管理される Transfer Family ユーザーを作成する場合は、[ホーム フォルダーに基づいてポリシーを自動生成] を選択できます。これは、ユーザーのアクセスを自分のフォルダーに制限する場合に便利なショートカットです。また、セッション ポリシーの詳細と例をで確認できますセッションポリシーの仕組み。セッションポリシーの詳細については、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。

トピック

ユーザーロールの作成

ユーザーを作成する際には、ユーザーアクセスについて決定すべきことがいくつもあります。具体的には、ユーザーがどのAmazon S3 バケットまたは Amazon EFS ファイルシステムにアクセスできるようにするか、各 Amazon S3 バケットのどの部分をアクセス可能にするか、ユーザーにどのアクセス許可を与えるか (たとえば、PUT または GET) などを決定する必要があります。

アクセスを設定するには、そのアクセス情報を提供するアイデンティティベースの AWS Identity and Access Management (IAM) ポリシーとロールを作成します。このプロセスの一環として、ユーザーに、ファイルオペレーションの送信先または送信元である Amazon S3 バケットまたは Amazon EFS ファイルシステムへのアクセス権を提供します。これを行うには以下のような手順を実行します。手順については、後で詳しい説明があります。

ユーザーロールの作成

1. の IAM ポリシーを作成します AWS Transfer Family。これについては、「の IAM ポリシーを作成するには AWS Transfer Family」で説明しています。
2. IAM ロールを作成して、新しいポリシーをアタッチします。例については、読み書きアクセスポリシーの例を参照してください。
3. AWS Transfer Family と IAM ロールの間に信頼関係を確立します。これについては、「信頼関係を確立するには」で説明しています。

以下の手順では、IAM ポリシーおよびロールを作成する方法について説明します。

の IAM ポリシーを作成するには AWS Transfer Family

1. https://console.aws.amazon.com/iam/ で IAM コンソール を開きます。
2. ナビゲーションペインで ポリシーを選択してから ポリシーの作成を選択します。
3. [Create Policy] (ポリシーの作成) ページで [JSON] タブを選択します。
4. エディタが表示されたら、エディタの内容を IAM ロールにアタッチする IAM ポリシーで置き換えます。

   読み書きアクセス権を付与するか、またはユーザーをホームディレクトリに制限できます。詳細については、「読み書きアクセスポリシーの例」を参照してください。
5. [Review policy] (ポリシーの確認) を選択してポリシーの名前と説明を入力し、[Create policy] (ポリシーの作成) を選択します。

次に、IAM ロールを作成してそれに新しい IAM ポリシーをアタッチします。

の IAM ロールを作成するには AWS Transfer Family

1. ナビゲーションペインで [Roles] (ロール) を選択してから [Create role] (ロールの作成) を選択します。

   [Create role] (ロールの作成) ページで [AWS service] (サービス) が選択されていることを確認します。
2. サービスリストから [Transfer] (転送) を選択してから [Next: Permissions] (次へ: アクセス許可) を選択します。これにより、 AWS Transfer Family と の間に信頼関係が確立されます AWS。
3. [Attach permissions policies] (アクセス許可ポリシーをアタッチする) セクションで、先ほど作成したポリシーを選択して [Next: Tags] (次へ: タグ) を選択します。
4. (オプション) タグのキーと値を入力して [Next: Review] (次へ: レビュー) を選択します。
5. [Review] (レビュー) ページで新しいロールの名前と説明を入力してから [Create role] (ロールの作成) を選択します。

次に、 AWS Transfer Family と の間に信頼関係を確立します AWS。

信頼関係を確立するには

> **Note**
>
> この例では、ArnLike と ArnEquals の両方を使用しています。これらは機能的には同じなので、ポリシーを作成する際にはどちらを使用してもかまいません。Transfer Family ドキュメントでは、条件にワイルドカード文字が含まれる場合は ArnLikeを使用し、完全一致の条件を示す場合は ArnEquals を使用しています。

1. IAM コンソールで、作成したロールを選択します。
2. [Summary] (概要) ページで [Trust relationships] (信頼関係) を選択してから [Edit trust relationship] (信頼関係の編集) を選択します。
3. [Edit Trust Relationship] (信頼関係の編集) エディタでサービスが "transfer.amazonaws.com" であることを確認します。アクセスポリシーは次のように表示されます。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "transfer.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Confused Deputy Problem (混乱した使節の問題) から自分を守るために、aws:SourceAccount および aws:SourceArn の条件キーを使用することをお勧めします。ソースアカウントはサーバーの所有者であり、ソース ARN はユーザーの ARN です。例:

```
"Condition": {
    "StringEquals": {
        "aws:SourceAccount": "account_id"
    },
    "ArnLike": {
```

```
            "aws:SourceArn": "arn:aws:transfer:region:account_id:user/*"
        }
}
```

ユーザー アカウント内のサーバーではなく特定のサーバーに制限したい場合は、ArnLike 条件を使用することもできます。例:

```
"Condition": {
    "ArnLike": {
        "aws:SourceArn": "arn:aws:transfer:region:account-id:user/server-id/*"
    }
}
```

> **Note**
>
> 上記の例では、それぞれの############をあなた自身の情報で置き換えます。

混乱した代理人問題の詳細とその他の例については、サービス間の混乱した代理の防止 を参照してください。

4. [Update Trust Policy] (信頼ポリシーの更新) を選択してアクセスポリシーを更新します。

これで、 がユーザーに代わって AWS サービスを呼び出すことができる IAM AWS Transfer Family ロールが作成されました。ロールには、ユーザーにアクセス権を付与するための IAM ポリシーをアタッチしてあります。「AWS Transfer Family サーバーエンドポイントの開始方法」セクションで、このロールとポリシーを 1 人以上のユーザーに割り当てます。

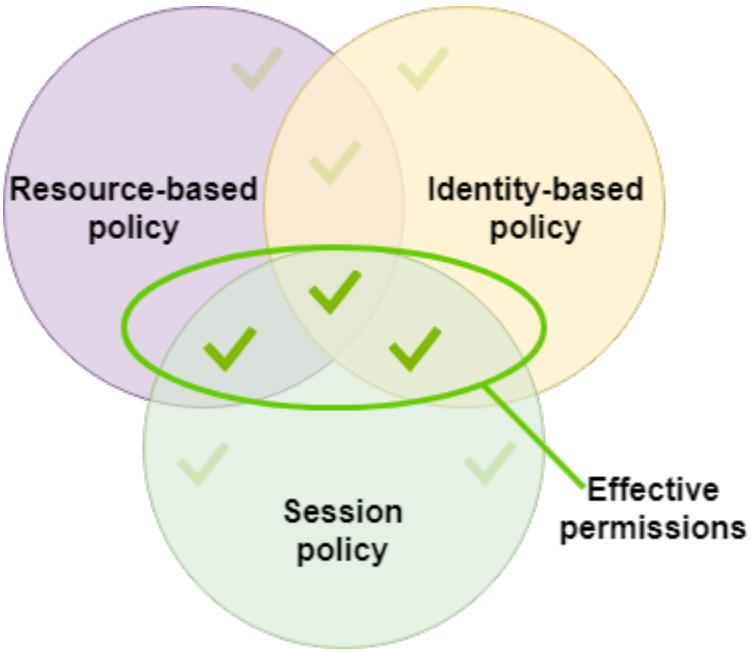
以下の資料も参照してください。

- IAM ロールの詳細については、「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。
- Amazon S3 バケットに関するアイデンティティベースのポリシーの詳細については、Amazon Simple Storage Service ユーザーガイドの「Amazon S3 での Identity and Access Management」を参照してください。
- Amazon EFS リソースのアイデンティティベースのポリシーの詳細については、Amazon Elastic File System ユーザーガイドの「IAM を使用したファイル システム データ アクセスの制御」 を参照してください。

セッションポリシーの仕組み

管理者がロールを作成する場合、そのロールには、複数のユースケースやチームメンバーをカバーする広範な権限が含まれることがよくあります。「管理者がコンソール URL」を構成すると、セッション ポリシーを使用して、結果として得られるセッションのアクセス許可を減らすことができます。たとえば、読み取り/書き込「みアクセス」権を持つロールを作成する場合、ユーザーのアクセスをホーム ディレクトリのみに制限する URL を設定できます。

セッションポリシーは、ロールまたはユーザーの一時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。セッションポリシーは、ユーザーをロックするのに便利で、オブジェクトのプレフィックスにユーザー名が含まれるバケツの一部だけにアクセスできるようにします。次の図は、セッションポリシーの許可が、セッションポリシーとリソースベースポリシーの交差点、およびセッションポリシーと ID ベースポリシーの交差点であることを示しています。



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。

では AWS Transfer Family、セッションポリシーは Amazon S3 との間で転送する場合にのみサポートされます。以下のサンプルポリシーは、ユーザーのアクセスを home ディレクトリのみに限定するセッションポリシーです。次の点に注意してください。

- `GetObjectACL`および`PutObjectACL`ステートメントは、クロスアカウントアクセスを有効にする必要がある場合にのみ必要です。つまり、Transfer Family サーバーは、別のアカウントのバケットにアクセスする必要があります。

- セッションポリシーの最大長は 2048 文字です。詳細については、API リファレンスで CreateUser アクションに関する「ポリシーリクエストパラメータ」を参照してください。
- Amazon S3 バケットが AWS Key Management Service（ AWS KMS) を使用して暗号化されている場合は、ポリシーで追加のアクセス許可を指定する必要があります。詳細については、「Amazon S3 でのデータ暗号化」を参照してください。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowListingOfUserFolder",
            "Action": [
                "s3:ListBucket"
            ],
            "Effect": "Allow",
            "Resource": [
                "arn:aws:s3:::${transfer:HomeBucket}"
            ],
            "Condition": {
                "StringLike": {
                    "s3:prefix": [
                        "${transfer:HomeFolder}/*",
                        "${transfer:HomeFolder}"
                    ]
                }
            }
        },
        {
            "Sid": "HomeDirObjectAccess",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:GetObject",
                "s3:DeleteObject",
                "s3:DeleteObjectVersion",
                "s3:GetObjectVersion",
                "s3:GetObjectACL",
                "s3:PutObjectACL"
            ],
            "Resource": "arn:aws:s3:::${transfer:HomeDirectory}/*"
        }
```

```
    ]
}
```

> **Note**
>
> 前述のポリシーの例では、ユーザーのホームディレクトリがディレクトリであることを示すために、末尾のスラッシュを含むように設定されていることを前提としています。一方、ユーザーの HomeDirectory の末尾にスラッシュを付けない場合、それをポリシーの一部として含める必要があります。

前のポリシー例では、transfer:HomeFolder、transfer:HomeBucket、およびtransfer:HomeDirectory ポリシーパラメータの使用に注目してください。これらのパラメータは、HomeDirectory「」および「」で説明HomeDirectoryされているように、ユーザー用に設定された に設定されますAPI Gateway メソッドを実装する。これらのパラメータには、次のような定義があります。

- transfer:HomeBucket パラメータは HomeDirectory の 1 つ目のコンポーネントに置き換わります。
- transfer:HomeFolder パラメータは HomeDirectory パラメータの残り部分に置き換わります。
- transfer:HomeDirectory パラメータの先頭に付いていた (/) が削除され、Resource ステートメントで S3 Amazon リソースネーム (ARN) の一部としてこのパラメータを使用できるようになりました。

> **Note**
>
> 論理ディレクトリを使用する場合 (つまり、ユーザーの homeDirectoryType が LOGICAL である場合) 、これらのポリシーパラメータ (HomeBucket、HomeDirectory、および HomeFolder) はサポートされません。

たとえば、Transfer Family ユーザー用に設定された HomeDirectory パラメータは /home/bob/amazon/stuff/ です。

- transfer:HomeBucket は /home に設定されます。
- transfer:HomeFolder は /bob/amazon/stuff/ に設定されます。

- `transfer:HomeDirectory` は home/bob/amazon/stuff/ になります。

1 つ目の "Sid" は、/home/bob/amazon/stuff/ から始まるすべてのディレクトリの一覧表示をユーザーに許可します。

2 つ目の "Sid" は、ユーザーの put と get のアクセス権をその同じパスである /home/bob/amazon/stuff/ に限定します。

読み書きアクセスポリシーの例

Amazon S3 バケットへの読み書きアクセス権の付与

次の のポリシー例では、Amazon S3 バケット内のオブジェクトへの読み取り/書き込みアクセス AWS Transfer Family を許可します。

次の点に注意してください。

- DOC-EXAMPLE-BUCKET を Amazon S3 バケットの名前に置き換えます。
- `GetObjectACL`および`PutObjectACL`ステートメントは、クロスアカウントアクセスを有効にする必要がある場合にのみ必要です。つまり、Transfer Family サーバーは、別のアカウントのバケットにアクセスする必要があります。
- `GetObjectVersion` および `DeleteObjectVersion` ステートメントは、アクセスされている Amazon S3 バケットでバージョニングが有効になっている場合にのみ必要です。

> **Note**
>
> バケットのバージョニングを有効にしたことがある場合は、Amazon S3 でバージョニングを停止するだけで、完全にオフにすることはできないため、これらのアクセス許可が必要です。詳細については、「バージョニングが有効なバケット」、「バージョニングが停止されたバケット」を参照してください。

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

```
                "s3:ListBucket",
                "s3:GetBucketLocation"
            ],
            "Effect": "Allow",
            "Resource": [
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET"
            ]
        },
        {
            "Sid": "HomeDirObjectAccess",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:GetObject",
                "s3:DeleteObject",
                "s3:DeleteObjectVersion",
                "s3:GetObjectVersion",
                "s3:GetObjectACL",
                "s3:PutObjectACL"
            ],
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
        }
    ]
}
```

Amazon EFS ファイルシステム内のファイルへのアクセス権を付与する

> **Note**
>
> ポリシーに加えて、POSIX ファイルアクセス許可で適切なアクセス権を付与していることを確認する必要があります。詳細については、Amazon Elastic File System ユーザーガイドの「ネットワークファイルシステム (NFS) レベルでのユーザー、グループ、アクセス許可の操作」を参照してください。

次のポリシー例では、Amazon EFS ファイルシステム内のファイルに root ファイルシステムアクセス権を付与します。

> **Note**
>
> 次の例では、*region* を自分のリージョンに、*account-id* をファイルが存在するアカウントと*file-system-id*に置き換え、 を Amazon Elastic File System (Amazon EFS) の ID に置き換えます。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "RootFileSystemAccess",
            "Effect": "Allow",
            "Action": [
                "elasticfilesystem:ClientRootAccess",
                "elasticfilesystem:ClientMount",
                "elasticfilesystem:ClientWrite"
            ],
            "Resource": "arn:aws:elasticfilesystem:region:account-id:file-system/file-system-id"
        }
    ]
}
```

次のポリシー例では、Amazon EFS ファイルシステム内のファイルにユーザーファイルシステムアクセス権を付与します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "UserFileSystemAccess",
            "Effect": "Allow",
            "Action": [
                "elasticfilesystem:ClientMount",
                "elasticfilesystem:ClientWrite"
            ],
            "Resource": "arn:aws:elasticfilesystem:region:account-id:file-system/file-system-id"
        }
    ]
```

```
}
```

Transfer Family チュートリアル

AWS Transfer Family ユーザーガイドでは、いくつかのユースケースの詳細なチュートリアルを提供します。

- AWS Transfer Family サーバーエンドポイントの開始方法: このチュートリアルでは、SFTP Transfer Family サーバーとサービスマネージドユーザーを作成し、クライアントを使用してファイルを転送する方法について説明します。
- SFTP コネクタのセットアップと使用: このチュートリアルでは、SFTP コネクタをセットアップし、Amazon S3 ストレージと SFTP サーバー間でファイルを転送する方法を説明します。
- カスタム ID プロバイダーとしての Amazon API Gateway メソッドの設定 : このチュートリアルでは、Amazon API Gateway メソッドをセットアップし、それをカスタム ID プロバイダーとして使用して AWS Transfer Family サーバーにファイルをアップロードする方法を示します。
- ファイルを復号するためのマネージドワークフローの設定: このチュートリアルでは、復号化ステップを含むマネージドワークフローをセットアップする方法と、暗号化されたファイルを Amazon S3 バケットにアップロードして復号化されたファイルを表示する方法について説明します。
- AS2 設定のセットアップ: このチュートリアルでは、AS2 Transfer Family サーバーの設定に必要な手順について説明します。証明書のインポート、プロファイルと契約の作成、オプションで AS2 コネクタの作成、設定のテストの手順があります。

トピック

AWS Transfer Family サーバーエンドポイントの開始方法

このチュートリアルを使用して、 AWS Transfer Family (Transfer Family) の使用を開始します。Amazon S3 ストレージを使用してパブリックアクセス可能なエンドポイントを持つ SFTP 対応サーバーを作成し、サービスマネージド認証でユーザーを追加し、Cyberduck でファイルを転送する方法について説明します。

トピック

前提条件

開始する前に、前提条件 の要件を満たしていることを確認してください。この設定の一環として、Amazon Simple Storage Service (Amazon S3) バケットと AWS Identity and Access Management (IAM) ユーザーロールを作成します。

AWS Transfer Family コンソールを使用するために必要なアクセス許可があり、Amazon Simple Storage Service、Amazon Elastic File System、Amazon Route 53 など AWS Certificate Manager、Transfer Family が使用する他の AWS サービスの設定に必要なアクセス許可があります。例えば、Transfer Family AWS を使用してファイルを送受信するユーザーの場合、AmazonS3FullAccess は Amazon S3 バケットをセットアップして使用するアクセス許可を付与します。Amazon S3 バケットを作成するには、このポリシーのアクセス許可の一部が必要です。

Transfer Family コンソールを使用するには、以下が必要です。

- AWSTransferConsoleFullAccess は、SFTP ユーザーに Transfer Family リソースを作成するためのアクセス許可を付与します。
- IAMFullAccess (または具体的には IAM ロールの作成を許可するポリシー) は、Transfer Family が Amazon CloudWatch Logs でサーバーのログ記録ロールを自動的に作成するか、ユーザーがサーバーにログインするためのユーザーロールを作成する場合にのみ必要です。
- VPC サーバータイプを作成および削除するには、ポリシーに ec2:CreateVpcEndpoint および ec2:DeleteVpcEndpoints アクションを追加する必要があります。

> **Note**
>
> AmazonS3FullAccess および IAMFullAccess ポリシー自体は、 の一般的な使用には必要ありません AWS Transfer Family。ここでは、必要なすべての権限が適用されていることを確認するための簡単な方法として紹介しています。さらに、これらは AWS マネージドポリシー

であり、すべての AWS お客様が利用できる標準ポリシーです。これらのポリシーに含まれる個々の権限を確認して、目的に必要な最小限の権限セットを決定できます。

ステップ 1: AWS Transfer Family コンソールにサインインする

Transfer Family にサインインするには

1. にサインイン AWS Management Console し、https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開きます。
2. 「アカウント ID または エイリアス」には、自分の AWS アカウントの ID を入力します。
3. [IAM user name] (IAM ユーザー名) に、Transfer Family 用に作成したユーザーロールの名前を入力します。
4. パスワード には、 AWS アカウントのパスワードを入力します。
5. [Sign in] (サインイン) を選択します。

ステップ 2: SFTP 対応サーバーを作成する

SFTP は Secure Shell (SSH) File Transfer Protocol の略称で、インターネット経由の安全なデータ転送に使用されるネットワークプロトコルです。このプロトコルは、SSH のセキュリティおよび認証機能をすべてサポートします。金融サービス、医療、小売、広告など、さまざまな業界のビジネスパートナー間の機密情報を含むデータを交換するために広く使用されています。

SFTP 対応サーバーを作成するには

1. [Servers] (サーバー) ナビゲーションペインから [Create server] (サーバーの作成) を選択します。
2. [Choose protocols] (プロトコルの選択) で SFTP を選択し、[Next] (次へ) を選択します。
3. [Choose an identity provider] (ID プロバイダーの選択) で[Service managed] (マネージドサービス) を選択してユーザー ID とキーを Transfer Family に選択してから [Next] (次へ) を選択します。
4. [Choose an endpoint] (エンドポイントの選択) で次のように操作します。
    a. [Endpoint type] (エンドポイントタイプ) として [Publicly accessible] (パブリックにアクセス可能な) エンドポイントタイプを選択します。
    b. [Custom hostname] (カスタムホスト名) で [None] (なし) を選択します。

c. [Next] (次へ) を選択します。

5. [Choose a domain] (ドメインの選択) で Amazon S3 を選択します。
6. 「追加の詳細を設定する」の「暗号化アルゴリズムオプション」で、サーバーで使用できる暗号化アルゴリズムを含むセキュリティポリシーを選択します。最新のセキュリティポリシーがデフォルトです。詳細については、「」を参照してくださいAWS Transfer Family サーバーのセキュリティポリシー。

> **Note**
>
> サーバーのマネージドワークフローを追加する場合にのみ、 をログ記録するための新しいロールを作成するを選択します。 CloudWatch サーバーイベントをログに記録するには、IAM ロールを作成する必要はありません。

7. [Review and create] (確認と作成) で [Create server] (サーバーの作成) を選択します。[Servers] (サーバー) ページが表示されます。

新しいサーバーのステータスが [Online] (オンライン) に変わるまでに数分かかることがあります。その時点で、サーバーでファイルオペレーションを実行できますが、まずユーザーを作成する必要があります。ユーザーの作成の詳細については、「」を参照してくださいサーバーエンドポイントのユーザーの管理。

ステップ 3: サービスマネージドユーザーを追加する

SFTP 対応サーバーにユーザーを追加するには

1. サーバーページで、ユーザーを追加するサーバーを選択します。
2. [ユーザーを追加] を選択します。
3. [ユーザー設定] セクションの[ユーザー名] にユーザー名を入力します。このユーザー名は最低 3 文字、最高 100 文字である必要があります。ユーザー名には、a ~ z、A ~ Z、0 ~ 9、アンダースコア「_」、ハイフン「-」、ピリオド「.」、アットマーク (@) を使用できます。ユーザー名はハイフン、ピリオド、アットマークで始めることはできません。
4. アクセス で、 で作成した IAM ロールを選択しますIAM ポリシーとロールを作成する。この IAM ロールには、Amazon S3 バケットにアクセスするためのアクセス許可と、 サービスとの信頼関係を含む IAM ポリシーが含まれています。 AWS Transfer Family で概説されている手順は、適切な信頼関係を確立する方法信頼関係を確立するには を示しています。
5. [Policy] (ポリシー) で [None] (なし) を選択します。

6. ホームディレクトリ で、 を使用して転送するデータを保存する Amazon S3 バケットを選択します AWS Transfer Family。home ディレクトリへのパスを入力します。これは、ユーザーがクライアントを使用してログインしたときに表示されるディレクトリです。

   セッションポリシーを使用できるように、ユーザー名を含むディレクトリパスを使用することをお勧めします。セッションポリシーは、Amazon S3 バケット内のユーザーのアクセスをそのユーザーのhomeディレクトリに制限します。セッションポリシーの使用の詳細については、「」を参照してくださいセッションポリシーの仕組み。

   必要に応じて、このパラメータを空白のままにして Amazon S3 バケットの `root` ディレクトリを使用できます。このオプションを選択した場合は、IAM ロールが ディレクトリへのアクセス`root`を許可していることを確認してください。
7. 制限付きチェックボックスをオンにすると、ユーザーがhomeディレクトリ外のものにアクセスできなくなります。これにより、ユーザーが Amazon S3 バケット名またはフォルダ名を表示できなくなります。
8. SSH パブリックキー には、SSH キーペアのパブリック SSH キー部分を `ssh-rsa <string>`形式で入力します。

   新しいユーザーを追加する前に、 サービスによってキーが検証されている必要があります。SSH キーペアを生成する方法の詳細については、「」を参照してくださいサービス管理ユーザーの SSH キーの生成。
9. (オプション) [Key] (キー) と [Value] (値) にキーバリューペアとして 1 つ以上のタグを入力して [Add tag] (タグの追加) を選択します。
10. [Add] (追加) を選択して、選択したサーバーに新しいユーザーを追加します。

    [Server details] (サーバーの詳細) ページの [Users] (ユーザー) セクションに新しいユーザーが表示されます。

ステップ 4: クライアントを使用してファイルを転送する

クライアントで転送オペレーションを指定して、 AWS Transfer Family サービス経由でファイルを転送します。 は複数のクライアント AWS Transfer Family をサポートしています。詳細については、「クライアントを使用してサーバーエンドポイント経由でファイルを転送する」を参照してください。

このセクションでは、Cyberduck と OpenSSH を使用する手順について説明します。

トピック

- サイバーダックを使用
- OpenSSH を使用する

サイバーダックを使用

Cyberduck AWS Transfer Family を使用してファイルを 経由で転送するには

1. Cyberduck クライアントを開きます。
2. [Open Connection] (接続を開く) を選択します。
3. [Open Connection] (接続を開く) ダイアログボックスで SFTP (SSH File Transfer Protocol) を選択します。
4. [Servers] (サーバー) にサーバーのエンドポイントを入力します。サーバエンドポイントは [Server details] (サーバーの詳細) ページにあります。「SFTP、FTPS、FTP サーバーの詳細を表示する」を参照してください。
5. [Port number] (ポート番号) に SFTP の **22** を入力します。
6. [Username] (ユーザー名) に、サーバーエンドポイントのユーザーの管理 で作成したユーザーの名前を入力します。
7. [SSH Private Key] (SSH プライベートキー) で SSH プライベートキーを選択します。
8. [Connect] (接続) を選択します。
9. ファイル転送を実行します。

   ファイルの場所に応じて、次のいずれかの操作をします。

   - ローカルディレクトリ (転送元) で転送したいファイルを選択して Amazon S3 ディレクトリ (転送先) にドラッグアンドドロップします。
   - Amazon S3 ディレクトリ (転送元) で転送したいファイルを選択してローカルディレクトリ (転送先) にドラッグアンドドロップします。

OpenSSH を使用する

次の手順に従って、OpenSSH を使用してコマンドラインからファイルを転送します。

> **Note**
>
> このクライアントは SFTP 対応サーバーでのみ動作します。

OpenSSH コマンドラインユーティリティ AWS Transfer Family を使用してファイルを 経由で転送するには

1. Linux または Macintosh の場合、コマンドターミナルを開きます。
2. プロンプトで、次のコマンドを入力します: `% sftp -i transfer-key sftp_user@service_endpoint`

   前述のコマンドでは、`sftp_user` がユーザー名、`transfer-key` が SSH プライベートキーです。ここで、 `service_endpoint`は、選択したサーバーの AWS Transfer Family コンソールに表示されるサーバーのエンドポイントです。

   `sftp` プロンプトが表示されます。
3. (オプション) ユーザーのホームディレクトリを表示するには、`sftp` プロンプトで次のコマンドを入力します: `sftp> pwd`
4. 次の行で、次のテキストを入力します: `sftp> cd /mybucket/home/sftp_user`

   この「使用開始」演習では、この Amazon S3 バケットがファイル転送先です。
5. 次の行で、次のコマンドを入力します: `sftp> put filename.txt`

   put コマンドはファイルを Amazon S3 バケットに転送します。

   次のようなメッセージが表示され、ファイル転送が進行中であるか、または完了したことを示します。

   ```
   Uploading filename.txt to /my-bucket/home/sftp_user/filename.txt

   some-file.txt 100% 127 0.1KB/s 00:00
   ```

ファイルを復号するためのマネージドワークフローの設定

このチュートリアルでは、復号化ステップを含む管理ワークフローを設定する方法を示します。このチュートリアルでは、暗号化されたファイルを Amazon S3 バケットにアップロードし、同じバケット内の復号化されたファイルを表示する方法も示しています。

> **Note**
>
> AWS ストレージブログには、Transfer Family Managed ワークフロー、PGP と を使用したファイルの暗号化と復号化を使用してコードを記述せずにファイルを単純に復号する方法 AWS Transfer Familyを説明する投稿があります。

トピック

ステップ 1：実行ロールを設定する

Transfer Family がワークフローの起動に使用できる AWS Identity and Access Management (IAM) 実行ロールを作成します。実行ロールを作成するプロセスについては、ワークフローの IAM ポリシーで説明しています。

> **Note**
>
> 実行ロールを作成する際、信頼関係を確立するには で説明されているように、実行ロールと Transfer Family の間に信頼関係を確立してください。

次の実行ロールポリシーには、このチュートリアルで作成したワークフローを開始するために必要なすべてのアクセス許可が含まれています。このポリシーの例を実行するには、*`user input placeholders`* をユーザー自身の情報に置き換えます。を、暗号化されたファイルをアップロードする Amazon S3 バケットの名前`DOC-EXAMPLE-BUCKET`に置き換えます。

> **Note**
>
> すべてのワークフローに、この例に記載されているすべての権限が必要なわけではありません。特定のワークフローのステップのタイプに基づいてアクセス許可を制限できます。定義済みの各ステップタイプに必要な権限については、事前定義されたステップを使用するで説明しています。カスタムステップに必要な権限については、カスタムステップの IAM 権限で説明しています。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "WorkflowsS3Permissions",
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:GetObjectTagging",
                "s3:GetObjectVersion",
                "s3:PutObject",
                "s3:PutObjectTagging",
                "s3:ListBucket",
                "s3:PutObjectTagging",
                "s3:PutObjectVersionTagging",
                "s3:DeleteObjectVersion",
                "s3:DeleteObject"
            ],
            "Resource": ["arn:aws:s3:::DOC-EXAMPLE-BUCKET/*",
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET"]
            "Condition": {
                "StringEquals": {
                    "s3:RequestObjectTag/Archive": "yes"
                }
            }
        },
        {
            "Sid": "DecryptSecret",
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetSecretValue"
            ],
```

```
            "Resource": "arn:aws:secretsmanager:region:account-id:secret:aws/transfer/
*"
        }
    ]
}
```

ステップ 2: マネージドワークフローを作成する

次に、復号化ステップを含むワークフローを作成する必要があります。

復号化ステップを含むワークフローを作成するには

1. https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開きます。
2. 左のナビゲーションペインで「ワークフロー」を選択し、「ワークフローの作成」を選択します。
3. 次の詳細情報を入力します。
   - **Decrypt workflow example**などの説明を入力します。
   - 「ノミナルステップ」セクションで「ステップを追加」を選択します。
4. 「ステップタイプを選択」で「ファイルを復号化」を選択し、「次へ」を選択します。
5. 「パラメータの設定」ダイアログボックスで、以下を指定します。
   - **decrypt-step**など、わかりやすいステップ名を入力します。ステップ名にはスペースを使用できません。
   - 「復号化されたファイルの宛先」には、Amazon S3 を選択します。
   - 「宛先バケット名」には、ステップ 1 で作成した IAM ポリシーでDOC-EXAMPLE-BUCKETとして指定したのと同じ Amazon S3 バケットを選択します。
   - 「「宛先キープレフィクス」には、復号したファイルを保存するプレフィックス (フォルダ) の名前を、保存先バケットに入力します (例 : **decrypted-files/**) 。

     > **Note**
     >
     > プレフィックスの末尾には必ずスラッシュ (**/**) を追加してください。

   - このチュートリアルでは、「既存を上書き」はオフのままにしておきます。この設定をクリアすると、既存のファイルと同じ名前のファイルを復号しようとしても、ワークフロー処理は停止し、新しいファイルは処理されません。

「次へ」を選択して、次の画面に移動します。

6. ステップの詳細を確認します。すべてが正しい場合は、「ステップを作成」を選択します。
7. ワークフローに必要な復号化ステップは 1 つだけなので、追加のステップを設定する必要はありません。「ワークフローの作成」を選択して新しいワークフローを作成します。

新しいワークフローのワークフロー ID を書き留めます。この ID は次のステップで必要となります。このチュートリアルでは、ワークフロー ID の例として「*w-1234abcd5678efghi*」を使用します。

ステップ 3: サーバーにワークフローを追加してユーザーを作成する

復号化ステップを含むワークフローができたので、そのワークフローを Transfer Family サーバーに関連付ける必要があります。このチュートリアルでは、ワークフローを既存のTransfer Family サーバーに接続する方法を示します。または、ワークフローで使用する新しいサーバーを作成することもできます。

ワークフローをサーバーに接続したら、サーバーに SFTP 接続してワークフローを実行できるユーザーを作成する必要があります。

Transfer Family サーバーを設定してワークフローを実行するには

1. https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開きます。
2. 左ナビゲーションペインで、「サーバ」を選択し、リストからサーバを選択します。このサーバーが SFTP プロトコルをサポートしていることを確認してください。
3. サーバーの詳細ページで下にスクロールして [Additional details] (その他の詳細) セクションで [Edit] (編集) を選択します。
4. 「詳細の編集」ページの「マネージドワークフロー」セクションで、ワークフローを選択し、対応する実行ロールを選択します。
   - 「ファイルをアップロードするワークフロー」では、ステップ 2: マネージドワークフローを作成するで作成したワークフロー (例:**w-1234abcd5678efghi**) を選択します。
   - 「マネージドワークフロー実行ロール」の場合は、ステップ 1 : 実行ロールを設定するで作成した IAM ロールを選択します。
5. ページの最下部までスクロールして、「保存」を選択して変更を保存します。

使用しているサーバーの ID を書き留めます。PGP キーの保存に使用する AWS Secrets Manager シークレットの名前は、サーバー ID に一部基づいています。

ワークフローをトリガーできるユーザーを追加するには

1. https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開きます。
2. 左ナビゲーションペインで、「サーバー」を選択し、復号ワークフローに使用しているサーバーを選択します。
3. サーバーの詳細ページで、「ユーザー」セクションまでスクロールダウンし、「ユーザーの追加」を選択します。
4. 新しいユーザーに、次の詳細情報を入力します。
   - [Username] (ユーザーネーム) に、**`decrypt-user`** と入力します。
   - 「ロール」で、サーバーにアクセスできるユーザーロールを選択します。
   - 「ホームディレクトリ」には、以前に使用した Amazon S3 バケット (例:`DOC-EXAMPLE-BUCKET`) を選択します。
   - 「SSH パブリックキー」の場合は、お持ちのプライベートキーに対応するパブリックキーを貼り付けます。詳細については、「サービス管理ユーザーの SSH キーの生成」を参照してください。
5. 「追加」を選択して新しいユーザーを保存します。

このサーバーのTransfer Familyユーザーの名前を書き留めます。シークレットの一部はユーザーの名前に基づいています。わかりやすくするために、このチュートリアルではサーバーのどのユーザーも使用できるデフォルトのシークレットを使用しています。

ステップ 4：PGP キーペアを作成する

「サポートされている PGP クライアント」のいずれかを使用して、PGP キーペアを生成します。このプロセスについては、PGP キーを生成する に説明されています。

PGP キーペアを生成するには

1. このチュートリアルでは、gpg (GnuPG) バージョン 2.0.22 クライアントを使用して、RSA を暗号化アルゴリズムとして使用する PGP キーペアを生成できます。このクライアントでは、以下のコマンドを実行して E メールアドレスとパスフレーズを指定します。任意の名前またはメールアドレスを使用できます。使用する値は、チュートリアルの後半で入力する必要があるため、忘れないようにしてください。

```
gpg --gen-key
```

> **Note**
>
> GnuPG バージョン 2.3.0 以降を使用している場合は、`gpg --full-gen-key` を実行する必要があります。作成する鍵の種類を求められたら、RSA または ECC を選択します。ただし、ECC を選択した場合は、必ず楕円曲線用に NIST または BrainPool を選択してください。Curve 25519 を「選びません」。

2. 次のコマンドを実行して、プライベートキーをエクスポートします。*user@example.com*は、キーを生成したときに使用したメールアドレスに置き換えます。

```
gpg --output workflow-tutorial-key.pgp --armor --export-secret-key user@example.com
```

このコマンドは秘密鍵を**workflow-tutorial-key.pgp**ファイルにエクスポートします。出力ファイルには任意の名前を付けることができます。プライベートキーファイルは、 AWS Secrets Managerに追加した後で削除することもできます。

ステップ 5: PGP 秘密鍵を AWS Secrets Managerに格納します。

ワークフローがアップロードされたファイルに対して復号化ステップを実行したときにワークフローが秘密鍵を見つけることができるように、秘密鍵を非常に特殊な方法でSecrets Managerに保存する必要があります。

> **Note**
>
> Secrets Manager にシークレットを保存すると、 AWS アカウント に料金が発生します。料金については、「AWS Secrets Manager 料金表」を参照してください。

PGP プライベートキーを Secrets Manager に保存するには

1. にサインイン AWS Management Console し、https://console.aws.amazon.com/secretsmanager/ で AWS Secrets Manager コンソールを開きます。
2. 左側のナビゲーションペインで [サーバー] を選択します。
3. [シークレット]ページで、[新しいシークレットの保存]を選択します。

4. [シークレットタイプの選択] ページの[シークレットタイプ] で[その他のシークレットタイプ] を選択します。
5. [キー/値のペア] セクションで、[キー/値] タブを選択します。
   - キー — **PGPPrivateKey**と入力します。
   - 「値」 — 秘密鍵のテキストを値フィールドに貼り付けます。
6. [行を追加] を選択し、[キー/値のペア] セクションで[キー/値] タブを選択します。
   - キー — **PGPPassphrase**と入力します。
   - 値 — ステップ 4：PGP キーペアを作成するで PGP キーペアを生成したときに使用したパスフレーズを入力します。
7. [次へ] をクリックします。
8. [シークレットの設定] ページで、シークレットの名前と説明を入力します。シークレットは、特定のユーザーに対して作成することも、すべてのユーザーが使用できるシークレットを作成することもできます。サーバー ID が の場合は*s-11112222333344445*、次のようにシークレットに名前を付けます。
   - すべてのユーザーのデフォルトのシークレットを作成するには、シークレットに という名前を付けます**aws/transfer/*s-11112222333344445*/@pgp-default**。
   - 前に作成したユーザー専用のシークレットを作成するには、シークレット**aws/transfer/*s-11112222333344445*/decrypt-user**に名前を付けます。
9. [次へ] を選択し、[ローテーションの設定] ページのデフォルトを受け入れます。次いで、[次へ] を選択します。
10. [レビュー] ページで[ストア] を選択し、シークレットを作成して保存します。

PGP プライベートキーを Secrets Manager に追加する方法の詳細については、「AWS Secrets Manager を使用して PGP キーを保存する」を参照してください。

ステップ 6: ファイルを暗号化する

gpgプログラムを使用して、ワークフローで使用するファイルを暗号化します。以下のコマンドを実行してファイルを暗号化する：

```
gpg -e -r marymajor@example.com --openpgp testfile.txt
```

このコマンドを実行する前に、以下のことに注意する：

- -r引数の場合は、*marymajor@example.com*を PGP キーペアの作成時に使用した電子メールアドレスに置き換えます。
- --openpgp フラグはオプションです。このフラグは、暗号化されたファイルを「OpenPGP RFC4880」標準に準拠させます。
- このコマンドは、**testfile.txt**と同じ場所に**testfile.txt.gpg**という名前のファイルを作成します。

ステップ 7: ワークフローを実行して結果を表示する

ワークフローを実行するには、ステップ 3 で作成したユーザーを使用して Transfer Family サーバーに接続します。そして、「ステップ2.5、保存先パラメーターの設定」で指定したAmazon S3バケットで、復号化されたファイルを見ることができます。

復号化ワークフローを実行するには

1. コマンドターミナルを開きます。
2. 次のコマンドを実行し、*your-endpoint*を実際のエンドポイントに、*transfer-key*をユーザーの SSH 秘密鍵に置き換えます。

   ```
   sftp -i transfer-key decrypt-user@your-endpoint
   ```

   たとえば、秘密鍵が~/.ssh/decrypt-userに保存されていて、エンドポイントがs-11112222333344445.server.transfer.us-east-2.amazonaws.comの場合、コマンドは次のようになります。

   ```
   sftp -i  ~/.ssh/decrypt-user decrypt-user@s-11112222333344445.server.transfer.us-east-2.amazonaws.com
   ```

3. pwd コマンドを実行します。成功すると、このコマンドは以下を返す：

   ```
   Remote working directory: /DOC-EXAMPLE-BUCKET/decrypt-user
   ```

   ディレクトリには、Amazon S3 バケットの名前が反映されます。
4. 次のコマンドを実行してファイルをアップロードし、ワークフローを実行するようにトリガーします。

```
put testfile.txt.gpg
```

5. 復号化されたファイルの保存先として、ワークフローを作成したときにdecrypted-files/フォルダーを指定しました。これで、そのフォルダーに移動して内容を一覧表示できます。

   ```
   cd ../decrypted-files/
   ls
   ```

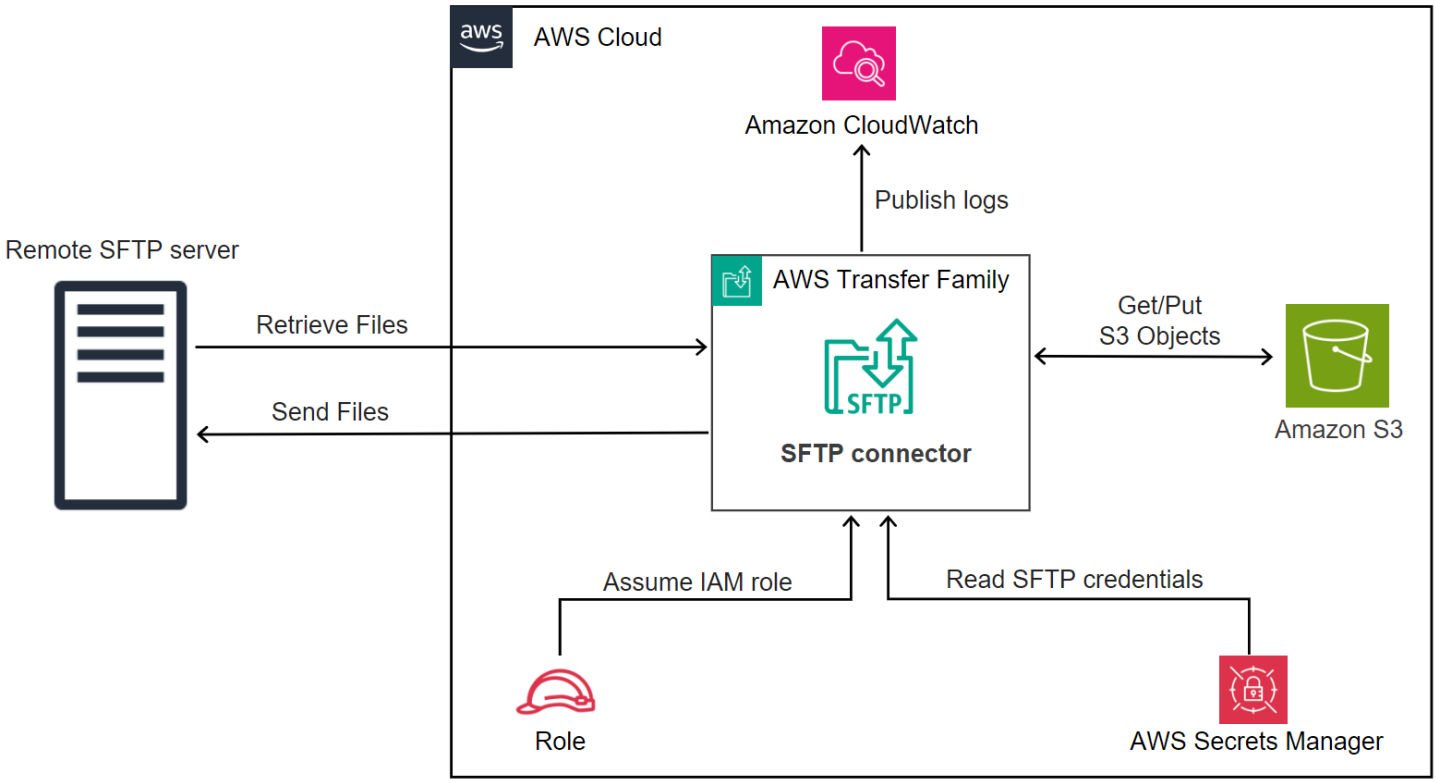
   成功すると、lsコマンドはtestfile.txtファイルを一覧表示します。このファイルをダウンロードして、以前に暗号化した元のファイルと同じかどうかを確認できます。

SFTP コネクタのセットアップと使用

コネクタの目的は、 AWS ストレージとパートナーの SFTP サーバーとの関係を確立することです。Amazon S3 からパートナーが所有する外部の宛先にファイルを送信できます。SFTP コネクタを使用して、パートナーの SFTP サーバーからファイルを取得することもできます。

このチュートリアルでは、SFTP コネクタをセットアップし、Amazon S3 ストレージと SFTP サーバー間でファイルを転送する方法を説明します。

SFTP コネクタは、 から SFTP 認証情報を取得 AWS Secrets Manager してリモート SFTP サーバーに認証し、接続を確立します。コネクタは、リモートサーバーとの間でファイルを送信または取得し、Amazon S3 にファイルを保存します。IAM ロールは、Amazon S3 バケットと Secrets Manager に保存されている認証情報へのアクセスを許可するために使用されます。また、Amazon にログインすることもできます CloudWatch。



次のブログ記事では、SFTP コネクタを使用して MFT ワークフローを構築するためのリファレンスアーキテクチャを提供します。これには、SFTP コネクタを使用してリモート SFTP サーバーに送信する前に PGP を使用してファイルを暗号化する、SFTP AWS Transfer Family コネクタと PGP 暗号化を使用した安全で準拠のマネージドファイル転送の設計が含まれます。

トピック

- ステップ 1: 必要なサポートリソースを作成する
- ステップ 2: SFTP コネクタを作成してテストする
- ステップ 3: SFTP コネクタを使用してファイルを送信および取得する
- リモート SFTP サーバーとして使用する Transfer Family サーバーを作成する手順

ステップ 1: 必要なサポートリソースを作成する

SFTP コネクタを使用して、Amazon S3 と任意のリモート SFTP サーバー間でファイルをコピーできます。このチュートリアルでは、リモート SFTP AWS Transfer Family サーバーとして サーバーを使用しています。次のリソースを作成して設定する必要があります。

- AWS 環境にファイルを保存し、リモート Amazon S3 バケットを作成します。 Amazon S3 バケットを作成する

- Secrets Manager で Amazon S3 ストレージとシークレットにアクセスするための AWS Identity and Access Management ロールを作成します。 必要なアクセス許可を持つ IAM ロールを作成する
- SFTP プロトコルを使用する Transfer Family サーバーと、SFTP コネクタを使用して SFTP サーバーとの間でファイルを転送するサービスマネージドユーザーを作成しますTransfer Family SFTP サーバーとユーザーを作成する。
- SFTP コネクタがリモート SFTP サーバーにログインするために使用する認証情報を保存する AWS Secrets Manager シークレットを作成しますシークレットを作成して保存する AWS Secrets Manager。

Amazon S3 バケットを作成する

Amazon S3 バケットを作成するには

1. https://console.aws.amazon.com/s3/ で AWS Transfer Family コンソールにサインインします。
2. リージョンを選択し、名前を入力します。

   このチュートリアルでは、バケットは にあり**US East (N. Virginia) us-east-1**、名前はです**sftp-server-storage-east**。
3. デフォルトを受け入れ、バケットの作成 を選択します。

Amazon S3 バケットの作成の詳細については、Amazon Simple Storage Service ユーザーガイドの S3 バケットの作成方法」を参照してください。

必要なアクセス許可を持つ IAM ロールを作成する

アクセスロールには、次のアクセス許可を持つポリシーを作成します。

次の例では、Amazon S3 の *DOC-EXAMPLE-BUCKET* および Secrets Manager に保存されている指定されたシークレットにアクセスするために必要なアクセス許可を付与します。 Amazon S3

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
        "Sid": "AllowListingOfUserFolder",
        "Action": [
            "s3:ListBucket",
```

```
            "s3:GetBucketLocation"
        ],
        "Effect": "Allow",
        "Resource": [
            "arn:aws:s3:::DOC-EXAMPLE-BUCKET"
        ]
    },
    {
        "Sid": "HomeDirObjectAccess",
        "Effect": "Allow",
        "Action": [
            "s3:PutObject",
            "s3:GetObject",
            "s3:DeleteObject",
            "s3:DeleteObjectVersion",
            "s3:GetObjectVersion",
            "s3:GetObjectACL",
            "s3:PutObjectACL"
        ],
        "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
    },
    {
        "Sid": "GetConnectorSecretValue",
        "Effect": "Allow",
        "Action": [
            "secretsmanager:GetSecretValue"
        ],
        "Resource": "arn:aws:secretsmanager:region:account-id:secret:aws/
transfer/SecretName-6RandomCharacters"
    }
  ]
}
```

次のように項目を置き換えます。

- *DOC-EXAMPLE-BUCKET* の場合、チュートリアルでは を使用します**s3-storage-east**。
- ***#####*** の場合、チュートリアルでは を使用します**us-east-1**。
- *account-id* には AWS アカウント 、ID を使用します。
- *SecretName-6RandomCharacters* の場合、名前は **using sftp-connector1** です (シークレットには独自の 6 つのランダムな文字があります)。

また、このロールに、ユーザーの転送リクエストを処理するときにコネクタがリソースにアクセスできるようにする信頼関係が含まれていることを確認する必要があります。信頼関係の確立の詳細については、信頼関係を確立するには を参照してください。

> **Note**
>
> チュートリアルで使用しているロールの詳細については、「」を参照してくださいユーザーロールとアクセスロールの組み合わせ。

シークレットを作成して保存する AWS Secrets Manager

SFTP コネクタのユーザー認証情報を保存するには、Secrets Manager にシークレットを保存する必要があります。パスワード、SSH プライベートキー、またはその両方を使用できます。チュートリアルでは、プライベートキーを使用しています。

> **Note**
>
> Secrets Manager にシークレットを保存すると、 AWS アカウント に料金が発生します。料金については、「AWS Secrets Manager 料金表」を参照してください。

シークレットを保存する手順を開始する前に、プライベートキーを取得してフォーマットします。プライベートキーは、リモート SFTP サーバー上のユーザー用に設定されたパブリックキーに対応している必要があります。このチュートリアルでは、プライベートキーは、リモートサーバーとして使用している Transfer Family SFTP サーバーにテストユーザー用に格納されているパブリックキーに対応している必要があります。

これを行うには、次のコマンドを実行します。

```
jq -sR . path-to-private-key-file
```

例えば、プライベートキーファイルが にある場合`~/.ssh/sftp-testuser-privatekey`、コマンドは次のようになります。

```
jq -sR . ~/.ssh/sftp-testuser-privatekey
```

これにより、キーが正しい形式 (改行文字が埋め込まれている) で標準出力に出力されます。このテキストは、次の手順 (ステップ 6) で貼り付ける必要があるため、どこかにコピーします。

SFTPコネクタのユーザー資格情報をSecrets Managerに保存するために

1. にサインイン AWS Management Console し、https://console.aws.amazon.com/secretsmanager/ で AWS Secrets Manager コンソールを開きます。
2. 左側のナビゲーションペインで [サーバー] を選択します。
3. [シークレット]ページで、[新しいシークレットの保存]を選択します。
4. [シークレットタイプの選択] ページの[シークレットタイプ] で[その他のシークレットタイプ] を選択します。
5. [キー/値のペア] セクションで、[キー/値] タブを選択します。
   - キー — **Username**と入力します。
   - value – ユーザーの名前を入力します**sftp-testuser**。
6. キーを入力するには、プレーンテキストタブを使用することをお勧めします。
   a. 行を追加 を選択し、 と入力します**PrivateKey**。
   b. [プレーンテキスト] タブを選択します。フィールドには、次のテキストが含まれるようになりました。

      ```
      {"Username":"sftp-testuser","PrivateKey":""}
      ```

   c. 空の二重引用符 ("") の間にプライベートキー (先ほど保存) のテキストを貼り付けます。

      画面は次のようになります (キーデータはグレー表示されます) 。

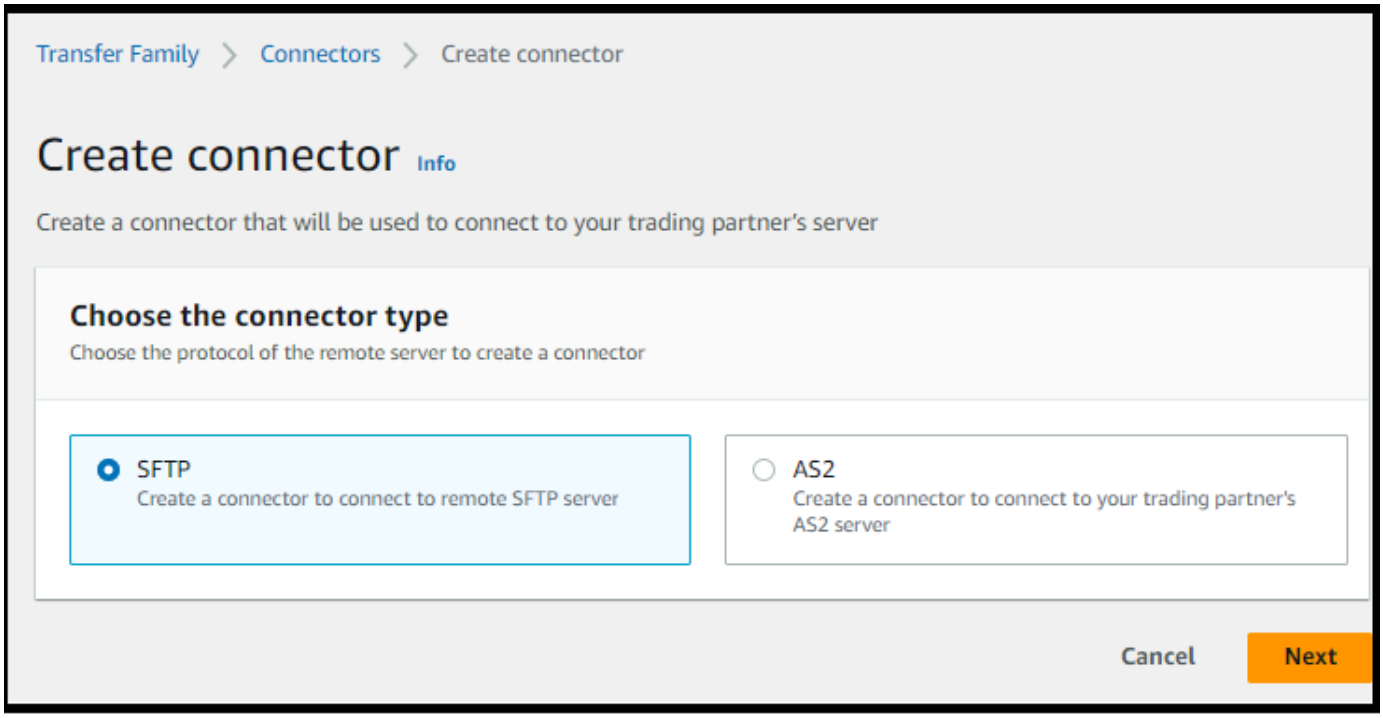
7. [次へ] をクリックします。
8. シークレットの設定ページで、シークレットの名前を入力します。このチュートリアルでは、シークレットに という名前を付けます**aws/transfer/sftp-connector1**。
9. [次へ] を選択し、[ローテーションの設定] ページのデフォルトを受け入れます。次いで、[次へ] を選択します。
10. [レビュー] ページで[ストア] を選択し、シークレットを作成して保存します。

ステップ 2: SFTP コネクタを作成してテストする

このセクションでは、前に作成したすべてのリソースを使用する SFTP コネクタを作成します。詳細については、「SFTP コネクタを設定する」を参照してください。

SFTPコネクタを作成するには

1. https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開きます。
2. 左側のナビゲーションペインで、[コネクタ] を選択し、[コネクタの作成] を選択します。
3. SFTP コネクタを作成するには、コネクタの種類として[SFTP] を選択し、[次へ] を選択します。



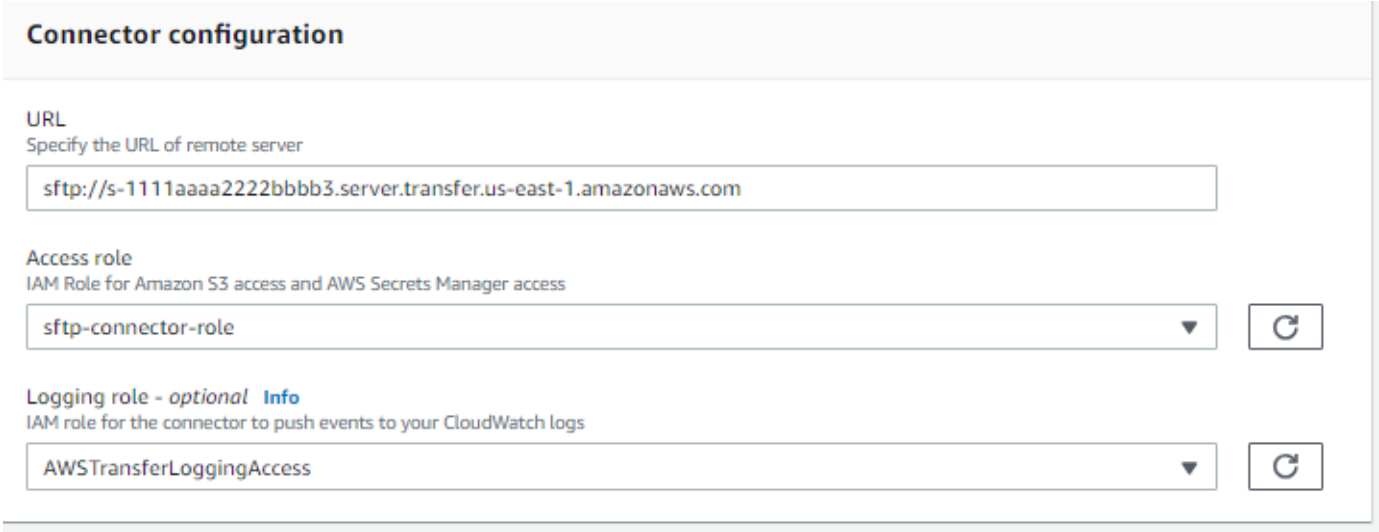
4. [コネクタ構成] セクションで、次の情報を入力します。

   - URL には、リモート SFTP サーバーの URL を入力します。このチュートリアルでは、リモート SFTP サーバーとして使用している Transfer Family サーバーの URL を入力します。

     ```
     sftp://s-1111aaaa2222bbbb3.server.transfer.us-east-1.amazonaws.com
     ```

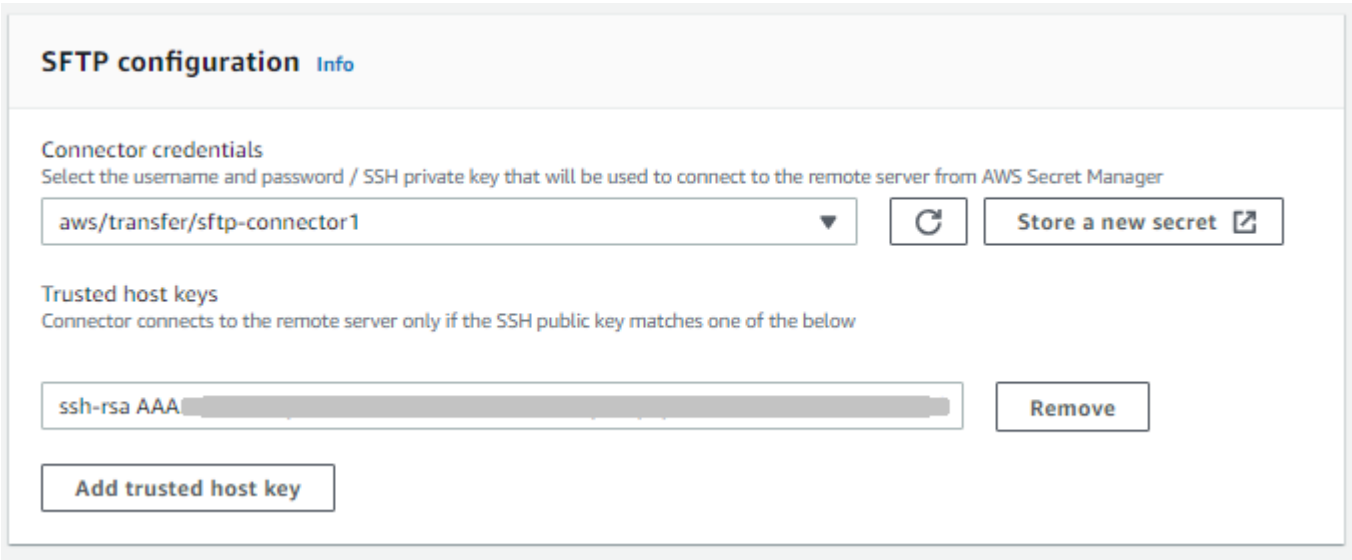
     1111aaaaa2222bbbb3 を Transfer Family サーバー ID に置き換えます。
   - アクセスロール には、前に作成したロール を入力します**sftp-connector-role**。
   - ログ記録ロール で、 を選択します**AWSTransferLoggingAccess**。

     > **Note**
     >
     > AWSTransferLoggingAccess は AWS マネージドポリシーです。このポリシーの詳細については、「」を参照してくださいAWS マネージドポリシー : AWSTransferLoggingAccess。



5. 「SFTPの設定」セクションで、以下の情報を入力する：

   - コネクタ認証情報 で、SFTP 認証情報を含む Secrets Manager リソースの名前を選択します。チュートリアルでは、 を選択します**aws/transfer/sftp-connector1**。
   - 信頼できるホストキー については、ホストキーのパブリック部分を貼り付けます。このキーを取得するには、SFTP サーバーssh-keyscanで を実行します。信頼できるホストキーのフォーマットと保存方法の詳細については、SftpConnectorConfigデータ型のドキュメントを参照してください。



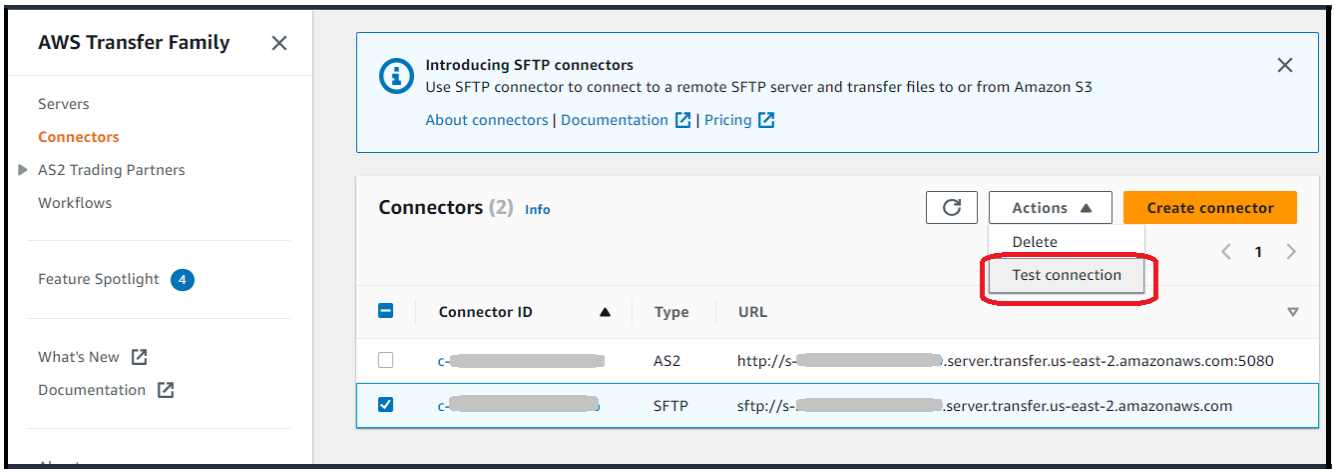
6. すべての設定を確認したら、[コネクタの作成] を選択して SFTP コネクタを作成します。

SFTP コネクタを作成した後、新しいコネクタを使用してファイルを転送する前にテストすることをお勧めします。

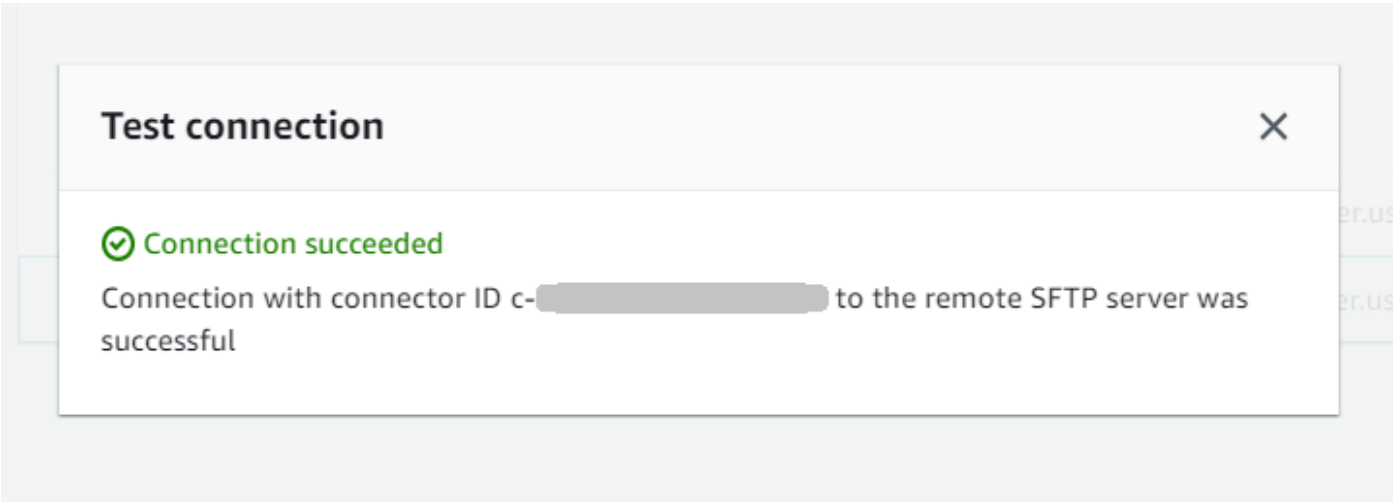
Test a connector using the console

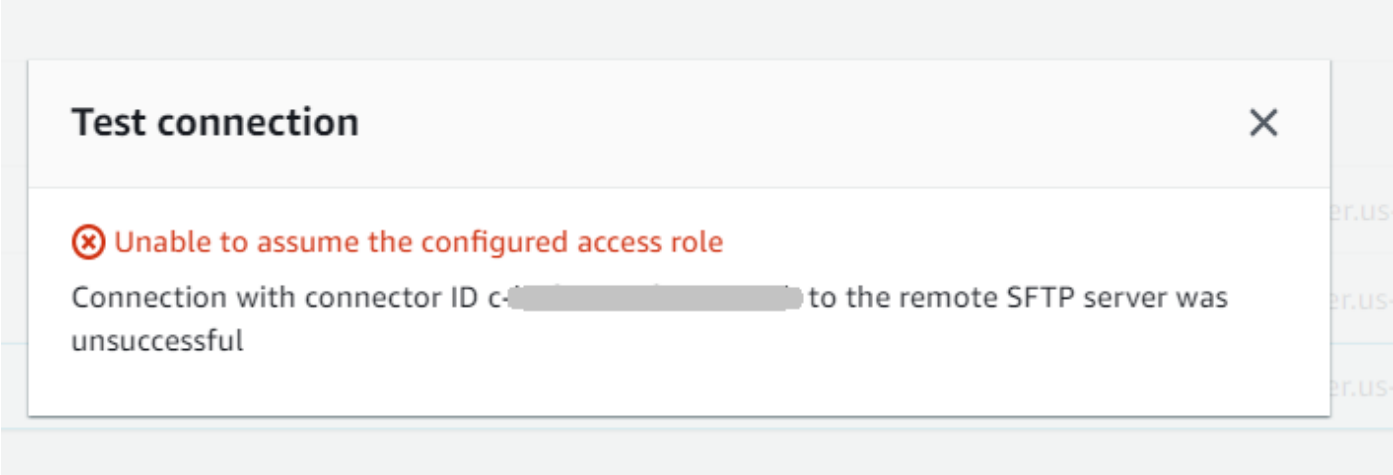
SFTP コネクタをテストするには

1. https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開きます。
2. 左側のナビゲーション ウィンドウで、[コネクタ] を選択し、コネクタを選択します。
3. [アクション] メニューから[テスト接続] を選択します。



システムはテストが合格したかどうかを示すメッセージを返します。テストに失敗した場合、システムは失敗の理由に基づいたエラーメッセージを提供します。





Test a connector using the CLI

を使用してコネクタをテストするには AWS Command Line Interface、コマンドプロンプトで次のコマンドを実行します (*connector-id* を実際のコネクタ ID に置き換えます)。

```
aws transfer test-connection --connector-id c-connector-id
```

テストが成功すると、次の行が返されます。

```
{
   "Status": "OK",
   "StatusMessage": "Connection succeeded"
}
```

テストが失敗すると、次のような説明的なエラーメッセージが表示されます。

```
{
   "Status": "ERROR",
   "StatusMessage": "Unable to assume the configured access role"
}
```

ステップ 3: SFTP コネクタを使用してファイルを送信および取得する

わかりやすくするために、Amazon S3 バケットにファイルがすでにあることを前提としています。

> **Note**
>
> このチュートリアルでは、送信元と送信先の両方のストレージロケーションに Amazon S3 バケットを使用しています。SFTP サーバーが Amazon S3 ストレージを使用しない場合、次のコマンド`sftp-server-storage-east`で表示される場所であれば、パスを SFTP サーバーからアクセス可能なファイルの場所へのパスに置き換えることができます。

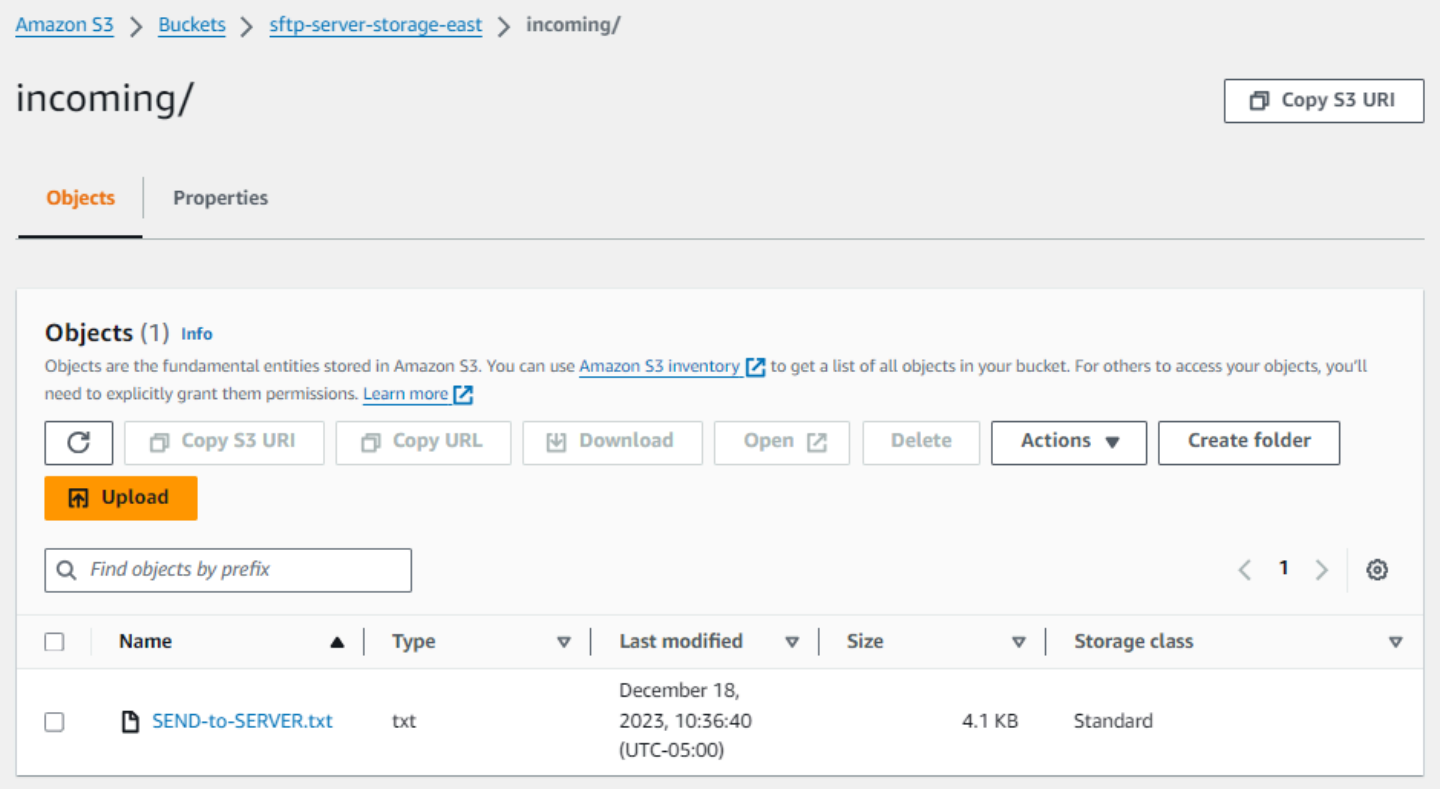
- Amazon S3 ストレージ`SEND-to-SERVER.txt`から という名前のファイルを SFTP サーバーに送信します。 Amazon S3
- という名前のファイルを SFTP サーバー`RETRIEVE-to-S3.txt`から Amazon S3 ストレージに取得します。

> **Note**
>
> 次のコマンドで、*`connector-id`* をコネクタ ID に置き換えます。

まず、Amazon S3 バケットからリモート SFTP サーバーにファイルを送信します。コマンドプロンプトから、次のコマンドを実行します。

```
aws transfer start-file-transfer --connector-id c-connector-id --send-file-paths "/s3-storage-east/SEND-to-SERVER.txt" /
   --remote-directory-path "/sftp-server-storage-east/incoming"
```

これで、`sftp-server-storage-east`バケットは次のようになります。



ファイルが期待どおりに表示されない場合は、 CloudWatch ログを確認してください。

CloudWatch ログを確認するには

1. https://console.aws.amazon.com/cloudwatch/ で Amazon CloudWatch コンソールを開きます。
2. 左側のナビゲーションメニューからロググループを選択します。
3. 検索バーにコネクタ ID を入力して、ログを検索します。
4. 検索から返されるログストリームを選択します。
5. 最新のログエントリを展開します。

成功した場合、ログエントリは次のようになります。

```
{
    "operation": "SEND",
    "timestamp": "2023-12-18T15:26:57.346283Z",
    "connector-id": "connector-id",
    "transfer-id": "transfer-id",
    "file-transfer-id": "transfer-id/file-transfer-id",
    "url": "sftp://server-id.server.transfer.us-east-1.amazonaws.com",
    "file-path": "/s3-storage-east/SEND-to-SERVER.txt",
```

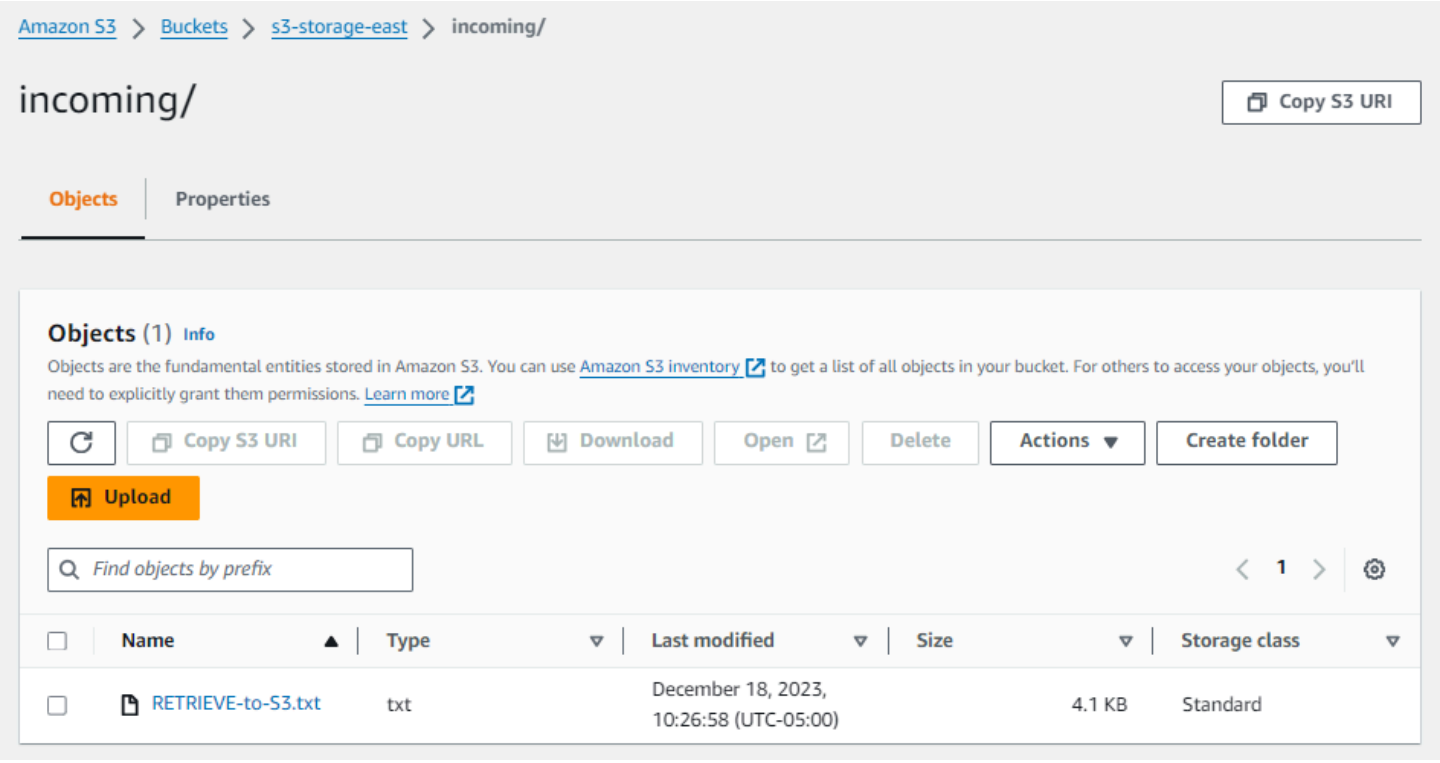
```
    "status-code": "COMPLETED",
    "start-time": "2023-12-18T15:26:56.915864Z",
    "end-time": "2023-12-18T15:26:57.298122Z",
    "account-id": "500655546075",
    "connector-arn": "arn:aws:transfer:us-east-1:500655546075:connector/connector-id",
    "remote-directory-path": "/sftp-server-storage-east/incoming"
}
```

ファイル転送が失敗した場合、ログエントリには問題を指定するエラーメッセージが含まれます。エラーの一般的な原因は、IAM アクセス許可の問題と誤ったファイルパスです。

次に、SFTP サーバーから Amazon S3 バケットにファイルを取得します。コマンドプロンプトから、次のコマンドを実行します。

```
aws transfer start-file-transfer --connector-id c-connector-id --retrieve-file-paths "/
sftp-server-storage-east/RETRIEVE-to-S3.txt" --local-directory-path "/s3-storage-east/
incoming"
```

転送が成功すると、Amazon S3 バケットには、次に示すように、転送されたファイルが含まれます。



成功した場合、ログエントリは次のようになります。

```
{
    "operation": "RETRIEVE",
    "timestamp": "2023-12-18T15:36:40.017800Z",
    "connector-id": "c-connector-id",
    "transfer-id": "transfer-id",
    "file-transfer-id": "transfer-id/file-transfer-id",
    "url": "sftp://s-server-id.server.transfer.us-east-1.amazonaws.com",
    "file-path": "/sftp-server-storage-east/RETRIEVE-to-S3.txt",
    "status-code": "COMPLETED",
    "start-time": "2023-12-18T15:36:39.727626Z",
    "end-time": "2023-12-18T15:36:39.895726Z",
    "account-id": "500655546075",
    "connector-arn": "arn:aws:transfer:us-east-1:500655546075:connector/c-connector-
id",
    "local-directory-path": "/s3-storage-east/incoming"
}
```

リモート SFTP サーバーとして使用する Transfer Family サーバーを作成する手順

以下に、このチュートリアルのリモート SFTP サーバーとして機能する Transfer Family サーバーを作成する手順の概要を示します。次の点に注意してください。

- Transfer Family サーバーを使用してリモート SFTP サーバーを表します。一般的な SFTP コネクタユーザーには、独自のリモート SFTP サーバーがあります。Transfer Family SFTP サーバーとユーザーを作成する を参照してください。
- Transfer Family サーバーを使用しているため、サービスマネージド SFTP ユーザーも使用しています。また、わかりやすくするために、このユーザーが Transfer Family サーバーにアクセスするために必要なアクセス許可と、コネクタを使用するために必要なアクセス許可を結合しました。繰り返しになりますが、ほとんどの SFTP コネクタのユースケースには、Transfer Family サーバーに関連付けられていない個別の SFTP ユーザーがあります。Transfer Family SFTP サーバーとユーザーを作成する を参照してください。
- このチュートリアルでは、リモート SFTP サーバーに Amazon S3 ストレージを使用しているため、あるバケットから別のバケットにファイルを転送できるように**s3-storage-east**、2 つ目のバケット を作成する必要があります。

Transfer Family SFTP サーバーとユーザーを作成する

ほとんどのユーザーは Transfer Family SFTP サーバーとユーザーを作成する必要はありません。これは、ユーザーを持つ SFTP サーバーが既にあるためで、このサーバーを使用してファイルを送受信できます。ただし、このチュートリアルでは、わかりやすくするために、Transfer Family サーバーを使用してリモート SFTP サーバーとして機能します。

SFTP 対応サーバーの作成「」で説明されている手順に従ってサーバーを作成し、ユーザーステップ 3: サービスマネージドユーザーを追加するを追加します。チュートリアルで使用しているユーザーの詳細は次のとおりです。

- サービスマネージドユーザー を作成しますsftp-testuser。
  - ホームディレクトリを に設定する /sftp-server-storage-east/sftp-testuser
  - ユーザーを作成するときは、パブリックキーを保存します。後で Secrets Manager でシークレットを作成するときに、対応するプライベートキーを指定する必要があります。
- ロール: sftp-connector-role。このチュートリアルでは、SFTP ユーザーと SFTP コネクタへのアクセスの両方に同じ IAM ロールを使用しています。組織のコネクタを作成するときは、ユーザーロールとアクセスロールが別々になる場合があります。
- サーバーホストキー: コネクタを作成するときは、サーバーホストキーを使用する必要があります。このキーを取得するには、サーバーssh-keyscanで を実行します。例えば、サーバー ID が でs-1111aaaa2222bbbb3、そのエンドポイントが にある場合us-east-1、次のコマンドはサーバーホストキーを取得します。

  ```
  ssh-keyscan s-1111aaaa2222bbbb3.server.transfer.us-east-1.amazonaws.com
  ```

  このテキストは、ステップ 2: SFTP コネクタを作成してテストする手順で貼り付ける必要があるため、どこかにコピーします。

ユーザーロールとアクセスロールの組み合わせ

このチュートリアルでは、単一の結合ロールを使用しています。このロールは、SFTP ユーザーとコネクタへのアクセスの両方に使用されます。次の例には、チュートリアルのタスクを実行する場合に備えて、このロールの詳細が含まれています。

次の例では、Amazon S3 の 2 つのバケットと、Secrets Manager に保存aws/transfer/sftp-connector1されている という名前のシークレットにアクセスするために必要なアクセス許可を付与します。このチュートリアルでは、このロールの名前は ですsftp-connector-role。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowListingOfUserFolder",
            "Action": [
                "s3:ListBucket",
                "s3:GetBucketLocation"
            ],
            "Effect": "Allow",
            "Resource": [
                "arn:aws:s3:::sftp-server-storage-east",
                "arn:aws:s3:::s3-storage-east"
            ]
        },
        {
            "Sid": "HomeDirObjectAccess",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:GetObject",
                "s3:DeleteObject",
                "s3:DeleteObjectVersion",
                "s3:GetObjectVersion",
                "s3:GetObjectACL",
                "s3:PutObjectACL"
            ],
            "Resource": [
                "arn:aws:s3:::sftp-server-storage-east/*",
                "arn:aws:s3:::s3-storage-east/*"
            ]
        },
        {
            "Sid": "GetConnectorSecretValue",
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetSecretValue"
            ],
            "Resource": "arn:aws:secretsmanager:us-east-1:500655546075:secret:aws/
transfer/sftp-connector1-6RandomCharacters"
        }
    ]
}
```

Transfer Family のロールの作成の詳細については、「」で説明されている手順に従ってロールユーザーロールの作成を作成します。

カスタム ID プロバイダーとしての Amazon API Gateway メソッドの設定

このチュートリアルでは、Amazon API Gateway メソッドをセットアップし、それをカスタム ID プロバイダーとして使用して AWS Transfer Family サーバーにファイルをアップロードする方法を示します。このチュートリアルでは、「基本スタックテンプレート」 やその他の基本的な機能のみを例として取り上げます。

トピック

前提条件

で Transfer Family リソースを作成する前に AWS CloudFormation、ストレージとユーザーロールを作成します。

ストレージを指定してユーザーの役割を作成する

1. 使用しているストレージに応じて、次のドキュメントを参照してください。
   - Amazon S3 バケットを作成するには、Amazon Simple Storage Service ユーザーガイドの「S3 バケットを作成するには？」を参照してください。
   - Amazon EFS ファイルシステムを作成するには、「」を参照してくださいAmazon EFS ファイルシステムを設定する。
2. ユーザーロールを作成するには、IAM ポリシーとロールを作成する を参照してください

AWS CloudFormation 次のセクションで スタックを作成する際に、ストレージとユーザーの役割の詳細を入力します。

ステップ 1: CloudFormation スタックを作成する

提供されたテンプレートから AWS CloudFormation スタックを作成するには

1. https://console.aws.amazon.com/cloudformation で AWS CloudFormation コンソールを開きます。
2. [Create stack] (スタックの作成) を選択し、[With new resources (standard)] (新しいリソースを使用 (標準)) を選択します。
3. [Prerequisite - Prepare template] (前提条件 - テンプレートを準備する) ペインで [Template is ready] (テンプレートの準備完了) を選択します。
4. このリンクで「基本スタックテンプレート」 をコピーして [Amazon S3 URL] フィールドに貼り付けます。
5. [Next] (次へ) をクリックします。
6. スタックの名前を含めてパラメータを指定します。必ず以下のことをしてください。
   - UserName および のデフォルト値を置き換えますUserPassword。
   - にはUserHomeDirectory、前に作成したストレージ (Amazon S3 バケットまたは Amazon EFS ファイルシステム) の詳細を入力します。
   - デフォルトを、前に作成したユーザーロールUserRoleArnに置き換えます。 AWS Identity and Access Management (IAM) ロールには適切なアクセス許可が必要です。IAM ロールとバケットポリシーの例については、「ステップ 6: バケットへのアクセスを制限する」を参照してください。
   - パスワードの代わりにパブリックキーを使用して認証する場合は、UserPublicKey1 フィールドにパブリックキーを入力します。SFTP を使用してサーバーに初めて接続する場合、パスワードの代わりにプライベートキーを指定します。
7. [Next] (次へ) を選択してから [Configure stack options] (スタックオプションの設定) ページでもう一度 [Next] (次へ) をクリックします。
8. 作成しようとするスタックの詳細を確認してから [Create stack] (スタックの作成) を選択します。

> **Note**
>
> ページ下部の [Capabilities] (機能) の下で、 AWS CloudFormation で IAM リソースが作成される可能性があることを承認する必要があります。

ステップ 2：サーバーの API Gateway メソッド設定を確認する

> **Note**
>
> セキュリティを強化するために、ウェブアプリケーションのファイアウォールを設定できます。 AWS WAF はウェブアプリケーションファイアウォールで、これにより Amazon API Gateway に転送される HTTP および HTTPS リクエストのモニタリングが可能です。詳細については、「ウェブアプリケーションファイアウォールを追加する」を参照してください

サーバーの API Gateway メソッドの設定を確認し、デプロイするには

1. API Gateway コンソール (「https://console.aws.amazon.com/apigateway」) を開きます。
2. テンプレートが生成した Transfer Custom Identity Provider の基本テンプレート API を選択します。 AWS CloudFormation
3. [Resource] (リソース) ペインで[GET] を選択してから[Method Request] (メソッドの作成) を選択します。
4. [Actions] (アクション) で [Deploy API] (API のデプロイ) を選択します。[Deplyment stage] (デプロイステージ) で [prod] を選択してから [Deploy] (デプロイ) を選択します。

   API Gateway メソッドが正常にデプロイされると、[Stage Editor] (ステージエディタ) セクションにそのパフォーマンスが表示されます。

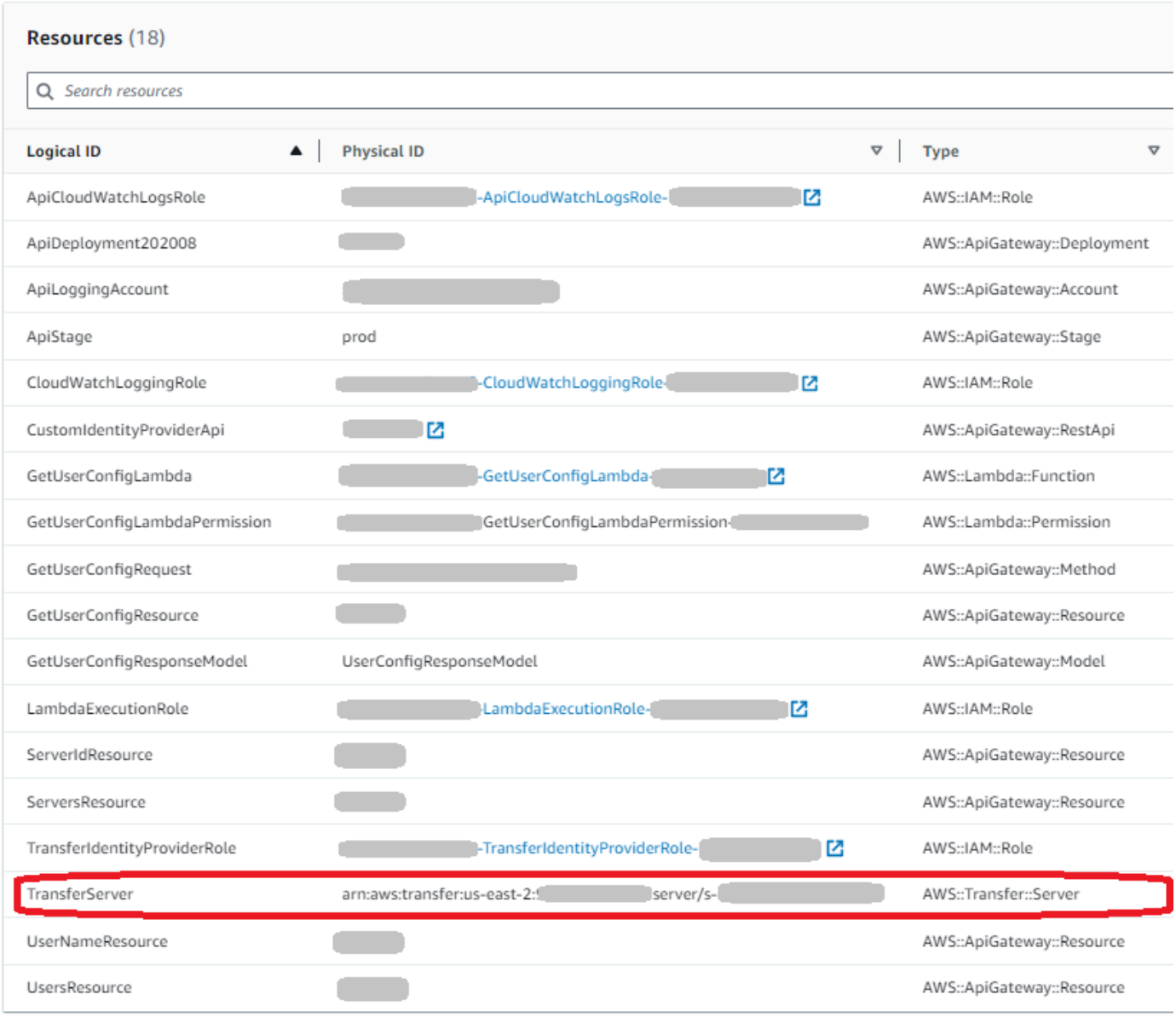
> **Note**
>
> ページの最上部に表示される [Invoke URL] (呼び出し URL) アドレスをコピーします。次のステップで必要になります。

ステップ 3: Transfer Family サーバーの詳細を表示する

テンプレートを使用して AWS CloudFormation スタックを作成すると、Transfer Family サーバーが自動的に作成されます。

Transfer Family サーバーの詳細を表示するには

1. https://console.aws.amazon.com/cloudformation で AWS CloudFormation コンソールを開きます。
2. 作成したスタックを選択します。
3. [リソース] タブを選択します。



サーバー ARN は、TransferServer行の物理 ID 列に表示されます。サーバー ID は ARN に含まれています (例: [s-11112222333344445])。

4. https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開き、サーバーページで新しいサーバーを選択します。

   サーバー ID は、 のTransferServerリソースに表示される ID と一致します AWS CloudFormation。

ステップ 4: ユーザーがサーバーに接続できることを確認する

Transfer Family コンソールを使用して、ユーザーがサーバーに接続できるかどうかをテストするには

1. https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開きます。
2. [Servers] (サーバー) ページで新しいサーバーを選択し、[Actions] (アクション) を選択してから [Test] (テスト) を選択します。
3. サインイン資格情報のテキストを[ユーザー名] フィールドと[パスワード] フィールドに入力します。これらは、 AWS CloudFormation スタックをデプロイしたときに設定した値です。
4. [Server Protocol] (サーバープロトコル) について、[SFTP] を選択し、[Source IP] (送信元 IP) に **`127.0.0.1`** を入力します。
5. [Test] (テスト) を選択します。

   ユーザー認証が成功すると、テストの結果として `StatusCode: 200` HTML レスポンスおよびユーザーのロールとパーミッションの詳細を含む JSON オブジェクトが返されます。例:

```
{
    "Response": "{\"Role\": \"arn:aws:iam::123456789012:role/my-user-role\",
\"HomeDirectory\": \"/${transfer:HomeBucket}/\"}",
    "StatusCode": 200,
    "Message": "",
    "Url": "https://1a2b3c4d5e.execute-api.us-east-2.amazonaws.com/prod/servers/
s-1234abcd5678efgh0/users/myuser/config"
}
```

   テストが失敗した場合は、API に使用しているロールに API Gateway AWS 管理ポリシーのいずれかを追加します。

ステップ 5: SFTP 接続とファイル転送をテストする

SFTP 接続をテストするには

1. Linux または macOS の場合、コマンドターミナルを開きます。
2. 認証にパスワードまたはキーペアのどちらを使用するかに応じて、次のいずれかのコマンドを入力します。

   - パスワードを使用する場合、このコマンドを入力します。

     ```
     sftp -o PubkeyAuthentication=no myuser@server-ID.server.transfer.region-code.amazonaws.com
     ```

     プロンプトが表示されたら、パスワードを入力します。
   - キーペアを使用する場合、このコマンドを入力します。

     ```
     sftp -i private-key-file myuser@server-ID.server.transfer.region-code.amazonaws.com
     ```

   > **Note**
   >
   > これらの `sftp` コマンドには、Transfer Family サーバーが置かれている AWS リージョン のコードを挿入します。たとえば、サーバーが米国東部 (オハイオ) にあるならば「**`us-east-2`**」を入力します。

3. `sftp>` プロンプトで、ディレクトリとファイル (`pwd` と `ls`) をアップロード (`put`)、ダウンロード (`get`)、および表示できることを確認します。

ステップ 6: バケットへのアクセスを制限する

特定の Amazon S3 バケットにアクセスできるユーザーを制限できます。次の例は、CloudFormation スタックおよびユーザー用に選択したポリシーで使用する設定を示しています。

この例では、 AWS CloudFormation スタックに次のパラメータを設定します。

- CreateServer: `true`
- UserHomeDirectory: `/myuser-bucket`
- UserName: `myuser`

- UserPassword: `MySuperSecretPassword`

> **Important**
>
> これはパスワードの例です。API Gateway メソッドを設定する際には、必ず強力なパスワードを入力してください。

- UserPublicKey1: *`your-public-key`*
- UserRoleArn: `arn:aws:iam::`*`role-id`*`:role/myuser-api-gateway-role`

UserPublicKey1 は、パブリック/プライベートキーペアの一部として生成したパブリックキーです。

`role-id` は、作成するユーザーロールに固有です。`myuser-api-gateway-role` にアタッチされるポリシーは次のとおりです。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": "s3:ListBucket",
            "Resource": "arn:aws:s3:::myuser-bucket"
        },
        {
            "Sid": "VisualEditor1",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:GetObjectAcl",
                "s3:GetObject",
                "s3:DeleteObjectVersion",
                "s3:DeleteObject",
                "s3:PutObjectAcl",
                "s3:GetObjectVersion"
            ],
            "Resource": "arn:aws:s3:::myuser-bucket/*"
        }
    ]
}
```

SFTP を使用してサーバに接続するには、プロンプトで次のいずれかのコマンドを入力します。

- パスワードで認証する場合、次のコマンドを実行します。

  ```
  sftp -o PubkeyAuthentication=no myuser@transfer-server-ID.server.transfer.region-id.amazonaws.com
  ```

  プロンプトが表示されたら、パスワードを入力します。
- キーペアで認証する場合、次のコマンドを実行します。

  ```
  sftp -i private-key-file myuser@transfer-server-ID.server.transfer.region-id.amazonaws.com
  ```

> **Note**
>
> これらのsftpコマンドには、Transfer Family サーバー AWS リージョン がある の ID を使用します。たとえば、サーバーが米国東部 (オハイオ) にあるならば「us-east-2」を使用します。

sftp プロンプトで、ホームディレクトリに誘導され、pwd コマンドを実行することによってそれを表示できます。例:

```
sftp> pwd
Remote working directory: /myuser-bucket
```

ユーザーは、ホームディレクトリよりも上位のディレクトリを表示できません。例:

```
sftp> pwd
Remote working directory: /myuser-bucket
sftp> cd ..
sftp> ls
Couldn't read directory: Permission denied
```

Amazon EFS を使用する場合に Lambda を更新する

Transfer Family サーバーのストレージオプションとして Amazon EFS を選択した場合、スタックの lambda 関数を編集する必要があります。

Lambda 関数に Posix プロファイルを追加するには

1. Lambda コンソール (https://console.aws.amazon.com/lambda/) を開きます。
2. 先ほど作成した Lambda 関数を選択します。Lambda 関数の形式は ***stack-name*** - GetUserConfigLambda-***lambda-identifier*** です。*### stack-name* は CloudFormation スタック名、*lambda-identifier* は関数の識別子です。
3. [Code] (コード) タブで index.js を選択して関数のコードを表示します。
4. `response` で `Policy` と `HomeDirectory` の間に次の行を追加します。

   `PosixProfile: {"Uid": uid-value, "Gid": gid-value},`

   uid-value と *gid-value* はそれぞれユーザー ID とグループ ID を表す 0 以上の整数です。

   たとえば、Posix プロファイルを追加すると、レスポンスフィールドは次のようになります。

```
response = {
     Role: 'arn:aws:iam::123456789012:role/api-gateway-transfer-efs-role', // The
user will be authenticated if and only if the Role field is not blank
     Policy: '', // Optional JSON blob to further restrict this user's permissions
     PosixProfile: {"Gid": 65534, "Uid": 65534},
     HomeDirectory: '/fs-fab2c234' // Not required, defaults to '/'
   };
```

AS2 設定のセットアップ

このチュートリアルでは、 で適用性ステートメント 2 (AS2) 設定を設定する方法について説明します AWS Transfer Family。ここで説明する手順を完了すると、AS2 対応サーバーが完成し、サンプルの取引相手からの AS2 メッセージを受け入れる準備が整います。また、サンプルの取引相手に AS2 メッセージを送信するために使用できるコネクタもあります。

> **Note**
>
> サンプルセットアップの一部では、 AWS Command Line Interface () を使用しますAWS CLI。をまだインストールしていない場合は AWS CLI、「 AWS Command Line Interface ユーザーガイド」の「 の最新バージョンのインストールまたは更新 AWS CLI」を参照してください。

1. 自分用と取引相手用の証明書を作成します。使用できる既存の証明書がある場合は、このセクションをスキップできます。

   このプロセスは、「ステップ 1: AS2 の証明書を作成する」で説明されています。

2. AS2 プロトコルを使用する AWS Transfer Family サーバーを作成します。オプションで、サーバーに Elastic IP アドレスを追加してインターネットに接続できるようにすることもできます。

   このプロセスは、「ステップ 2: AS2 プロトコルを使用する Transfer Family サーバーを作成する」で説明されています。

   > **Note**
   >
   > Transfer Family サーバーは、インバウンド転送専用に作成する必要があります。アウトバウンド転送のみを行う場合は、Transfer Family サーバーは必要ありません。

3. ステップ 1 で作成した証明書をインポートします。

   このプロセスは、「ステップ 3: 証明書を Transfer Family 証明書リソースとしてインポートする」で説明されています。

4. 取引相手を設定するには、ローカルプロファイルとパートナープロファイルを作成します。

   このプロセスは、「ステップ 4: 自分と取引相手のプロフィールを作成する」で説明されています。

5. 自分と取引相手との間で契約を作成してください。

   このプロセスは、「ステップ 5: 自分とパートナーとの間で契約を作成する」で説明されています。

   > **Note**
   >
   > 契約を作成する必要があるのはインバウンド転送の場合のみです。アウトバウンド転送のみを行う場合は、契約は必要ありません。

6. 自分と取引相手との間でコネクタを作成してください。

   このプロセスは、「ステップ 6: 自分とパートナーとの間でコネクタを作成する」で説明されています。

> **Note**
>
> コネクタはアウトバウンド転送専用に作成する必要があります。インバウンド転送のみを行う場合は、コネクタは必要ありません。

7. AS2 ファイル交換をテストします。

   このプロセスは、「ステップ 7: Transfer Family を使用して AS2 経由でのファイル交換をテストする」で説明されています。

このステップを完了したら、以下を実行できます。

- Transfer Family `start-file-transfer` AWS Command Line Interface（ AWS CLI) コマンドを使用して、リモート AS2-enabledパートナーサーバーにファイルを送信します。
- 仮想プライベートクラウド (VPC) エンドポイントを介して、ポート 5080 にあるリモート AS2 対応パートナーサーバーからファイルを受信します。

ステップ 1: AS2 の証明書を作成する

AS2 交換の両当事者には X.509 証明書が必要です。これらの証明書は好きな方法で作成できます。このトピックでは、コマンドラインから OpenSSL を使用してルート証明書を作成し、下位証明書に署名する方法について説明します。両当事者はそれぞれ独自の証明書を生成する必要があります。

> **Note**
>
> AS2 証明書のキー長は 2048 ビット以上、最大で 4096 ビットでなければなりません。

パートナーとファイルを転送する場合は、次の点に注意してください。

- 証明書はプロファイルに添付できます。証明書にはパブリックキーまたはプライベートキーが含まれます。
- 取引相手がパブリックキーを送り、自分のパブリックキーを彼らに送ります。
- 取引相手はパブリックキーでメッセージを暗号化し、プライベートキーで署名します。逆に、パートナーのパブリックキーでメッセージを暗号化し、自分のプライベートキーで署名します。

> **Note**
>
> キーを GUI で管理したい場合に使用できるオプションの 1 つは Portecle です。

証明書の例を生成するには

> **Important**
>
> パートナーにプライベートキーを送らないでください。この例では、一方の当事者用に自己署名パブリックキーとプライベートキーのセットを生成します。テスト目的で両方の取引相手として行動する場合は、これらの手順を繰り返して、取引相手ごとに 1 つずつ、合計 2 つのキーセットを生成できます。この場合、ルート認証機関 (CA) を 2 つ生成する必要はありません。

1. 次のコマンドを実行して、2048 ビット長のモジュラスで RSA プライベートキーを生成します。

```
/usr/bin/openssl genrsa -out root-ca-key.pem 2048
```

2. 次のコマンドを実行して、root-ca-key.pem ファイルを使用して自己署名証明書を作成します。

```
/usr/bin/openssl req \
-x509 -new -nodes -sha256 \
-days 1825 \
-subj "/C=US/ST=MA/L=Boston/O=TransferFamilyCustomer/OU=IT-dept/CN=ROOTCA" \
-key root-ca-key.pem \
-out root-ca.pem
```

-subj 引数は、次の値で構成されます。

	名前	説明
C	国コード	組織が所在する国を表す 2 文字のコード。

	名前	説明
ST	州、地域、または都道府県	あなたが所属する組織の所在地の州または県。(この場合、リージョンは AWS リージョンを参照しません。)
L	市区町村	あなたが所属する組織の所在地の市区町村。
O	[Organization name] (組織名)	LLC、Corp などのサフィックスを含む、組織の正式名称です。
OU	部門名	この証明書を扱う組織内の部門。
CN	共通名または完全修飾ドメイン名 (FQDN)	このケースでは、ルート証明書を作成するので、値は ROOTCA です。これらの例では、CN を使用して証明書の目的を説明しています。

3. ローカルプロファイル用の署名キーと暗号化キーを作成します。

```
/usr/bin/openssl genrsa -out signing-key.pem 2048
/usr/bin/openssl genrsa -out encryption-key.pem 2048
```

> **Note**
>
> OpenAS2 など、一部の AS2 対応サーバーでは、署名と暗号化の両方に同じ証明書を使用する必要があります。この場合、両方の目的で同じプライベートキーと証明書をインポートできます。そのためには、前の 2 つのコマンドの代わりに次のコマンドを実行します。
>
> ```
> /usr/bin/openssl genrsa -out signing-and-encryption-key.pem 2048
> ```

4. 次のコマンドを実行して、ルートキーに署名する証明書署名リクエスト (CSR) を作成します。

```
/usr/bin/openssl req -new -key signing-key.pem -subj \
"/C=US/ST=MA/L=Boston/O=TransferFamilyCustomer/OU=IT-dept/CN=Signer" -out signing-
key-csr.pem
```

```
/usr/bin/openssl req -new -key encryption-key.pem -subj \
"/C=US/ST=MA/L=Boston/O=TransferFamilyCustomer/OU=IT-dept/CN=Encrypter" -out
 encryption-key-csr.pem
```

5. 次に、signing-cert.conf ファイルと encryption-cert.conf ファイルを作成する必要があります。

   - テキストエディタを使用して、signing-cert.conf ファイルを作成し、次の内容を記述します。

     ```
     authorityKeyIdentifier=keyid,issuer
     keyUsage = digitalSignature, nonRepudiation
     ```

   - テキストエディタを使用して、encryption-cert.conf ファイルを作成し、次の内容を記述します。

     ```
     authorityKeyIdentifier=keyid,issuer
     keyUsage = dataEncipherment
     ```

6. 最後に、以下のコマンドを実行して署名付き証明書を作成します。

```
/usr/bin/openssl x509 -req -sha256 -CAcreateserial -days 1825 -in signing-key-
csr.pem -out signing-cert.pem -CA \
root-ca.pem -CAkey root-ca-key.pem -extfile signing-cert.conf
```

```
/usr/bin/openssl x509 -req -sha256 -CAcreateserial -days 1825 -in encryption-key-
csr.pem -out encryption-cert.pem \
-CA root-ca.pem -CAkey root-ca-key.pem -extfile encryption-cert.conf
```

ステップ 2: AS2 プロトコルを使用する Transfer Family サーバーを作成する

この手順では、Transfer Family AWS CLIを使用して AS2 対応サーバーを作成する方法について説明します。

> **Note**
>
> サンプルステップの多くは、ファイルからパラメータを読み込むコマンドを使用しています。ファイルを使用してパラメータを読み込む方法について詳しくは、「ファイルからパラメータをロードする方法」を参照してください。

代わりにコンソールを使用する場合は、「Transfer Family コンソールを使用して AS2 サーバーを作成する」を参照してください。

SFTP または FTPS AWS Transfer Family サーバーの作成方法と同様に、 `create-server` AWS CLI コマンドの `--protocols AS2`パラメータを使用して AS2-enabledサーバーを作成します。現在、Transfer Family は VPC エンドポイントタイプと AS2 プロトコルの Amazon S3 ストレージのみをサポートしています。

`create-server` コマンドを使用して Transfer Family 用の AS2 対応サーバーを作成すると、VPC エンドポイントが自動的に作成されます。このエンドポイントは TCP ポート 5080 を公開し、AS2 メッセージを受け付けることができるようにします。

VPC エンドポイントをインターネット向けに公開する場合、Elastic IP アドレスを VPC エンドポイントに関連付けることができます。

この指示書を使用するには、次が必要です。

- VPC の ID (例えば vpc-abcdef01)。
- VPC サブネットの IDs (例: subnet-abcdef01、subnet-subnet-abcdef01、subnet-021345ab) 。
- 取引相手からの TCP ポート 5080 への着信トラフィックを許可するセキュリティグループの 1 つ以上の ID (例えば、sg-1234567890abcdef0 と sg-abcdef01234567890)。
- (オプション) VPC エンドポイントに関連付けたい Elastic IP アドレス。
- 取引相手が VPN 経由で VPC に接続されていない場合は、インターネットゲートウェイが必要です。詳細については、「Amazon VPC ユーザーガイド」の「インターネットゲートウェイを使用してインターネットに接続する」 を参照してください。

AS2 対応サーバーを作成するには

1. 以下のコマンドを実行します。*user input placeholder* を、ユーザー自身の情報に置き換えます。

```
aws transfer create-server --endpoint-type VPC \
--endpoint-details VpcId=vpc-abcdef01,SubnetIds=subnet-abcdef01,subnet-
abcdef01,subnet-
021345ab,SecurityGroupIds=sg-abcdef01234567890,sg-1234567890abcdef0 --protocols AS2
 \
--protocol-details As2Transports=HTTP
```

2. (オプション) VPC エンドポイントをパブリックにすることができます。Elastic IP アドレスは、update-server オペレーションを通じてのみ Transfer Family サーバーにアタッチできます。以下のコマンドはサーバーを停止し、Elastic IP アドレスで更新してから再起動します。

```
aws transfer stop-server --server-id your-server-id
```

```
aws transfer update-server --server-id your-server-id --endpoint-details \
AddressAllocationIds=eipalloc-abcdef01234567890,eipalloc-
1234567890abcdef0,eipalloc-abcd012345ccccccc
```

```
aws transfer start-server --server-id your-server-id
```

start-server コマンドは、サーバーのパブリック IP アドレスを含む DNS レコードを自動的に作成します。取引相手にサーバーへのアクセスを許可するには、次の情報を提供します。この場合、*your-region* は AWS リージョンのことを指します。

s-*your-server-id*.server.transfer.*your-region*.amazonaws.com

取引相手に提供する完全な URL は次のとおりです。

http://s-*your-server-id*.server.transfer.*your-region*.amazonaws.com:5080

3. AS2 対応サーバーにアクセスできるかどうかをテストするには、以下のコマンドを使用します。VPC エンドポイントのプライベート DNS アドレスまたはパブリックエンドポイント (エンドポイントに Elastic IP アドレスを関連付けている場合) からサーバーにアクセスできることを確認します。

サーバーが正しく設定されていれば、接続は成功します。ただし、有効な AS2 メッセージを送信していないため、HTTP ステータスコード 400 (Bad Request) のレスポンスが返されます。

- パブリックエンドポイント (前のステップで Elastic IP アドレスを関連付けた場合) では、サーバー ID とリージョンを代入して以下のコマンドを実行します。

```
curl -vv -X POST http://s-your-server-id.transfer.your-region.amazonaws.com:5080
```

- VPC 内で接続している場合は、次のコマンドを実行して VPC エンドポイントのプライベート DNS 名を検索します。

```
aws transfer describe-server --server-id s-your-server-id
```

`describe-server` コマンドは、VPC エンドポイント ID を `VpcEndpointId` パラメータに返します。次のコマンドを実行するには、この値を使用します。

```
aws ec2 describe-vpc-endpoints --vpc-endpoint-ids vpce-your-vpc-endpoint-id
```

`describe-vpc-endpoints` コマンドは、複数の `DnsName` パラメータを含む `DNSEntries` 配列を返します。以下のコマンドでは、リージョン DNS 名 (アベイラビリティーゾーンを含まないもの) を使用します。

```
curl -vv -X POST http://vpce-your-vpce.vpce-svc-your-vpce-svc.your-region.vpce.amazonaws.com:5080
```

例えば、次のコマンドでは、前のコマンドのプレースホルダーのサンプル値が表示されます。

```
curl -vv -X POST http://vpce-0123456789abcdefg-fghij123.vpce-svc-11111aaaa2222bbbb.us-east-1.vpce.amazonaws.com:5080
```

4. (オプション) ロギングロールを設定します。Transfer Family は、送受信されたメッセージのステータスを構造化された JSON 形式で Amazon CloudWatch ログに記録します。Transfer Family にアカウントの CloudWatch ログへのアクセスを提供するには、サーバーでログ記録ロールを設定する必要があります。

   を信頼する AWS Identity and Access Management (IAM) ロールを作成しtransfer.amazonaws.com、 AWSTransferLoggingAccessマネージドポリシーをアタッチします。詳細については、「IAM ポリシーとロールを作成する」を参照してください。先ほ

ど作成した IAM ロールの Amazon リソースネーム (ARN) を書き留め、次の update-server コマンドを実行して、サーバーに関連付けます。

```
aws transfer update-server --server-id your-server-id --logging-role
 arn:aws:iam::your-account-id:role/logging-role-name
```

> **Note**
>
> ログ記録ロールはオプションですが、メッセージのステータスを確認したり、設定上の問題をトラブルシューティングしたりできるように設定することを強くお勧めします。

ステップ 3: 証明書を Transfer Family 証明書リソースとしてインポートする

この手順では、 AWS CLIを使用して証明書をインポートする方法について説明します。代わりに Transfer Family コンソールを使用する場合は、the section called “AS2 証明書のインポート” を参照してください。

ステップ 1 で作成した署名用証明書と暗号化証明書をインポートするには、次の import-certificate コマンドを実行します。暗号化と署名に同じ証明書を使用している場合は、同じ証明書を 2 回インポートします (1 回目は SIGNING 用、もう 1 回は ENCRYPTION 用)。

```
aws transfer import-certificate --usage SIGNING --certificate file://signing-cert.pem \
            --private-key file://signing-key.pem --certificate-chain file://root-ca.pem
```

このコマンドは署名の CertificateId を返します。次のセクションでは、この証明書 ID を *my-signing-cert-id* と呼びます。

```
aws transfer import-certificate --usage ENCRYPTION --certificate file://encryption-
cert.pem \
            --private-key file://encryption-key.pem --certificate-chain file://root-
ca.pem
```

このコマンドは暗号化 CertificateId を返します。次のセクションでは、この証明書 ID を *my-encrypt-cert-id* と呼びます。

次に、次のコマンドを実行して、パートナーの暗号化証明書と署名用証明書をインポートします。

```
aws transfer import-certificate --usage ENCRYPTION --certificate file://partner-
encryption-cert.pem \
--certificate-chain file://partner-root-ca.pem
```

このコマンドはパートナーの暗号化 CertificateId を返します。次のセクションでは、この証明書 ID を *partner-encrypt-cert-id* と呼びます。

```
aws transfer import-certificate --usage SIGNING --certificate file://partner-signing-
cert.pem \
--certificate-chain file://partner-root-ca.pem
```

このコマンドはパートナーの署名 CertificateId を返します。次のセクションでは、この証明書 ID を *partner-signing-cert-id* と呼びます。

ステップ 4: 自分と取引相手のプロフィールを作成する

この手順では、 を使用して AS2 プロファイルを作成する方法について説明します AWS CLI。代わりに Transfer Family コンソールを使用する場合は、the section called “AS2 プロファイルの作成” を参照してください。

次のコマンドを実行してローカル AS2 プロファイルを作成します。このコマンドは、パブリックキーとプライベートキーを含む証明書を参照します。

```
aws transfer create-profile --as2-id MYCORP --profile-type LOCAL --certificate-ids \
my-signing-cert-id my-encrypt-cert-id
```

このコマンドはプロファイル ID を返します。次のセクションでは、この ID を *my-profile-id* と呼びます。

次のコマンドを実行して、パートナープロファイルを作成します。このコマンドは、パートナーのパブリックキー証明書のみを使用します。このコマンドを使用するには、*user input placeholders* を自身の情報 (パートナーの AS2 名や証明書 ID など) に置き換えます。

```
aws transfer create-profile --as2-id PARTNER-COMPANY --profile-type PARTNER --
certificate-ids \
partner-signing-cert-id partner-encrypt-cert-id
```

このコマンドはパートナーのプロファイル ID を返します。次のセクションでは、この ID を *partner-profile-id* と呼びます。

> **Note**
>
> 前のコマンドでは、*MYCORP* を自分の組織の名前に置き換え、*PARTNER-COMPANY* を取引相手の組織の名前に置き換えます。

ステップ 5: 自分とパートナーとの間で契約を作成する

ここでは、 AWS CLIを使用して AS2 契約を作成する方法を説明します。代わりに Transfer Family コンソールを使用する場合は、the section called “AS2 契約の作成” を参照してください。

契約には、2 つのプロファイル (ローカルとパートナー)、それらの証明書、および両者間のインバウンド AS2 転送を可能にするサーバー設定がまとめられています。次のコマンドを実行すると、項目を一覧表示できます。

```
aws transfer list-profiles --profile-type LOCAL
aws transfer list-profiles --profile-type PARTNER
aws transfer list-servers
```

このステップには、バケットへの、およびバケットからの読み取り/書き込みアクセス権限を持つ Amazon S3 バケットと IAM ロールが必要です。このロールを作成する手順は、Transfer Family の SFTP、FTP、FTPS プロトコルの場合と同じで、IAM ポリシーとロールを作成する に記載されています。

契約を作成するには、次のアイテムが必要です。

- Amazon S3 バケット名 (指定されている場合はオブジェクトプレフィックスも)
- バケットへのアクセス権を持つ IAM ロールの ARN
- Transfer Family サーバー ID
- プロファイル ID とパートナーのプロファイル ID

次のコマンドを実行して契約を作成します。

```
aws transfer create-agreement --description "ExampleAgreementName" --server-id your-
server-id \
--local-profile-id your-profile-id --partner-profile-id your-partner-profile-id --base-
directory /DOC-EXAMPLE-DESTINATION-BUCKET/AS2-inbox \
--access-role arn:aws:iam::111111111111:role/TransferAS2AccessRole
```

成功した場合、このコマンドは契約の ID を返します。その後、次のコマンドを使用して契約の詳細を表示できます。

```
aws transfer describe-agreement --agreement-id agreement-id --server-id your-server-id
```

ステップ 6: 自分とパートナーとの間でコネクタを作成する

ここでは、 AWS CLIを使用して AS2 コネクタを作成する方法を説明します。代わりに Transfer Family コンソールを使用する場合は、the section called “AS2 コネクタを設定する” を参照してください。

StartFileTransfer API オペレーションを使用して、Amazon S3 に保存されているファイルをコネクタを使用して取引相手の AS2 エンドポイントに送信できます。次のコマンドを実行すると、先ほど作成したプロファイルを見つけることができます。

```
aws transfer list-profiles
```

コネクタを作成するときは、パートナーの AS2 サーバー URL を指定する必要があります。次の テキストを testAS2Config.json という名前のファイルにコピーします。

```
{
"Compression": "ZLIB",
"EncryptionAlgorithm": "AES256_CBC",
"LocalProfileId": "your-profile-id",
"MdnResponse": "SYNC",
"MdnSigningAlgorithm": "DEFAULT",
"MessageSubject": "Your Message Subject",
"PartnerProfileId": "partner-profile-id",
"SigningAlgorithm": "SHA256"
}
```

> **Note**
>
> ではEncryptionAlgorithm、アルゴリズムは弱い暗号化DES_EDE3_CBCアルゴリズムであるため、それを必要とするレガシークライアントをサポートする必要がある場合を除き、アルゴリズムを指定しないでください。

その後、次のコマンドを実行してコネクタを作成します。

```
aws transfer create-connector --url "http://partner-as2-server-url" \
--access-role your-IAM-role-for-bucket-access \
--logging-role arn:aws:iam::your-account-id:role/service-role/AWSTransferLoggingAccess
 \
--as2-config file:///path/to/testAS2Config.json
```

ステップ 7: Transfer Family を使用して AS2 経由でのファイル交換をテストする

取引相手からファイルを受け取ります

パブリック Elastic IP アドレスを VPC エンドポイントに関連付けた場合、Transfer Family はパブリック IP アドレスを含む DNS 名を自動的に作成します。サブドメインは AWS Transfer Family 、サーバー ID (形式) ですs-1234567890abcdef0。以下の形式で、取引相手にサーバー URL を提供します。

```
http://s-1234567890abcdef0.server.transfer.us-east-1.amazonaws.com:5080
```

パブリック Elastic IP アドレスを VPC エンドポイントに関連付けていない場合は、ポート 5080 で取引相手からの HTTP POST 経由で AS2 メッセージを受け付ける VPC エンドポイントのホスト名を検索します。VPC エンドポイントの詳細情報を取得するには、次のコマンドを使用します。

```
aws transfer describe-server --server-id s-1234567890abcdef0
```

例えば、前のコマンドが VPC エンドポイント ID を vpce-1234abcd5678efghi と返すとします。次に、次のコマンドを使用して DNS 名を取得します。

```
aws ec2 describe-vpc-endpoints --vpc-endpoint-ids vpce-1234abcd5678efghi
```

このコマンドは、次のコマンドを実行するために必要な VPC エンドポイントのすべての詳細を返します。

DNS 名は DnsEntries 配列に一覧表示されます。VPC エンドポイントにアクセスするには、取引相手が VPC 内にいる必要があります (例えば、 AWS PrivateLink または VPN 経由)。以下の形式で、パートナーに VPC エンドポイント URL を提供します。

```
http://vpce-your-vpce-id.vpce-svc-your-vpce-svc-id.your-region.vpce.amazonaws.com:5080
```

例えば、次の URL では、前のコマンドのプレースホルダーのサンプル値が表示されます。

```
http://vpce-0123456789abcdefg-fghij123.vpce-svc-11111aaaa2222bbbb.us-
east-1.vpce.amazonaws.com:5080
```

この例では、成功した転送は、ステップ 5: 自分とパートナーとの間で契約を作成する で指定した base-directory パラメータで指定された場所に保存されます。myfile1.txt および myfile2.txt という名前のファイルを正常に受信すると、ファイルは */path-defined-in-the-agreement*/processed/*original_filename.messageId.original_extension* として保存されます。ここでは、ファイルは /DOC-EXAMPLE-DESTINATION-BUCKET/AS2-inbox/processed/myfile1.*messageId*.txt と /DOC-EXAMPLE-DESTINATION-BUCKET/AS2-inbox/processed/myfile2.*messageId*.txt として保存されます。

Transfer Family サーバーの作成時にログ記録ロールを設定した場合は、 CloudWatch ログで AS2 メッセージのステータスを確認することもできます。

取引相手にファイルを送信します

Transfer Family を使用して AS2 メッセージを送信するには、次の start-file-transfer AWS Command Line Interface （ AWS CLI) コマンドに示すように、コネクタ ID とファイルへのパスを参照します。

```
aws transfer start-file-transfer --connector-id c-1234567890abcdef0 \
--send-file-paths "/DOC-EXAMPLE-SOURCE-BUCKET/myfile1.txt" "/DOC-EXAMPLE-SOURCE-BUCKET/
myfile2.txt"
```

コネクタの詳細情報を取得するには、次のコマンドを実行しする：

```
aws transfer list-connectors
```

list-connectors コマンドは、コネクタのコネクタ ID、URL、Amazon リソースネーム (ARN) を返します。

特定のコネクタのプロパティを返すには、使用する ID を指定して以下のコマンドを実行します。

```
aws transfer describe-connector --connector-id your-connector-id
```

describe-connector コマンドは、URL、ロール、プロファイル、メッセージ処理通知 (MDN)、タグ、モニタリングメトリックなど、コネクタのすべてのプロパティを返します。

JSON と MDN ファイルを表示すると、パートナーがファイルを正常に受信したことを確認できます。これらのファイルには、ファイル名と場所 で説明されている規則に従って名前が付けられます。コネクタの作成時にログ記録ロールを設定した場合は、 CloudWatch ログで AS2 メッセージのステータスを確認することもできます。

SFTP、FTPS、または FTP サーバーエンドポイントの設定

このトピックでは、1 つ以上の SFTP、FTPS、および FTP プロトコルを使用する AWS Transfer Family サーバーエンドポイントの作成と使用について詳しく説明します。

トピック

ID プロバイダーオプション

AWS Transfer Family には、ユーザーを認証および管理するための方法がいくつか用意されています。次の表は、Transfer Family で使用できる ID プロバイダを比較したものです。

アクション	AWS Transfer Family サービスマネージド	AWS Managed Microsoft AD	Amazon API Gateway	AWS Lambda
サポートされるプロトコル	SFTP	SFTP、FTPS、FTP	SFTP、FTPS、FTP	SFTP、FTPS、FTP
SAML ベースの認証	はい	いいえ	はい	はい
パスワード認証	いいえ	はい	はい	はい
AWS Identity and Access Management (IAM) と POSIX	はい	はい	はい	はい

アクション	AWS Transfer Family サービスマネージド	AWS Managed Microsoft AD	Amazon API Gateway	AWS Lambda
論理ホームディレクトリ	はい	はい	はい	はい
パラメータ化されたアクセス (ユーザー名ベース)	はい	はい	はい	はい
アドホックアクセス構造	はい	いいえ	はい	はい
AWS WAF	いいえ	いいえ	はい	いいえ

注記:

- IAM は Amazon S3 バッキングストレージへのアクセスを制御するために使用され、POSIX は Amazon EFS に使用されます。
- アドホックとは、実行時にユーザープロファイルを送信する機能のことです。たとえば、ユーザー名を変数として渡すことで、ユーザーをホームディレクトリに配置できます。
- の詳細については AWS WAF、「」を参照してくださいウェブアプリケーションファイアウォールを追加する。
- Lambda関数をMicrosoft Azure ADと統合してTransfer Familyのアイデンティティプロバイダーとして使用する方法について説明したブログ記事があります。詳細については、「Azure Active Directory AWS Transfer Family による への認証」および AWS Lambda「」を参照してください。
- カスタム ID プロバイダーを使用する Transfer Family サーバーを迅速にデプロイするのに役立つ AWS CloudFormation テンプレートがいくつか用意されています。詳細については、「Lambda 関数のテンプレート」を参照してください。

以下の手順では、SFTP 対応サーバー、FTPS 対応サーバー、FTP 対応サーバー、または AS2 対応サーバーを作成できます。

次のステップ

- SFTP 対応サーバーの作成
- FTPS 対応サーバーを作成する
- FTP 対応サーバーの作成
- AS2 の設定

AWS Transfer Family エンドポイントタイプマトリックス

Transfer Familyサーバーを作成する際には、使用するエンドポイントのタイプを選択します。以下の表は、各エンドポイントタイプの特性を説明しています。

エンドポイントタイプマトリックス

特徴	Public	VPC-インターネット	VPC-インターナル	VPC_エンドポイント (非推奨)
サポートされるプロトコル	SFTP	SFTP, FTPS, AS2	SFTP, FTP, FTPS, AS2	SFTP
アクセス	インターネット経由で。このエンドポイント タイプでは、VPC に特別な構成は必要ありません。	インターネット経由、および AWS Direct Connect または VPN 経由のオンプレミスデータセンターなどの VPC および VPC 接続環境内から。	AWS Direct Connect または VPN 経由のオンプレミスデータセンターなど、VPC および VPC 接続環境内から。	AWS Direct Connect または VPN 経由のオンプレミスデータセンターなど、VPC および VPC 接続環境内から。
静的 IP アドレス	静的 IP アドレスをアタッチすることはできません。 は、変更される可能性のある IP アドレス AWS を提供します。	Elastic IP アドレスをエンドポイントに接続できます。 AWS所有の IP アドレスまたは(「独自の IP アドレス (Bring your own	エンドポイントに接続されているプライベート IP アドレスは変更されません。	エンドポイントに接続されているプライベート IP アドレスは変更されません。

特徴	Public	VPC-インターネット	VPC-インターナル	VPC_エンドポイント (非推奨)
		IP address) を使用できます」)。エンドポイントに接続されている Elastic IP アドレスは変更されません。 サーバーに接続されているプライベート IP アドレスも変更されません。		

特徴	Public	VPC-インターネット	VPC-インターナル	VPC_エンドポイント (非推奨)
送信元 IP 許可リスト	このエンドポイント タイプは、送信元 IP アドレスによる許可リストをサポートしません。 エンドポイントはパブリックにアクセスでき、ポート 22 経由のトラフィックをリッスンします。 **Note** VPC でホストされるエンドポイントの場合、SFTP Transfer Family サーバーはポート 22 (デフォルト)、ポート 2222、または	送信元 IP アドレスによるアクセスを許可するには、サーバー エンドポイントに接続されているセキュリティ グループと、エンドポイントが存在するサブネットに接続されているネットワーク ACL を使用できます。	送信元 IP アドレスによるアクセスを許可するには、サーバー エンドポイントにアタッチされているセキュリティ グループと、エンドポイントが存在するサブネットにアタッチされているネットワーク アクセス コントロール リスト (ネットワーク ACL) を使用できます。	送信元 IP アドレスによるアクセスを許可するには、サーバー エンドポイントに接続されているセキュリティ グループと、エンドポイントが存在するサブネットに接続されているネットワーク ACL を使用できます。

特徴	Public	VPC-インターネット	VPC-インターナル	VPC_エンドポイント (非推奨)
	ポート 22000 で動作できます。			
クライアントファイアウォールの許可リスト	サーバーの DNS 名を許可する必要があります。 IP アドレスは変更される可能性があるため、クライアント ファイアウォールの許可リストに IP アドレスを使用することは避けてください。	サーバーの DNS 名、またはサーバーに接続されている Elastic IP アドレスを許可できます。	エンドポイントのプライベート IP アドレスまたは DNS 名を許可できます。	エンドポイントのプライベート IP アドレスまたは DNS 名を許可できます。

> **Note**
>
> このエンドポイント VPC_ENDPOINT タイプは現在非推奨となっており、新しいサーバーの作成には使用できません。EndpointType=VPC_ENDPOINT を使用する代わりに、新しい VPC エンドポイント タイプ (EndpointType=VPC) を使用します。これは、前の表で説明したように、内部またはインターネット接続として使用できます。詳細については、「VPC_ENDPOINT のサポート終了」を参照してください。

AWS Transfer Family サーバーのセキュリティ体制を強化するには、次のオプションを検討してください。

- VPC エンドポイントを内部アクセスで使用して、サーバーが VPC 内のクライアント、または AWS Direct Connect や VPN 経由でオンプレミスデータセンターなどの VPC 接続環境にのみアクセスできるようにします。
- クライアントがインターネット経由でエンドポイントにアクセスできるようにし、サーバーを保護するには、インターネットに接続されたアクセスを持つ VPC エンドポイントを使用します。次に、ユーザーのクライアントをホストする特定の IP アドレスからのトラフィックのみを許可するように VPC のセキュリティ グループを変更します。
- パスワード ベースの認証が必要で、サーバーでカスタム ID プロバイダーを使用している場合は、パスワード ポリシーでユーザーが弱いパスワードを作成できないようにし、ログイン試行の失敗回数を制限することがベスト プラクティスです。
- AWS Transfer Family はマネージドサービスであるため、シェルアクセスは提供されません。基盤となる SFTP サーバーに直接アクセスして、Transfer Family サーバー上で OS ネイティブ コマンドを実行することはできません。
- 内部アクセスのある VPC エンドポイントの前で Network Load Balancer を使用します。ロード バランサーのリスナー ポートをポート 22 から別のポートに変更します。これにより、ポート 22 がスキャンに最も一般的に使用されるため、ポート スキャナーやボットがサーバーを調査するリスクは軽減されますが、排除されるわけではありません。詳細については、ブログ記事「Network Load Balancer がセキュリティグループ をサポートするようになりました」を参照してください。

> **Note**
>
> Network Load Balancer を使用する場合、 AWS Transfer Family CloudWatch ログには実際のクライアント IP アドレスではなく NLB の IP アドレスが表示されます。

SFTP、FTPS、または FTP サーバーエンドポイントの設定

AWS Transfer Family サービスを使用してファイル転送サーバーを作成できます。次のファイル転送プロトコルを使用できます。

- Secure Shell (SSH) File Transfer Protocol (SFTP) – SSH 経由のファイル転送 詳細については、「the section called “SFTP 対応サーバーの作成”」を参照してください。

> **Note**
>
> SFTP Transfer Family サーバーを作成する AWS CDK 例を示します。この例では を使用しており TypeScript、 GitHub ここで で利用できます。

- File Transfer Protocol Secure (FTPS) – TLS 暗号化によるファイル転送 詳細については、「the section called “FTPS 対応サーバーを作成する”」を参照してください。
- File Transfer Protocol (FTP) - 暗号化されていないファイル転送 詳細については、「the section called “FTP 対応サーバーの作成”」を参照してください。
- 適用性ステートメント 2 (AS2) — 構造化 business-to-business データを転送するためのファイル転送。詳細については、「the section called “AS2 を設定する”」を参照してください。AS2 では、デモンストレーション目的で AWS CloudFormation スタックをすばやく作成できます。この手順は、「テンプレートを使用して Transfer Family AS2 スタックのデモを作成する」で説明されています。

複数のプロトコルを持つサーバーを作成できます。

> **Note**
>
> 同じサーバエンドポイントに対して複数のプロトコルを有効にしており、複数のプロトコルで同じユーザ名を使用してアクセスを提供したい場合、プロトコルに固有の認証情報がIDプロバイダに設定されている限り、そうすることができます。FTP の場合は、SFTP および FTPS とは別の認証情報を維持することをお勧めします。SFTP や FTPS と異なり、FTP は認証情報を平文で送信します。FTP の認証情報を SFTP または FTPS から隔離することを推奨します。そうすれば、FTP の認証情報が共有または公開されても、SFTP または FTPS を使用しているワークロードは引き続き安全だからです。

サーバーを作成するときは、特定の を選択して、そのサーバー AWS リージョン に割り当てられているユーザーのファイルオペレーションリクエストを実行します。サーバーに 1 つ以上のプロトコルを割り当てるとともに、次のいずれかの ID プロバイダーのタイプも割り当てます。

- SSH キーを使用してサービスを管理します。詳細については、「サービスマネージドユーザーの使用」を参照してください。
- AWS Directory Service for Microsoft Active Directory （ AWS Managed Microsoft AD ）。 この方法では、Microsoft Active Directory グループを統合して、Transfer Family サーバーへのアクセスを提

供できます。詳細については、「AWS Directory Service ID プロバイダーの使用」を参照してください。

- カスタムメソッド。カスタム ID プロバイダー方式では、 AWS Lambda または Amazon API Gateway を使用し、ディレクトリサービスを統合して SFTP ユーザーの認証と許可を実行できます。サービスは、サーバーを一意に識別する識別子を自動的に割り当てます。詳細については、「カスタム ID プロバイダーの使用」を参照してください。Transfer Family には AWS CloudFormation 、カスタム ID プロバイダーを使用するサーバーをすばやくデプロイするために使用できるテンプレートが用意されています。
  - 認証用の Lambda 関数 では、認証に Lambda 関数を使用する CloudFormation テンプレートについて説明します。
  - API Gateway メソッドを使用した認証 では、認証に Amazon API Gateway メソッドを使用するテンプレートについて説明します CloudFormation。

また、デフォルトのサーバーエンドポイントを使用するか、Amazon Route 53サービスを使用するか、好みのドメインネームシステム（DNS）サービスを使用して、サーバーにエンドポイントタイプ（一般にアクセス可能か、VPCがホストしているか）とホスト名を割り当てます。サーバーのホスト名は、作成 AWS リージョン 先の で一意である必要があります。

さらに、Amazon CloudWatch ログ記録ロールを割り当てて CloudWatch 、ログにイベントをプッシュしたり、サーバーで使用できる暗号化アルゴリズムを含むセキュリティポリシーを選択したり、キーと値のペアであるタグの形式でメタデータをサーバーに追加したりできます。

> **Important**
>
> 料金は、インスタンス化された SFTP サーバーとデータ転送について発生します。Transfer Family の使用コストの見積もりを取得するために の料金と を使用する方法については、「 のAWS Transfer Family 料金 AWS Pricing Calculator 」を参照してください。

SFTP 対応サーバーの作成

SFTP は Secure Shell (SSH) File Transfer Protocol の略称で、インターネット経由の安全なデータ転送に使用されるネットワークプロトコルです。このプロトコルは、SSH のセキュリティおよび認証機能をすべてサポートします。金融サービス、医療、小売、広告などさまざまな業界で、ビジネスパートナー間の機密情報を含むデータ交換に広く使われています。

> **Note**
>
> Transfer Family 用の SFTP サーバーは、ポート 22 経由で動作します。VPC でホストされるエンドポイントの場合、SFTP Transfer Family サーバーはポート 2222 またはポート 22000 で動作することもできます。詳細については、「Virtual Private Cloud でサーバーを作成する」を参照してください。

以下の資料も参照してください。

- SFTP Transfer Family サーバーを作成する AWS CDK 例を示します。この例では を使用しており TypeScript、 GitHub ここで で利用できます。
- VPC 内に Transfer Family サーバーをデプロイする方法のチュートリアルについては、「IP 許可リストを使用して AWS Transfer Family サーバーを保護する」を参照してください。

SFTP 対応サーバーを作成するには

1. https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開き、ナビゲーションペインからサーバーを選択し、サーバーの作成を選択します。
2. [Choose protocols] (プロトコルの選択) で SFTP を選択し、[Next] (次へ) を選択します。
3. [Choose an identity provider] (ID プロバイダーの選択) で、ユーザーアクセスの管理に使用したい ID プロバイダーを選択します。次のオプションがあります。

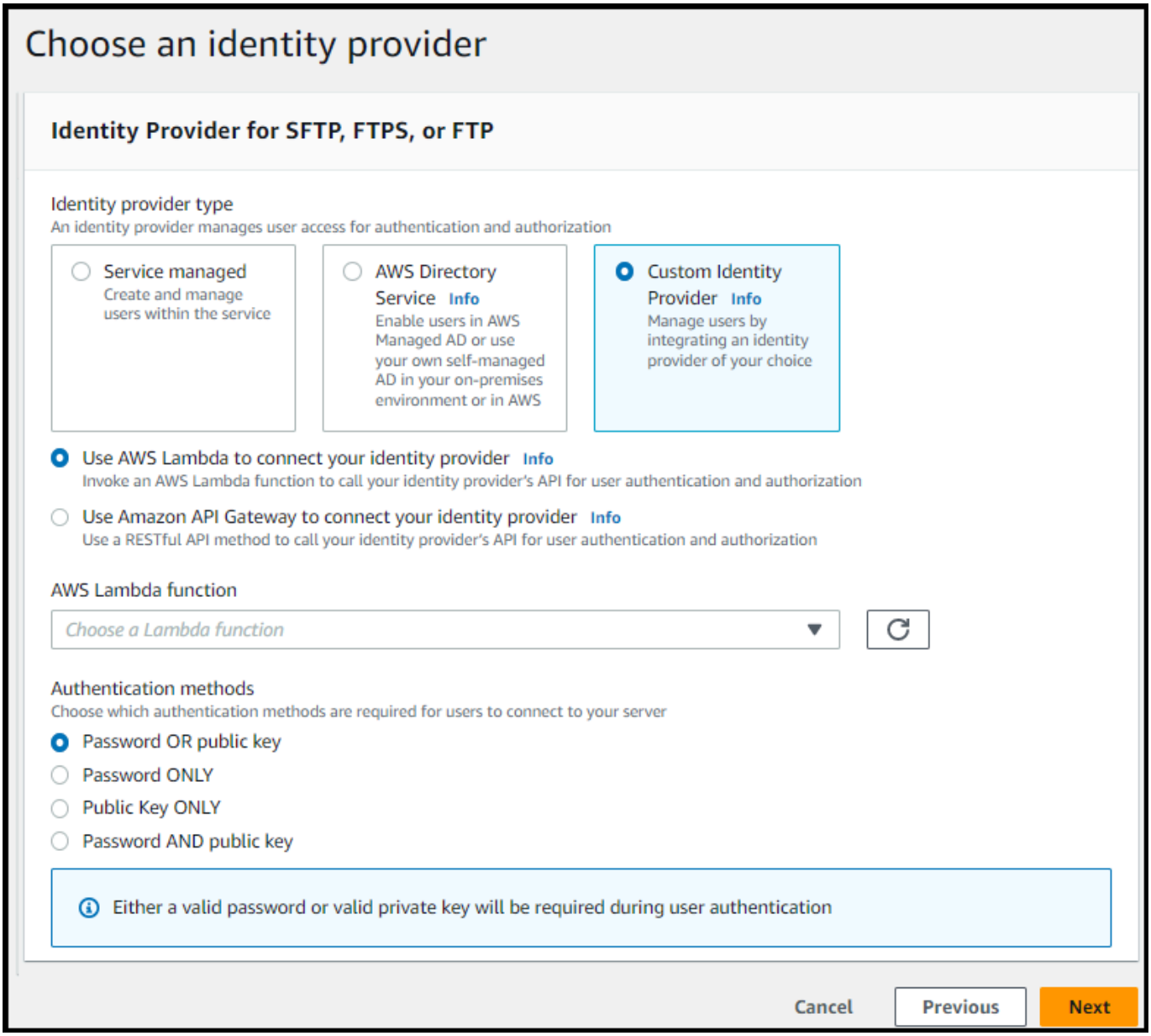
   - サービス管理 – ユーザー ID とキーは に保存します AWS Transfer Family。
   - AWS Directory Service for Microsoft Active Directory – エンドポイントにアクセスするための AWS Directory Service ディレクトリを指定します。そうすることで、Active Directory に保存されている認証情報を使用してユーザーを認証できるようになります。 AWS Managed Microsoft AD ID プロバイダーの操作の詳細については、「」を参照してくださいAWS Directory Service ID プロバイダーの使用。

     > **Note**
     >
     > - クロスアカウントディレクトリと共有ディレクトリは、 ではサポートされていません AWS Managed Microsoft AD。
     > - Directory Service を ID プロバイダーとしてサーバーを設定するには、いくつかの AWS Directory Service アクセス許可を追加する必要があります。詳細については、

「の使用を開始する前に AWS Directory Service for Microsoft Active Directory」を参照してください。

- 「Custom Identity Provider」 — 以下のいずれかのオプションを選択する：
  - AWS Lambda を使用して ID プロバイダーを接続する – Lambda 関数にバックアップされた既存の ID プロバイダーを使用できます。Lambda 関数の名前を指定します。詳細については、「AWS Lambda を使用して ID プロバイダーを統合する」を参照してください。
  - 「Amazon API Gateway を使って ID プロバイダーに接続する」 — ID プロバイダーとして使用するために、Lambda 関数に裏打ちされた API ゲートウェイメソッドを作成することができます。Amazon API Gateway URL と呼び出しロールを指定します。詳細については、「Amazon API Gateway を使用して ID プロバイダーを統合する」を参照してください。

  どちらのオプションでも、認証方法を指定することもできます。
  - パスワードまたはキー – ユーザーはパスワードまたはキーを使用して認証できます。これは、デフォルト値です。
  - パスワードのみ – ユーザーは接続するためにパスワードを入力する必要があります。
  - キーのみ — ユーザーは接続するためにプライベートキーを指定する必要があります。
  - パスワードとキー – ユーザーは接続するためにプライベートキーとパスワードの両方を指定する必要があります。サーバーは最初にキーを確認し、キーが有効な場合はパスワードの入力を求めます。提供されたプライベートキーが保存されたパブリックキーと一致しない場合、認証は失敗します。



4. [次へ] をクリックします。
5. [Choose an endpoint] (エンドポイントの選択) で次のように操作します。

   a. [Endpoint type] (エンドポイントタイプ) として [Publicly accessible] (パブリックにアクセス可能な) エンドポイントタイプを選択します。VPC ホステッドエンドポイントについては、「Virtual Private Cloud でサーバーを作成する」を参照してください。

   b. (オプション) [Custom hostname] (カスタムホスト名) で [None] (なし) を選択します。

      によって提供されるサーバーホスト名を取得します AWS Transfer Family。サーバーホスト名の形式は *serverId*.server.transfer.*regionId*.amazonaws.com です。

カスタムホスト名には、サーバーエンドポイントのカスタムエイリアスを指定します。カスタムホスト名の使用の詳細については、「カスタムホスト名の使用」を参照してください。

c. (オプション) [FIPS Enabled] (FIPS 対応) で、[FIPS Enabled endpoint] (FIPS 対応エンドポイント) チェックボックスをオンにして、エンドポイントが連邦情報処理標準 (FIPS) に準拠することを確認します。

> **Note**
>
> FIPS 対応エンドポイントは、北米 AWS リージョンでのみ使用できます。利用可能なリージョンについては、「AWS 全般のリファレンス」の「AWS Transfer Family エンドポイントとクォータ」を参照してください。FIPS 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。

d. [Next] (次へ) を選択します。

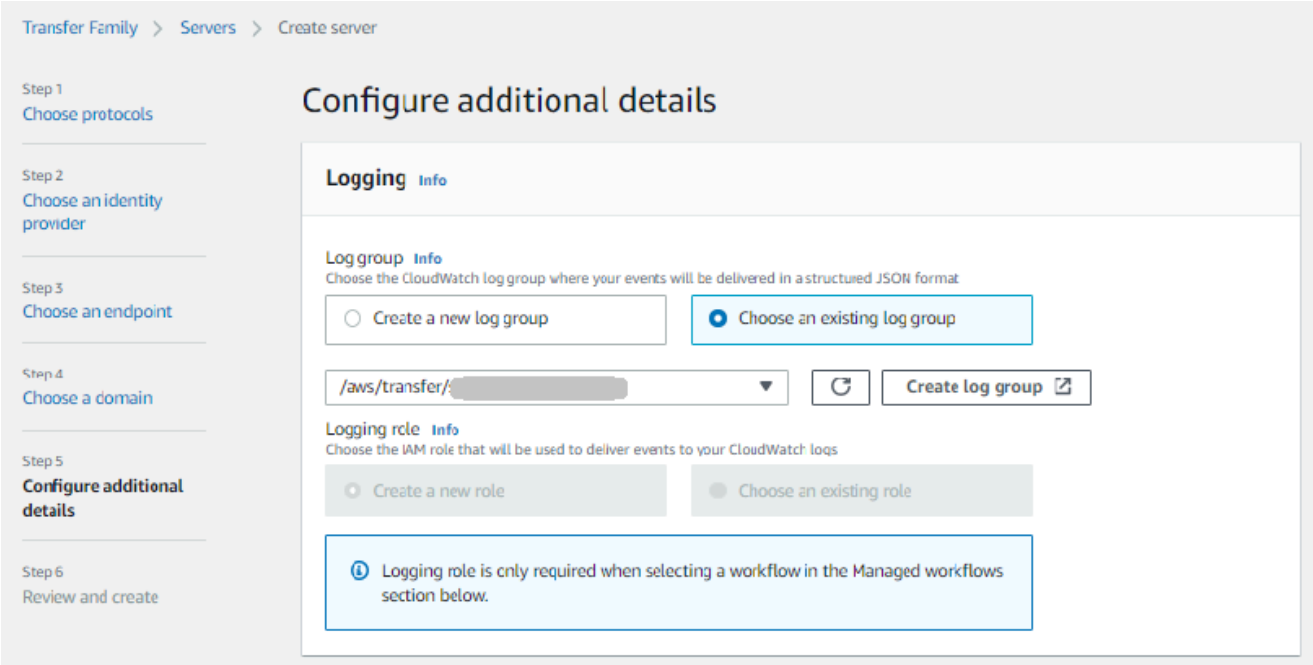
6. 「ドメインの選択」ページで、選択したプロトコルでデータを保存およびアクセスするために使用する AWS ストレージサービスを選択します。

   - Amazon S3 を選択すると、選択したプロトコル経由でファイルをオブジェクトとして保存してアクセスできます。
   - Amazon EFS を選択すると、選択したプロトコル経由で Amazon EFS ファイルシステム内でファイルを保存してアクセスできます。

   [Next] (次へ) を選択します。

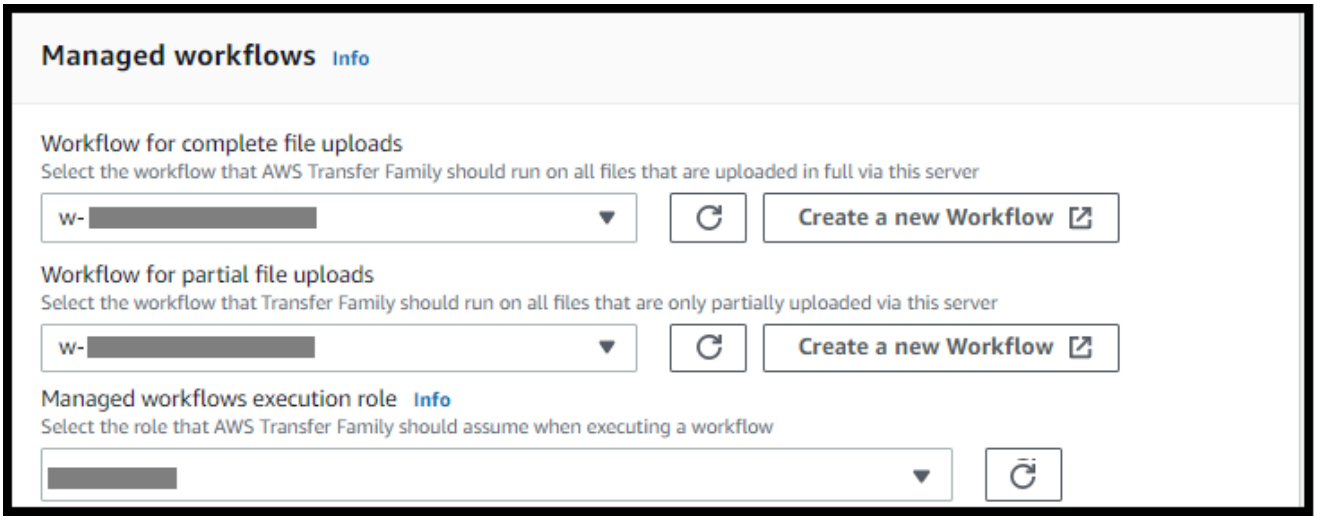
7. [Configure additional details] (その他の詳細の構成) で次のように操作します。

   a. ログの場合、既存のログ グループを指定するか、新しいログ グループを作成します (デフォルト オプション)。既存のロググループを選択する場合は、 に関連付けられているロググループを選択する必要があります AWS アカウント。



ロググループの作成 を選択すると、 CloudWatch コンソール (https://console.aws.amazon.com/cloudwatch/) がロググループの作成ページを開きます。詳細については、「 ログ」の CloudWatch 「ロググループの作成」を参照してください。

b. (オプション)[管理対象ワークフローの場合]、ワークフローの実行時に Transfer Family が引き受けるワークフロー ID (および対応するロール) を選択します。完全なアップロード時に実行するワークフローと、部分的なアップロード時に実行するワークフローを選択できます。マネージドワークフローを使用したファイルの処理の詳細については、AWS Transfer Family マネージドワークフローを参照してください。



c. [Cryptographic algorithm options] (暗号化アルゴリズムオプション) で、サーバーで使用できる暗号化アルゴリズムを含むセキュリティポリシーを選択します。最新のセキュリティポ

リシーがデフォルトです。詳細については、「」を参照してくださいAWS Transfer Family サーバーのセキュリティポリシー。

d. （オプション）[サーバーホストキー] には、クライアントが SFTP でサーバーに接続するときにサーバーを識別するために使用する RSA、ED25519、または ECDSA プライベートキーを入力します。複数のホストキーを区別するための説明を追加することもできます。

   サーバーを作成した後、追加のホスト キーを追加できます。複数のホスト キーを持つことは、キーをローテーションする場合、または RSA キーと ECDSA キーなどの異なるタイプのキーが必要な場合に便利です。

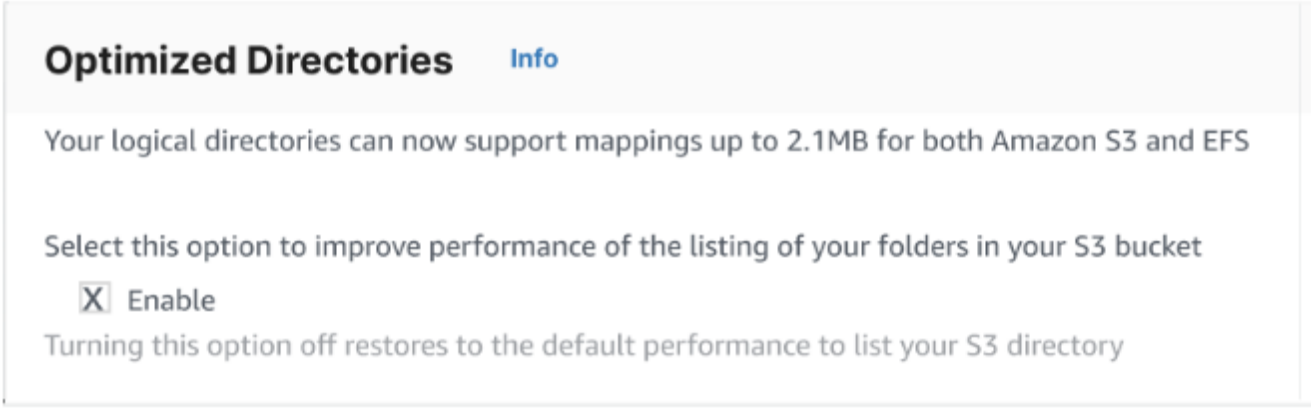
   > **Note**
   >
   > 「サーバーホストキー」セクションは、既存の SFTP 対応サーバーからユーザーを移行する場合のみ使用します。

e. (オプション) [Tags] (タグ) の [Key] (キー) と [Value] (値) にキーバリューペアとして 1 つ以上のタグを入力して [Add tag] (タグの追加) を選択します。

f. [次へ] をクリックします。

g. Amazon S3 ディレクトリのパフォーマンスを最適化できます。例えば、ホームディレクトリに移動し、10,000 個のサブディレクトリがあるとします。つまり、Amazon S3 バケットには 10,000 個のフォルダがあります。このシナリオでは、`ls` (list) コマンドを実行すると、リストオペレーションに 6～8 分かかります。ただし、ディレクトリを最適化すると、このオペレーションにかかる時間はわずか数秒です。

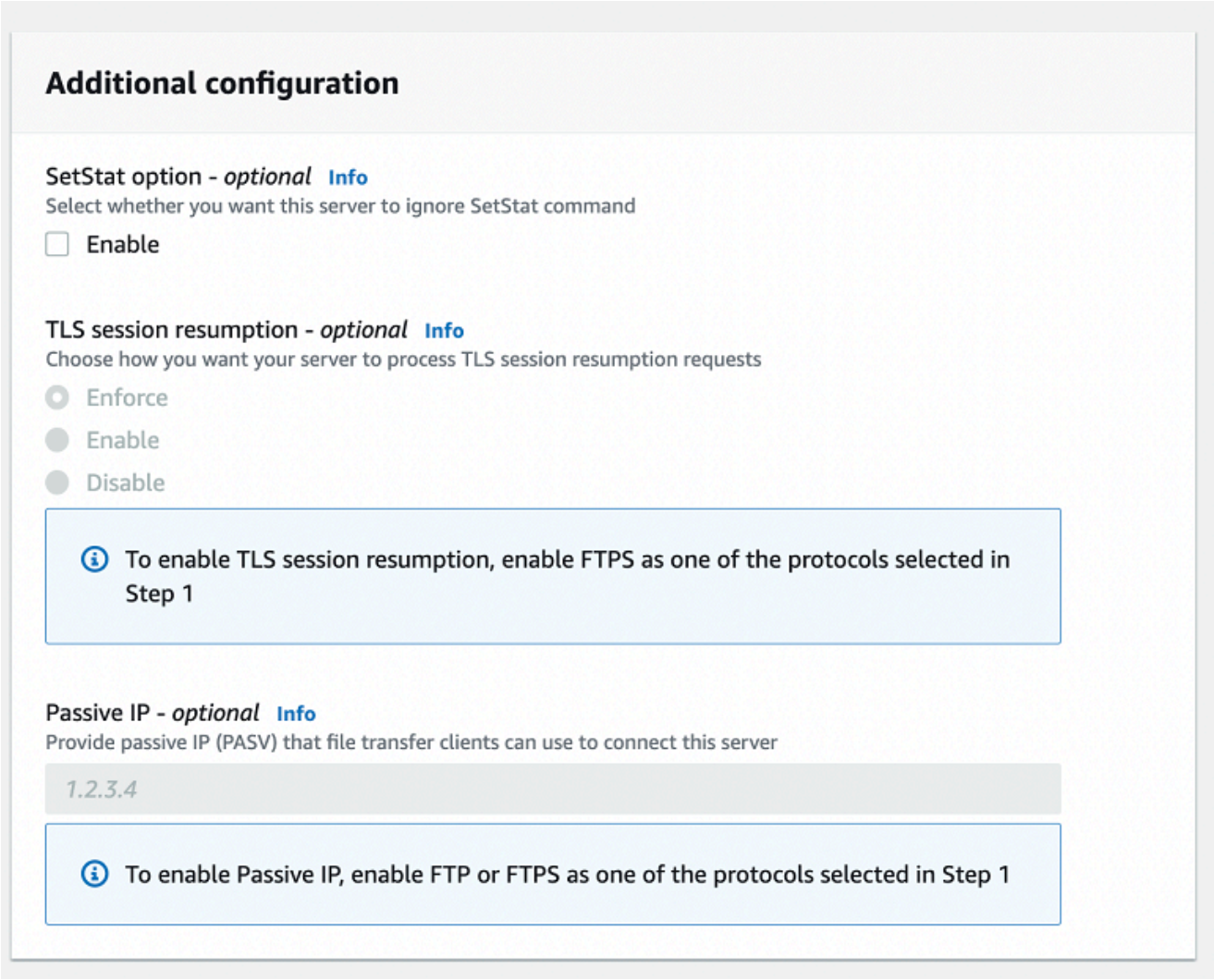
   コンソールを使用してサーバーを作成すると、最適化されたディレクトリがデフォルトで有効になります。API を使用してサーバーを作成する場合、この動作はデフォルトでは有効になっていません。



h. （オプション) 組織のポリシーや利用規約などのカスタマイズされたメッセージをエンドユーザーに表示するように AWS Transfer Family サーバーを設定します。[表示バナー] の [認証前表示バナー] テキスト ボックスに、ユーザーが認証する前に表示するテキスト メッセージを入力します。

i. (オプション) 次の追加オプションを構成できます。

- SetStat オプション : このオプションを有効にすると、クライアントが Amazon S3 バケットにアップロードするファイルSETSTATで を使用しようとしたときに生成されるエラーが無視されます。詳細については、「」のSetStatOptionドキュメントを参照してくださいProtocolDetails。
- [TLS セッション再開]:このオプションは、このサーバーのプロトコルの 1 つとして FTPS を有効にしている場合にのみ使用できます。
- [パッシブ IP]: このオプションは、このサーバーのプロトコルの 1 つとして FTPS または FTP を有効にしている場合にのみ使用できます。



8. [Review and create] (確認と作成) で選択内容を確認します。

   - いずれかを編集するには、ステップの隣にある [Edit] (編集) を選択します。

     > **Note**
     >
     > 編集を選んだステップの後に、各ステップを見直す必要があります。

   - 変更がない場合、[Create server] (サーバーの作成) を選択してサーバーを作成します。[Servers] (サーバー) ページに誘導され、次に示すように新しいサーバーの一覧が表示されます。

新しい SFTP サーバーのステータスが [Online] (オンライン) に変わるまでに数分かかる場合があります。その時点で、サーバーでファイルオペレーションを実行できますが、まずユーザーを作成する必要があります。ユーザーの作成の詳細については、「」を参照してくださいサーバーエンドポイントのユーザーの管理。

FTPS 対応サーバーを作成する

File Transfer Protocol over SSL (FTPS) は FTP の拡張版です。Transport Layer Security (TLS) および Secure Sockets Layer (SSL) 暗号化プロトコルを使用して、トラフィックを暗号化します。FTPS を利用すると、制御チャネルとデータチャネルの接続の両方を同時に、または個別に暗号化できます。

FTPS 対応サーバーを作成するには

1. https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開き、ナビゲーションペインからサーバーを選択し、サーバーの作成を選択します。
2. [Choose protocols] (プロトコルの選択) で FTPS を選択してから [Next] (次へ) を選択します。

   [Server certificate] (サーバー証明書) で、クライアントが FTPS 経由でサーバーに接続するときにサーバーを識別するために使用される AWS Certificate Manager (ACM) に保存されている証明書を選択してから [Next] (次へ) を選択します。

   新しいパブリック証明書をリクエストするには、AWS Certificate Manager ユーザーガイドの「パブリック証明書をリクエストする」を参照してください。

   既存の証明書を ACM にインポートするには、AWS Certificate Manager ユーザーガイドの「ACM に証明書をインポートする」を参照してください。

   プライベート IP アドレス経由で FTPS を使用できるようにプライベート証明書をリクエストするには、AWS Certificate Manager ユーザーガイドの「プライベート証明書のリクエスト」を参照してください。

   以下の暗号化アルゴリズムとキーサイズの証明書がサポートされています。

   - 2048 ビット RSA (RSA_2048)
   - 4096 ビット RSA (RSA_4096)
   - 楕円素数曲線 256 ビット (EC_Prime256v1)
   - 楕円素数曲線 384 ビット (EC_secp384R1)
   - 楕円素数曲線 521 ビット (EC_secp521R1)

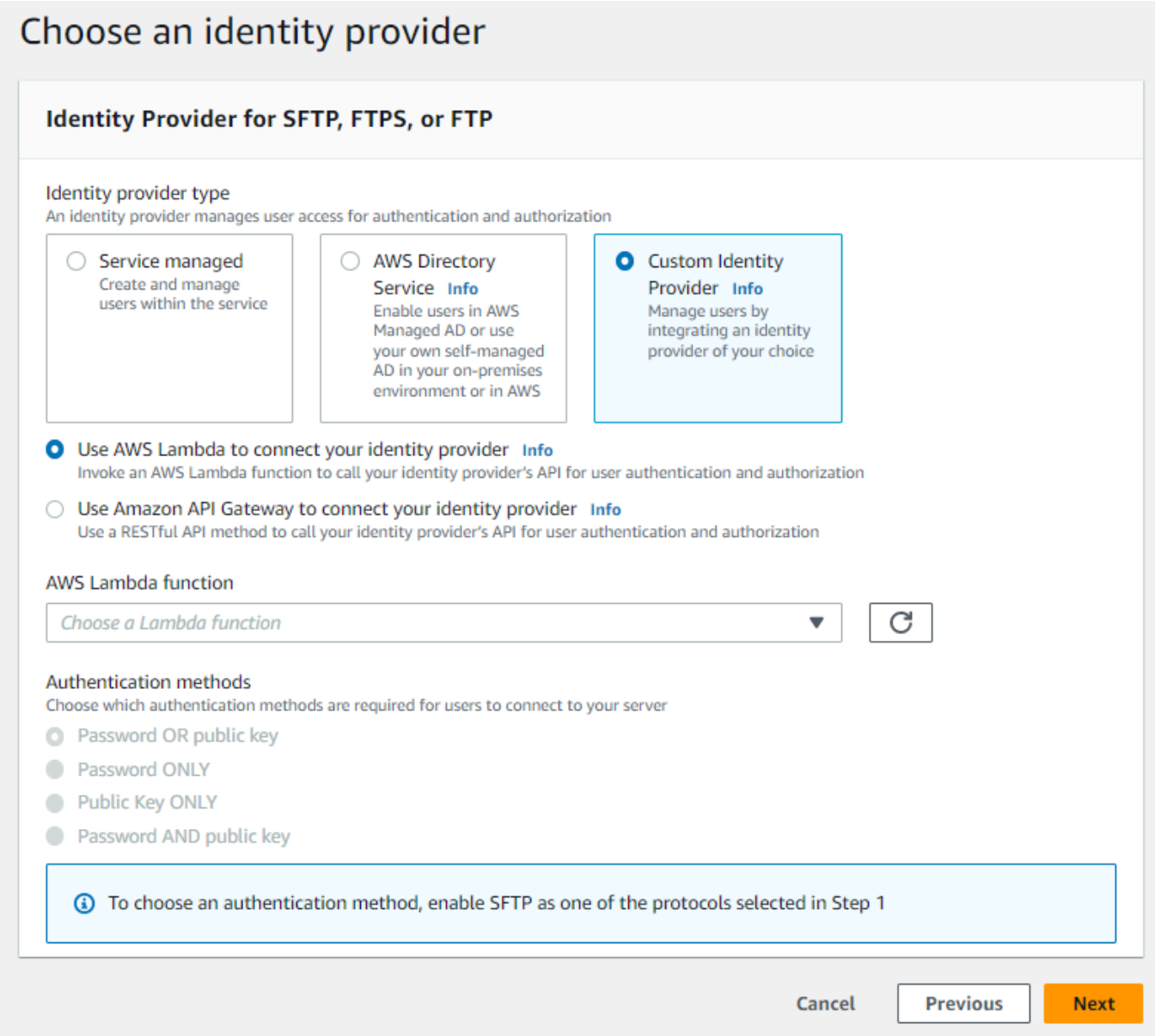
> **Note**
>
> 証明書は、FQDN または IP アドレスが指定された有効な SSL/TLS X.509 バージョン 3 証明書で、発行者に関する情報が含まれている必要があります。

3. [Choose an identity provider] (ID プロバイダーの選択) で、ユーザーアクセスの管理に使用したい ID プロバイダーを選択します。次のオプションがあります。

   - AWS Directory Service for Microsoft Active Directory – エンドポイントにアクセスするための AWS Directory Service ディレクトリを指定します。そうすることで、Active Directory に保存されている認証情報を使用してユーザーを認証できるようになります。 AWS Managed Microsoft AD ID プロバイダーの操作の詳細については、「」を参照してくださいAWS Directory Service ID プロバイダーの使用。

     > **Note**
     >
     > - クロスアカウントディレクトリと共有ディレクトリは、 ではサポートされていません AWS Managed Microsoft AD。
     > - Directory Service を ID プロバイダーとしてサーバーを設定するには、いくつかの AWS Directory Service アクセス許可を追加する必要があります。詳細については、「の使用を開始する前に AWS Directory Service for Microsoft Active Directory」を参照してください。

   - 「Custom Identity Provider」 ー 以下のいずれかのオプションを選択する：
     - AWS Lambda を使用して ID プロバイダーを接続する – Lambda 関数にバックアップされた既存の ID プロバイダーを使用できます。Lambda 関数の名前を指定します。詳細については、「AWS Lambda を使用して ID プロバイダーを統合する」を参照してください。
     - 「Amazon API Gateway を使って ID プロバイダーに接続する」 ー ID プロバイダーとして使用するために、Lambda 関数に裏打ちされた API ゲートウェイメソッドを作成することができます。Amazon API Gateway URL と呼び出しロールを指定します。詳細については、「Amazon API Gateway を使用して ID プロバイダーを統合する」を参照してください。



4. [次へ] をクリックします。
5. [Choose an endpoint] (エンドポイントの選択) で、次のように操作します。

> **Note**
>
> Transfer Family の FTPS サーバーは、ポート 21 (コントロールチャネル) とポート範囲 8192-8200 (データチャネル) で動作します。

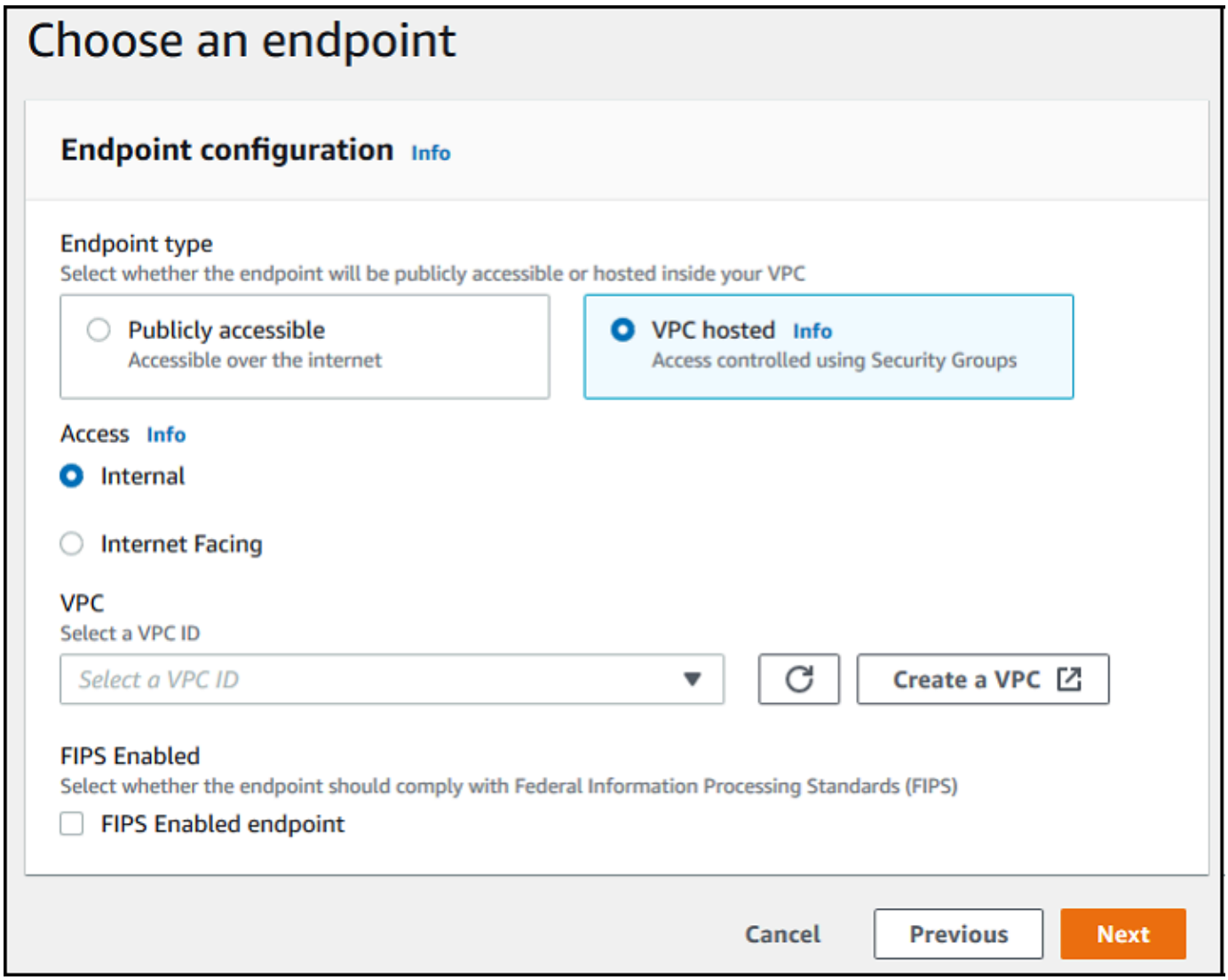
a. [Endpoint type] (エンドポイントタイプ) で、サーバーのエンドポイントのホストになる VPC ホステッドエンドポイントタイプを選択します。VPC ホストエンドポイントの設定については、「Virtual Private Cloud でサーバーを作成する」を参照してください。

> **Note**
>
> パブリックにアクセス可能なエンドポイントはサポートされません。

b. (オプション) [FIPS Enabled] (FIPS 対応) で、[FIPS Enabled endpoint] (FIPS 対応エンドポイント) チェックボックスをオンにして、エンドポイントが連邦情報処理標準 (FIPS) に準拠することを確認します。

> **Note**
>
> FIPS 対応エンドポイントは、北米 AWS リージョンでのみ使用できます。利用可能なリージョンについては、「AWS 全般のリファレンス」の「AWS Transfer Family エンドポイントとクォータ」を参照してください。FIPS 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。

c. [Next] (次へ) を選択します。



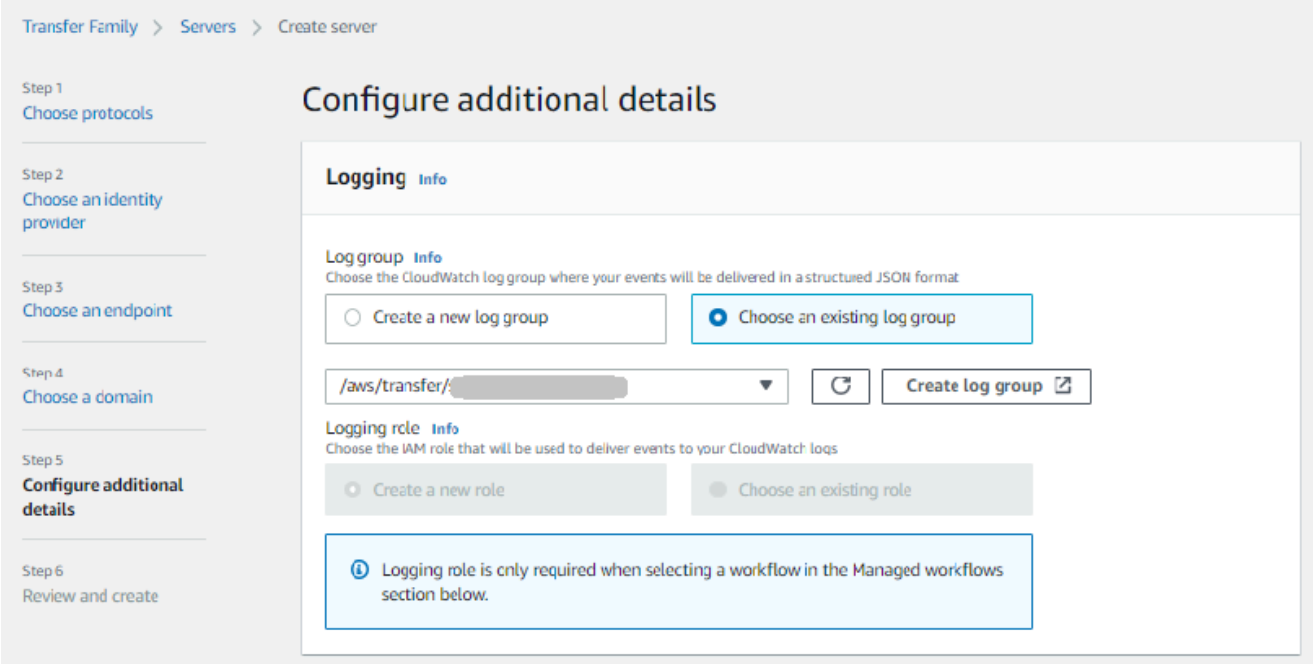
6. 「ドメインの選択」ページで、選択したプロトコルでデータを保存およびアクセスするために使用する AWS ストレージサービスを選択します。

   - Amazon S3 を選択すると、選択したプロトコル経由でファイルをオブジェクトとして保存してアクセスできます。
   - Amazon EFS を選択すると、選択したプロトコル経由で Amazon EFS ファイルシステム内でファイルを保存してアクセスできます。

   [Next] (次へ) を選択します。

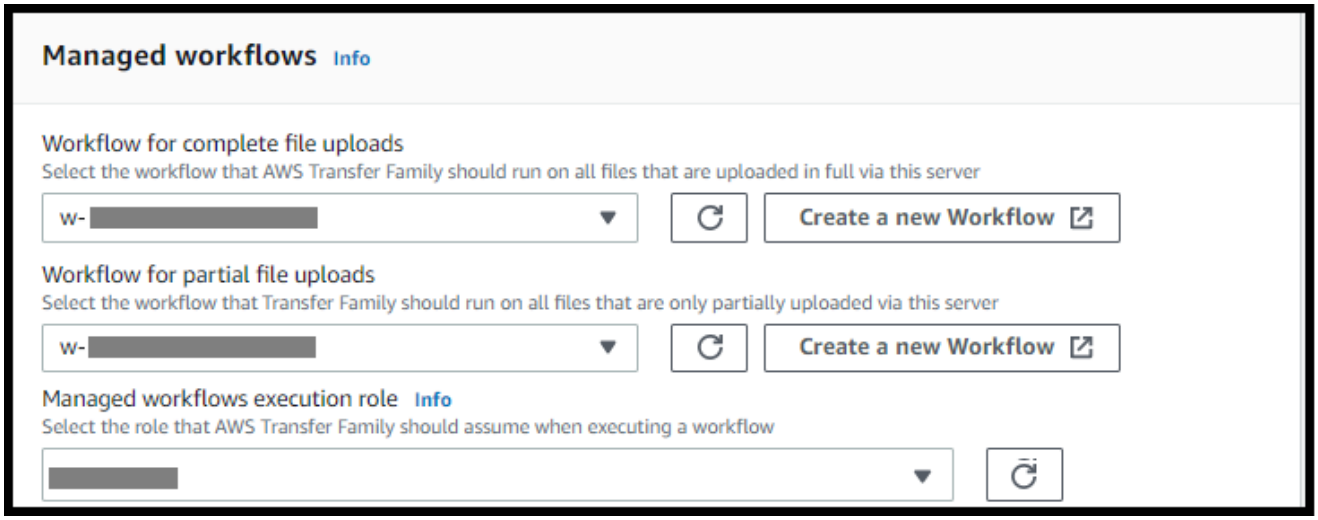
7. [Configure additional details] (その他の詳細の構成) で次のように操作します。

a. ログの場合、既存のログ グループを指定するか、新しいログ グループを作成します (デフォルト オプション)。



ロググループの作成 を選択すると、 CloudWatch コンソール (https://console.aws.amazon.com/cloudwatch/) がロググループの作成ページを開きます。詳細については、「 ログ」の CloudWatch 「ロググループの作成」を参照してください。

b. (オプション)[管理対象ワークフローの場合]、ワークフローの実行時に Transfer Family が引き受けるワークフロー ID (および対応するロール) を選択します。完全なアップロード時に実行するワークフローと、部分的なアップロード時に実行するワークフローを選択できます。マネージドワークフローを使用したファイルの処理の詳細については、AWS Transfer Family マネージドワークフローを参照してください。



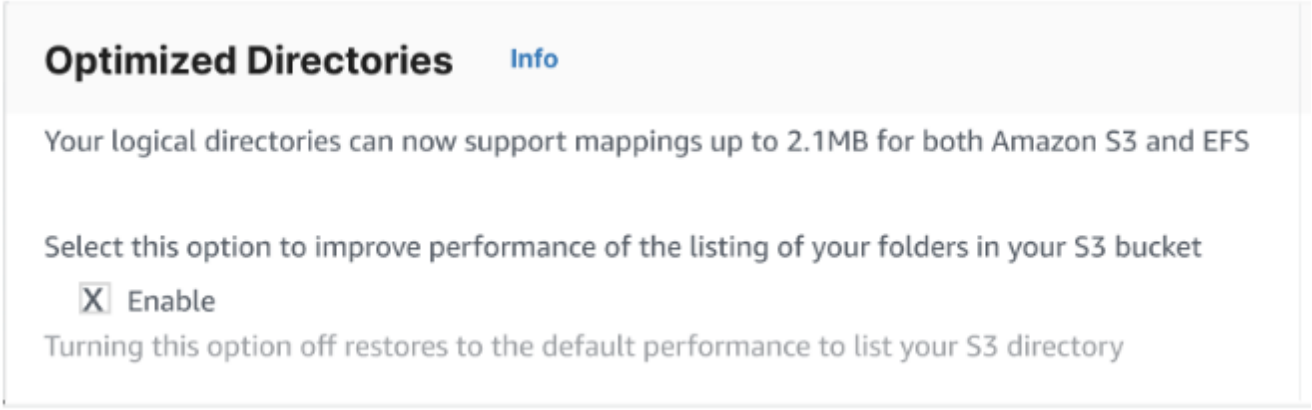
c. [Cryptographic algorithm options] (暗号化アルゴリズムオプション) で、サーバーで使用できる暗号化アルゴリズムを含むセキュリティポリシーを選択します。最新のセキュリティポリシーがデフォルトです。詳細については、「」を参照してくださいAWS Transfer Family サーバーのセキュリティポリシー。

d. 「サーバーホストキー」は空白のままにしておきます。

e. (オプション) [Tags] (タグ) の [Key] (キー) と [Value] (値) にキーバリューペアとして 1 つ以上のタグを入力して [Add tag] (タグの追加) を選択します。

f. Amazon S3 ディレクトリのパフォーマンスを最適化できます。例えば、ホームディレクトリに移動し、10,000 個のサブディレクトリがあるとします。つまり、Amazon S3 バケットには 10,000 個のフォルダがあります。このシナリオでは、 `ls` (list) コマンドを実行すると、リストオペレーションに 6～8 分かかります。ただし、ディレクトリを最適化すると、このオペレーションにかかる時間はわずか数秒です。

   コンソールを使用してサーバーを作成すると、最適化されたディレクトリがデフォルトで有効になります。API を使用してサーバーを作成する場合、この動作はデフォルトでは有効になっていません。



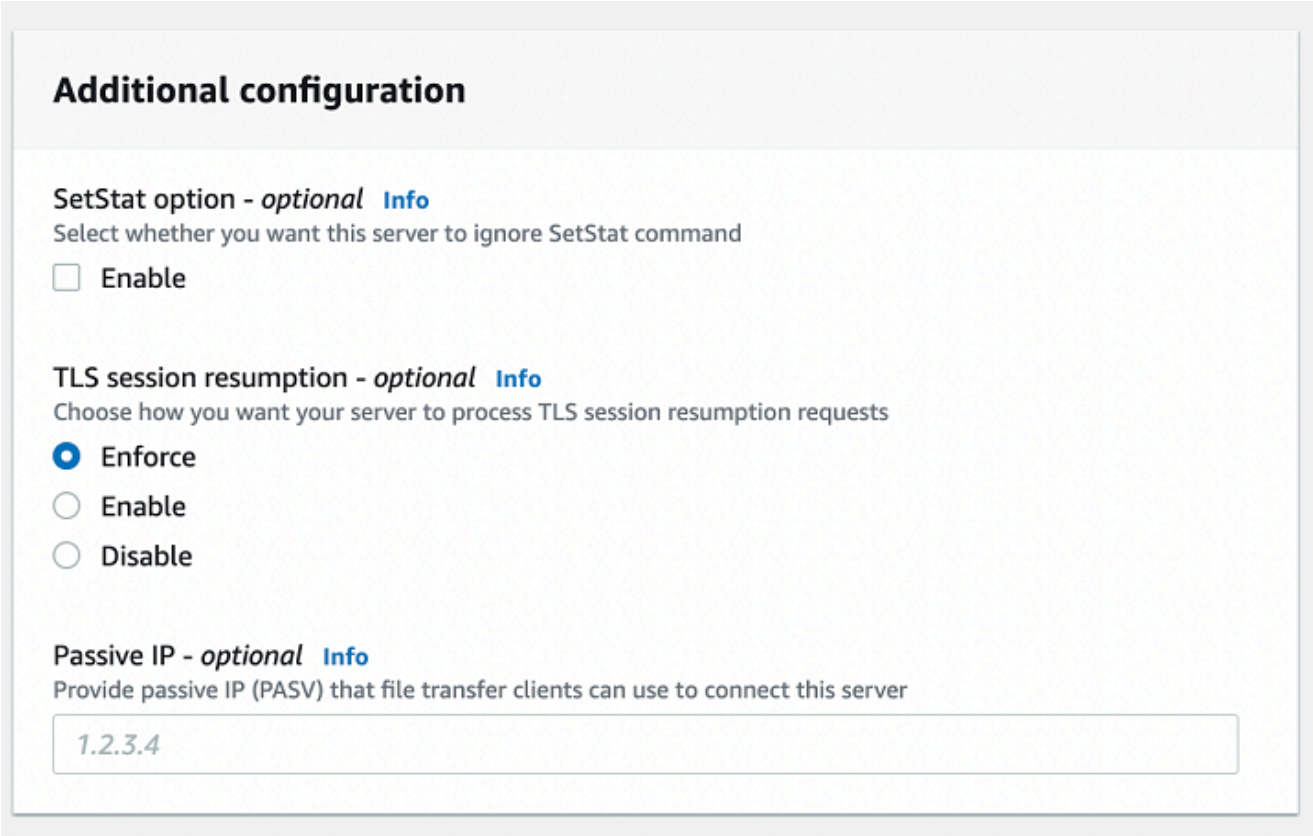
g. [次へ] をクリックします。

h. (オプション) 組織のポリシーや利用規約などのカスタマイズされたメッセージをエンドユーザーに表示するように AWS Transfer Family サーバーを設定できます。認証に成功したユーザーに対して、カスタマイズされた今日のメッセージ (MOTD) を表示することもできます。

[ディスプレイ バナー] の場合、[プリ認証ディスプレイ バナー] テキスト ボックスに、ユーザーが認証する前に表示したいテキスト メッセージを入力し、ポスト認証ディスプレイ バナー テキスト ボックスには、ユーザーが認証に成功した後に表示したいテキストを入力してください。

i. (オプション) 次の追加オプションを構成できます。

- SetStat オプション : このオプションを有効にすると、クライアントが Amazon S3 バケットにアップロードするファイルSETSTATで を使用しようとしたときに生成されるエラーが無視されます。詳細については、ProtocolDetails「」トピックのSetStatOptionドキュメントを参照してください。
- [TLS セッション再開]:FTPS セッションの制御接続とデータ接続の間でネゴシエートされた秘密キーを再開または共有するメカニズムを提供します。詳細については、ProtocolDetails「」トピックのTlsSessionResumptionModeドキュメントを参照してください。
- [Passive IP] : FTP および FTPS プロトコルのパッシブモードを示します。ファイアウォール、ルーター、ロードバランサーのパブリック IP アドレスなど、単一 IPv4 アドレスを入力します。詳細については、ProtocolDetails「」トピックのPassiveIpドキュメントを参照してください。



8. [Review and create] (確認と作成) で選択内容を確認します。

   - いずれかを編集するには、ステップの隣にある [Edit] (編集) を選択します。

     > **Note**
     >
     > 編集を選んだステップの後に、各ステップを見直す必要があります。

   - 変更がない場合、[Create server] (サーバーの作成) を選択してサーバーを作成します。[Servers] (サーバー) ページに誘導され、次に示すように新しいサーバーの一覧が表示されます。

新しい SFTP サーバーのステータスが [Online] (オンライン) に変わるまでに数分かかる場合があります。その時点で、サーバーはユーザーのためにファイルオペレーションを実行できます。

次のステップ： 次のステップでは、カスタム ID プロバイダーの使用 に進んでユーザーを設定します。

FTP 対応サーバーの作成

File Transfer Protocol (FTP) は、データの転送に使用されるネットワークプロトコルです。FTP は制御とデータ転送に別のチャネルを使用します。制御チャネルは、終了するか非アクティブでタイムアウトになるまで開いています。データチャネルは、転送期間にアクティブになります。FTP は平文を使用し、トラフィックの暗号化をサポートしません。

> **Note**
>
> FTP を有効にするときは、VPC でホストされるエンドポイントの内部アクセスオプションを選択する必要があります。サーバーでデータをパブリックネットワークを通過させる必要がある場合は、SFTP や FTPS などの安全なプロトコルを使用する必要があります。

FTP 対応サーバーを作成するには

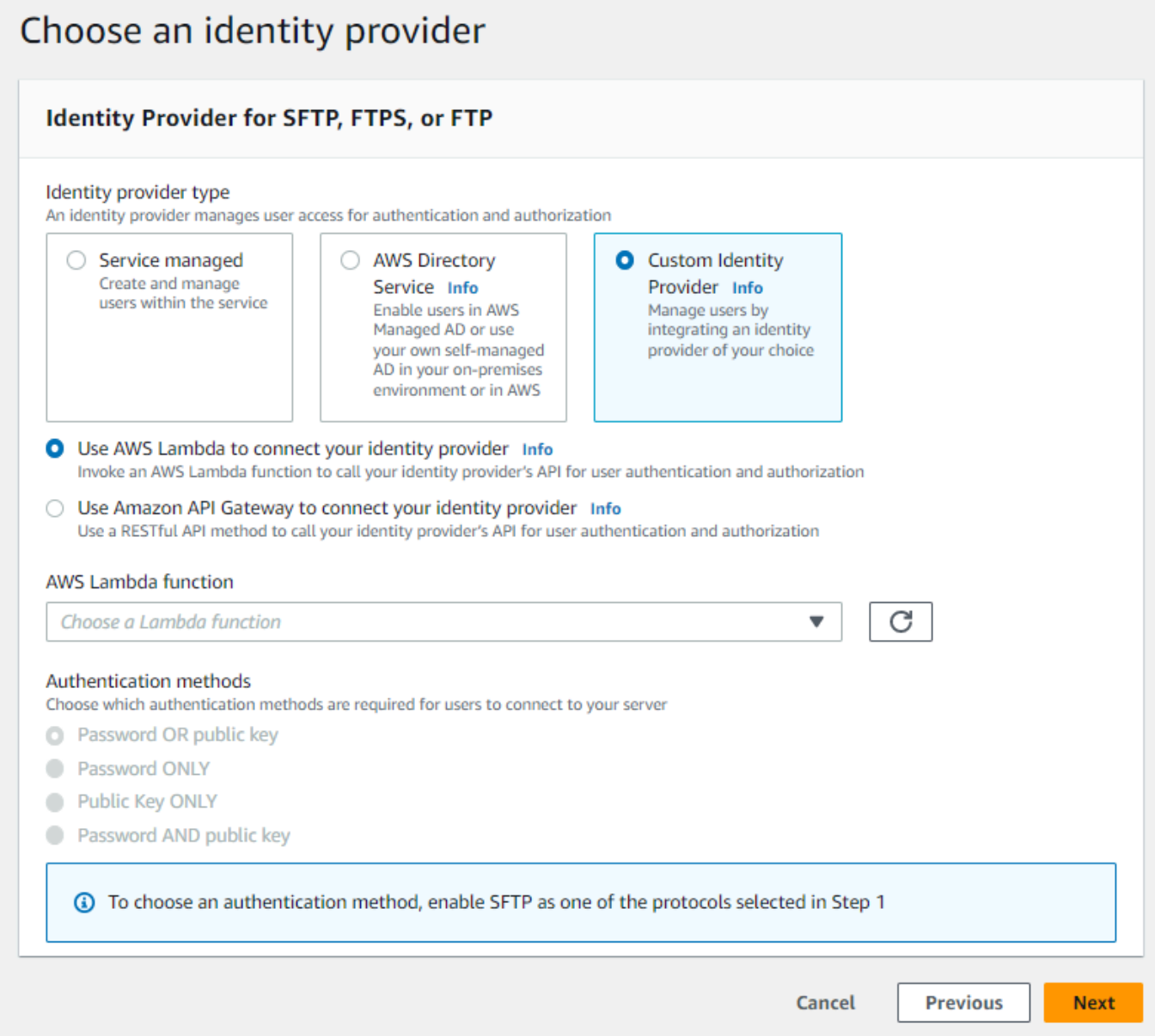
1. https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開き、ナビゲーションペインからサーバーを選択し、サーバーの作成を選択します。
2. [Choose protocols] (プロトコルの選択) で FTP を選択してから [Next] (次へ) を選択します。
3. [Choose an identity provider] (ID プロバイダーの選択) で、ユーザーアクセスの管理に使用したい ID プロバイダーを選択します。次のオプションがあります。

   - AWS Directory Service for Microsoft Active Directory – エンドポイントにアクセスするための AWS Directory Service ディレクトリを指定します。そうすることで、Active Directory に保存されている認証情報を使用してユーザーを認証できるようになります。 AWS Managed Microsoft AD ID プロバイダーの操作の詳細については、「」を参照してくださいAWS Directory Service ID プロバイダーの使用。

     > **Note**
     >
     > - クロスアカウントディレクトリと共有ディレクトリは、 ではサポートされていません AWS Managed Microsoft AD。
     > - Directory Service を ID プロバイダーとしてサーバーを設定するには、いくつかの AWS Directory Service アクセス許可を追加する必要があります。詳細については、「の使用を開始する前に AWS Directory Service for Microsoft Active Directory」を参照してください。

   - 「Custom Identity Provider」 ー 以下のいずれかのオプションを選択する：

- AWS Lambda を使用して ID プロバイダーを接続する – Lambda 関数にバックアップされた既存の ID プロバイダーを使用できます。Lambda 関数の名前を指定します。詳細については、「AWS Lambda を使用して ID プロバイダーを統合する」を参照してください。
- 「Amazon API Gateway を使って ID プロバイダーに接続する」 – ID プロバイダーとして使用するために、Lambda 関数に裏打ちされた API ゲートウェイメソッドを作成することができます。Amazon API Gateway URL と呼び出しロールを指定します。詳細については、「Amazon API Gateway を使用して ID プロバイダーを統合する」を参照してください。



4. [次へ] をクリックします。
5. [Choose an endpoint] (エンドポイントの選択) で、次のように操作します。

> **Note**
>
> Transfer Family の FTP サーバーは、ポート21（コントロールチャネル）とポートレンジ 8192-8200（データチャネル）で動作します。

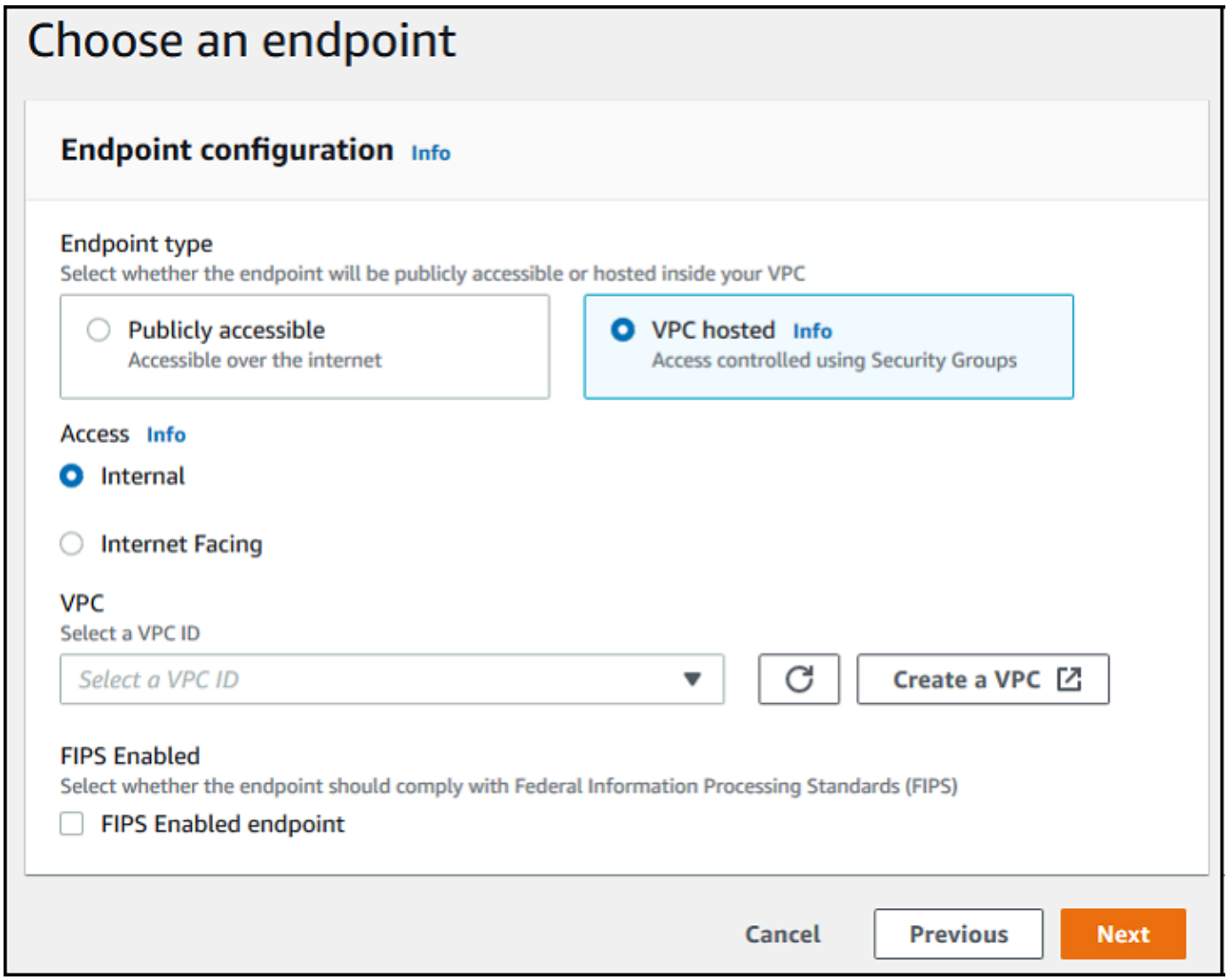
a. [Endpoint type] (エンドポイントタイプ) で、サーバーのエンドポイントのホストになる VPC ホストテッドエンドポイントタイプを選択します。VPC ホストエンドポイントの設定については、「Virtual Private Cloud でサーバーを作成する」を参照してください。

> **Note**
>
> パブリックにアクセス可能なエンドポイントはサポートされません。

b. [FIPS Enabled] (FIPS 対応) で [FIPS Enabled endpoint] (FIPS 対応エンドポイント) チェックボックスをオフのままにします。

> **Note**
>
> IPS 対応エンドポイントは、FTP サーバーではサポートされていません。

c. [次へ] をクリックします。



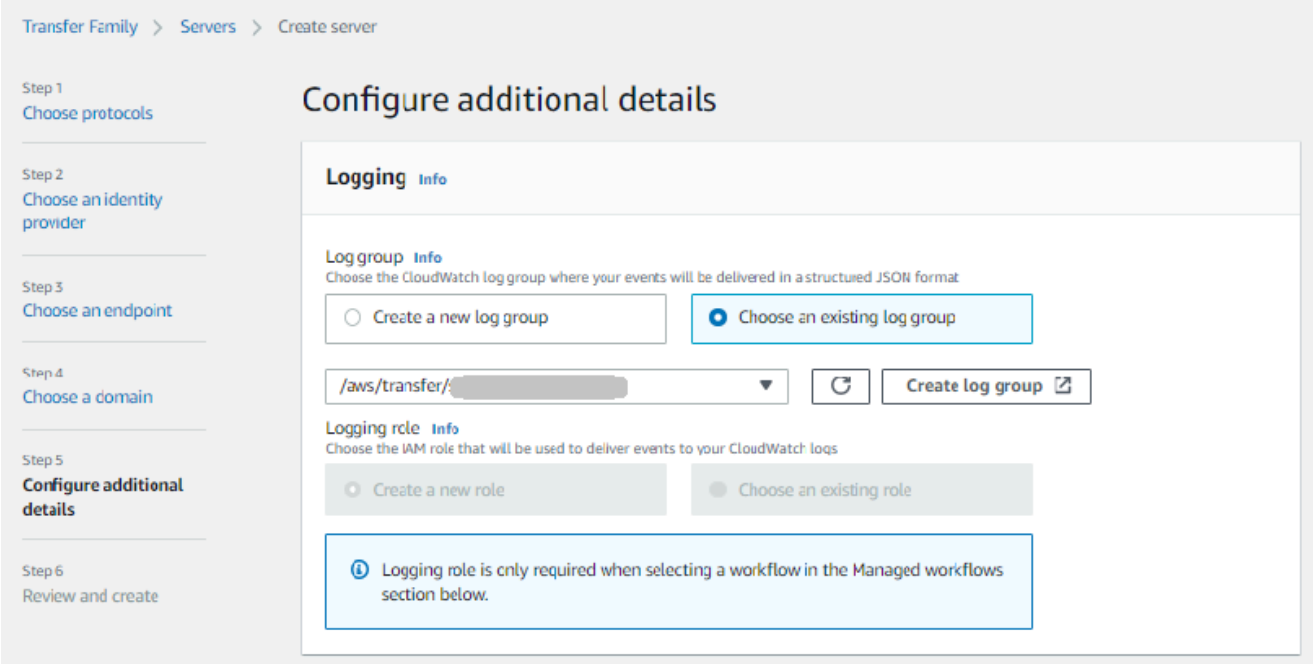
6. 「ドメインの選択」ページで、選択したプロトコルでデータを保存してアクセスするために使用する AWS ストレージサービスを選択します。

   - Amazon S3 を選択すると、選択したプロトコル経由でファイルをオブジェクトとして保存してアクセスできます。
   - Amazon EFS を選択すると、選択したプロトコル経由で Amazon EFS ファイルシステム内でファイルを保存してアクセスできます。

   [Next] (次へ) を選択します。

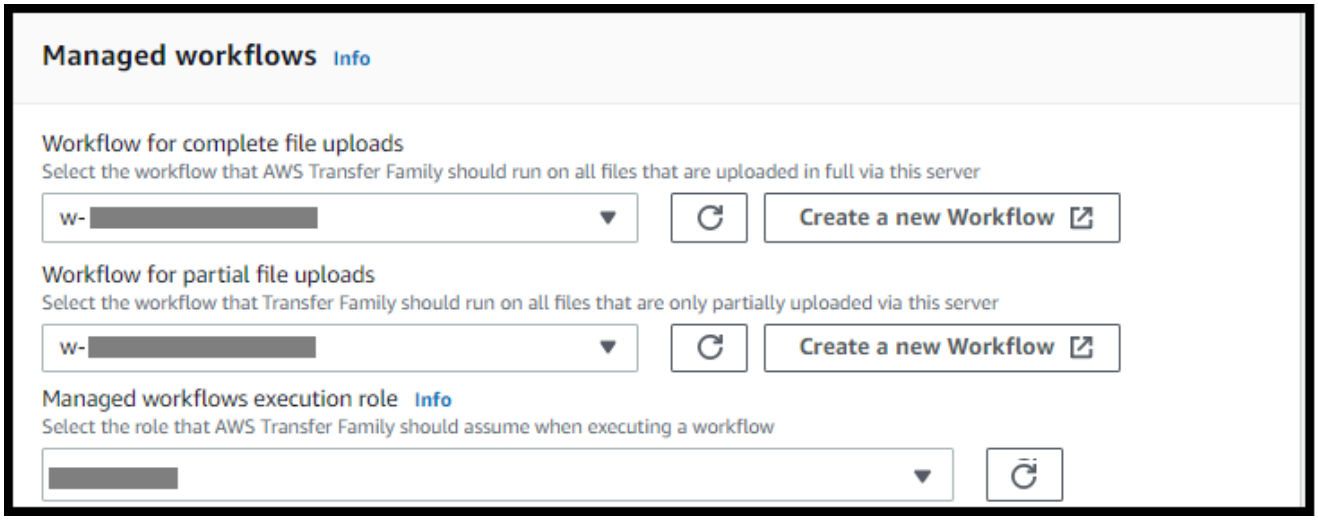
7. [Configure additional details] (その他の詳細の構成) で次のように操作します。

a. ログの場合、既存のログ グループを指定するか、新しいログ グループを作成します (デフォルト オプション)。



ロググループの作成 を選択すると、 CloudWatch コンソール (https://console.aws.amazon.com/cloudwatch/) がロググループの作成ページを開きます。詳細については、「 ログ」の CloudWatch 「ロググループの作成」を参照してください。

b. (オプション)[管理対象ワークフローの場合]、ワークフローの実行時に Transfer Family が引き受けるワークフロー ID (および対応するロール) を選択します。完全なアップロード時に実行するワークフローと、部分的なアップロード時に実行するワークフローを選択できます。マネージドワークフローを使用したファイルの処理の詳細については、AWS Transfer Family マネージドワークフローを参照してください。



c. [Cryptographic algorithm options] (暗号化アルゴリズムオプション) で、サーバーで使用できる暗号化アルゴリズムを含むセキュリティポリシーを選択します。

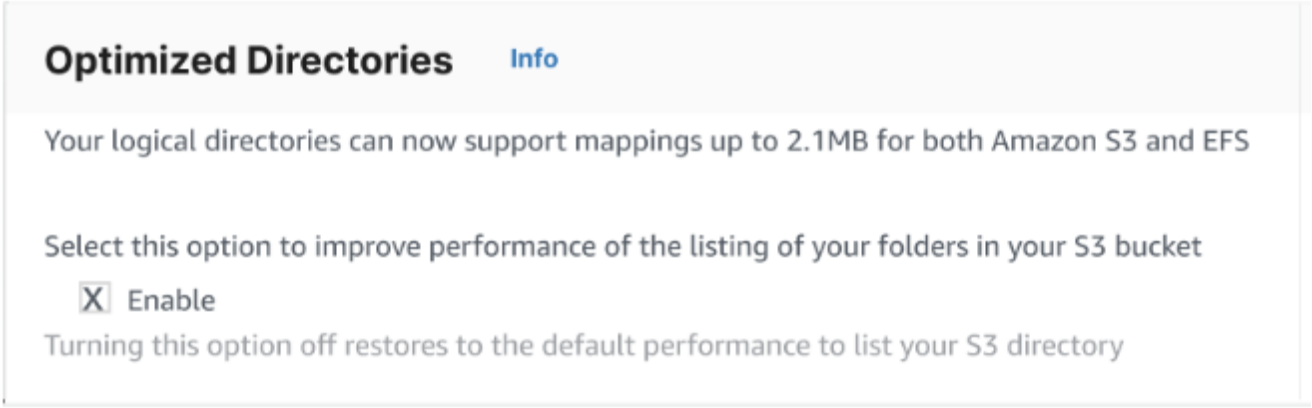
> **Note**
>
> Transfer Family は、FTP サーバーに最新のセキュリティポリシーを割り当てます。ただし、FTP プロトコルは暗号化を使用しないため、FTP サーバーはセキュリティポリシーアルゴリズムを使用しません。サーバーが FTPS または SFTP プロトコルも使用していない限り、セキュリティポリシーは使用されません。

d. 「サーバーホストキー」は空白のままにしておきます。

e. (オプション) [Tags] (タグ) の [Key] (キー) と [Value] (値) にキーバリューペアとして 1 つ以上のタグを入力して [Add tag] (タグの追加) を選択します。

f. Amazon S3 ディレクトリのパフォーマンスを最適化できます。例えば、ホームディレクトリに移動し、10,000 個のサブディレクトリがあるとします。つまり、Amazon S3 バケットには 10,000 個のフォルダがあります。このシナリオでは、`ls` (list) コマンドを実行すると、リストオペレーションに 6~8 分かかります。ただし、ディレクトリを最適化する場合、このオペレーションには数秒しかかかりません。

   コンソールを使用してサーバーを作成すると、最適化されたディレクトリがデフォルトで有効になります。API を使用してサーバーを作成する場合、この動作はデフォルトでは有効になっていません。



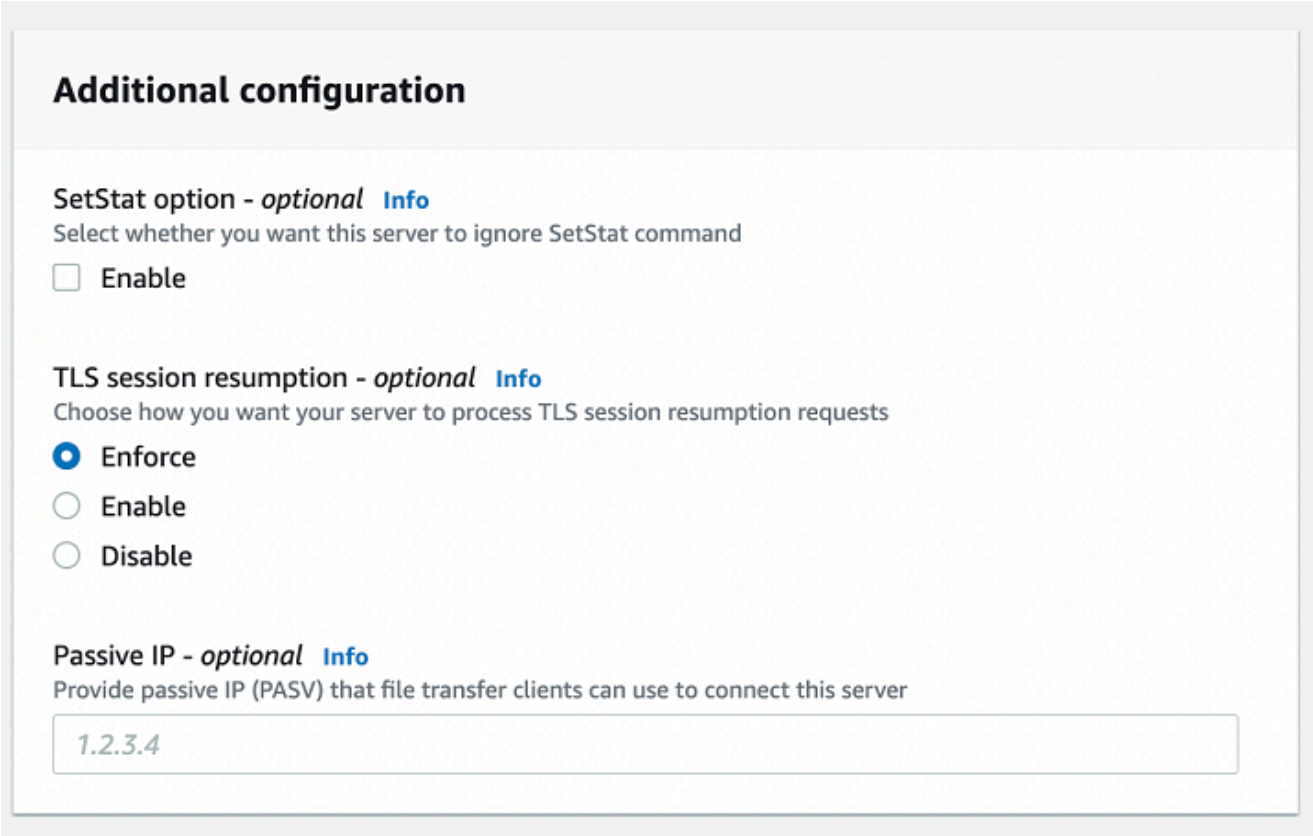
g. [次へ] をクリックします。

h. (オプション) 組織のポリシーや利用規約などのカスタマイズされたメッセージをエンドユーザーに表示するように AWS Transfer Family サーバーを設定できます。認証に成功したユーザーに対して、カスタマイズされた今日のメッセージ (MOTD) を表示することもできます。

[ディスプレイ バナー] の場合、[プリ認証ディスプレイ バナー] テキスト ボックスに、ユーザーが認証する前に表示したいテキスト メッセージを入力し、ポスト認証ディスプレイ バナー テキスト ボックスには、ユーザーが認証に成功した後に表示したいテキストを入力してください。

i. (オプション) 次の追加オプションを構成できます。

- SetStat オプション : このオプションを有効にすると、クライアントが Amazon S3 バケットにアップロードするファイルSETSTATで を使用しようとしたときに生成されるエラーを無視できます。詳細については、ProtocolDetails「」トピックのSetStatOptionドキュメントを参照してください。
- [TLS セッション再開]:FTPS セッションの制御接続とデータ接続の間でネゴシエートされた秘密キーを再開または共有するメカニズムを提供します。詳細については、ProtocolDetails「」トピックのTlsSessionResumptionModeドキュメントを参照してください。
- [Passive IP] : FTP および FTPS プロトコルのパッシブモードを示します。ファイアウォール、ルーター、ロードバランサーのパブリック IP アドレスなど、単一 IPv4 アドレスを入力します。詳細については、ProtocolDetails「」トピックのPassiveIpドキュメントを参照してください。



8. [Review and create] (確認と作成) で選択内容を確認します。

   - いずれかを編集するには、ステップの隣にある [Edit] (編集) を選択します。

     > **Note**
     >
     > 編集を選んだステップの後に、各ステップを見直す必要があります。

   - 変更がない場合、[Create server] (サーバーの作成) を選択してサーバーを作成します。[Servers] (サーバー) ページに誘導され、次に示すように新しいサーバーの一覧が表示されます。

新しい SFTP サーバーのステータスが [Online] (オンライン) に変わるまでに数分かかる場合があります。その時点で、サーバーはユーザーのためにファイルオペレーションを実行できます。

次のステップ — 次のステップとして、「カスタム ID プロバイダーの使用」に進んでユーザーを設定します。

Virtual Private Cloud でサーバーを作成する

Virtual Private Cloud (VPC) 内でサーバーのエンドポイントをホストし、それを使用してパブリックインターネットを経由せずに Amazon S3 バケットや Amazon EFS ファイルシステムとの間でデータを転送できます。

> **Note**
>
> 2021 年 5 月 19 日以降、 AWS 2021 年 5 月 19 日より前にアカウントEndpointType=VPC_ENDPOINTで を使用してサーバーを作成していない場合、そのサーバーを作成することはできません。2021 年 2 月 21 日以前に AWS アカウントEndpointType=VPC_ENDPOINTで を使用して既にサーバーを作成している場合は、影響を受けません。この日付を過ぎたら EndpointType=**VPC** を使用します。詳細については、「the section called “VPC_ENDPOINT のサポート終了”」を参照してください。

Amazon Virtual Private Cloud (Amazon VPC) を使用して AWS リソースをホストする場合、VPC とサーバーの間にプライベート接続を確立できます。このサーバーを使用すれば、パブリック IP アドレスを使用したり、インターネットゲートウェイを必要とすることなく、クライアント経由で Amazon S3 バケットとの間でデータを転送できます。

Amazon VPC を使用すると、カスタム仮想ネットワークで AWS リソースを起動できます。VPCを使用して、IP アドレス範囲、サブネット、ルートテーブル、ネットワークゲートウェイなどのネットワーク設定を制御できます。Amazon VPC の詳細については、Amazon VPC ユーザーガイドの「Amazon VPC とは」を参照してください。

以下のセクションでは、VPC をサーバーに接続する手順を示しています。手順の概要は、以下のとおりです。

1. VPC エンドポイントを使用してサーバーを設定します。
2. VPC エンドポイントを通して VPC 内の SFTP クライアントでサーバーに接続します。これにより、 AWS Transfer Familyを使用して Amazon S3 バケットに保存されたデータをクライアント経由で転送できます。ネットワークがパブリックインターネットから切断されていても、この転送を実行できます。
3. さらに、サーバーのエンドポイントをインターネット向けにしようとする場合、Elastic IP アドレスをエンドポイントに関連付けることができます。これで VPC の外部のクライアントがサーバーに接続できるようになります。VPC セキュリティグループを使用して、許可されたアドレスのみからリクエストが送信される認証済みユーザーへのアクセスを制御できます。

トピック

- VPC 内でのみアクセスできるサーバーエンドポイントを作成する
- サーバー用のインターネット向けエンドポイントを作成する
- SFTP サーバーのエンドポイントタイプの変更
- VPC_ENDPOINT のサポート終了
- AWS Transfer Family サーバーエンドポイントタイプを VPC_ENDPOINT から VPC に更新する

VPC 内でのみアクセスできるサーバーエンドポイントを作成する

以下の手順に従って、VPC 内のリソースにのみアクセスできるサーバーエンドポイントを作成します。

VPC の内部にエンドポイントを作成するには

1. https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開きます。
2. ナビゲーションペインで、[Servers] (サーバー) を選択してから [Create server] (サーバーの作成) を選択します。
3. [Choose protocols] (プロトコルの選択) で 1 つ以上のプロトコルを選択してから [Next] (次へ) を選択します。プロトコルの詳細については、「ステップ 2: SFTP 対応サーバーを作成する」を参照してください。
4. ID プロバイダーを選択 で、サービスマネージド を選択してユーザー ID とキーを に保存し AWS Transfer Family、次へ を選択します。

   > **Note**
   >
   > この手順では、サービスマネージドオプションを使用します。[Custom] (カスタム) を選択した場合、 Amazon API Gateway エンドポイントにアクセスするために Amazon API Gateway エンドポイントと AWS Identity and Access Management (IAM) ロールを提供します。そうすることで、ディレクトリサービスを統合して SFTP ユーザーの認証と許可を実行できます。カスタム ID プロバイダーの使用に関する詳細については、「カスタム ID プロバイダーの使用」を参照してください。

5. [Choose an endpoint] (エンドポイントの選択) で、次のように操作します。

> **Note**
>
> Transfer Family 用の FTP および FTPS サーバーは、ポート 21 (制御チャネル) およびポート範囲 8192 ~ 8200 (データチャネル) を経由して動作します。

a. [Endpoint type] (エンドポイントタイプ) で、サーバーのエンドポイントをホストする VPC hosted (VPC ホステッド) エンドポイントタイプを選択します。

b. [Access] (アクセス) で [Internal] (社内) すると、エンドポイントのプライベート IP アドレスを使用しているクライアントのみがエンドポイントにアクセスできるようになります。

> **Note**
>
> Internet Facing (インターネット向け) オプションの詳細については、「サーバー用のインターネット向けエンドポイントを作成する」を参照してください。社内アクセスのみの VPC 内で作成されたサーバーはカスタムホスト名をサポートしません。

c. VPC では、既存の VPC ID を選択するか、[Create a VPC] (VPC を作成する) をクリックして新しい VPC を作成します。

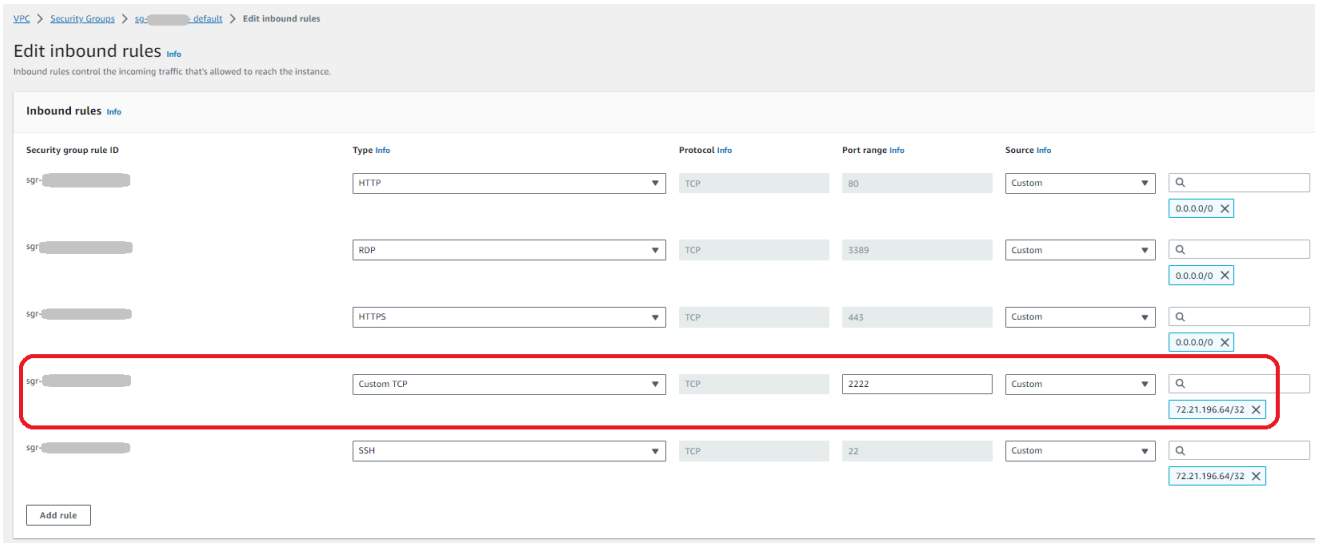
d. [Availability Zones] (アベイラビリティーゾーン) セクションで、最大 3 つのアベイラビリティーゾーンと関連サブネットを選択します。

e. [Security Groups] (セキュリティグループ) セクションで、既存のセキュリティグループ ID を選択するか、[Create a security group] (セキュリティグループの作成) をクリックして新しいセキュリティグループを作成します。VPC セキュリティグループの詳細については、Amazon Virtual Private Cloud ユーザーガイドの「VPC のセキュリティグループ」を参照してください。セキュリティグループを作成するには、Amazon Virtual Private Cloud ユーザーガイドの「セキュリティグループの作成」を参照してください。

> **Note**
>
> VPC にはデフォルトセキュリティグループが付属していて自動的に設定されます。サーバーの起動時に別のセキュリティグループを指定しない場合、デフォルトのセキュリティグループがサーバーに関連付けられます。

セキュリティグループのインバウンドルールでは、ポート 22、2222、22000、または任意の組み合わせを使用するように SSH トラフィックを設定できます。ポート 22 はデフォルトで設定されています。ポート 2222 またはポート 22000 を使用するには、セキュリティグループにインバウンドルールを追加します。タイプにはカスタム TCP を選択し、ポート範囲 **22000**には **2222**または を入力し、ソースには SSH ポート 22 ルールと同じ CIDR 範囲を入力します。

> **Note**
>
> また、TCP の「piggy-backACKs、または TCP 3 ウェイハンドシェイクの最終確認機能にデータを含める機能にポート 2223 を使用することもできます。
> 一部のクライアントソフトウェアは、ポート 2223 と互換性がない場合があります。例えば、クライアントが送信する前にサーバーが SFTP 識別文字列を送信する必要があるクライアントなどです。



f. (オプション) [FIPS Enabled] (FIPS 対応) で、[FIPS Enabled endpoint] (FIPS 対応エンドポイント) チェックボックスをオンにして、エンドポイントが連邦情報処理標準 (FIPS) に準拠していることを確認します。

> **Note**
>
> FIPS 対応エンドポイントは、北米 AWS リージョンでのみ使用できます。利用可能なリージョンについては、「AWS 全般のリファレンス」の「AWS Transfer Family

エンドポイントとクォータ」を参照してください。FIPS 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。

g. [Next] (次へ) を選択します。

6. [Configure additional details] (その他の詳細の構成) で次のように操作します。

a. CloudWatch ログ記録 では、次のいずれかを選択して、ユーザーアクティビティの Amazon CloudWatch ログ記録を有効にします。

- 新しいロールを作成するための正しいアクセス許可を受けている場合、[Create a new role] (新しいロールを作成する) を選択すれば Transfer Family で自動的に IAM ロールを作成できます。作成される IAM ロールは `AWSTransferLoggingAccess` と呼ばれます。
- [Choose an existing role] (既存のロールを選択) でアカウントから既存のロール を選択します。[Logging role] (ログ記録ロール) でロールを選択します。この IAM ロールには [Service] (サービス) を `transfer.amazonaws.com` に設定した信頼ポリシーを含めてください。

  CloudWatch ログ記録の詳細については、「」を参照してください CloudWatch ログ記録ロールを設定する。

> **Note**
>
> - ログ記録ロールを指定 CloudWatch しない場合、 でエンドユーザーアクティビティを表示することはできません。
> - CloudWatch ログ記録ロールをセットアップしない場合は、既存のロールを選択を選択しますが、ログ記録ロールは選択しません。

b. [Cryptographic algorithm options] (暗号化アルゴリズムオプション) で、サーバーで使用できる暗号化アルゴリズムを含むセキュリティポリシーを選択します。

> **Note**
>
> デフォルトでは、`TransferSecurityPolicy-2020-06` セキュリティポリシーは、別のポリシーを選択しない限り、サーバーに関連付けられます。

セキュリティポリシーの詳細については、「AWS Transfer Family サーバーのセキュリティポリシー」を参照してください。

c. （オプション: このセクションは、既存の SFTP 対応サーバーからユーザーを移行する場合のみです）。サーバーホストキー には、クライアントが SFTP 経由でサーバーに接続するときにサーバーを識別するために使用される RSA、ED25519、または ECDSA プライベートキーを入力します。

d. (オプション) [Tags] (タグ) の [Key] (キー) と [Value] (値) にキーバリューペアとして 1 つ以上のタグを入力して [Add tag] (タグの追加) を選択します。

e. [次へ] をクリックします。

7. [Review and create] (確認と作成) で選択内容を見直します。オプション:

- いずれかを編集するには、ステップの隣にある [Edit] (編集) を選択します。

> **Note**
>
> 編集するステップの後に、各ステップを確認する必要があります。

- 変更の必要がなければ、[Create server] (サーバーの作成) を選択してサーバーを作成します。次に示すとおり、新しいサーバーが一覧表示されている、[Servers] (サーバー) ページに移動します。

新しい SFTP サーバーのステータスが [Online] (オンライン) に変わるまでに数分かかる場合があります。その時点で、サーバーでファイルオペレーションを実行できますが、まずユーザーを作成する必要があります。ユーザーの作成の詳細については、「」を参照してくださいサーバーエンドポイントのユーザーの管理。

サーバー用のインターネット向けエンドポイントを作成する

以下の手順に従って、サーバーエンドポイントを作成します。このエンドポイントは、VPC のデフォルトセキュリティグループで送信元 IP アドレスが許可されているクライアントに対してのみ、インターネット経由でアクセスできます。さらに、Elastic IP アドレスを使用してエンドポイントをインターネット向けにすることで、クライアントは Elastic IP アドレスを使用してファイアウォール内のエンドポイントへのアクセスを許可できます。

> **Note**
>
> インターネット向け VPC でホストされるエンドポイントで使用できるのは SFTP と FTPS のみです。

インターネット向けエンドポイントを作成するには

1. https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開きます。
2. ナビゲーションペインで、[Servers] (サーバー) を選択してから [Create server] (サーバーの作成) を選択します。
3. [Choose protocols] (プロトコルの選択) で 1 つ以上のプロトコルを選択してから [Next] (次へ) を選択します。プロトコルの詳細については、「ステップ 2: SFTP 対応サーバーを作成する」を参照してください。
4. ID プロバイダーを選択 で、サービスマネージド を選択してユーザー ID とキーを に保存し AWS Transfer Family、次へ を選択します。

   > **Note**
   >
   > この手順では、サービスマネージドオプションを使用します。[Custom] (カスタム) を選択した場合、 Amazon API Gateway エンドポイントにアクセスするために Amazon API Gateway エンドポイントと AWS Identity and Access Management (IAM) ロールを提供します。そうすることで、ディレクトリサービスを統合して SFTP ユーザーの認証と許可を実行できます。カスタム ID プロバイダーの使用に関する詳細については、「カスタム ID プロバイダーの使用」を参照してください。

5. [Choose an endpoint] (エンドポイントの選択) で、次のように操作します。

   a. [Endpoint type] (エンドポイントタイプ) で、サーバーのエンドポイントをホストする VPC hosted (VPC ホステッド) エンドポイントタイプを選択します。

   b. [Access] (アクセス) で [Internet Facing] (インターネット向け) をクリックして、インターネット経由でクライアントからエンドポイントにアクセスできるようにします。

      > **Note**
      >
      > [Internet Facing] (インターネット向け) を選択すると、各サブネット内の既存の Elastic IP アドレスを選択できます。または、VPC コンソール (https://

console.aws.amazon.com/vpc/) に移動して、1 つ以上の新しい Elastic IP アドレスを割り当てることもできます。これらのアドレスは、 AWS または が所有できます。既に使用されている Elastic IP アドレスをエンドポイントに関連付けることはできません。

c. (オプション) [Custom hostname (カスタムホスト名) で以下のいずれかのオプションを選択します。

> **Note**
>
> AWS GovCloud (US) のお客様は、Elastic IP アドレス経由で直接接続するか、EIP を指すホスト名レコードを Commercial Route 53 内に作成する必要があります。GovCloud エンドポイントに Route 53 を使用する方法の詳細については、「 ユーザーガイド」の「 AWS GovCloud (US) リソースでの Amazon Route 53 のセットアップAWS GovCloud (US) 」を参照してください。

- Amazon Route 53 DNS alias (Amazon Route 53 DNS エイリアス) – 使用するホスト名が、Route 53 に登録されている場合。その後、ホスト名を入力します。
- Other DNS (その他の DNS) – 使用するホスト名が別の DNS プロバイダーに登録されている場合。その後、ホスト名を入力します。
- None (なし) – サーバーのエンドポイントを使用し、カスタムホスト名は使用しない場合。サーバーホスト名の形式は *server-id*.server.transfer.*region*.amazonaws.com です。

> **Note**
>
> のお客様の場合 AWS GovCloud (US)、None を選択しても、この形式でホスト名は作成されません。

カスタムホスト名の使用に関する詳細については、「カスタムホスト名の使用」を参照してください 。

d. VPC では、既存の VPC ID を選択するか、[Create a VPC] (VPC を作成する) をクリックして新しい VPC を作成します。

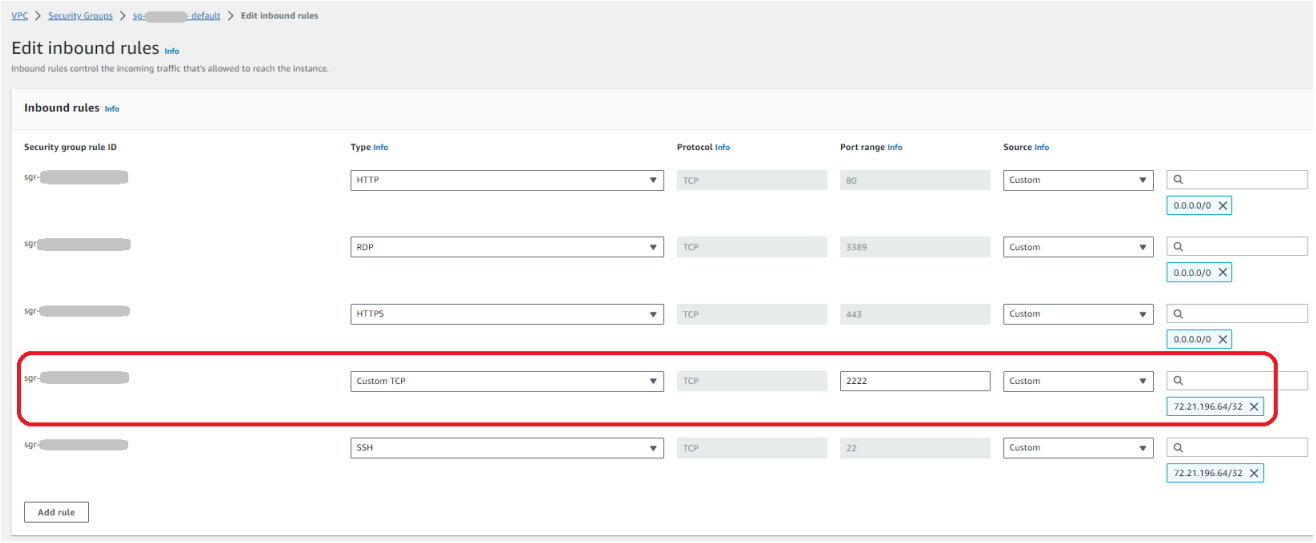
e. [Availability Zones] (アベイラビリティーゾーン) セクションで、最大 3 つのアベイラビリティーゾーンと関連サブネットを選択します。[IPv4 Addresses] (IPv4 アドレス) でサブネットごとに Elastic IP アドレスを選択します。これは、クライアントがファイアウォール内のエンドポイントへのアクセスを許可するために使用できる IP アドレスです。

f. [Security Groups] (セキュリティグループ) セクションで、既存のセキュリティグループ ID を選択するか、[Create a security group] (セキュリティグループの作成) をクリックして新しいセキュリティグループを作成します。VPC セキュリティグループの詳細については、Amazon Virtual Private Cloud ユーザーガイドの「VPC のセキュリティグループ」を参照してください。セキュリティグループを作成するには、Amazon Virtual Private Cloud ユーザーガイドの「セキュリティグループの作成」を参照してください。

> **Note**
>
> VPC にはデフォルトセキュリティグループが付属していて自動的に設定されます。サーバーの起動時に別のセキュリティグループを指定しない場合、デフォルトのセキュリティグループがサーバーに関連付けられます。

セキュリティグループのインバウンドルールでは、ポート 22、2222、22000、または任意の組み合わせを使用するように SSH トラフィックを設定できます。ポート 22 はデフォルトで設定されています。ポート 2222 またはポート 22000 を使用するには、セキュリティグループにインバウンドルールを追加します。タイプにはカスタム TCP を選択し、ポート範囲 **22000**には **2222**または を入力し、ソースには SSH ポート 22 ルールと同じ CIDR 範囲を入力します。

> **Note**
>
> また、TCP の「piggy-backACKs、または TCP 3 ウェイハンドシェイクの最終確認機能にデータを含める機能にポート 2223 を使用することもできます。
> 一部のクライアントソフトウェアは、ポート 2223 と互換性がない場合があります。例えば、クライアントが送信する前にサーバーが SFTP 識別文字列を送信する必要があるクライアントなどです。



g. (オプション) [FIPS Enabled] (FIPS 対応) で、[FIPS Enabled endpoint] (FIPS 対応エンドポイント) チェックボックスをオンにして、エンドポイントが連邦情報処理標準 (FIPS) に準拠していることを確認します。

> **Note**
>
> FIPS 対応エンドポイントは、北米 AWS リージョンでのみ使用できます。利用可能なリージョンについては、「AWS 全般のリファレンス」の「AWS Transfer Family エンドポイントとクォータ」を参照してください。FIPS 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。

h. [Next] (次へ) を選択します。

6. [Configure additional details] (その他の詳細の構成) で次のように操作します。

a. CloudWatch ログ記録 では、次のいずれかを選択して、ユーザーアクティビティの Amazon CloudWatch ログ記録を有効にします。

- 新しいロールを作成するための正しいアクセス許可を受けている場合、[Create a new role] (新しいロールを作成する) を選択すれば Transfer Family で自動的に IAM ロールを作成できます。作成される IAM ロールは `AWSTransferLoggingAccess` と呼ばれます。
- [Choose an existing role] (既存のロールを選択) でアカウントから既存のロール を選択します。[Logging role] (ログ記録ロール) でロールを選択します。この IAM ロールには [Service] (サービス) を `transfer.amazonaws.com` に設定した信頼ポリシーを含めてください。

CloudWatch ログ記録の詳細については、「」を参照してください CloudWatch ログ記録ロールを設定する。

> **Note**
>
> - ログ記録ロールを指定 CloudWatch しない場合、 でエンドユーザーアクティビティを表示することはできません。
> - CloudWatch ログ記録ロールをセットアップしない場合は、既存のロールを選択を選択しますが、ログ記録ロールは選択しません。

b. [Cryptographic algorithm options] (暗号化アルゴリズムオプション) で、サーバーで使用できる暗号化アルゴリズムを含むセキュリティポリシーを選択します。

> **Note**
>
> デフォルトでは、`TransferSecurityPolicy-2020-06` セキュリティポリシーは、別のポリシーを選択しない限り、サーバーに関連付けられます。

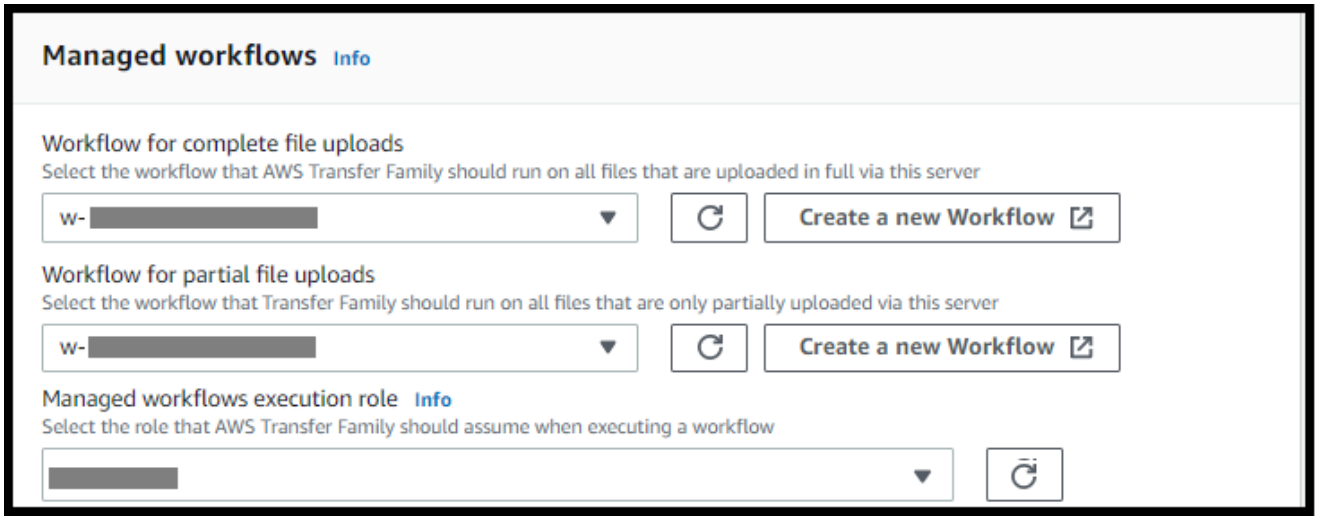
セキュリティポリシーの詳細については、「AWS Transfer Family サーバーのセキュリティポリシー」を参照してください。

c. (オプション: このセクションは、既存の SFTP 対応サーバーからユーザーを移行する場合のみです) 。サーバーホストキー には、クライアントが SFTP 経由でサーバーに接続するときにサーバーを識別するために使用される RSA、ED25519、または ECDSA プライベートキーを入力します。

d. (オプション) [Tags] (タグ) の [Key] (キー) と [Value] (値) にキーバリューペアとして 1 つ以上のタグを入力して [Add tag] (タグの追加) を選択します。

e. [次へ] をクリックします。

f. (オプション)[管理対象ワークフローの場合]、ワークフローの実行時に Transfer Family が引き受けるワークフロー ID (および対応するロール) を選択します。完全なアップロード時に実行するワークフローと、部分的なアップロード時に実行するワークフローを選択できます。マネージドワークフローを使用したファイルの処理の詳細については、AWS Transfer Family マネージドワークフローを参照してください。



7. [Review and create] (確認と作成) で選択内容を見直します。オプション:

   - いずれかを編集するには、ステップの隣にある [Edit] (編集) を選択します。

     > **Note**
     >
     > 編集するステップの後に、各ステップを確認する必要があります。

   - 変更の必要がなければ、[Create server] (サーバーの作成) を選択してサーバーを作成します。次に示すとおり、新しいサーバーが一覧表示されている、[Servers] (サーバー) ページに移動します。

サーバー ID を選択して、作成したサーバーのデフォルト設定を確認できます。[Public IPv4 address] (パブリック IPv4 アドレス) 列にアドレスが入力された後、指定した Elastic IP アドレスがサーバーのエンドポイントに正常に関連付けられます。

> **Note**
>
> VPC 内のサーバーがオンラインの場合、サブネットのみを変更でき、UpdateServerAPI を介してのみ変更できます。サーバーエンドポイントの Elastic IP アドレスを追加または変更するには、サーバーを停止する必要があります。

SFTP サーバーのエンドポイントタイプの変更

インターネット経由でアクセス可能な既存のサーバー (パブリックエンドポイントタイプを持つサーバー) がある場合 、そのエンドポイントを VPC エンドポイントに変更できます。

> **Note**
>
> VPC_ENDPOINT として表示される VPC 内に既存のサーバーがある場合、新しい VPC エンドポイントタイプに変更することをお勧めします。この新しいエンドポイントタイプでは、Network Load Balancer (NLB) を使用して Elastic IP アドレスをサーバーのエンドポイントに関連付ける必要がなくなりました。また、VPC セキュリティグループを使用して、サーバーのエンドポイントへのアクセスを制限できます。ただし、必要に応じて VPC_ENDPOINT エンドポイントタイプを引き続き使用できます。

次の手順では、現在のパブリックエンドポイントタイプまたは古い VPC_ENDPOINT エンドポイントタイプのいずれかを使用するサーバーがあることを前提にしています。

サーバーのエンドポイントタイプを変更するには

1. https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開きます。
2. ナビゲーションペインで [Servers] (サーバー) を選択します。
3. エンドポイントタイプを変更したいサーバーのチェックボックスをオンにします。

   > **Important**
   >
   > エンドポイントを変更するには、その前にサーバーを停止する必要があります。

4. [Actions] (アクション) で [Stop] (停止) を選択します。
5. 表示される確認ダイアログボックスで [Stop] (停止) を選択して、サーバーを停止したいことを確認します。

   > **Note**
   >
   > 次のステップに進む前に、[Endpoint details] (エンドポイントの詳細) でサーバーの [Status] (ステータス) が [Offline] (オフライン) に変わるのを待ちますが、これには数分かかる場合があります。ステータスの変更を見るには、[Servers] (サーバー) ページで [Refresh] (更新) の選択が必要になる場合があります。

サーバーが [Offline] (オフライン) になるまで編集はできません。

6. [Endpoint details] (エンドポイントの詳細) で [Edit] (編集) を選択します。
7. [Edit endpoint configuration] (エンドポイント設定の編集) で次のように操作します。

   a. [Edit endpoint type] (エンドポイントタイプ) で [VPC hosted] (VPC ホステッド) を選択します。

   b. [Access] (アクセス) について以下のいずれかを選択します。

      - [Internal] (社内) を選択すると、エンドポイントのプライベート IP アドレスを使用しているクライアントのみがエンドポイントにアクセスできるようになります。
      - [Internet Facing] (インターネット向け) を選択すると、パブリックインターネット経由でクライアントがエンドポイントにアクセスできるようになります。

      > **Note**
      >
      > [Internet Facing] (インターネット向け) を選択すると、各サブネット内の既存の Elastic IP アドレスを選択できます。または、VPC コンソール (https://console.aws.amazon.com/vpc/) に移動して、1 つ以上の新しい Elastic IP アドレスを割り当てることもできます。これらのアドレスは、 AWS または が所有できます。既に使用されている Elastic IP アドレスをエンドポイントに関連付けることはできません。

   c. (インターネット向けアクセスの場合のみのオプション) [Custom hostname] (カスタムホスト名) で、以下のいずれかのオプションを選択します。

      - Amazon Route 53 DNS alias (Amazon Route 53 DNS エイリアス) – 使用するホスト名が、Route 53 に登録されている場合。その後、ホスト名を入力します。
      - Other DNS (その他の DNS) – 使用するホスト名が別の DNS プロバイダーに登録されている場合。その後、ホスト名を入力します。
      - None (なし) – サーバーのエンドポイントを使用し、カスタムホスト名は使用しない場合。サーバーホスト名の形式は *serverId*.server.transfer.*regionId*.amazonaws.com です。

        カスタムホスト名の使用に関する詳細については、「カスタムホスト名の使用」を参照してください 。

d. VPC では、既存の VPC ID を選択するか、[Create a VPC] (VPC を作成する) をクリックして新しい VPC を作成します。

e. [Availability Zones] (アベイラビリティーゾーン) セクションで、最大 3 つのアベイラビリティーゾーンと関連サブネットを選択します。[Internet Facing] (インターネット向け) が選択されている場合、サブネットごとに Elastic IP アドレスを選択します。

> **Note**
>
> アベイラビリティーゾーンは最大 3 つまで使用でき、足りない場合には VPC コンソール (https://console.aws.amazon.com/vpc/) で作成します。
> サブネットまたは Elastic IP アドレスを変更した場合、サーバーの更新に数分かかります。サーバーの更新が完了するまで、変更内容を保存することはできません。

f. [Save] (保存) を選択します。

8. [Actions] (アクション) で [Start] (開始) を選択してサーバーのステータスが [Online] (オンライン) になるのを待ちますが、これには数分かかる場合があります。

> **Note**
>
> パブリックエンドポイントタイプを VPC エンドポイントタイプに変更した場合、サーバーの [Endpoint type] (エンドポイントタイプ) が VPC に変わることに着目してください。

デフォルトのセキュリティグループがエンドポイントにアタッチされます。セキュリティグループを変更または新しく追加するには、「セキュリティグループの作成」を参照してください。

VPC_ENDPOINT のサポート終了

AWS Transfer Family は、新しい AWS アカウント`EndpointType=VPC_ENDPOINT`用に でサーバーを作成する機能を中止します。2021 年 5 月 19 日現在、エンドポイントタイプが の AWS Transfer Family サーバーを所有していない AWS アカウント`VPC_ENDPOINT`は、 で新しいサーバーを作成できません`EndpointType=VPC_ENDPOINT`。`VPC_ENDPOINT` エンドポイントタイプを使用するサーバーを既に所有している場合、できる限り早急に `EndpointType=VPC` の使用を開始することをお勧めします。詳細については、AWS Transfer Family 「サーバーエンドポイントタイプを VPC_ENDPOINT から VPC に更新する」を参照してください。

2020 年に新しい VPC エンドポイントタイプを発表いたしました。詳細については、「AWS Transfer Family for SFTP で VPC セキュリティグループと Elastic IP アドレスが利用可能に」を参照してください。この新しいエンドポイントは、より機能豊富で費用対効果が高く、 PrivateLink 料金はかかりません。詳細については、「 のAWS PrivateLink 料金」を参照してください。

このエンドポイントタイプは、以前のエンドポイントタイプ (VPC_ENDPOINT) と機能的に同等です。Elastic IP アドレスをエンドポイントに直接アタッチして、インターネット向けにし、送信元 IP フィルタリングにセキュリティグループを使用できます。詳細については、「IP 許可リストを使用して for SFTP サーバーの を保護する AWS Transfer Family」ブログ記事を参照してください。

共有 VPC 環境でこのエンドポイントをホストすることもできます。詳細については、「AWS Transfer Family が共有サービスの VPC 環境をサポート」を参照してください。

SFTP に加えて VPC EndpointType を使用して FTPS と FTP を有効にすることもできます。これらの機能と FTPS/FTP サポートを EndpointType=VPC_ENDPOINT に追加する予定はありません。また、コンソールからこのエンドポイントタイプをオプションとして削除しました AWS Transfer Family 。

Transfer Family コンソール、API、 SDKs AWS CLI、または を使用して、サーバーのエンドポイントタイプを変更できます AWS CloudFormation。サーバーのエンドポイントタイプを変更するには、「AWS Transfer Family サーバーエンドポイントタイプを VPC_ENDPOINT から VPC に更新する」を参照してください。

ご質問がある場合は、 AWS Support または AWS アカウントチームにお問い合わせください。

> **Note**
>
> これらの機能や FTPS または FTP サポートを EndpointType=VPC_ENDPOINT に追加する予定はありません。 AWS Transfer Family コンソールのオプションとして提供されなくなりました。

他にご質問がございましたら、 AWS Support または アカウントチームにお問い合わせください。

AWS Transfer Family サーバーエンドポイントタイプを VPC_ENDPOINT から VPC に更新する

AWS Management Console、、または Transfer Family API を使用して AWS CloudFormation、サーバーの を EndpointTypeから VPC_ENDPOINTに更新できますVPC。以下のセクションでは、これらの各メソッドを使用してサーバーエンドポイントタイプを更新するための詳細な手順と例について

説明します。複数の AWS リージョンと複数の AWS アカウントにサーバーがある場合は、次のセクションで提供されるスクリプト例を変更して使用して、更新する必要があるVPC_ENDPOINTタイプを使用するサーバーを識別できます。

トピック

- VPC_ENDPOINT エンドポイントタイプを使用したサーバーの識別
- を使用したサーバーエンドポイントタイプの更新 AWS Management Console
- を使用したサーバーエンドポイントタイプの更新 AWS CloudFormation
- API EndpointType を使用したサーバーの更新

VPC_ENDPOINT エンドポイントタイプを使用したサーバーの識別

AWS Management Consoleを使用することで、どのサーバーで VPC_ENDPOINT を使用しているかを特定できます。

VPC_ENDPOINT エンドポイントタイプを使用するサーバーをコンソールで識別するには

1. https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開きます。
2. ナビゲーションペインの [Servers] (サーバー) で、そのリージョンのアカウント内のサーバーの一覧を表示します。
3. [Endpoint type] (エンドポイントタイプ) を使用してサーバーのリストを並べ替えると、VPC_ENDPOINT を使用してすべてのサーバーを見ることができます。

複数の AWS リージョンとアカウントVPC_ENDPOINTで を使用しているサーバーを特定するには

複数の AWS リージョンと複数の AWS アカウントにサーバーがある場合は、次のスクリプト例を変更して、VPC_ENDPOINTエンドポイントタイプを使用してサーバーを識別できます。サンプルスクリプトは、Amazon EC2 DescribeRegionsと Transfer Family ListServers API コールを使用して、を使用するすべてのサーバーのサーバー IDsとリージョンのリストを取得しますVPC_ENDPOINT。AWS アカウントが多数ある場合は、ID プロバイダーへのセッションプロファイルを使用して認証する場合、読み取り専用の監査アクセス権を持つ IAM ロールを使用してアカウントをループスルーできます。

1. 以下に単純な例を示します。

```
import boto3
```

```
profile = input("Enter the name of the AWS account you'll be working in: ")
session = boto3.Session(profile_name=profile)

ec2 = session.client("ec2")

regions = ec2.describe_regions()

for region in regions['Regions']:
    region_name = region['RegionName']
    if region_name=='ap-northeast-3': #https://github.com/boto/boto3/issues/1943
        continue
    transfer = session.client("transfer", region_name=region_name)
    servers = transfer.list_servers()
    for server in servers['Servers']:
       if server['EndpointType']=='VPC_ENDPOINT':
           print(server['ServerId'], region_name)
```

2. 更新するサーバーのリストができたら、以下のセクションで説明する方法のいずれかを使って、EndpointType を VPC に更新することができます。

を使用したサーバーエンドポイントタイプの更新 AWS Management Console

1. https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開きます。
2. ナビゲーションペインで [Servers] (サーバー) を選択します。
3. エンドポイントタイプを変更したいサーバーのチェックボックスをオンにします。

> **Important**
>
> エンドポイントを変更するには、その前にサーバーを停止する必要があります。

4. [Actions] (アクション) で [Stop] (停止) を選択します。
5. 表示される確認ダイアログボックスで [Stop] (停止) を選択して、サーバーを停止したいことを確認します。

> **Note**
>
> 次のステップに進む前に、サーバーの [Status] (ステータス) が [Offline] (オフライン) に変わるのを待ちますが、これには数分かかる場合があります。ステータスの変更を見る

には、[Servers] (サーバー) ページで [Refresh] (更新) の選択が必要になる場合があります。

6. ステータスが「オフライン」に変わったら、サーバーを選択してサーバーの詳細ページを表示します。
7. 「エンドポイントの詳細」セクションで、「編集」を選択します。
8. [Edit endpoint type] (エンドポイントタイプ) で [VPC hosted] (VPC ホステッド) を選択します。
9. [Save] (保存) を選択します。
10. [Actions] (アクション) で [Start] (開始) を選択してサーバーのステータスが [Online] (オンライン) になるのを待ちますが、これには数分かかる場合があります。

を使用したサーバーエンドポイントタイプの更新 AWS CloudFormation

このセクションでは、 AWS CloudFormation を使用してサーバーの を EndpointTypeに更新する方法について説明しますVPC。を使用してデプロイした Transfer Family サーバーでは、この手順を使用します AWS CloudFormation。この例では、Transfer Family サーバーのデプロイに使用される元の AWS CloudFormation テンプレートを次のように示しています。

```
AWS TemplateFormatVersion: '2010-09-09'
Description: 'Create AWS Transfer Server with VPC_ENDPOINT endpoint type'
Parameters:
  SecurityGroupId:
    Type: AWS::EC2::SecurityGroup::Id
  SubnetIds:
    Type: List<AWS::EC2::Subnet::Id>
  VpcId:
    Type: AWS::EC2::VPC::Id
Resources:
  TransferServer:
    Type: AWS::Transfer::Server
    Properties:
      Domain: S3
      EndpointDetails:
        VpcEndpointId: !Ref VPCEndpoint
      EndpointType: VPC_ENDPOINT
      IdentityProviderType: SERVICE_MANAGED
      Protocols:
        - SFTP
  VPCEndpoint:
    Type: AWS::EC2::VPCEndpoint
```

```
    Properties:
      ServiceName: com.amazonaws.us-east-1.transfer.server
      SecurityGroupIds:
        - !Ref SecurityGroupId
      SubnetIds:
        - !Select [0, !Ref SubnetIds]
        - !Select [1, !Ref SubnetIds]
        - !Select [2, !Ref SubnetIds]
      VpcEndpointType: Interface
      VpcId: !Ref VpcId
```

テンプレートは以下の変更によって更新されます。

- EndpointType は VPC に変更されました。
- AWS::EC2::VPCEndpoint リソースは削除されます。
- SecurityGroupId、SubnetIds、および VpcId を AWS::Transfer::Server リソースの EndpointDetails セクションに移動しました。
- EndpointDetails の VpcEndpointId プロパティは削除されました。

更新されたテンプレートは次のようになります。

```
AWS TemplateFormatVersion: '2010-09-09'
Description: 'Create AWS Transfer Server with VPC endpoint type'
Parameters:
  SecurityGroupId:
    Type: AWS::EC2::SecurityGroup::Id
  SubnetIds:
    Type: List<AWS::EC2::Subnet::Id>
  VpcId:
    Type: AWS::EC2::VPC::Id
Resources:
  TransferServer:
    Type: AWS::Transfer::Server
    Properties:
      Domain: S3
      EndpointDetails:
        SecurityGroupIds:
          - !Ref SecurityGroupId
        SubnetIds:
          - !Select [0, !Ref SubnetIds]
          - !Select [1, !Ref SubnetIds]
```

```
        - !Select [2, !Ref SubnetIds]
      VpcId: !Ref VpcId
    EndpointType: VPC
    IdentityProviderType: SERVICE_MANAGED
    Protocols:
      - SFTP
```

を使用してデプロイされた Transfer Family サーバーのエンドポイントタイプを更新するには AWS CloudFormation

1. 次の手順に従って、更新したいサーバーを停止します。

   a. https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開きます。

   b. ナビゲーションペインで [Servers] (サーバー) を選択します。

   c. エンドポイントタイプを変更したいサーバーのチェックボックスをオンにします。

   > **Important**
   >
   > エンドポイントを変更するには、その前にサーバーを停止する必要があります。

   d. [Actions] (アクション) で [Stop] (停止) を選択します。

   e. 表示される確認ダイアログボックスで [Stop] (停止) を選択して、サーバーを停止したいことを確認します。

   > **Note**
   >
   > 次のステップに進む前に、サーバーの [Status] (ステータス) が [Offline] (オフライン) に変わるのを待ちますが、これには数分かかる場合があります。ステータスの変更を見るには、[Servers] (サーバー) ページで [Refresh] (更新) の選択が必要になる場合があります。

2. CloudFormation スタックを更新する

   a. https://console.aws.amazon.com/cloudformation で AWS CloudFormation コンソールを開きます。

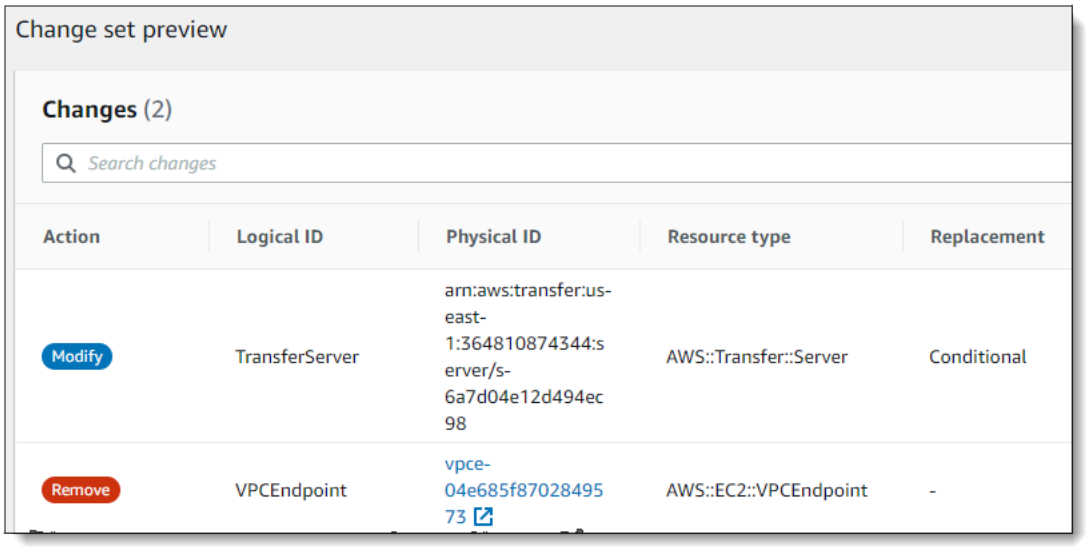
   b. Transfer Family サーバーの作成に使用するスタックを選択します。

   c. [Update] (更新) を選択します。

   d. [Replace current template] (現在のテンプレートを置換) を選択します。

e. 新しいテンプレートをアップロードします。 CloudFormation 変更セットは、テンプレートの変更が実行中のリソースにどのように影響するかを理解してから実装するのに役立ちます。この例では、Transfer サーバーリソースが変更され、vpcendPoint リソースが削除されます。VPC エンドポイントタイプサーバーはユーザーに代わって VPC エンドポイントを作成し、元の `VPCEndpoint` リソースを置き換えます。

新しいテンプレートをアップロードすると、変更セットは以下のようになります。



f. スタックを更新します。

3. スタックの更新が完了したら、Transfer Family 管理コンソール (https://console.aws.amazon.com/transfer/) に移動します。
4. サーバーを再起動します。で更新したサーバーを選択し AWS CloudFormation、アクションメニューから開始を選択します。

API EndpointType を使用したサーバーの更新

describe-server AWS CLI コマンドまたは UpdateServer API コマンドを使用できます。次のサンプルスクリプトは、Transfer Family サーバーを停止し、 を更新し EndpointType、VPC_ENDPOINT を削除して、サーバーを起動します。

```
import boto3
import time

profile = input("Enter the name of the AWS account you'll be working in: ")
region_name = input("Enter the AWS Region you're working in: ")
server_id = input("Enter the AWS Transfer Server Id: ")
```

```
session = boto3.Session(profile_name=profile)

ec2 = session.client("ec2", region_name=region_name)
transfer = session.client("transfer", region_name=region_name)

group_ids=[]

transfer_description = transfer.describe_server(ServerId=server_id)
if transfer_description['Server']['EndpointType']=='VPC_ENDPOINT':
    transfer_vpc_endpoint = transfer_description['Server']['EndpointDetails']
['VpcEndpointId']
    transfer_vpc_endpoint_descriptions =
 ec2.describe_vpc_endpoints(VpcEndpointIds=[transfer_vpc_endpoint])
    for transfer_vpc_endpoint_description in
 transfer_vpc_endpoint_descriptions['VpcEndpoints']:
        subnet_ids=transfer_vpc_endpoint_description['SubnetIds']
        group_id_list=transfer_vpc_endpoint_description['Groups']
        vpc_id=transfer_vpc_endpoint_description['VpcId']
        for group_id in group_id_list:
             group_ids.append(group_id['GroupId'])
    if transfer_description['Server']['State']=='ONLINE':
        transfer_stop = transfer.stop_server(ServerId=server_id)
        print(transfer_stop)
        time.sleep(300) #safe
        transfer_update =
 transfer.update_server(ServerId=server_id,EndpointType='VPC',EndpointDetails={'SecurityGroupId
        print(transfer_update)
        time.sleep(10)
        transfer_start = transfer.start_server(ServerId=server_id)
        print(transfer_start)
        delete_vpc_endpoint =
 ec2.delete_vpc_endpoints(VpcEndpointIds=[transfer_vpc_endpoint])
```

カスタムホスト名の使用

サーバーホスト名は、ユーザーがサーバーに接続するときにクライアントに入力するホスト名です。の使用時に、サーバーホスト名に登録したカスタムドメインを使用できます AWS Transfer Family。たとえば、mysftpserver.mysubdomain.domain.com といったカスタムのホスト名を使用します。

登録したカスタムドメインからトラフィックをサーバーエンドポイントにリダイレクトするには、Amazon Route 53 または任意のドメインネームシステム (DNS) プロバイダーを使用します。Route 53 は AWS Transfer Family がネイティブにサポートする DNS サービスです。

トピック

- Amazon Route 53 を DNS プロバイダーとして使用します。
- 他の DNS プロバイダーを使用する
- コンソール以外で作成したサーバーのカスタムホスト名

コンソールで、次のいずれかのオプションを選択し、カスタムホスト名を設定します。

- Amazon Route 53 DNS alias (Amazon Route 53 DNS エイリアス) – 使用するホスト名が、Route 53 に登録されている場合。その後、ホスト名を入力します。
- Other DNS (その他の DNS) – 使用するホスト名が別の DNS プロバイダーに登録されている場合。その後、ホスト名を入力します。
- None (なし) – サーバーのエンドポイントを使用し、カスタムホスト名は使用しない場合。

このオプションは、新しいサーバーを作成する際または既存のサーバーの設定を編集する際に設定します。新しいサーバーの作成方法の詳細については、「ステップ 2: SFTP 対応サーバーを作成する」を参照してください。既存のサーバー設定の編集方法の詳細については、「サーバーの詳細を編集する」を参照してください。

サーバーのホスト名に独自のドメインを使用する方法と、 が Route 53 AWS Transfer Family を使用する方法の詳細については、以下のセクションを参照してください。

Amazon Route 53 を DNS プロバイダーとして使用します。

サーバーを作成する際には、Amazon Route 53 を DNS プロバイダーとして使用できます。Route 53 でドメインを使用する前に、ドメインを登録します。詳細については、Amazon Route 53 デベロッパーガイドの「ドメイン登録の仕組み」を参照してください。

Route 53 を使用してサーバーに DNS ルーティングを提供する場合、 は入力したカスタムホスト名 AWS Transfer Family を使用してホストゾーンを抽出します。がホストゾーンを AWS Transfer Family 抽出すると、次の 3 つのことが発生する可能性があります。

1. Route 53 を初めて使用し、ホストゾーンがない場合は、 によって新しいホストゾーンとCNAMEレコード AWS Transfer Family が追加されます。この CNAME レコードの値は、サーバーのエンドポイントホスト名です。CNAME は、代替ドメイン名です。
2. Route 53 にホストゾーンがあって CNAME レコードがない場合、 AWS Transfer Family は、ホストゾーンに CNAME レコードを追加します。
3. ホストゾーンに CNAME レコードがあることがサービスによって検出された場合、CNAME レコードがすでに存在していることを示すエラーが表示されます。その場合、CNAME レコードの値をサーバーのホスト名に変更します。

Route 53 のホストゾーンの詳細については、Amazon Route 53 デベロッパーガイドの「ホストゾーンの使用」を参照してください。

他の DNS プロバイダーを使用する

サーバーを作成する際には、Amazon Route 53 以外の DNS プロバイダーも使用できます。代替 DNS プロバイダーを使用する場合は、ドメインからのトラフィックが サーバーエンドポイントに向けられている必要があります。

そのためには、ドメインをサーバーのエンドポイントホスト名に設定します。コンソールには、以下のようなエンドポイントホスト名が表示されます。

serverid.server.transfer.*region*.amazonaws.com

> **Note**
>
> サーバーに VPC エンドポイントがある場合、ホスト名の形式は上記のものとは異なります。VPC エンドポイントを見つけるには、サーバーの詳細ページで VPC を選択し、VPC ダッシュボードで [VPC エンドポイント ID] を選択します。エンドポイントは、リストされている DNS 名のうちの最初の DNS 名です。

コンソール以外で作成したサーバーのカスタムホスト名

AWS Cloud Development Kit (AWS CDK)、 AWS CloudFormation、または CLI を使用してサーバーを作成する場合、そのサーバーにカスタムホスト名を付けるには、タグを追加する必要があります。コンソールを使用してTransfer Family サーバーを作成すると、タグ付けが自動的に完了します。

> **Note**
>
> また、ドメインからサーバーエンドポイントにトラフィックをリダイレクトするための DNS レコードも作成する必要があります。詳細については、「Amazon Route 53 デベロッパーガイド」の「レコードの使用」を参照してください。

カスタム ホスト名には次のキーを使用します。

- `transfer:customHostname` を追加すると、コンソールにカスタム ホスト名が表示されます。
- DNS プロバイダーとして Route 53 を使用している場合は、`transfer:route53HostedZoneId` を追加します。このタグは、カスタムホスト名を Route 53 ホストゾーン ID にリンクします。

カスタム ホスト名を追加するには、次の CLI コマンドを発行します。

```
aws transfer tag-resource --arn arn:aws:transfer:region:AWS #####:server/server-ID --
tags Key=transfer:customHostname,Value="custom-host-name"
```

例:

```
aws transfer tag-resource --arn arn:aws:transfer:us-east-1:111122223333:server/
s-1234567890abcdef0 --tags Key=transfer:customHostname,Value="abc.example.com"
```

Route 53 を使用している場合は、次のコマンドを実行して、カスタム ホスト名を Route 53 ホストゾーン ID にリンクします。

```
aws transfer tag-resource --arn server-ARN:server/server-ID --tags
 Key=transfer:route53HostedZoneId,Value=HOSTED-ZONE-ID
```

例:

```
aws transfer tag-resource --arn arn:aws:transfer:us-east-1:111122223333:server/
s-1234567890abcdef0 --tags Key=transfer:route53HostedZoneId,Value=ABCDE1111222233334444
```

前のコマンドのサンプル値を仮定して、次のコマンドを実行してタグを表示します。

```
aws transfer list-tags-for-resource --arn arn:aws:transfer:us-
east-1:111122223333:server/s-1234567890abcdef0
```

```
"Tags": [
   {
      "Key": "transfer:route53HostedZoneId",
      "Value": "/hostedzone/ABCDE1111222233334444"
   },
   {
      "Key": "transfer:customHostname",
      "Value": "abc.example.com"
   }
 ]
```

> **Note**
>
> パブリックホストゾーンとその ID を Amazon Route 53 で利用できます。
> にサインイン AWS Management Console し、https://console.aws.amazon.com/route53/ で Route 53 コンソールを開きます。

クライアントを使用してサーバーエンドポイント経由でファイルを転送する

クライアントで転送オペレーションを指定して、 AWS Transfer Family サービス経由でファイルを転送します。 は次のクライアント AWS Transfer Family をサポートします。

- SFTP プロトコルのバージョン 3 をサポートしています。
- OpenSSH (macOS および Linux)

> **Note**
>
> このクライアントは、Secure Shell (SSH) File Transfer Protocol (SFTP) が有効なサーバーとの組み合わせでのみ機能します。

- WinSCP (Microsoft Windows のみ)
- Cyberduck (Windows、macOS、および Linux)
- FileZilla (Windows、macOS、Linux)

どのクライアントにも以下の制限が適用されます。

- 1 接続あたりの、同時、多重化された SFTP セッションの最大数は 10 です。
- SFTP/FTP/FTPS 接続には 2 つのタイムアウト値があります。アイドル接続の場合、タイムアウト値は 1800 秒 (30 分) です。期間が経過した後にアクティビティがない場合、クライアントは切断される可能性があります。また、クライアントが完全に応答しない場合、300 秒 (5 分) のタイムアウトが発生します。
- Amazon S3 と Amazon EFS (NFSv4 プロトコルのため) では、ファイル名は UTF-8 エンコーディングである必要があります。異なるエンコーディングを使用すると、予期しない結果になることがあります。Amazon S3 については、「オブジェクトキーの命名ガイドライン」を参照してください。
- File Transfer Protocol over SSL (FTPS) については、明示モードのみがサポートされます。暗黙モードはサポートされません。
- FTP および FTPS では、パッシブモードのみがサポートされます。
- FTP および FTPS では、STREAM モードのみがサポートされます。
- FTP および FTPS では、イメージ/バイナリモードのみがサポートされます。
- FTP および FTPS の場合、データ接続の TLS - PROT C (保護されていない) TLS がデフォルトですが、 AWS Transfer Family FTPS プロトコルでは PROT C はサポートされていません。つまり、FTPS の場合は、PROT P を発行してデータ操作を受け付ける必要があります。
- サーバーのストレージに Amazon S3 を使用していて、1 回の転送に複数の接続を使用するオプションがクライアントに含まれている場合は、必ずそのオプションを無効にしてください。そうしないと、大容量ファイルのアップロードが失敗して予測外の状況になる可能性があります。Amazon EFS をストレージバックエンドとして使用している場合、EFS は 1 回の転送で複数の接続をサポート「する」ことに注意してください。

以下に示すのは、FTP および FTPS で使用できるコマンドの一覧です。

使用できるコマンド					
ABOR	FEAT	MLST	PASS	RETR	STOR
AUTH	LANG	MKD	PASV	RMD	STOU
CDUP	LIST	MODE	PBSZ	RNFR	STRU
CWD	MDTM	NLST	PROT	RNTO	SYST
DELE	MFMT	NOOP	PWD	SIZE	TYPE

使用できるコマンド					
EPSV	MLSD	OPTS	QUIT	STAT	USER

> **Note**
>
> APPE はサポートされません。

SFTP については、Amazon Elastic File System (Amazon EFS) を使用しているサーバーで論理ホームディレクトリを使用するユーザーの場合、現時点で以下のオペレーションはサポートされていません。

サポートされない SFTP コマンド			
SSH_FXP_READLINK	SSH_FXP_SYMLINK	リクエストされたファイルがシンボリックリンクの場合、SSH_FXP_STAT	リクエストされたパスにシンボリックリンクコンポーネントが含まれている場合、SSH_FXP_REALPATH

パブリックキーとプライベートキーのペアを生成する

ファイルを転送する前に、パブリックキーとプライベートキーのキーペアを用意する必要があります。以前にキーペアを生成していない場合、「サービス管理ユーザーの SSH キーの生成」を参照してください。

トピック

- 使用可能な SFTP/FTPS/FTP コマンド
- Amazon VPC エンドポイントを検索する
- setstatエラーを回避する
- OpenSSH を使用する
- WinSCP を使用する
- Cyberduck を使用する

- を使用する FileZilla
- Perl クライアントを使用する
- アップロード後の処理

使用可能な SFTP/FTPS/FTP コマンド

次の表に、SFTP AWS Transfer Family、FTPS、および FTP プロトコルで使用可能な コマンドを示します。

> **Note**
>
> この表には、バケットとオブジェクトのみをサポートする Amazon S3 の「ファイル」と「ディレクトリ」が記載されています。階層はありません。ただし、オブジェクトキー名にプレフィックスを使って階層を示したり、フォルダのようにデータを整理したりすることはできます。この動作については、「Amazon Simple Storage Service ユーザーガイド」の「オブジェクトメタデータの処理」で説明されています。

SFTP/FTPS/FTP コマンド

Command	Amazon S3	Amazon EFS
`cd`	サポート	サポート
`chgrp`	サポートされていません	対応 (`root` または `owner` のみ)
`chmod`	サポートされていません	サポート (`root`のみ)
`chmtime`	サポート外	サポート
`chown`	サポートされていません	サポート (`root`のみ)
`get`	サポート	サポート (シンボリックリンクの解決を含む)
`ln -s`	サポート外	サポート対象
`ls/dir`	サポート対象	サポート対象

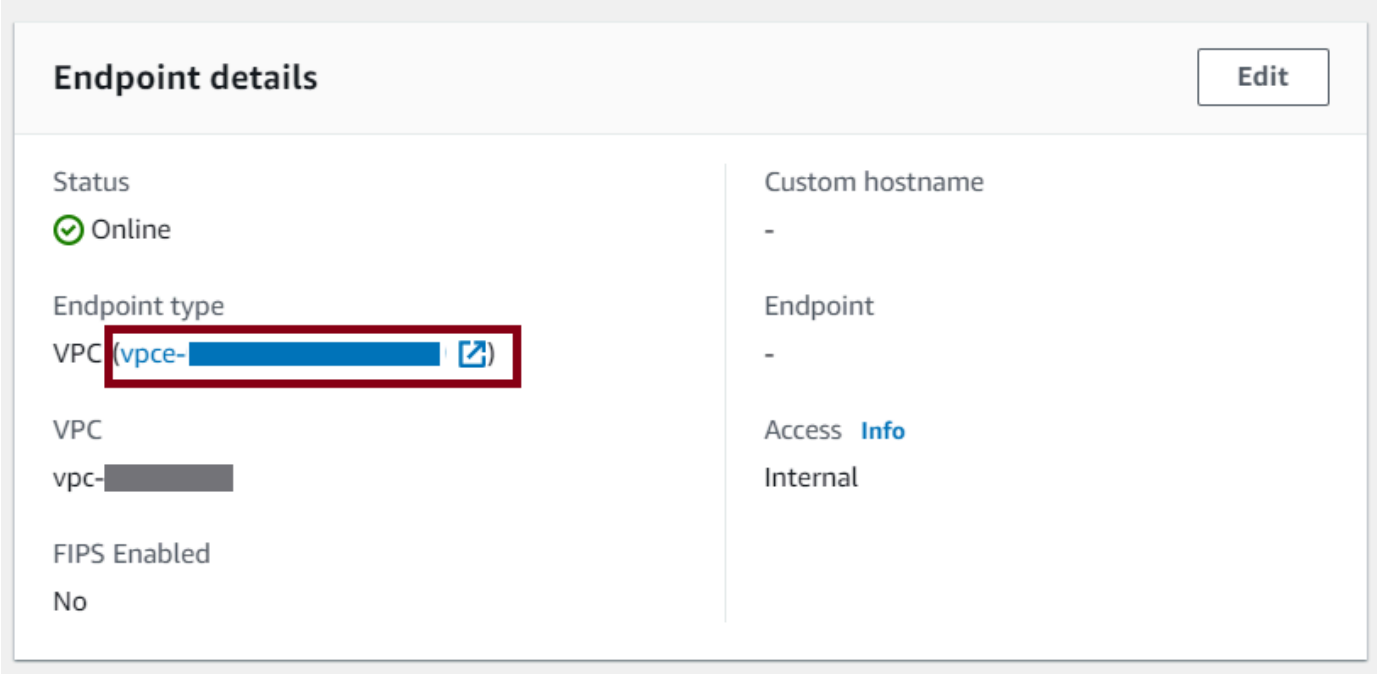
Command	Amazon S3	Amazon EFS
`mkdir`	サポート対象	サポート対象
`put`	サポート対象	サポート対象
`pwd`	サポート対象	サポート
`rename`	ファイルでのみサポート	サポート **Note** 既存のファイルやディレクトリを上書きするような名前変更はサポートされていません。
`rm`	サポート	サポート
`rmdir`	サポート (空のディレクトリのみ)	サポート
`version`	サポート対象	サポート

Amazon VPC エンドポイントを検索する

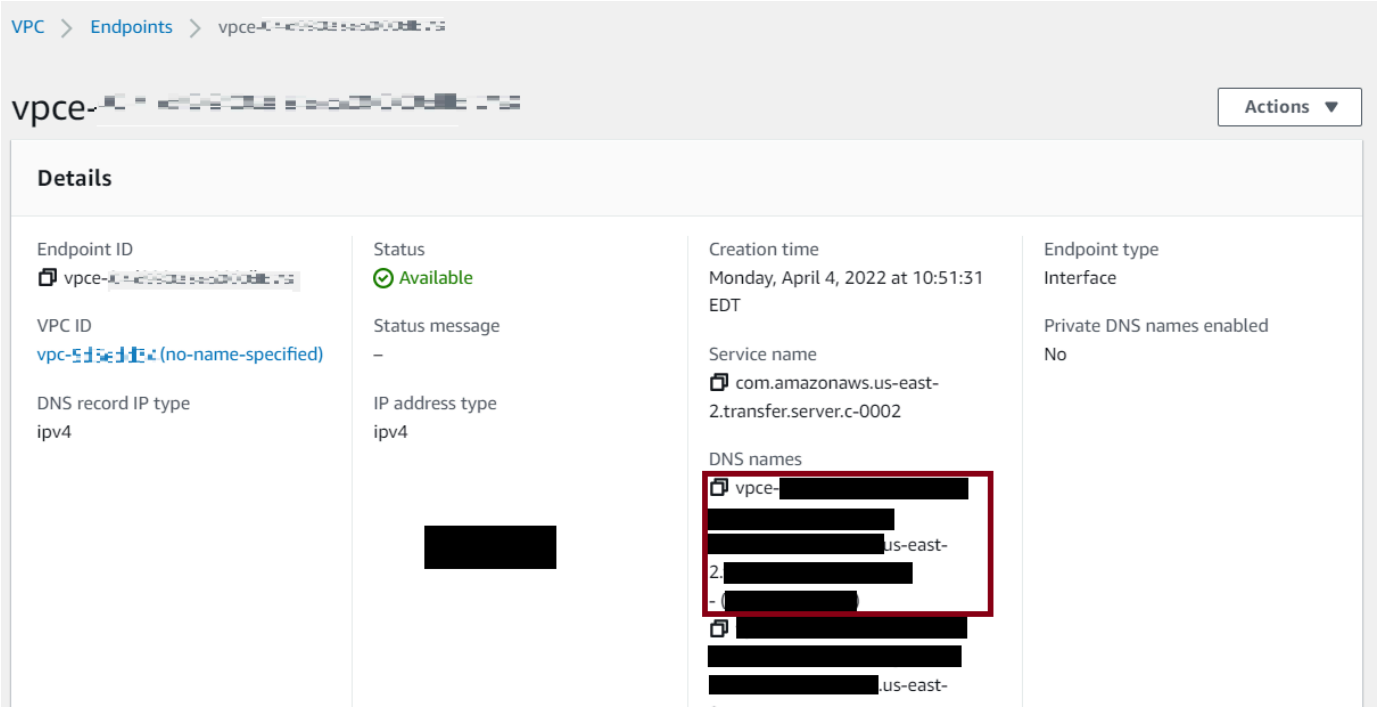
Transfer Familyサーバーのエンドポイントタイプが VPC の場合、ファイルの転送に使用するエンドポイントを特定するのは簡単ではありません。この場合、次の手順を使用して、Amazon VPC エンドポイントを検索します。

Amazon VPC エンドポイントを検索する

1. サーバーの詳細ページに移動します。
2. 「エンドポイントの詳細」ペインで、「VPC」を選択します。



3. Amazon VPC ダッシュボードで「VPC エンドポイント ID」を選択します。
4. 「DNS 名」のリストでは、サーバーエンドポイントが最初に表示されます。



setstatエラーを回避する

SFTP ファイル転送クライアントの中には、ファイルをアップロードする 際に SETSTAT などのコマンドを使用して、タイムスタンプやパーミッションな ど、リモートファイルの属性を変更しようとするものがあります。ただし、これらのコマンドは Amazon S3 などのオブジェクトストレージシステムと互換性がありません。この非互換性が原因で、これらのクライアントからファイルをアップロードする際に、ファイルが正常にアップロードされた場合でもエラーが発生する可能性があります。

- `CreateServer` または `UpdateServer` API を呼び出すときに、`ProtocolDetails` オプション `SetStatOption` を使用して、クライアントが S3 バケットにアップロードするファイルに SETSTAT を使用しようとしたときに生成されるエラーを無視します。
- Transfer Family サーバーが SETSTAT コマンドを無視するように値を `ENABLE_NO_OP` に設定し、SFTP クライアントに変更を加えることなくファイルをアップロードできます。
- `SetStatOption` `ENABLE_NO_OP` 設定ではエラーは無視されますが、 CloudWatch Logs にログエントリが生成されるため、クライアントが SETSTAT 呼び出しをいつ実行しているかを判断できます。

このオプションの API の詳細については、「」を参照してくださいProtocolDetails。

OpenSSH を使用する

次の手順に従って、OpenSSH を使用してコマンドラインからファイルを転送します。

> **Note**
>
> このクライアントは SFTP 対応サーバーでのみ動作します。

OpenSSH コマンドラインユーティリティ AWS Transfer Family を使用してファイルを 経由で転送するには

1. Linux、macOS、Windows では、コマンドターミナルを開きます。
2. プロンプトで、次のコマンドを入力します。

```
sftp -i transfer-key sftp_user@service_endpoint
```

前述のコマンドでは、*sftp_user* がユーザー名、*transfer-key* が SSH プライベートキーです。ここで、 *service_endpoint*は、選択したサーバーの AWS Transfer Family コンソールに表示されるサーバーのエンドポイントです。

> **Note**
>
> このコマンドは、デフォルトの ssh_config ファイルにある設定を使用します。このファイルを以前に編集していない限り、SFTP はポート 22 を使用します。次のようにコマンドに**-P**フラグを追加することで、別のポート (2222 など) を指定できます。
>
> ```
> sftp -P 2222 -i transfer-key sftp_user@service_endpoint
> ```
>
> または、常にポート 2222 またはポート 22000 を使用する場合は、 ssh_config ファイル内のデフォルトポートを更新できます。

sftp プロンプトが表示されます。

3. (オプション) ユーザーのホームディレクトリを表示するには、sftp プロンプトで次のコマンドを入力します。

   pwd

4. ご使用のファイルシステムから Transfer Family サーバーにファイルをアップロードするには、put コマンドを使用します。たとえば、hello.txt をアップロードするには (ファイルがファイルシステム上のカレントディレクトリにあると仮定して)、sftp プロンプトで次のコマンドを実行します。

   ```
   put hello.txt
   ```

   ファイル転送が進行中か完了したことを示す次のようなメッセージが表示されます。

   ```
   Uploading hello.txt to /my-bucket/home/sftp_user/hello.txt

   hello.txt 100% 127 0.1KB/s 00:00
   ```

> **Note**
>
> サーバーの作成後、お使いの環境の DNS サービスによってサーバーエンドポイントのホスト名が解決可能になるまでに数分かかることがあります。

WinSCP を使用する

次の手順に従って、WinSCP を使用してコマンドラインからファイルを転送します。

> **Note**
>
> WinSCP 5.19 を使用している場合は、 AWS 認証情報を使用して Amazon S3 に直接接続し、ファイルをアップロード/ダウンロードできます。詳細については、「Amazon S3 サービスに接続する」を参照してください。

WinSCP AWS Transfer Family を使用して 経由でファイルを転送するには

1. WinSCP クライアントを開きます。
2. [Login] (ログイン) ダイアログボックスで [File protocol] (ファイルプロトコル) として SFTP または FTP を選択します。

   [FTP] を選択した場合、[Encryption] (暗号化) で次のいずれかを選択します。

   - FTP の場合、暗号化なし
   - FTPS の場合、TLS/SSL 明示的な暗号化
3. [Host name] (ホスト名) にサーバーエンドポイントを入力します。サーバーエンドポイントは、[Server details] (サーバーの詳細) ページにあります 。詳細については、「SFTP、FTPS、FTP サーバーの詳細を表示する」を参照してください。

   > **Note**
   >
   > サーバーが VPC エンドポイントを使用している場合は、Amazon VPC エンドポイントを検索するを参照してください。
4. [Port number] (ポート番号) に次の値を入力します。

   - SFTP の場合、**22**

- FTP/FTPS の場合、**21**

5. 「ユーザー名」には、特定の ID プロバイダー用に作成したユーザーの名前を入力します。

> **Note**
>
> ユーザー名は、ID プロバイダー用に作成または設定したユーザーの 1 人である必要があります。 は、次の ID プロバイダー AWS Transfer Family を提供します。
>
> - サービスマネージドユーザーの使用
> - AWS Directory Service ID プロバイダーの使用
> - カスタム ID プロバイダーの使用

6. [Advanced] (アドバンスト) を選択して、[Advanced Site Settings] (サイトのアドバンスト設定) ダイアログボックスを開きます。[SSH] セクションで [Authentication] (認証) を選択します。
7. [Private key file] (プライベートキーファイル) でファイルシステムから SSH プライベートキーを参照して選択します。

> **Note**
>
> WinSCP から SSH プライベートキーを PPK 形式に変換するオプションが表示されたら [OK] を選択します。

8. [OK] を選択して [Login] (ログイン) ダイアログボックスに戻り、[Save] (保存) を選択します。
9. [Save session as site] (セッションをサイトとして保存) ダイアログボックスで [OK] を選択して接続セットアップを完了します。
10. [Login] (ログイン) ダイアログボックスで [Tools] (ツール) を選択してから[Preferences] (設定) を選択します。
11. [Preferences] (設定) ダイアログボックスの [Transfer] (転送) で [Endurance] (耐久性) を選択します。

    [Enable transfer resume/transfer to temporary filename for] オプションについて [Disable] (無効にする) を選択します。

> **Note**
>
> このオプションを有効にしたままにすると、アップロードコストが増加し、アップロードのパフォーマンスが大幅に低下します。大容量ファイルのアップロードが失敗する可能性もあります。

12. [Transfer] (転送) で [Background] (バックグラウンド) を選択し、[Use multiple connections for single transfer] (単一の転送に複数の接続を使用する) チェックボックスをオフにします。

> **Note**
>
> このオプションを選択したままにすると、大容量ファイルのアップロードが失敗して予測外の状況になる可能性があります。たとえば、孤立マルチパートアップロードが生成されて Amazon S3 の料金が発生する可能性があります。サイレントデータ破損が発生することもあります。

13. ファイル転送を実行します。

    drag-and-drop メソッドを使用して、ターゲットウィンドウとソースウィンドウ間でファイルをコピーできます。ツールバーアイコンを使用して、WinSCP にあるファイルのプロパティをアップロード/ダウンロード、削除、編集、または変更できます。

> **Note**
>
> Amazon EFS をストレージに使用している場合、この注意事項は適用されません。
> タイムスタンプを含むリモートファイルの属性を変更しようとするコマンドは Amazon S3 などのオブジェクトストレージシステムと互換性がありません。そのため、Amazon S3 をストレージとして使用している場合は、ファイル転送を実行する前に、WinSCP タイムスタンプ設定を必ず無効にしてください (または、`setstat`エラーを回避するでで説明したように`SetStatOption`を使う)。そのためには、WinSCP Transfer settings (WinSCP 転送設定) ダイアログボックスで [Set permissions] (パーミッションの設定) オプションと [Preserve timestamp] (タイムスタンプを保持) オプションを無効にします。

Cyberduck を使用する

次の手順に従って、Cyberduck を使用してコマンドラインからファイルを転送します。

Cyberduck AWS Transfer Family を使用してファイルを 経由で転送するには

1. Cyberduck クライアントを開きます。
2. [Open connection] (接続を開く) を選択します。
3. [Open Connection] (接続を開く) ダイアログボックスでプロトコルとして SFTP (SSH File Transfer Protocol)、FTP-SSL (Explicit AUTH TLS)、または FTP (File Transfer Protocol) を選択します。
4. [Servers] (サーバー) にサーバーのエンドポイントを入力します。サーバーエンドポイントは、[Server details] (サーバーの詳細) ページにあります 。詳細については、「SFTP、FTPS、FTP サーバーの詳細を表示する」を参照してください。

   > **Note**
   >
   > サーバーが VPC エンドポイントを使用している場合は、Amazon VPC エンドポイントを検索するを参照してください。

5. [Port number] (ポート番号) に次の値を入力します。

   - SFTP の場合、**22**
   - FTP/FTPS の場合、**21**

6. [Username] (ユーザー名) に サーバーエンドポイントのユーザーの管理 で作成したユーザーの名前を入力します。
7. SFTP を選択した場合、[SSH Private Key] (SSH プライベートキー) で SSH プライベートキー] を選択するか入力します。
8. [Connect] (接続) を選択します。
9. ファイル転送を実行します。

   ファイルの場所に応じて、次のいずれかの操作をします。

   - ローカルディレクトリ (転送元) で転送したいファイルを選択して Amazon S3 ディレクトリ (転送先) にドラッグアンドドロップします。
   - Amazon S3 ディレクトリ (転送元) で転送したいファイルを選択してローカルディレクトリ (転送先) にドラッグアンドドロップします。

を使用する FileZilla

を使用してファイルを転送するには、以下の手順に従います FileZilla。

ファイル転送 FileZilla をセットアップするには

1. FileZilla クライアントを開きます。
2. [File] (ファイル) を選択してから [Site Manager] を選択します。
3. [Site Manager] ダイアログボックスで [New site] (新しいサイト) を選択します。
4. [General] (全般) タブの [Protocol] (プロトコル) で SFTP または FTP のプロトコルを選択します。

   [FTP] を選択した場合、[Encryption] (暗号化) で次のいずれかを選択します。

   - 平文 FTP のみを使用する (安全でない) — FTP の場合
   - TLS 経由の明示的な FTP を (可能であれば) 使用する — FTPS の場合
5. [Host name] (ホスト名) には、使用するプロトコルを入力し、その後にサーバーエンドポイントを続けます。サーバーエンドポイントは、[Server details] (サーバーの詳細) ページにあります 。詳細については、「SFTP、FTPS、FTP サーバーの詳細を表示する」を参照してください。

   > **Note**
   >
   > サーバーが VPC エンドポイントを使用している場合は、Amazon VPC エンドポイントを検索するを参照してください。

   - SFTP を使用する場合、`sftp://`*`hostname`* を入力します。
   - FTPS を使用する場合、`ftps://`*`hostname`* を入力します。

   `hostname` を実際のサーバーエンドポイントに確実に置き換えてください。
6. [Port number] (ポート番号) に次の値を入力します。

   - SFTP の場合、**22**
   - FTP/FTPS の場合、**21**

7. [SFTP] を選択した場合、[Logon type] (ログオンの種類) で [Key file] (キーファイル) を選択します。

   [Key file] (キーファイル) で SSH プライベートキーを選択するか入力します。

8. [Username] (ユーザー名) に、サーバーエンドポイントのユーザーの管理 で作成したユーザーの名前を入力します。
9. [Connect] (接続) を選択します。
10. ファイル転送を実行します。

> **Note**
>
> 進行中のファイル転送を中断すると、Amazon S3 バケットに部分的なオブジェクトが書き込まれる AWS Transfer Family 可能性があります。アップロードを中断する場合、続行する前に Amazon S3 バケットのファイルサイズがソースオブジェクトのファイルサイズと一致することを確認してください。

Perl クライアントを使用する

perl クライアントを使用する場合は NET::SFTP::Foreign、 queue_sizeを に設定する必要があります1。例:

```
my $sftp = Net::SFTP::Foreign->new('user@s-12345.server.transfer.us-east-2.amazonaws.com', queue_size => 1);
```

> **Note**
>
> この回避策は、1.92.02 以前の Net::SFTP::Foreign のリビジョンに必要です。

アップロード後の処理

Amazon S3 オブジェクトのメタデータやイベント通知など、アップロード後の処理情報を表示できます。

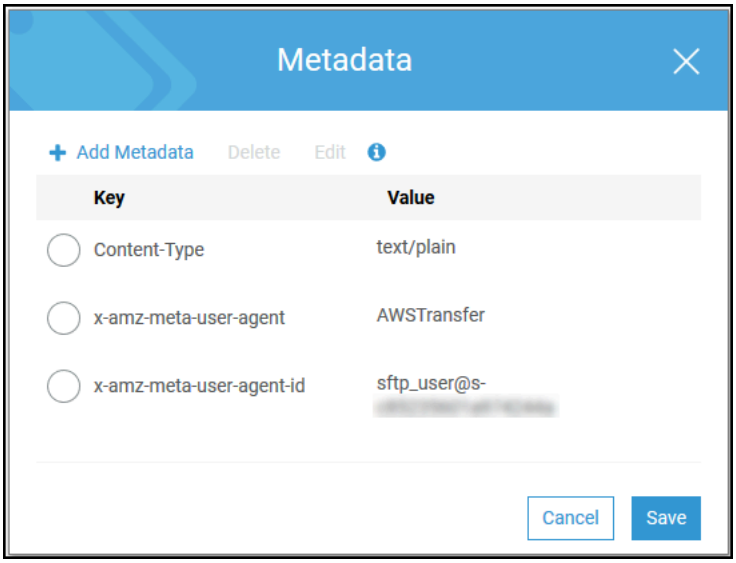
トピック

- Amazon S3 オブジェクトメタデータ

- Amazon S3 イベント通知

Amazon S3 オブジェクトメタデータ

オブジェクトのメタデータの一部として、値が AWSTransfer である x-amz-meta-user-agent と値が username@server-id である x-amz-meta-user-agent-id というキーが見えます。username はファイルをアップロードしたTransfer Family ユーザー、server-id はアップロードに使用したサーバーです。この情報は、Lambda 関数内の S3 オブジェクトに対する HeadObjectオペレーションを使用してアクセスできます。



Amazon S3 イベント通知

Transfer Family を使用してオブジェクトを S3 バケットにアップロードすると、S3 イベント通知構造で RoleSessionName が [Requester] (リクエスタ) フィールドに [AWS:Role Unique Identifier]/username.sessionid@server-id として含まれます。たとえば、次に示すのは S3 バケットにコピーされたファイルの S3 アクセスログから得たサンプル [Requester] (リクエスタ) フィールドのコンテンツです。

```
arn:aws:sts::AWS-Account-ID:assumed-role/IamRoleName/
username.sessionid@server-id
```

上記の [Requester] (リクエスタ) フィールドには、IamRoleName と呼ばれる IAM ロールが表示されます。S3 イベント通知を設定する方法の詳細については、Amazon Simple Storage Service デベロッパーガイドの「Amazon S3 イベント通知の設定」を参照してください。 AWS Identity and Access Management (IAM) ロールの一意の識別子の詳細については、AWS Identity and Access Management 「 ユーザーガイド」の「一意の識別子」を参照してください。

サーバーエンドポイントのユーザーの管理

以下のセクションでは、AWS Transfer Family、AWS Directory Service for Microsoft Active Directory、またはカスタム ID プロバイダーを使用してユーザーを追加する方法について説明します。

サービス管理型 ID タイプを使用する場合は、ファイル転送プロトコル対応のサーバーにユーザーを追加します。その場合、各ユーザー名はサーバーで一意である必要があります。

各ユーザーのプロパティの一部として、そのユーザーの Secure Shell (SSH) パブリックキーも保存します。この手順で使用する、キーベースの認証では、そうする必要があります。プライベートキーは、ユーザーのコンピュータにローカルに保存されます。ユーザーがクライアントを使用してサーバーに認証要求を送信すると、サーバーはまず、ユーザーが関連付けられた SSH 秘密キーにアクセスできることを確認します。その後、サーバーはユーザーを正常に認証します。

さらに、ユーザーのホームディレクトリ、またはランディングディレクトリを指定して、ユーザーに AWS Identity and Access Management (IAM) ロールを割り当てます。必要に応じて、セッションポリシーを提供してユーザーアクセスを Amazon S3 バケットのホームディレクトリのみに限定できます。

> **Important**
>
> AWS Transfer Family は、1 文字または 2 文字の長さのユーザー名を SFTP サーバーへの認証からブロックします。さらに、`root` ユーザー名もブロックされます。
> これは、パスワードスキャナによる悪意のあるログイン試行の大量発生が原因です。

Amazon EFS と Amazon S3 の比較

各ストレージオプションの特性:

- アクセスを制限するために、Amazon S3 はセッションポリシーをサポートし、Amazon EFS は POSIX ユーザー、グループ、およびセカンダリグループ ID をサポートします。
- いずれもパブリックキー/プライベートキーをサポートする
- いずれもホームディレクトリをサポートする
- いずれも論理ディレクトリをサポートする

> **Note**
>
> Amazon S3 では、論理ディレクトリのサポートの大部分は API/CLI 経由です。コンソールの [Restricted] (制限) チェックボックスを使用すると、ユーザーをホームディレクトリにロックダウンできますが、仮想ディレクトリ構造は指定できません。

論理ディレクトリ

ユーザーに対して論理ディレクトリ値を指定する場合、使用するパラメータはユーザーのタイプに依存します。

- サービス管理型ユーザーの場合、論理ディレクトリの値を `HomeDirectoryMappings` に指定してください。
- カスタム ID プロバイダーユーザーの場合は、 で論理ディレクトリ値を指定しますHomeDirectoryDetails。

トピック

- サービスマネージドユーザーの使用
- AWS Directory Service ID プロバイダーの使用
- カスタム ID プロバイダーの使用

サービスマネージドユーザーの使用

サーバーのドメイン設定に応じて、Amazon S3 または Amazon EFS いずれかのサービスマネージドユーザーをサーバーに追加できます。詳細については、「SFTP、FTPS、または FTP サーバーエンドポイントの設定」を参照してください。

サービスマネージドユーザーをプログラムで追加するには、 CreateUser API の例を参照してください。

> **Note**
>
> サービスマネージドユーザーの場合、論理ディレクトリエントリの数は 2,000 に制限されています。論理ディレクトリの使用については、「」を参照してください論理ディレクトリを使用して Transfer Family ディレクトリ構造を簡素化する。

トピック

- Amazon S3 サービスマネージドユーザーの追加
- Amazon EFS サービスマネージドユーザーの追加
- サービスマネージドユーザーの管理

Amazon S3 サービスマネージドユーザーの追加

> **Note**
>
> クロスアカウントの Amazon S3 バケットを設定する場合は、Knowledge Center の記事「AWS 別のアカウントにある Amazon Simple Storage Service AWS Transfer Family バケットを使用するようにサーバーを設定するにはどうすればよいですか?」で説明されている手順に従ってください。

Amazon S3 サービスマネージドユーザーをサーバーに追加するには

1. https://console.aws.amazon.com/transfer で AWS Transfer Family コンソールを開いてからナビゲーションペインで [Servers] (サーバー) を選択します。
2. [Servers] (サーバー) ページで、ユーザーの追加先になるサーバーのチェックボックスをオンにします。
3. [Add user] (ユーザーを追加) を選択します。
4. [ユーザー設定] セクションの[ユーザー名] にユーザー名を入力します。このユーザー名は最低 3 文字、最高 100 文字である必要があります。ユーザー名に使用できる文字は、a-z、A-Z、0-9、アンダースコア「_」、ハイフン「-」、ピリオド「.」、およびアットマーク「@」です。ユーザー名をハイフン「-」、ピリオド「.」、アットマーク「@」で始めることはできません。
5. [Access] (アクセス) で、以前に作成した Amazon S3 バケットへのアクセスを提供する IAM ロールを選択します。

   この IAM ロールは、「IAM ポリシーとロールを作成する」の手順に従って作成されました。その IAM ロールには、Amazon S3 バケットへのアクセスを提供する IAM ポリシーが含まれます。別の IAM ポリシーで定義された AWS Transfer Family サービスとの信頼関係も含まれます。ユーザーに対してきめ細かいアクセス制御が必要な場合は、「Amazon S3 によるデータ アクセス制御の強化」 AWS Transfer Family に関するブログ投稿を参照してください。
6. （オプション）「Policy」では、以下のいずれかを選択する：

- None (なし)
- Existing policy (既存のポリシー)
- 「IAM からポリシーを選択する」: 既存のセッションポリシーを選択できます。[View] (表示) を選択すると、ポリシーの詳細を含む JSON オブジェクトが表示されます。
- [ホームフォルダーに基づいてポリシーを自動生成]:セッションポリシーを生成します。[View] (表示) を選択すると、ポリシーの詳細を含む JSON オブジェクトが表示されます。

> **Note**
>
> [ホームフォルダーに基づいてポリシーを自動生成] を選択した場合は、[このユーザーに対して制限付き] を選択しないでください。

セッションポリシーの詳細については、「IAM ポリシーとロールを作成する」を参照してください。セッションポリシーの作成方法の詳細については、「Amazon S3 バケットのセッションポリシーの作成」を参照してください 。

7. [Home Directory] (ホームディレクトリ) で AWS Transfer Family を使用して転送するデータを保存する Amazon S3 バケットを選択します。ユーザーが SFTP クライアントを使用してログインしたときにアクセスする home ディレクトリのパスを入力します。

   このパラメータを空白のままにした場合、Amazon S3 バケットの `root` ディレクトリが使用されます。この場合、IAM ロールが `root` ディレクトリへのアクセスを提供することを確認します。

   > **Note**
   >
   > セッションポリシーを効果的に使用できるように、ユーザーのユーザー名が含まれるディレクトリパスを選択することをお勧めします。セッションポリシーは、Amazon S3 バケットでのユーザーアクセスを home ディレクトリに制限します。

8. (オプション) [Restricted] (制限) チェックボックスをオンにすると、ユーザーはそのフォルダの外部にあるものにアクセスしたり、Amazon S3 バケット名やフォルダ名を表示したりできなくなります。

> **Note**
>
> ユーザーにホームディレクトリを割り当て、そのホームディレクトリへのアクセスを制限すれば、指定されたフォルダへのアクセスをロックダウンするのに十分なはずです。さらにコントロールの適用が必要な場合、セッションポリシーを使用します。
> このユーザーに対して[制限付き] を選択した場合、ホーム フォルダーは制限付きユーザーに対して定義された値ではないため、[ホーム フォルダーに基づいてポリシーを自動生成] を選択できません。

9. [SSH public key] (SSH 公開キー) に SSH キーペアの SSH 公開キー部分を入力します。

   新しいユーザーを追加する前に、サービスによってキーが検証されます。

   > **Note**
   >
   > SSH キーペアを生成する手順については、「サービス管理ユーザーの SSH キーの生成」を参照してください。

10. (オプション) [Key] (キー) と [Value] (値) にキーバリューペアとして 1 つ以上のタグを入力して [Add tag] (タグの追加) を選択します。
11. [Add] (追加) を選択して、選択したサーバーに新しいユーザーを追加します。

    [Server details] (サーバーの詳細) ページの [Users] (ユーザー) セクションに新しいユーザーが表示されます。

次のステップ— 次のステップについては、「クライアントを使用してサーバーエンドポイント経由でファイルを転送する」に進みます。

Amazon EFS サービスマネージドユーザーの追加

Amazon EFS は、Portable Operating System Interface (POSIX) ファイルアクセス許可モデルを使用して、ファイルの所有権を表します。

- Amazon EFS ファイルの所有権の詳細については、「Amazon EFS ファイルの所有権」を参照してください。
- EFSユーザー向けのディレクトリの設定に関する詳細は、 を参照してくださいTransfer Family用にAmazon EFSユーザーを設定します。。

Amazon EFS サービスマネージドユーザーをサーバーに追加するには

1. https://console.aws.amazon.com/transfer で AWS Transfer Family コンソールを開いてからナビゲーションペインで [Servers] (サーバー) を選択します。
2. [Servers] (サーバー) ページで、ユーザーの追加先になる Amazon EFS サーバーのチェックボックスをオンにします。
3. [Add user] (ユーザーの追加) を選択して [Add user] (ユーザーの追加) ページを表示します。
4. [User configuration] (ユーザー設定) セクションで、次のように設定します。

   a. [ユーザー名] は、最小 3 文字、最大 100 文字にする必要があります。ユーザー名に使用できる文字は、a-z、A-Z、0-9、アンダースコア「_」、ハイフン「-」、ピリオド「.」、およびアットマーク「@」です。ユーザー名をハイフン「-」、ピリオド「.」、アットマーク「@」で始めることはできません。

   b. ユーザー ID およびグループ ID については、以下の点に注意してください。

      - 最初に作成するユーザーには、グループ ID とユーザー ID の両方に **0** の値を入力することをお勧めします。これにより、Amazon EFS のユーザー管理者権限が付与されます。
      - 追加のユーザーについて、ユーザーの POSIX ユーザー ID とグループ ID を入力します。これらの ID は、ユーザーが実行するすべての Amazon Elastic File System オペレーションに使用されます。
      - [User ID] (ユーザー ID) と [Group ID] (グループ ID) には、先頭にゼロを使用しないでください。たとえば、**12345** は許容されますが **012345** は許容されません。

   c. (オプション) [Secondary Group IDs] (セカンダリグループ ID) には、ユーザーごとに 1 つ以上の追加の POSIX グループ ID をカンマで区切って入力します。

   d. [Access] (アクセス) で、以下のような IAM ロールを選択します。

      - アクセスしたい Amazon EFS リソース (ファイルシステム) にのみアクセス権を付与します。
      - ユーザーが実行できるファイルシステム操作と実行できないファイルシステムシステムオペレーションを定義します。

      Amazon EFS ファイルシステムの選択にあたっては、マウントアクセスと読み書きアクセス許可を持つ IAM ロールを使用することをお勧めします。たとえば、次の 2 つの AWS マネージドポリシーの組み合わせは、かなり寛容でありながら、ユーザーに必要なアクセス許可を付与します。

- AmazonElasticFileSystemClientFullAccess
- AWSTransferConsoleFullAccess

詳細は、ブログ記事「Amazon Elastic File Systemの AWS Transfer Family 対応」を参照してください。

e. [Home directory] (ホームディレクトリ) について、次のように操作します。

- AWS Transfer Family で転送するデータの保存に使用したい Amazon EFS ファイルシステムを選択します。
- ホームディレクトリを制限するかどうかを決めます。ホームディレクトリを [Restricted] (制限) に設定すると次の効果があります。
  - Amazon EFS ユーザーは、そのフォルダ外のファイルやディレクトリにアクセスできません。
  - Amazon EFS ユーザーは Amazon EFS ファイルシステム名 (fs-xxxxxxx) を表示できません。

> **Note**
>
> [Restricted] (制限) オプションを使用すると、Amazon EFS ユーザーのシンボリックリンクは解決されません。

- (オプション) ユーザーがクライアントとしてログインしたときのホームディレクトリのパスを入力します。

  ホームディレクトリを指定しない場合、Amazon EFS ファイルシステムのルートディレクトリが使用されます。この場合、IAM ロールがルートディレクトリへのアクセスを提供することを確認します。

5. [SSH public key] (SSH パブリックキー) に SSH キーペアの SSH パブリックキー部分を入力します。

   新しいユーザーを追加する前に、サービスによってキーが検証されます。

> **Note**
>
> SSH キーペアを生成する手順については、「サービス管理ユーザーの SSH キーの生成」を参照してください。

6. (オプション) ユーザーに任意のタグを入力します。[Key] (キー) と [Value] (値) にキーバリューペアとして 1 つ以上のタグを入力して [Add tag] (タグの追加) を選択します。
7. [Add] (追加) を選択して、選択したサーバーに新しいユーザーを追加します。

   [Server details] (サーバーの詳細) ページの [Users] (ユーザー) セクションに新しいユーザーが表示されます。

初めて SFTP で Transfer Family サーバーに接続する際に発生する可能性がある問題:

- `sftp` コマンドを実行した場合にプロンプトが表示されず、次のメッセージが表示されることがあります。

  ```
  Couldn't canonicalize: Permission denied

  Need cwd
  ```

  この場合、ユーザーのロールのポリシー許可を増やす必要があります。AWS などの `AmazonElasticFileSystemClientFullAccess` マネージドポリシーを追加できます。
- `sftp` プロンプトで `pwd` を入力してユーザーのホームディレクトリを表示すると、以下のようなメッセージが表示されます。*`USER-HOME-DIRECTORY`*は SFTP ユーザーのホームディレクトリです。

  ```
  remote readdir("/USER-HOME-DIRECTORY"): No such file or directory
  ```

  この場合、親ディレクトリ (`cd ..`) をクリックし、ユーザーのホームディレクトリ (`mkdir` *`username`*) を作成します。

次のステップ— 次のステップについては、「クライアントを使用してサーバーエンドポイント経由でファイルを転送する」に進みます。

サービスマネージドユーザーの管理

このセクションでは、ユーザーのリストを表示する方法、ユーザーの詳細を編集する方法、および SSH パブリックキーを追加する方法について説明します。

- ユーザーリストを表示する
- ユーザーの詳細を表示または編集する
- ユーザーを削除する
- SSH パブリックキーを追加する
- SSH パブリックキーを削除する

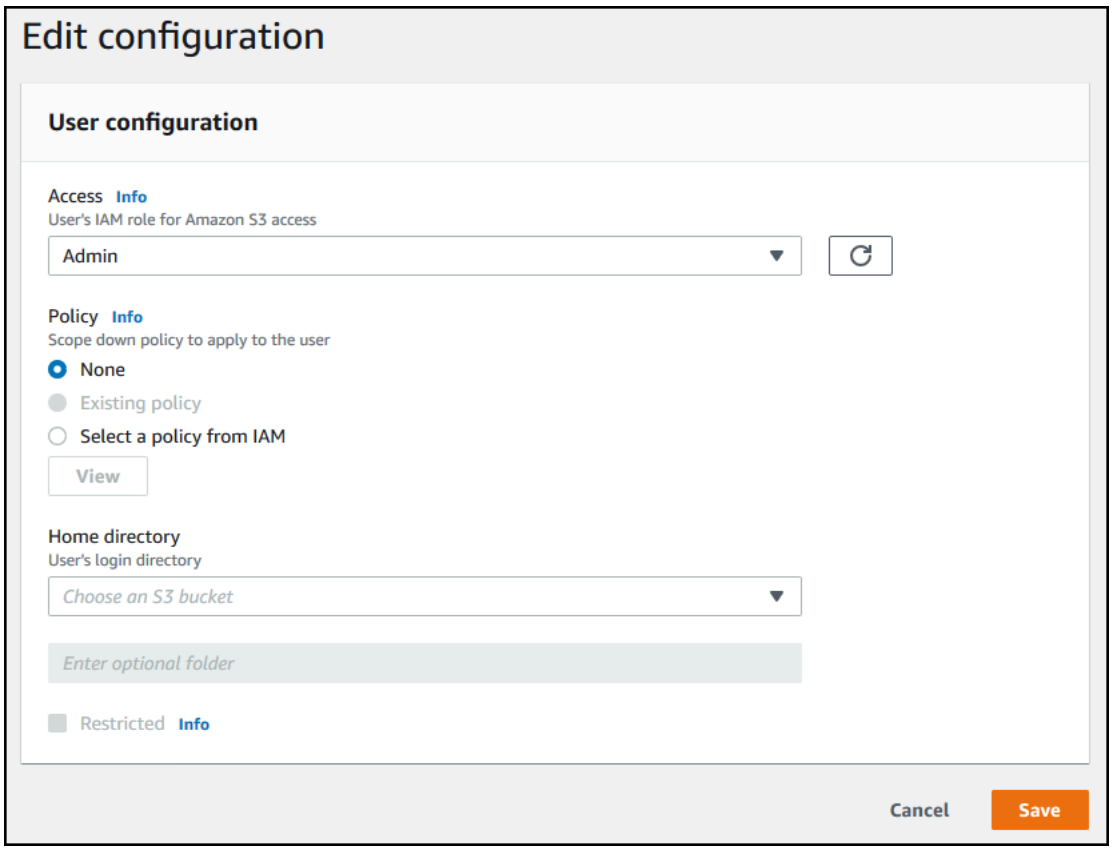
ユーザーのリストを検索するには

1. https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開きます。
2. ナビゲーションペインで [Servers] (サーバー) を選択して [Servers] (サーバー) ページを表示します。
3. [Server ID] (サーバー ID) 列で ID を選択すると、[Server Configuration] (サーバーの構成) ページが表示されます。
4. [Users] (ユーザー) の下にユーザーのリストが表示されます。

ユーザーの詳細を表示または編集するには

1. https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開きます。
2. ナビゲーションペインで [Servers] (サーバー) を選択して [Servers] (サーバー) ページを表示します。
3. [Server ID] (サーバー ID) 列で ID を選択すると、[Server Configuration] (サーバーの構成) ページが表示されます。
4. [ユーザー] でユーザー名を選択し、[ユーザーの詳細] ページを見ます。

   このページで [Edit] (編集) を選択すると、ユーザーのプロパティを変更できます。
5. [User details] (ユーザーの詳細) ページで [User configuration] (ユーザー設定) の隣にある [Edit] (編集) を選択します。



6. [Edit onfigu] (設定の編集) ページで [Access] (アクセス) について、以前に作成した Amazon S3 バケットへのアクセスを提供する IAM ロールを選択します。

   この IAM ロールは、「IAM ポリシーとロールを作成する」の手順に従って作成されました。その IAM ロールには、Amazon S3 バケットへのアクセスを提供する IAM ポリシーが含まれます。別の IAM ポリシーで定義された AWS Transfer Family サービスとの信頼関係も含まれます。

7. (オプション) [Policy] (ポリシー) で、次のいずれかを選択します。

   - None (なし)
   - Existing policy (既存のポリシー)
   - 既存のポリシーを選択するには、IAM からから ポリシーを選択します。[View] (表示) を選択すると、ポリシーの詳細を含む JSON オブジェクトが表示されます。

   セッションポリシーの詳細については、「IAM ポリシーとロールを作成する」を参照してください。セッションポリシーの作成方法の詳細については、「Amazon S3 バケットのセッションポリシーの作成」を参照してください 。

8. [Home Directory] (ホームディレクトリ) で AWS Transfer Family を使用して転送するデータを保存する Amazon S3 バケットを選択します。ユーザーが SFTP クライアントを使用してログインしたときにアクセスする home ディレクトリのパスを入力します。

   このパラメータを空白のままにした場合、Amazon S3 バケットの root ディレクトリが使用されます。この場合、IAM ロールが root ディレクトリへのアクセスを提供することを確認します。

   > **Note**
   >
   > セッションポリシーを効果的に使用できるように、ユーザーのユーザー名が含まれるディレクトリパスを選択することをお勧めします。セッションポリシーは、Amazon S3 バケットでのユーザーアクセスを home ディレクトリに制限します。

9. (オプション) [Restricted] (制限) チェックボックスをオンにすると、ユーザーはそのフォルダの外部にあるものにアクセスしたり、Amazon S3 バケット名やフォルダ名を表示したりできなくなります。

   > **Note**
   >
   > ユーザーにホームディレクトリを割り当てて、ユーザーをそのホームディレクトリに制限する場合、指定したフォルダへのユーザーのアクセスをロックダウンするにはこれで十分なはずです。さらなるコントロールの適用が必要な場合、セッションポリシーを使用します。

10. [Save] (保存) を選択して変更内容を保存します。

ユーザーを削除するには

1. https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開きます。
2. ナビゲーションペインで [Servers] (サーバー) を選択して [Servers] (サーバー) ページを表示します。
3. [Server ID] (サーバー ID) 列で ID を選択すると、[Server Configuration] (サーバーの構成) ページが表示されます。
4. [ユーザー] でユーザー名を選択し、[ユーザーの詳細] ページを見ます。
5. [ユーザーの詳細] ページで、ユーザー名の右にある[削除] を選択します。

6. 表示される確認ダイアログボックスで、「**delete**」という語を入力してから [Delete] (削除) を選択してユーザーを削除してよいことを確認します。

[Users] (ユーザー) リストからユーザーが削除されます。

ユーザーの SSH パブリックキーを追加するには

1. https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開きます。
2. ナビゲーションペインで、[Servers] (サーバー) を選択します。
3. [Server ID] (サーバー ID) 列で ID を選択すると、[Server Configuration] (サーバーの構成) ページが表示されます。
4. [ユーザー] でユーザー名を選択し、[ユーザーの詳細] ページを見ます。
5. [Add SSH public key] (SSH パブリックキーの追加) を選択して、新しい SSH パブリックキーをユーザーに追加します。

   > **Note**
   >
   > SSH キーは、Secure Shell (SSH) File Transfer Protocol (SFTP)が有効になっているサーバーでのみ使用されます。SSH キーペアを生成する方法については、「サービス管理ユーザーの SSH キーの生成」を参照してください。

6. [SSH public key] (SSH パブリックキー) に SSH キーペアの SSH パブリックキー部分を入力します。

   新しいユーザーを追加する前に、サービスによってキーが検証されます。SSH キーの形式は `ssh-rsa` *`string`* です。SSH キーペアを生成するには、「サービス管理ユーザーの SSH キーの生成」を参照してください。
7. [Add key] (キーの追加) を選択します。

ユーザーの SSH パブリックキーを削除するには

1. https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開きます。
2. ナビゲーションペインで、[Servers] (サーバー) を選択します。
3. [Server ID] (サーバー ID) 列で ID を選択すると、[Server Configuration] (サーバーの構成) ページが表示されます。
4. [ユーザー] でユーザー名を選択し、[ユーザーの詳細] ページを見ます。

5. パブリックキーを削除するには、その SSH キーチェックボックスを選択し、削除を選択します。

AWS Directory Service ID プロバイダーの使用

このトピックでは、 の AWS Directory Service ID プロバイダーを使用する方法について説明します AWS Transfer Family。

トピック

- の使用 AWS Directory Service for Microsoft Active Directory
- AWS Directory Service for Azure Active Directory ドメインサービスの使用

の使用 AWS Directory Service for Microsoft Active Directory

を使用して AWS Transfer Family 、 でファイル転送エンドユーザーを認証できます AWS Directory Service for Microsoft Active Directory。そうすることで、エンドユーザーの認証情報を変更したり、カスタムオーソライザーを必要とせずに、Active Directory 認証に依存するファイル転送ワークフローをシームレスに移行できます。

を使用すると AWS Managed Microsoft AD、Amazon Simple Storage Service (Amazon S3) または Amazon Elastic File System (Amazon EFS) に保存されているデータに対して、SFTP、FTPS、および FTP 経由で AWS Directory Service ユーザーおよびグループに安全にアクセスできます。Active Directory を使用してユーザーの認証情報を保存する場合、これらのユーザーがこれまでよりも簡単にファイル転送できるようになりました。

オンプレミス環境 AWS Managed Microsoft AD の または Active Directory コネクタを使用して AWS クラウドの Active Directory グループへのアクセスを提供できます。Microsoft Windows 環境で設定済みのユーザーに、 AWS クラウドまたはオンプレミスネットワークのいずれかで、 が ID AWS Managed Microsoft AD に使用する AWS Transfer Family サーバーへのアクセスを許可できます。

> **Note**
>
> - AWS Transfer Family シンプルADには対応しておりません。
> - Transfer Familyはクロスリージョンの Active Directory構成をサポートしていません。Transfer Familyサーバーと同じリージョンにあるActive Directory統合のみをサポートしています。

- Transfer Family は、既存の RADIUS ベースの MFA インフラストラクチャで多要素認証 (MFA) を有効にするための AWS Managed Microsoft AD または AD Connector の使用をサポートしていません。
- AWS Transfer Family は、 Managed Active Directory のレプリケートされたリージョンをサポートしていません。

を使用するには AWS Managed Microsoft AD、次のステップを実行する必要があります。

1. AWS Directory Service コンソールを使用して 1 つ以上の AWS Managed Microsoft AD ディレクトリを作成します。
2. Transfer Family コンソールを使用して、 を ID プロバイダー AWS Managed Microsoft AD として使用するサーバーを作成します。
3. 1 つ以上の AWS Directory Service グループからアクセスを追加します。
4. 必須ではありませんが、ユーザーアクセスのテストと検証をお勧めします。

トピック

の使用を開始する前に AWS Directory Service for Microsoft Active Directory

ADグループに固有の識別子を提供してください。

を使用する前に AWS Managed Microsoft AD、Microsoft AD ディレクトリ内のグループごとに一意の識別子を指定する必要があります。そのためには、各グループごとのセキュリティ識別子 (SID) を使用します。関連付けるグループのユーザーは、 AWS Transfer Family を使用して、有効なプロトコル経由で Amazon S3 または Amazon EFS リソースにアクセスできます。

次の Windows PowerShell コマンドを使用してグループの SID を取得し、 をグループの名前*YourGroupName*に置き換えます。

```
Get-ADGroup -Filter {samAccountName -like "YourGroupName*"} -Properties * | Select
 SamAccountName,ObjectSid
```

> **Note**
>
> を ID プロバイダー AWS Directory Service として使用しており、 `userPrincipalName`との値が異なる場合、 `SamAccountName`は の値 AWS Transfer Family を受け入れますSamAccountName。Transfer Familyでは`userPrincipalName` . で指定した値は受け付けません。

ロールにアクセス AWS Directory Service 許可を追加する

ID プロバイダー AWS Directory Service として使用する AWS Directory Service API アクセス許可も必要です。以下のパーミッションが必要または推奨される：

- `ds:DescribeDirectories` Transfer Family がディレクトリを検索するために必要です
- `ds:AuthorizeApplication` Transfer Family の承認を追加する必要があります
- `ds:UnauthorizeApplication` サーバー作成プロセス中に問題が発生した場合に備えて、暫定的に作成されたリソースを削除することをお勧めします。

Transfer Family サーバーの作成に使用しているロールにこれらの権限を追加します。これらの権限の詳細については、「AWS Directory Service API 権限: アクション、リソース、および条件のリファレンスを参照してください」。

Active Directoryレルムでの作業

Active Directory AWS Transfer Family ユーザーにサーバーにアクセスさせる方法を検討するときは、ユーザーのレルムとそのグループのレルムに留意してください。理想的には、ユーザーのレルムとそのグループのレルムが一致している必要があります。つまり、ユーザーとグループの両方がデフォルト レルム内にあるか、両方とも信頼されたレルム内にあります そうでない場合、ユーザーは Transfer Family によって認証されません。

ユーザーをテストして、構成が正しいことを確認できます。詳細については、「ユーザーのテスト」を参照してください。ユーザー/グループ レルムに問題がある場合は、ユーザーのグループに関連付けられたアクセスが見つかりませんというエラーが表示されます。

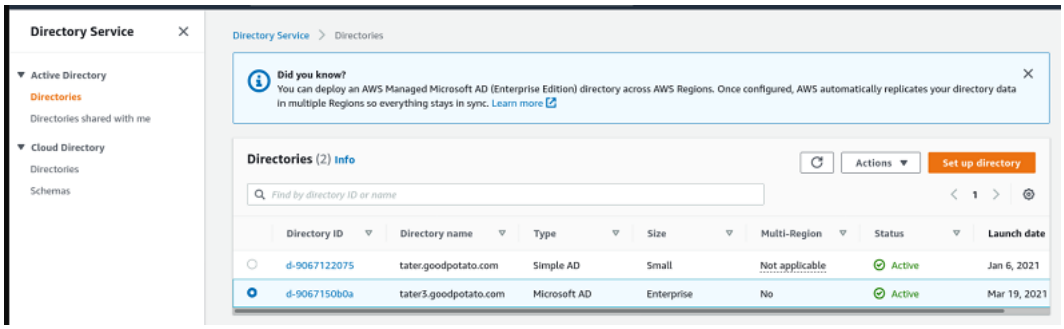
を ID プロバイダー AWS Managed Microsoft AD として選択する

このセクションでは、サーバー AWS Directory Service for Microsoft Active Directory で を使用する方法について説明します。

Transfer Family AWS Managed Microsoft AD で を使用するには

1. にサインイン AWS Management Console し、https://console.aws.amazon.com/directoryservicev2/ で AWS Directory Service コンソールを開きます。

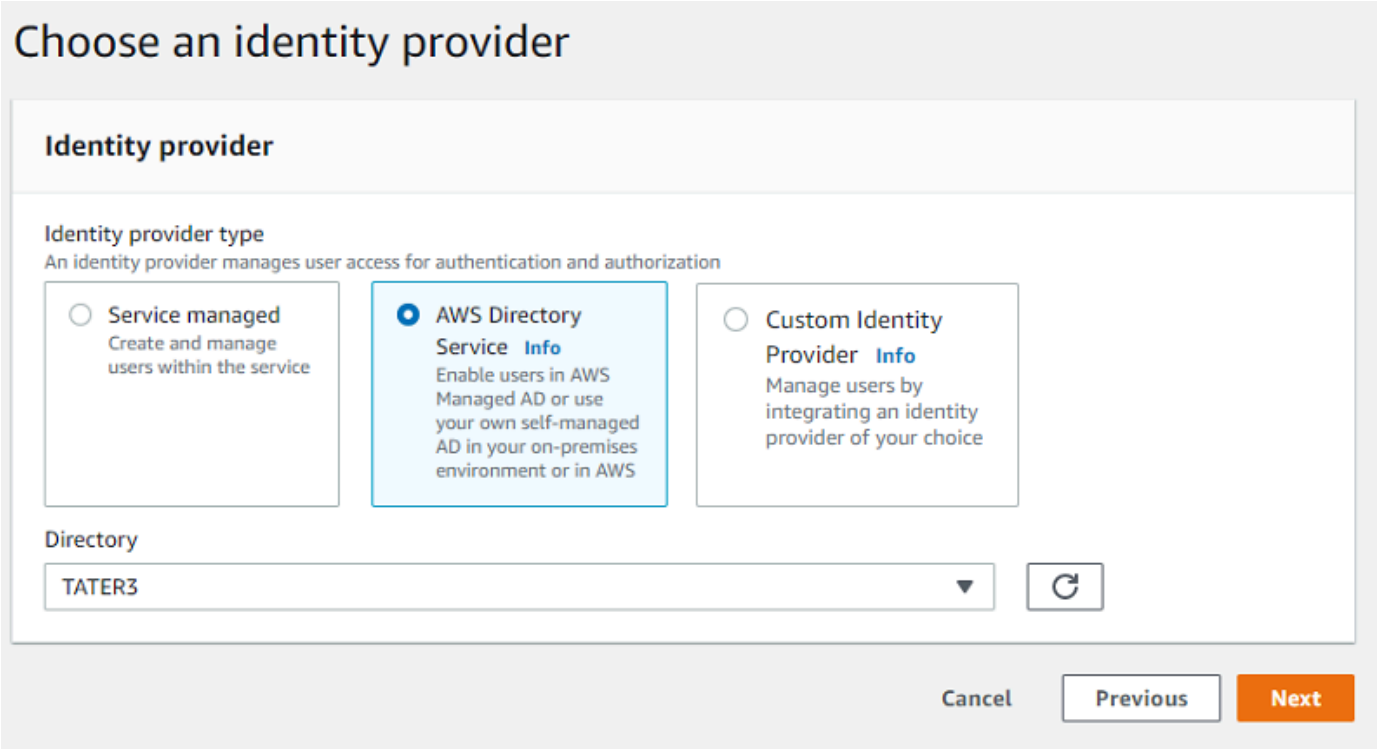
   AWS Directory Service コンソールを使用して、1 つ以上のマネージドディレクトリを設定します。詳細については、 AWS Directory Service 管理者ガイドの AWS Managed Microsoft AD を参照してください。



2. https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開き、サーバーの作成を選択します。
3. [Choose protocols] (プロトコルの選択) ページのリストから 1 つ以上のプロトコルを選択します。

   > **Note**
   >
   > FTPS を選択した場合、 AWS Certificate Manager 証明書を提供する必要があります。

4. [Choose an identity provider] (ID プロバイダーを選択する) で[AWS Directory Service] を選択します。



5. [Directory] (ディレクトリ) リストには、設定したすべてのマネージドディレクトリが含まれます。リストからディレクトリを選択し、[Next] (次へ) を選択します。

> **Note**
>
> - クロスアカウントディレクトリと共有ディレクトリは、 ではサポートされていません AWS Managed Microsoft AD。
> - Directory Service を ID プロバイダーとしてサーバーを設定するには、いくつかの AWS Directory Service アクセス許可を追加する必要があります。詳細については、「の使用を開始する前に AWS Directory Service for Microsoft Active Directory」を参照してください。

6. サーバーの作成を終了するには、以下のいずれかの手順に従います。

- SFTP 対応サーバーの作成
- FTPS 対応サーバーを作成する
- FTP 対応サーバーの作成

これらの手順では、ID プロバイダーを選択するステップに進みます。

> **Important**
>
> Transfer Family サーバーで Microsoft AD ディレクトリを使用した AWS Directory Service 場合、 で Microsoft AD ディレクトリを削除することはできません。まずサーバーを削除してから、ディレクトリを削除する必要があります。

グループへのアクセス権の付与

サーバーを作成したら、 を使用して有効なプロトコル経由でファイルをアップロードおよびダウンロードするためのアクセス権をディレクトリ内のどのグループに付与するかを選択する必要があります AWS Transfer Family。そのためには、アクセス権を作成します。

> **Note**
>
> ユーザーはアクセスを付与されたグループに直接属していなければなりません。たとえば、Bob というユーザーが GroupA に属し、GroupA 自体が GroupB に含まれているとします。
>
> - GroupA へのアクセス権を付与すると、Bob にはアクセス権が付与されます。
> - GroupB にアクセス権を付与する (そして GroupA しない) と、Bob にはアクセス権がありません。

グループへのアクセス権を付与するには

1. https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開きます。
2. サーバーの詳細ページに移動します。
3. [アクセス] セクションで、[アクセスの追加] を選択します。
4. このサーバーへのアクセスを許可する AWS Managed Microsoft AD ディレクトリの SID を入力します。

   > **Note**
   >
   > グループの SID を検索する方法については、「the section called “の使用を開始する前に AWS Directory Service for Microsoft Active Directory”」を参照してください。

5. アクセス で、グループの AWS Identity and Access Management (IAM) ロールを選択します。

6. [Policy] (ポリシー) セクションでポリシーを選択します。デフォルト設定は [None] (なし) です。
7. [Home directory] (ホームディレクトリ) で、グループのホームディレクトリに対応する S3 バケットを選択します。

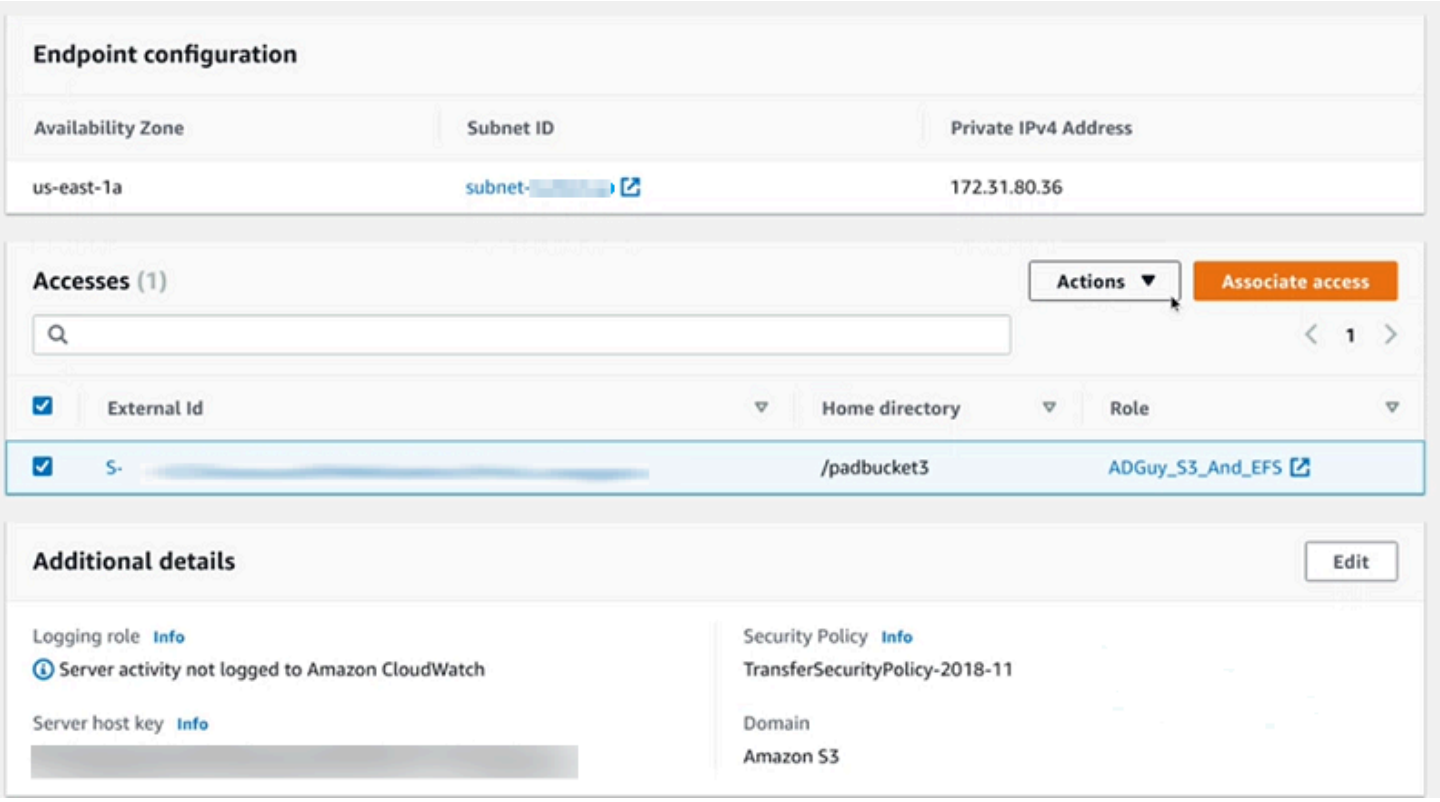
> **Note**
>
> セッションポリシーを作成して、ユーザーに表示されるバケットの部分を限定できます。たとえば、ユーザーを `/filetest` ディレクトリの下位にある各自のフォルダに限定するには、フィールドに次のテキストを入力します。
>
> ```
> /filetest/${transfer:UserName}
> ```
>
> セッションポリシーの作成方法の詳細については、「Amazon S3 バケットのセッションポリシーの作成」を参照してください 。

8. [Add] (追加) をクリックして関連付けを作成します。
9. サーバーを選択します。
10. 「アクセスの追加」を選択します。
    - グループの SID を入力します。

    > **Note**
    >
    > DID を見つける方法については、「the section called “の使用を開始する前に AWS Directory Service for Microsoft Active Directory”」を参照してください。

11. 「アクセスの追加」を選択します。

[Accesses] (アクセス) セクションにサーバアクセス権が一覧表示されます。



ユーザーのテスト

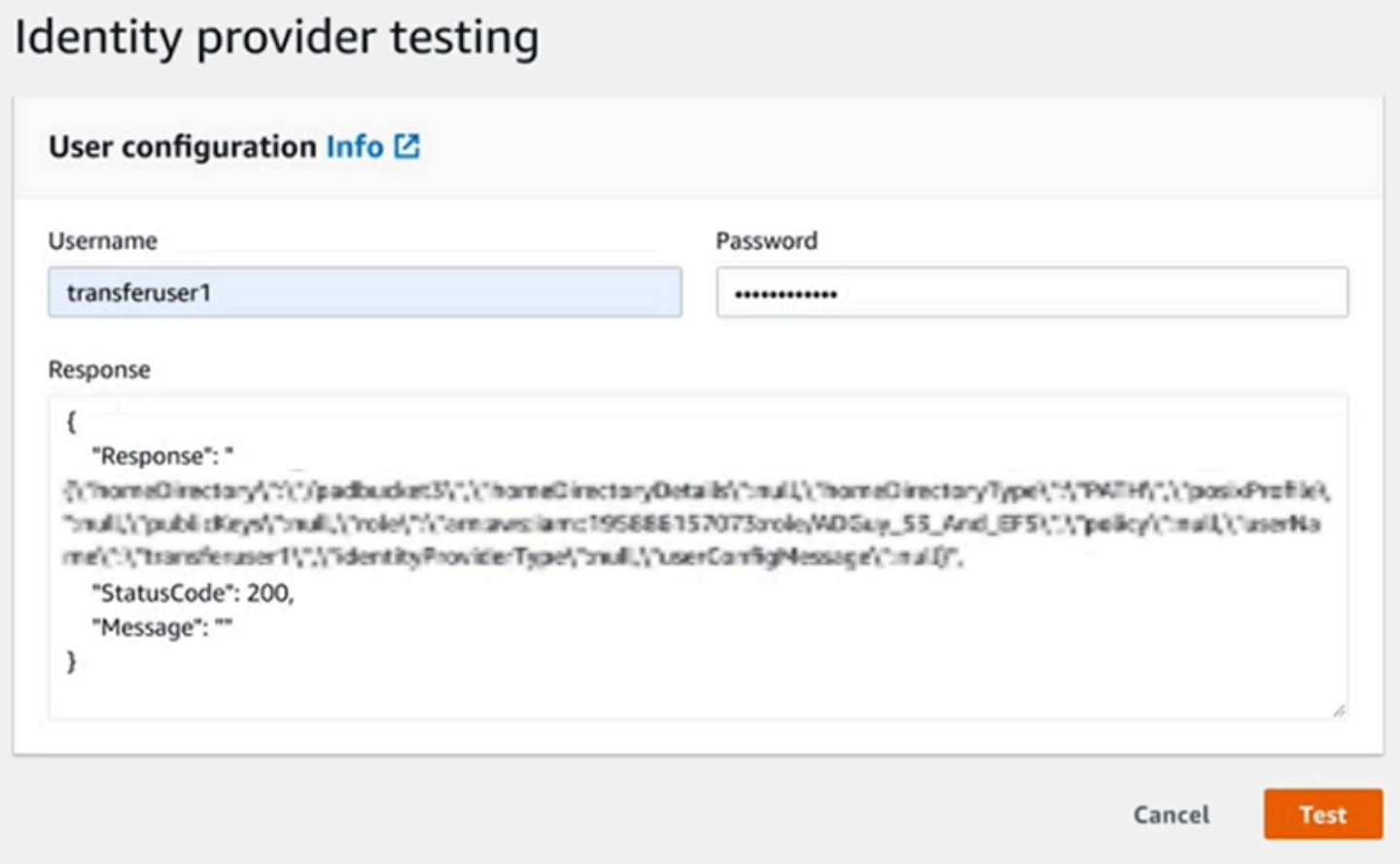
ユーザーがサーバーの AWS Managed Microsoft AD ディレクトリにアクセスできるかどうかをテストできます。

> **Note**
>
> ユーザーは、[Endpoint configuration] (エンドポイント設定) ページの [Access] (アクセス) セクションのリストにあるいずれかのグループ (外部 ID) に属している必要があります。ユーザーがグループに属していない場合、または複数のグループに属している場合、そのユーザーにはアクセス権が付与されません。

特定のユーザーがアクセス権を持っているかどうかをテストするには

1. サーバーの詳細ページで [Actions] (アクション) を選択してから [Test] (テスト) を選択します。
2. [ID プロバイダのテスト] には、アクセス権を持つグループの 1 つに属しているユー ザーのサインイン認証情報を入力します。
3. [Test] (テスト) を選択します。

選択したユーザーにサーバーへのアクセスが許可されていることを示す、ID プロバイダーのテストが成功しました。



ユーザーがアクセス権のある複数のグループに属している場合、次のレスポンスが表示されます。

```
"Response":"",
"StatusCode":200,
"Message":"More than one associated access found for user's groups."
```

グループのサーバーアクセスの削除

グループのサーバーアクセスを削除するには

1. サーバーの詳細ページで [Actions] (アクション) を選択してから [Delete Access] (アクセスの削除) を選択します。
2. ダイアログボックスで、このグループのアクセスを削除したいことを確認します。

サーバーの詳細ページに戻ると、このグループのアクセス権がリストから消えたことがわかります。

SSH (Secure Shell) を使用してサーバーに接続する

サーバーとユーザーを設定したら、SSH を使ってサーバーに接続し、アクセス権を持つユーザーの完全修飾ユーザー名を使うことができます。

```
sftp user@active-directory-domain@vpc-endpoint
```

例: `transferuserexample@mycompany.com@vpce-0123456abcdef-789xyz.vpc-svc-987654zyxabc.us-east-1.vpce.amazonaws.com`。

この形式は、潜在的に大規模な Active Directory の検索を限定し、フェデレーションの検索を対象とします。

> **Note**
>
> 単純なユーザー名を指定することができます。ただし、この場合、Active Directory コードはフェデレーション内のすべてのディレクトリを検索する必要があります。そうすると検索が限定され、ユーザーにアクセス権があっても認証が失敗する可能性があります。

認証後、ユーザーは、ユーザーの設定時に指定されたホームディレクトリ内にいます。

フォレストと信頼を使用したセルフマネージド Active Directory AWS Transfer Family への接続

セルフマネージド Active Directory (AD) のユーザーは、 AWS アカウント および Transfer Family サーバーへのシングルサインオンアクセス AWS IAM Identity Center に を使用することもできます。そのために、 AWS Directory Service には以下のオプションがあります。

- 一方向のフォレストの信頼 (AWS Managed Microsoft AD オンプレミスの Active Directory との間で送受信) は、ルートドメインでのみ機能します。
- 子ドメインの場合、以下のいずれかを使用できます:
  - AWS Managed Microsoft AD とオンプレミスの Active Directory の間で双方向の信頼を使用する
  - 各子ドメインに対して一方向の外部信頼を使用します。

信頼できるドメインを使用してサーバーに接続する場合、ユーザーは信頼できるドメインを指定する必要があります (例: `transferuserexample@mycompany.com`)。

AWS Directory Service for Azure Active Directory ドメインサービスの使用

- SFTP 転送のニーズに合わせて既存の Active Directory フォレストを利用するには、Active Directory Connector を使用できます。
- フルマネージド サービスで Active Directory と高可用性のメリットを享受したい場合は、 AWS Directory Service for Microsoft Active Directoryを使用できます。詳細については、「AWS Directory Service ID プロバイダーの使用」を参照してください。

このトピックでは、Active Directory Connectorと 「Azure Active Directory Domain Services (Azure ADDS)」 を使用して、 「Azure Active Directory」 で SFTP Transfer ユーザーを認証する方法について説明します。

トピック

AWS Directory Service for Azure Active Directory Domain Services の使用を開始する前に

には AWS、以下が必要です。

- Transfer Family サーバーを使用している AWS リージョンの Virtual Private Cloud (VPC)
- VPC 内に少なくとも 2 つのプライベート サブネット
- VPC はインターネットに接続していなければならない。
- Microsoft Azure との site-to-site VPN 接続用のカスタマーゲートウェイと仮想プライベートゲートウェイ

Microsoft Azure の場合は、次のものが必要です

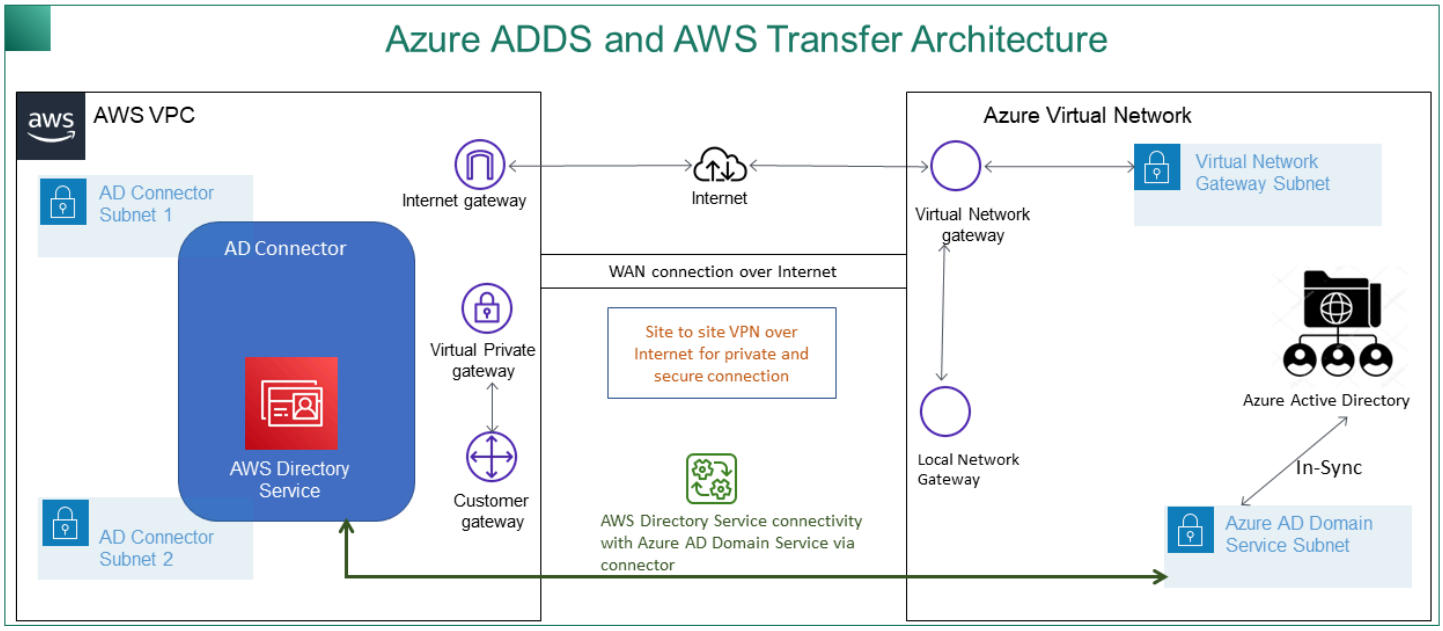
- Azure Active Directory および Active Directory ドメイン サービス (Azure ADDS)
- Azure リソース グループ

- Azure仮想ネットワーク
- Amazon VPC と Azure リソース グループ間の VPN 接続

> **Note**
>
> これは、ネイティブ IPSEC トンネル経由、または VPN アプライアンスを使用して行うことができます。このトピックでは、Azure 仮想ネットワーク ゲートウェイとローカル ネットワーク ゲートウェイの間で IPSEC トンネルを使用します。トンネルは、Azure ADDS エンドポイントと AWS VPC を格納するサブネット間のトラフィックを許可するように設定する必要があります。

- Microsoft Azure との site-to-site VPN 接続用のカスタマーゲートウェイと仮想プライベートゲートウェイ

次の図は、開始する前に必要な構成を示しています。

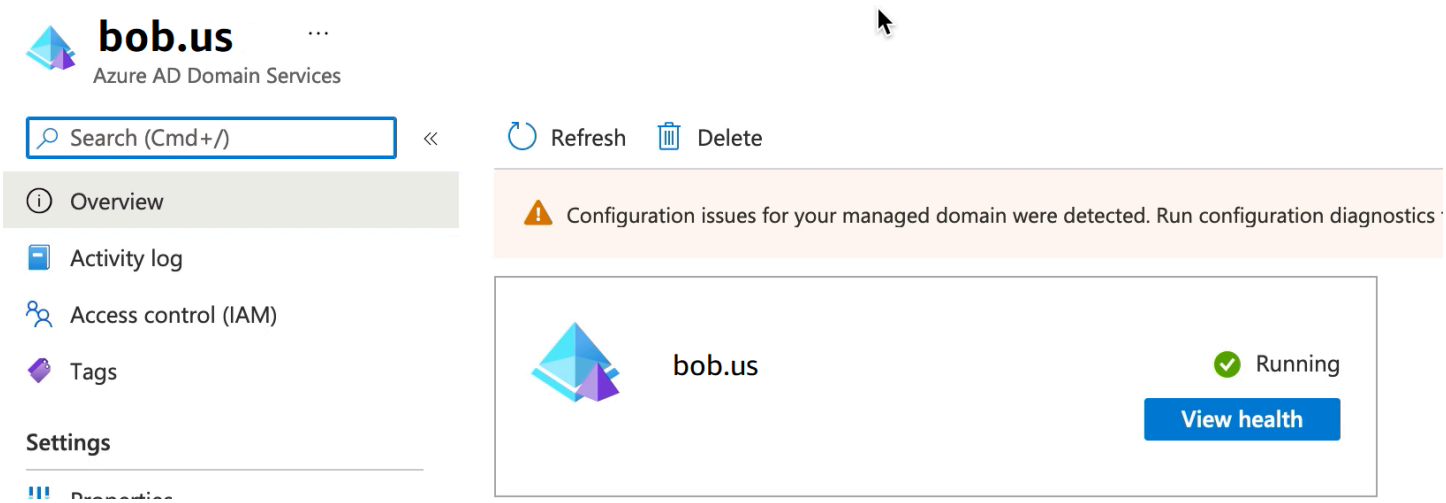


ステップ 1: Azure Active Directory ドメイン サービスを追加する

Azure AD は、デフォルトではドメイン参加インスタンスをサポートしません。ドメイン参加などのアクションを実行したり、グループ ポリシーなどのツールを使用したりするには、管理者は Azure Active Directory ドメイン サービスを有効にする必要があります。Azure AD DS をまだ追加していない場合、または既存の実装が SFTP 転送サーバーで使用するドメインに関連付けられていない場合は、新しいインスタンスを追加する必要があります。

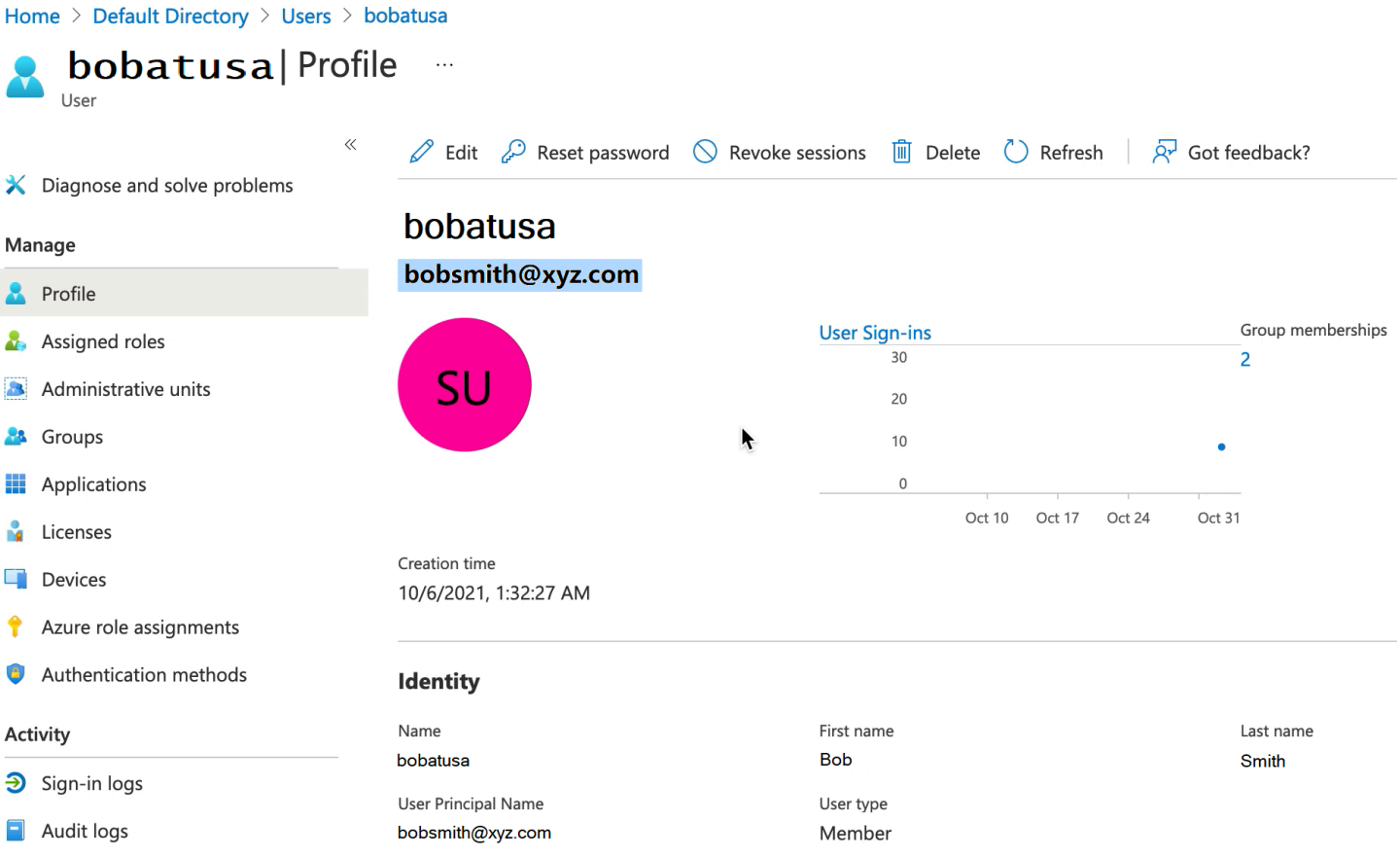
Azure Active Directory Domain Services (Azure ADDS) を有効にする方法については、「チュートリアル: Azure Active Directory Domain Services マネージド ドメインを作成および構成する」を参照してください」。

> **Note**
>
> Azure ADDS を有効にする場合は、SFTP 転送サーバーの接続先のリソース グループと Azure AD ドメインに対して Azure ADDS が構成されていることを確認してください。



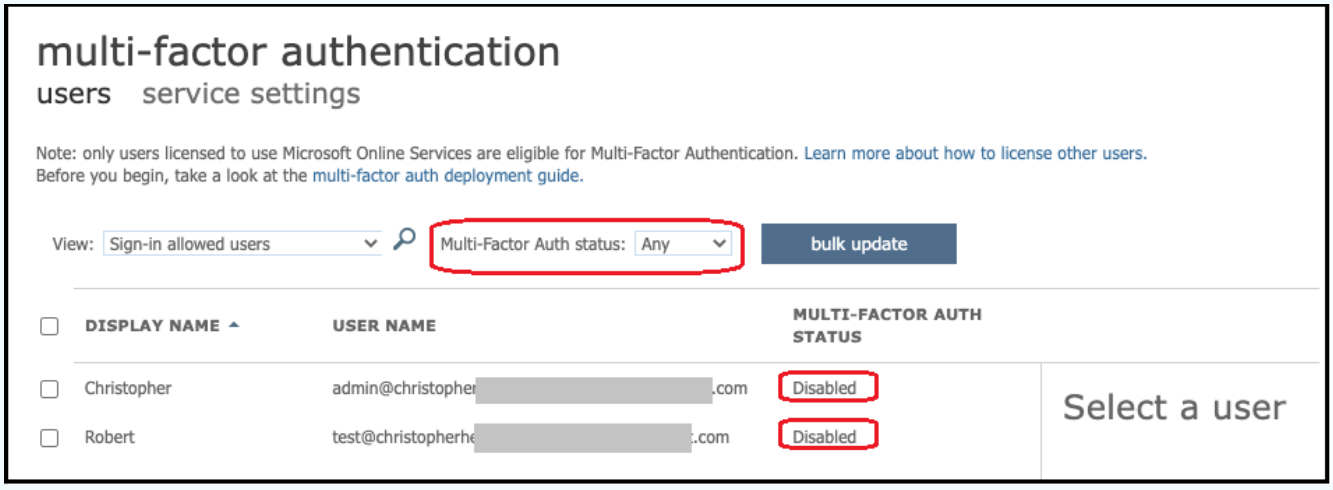
ステップ 2：サービスアカウントの作成

Azure AD には、Azure ADDS の管理者グループの一部であるサービス アカウントが 1 つ必要です。このアカウントは AWS Active Directory コネクタで使用されます。このアカウントが Azure ADDS と同期していることを確認してください。



Tip

Azure Active Directory の多要素認証は、SFTP プロトコルを使用する Transfer Family サーバーではサポートされていません。Transfer Family サーバーは、ユーザーが SFTP に対して認証された後、MFA トークンを提供できません。接続を試行する前に、必ず MFA を無効にしてください。



ステップ 3: AD Connector を使用して AWS ディレクトリを設定する

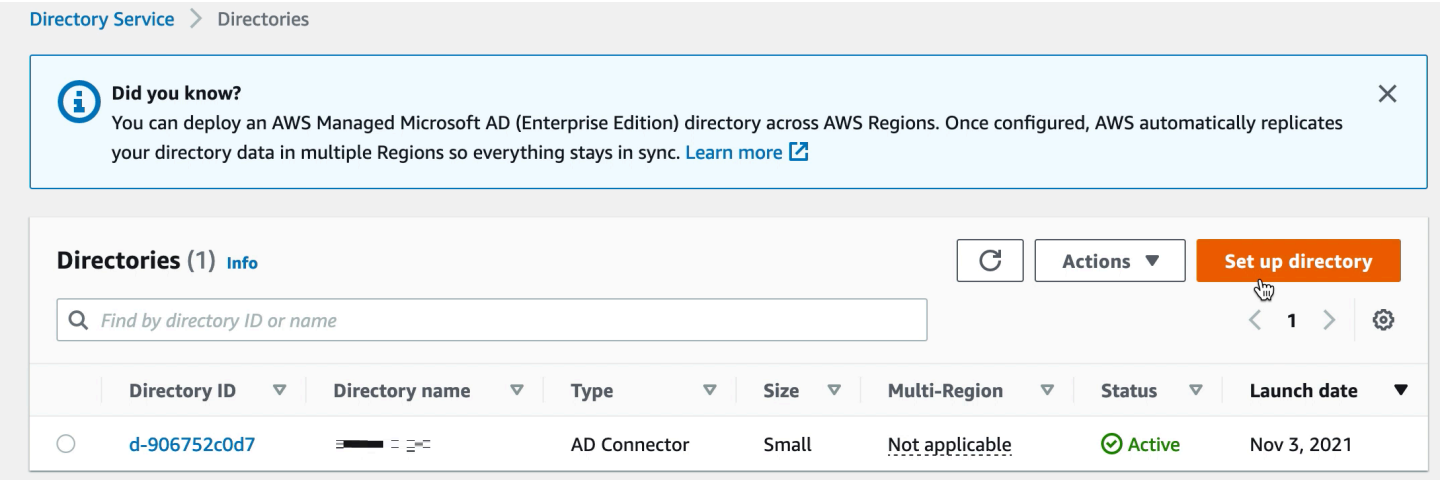
Azure ADDS を設定し、 AWS VPC と Azure 仮想ネットワークの間に IPSEC VPN トンネルを持つサービスアカウントを作成したら、任意の AWS EC2 インスタンスから Azure ADDS DNS IP アドレスに ping を送信することで接続をテストできます。

接続がアクティブであることを確認したら、以下に進むことができます。

AD Connector を使用して AWS ディレクトリを設定するには

1. 「ディレクトリ サービス」コンソールを開き、[ディレクトリ] を選択します。
2. [Set up directory] (ディレクトリをセットアップする) を選択します。
3. ディレクトリタイプには、[AD Connector] を選択します。
4. ディレクトリ サイズを選択し、[次へ] を選択して、VPC とサブネットを選択します。
5. [Next] を選択し、次のようにフィールドに入力します。
   - [ディレクトリ DNS 名] : Azure ADDS に使用しているドメイン名を入力します。
   - [DNS IP アドレス] : Azure ADDS IP アドレスを入力します
   - [サーバー アカウントのユーザー名] [とパスワード]: ステップ 2: サービス アカウントを作成するで作成したサービス アカウントの詳細を入力します。
6. 画面に入力してディレクトリ サービスを作成します

これで、ディレクトリのステータスが[アクティブ] になり、SFTP 転送サーバーで使用する準備が整いました。



ステップ 4: AWS Transfer Family サーバーをセットアップする

SFTP プロトコルと[AWS ディレクトリ サービス] の ID プロバイダー タイプを使用して Transfer Family サーバーを作成します。ディレクトリのドロップダウンリストから、「ステップ 3: AD Connector を使用して AWS ディレクトリをセットアップする」で追加したディレクトリを選択します。

> **Note**
>
> Transfer Family サーバーで Microsoft AD ディレクトリを使用した場合、 AWS Directory Service で Microsoft AD ディレクトリを削除することはできません。まずサーバーを削除してから、ディレクトリを削除する必要があります。

ステップ5：グループへのアクセス権の付与

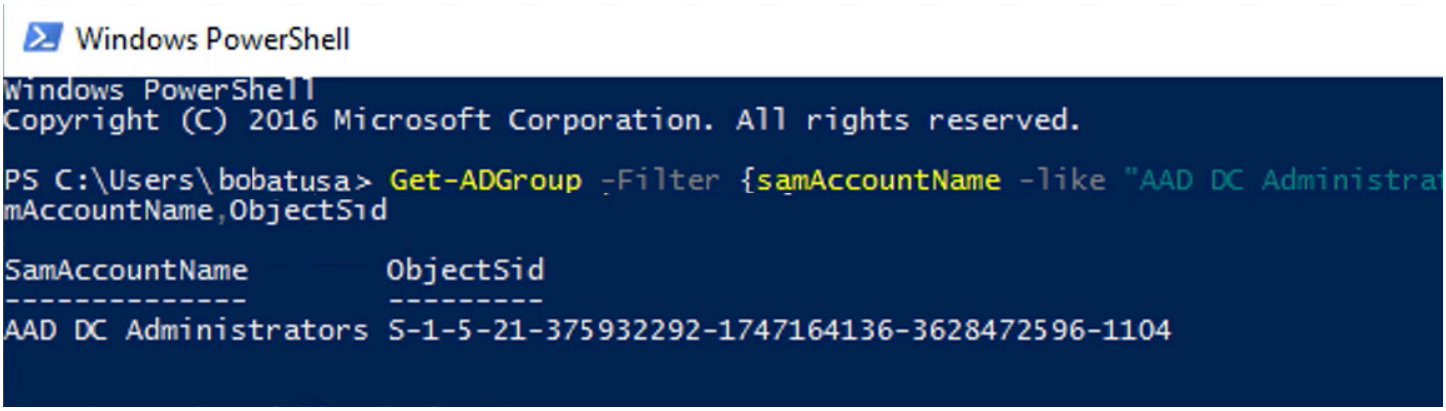
サーバーを作成したら、 を使用して有効なプロトコル経由でファイルをアップロードおよびダウンロードするためのアクセス権をディレクトリ内のどのグループに付与するかを選択する必要があります AWS Transfer Family。そのためには、アクセス権を作成します。

> **Note**
>
> ユーザーはアクセスを付与されたグループに直接属していなければなりません。たとえば、Bob というユーザーが GroupA に属し、GroupA 自体が GroupB に含まれているとします。
>
> - GroupA へのアクセス権を付与すると、Bob にはアクセス権が付与されます。
> - GroupB にアクセス権を付与する (そして GroupA しない) と、Bob にはアクセス権がありません。

アクセスを許可するには、グループの SID を取得する必要があります。

次の Windows PowerShell コマンドを使用してグループの SID を取得し、 をグループの名前*YourGroupName*に置き換えます。

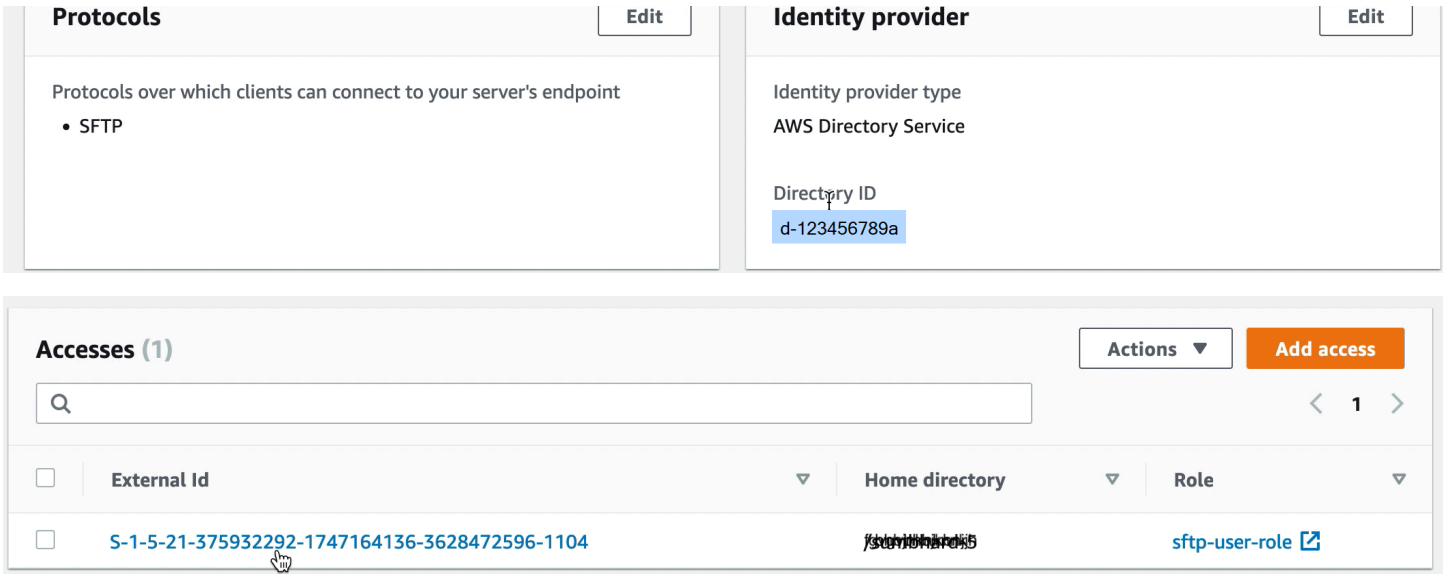
```
Get-ADGroup -Filter {samAccountName -like "YourGroupName*"} -Properties * | Select
 SamAccountName,ObjectSid
```



グループへのアクセスを許可する

1. https://console.aws.amazon.com/transfer/ を開きます。
2. サーバーの詳細ページに移動し、[アクセス] セクションで[アクセスの追加] を選択します。
3. 前の手順の出力から受け取ったSIDを入力してください。
4. アクセス で、グループの AWS Identity and Access Management ロールを選択します。
5. [Policy] (ポリシー) セクションでポリシーを選択します。デフォルト値は [なし] です。
6. [Home directory] (ホームディレクトリ) で、グループのホームディレクトリに対応する S3 バケットを選択します。
7. [Add] (追加) をクリックして関連付けを作成します。

Transferサーバーの詳細は、次のように見えるべきです:



ステップ6: ユーザーのテスト

ユーザーがサーバーの AWS Managed Microsoft AD ディレクトリにアクセスできるかどうかをテスト (ユーザーのテスト) できます。ユーザーは、[Endpoint configuration] (エンドポイント設定) ページの [Access] (アクセス) セクションのリストにあるいずれかのグループ (外部 ID) に属している必要があります。ユーザーがグループに属していない場合、または複数のグループに属している場合、そのユーザーにはアクセス権が付与されません。

カスタム ID プロバイダーの使用

ユーザーを認証するには、既存の ID プロバイダーを で使用できます AWS Transfer Family。ID プロバイダーは、Amazon S3 または Amazon Elastic File System (Amazon EFS) へのアクセスを認証および承認する AWS Lambda 関数を使用して統合します。詳細については、「AWS Lambda を使用して ID プロバイダーを統合する」を参照してください。また、 AWS Transfer Family マネジメントコンソールで転送されたファイル数やバイト数などのメトリクスのグラフにアクセス CloudWatch することもできます。これにより、一元化されたダッシュボードを使用してファイル転送をモニタリングする 1 つのペインが提供されます。

または、単一の Amazon API Gateway メソッドで RESTful インターフェイスを提供することもできます。Transfer Family は、このメソッドを呼び出して ID プロバイダーに接続し、ID プロバイダーは Amazon S3 または Amazon EFS へのアクセスを認証および承認します。ID プロバイダーを統合するために RESTful API が必要な場合、または AWS WAF を使用してその機能をジオブロッキングまたはレート制限リクエストに活用する場合は、このオプションを使用します。詳細については、「Amazon API Gateway を使用して ID プロバイダーを統合する」を参照してください。

いずれの場合も、「AWS Transfer Family コンソール」または「CreateServer」API 操作を使って新しいサーバーを作成することができます。

> **Note**
>
> Transfer Family は、ブログ記事と、ファイル転送ソリューションの構築を順を追って説明するワークショップを提供します。このソリューションは、マネージド SFTP/FTPS エンドポイントに 、ユーザー管理 AWS Transfer Family に Amazon Cognito と DynamoDB を活用します。
> ブログ記事は、「 AWS Transfer Family および Amazon S3 での ID プロバイダーとしての Amazon Cognito Amazon S3の使用」で入手できます。ワークショップの詳細はこちらで確認できます。

AWS Transfer Family には、カスタム ID プロバイダーを操作するための以下のオプションが用意されています。

- AWS Lambda を使用して ID プロバイダーを接続する – Lambda 関数にバックアップされた既存の ID プロバイダーを使用できます。Lambda 関数の名前を指定します。詳細については、「AWS Lambda を使用して ID プロバイダーを統合する」を参照してください。
- 「Amazon API Gateway を使って ID プロバイダーに接続する」 — ID プロバイダーとして使用するために、Lambda 関数に裏打ちされた API ゲートウェイメソッドを作成することができます。Amazon API Gateway URL と呼び出しロールを指定します。詳細については、「Amazon API Gateway を使用して ID プロバイダーを統合する」を参照してください。

どちらのオプションでも、認証方法を指定することもできます。

- パスワードまたはキー – ユーザーはパスワードまたはキーを使用して認証できます。これは、デフォルト値です。
- パスワードのみ – ユーザーは接続するためにパスワードを入力する必要があります。
- キーのみ — ユーザーは接続するためにプライベートキーを指定する必要があります。
- パスワードとキー – ユーザーは接続するためにプライベートキーとパスワードの両方を指定する必要があります。サーバーは最初にキーを確認し、キーが有効な場合はパスワードの入力を求めます。提供されたプライベートキーが保存されたパブリックキーと一致しない場合、認証は失敗します。

複数の認証方法を使用してカスタム ID プロバイダーで認証する

Transfer Family サーバーは、複数の認証方法を使用するときに AND ロジックを制御します。Transfer Family は、これをカスタム ID プロバイダーへの 2 つの個別のリクエストとして扱います。ただし、その効果は結合されます。

認証を完了できるようにするには、両方のリクエストが適切なレスポンスで正常に返される必要があります。Transfer Family では、2 つのレスポンスを完了する必要があります。つまり、必要な要素 (Amazon EFS をストレージに使用する場合、ロール、ホームディレクトリ、ポリシー、POSIX プロファイル) がすべて含まれています。Transfer Family では、パスワードレスポンスにパブリックキーを含めないようにする必要があります。

パブリックキーリクエストには、ID プロバイダーとは別のレスポンスが必要です。パスワード または キー または パスワード および キー を使用する場合、その動作は変更されません。

SSH/SFTP プロトコルは、最初にパブリックキー認証を使用してソフトウェアクライアントにチャレンジし、次にパスワード認証を要求します。このオペレーションでは、ユーザーが認証を完了する前に、両方が成功することを義務付けています。

トピック

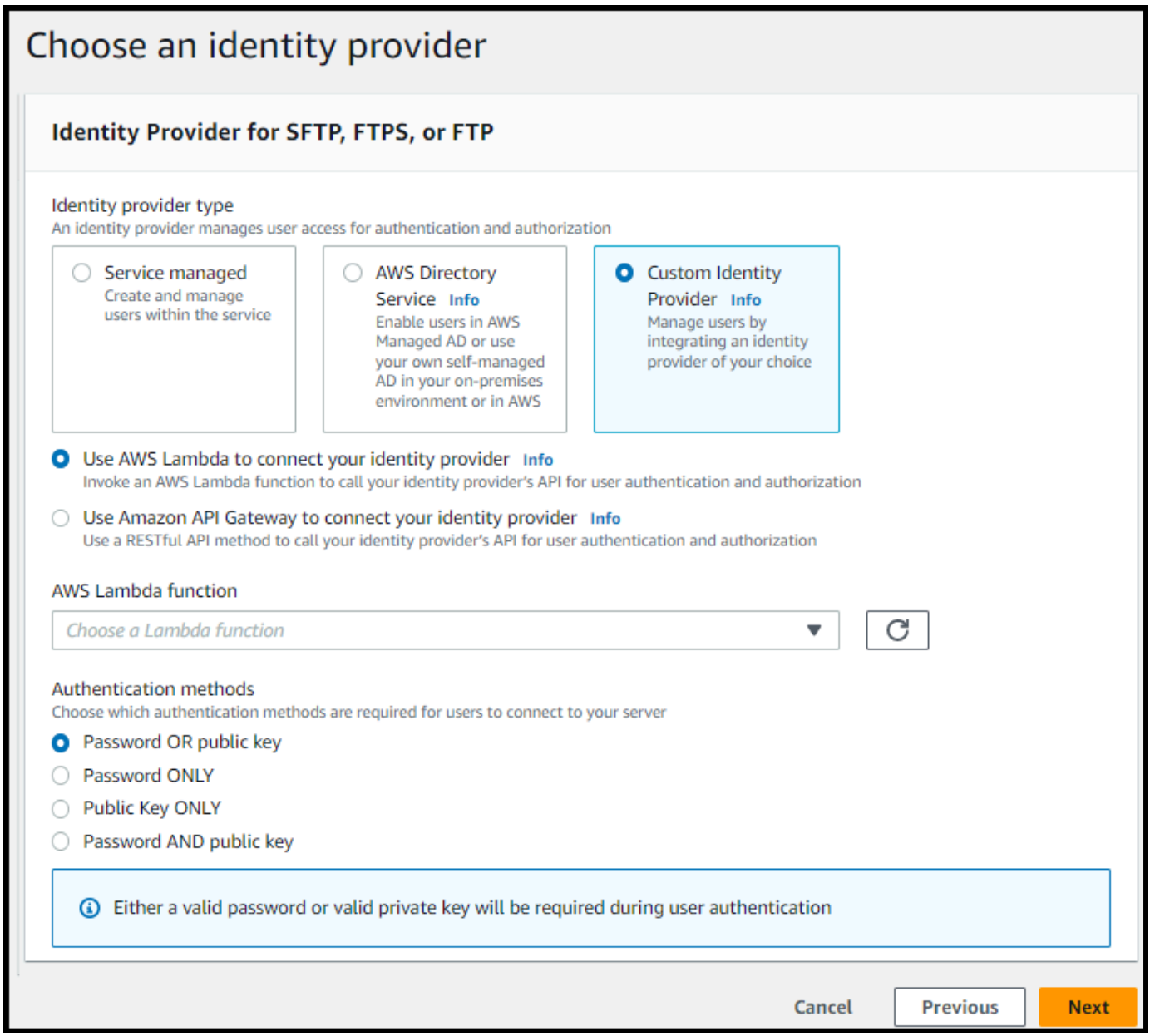
- AWS Lambda を使用して ID プロバイダーを統合する
- Amazon API Gateway を使用して ID プロバイダーを統合する

AWS Lambda を使用して ID プロバイダーを統合する

カスタム ID プロバイダーに接続する AWS Lambda 関数を作成します。Okta、Secrets Manager、、または認証ロジックを含むカスタムデータストアなど OneLogin、任意のカスタム ID プロバイダーを使用できます。

> **Note**
>
> Lambda を ID プロバイダーとして使用する Transfer Family サーバーを作成する前に、関数を作成する必要があります。サンプルの Lambda 関数については、「Lambda 関数の例」を参照してください。または、 のいずれかを使用する CloudFormation スタックをデプロイすることもできますLambda 関数のテンプレート。また、Lambda 関数が Transfer Family と信頼関係にあるリソースベースのポリシーを使用していることを確認してください。ポリシーの例については、「Lambda リソースベースのポリシー」を参照してください。

1. AWS Transfer Family コンソールを開きます。
2. [Create server] (サーバーの作成) を選択すると [Create server] (サーバーの作成) ページが開きます。[Choose an identity provider] (ID プロバイダーの選択) で [Custom Identity Provider] (カスタム ID プロバイダー) を選択します (次の画面例を参照)。



> **Note**
>
> 認証方法を選択できるのは、Transfer Familyサーバーのプロトコルの1つとしてSFTPを有効にした場合のみです。

3. デフォルト値である AWS Lambda を使用して ID プロバイダー を接続します。
4. 「AWS Lambda 関数」では、Lambda 関数の名前を選択します。

5. 残りのフィールドに値を入力してから [Create server] (サーバーの作成) を選択します。サーバーを作成するための残りの手順の詳細については、「SFTP、FTPS、または FTP サーバーエンドポイントの設定」を参照してください。

Lambda リソースベースのポリシー

Transfer Family サーバーと Lambda ARN を参照するポリシーが必要です。たとえば、ID プロバイダーに接続する Lambda 関数で次のポリシーを使用できます。ポリシーは、JSON で文字列としてエスケープされます。

```
"Policy":
"{
  "Version": "2012-10-17",
  "Id": "default",
  "Statement": [
    {
      "Sid": "AllowTransferInvocation",
      "Effect": "Allow",
      "Principal": {
        "Service": "transfer.amazonaws.com"
      },
      "Action": "lambda:InvokeFunction",
      "Resource": "arn:aws:transfer:region:account-id:function:my-lambda-auth-
function",
      "Condition": {
        "ArnLike": {
          "AWS:SourceArn": "arn:aws:transfer:region:account-id:server/server-id"
        }
      }
    }
  ]
}"
```

> **Note**
>
> 上記のポリシー例では、各「*###############*」をあなた自身の情報で置き換えます。

イベントメッセージの構造

SFTP サーバーからカスタム IDP のオーソライザー Lambda 関数に送信されるイベントメッセージ構造は次のとおりです。

```
{
    "username": "value",
    "password": "value",
    "protocol": "SFTP",
    "serverId": "s-abcd123456",
    "sourceIp": "192.168.0.100"
}
```

ここで、usernameおよびpasswordはサーバーに送信されるサインイン認証情報の値です。

例えば、以下のコマンドを入力して接続する：

```
sftp bobusa@server_hostname
```

その後、パスワードの入力を求められる：

```
Enter password:
    mysecretpassword
```

Lambda 関数内から渡されたイベントを出力することで、Lambda 関数からこれを確認できます。以下のテキストブロックのように見えるはずです。

```
{
    "username": "bobusa",
    "password": "mysecretpassword",
    "protocol": "SFTP",
    "serverId": "s-abcd123456",
    "sourceIp": "192.168.0.100"
}
```

イベントの構造は FTP と FTPS で似ています。唯一の違いは、SFTP ではなく、これらの値がprotocolパラメータに使用されることです。

認証用の Lambda 関数

さまざまな認証戦略を実装するには、Lambda 関数を編集します。アプリケーションのニーズを満たすために、 CloudFormation スタックをデプロイできます。Lambda の詳細については、AWS Lambda デベロッパーガイドまたは「Node.js による Lambda 関数の構築」を参照してください。

トピック

- Lambda 関数のテンプレート
- 有効な Lambda 値
- Lambda 関数の例
- 設定をテストする

Lambda 関数のテンプレート

認証に Lambda 関数を使用する AWS CloudFormation スタックをデプロイできます。ログイン認証情報を使用してユーザーを認証および承認するテンプレートがいくつか用意されています。これらのテンプレートまたは AWS Lambda コードを変更して、ユーザーアクセスをさらにカスタマイズできます。

> **Note**
>
> テンプレートで FIPS 対応のセキュリティポリシーを指定 AWS CloudFormation することで、 を通じて FIPS 対応 AWS Transfer Family サーバーを作成できます。使用可能なセキュリティポリシーについては、AWS Transfer Family サーバーのセキュリティポリシーで説明しています。

認証に使用する AWS CloudFormation スタックを作成するには

1. https://console.aws.amazon.com/cloudformation で AWS CloudFormation コンソールを開きます。
2. 「 AWS CloudFormation ユーザーガイド」の AWS CloudFormation 「スタックテンプレートの選択」にある既存のテンプレートからスタックをデプロイする手順に従ってください。
3. 以下のテンプレートのいずれかを使用して、Transfer Family で認証に使用するLambda 関数を作成します。

   - クラシック (Amazon Cognito) スタックテンプレート

でカスタム ID プロバイダー AWS Lambda として使用する を作成するための基本的なテンプレート AWS Transfer Family。Amazon Cognito に対してパスワードベースの認証を行い、パブリックキーベースの認証が使用されている場合、パブリックキーは Amazon S3 バケットから返されます。デプロイ後に Lambda 関数コードを変更すれば異なる処理を実行できます。

- AWS Secrets Manager スタックテンプレート

  Secrets Manager を ID プロバイダーとして統合するために、 を AWS Transfer Family サーバー AWS Lambda で使用する基本的なテンプレート。形式の AWS Secrets Manager のエントリに対して認証されます`aws/transfer/`*`server-id`*`/`*`username`*。さらに、シークレットは、Transfer Family に返されるすべてのユーザープロパティのキーバリューペアを保持する必要があります。デプロイ後に Lambda 関数コードを変更すれば異なる処理を実行できます。

- Okta スタックテンプレート: が AWS Transfer Family サーバー AWS Lambda と使用して Okta をカスタム ID プロバイダーとして統合する基本的なテンプレート。
- Okta-mfa スタックテンプレート : を AWS Transfer Family サーバー AWS Lambda と使用して Okta を MultiFactor 認証と統合する基本的なテンプレート。カスタム ID プロバイダーです。
- Azure Active Directory テンプレート: このスタックの詳細については、ブログ記事「Azure Active Directory AWS Transfer Family と を使用した への認証」で説明されています AWS Lambda。

スタックがデプロイされたら、 CloudFormation コンソールの出力タブでスタックの詳細を表示できます。

これらのスタックのいずれかをデプロイすることが、カスタム ID プロバイダーをTransfer Family ワークフローに統合するうえで最も簡単な方法です。

有効な Lambda 値

次の表は、カスタム ID プロバイダーに使用される Lambda 関数について Transfer Family が受け入れる値の詳細についての説明です。

値	説明	必須
`Role`	Amazon S3 バケットまたは Amazon EFS ファイルシステムへのユーザーのアク	必須

値	説明	必須
	セスを制御する IAM ロールの Amazon Resource Name (ARN) を指定します。このロールにアタッチされたポリシーにより、ファイルを Amazon S3 または Amazon EFS ファイルシステム間で転送する際のユーザーに付与するアクセスのレベルが決定されます。IAM ロールには、ユーザーの転送リクエストを処理する際に、サーバーによるリソースへのアクセスを許可する信頼関係も含まれる必要があります。 信頼関係の確立の詳細については、信頼関係を確立するには を参照してください。	
`PosixProfile`	ユーザーの Amazon EFS ファイルシステムへのアクセスを制御する、ユーザー ID (`Uid`)、グループ ID (`Gid`)、およびセカンダリグループ ID (`SecondaryGids`) を含む完全な POSIX ID。ファイルシステム内のファイルとディレクトリに設定される POSIX アクセス許可によって、Amazon EFS ファイルシステムとの間でファイルを転送するときにユーザーが得るアクセスのレベルが決まります。	Amazon EFS バッキングストレージに必須

値	説明	必須
PublicKeys	このユーザーに有効な SSH パブリックキー値のリスト。空のリストはこれが有効なログインではないことを示します。パスワード認証中に返してはなりません。	オプションです。
Policy	複数のユーザーに同じ IAM ロールの使用を可能にするユーザーのセッションポリシー。このポリシーは、ユーザーアクセスのスコープを Amazon S3 バケットの一部に絞り込みます。	オプションです。
HomeDirectoryType	ユーザーがサーバーにログインするときにホームディレクトリにするランディングディレクトリ (フォルダ) のタイプ。 • PATH に設定した場合、ユーザーにはファイル転送プロトコルクライアント内の Amazon S3 バケットまたは Amazon EFS の絶対パスが表示されます。 • LOGICAL に設定した場合、Amazon S3 または Amazon EFS のパスをユーザーから見えるようにするには、HomeDirectoryDetails パラメーターでマッピングを提供する必要があります。	オプションです。

値	説明	必須
HomeDirectoryDetails	ユーザーに表示する Amazon S3 または Amazon EFS のパスとキー、およびそれらをどのように表示するかを指定する論理ディレクトリマッピング。「Entry」と「Target」のペアを指定する必要があり、Entry はパスの表示方法を示し、Target は実際の Amazon S3 または Amazon EFS のパスです。	HomeDirectoryType に LOGICAL の値がある場合に必須
HomeDirectory	ユーザーがクライアントを使用してサーバーにログインするときの、ユーザーのランディングディレクトリ。	オプションです。

> **Note**
>
> HomeDirectoryDetailsはJSON マップの文字列表現です。これは、実際のJSONマップ・オブジェクトであるPosixProfileや、文字列のJSON配列であるPublicKeysとは対照的です。言語固有の詳細については、コード例を参照してください。

Lambda 関数の例

このセクションでは、NodeJS と Python の両方で使用される Lambda 関数の例をいくつか紹介します。

> **Note**
>
> これらの例では、ユーザー、ロール、POSIX プロファイル、パスワード、ホームディレクトリの詳細はすべて例であり、実際の値に置き換える必要があります。

Logical home directory, NodeJS

以下の NodeJS サンプル関数は、「論理ホームディレクトリ」を持つユーザーの詳細を提供します。

```
// GetUserConfig Lambda

exports.handler = (event, context, callback) => {
  console.log("Username:", event.username, "ServerId: ", event.serverId);

  var response;
  // Check if the username presented for authentication is correct. This doesn't
 check the value of the server ID, only that it is provided.
  if (event.serverId !== "" && event.username == 'example-user') {
    var homeDirectoryDetails = [
      {
        Entry: "/",
        Target: "/fs-faa1a123"
      }
    ];
    response = {
      Role: 'arn:aws:iam::123456789012:role/transfer-access-role', // The user is
 authenticated if and only if the Role field is not blank
      PosixProfile: {"Gid": 65534, "Uid": 65534}, // Required for EFS access, but
 not needed for S3
      HomeDirectoryDetails: JSON.stringify(homeDirectoryDetails),
      HomeDirectoryType: "LOGICAL",
    };

    // Check if password is provided
    if (!event.password) {
      // If no password provided, return the user's SSH public key
      response['PublicKeys'] = [ "ssh-
rsa abcdef0123456789abcdef0123456789abcdef0123456789abcdef0123456789" ];
    // Check if password is correct
    } else if (event.password !== 'Password1234') {
      // Return HTTP status 200 but with no role in the response to indicate
 authentication failure
      response = {};
    }
  } else {
    // Return HTTP status 200 but with no role in the response to indicate
 authentication failure
    response = {};
```

```
  }
  callback(null, response);
};
```

Path-based home directory, NodeJS

以下の NodeJS サンプル関数は、パスベースのホーム・ディレクトリを持つユーザーの詳細を提供します。

```
// GetUserConfig Lambda

exports.handler = (event, context, callback) => {
  console.log("Username:", event.username, "ServerId: ", event.serverId);

  var response;
  // Check if the username presented for authentication is correct. This doesn't
 check the value of the server ID, only that it is provided.
  // There is also event.protocol (one of "FTP", "FTPS", "SFTP") and event.sourceIp
 (e.g., "127.0.0.1") to further restrict logins.
  if (event.serverId !== "" && event.username == 'example-user') {
    response = {
      Role: 'arn:aws:iam::123456789012:role/transfer-access-role', // The user is
 authenticated if and only if the Role field is not blank
      Policy: '', // Optional, JSON stringified blob to further restrict this user's
 permissions
      HomeDirectory: '/fs-faa1a123' // Not required, defaults to '/'
    };

    // Check if password is provided
    if (!event.password) {
      // If no password provided, return the user's SSH public key
     response['PublicKeys'] = [ "ssh-
rsa abcdef0123456789abcdef0123456789abcdef0123456789abcdef0123456789" ];
    // Check if password is correct
    } else if (event.password !== 'Password1234') {
      // Return HTTP status 200 but with no role in the response to indicate
 authentication failure
      response = {};
    }
  } else {
    // Return HTTP status 200 but with no role in the response to indicate
 authentication failure
    response = {};
```

```
  }
  callback(null, response);
};
```

Logical home directory, Python

以下の Python サンプル関数は、「論理ホームディレクトリ」を持つユーザーの詳細を提供します。

```
# GetUserConfig Python Lambda with LOGICAL HomeDirectoryDetails
import json

def lambda_handler(event, context):
  print("Username: {}, ServerId: {}".format(event['username'], event['serverId']))

  response = {}

  # Check if the username presented for authentication is correct. This doesn't
 check the value of the server ID, only that it is provided.
  if event['serverId'] != '' and event['username'] == 'example-user':
    homeDirectoryDetails = [
      {
        'Entry': '/',
        'Target': '/fs-faa1a123'
      }
    ]
    response = {
      'Role': 'arn:aws:iam::123456789012:role/transfer-access-role', # The user will
 be authenticated if and only if the Role field is not blank
      'PosixProfile': {"Gid": 65534, "Uid": 65534}, # Required for EFS access, but
 not needed for S3
      'HomeDirectoryDetails': json.dumps(homeDirectoryDetails),
      'HomeDirectoryType': "LOGICAL"
    }

    # Check if password is provided
    if event.get('password', '') == '':
      # If no password provided, return the user's SSH public key
     response['PublicKeys'] = [ "ssh-
rsa abcdef0123456789abcdef0123456789abcdef0123456789abcdef0123456789" ]
    # Check if password is correct
    elif event['password'] != 'Password1234':
```

```
      # Return HTTP status 200 but with no role in the response to indicate
 authentication failure
      response = {}
  else:
    # Return HTTP status 200 but with no role in the response to indicate
 authentication failure
    response = {}

  return response
```

Path-based home directory, Python

以下の Python サンプル関数は、パスベースのホームディレクトリを持つユーザーの詳細を提供します。

```
# GetUserConfig Python Lambda with PATH HomeDirectory

def lambda_handler(event, context):
  print("Username: {}, ServerId: {}".format(event['username'], event['serverId']))

  response = {}

  # Check if the username presented for authentication is correct. This doesn't
 check the value of the server ID, only that it is provided.
  # There is also event.protocol (one of "FTP", "FTPS", "SFTP") and event.sourceIp
 (e.g., "127.0.0.1") to further restrict logins.
  if event['serverId'] != '' and event['username'] == 'example-user':
    response = {
      'Role': 'arn:aws:iam::123456789012:role/transfer-access-role', # The user will
 be authenticated if and only if the Role field is not blank
      'Policy': '', #  Optional, JSON stringified blob to further restrict this
 user's permissions
      'HomeDirectory': '/fs-fs-faa1a123',
      'HomeDirectoryType': "PATH" # Not strictly required, defaults to PATH
    }

    # Check if password is provided
    if event.get('password', '') == '':
      # If no password provided, return the user's SSH public key
     response['PublicKeys'] = [ "ssh-
rsa abcdef0123456789abcdef0123456789abcdef0123456789abcdef0123456789" ]
    # Check if password is correct
    elif event['password'] != 'Password1234':
```

```
     # Return HTTP status 200 but with no role in the response to indicate
authentication failure
     response = {}
 else:
   # Return HTTP status 200 but with no role in the response to indicate
authentication failure
   response = {}

 return response
```

設定をテストする

カスタム ID プロバイダーを作成したら、設定をテストする必要があります。

Console

AWS Transfer Family コンソールを使用して設定をテストするには

1. AWS Transfer Family コンソールを開きます。
2. [Servers] (サーバー) ページで新しいサーバーを選択し、[Actions] (アクション) を選択してから [Test] (テスト) を選択します。
3. AWS CloudFormation スタックをデプロイしたときに設定したユーザー名とパスワードのテキストを入力します。デフォルトのオプションのままだと、ユーザー名は `myuser`、パスワードは `MySuperSecretPassword` となります。
4. AWS CloudFormation スタックのデプロイ時に設定する場合は、サーバープロトコルを選択し、送信元 IP の IP アドレスを入力します。

CLI

AWS CLI を使用して設定をテストするには

1. test-identity-provider コマンドを実行します。以降のステップで説明するように、それぞれの *user input placeholder* を独自の情報に置き換えます。

```
aws transfer test-identity-provider --server-id s-1234abcd5678efgh --user-
name myuser --user-password MySuperSecretPassword --server-protocol FTP --
source-ip 127.0.0.1
```

2. サーバー ID を入力します。

3. AWS CloudFormation スタックをデプロイしたときに設定したユーザー名とパスワードを入力します。デフォルトのオプションのままだと、ユーザー名は `myuser`、パスワードは `MySuperSecretPassword` となります。
4. AWS CloudFormation スタックをデプロイしたときにサーバープロトコルと送信元 IP アドレスを設定した場合は、サーバープロトコルと送信元 IP アドレスを入力します。

ユーザー認証が成功した場合、テストは`StatusCode: 200` HTTP レスポンス、空の文字列`Message: ""`（これがなければ失敗の理由を含む）、および`Response`フィールドを返します。

> **Note**
>
> 以下のレスポンス例では、`Response`フィールドは「文字列化」された (プログラム内で使用できるフラットな JSON 文字列に変換された) JSON オブジェクトで、ユーザーのロールと権限の詳細が含まれています。

```
{
    "Response":"{\"Policy\":\"{\\\"Version\\\":\\\"2012-10-17\\\",\\\"Statement\\\":
[{\\\"Sid\\\":\\\"ReadAndListAllBuckets\\\",\\\"Effect\\\":\\\"Allow\\\",\\\"Action\\
\":[\\\"s3:ListAllMybuckets\\\",\\\"s3:GetBucketLocation\\\",\\\"s3:ListBucket\\\",\\
\"s3:GetObjectVersion\\\",\\\"s3:GetObjectVersion\\\"],\\\"Resource\\\":\\\"*\\\"}]}\",
\"Role\":\"arn:aws:iam::000000000000:role/MyUserS3AccessRole\",\"HomeDirectory\":\"/
\"}",
    "StatusCode": 200,
    "Message": ""
}
```

Amazon API Gateway を使用して ID プロバイダーを統合する

このトピックでは、 AWS Lambda 関数を使用して API Gateway メソッドをバックアップする方法について説明します。このオプションは、ID プロバイダーを統合するために RESTful API が必要な場合、または AWS WAF を使用してその機能をジオブロッキングまたはレート制限リクエストに活用する場合に使用します。

「API ゲートウェイを使用して ID プロバイダを統合する場合の制限事項」

- この構成はカスタムドメインをサポートしていません。
- この構成は、プライベート API Gateway URL をサポートしていません。

これらのいずれかが必要な場合は、API Gateway なしで Lambda を ID プロバイダーとして使用できます。詳細については、「AWS Lambda を使用して ID プロバイダーを統合する」を参照してください。

API Gateway メソッドを使用した認証

Transfer Family の ID プロバイダーとして使用する API Gateway メソッドを作成できます。このアプローチは、API を作成して提供するための非常に安全な方法です。API Gateway では、HTTPS エンドポイントを作成して、すべての着信 API コールをより高いセキュリティで送信できます。API Gateway サービスの詳細については、API Gateway デベロッパーガイドを参照してください。

API Gateway には、 という名前の認証方法が用意されています。これによりAWS_IAM、 が内部的に AWS 使用する AWS Identity and Access Management (IAM) に基づく認証と同じ認証が提供されます。AWS_IAM で認証を有効にした場合、API を呼び出す明示的なアクセス権限を持つ呼び出し側のみがその API Gateway メソッドに到達できます。

API Gateway メソッドをTransfer Family カスタム ID プロバイダーとして使用するには、API Gateway メソッドの IAM を有効にします。このプロセスの一環として、Transfer Family についてゲートウェイを使用するためのアクセス許可を持つ IAM ロールを提供します。

> **Note**
>
> セキュリティを強化するために、ウェブアプリケーションのファイアウォールを設定できます。 AWS WAF はウェブアプリケーションファイアウォールで、これにより Amazon API Gateway に転送される HTTP および HTTPS リクエストのモニタリングが可能です。詳細については、「ウェブアプリケーションファイアウォールを追加する」を参照してください

Transfer Family でカスタム認証に API Gateway メソッドを使用するには

1. AWS CloudFormation スタックを作成します。これを実行するには:

   > **Note**
   >
   > スタックテンプレートが更新され、BASE64-encodedパスワードが使用されました。詳細については、「」を参照してくださいAWS CloudFormation テンプレートの改善。

   a. https://console.aws.amazon.com/cloudformation で AWS CloudFormation コンソールを開きます。

b. 「 AWS CloudFormation ユーザーガイド」の AWS CloudFormation 「スタックテンプレートの選択」にある既存のテンプレートからスタックをデプロイする手順に従ってください。

c. Transfer Family でカスタム ID プロバイダーとして使用する AWS Lambda-backed API Gateway メソッドを作成するには、次のいずれかの基本テンプレートを使用します。

- 基本スタックテンプレート

  デフォルトでは、API Gateway メソッドはカスタム ID プロバイダーとして使用され、ハードコードされた SSH (Secure Shell) キーまたはパスワードを使用して 1 つのサーバーで 1 人のユーザーを認証します。デプロイ後に Lambda 関数コードを変更すれば異なる処理を実行できます。

- AWS Secrets Manager スタックテンプレート

  デフォルトでは、API Gateway メソッドは、Secrets Manager 内の `aws/transfer/server-id/username` 形式のエントリに対して認証します。さらに、シークレットは、Transfer Family に返されるすべてのユーザープロパティのキーバリューペアを保持する必要があります。デプロイ後に Lambda 関数コードを変更すれば異なる処理を実行できます。詳細については、ブログ記事「 AWS Transfer Family を使用したパスワード認証の有効化 AWS Secrets Manager」を参照してください。

- Okta スタックテンプレート

  API Gateway メソッドは、Transfer Family のカスタム ID プロバイダーとして Okta と統合されます。詳細については、ブログ記事「AWS Transfer Familyで Okta を ID プロバイダーとして使う」を参照してください。

これらのスタックのいずれかをデプロイすることが、カスタム ID プロバイダーをTransfer Family ワークフローに統合するうえで最も簡単な方法です。各スタックは Lambda 関数を使用して、API Gateway に基づく API メソッドをサポートします。その後、Transfer Family でカスタム ID プロバイダーとして API メソッドを使用できます。デフォルトでは、Lambda 関数は、`myuser` という単一のユーザーを `MySuperSecretPassword` のパスワードで認証します。デプロイ後に、これらの認証情報を編集するか Lambda 関数コードを更新すれば異なる処理を実行できます。

> **Important**
>
> デフォルトのユーザーとパスワード認証情報を編集することをお勧めします。

スタックがデプロイされたら、 CloudFormation コンソールの出力タブでスタックの詳細を表示できます。具体的には、スタックの Amazon リソースネーム (ARN)、スタックが作成した IAM ロールの ARN、および新しいゲートウェイの URL が含まれます。

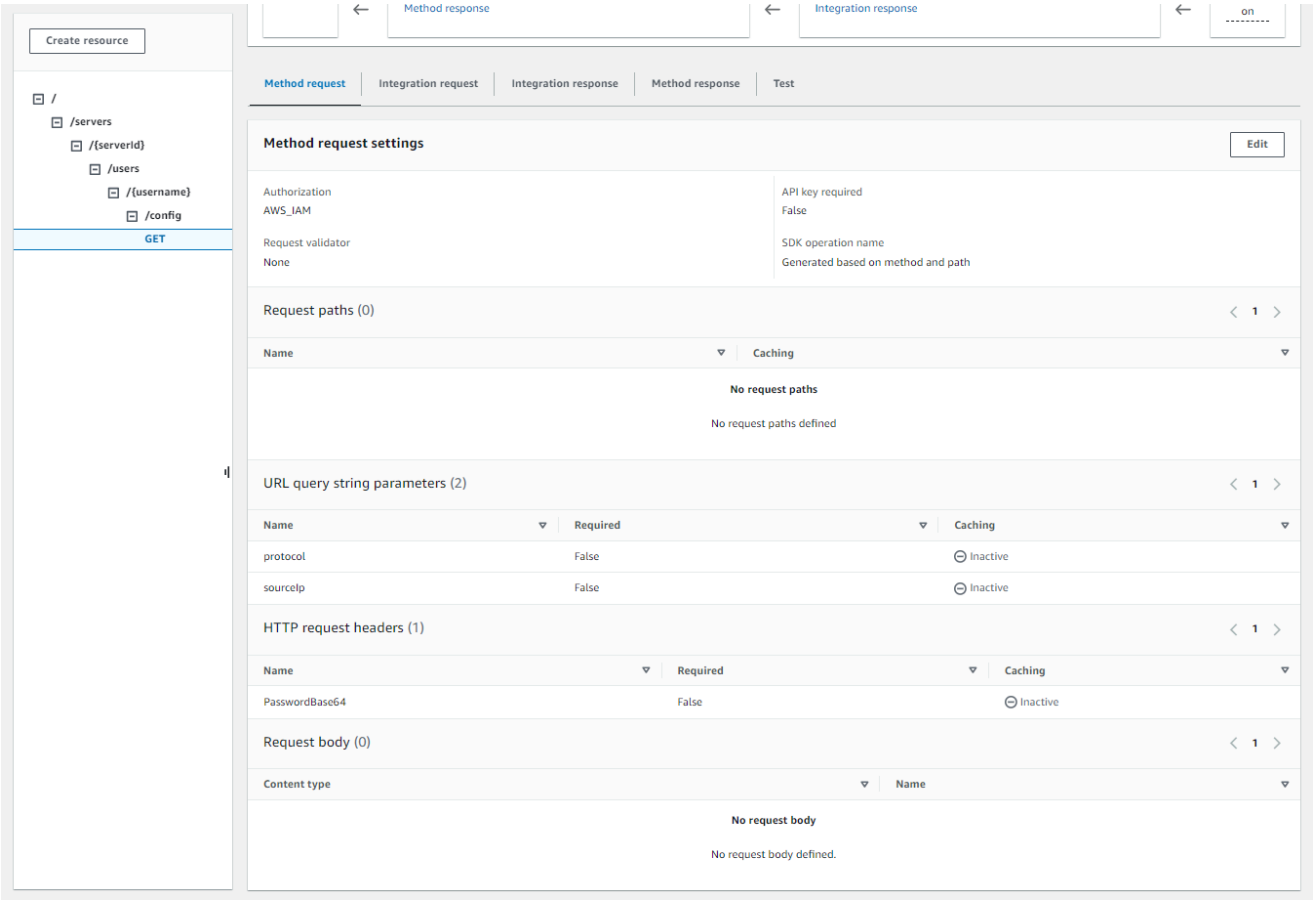
> **Note**
>
> カスタム ID プロバイダーオプションを使用してユーザーのパスワードベースの認証を有効にし、API Gateway が提供するリクエストとレスポンスのログ記録を有効にすると、API Gateway はユーザーのパスワードを Amazon CloudWatch Logs に記録します。本稼働環境でこのログを使用することは推奨されません。詳細については、CloudWatch 「API Gateway デベロッパーガイド」の「API Gateway での API ログ記録の設定」を参照してください。

2. サーバーの API Gateway メソッドの設定を確認します。これを実行するには:

   a. API Gateway コンソール (https://console.aws.amazon.com/apigateway) を開きます。

   b. テンプレートが生成した Transfer Custom Identity Provider の基本テンプレート API を選択します。 AWS CloudFormation ゲートウェイを表示するには、リージョンを選択する必要がある場合があります。

   c. リソースペインで、GET を選択します。次の画面例は、正しいメソッド設定を示しています。

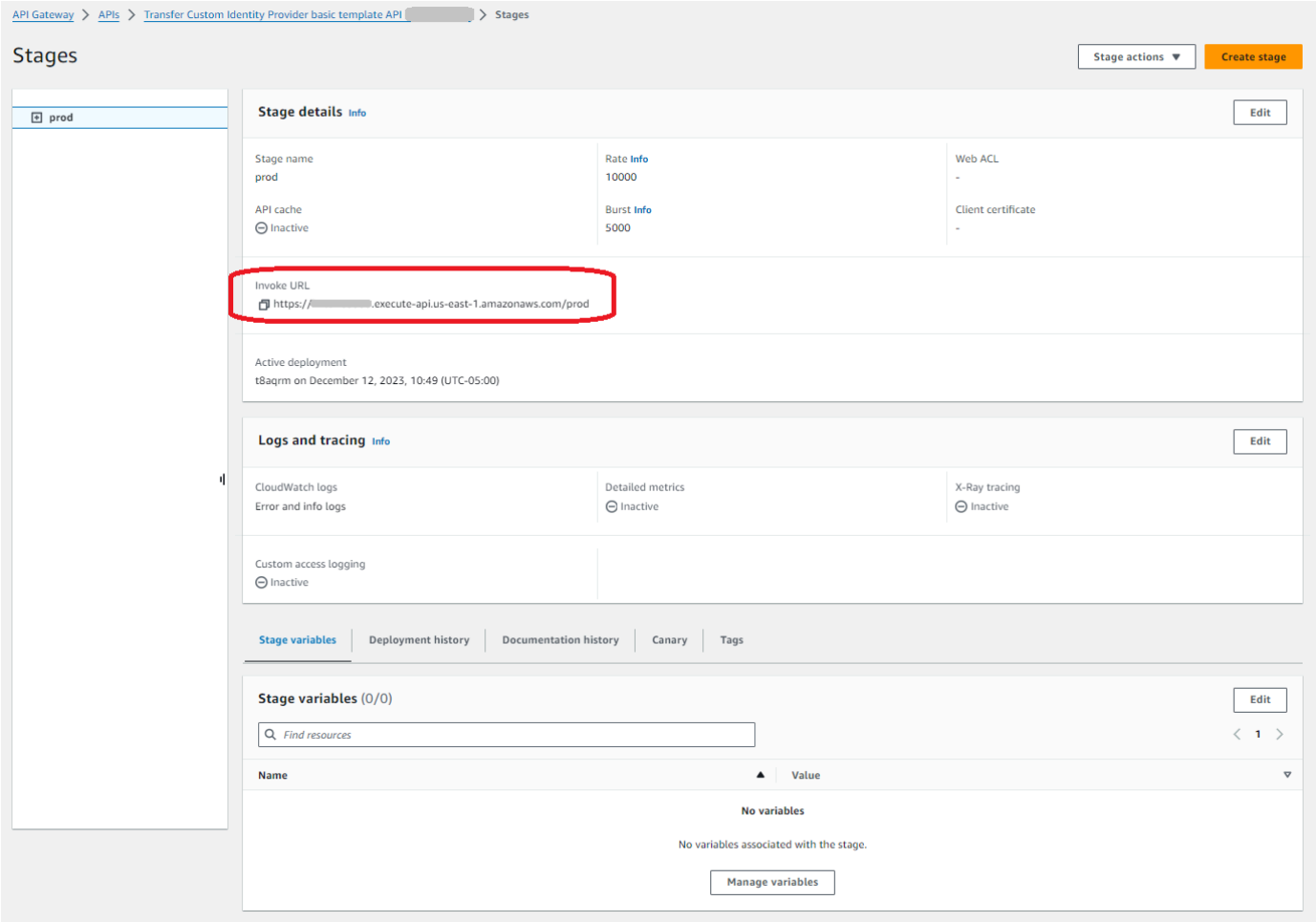


この時点で、API Gateway をデプロイする準備が整いました。

3. [Actions] (アクション) で [Deploy API] (API のデプロイ) を選択します。[Deplyment stage] (デプロイステージ) で [prod] を選択してから [Deploy] (デプロイ) を選択します。

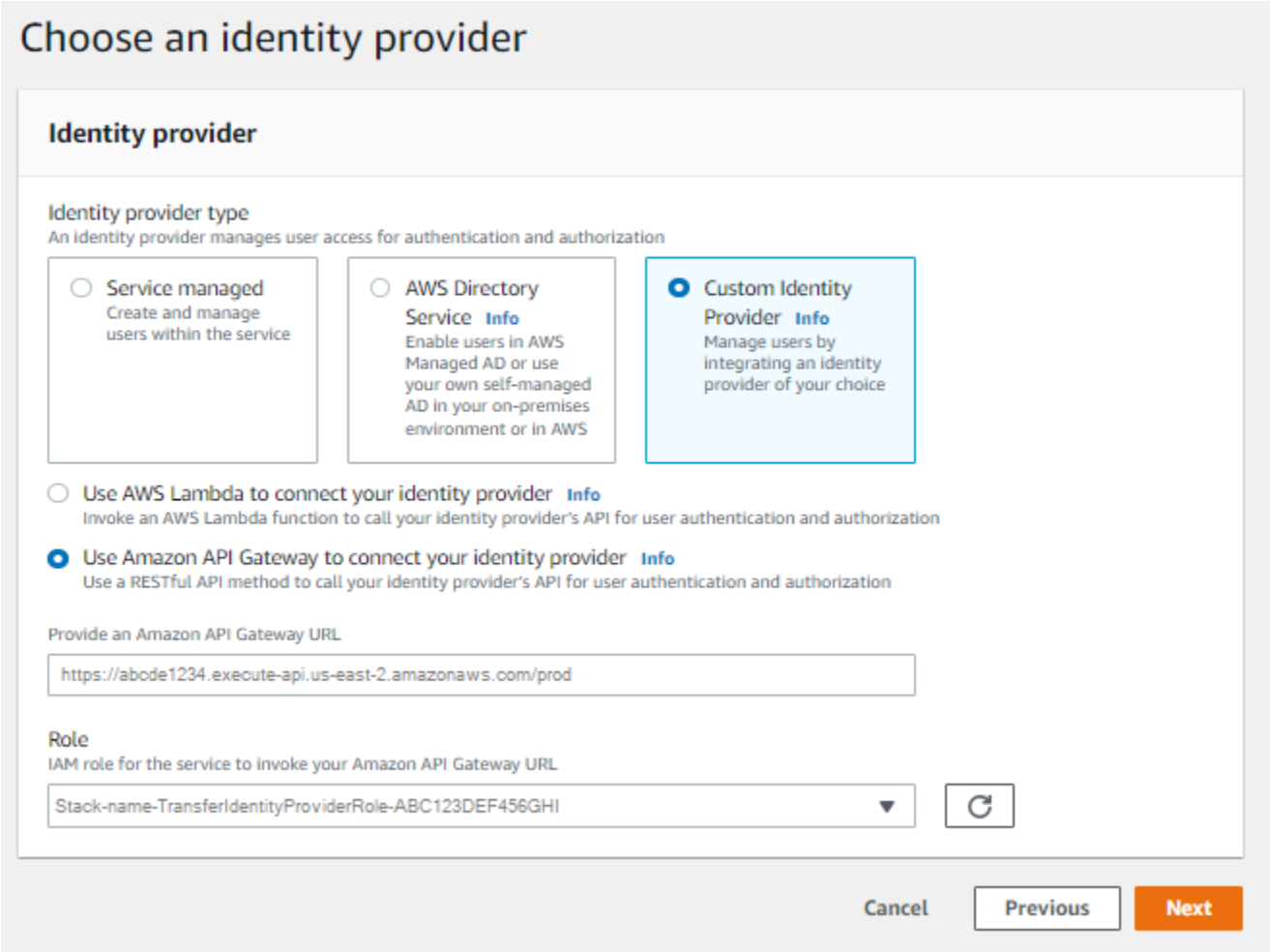
   API Gateway メソッドが正常にデプロイされたら、次のスクリーンショットに示すように、ステージ > ステージの詳細 でそのパフォーマンスを表示します。

> **Note**
>
> 画面の最上部に表示される [Invoke URL] (呼び出し URL) アドレスをコピーします。次のステップで必要になる場合があります。



4. https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開きます。
5. Transfer Family は、スタックの作成時に自動的に作成されたはずです。そうでない場合は、以下の手順を使用してサーバーを設定します。

   a. [Create server] (サーバーの作成) を選択すると [Create server] (サーバーの作成) ページが開きます。[Choose an identity provider] (ID プロバイダーの選択) で [Custom] (カスタム) を選択してから、[Use Amazon API Gateway to connect to your identity provider] (Amazon API Gateway を使用してアイデンティティプロバイダーに接続する) を選択します (次の画面例を参照)。



b. [Provide an Amazon API Gateway URL] (Amazon API Gateway URL を指定) テキストボックスに、ステップ 3 で作成した API Gateway エンドポイントの [Invoke URL] (呼び出し URL) アドレスを貼り付けます。

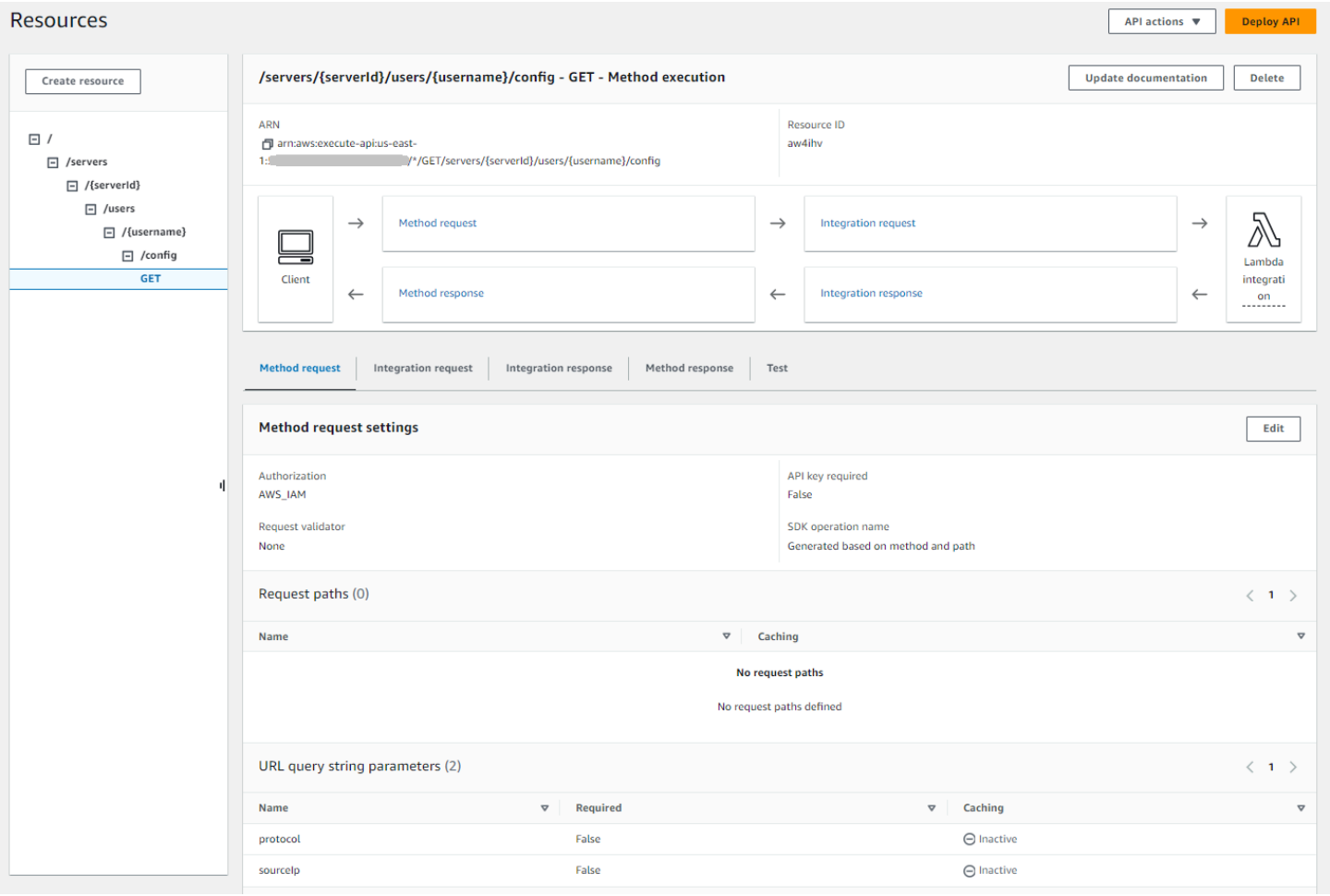
c. ロール で、 AWS CloudFormation テンプレートによって作成された IAM ロールを選択します。このロールにより、Transfer Family が API Gateway メソッドを呼び出せるようになります。

   呼び出しロールには、ステップ 1 で作成した AWS CloudFormation スタック用に選択したスタック名が含まれます。次のような形式です: *CloudFormation-stack-name*-TransferIdentityProviderRole-*ABC123DEF456GHI*

d. 残りのフィールドに値を入力してから [Create server] (サーバーの作成) を選択します。サーバーを作成するための残りの手順の詳細については、「SFTP、FTPS、または FTP サーバーエンドポイントの設定」を参照してください。

API Gateway メソッドを実装する

Transfer Family のカスタム ID プロバイダーを作成するには、API Gateway メソッドで/servers/*serverId*/users/*username*/config のリソースパスを持つ単一のメソッドを実装する必要があります。*serverId* と *username* の値は RESTful リソースパスから取得されます。また、下図のように、sourceIpとprotocolを「URL クエリ文字列パラメータ」として「メソッドリクエスト」に追加します。



> **Note**
>
> ユーザー名は、最小 3 文字、最大 100 文字にする必要があります。ユーザー名に使用できる文字は、a-z、A-Z、0-9、アンダースコア (_) 、ハイフン (-) 、ピリオド (.) 、アットマーク (@) です。ただし、ユーザー名はハイフン (-) 、ピリオド (.) 、アットマーク (@) で始めることはできません。

Transfer Family がユーザーに代わってパスワード認証を試みると、サービスが `Password:` ヘッダーフィールドを提供します。`Password:` ヘッダーが存在しない場合、Transfer Family はパブリックキー認証でユーザーを認証しようと試みます。

エンドユーザーの認証と承認に ID プロバイダーを使用する場合、エンドユーザーが使用するクライアントの IP アドレスに基づいてアクセスリクエストを許可または拒否できます。この機能を使用すると、S3 バケットまたは Amazon EFS ファイルシステムに保存されたデータに、信頼済みとして指定した IP アドレスからのみ、サポートされているプロトコル経由でアクセスできるようになります。この機能を有効にするには、`sourceIp`をクエリ文字列に含める必要があります。

サーバーで複数のプロトコルを有効にしており、複数のプロトコルで同じユーザー名を使用してアクセスを提供したい場合、各プロトコルに固有の認証情報が ID プロバイダーに設定されている限り、そうすることができます。この機能を有効にするには、RESTful リソースパスに *`protocol`* 値を含める必要があります。

API Gateway メソッドは常に HTTP ステータスコード `200` を返す必要があります。その他の HTTP ステータスコードは、API へのアクセスエラーがあったことを示します。

Amazon S3 のレスポンスの例

この例のレスポンス本文は、Amazon S3 では次の形式の JSON ドキュメントになります。

```
{
 "Role": "IAM role with configured S3 permissions",
 "PublicKeys": [
     "ssh-rsa public-key1",
     "ssh-rsa public-key2"
  ],
 "Policy": "STS Assume role session policy",
 "HomeDirectory": "/bucketName/path/to/home/directory"
}
```

> **Note**
>
> ポリシーは、JSON で文字列としてエスケープされます。例:
>
> ```
> "Policy":
> "{
>   \"Version\": \"2012-10-17\",
>   \"Statement\":
>      [
> ```

```
      {\"Condition\":
         {\"StringLike\":
             {\"s3:prefix\":
                [\"user/*\", \"user/\"]}},
      \"Resource\": \"arn:aws:s3:::bucket\",
      \"Action\": \"s3:ListBucket\",
      \"Effect\": \"Allow\",
      \"Sid\": \"ListHomeDir\"},
      {\"Resource\": \"arn:aws:s3:::*\",
         \"Action\": [\"s3:PutObject\",
         \"s3:GetObject\",
         \"s3:DeleteObjectVersion\",
         \"s3:DeleteObject\",
         \"s3:GetObjectVersion\",
         \"s3:GetObjectACL\",
         \"s3:PutObjectACL\"],
      \"Effect\": \"Allow\",
      \"Sid\": \"HomeDirObjectAccess\"}]
}"
```

次のレスポンスの例は、論理ホームディレクトリタイプを有するユーザーを示しています。

```
{
   "Role": "arn:aws:iam::123456789012:role/transfer-access-role-s3",
   "HomeDirectoryType":"LOGICAL",
   "HomeDirectoryDetails":"[{\"Entry\":\"/\",\"Target\":\"/MY-HOME-BUCKET\"}]",
   "PublicKeys":[""]
}
```

Amazon EFS のレスポンスの例

この例のレスポンス本文は、Amazon EFS では次の形式の JSON ドキュメントになります。

```
{
 "Role": "IAM role with configured EFS permissions",
 "PublicKeys": [
     "ssh-rsa public-key1",
     "ssh-rsa public-key2"
  ],
 "PosixProfile": {
   "Uid": "POSIX user ID",
```

```
    "Gid": "POSIX group ID",
    "SecondaryGids": [Optional list of secondary Group IDs],
  },
  "HomeDirectory": "/fs-id/path/to/home/directory"
}
```

Role フィールドは認証に成功したことを示します。パスワード認証を実行する場合 (Password: ヘッダーの提供時) には、SSH パブリックキーを提供する必要はありません。ユーザーが認証できない場合 (たとえば、パスワードが正しくない場合)、メソッドは Role を設定せずにレスポンスを返す必要があります。このようなレスポンスの例として、空の JSON オブジェクトがあります。

次のレスポンスの例は、論理ホームディレクトリタイプを持つユーザーを示しています。

```
{
    "Role": "arn:aws:iam::123456789012:role/transfer-access-role-efs",
    "HomeDirectoryType": "LOGICAL",
    "HomeDirectoryDetails":"[{\"Entry\":\"/\",\"Target\":\"/faa1a123\"}]",
    "PublicKeys":[""],
    "PosixProfile":{"Uid":65534,"Gid":65534}
}
```

Lambda 関数にユーザーポリシーを JSON 形式で含めることができます。Transfer Family でのユーザーポリシーの設定の詳細については、「アクセスコントロールの管理」を参照してください。

デフォルトの Lambda 関数

異なる認証戦略を実装するには、ゲートウェイで使用する Lambda 関数を編集します。アプリケーションのニーズを満たすために、Node.js 内で次の Lambda 関数の例を使用できます。Lambda の詳細については、AWS Lambda デベロッパーガイドまたは「Node.js による Lambda 関数の構築」を参照してください。

以下の Lambda 関数の例では、ユーザー名、パスワード (パスワード認証を行っている場合)、サーバー ID、プロトコル、クライアント IP アドレスを受け取っています。これらの入力を組み合わせて使用すると、ID プロバイダーを検索し、ログインを許可するかどうかを判断できます。

> **Note**
>
> サーバーで複数のプロトコルを有効にしており、複数のプロトコルで同じユーザー名を使用してアクセスを提供したい場合、プロトコルに固有の認証情報が ID プロバイダーに設定されている限り、そうすることができます。

File Transfer Protocol (FTP) については、Secure Shell (SSH) File Transfer Protocol (SFTP) and File Transfer Protocol over SSL (FTPS) とは異なる認証情報を維持することをお勧めします。SFTP や FTPS と異なり、FTP は認証情報を平文で送信します。FTP の認証情報を SFTP または FTPS から隔離することを推奨します。そうすれば、FTP の認証情報が共有または公開されても、SFTP または FTPS を使用しているワークロードは引き続き安全だからです。

この関数の例は、ロールと論理ホームディレクトリの詳細、ならびに (公開鍵認証を実行する場合には) パブリックキーを返します。

サービス管理対象ユーザーを作成するときは、そのユーザーのホームディレクトリを論理ディレクトリまたは物理ディレクトリに設定します。同様に、目的のユーザーに物理的または論理的なディレクトリ構造を伝えるには、Lambda 関数の結果が必要です。設定するパラメータは「HomeDirectoryType」フィールドの値によって異なります。

- `HomeDirectoryType`が`PATH`に設定される場合 — `HomeDirectory`フィールドはユーザーに表示される Amazon S3 バケットの絶対プレフィックスまたは Amazon EFS 絶対パスである必要があります。
- `HomeDirectoryType`が`LOGICAL`に設定される場合 — `HomeDirectory`フィールドを設定「しないで」ください。代わりに、サービス管理対象ユーザー向けの「HomeDirectoryDetails」パラメーターで説明されている値と同様に、必要なエントリ/ターゲットのマッピングを提供する`HomeDirectoryDetails`フィールドを設定します。

関数の例はLambda 関数の例に記載されています。

で使用する Lambda 関数 AWS Secrets Manager

を ID プロバイダー AWS Secrets Manager として使用するには、サンプル AWS CloudFormation テンプレートで Lambda 関数を使用します。Lambda 関数は、認証情報を使用して Secrets Manager サービスにクエリを実行し、成功すると指定されたシークレットを返します。Secrets Manager の詳細については、AWS Secrets Manager ユーザーガイドを参照してください。

この Lambda 関数を使用するサンプル AWS CloudFormation テンプレートをダウンロードするには、 が提供する Amazon S3 バケット AWS Transfer Familyに移動します。

AWS CloudFormation テンプレートの改善

API Gateway インターフェイスの改良が、公開 CloudFormationされたテンプレートに対して行われました。テンプレートでは、API Gateway で BASE64-encodedされたパスワードを使用するようになりました。既存のデプロイは、この機能強化なしで引き続き機能しますが、基本的な US-ASCII 文字セット以外の文字を含むパスワードは許可されません。

この機能を有効にするテンプレートの変更は次のとおりです。

- GetUserConfigRequest AWS::ApiGateway::Method リソースにはこのRequestTemplatesコードが必要です (斜体の行は更新された行です)

```
RequestTemplates:
   application/json: |
   {
      "username": "$util.urlDecode($input.params('username'))",
      "password":
 "$util.escapeJavaScript($util.base64Decode($input.params('PasswordBase64'))).replaceAll("\
\'","''")",
      "protocol": "$input.params('protocol')",
      "serverId": "$input.params('serverId')",
      "sourceIp": "$input.params('sourceIp')"
}
```

- PasswordBase64 ヘッダーを使用するには、GetUserConfigリソースRequestParameterの を変更する必要があります (斜体の行は更新された行です)。

```
RequestParameters:
   method.request.header.PasswordBase64: false
   method.request.querystring.protocol: false
   method.request.querystring.sourceIp: false
```

スタックのテンプレートが最新かどうかを確認するには

1. https://console.aws.amazon.com/cloudformation で AWS CloudFormation コンソールを開きます。
2. スタックのリストからスタックを選択します。
3. 詳細パネルから、テンプレートタブを選択します。
4. 以下を探します。

- を検索しRequestTemplates、次の行があることを確認します。

```
"password":
 "$util.escapeJavaScript($util.base64Decode($input.params('PasswordBase64'))).replaceAll(
\'","'")",
```

- を検索しRequestParameters、次の行があることを確認します。

```
method.request.header.PasswordBase64: false
```

更新された行が表示されない場合は、スタックを編集します。 AWS CloudFormation スタックの更新方法の詳細については、「 ユーザーガイド」の「スタックテンプレートの変更AWS CloudFormation」を参照してください。

論理ディレクトリを使用して Transfer Family ディレクトリ構造を簡素化する

AWS Transfer Family サーバーディレクトリ構造を簡素化するために、論理ディレクトリを使用できます。論理ディレクトリを使用すると、ユーザーが Amazon S3 バケットまたは Amazon EFS ファイルシステムに接続するときにナビゲートしやすいユーザーフレンドリーな名前で仮想ディレクトリ構造を構築できます。論理ディレクトリを使用する場合、絶対ディレクトリパス、Amazon S3 バケット名、EFS ファイルシステム名をエンドユーザーに開示しないようにできます。

> **Note**
>
> エンドユーザーが実行を許可したオペレーションのみを実行できるように、セッションポリシーを使用する必要があります。
> 論理ディレクトリを使用して、エンドユーザー向けの使いやすい仮想ディレクトリを作成し、バケット名を抽象化する必要があります。論理ディレクトリマッピングでは、ユーザーは指定された論理パスとサブディレクトリにのみアクセスでき、論理ルートを通過する相対パスは禁止されます。
> Transfer Family は、相対要素を含む可能性のあるすべてのパスを検証し、これらのパスを Amazon S3 に渡す前にこれらのパスが解決されないようにアクティブにブロックします。これにより、ユーザーは論理マッピングを超えて移動できなくなります。

Transfer Family は、エンドユーザーが論理ディレクトリの外部にあるディレクトリにアクセスできないようにしていますが、ストレージレベルで最小特権を適用するために、一意のロールまたはセッションポリシーを使用することをお勧めします。

論理ディレクトリを使用すると、chroot オペレーションと呼ばれるものを実行することで、ストレージ階層内の目的の場所にユーザーのルートディレクトリを設定できます。このモードでは、ユーザーが構成したホームディレクトリまたはルートディレクトリの外部にあるディレクトリには移動できません。

たとえば、Amazon S3 ユーザーはアクセス専用 */mybucket/home/*${transfer:*UserName*} にスコープダウンされていますが、一部のクライアントでは、ユーザーがフォルダを */mybucket/home* にトラバースアップできます。この場合、ユーザーは、Transfer Family サーバーからログアウトして再びログインした後でのみ、本来のホームディレクトリに戻ります。chroot オペレーションを実行すると、この状況が発生するのを防止できます。

バケットとプレフィックスで独自のディレクトリ構造を作成できます。この機能は、バケットプレフィクスを介してレプリケートできない特定のディレクトリ構造を想定したワークフローがある場合に便利です。また、Amazon S3 内の複数の非連続な場所にリンクすることもでき、これはディレクトリパスがファイルシステム内の別の場所を参照する Linux ファイルシステム内にシンボリックリンクを作成するのと似ています。

論理ディレクトリ FILE マッピング

HomeDirectoryMapEntry データ型に Typeパラメータが含まれるようになりました。このパラメータが存在する前に、ターゲットが ファイルである論理ディレクトリマッピングを作成していた可能性があります。以前にこれらの種類の論理ディレクトリマッピングを作成したことがある場合は、 Typeを明示的に に設定する必要があります。設定しないとFILE、これらのマッピングは今後正しく機能しません。
これを行う 1 つの方法は、 UpdateUser API を呼び出し、既存のマッピングFILEの Typeを に設定することです。

論理ディレクトリを使用する際のルール

論理ディレクトリマッピングを構築する前に、以下のルールを理解する必要があります。

- Entry が "/" の場合、重複パスは許可されないので、設定できるマッピングは 1 つのみです。

- 論理ディレクトリは、最大 2.1 MB のマッピングをサポートします (サービスマネージドユーザーの場合、この制限は 2,000 エントリです) 。つまり、マッピングを含むデータ構造の最大サイズは 2.1 MB です。マッピングが多数ある場合は、次のようにマッピングのサイズを計算できます。

  1. 形式 で一般的なマッピングを記述します。ここで`{"Entry":"/`*`entry-path`*`","Target":"/`*`target-path`*`"}`、 *`entry-path`*と *`target-path`*は使用する実際の値です。
  2. その文字列の文字をカウントし、(1) を追加します。
  3. その数にサーバー用のマッピングのおおよその数を掛けます。

  ステップ 3 で推定した数値が 2.1 MB 未満の場合、マッピングは許容範囲内です。

- バケットまたはファイルシステムのパスがユーザー名に基づいてパラメータ化されている場合、ターゲットは `${transfer:UserName}` 変数を使用できます。
- ターゲットは、異なるバケットまたはファイルシステムのパスにすることができますが、マッピングされた AWS Identity and Access Management (IAM) ロール (レスポンスの `Role`パラメータ) がそれらのバケットまたはファイルシステムへのアクセスを提供することを確認する必要があります。
- `HomeDirectory` パラメータを指定しないでください。この値は、 `HomeDirectoryType` パラメータの値を使用しているときに`EntryTarget`ペアによって暗示されるため`LOGICAL`です。
- ターゲットはスラッシュ (/) 文字で始める必要がありますが、 を指定するときは末尾のスラッシュ (/) を使用しないでください`Target`。例えば、 `/DOC-EXAMPLE-BUCKET/images`は許容されますが、 `DOC-EXAMPLE-BUCKET/images`と `/DOC-EXAMPLE-BUCKET/images/`は許容されません。
- Amazon S3 はオブジェクトストアです。つまり、フォルダは仮想概念であり、実際のディレクトリ階層はありません。アプリケーションがクライアントから`stat`オペレーションを発行する場合、ストレージに Amazon S3 を使用していると、すべてがファイルとして分類されます。この動作については、「Amazon Simple Storage Service ユーザーガイド」の「フォルダを使用した Amazon S3 コンソールでのオブジェクトの整理」で説明されています。 アプリケーションで、何かがファイルかフォルダか`stat`を正確に表示する必要がある場合は、Transfer Family サーバーのストレージオプションとして Amazon Elastic File System (Amazon EFS) を使用できます。
- ユーザーの論理ディレクトリ値を指定する場合、使用する パラメータはユーザーのタイプによって異なります。
  - サービス管理型ユーザーの場合、論理ディレクトリの値を `HomeDirectoryMappings` に指定してください。

- カスタム ID プロバイダーユーザーの場合は、 で論理ディレクトリ値を指定しますHomeDirectoryDetails。

> **Important**
>
> Amazon S3 ディレクトリのパフォーマンスを最適化することを選択しない限り (サーバーを作成または更新するとき)、起動時にルートディレクトリが存在している必要があります。Amazon S3 の場合、ルートフォルダを作成するには、スラッシュ (/) で終わる 0 バイトのオブジェクトが既に作成されている必要があります。この問題を回避することは、Amazon S3 のパフォーマンスの最適化を検討する理由です。

論理ディレクトリと の実装 chroot

論理ディレクトリと chroot 機能を使用するには、以下のことをする必要があります。

ユーザーごとの論理ディレクトリをオンにします。そのためには、ユーザーを作成または更新するときに HomeDirectoryType パラメータを LOGICAL に設定します。

```
"HomeDirectoryType": "LOGICAL"
```

chroot

chroot の場合、ユーザーごとに単一の Entry と Target のペア構成されるディレクトリ構造を作成します。ルートフォルダは Entry ポイントであり、Target はマップ先になるバケットまたはファイルシステム内の場所です。

Example for Amazon S3

```
[{"Entry": "/", "Target": "/mybucket/jane"}]
```

Example for Amazon EFS

```
[{"Entry": "/", "Target": "/fs-faa1a123/jane"}]
```

前の例のように絶対パスを使うこともできるし、次の例のようにユーザー名を ${transfer:UserName} でダイナミックに置換できます。

```
[{"Entry": "/", "Target":
"/mybucket/${transfer:UserName}"}]
```

前の例では、ユーザーはルートディレクトリにロックされ、階層の上位には移動できません。

仮想ディレクトリ構造

仮想ディレクトリ構造の場合、S3 バケットまたは EFS ファイルシステム内の任意の場所にあるターゲットと組み合わせて複数の Entry Target ペアを作成でき、ユーザーの IAM ロールマッピングがそれらにアクセスする権限を持つ限り、複数のバケットまたはファイルシステムにわたっても可能です。

次の仮想構造の例では、ユーザーが AWS SFTP にログインすると、、/pics、、/doc/reportingおよび のサブディレクトリを持つルートディレクトリにあります/anotherpath/subpath/financials。

> **Note**
>
> Amazon S3 ディレクトリのパフォーマンスを最適化することを選択しない限り (サーバーを作成または更新するとき)、ディレクトリが存在しない場合は、ユーザーまたは管理者がディレクトリを作成する必要があります。この問題を回避することは、Amazon S3 のパフォーマンスの最適化を検討する理由です。
> Amazon EFS の場合、論理マッピングまたは / ディレクトリを作成する管理者が必要です。

```
[
{"Entry": "/pics", "Target": "/bucket1/pics"},
{"Entry": "/doc", "Target": "/bucket1/anotherpath/docs"},
{"Entry": "/reporting", "Target": "/reportingbucket/Q1"},
{"Entry": "/anotherpath/subpath/financials", "Target": "/reportingbucket/financials"}]
```

> **Note**
>
> ファイルは、マップした特定のフォルダにのみアップロードできます。つまり、前の例では、/anotherpath または anotherpath/subpath ディレクトリにはアップロードできず、anotherpath/subpath/financials のみが可能です。また、重複するパスは許可されないため、これらのパスに直接マップすることはできません。

たとえば、次のマッピングを作成するとします。

```
{
   "Entry": "/pics",
   "Target": "/mybucket/pics"
},
{
   "Entry": "/doc",
   "Target": "/mybucket/mydocs"
},
{
   "Entry": "/temp",
   "Target": "/mybucket"
}
```

ファイルは、それらのバケットのみにアップロードできます。sftp を通して初めて接続すると、ルートディレクトリ (/) の階層まで落ちます。そのディレクトリにファイルをアップロードしようとすると、アップロードは失敗します。次のコマンドは、シーケンスの例を示しています。

```
sftp> pwd
Remote working directory: /
sftp> put file
Uploading file to /file
remote open("/file"): No such file or directory
```

任意の directory/sub-directory にアップロードするには、パスを sub-directory に明示的にマップする必要があります。

ダウンロードして使用できる AWS CloudFormation テンプレートなど、論理ディレクトリとユーザーの設定の詳細については、 AWS ストレージブログの「chroot および論理ディレクトリを使用して AWS SFTP 構造を簡素化するchroot」を参照してください。

論理ディレクトリの構成例

この例では、ユーザーを作成し、2 つの論理ディレクトリを割り当てます。次のコマンドは、論理ディレクトリpicsとdocを使用して新しいユーザー (既存のTTransfer Family ilyサーバー用) を作成します。

```
aws transfer create-user --user-name marymajor-logical --server-id s-11112222333344445
 --role arn:aws:iam::1234abcd5678:role/marymajor-role --home-directory-type LOGICAL \
 --home-directory-mappings "[{\"Entry\":\"/pics\", \"Target\":\"/DOC-EXAMPLE-BUCKET1/
pics\"}, {\"Entry\":\"/doc\", \"Target\":\"/DOC-EXAMPLE-BUCKET2/test/mydocs\"}]" \
 --ssh-public-key-body file://~/.ssh/id_rsa.pub
```

marymajorが既存のユーザーで、ホームディレクトリのタイプがPATHの場合、前のユーザーと同様のコマンドを使ってこれをLOGICALに変更できます。

```
aws transfer update-user --user-name marymajor-logical \
 --server-id s-11112222333344445 --role arn:aws:iam::1234abcd5678:role/marymajor-role \
 --home-directory-type LOGICAL --home-directory-mappings "[{\"Entry\":\"/pics\",
 \"Target\":\"/DOC-EXAMPLE-BUCKET1/pics\"}, \
 {\"Entry\":\"/doc\", \"Target\":\"/DOC-EXAMPLE-BUCKET2/test/mydocs\"}]"
```

次の点に注意してください。

- ディレクトリ/DOC-EXAMPLE-BUCKET1/picsと/DOC-EXAMPLE-BUCKET2/test/mydocsがまだ存在しない場合は、ユーザ (または管理者) がディレクトリを作成する必要があります。
- **marymajor**をサーバーに接続してls -lコマンドを実行すると、次のように表示されます。

  ```
  drwxr--r--   1        -        -        0 Mar 17 15:42 doc
  drwxr--r--   1        -        -        0 Mar 17 16:04 pics
  ```

- **marymajor**はこのレベルではファイルやディレクトリは作成できません。ただし、picsおよびdoc内には、サブディレクトリを追加できます。
- 彼女がpicsとdocに追加したファイルと、それぞれAmazon S3 パス/DOC-EXAMPLE-BUCKET1/picsと/DOC-EXAMPLE-BUCKET2/test/mydocs にそれぞれ追加されます。
- この例では、その可能性を説明するために 2 つの異なるバケットを指定しています。ただし、ユーザーに指定した複数またはすべての論理ディレクトリに同じバケットを使用できます。

Amazon EFS の論理ディレクトリを設定する

Transfer Family サーバーが Amazon EFS を使用している場合は、ユーザーが論理ホームディレクトリで作業するには、ユーザーのホームディレクトリを読み取りおよび書き込みアクセスで作成する必要があります。ユーザーは自分の論理ホームディレクトリに対する mkdir の権限がないため、このディレクトリを自分で作成することはできません。

ユーザーのホームディレクトリが存在せず、ユーザーが `ls` コマンドを実行すると、システムは次のように応答します。

```
sftp> ls
remote readdir ("/"): No such file or directory
```

親ディレクトリへの管理アクセス権を持つユーザーは、ユーザーの論理ホームディレクトリを作成する必要があります。

カスタム AWS Lambda レスポンス

論理ディレクトリは、カスタム ID プロバイダーに接続する Lambda 関数で使用できます。そのためには、Lambda 関数で`HomeDirectoryType`を**`LOGICAL`**として指定し、`HomeDirectoryDetails`パラメータに`Entry`と`Target`の値を追加します。例:

```
HomeDirectoryType: "LOGICAL"
HomeDirectoryDetails: "[{\"Entry\": \"/\", \"Target\": \"/DOC-EXAMPLE-BUCKET/
theRealFolder"}]"
```

以下のコードは、カスタム Lambda 認証コールからの成功レスポンスの例です。

```
aws transfer test-identity-provider --server-id s-1234567890abcdef0 --user-name myuser
{
    "Url": "https://a1b2c3d4e5.execute-api.us-east-2.amazonaws.com/prod/servers/
s-1234567890abcdef0/users/myuser/config",
    "Message": "",
    "Response": "{\"Role\": \"arn:aws:iam::123456789012:role/bob-usa-role\",
\"HomeDirectoryType\": \"LOGICAL\",\"HomeDirectoryDetails\": \"[{\\\"Entry\\\":\\\"/
myhome\\\",\\\"Target\\\":\\\"/DOC-EXAMPLE-BUCKET/theRealFolder\\\"}]\",\"PublicKeys\":
 \"[ssh-rsa myrsapubkey]\"}",
    "StatusCode": 200
}
```

> **Note**
>
> `"Url":` の行は、カスタム ID プロバイダーとして API Gateway メソッドを使用している場合にのみ返されます。

AWS Transfer Family SFTP コネクタ

AWS Transfer Family SFTP コネクタは、SFTP プロトコルを使用して、Amazon ストレージと外部パートナー間でファイルとメッセージを送信するための関係を確立します。Amazon S3 からパートナーが所有する外部の宛先にファイルを送信できます。SFTP コネクタを使用して、パートナーの SFTP サーバーからファイルを取得することもできます。

> **Note**
>
> 現在、SFTP コネクタは、インターネットにアクセスできるエンドポイントを提供するリモート SFTP サーバーに接続するためにのみ使用できます。

次のブログ記事では、SFTP コネクタを使用して MFT ワークフローを構築するためのリファレンスアーキテクチャを提供します。これには、SFTP コネクタを使用してリモート SFTP サーバーに送信する前に PGP を使用してファイルを暗号化する、AWS Transfer Family SFTP コネクタと PGP 暗号化を使用した安全で準拠のマネージドファイル転送の設計が含まれます。

Transfer Family SFTP コネクタの簡単な概要については、「AWS Transfer Family SFTP コネクタ」を参照してください。

トピック

- SFTP コネクタを設定する
- SFTP コネクタを使用してファイルを送受信します。
- リモートディレクトリの内容を一覧表示する
- SFTP コネクタの管理

SFTP コネクタを設定する

このトピックでは、SFTP コネクタの作成方法、それらに関連付けられたセキュリティアルゴリズム、認証情報を保持するためのシークレットの保存方法、プライベートキーの書式設定の詳細、コネクタのテスト手順について説明します。

トピック

- SFTPコネクタを作成する

- SFTP コネクタで使用するシークレットを保存する
- SFTP コネクタプライベートキーの生成とフォーマット
- SFTP コネクタをテストします。

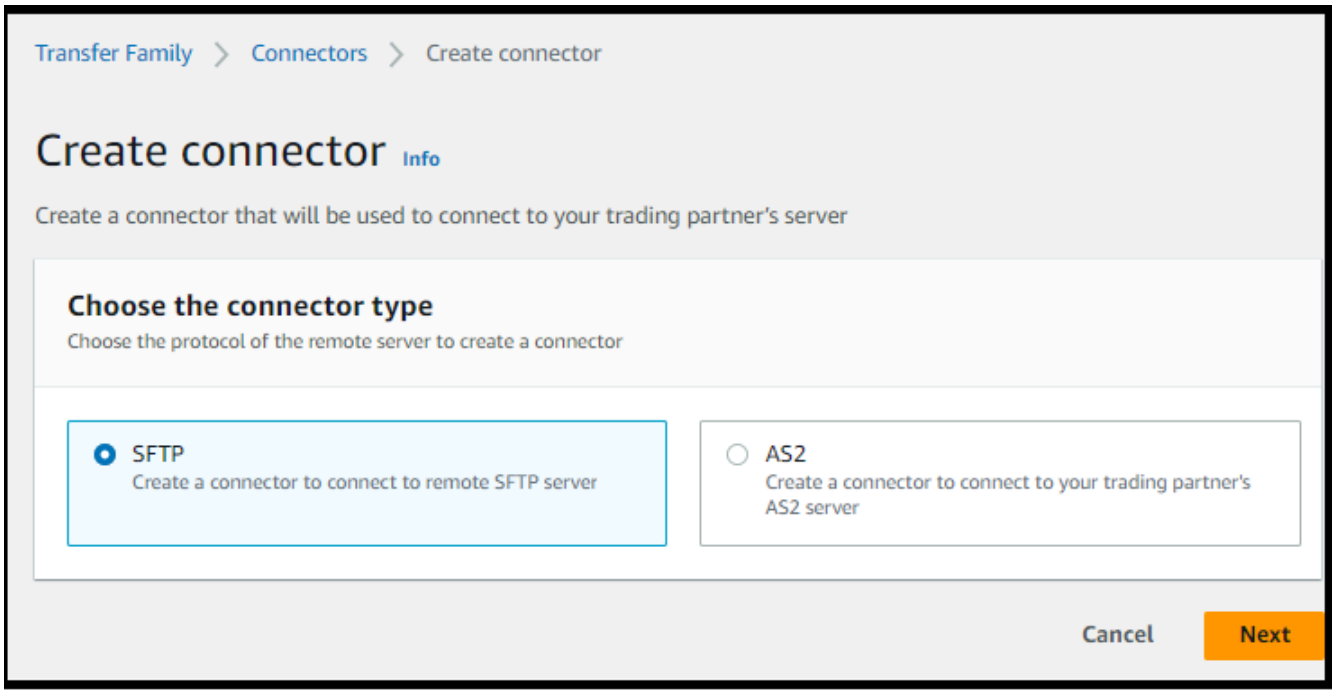
SFTPコネクタを作成する

この手順では、 AWS Transfer Family コンソールまたは を使用して SFTP コネクタを作成する方法について説明します AWS CLI。

Console

SFTPコネクタを作成するには

1. https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開きます。
2. 左側のナビゲーションペインで、[コネクタ] を選択し、[コネクタの作成] を選択します。
3. SFTP コネクタを作成するには、コネクタの種類として[SFTP] を選択し、[次へ] を選択します。



4. [コネクタ構成] セクションで、次の情報を入力します。
   - [URL] には、リモート SFTP サーバーの URL を入力します。この URL は `sftp://partner-SFTP-server-url`、たとえば次のような形式にする必要があります。`sftp://AnyCompany.com`

> **Note**
>
> オプションで、URL にポート番号を指定できます。形式は `sftp://partner-SFTP-server-url:port-number` です。デフォルトのポート番号 (ポートが指定されていない場合) はポート 22 です。

- アクセスロール で、使用する (IAM) ロールの Amazon リソースネーム AWS Identity and Access Management (ARN) を選択します。
  - `StartFileTransfer` リクエストで使用されるファイルロケーションの親ディレクトリに対して、このロールが読み取りと書き込みのアクセスを提供することを確認します。
  - **`secretsmanager:GetSecretValue`** このロールがシークレットへのアクセス許可を提供していることを確認してください。

    > **Note**
    >
    > ポリシーでは、シークレットの ARN を指定する必要があります。ARN にはシークレット名が含まれていますが、名前にはランダムな英数字 6 文字を追加します。シークレットの ARN の形式は次のとおりです。
    >
    > ```
    > arn:aws:secretsmanager:region:account-id:secret:aws/
    > transfer/SecretName-6RandomCharacters
    > ```

  - 「このロールに信頼関係が含まれていることを確認する」ことで、ユーザーの転送要求に対応する際に、コネクタがリソースにアクセスできません。信頼関係の確立の詳細については、信頼関係を確立するには を参照してください。

次の例では、Amazon S3 の *DOC-EXAMPLE-BUCKET* および Secrets Manager に保存されている指定されたシークレットにアクセスするために必要なアクセス許可を付与します。

Amazon S3

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
        "Sid": "AllowListingOfUserFolder",
        "Action": [
            "s3:ListBucket",
```

```
            "s3:GetBucketLocation"
        ],
        "Effect": "Allow",
        "Resource": [
            "arn:aws:s3:::DOC-EXAMPLE-BUCKET"
        ]
    },
    {
        "Sid": "HomeDirObjectAccess",
        "Effect": "Allow",
        "Action": [
            "s3:PutObject",
            "s3:GetObject",
            "s3:DeleteObject",
            "s3:DeleteObjectVersion",
            "s3:GetObjectVersion",
            "s3:GetObjectACL",
            "s3:PutObjectACL"
        ],
        "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
    },
    {
        "Sid": "GetConnectorSecretValue",
        "Effect": "Allow",
        "Action": [
            "secretsmanager:GetSecretValue"
        ],
        "Resource": "arn:aws:secretsmanager:region:account-id:secret:aws/
transfer/SecretName-6RandomCharacters"
    }
  ]
}
```

> **Note**
>
> アクセス ロールの場合、この例では 1 つのシークレットへのアクセスを許可します。ただし、ワイルドカード文字を使用すると、複数のユーザーとシークレットに対して同じ IAM ロールを再利用する場合に作業を節約できます。たとえば、次のリソース ステートメントは、aws/transfer で始まる名前を持つすべてのシークレットに対するアクセス許可を付与します。

```
"Resource": "arn:aws:secretsmanager:region:account-id:secret:aws/
transfer/*"
```

SFTP 認証情報を含むシークレットを別の AWS アカウントに保存することもできます。クロスアカウントシークレットアクセスの有効化の詳細については、「別のアカウント のユーザーの AWS Secrets Manager シークレットへのアクセス許可」を参照してください。

- (オプション) ログ記録ロール で、 CloudWatch ログにイベントをプッシュするために使用するコネクタの IAM ロールを選択します。次のポリシー例では、SFTP コネクタのイベントをログに記録するために必要なアクセス許可を一覧表示します。

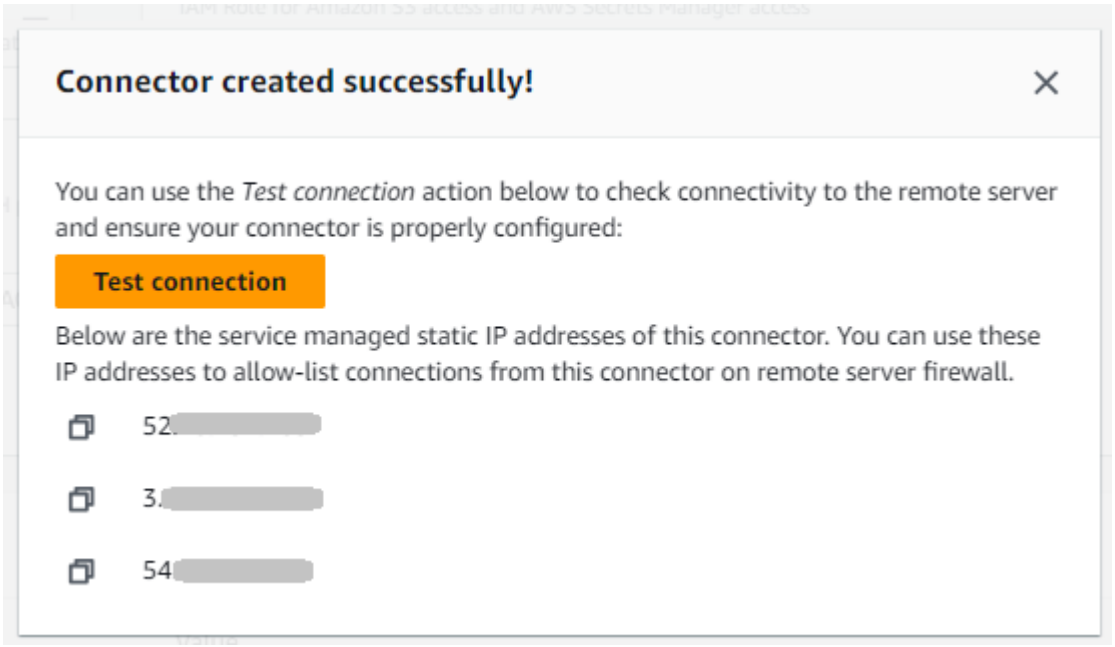
```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Sid": "SFTPConnectorPermissions",
        "Effect": "Allow",
        "Action": [
            "logs:CreateLogStream",
            "logs:DescribeLogStreams",
            "logs:CreateLogGroup",
            "logs:PutLogEvents"
        ],
        "Resource": [
            "arn:aws:logs:*:*:log-group:/aws/transfer/*"
        ]
    }]
}
```

5. 「SFTPの設定」セクションで、以下の情報を入力する：

- コネクタ認証情報 では、ドロップダウンリストから、SFTP ユーザーのプライベートキーまたはパスワード AWS Secrets Manager を含む のシークレットの名前を選択します。シークレットを作成し、特定の方法で保存する必要があります。詳細については、「SFTP コネクタで使用するシークレットを保存する」を参照してください。
- 信頼できるホストキー については、外部サーバーを識別するために使用されるホストキーのパブリック部分を貼り付けます。[信頼できるホストキーを追加] を選択してキーを追加することで、複数のキーを追加できます。SFTP サーバーに対して ssh-keyscan コマンドを使用して、必要なキーを取得できます。Transfer Familyがサポー

トするトラステッドホストキーの形式とタイプの詳細については、を参照してください「SFTPConnectorConfig」。

6. 暗号化アルゴリズムオプションセクションで、セキュリティポリシーフィールドのドロップダウンリストからセキュリティポリシーを選択します。セキュリティポリシーを使用すると、コネクタがサポートする暗号化アルゴリズムを選択できます。使用可能なセキュリティポリシーとアルゴリズムの詳細については、「」を参照してくださいAWS Transfer Family SFTP コネクタのセキュリティポリシー。
7. （オプション）[Tags] セクションの[Key] と[Value] に、1 つ以上のタグをキーと値のペアとして入力します。
8. すべての設定を確認したら、[コネクタの作成] を選択して SFTP コネクタを作成します。コネクタが正常に作成されると、割り当てられた静的 IP アドレスのリストと接続のテストボタンが画面に表示されます。ボタンを使用して、新しいコネクタの設定をテストします。



新しい SFTP [コネクタの] ID がリストに追加されたコネクタページが表示されます。コネクタの詳細を表示するには、SFTP コネクタの詳細を表示します。 を参照してください。

CLI

コネクタを作成するには、「create-connector」コマンドを使用します。このコマンドを使用して SFTP コネクタを作成するには、次の情報を指定する必要があります。

- リモート SFTP サーバーの URL。この URL は `sftp://`*`partner-SFTP-server-url`*、たとえば次のような形式にする必要があります。`sftp://AnyCompany.com`

- アクセスロール 使用する AWS Identity and Access Management （ IAM ）ロールの Amazon リソースネーム（ ARN ）を選択します。
  - `StartFileTransfer` リクエストで使用されるファイルロケーションの親ディレクトリに対して、このロールが読み取りと書き込みのアクセスを提供することを確認します。
  - **`secretsmanager:GetSecretValue`** このロールがシークレットへのアクセス許可を提供していることを確認してください。

    > **Note**
    >
    > ポリシーでは、シークレットの ARN を指定する必要があります。ARN にはシークレット名が含まれていますが、名前にはランダムな英数字 6 文字を追加します。シークレットの ARN の形式は次のとおりです。
    >
    > ```
    > arn:aws:secretsmanager:region:account-id:secret:aws/
    > transfer/SecretName-6RandomCharacters
    > ```

- 「このロールに信頼関係が含まれていることを確認する」ことで、ユーザーの転送要求に対応する際に、コネクタがリソースにアクセスできません。信頼関係の確立の詳細については、信頼関係を確立するには を参照してください。

次の例では、Amazon S3 の *DOC-EXAMPLE-BUCKET* および Secrets Manager に保存されている指定されたシークレットにアクセスするために必要なアクセス許可を付与します。 Amazon S3

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
        "Sid": "AllowListingOfUserFolder",
        "Action": [
            "s3:ListBucket",
            "s3:GetBucketLocation"
        ],
        "Effect": "Allow",
        "Resource": [
            "arn:aws:s3:::DOC-EXAMPLE-BUCKET"
        ]
    },
    {
```

```
        "Sid": "HomeDirObjectAccess",
        "Effect": "Allow",
        "Action": [
            "s3:PutObject",
            "s3:GetObject",
            "s3:DeleteObject",
            "s3:DeleteObjectVersion",
            "s3:GetObjectVersion",
            "s3:GetObjectACL",
            "s3:PutObjectACL"
        ],
        "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
    },
    {
        "Sid": "GetConnectorSecretValue",
        "Effect": "Allow",
        "Action": [
            "secretsmanager:GetSecretValue"
        ],
        "Resource": "arn:aws:secretsmanager:region:account-id:secret:aws/
transfer/SecretName-6RandomCharacters"
    }
  ]
}
```

Note

アクセス ロールの場合、この例では 1 つのシークレットへのアクセスを許可します。ただし、ワイルドカード文字を使用すると、複数のユーザーとシークレットに対して同じ IAM ロールを再利用する場合に作業を節約できます。たとえば、次のリソース ステートメントは、aws/transfer で始まる名前を持つすべてのシークレットに対するアクセス許可を付与します。

```
"Resource": "arn:aws:secretsmanager:region:account-id:secret:aws/transfer/
*"
```

SFTP 認証情報を含むシークレットを別の AWS アカウントに保存することもできます。クロスアカウントシークレットアクセスの有効化の詳細については、「別のアカウント のユーザーの AWS Secrets Manager シークレットへのアクセス許可」を参照してください。

- （オプション) CloudWatch ログにイベントをプッシュするために使用するコネクタの IAM ロールを選択します。次のポリシー例では、SFTP コネクタのイベントをログに記録するために必要なアクセス許可を一覧表示します。

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Sid": "SFTPConnectorPermissions",
        "Effect": "Allow",
        "Action": [
            "logs:CreateLogStream",
            "logs:DescribeLogStreams",
            "logs:CreateLogGroup",
            "logs:PutLogEvents"
        ],
        "Resource": [
            "arn:aws:logs:*:*:log-group:/aws/transfer/*"
        ]
    }]
}
```

- 次の SFTP 構成情報を入力します。
  - SFTP ユーザーのプライベートキーまたはパスワード AWS Secrets Manager を含む のシークレットの ARN。
  - 外部サーバーを識別するために使用されるホストキーの公開部分。必要に応じて、複数の信頼できるホスト キーを指定できます。

  SFTP 情報を提供する最も簡単な方法は、SFTP 情報をファイルに保存することです。たとえば、testSFTPConfig.json 次のサンプルテキストをという名前のファイルにコピーします。

```
// Listing for testSFTPConfig.json
{
   "UserSecretId": "arn:aws::secretsmanager:us-east-2:123456789012:secret:aws/
transfer/example-username-key",
   "TrustedHostKeys": [
      "sftp.example.com ssh-rsa AAAAbbbb...EEEE="
   ]
}
```

- コネクタのセキュリティポリシーを指定し、セキュリティポリシー名を入力します。

> **Note**
>
> `SecretId` は、ARN 全体またはシークレットの名前 (前のリストの *example-username-key*) のいずれかになります。

その後、次のコマンドを実行してコネクタを作成します。

```
aws transfer create-connector --url "sftp://partner-SFTP-server-url" \
--access-role your-IAM-role-for-bucket-access \
--logging-role arn:aws:iam::your-account-id:role/service-role/
AWSTransferLoggingAccess \
--sftp-config file:///path/to/testSFTPConfig.json
--security-policy-name security-policy-name
```

SFTP コネクタで使用するシークレットを保存する

Secrets Manager を使用して、SFTP コネクタのユーザー資格情報を保存できます。シークレットを作成するときは、ユーザー名を指定する必要があります。追加で、パスワード、プライベートキー、またはその両方を提供できます。詳細については、「SFTPコネクタのクォータ」を参照してください。

> **Note**
>
> Secrets Manager にシークレットを保存すると、 AWS アカウント に料金が発生します。料金については、「AWS Secrets Manager 料金表」を参照してください。

SFTPコネクタのユーザー資格情報をSecrets Managerに保存するために

1. にサインイン AWS Management Console し、https://console.aws.amazon.com/secretsmanager/ で AWS Secrets Manager コンソールを開きます。
2. 左側のナビゲーションペインで [サーバー] を選択します。
3. [シークレット]ページで、[新しいシークレットの保存]を選択します。
4. [シークレットタイプの選択] ページの[シークレットタイプ] で[その他のシークレットタイプ] を選択します。
5. [キー/値のペア] セクションで、[キー/値] タブを選択します。

- キー — **Username**と入力します。
- [値] — パートナーのサーバーへの接続を許可されているユーザーの名前を入力します。

6. パスワードを入力する場合は、[行を追加] を選択し、[キー/値のペア] セクションで[Key/Value] タブを選択します。

   [行を追加] を選択し、[キー/値のペア] セクションで[キー/値] タブを選択します。

   - キー — **Password**と入力します。
   - [値] ー ユーザーのパスワードを入力します。

7. プライベートキーを提供したい場合は、SFTP コネクタプライベートキーの生成とフォーマットを参照して、プライベートキーのデータの入力方法を説明しています。

   > **Note**
   >
   > 入力するプライベートキーデータは、リモートのSFTPサーバーにこのユーザーのために保存されている公開キーと対応している必要があります。

8. [次へ] をクリックします。
9. [シークレットの設定] ページで、シークレットの名前と説明を入力します。名前には **aws/transfer/** というプレフィックスを使用することをお勧めします。例えば、シークレットを **aws/transfer/connector-1** と名付けることができます。
10. [次へ] を選択し、[ローテーションの設定] ページのデフォルトを受け入れます。次いで、[次へ] を選択します。
11. [レビュー] ページで[ストア] を選択し、シークレットを作成して保存します。

SFTP コネクタプライベートキーの生成とフォーマット

パブリックキーとプライベートキーのペアを生成するための完全な詳細については、「」を参照してくださいmacOS、Linux、または UNIX で SSH キーを作成する。

例えば、SFTP コネクタで使用するプライベートキーを生成するには、次のサンプルコマンドで正しいタイプのキーを生成します (*key_name* をキーペアの実際のファイル名に置き換えます)。

```
ssh-keygen -t rsa -b 4096 -m PEM -f key_name -N ""
```

> **Note**
>
> SFTP コネクタで使用するキーペアを作成するときは、パスフレーズを使用しないでください。SFTP 設定が正しく機能するには、空のパスフレーズが必要です。

このコマンドは、キーサイズが 4096 ビットの RSA キーペアを作成します。キーはレガシー PEM 形式で生成されます。これは、Transfer Family が SFTP コネクタシークレットで使用するのに必要です。キーは、現在のディレクトリの *key_name* (プライベートキー) と *key_name*.pub (パブリックキー) に保存されます。つまり、ssh-keygenコマンドを実行するディレクトリです。

> **Note**
>
> Transfer Family は、SFTP コネクタに使用されるキーの OpenSSH 形式 (`-----BEGIN OPENSSH PRIVATE KEY-----`) をサポートしていません。キーはレガシー PEM 形式 (`-----BEGIN RSA PRIVATE KEY-----` または `-----BEGIN EC PRIVATE KEY-----`) である必要があります。コマンドを実行するときに `-m PEM` オプションを指定すると、ssh-keygen ツールを使用してキーを変換できます。

キーを生成したら、プライベートキーが JSON 形式の改行文字 (「\n」) で埋め込まれていることを確認する必要があります。

コマンドを使用して、既存のプライベートキーを改行文字が埋め込まれた JSON 形式に変換します。ここでは、 jqと Powershell の例を示します。プライベートキーを改行文字が埋め込まれた JSON 形式に変換する任意のツールまたはコマンドを使用できます。

jq command

この例では、 jq コマンドを使用します。 コマンドは、Download jq からダウンロードできます。

```
jq -sR . path-to-private-key-file
```

例えば、プライベートキーファイルが にある場合~/.ssh/my_private_key、コマンドは次のようになります。

```
jq -sR . ~/.ssh/my_private_key
```

これにより、キーが正しい形式 (改行文字が埋め込まれている) で標準出力に出力されます。

PowerShell

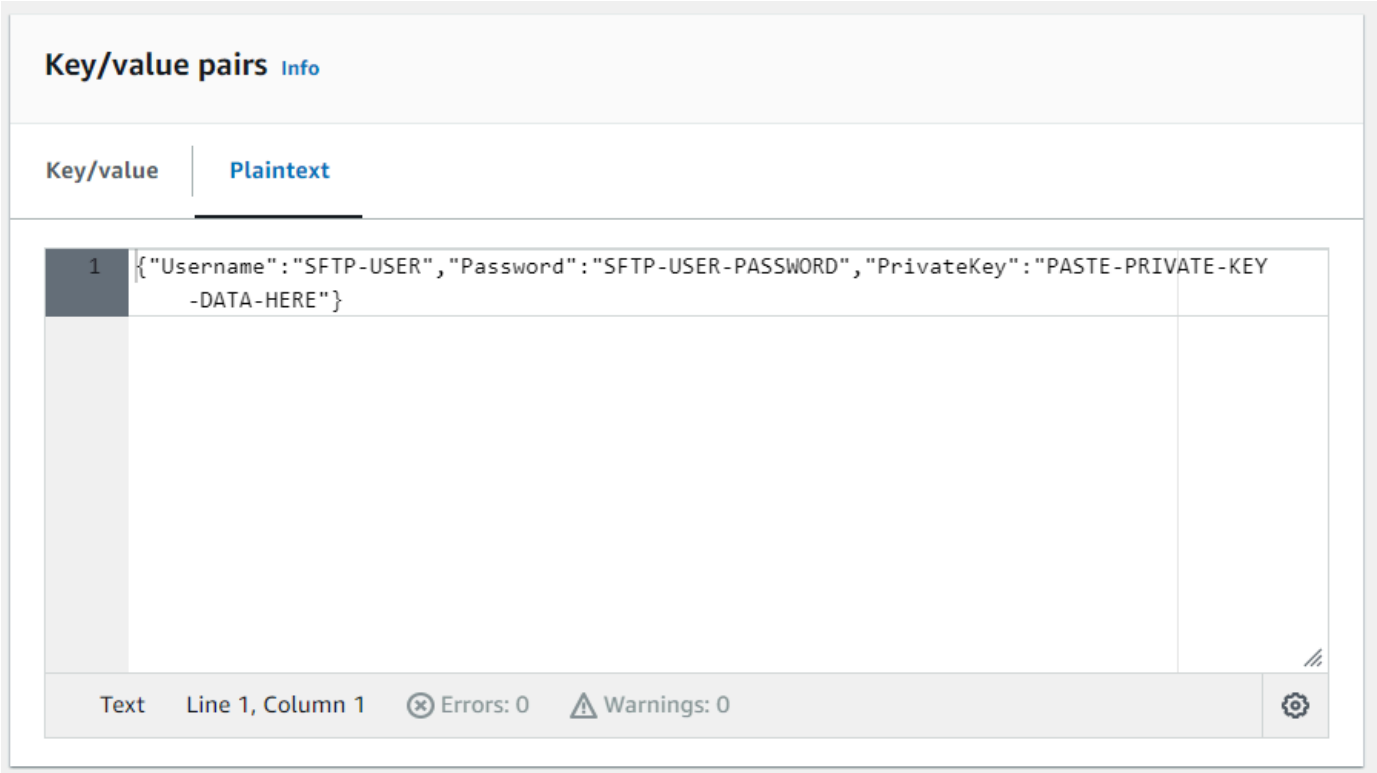
Windows を使用している場合は、 PowerShell を使用してキーを正しい形式に変換できます。次の Powershell コマンドは、プライベートキーを正しい形式に変換します。

```
Get-Content -Raw path-to-private-key-file | ConvertTo-Json
```

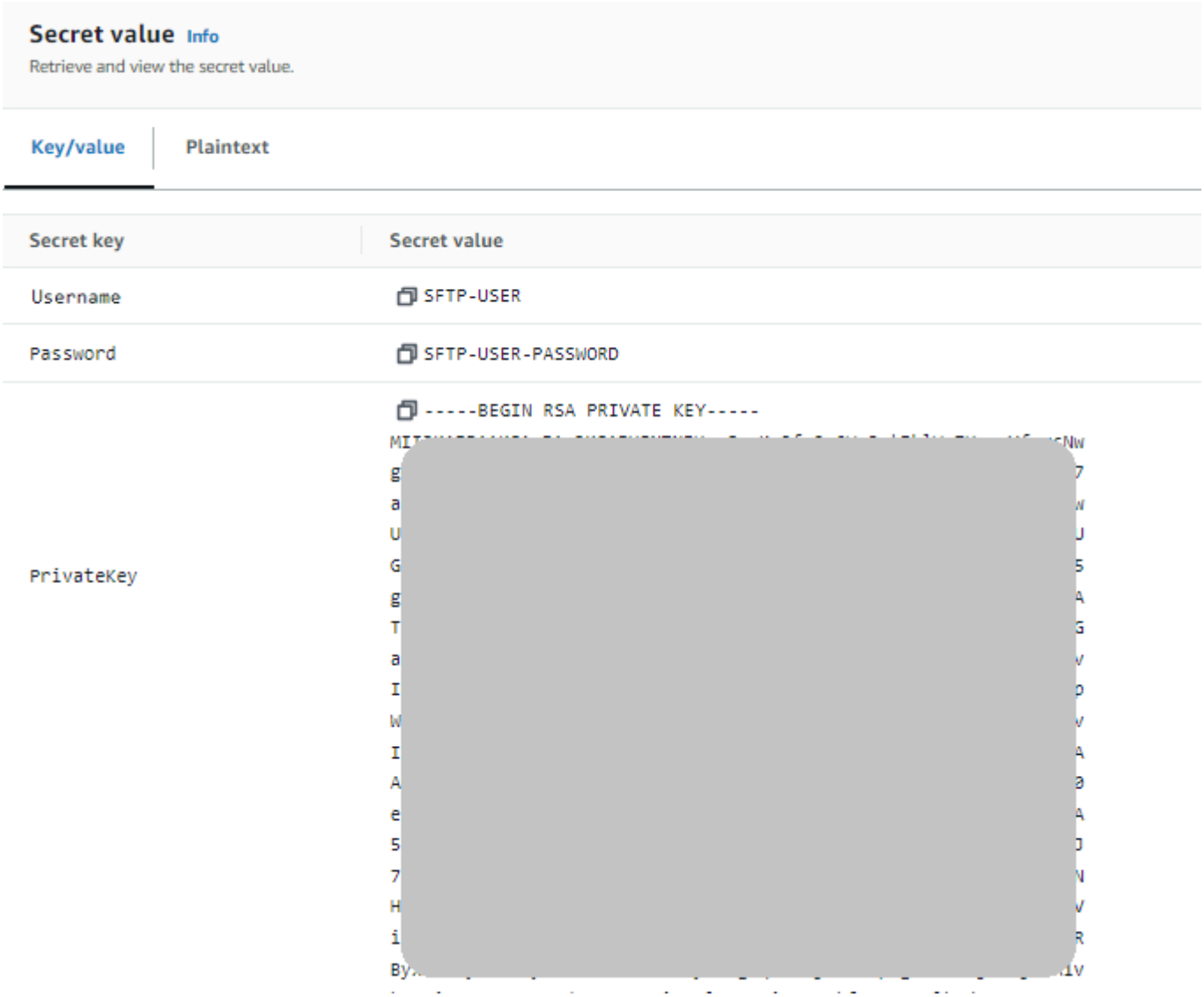
SFTPコネクタで使用するためにシークレットにプライベートキーデータを追加するには

1. Secrets Manager コンソールで、[その他のタイプのシークレットを] 保存する場合は、[プレーンテキスト] タブを選択します。テキストは空であり、左中括弧と右中括弧 {} のみが含まれている必要があります。
2. ユーザー名、プライベートキーデータ、および/またはパスワードを以下の形式で貼り付けてください。プライベートキーデータについては、ステップ1で実行したコマンドの出力を貼り付けてください。

```
{"Username":"SFTP-USER","Password":"SFTP-USER-PASSWORD","PrivateKey":"PASTE-PRIVATE-KEY-DATA-HERE"}
```



。プライベートキーデータを正しく貼り付けた場合、[キー/バリュー] タブを選択すると以下が表示されます。プライベートキーデータは、テキストの連続文字列としてではなく line-by-line、 として表示されることに注意してください。



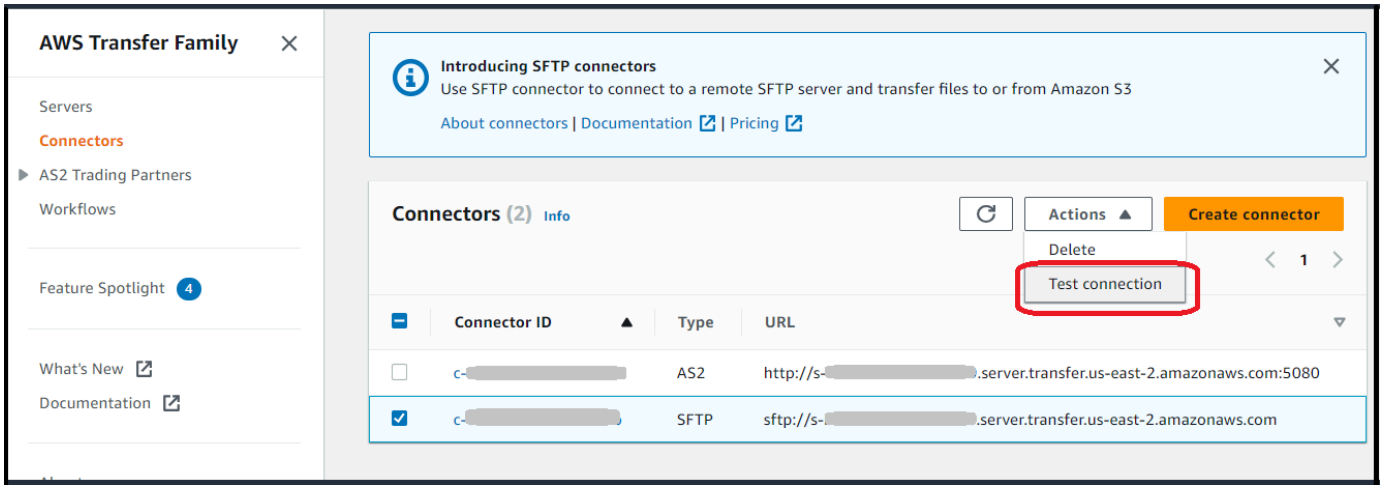
3. 引き続き SFTP コネクタで使用するシークレットを保存する の手順8の手順を最後まで行ってください。

SFTP コネクタをテストします。

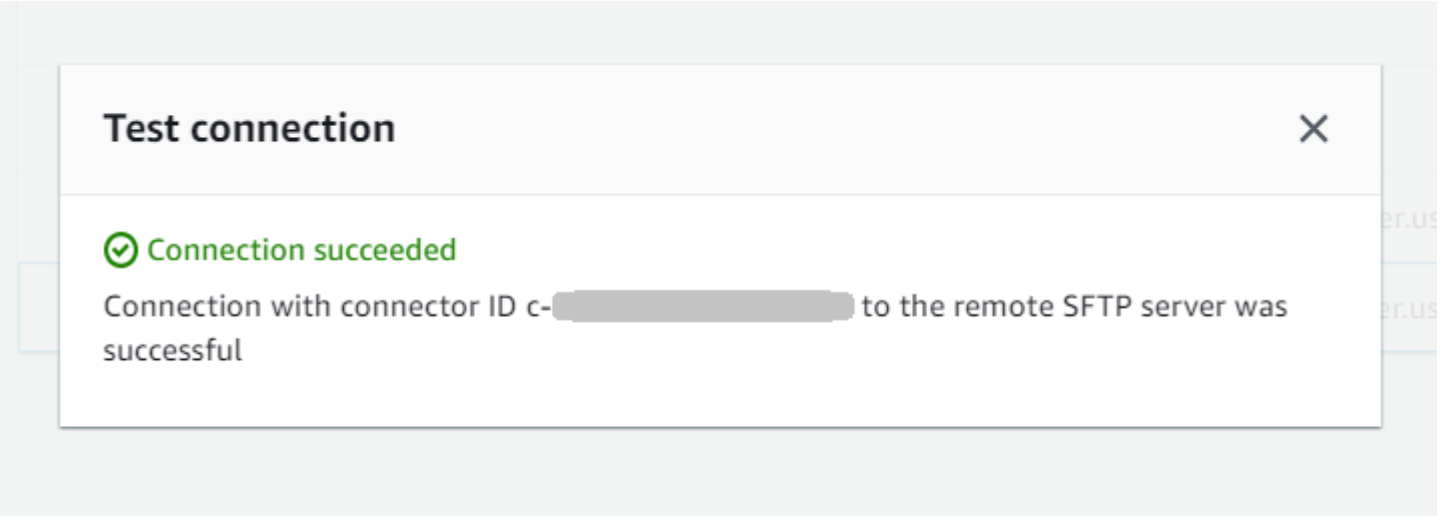
SFTP コネクタを作成した後、新しいコネクタを使用してファイルを転送する前にテストすることをお勧めします。

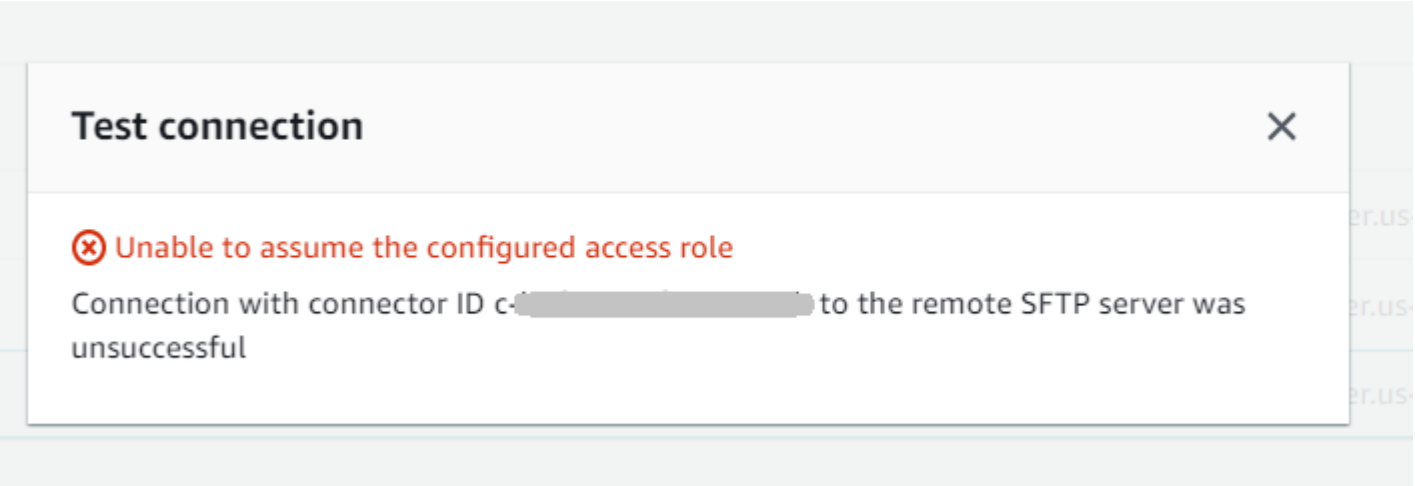
SFTP コネクタをテストするには

1. https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開きます。
2. 左側のナビゲーション ウィンドウで、[コネクタ] を選択し、コネクタを選択します。
3. [アクション] メニューから[テスト接続] を選択します。



システムはテストが合格したかどうかを示すメッセージを返します。テストに失敗した場合、システムは失敗の理由に基づいたエラーメッセージを提供します。





> **Note**
>
> コネクタをテストするためにAPIを使用するには、「TestConnection」 APIドキュメントを参照してください。

SFTP コネクタを使用してファイルを送受信します。

SFTP コネクタは、クラウドとオンプレミスの両方でリモートサーバーと通信 AWS Transfer Family するために の機能を拡張します。リモートソースで生成および保存されたデータを、分析、ビジネスアプリケーション、レポート、監査のために AWS ホストされたデータウェアハウスと統合できます。

リモート SFTP サーバーへのファイル転送を開始するには、SFTP コネクターを使用して転送を行う StartFileTransferAPI 操作を使用します。各`StartFileTransfer`リクエストには 10 個の異なるパスを含めることができます。

サーバーログを確認することで、ファイル転送を監視できます。コネクタアクティビティは、`aws/transfer/c-1234567890abcdef0` などの `aws/transfer/`*`connector-id`* 形式のログストリームに記録されます。コネクタのログが表示されない場合は、コネクタの正しい権限を持つロギングロールを指定していることを確認してください。

コネクターの作成については、SFTP コネクタを設定する を参照してください。

SFTP コネクタを使用してファイルを送信および取得するには、 `start-file-transfer` AWS Command Line Interface （ AWS CLI) コマンドを使用します。ファイルの送信 (アウトバウンド転送) か、ファイルの受信 (インバウンド転送) かによって、次のパラメータを指定します。

- アウトバウンド転送
  - send-file-paths には、パートナーのSFTPサーバーに転送するファイルのソースファイ ルパスが1～10個含まれています。
  - remote-directory-pathは、顧客の SFTP サーバ上でファイルを送信するリモートパスです。
- インバウンド転送
  - retrieve-file-pathsには1本から10本のリモートパスが含まれます。各パスは、パートナーのSFTPサーバーからTransfer Familyサーバーにファイルを転送する場所を指定します。
  - local-directory-pathは、ファイルが保存されている Amazon S3 の場所 (バケットとオプションのプレフィックス) です。

ファイルを送信するには、send-file-pathsとremote-directory-pathパラメータを指定します。send-file-pathsパラメータには最大 10 個のファイルを指定できます。以下のコマンド例では、Amazon S3 ストレージにある /DOC-EXAMPLE-SOURCE-BUCKET/file1.txt と /DOC-EXAMPLE-SOURCE-BUCKET/file2.txt という名前のファイルを、パートナーの SFTP サーバー上の /tmp ディレクトリに送信します。この例のコマンドを使うには、*DOC-EXAMPLE-SOURCE-BUCKET* を自分のバケツに置き換えます。

```
aws transfer start-file-transfer --send-file-paths /DOC-EXAMPLE-SOURCE-BUCKET/
file1.txt /DOC-EXAMPLE-SOURCE-BUCKET/file2.txt \
    --remote-directory-path /tmp --connector-id c-1111AAAA2222BBBB3 --region us-east-2
```

ファイルを受信するには、retrieve-file-pathsとlocal-directory-pathパラメータを指定します。次の例では、パートナーの SFTP サーバー/my/remote/file1.txt/my/remote/file2.txt上のファイル と を取得し、Amazon S3 の場所 /DOC-EXAMPLE-BUCKET/*prefix* に配置します。このコマンドの例を実行するには、*user input placeholders* をユーザー自身の情報に置き換えます。

```
aws transfer start-file-transfer --retrieve-file-paths /my/remote/file1.txt  /my/
remote/file2.txt \
   --local-directory-path /DOC-EXAMPLE-BUCKET/prefix --connector-id c-2222BBBB3333CCCC4
 --region us-east-2
```

前の例では、SFTP サーバー上の絶対パスを指定しています。相対パスを使用することもできます。相対パスは、SFTPユーザーのホームディレクトリへの相対パスです。たとえば、SFTP ユーザーがmarymajorで、SFTP サーバー上のホームディレクトリが/users/marymajor/の場合、

次のコマンドは/DOC-EXAMPLE-SOURCE-BUCKET/file1.txtを/users/marymajor/test-connectors/file1.txtに送信します。

```
aws transfer start-file-transfer --send-file-paths /DOC-EXAMPLE-SOURCE-BUCKET/file1.txt
 \
   --remote-directory-path test-connectors --connector-id c-2222BBBB3333CCCC4 --
region us-east-2
```

リモートディレクトリの内容を一覧表示する

リモート SFTP サーバーからファイルを取得する前に、リモート SFTP サーバー上のディレクトリの内容を取得できます。これを行うには、 StartDirectoryListing API コールを使用します。

次の例では、コネクタの設定で指定されているリモート SFTP サーバー上の homeフォルダの内容を一覧表示します。結果は Amazon S3 の場所 に配置され/DOC-EXAMPLE-BUCKET/connector-files、 という名前のファイルに配置されますc-AAAA1111BBBB2222C-6666abcd-11aa-22bb-cc33-0000aaaa3333.json。

```
aws transfer start-directory-listing  \
   --connector-id c-AAAA1111BBBB2222C  \
   --output-directory-path /DOC-EXAMPLE-BUCKET/example/connector-files  \
   --remote-directory-path /home
```

この AWS CLI コマンドは、リスト ID と結果を含むファイルの名前を返します。

```
{
    "ListingId": "6666abcd-11aa-22bb-cc33-0000aaaa3333",
    "OutputFileName": "c-AAAA1111BBBB2222C-6666abcd-11aa-22bb-cc33-0000aaaa3333.json"
}
```

> **Note**
>
> 出力ファイルの命名規則は ですconnector-ID-listing-ID.json。

JSON ファイルには、次の情報が含まれています。

- filePath: リモートサーバー上の SFTP コネクタのリストリクエストのディレクトリを基準とした、リモートファイルの完全なパス。

- modifiedTimestamp: ファイルが最後に変更された時刻、秒単位、協定世界時 (UTC) 形式。このフィールドはオプションです。リモートファイル属性にタイムスタンプが含まれていない場合、ファイルリストから除外されます。
- size: ファイルのサイズ、バイト単位。このフィールドはオプションです。リモートファイル属性にファイルサイズが含まれていない場合は、ファイルリストから省略されます。
- path: リモートサーバー上の SFTP コネクタのリストリクエストのディレクトリに対するリモートディレクトリの完全なパス。
- truncated: リスト出力にリモートディレクトリに含まれるすべての項目が含まれているかどうかを示すフラグ。truncated 出力値が true の場合、オプションのmax-items入力属性で指定された値を増やして、より多くの項目を一覧表示できます (最大許容リストサイズは 10,000 項目まで) 。

以下は、リモートディレクトリに 2 つのファイルと 2 つのサブディレクトリ (パスc-AAAA1111BBBB2222C-6666abcd-11aa-22bb-cc33-0000aaaa3333.json) が含まれる出力ファイル () の内容の例です。

```
{
    "files": [
        {
            "filePath": "/home/what.txt",
            "modifiedTimestamp": "2024-01-30T20:34:54Z",
            "size" : 2323
        },
        {
            "filePath": "/home/how.pgp",
            "modifiedTimestamp": "2024-01-30T20:34:54Z",
            "size" : 4691
        }
    ],
    "paths": [
        {
            "path": "/home/magic"
        },
        {
            "path": "/home/aws"
        },
    ],
    "truncated": "false"
}
```

SFTP コネクタの管理

このトピックでは、SFTP コネクタを表示および更新する方法を説明し、SFTP コネクタに関連するクォータを一覧表示します。

> **Note**
>
> 各コネクタには、コネクタの存続期間中変更されない静的 IP アドレスが自動的に割り当てられます。これにより、既知の IP アドレスからのインバウンド接続のみを受け入れるリモート SFTP サーバーに接続できます。コネクタには、 で同じプロトコル (SFTP または AS2) を使用してすべてのコネクタによって共有される静的 IP アドレスのセットが割り当てられます AWS アカウント。

トピック

SFTP コネクタのアップデート

コネクタの既存のパラメータ値を変更するには、update-connectorコマンドを実行します。以下のコマンドは、*region-id*から*secret-ARN*のリージョン内のコネクター*connector-id*のシークレットをに更新します。このコマンドの例を実行するには、*user input placeholders* をユーザー自身の情報に置き換えます。

```
aws transfer update-connector --sftp-config '{"UserSecretId":"secret-ARN"}' \
   --connector-id connector-id --region region-id
```

SFTP コネクタの詳細を表示します。

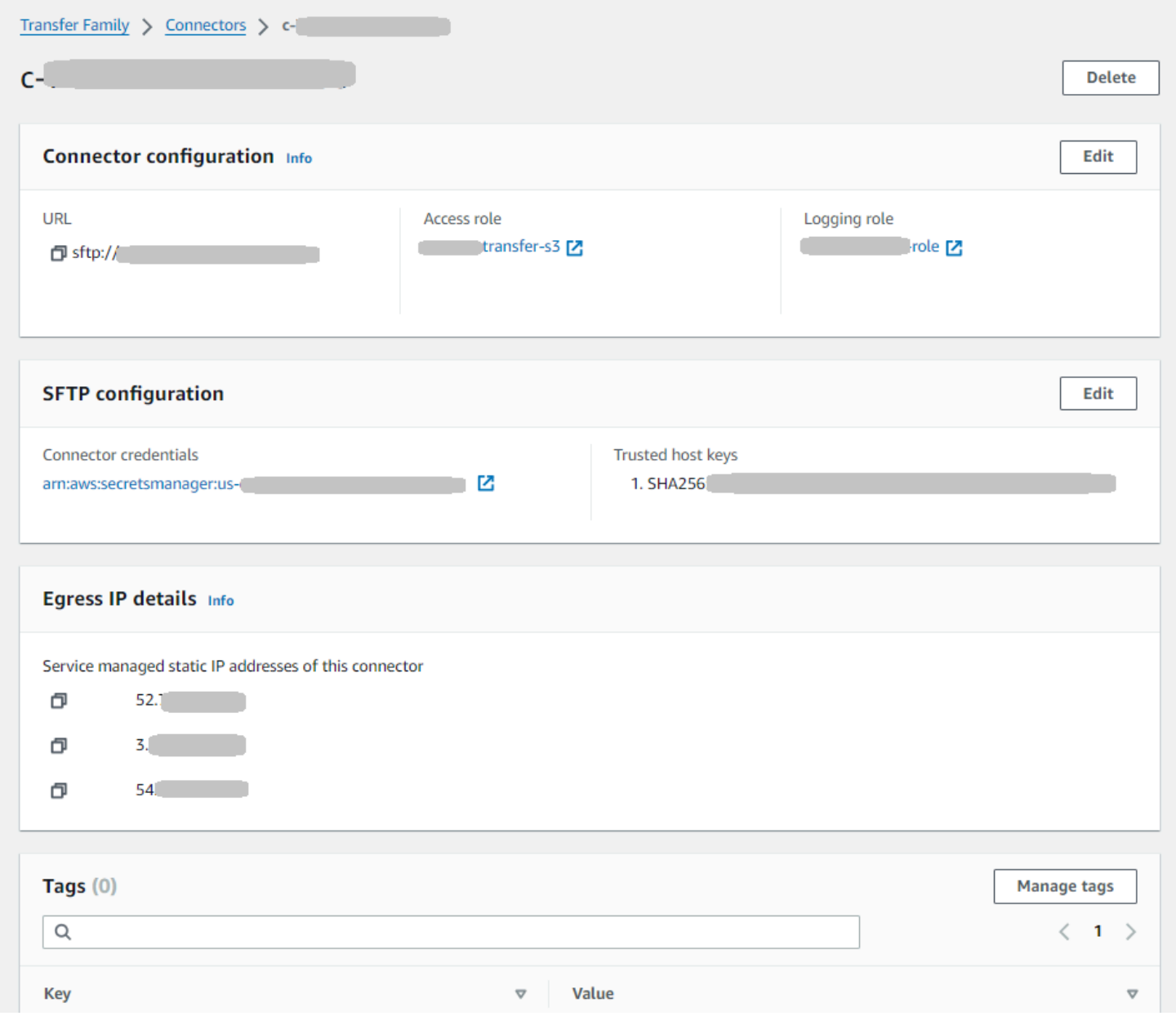
AWS Transfer Family コンソールで、SFTP コネクタの詳細とプロパティのリストを確認できます。

コネクタの詳細を表示するには

1. https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開きます。

2. 左側のナビゲーションペインで、[Connectors (コネクタ)] を選択します。
3. [コネクタ ID] 列の識別子を選択すると、選択したコネクタの詳細ページが表示されます。

SFTP コネクタのプロパティは、コネクタの詳細ページで 「編集」を選択して変更できます。



> **Note**
>
> 次の AWS Command Line Interface （ AWS CLI) コマンドを実行すると、この情報の多くを別の形式で取得できます。このコマンドの例を実行するには、*user input placeholders* をユーザー自身の情報に置き換えます。

```
aws transfer describe-connector --connector-id your-connector-id
```

詳細については、API リファレンスのDescribeConnectorを参照してください。

SFTPコネクタのクォータ

SFTP コネクタには次のクォータが設定されています。

> **Note**
>
> SFTP コネクタのその他のサービスクォータは、 のAWS Transfer Family エンドポイントとクォータに記載されていますAmazon Web Services 全般のリファレンス。

SFTP Connector クォータ

名前	デフォルト	引き上げ可能
最大テスト接続トランザクション / 秒 (TPS)	1 アカウントにつき毎秒 1 リクエスト	いいえ
保留中のファイル転送の最大キューサイズ	1,000	いいえ
最大ファイルサイズ	50 ギビバイト (GiB)	いいえ
1 ファイルあたりの最大転送時間	6 時間	いいえ
1 ファイルあたりの最大リクエスト待機時間	6 時間	いいえ
アカウントあたりのコネクタの最大帯域幅 (SFTP コネクタと AS2 コネクタの両方がこの値に影響します)	50 MBps	いいえ

SFTP コネクタの認証情報を保存するには、各 Secrets Manager シークレットに関連付けられたクォータがあります。同じシークレットを使用して複数のタイプのキーを保存すると、複数の目的でこれらのクォータが発生する可能性があります。

- 1 つのシークレットの合計長: 12,000 文字
- **`Password`** 文字列の最大長: 1024 文字
- **`PrivateKey`** 文字列の最大長: 8,192 文字
- **`Username`** 文字列の最大長: 100 文字

AWS Transfer Family AS2 の

適用性ステートメント 2 (AS2) は RFC が定義したファイル転送仕様で、強力なメッセージ保護と検証メカニズムが含まれています。AS2 プロトコルは、データ保護とセキュリティ機能をプロトコルに組み込むことが前提となるコンプライアンス要件のあるワークフローにとって重要です。

> **Note**
>
> Transfer Family AS2は「ドラモンド認定」を受けています。

サプライチェーン、物流、決済のワークフローで AS2 を利用している小売、ライフサイエンス、製造、金融サービス、公益事業などの業界のお客様は、 AWS Transfer Family AS2 エンドポイントを使用してビジネスパートナーと安全に取引できます。トランザクションされたデータは、処理、分析、機械学習 AWS のために でネイティブにアクセスできます。このデータは、 AWS上で動作する企業資源計画（ERP）や顧客関係管理（CRM）システムとの統合にも利用できます。AS2 を使用すると、既存のビジネスパートナーの統合とコンプライアンス AWS を維持しながら、 business-to-business (B2B) トランザクションを大規模に実行できます。

Transfer Familyのお客様で、AS2対応サーバーを設定しているパートナーとファイルを交換したい場合、セットアップでは、暗号化用の公開鍵と秘密鍵のペアを1つ生成し、もう1つの公開鍵に署名してパートナーと交換します。

Transfer Family には、AS2 を有効にして Transfer Family エンドポイントを設定するワークショップと、Transfer Family AS2 コネクタが用意されています。このワークショップの詳細については、こちらを参照してください。

転送中の AS2 ペイロードを保護するには通常、暗号メッセージ構文 (CMS) を使用し、通常は暗号化とデジタル署名を使用してデータ保護とピア認証を行います。署名付きメッセージ処理通知 (MDN) レスポンスペイロードは、メッセージが受信され、正常に復号化されたことを確認 (否認防止) します。

これらの CMS ペイロードと MDN 応答は HTTP 経由で転送されます。

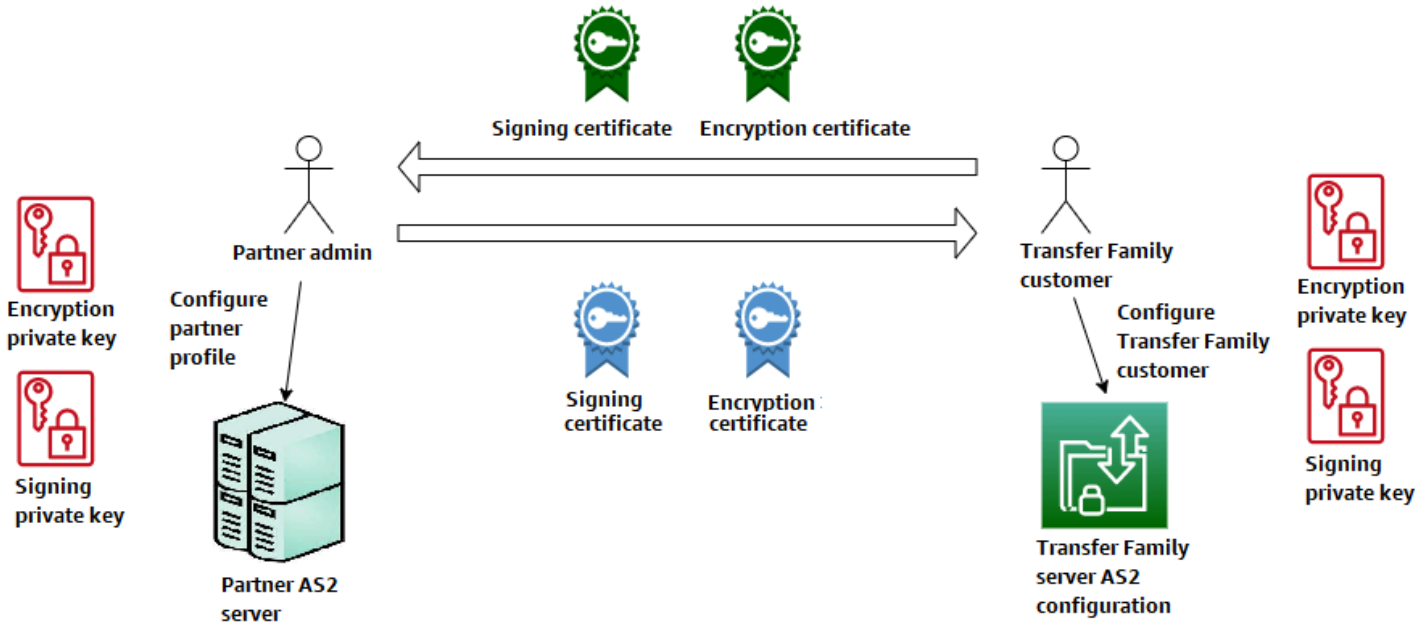
> **Note**
>
> HTTPS AS2 サーバーのエンドポイントは現在サポートされていません。現在、TLS の終了はお客様の責任です。

適用性ステートメント 2 (AS2) 設定の設定の詳細な step-by-step ウォークスルーについては、「」チュートリアルを参照してくださいAS2 設定のセットアップ。

トピック

AS2 ユースケース

AS2 サーバーが設定されたパートナーとファイル AWS Transfer Family を交換する場合は、セットアップの最も複雑な部分として、暗号化用に 1 つのパブリックキーとプライベートキーのペアを生成し、もう 1 つのパブリックキーに署名してパートナーと交換します。



AS2 で を使用する場合は AWS Transfer Family 、次のバリエーションを考慮してください。

> **Note**
>
> 取引先は、そのパートナープロファイルに関連付けられたパートナーです。

以下の表の「MDN」に関する記述はすべて、「署名付き」MDN を前提としています。

AS2 ユースケース

インバウンドのみのユースケース

- 暗号化された AS2 メッセージを取引相手から Transfer Family サーバーに転送します。

  この場合、次の操作を行います。

  1. 取引相手と自分自身のプロファイルを作成します。
  2. AS2 プロトコルを使用するTransfer Family サーバーを作成します。
  3. 契約書を作成してサーバーに追加します。
  4. プライベートキーを使用して証明書をインポートし、プロファイルに追加してから、パブリックキーをパートナープロファイルにインポートして暗号化します。
  5. これらの項目を入手したら、証明書の公開鍵を取引相手に送信します。

  これで、パートナーは暗号化されたメッセージを送信できるようになり、ユーザーはそれらを復号して Amazon S3 バケットに保存できます。
- 暗号化された AS2 メッセージを取引相手から Transfer Family サーバーに転送し、署名を追加します。

  このシナリオでは、まだ受信転送のみを行っていますが、今度はパートナーが送信するメッセージに署名してもらいたいと考えています。この場合、取引相手の署名パブリックキー (パートナーのプロファイルに追加された署名証明書として) をインポートします。
- 暗号化された AS2 メッセージを取引相手から Transfer Family サーバーに転送し、署名と MDN レスポンスの送信を追加します。

  このシナリオでは、まだインバウンド転送のみを行っていますが、取引相手は署名されたペイロードの受信に加えて、署名済みの MDN レスポンスを受信したいと考えています。

  1. 公開署名鍵と秘密署名鍵を (署名証明書としてプロファイルに) インポートします。
  2. 公開署名キーを取引相手に送信します。

アウトバウンドのみのユースケース

- 暗号化された AS2 メッセージを Transfer Family サーバーから取引相手に転送します。

このケースは受信専用転送のユースケースと似ていますが、AS2 サーバーに契約を追加する代わりにコネクタを作成するという点が異なります。この場合、取引相手のパブリックキーをそのプロファイルにインポートします。

- 暗号化された AS2 メッセージを Transfer Family サーバーから取引相手に転送し、署名を追加します。

  まだアウトバウンド転送のみを行っていますが、取引相手が送信するメッセージに署名するように要求するようになりました。

  1. (プロファイルに追加されたサイン証明書として) サイン用プライベートキーをインポートします。
  2. 取引相手にパブリックキーを送信します。
- 暗号化された AS2 メッセージを Transfer Family サーバーから取引相手に転送し、署名を追加して MDN レスポンスを送信します。

  まだアウトバウンド転送のみを行っていますが、現在は署名付きペイロードの送信に加えて、取引相手から署名付き MDN レスポンスを受け取る必要があります。

  1. 取引相手が公開署名キーを送信します。
  2. 取引相手のパブリックキー (パートナープロファイルに追加された署名証明書として) をインポートします。

インバウンドとアウトバウンドのユースケース

- Transfer Family サーバーと取引相手の間で、暗号化された AS2 メッセージを双方向に転送します。

  この場合、次の操作を行います。

  1. 取引相手と自分自身のプロファイルを作成します。
  2. AS2 プロトコルを使用するTransfer Family サーバーを作成します。
  3. 契約書を作成してサーバーに追加します。
  4. コネクタを作成します。
  5. プライベートキーを使用して証明書をインポートし、プロファイルに追加してから、パブリックキーをパートナープロファイルにインポートして暗号化します。
  6. 取引相手からパブリックキーを受け取り、暗号化のためにプロファイルに追加します。
  7. これらの項目を入手したら、証明書の公開鍵を取引相手に送信します。

  これで、あなたと取引相手は暗号化されたメッセージを交換でき、両方とも暗号化を解除できます。受信したメッセージは Amazon S3 バケットに保存でき、パートナーは送信したメッセージを復号化して保存できます。
- Transfer Family サーバーと取引相手の間で暗号化された AS2 メッセージを双方向に転送し、署名を追加します。

  これで、あなたとパートナーは署名付きのメッセージを求めます。

  1. (プロファイルに追加されたサイン証明書として) サイン用プライベートキーをインポートします。
  2. 取引相手にパブリックキーを送信します。
  3. 取引相手の署名パブリックキーをインポートし、プロファイルに追加します。
- Transfer Family サーバーと取引相手の間で暗号化された AS2 メッセージを双方向に転送し、署名を追加して MDN レスポンスを送信します。

  次に、署名付きペイロードを交換する場合、自分と取引相手の両方が MDN レスポンスを必要としています。

  1. 取引相手が公開署名キーを送信します。
  2. 取引相手のパブリックキーを (パートナープロファイルの署名証明書として) インポートします。

3. 公開キーを取引相手に送信します。

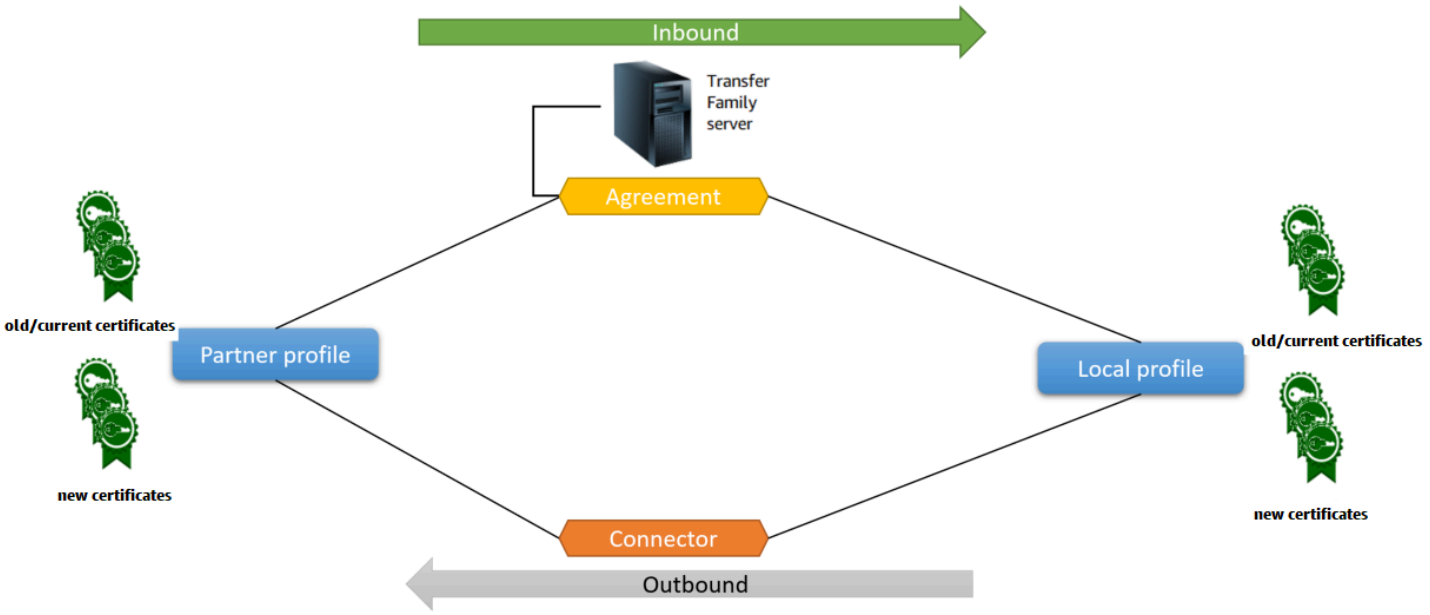
AS2 の設定

AS2 対応サーバーを作成するには、以下のコンポーネントも指定する必要があります。

- 「契約」 — 二国間取引先「契約」 (パートナーシップ) は、メッセージ (ファイル) を交換する 2 者間の関係を定義します。アグリーメントを定義するために、Transfer Family はサーバ、ローカルプロファイル、パートナープロファイル、証明書情報を組み合わせます。Transfer Family AS2-インバウンドプロセス使用契約。
- 「証明書」 — 「公開鍵 (X.509) 証明書」は AS2 通信でメッセージの暗号化と検証に使用されます。証明書はコネクタエンドポイントにも使用されます。
- 「ローカルプロファイルとパートナープロファイル」 — 「ローカルプロファイル」は、ローカル（AS2対応のTransTransfer Family ilyサーバー）組織または「パーティ」を定義します。同様に、「パートナープロファイル」は、TTransfer Family ilyの外部にあるリモートパートナー組織を定義します。

すべての AS2 対応サーバーには必要ありませんが、アウトバウンド転送には「コネクタ」が必要です。コネクタは、アウトバウンド接続のパラメータを取得します。コネクタは、お客様の外部非 AWS サーバーにファイルを送信するために必要です。

次の図表は、インバウンドプロセスとアウトバウンドプロセスに関係する AS2 オブジェクトの関係を示しています。



AS2 設定 end-to-end の例については、「」を参照してくださいAS2 設定のセットアップ。

トピック

- Transfer Family コンソールを使用して AS2 サーバーを作成する
- テンプレートを使用して Transfer Family AS2 スタックのデモを作成する
- AS2 の設定とクォータ
- AS2 の特徴と機能

Transfer Family コンソールを使用して AS2 サーバーを作成する

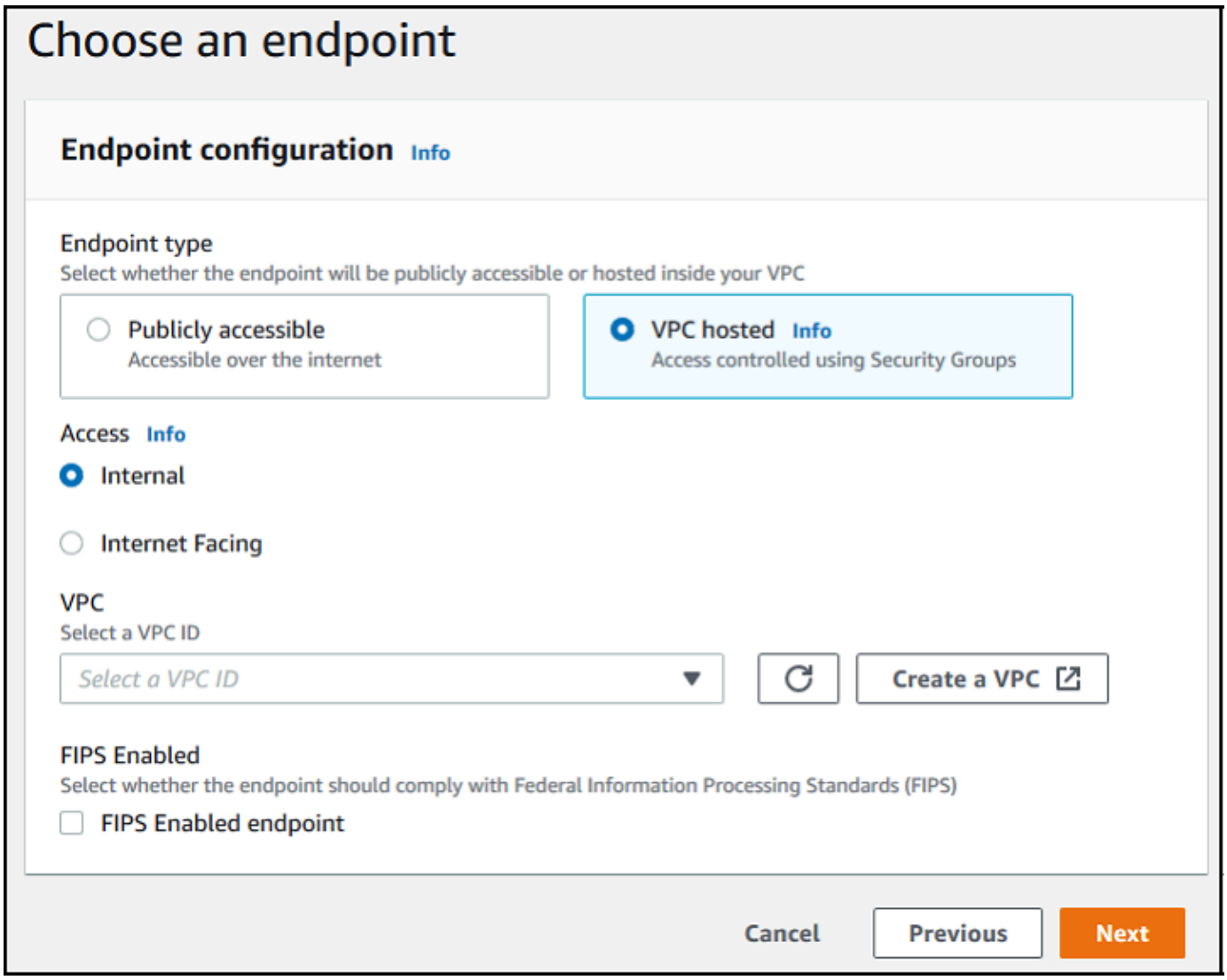
ここでは、Transfer Family コンソールを使用して AS2 対応サーバーを作成する方法について説明します。 AWS CLI 代わりに を使用する場合は、「」を参照してくださいthe section called “ステップ 2: AS2 プロトコルを使用する Transfer Family サーバーを作成する”。

AS2 対応サーバーを作成するには

1. https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開きます。
2. 左のナビゲーションペインで「サーバー」を選択し、「サーバーの作成」を選択します。
3. 「プロトコルの選択」ページで「AS2 (適用性ステートメント 2)」を選択し、「次へ」を選択します。
4. 「ID プロバイダーの選択」ページで、「次へ」を選択します。

> **Note**
>
> AS2 では、AS2 プロトコルでは基本認証がサポートされていないため、ID プロバイダーを選択できません。代わりに、仮想プライベートクラウド (VPC) セキュリティグループを介してアクセスを制御します。

5. 「エンドポイントの選択」ページで、次のようにする：



a. [Endpoint type] (エンドポイントタイプ) で、サーバーのエンドポイントのホストになる VPC ホストテッドエンドポイントタイプを選択します。VPC ホスト・エンドポイントの設定については、Virtual Private Cloud でサーバーを作成するを参照してください。

> **Note**
>
> AS2 プロトコルでは、パブリックにアクセス可能なエンドポイントはサポートされません。VPC エンドポイントにインターネット経由でアクセスできるようにするには、「アクセス」で「インターネット向け」を選択し、Elastic IP アドレスを指定します。

b. 「アクセス」の場合は、以下のオプションのいずれかを選択する：

- 「内部」 — このオプションを選択すると、VPC内およびVPCに接続された環境 (オンプレミスのデータセンターなど) から AWS Direct Connect またはVPN経由でアクセス可能になります。
- インターネット向け – このオプションを選択すると、インターネット経由で、VPC 内および VPC 接続環境内から、 AWS Direct Connect または VPN 経由のオンプレミスデータセンターなどにアクセスできるようになります。

  「インターネット向け」を選択した場合は、プロンプトが表示されたら Elastic IP アドレスを入力します。

c. 「VPC」では、既存の VPC を選択するか、「VPC の作成」を選択して新しい VPC を作成します。

d. [FIPS Enabled] (FIPS 対応) で [FIPS Enabled endpoint] (FIPS 対応エンドポイント) チェックボックスをオフのままにします。

> **Note**
>
> AS2 プロトコルでは、FIPS 対応のエンドポイントはサポートされません。

e. [次へ] をクリックします。

6. 「ドメインの選択」ページで「Amazon S3」を選択し、選択したプロトコルを使用してファイルをオブジェクトとして保存およびアクセスします。

   [次へ] をクリックします。

7. 「追加詳細の設定」ページで、必要な設定を選択します。

> **Note**
>
> AS2 と一緒に他のプロトコルを設定する場合は、追加の詳細設定がすべて適用されます。ただし、AS2 プロトコルでは、適用される設定はCloudWatch ロギングセクションとタグセクションの設定のみです。
> CloudWatch ログ記録ロールの設定はオプションですが、メッセージのステータスを確認し、設定の問題をトラブルシューティングできるように設定することを強くお勧めします。

8. 「レビューと作成」ページで、選択内容が正しいことを確認してください。

   - 設定を編集する場合は、変更するステップの横にある「編集」を選択します。

     > **Note**
     >
     > ステップを編集する場合は、編集の対象として選択したステップの後で各ステップの確認をお勧めします。

   - 変更がない場合、[Create server] (サーバーの作成) を選択してサーバーを作成します。[Servers] (サーバー) ページに誘導され、次に示すように新しいサーバーの一覧が表示されます。

     新しいサーバーのステータスが「オンライン」に変わるまで、数分かかることがあります。その時点で、サーバーはユーザーのためにファイルオペレーションを実行できます。

テンプレートを使用して Transfer Family AS2 スタックのデモを作成する

AS2-enabled Transfer Family サーバーをすばやく作成するために、自己完結型の AWS CloudFormation テンプレートを提供しています。このテンプレートは、パブリック Amazon VPC エンドポイント、証明書、ローカルプロファイルとパートナープロファイル、契約、コネクタを使用してサーバーを設定します。

このテンプレートを使用する前に、次の点に注意してください。

- このテンプレートからスタックを作成した場合、使用した AWS リソースに対する料金が発生します。

- テンプレートは複数の証明書を作成し、安全に保存 AWS Secrets Manager するために に配置します。このサービスの使用には料金がかかるため、これらの証明書は必要に応じて Secrets Manager から削除できます。Secrets Manager でこれらの証明書を削除しても、Transfer Family サーバーからは削除されません。したがって、デモスタックの機能には影響しません。ただし、本番環境の AS2 サーバーで使用する証明書については、Secrets Manager を使用して、保存されている証明書を管理し、定期的にローテーションすることをおすすめします。
- テンプレートはベースとしてのみ使用し、主にデモンストレーションの目的で使用することが推奨されます。このデモスタックを本番環境で使用する場合は、テンプレートの YAML コードを変更して、より堅牢なスタックを作成することをお勧めします。例えば、本番稼働レベルの証明書を作成し、本番環境で使用できる AWS Lambda 関数を作成します。

CloudFormation テンプレートから AS2-enabled Transfer Family サーバーを作成するには

1. https://console.aws.amazon.com/cloudformation で AWS CloudFormation コンソールを開きます。
2. 左のナビゲーションペインで [Stacks] (スタック) をクリックします。
3. [スタックの作成] を選択し、[With new resources (standard) 新しいリソースを使用 (標準)] を選択します。
4. [前提条件 - テンプレートを準備する] セクションで [テンプレートの準備完了] を選択します。
5. このリンクで AS2 デモテンプレートをコピーして [Amazon S3 URL] フィールドに貼り付けます。
6. [次へ] をクリックします。
7. [スタックの詳細を指定] ページで、スタックに名前を付け、次のパラメータを指定します。
   - [AS2] で、[ローカル AS2 ID] と [パートナー AS2 ID] の値を入力するか、デフォルトの `local` と `partner` をそれぞれ受け入れます。
   - [ネットワーク] で、[セキュリティグループの進入 CIDR IP] の値を入力するか、デフォルトの `0.0.0.0/0` を受け入れます。

     > **Note**
     >
     > この値は CIDR 形式で、AS2 サーバーへの受信トラフィックに許可される IP アドレスを指定します。デフォルト値の `0.0.0.0/0` では、すべての IP アドレスを許可します。

- [全般] で、[プレフィックス] の値を入力するか、デフォルトの `transfer-as2` を受け入れます。このプレフィックスは、スタックによって作成されるリソース名の前に置かれます。例えば、デフォルトのプレフィックスを使用する場合、Amazon S3 バケットには `transfer-as2-`*`TransferS3BucketName`* という名前が付けられます。

8. [次へ] をクリックします。[スタックオプションの設定] ページで、[次へ] を選択します。
9. 作成しようとするスタックの詳細を確認してから [スタックの作成] を選択します。

> **Note**
>
> ページの下部にある 機能 で、 が AWS Identity and Access Management (IAM) リソースを作成する AWS CloudFormation 可能性があることを確認する必要があります。

スタックが作成されたら、 AWS Command Line Interface () を使用して、パートナーサーバーからローカル Transfer Family サーバーにテスト AS2 メッセージを送信できますAWS CLI。テストメッセージを送信するためのサンプル AWS CLI コマンドが、スタック内の他のすべてのリソースとともに作成されます。

このサンプルコマンドを使用するには、スタックの出力タブに移動し、TransferExampleAs2Command をコピーします。その後、 AWS CLIを使用してコマンドを実行できます。をまだインストールしていない場合は AWS CLI、「 AWS Command Line Interface ユーザーガイド」の「 の最新バージョンのインストールまたは更新 AWS CLI」を参照してください。

このサンプルコマンドの形式は次のとおりです。

```
aws s3api put-object --bucket TransferS3BucketName --key test.txt && aws transfer
 start-file-transfer --region aws-region --connector-id TransferConnectorId --send-
file-paths /TransferS3BucketName/test.txt
```

> **Note**
>
> このコマンドのバージョンには、スタック内の *`TransferS3BucketName`* および *`TransferConnectorId`* リソースの実際の値が含まれています。

このサンプルコマンドは、&& 文字列を使用して連結された 2 つのコマンドで構成されています。

最初のコマンドは、バケットに新しい空のテキストファイルを作成します。

```
aws s3api put-object --bucket TransferS3BucketName --key test.txt
```

次に、2 番目のコマンドはコネクタを使用して、パートナープロファイルからローカルプロファイルにファイルを送信します。Transfer Family サーバーには、ローカルプロファイルがパートナープロファイルからのメッセージを受け入れることを許可する契約が設定されています。

```
aws transfer start-file-transfer --region aws-region --connector-id TransferConnectorId
 --send-file-paths /TransferS3BucketName/test.txt
```

コマンドを実行したら、Amazon S3 バケット (*TransferS3BucketName*) に移動して内容を表示できます。コマンドが成功した場合は、バケットには次のオブジェクトが表示されます。

- `processed/` - このフォルダには、転送されたファイルと MDN レスポンスを記述した JSON ファイルが含まれています。
- `processing/` - このフォルダには、処理中のファイルが一時的に保存されますが、転送が完了すると、このフォルダは空になるはずです。
- *`server-id`*`/` - このフォルダは、Transfer Family サーバー ID に基づいて名前が付けられています。これには `from-`*`partner`* (このフォルダにはパートナーの AS2 ID に基づいて動的に名前が付けられます) が含まれており、フォルダ自体に `failed/`、`processed/`、`processing/` フォルダが含まれています。*`/server-id`*`/from-`*`partner`*`/processed/` フォルダには、転送されたテキストファイルのコピーと、対応する JSON ファイルと MDN ファイルが格納されます。
- `test.txt` - このオブジェクトは転送された (空の) ファイルです。

AS2 の設定とクォータ

このトピックでは、適用性ステートメント 2 (AS2) プロトコルを使用する転送でサポートされる設定、機能、能力について説明します。これには、承認された暗号やダイジェストも含まれます。このセクションでは、AS2 転送の制限と既知の問題についても説明します。

トピック

- サポートされる AS2 設計
- AS2 クォータと制限事項

サポートされる AS2 設計

署名、暗号化、圧縮、MDN

インバウンドとアウトバウンドのどちらの転送でも、以下の項目は必須またはオプションです。

- 暗号化 - 必須 (現在サポートされている唯一のトランスポート方法である HTTP トランスポート用)。暗号化されていないメッセージは、Application Load Balancer (ALB) などの TLS 終了プロキシによって転送され、`X-Forwarded-Proto: https` ヘッダーが存在する場合にのみ受け入れられます。
- 署名 - オプション
- 圧縮 - オプション (現在サポートされている圧縮アルゴリズムは ZLIB だけです)
- メッセージ処理通知 (MDN) - オプション

暗号

次の暗号は、インバウンド転送とアウトバウンド転送の両方でサポートされています。

- AES128_CBC
- AES192_CBC
- AES256_CBC
- 3DES (下位互換性のみ)

ダイジェスト

以下のダイジェストをサポートしています。

- インバウンド署名と MDN - SHA1、SHA256、SHA384、SHA512
- アウトバウンド署名と MDN - SHA1、SHA256、SHA384、SHA512

MDN

MDN レスポンスでは、以下のように特定のタイプがサポートされています。

- インバウンド転送 - 同期と非同期
- アウトバウンド転送 - 同期のみ
- Simple Mail Transfer Protocol (SMTP) (E メール MDN) - サポートされていません

トランスポート

- インバウンド転送 - 現在サポートされているトランスポートは HTTP のみで、明示的に指定する必要があります。

> **Note**
>
> インバウンド転送に HTTPS を使用する必要がある場合は、Application Load Balancer または Network Load Balancer で TLS を終了できます。これについては、「HTTPS 経由で AS2 メッセージを受信」で説明しています。

- アウトバウンド転送 - HTTP URL を指定する場合は、暗号化アルゴリズムも指定する必要があります。HTTPS URL を指定する場合、暗号化アルゴリズムに [なし] を指定することもできます。

AS2 クォータと制限事項

このセクションでは、AS2 のクォータと制限について説明します。

トピック

AS2 クォータ

AS2 ファイル転送には次のクォータが設定されています。調整可能なクォータの増額をリクエストするには、AWS 全般のリファレンスの「AWS のサービス のクォータ」を参照してください。

AS2 クォータ

名前	デフォルト	引き上げ可能
1 秒あたりに受信されるインバウンドファイルの最大数	100	いいえ
1 秒あたりに送信されるアウトバウンドファイルの最大数	100	いいえ
同時インバウンドファイルの最大数	400	いいえ

名前	デフォルト	引き上げ可能
同時アウトバウンドファイルの最大数	400	いいえ
インバウンドファイルの最大サイズ (非圧縮)	1 GB	いいえ
アウトバウンドファイルの最大サイズ (非圧縮)	1 GB	いいえ
アウトバウンドリクエストあたりのファイルの最大数	10	いいえ
1 秒あたりのアウトバウンドリクエストの最大数	100	いいえ
1 秒あたりのインバウンドリクエストの最大数	100	いいえ
アカウントあたりの最大アウトバウンド帯域幅 (アウトバウンド SFTP リクエストと AS2 リクエストの両方がこの値に影響します)	50 MB/秒	いいえ
サーバーあたりの契約の最大数	100	はい
アカウントあたりのコネクタの最大数 (SFTP コネクタと AS2 コネクタの両方がこの制限に影響します)	100	はい
パートナープロファイルあたりの証明書の最大数	10	いいえ
アカウントあたりの証明書の最大数	1,000	はい

名前	デフォルト	引き上げ可能
アカウントあたりのパートナープロファイルの最大数	1,000	はい

シークレットの処理クォータ

AWS Transfer Family は、基本認証を使用している AS2 のお客様 AWS Secrets Manager に代わってを呼び出します。さらに、Secrets Manager は を呼び出します AWS KMS。

> **Note**
>
> これらのクォータは、Transfer Family のシークレットの使用に固有のものではなく、 内のすべてのサービス間で共有されます AWS アカウント。

Secrets Manager の場合`GetSecretValue`、適用されるクォータは、クォータ で説明されているように、 DescribeSecret および GetSecretValue API リクエスト の合計レートです。 AWS Secrets Manager

Secrets Manager **`GetSecretValue`**

名前	値	説明
DescribeSecret および GetSecretValue API リクエストの合計レート	サポートされている各リージョン: 10,000/秒	DescribeSecret と GetSecretValue API リクエストの合計における 1 秒あたりの最大トランザクション数。

の場合 AWS KMS、 には次のクォータが適用されます`Decrypt`。詳細については、「各 AWS KMS API オペレーションのリクエストクォータ」を参照してください。

AWS KMS **`Decrypt`**

クォータ名	デフォルト値 (1 秒あたりのリクエスト)
暗号化オペレーション (対称) リクエストレート	これらの共有クォータは、 AWS リージョン およびリクエストで使用される AWS KMS キー

クォータ名	デフォルト値 (1 秒あたりのリクエスト)

	のタイプによって異なります。各クォータは個別に計算されます。
	• 5,500 (共有)
	• 以下のリージョンでは 10,000 (共有):
	  • 米国東部 (オハイオ)、us-east-2
	  • アジアパシフィック (シンガポール)、ap-southeast-1
	  • アジアパシフィック (シドニー)、ap-southeast-2
	  • アジアパシフィック (東京)、ap-norteast-1
	  • ヨーロッパ (フランクフルト)、eu-central-1
	  • ヨーロッパ (ロンドン)、eu–west-2
	• 以下のリージョンでは 50,000 (共有):
	  • 米国東部 (バージニア北部)、us-east-1
	  • 米国西部 (オレゴン)、us-west-2
	  • ヨーロッパ (アイルランド)、eu-west-1

カスタムキーストアのリクエストクォータ **Note** このクォータは、外部キーストアを使用している場合にのみ適用されます。	カスタムキーストアのリクエストクォータは、カスタムキーストアごとに個別に計算されます。 • AWS CloudHSM キーストアごとに 1,800 (共有) • 外部キーストアごとに 1,800 (共有)

既知の制限事項

- サーバー側の TCP キープアライブはサポートされていません。クライアントがキープアライブパケットを送信しない限り、350 秒間何も操作しないと接続はタイムアウトします。
- アクティブな契約がサービスによって承諾され、Amazon CloudWatch ログに表示されるには、メッセージに有効な AS2 ヘッダーが含まれている必要があります。

- AS2 AWS Transfer Family の からメッセージを受信するサーバーは、RFC 6211 で定義されているように、メッセージ署名を検証するための暗号化メッセージ構文 (CMS) アルゴリズム保護属性をサポートしている必要があります。この属性は、一部の古い IBM Sterling 製品ではサポートされていません。
- メッセージ ID が重複していると、「processed/Warning: duplicate-document」メッセージが表示されます。
- AS2 証明書のキー長は 2048 ビット以上、最大で 4096 ビットでなければなりません。
- AS2 メッセージまたは非同期 MDN を取引相手の HTTPS エンドポイントに送信する場合、メッセージまたは MDN は、信頼された公的認証機関 (CA) によって署名された有効な SSL 証明書を使用する必要があります。現在、自己署名証明書はアウトバウンド転送でのみサポートされています。
- エンドポイントは TLS バージョン 1.2 プロトコルと、セキュリティポリシー (AWS Transfer Family サーバーのセキュリティポリシー を参照) で許可されている暗号化アルゴリズムをサポートしている必要があります。
- AS2 バージョン 1.2 の複数の添付ファイルと証明書交換メッセージ (CEM) は、現在サポートされていません。
- Basic 認証は現在、アウトバウンドメッセージでのみサポートされています。

AS2 の特徴と機能

次の表は、AS2 を使用する Transfer Family リソースで使用できる機能と能力の一覧です。

AS2 の機能

Transfer Family は AS2 向けに以下の機能を提供します。

機能	でサポート AWS Transfer Family
Drummond 認定	はい
AWS CloudFormation のサポート	はい
Amazon CloudWatch メトリクス	はい
SHA-2 暗号化アルゴリズム	はい
Amazon S3 のサポート	はい

機能	でサポート AWS Transfer Family
Amazon EFS のサポート	いいえ
スケジュールされたメッセージ	はい [1]
AWS Transfer Family マネージドワークフロー	いいえ
Certificate Exchange Messaging (CEM)	いいえ
相互 TLS (mTLS)	いいえ
自己署名証明書のサポート	はい

1. Amazon を使用して AWS Lambda 関数をスケジュール EventBridgeすることで利用可能なアウトバウンドのスケジュールされたメッセージ

AS2 の送受信機能

次の表は、 AWS Transfer Family AS2 の送受信機能のリストです。

機能	インバウンド: サーバーによる受信	アウトバウンド: コネクタを使用した送信
TLS 暗号化トランスポート (HTTPS)	はい [1]	はい
TLS 以外のトランスポート (HTTP)	はい	はい [2]
同期 MDN	はい	はい
メッセージ圧縮	はい	はい
非同期 MDN	はい	いいえ
静的 IP アドレス	はい	はい
自分の IP アドレスを持ち込む	はい	いいえ

機能	インバウンド: サーバーによる受信	アウトバウンド: コネクタを使用した送信
複数のファイルアタッチメント	いいえ	いいえ
基本認証	いいえ	はい
AS2 再起動	該当しない	いいえ
AS2 の信頼性	いいえ	いいえ
メッセージあたりのカスタム件名	該当しない	いいえ

1. Network Load Balancer (NLB) で使用できるインバウンド TLS 暗号化トランスポート

2. 暗号化が有効な場合にのみアウトバウンドの非 TLS トランスポートが使用可能

AS2 コネクタを設定する

コネクタの目的は、「アウトバウンド」転送について取引パートナー間の関係を確立することです。つまり、AS2ファイルをTransfer Familyサーバーからパートナーが所有する外部の宛先に送信します。コネクタには、ローカルパーティ、リモートパートナー、およびそれらの証明書を (ローカルプロファイルとパートナープロファイルを作成して) 指定します。

コネクタを設定したら、取引相手に情報を転送できます。各 AS2 サーバーには 3 つの静的 IP アドレスが割り当てられます。AS2 コネクタは、これらの IP アドレスを使用して、AS2 経由で取引相手に非同期 MDNs を送信します。 AS2

> **Note**
>
> 取引相手が受信するメッセージサイズは、Amazon S3 のオブジェクトサイズと一致しません。この不一致は、AS2 メッセージが送信前にファイルをエンベロープに包んでしまうために発生します。そのため、ファイルを圧縮して送信しても、ファイルサイズが大きくなる可能性があります。そのため、取引相手の最大ファイルサイズが、送信するファイルのサイズよりも大きいことを確認してください。

AS2 コネクタを作成する

この手順では、 AWS Transfer Family コンソールを使用して AS2 コネクタを作成する方法について説明します。 AWS CLI 代わりに を使用する場合は、「」を参照してくださいthe section called “ステップ 6: 自分とパートナーとの間でコネクタを作成する”。

AS2 コネクターを作成するには

1. https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開きます。
2. 左のナビゲーションペインで[コネクタ] を選択し、[コネクタの作成]を選択します。
3. [コネクタ設定] セクションで、以下の情報を指定します。

   - [URL] — アウトバウンド接続の URL を入力します。
   - アクセスロール — 使用する (IAM) ロールの Amazon リソースネーム AWS Identity and Access Management (ARN) を選択します。`StartFileTransfer` リクエストで使用されるファイルロケーションの親ディレクトリに対して、このロールが読み取りと書き込みのアクセスを提供することを確認します。さらに、`StartFileTransfer` で送信するファイルの親ディレクトリに対する読み取り/書き込みアクセスを提供していることを確認してください。

     > **Note**
     >
     > コネクタに Basic 認証を使用している場合、アクセスロールにはシークレットの `secretsmanager:GetSecretValue` 権限が必要です。シークレットが AWS マネージドキー の ではなくカスタマーマネージドキーを使用して暗号化されている場合 AWS Secrets Manager、ロールにはそのキーに対する アクセス`kms:Decrypt`許可も必要です。シークレットにプレフィックス `aws/transfer/` を付けた名前を付けると、「シークレットを作成する権限の例」に示すように、ワイルドカード文字 (*) を使用して必要な権限を追加できます。

   - ログ記録ロール (オプション) — CloudWatch ログにイベントをプッシュするために使用するコネクタの IAM ロールを選択します。
4. [AS2 設定] セクションで、ローカルプロファイルとパートナープロファイル、暗号化アルゴリズムと署名アルゴリズム、転送された情報を圧縮するかどうかを選択します。次の点に注意してください。

   - 暗号化アルゴリズムでは、それを必要とするレガシークライアントをサポートする必要がある`DES_EDE3_CBC`場合を除き、 を選択しないでください。これは弱い暗号化アルゴリズムです。

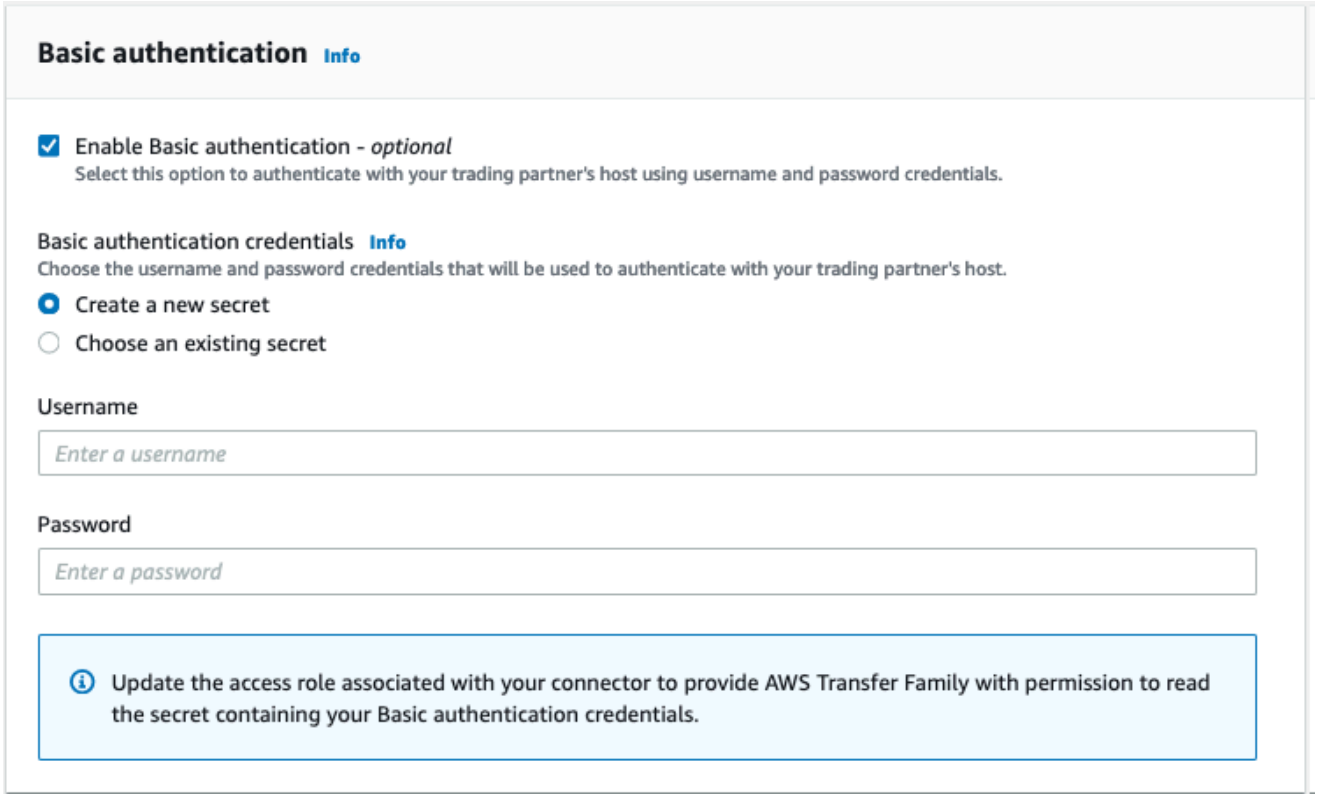
- [件名] は、コネクタで送信される AS2 `subject` メッセージの HTTP ヘッダー属性として使用されます。
- 暗号化アルゴリズムなしでコネクタを作成する場合は、プロトコルHTTPSとして を指定する必要があります。

5. [MDN 設定] セクションで、以下の情報を指定します。

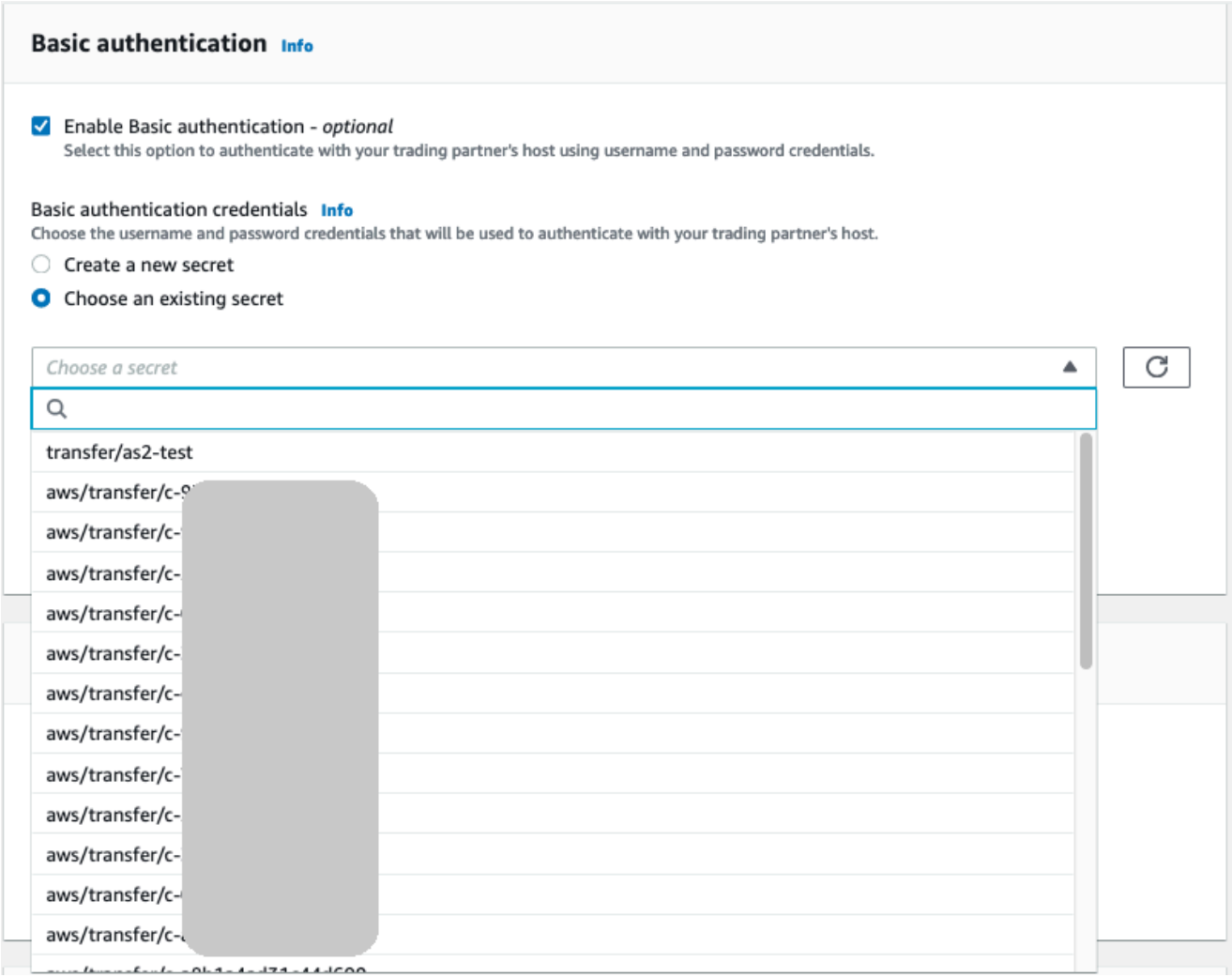
- [MDN をリクエスト] — AS2 経由でメッセージを正常に受信した後に、取引相手に MDN の送信を要求することができます。
- [署名済み MDN] — MDN への署名を要求するオプションがあります。このオプションは、[MDN をリクエスト] を選択した場合にのみ使用できます。

6. [基本認証] セクションで、以下の情報を指定します。

- サインオン認証情報を送信メッセージと一緒に送信するには、[基本認証を有効にする] を選択します。送信メッセージで認証情報を送信したくない場合は、[基本認証を有効にする] をオフのままにします。
- 認証を使用している場合は、シークレットを選択するか作成してください。
  - 新しいシークレットを作成するには、[新しいシークレットを作成] を選択し、ユーザー名とパスワードを入力します。これらの認証情報は、パートナーのエンドポイントに接続するユーザーと一致する必要があります。



- 既存のシークレットを使用するには、「既存シークレットの選択」を選択し、ドロップダウンメニューからシークレットを選択します。Secrets Manager で正しくフォーマットされたシークレットを作成する方法については、AS2 コネクタの基本認証を有効にします。 を参照してください。



7. すべての設定を確認したら、[コネクタを作成] を選択してコネクタを作成します。

新しいコネクタの ID がリストに追加された[コネクタ] ページが表示されます。コネクタの詳細を表示するには、AS2 コネクターの詳細を表示 を参照してください。

AS2 コネクタアルゴリズム

AS2 コネクタを作成すると、次のセキュリティアルゴリズムがコネクタにアタッチされます。

タイプ	アルゴリズム
TLS 暗号	TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384 TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384

AS2 コネクタの基本認証

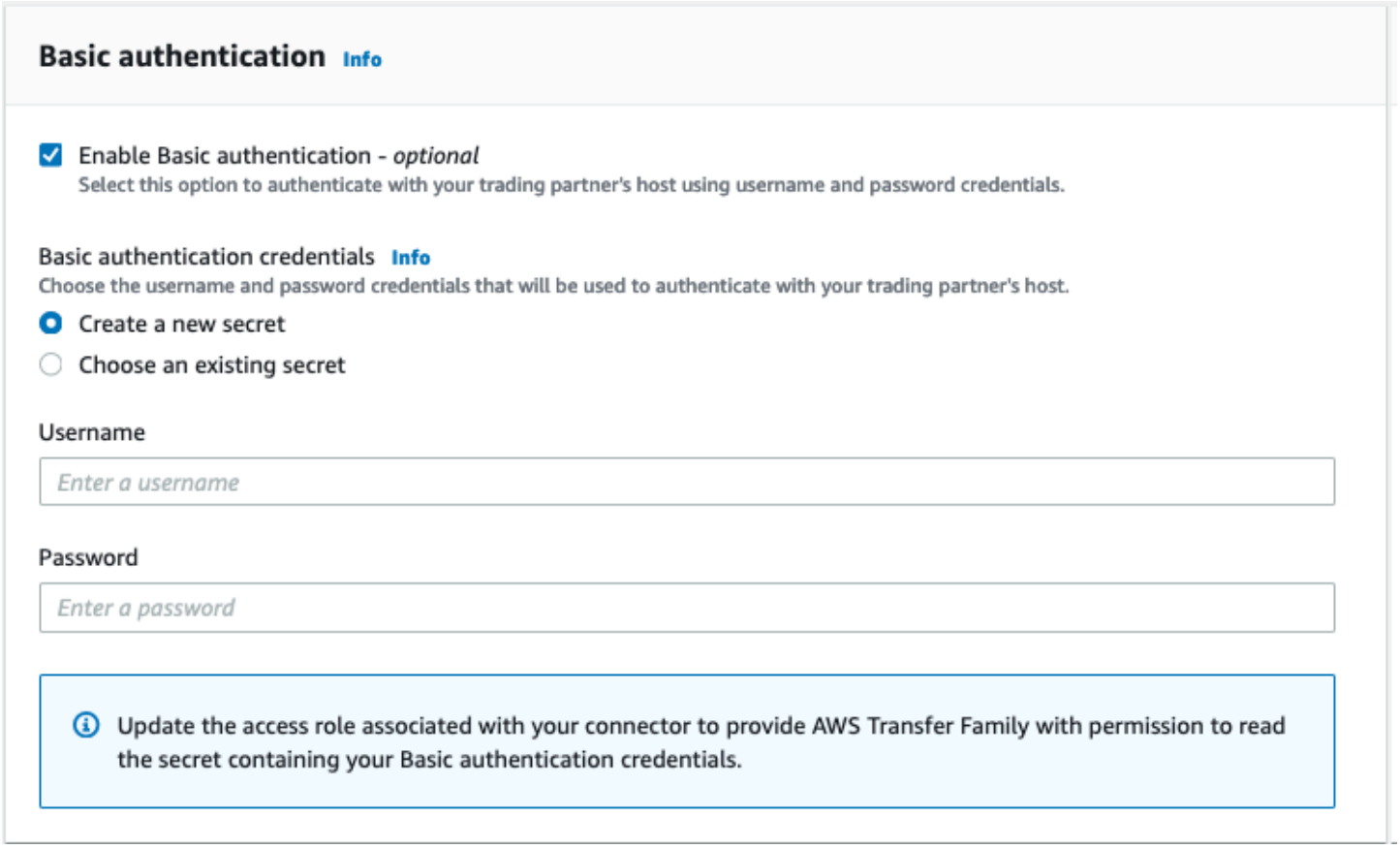
AS2プロトコルを使用するTransfer Family サーバーを作成または更新すると、送信メッセージに基本認証を追加できます。これを行うには、コネクタに認証情報を追加します。

> **Note**
>
> 基本認証は HTTPS を使用している場合にのみ使用できます。

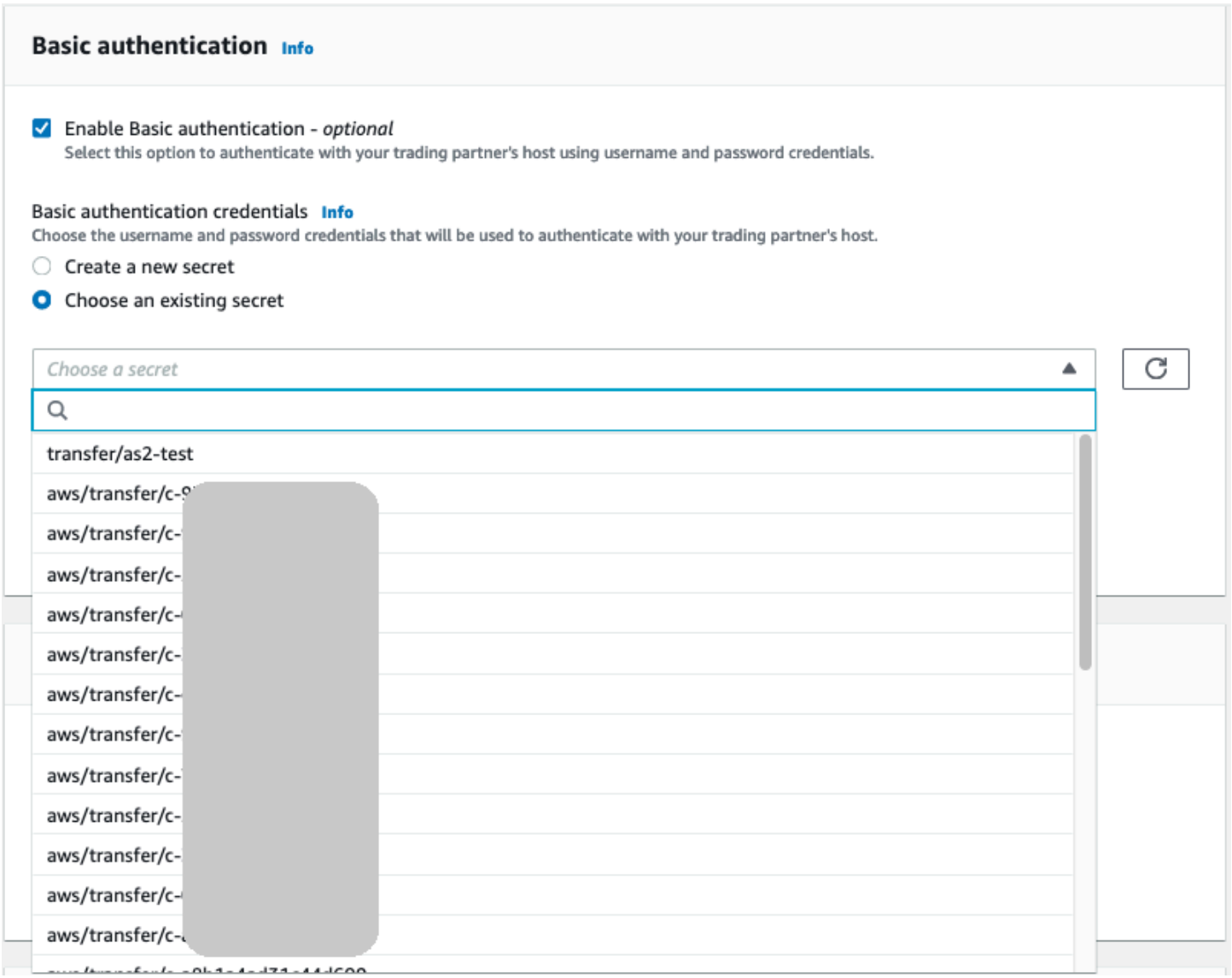
コネクタに認証を使用するには、「基本認証」セクションの「基本認証を有効にする」を選択します。基本認証を有効にすると、新しいシークレットを作成するか、既存のシークレットを使用するか

を選択できます。いずれの場合も、シークレット内の認証情報は、このコネクタを使用する送信メッセージとともに送信されます。認証情報は、取引相手のリモートエンドポイントに接続しようとするユーザーと一致する必要があります。

次のスクリーンショットは、「基本認証を有効にする」と「新しいシークレットを作成する」が選択されていることを示しています。これらの選択をした後、秘密のためのユーザー名とパスワードを入力できます。



次のスクリーンショットは、「基本認証を有効にする」が選択され、「既存のシークレットを選択する」が選択されていることを示しています。シークレットは、AS2 コネクタの基本認証を有効にします。 で説明されているように、正しい形式でなければなりません。



AS2 コネクタの基本認証を有効にします。

AS2 コネクタの基本認証を有効にすると、Transfer Family コンソールで新しいシークレットを作成することも、 AWS Secrets Managerで作成したシークレットを使用することもできます。いずれの場合も、シークレットは Secrets Manager に保存されます。

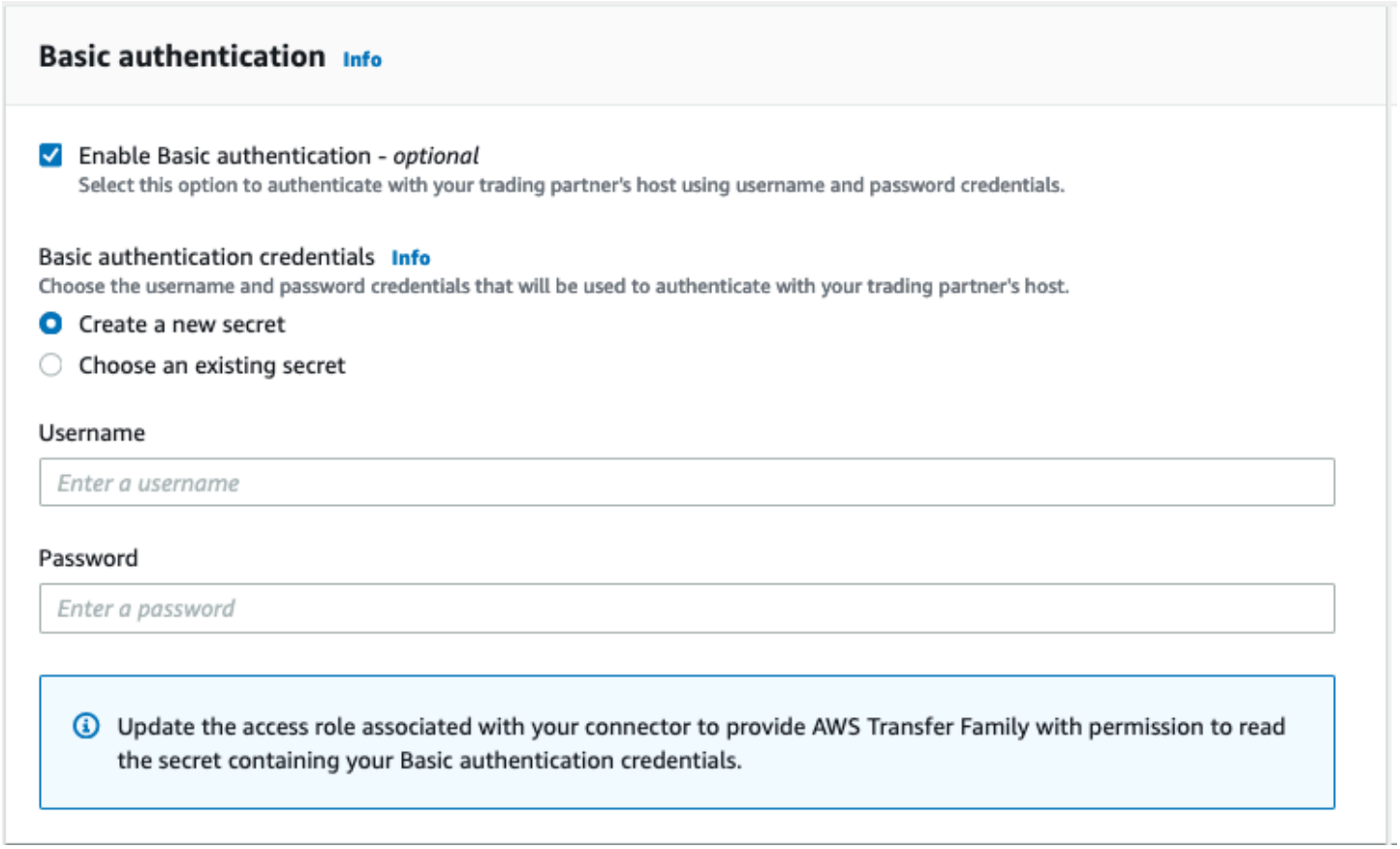
トピック

- コンソールで新しいシークレットを作成する
- 既存の シークレットを使用
- でシークレットを作成する AWS Secrets Manager

コンソールで新しいシークレットを作成する

コンソールでコネクタを作成する場合、新しいシークレットを作成できます。

新しいシークレットを作成するには、[新しいシークレットを作成] を選択し、ユーザー名とパスワードを入力します。これらの認証情報は、パートナーのエンドポイントに接続するユーザーと一致する必要があります。

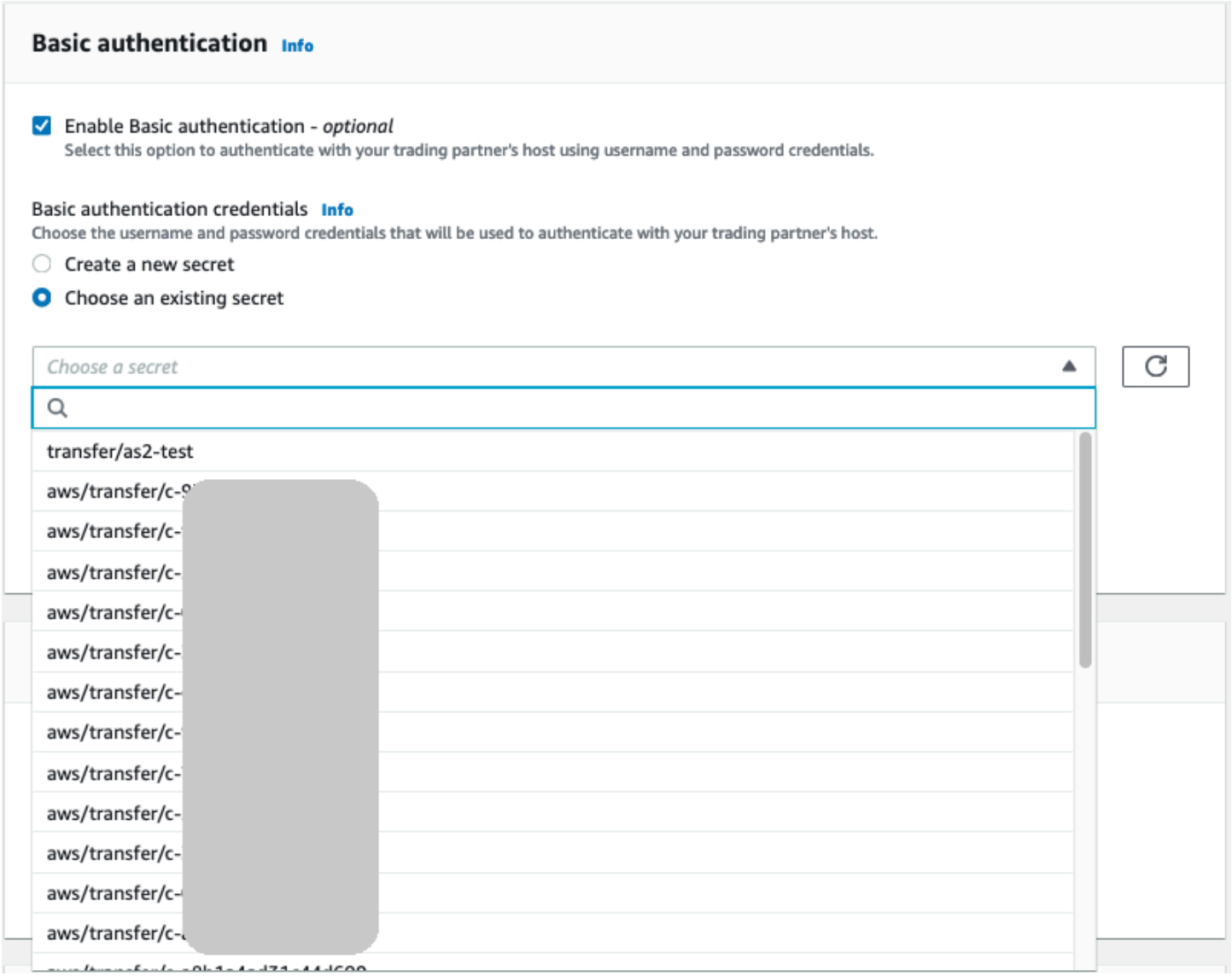


> **Note**
>
> コンソールで新しいシークレットを作成する場合、シークレット名は **/aws/transfer/*connector-id*** という命名規則に従います。ここで、「*connector-id*」は作成するコネクタのIDです。 AWS Secrets Managerでシークレットを探す際には、この点を考慮してください。

既存の シークレットを使用

コンソールでコネクタを作成する場合、既存のシークレットを指定できます。

既存のシークレットを使用するには、「既存シークレットの選択」を選択し、ドロップダウンメニューからシークレットを選択します。Secrets Manager で正しくフォーマットされたシークレットを作成する方法については、でシークレットを作成する AWS Secrets Manager を参照してください。



でシークレットを作成する AWS Secrets Manager

次の手順では、AS2 コネクタで使用する適切なシークレットを作成する方法について説明します。

> **Note**
>
> 基本認証は HTTPS を使用している場合にのみ使用できます。

AS2 基本認証用のSecrets Manager にユーザー認証情報を保存するには

1. にサインイン AWS Management Console し、https://console.aws.amazon.com/secretsmanager/ で AWS Secrets Manager コンソールを開きます。
2. 左側のナビゲーションペインで [サーバー] を選択します。
3. [シークレット]ページで、[新しいシークレットの保存]を選択します。
4. [シークレットタイプの選択] ページの[シークレットタイプ] で[その他のシークレットタイプ] を選択します。
5. [キー/値のペア] セクションで、[キー/値] タブを選択します。
   - キー — **Username**と入力します。
   - [値] — パートナーのサーバーへの接続を許可されているユーザーの名前を入力します。
6. パスワードを入力する場合は、[行を追加] を選択し、[キー/値のペア] セクションで[Key/Value] タブを選択します。

   [行を追加] を選択し、[キー/値のペア] セクションで[キー/値] タブを選択します。
   - キー — **Password**と入力します。
   - [値] ー ユーザーのパスワードを入力します。
7. プライベートキーを指定する場合は、[行を追加] を選択し、「キー/値のペア」セクションで「Key/Value」タブを選択します。
   - キー — **PrivateKey**と入力します。
   - 「値」 — ユーザーの秘密鍵を入力します。この値は OpenSSH 形式で保存する必要があり、リモートサーバーでこのユーザー用に保存されている公開鍵に対応している必要があります。
8. [次へ] をクリックします。
9. [シークレットの設定] ページで、シークレットの名前と説明を入力します。名前には **aws/transfer/** というプレフィックスを使用することをお勧めします。例えば、シークレットを **aws/transfer/connector-1** と名付けることができます。
10. [次へ] を選択し、[ローテーションの設定] ページのデフォルトを受け入れます。次いで、[次へ] を選択します。
11. [レビュー] ページで[ストア] を選択し、シークレットを作成して保存します。

シークレットを作成したら、コネクタの作成時に選択できます (AS2 コネクタを設定する を参照)。基本認証を有効にするステップで、使用可能なシークレットのドロップダウンリストからシークレットを選択します。

AS2 コネクターの詳細を表示

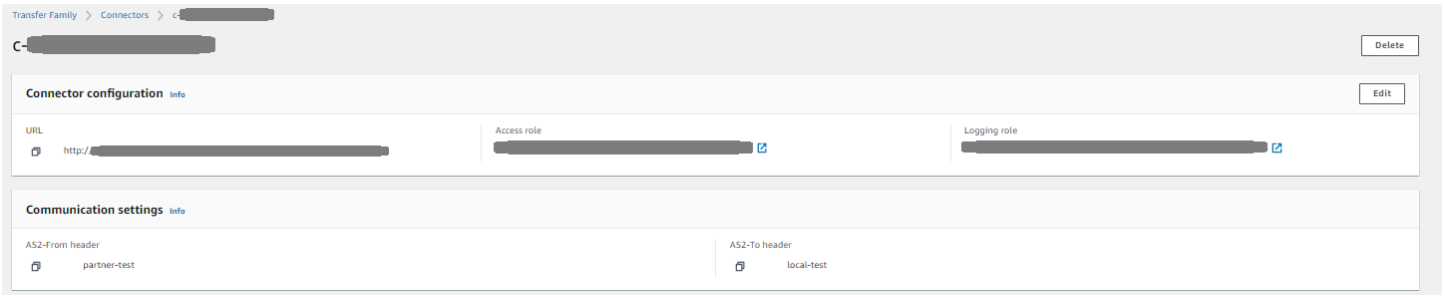
AS2 AWS Transfer Family コネクタの詳細とプロパティのリストは、 AWS Transfer Family コンソールで確認できます。AS2 コネクタのプロパティには、URL、ロール、プロファイル、mDNS、タグ、監視指標が含まれます。

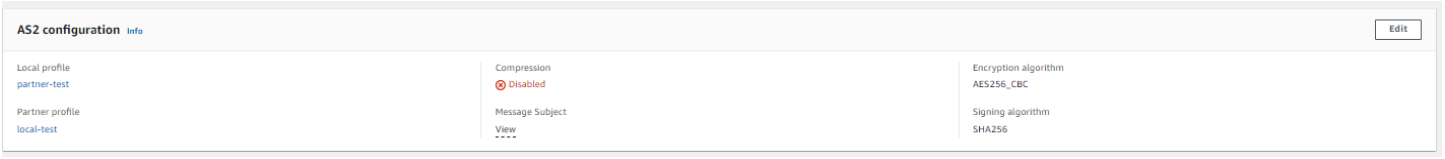
これはコネクタの詳細を表示する手順です。

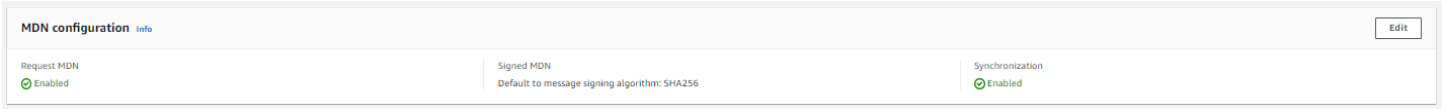
コネクタの詳細を表示するには

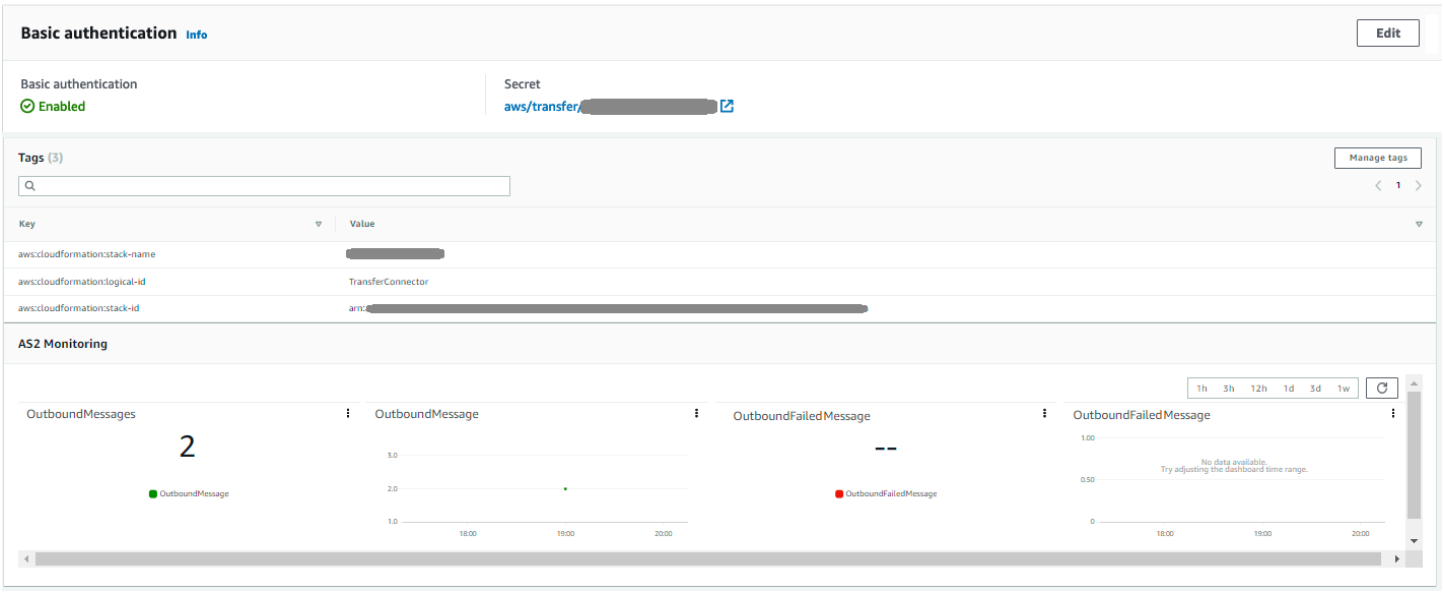
1. https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開きます。
2. 左側のナビゲーションペインで、[Connectors (コネクタ)] を選択します。
3. [コネクタ ID] 列の識別子を選択すると、選択したコネクタの詳細ページが表示されます。

AS2 コネクタのプロパティは、コネクタの詳細ページで 「編集」を選択して変更できます。









> **Note**
>
> この情報の多くは、形式は異なりますが、次の AWS Command Line Interface (AWS CLI コマンド) を実行することで取得できます。
>
> ```
> aws transfer describe-connector --connector-id your-connector-id
> ```
>
> 詳細については、API リファレンスのDescribeConnectorを参照してください。

AS2 パートナーを管理する

このトピックでは、AS2 証明書、プロファイル、および契約を管理する方法について説明します。

AS2 証明書のインポート

Transfer Family AS2 プロセスでは、転送された情報の暗号化と署名の両方に証明書キーを使用します。パートナーは両方の目的で同じキーを使用することも、それぞれに別のキーを使用することもできます。災害やセキュリティ侵害が発生した場合にデータを復号化できるように、信頼できる第三者が共通の暗号鍵をエスクローに保管している場合は、別々の署名鍵を用意することをおすすめします。個別の署名鍵 (エスクローは行わない) を使用することで、電子署名の否認防止機能を損なうことがなくなります。

> **Note**
>
> AS2 証明書のキー長は 2048 ビット以上、最大で 4096 ビットでなければなりません。

次の点では、この処理中に AS2 証明書がどのように使用されるかを詳しく説明します。

- インバウンド AS2
  - 取引相手は署名証明書の公開鍵を送信し、この鍵はパートナープロファイルにインポートされます。
  - ローカルパーティは、暗号化証明書と署名証明書用の公開鍵を送信します。その後、パートナーは 1 つまたは複数の秘密鍵をインポートします。ローカルパーティは、署名用と暗号化用に別々の証明書キーを送信することも、両方の目的で同じキーを使用することもできます。
- アウトバウンド AS2
  - パートナーは暗号化証明書の公開鍵を送信し、この鍵はパートナープロファイルにインポートされます。
  - ローカルパーティは証明書の公開鍵を署名用に送信し、証明書の秘密鍵をインポートして署名します。
  - HTTPS を使用している場合は、自己署名 Transport Layer Security (TLS) 証明書をインポートできます。

証明書を作成する方法については、the section called “ステップ 1: AS2 の証明書を作成する” を参照してください。

この手順では、Transfer Family コンソールを使用して証明書をインポートする方法について説明します。 AWS CLI 代わりに を使用する場合は、「」を参照してくださいthe section called “ステップ 3: 証明書を Transfer Family 証明書リソースとしてインポートする”。

AS2 対応証明書を指定するには

1. https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開きます。
2. 左側のナビゲーションペインで、「AS2 トレーディングパートナー」の下にある「証明書」を選択します。
3. [証明書のインポート] を選択します。
4. 「証明書の説明」セクションに、簡単に識別できる証明書の名前を入力します。説明から証明書の目的を特定できることを確認します。さらに、証明書の役割を選択します。

5. 「証明書コンテンツ」セクションで、取引相手からの公開証明書、またはローカル証明書の公開鍵と秘密鍵を入力します。
6. 「証明書の使用方法」セクションで、この証明書の目的を選択します。暗号化、署名、またはその両方に使用できます。

> **Note**
>
> 使用方法として「暗号化と署名」を選択した場合、TTransfer Family ilyは2つの同一の証明書 (それぞれに独自のIDを持つ) を作成します。1つは `ENCRYPTION` の使用値がで、もう1つは `SIGNING` の使用値です。

7. 「証明書の内容」セクションに適切な詳細を入力します。

   - 「自己署名証明書」を選択した場合は、証明書チェーンを提供しません。
   - 証明書の内容を貼り付けます。
   - 証明書が自己署名証明書でない場合は、証明書チェーンを提供します。
   - この証明書がローカル証明書の場合は、秘密鍵を貼り付けます。
8. 「証明書のインポート」を選択して処理を完了し、インポートした証明書の詳細を保存します。

> **Note**
>
> TLS 証明書は、パートナーのパブリック証明書としてのみインポートできます。パートナーからパブリック証明書を選択し、使用のために Transport Layer Security (TLS) を選択すると、警告が表示されます。また、TLS 証明書は自己署名する必要があります (つまり、TLS 証明書をインポートするには自己署名証明書を選択する必要があります)。

AS2 証明書のローテーション

多くの場合、証明書は 6 か月から 1 年間有効です。プロファイルを設定して、長期間保存したい場合があります。これを容易にするために、Transfer Familyは証明書のローテーションを提供しています。プロファイルには複数の証明書を指定できるため、プロファイルを複数年間使用し続けることができます。Transfer Familyは、署名 (オプション) と暗号化 (必須) に証明書を使用します。希望すれば、両方の目的で単一の証明書を指定できます。

証明書のローテーションは、期限切れの古い証明書を新しい証明書に置き換えるプロセスです。契約のパートナーがアウトバウンド転送用の新しい証明書をまだ設定していない場合や、新しい証明書が

使用されている可能性のある期間に、古い証明書で署名または暗号化されたペイロードを送信している可能性がある場合に、転送が中断されないように、移行は段階的に行われます。古い証明書と新しい証明書の両方が有効になる中間期間を「猶予期間」と呼びます。

X.509 証明書には Not Before 日付と Not After 日付があります。ただし、これらのパラメータでは管理者が十分に制御できない場合があります。Transfer Family は、アウトバウンドのペイロードにどの証明書を使用し、インバウンドのペイロードにど の証明書を受け入れるかを制御するために、Active Date および Inactive Date 設定を提供します。

アウトバウンド証明書の選択では、転送日より前の最大値が Inactive Date として使用されます。インバウンドプロセスでは、Not Before および Not After の範囲内、Active Date および Inactive Date の範囲内の証明書を受け付けます。

次の表は、1 つのプロファイルに 2 つの証明書を設定する方法の 1 つを示しています。

2 つの証明書をローテーション中

名前	NOT BEFORE(認証局による管理)	ACTIVE DATE (Transfer Family が設定)	INACTIVE DATE (Transfer Family が設定)	NOT AFTER (認証局によって設定される)
Cert1 (古い証明書)	2019-11-01	2020-01-01	2020-12-31	2024-01-01
証明書 2 (新しい証明書)	2020 年 11 月 1 日	2020-06-01	2021-06-01	2025-01-01

次の点に注意してください。

- 証明書に Active Date と Inactive Date を指定する場合、範囲は Not Before と Not After の間の範囲内である必要があります。
- プロファイルごとに複数の証明書を設定して、すべての証明書を組み合わせた有効な日付範囲がプロファイルを使用する期間をカバーするようにすることをお勧めします。
- 古い証明書が非アクティブになってから新しい証明書が有効になるまでの間にある程度の猶予時間を設定することをお勧めします。前の例では、最初の証明書は 2020-12-31 まで非アクティブにならず、2 番目の証明書は 2020-06-01 に有効になり、6 か月の猶予期間が与えられます。2020 年 6 月 1 日から 2020 年 12 月 31 日までの期間は、両方の証明書が有効です。

AS2 プロファイルの作成

この手順を使用して、ローカルプロファイルとパートナープロファイルの両方を作成します。ここでは、Transfer Family コンソールを使用して AS2 プロファイルを作成する方法について説明します。AWS CLI を代わりに使いたい場合は、the section called “ステップ 4: 自分と取引相手のプロフィールを作成する” を参照してください。

AS2 プロファイルを作成するには

1. https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開きます。
2. 左側のナビゲーションペインの「AS2 Trading Partners」で、「プロフィール」を選択し、「プロフィールを作成」を選択します。
3. 「プロファイル設定」セクションで、プロファイルの AS2 ID を入力します。この値は、AS2プロトコル固有のHTTPヘッダ `as2-from` と `as2-to` に使用され、取引パートナーシップを識別し、使用する証明書などを決定します。
4. 「プロファイルタイプ」セクションで、「ローカルプロファイル」または「パートナープロファイル」を選択します。
5. 「証明書」セクションで、ドロップダウンメニューから 1 つまたは複数の証明書を選択します。

   > **Note**
   >
   > ドロップダウンメニューに表示されていない証明書をインポートする場合は、「新しい証明書をインポートする」を選択します。これにより、「証明書のインポート」画面に新しいブラウザウィンドウが開きます。証明書をインポートする手順については、AS2 証明書のインポート を参照してください。

6. (オプション)「Tags」セクションで、このプロファイルを識別するためにキーと値のペアを 1 つ以上指定します。
7. 「プロファイルの作成」を選択してプロセスを完了し、新しいプロファイルを保存します。

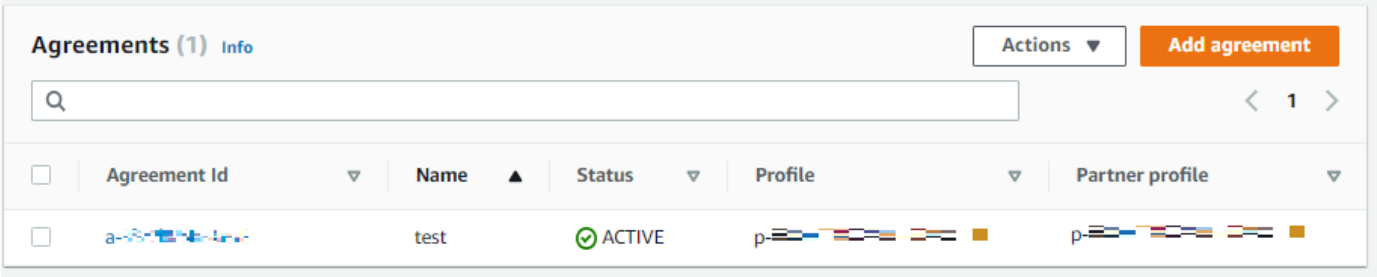
AS2 契約の作成

契約はTransfer Family サーバーに関連付けられています。AS2プロトコルを使用してTransferFamilyを使用してメッセージまたはファイルを交換する取引パートナー向けの詳細を指定します。「インバウンド」転送 (パートナー所有の外部ソースからTransTransfer Family ilyサーバーへのAS2ファイルの送信)

ここでは、Transfer Family コンソールを使用して AS2 契約を作成する方法について説明します。AWS CLI 代わりに を使用する場合は、「」を参照してくださいthe section called “ステップ 5: 自分とパートナーとの間で契約を作成する”。

Transfer Family サーバーの契約を作成するには

1. https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開きます。
2. 左側のナビゲーションペインで、「Servers」を選択し、AS2 プロトコルを使用するサーバーを選択します。
3. [サーバー詳細] ページで、「契約」セクションまで下にスクロールします。



4. 「契約を追加」を選択します。
5. 契約パラメータを次のように入力します。

   a. 「契約設定」セクションで、説明的な名前を入力します。契約の目的を特定できることを確認します。また、契約書の「ステータス」を「有効」(デフォルトで選択されている) または「無効」のいずれかに設定します。

   b. 「コミュニケーション設定」セクションで、ローカルプロファイルとパートナープロファイルを選択します。

   c. 「Inbox フォルダ設定」セクションで、受信ファイルを保存する Amazon S3 バケットと、バケットにアクセスできる IAM ロールを選択します。オプションで、バケットにファイルを保存するために使用するプレフィックス (フォルダ) を入力できます。

      例えば、バケットに **DOC-EXAMPLE-BUCKET**、プレフィックスに **incoming** を入力すると、受信ファイルは `/DOC-EXAMPLE-BUCKET/incoming` フォルダに保存されます。

   d. (オプション) 「タグ」セクションにタグを追加します。

   e. 契約書のすべての情報を入力したら、「契約を作成」を選択します。

新しい契約がサーバー詳細ページの「契約」セクションに表示されます。

AS2 メッセージの送受信

このセクションでは、AS2 メッセージを送受信するプロセスについて説明します。また、AS2 メッセージに関連付けられたファイル名と場所に関する詳細も提供します。

次の表に、AS2 メッセージで使用できる暗号化アルゴリズムと、それらを使用できるタイミングを示します。

暗号化アルゴリズム	HTTP	HTTPS	メモ
AES128_CBC	はい	はい	
AES192_CBC	はい	はい	
AES256_CBC	はい	はい	
DES_EDE3_CBC	はい	はい	このアルゴリズムは弱い暗号化アルゴリズムであるため、それを必要とするレガシークライアントをサポートする必要がある場合のみ使用してください。
なし	いいえ	はい	Transfer Family サーバーにメッセージを送信する場合は、Application Load Balancer (ALB) `NONE`を使用している場合にのみ選択できます。

トピック

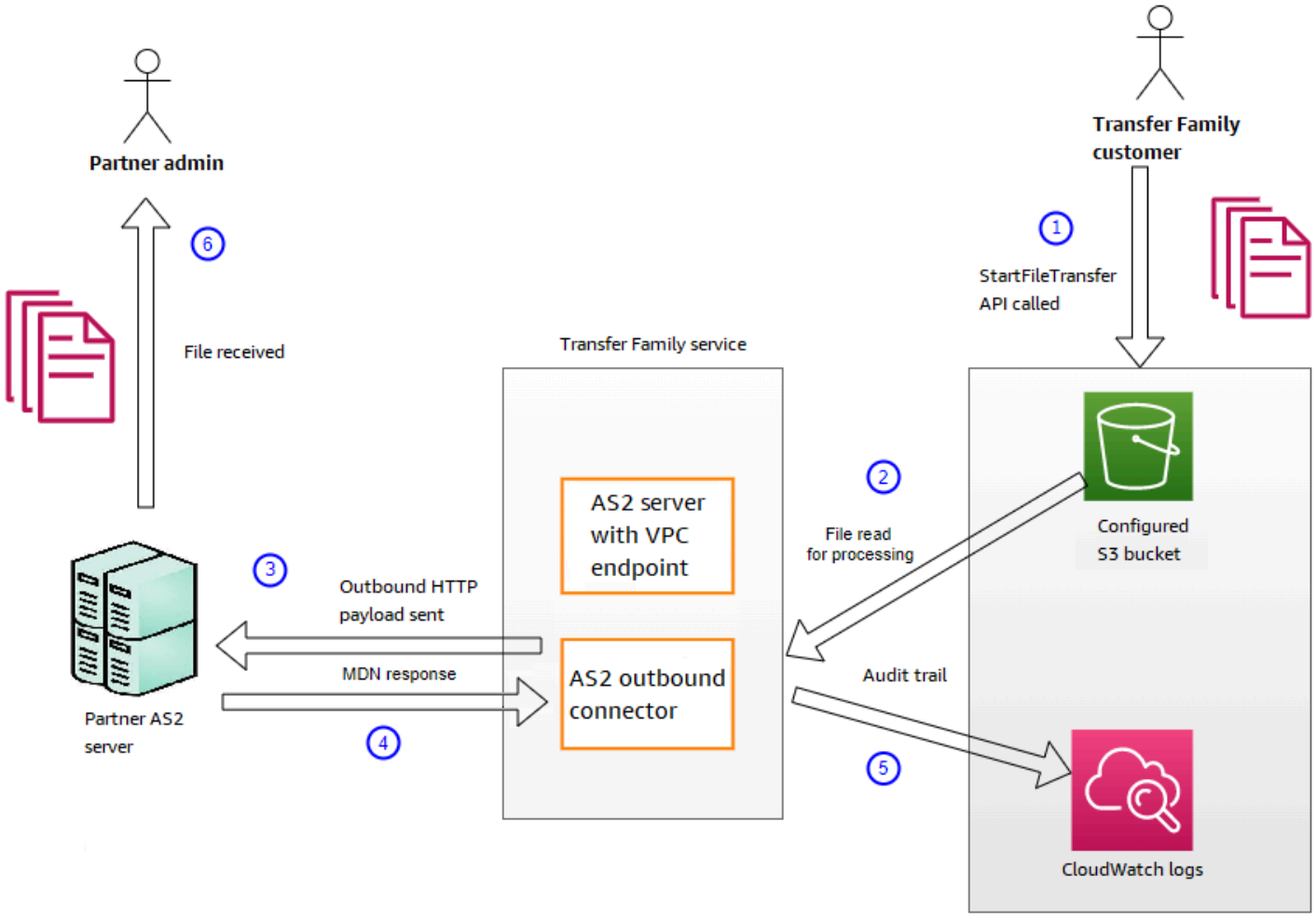
- AS2 メッセージプロセスの送信
- AS2 メッセージプロセスを受信する

- HTTPS 経由の AS2 メッセージの送受信
- AS2 コネクタを使用したファイルの転送
- ファイル名と場所
- ステータスコード
- JSON ファイルの例

AS2 メッセージプロセスの送信

アウトバウンドプロセスは、 から外部クライアントまたはサービス AWS に送信されるメッセージまたはファイルとして定義されます。アウトバウンドメッセージのシーケンスは以下の通りである：

1. 管理者は `start-file-transfer` AWS Command Line Interface （ AWS CLI) コマンドまたは `StartFileTransfer` API オペレーションを呼び出します。この操作は `connector` 設定を参照します。
2. Transfer Family は新しいファイルリクエストを検出し、ファイルを見つけます。ファイルは圧縮、署名、暗号化されます。
3. 転送 HTTP クライアントは HTTP POST リクエストを実行して、ペイロードをパートナーの AS2 サーバーに送信します。
4. このプロセスは、署名された MDN 応答を HTTP 応答 (同期 MDN) とインラインで返します。
5. ファイルがさまざまな転送段階の間を移動すると、プロセスは MDN 応答の受信と処理の詳細を顧客に配信します。
6. リモート AS2 サーバーは、復号化され検証されたファイルをパートナー管理者が利用できるようにします。



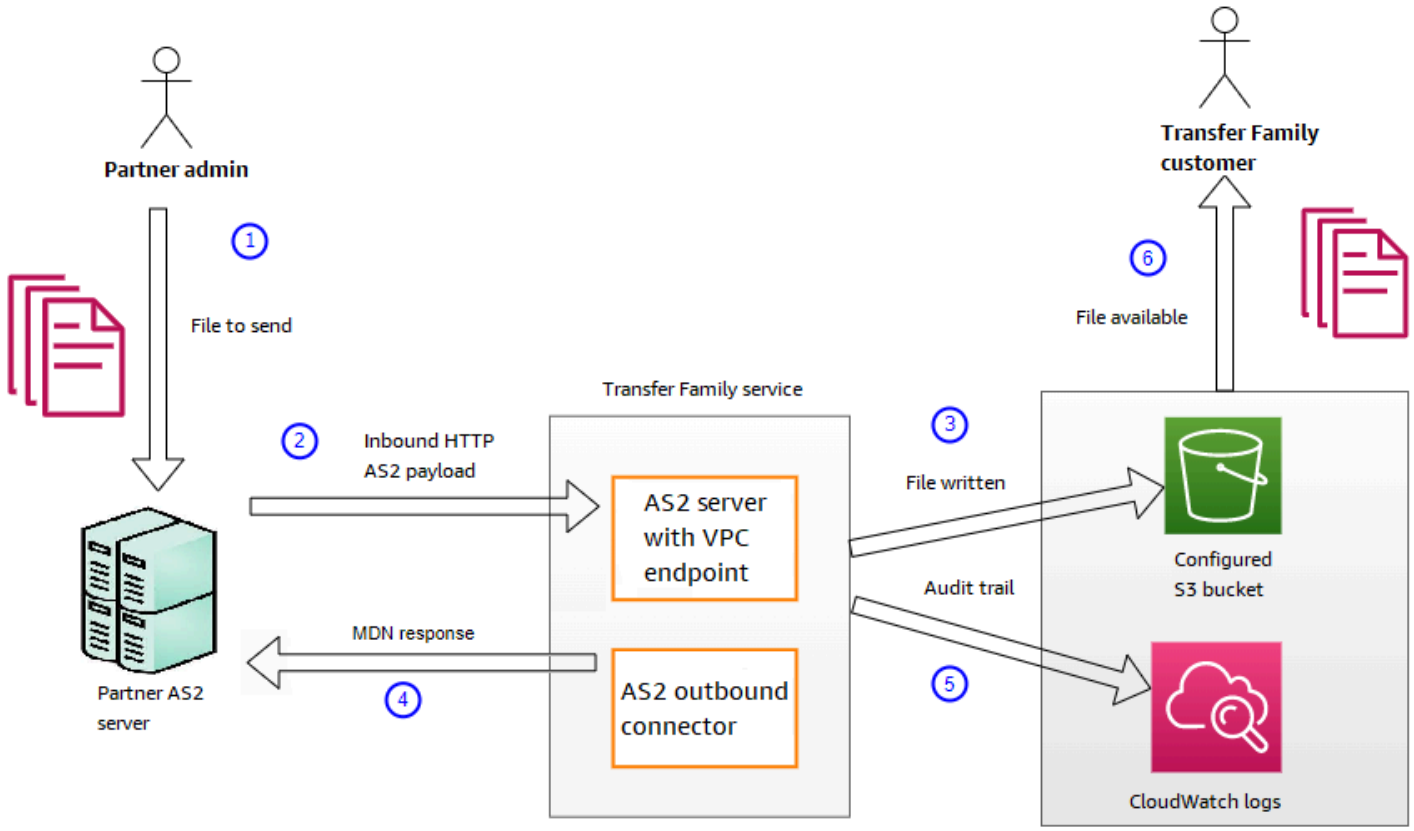
AS2 処理は、一般的なユースケースと既存の AS2 対応サーバー実装との統合に重点を置いて、RFC 4130 プロトコルの多くをサポートしています。サポートされている構成の詳細については、[サポートされる AS2 設計](#) を参照してください。

AS2 メッセージプロセスを受信する

インバウンドプロセスは、 AWS Transfer Family サーバーに転送されるメッセージまたはファイルとして定義されます。受信メッセージの順序は次のとおりです。

1. 管理プロセスまたは自動プロセスが、パートナーのリモート AS2 サーバーで AS2 ファイル転送を開始します。
2. パートナーのリモートAS2サーバーは、ファイルの内容に署名して暗号化し、Transfer Family でホストされているAS2インバウンドエンドポイントにHTTP POSTリクエストを送信します。

3. サーバー、パートナー、証明書、および契約の設定値を使用して、Transfer Family は AS2 ペイロードを復号化して検証します。ファイルの内容は、設定された Amazon S3 ファイルストアに保存されます。
4. 署名された MDN レスポンスは HTTP レスポンスと一緒に返されるか、別の HTTP POST リクエストを通じて非同期的に元のサーバーに返されます。
5. 監査証跡は、交換の詳細 CloudWatch とともに Amazon に書き込まれます。
6. 復号されたファイルは、`inbox/processed` という名前のフォルダにあります。



HTTPS 経由の AS2 メッセージの送受信

このセクションでは、AS2 プロトコルを使用して HTTPS 経由でメッセージを送受信する Transfer Family サーバーを設定する方法について説明します。

トピック

- HTTPS 経由で AS2 メッセージを送信
- HTTPS 経由で AS2 メッセージを受信

HTTPS 経由で AS2 メッセージを送信

HTTPS を使用して AS2 メッセージを送信するには、次の情報を含むコネクタを作成します。

- URL には HTTPS URL を指定
- 暗号化アルゴリズムでは、使用可能なアルゴリズムのいずれかを選択します。

> **Note**
>
> 暗号化を使用しない (つまり、暗号化アルゴリズムNONEに を選択する) ときに Transfer Family サーバーにメッセージを送信するには、Application Load Balancer (ALB) を使用する必要があります。

- 「AS2 コネクタを設定する」の説明に従って、コネクタの残りの値を指定します。

HTTPS 経由で AS2 メッセージを受信

AWS Transfer Family AS2 サーバーは現在、ポート 5080 経由の HTTP トランスポートのみを提供します。ただし、選択したポートと証明書を使用して、Transfer Family サーバー VPC エンドポイントの前にあるネットワークまたはアプリケーションロードバランサーで TLS を終了できます。この方法では、受信 AS2 メッセージに HTTPS を使用させることができます。

前提条件

- VPC は Transfer Family サーバー AWS リージョン と同じ にある必要があります。
- VPC のサブネットは、サーバーを使用するアベイラビリティーゾーン内にある必要があります。

> **Note**
>
> Transfer Family サーバーごとに、最大 3 つのアベイラビリティーゾーンをサポートできます。

- サーバーと同じリージョンに最大 3 つの Elastic IP アドレスを割り当てます。または、独自の IP アドレス範囲 (BYOIP) を使用することもできます。

> **Note**
>
> Elastic IP アドレスの数は、サーバーエンドポイントで使用するアベイラビリティーゾーンの数と一致する必要があります。

Network Load Balance (NLB) または Application Load Balancer (ALB) のいずれかを設定できます。次の表に、各アプローチの長所と短所を示します。

以下の表は、NLB と ALB を使用して TLS を終了する場合の機能の違いを示しています。

機能	Network Load Balancer (NLB)	Application Load Balancer (ALB)
レイテンシー	ネットワークレイヤーで動作するため、レイテンシーが低くなります。	アプリケーションレイヤーで動作するため、レイテンシーが長くなります。
静的 IP のサポート	静的な Elastic IP アドレスをアタッチできます。	Elastic IP アドレスをアタッチできません: 基になる IP アドレスを変更できるドメインを提供します。
高度なルーティング	アドバンストルーティングをサポートしていません。	高度なルーティングをサポートします。AS2 に必要な`X-Forwarded-Proto` ヘッダーを暗号化なしで挿入できます。 このヘッダーは、developer.mozilla.org ウェブサイトの X-Forwarded-Proto で説明されています。
TLS/SSL 終了	TLS/SSL ターミネーションをサポート	TLS/SSL ターミネーションをサポート

機能	Network Load Balancer (NLB)	Application Load Balancer (ALB)
相互 TLS (mTLS)	Transfer Family は現在、mTLS 用の NLB の使用をサポートしていません	mTLS のサポート

Configure NLB

この手順では、VPC でインターネット向け Network Load Balancer (NLB) を設定する方法について説明します。

Network Load Balancer を作成し、サーバーの VPC エンドポイントをロードバランサーのターゲットとして定義するには

1. Amazon Elastic Compute Cloud コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[ロードバランサー] を選択し、[ロードバランサーを作成] を選択します。
3. [Network Load Balancer] で、[Create] (作成) を選択します。
4. [基本設定] セクションで、以下の情報を入力します。
   - [Name] (名前) に、対象のロードバランサーを表現した説明的な名前を入力します。
   - [スキーム] で、[インターネット接続] を選択します。
   - [IP アドレスタイプ] で、[ipv4] を選択します。
5. [ネットワークマッピング] セクションで、以下の情報を入力します。
   - [VPC] で、作成した仮想プライベートクラウド (VPC) を選択します。
   - [マッピング] で、サーバーエンドポイントで使用しているのと同じ VPC で使用可能なパブリックサブネットに関連付けられているアベイラビリティーゾーンを選択します。
   - 各サブネットの IPv4 アドレスには、割り当てた Elastic IP アドレスのいずれかを選択します。
6. [リスナーとルーティング] セクションに、以下の情報を入力します。
   - [プロトコル] で [TLS] を選択します。
   - [Port (ポート)] に「**`5080`**」と入力します。

- [デフォルトアクション] で、[ターゲットグループの作成] を選択します。新しいターゲットグループの作成の詳細については、「対象グループを作成するには」を参照してください。

  ターゲットグループを作成した後で、[デフォルトアクション] フィールドにその名前を入力します。

7. [セキュアリスナー設定] セクションの [デフォルト SSL/TLS 証明書] 領域で証明書を選択します。
8. [ロードバランサーの作成] を選択し、NLB を作成します。
9. (オプションですが、推奨されます) Network Load Balancer のアクセスログを有効にして、完全な監査証跡を維持します。これについては、「Network Load Balancer のアクセスログ」を参照してください。

   TLS 接続は NLB で終了するため、このステップをお勧めします。したがって、Transfer Family AS2 CloudWatch ロググループに反映される送信元 IP アドレスは、取引相手の外部 IP アドレスではなく、NLB のプライベート IP アドレスです。

Configure ALB

この手順では、VPC で Application Load Balancer (NLB) を設定する方法について説明します。

Application Load Balancer を作成し、サーバーの VPC エンドポイントをロードバランサーのターゲットとして定義するには

1. Amazon Elastic Compute Cloud コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[ロードバランサー] を選択し、[ロードバランサーを作成] を選択します。
3. [Application Load Balancer] で [作成] を選択します。
4. ALB コンソールで、ポート 443 (HTTPS) に新しい HTTP リスナーを作成します。
5. (オプション)。相互認証 (mTLS) を設定する場合は、セキュリティ設定と信頼ストアを設定します。

   a. SSL/TLS 証明書をリスナーにアタッチします。
   b. クライアント証明書処理 で、相互認証 (mTLS) を選択します。
   c. 信頼ストア で検証を選択します。

d. 詳細 mTLS 設定 で、CA 証明書をアップロードして信頼ストアを選択または作成します。

6. 新しいターゲットグループを作成し、Transfer Family AS2 サーバーエンドポイントのプライベート IP アドレスをポート 5080 のターゲットとして追加します。新しいターゲットグループの作成の詳細については、「対象グループを作成するには」を参照してください。
7. ポート 5080 で TCP プロトコルを使用するようにターゲットグループのヘルスチェックを設定します。
8. リスナーからターゲットグループに HTTPS トラフィックを転送する新しいルールを作成します。
9. SSL/TLS 証明書を使用するようにリスナーを設定します。

ロードバランサーを設定すると、クライアントはカスタムポートリスナーを介してロードバランサーと通信します。次に、ロードバランサーはポート 5080 を介してサーバーと通信します。

対象グループを作成するには

1. 前の手順で [ターゲットグループの作成] を選択すると、新しいターゲットグループの [グループの詳細を指定] ページが表示されます。
2. [基本設定] セクションで、以下の情報を入力します。
   - [ターゲットタイプを選択] で [IP アドレス] を選択します。
   - [ターゲットグループ名] に、ターゲットグループの名前を入力します。
   - [プロトコル] で [TCP] を選択します。
   - [Port (ポート)] に「**5080**」と入力します。
   - [IP アドレス] タイプで、[ipv4] を選択します。
   - [VPC] で、Transfer Family AS2 サーバー用に作成した VPC を選択します。
3. [ヘルスチェック] セクションで、[ヘルスチェックプロトコル] に [TCP] を選択します。
4. [次へ] をクリックします。
5. [ターゲットの登録] ページで、次の情報を入力します。
   - [ネットワーク] では、Transfer Family AS2 サーバー用に作成した VPC が指定されていることを確認します。
   - [IPv4 アドレス] では、Transfer Family AS2 サーバーのエンドポイントのプライベート IPv4 アドレスを入力します。

サーバーのエンドポイントが複数ある場合は、[IPv4 アドレスを追加] を選択して別の IPv4 アドレスを入力する行をもう 1 つ追加します。サーバーのすべてのエンドポイントのプライベート IP アドレスを入力し終わるまで、このプロセスを繰り返します。

- [ポート] が **5080** に設定されていることを確認します。
- [保留中として以下を含める] を選択し、[レビューターゲット] セクションにエントリを追加します。

6. [レビューターゲット] セクションで、IP ターゲットをレビューします。
7. [ターゲットグループの作成] を選択し、前の NLB の作成手順に戻り、指示された場所に新しいターゲットグループを入力します。

Elastic IP アドレスからサーバーへのアクセスをテストします

Elastic IP アドレスまたは Network Load Balancer の DNS 名を使用して、カスタムポート経由でサーバーに接続します。

> **Important**
>
> ロードバランサーに設定されたサブネットのネットワークアクセスコントロールリスト (ネットワーク ACL) を使用して、クライアント IP アドレスからサーバーへのアクセスを管理します。ネットワーク ACL のアクセス許可はサブネットレベルで設定されるため、ルールはサブネットを使用するすべてのリソースに適用されます。ロードバランサーのターゲットタイプはインスタンスではなく IP アドレスに設定されているため、セキュリティグループを使用してクライアント IP アドレスからのアクセスを制御することはできません。そのため、ロードバランサーはソース IP アドレスを保持しません。Network Load Balancer のヘルスチェックが失敗すると、ロードバランサーはサーバーエンドポイントに接続できなくなります。この問題のトラブルシューティングを行うには、次を確認します。
>
> - サーバーのエンドポイントに関連付けられたセキュリティグループが、ロードバランサーに設定されているサブネットからのインバウンド接続を許可していることを確認します。ロードバランサーは、ポート 5080 を介してサーバーエンドポイントに接続できる必要があります。
> - サーバーの [状態] が [オンライン] であることを確認します。

AS2 コネクタを使用したファイルの転送

AS2コネクタは、Transfer Familyサーバーからパートナーが所有する外部の宛先にAS2メッセージを転送するための取引パートナー間の関係を確立します。

Transfer Family を使用して AS2 メッセージを送信するには、次の `start-file-transfer` AWS Command Line Interface（ AWS CLI) コマンドに示すように、コネクタ ID とファイルへのパスを参照します。

```
aws transfer start-file-transfer --connector-id c-1234567890abcdef0 \
--send-file-paths "/DOC-EXAMPLE-SOURCE-BUCKET/myfile1.txt" "/DOC-EXAMPLE-SOURCE-BUCKET/
myfile2.txt"
```

コネクタの詳細情報を取得するには、次のコマンドを実行しする：

```
aws transfer list-connectors
```

`list-connectors` コマンドは、コネクタのコネクタ ID、URL、Amazon リソースネーム (ARN) を返します。

特定のコネクタのプロパティを返すには、使用する ID を指定して以下のコマンドを実行します。

```
aws transfer describe-connector --connector-id your-connector-id
```

`describe-connector` コマンドは、URL、ロール、プロファイル、メッセージ処理通知 (MDN)、タグ、モニタリングメトリックなど、コネクタのすべてのプロパティを返します。

JSON と MDN ファイルを表示すると、パートナーがファイルを正常に受信したことを確認できます。これらのファイルには、ファイル名と場所 で説明されている規則に従って名前が付けられます。コネクタの作成時にログ記録ロールを設定した場合は、 CloudWatch ログで AS2 メッセージのステータスを確認することもできます。

AS2 コネクターの詳細を表示するには、AS2 コネクターの詳細を表示を参照してください。AS2 コネクターの作成についての詳細は、AS2 コネクタを設定する を参照してください。

ファイル名と場所

このセクションでは、AS2 転送のファイル命名規則について説明します。

インバウンドファイル転送については、次の点に注意してください。

- 基本ディレクトリは契約書で指定します。ベースディレクトリは、Amazon S3 バケット名にプレフィックス (ある場合) を組み合わせたものです。例えば /DOC-EXAMPLE-BUCKET/AS2-folder です。
- 受信ファイルが正常に処理されると、ファイル (および対応する JSON ファイル) が /processed フォルダに保存されます。例えば /DOC-EXAMPLE-BUCKET/AS2-folder/processed です。

  JSONファイルには以下のフィールドが含まれる：

  - agreement-id
  - as2-from
  - as2-to
  - as2-message-id
  - transfer-id
  - client-ip
  - connector-id
  - failure-message
  - file-path
  - message-subject
  - mdn-message-id
  - mdn-subject
  - requester-file-name
  - requester-content-type
  - server-id
  - status-code
  - failure-code
  - transfer-size
- 受信ファイルを正常に処理できない場合、ファイル (および対応する JSON ファイル) は /failed フォルダーに保存されます。例えば /DOC-EXAMPLE-BUCKET/AS2-folder/failed です。
- 転送されたファイルは、*original_filename*.*messageId*.*original_extension* として processed フォルダーに保存されます。つまり、転送のメッセージ ID がファイル名の元の拡張子の前に追加されます。

- JSON ファイルは *original_filename*.*messageId*.*original_extension*.json として作成され、保存されます。追加されるメッセージ ID に加えて、文字列 .json は転送されたファイルの名前に追加されます。
- メッセージ処理通知 (MDN) ファイルが作成され、*original_filename*.*messageId*.*original_extension*.mdn という名前で保存されます。追加されるメッセージ ID に加えて、文字列 .mdn は転送されたファイルの名前に追加されます。
- ExampleFileInS3Payload.dat という名前の受信ファイルがある場合、次のファイルが作成されます。
  - File – ExampleFileInS3Payload.c4d6b6c7-23ea-4b8c-9ada-0cb811dc8b35@44313c54b0a46a36.
  - JSON – ExampleFileInS3Payload.c4d6b6c7-23ea-4b8c-9ada-0cb811dc8b35@44313c54b0a46a36.
  - MDN – ExampleFileInS3Payload.c4d6b6c7-23ea-4b8c-9ada-0cb811dc8b35@44313c54b0a46a36.

アウトバウンド転送の場合、名前は似ていますが、受信メッセージファイルがないことと、転送されたメッセージの転送 ID がファイル名に追加される点が異なります。転送 ID は StartFileTransfer API 操作によって (または別のプロセスまたはスクリプトがこの操作を呼び出したときに) 返されます。

- transfer-id はファイル転送に関連付けられる識別子です。StartFileTransfer コールの一部であるすべてのリクエストは transfer-id を共有します。
- ベースディレクトリは、ソースファイルに使用するパスと同じです。つまり、ベースディレクトリは StartFileTransfer API オペレーションまたはstart-file-transfer AWS CLI コマンドで指定するパスです。例:

```
aws transfer start-file-transfer --send-file-paths /DOC-EXAMPLE-BUCKET/AS2-folder/
file-to-send.txt
```

このコマンドを実行すると、MDN ファイルと JSON ファイルは /DOC-EXAMPLE-BUCKET/AS2-folder/processed (転送が成功した場合) または /DOC-EXAMPLE-BUCKET/AS2-folder/failed (転送に失敗した場合) に保存されます。

- JSON ファイルは *original_filename*.*transferId*.*messageId*.*original_extension*.json として作成され、保存されます。
- MDN ファイルは *original_filename*.*transferId*.*messageId*.*original_extension*.mdn として作成され、保存されます。
- ExampleFileOutTestOutboundSyncMdn.dat という名前のアウトバウンドファイルがある場合、次のファイルが作成されます。
  - JSON – ExampleFileOutTestOutboundSyncMdn.dedf4601-4e90-4043-b16b-579af35e0d83.fbe18db8-7361-42ff-8ab6-49ec1e435f34@c9c705f0baaaabaa.dat.j
  - MDN – ExampleFileOutTestOutboundSyncMdn.dedf4601-4e90-4043-b16b-579af35e0d83.fbe18db8-7361-42ff-8ab6-49ec1e435f34@c9c705f0baaaabaa.dat.m

また、 CloudWatch ログを確認して、失敗した転送の詳細を表示することもできます。

ステータスコード

次の表に、ユーザーまたはパートナーが AS2 メッセージを送信したときに CloudWatch ログに記録できるすべてのステータスコードを示します。異なるメッセージ処理ステップは、異なるメッセージタイプに適用され、モニタリングのみを目的としています。COMPLETED 状態と FAILED 状態は処理の最終ステップを表し、JSON ファイルに表示されます。

Code	説明	処理が完了しましたか？
処理	メッセージは最終形式に変換中です。例えば、解凍ステップと復号ステップの両方にこのステータスがあります。	いいえ
MDN_TRANSMIT	メッセージ処理が MDN レスポンスを送信しています。	いいえ
MDN_RECEIVE	メッセージ処理が MDN レスポンスを受信しています。	いいえ
COMPLETED	メッセージ処理が正常に完了しました。この状態には、イ	はい

Code	説明	処理が完了しましたか？
	ンバウンドメッセージまたはアウトバウンドメッセージの MDN 検証に MDN が送信されるときが含まれます。	
FAILED	メッセージ処理が失敗しました。エラーコードのリストについては、「」を参照してくださいAS2 エラーコード。	はい

JSON ファイルの例

このセクションには、転送が成功した場合と失敗した転送のサンプルファイルなど、インバウンドとアウトバウンドの両方の転送のサンプル JSON ファイルが表示されます。

正常に転送されたアウトバウンドファイルの例:

```
{
  "requester-content-type": "application/octet-stream",
  "mesage-subject": "File xyzTest from MyCompany_OID to partner YourCompany",
  "requester-file-name": "TestOutboundSyncMdn-9lmCr79hV.dat",
  "as2-from": "MyCompany_OID",
  "connector-id": "c-c21c63ceaaf34d99b",
  "status-code": "COMPLETED",
  "disposition": "automatic-action/MDN-sent-automatically; processed",
  "transfer-size": 3198,
  "mdn-message-id": "OPENAS2-11072022063009+0000-df865189-1450-435b-9b8d-
d8bc0cee97fd@PartnerA_OID_MyCompany_OID",
  "mdn-subject": "Message be18db8-7361-42ff-8ab6-49ec1e435f34@c9c705f0baaaabaa has been
 accepted",
  "as2-to": "PartnerA_OID",
  "transfer-id": "dedf4601-4e90-4043-b16b-579af35e0d83",
  "file-path": "/DOC-EXAMPLE-BUCKET/as2testcell0000/openAs2/
TestOutboundSyncMdn-9lmCr79hV.dat",
  "as2-message-id": "fbe18db8-7361-42ff-8ab6-49ec1e435f34@c9c705f0baaaabaa",
  "timestamp": "2022-07-11T06:30:10.791274Z"
}
```

転送に失敗したアウトバウンドファイルの例:

```
{
  "failure-code": "HTTP_ERROR_RESPONSE_FROM_PARTNER",
  "status-code": "FAILED",
  "requester-content-type": "application/octet-stream",
  "subject": "Test run from Id da86e74d6e57464aae1a55b8596bad0a to partner
 9f8474d7714e476e8a46ce8c93a48c6c",
  "transfer-size": 3198,
  "requester-file-name": "openAs2TestOutboundWrongAs2Ids-necco-3VYn5n8wE.dat",
  "as2-message-id": "9a9cc9ab-7893-4cb6-992a-5ed8b90775ff@718de4cec1374598",
  "failure-message": "http://Test123456789.us-east-1.elb.amazonaws.com:10080 returned
 status 500 for message with ID 9a9cc9ab-7893-4cb6-992a-5ed8b90775ff@718de4cec1374598",
  "transfer-id": "07bd3e07-a652-4cc6-9412-73ffdb97ab92",
  "connector-id": "c-056e15cc851f4b2e9",
  "file-path": "/testbucket-4c1tq6ohjt9y/as2IntegCell0002/openAs2/
openAs2TestOutboundWrongAs2Ids-necco-3VYn5n8wE.dat",
  "timestamp": "2022-07-11T21:17:24.802378Z"
}
```

正常に転送された受信ファイルの例:

```
{
  "requester-content-type": "application/EDI-X12",
  "subject": "File openAs2TestInboundAsyncMdn-necco-5Ab6bTfCO.dat sent from MyCompany
 to PartnerA",
  "client-ip": "10.0.109.105",
  "requester-file-name": "openAs2TestInboundAsyncMdn-necco-5Ab6bTfCO.dat",
  "as2-from": "MyCompany_OID",
  "status-code": "COMPLETED",
  "disposition": "automatic-action/MDN-sent-automatically; processed",
  "transfer-size": 1050,
  "mdn-subject": "Message Disposition Notification",
  "as2-message-id": "OPENAS2-11072022233606+0000-5dab0452-0ca1-4f9b-b622-
fba84effff3c@MyCompany_OID_PartnerA_OID",
  "as2-to": "PartnerA_OID",
  "agreement-id": "a-f5c5cbea5f7741988",
  "file-path": "processed/openAs2TestInboundAsyncMdn-
necco-5Ab6bTfCO.OPENAS2-11072022233606+0000-5dab0452-0ca1-4f9b-b622-
fba84effff3c@MyCompany_OID_PartnerA_OID.dat",
  "server-id": "s-5f7422b04c2447ef9",
  "timestamp": "2022-07-11T23:36:36.105030Z"
}
```

転送に失敗した受信ファイルの例:

```
{
  "failure-code": "INVALID_REQUEST",
  "status-code": "FAILED",
  "subject": "Sending a request from InboundHttpClientTests",
  "client-ip": "10.0.117.27",
  "as2-message-id": "testFailedLogs-TestRunConfig-Default-inbound-direct-
integ-0c97ee55-af56-4988-b7b4-a3e0576f8f9c@necco",
  "as2-to": "0beff6af56c548f28b0e78841dce44f9",
  "failure-message": "Unsupported date format: 2022/123/456T",
  "agreement-id": "a-0ceec8ca0a3348d6a",
  "as2-from": "ab91a398aed0422d9dd1362710213880",
  "file-path": "failed/01187f15-523c-43ac-9fd6-51b5ad2b08f3.testFailedLogs-
TestRunConfig-Default-inbound-direct-integ-0c97ee55-af56-4988-b7b4-a3e0576f8f9c@necco",
  "server-id": "s-0582af12e44540b9b",
  "timestamp": "2022-07-11T06:30:03.662939Z"
}
```

AS2 使用状況のモニタリング

Amazon CloudWatch および を使用して AS2 アクティビティをモニタリングできますAWS CloudTrail。他の Transfer Family サーバーメトリックスを表示するには、の Amazon CloudWatch ログ記録 AWS Transfer Family を参照してください

AS2 メトリクス

メトリクス	説明
InboundMessage	取引相手から正常に受信した AS2 メッセージの総数。 単位: カウント 期間: 5 分
InboundFailedMessage	取引相手から正常に受信されなかった AS2 メッセージの総数。つまり、取引相手がメッセージを送信しましたが、Transfer Family サーバーはそれを正常に処理できませんでした。

メトリクス	説明
	単位: カウント 期間: 5 分
OutboundMessage	Transfer Family サーバーから取引相手に正常に送信された AS2 メッセージの総数。 単位: カウント 期間 = 5 分
OutboundFailedMessage	取引相手への送信に受信失敗した AS2 メッセージの総数。つまり、Transfer Family サーバーから送信されましたが、取引相手には正常に受信されませんでした。 単位: カウント 期間: 5 分

AS2 ステータスコード

次の表は、ユーザーまたはパートナーが AS2 メッセージを送信したときに CloudWatch ログに記録できるすべてのステータスコードの一覧です。異なるメッセージ処理ステップは、異なるメッセージタイプに適用され、モニタリングのみを目的としています。COMPLETED および FAILED 状態は、処理の最後のステップを表し、JSON ファイルに表示されます。

Code	説明	処理は完了していますか？
保護	メッセージは最終形式に変換中です。例えば、解凍ステップと復号ステップの両方がこのステータスになります。	いいえ
MDN_TRANSMIT	メッセージ処理が MDN レスポンスを送信しています。	いいえ

Code	説明	処理は完了していますか？
MDN_RECEIVE	メッセージ処理は MDN レスポンスを受信しています。	いいえ
COMPLETED	メッセージ処理が正常に完了しました。この状態には、インバウンドメッセージまたはアウトバウンドメッセージの MDN 検証のために MDN が送信されるときが含まれます。	はい
FAILED	メッセージ処理が失敗しました。エラーコードのリストについては、「」を参照してくださいAS2 エラーコード。	はい

AS2 エラーコード

次の表に、AS2 ファイル転送から受け取る可能性のあるエラーコードの一覧と説明を示します。

AS2 エラーコード

Code	エラー	説明と解決策
`ACCESS_DENIED`	• アクセスが拒否されました。アクセスロールに必要な権限があるかどうかを確認してください。 • 無効なファイルパス *send-file-path* • で認証情報を取得できませんでした ErrorCode: *error-code*	`SendFilePaths` にいずれも有効でない、または形式に誤りがある `StartFileTransfer` リクエストを処理する際に発生します。つまり、パスに Amazon S3 バケット名がないか、パスに無効な文字が含まれています。Transfer Family がアクセスロールまたは口グ記録用の口ールを引き受けられない場合にも発生します。

Code	エラー	説明と解決策
		パスに有効な Amazon S3 バケット名とキー名が含まれていることを確認します。
AGREEMENT_NOT_FOUND	契約が見つかりませんでした。	契約が見つからなかったか、非アクティブなプロファイルに関連付けられているかのどちらかです。 Transfer Family サーバー内の契約を更新して、アクティブなプロフィールを含めてください。
CONNECTOR_NOT_FOUND	コネクタまたは関連する設定が見つかりませんでした。	コネクタが見つからなかったか、非アクティブなプロファイルに関連付けられているかのどちらかです。 コネクタを更新して、アクティブなプロファイルを含めてください。

Code	エラー	説明と解決策
CREDENTIALS_RETRIEVAL_FAILED	1. シークレットが Secrets Manager に見つかりません。 2. Secrets Manager にアクセスできません。 3. Secrets Manager でシークレットの値を復号できませんでした。 4. スロットリングのためシークレット値を取得できません。	AS2 Basic 認証では、シークレットを正しくフォーマットする必要があります。以下の解決策は前のコラムにリストされたエラーに対応しています。 1. シークレット ID が正しいことを確認してください。 2. アクセスロールにシークレットを読み取るための適切なアクセス許可があることを確認してください。アクセスロールは、StartFileTransfer リクエストで使用されるファイルの場所の親ディレクトリに対する読み取り/書き込みアクセスを提供する必要があります。さらに、StartFileTransfer で送信するファイルの親ディレクトリに対する読み取り/書き込みアクセスを提供していることを確認してください。 3. カスタマー管理キーがシークレットに使用されている場合は、アクセスロールに AWS Key Management Service (AWS KMS) キーに対する権限があることを確認してください。

Code	エラー	説明と解決策
		4. 該当するクォータについては、シークレットの処理クォータ を参照してください。
DECOMPRESSION_FAILED	メッセージを解凍できませんでした。	送信されたファイルが破損しているか、圧縮アルゴリズムが無効です。 メッセージを再送信して ZLIB 圧縮が使用されていることを確認するか、圧縮を有効にせずにメッセージを再送信してください。
DECRYPT_FAILED	メッセージの *message-ID* を復号化できませんでした。パートナーが正しい公開暗号鍵を持っていることを確認してください。	復号化に失敗しました。 パートナーが有効な証明書を使用してペイロードを送信したこと、および暗号化が有効な暗号化アルゴリズムを使用して実行されたことを確認してください。
DECRYPT_FAILED_INVALID_SMIME_FORMAT	エンベロープされた mimePart を解析できません。	MIME ペイロードが壊れているか、サポートされていない SMIME 形式です。 送信者は、使用している形式がサポートされていることを確認し、ペイロードを再送信する必要があります。

Code	エラー	説明と解決策
DECRYPT_FAILED_NO_DECRYPTION_KEY_FOUND	一致する復号化キーは見つかりませんでした。	パートナープロファイルには、メッセージと一致する証明書が割り当てられていなかったか、メッセージと一致した証明書の有効期限が切れているか、無効になっています。 パートナープロファイルを更新し、有効な証明書が含まれていることを確認する必要があります。
DECRYPT_FAILED_UNSUPPORTED_ENCRYPTION_ALG	サポートされていないアルゴリズムを使用して、ID: *encryption-ID* で SMIME ペイロード復号化が要求されました。	リモート送信者が、サポートされていない暗号化アルゴリズムで AS2 ペイロードを送信しました。 送信者は、AWS Transfer Family でサポートされている暗号化アルゴリズムを選択する必要があります。
DUPLICATE_MESSAGE	重複または二重処理されたステップ。	ペイロードには重複した処理ステップがあります。例えば、暗号化には 2 つのステップがあります。 署名、圧縮、暗号化を単一の手順でメッセージを再送信します。

Code	エラー	説明と解決策
ENCRYPT_FAILED_NO_ENCRYPTION_KEY_FOUND	プロファイル: *local-profile-ID* に有効な公開暗号化証明書が見つかりません	Transfer Family はアウトバウンドメッセージを暗号化しようとしていますが、ローカルプロファイルの暗号化証明書が見つかりません。 解決オプション: • ローカルプロファイルには、暗号化用の証明書とプライベートキーが添付されていることを確認してください。 • 暗号化証明書が現在有効であることを確認してください。
ENCRYPTION_FAILED	ファイル *file-name* の暗号化に失敗しました。	送信するファイルは暗号化できません。 ファイルが AS2 の想定どおりの場所にあり、AWS Transfer Family にファイルを読み取る権限があることを確認してください。
FILE_SIZE_TOO_LARGE	ファイルサイズが大きすぎます。	これは、ファイルサイズの制限を超えるファイルを送受信したときに発生します。

Code	エラー	説明と解決策
HTTP_ERROR_RESPONSE_FROM_PARTNER	*partner-URL* が ID=*message-ID* のメッセージに対してステータス 400 を返しました。	パートナーの AS2 サーバーと通信すると、予期しない HTTP レスポンスコードが返されました。 パートナーは AS2 サーバーログからより多くの診断を提供できる可能性があります。
INSUFFICENT_MESSAGE_SECURITY_UNENCRYPTED	暗号化が必要です。	パートナーは暗号化されていないメッセージを Transfer Family に送信しましたが、これはサポートされていません。送信者は暗号化されたペイロードを使用する必要があります。
INVALID_ENDPOINT_PROTOCOL	HTTP と HTTPS のみがサポートされます。	AS2 コネクタ設定のプロトコルとして HTTP または HTTPS を指定する必要があります。

Code	エラー	説明と解決策
INVALID_REQUEST	1. メッセージヘッダーに問題があります。 2. シークレット JSON を解析できませんでした。 シークレット JSON が予想された形式と一致しませんでした。 3. シークレットは JSON 文字列でなければなりません。 4. ユーザー名にはコロンを含めないでください。 ユーザー名には制御文字を含めないでください。 ユーザー名には ASCII 文字のみを使用します。 パスワードには制御文字を含めないでください。 パスワードには ASCII 文字のみを使用します。	このエラーには複数の原因があります。以下の解決策は前のコラムにリストされたエラーに対応しています。 1. as2-from および as2-to フィールドを確認してください。元のメッセージ ID が MDN 形式に合っていることを確認してください。また、メッセージ ID 形式に AS2 ヘッダーが欠落していないことも確認してください。 2. 「AS2 コネクタの基本認証を有効にします。」で説明されているように、シークレット値が文書化されている形式と一致していることを確認してください。 3. シークレットはバイナリではなく文字列として指定してください。 4. ユーザー名またはパスワードに必要な修正を行います。

Code	エラー	説明と解決策
INVALID_URL_FORMAT	無効な URL 形式: *URL*	これは、不正な URL で設定されたコネクタを使用してアウトバウンドメッセージを送信している場合に発生します。 コネクタが有効な HTTP または HTTPS URL で設定されていることを確認します。
MDN_RESPONSE_INDICATES_AUTHENTICATION_FAILED	該当しない	受信者は送信者を認証できません。取引相手は、処理修飾子「Error: authentication-failed」を含む MDN を Transfer Family に返します。
MDN_RESPONSE_INDICATES_DECOMPRESSION_FAILED	該当しない	これは、受信者がメッセージの内容を解凍できない場合に発生します。取引相手は、処理修飾子「Error: decompression-failed」を含む MDN を Transfer Family に返します。
MDN_RESPONSE_INDICATES_DECRYPTION_FAILED	該当しない	受信者がメッセージの内容を復号できません。取引相手は、処理修飾子「Error: authentication-failed」を含む MDN を Transfer Family に返します。

Code	エラー	説明と解決策
`MDN_RESPONSE_INDICATES_INSUFFICIENT_MESSAGE_SECURITY`	該当しない	受信者はメッセージが署名されているか暗号化されていることを期待していますが、そうではありません。取引相手は、処理修飾子 Error: insufficient-message-security を持つ MDN を Transfer Family に返します。 取引相手の期待に合わせて、コネクタの署名や暗号化を有効にします。
`MDN_RESPONSE_INDICATES_INTEGRITY_CHECK_FAILED`	該当しない	受信者はコンテンツの完全性を検証できません。取引相手は、処理修飾子 Error: integrity-check-failed を持つ MDN を Transfer Family に返します。
`PATH_NOT_FOUND`	ディレクトリ *`file-path`* を作成できません。親パスが見つかりませんでした。	Transfer Family はお客様の Amazon S3 バケットにディレクトリを作成しようとしていますが、バケットが見つかりません。 `StartFileTransfer` コマンドに記載されている各パスに既存のバケットの名前が含まれていることを確認します。

Code	エラー	説明と解決策
SEND_FILE_NOT_FOUND	ファイルパス *file-path* が見つかりません。	Transfer Family はファイル送信操作でファイルを見つけることができません。 設定したホームディレクトリとパスが有効であること、および Transfer Family にファイルの読み取りアクセス許可があることを確認してください。
SERVER_NOT_FOUND	メッセージに関連付けられたサーバーが見つかりません。	Transfer Family は、メッセージを受信したときにサーバーを見つけることができませんでした。これは、受信メッセージの処理中にサーバーが削除された場合に発生する可能性があります。
SERVER_NOT_ONLINE	サーバー *server-ID* はオンラインではありません。	Transfer Family サーバーはオフラインです。 メッセージを受信して処理できるよう、サーバーを起動します。
SIGNING_FAILED	ファイルに署名できませんでした。	送信するファイルには署名ができないか、署名を実行することができませんでした。 ファイルが AS2 の想定どおりの場所にあり、AWS Transfer Family にファイルを読み取る権限があることを確認してください。

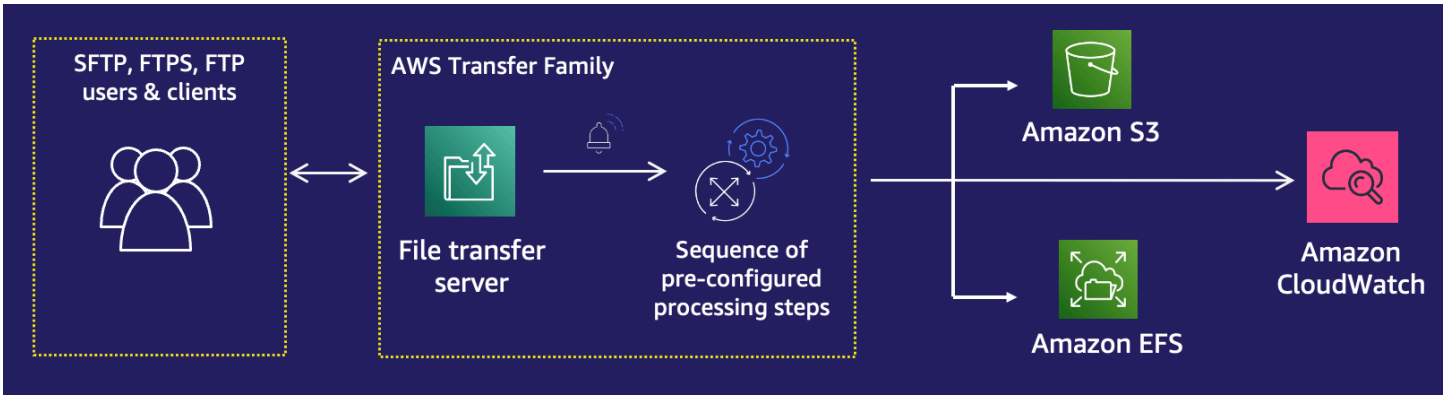
Code	エラー	説明と解決策
SIGNING_FAILED_NO_SIGNING_KEY_FOUND	プロファイル: *local-profile-ID* の証明書が見つかりません。	アウトバウンドメッセージを署名しようとしていますが、ローカルプロファイルの署名用証明書が見つかりません。 解決オプション: • ローカルプロファイルには、署名用の証明書とプライベートキーが添付されていることを確認してください。 • 署名用証明書が現在有効であることを確認してください。
UNABLE_RESOLVE_HOST_TO_IP_ADDRESS	ホスト名を IP アドレスに変換できません。	Transfer Family は、AS2 コネクタで設定されているパブリック DNS サーバー上で DNS から IP アドレスへの解決を実行できません。 有効なパートナー URL を指定するコネクタを更新してください。
UNABLE_TO_CONNECT_TO_REMOTE_HOST_OR_IP	エンドポイントへの接続がタイムアウトしました。	Transfer Family は、設定されたパートナーの AS2 サーバーへのソケット接続を確立できません。 パートナーの AS2 サーバーが設定された IP アドレスで利用できることを確認してください。

Code	エラー	説明と解決策
UNABLE_TO_RESOLVE_HOSTNAME	ホスト名 *hostname* を解決できません。	Transfer Family サーバーは、パブリック DNS サーバーを使用してパートナーのホスト名を解決できませんでした。 設定したホストが登録されていることと、DNS レコードが公開されるまでの時間があることを確認してください。
VERIFICATION_FAILED	AS2 メッセージ *message-ID* の署名検証に失敗したか、MIC コードが一致しませんでした。	送信者の署名用証明書がリモートプロファイルの署名用証明書と一致することを確認してください。また、MIC アルゴリズムが AWS Transfer Family と互換性があることも確認してください。

Code	エラー	説明と解決策
VERIFICATION_FAILED_NO_MATCHING_KEY_FOUND	• メッセージ署名と一致する公開証明書がプロファイル: *partner-profile-ID* に見つかりませんでした。 • 存在しないプロファイル: *partner-profile-ID* の証明書を取得できません。 • プロファイル: *partner-profile-ID* に有効な証明書が見つかりませんでした。	AWS Transfer Family が受信したメッセージの署名を検証しようとしていますが、パートナープロファイルに一致する署名用証明書が見つかりません。 解決オプション: • パートナープロフィールに署名用証明書が添付されていることを確認してください。 • 証明書が現在有効であることを確認してください。 • 証明書がパートナーの正しい署名用証明書であることを確認してください。

AWS Transfer Family マネージドワークフロー

AWS Transfer Family は、ファイル処理のマネージドワークフローをサポートします。管理されたワークフローを使えば、SFTP、FTPS、または FTP でファイルが転送された後に、ワークフローを開始することができます。この機能を使用すると、ファイル処理に必要なすべてのステップを調整することで、 business-to-business (B2B) ファイル交換のコンプライアンス要件を安全かつ費用対効果の高い方法で満たすことができます。さらに、 end-to-end 監査と可視性のメリットもあります。



マネージドワークフローは、ファイル処理タスクをオーケストレーションすることで、ダウンストリームのアプリケーションで消費される前にデータを前処理するのに役立ちます。このようなファイル処理タスクには以下が含まれる場合があります。

- ファイルをユーザー固有のフォルダーに移動します。
- ワークフローの一部としてファイルを復号化します。
- ファイルのタグ付け
- AWS Lambda 関数を作成してワークフローにアタッチすることで、カスタム処理を実行します。
- ファイルが正常に転送されたら通知を送信する。(このユースケースを詳しく説明するブログ記事については、AWS Transfer Family「マネージドワークフローを使用してファイル配信通知をカスタマイズする」を参照してください。)

組織内の複数の事業部門にまたがる一般的なアップロード後のファイル処理タスクを迅速に複製して標準化するには、Infrastructure as Code (IaC) を使用してワークフローを展開できます。完全にアップロードされたファイルに対して開始する管理ワークフローを指定できます。また、セッションが途中で切断されたために部分的にしかアップロードされないファイルに対して、別の管理ワークフローを開始するように指定することもできます。組み込みの例外処理機能により、ファイル処理の結果にすばやく対応できるとともに、失敗の処理方法を制御できます。さらに、ワークフローの各ステップでは詳細なログが生成され、それを監査してデータの系統を追跡できます。

始めるには、以下のタスクを実行する：

1. 要件に基づいて、コピー、タグ付け、およびその他のステップなどの前処理アクションを含めるようにワークフローを設定します。詳細については、「ワークフローを作成する」を参照してください。
2. Transfer Familyがワークフローを実行するための実行ロールを設定します。詳細については、「ワークフローの IAM ポリシー」を参照してください。
3. ワークフローをサーバーにマップすると、ファイルの到着時に、このワークフローで指定されたアクションがリアルタイムで評価され、開始されます。詳細については、「 ワークフローの設定と実行」を参照してください。

関連情報

- ワークフローの実行を監視するには、Transfer Family の CloudWatch メトリクスの使用を参照してください。
- 詳細な実行ログとトラブルシューティング情報については、Amazon を使用したワークフロー関連のエラーのトラブルシューティング CloudWatchを参照してください。
- Transfer Family は、ブログ記事と、ファイル転送ソリューションの構築を順を追って説明するワークショップを提供します。このソリューションは、マネージド SFTP/FTPS エンドポイントに 、ユーザー管理 AWS Transfer Family に Amazon Cognito と DynamoDB を活用します。

  ブログ記事は、「 AWS Transfer Family および Amazon S3 での ID プロバイダーとしての Amazon Cognito Amazon S3の使用」で入手できます。ワークショップの詳細はこちらで確認できます。
- Transfer Familyのワークフローについては、「AWS Transfer Family Managed Workflows」をご覧ください。

トピック

- ワークフローを作成する
- 事前定義されたステップを使用する
- カスタムのファイル処理ステップを使用してください。
- ワークフローの IAM ポリシー
- ワークフローの例外処理
- ワークフロー実行のモニタリング

- テンプレートからワークフローを作成する
- Transfer Family サーバーからワークフローを削除する
- マネージドワークフローの制限事項と機能制限

マネージドワークフローを開始する際のヒントについては、次のリソースを参照してください。

- AWS Transfer Family マネージドワークフローのデモビデオ
- AWS Transfer Family ワークフローを使用したクラウドネイティブなファイル転送プラットフォームの構築に関するブログ記事

ワークフローを作成する

このトピックで説明されているように AWS Management Console、 を使用してマネージドワークフローを作成できます。ワークフローの作成プロセスをできるだけ簡単にするために、コンソールのほとんどのセクションでコンテキストヘルプパネルを使用できます。

ワークフローには次の 2 種類のステップがあります。

- ノミナルステップ — ノミナルステップは、受信ファイルに適用したいファイル処理ステップです。ノミナルステップを複数選択した場合、各ステップは線形シーケンスで処理されます。
- 例外処理ステップ – 例外ハンドラーは、わずかなステップが失敗したり、検証エラーになったりした場合に AWS Transfer Family 実行されるファイル処理ステップです。

ワークフローを作成する

1. https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開きます。
2. ナビゲーションペインで [Workflows] (ワークフロー) を選択します。
3. [Workflows] (ワークフロー) ページで [Create workflow] (ワークフローの作成) を選択します。
4. 「ワークフローの作成」ページで、説明を入力します。この説明は [Workflows] (ワークフロー) ページに表示されます。
5. 「ノミナルステップ」セクションで「ステップを追加」を選択します。1 つ以上のステップを追加します。

    a. 使用可能なオプションからステップタイプを選択します。各種ステップの詳細については、the section called “事前定義されたステップを使用する” を参照してください。

b. [Next] (次へ) を選択してからステップのパラメータを設定します。
c. [Next] (次へ) を選択してからステップの詳細を見直します。
d. 「ステップの作成」を選択してステップを追加し、次に進みます。
e. 必要に応じて、ステップを追加し続けます。ワークフローの最大ステップ数は 8 です。
f. 必要なノミナルステップをすべて追加したら、「例外ハンドラー — オプション」セクションまでスクロールし、「ステップを追加」を選択します。

> **Note**
> リアルタイムで失敗を通知できるように、ワークフローが失敗したときに実行する例外ハンドラーとステップを設定することをお勧めします。

6. 例外ハンドラを設定するには、前述と同じ方法でステップを追加します。いずれかのステップでファイルに起因する例外が発生すると、例外ハンドラが 1 つずつ呼び出されます。
7. (オプション)「タグ」セクションまでスクロールダウンし、ワークフローのタグを追加します。
8. 設定を見直して [Create workflow] (ワークフローの作成) を選択します。

> **Important**
> ワークフローを作成した後は編集できないため、設定を注意深く確認してください。

ワークフローの設定と実行

ワークフローを実行する前に、そのワークフローを Transfer Family サーバーに関連付ける必要があります。

アップロードされたファイルに対してワークフローを実行するように Transfer Family を設定するには

1. https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開きます。
2. 左側のナビゲーションペインで [Servers] (サーバー) を選択します。

- ワークフローを既存のサーバーに追加するには、ワークフローで使用するサーバーを選択します。
- または、新しいサーバーを作成して、そのサーバーにワークフローを追加します。詳細については、「SFTP、FTPS、または FTP サーバーエンドポイントの設定」を参照してください。

3. サーバーの詳細ページで下にスクロールして [Additional details] (その他の詳細) セクションで [Edit] (編集) を選択します。

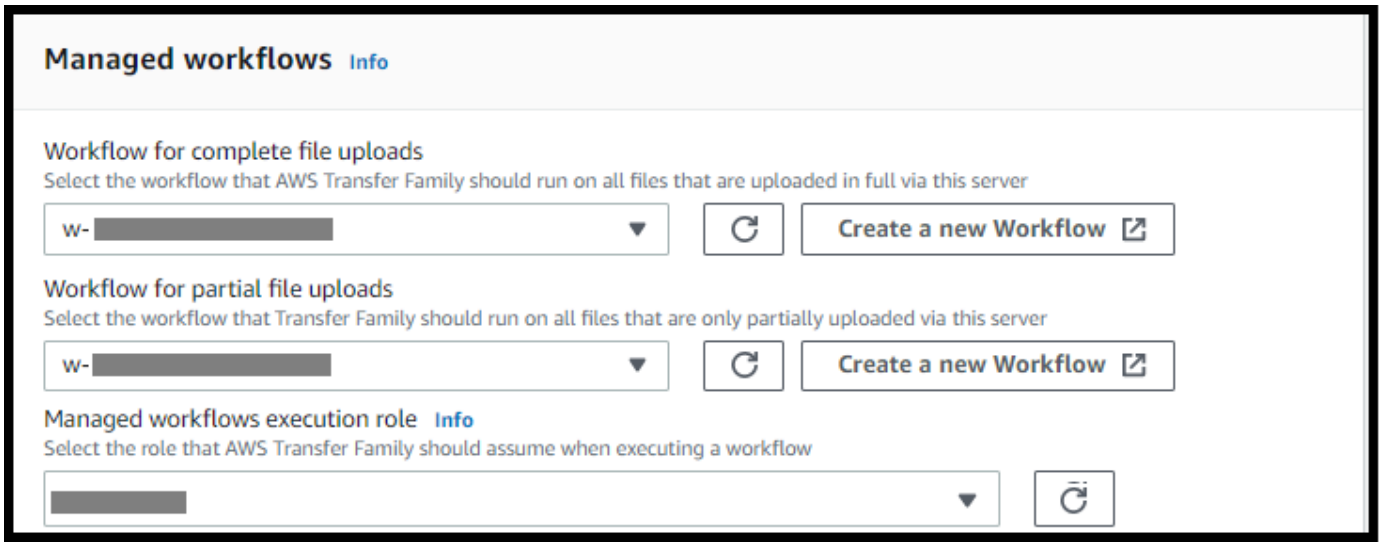
> **Note**
>
> デフォルトでは、サーバーに関連付けられたワークフローはありません。[Additional details] (その他の詳細) セクションを使用して、選択したサーバーにワークフローを関連付けます。

4. 「追加詳細の編集」ページの「管理ワークフロー」セクションで、すべてのアップロードに対して実行するワークフローを選択します。

> **Note**
>
> ワークフローがまだない場合は、「新しいワークフローを作成する」を選択し、ワークフローを作成します。

a. 使用するワークフロー ID を選択します。

b. 実行ロールを選択します。これは、Transfer Family がワークフローのステップを実行するときに引き受ける役割です。詳細については、「ワークフローの IAM ポリシー」を参照してください 保存を選択します。



> **Note**
>
> ワークフローをサーバーに関連付けたくない場合は、関連付けを削除できます。詳細については、「Transfer Family サーバーからワークフローを削除する」を参照してください。

「ワークフローを実行する」

ワークフローを実行するには、関連付けられたワークフローで設定した Transfer Family サーバーにファイルをアップロードします。

> **Note**
>
> サーバーからワークフローを削除し、新しいワークフローに置き換える場合、またはサーバー構成を更新する場合 (ワークフローの実行ロールに影響する) 、新しいワークフローを実行する前に約 10 分間待機する必要があります。Transfer Familyサーバーはワークフローの詳細をキャッシュし、サーバーがキャッシュを更新するのに10分かかります。
> さらに、アクティブなSFTPセッションからログアウトし、10分間の待ち時間の後にログインし直すと、変更を確認できます。

Example

```
# Execute a workflow
> sftp bob@s-1234567890abcdef0.server.transfer.us-east-1.amazonaws.com

Connected to s-1234567890abcdef0.server.transfer.us-east-1.amazonaws.com.
sftp> put doc1.pdf
Uploading doc1.pdf to /DOC-EXAMPLE-BUCKET/home/users/bob/doc1.pdf
doc1.pdf                                                                    100% 5013KB
 601.0KB/s   00:08
sftp> exit
>
```

ファイルがアップロードされると、定義されたアクションがファイルに対して実行されます。たとえば、ワークフローにコピーステップが含まれている場合、そのステップで定義した場所にファイルがコピーされます。Amazon CloudWatch Logs を使用して、実行されたステップとその実行ステータスを追跡できます。

ワークフロー詳細の表示

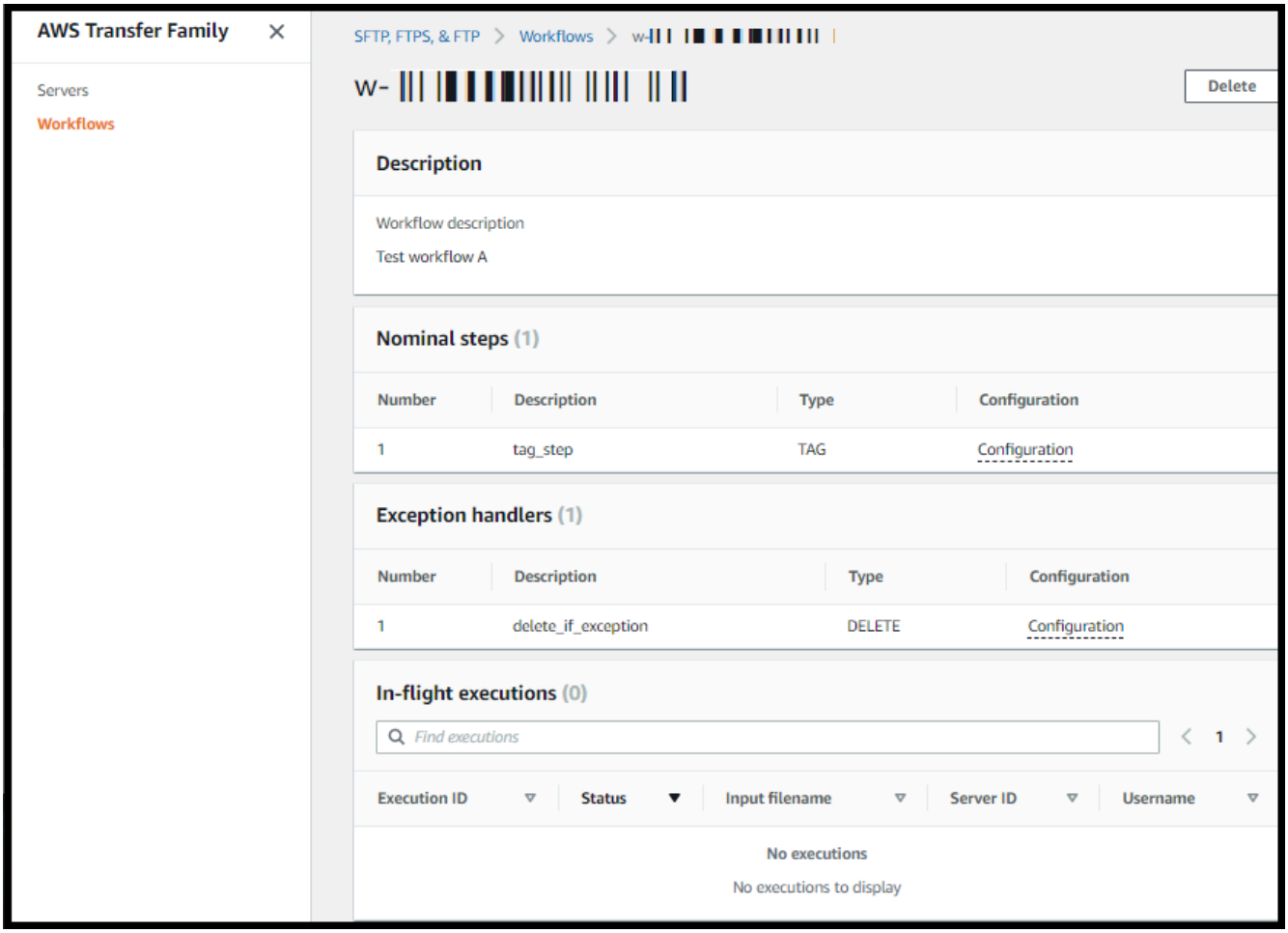
以前に作成したワークフローの詳細や、ワークフローの実行状況を見ることができます。これらの詳細を表示するには、 コンソールまたは AWS Command Line Interface () を使用しますAWS CLI。

Console

ワークフロー詳細の表示

1. https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開きます。
2. ナビゲーションペインで [Workflows] (ワークフロー) を選択します。
3. [Workflows] (ワークフロー) ページでワークフローを選択します。

ワークフローの詳細ページが開きます。

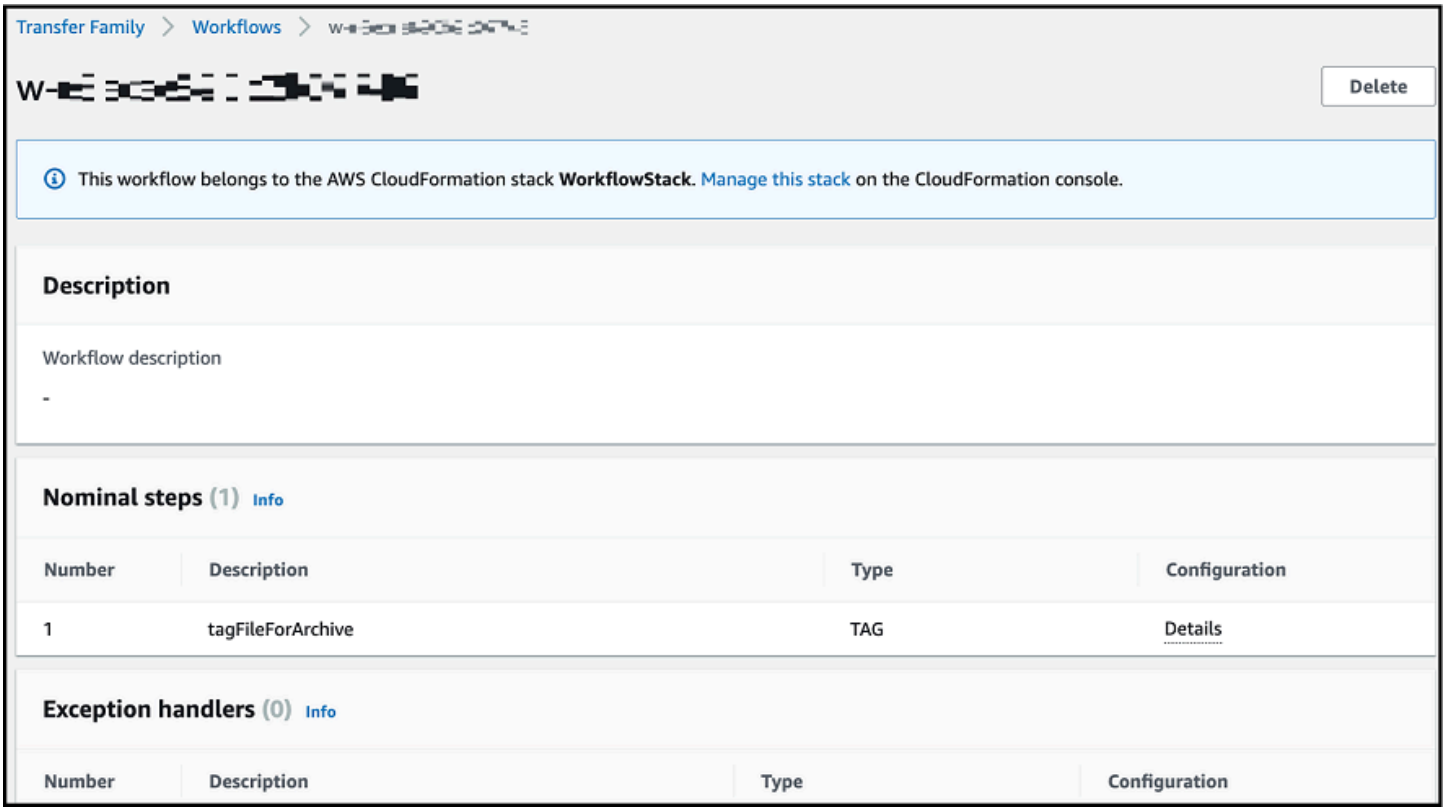


CLI

ワークフローの詳細を表示するには、次の例に示すように CLI で describe-workflow コマンドを使用します。ワークフロー ID *w-1234567890abcdef0* を独自の値に置き換えます。詳細については、「AWS CLI コマンドリファレンス」の「describe-workflow」を参照してください。

```
# View Workflow details
> aws transfer describe-workflow --workflow-id w-1234567890abcdef0
{
    "Workflow": {
        "Arn": "arn:aws:transfer:us-east-1:111122223333:workflow/
w-1234567890abcdef0",
        "WorkflowId": "w-1234567890abcdef0",
        "Name": "Copy file to shared_files",
        "Steps": [
            {
                "Type": "COPY",
                "CopyStepDetails": {
                "Name": "Copy to shared",
                "FileLocation": {
                    "S3FileLocation": {
                        "Bucket": "DOC-EXAMPLE-BUCKET",
                        "Key": "home/shared_files/"
                    }
                }
                }
            }
        ],
        "OnException": {}
    }
}
```

ワークフローが AWS CloudFormation スタックの一部として作成された場合は、 AWS CloudFormation コンソール (https://console.aws.amazon.com/cloudformation) を使用してワークフローを管理できます。



事前定義されたステップを使用する

ワークフローを作成する場合、このトピックで説明した以下の定義済みステップのいずれかを追加することを選択できます。独自のカスタムファイル処理ステップを追加することもできます。詳細については、「the section called “カスタムのファイル処理ステップを使用してください。”」を参照してください。

トピック

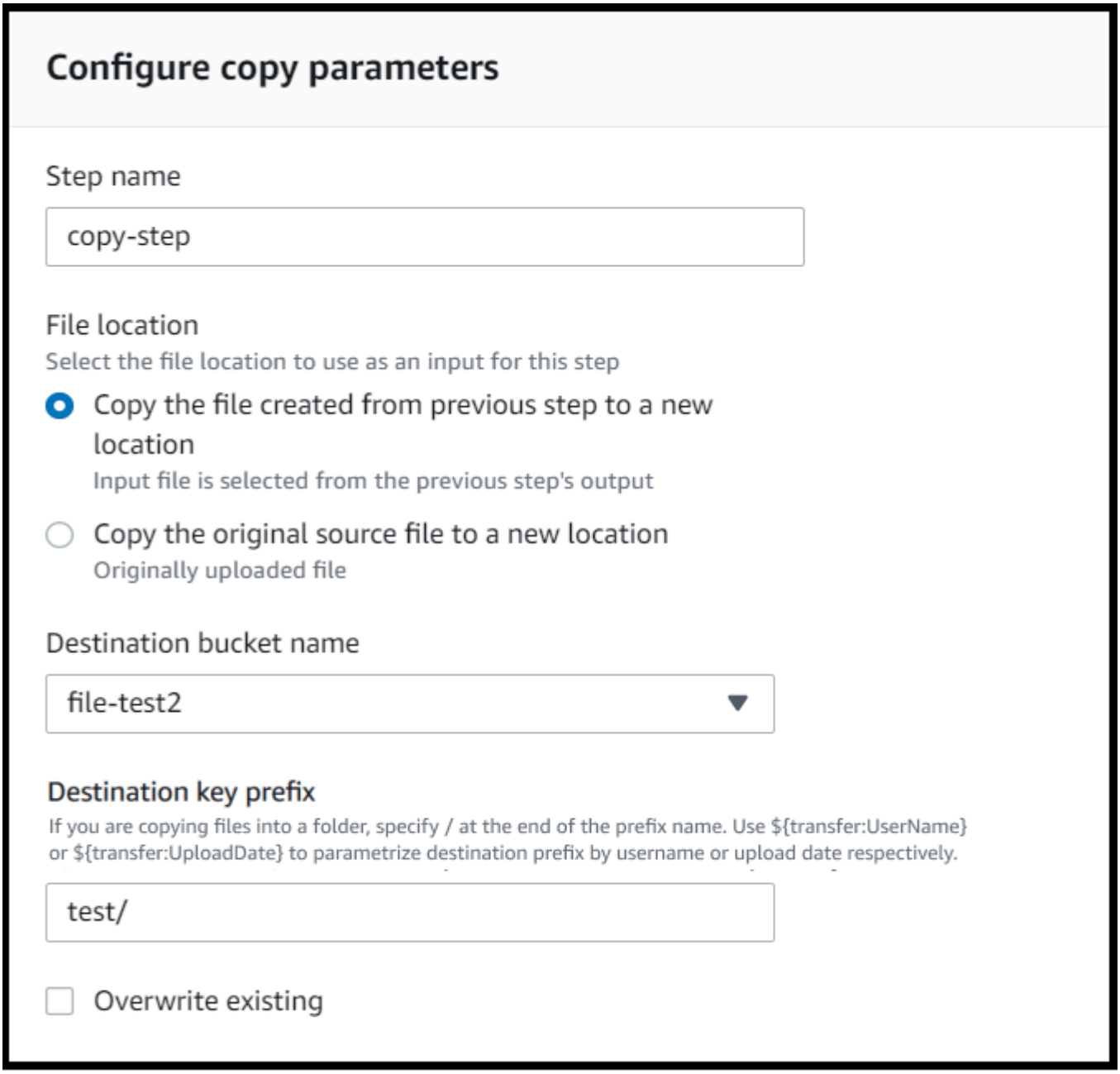
- ファイルをコピーする
- ファイルを復号化します。
- タグファイル
- ファイルを削除する
- ワークフローの名前付き変数
- タグ付けと削除のワークフローの例

ファイルをコピーする

ファイルのコピーステップでは、アップロードされたファイルのコピーを新しい Amazon S3 ロケーションに作成します。現在、ファイルコピーステップは Amazon S3 でのみ使用できます。

次のファイルコピーステップでは、ファイルを`file-test`宛先バケットの`test`フォルダにコピーします。

ファイルのコピーステップがワークフローの最初のステップでない場合は、「ファイルロケーション」を指定できます。ファイルの場所を指定することで、前のステップで使用したファイルまたはアップロードされた元のファイルのいずれかをコピーできます。この機能を使用すると、ファイルのアーカイブや記録の保持のためにソースファイルをそのまま保持したまま、元のファイルの複数のコピーを作成できます。例については、タグ付けと削除のワークフローの例を参照してください。



バケットとキーの詳細を提供する

ファイルのコピーステップのバケット名とキーを指定する必要があります。キーには、パス名またはファイル名を指定できます。キーがパス名またはファイル名のどちらとして扱われるかは、キーの末尾にスラッシュ (/) を付けるかどうかで決まります。

最後の文字が「/」の場合、ファイルはフォルダにコピーされ、その名前は変わりません。最後の文字が英数字の場合、アップロードしたファイルの名前が path 値に変更されます。その名前のファイルがすでに存在する場合、動作は「既存を上書き」フィールドの設定によって異なります。

- 「既存を上書き」を選択した場合、既存のファイルは処理中のファイルに置き換えられます。
- 「既存を上書き」が選択されていない場合は何も起こらず、ワークフロー処理は停止します。

> **Tip**
>
> 同じファイルパスで同時書き込みを実行すると、ファイルの上書き時に予期しない動作が発生する可能性があります。

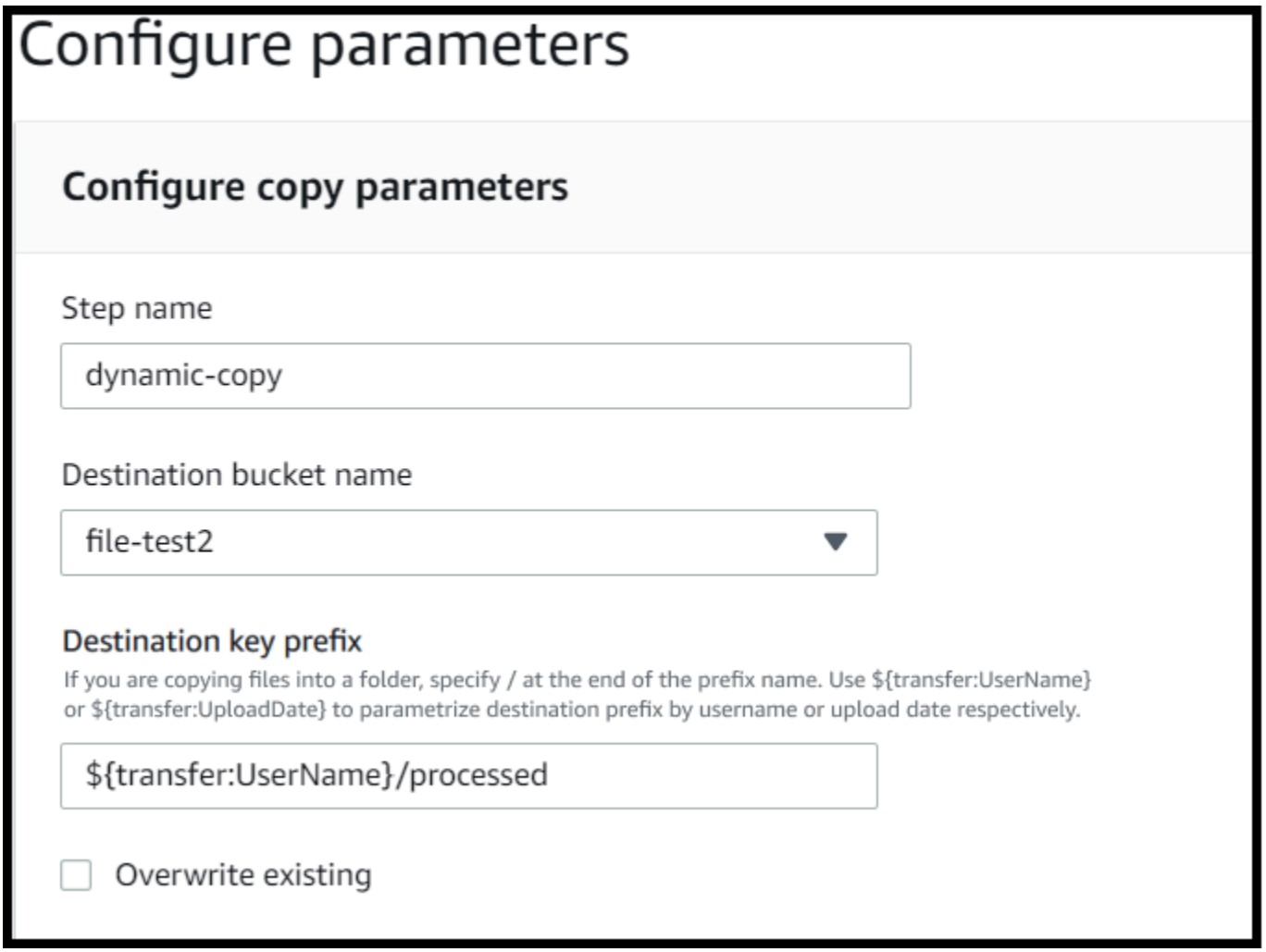
たとえば、キー値が `test/` であれば、アップロードされたファイルは `test` フォルダにコピーされます。キーの値が `test/today` で、「Overwrite existing」が選択されている場合、アップロードしたファイルはすべて `test` フォルダー内の `today` という名前のファイルにコピーされ、後続のファイルはそれぞれ前のファイルが上書きされます。

> **Note**
>
> Amazon S3 ではバケットとオブジェクトをサポートしており、階層はありません。しかし、オブジェクトキーの名前に接頭辞や区切り文字を使うことで、フォルダに似た方法で階層を暗示し、データを整理することができます。

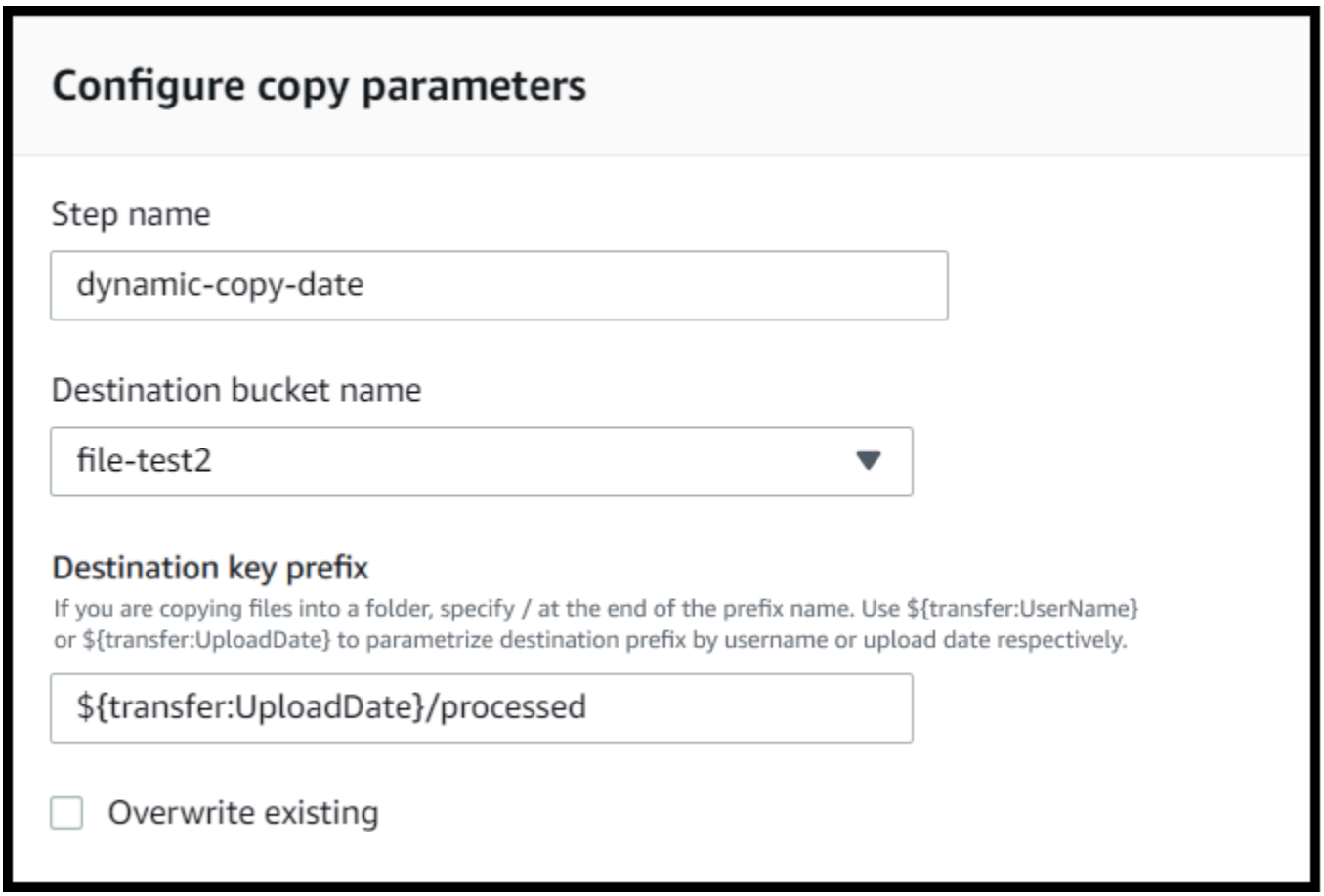
ファイルのコピーステップでは名前付き変数を使用します。

ファイルのコピーステップでは、変数を使用してファイルをユーザー固有のフォルダーに動的にコピーできます。現在、`${transfer:UserName}`または`${transfer:UploadDate}`を変数として使用して、ファイルをアップロードしている特定のユーザーの宛先に、または現在の日付に基づいてファイルをコピーできます。

次の例では、ユーザー`richard-roe`がファイルをアップロードすると、そのファイルは`file-test2/richard-roe/processed/`フォルダーにコピーされます。ユーザー`mary-major`がファイルをアップロードすると、そのファイルは`file-test2/mary-major/processed/`フォルダーにコピーされます。



同様に、`${transfer:UploadDate}`を変数として使って、現在の日付にちなんだ名前のコピー先にファイルをコピーすることができます。次の例では、コピー先を 2022 年 2 月 1 日の`${transfer:UploadDate}/processed`に設定すると、アップロードされたファイルは`file-test2/2022-02-01/processed/`フォルダーにコピーされます。



これらの変数は両方を組み合わせて使用することもできます。例:

- 「宛先キープレフィックス」を**folder/${transfer:UserName}/${transfer:UploadDate}/**に設定すると、folder/marymajor/2023-01-05/のように、ネストされたフォルダが作成されます。
- 「宛先キープレフィックス」を**folder/${transfer:UserName}-${transfer:UploadDate}/**にに設定すると、folder/marymajor-2023-01-05/のように、2 つの変数を連結できます。

コピーステップの IAM 権限

コピーステップを成功させるには、ワークフローの実行ロールに次の権限が含まれていることを確認してください。

```
{
            "Sid": "ListBucket",
            "Effect": "Allow",
```

```
            "Action": "s3:ListBucket",
            "Resource": [
                "arn:aws:s3:::destination-bucket-name"
            ]
        },
        {
            "Sid": "HomeDirObjectAccess",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:GetObject",
                "s3:DeleteObjectVersion",
                "s3:DeleteObject",
                "s3:GetObjectVersion"
            ],
            "Resource": "arn:aws:s3:::destination-bucket-name/*"
        }
```

> **Note**
>
> `s3:ListBucket`権限は、「既存を上書き」を選択しない場合にのみ必要です。この権限は、バケットをチェックして、同じ名前のファイルがすでに存在するかどうかを確認します。「既存を上書き」を選択した場合、ワークフローはファイルを確認する必要はなく、書き込むだけで済みます。
> Amazon S3 ファイルにタグがある場合は、IAM ポリシーに 1 つまたは 2 つのアクセス権限を追加する必要があります。
>
> - バージョン管理されていない Amazon S3 ファイルに`s3:GetObjectTagging`を追加します。
> - バージョン管理されている Amazon S3 ファイルに`s3:GetObjectVersionTagging`を追加します。

ファイルを復号化します。

AWS ストレージブログには、Transfer Family Managed ワークフロー、PGP および を使用したファイルの暗号化と復号化を使用してコードを記述せずにファイルを単純に復号 AWS Transfer Familyする方法を説明する投稿があります。

PGP 復号化をワークフローで使用してください。

Transfer Family には完璧なプライバシー (PGP) 復号化のサポートが組み込まれています。SFTP、FTPS、またはFTP経由でAmazon Simple Storage Service（Amazon S3）またはAmazon Elastic File System（Amazon EFS）にアップロードされたファイルでPGP復号化を使用できます。

PGP 復号化を使用するには、ファイルの復号に使用する PGP プライベートキーを作成して保存する必要があります。ユーザーは、Transfer Familyサーバーにファイルをアップロードする前に、対応するPGP暗号鍵を使用してファイルを暗号化することができます。暗号化されたファイルを受信したら、ワークフローでそれらのファイルを復号化できます。詳細なチュートリアルについては、「ファイルを復号するためのマネージドワークフローの設定」を参照してください。

PGP 復号化をワークフローで使用するには

1. ワークフローをホストする Transfer Family サーバーを特定するか、新しいサーバーを作成します。PGP キーを正しいシークレット名 AWS Secrets Manager で に保存する前に、サーバー ID が必要です。
2. PGP キーを必要なシークレット名 AWS Secrets Manager で に保存します。詳細については、「キーペアを管理する」を参照してください。ワークフローは、Secrets Manager のシークレット名に基づいて、復号化に使用する正しいPGPキーを自動的に見つけることができます。

   > **Note**
   >
   > Secrets Manager にシークレットを保存すると、 AWS アカウント に料金が発生します。料金については、「AWS Secrets Manager 料金表」を参照してください。

3. PGP キーペアを使用してファイルを暗号化します。（対応クライアントのリストは サポートされている PGP クライアント を参照のこと）。コマンドラインを使用している場合は、以下のコマンドを実行します。このコマンドを使用するには、*username@example.com*をPGPキー･ペアの作成に使用した電子メール･アドレスに置き換えます。*testfile.txt* を暗号化したいファイル名に置き換えます。

   ```
   gpg -e -r username@example.com testfile.txt
   ```

4. 暗号化されたファイルを Transfer Family サーバーにアップロードします。
5. ワークフローで復号化ステップを設定します。詳細については、「復号ステップを追加する」を参照してください。

復号ステップを追加する

復号化ステップでは、ワークフローの一部として Amazon S3 または Amazon EFS にアップロードされた暗号化されたファイルを復号化します。復号化の設定の詳細については、PGP 復号化をワークフローで使用してください。 を参照してください。

ワークフローの復号化ステップを作成するときは、復号化されたファイルの保存先を指定する必要があります。また、宛先にファイルがすでに存在する場合、既存のファイルを上書きするかどうかも選択する必要があります。Amazon CloudWatch Logs を使用すると、復号ワークフローの結果をモニタリングし、各ファイルの監査ログをリアルタイムで取得できます。

ステップの「Decrypt ファイル」タイプを選択すると、「パラメータの設定」ページが表示されます。「PGP 復号化パラメーターの設定」セクションに値を入力します。

利用可能なオプションは以下の通りである：

- ステップ名 — ステップの説明的な名前を入力します。
- ファイルロケーション — ファイルの場所を指定することで、前のステップで使用したファイルまたはアップロードされた元のファイルのいずれかを復号化できます。

> **Note**
>
> このパラメータは、このステップがワークフローの最初のステップである場合は使用できません。

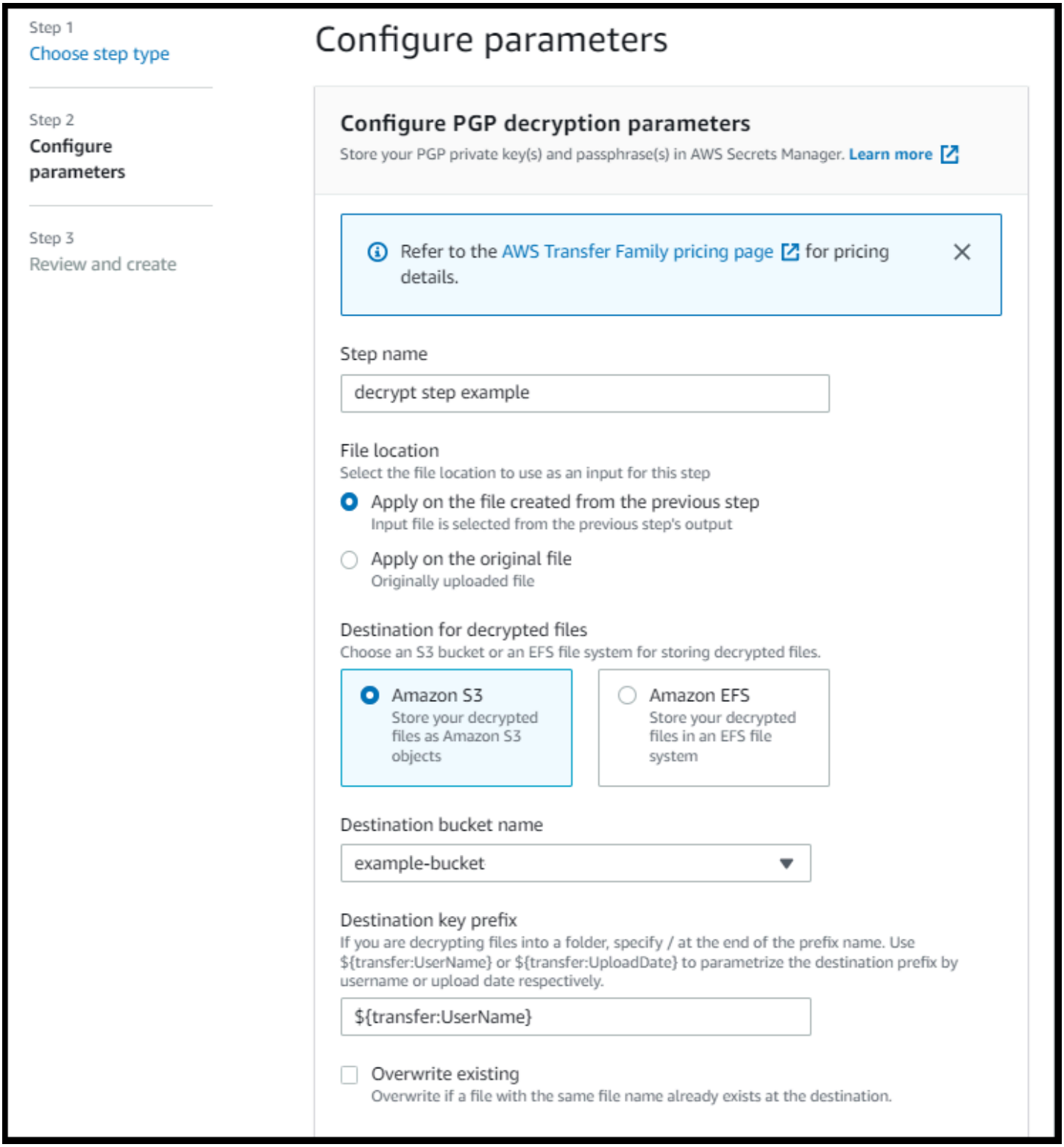
- 復号化されたファイルの宛先 — 復号化されたファイルの宛先として Amazon S3 バケットまたは Amazon EFS ファイルシステムを選択します。
  - Amazon S3 を選択した場合、宛先バケット名と宛先キープレフィックスを指定する必要があります。宛先キープレフィックスをユーザー名でパラメータ化するには、「宛先キープレフィックス」に**`${transfer:UserName}`**と入力します。同様に、アップロード日ごとに宛先キープレフィックスをパラメータ化するには、「宛先キープレフィックス」に**`${Transfer:UploadDate}`**と入力します。
  - Amazon EFS を選択する場合は、デスティネーションファイルシステムとパスを指定する必要があります。

> **Note**
>
> ここで選択するストレージオプションは、このワークフローが関連するTransfer Family サーバが使用するストレージシステムと一致する必要があります。作成されていない場合は、このワークフローを実行しようとしたときにエラーが発生します。

- 既存の上書き — ファイルをアップロードし、同じファイル名のファイルが宛先にすでに存在する場合、動作はこのパラメーターの設定によって異なります。
  - 「既存を上書き」を選択した場合、既存のファイルは処理中のファイルに置き換えられます。
  - 「既存を上書き」が選択されていない場合は何も起こらず、ワークフロー処理は停止します。

> **Tip**
>
> 同じファイルパスで同時書き込みを実行すると、ファイルの上書き時に予期しない動作が発生する可能性があります。

次のスクリーンショットは、ファイルの復号化ステップで選択できるオプションの例を示しています。


Step 1
Choose step type
Step 2
Configure parameters
Step 3
Review and create
Configure parameters
Configure PGP decryption parameters
Store your PGP private key(s) and passphrase(s) in AWS Secrets Manager. Learn more
Refer to the AWS Transfer Family pricing page for pricing details.
Step name
decrypt step example
File location
Select the file location to use as an input for this step
Apply on the file created from the previous step
Input file is selected from the previous step's output
Apply on the original file
Originally uploaded file
Destination for decrypted files
Choose an S3 bucket or an EFS file system for storing decrypted files.
Amazon S3
Store your decrypted files as Amazon S3 objects
Amazon EFS
Store your decrypted files in an EFS file system
Destination bucket name
example-bucket
Destination key prefix
If you are decrypting files into a folder, specify / at the end of the prefix name. Use ${transfer:UserName} or ${transfer:UploadDate} to parametrize the destination prefix by username or upload date respectively.
${transfer:UserName}
Overwrite existing
Overwrite if a file with the same file name already exists at the destination.

復号化ステップの IAM 権限

復号化ステップを成功させるには、ワークフローの実行ロールに次の権限が含まれていることを確認してください。

```
{
            "Sid": "ListBucket",
            "Effect": "Allow",
            "Action": "s3:ListBucket",
            "Resource": [
                "arn:aws:s3:::destination-bucket-name"
            ]
        },
        {
            "Sid": "HomeDirObjectAccess",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:GetObject",
                "s3:DeleteObjectVersion",
                "s3:DeleteObject",
                "s3:GetObjectVersion"
            ],
            "Resource": "arn:aws:s3:::destination-bucket-name/*"
        },
        {
            "Sid": "Decrypt",
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetSecretValue",
            ],
            "Resource": "arn:aws:secretsmanager:region:account-id:secret:aws/transfer/
*"
        }
```

> **Note**
>
> s3:ListBucket権限は、「既存を上書き」を選択しない場合にのみ必要です。この権限は、バケットをチェックして、同じ名前のファイルがすでに存在するかどうかを確認します。「既存を上書き」を選択した場合、ワークフローはファイルを確認する必要はなく、書き込むだけで済みます。

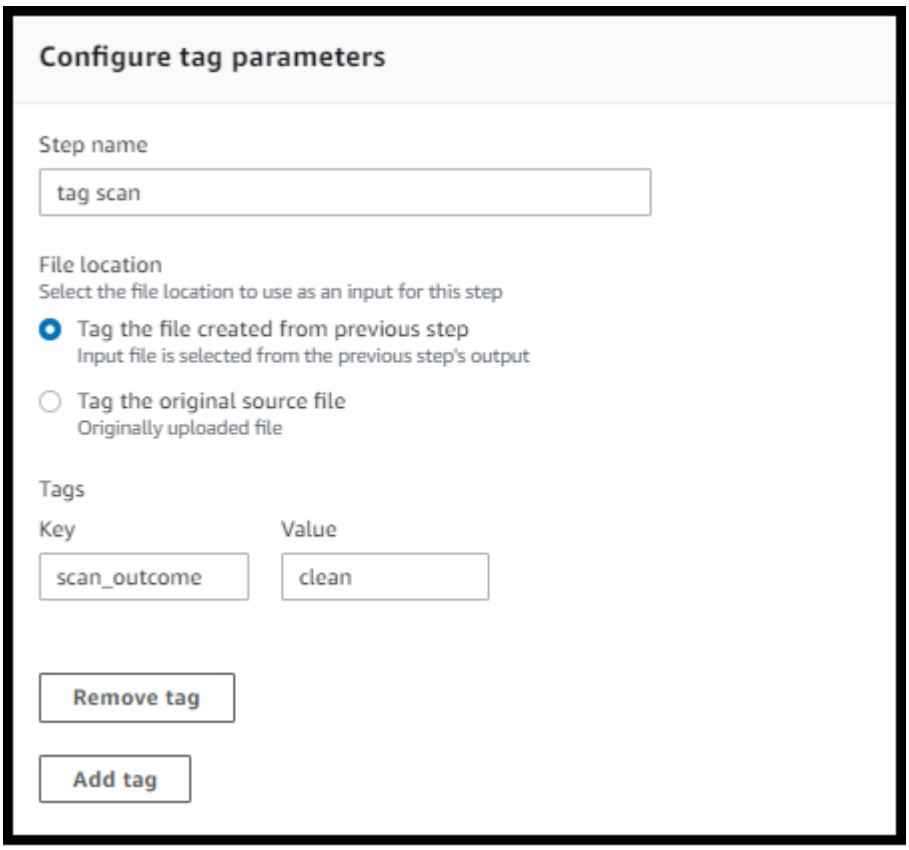
Amazon S3 ファイルにタグがある場合は、IAM ポリシーに 1 つまたは 2 つのアクセス権限を追加する必要があります。

- バージョン管理されていない Amazon S3 ファイルに`s3:GetObjectTagging`を追加します。
- バージョン管理されている Amazon S3 ファイルに`s3:GetObjectVersionTagging`を追加します。

タグファイル

受信ファイルにタグを付けてさらに下流処理を行うには、タグステップを使用します。受信ファイルに割り当てるタグの値を入力します。現在、タグオペレーションは Transfer Family サーバーストレージに Amazon S3 を使用している場合にのみサポートされます。

次のタグステップ例では、`scan_outcome`と`clean`をそれぞれタグキーと値として割り当てます。



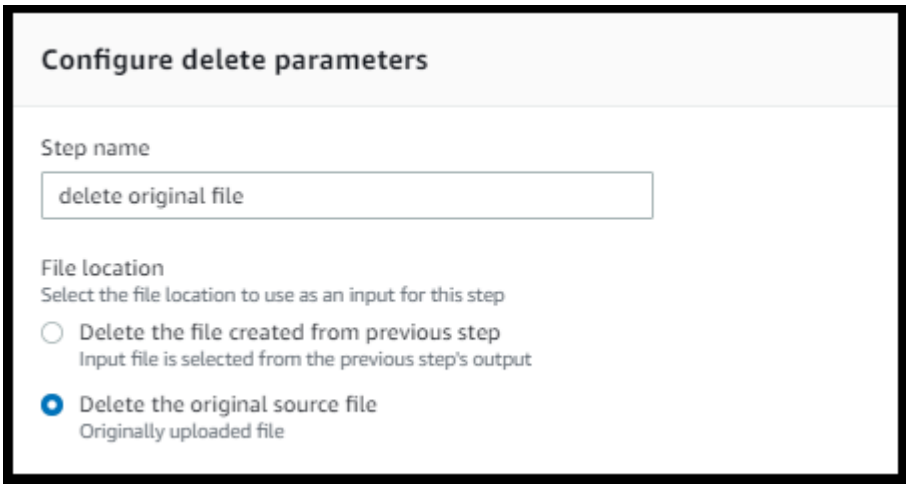
タグステップを成功させるには、ワークフローの実行ロールに次の権限が含まれていることを確認してください。

```
{
            "Sid": "Tag",
            "Effect": "Allow",
            "Action": [
                "s3:PutObjectTagging",
                "s3:PutObjectVersionTagging"
            ],
            "Resource": [
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
            ]
}
```

> **Note**
>
> ワークフローに、コピーまたは復号化ステップの前に実行されるタグステップが含まれている場合は、IAM ポリシーに 1 つまたは 2 つのアクセス権限を追加する必要があります。
>
> - バージョン管理されていない Amazon S3 ファイルに`s3:GetObjectTagging`を追加します。
> - バージョン管理されている Amazon S3 ファイルに`s3:GetObjectVersionTagging`を追加します。

ファイルを削除する

処理済みのファイルを前のワークフローステップから削除したり、最初にアップロードしたファイルを削除したりするには、ファイル削除ステップを使用します。



削除ステップを正常に実行するには、ワークフローの実行ロールに次の権限が含まれていることを確認してください。

```
{
            "Sid": "Delete",
            "Effect": "Allow",
            "Action": [
                "s3:DeleteObjectVersion",
                "s3:DeleteObject"
            ],
            "Resource": "arn:aws:secretsmanager:region:account-ID:secret:aws/transfer/
*"
        }
```

ワークフローの名前付き変数

コピーと復号化のステップでは、変数を使用してアクションを動的に実行できます。現在、 は次の名前付き変数 AWS Transfer Family をサポートしています。

- `${transfer:UserName}`を使用して、ファイルをアップロードしているユーザーに基づいて、ファイルを宛先にコピーまたは復号化します。
- `${transfer:UploadDate}`を使用して、現在の日付に基づいて、ファイルを宛先にロケーションにコピーまたは復号化します。

タグ付けと削除のワークフローの例

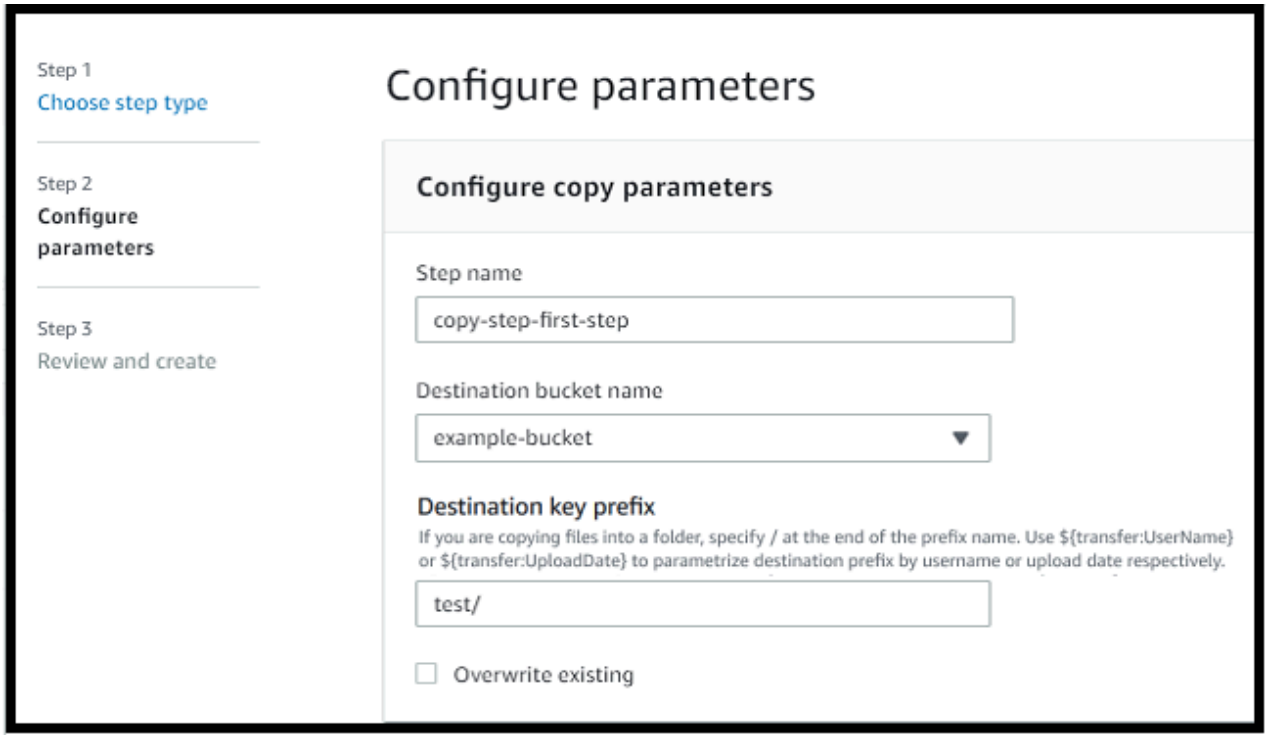
次の例は、データ分析プラットフォームなどのダウンストリームアプリケーションで処理する必要がある受信ファイルにタグを付けるワークフローを示しています。受信ファイルにタグを付けた後、ワークフローはストレージコストを節約するために最初にアップロードされたファイルを削除します。

Console

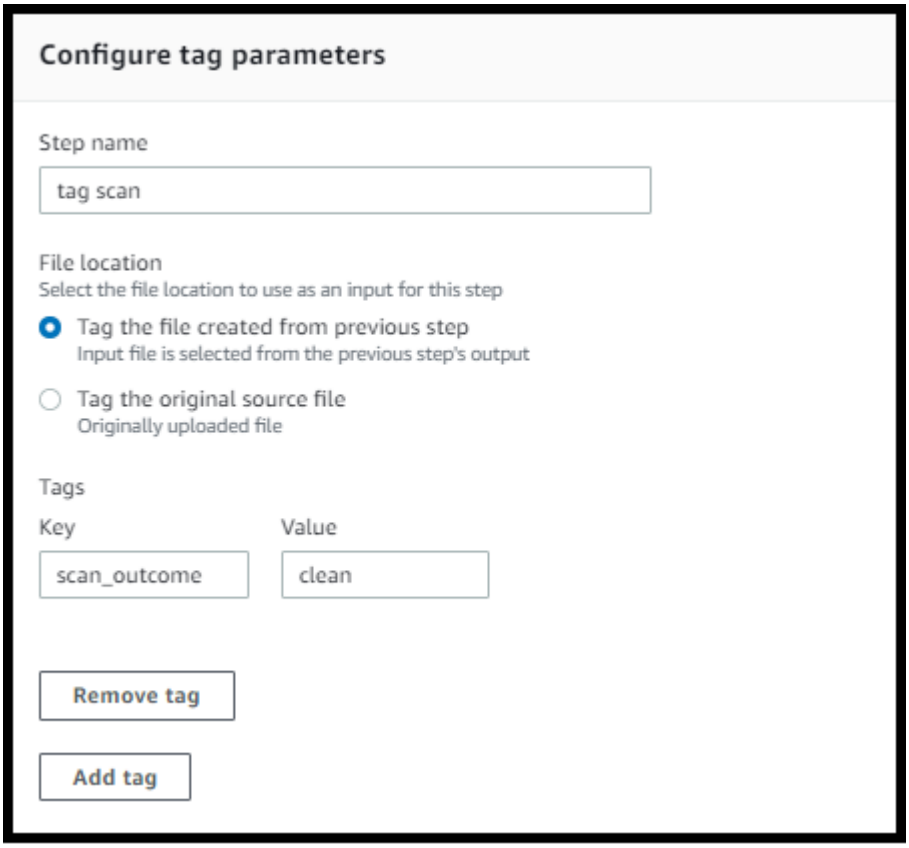
例:タグ付けして移動するワークフロー

1. https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開きます。
2. ナビゲーションペインで [Workflows] (ワークフロー) を選択します。
3. [Workflows] (ワークフロー) ページで [Create workflow] (ワークフローの作成) を選択します。

4. 「ワークフローの作成」ページで、説明を入力します。この説明は [Workflows] (ワークフロー) ページに表示されます。
5. 最初のステップ (コピー) を追加します。
   a. 「ノミナルステップ」セクションで「ステップを追加」を選択します。
   b. 「ファイルのコピー」を選択し、「次へ」を選択します。
   c. ステップ名を入力し、宛先バケットとkey prefix スを選択します。



   d. [Next] (次へ) を選択してからステップの詳細を見直します。
   e. 「ステップの作成」を選択してステップを追加し、次に進みます。
6. 2 番目のステップ (タグ) を追加します。
   a. 「ノミナルステップ」セクションで「ステップを追加」を選択します。
   b. 「ファイルのタグ付け」を選択し、「次へ」を選択します。
   c. ステップ名を入力します。
   d. 「ファイルロケーション」で、「前の手順で作成したファイルにタグを付ける」を選択します。
   e. [Key] (キー) と [Value] (値) を入力します。



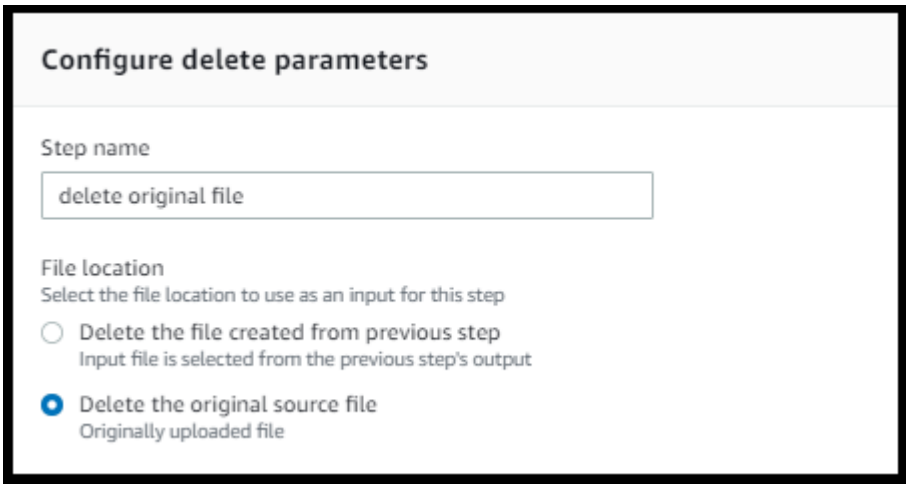
f. [Next] (次へ) を選択してからステップの詳細を見直します。

g. 「ステップの作成」を選択してステップを追加し、次に進みます。

7. 3 番目のステップ (削除) を追加します。

a. 「ノミナルステップ」セクションで「ステップを追加」を選択します。

b. 「ファイルの削除」を選択し、「次へ」を選択します。



c. ステップ名を入力します。

d. 「ファイルロケーション」で「元のソースファイルを削除」を選択します。

e. [Next] (次へ) を選択してからステップの詳細を見直します。

f. 「ステップの作成」を選択してステップを追加し、次に進みます。

8. ワークフロー設定を確認し、し、「ワークフローの作成」を選択します。

CLI

例:タグ付けして移動するワークフロー

1. 以下のコードをファイルに保存します；例えば、tagAndMoveWorkflow.json。*user input placeholder* を、ユーザー自身の情報に置き換えます。

```
[
   {
       "Type": "COPY",
       "CopyStepDetails": {
          "Name": "CopyStep",
          "DestinationFileLocation": {
             "S3FileLocation": {
                "Bucket": "DOC-EXAMPLE-BUCKET",
                "Key": "test/"
             }
          }
       }
   },
   {
       "Type": "TAG",
       "TagStepDetails": {
          "Name": "TagStep",
          "Tags": [
             {
                "Key": "name",
                "Value": "demo"
             }
          ],
          "SourceFileLocation": "${previous.file}"
       }
   },
   {
      "Type": "DELETE",
      "DeleteStepDetails":{
```

```
        "Name":"DeleteStep",
        "SourceFileLocation": "${original.file}"
      }
  }
]
```

最初のステップでは、アップロードしたファイルを新しい Amazon S3 ロケーションにコピーします。2 番目のステップでは、新しいロケーションにコピーされたファイル (previous.file) にタグ (キーと値のペア) を追加します。最後に、3 番目のステップで元のファイル (original.file) を削除します。

2. 保存したファイルからワークフローを作成します。*user input placeholder* を、ユーザー自身の情報に置き換えます。

```
aws transfer create-workflow --description "short-description" --steps
 file://path-to-file --region region-ID
```

例:

```
aws transfer create-workflow --description "copy-tag-delete workflow" --steps
 file://tagAndMoveWorkflow.json --region us-east-1
```

> **Note**
>
> ファイルを使用してパラメータを読み込む方法について詳しくは、「ファイルからパラメータをロードする方法」を参照してください。

3. 既存のサーバーを更新します。

> **Note**
>
> この手順では、すでに Transfer Family サーバーがあり、それにワークフローを関連付けることを前提としています。そうでない場合は、「SFTP、FTPS、または FTP サーバーエンドポイントの設定」を参照してください。*user input placeholder* を、ユーザー自身の情報に置き換えます。

```
aws transfer update-server --server-id server-ID --region region-ID
```

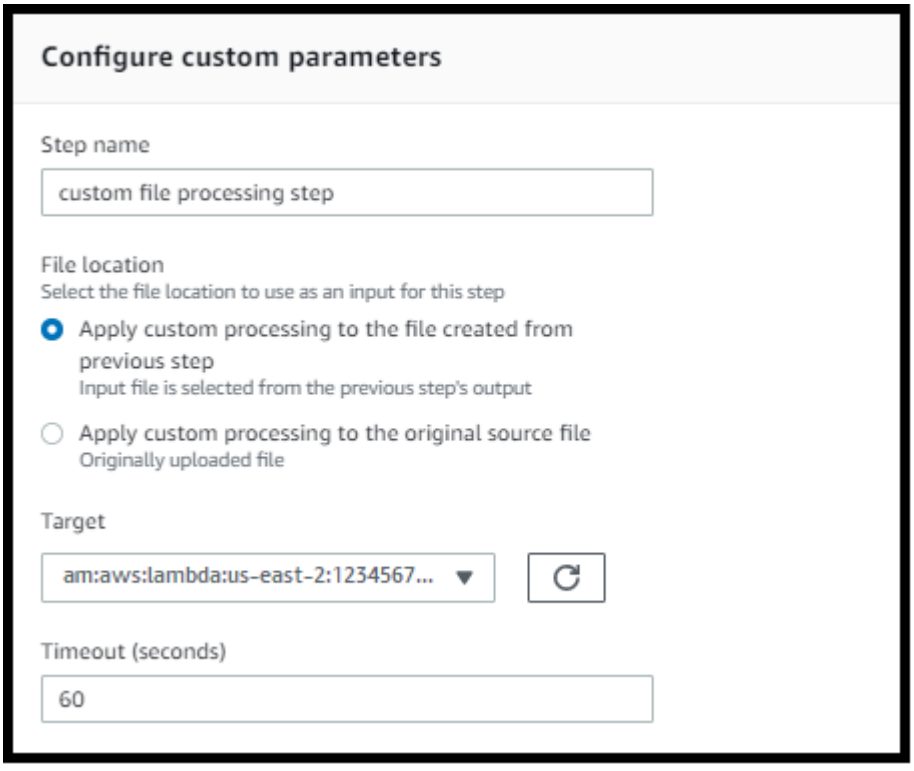
```
  --workflow-details '{"OnUpload":[{ "WorkflowId": "workflow-
ID","ExecutionRole": "execution-role-ARN"}]}'
```

例:

```
aws transfer update-server --server-id s-1234567890abcdef0 --region us-east-2
  --workflow-details '{"OnUpload":[{ "WorkflowId": "w-
abcdef01234567890","ExecutionRole": "arn:aws:iam::111111111111:role/nikki-wolf-
execution-role"}]}'
```

カスタムのファイル処理ステップを使用してください。

カスタムファイル処理ステップを使用することで、 AWS Lambdaを使用して独自のファイル処理ロジックを実現できます。ファイルが到着すると、Transfer Family サーバーは、ファイルの暗号化、マルウェアのスキャン、不正なファイルタイプのチェックなど、カスタムファイル処理ロジックを含む Lambda 関数を呼び出します。次の例では、ターゲット AWS Lambda 関数が前のステップの出力ファイルを処理するのに使われています。



> **Note**
>
> サンプルの Lambda 関数については、「カスタムワークフローステップの Lambda 関数の例」を参照してください。イベント (Lambda に渡されるファイルの場所を含む) の例については、ファイルのアップロード AWS Lambda 時に に送信されるイベントの例を参照してください。

カスタムワークフローステップでは、 SendWorkflowStepState API オペレーションを呼び出すように Lambda 関数を設定する必要があります。 は、ステップが完了したことをワークフロー実行に成功ステータスまたは失敗ステータスでSendWorkflowStepState通知します。SendWorkflowStepStateAPI オペレーションのステータスにより、Lambda 関数の結果に基づいて、例外ハンドラーステップまたは線形シーケンスのノミナルステップが呼び出されます。

Lambda 関数が失敗またはタイムアウトすると、ステップは失敗し、 CloudWatch ログStepErroredに が表示されます。Lambda 関数がノミナルステップの一部であり、その関数がSendWorkflowStepStateにStatus="FAILURE"で応答するかタイムアウトした場合、フローは例外ハンドラステップに進みます。この場合、ワークフローは残りの (もしあれば) 名目上のステップを実行し続けません。詳細については、「ワークフローの例外処理」を参照してください。

SendWorkflowStepState API オペレーションを呼び出す際には、以下のパラメーターを送信する必要があります。

```
{
    "ExecutionId": "string",
    "Status": "string",
    "Token": "string",
    "WorkflowId": "string"
}
```

Lambda 関数実行時に渡される入力イベントから、ExecutionId、Token、WorkflowId を抽出することができます (以下のセクションに例を示す)。Status 値は SUCCESS または FAILURE のいずれかです。

Lambda 関数から SendWorkflowStepState API オペレーションを呼び出すには、 マネージドワークフローが導入された後に発行された AWS SDK のバージョンを使用する必要があります。 ???

複数の Lambda 関数を続けて使用する

複数のカスタムステップを次々に使用する場合、「ファイルロケーション」オプションは 1 つのカスタムステップのみを使用する場合とは動作が異なります。Transfer Family は、Lambda で処理されたファイルを戻して次のステップの入力として使用することをサポートしていません。そのため、previous.fileオプションを使用するように構成された複数のカスタム・ステップがある場合、それらはすべて同じファイルの場所 (最初のカスタム・ステップの入力ファイルの場所) を使用します。

> **Note**
>
> カスタムステップの後に定義済みのステップ (タグ付け、コピー、復号化、または削除) がある場合も、previous.file設定の動作は異なります。定義済みステップがprevious.file設定を使用するように構成されている場合、定義済みステップはカスタム・ステップで使用されるのと同じ入力ファイルを使用します。カスタムステップで処理されたファイルは、定義済みのステップには渡されません。

カスタム処理後のファイルへのアクセス

Amazon S3 をストレージとして使用していて、ワークフローに最初にアップロードされたファイルに対してアクションを実行するカスタムステップが含まれている場合、以降のステップでは処理されたファイルにアクセスできません。つまり、カスタムステップの後のどのステップも、カスタムステップの出力から更新されたファイルを参照することはできません。

例えば、ワークフローに次の 3 つがあるとします。

- ステップ 1 — example-file.txt という名前のファイルをアップロードします。
- ステップ 2 — example-file.txtを何らかの方法で変更する Lambda 関数を呼び出します。
- ステップ 3 — 更新されたバージョンのexample-file.txtに対して、さらなる処理を試みます。

ステップ 3 のsourceFileLocationを${original.file}として設定した場合、ステップ 3 では、でサーバがステップ 1 ファイルをストレージにアップロードしたときの元のファイルロケーションが使用されます。ステップ 3 で${previous.file}を使用している場合、ステップ 3 はステップ 2 が入力として使用したファイルの場所を再利用します。

そのため、ステップ 3 ではエラーが発生します。例えば、ステップ 3 で更新されたexample-file.txtをコピーしようとすると、以下のエラーが発生します。

```
{
    "type": "StepErrored",
    "details": {
        "errorType": "NOT_FOUND",
        "errorMessage": "ETag constraint not met (Service: null; Status Code: 412;
 Error Code: null; Request ID: null; S3 Extended Request ID: null; Proxy: null)",
        "stepType": "COPY",
        "stepName": "CopyFile"
    },
```

このエラーは、カスタム・ステップでexample-file.txt用のエンティティ・タグ（ETag）が変更され、元のファイルと一致しなくなるために発生します。

> **Note**
>
> Amazon EFS ではファイルの識別にエンティティタグを使用しないため、Amazon EFS を使用している場合はこの動作は発生しません。

ファイルのアップロード AWS Lambda 時に に送信されるイベントの例

次の例は、ファイルのアップロードが完了した AWS Lambda ときに に送信されるイベントを示しています。ある例では、ドメインが Amazon S3 で構成されている Transfer Family サーバーを使用しています。もう 1 つは Transfer Family サーバーでドメインで Amazon EFS を使用する例です。

Custom step that uses an Amazon S3 domain

```
{
    "token": "MzI0Nzc4ZDktMGRmMi00MjFhLTgxMjUtYWZmZmRmODNkYjc0",
    "serviceMetadata": {
        "executionDetails": {
            "workflowId": "w-1234567890example",
            "executionId": "abcd1234-aa11-bb22-cc33-abcdef123456"
        },
        "transferDetails": {
            "sessionId": "36688ff5d2deda8c",
            "userName": "myuser",
            "serverId": "s-example1234567890"
        }
    },
    "fileLocation": {
```

```
        "domain": "S3",
        "bucket": "DOC-EXAMPLE-BUCKET",
        "key": "path/to/mykey",
        "eTag": "d8e8fca2dc0f896fd7cb4cb0031ba249",
        "versionId": null
    }
}
```

Custom step that uses an Amazon EFS domain

```
{
    "token": "MTg0N2Y3N2UtNWI5Ny00ZmZlLTk5YTgtZTU3YzViYjllNmZm",
    "serviceMetadata": {
        "executionDetails": {
            "workflowId": "w-1234567890example",
            "executionId": "abcd1234-aa11-bb22-cc33-abcdef123456"
        },
        "transferDetails": {
            "sessionId": "36688ff5d2deda8c",
            "userName": "myuser",
            "serverId": "s-example1234567890"
        }
    },
    "fileLocation": {
        "domain": "EFS",
        "fileSystemId": "fs-1234567",
        "path": "/path/to/myfile"
    }
}
```

カスタムワークフローステップの Lambda 関数の例

次の Lambda 関数は、実行ステータスに関する情報を抽出し、 SendWorkflowStepState API オペレーションを呼び出して、 SUCCESSまたは のいずれかのステップのワークフローにステータスを返しますFAILURE。関数がSendWorkflowStepState API オペレーションを呼び出す前に、ワークフローロジックに基づいたアクションを取るように Lambda を設定できます。

```
import json
import boto3

transfer = boto3.client('transfer')
```

```
def lambda_handler(event, context):
    print(json.dumps(event))

    # call the SendWorkflowStepState API to notify the workflow about the step's
 SUCCESS or FAILURE status
    response = transfer.send_workflow_step_state(
        WorkflowId=event['serviceMetadata']['executionDetails']['workflowId'],
        ExecutionId=event['serviceMetadata']['executionDetails']['executionId'],
        Token=event['token'],
        Status='SUCCESS|FAILURE'
    )

    print(json.dumps(response))

    return {
      'statusCode': 200,
      'body': json.dumps(response)
    }
```

カスタムステップの IAM 権限

Lambda を呼び出すステップを成功させるには、ワークフローの実行ロールに次の権限が含まれていることを確認してください。

```
{
            "Sid": "Custom",
            "Effect": "Allow",
            "Action": [
                "lambda:InvokeFunction"
            ],
            "Resource": [
                "arn:aws:lambda:region:account-id:function:function-name"
            ]
        }
```

ワークフローの IAM ポリシー

ワークフローをサーバーに追加する際、実行ロールを選択する必要があります。サーバーは、ワークフローの実行時にこのロールを使用します。ロールに適切なアクセス許可がない場合は、ワークフローを実行 AWS Transfer Family できません。

このセクションでは、ワークフローの実行に使用できる AWS Identity and Access Management (IAM) アクセス許可の 1 つのセットについて説明します。このトピックの後半で、他の例を紹介しています。

> **Note**
>
> Amazon S3 ファイルにタグがある場合は、IAM ポリシーに 1 つまたは 2 つのアクセス権限を追加する必要があります。
>
> - バージョン管理されていない Amazon S3 ファイルに`s3:GetObjectTagging`を追加します。
> - バージョン管理されている Amazon S3 ファイルに`s3:GetObjectVersionTagging`を追加します。

ワークフローの実行ロールを作成するには

1. 新しい IAM ロールを作成し、 AWS マネージドポリシー`AWSTransferFullAccess`をロールに追加します。新しい IAM ロールの作成についての詳細は、the section called “IAM ポリシーとロールを作成する” を参照してください。
2. 次のアクセス許可を持つポリシーを作成してロールにアタッチします。*`user input placeholder`* を、ユーザー自身の情報に置き換えます。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ConsoleAccess",
            "Effect": "Allow",
            "Action": "s3:GetBucketLocation",
            "Resource": "*"
        },
        {
            "Sid": "ListObjectsInBucket",
            "Effect": "Allow",
            "Action": "s3:ListBucket",
            "Resource": [
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET"
            ]
        },
```

```
        {
            "Sid": "AllObjectActions",
            "Effect": "Allow",
            "Action": "s3:*Object",
            "Resource": [
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
            ]
        },
        {
            "Sid": "GetObjectVersion",
            "Effect": "Allow",
            "Action": "s3:GetObjectVersion",
            "Resource": [
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
            ]
        },
        {
            "Sid": "Custom",
            "Effect": "Allow",
            "Action": [
                "lambda:InvokeFunction"
            ],
            "Resource": [
                "arn:aws:lambda:region:account-id:function:function-name"
            ]
        },
        {
            "Sid": "Tag",
            "Effect": "Allow",
            "Action": [
                "s3:PutObjectTagging",
                "s3:PutObjectVersionTagging"
            ],
            "Resource": [
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
            ]
        }
    ]
}
```

3. このロールを保存し、ワークフローをサーバーに追加する際に実行ロールとして指定します。

> **Note**
>
> IAM ロールを構築する場合、 では、ワークフローに対して可能な限りリソースへのアクセスを制限することを AWS 推奨しています。

ワークフローの信頼関係

ワークフロー実行ロールには、`transfer.amazonaws.com`との信頼関係も必要です。 AWS Transfer Familyとの信頼関係を築くには、信頼関係を確立するには を参照してください。

信頼関係を築いている間は、「混乱した代理人」問題を避けるための対策を講じることもできます。この問題の説明と回避方法の例については、the section called “サービス間の混乱した代理の防止”を参照してください。

実行ロールの例:復号化、コピー、タグ付け

タグ付け、コピー、復号化のステップを含むワークフローがある場合は、次の IAM ポリシーを使用できます。*`user input placeholder`* を、ユーザー自身の情報に置き換えます。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "CopyRead",
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:GetObjectTagging",
                "s3:GetObjectVersionTagging"
            ],
            "Resource": "arn:aws:s3:::source-bucket-name/*"
        },
        {
            "Sid": "CopyWrite",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:PutObjectTagging"
            ],
            "Resource": "arn:aws:s3:::destination-bucket-name/*"
```

```
        },
        {
            "Sid": "CopyList",
            "Effect": "Allow",
            "Action": "s3:ListBucket",
            "Resource": [
                "arn:aws:s3:::source-bucket-name",
                "arn:aws:s3:::destination-bucket-name"
            ]
        },
        {
            "Sid": "Tag",
            "Effect": "Allow",
            "Action": [
                "s3:PutObjectTagging",
                "s3:PutObjectVersionTagging"
            ],
            "Resource": "arn:aws:s3:::destination-bucket-name/*",
            "Condition": {
                "StringEquals": {
                    "s3:RequestObjectTag/Archive": "yes"
                }
            }
        },
        {
            "Sid": "ListBucket",
            "Effect": "Allow",
            "Action": "s3:ListBucket",
            "Resource": [
                "arn:aws:s3:::destination-bucket-name"
            ]
        },
        {
            "Sid": "HomeDirObjectAccess",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:GetObject",
                "s3:DeleteObjectVersion",
                "s3:DeleteObject",
                "s3:GetObjectVersion"
            ],
            "Resource": "arn:aws:s3:::destination-bucket-name/*"
        },
```

```
        {
            "Sid": "Decrypt",
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetSecretValue"
            ],
            "Resource": "arn:aws:secretsmanager:region:account-ID:secret:aws/transfer/
*"
        }
    ]
}
```

実行ロールの例:関数の実行と削除

この例では、 AWS Lambda 関数を呼び出すワークフローがあります。ワークフローがアップロードされたファイルを削除し、前のステップで失敗したワークフロー実行を処理する例外ハンドラーステップがある場合は、次の IAM ポリシーを使用してください。*user input placeholder* を、ユーザー自身の情報に置き換えます。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Delete",
            "Effect": "Allow",
            "Action": [
                "s3:DeleteObject",
                "s3:DeleteObjectVersion"
            ],
            "Resource": "arn:aws:s3:::bucket-name"
        },
        {
            "Sid": "Custom",
            "Effect": "Allow",
            "Action": [
                "lambda:InvokeFunction"
            ],
            "Resource": [
                "arn:aws:lambda:region:account-id:function:function-name"
            ]
        }
    ]
```

```
}
```

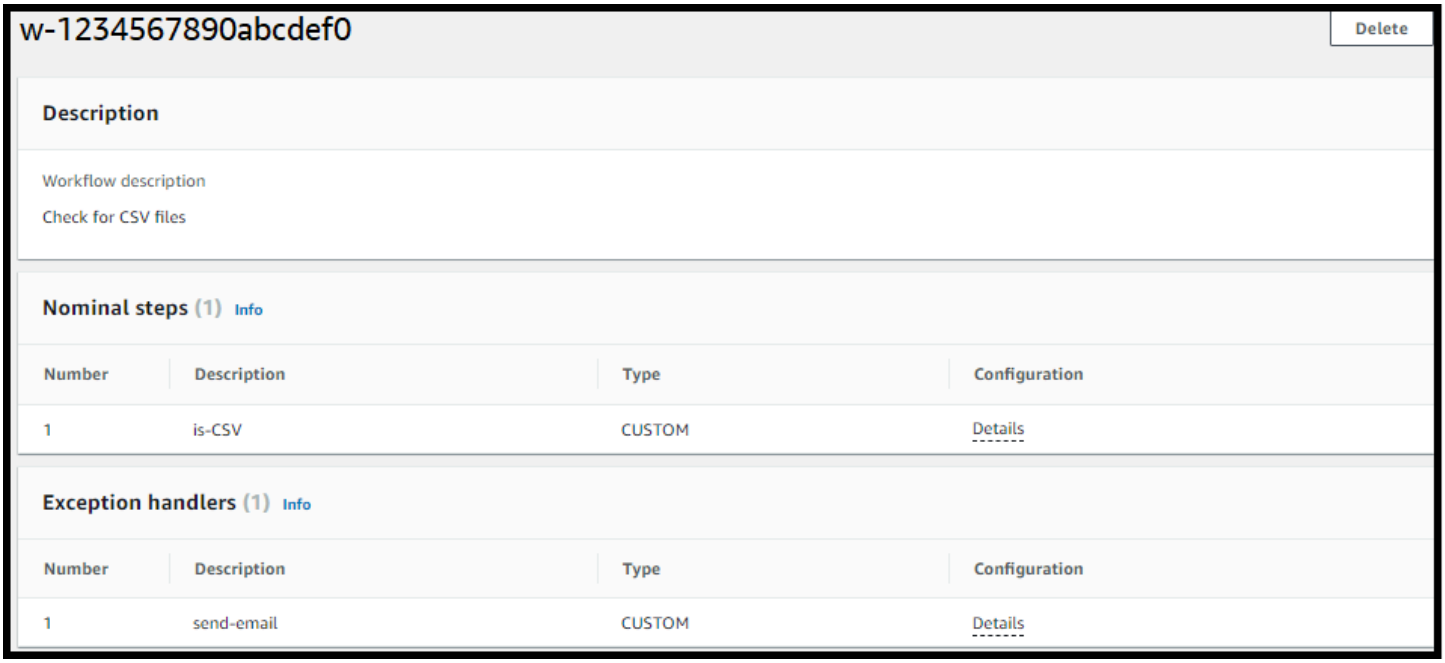
ワークフローの例外処理

ワークフローの実行中にエラーが発生した場合、指定した例外処理ステップが実行されます。ワークフローのエラー処理手順は、ワークフローの指名手順を指定するのと同じ方法で指定します。たとえば、受信ファイルを検証するためのカスタム処理をわずかな手順で設定したとします。ファイルの検証に失敗した場合、例外処理ステップで管理者にメールを送信できます。

以下のワークフロー例には、2 つのステップが含まれている：

- アップロードされたファイルが CSV 形式かどうかをチェックする、ごく普通のステップの 1 つです。
- アップロードされたファイルが CSV 形式でない場合に電子メールを送信する例外処理ステップで、通常のステップは失敗します。

例外処理ステップを開始するには、ノミナルステップの AWS Lambda 関数が で応答する必要がありますStatus="FAILURE"。ワークフローにおけるエラー処理の詳細については、the section called “カスタムのファイル処理ステップを使用してください。” を参照してください。



ワークフロー実行のモニタリング

Amazon は、 で実行している AWS リソースとアプリケーションを AWS クラウド リアルタイムで CloudWatch モニタリングします。Amazon を使用して、ワークフローで測定できる変数であるメトリクス CloudWatch を収集および追跡できます。Amazon を使用して、ワークフローメトリクスと統合ログを表示できます CloudWatch。

CloudWatch ワークフローの ログ記録

CloudWatch は、ワークフローの進行状況と結果の監査とログ記録を統合します。

ワークフローの Amazon CloudWatch ログを表示する

1. https://console.aws.amazon.com/cloudwatch/ で Amazon CloudWatch コンソールを開きます。
2. 左側のナビゲーションペインで、「ログ」を選択し、「ロググループ」を選択します。
3. ロググループページのナビゲーションバーで、 AWS Transfer Family サーバーに適したリージョンを選択します。
4. サーバーに対応するロググループを選択します。

   例えば、サーバー ID が `s-1234567890abcdef0` であれば、ロググループは `/aws/transfer/s-1234567890abcdef0` です。
5. サーバーのロググループの詳細ページに、最新のログストリームが表示されます。調査対象のユーザーには 2 つのログストリームがあります。

   - Secure Shell (SSH) File Transfer Protocol (SFTP) セッションごとに 1 つずつ。
   - 1 つはサーバーで実行中のワークフロー用です。ワークフローのログストリームの形式は *`username`*`.`*`workflowID`*`.`*`uniqueStreamSuffix`* です。

   例えば、ユーザーが `mary-major` である場合、以下のようなログストリームがある：

   ```
   mary-major-east.1234567890abcdef0
   mary.w-abcdef01234567890.021345abcdef6789
   ```

   > **Note**
   >
   > この例に挙げた 16 桁の英数字の識別子は架空のものです。Amazon に表示される値は CloudWatch 異なります。

mary-major-usa-east.1234567890abcdef0の「ログイベント」ページには各ユーザーセッションの詳細が表示され、mary.w-abcdef01234567890.021345abcdef6789ログストリームにはワークフローの詳細が含まれます。

以下は、コピーステップを含むワークフロー (w-abcdef01234567890) に基づく、mary.w-abcdef01234567890.021345abcdef6789 のログストリームのサンプルです。

```
{
    "type": "ExecutionStarted",
    "details": {
        "input": {
            "initialFileLocation": {
                "bucket": "DOC-EXAMPLE-BUCKET",
                "key": "mary/workflowSteps2.json",
                "versionId": "version-id",
                "etag": "etag-id"
            }
        }
    },
    "workflowId":"w-abcdef01234567890",
    "executionId":"execution-id",
    "transferDetails": {
        "serverId":"s-server-id",
        "username":"mary",
        "sessionId":"session-id"
    }
},
{
    "type":"StepStarted",
    "details": {
        "input": {
            "fileLocation": {
                "backingStore":"S3",
                "bucket":"DOC-EXAMPLE-BUCKET",
                "key":"mary/workflowSteps2.json",
                "versionId":"version-id",
                "etag":"etag-id"
            }
        },
        "stepType":"COPY",
        "stepName":"copyToShared"
    },
    "workflowId":"w-abcdef01234567890",
```

```
    "executionId":"execution-id",
    "transferDetails": {
        "serverId":"s-server-id",
        "username":"mary",
        "sessionId":"session-id"
    }
},
{
    "type":"StepCompleted",
    "details":{
        "output":{},
        "stepType":"COPY",
        "stepName":"copyToShared"
    },
    "workflowId":"w-abcdef01234567890",
    "executionId":"execution-id",
    "transferDetails":{
        "serverId":"server-id",
        "username":"mary",
        "sessionId":"session-id"
    }
},
{
    "type":"ExecutionCompleted",
    "details": {},
    "workflowId":"w-abcdef01234567890",
    "executionId":"execution-id",
    "transferDetails":{
        "serverId":"s-server-id",
        "username":"mary",
        "sessionId":"session-id"
    }
}
```

CloudWatch ワークフローの メトリクス

AWS Transfer Family には、ワークフローの複数のメトリクスが用意されています。過去 1 分間に開始された、正常に完了した、失敗したワークフロー実行の数を示す指標を表示できます。Transfer Family のすべての CloudWatch メトリクスについては、「」で説明されています[Transfer Family の CloudWatch メトリクスの使用]。

テンプレートからワークフローを作成する

テンプレートからワークフローとサーバーを作成する AWS CloudFormation スタックをデプロイできます。この手順には、ワークフローをすばやくデプロイするために使用できる例が含まれています。

AWS Transfer Family ワークフローとサーバーを作成する AWS CloudFormation スタックを作成するには

1. https://console.aws.amazon.com/cloudformation で AWS CloudFormation コンソールを開きます。
2. 以下のコードをファイルに保存します。

   YAML

```
AWSTemplateFormatVersion: 2010-09-09
Resources:
  SFTPServer:
    Type: 'AWS::Transfer::Server'
    Properties:
      WorkflowDetails:
        OnUpload:
          - ExecutionRole: workflow-execution-role-arn
            WorkflowId: !GetAtt
              - TransferWorkflow
              - WorkflowId
  TransferWorkflow:
    Type: AWS::Transfer::Workflow
    Properties:
      Description: Transfer Family Workflows Blog
      Steps:
        - Type: COPY
          CopyStepDetails:
            Name: copyToUserKey
            DestinationFileLocation:
              S3FileLocation:
                Bucket: archived-records
                Key: ${transfer:UserName}/
            OverwriteExisting: 'TRUE'
        - Type: TAG
          TagStepDetails:
            Name: tagFileForArchive
```

```
            Tags:
              - Key: Archive
                Value: yes
        - Type: CUSTOM
          CustomStepDetails:
            Name: transferExtract
            Target: arn:aws:lambda:region:account-id:function:function-name
            TimeoutSeconds: 60
        - Type: DELETE
          DeleteStepDetails:
            Name: DeleteInputFile
            SourceFileLocation: '${original.file}'
      Tags:
        - Key: Name
          Value: TransferFamilyWorkflows
```

JSON

```
{
    "AWSTemplateFormatVersion": "2010-09-09",
    "Resources": {
        "SFTPServer": {
            "Type": "AWS::Transfer::Server",
            "Properties": {
                "WorkflowDetails": {
                    "OnUpload": [
                        {
                            "ExecutionRole": "workflow-execution-role-arn",
                            "WorkflowId": {
                                "Fn::GetAtt": [
                                    "TransferWorkflow",
                                    "WorkflowId"
                                ]
                            }
                        }
                    ]
                }
            }
        },
        "TransferWorkflow": {
            "Type": "AWS::Transfer::Workflow",
            "Properties": {
                "Description": "Transfer Family Workflows Blog",
```

```
                "Steps": [
                    {
                        "Type": "COPY",
                        "CopyStepDetails": {
                            "Name": "copyToUserKey",
                            "DestinationFileLocation": {
                                "S3FileLocation": {
                                    "Bucket": "archived-records",
                                    "Key": "${transfer:UserName}/"
                                }
                            },
                            "OverwriteExisting": "TRUE"
                        }
                    },
                    {
                        "Type": "TAG",
                        "TagStepDetails": {
                            "Name": "tagFileForArchive",
                            "Tags": [
                                {
                                    "Key": "Archive",
                                    "Value": "yes"
                                }
                            ]
                        }
                    },
                    {
                        "Type": "CUSTOM",
                        "CustomStepDetails": {
                            "Name": "transferExtract",
                            "Target": "arn:aws:lambda:region:account-
id:function:function-name",
                            "TimeoutSeconds": 60
                        }
                    },
                    {
                        "Type": "DELETE",
                        "DeleteStepDetails": {
                            "Name": "DeleteInputFile",
                            "SourceFileLocation": "${original.file}"
                        }
                    }
                ],
                "Tags": [
```

```
                    {
                        "Key": "Name",
                        "Value": "TransferFamilyWorkflows"
                    }
                ]
            }
        }
    }
}
```

3. 次の項目を実際の値に置き換えます。

   - *workflow-execution-role-arn* を実際のワークフロー実行ロールの ARN に置き換えます。例えば、次のようになります: `arn:aws:transfer:us-east-2:111122223333:workflow/w-1234567890abcdef0`
   - `arn:aws:lambda:`*region*`:`*account-id*`:function:`*function-name* を Lambda 関数の ARN に置き換えます。例えば `arn:aws:lambda:us-east-2:123456789012:function:example-lambda-idp` です。

4. 「 AWS CloudFormation ユーザーガイド」の AWS CloudFormation 「スタックテンプレートの選択」にある既存のテンプレートから スタックをデプロイする手順に従います。

スタックがデプロイされたら、 CloudFormation コンソールの出力タブでスタックの詳細を表示できます。テンプレートは、サービスマネージドユーザーを使用する新しい AWS Transfer Family SFTP サーバーと新しいワークフローを作成し、ワークフローを新しいサーバーに関連付けます。

Transfer Family サーバーからワークフローを削除する

ワークフローを Transfer Family サーバーに関連付けていて、その関連付けを削除したい場合は、コンソールを使用するか、プログラムを使用して削除できます。

Console

Transfer Family サーバーからワークフローを削除するには

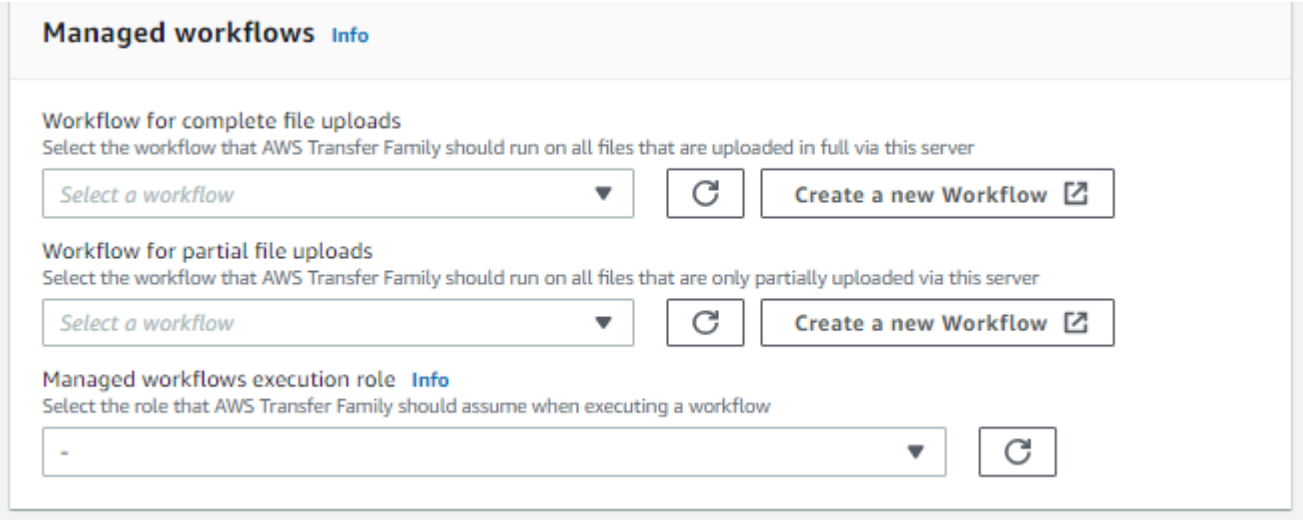
1. https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開きます。
2. 左側のナビゲーションペインで [Servers] (サーバー) を選択します。
3. 「サーバー ID」欄でサーバーの識別子を選択します。

4. サーバーの詳細ページで下にスクロールして [Additional details] (その他の詳細) セクションで [Edit] (編集) を選択します。
5. 「追加情報の編集」ページの「管理されたワークフロー」セクションで、すべての設定の情報をクリアします。

   - ワークフローのリストから「ファイルアップロードを完了するワークフロー」のダッシュ (-) を選択します。
   - まだクリアされていない場合は、「部分ファイルアップロード用ワークフロー」のワークフローリストからダッシュ (-) を選択します。
   - 「マネージドワークフロー実行ロール」のロールのリストからダッシュ (-) を選択します。

   ダッシュが表示されていない場合は、表示されるまで上にスクロールしてください。ダッシュは各メニューの最初の値です。

   画面は以下のようになるはずです。



6. 下にスクロールし、「保存」を選択して変更を保存します。

CLI

`update-server`(または API の`UpdateServer`) 呼び出しを使用し、`OnUpload`および`OnPartialUpload`パラメータには空の引数を指定します。

から AWS CLI、次のコマンドを実行します。

```
aws transfer update-server --server-id your-server-id --workflow-details
 '{"OnPartialUpload":[],"OnUpload":[]}'
```

*your-server-id*をサーバーの ID に置き換えます。たとえば、サーバー ID がs-01234567890abcdefの場合、コマンドは次のようになります。

```
aws transfer update-server --server-id s-01234567890abcdef --workflow-details
 '{"OnPartialUpload":[],"OnUpload":[]}'
```

マネージドワークフローの制限事項と機能制限

制限事項

現時点では AWS Transfer Familyのアップロード後処理ワークフローに以下の制限事項が適用されます。

- クロスアカウント関数とクロスリージョン AWS Lambda 関数はサポートされていません。ただし、 AWS Identity and Access Management (IAM) ポリシーが正しく設定されていれば、アカウント間でコピーできます。
- すべてのワークフローステップで、ワークフローがアクセスする Amazon S3 バケットは、ワークフロー自体と同じリージョンにある必要があります。
- 復号ステップでは、復号化先がリージョンとバッキングストアのソースと一致する必要があります (たとえば、復号するファイルが Amazon S3 に保存されている場合、指定された宛先も Amazon S3 にある必要がある)。
- 非同期カスタムステップのみがサポートされます。
- カスタムステップタイムアウトは概算です。つまり、指定された時間よりも若干タイムアウトに時間がかかる可能性があります。さらに、ワークフローは Lambda 関数に依存します。したがって、実行中に関数が遅延した場合、ワークフローは遅延を認識しません。
- スロットリング制限を超えた場合、Transfer Family はワークフローオペレーションをキューに追加しません。
- ワークフローは、サイズが 0 のファイルに対しては開始されません。サイズが 0 より大きいファイルは、関連するワークフローを開始します。

機能制限

さらに、Transfer Family のワークフローには、次の機能制限が適用されます。

- リージョンごと、アカウントごとのワークフロー数は 10 に制限されています。
- カスタムステップの最大タイムアウト値は 30 分です。
- ワークフローの最大ステップ数は 8 です。
- ワークグループあたりのタグの最大数は 50 です。
- 復号ステップを含む同時実行の最大数は、ワークフローあたり 250 です。
- Transfer Family サーバー 1 台につき、1 ユーザーにつき最大 3 つの PGP 秘密鍵を保存できます。
- 復号化されたファイルの最大サイズは 10 GB です。
- 新しい実行速度は、バースト容量が 100 でリフィル率が 1 の「トークンバケット」システムを使用して調整しています。
- サーバーからワークフローを削除し、新しいワークフローに置き換える場合、またはサーバー構成を更新する場合（ワークフローの実行ロールに影響する）、新しいワークフローを実行する前に約 10 分間待機する必要があります。Transfer Familyサーバーはワークフローの詳細をキャッシュし、サーバーがキャッシュを更新するのに10分かかります。

  さらに、アクティブなSFTPセッションからログアウトし、10分間の待ち時間の後にログインし直すと、変更を確認できます。

サーバーの管理

このセクションでは、サーバーのリストを表示する方法、サーバーの詳細を表示する方法、サーバーの詳細を編集する方法、および SFTP 対応サーバーのホストキーを変更する方法について説明します。

トピック

サーバーのリストを表示する

AWS Transfer Family コンソールには、選択した AWS リージョンにあるすべてのサーバーのリストが表示されます。

AWS リージョンに存在するサーバーのリストを検索するには

- https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開きます。

  現在の AWS リージョンに 1 つ以上のサーバーがある場合、コンソールが開き、サーバーのリストが表示されます。サーバーのリストが表示されない場合は、正しいリージョンにいることを確認してください。ナビゲーションペインで [Servers] (サーバー) を選択することもできます。

  サーバーの詳細を表示する方法の詳細については、「SFTP、FTPS、FTP サーバーの詳細を表示する」を参照してください。

サーバーを削除する

この手順では、 AWS Transfer Family コンソールまたは を使用して Transfer Family サーバーを削除する方法について説明します AWS CLI。

> **Important**
>
> サーバーを削除するまでは、エンドポイントへのアクセスが有効なプロトコルごとに課金が発生します。

> **Warning**
>
> サーバーを削除すると、そのサーバーのすべてのユーザーが削除されます。サーバーを使用してアクセスされたバケット内のデータは削除されず、それらの Amazon S3 バケットへの権限を持つ AWS ユーザーがアクセス可能なままであります。

Console

コンソールを使用してサーバーを削除するには

1. https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開きます。
2. 左側のナビゲーションペインで [Servers] (サーバー) を選択します。
3. 削除したいプリセットのチェックボックスを選択します。
4. [Actions] (アクション) について [Delete] (削除) を選択します。
5. 表示される確認ダイアログボックスで、「**delete**」という語を入力してから [Delete] (削除) を選択してサーバーを削除してよいことを確認します。

[Servers] (サーバー) ページからサーバーが削除されれば、その料金は請求されなくなります。

AWS CLI

CLI を使用してサーバーを削除するには

1. (オプション) 次のコマンドを実行して、完全に削除するサーバーの詳細を表示します。

```
aws transfer describe-server --server-id your-server-id
```

このdescribe-serverコマンドは、サーバーの詳細をすべて返します。

2. 次のコマンドを実行してサーバーを削除します。

```
aws transfer delete-server --server-id your-server-id
```

成功すると、コマンドはサーバーを削除し、情報を返しません。

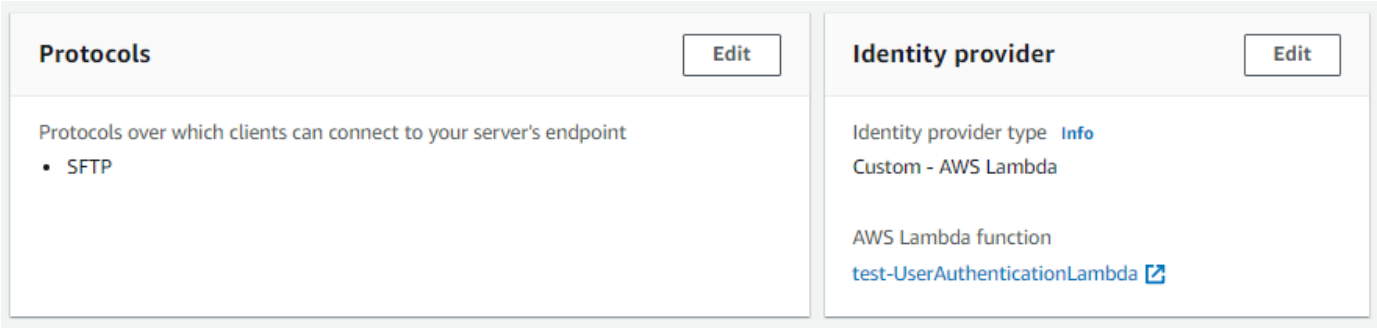
SFTP、FTPS、FTP サーバーの詳細を表示する

個々の AWS Transfer Family サーバーの詳細とプロパティのリストを確認できます。サーバーのプロパティには、プロトコル、ID プロバイダー、ステータス、エンドポイントタイプ、カスタムホスト名、エンドポイント、ユーザー、ログ記録ロール、サーバーホストキー、およびタグが含まれます。

サーバーの詳細を表示するには

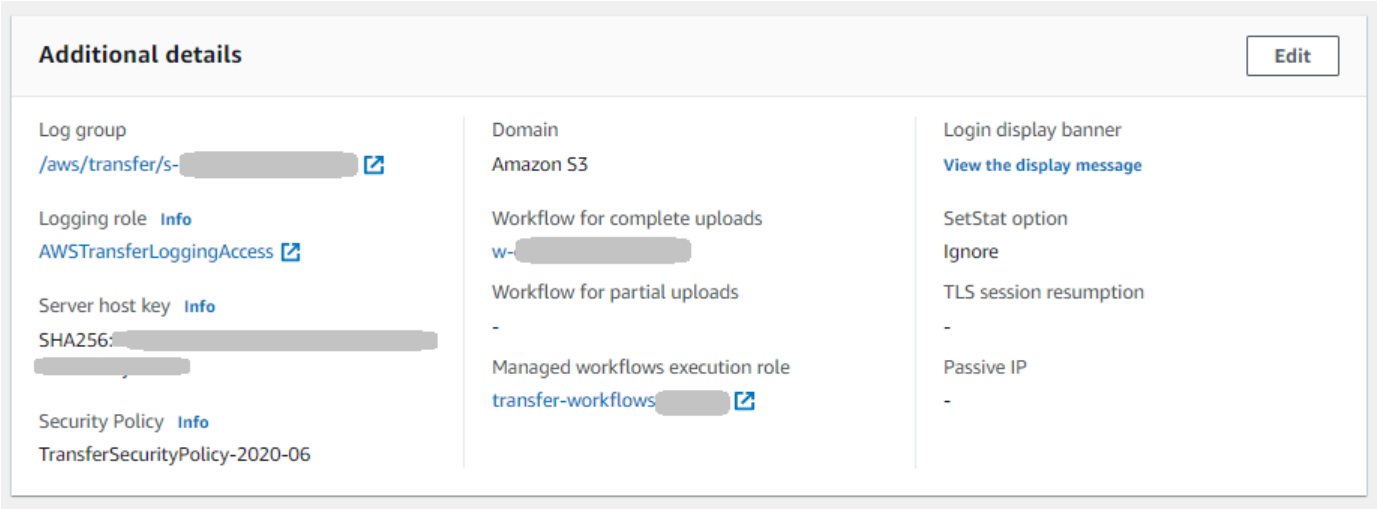
1. https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開きます。
2. ナビゲーションペインで、[Servers] (サーバー) を選択します。
3. [Server ID] (サーバー ID) 列で ID を選択すると、次のように [Server Configuration] (サーバーの構成) ページが開きます。

   このページで [Edit] (編集) を選択して、サーバーのプロパティを変更できます。サーバーの詳細を編集する方法の詳細については、「サーバーの詳細を編集する」を参照してください。AS2 サーバーの詳細ページは少し異なります。AS2 サーバーについては、「」を参照してくださいAS2 サーバーの詳細を表示する。



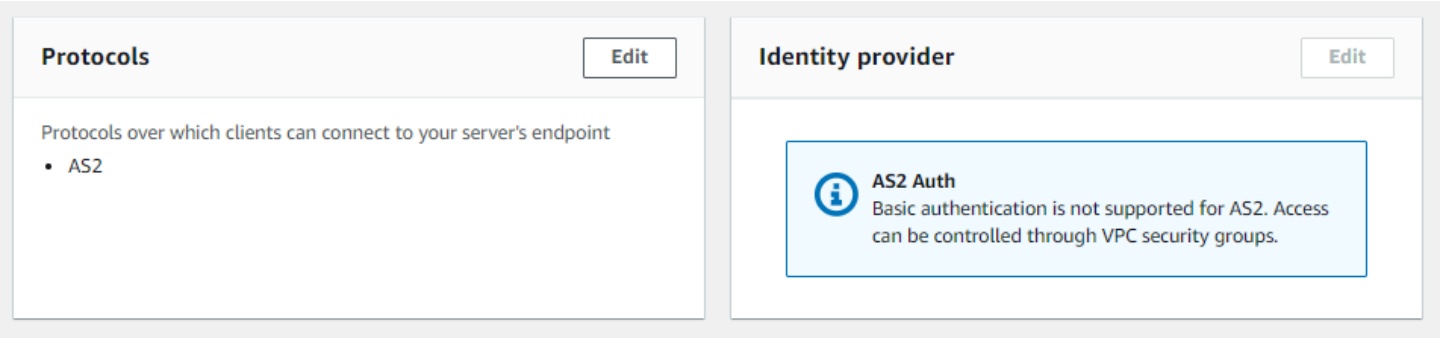
> **Note**
>
> サーバーホストキーの [説明]と [インポートされた日付] の値は、2022 年 9 月時点で新しくなりました。これらの値は、複数のホストキー特徴量をサポートするために導入さ

れました。この特徴量では、複数のホストキーが導入される前に使用されていたすべてのホストキーを移行する必要がありました。
移行されたサーバーホストキーの [インポートされた日付] の値は、サーバーの最終更新日に設定されます。つまり、移行されたホストキーに表示される日付は、サーバーホストキーの移行前に、何らかの方法でサーバーを最後に変更した日付に対応しています。
移行された唯一のキーは、最も古い、または唯一のサーバーホストキーです。その他のキーには、インポートした時点の実際の日付が反映されます。さらに、移行されたキーには、移行されたものであることを簡単に識別できるように説明が付いています。
移行は 9 月 2 日から 9 月 13 日の間に行われました。この範囲内の実際の移行日は、サーバーのリージョンによって異なります。

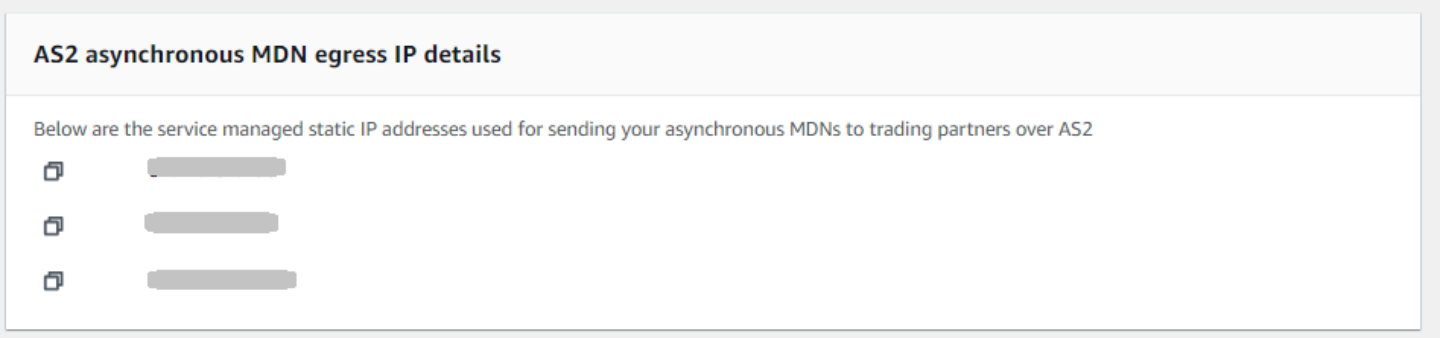


AS2 サーバーの詳細を表示する

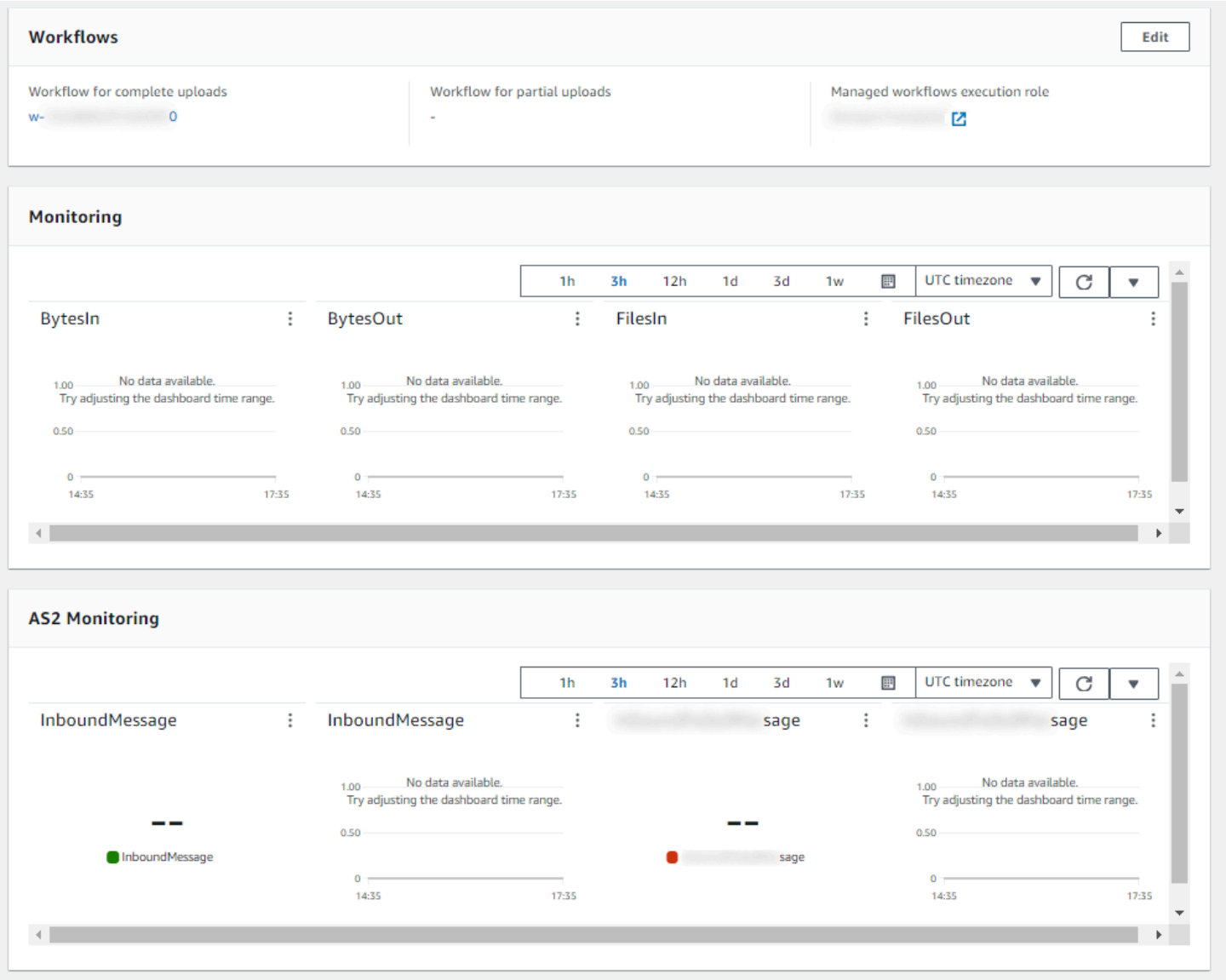
個々の AWS Transfer Family サーバーの詳細とプロパティのリストを確認できます。サーバープロパティには、プロトコル、ステータスなどが含まれます。AS2 サーバーの場合、AS2 非同期 MDN 出力 IP アドレスを表示することもできます。



各 AS2 サーバーには 3 つの静的 IP アドレスが割り当てられます。これらの IP アドレスを使用して、AS2 経由で取引相手に非同期 MDNsを送信します。



AS2 サーバーの詳細ページの下部には、アタッチされたワークフローとモニタリングおよびタグ付け情報の詳細が含まれています。



サーバーの詳細を編集する

AWS Transfer Family サーバーを作成したら、サーバー設定を編集できます。

トピック

- ファイル転送プロトコルを編集する
- カスタム ID プロバイダーパラメーターを編集
- サーバーエンドポイントを編集する
- ログ設定を編集する
- セキュリティポリシーを編集する

- SFTP サーバーの管理ワークフローを変更する
- SFTP サーバーのディスプレイバナーを変更します
- サーバーのオンラインとオフラインを切り替える

サーバーの設定を編集するには

1. https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開きます。
2. 左側のナビゲーションペインで [Servers] (サーバー) を選択します。
3. [Server ID] (サーバー ID) 列で ID を選択すると、次のように [Server Configuration] (サーバーの構成) ページが開きます。

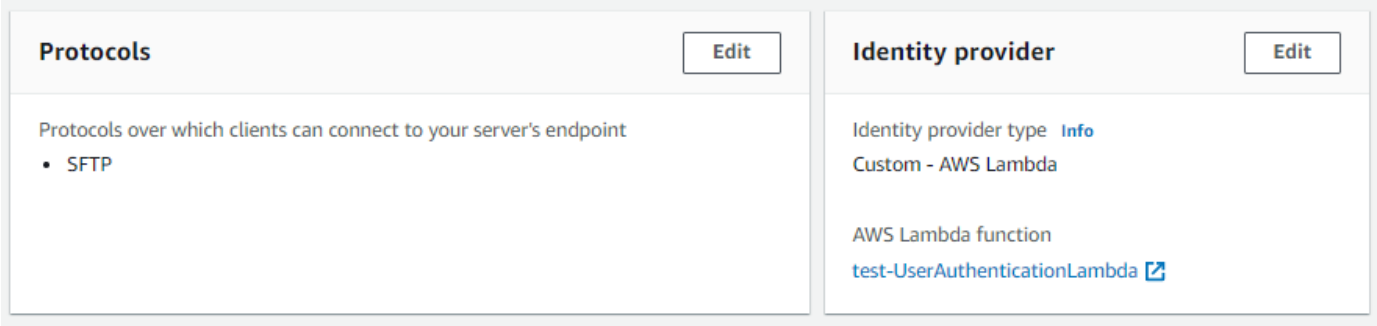
   このページで [Edit] (編集) を選択して、サーバーのプロパティを変更できます。

   - プロトコルを変更するには、「ファイル転送プロトコルを編集する」を参照してください。
   - ID プロバイダについては、サーバを作成した後にサーバの ID プロバイダタイプを変更することはできません。ID プロバイダーを変更するには、サーバーを削除し、希望の ID プロバイダーで新しいサーバーを作成する必要があります。

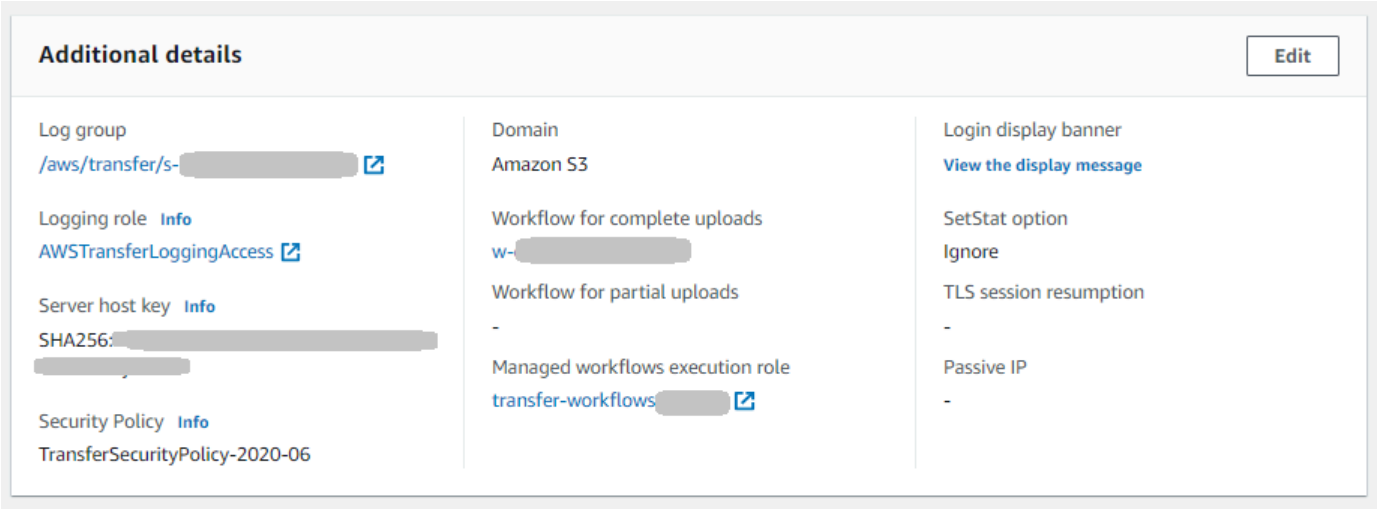
     > **Note**
     >
     > サーバーがカスタム ID プロバイダーを使用している場合は、一部のプロパティを編集できます。詳細については、「カスタム ID プロバイダーパラメーターを編集」を参照してください。

   - エンドポイントタイプまたはカスタムホスト名を変更するには、「サーバーエンドポイントを編集する」を参照してください。
   - 契約を追加するには、まず AS2 をプロトコルとしてサーバーに追加する必要があります。詳細については、「ファイル転送プロトコルを編集する」を参照してください。
   - サーバーのホストキーを管理するには、SFTP 対応サーバーのホストキーを管理を参照してください。
   - 「その他の詳細」では、以下の情報を編集できます。
     - ログ記録ロールを変更する手順については、「ログ設定を編集する」を参照してください。
     - セキュリティポリシーを変更するには、「セキュリティポリシーを編集する」を参照してください。

  - サーバーのホストキーを変更するには、「SFTP 対応サーバーのホストキーを管理」を参照してください。
  - サーバのマネージドワークフローを変更するには、SFTP サーバーの管理ワークフローを変更するを参照してください。
  - サーバのディスプレイバナーを編集するには、SFTP サーバーのディスプレイバナーを変更しますを参照してください。
- [追加設定] で、以下の情報を編集できます。
  - SetStat オプション : このオプションを有効にすると、クライアントが Amazon S3 バケットにアップロードするファイルSETSTATで を使用しようとしたときに生成されるエラーを無視できます。詳細については、ProtocolDetails「」トピックのSetStatOptionドキュメントを参照してください。
  - [TLS セッション再開]:FTPS セッションの制御接続とデータ接続の間でネゴシエートされた秘密キーを再開または共有するメカニズムを提供します。詳細については、ProtocolDetails「」トピックのTlsSessionResumptionModeドキュメントを参照してください。
  - [Passive IP] : FTP および FTPS プロトコルのパッシブモードを示します。ファイアウォール、ルーター、ロードバランサーのパブリック IP アドレスなど、単一 IPv4 アドレスを入力します。詳細については、ProtocolDetails「」トピックのPassiveIpドキュメントを参照してください。
- サーバを起動または停止するには、「サーバーのオンラインとオフラインを切り替える」を参照してください。
- サーバーを削除するには、「サーバーを削除する」を参照してください。
- ユーザーのプロパティを編集するには、「アクセスコントロールの管理」を参照してください。



> **Note**
>
> サーバーホストキーの [説明]と [インポートされた日付] の値は、2022 年 9 月時点で新しくなりました。これらの値は、複数のホストキー特徴量をサポートするために導入されました。この特徴量では、複数のホストキーが導入される前に使用されていたすべてのホストキーを移行する必要がありました。
> 移行されたサーバーホストキーの [インポートされた日付] の値は、サーバーの最終更新日に設定されます。つまり、移行されたホストキーに表示される日付は、サーバーホストキーの移行前に、何らかの方法でサーバーを最後に変更した日付に対応しています。
> 移行された唯一のキーは、最も古い、または唯一のサーバーホストキーです。その他のキーには、インポートした時点の実際の日付が反映されます。さらに、移行されたキーには、移行されたものであることを簡単に識別できるように説明が付いています。
> 移行は 9 月 2 日から 9 月 13 日の間に行われました。この範囲内の実際の移行日は、サーバーのリージョンによって異なります。



ファイル転送プロトコルを編集する

AWS Transfer Family コンソールでは、ファイル転送プロトコルを編集できます。ファイル転送プロトコルは、クライアントをサーバーのエンドポイントに接続します。

プロトコルを編集するには

1. [Server details] (サーバーの詳細) ページで [Protocols] (プロトコル) の隣にある [Edit] (編集) を選択します。
2. [Edit protocols] (プロトコルの編集) ページで、1 つ以上のプロトコルチェックボックスをオンまたはオフにして、以下のファイル転送プロトコルを追加または削除します。

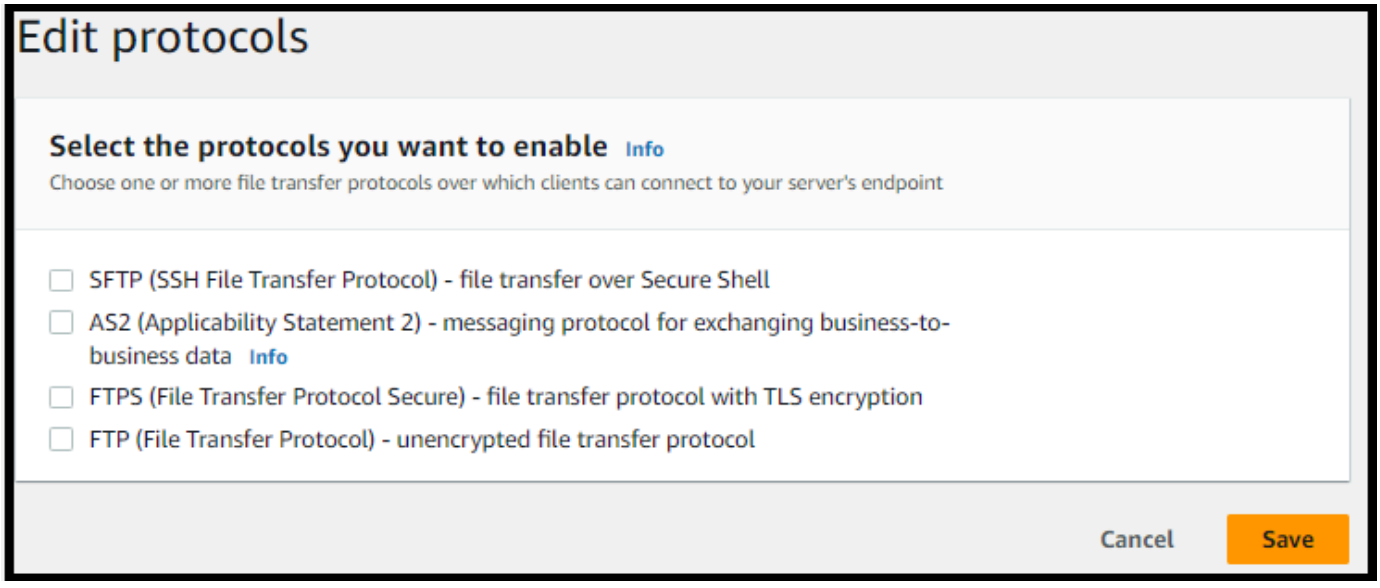
   - Secure Shell (SSH) File Transfer Protocol (SFTP) – SSH 経由のファイル転送

     SFTP の詳細については、「SFTP 対応サーバーの作成」を参照してください。
   - File Transfer Protocol Secure (FTPS) – TLS 暗号化によるファイル転送

     FTP の詳細については、「FTPS 対応サーバーを作成する」を参照してください。
   - File Transfer Protocol (FTP) - 暗号化されていないファイル転送

     FTPS の詳細については、「FTP 対応サーバーの作成」を参照してください。

> **Note**
>
> 既存のサーバーが SFTP についてのみ有効で FTPS と FTP を追加したい場合、FTPS および FTP と互換性のある適切な ID プロバイダーとエンドポイントタイプの設定ができていることを確認する必要があります。



FTPS を選択した場合は、 AWS Certificate Manager (ACM) に保存されている証明書を選択する必要があります。この証明書は、クライアントが FTPS 経由でサーバーに接続するときにサーバーを識別するために使用されます。

新しいパブリック証明書をリクエストするには、AWS Certificate Manager ユーザーガイドの「パブリック証明書をリクエストする」を参照してください。

既存の証明書を ACM にインポートするには、AWS Certificate Manager ユーザーガイドの「ACM に証明書をインポートする」を参照してください。

プライベート IP アドレス経由で FTPS を使用するためにプライベート証明書を要求するには、「AWS Certificate Manager ユーザーガイド」の「プライベート証明書の要求」を参照してください。

以下の暗号化アルゴリズムとキーサイズの証明書がサポートされています。

- 2048 ビット RSA (RSA_2048)
- 4096 ビット RSA (RSA_4096)
- 楕円素数曲線 256 ビット (EC_Prime256v1)
- 楕円素数曲線 384 ビット (EC_secp384R1)
- 楕円素数曲線 521 ビット (EC_secp521R1)

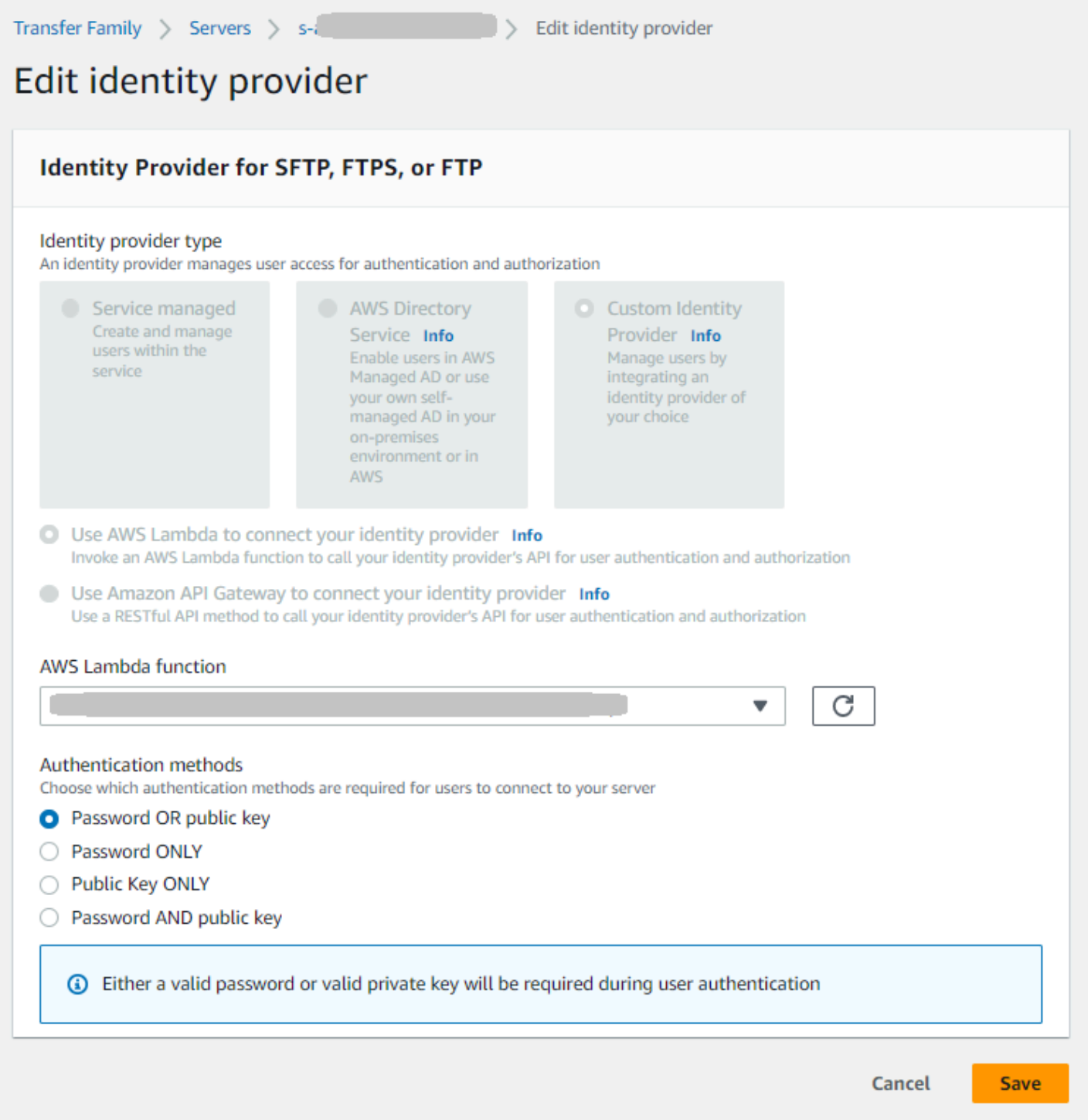
> **Note**
>
> 証明書は、FQDN または IP アドレスが指定された有効な SSL/TLS X.509 バージョン 3 証明書で、発行者に関する情報が含まれている必要があります。

3. [Save] (保存) を選択します。[Server details] (サーバーの詳細) ページが再表示されます。

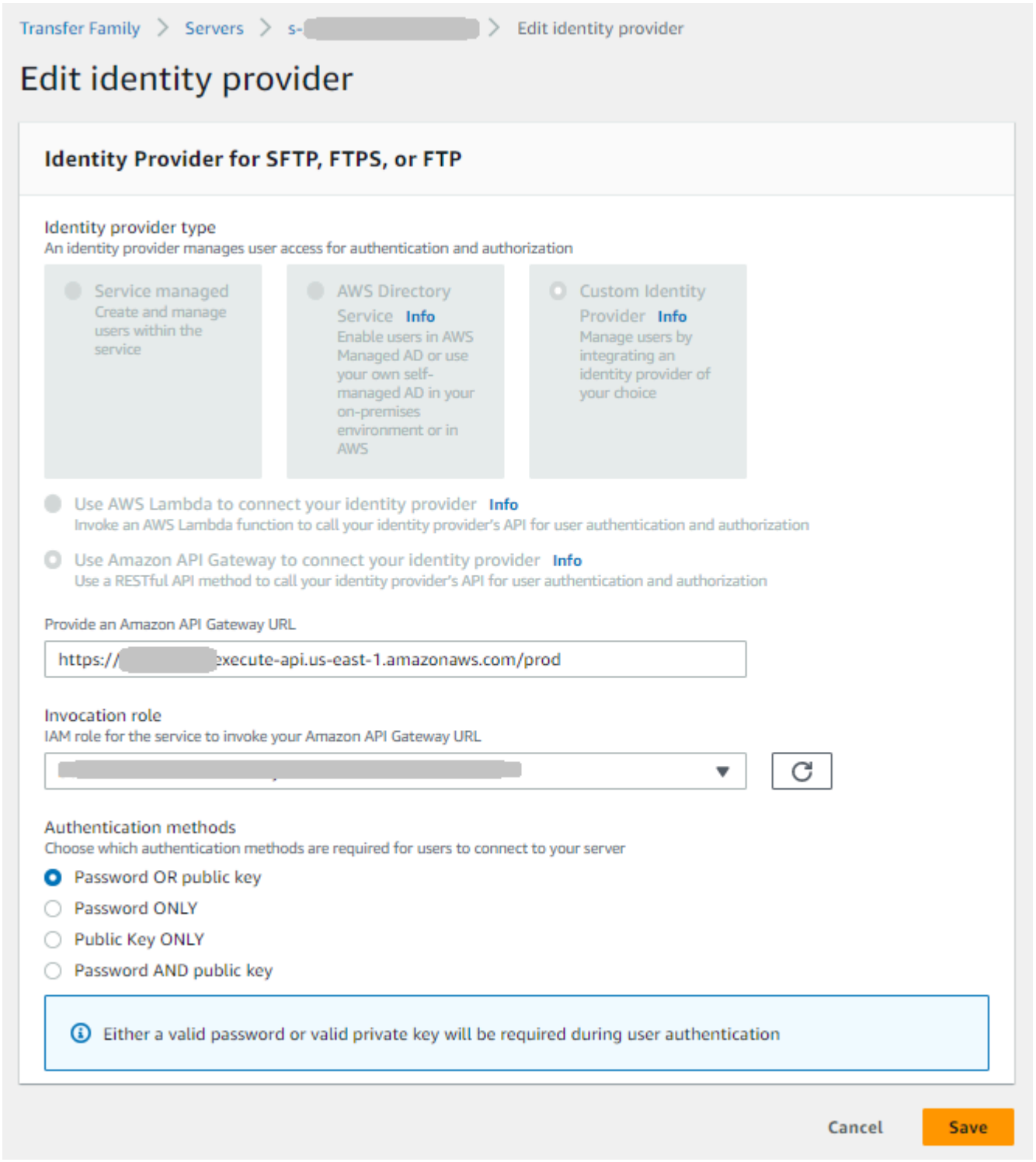
カスタム ID プロバイダーパラメーターを編集

AWS Transfer Family コンソールでは、カスタム ID プロバイダーの場合、Lambda 関数と API Gateway のどちらを使用しているかに応じて、一部の設定を変更できます。いずれの場合も、サーバーが SFTP プロトコルを使用している場合は、認証方法を編集できます。

- ID プロバイダーとして Lambda を使用している場合は、基になる Lambda 関数を変更できます。



- ID プロバイダーとして API ゲートウェイを使用している場合は、ゲートウェイ URL または呼び出しロール、あるいはその両方を更新できます。



サーバーエンドポイントを編集する

AWS Transfer Family コンソールでは、サーバーエンドポイントタイプとカスタムホスト名を変更できます。さらに、VPC エンドポイントでは、アベイラビリティーゾーン情報を編集できます。

サーバーエンドポイントの詳細を編集するには

1. [Server details] (サーバーの詳細) ページで [Endpoint details] (エンドポイントの詳細) の隣にある [Edit] (編集) を選択します。
2. エンドポイントタイプ を編集する前に、まずサーバーを停止する必要があります。次に、エンドポイント設定の編集ページのエンドポイントタイプ で、次のいずれかの値を選択できます。

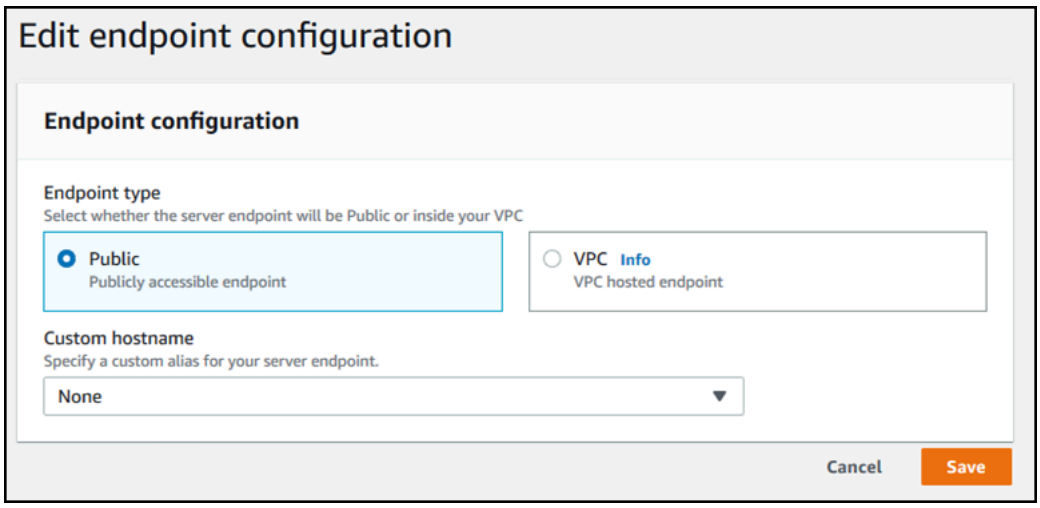
   - [公開] ー このオプションは、インターネット経由でサーバーにアクセスできるようにします。
   - [VPC] ー このオプションは、仮想プライベートクラウド (VPC) 内でサーバーにアクセスできます。VPC については、「Virtual Private Cloud でサーバーを作成する」を参照してください。
3. [Custom hostname] (カスタムホスト名) の場合、次のいずれかを選択します。

   - [なし] ー カスタムドメインを使用したくない場合は、「なし」を選択します。

     によって提供されるサーバーホスト名を取得します AWS Transfer Family。サーバーホスト名の形式は *serverId*.server.transfer.*regionId*.amazonaws.com です。
   - [Amazon Route 53 DNS エイリアス] ー Route 53 で自動的に作成されたDNSエイリアスを使用するには、このオプションを選択します。
   - [ほかの DNS] ー 外部 DNS サービスですでに所有しているホスト名を使用するには、[ほかの DNS]を選択します。

   [Amazon Route 53 DNS alias] (Amazon Route 53 DNS エイリアス) または [Other DNS] (その他のDNS) を選択することでサーバーのエンドポイントに関連付ける名前解決メソッドを指定します。

   たとえば、カスタムドメインは sftp.inbox.example.com である場合があります。カスタムホスト名は、ユーザーが提供し、DNS サービスが解決できる DNS 名を使用します。Route 53 を DNS リゾルバーとして使用するか、独自の DNS サービスプロバイダーを使用できます。AWS Transfer Family が Route 53 を使用してカスタムドメインから SFTP エンドポイントにトラフィックをルーティングするしくみについては、「カスタムホスト名の使用」を参照してください。



4. VPC エンドポイントの場合、アベイラビリティーゾーンペインで情報を変更できます。
5. [Save] (保存) を選択します。[Server details] (サーバーの詳細) ページが再表示されます。

ログ設定を編集する

AWS Transfer Family コンソールでは、ログ記録設定を変更できます。

> **Note**
>
> Transfer Family がサーバーの作成時に CloudWatch ログ記録 IAM ロールを作成した場合、IAM ロールは と呼ばれます`AWSTransferLoggingAccess`。トランスファーファミリーのすべてのサーバーに使用できます。

ロギング設定を変更するには

1. [Server details] (サーバーの詳細) ページで [Additional details] (その他の詳細) の隣にある [Edit] (編集) を選択します。
2. 設定に基づいて、ロギングロール、構造化 JSON ロギング、またはその両方を選択します。詳細については、「サーバーのロギングを更新します。」を参照してください。

セキュリティポリシーを編集する

この手順では、 AWS Transfer Family コンソールまたは を使用して Transfer Family サーバーのセキュリティポリシーを変更する方法について説明します AWS CLI。

> **Note**
>
> エンドポイントが FIPS 対応の場合、FIPS セキュリティポリシーを非 FIPS セキュリティポリシーに変更することはできません。

Console

コンソールを使用してセキュリティポリシーを編集するには

1. [Server details] (サーバーの詳細) ページで [Additional details] (その他の詳細) の隣にある [Edit] (編集) を選択します。
2. [暗号アルゴリズムオプション]セクションで、サーバーで使用可能な暗号アルゴリズムを含むセキュリティ・ ポリシーを選択します。

   セキュリティポリシーの詳細については、「AWS Transfer Family サーバーのセキュリティポリシー」を参照してください。
3. [保存] を選択します。

   サーバーの詳細ページに戻り、更新されたセキュリティポリシーが表示されます。

AWS CLI

CLI を使用してセキュリティポリシーを編集するには

1. 次のコマンドを実行して、サーバーにアタッチされている現在のセキュリティポリシーを表示します。

   ```
   aws transfer describe-server --server-id your-server-id
   ```

   このdescribe-serverコマンドは、次の行を含むサーバーの詳細をすべて返します。

   ```
   "SecurityPolicyName": "TransferSecurityPolicy-2018-11"
   ```

   この場合、サーバーのセキュリティポリシーは でTransferSecurityPolicy-2018-11すす。
2. コマンドには、セキュリティポリシーの正確な名前を必ず指定してください。例えば、次のコマンドを実行してサーバーを に更新しますTransferSecurityPolicy-2023-05。

```
aws transfer update-server --server-id your-server-id --security-policy-name
 "TransferSecurityPolicy-2023-05"
```

> **Note**
>
> 使用可能なセキュリティポリシーの名前は、「」に記載されていますAWS Transfer Family サーバーのセキュリティポリシー。

成功すると、コマンドは次のコードを返し、サーバーのセキュリティポリシーを更新します。

```
{
    "ServerId": "your-server-id"
}
```

SFTP サーバーの管理ワークフローを変更する

AWS Transfer Family コンソールでは、サーバーに関連付けられた管理ワークフローを変更できます。

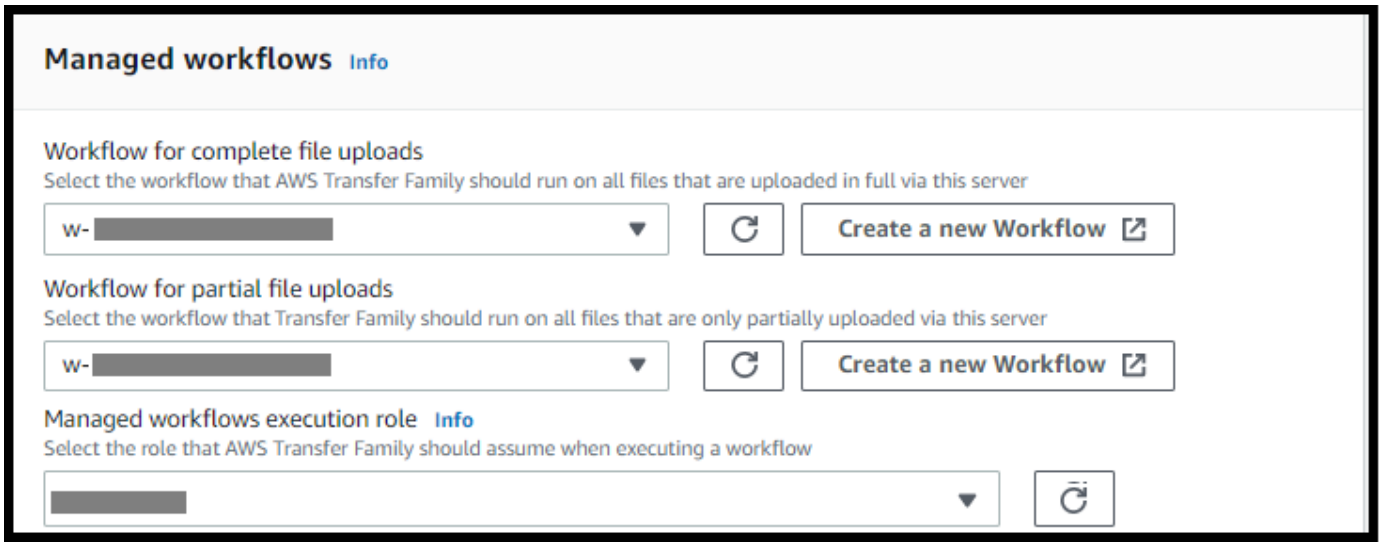
管理対象ワークフローを変更するには

1. [Server details] (サーバーの詳細) ページで [Additional details] (その他の詳細) の隣にある [Edit] (編集) を選択します。
2. 「追加詳細の編集」ページの「管理ワークフロー」セクションで、すべてのアップロードに対して実行するワークフローを選択します。

> **Note**
>
> ワークフローがまだない場合は、「新しいワークフローを作成する」を選択し、ワークフローを作成します。

   a. 使用するワークフロー ID を選択します。

b. 実行ロールを選択します。これは、Transfer Family がワークフローのステップを実行するときに引き受ける役割です。詳細については、「ワークフローの IAM ポリシー」を参照してください 保存を選択します。



3. [Save] (保存) を選択します。[Server details] (サーバーの詳細) ページが再表示されます。

SFTP サーバーのディスプレイバナーを変更します

AWS Transfer Family コンソールでは、サーバーに関連付けられた表示バナーを変更できます。

表示バナーを変更するには

1. [Server details] (サーバーの詳細) ページで [Additional details] (その他の詳細) の隣にある [Edit] (編集) を選択します。
2. 「詳細情報の編集」 ページの「表示バナー」セクションで、使用可能なディスプレイバナーのテキストを入力します。
3. [Save] (保存) を選択します。[Server details] (サーバーの詳細) ページが再表示されます。

サーバーのオンラインとオフラインを切り替える

AWS Transfer Family コンソールでは、サーバーをオンラインにすることも、オフラインにすることもできます。

サーバーをオンラインにするには

1. https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開きます。
2. ナビゲーションペインで [Servers] (サーバー) を選択します。
3. オフラインになっているサーバーのチェックボックスをオンにします。
4. [Actions] (アクション) で [Start] (開始) を選択します。

サーバーがオフラインからオンラインに切り替わるまでに数分かかる場合があります。

> **Note**
>
> サーバーを停止してオフラインにしても、現状ではそのサーバーのサービス料金は発生し続けます。サーバーの追加料金が発生しないようにするには、そのサーバーを削除してください。

サーバーをオフラインにするには

1. https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開きます。
2. ナビゲーションペインで [Servers] (サーバー) を選択します。
3. オンラインになっているサーバーのチェックボックスをオンにします。
4. [Actions] (アクション) で [Stop] (停止) を選択します。

サーバーのスタートアップ時またはシャットダウン時には、サーバーをファイルオペレーションに利用することはできません。コンソールには、開始状態や停止状態が表示されません。

エラー状態 `START_FAILED`または が見つかった場合は`STOP_FAILED`、 AWS Support に連絡して問題を解決してください。

SFTP 対応サーバーのホストキーを管理

> **Important**
>
> 既存のユーザーを既存の SFTP サーバーから新しい SFTP サーバーに移行する計画がない場合、このセクションは無視してください。

サーバーのホストキーを誤って変更することは、破壊的な操作になり得えます。SFTP クライアントの設定方法によっては、信頼できるホストキーが存在しないというメッセージが表示されたり、すぐに障害が発生したり、脅迫的なプロンプトが表示されることがあります。接続を自動化するスクリプトがあれば、そのスクリプトも失敗する可能性が高いでしょう。

デフォルトでは、 は SFTP 対応サーバーのホストキー AWS Transfer Family を提供します。デフォルトのホストキーは、別のサーバーのホストキーに置き換えることができます。この操作をするのは、既存のユーザーを既存の SFTP 対応サーバーから新しい SFTP 対応サーバーに移動する計画がある場合のみに限定してください。

SFTP 対応サーバーの真正性を確認するプロンプトがユーザーに再度表示されないようにするには、オンプレミスサーバーのホストキーを SFTP 対応サーバーにインポートします。これにより、ユーザーは潜在的な man-in-the-middle 攻撃について警告を受けるのを防ぐことができます。

追加のセキュリティ対策として、ホストキーを定期的にローテーションすることもできます。

> **Note**
>
> Transfer Familyコンソールではすべてのサーバーのサーバーホストキーを指定して追加できますが、これらのキーはSFTPプロトコルを使用するサーバーでのみ役立ちます。

トピック

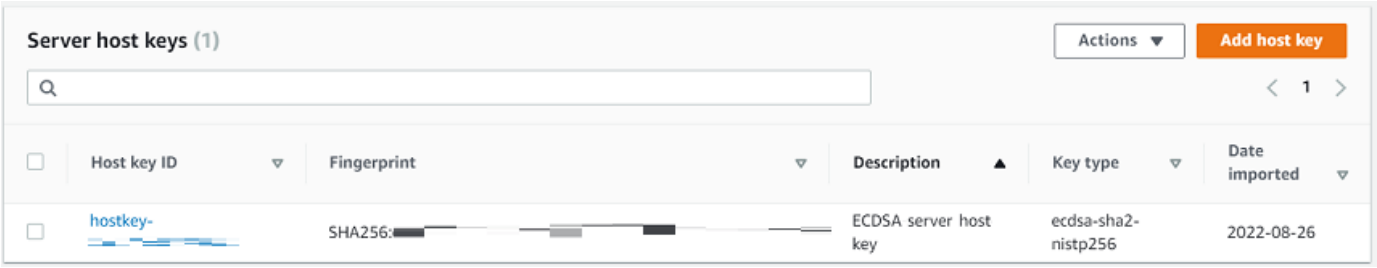
サーバーホストキーを追加する

AWS Transfer Family コンソールでは、サーバーホストキーを追加できます。異なる形式のホストキーを追加すると、クライアントが接続したときにサーバーを識別したり、セキュリティプロファイルを改善したりするのに役立ちます。たとえば、元のキーが RSA キーの場合、ECDSA キーを追加することができます。

> **Note**
>
> SFTP クライアントは、アクティブなサーバー鍵の 1 つと一致する最初の公開鍵を使用して接続します。

サーバーホストキーを追加するには

1. https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開きます。
2. 左側のナビゲーションペインで、[サーバー] を選択し、それから、SFTP プロトコルを使用するサーバーを選択します。
3. サーバーの詳細ページで [サーバーホストキー]セクションまでスクロールします。



4. 「ホストキーの追加」を選択します。

   「サーバーホストキーを追加」ページが表示されます。
5. 「サーバーホストキー」セクションに、クライアントが SFTP 対応サーバーを介してサーバーに接続するときにサーバーを識別するために使用する RSA、ECDSA、または ED25519 プライベートキーを入力します。

   > **Note**
   >
   > サーバーホストキーを作成するときは、必ず `-N ""` (パスフレーズなし) を指定してください。キーペアの生成方法の詳細については、macOS、Linux、または UNIX で SSH キーを作成するを参照してください。
6. (オプション) 複数のサーバーホストキーを区別するための説明を追加します。キーにタグを追加することもできます。
7. [Add key] (キーの追加) を選択します。[Server details] (サーバーの詳細) ページが再表示されます。

AWS Command Line Interface （ AWS CLI) を使用してホストキーを追加するには、 the section called “ImportHostKey” API オペレーションを使用して新しいホストキーを指定します。新しいサーバーを作成する場合、the section called “CreateServer” API オペレーションでパラメータとしてホストキーを指定します。 AWS CLI を使用して既存のホストキーの説明を更新することもできます。

次のimport-host-key AWS CLI コマンド例では、指定された SFTP 対応サーバーのホストキーをインポートします。

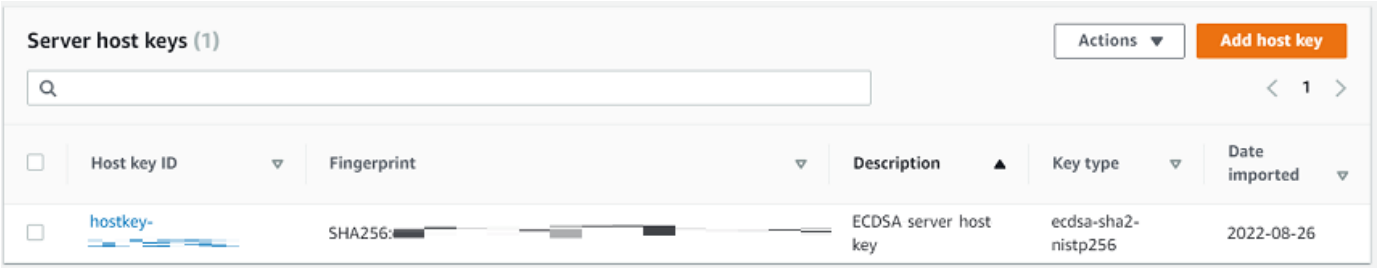
```
aws transfer import-host-key --description key-description --server-id your-server-id
 --host-key-body file://my-host-key
```

サーバーのホストキーを削除する

AWS Transfer Family コンソールでは、サーバーホストキーを削除できます。

サーバーのホストキーを削除するには

1. https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開きます。
2. 左側のナビゲーションペインで、[サーバー] を選択し、それから、SFTP プロトコルを使用するサーバーを選択します。
3. サーバーの詳細ページで [サーバーホストキー]セクションまでスクロールします。



4. 「サーバーホストキー」セクションでキーを選択し、「アクション」で「削除」を選択します。
5. 表示される確認ダイアログボックスで **delete** と入力し、「Delete」を選択してホストキーを削除することを確認します。

ホストキーが「サーバー」ページから削除されます。

を使用してホストキーを削除するには AWS CLI、 the section called “DeleteHostKey” API オペレーションを使用し、サーバー ID とホストキー ID を指定します。

次のdelete-host-key AWS CLI コマンド例では、指定された SFTP 対応サーバーのホストキーを削除します。

```
aws transfer delete-host-key --server-id your-server-id --host-key-id your-host-key-id
```

サーバーのホストキーをローテーションする

サーバーホストキーは定期的にローテーションできます。

クライアントがサーバーホストキーを選択する方法

Transfer Family が適用するサーバーキーを選択する方法は、ここで説明するように、SFTP クライアントの条件によって異なります。古いキーが 1 つあり、新しいキーが 1 つあることを前提としています。

- SFTP クライアントには、サーバー用の以前のパブリックホストキーがありません。クライアントがサーバーに初めて接続すると、次のいずれかが発生します。
  - クライアントが接続に失敗するように設定されている場合、クライアントは接続に失敗します。
  - または、クライアントは使用可能なアルゴリズムに一致する最初のキーを選択し、そのキーを信頼できるかどうかをユーザーに尋ねます。その場合、クライアントはknown_hostsファイル (またはクライアントが信頼の決定を記録するために使用するローカル設定ファイルまたはリソース) を自動更新し、そのキーを入力します。
- SFTP クライアントには、known_hostsファイルに古いキーがあります。クライアントは、新しいキーが存在する場合でも、このキーのアルゴリズムまたは別のアルゴリズムにこのキーを使用することを好みます。これは、クライアントが known_hosts ファイルにあるキーに対する信頼度が高いためです。
- SFTP クライアントは、キーファイルに新しいknown_hostsキー (使用可能なアルゴリズムのいずれか) を持っています。クライアントは古いキーを無視します。古いキーは信頼されておらず、新しいキーを使用するためです。
- SFTP クライアントには、known_hostsファイル内に両方のキーがあります。クライアントは、サーバーが提供する使用可能なキーのリストと一致するインデックスで最初のキーを選択します。

Transfer Family は、SFTP クライアントがknown_hostsファイル内のすべてのキーを持つことを優先します。これにより、Transfer Family サーバーへの接続時に最も柔軟性が高くなります。キーローテーションは、同じ Transfer Family サーバーの known_hosts ファイルに複数のエントリが存在する可能性があることに基づいています。

サーバーホストキーのローテーション手順

例として、Transfer Family サーバーに次のサーバーホストキーのセットを追加したとします。

サーバーホストキー

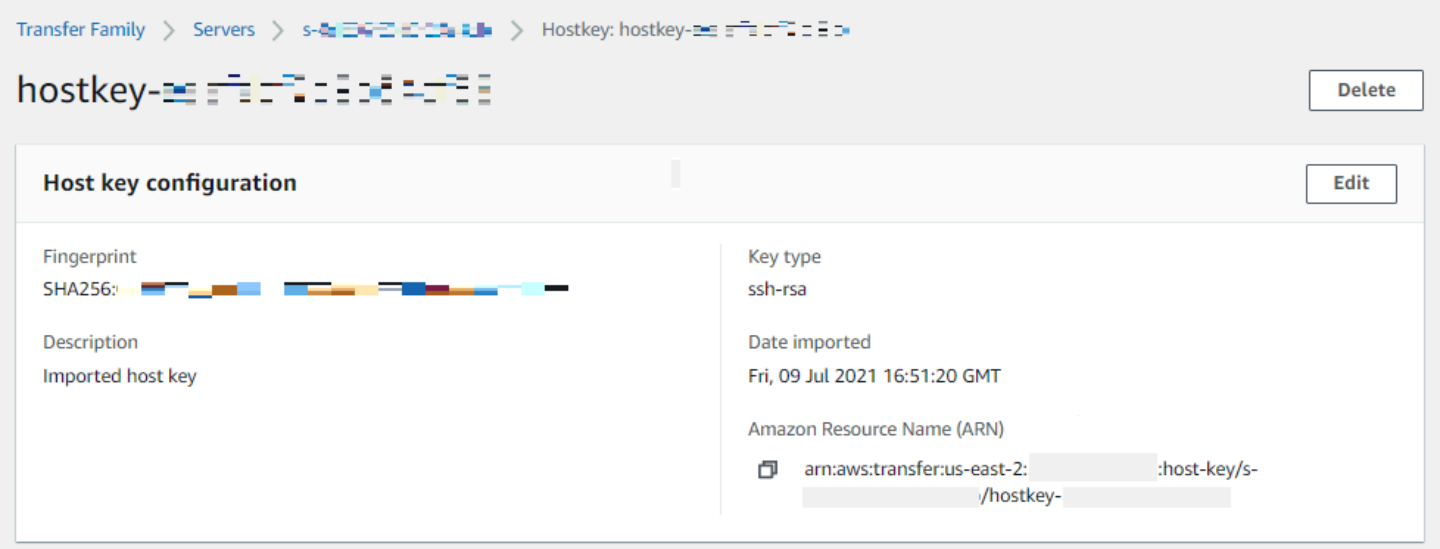
ホストキータイプ	サーバーに追加された日付
RSA	2020 年 4 月 1 日
ECDSA	2020 年 2 月 1 日
ED25519	2019年12月1日
RSA	2019 年 10 月 1 日
ECDSA	2019 年 6 月 1 日
ED25519	2019 年 3 月 1 日

サーバーのホストキーをローテーションするには

1. 新しいサーバーホストキーを追加します。この手順は、「サーバーホストキーを追加する」で説明されています。
2. 以前に追加した同じタイプのホストキーを 1 つ以上削除します。この手順は、「サーバーのホストキーを削除する」で説明されています。
3. 前述の で説明した動作に従って、すべてのキーが表示され、アクティブにすることができますクライアントがサーバーホストキーを選択する方法。

サーバーホストキーの追加情報

ホストキーを選択すると、そのキーの詳細を表示できます。



ホストキーは、サーバー詳細画面の「アクション」メニューから削除したり、説明を編集したりできます。ホストキーを選択し、メニューから適切なアクションを選択します。

コンソールで使用状況をモニターする

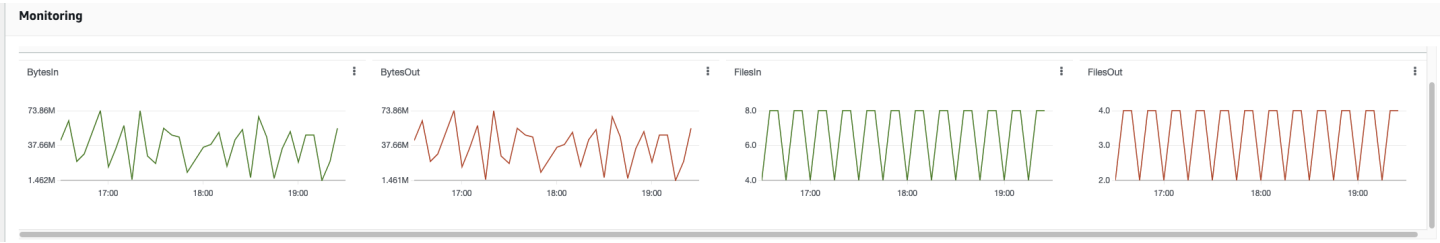
[Server details] (サーバーの詳細) ページでサーバーのメトリクスに関する情報を取得できます。これにより、ファイル転送のワークロードを 1 か所でモニタリングできます。一元化されたダッシュボードを使用して、パートナーと交換したファイルの数を追跡し、その詳細な使用状況を追跡できます。詳細については、「SFTP、FTPS、FTP サーバーの詳細を表示する」を参照してください。Transfer Family で使用できるメトリクスの説明を次の表に示します。

名前空間	メトリクス	説明
AWS/Transfer	BytesIn	サーバーに転送された総バイト数。 単位: カウント

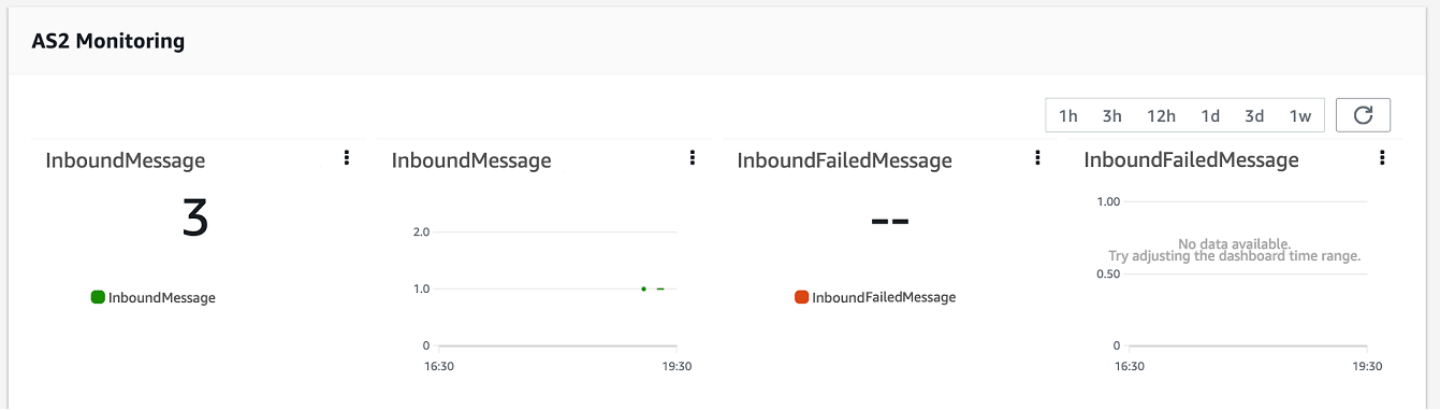
名前空間	メトリクス	説明
		期間: 5 分
	BytesOut	サーバーから転送された総バイト数。 単位: 個 期間: 5 分
	FilesIn	サーバーに転送されたファイルの合計数。 AS2 プロトコルを使用するサーバーの場合、このメトリックは受信したメッセージの数を表します。 単位: カウント 期間: 5 分
	FilesOut	サーバーから転送されたファイルの合計数。 単位: カウント 期間: 5 分
	InboundMessage	取引相手から正常に受信した AS2 メッセージの総数。 単位: カウント 期間: 5 分
	InboundFailedMessage	取引相手から正常に受信されなかった AS2 メッセージの総数。つまり、取引相手がメッセージを送信しましたが、Transfer Family サーバーはそれを正常に処理できませんでした。 単位: カウント 期間: 5 分

名前空間	メトリクス	説明
	OnUploadExecutionsStarted	サーバーで開始されたワークフロー実行の総数。 単位: カウント 期間 = 1 分
	OnUploadExecutionsSuccess	サーバーで成功したワークフロー実行の合計数。 単位: カウント 期間 = 1 分
	OnUploadExecutionsFailed	サーバーで失敗したワークフロー実行の合計数。 単位: カウント 期間 = 1 分

- [Monitoring] (モニタリング) セクションには、4 つの個別グラフが含まれています。これらのグラフには、受信バイト数、出力バイト数、受信ファイル数、出力ファイル数が表示されます。



AS2 プロトコルが有効になっているサーバーの場合、監視情報の下に AS2 監視セクションがあります。このセクションには、受信メッセージ数 (成功と失敗の両方) の詳細が含まれています。



選択したグラフを独自のウィンドウで開くには、展開アイコン
( )
を選択します。また、グラフの縦の省略記号アイコン
( )
をクリックすると、以下の項目を含むドロップダウンメニューを開くことができる：

- 「拡大」 — 選択したグラフを専用のウィンドウで開きます。
- 「リフレッシュ」 — 最新のデータでグラフを再読み込みします。
- メトリクスで表示 — Amazon で対応するメトリクスの詳細を開きます CloudWatch。
- ログの表示 — 対応するロググループを で開きます CloudWatch。

アクセスコントロールの管理

(IAM) ポリシーを使用して、 AWS Transfer Family リソースへのユーザーのアクセスを AWS Identity and Access Management 制御できます。IAM ポリシーは、通常は JSON 形式のステートメントで、リソースへの特定のレベルのアクセスを許可するものです。IAM ポリシーを使用して、ユーザーの実行を許可するファイル操作を定義します。IAM ポリシーを使用すると、ユーザーにアクセス権を与える Amazon S3 バケットも定義できます。ユーザー向けにこれらのポリシーを定義するには、IAM ポリシーとそれに関連付けられた信頼関係を含んでいる AWS Transfer Family の IAM ロールを作成します。

ユーザーごとにロールが割り当てられます。が AWS Transfer Family 使用する IAM ロールのタイプは、サービスロール と呼ばれます。ユーザーがサーバーにログインすると、 はユーザーにマッピングされた IAM ロールを AWS Transfer Family 引き受けます。Amazon S3 バケットへのユーザーアクセスを提供する IAM ロールの作成については、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。

IAM ポリシー内の特定のアクセス許可を使用して、Amazon S3 オブジェクトへの書き込み専用アクセスを許可できます。詳細については、「ファイルの書き込みとリストのみの権限を付与します。」を参照してください。

AWS ストレージブログには、最小特権アクセスを設定する方法の詳細が記載された投稿が含まれています。詳細については、AWS Transfer Family 「ワークフローでの最小特権アクセスの実装」を参照してください。

> **Note**
>
> Amazon S3 バケットが AWS Key Management Service （ AWS KMS) を使用して暗号化されている場合は、ポリシーで追加のアクセス許可を指定する必要があります。詳細については、「Amazon S3 でのデータ暗号化」を参照してください。さらに、IAM ユーザーガイドでもセッションポリシーに関する詳細をご覧いただけます。

トピック

- Amazon S3 バケットへの読み書きアクセスの許可
- Amazon S3 バケットのセッションポリシーの作成
- ユーザーによる S3 mkdir バケットでの実行を無効にする

Amazon S3 バケットへの読み書きアクセスの許可

このセクションでは、特定の Amazon S3 バケットへの読み取りと書き込みアクセスを許可する IAM ポリシーを作成する方法について説明します。この IAM ポリシーを持つ IAM ロールをユーザーに割り当てると、指定された Amazon S3 バケットへの読み書きアクセスがそのユーザーに付与されます。

以下のポリシーは、Amazon S3 バケットへのプログラムによる読み取り、書き込み、タグ付けアクセスを提供します。`GetObjectACL`および`PutObjectACL`ステートメントは、クロスアカウントアクセスを有効にする必要がある場合にのみ必要です。つまり、Transfer Family サーバーは、別のアカウントのバケットにアクセスする必要があります。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid":"ReadWriteS3",
      "Action": [
            "s3:ListBucket"
                ],
      "Effect": "Allow",
      "Resource": ["arn:aws:s3:::DOC-EXAMPLE-BUCKET"]
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:PutObject",
        "s3:GetObject",
        "s3:GetObjectTagging",
        "s3:DeleteObject",
        "s3:DeleteObjectVersion",
        "s3:GetObjectVersion",
        "s3:GetObjectVersionTagging",
        "s3:GetObjectACL",
        "s3:PutObjectACL"
      ],
      "Resource": ["arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"]
    }
  ]
}
```

ListBucket アクションには、バケット自体のアクセス許可が必要です。PUT、GET、および DELETE アクションにはオブジェクトのアクセス許可が必要です。これらは異なるリソースなので、異なるAmazonリソースネーム（ARN）を使って指定されます。

指定した Amazon S3 バケットの home プレフィックスのみにユーザーのアクセスをさらに制限するには、Amazon S3 バケットのセッションポリシーの作成 を参照してください。

Amazon S3 バケットのセッションポリシーの作成

セッションポリシーは、Amazon S3 バケットの特定の部分にユーザーを制限する AWS Identity and Access Management (IAM) ポリシーです。 Amazon S3 そのために、リアルタイムでアクセスが評価されます。

> **Note**
>
> セッションポリシーは Amazon S3 でのみ使用されます。Amazon EFS では、POSIX ファイルパーミッションを使用してアクセスを制限します。

Amazon S3 バケットの特定の部分への同じアクセス権をユーザーのグループに付与する必要がある場合は、セッションポリシーを使用できます。たとえば、あるユーザーのグループが home ディレクトリのみにアクセスする必要があるとします。そのユーザーのグループは同じ IAM ロールを共有します。

> **Note**
>
> セッションポリシーの最大長は 2048 文字です。詳細については、API リファレンスで CreateUser アクションの「ポリシーリクエストパラメータ」を参照してください。

セッションポリシーを作成するには、以下の IAM ポリシーのポリシー変数を使用します。

- ${transfer:HomeBucket}
- ${transfer:HomeDirectory}
- ${transfer:HomeFolder}
- ${transfer:UserName}

> **Important**
>
> 管理ポリシーで先に表示されている変数を先に表示することはできません。IAM ロールの定義のリストでこれらをポリシー変数として使用することもできません。このような変数は、IAM ポリシーで作成してから、ユーザーの設定時に直接指定します。また、セッションポリシーの ${aws:Username} 変数を使用することはできません。この変数は IAM ユーザー名を指し、 AWS Transfer Familyが要求するユーザー名ではありません。

次のコードは、セッションポリシーの例を示しています。

```
{
  "Version": "2012-10-17",
  "Statement": [
      {
          "Sid": "AllowListingOfUserFolder",
          "Action": [
              "s3:ListBucket"
          ],
          "Effect": "Allow",
          "Resource": [
              "arn:aws:s3:::${transfer:HomeBucket}"
          ],
          "Condition": {
              "StringLike": {
                  "s3:prefix": [
                      "${transfer:HomeFolder}/*",
                      "${transfer:HomeFolder}"
                  ]
              }
          }
      },
      {
          "Sid": "HomeDirObjectAccess",
          "Effect": "Allow",
          "Action": [
              "s3:PutObject",
              "s3:GetObject",
              "s3:DeleteObjectVersion",
              "s3:DeleteObject",
              "s3:GetObjectVersion",
              "s3:GetObjectACL",
```

```
                "s3:PutObjectACL"
            ],
            "Resource": "arn:aws:s3:::${transfer:HomeDirectory}/*"
        }
    ]
}
```

> **Note**
>
> 前述のポリシーの例では、ユーザーのホームディレクトリがディレクトリであることを示すために、末尾のスラッシュを含むように設定されていることを前提としています。一方、ユーザーの HomeDirectory の末尾にスラッシュを付けない場合、それをポリシーの一部として含める必要があります。

前のポリシー例では、transfer:HomeFolder、transfer:HomeBucket、およびtransfer:HomeDirectory ポリシーパラメータの使用に注目してください。これらのパラメータは、HomeDirectory「」および「」で説明HomeDirectoryされているように、ユーザー用に設定された に設定されますAPI Gateway メソッドを実装する。これらのパラメータには、次のような定義があります。

- transfer:HomeBucket パラメータは HomeDirectory の 1 つ目のコンポーネントに置き換わります。
- transfer:HomeFolder パラメータは HomeDirectory パラメータの残り部分に置き換わります。
- transfer:HomeDirectory パラメータの先頭に付いていた (/) が削除され、Resource ステートメントで S3 Amazon リソースネーム (ARN) の一部としてこのパラメータを使用できるようになりました。

> **Note**
>
> 論理ディレクトリを使用する場合 (つまり、ユーザーの homeDirectoryType が LOGICAL である場合) 、これらのポリシーパラメータ (HomeBucket、HomeDirectory、および HomeFolder) はサポートされません。

たとえば、Transfer Family ユーザー用に設定された HomeDirectory パラメータは /home/bob/amazon/stuff/ です。

- transfer:HomeBucket は /home に設定されます。
- transfer:HomeFolder は /bob/amazon/stuff/ に設定されます。
- transfer:HomeDirectory は home/bob/amazon/stuff/ になります。

1 つ目の "Sid" は、/home/bob/amazon/stuff/ から始まるすべてのディレクトリの一覧表示をユーザーに許可します。

2 つ目の "Sid" は、ユーザーの put およびその同じパス /home/bob/amazon/stuff/ への get アクセスを制限します。

前述のポリシーが使用されている場合、ユーザーがログインすると、ユーザーはホームディレクトリにあるオブジェクトにのみアクセスできます。接続時に、 はこれらの変数をユーザーに適した値 AWS Transfer Family に置き換えます。これによって、同じポリシードキュメントを複数のユーザーに簡単に適用できます。この方法により、IAM ロールのオーバーヘッドおよび Amazon S3 バケットへのユーザーのアクセスを管理するためのポリシー管理が軽減されます。

セッションポリシーを使用すると、業務要件に基づいて、ユーザーごとにアクセス権をカスタマイズすることもできます。詳細については、IAM ユーザーガイドの AssumeRole「、AssumeRoleWithSAML、および のアクセス許可 AssumeRoleWithWebIdentity」を参照してください。

> **Note**
>
> AWS Transfer Family は、ポリシーの Amazon リソースネーム (ARN) の代わりにポリシー JSON を保存します。したがって、IAM コンソールでポリシーを変更する場合、 AWS Transfer Family コンソールに戻り、最新のポリシーコンテンツでユーザーを更新する必要があります。[ユーザー設定] セクションの[ポリシー情報] タブでユーザーを更新できます。を使用している場合は AWS CLI、次のコマンドを使用してポリシーを更新できます。
>
> ```
> aws transfer update-user --server-id server --user-name user --policy \
>    "$(aws iam get-policy-version --policy-arn policy --version-id version --
> output json)"
> ```

ユーザーによる S3 mkdir バケットでの実行を無効にする

ユーザーによる Amazon S3 バケットでのディレクトリの作成を無効にできます。そのためには、s3:PutObject アクションを許可するが、キーが「/」(スラッシュ) で終わる場合にこのアクションを拒否する IAM ポリシーを作成します。次のポリシー例では、ユーザーに Amazon S3 バケットへのファイルのアップロードが許可されますが、Amazon S3 mkdir バケットでのコマンドは拒否されます。

```
{
   "Sid":"DenyMkdir",
   "Action":[
      "s3:PutObject"
   ],
   "Effect":"Deny",
   "Resource":[
      "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*/",
      "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*/*"
    ]
}
```

> **Note**
>
> 2 行目のリソース行では、ユーザーがput my-file DOC-EXAMPLE-BUCKET/new-folder/my-fileなどのコマンドを実行してもサブフォルダを作成できなくなります。

AWS Transfer Family のログ記録

AWS Transfer Family は、 AWS CloudTrailと Amazon の両方と統合 CloudTrail され CloudWatch、異なる補完的な目的 CloudWatch を提供します。

- CloudTrail は、 内で実行されたアクションのレコードを作成する AWSのサービスですAWS アカウント。コンソールのサインイン、AWS Command Line Interfaceコマンド、SDK/API コールなどのアクティビティの API コールを継続的にモニタリングして記録します。これにより、AWS環境内のすべてのアクティビティの履歴を提供することで、監査、アクセス管理、規制コンプライアンス CloudTrail により、誰がいつどこでどのようなアクションを実行したかのログを保持できます。詳細については、AWS CloudTrail「 ユーザーガイド」を参照してください。
- CloudWatch は、 AWSリソースとアプリケーションのモニタリングサービスです。メトリクスとログを収集して、リソースの使用率、アプリケーションのパフォーマンス、システム全体のヘルスを可視化し、問題のトラブルシューティング、アラームの設定、自動スケーリングなどの CloudWatch 運用タスクを提供します。詳細については、「Amazon ユーザーガイド CloudWatch」を参照してください。

トピック

- AWS CloudTrail の ログ記録 AWS Transfer Family
- の Amazon CloudWatch ログ記録 AWS Transfer Family

AWS CloudTrail の ログ記録 AWS Transfer Family

AWS Transfer Family は、 のユーザーAWS CloudTrail、ロール、または AWSのサービスによって実行されたアクションを記録するサービスである と統合されていますAWS Transfer Family。 は、 のすべての API コールをイベントAWS Transfer Familyとして CloudTrail キャプチャします。キャプチャされたコールには、AWS Transfer Family コンソールのコールと、AWS Transfer Family API オペレーションへのコードのコールが含まれます。

証跡は、指定した Amazon S3 バケットにイベントをログファイルとして配信できるようにする設定です。 CloudTrail ログファイルには、1 つ以上のログエントリが含まれます。イベントは任意の送信元からの単一のリクエストを表し、リクエストされたアクション、アクションの日時、リクエストパラメータなどに関する情報が含まれます。 CloudTrail ログファイルは、パブリック API コールの順序付けられたスタックトレースではないため、特定の順序では表示されません。

AWSのイベントなど、AWS Transfer Familyアカウントのイベントの継続的なレコードについては、追跡を作成します。証跡により、 はログファイル CloudTrail を Amazon S3 バケットに配信できます。デフォルトでは、コンソールで証跡を作成すると、すべての AWS リージョンに証跡が適用されます。追跡は、AWS パーティションのすべてのリージョンからのイベントをログに記録し、指定した Amazon S3 バケットにログファイルを配信します。さらに、 CloudTrail ログで収集されたデータをより詳細に分析し、それに基づく対応を行うように他の AWSサービスを設定できます。詳細については、次を参照してください:

- 「証跡作成の概要」
- CloudTrail でサポートされているサービスと統合
- の Amazon SNS 通知の設定 CloudTrail
- 複数のリージョンからの CloudTrail ログファイルの受信と複数のアカウントからの CloudTrail ログファイルの受信

すべての AWS Transfer Family アクションは によってログに記録 CloudTrail され、 Actions に文書化されますAPI reference。例えば、、、および `StopServer`アクションを呼び出すと`ListUsers`と`CreateServer`、 CloudTrail ログファイルにエントリが生成されます。

各イベントまたはログエントリには、誰がリクエストを生成したかという情報が含まれます。ID 情報は次の判断に役立ちます。

- リクエストが、ルートと AWS Identity and Access Management ユーザー認証情報のどちらを使用して送信されたか。
- リクエストが、ロールとフェデレーションユーザーの一時的なセキュリティ認証情報のどちらを使用して送信されたか。
- リクエストが、別の AWS サービスによって送信されたかどうか。

詳細については、「CloudTrail userIdentity 要素」を参照してください。

証跡を作成する場合は、 の CloudTrail イベントなど、Amazon S3 バケットへのイベントの継続的な配信を有効にすることができますAWS Transfer Family。証跡を設定しない場合でも、コンソールのイベント履歴 で最新の CloudTrail イベントを表示できます。

で収集された情報を使用して CloudTrail、 に対するリクエストAWS Transfer Family、リクエスト元の IP アドレス、リクエスト者、リクエスト日時などの詳細を確認できます。

の詳細については CloudTrail、「 AWS CloudTrailユーザーガイド」を参照してください。

トピック

AWS CloudTrail ログ記録の有効化

AWS CloudTrail を使用して AWS Transfer Family API コールをモニタリングできます。API コールをモニタリングすることで、有益なセキュリティ情報や動作情報を取得できます。Amazon S3 オブジェクトレベルのログ記録が有効になっている場合、RoleSessionName は [Requeser] (リクエスタ) フィールドに [AWS:Role Unique Identifier]/username.sessionid@server-id として含まれます。AWS Identity and Access Management (IAM) ロールの一意の識別子の詳細については、AWS Identity and Access Management ユーザーガイドの「一意の識別子」を参照してください。

> **Important**
>
> RoleSessionName の最大長は 64 文字です。RoleSessionName が長い場合、server-id は切り詰められます。

サーバーを作成するためのログエントリの例

次の例は、 CreateServerアクションを示す CloudTrail ログエントリ (JSON 形式) を示しています。

```
{
    "eventVersion": "1.09",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AAAA4FFF5HHHHH6NNWWWW:user1",
        "arn": "arn:aws:sts::123456789102:assumed-role/Admin/user1",
        "accountId": "123456789102",
        "accessKeyId": "AAAA52C2WWWWW3BB4Z",
        "sessionContext": {
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2018-12-18T20:03:57Z"
            },
            "sessionIssuer": {
```

```
                "type": "Role",
                "principalId": "AAAA4FFF5HHHHH6NNWWWW",
                "arn": "arn:aws:iam::123456789102:role/Admin",
                "accountId": "123456789102",
                "userName": "Admin"
            }
        }
    },
    "eventTime": "2024-02-05T19:18:53Z",
    "eventSource": "transfer.amazonaws.com",
    "eventName": "CreateServer",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "11.22.1.2",
    "userAgent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
 like Gecko) Chrome/121.0.0.0 Safari/537.36",
    "requestParameters": {
        "domain": "S3",
        "hostKey": "HIDDEN_DUE_TO_SECURITY_REASONS",
        "protocols": [
            "SFTP"
        ],
        "protocolDetails": {
            "passiveIp": "AUTO",
            "tlsSessionResumptionMode": "ENFORCED",
            "setStatOption": "DEFAULT"
        },
        "securityPolicyName": "TransferSecurityPolicy-2020-06",
        "s3StorageOptions": {
            "directoryListingOptimization": "ENABLED"
        }
    },
    "responseElements": {
        "serverId": "s-1234abcd5678efghi"
    },
    "requestID": "6fe7e9b1-72fc-45b0-a7f9-5840268aeadf",
    "eventID": "4781364f-7c1e-464e-9598-52d06aa9e63a",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "123456789102",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
```

```
        "clientProvidedHostHeader": "transfer.us-east-1.amazonaws.com"
    },
    "sessionCredentialFromConsole": "true"
}
```

の Amazon CloudWatch ログ記録 AWS Transfer Family

Amazon は、 AWS Transfer Family リソースと で実行しているアプリケーションを AWS リアルタイムで CloudWatch モニタリングします。 CloudWatch を使用して、リソースとアプリケーションに対して測定できる変数であるメトリクスを収集および追跡できます。

CloudWatch ホームページには、Transfer Family と、使用する他のすべての AWS サービスに関するメトリクスが自動的に表示されます。さらに、カスタムダッシュボードを作成してカスタムアプリケーションのメトリクスを表示したり、選択したメトリクスのカスタムコレクションを表示したりできます。

メトリクスを監視し、しきい値を超過したときに通知を送信したり、モニターリングしているリソースを自動的に変更したりするアラームを作成できます。例えば、Transfer Family サーバーに転送されるファイルをモニタリングし、そのデータを使用して、増加した負荷を処理するために追加のサーバーをデプロイする必要があるかどうかを判断できます。このデータを使用して、使用頻度の低いインスタンスを停止または削除することで、コストを節約することもできます。

Transfer Family の CloudWatch ログ記録のタイプ

Transfer Family には、イベントを に記録する 2 つの方法があります CloudWatch。

- JSON 構造化ログ記録
- ログ記録ロールによるログ記録

Transfer Family サーバーでは、希望するログ記録メカニズムを選択できます。コネクタとワークフローでは、ログ記録ロールのみがサポートされます。

JSON 構造化ログ記録

サーバーイベントのログ記録には、JSON 構造化ログ記録を使用することをお勧めします。これにより、ログクエリを可能にするより包括的な CloudWatch ログ記録形式が提供されます。このタイプのログ記録では、サーバーを作成する (またはサーバーのログ記録設定を編集する) ユーザーの IAM ポリシーに、次のアクセス許可が含まれている必要があります。

- logs:CreateLogDelivery
- logs:DeleteLogDelivery
- logs:DescribeLogGroups
- logs:DescribeResourcePolicies
- logs:GetLogDelivery
- logs:ListLogDeliveries
- logs:PutResourcePolicy
- logs:UpdateLogDelivery

以下は、ポリシーの例です。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogDelivery",
                "logs:GetLogDelivery",
                "logs:UpdateLogDelivery",
                "logs:DeleteLogDelivery",
                "logs:ListLogDeliveries",
                "logs:PutResourcePolicy",
                "logs:DescribeResourcePolicies",
                "logs:DescribeLogGroups"
            ],
            "Resource": "arn:aws:logs:region-id:AWS #####:log-group:/aws/transfer/*"
        }
    ]
}
```

JSON 構造化ログの設定の詳細については、「」を参照してくださいサーバーのログ記録の作成、更新、表示。

ログ記録ロール

サーバーにアタッチされているマネージドワークフローのイベントとコネクタのイベントをログに記録するには、ログ記録ロールを指定する必要があります。アクセス権を設定するには、リソースベー

スの IAM ポリシー、およびアクセス情報を提供する IAM ロールを作成します。サーバーイベントをログ AWS アカウント 記録できる のポリシーの例を次に示します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogStream",
                "logs:DescribeLogStreams",
                "logs:CreateLogGroup",
                "logs:PutLogEvents"
            ],
            "Resource": "arn:aws:logs:*:*:log-group:/aws/transfer/*"
        }
    ]
}
```

ワークフローイベントをログに記録するログ記録ロールの設定の詳細については、「」を参照してくださいワークフローのログ記録の管理。

トピック

- サーバーのログ記録の作成、更新、表示
- ワークフローのログ記録の管理
- CloudWatch ログ記録ロールを設定する
- Transfer Family ログストリームの表示
- Amazon CloudWatch アラームの作成
- Amazon S3 API コールを S3 アクセスログに記録する
- 混乱する代理問題の防止
- CloudWatch Transfer Family の ログ構造
- CloudWatch ログエントリの例
- Transfer Family の CloudWatch メトリクスの使用
- AWS User Notifications で を使用する AWS Transfer Family
- クエリを使用したログエントリのフィルタリング

サーバーのログ記録の作成、更新、表示

すべての AWS Transfer Family サーバーで、ログ記録の 2 つのオプション LoggingRole (サーバーにアタッチされたワークフローのログ記録に使用) または を選択できますStructuredLogDestinations。StructuredLogDestinations を使用すると、次のような利点があります。

- 構造化された JSON 形式でログを受信します。
- JSON 形式のフィールドを自動的に検出する Amazon CloudWatch Logs Insights を使用してログをクエリします。
- AWS Transfer Family リソース間でロググループを共有すると、複数のサーバーからのログストリームを 1 つのロググループに結合できるため、モニタリング設定とログ保持設定の管理が容易になります。
- ダッシュボードに追加 CloudWatchできる集計メトリクスと視覚化を作成します。
- ロググループを使用して使用状況とパフォーマンスデータを追跡し、統合されたログメトリクス、ビジュアライゼーション、ダッシュボードを作成します。

LoggingRoleまたはStructuredLogDestinationsのオプションは個別に設定および制御されます。サーバーごとに、一方または両方のログ記録方法を設定することも、ログ記録をまったく行わないようにサーバーを構成することもできます (ただし、これは推奨しません)。

Transfer Family コンソールを使用して新しいサーバーを作成する場合、ログ記録がデフォルトで有効になります。サーバーを作成したら、UpdateServer API コールを使用してロギング設定を変更できます。詳細については、StructuredLog「送信先」を参照してください。

現在、ワークフローでロギングを有効にする場合は、ロギングロールを指定する必要があります。

- CreateServerまたは UpdateServer API コールを使用してワークフローをサーバーに関連付ける場合、システムはロギングロールを自動的に作成しません。ワークフローイベントをログに記録する場合は、ロギングロールをサーバーに明示的にアタッチする必要があります。
- Transfer Family コンソールを使用してサーバーを作成し、ワークフローをアタッチすると、ログは名前にサーバー ID を含むロググループに送信されます。形式は/aws/transfer/*server-id*で、例えば、/aws/transfer/s-1111aaaa2222bbbb3です。サーバーログは、同じロググループに送信することも、別のロググループに送信することもできます。

コンソールでサーバーを作成および編集する際のロギングに関する考慮事項

- コンソールによって作成された新しいサーバーは、ワークフローがサーバーにアタッチされていない限り、構造化された JSON ロギングのみをサポートします。
- コンソールで作成する新しいサーバーでは、ログを記録しないという選択肢はありません。
- 既存のサーバーでは、いつでもコンソールによって構造化された JSON ロギングを有効にすることができます。
- コンソールによって構造化された JSON ロギングを有効にすると、既存のロギング方法が無効になるため、顧客への二重請求を防ぐことができます。ワークフローがサーバーにアタッチする場合は例外です。
- 構造化 JSON ロギングを有効にした場合、後でコンソールから無効にすることはできません。
- 構造化 JSON ロギングを有効にすると、コンソールからロググループの送信先をいつでも変更できます。
- 構造化 JSON ロギングを有効にした場合、API を使用して両方のロギングタイプを有効にしていれば、コンソールからロギングロールを編集することはできません。例外は、サーバーにワークフローがアタッチされている場合です。ただし、ロギングロールは引き続き[その他の詳細]に表示されます。

API または SDK を使用してサーバーを作成および編集する際のロギングに関する考慮事項

- API を使用して新しいサーバーを作成する場合、ロギングのどちらかまたは両方のタイプを設定することも、ロギングなしを選択することもできます。
- 既存のサーバーでは、構造化 JSON ロギングをいつでも有効または無効にできます。
- ロググループは、API を使用していつでも変更できます。
- ログ記録の役割は、API を使用していつでも変更できます。

構造化ログ記録を有効にするには、以下のアクセス許可でアカウントにログインする必要があります

- `logs:CreateLogDelivery`
- `logs:DeleteLogDelivery`
- `logs:DescribeLogGroups`
- `logs:DescribeResourcePolicies`
- `logs:GetLogDelivery`
- `logs:ListLogDeliveries`
- `logs:PutResourcePolicy`

- `logs:UpdateLogDelivery`

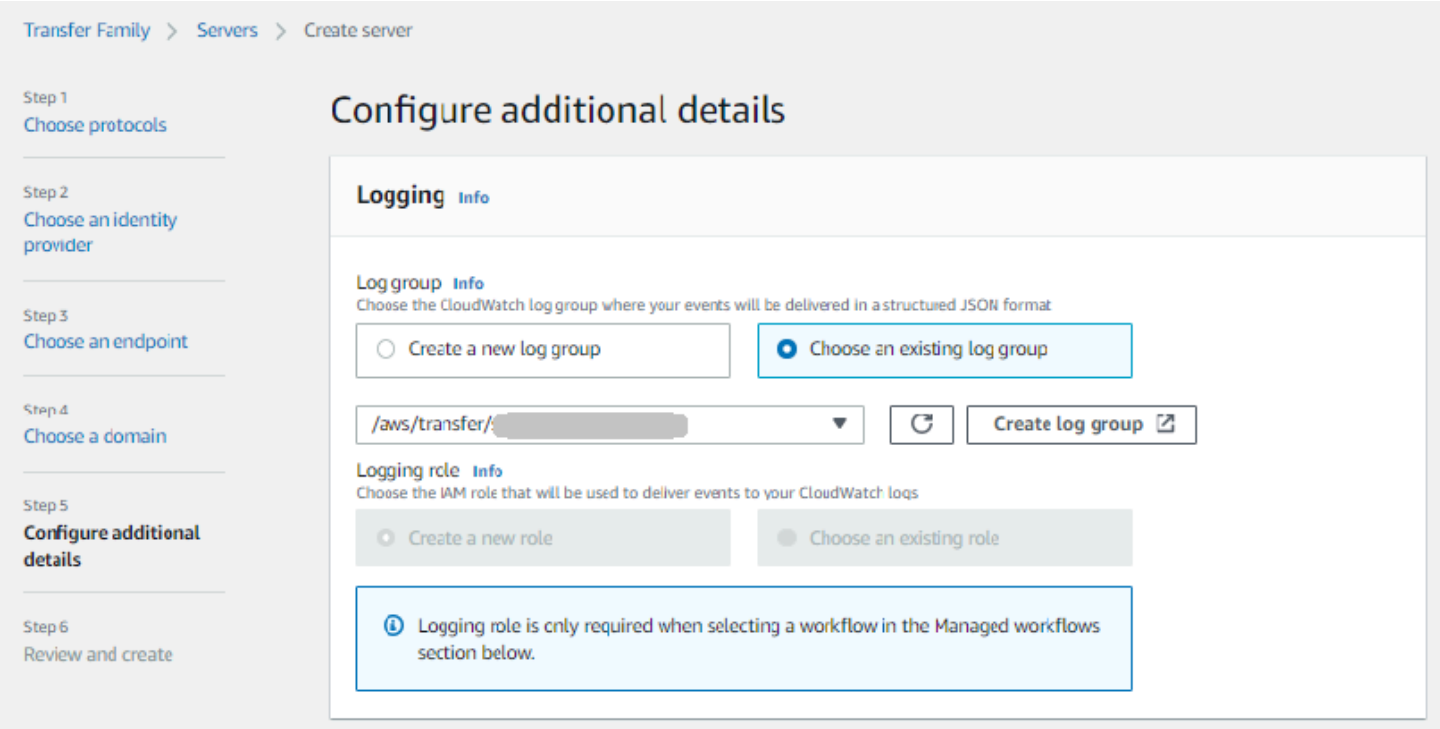
ポリシーの例は、「」セクションで入手できます CloudWatch ログ記録ロールを設定する。

トピック

- サーバーのログ記録の作成
- サーバーのロギングを更新します。
- サーバー設定の表示

サーバーのログ記録の作成

新しいサーバーを作成する場合、「詳細の設定」ページで、既存のロググループを指定するか、新しいロググループを作成できます。



ロググループの作成 を選択すると、 CloudWatch コンソール (https://console.aws.amazon.com/cloudwatch/) がロググループの作成ページを開きます。詳細については、「 ログ」の CloudWatch「ロググループの作成」を参照してください。

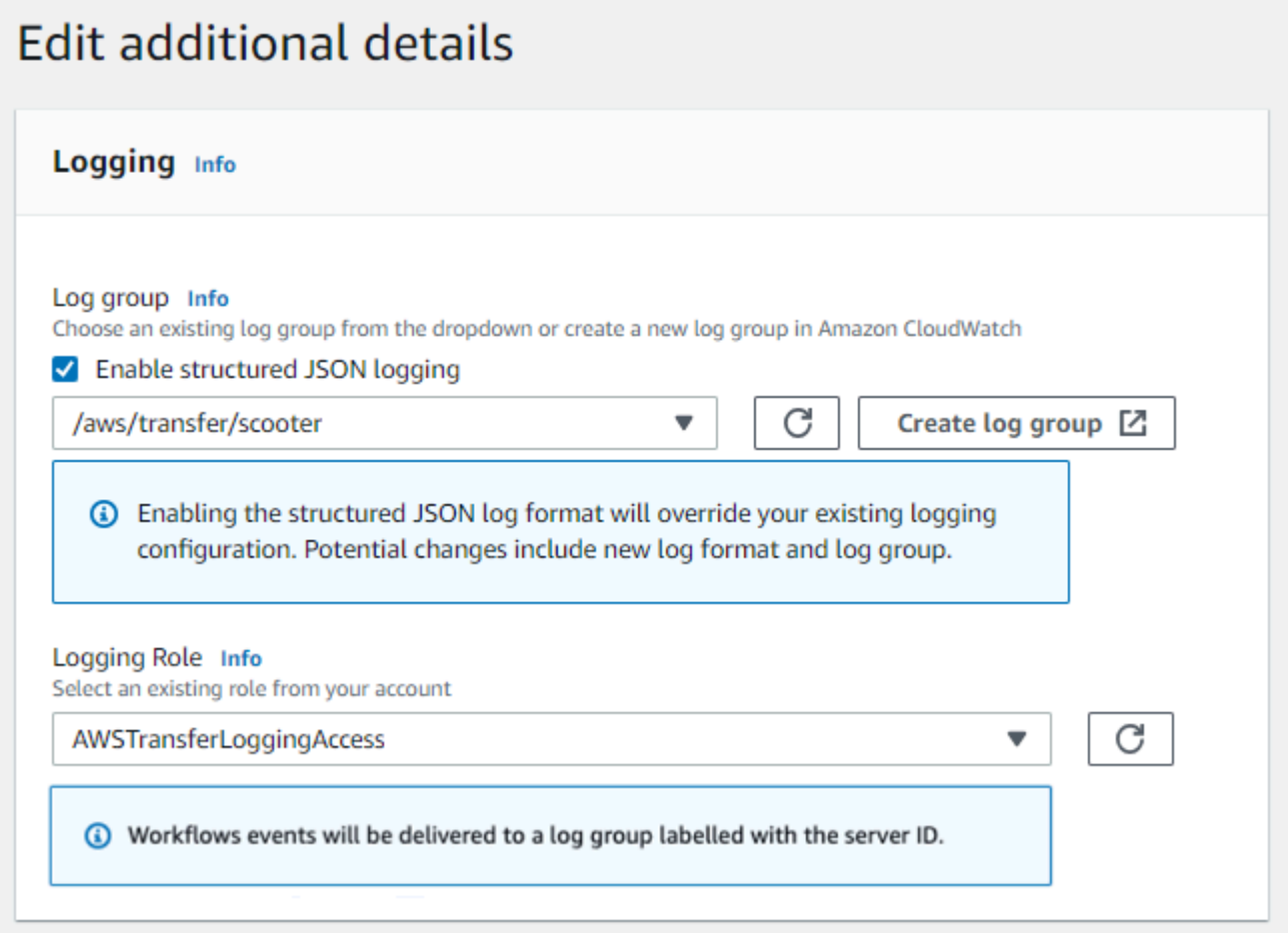
サーバーのロギングを更新します。

ロギングの詳細は、更新のシナリオによって異なります。

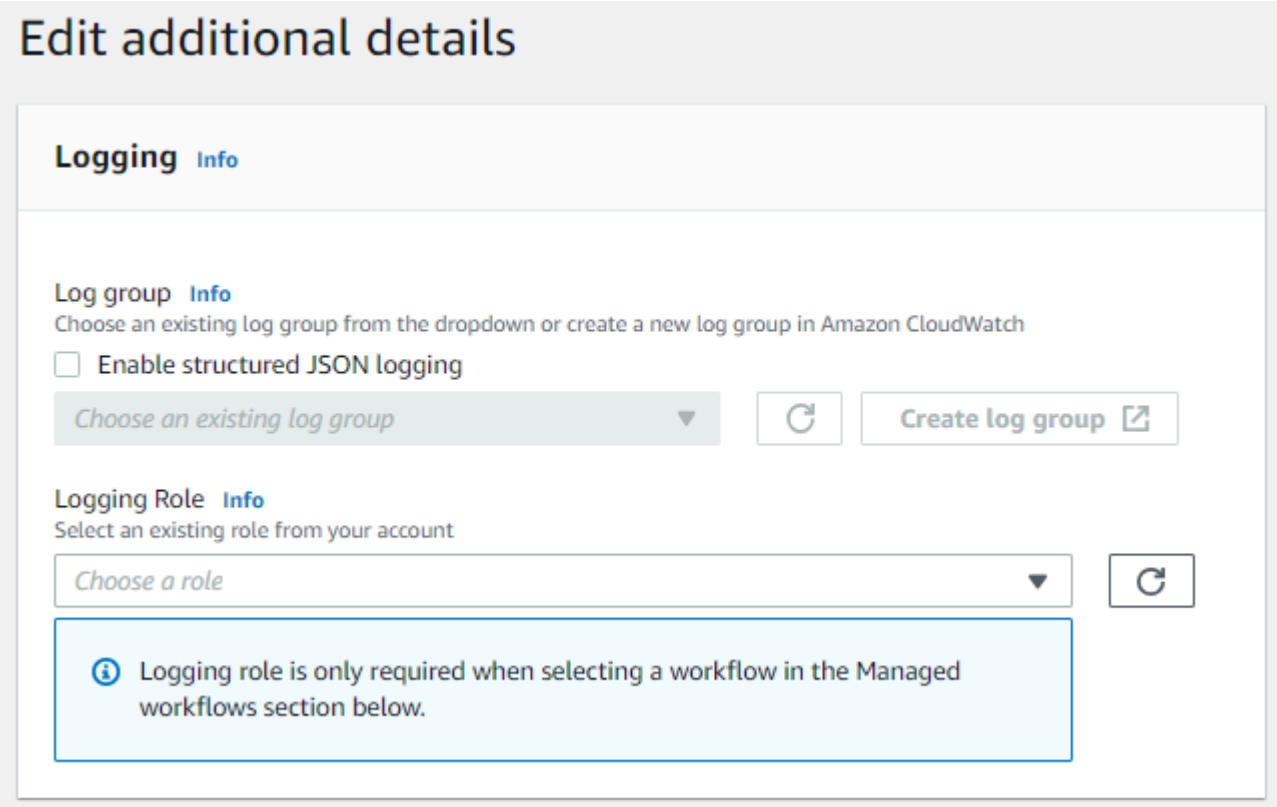
> **Note**
>
> 構造化JSONロギングを選択した場合、まれに、TTransfer Family ilyが古い形式でのロギングを停止し、新しいJSON形式でのロギングを開始するまでに時間がかかるという遅延が発生することがあります。その結果、イベントが記録されないことがあります。サービスが中断されることはありませんが、ログ記録方法を変更してから最初の 1 時間はログが削除される可能性があるため、ファイルの転送には注意が必要です。

既存のサーバーを編集する場合、オプションはサーバーの状態によって異なります。

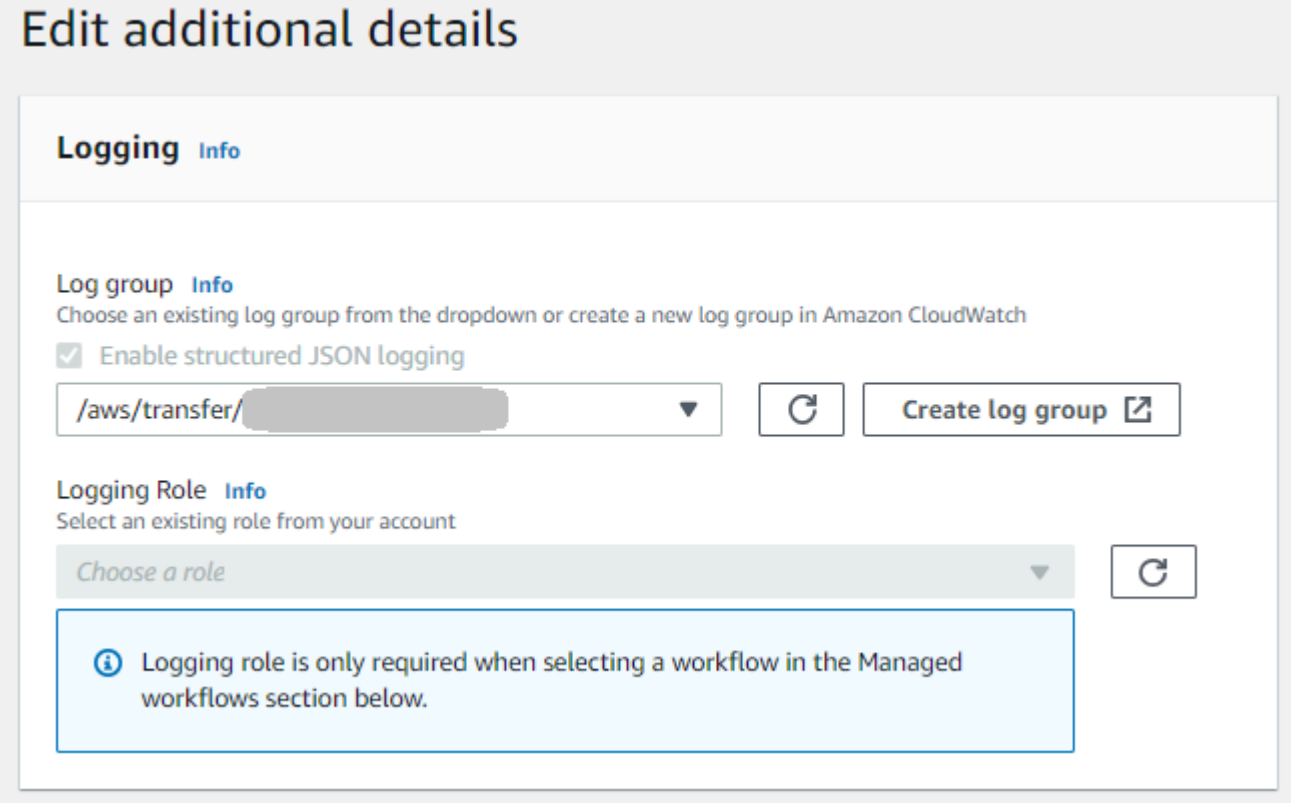
- サーバーではすでにロギングロールが有効になっていますが、Structured JSON ロギングは有効になっていません。



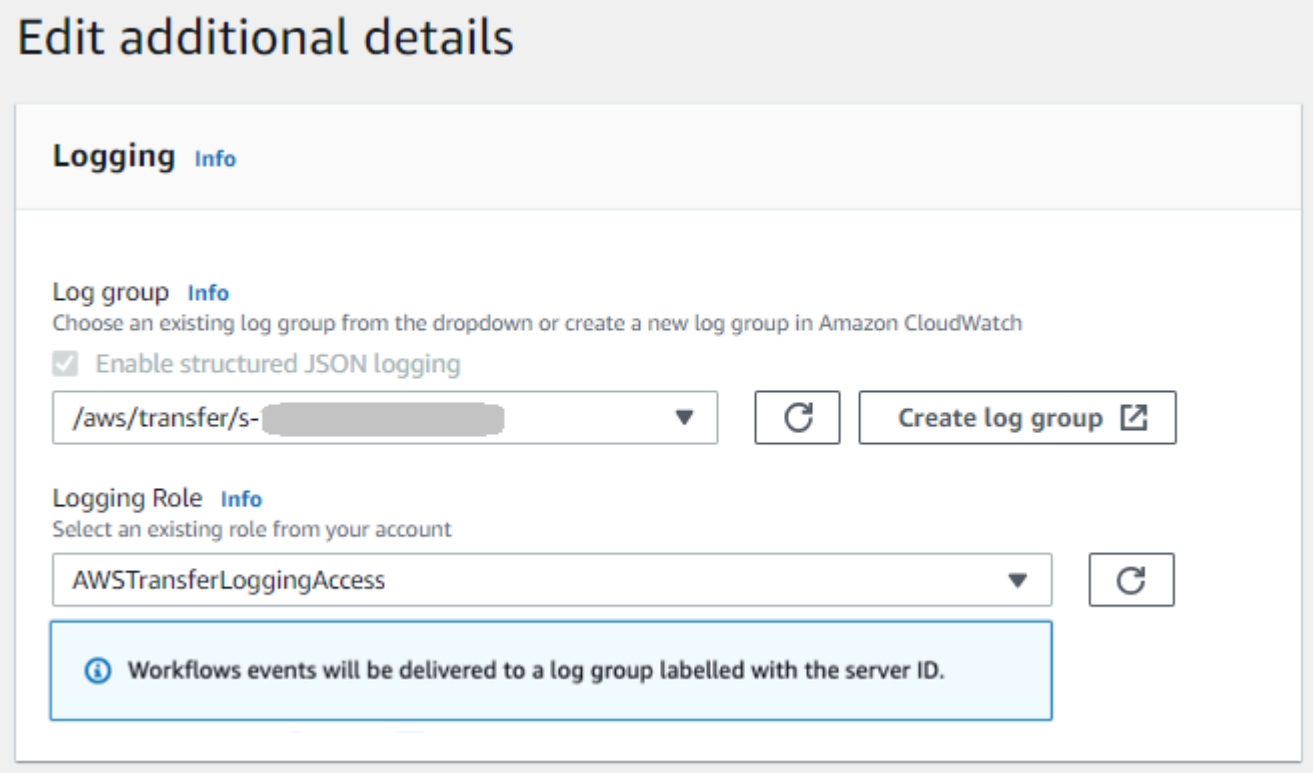
- サーバーでログ記録が有効になっていません。



- サーバーでは既に Structured JSON ロギングが有効になっていますが、ロギングロールは指定されていません。



- サーバーでは既に Structured JSON ロギングが有効になっており、ロギングロールも指定されています。

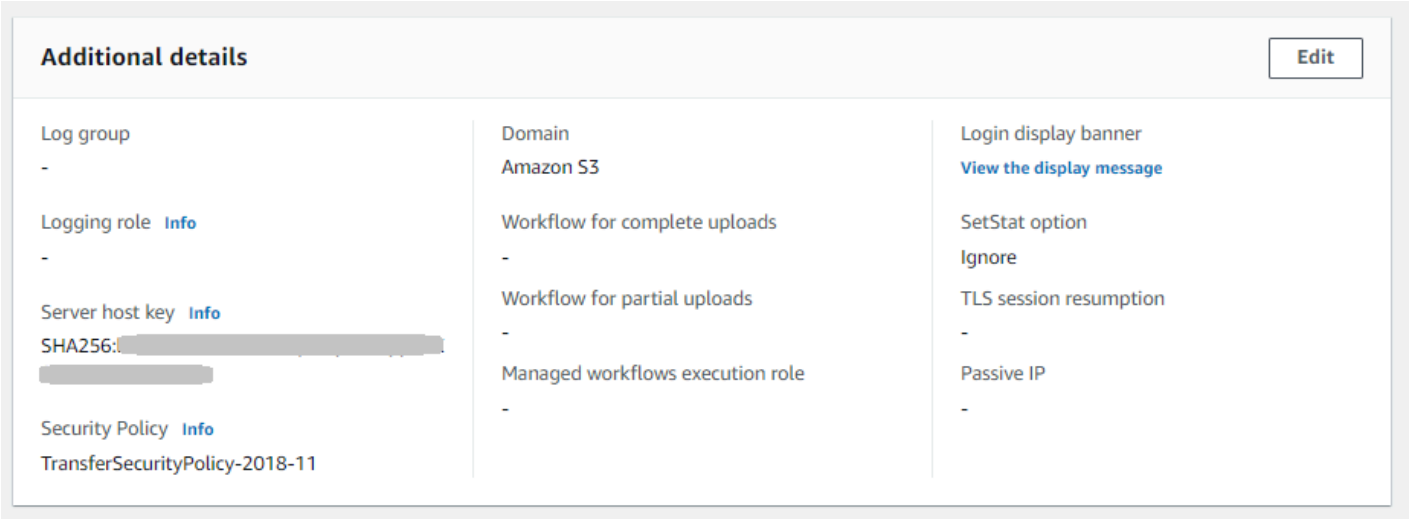


サーバー設定の表示

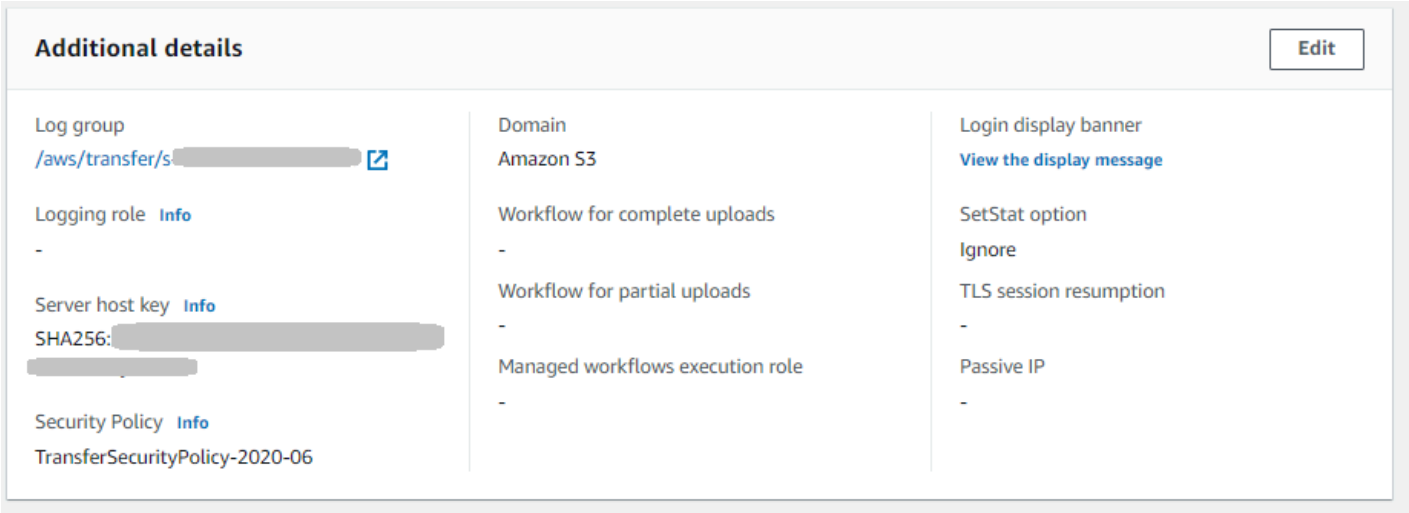
サーバー設定ページの詳細は、シナリオによって異なります。

シナリオによっては、サーバー設定ページが以下のいずれかの例のようになる場合があります。

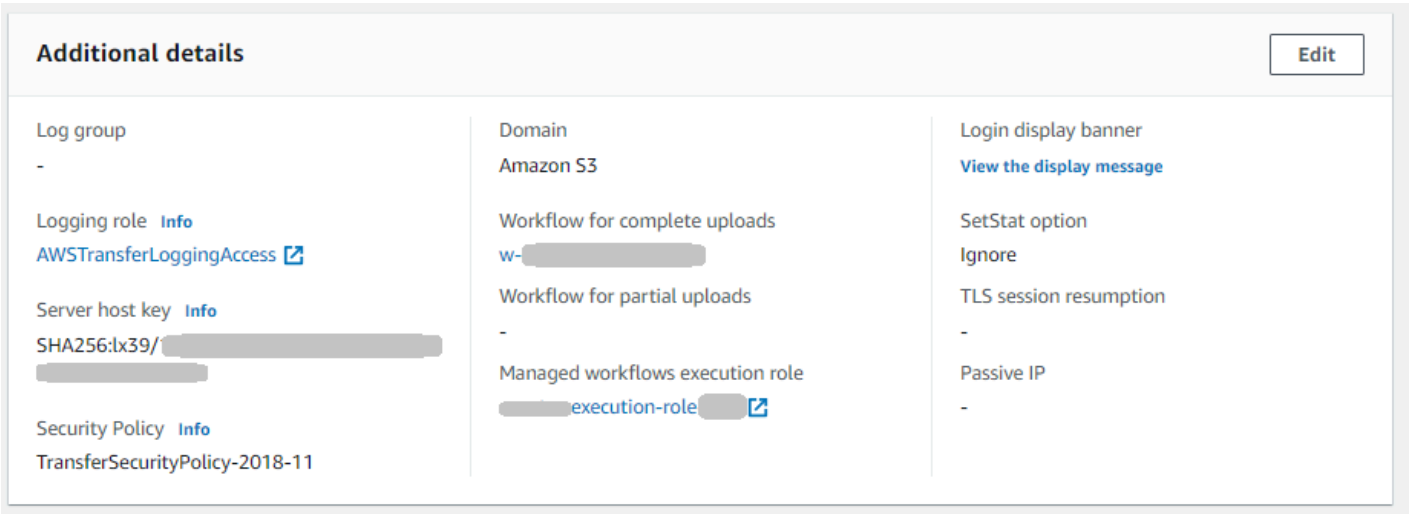
- ログは有効になっていません。



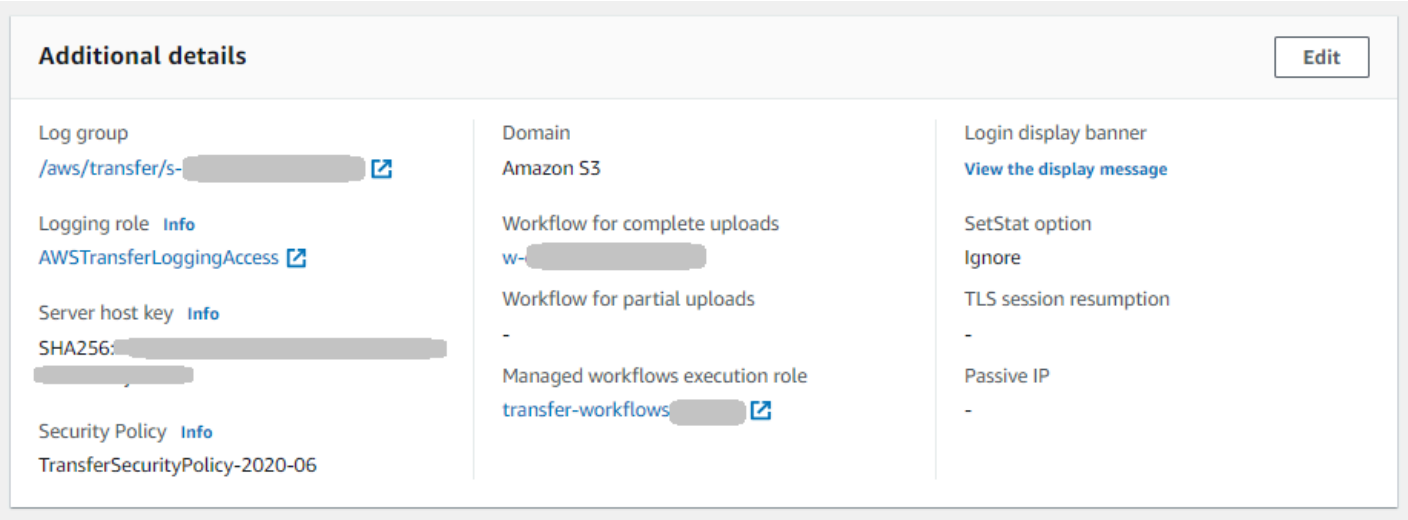
- 構造化 JSON ロギングが有効になっています。



- ロギングロールは有効ですが、構造化 JSON ロギングは有効になっていません。



- どちらの種類のロギング (ロギングロールと構造化 JSON ロギング) も有効になっています。



ワークフローのログ記録の管理

CloudWatch は、ワークフローの進行状況と結果の監査とログ記録を統合します。さらに、 はワークフローの複数のメトリクス AWS Transfer Family を提供します。過去 1 分間に開始された、正常に完了した、失敗したワークフロー実行の数を示す指標を表示できます。Transfer Family のすべての CloudWatch メトリクスについては、「」で説明されていますTransfer Family の CloudWatch メトリクスの使用。

ワークフローの Amazon CloudWatch ログを表示する

1. https://console.aws.amazon.com/cloudwatch/ で Amazon CloudWatch コンソールを開きます。
2. 左側のナビゲーションペインで、「ログ」を選択し、「ロググループ」を選択します。
3. ロググループページのナビゲーションバーで、 AWS Transfer Family サーバーに適したリージョンを選択します。
4. サーバーに対応するロググループを選択します。

   例えば、サーバー ID が `s-1234567890abcdef0` であれば、ロググループは `/aws/transfer/s-1234567890abcdef0` です。
5. サーバーのロググループの詳細ページに、最新のログストリームが表示されます。調査対象のユーザーには 2 つのログストリームがあります。

   - Secure Shell (SSH) File Transfer Protocol (SFTP) セッションごとに 1 つずつ。

- 1 つはサーバーで実行中のワークフロー用です。ワークフローのログストリームの形式は *username*.*workflowID*.*uniqueStreamSuffix* です。

例えば、ユーザーが mary-major である場合、以下のようなログストリームがある：

```
mary-major-east.1234567890abcdef0
mary.w-abcdef01234567890.021345abcdef6789
```

> **Note**
>
> この例に挙げた 16 桁の英数字の識別子は架空のものです。Amazon に表示される値は CloudWatch 異なります。

mary-major-usa-east.1234567890abcdef0の「ログイベント」ページには各ユーザーセッションの詳細が表示され、mary.w-abcdef01234567890.021345abcdef6789ログストリームにはワークフローの詳細が含まれます。

以下は、コピーステップを含むワークフロー (w-abcdef01234567890) に基づく、mary.w-abcdef01234567890.021345abcdef6789 のログストリームのサンプルです。

```
{
    "type": "ExecutionStarted",
    "details": {
        "input": {
            "initialFileLocation": {
                "bucket": "DOC-EXAMPLE-BUCKET",
                "key": "mary/workflowSteps2.json",
                "versionId": "version-id",
                "etag": "etag-id"
            }
        }
    },
    "workflowId":"w-abcdef01234567890",
    "executionId":"execution-id",
    "transferDetails": {
        "serverId":"s-server-id",
        "username":"mary",
        "sessionId":"session-id"
    }
```

```
},
{
    "type":"StepStarted",
    "details": {
        "input": {
            "fileLocation": {
                "backingStore":"S3",
                "bucket":"DOC-EXAMPLE-BUCKET",
                "key":"mary/workflowSteps2.json",
                "versionId":"version-id",
                "etag":"etag-id"
            }
        },
        "stepType":"COPY",
        "stepName":"copyToShared"
    },
    "workflowId":"w-abcdef01234567890",
    "executionId":"execution-id",
    "transferDetails": {
        "serverId":"s-server-id",
        "username":"mary",
        "sessionId":"session-id"
    }
},
{
    "type":"StepCompleted",
    "details":{
        "output":{},
        "stepType":"COPY",
        "stepName":"copyToShared"
    },
    "workflowId":"w-abcdef01234567890",
    "executionId":"execution-id",
    "transferDetails":{
        "serverId":"server-id",
        "username":"mary",
        "sessionId":"session-id"
    }
},
{
    "type":"ExecutionCompleted",
    "details": {},
    "workflowId":"w-abcdef01234567890",
    "executionId":"execution-id",
```

```
    "transferDetails":{
        "serverId":"s-server-id",
        "username":"mary",
        "sessionId":"session-id"
    }
}
```

CloudWatch ログ記録ロールを設定する

アクセス権を設定するには、リソースベースの IAM ポリシー、およびアクセス情報を提供する IAM ロールを作成します。

Amazon CloudWatch ログ記録を有効にするには、まずログ記録を有効にする CloudWatch IAM ポリシーを作成します。次いで、IAM ロールを作成して、ポリシーをアタッチします。そのためには、サーバーを作成するか既存のサーバーを編集します。の詳細については CloudWatch、「Amazon ユーザーガイド」の「Amazon CloudWatchとは」および CloudWatch「Amazon ログとは CloudWatch 」を参照してください。

CloudWatch ログ記録を許可するには、次の IAM ポリシーの例を使用します。

Use a logging role

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogStream",
                "logs:DescribeLogStreams",
                "logs:CreateLogGroup",
                "logs:PutLogEvents"
            ],
            "Resource": "arn:aws:logs:*:*:log-group:/aws/transfer/*"
        }
    ]
}
```

Use structured logging

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogDelivery",
                "logs:GetLogDelivery",
                "logs:UpdateLogDelivery",
                "logs:DeleteLogDelivery",
                "logs:ListLogDeliveries",
                "logs:PutResourcePolicy",
                "logs:DescribeResourcePolicies",
                "logs:DescribeLogGroups"
            ],
            "Resource": "arn:aws:logs:region-id:AWS #####:log-group:/aws/transfer/*"
        }
    ]
}
```

前述のサンプルポリシーでは、**Resource** について「*region-id*」と「*AWS #####*」を自分の値に置き換えます。例えば、次のようになります: **"Resource": "arn:aws::logs:us-east-1:111122223333:log-group:/aws/transfer/*"**

次に、ロールを作成し、作成した CloudWatch ログポリシーをアタッチします。

IAM ロールを作成し、ポリシーをアタッチするには

1. ナビゲーションペインで [Roles] (ロール) を選択してから [Create role] (ロールの作成) を選択します。

   [Create role] (ロールの作成) ページで [AWS service] (サービス) が選択されていることを確認します。

2. サービスリストから [Transfer] (転送) を選択してから [Next: Permissions] (次へ: アクセス許可) を選択します。これにより、 AWS Transfer Family と IAM ロールとの信頼関係が確立されます。さらに、aws:SourceAccount と aws:SourceArn のコンディション・キーを追加し、「混乱する代理」問題から守ります。詳細は以下のドキュメントを参照のこと：

   - との信頼関係を確立する手順 AWS Transfer Family: 信頼関係を確立するには
   - 混乱した代理問題の説明:混乱した代理問題

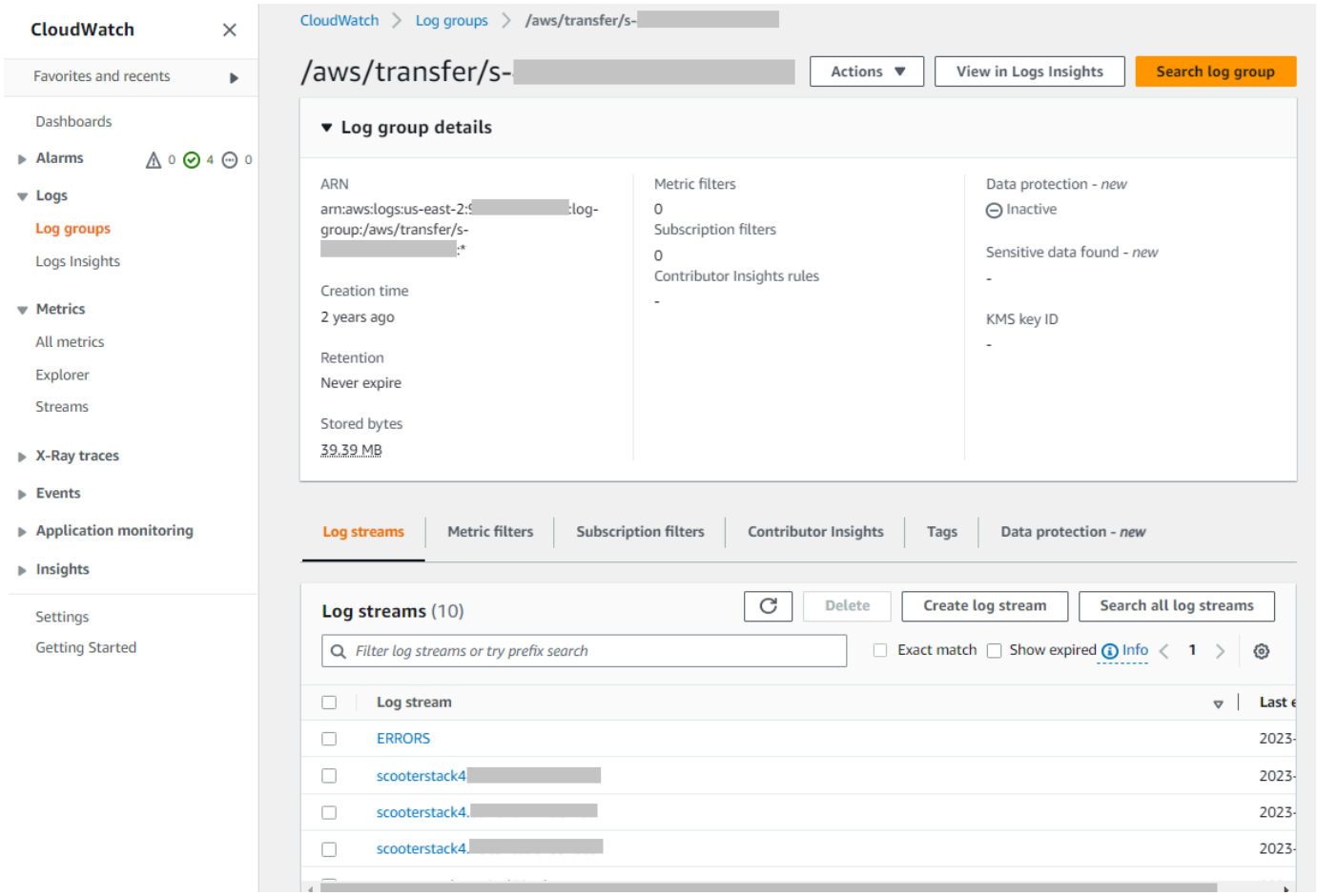
3. アクセス許可ポリシーのアタッチセクションで、先ほど作成した CloudWatch ログポリシーを見つけて選択し、次へ: タグ を選択します。
4. (オプション) タグのキーと値を入力して [Next: Review] (次へ: レビュー) を選択します。
5. [Review] (確認) ページで新しいロールの名前と説明を入力してから [Create role] (ロールの作成) を選択します。
6. ログを表示するには、[Server ID] (サーバー ID) を選択してサーバー設定ページを開き、[View logs] (ログの表示) を選択します。 CloudWatch コンソールにリダイレクトされ、ログストリームが表示されます。

CloudWatch サーバーのページには、ユーザー認証 (成功と失敗) 、データアップロード (PUT オペレーション) 、データダウンロード (GET オペレーション) のレコードが表示されます。

Transfer Family ログストリームの表示

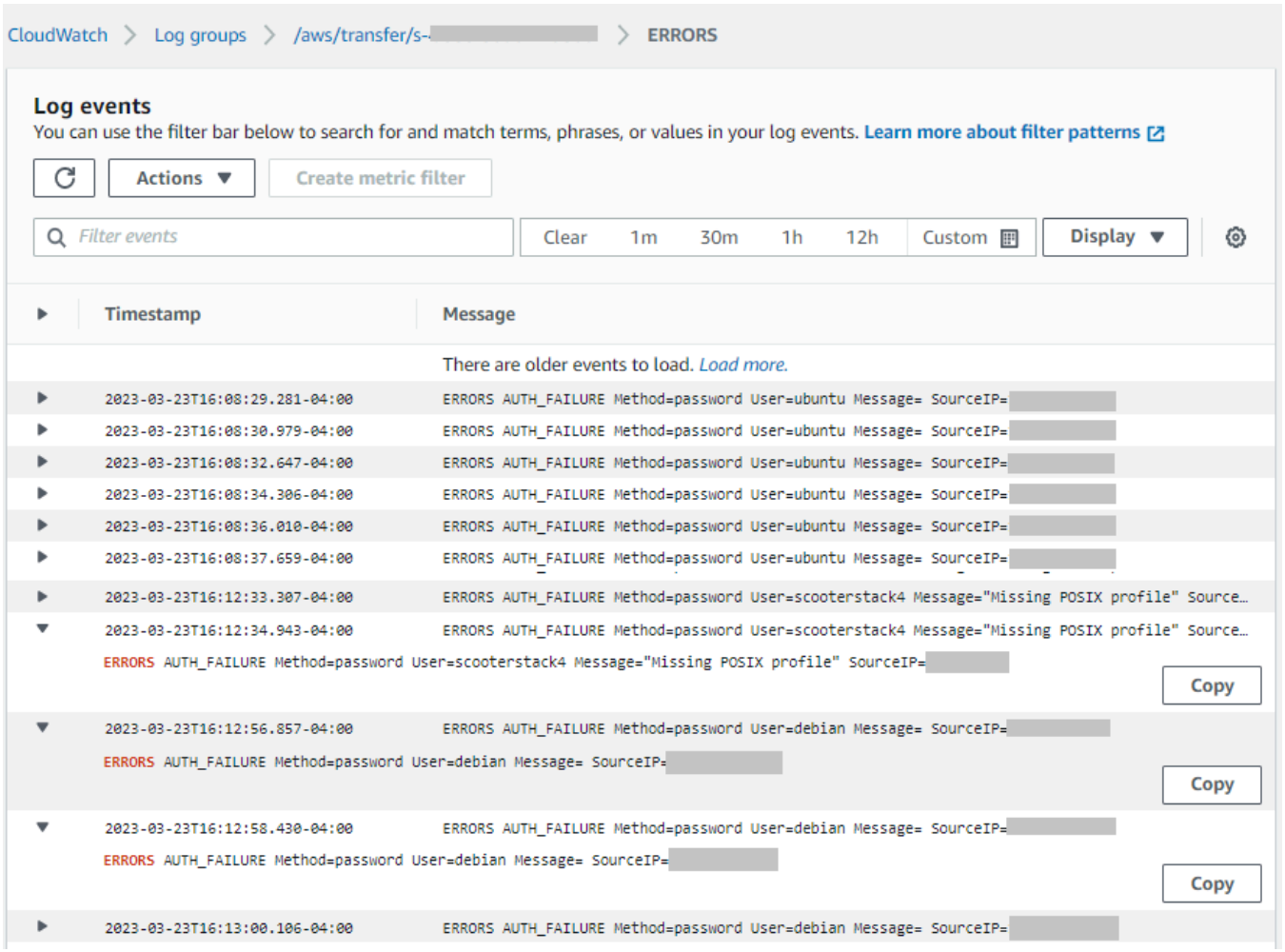
Transfer Familyのサーバーログを表示するには

1. サーバーの詳細ページに移動します。
2. [ログを表示] を選択します。これにより、Amazon が開きます CloudWatch。
3. 選択したサーバーのロググループが表示されます。

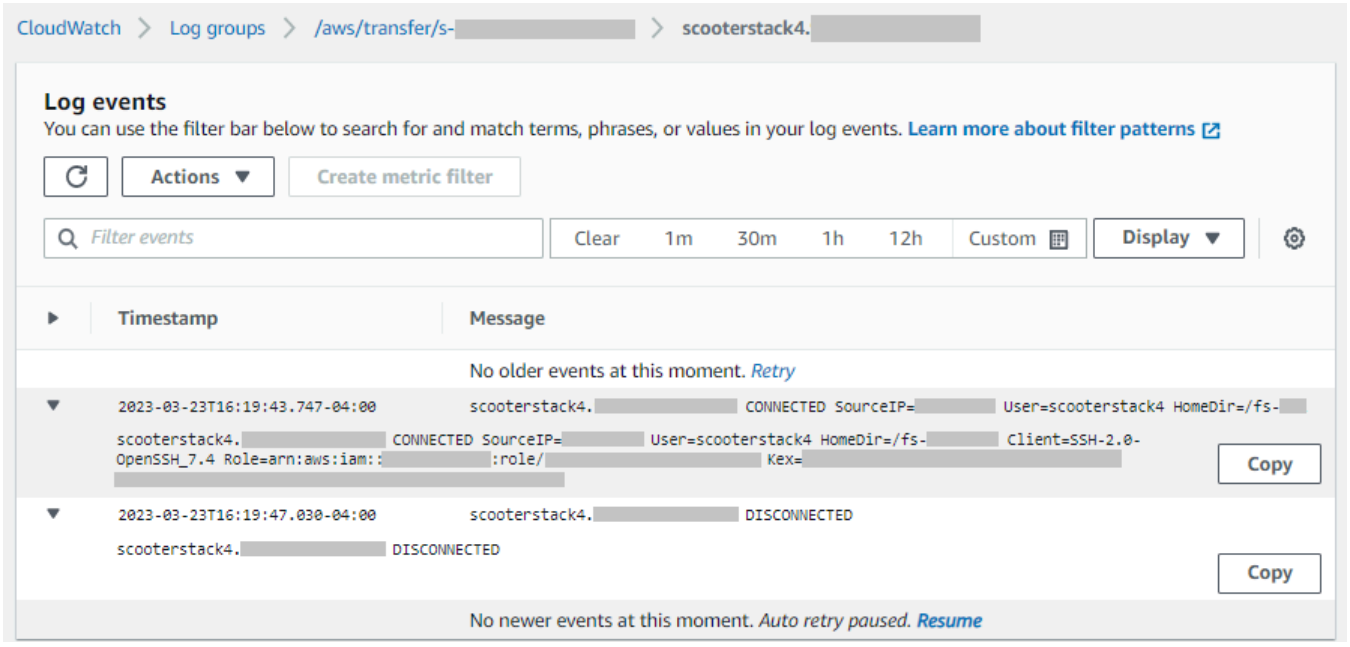


4. ログストリームを選択すると、そのストリームの詳細と個々のエントリを表示できます。

   - 「エラー」のリストがある場合は、それを選択してサーバーの最新のエラーの詳細を表示できます。

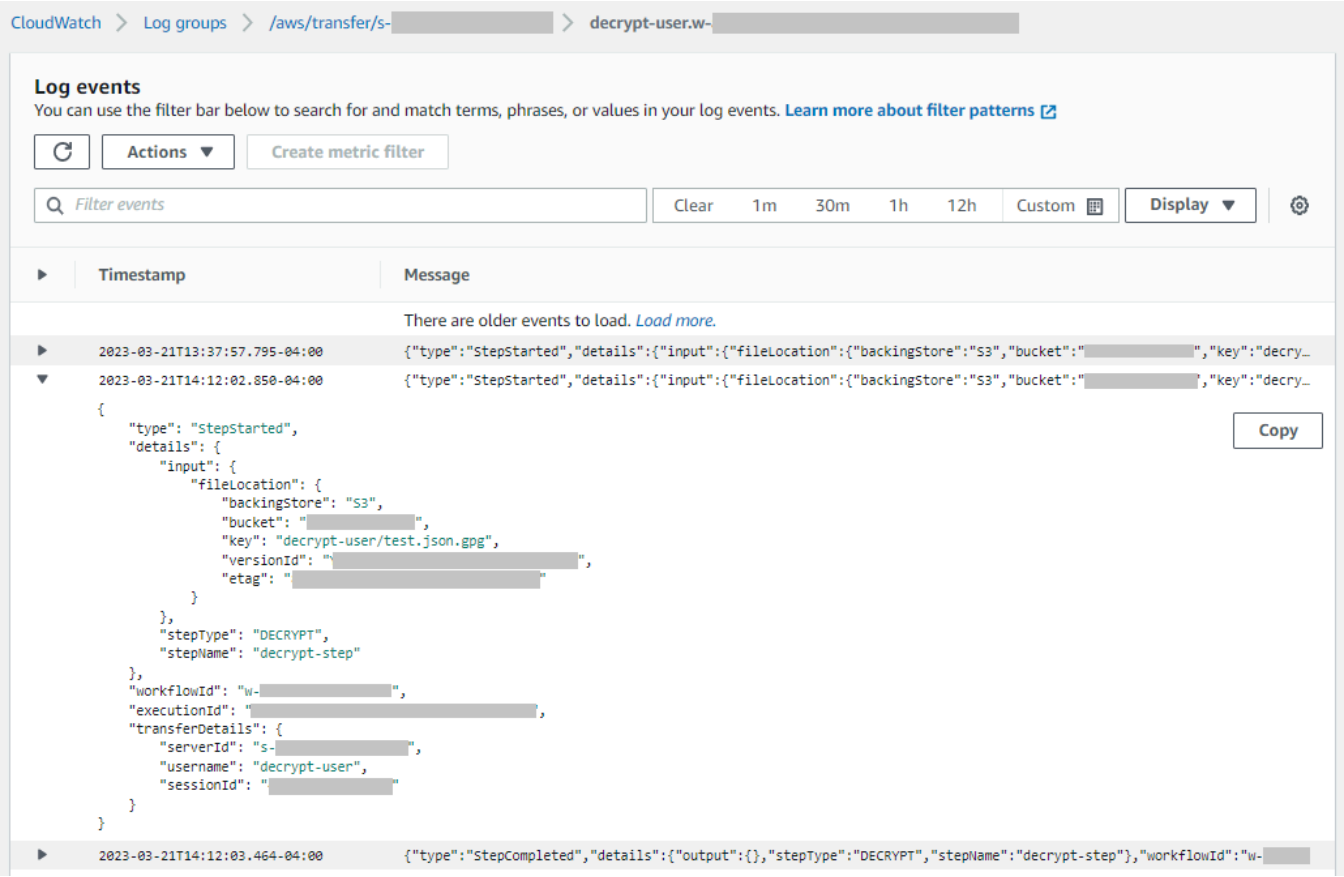


- 他のエントリを選択すると、ログストリームの例が表示されます。



- サーバーにマネージド型ワークフローが関連付けられている場合は、ワークフロー実行のログを表示できます。

> **Note**
>
> ワークフローのログストリームの形式は *username*.*workflowId*.*uniqueStreamSuffix* です。たとえば、「decrypt-user.w-a1111222233334444.aaaa1111bbbb2222」は、ユーザー **decrypt-user** とワークフロー **w-a1111222233334444** のログストリームの名前になる可能性があります。



> **Note**
>
> 拡張されたログエントリについては、「コピー」を選択してエントリをクリップボードにコピーできます。 CloudWatch ログの詳細については、「ログデータの表示」を参照してください。

Amazon CloudWatch アラームの作成

次の例は、 AWS Transfer Family メトリクス を使用して Amazon CloudWatch アラームを作成する方法を示していますFilesIn。

CDK

```
new cloudwatch.Metric({
  namespace: "AWS/Transfer",
  metricName: "FilesIn",
  dimensionsMap: { ServerId: "s-00000000000000000" },
  statistic: "Average",
  period: cdk.Duration.minutes(1),
}).createAlarm(this, "AWS/Transfer FilesIn", {
  threshold: 1000,
  evaluationPeriods: 10,
  datapointsToAlarm: 5,
  comparisonOperator:
 cloudwatch.ComparisonOperator.GREATER_THAN_OR_EQUAL_TO_THRESHOLD,
});
```

AWS CloudFormation

```
Type: AWS::CloudWatch::Alarm
Properties:
  Namespace: AWS/Transfer
  MetricName: FilesIn
  Dimensions:
    - Name: ServerId
      Value: s-00000000000000000
  Statistic: Average
  Period: 60
  Threshold: 1000
  EvaluationPeriods: 10
```

```
  DatapointsToAlarm: 5
  ComparisonOperator: GreaterThanOrEqualToThreshold
```

Amazon S3 API コールを S3 アクセスログに記録する

ユーザーに代わってファイル転送を要求した S3 リクエストを Amazon S3 アクセスログで特定しようとする場合、ファイル転送を処理すると想定された IAM ロールを表示する際には RoleSessionName が使用されます。また、転送に使用されるユーザー名、セッション ID、サーバー ID などの追加情報も表示されます。形式は `[AWS:Role Unique Identifier]/username.sessionid@server-id` で [Requester] (リクエスタ) フィールドに含まれています。たとえば、次に示すのは S3 バケットにコピーされたファイルの S3 アクセスログから得た[Requester] (リクエスタ) フィールドのコンテンツの例です。

```
arn:aws:sts::AWS-Account-ID:assumed-role/IamRoleName/
username.sessionid@server-id
```

上記の [Requester] (リクエスタ) フィールドには、IamRoleName と呼ばれる IAM ロールが表示されます。IAM ロールの一意の識別子の詳細については、AWS Identity and Access Management ユーザーガイドの「一意の識別子」を参照してください。

混乱する代理問題の防止

混乱した代理問題は、アクションを実行するためのアクセス許可を持たないエンティティが、より特権のあるエンティティにアクションの実行を強制できてしまう場合に生じる、セキュリティ上の問題です。では AWS、サービス間のなりすましにより、混乱した代理問題が発生する可能性があります。詳細については、「サービス間の混乱した代理の防止」を参照してください。

> **Note**
>
> 次の例では、*###############*をユーザー自身の情報で置き換えます。
> これらの例では、サーバーにワークフローがアタッチされていない場合、ワークフローの ARN の詳細を削除できます。

次のログ記録/呼び出しポリシーの例では、アカウント内のすべてのサーバー (およびワークフロー) がロールを引き受けることを許可します。

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAllServersWithWorkflowAttached",
            "Effect": "Allow",
            "Principal": {
                "Service": "transfer.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "account-id"
                },
                "ArnLike": {
                   "aws:SourceArn": [
                     "arn:aws:transfer:region:account-id:server/*",
                     "arn:aws:transfer:region:account-id:workflow/*"
                   ]
                }
            }
        }
    ]
}
```

次のログ記録/呼び出しポリシーの例では、特定のサーバー (およびワークフロー) がロールを引き受けることを許可します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowSpecificServerWithWorkflowAttached",
            "Effect": "Allow",
            "Principal": {
                "Service": "transfer.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "account-id"
                },
                "ArnEquals": {
                   "aws:SourceArn": [
```

```
                    "arn:aws:transfer:region:account-id:server/server-id",
                    "arn:aws:transfer:region:account-id:workflow/workflow-id"
                ]
            }
        }
    }
  ]
}
```

CloudWatch Transfer Family の ログ構造

このトピックでは、Transfer Family ログに入力されるフィールドについて説明します。JSON 構造化ログエントリとレガシーログエントリの両方についてです。

トピック

- Transfer Family の JSON 構造化ログ
- Transfer Family のレガシーログ

Transfer Family の JSON 構造化ログ

次の表に、Transfer Family SFTP/FTP/FTPS アクションのログエントリフィールドの詳細を、新しい JSON 構造化ログ形式で示します。

フィールド	説明	エントリ例
activity-type	The action by the user	OPEN | CLOSE | PARTIAL_CLOSE | DISCONNECTED | CONNECTED
bytes-in	Number of bytes uploaded by the user	29238420042
bytes-out	Number of bytes downloaded by the user	23094032490328
ciphers	Specifies the SSH cipher negotiated for the connection (available ciphers are listed in 暗号アルゴリズム)	aes256-gcm@openssh.com

フィールド	説明	エントリ例
client	The user's client software	SSH-2.0-OpenSSH_7.4
home-dir	The directory that the end user lands on when they connect to the endpoint if their home directory type is PATH: if they have a logical home directory, this value is always /	/user-home-bucket/test
kex	Specifies the negotiated SSH key exchange (KEX) for the connection (available KEX are listed in 暗号アルゴリズム)	diffie-hellman-group14-sha256
message	Provides more information related to the error	*<string>*
method	The authentication method	publickey
mode	Specifies how a client opens a file	CREATE | TRUNCATE | WRITE
operation	The client operation on a file	OPEN | CLOSE
path	Actual file path affected	/user-test-bucket/test-file-1.pdf
resource-arn	A system-assigned, unique identifier for a specific resource (for example, a server)	arn:aws:transfer:ap-northeast-1:12346789012:server/s-1234567890akeu2js2
role	The IAM role of the user	arn:aws:iam::0293883675:role/testuser-role
session-id	A system-assigned, unique identifier for a single session	9ca9a0e1cec6ad9d
source-ip	Client IP address	18.323.0.129

フィールド	説明	エントリ例
user	The end user's username	myname192
user-policy	The permissions specified for the end user: this field is populated if the user's policy is a session policy.	The JSON code for the session policy that is being used

Transfer Family のレガシーログ

次の表に、さまざまな Transfer Family アクションのログエントリの詳細を示します。

> **Note**
>
> これらのエントリは、新しい JSON 構造化ログ形式ではありません。

次の表に、さまざまな Transfer Family アクションのログエントリの詳細を新しい JSON 構造化ログ形式で示します。

[アクション]	Amazon CloudWatch Logs 内の対応するログ
認証の失敗	エラー AUTH_FAILURE METHOD=PU BLICKey user=LHR message=「RSA SHA256: LFZ3R2NMLY4RAK+B7RB1RSVUIBAE+A +HXG0C7L1JIZ0" SourceIP =3.8.172.211
コピー/タグ/削除/復号化ワークフロー	{「type":"StepStarted":"":"details」:{"input":{ "fileLocation ":{"backingStore ":"EFS "filesyst emId ":"fs-12345678 "path":"/lhr/regex.py"}}"st epType ":"TAG":"stepName ":"successful_tag_ step"},"workflowId ":"w-111aaaa222bb3 ,222bb3, の "executionId ":"81234abcd-1234- efggh-5678-ijklmnopqr90,transferDetails ":{"serverId ":"s-123abcabc5678efghi"use ruseruseruseruser"sessionId1234567890

[アクション]	Amazon CloudWatch Logs 内の対応するログ
カスタムステップワークフロー	{「type":" CustomStepInvoked"details":{"output" :{"token":"MzM4Mjg5YWUtYTEzMy 00YjlzLWI 3OGMtYz U4OGI2ZjQyMzE5"},"stepType ":"CUSTOM":"stepName ":"efs-s3_copy_2"} ,"workflowId ":"w-9283e49d3297c3333297,e xecutionId ":"1234abcd-1234-efglnopqr9 0」,transferDetails "{"serverId ":"111aa2 223":"username":"11111aaaaaa2 sessionId 123456789022223"
削除	lhr.33a8fb495ffb383b DELETE Path=/bucket/ user/123.jpg
ダウンロード	lhr.33a8fb495ffb383b オープンパス=/bucket/ user/123.jpg mode=READ llhr.33a8fb495fffb383b CLOSE Path=/bucket/ user/123.jpg BytesOut=3618546
ログイン/ログアウト	user.914984e553bcddb6 CONNECTED SourceIP =1.22.11.222 User=lhr HomeDir=L OGICAL Client=SSH-2.0-OpenSSH _7.4 Role=arn:aws::iam::123456789012:role/sftp-s3-access ユーザー.914984e553bcddb6 接続解除
Renames	lhr.33a8fb495fffb383b RENAME Path=/bucket/ user/lambo.png NewPath=/bucket/user/ferrar i.png

[アクション]	Amazon CloudWatch Logs 内の対応するログ
ワークフローエラーログの例	{「type":"StepErrored":"":"details」:{"errorTyp e ":"BAD_REQUEST":"errorMessage ":"Cannot tag Efs file":"stepType ":"TAG":"stepName ":"successful_tag_step"},"workflowId ":"w-1234 abcd5678efexecutionId ":"81234abcd-1234-efgh-5678-ijklmnopqr90「transferDetai ls "{"serverId":"s-1234abcd5678efghi」,u sername":"lhr,sessionId ":"1234567890abcde f0}}
Symlinks	lhr.eb49cf7b8651e6d5 CREATE_SYMTAK LinkPath=/fs-12345678/lhr/pqr.jpg TargetPat h=abc.jpg
アップロード	lhr.33a8fb495ffb383b OPEN PATH=/bucket/ user/123.jpg mode=create|truncate|Write lhr.33a8fb495ffb383b CLOSE Path=/bucket/ user/123.jpg BytesIn=3618546

[アクション]	Amazon CloudWatch Logs 内の対応するログ
ワークフロー	{「type":"ExecutionStarted":"""input":{"initia lFileLocation":{"backingStore ":"EFS ":"filesy stemId ":"fs-12345678":"path":"/lhr/regex.p y"}}},"workflowId ":"w-111aaa222bbbb bbbb3「executionId ":"1234abcd-1234-e fgh-5678-iklmnopqr90」,transferDetails ": {"serverId ":"s-":"111aaa223「username" :"lhr」,sessionId ":"1234567890abcdef0" {「type":"StepStarted":"":"details」:{"input":{ "fileLocation ":{"backingStore ":"EFS "filesyst emId ":"fs-12345678 "path":"/lhr/regex.py"}}"st epType ":"CUSTOM":"stepName ":"efs-s3 _copy_2"}"workflowId ":"w-9283e49d32932 97c333333297,executionId ":"1234abcd-1234-e fglnopqr90,transferDetails "{"serverId ":"s-18ca 49dce5d5d84e42e)"sessionId1234567890

CloudWatch ログエントリの例

このトピックでは、ログエントリの例を示します。

トピック

- 転送セッションログエントリの例
- SFTP コネクタのログエントリの例
- キー交換アルゴリズムの失敗のログエントリの例

転送セッションログエントリの例

この例では、SFTP ユーザーが Transfer Family サーバーに接続し、ファイルをアップロードしてから、セッションから切断します。

次のログエントリは、Transfer Family サーバーに接続する SFTP ユーザーを反映しています。

```
{
```

```
   "role": "arn:aws:iam::500655546075:role/scooter-transfer-s3",
   "activity-type": "CONNECTED",
   "ciphers": "chacha20-poly1305@openssh.com,chacha20-poly1305@openssh.com",
   "client": "SSH-2.0-OpenSSH_7.4",
   "source-ip": "52.94.133.133",
   "resource-arn": "arn:aws:transfer:us-east-1:500655546075:server/
s-3fe215d89f074ed2a",
   "home-dir": "/scooter-test/log-me",
   "user": "log-me",
   "kex": "ecdh-sha2-nistp256",
   "session-id": "9ca9a0e1cec6ad9d"
}
```

次のログエントリは、Amazon S3 バケットにファイルをアップロードする SFTP ユーザーを反映しています。

```
{
   "mode": "CREATE|TRUNCATE|WRITE",
   "path": "/scooter-test/log-me/config-file",
   "activity-type": "OPEN",
   "resource-arn": "arn:aws:transfer:us-east-1:500655546075:server/
s-3fe215d89f074ed2a",
   "session-id": "9ca9a0e1cec6ad9d"
}
```

次のログエントリは、SFTP ユーザーが SFTP セッションから切断したことを示しています。まず、クライアントはバケットへの接続を閉じ、次にクライアントは SFTP セッションを切断します。

```
{
   "path": "/scooter-test/log-me/config-file",
   "activity-type": "CLOSE",
   "resource-arn": "arn:aws:transfer:us-east-1:500655546075:server/
s-3fe215d89f074ed2a",
   "bytes-in": "121",
   "session-id": "9ca9a0e1cec6ad9d"
}

{
   "activity-type": "DISCONNECTED",
   "resource-arn": "arn:aws:transfer:us-east-1:500655546075:server/
s-3fe215d89f074ed2a",
   "session-id": "9ca9a0e1cec6ad9d"
```

```
}
```

SFTP コネクタのログエントリの例

このセクションでは、転送の成功と失敗の両方のログの例を示します。ログは という名前のロググループに生成されます。ここで/aws/transfer/*connector-id*、*connector-id* は SFTP コネクタの識別子です。

> **Note**
>
> SFTP コネクタのログエントリは、StartFileTransferコマンドを実行する場合にのみ生成されます。

このログエントリは、正常に完了した転送用です。

```
{
    "operation": "RETRIEVE",
    "timestamp": "2023-10-25T16:33:27.373720Z",
    "connector-id": "connector-id",
    "transfer-id": "transfer-id",
    "file-transfer-id": "transfer-id/file-transfer-id",
    "url": "sftp://192.0.2.0",
    "file-path": "/remotebucket/remotefilepath",
    "status-code": "COMPLETED",
    "start-time": "2023-10-25T16:33:26.945481Z",
    "end-time": "2023-10-25T16:33:27.159823Z",
    "account-id": "480351544584",
    "connector-arn": "arn:aws:transfer:us-east-1:480351544584:connector/connector-id",
    "local-directory-path": "/connectors-localbucket"
    "bytes": 514
}
```

このログエントリは、タイムアウトした転送用であるため、正常に完了しませんでした。

```
{
    "operation": "RETRIEVE",
    "timestamp": "2023-10-25T22:33:47.625703Z",
    "connector-id": "connector-id",
    "transfer-id": "transfer-id",
```

```
    "file-transfer-id": "transfer-id/file-transfer-id",
    "url": "sftp://192.0.2.0",
    "file-path": "/remotebucket/remotefilepath",
    "status-code": "FAILED",
    "failure-code": "TIMEOUT_ERROR",
    "failure-message": "Transfer request timeout.",
    "account-id": "480351544584",
    "connector-arn": "arn:aws:transfer:us-east-1:480351544584:connector/connector-id",
    "local-directory-path": "/connectors-localbucket"
}
```

このログエントリは、成功した SEND オペレーション用です。

```
{
    "operation": "SEND",
    "timestamp": "2024-04-24T18:16:12.513207284Z",
    "connector-id": "connector-id",
    "transfer-id": "transfer-id",
    "file-transfer-id": "transfer-id/file-transfer-id",
    "url": "sftp://server-id.server.transfer.us-east-1.amazonaws.com",
    "file-path": "/DOC-EXAMPLE-BUCKET/my-test-folder/connector-metrics-us-
east-1-2024-01-02.csv",
    "status-code": "COMPLETED",
    "start-time": "2024-04-24T18:16:12.295235884Z",
    "end-time": "2024-04-24T18:16:12.461840732Z",
    "account-id": "255443218509",
    "connector-arn": "arn:aws:transfer:us-east-1:255443218509:connector/connector-id",
    "bytes": 275
}
```

前の口グ例の一部の主要フィールドの説明。

- timestamp は、口グが に追加されるタイミングを表します CloudWatch。 start-timeと は、コネクタが実際に転送を開始および終了するタイミングend-timeに対応します。
- transfer-id は、各start-file-transferリクエストに割り当てられる一意の識別子です。ユーザーが 1 回の start-file-transfer API コールで複数のファイルパスを渡すと、すべてのファイルが同じ を共有しますtransfer-id。
- file-transfer-id は、転送されるファイルごとに生成される一意の値です。の最初の部分は file-transfer-idと同じであることに注意してくださいtransfer-id。

キー交換アルゴリズムの失敗のログエントリの例

このセクションでは、キー交換アルゴリズム (KEX) が失敗したログの例を示します。以下は、構造化ログの ERRORS ログストリームの例です。

このログエントリは、ホストキータイプエラーがある例です。

```
{
    "activity-type": "KEX_FAILURE",
    "source-ip": "999.999.999.999",
    "resource-arn": "arn:aws:transfer:us-east-1:999999999999:server/
s-999999999999999999",
    "message": "no matching host key type found",
    "kex": "ecdsa-sha2-nistp256,ecdsa-sha2-nistp384,ecdsa-sha2-nistp521,ecdsa-sha2-
nistp256-cert-v01@openssh.com,ecdsa-sha2-nistp384-cert-v01@openssh.com,ecdsa-sha2-
nistp521-cert-v01@openssh.com,ssh-ed25519,ssh-rsa,ssh-dss"
}
```

このログエントリは、KEX の不一致がある例です。

```
{
    "activity-type": "KEX_FAILURE",
    "source-ip": "999.999.999.999",
    "resource-arn": "arn:aws:transfer:us-east-1:999999999999:server/
s-999999999999999999",
    "message": "no matching key exchange method found",
    "kex": "diffie-hellman-group1-sha1,diffie-hellman-group14-sha1,diffie-hellman-
group14-sha256"
}
```

Transfer Family の CloudWatch メトリクスの使用

> **Note**
>
> Transfer Family コンソール自体で Transfer Family のメトリクスを取得することもできます。詳細については、「コンソールで使用状況をモニターする」を参照してください

CloudWatch メトリクスを使用してサーバーに関する情報を取得できます。メトリクスは、 に発行される時系列のデータポイントのセットを表します CloudWatch。メトリクスを使用するには、Transfer Family の名前空間、メトリクス名、およびディメンションを指定する必要がありま

す。メトリクスの詳細については、「Amazon ユーザーガイド」の「メトリクス」を参照してください。 CloudWatch

次の表に、Transfer Family の CloudWatch メトリクスを示します。

名前空間	メトリクス	説明
AWS/Transfer	BytesIn	サーバーに転送された総バイト数。 単位: カウント 期間: 5 分
	BytesOut	サーバーから転送された総バイト数。 単位: 個 期間: 5 分
	FilesIn	サーバーに転送されたファイルの合計数。 AS2 プロトコルを使用するサーバーの場合、このメトリックは受信したメッセージの数を表します。 単位: カウント 期間: 5 分
	FilesOut	サーバーから転送されたファイルの合計数。 単位: カウント 期間: 5 分
	InboundMessage	取引相手から正常に受信した AS2 メッセージの総数。 単位: カウント 期間: 5 分
	InboundFailedMessage	取引相手から正常に受信されなかった AS2 メッセージの総数。つまり、取引相手がメッセージを送信しました

名前空間	メトリクス	説明
		が、Transfer Family サーバーはそれを正常に処理できませんでした。 単位: カウント 期間: 5 分
	OnUploadExecutionsStarted	サーバーで開始されたワークフロー実行の総数。 単位: カウント 期間 = 1 分
	OnUploadExecutionsSuccess	サーバーで成功したワークフロー実行の合計数。 単位: カウント 期間 = 1 分
	OnUploadExecutionsFailed	サーバーで失敗したワークフロー実行の合計数。 単位: カウント 期間 = 1 分

Transfer Family ディメンション

ディメンションは、メトリクスのアイデンティティの一部である名前と値のペアです。ディメンションの詳細については、「Amazon CloudWatch ユーザーガイド」の「ディメンション」を参照してください。

次の表は、Transfer Family の CloudWatch ディメンションを示しています。

ディメンション	説明
ServerId	サーバーに付けられた一意の ID。

AWS User Notifications で を使用する AWS Transfer Family

AWS Transfer Family イベントに関する通知を受け取るには、 AWS User Notificationsを使用してさまざまな配信チャネルを設定できます。指定したルールにイベントが一致すると、通知を受け取ります。

イベントの通知は、E メール、AWS Chatbot チャット通知、AWS Console Mobile Application プッシュ通知など、複数のチャネルを通じて受け取ることができます。コンソール通知センター で通知を確認することもできます。 は集約をサポートしているため、特定のイベント中に受信する通知の数を減らすことができます。 User Notifications

詳細については、「 ユーザーガイド」のAWS Transfer Family 「マネージドワークフローを使用したファイル配信通知のカスタマイズ」ブログ記事「 AWS User NotificationsとはAWS User Notifications 」を参照してください。

クエリを使用したログエントリのフィルタリング

CloudWatch クエリを使用して、Transfer Family のログエントリをフィルタリングおよび識別できます。このセクションでは、いくつかの例を示します。

1. にサインイン AWS Management Console し、https://console.aws.amazon.com/cloudwatch/ で CloudWatch コンソールを開きます。
2. クエリまたはルールを作成できます。
   - Logs Insights クエリを作成するには、左側のナビゲーションパネルから Logs Insights を選択し、クエリの詳細を入力します。
   - Contributor Insights ルールを作成するには、左側のナビゲーションパネルから Insights > Contributor Insights を選択し、ルールの詳細を入力します。
3. 作成したクエリまたはルールを実行します。

認証失敗の上位の寄稿者を表示する

構造化ログでは、認証失敗ログエントリは次のようになります。

```
{
  "method":"password",
  "activity-type":"AUTH_FAILURE",
  "source-ip":"999.999.999.999",
```

```
  "resource-arn":"arn:aws:transfer:us-east-1:999999999999:server/s-0123456789abcdef",
  "message":"Invalid user name or password",
  "user":"exampleUser"
}
```

次のクエリを実行して、認証失敗の上位の寄与要因を表示します。

```
filter @logStream = 'ERRORS'
| filter `activity-type` = 'AUTH_FAILURE'
| stats count() as AuthFailures by user, method
| sort by AuthFailures desc
| limit 10
```

CloudWatch Logs Insights を使用する代わりに、CloudWatch Contributors Insights ルールを作成して認証の失敗を表示できます。次のようなルールを作成します。

```
{
    "AggregateOn": "Count",
    "Contribution": {
        "Filters": [
            {
                "Match": "$.activity-type",
                "In": [
                    "AUTH_FAILURE"
                ]
            }
        ],
        "Keys": [
            "$.user"
        ]
    },
    "LogFormat": "JSON",
    "Schema": {
        "Name": "CloudWatchLogRule",
        "Version": 1
    },
    "LogGroupARNs": [
        "arn:aws:logs:us-east-1:999999999999:log-group:/customer/structured_logs"
    ]
}
```

ファイルが開かれたログエントリを表示する

構造化ログでは、ファイル読み取りログエントリは次のようになります。

```
{
  "mode":"READ",
  "path":"/fs-0df669c89d9bf7f45/avtester/example",
  "activity-type":"OPEN",
  "resource-arn":"arn:aws:transfer:us-east-1:999999999999:server/s-0123456789abcdef",
  "session-id":"0049cd844c7536c06a89"
}
```

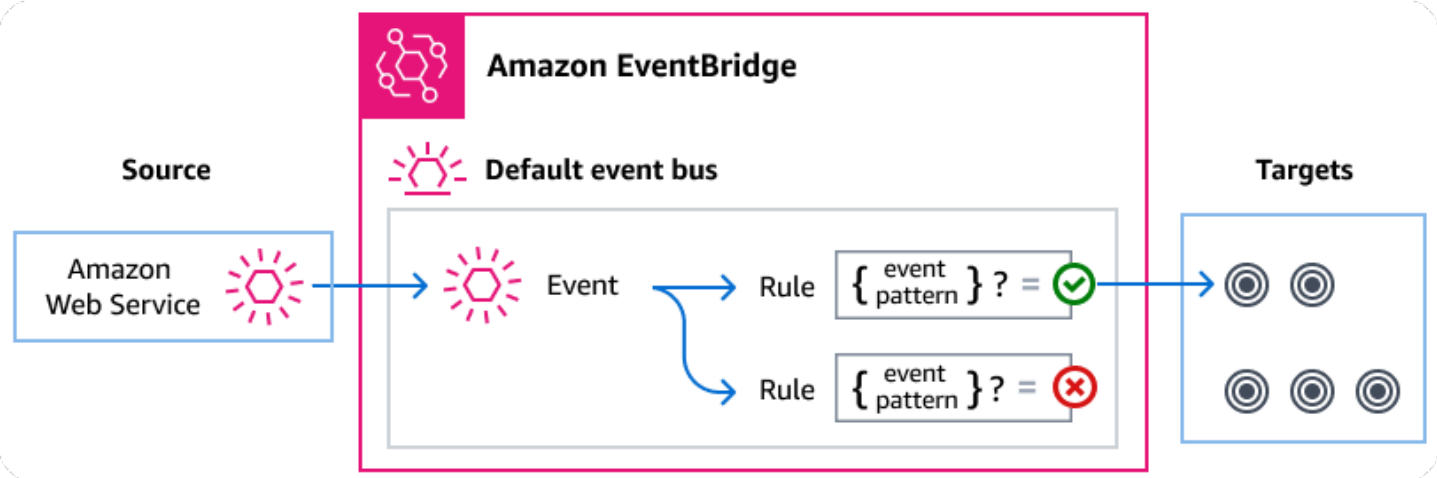
次のクエリを実行して、ファイルが開かれたことを示すログエントリを表示します。

```
filter `activity-type` = 'OPEN'
| display @timestamp, @logStream, `session-id`, mode, path
```

を使用した Transfer Family イベントの管理 Amazon EventBridge

Amazon EventBridge は、イベントを使用してアプリケーションコンポーネントを接続できるサーバーレスサービスです。これにより、スケーラブルなイベント駆動型アプリケーションを簡単に構築できます。イベント駆動型アーキテクチャは、イベントを発行して応答することで連携する疎結合ソフトウェアシステムを構築するスタイルです。イベントとは、リソースまたは環境で発生した変更を指します。

多くの AWS サービスと同様に、 はイベント Transfer Family を生成し、 EventBridge デフォルトのイベントバスに送信します。デフォルトのイベントバスは、すべての AWS アカウントで自動的にプロビジョニングされることに注意してください。イベントバスは、イベントを受信するルーターであり、ゼロ個以上の送信先やターゲットに配信します。イベントバスのルールを指定して、イベントが到着したときに評価します。各ルールは、イベントがルールのイベントパターンに一致するかどうかをチェックします。イベントが一致すると、イベントバスは 1 つ以上の指定されたターゲットにイベントを送信します。



トピック

- Transfer Family イベント
- EventBridge ルールを使用した Transfer Family イベントの送信
- Amazon EventBridge アクセス許可
- その他の EventBridge リソース
- Transfer Family イベント詳細リファレンス

Transfer Family イベント

Transfer Family は、デフォルトのイベントバスに EventBridge イベントを自動的に送信します。各ルールにイベントパターンと 1 つ以上のターゲットが含まれるイベントバスにルールを作成できます。ルールのイベントパターンに一致するイベントは、ベストエフォートベースで指定されたターゲットに配信されますが、一部のイベントは順不同で配信される場合があります。

次のイベントは によって生成されます Transfer Family。詳細については、「 ユーザーガイド」の「 EventBridge イベントAmazon EventBridge 」を参照してください。

SFTP、FTPS、および FTP サーバーイベント

イベントの詳細のタイプ	説明
FTP ファイルサーバーのダウンロードが完了しました	FTP プロトコルのファイルが正常にダウンロードされました。
FTP ファイルサーバーのダウンロードに失敗しました	FTP プロトコルでファイルのダウンロードに失敗しました。
FTP ファイルサーバーのアップロードが完了しました	FTP プロトコルのファイルが正常にアップロードされました。
FTP ファイルサーバーのアップロードに失敗しました	FTP プロトコルでファイルをアップロードしようとしましたが失敗しました。
FTPS ファイルサーバーのダウンロードが完了しました	FTPS プロトコルのファイルが正常にダウンロードされました。
FTPS ファイルサーバーのダウンロードに失敗しました	FTPS プロトコルでファイルのダウンロードに失敗しました。
FTPS ファイルサーバーのアップロードが完了しました	FTPS プロトコルのファイルが正常にアップロードされました。
FTPS ファイルサーバーのアップロードに失敗しました	FTPS プロトコルのファイルをアップロードできませんでした。

イベントの詳細のタイプ	説明
SFTP サーバーファイルのダウンロードが完了しました	SFTP プロトコルのファイルが正常にダウンロードされました。
SFTP サーバーファイルのダウンロードに失敗しました	SFTP プロトコルでファイルのダウンロードに失敗しました。
SFTP サーバーファイルのアップロードが完了しました	SFTP プロトコルのファイルが正常にアップロードされました。
SFTP サーバーファイルのアップロードに失敗しました	SFTP プロトコルでファイルをアップロードできませんでした。

SFTP コネクタイベント

イベントの詳細のタイプ	説明
SFTP コネクタファイルの送信が完了しました	コネクタからリモート SFTP サーバーへのファイル転送が正常に完了しました。
SFTP コネクタファイルの送信に失敗しました	コネクタからリモート SFTP サーバーへのファイル転送が失敗しました。
SFTP コネクタファイルの取得が完了しました	リモート SFTP サーバーからコネクタへのファイル転送が正常に完了しました。
SFTP コネクタファイルの取得に失敗しました	リモート SFTP サーバーからコネクタへのファイル転送が失敗しました。
SFTP コネクタディレクトリの一覧表示が完了しました	正常に完了した呼び出しを一覧表示する開始ファイルディレクトリ。
SFTP コネクタディレクトリの一覧表示に失敗しました	失敗した開始ファイルディレクトリのリスト。

A2S イベント

イベントの詳細のタイプ	説明
AS2 ペイロード受信完了	AS2 メッセージのペイロードが受信されました。
AS2 ペイロード受信失敗	AS2 メッセージのペイロードが受信されていません。
AS2 ペイロードの送信が完了しました	AS2 メッセージのペイロードが正常に送信されました。
AS2 ペイロードの送信に失敗しました	AS2 メッセージのペイロードの送信に失敗しました。
AS2 MDN 受信完了	AS2 メッセージのメッセージ処理通知を受信しました。
AS2 MDN 受信失敗	AS2 メッセージのメッセージ処理通知が受信されていません。
AS2 MDN 送信が完了しました	AS2 メッセージのメッセージ処理通知が正常に送信されました。
AS2 MDN 送信失敗	AS2 メッセージのメッセージ処理通知の送信に失敗しました。

EventBridge ルールを使用した Transfer Family イベントの送信

EventBridge デフォルトのイベントバスでターゲットに Transfer Family イベントを送信する場合は、目的のイベント内のデータと一致する Transfer Family イベントパターンを含むルールを作成する必要があります。

ルールを作成するには、以下の一般的なステップに従います。

1. 以下を指定するルールのイベントパターンを作成します。
   - Transfer Family は、ルールによって評価されるイベントのソースです。
   - (オプション) 照合するその他のイベントデータ。

   詳細については、「???」を参照してください。
2. (オプション) がルールのターゲットに情報 EventBridge を送信する前に、イベントからのデータをカスタマイズする入力トランスフォーマーを作成します。

詳細については、「EventBridge ユーザーガイド」の「Amazon EventBridge 入力変換」を参照してください。

3. イベントパターンに一致するイベントを配信 EventBridge するターゲットを指定します。

   ターゲットは、他の AWS のサービス、Software as a Service (SaaS) アプリケーション、API 送信先、またはその他のカスタムエンドポイントです。詳細については、EventBridge ユーザーガイドのターゲットを参照してください。

イベントバスルールの詳細な作成方法については、「EventBridge ユーザーガイド」の「イベントに反応する Amazon EventBridge ルールの作成」を参照してください。

イベントの Transfer Family イベントパターンの作成

Transfer Family がデフォルトのイベントバスにイベントを配信する場合、 は各ルールに定義されたイベントパターン EventBridge を使用して、イベントをルールのターゲットに配信する必要があるかどうかを判断します。イベントパターンは、目的の Transfer Family イベントのデータに一致します。各イベントパターンは、以下を含む JSON オブジェクトです。

- イベントを送信するサービスを識別する `source` 属性。 Transfer Family イベントの場合、ソースは ですaws.transfer。
- (オプション) 一致するイベントタイプの配列を含むdetail-type属性。
- (オプション) 一致する他のイベントデータを含むdetail属性。

例えば、次のイベントパターンは、 からのすべてのイベントに一致します Transfer Family。

```
{
  "source": ["aws.transfer"]
}
```

次のイベントパターンの例は、すべての SFTP コネクタイベントに一致します。

```
{
  "source": ["aws.transfer"],
  "detail-type": ["SFTP Connector File Send Completed", "SFTP Connector File Retrieve
 Completed",
                  "SFTP Connector File Retrieve Failed", "SFTP Connector File Send
 Failed"]
```

```
}
```

次のイベントパターンの例では、Transfer Family で失敗したすべてのイベントに一致します。

```
{
  "source": ["aws.transfer"],
  "detail-type": [{"wildcard", "*Failed"}]
}
```

次のイベントパターンの例では、ユーザー##### の正常な SFTP ダウンロードに一致します。

```
{
  "source": ["aws.transfer"],
  "detail-type": ["SFTP Server File Download Completed"],
  "detail": {
    "username": [username]
  }
}
```

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

でのイベントの Transfer Family イベントパターンのテスト EventBridge

EventBridge サンドボックスを使用すると、ルールを作成または編集するより広範なプロセスを完了することなく、イベントパターンをすばやく定義してテストできます。サンドボックスを使用すると、イベントパターンを定義し、サンプルイベントを使用して、そのパターンが目的のイベントと一致することを確認できます。 は、サンドボックスから直接そのイベントパターンを使用して新しいルールを作成するオプション EventBridge を提供します。

詳細については、「 ユーザーガイド」の EventBridge 「サンドボックスを使用したイベントパターンのテストEventBridge 」を参照してください。

Amazon EventBridge アクセス許可

Transfer Family では、 にイベントを配信するための追加のアクセス許可は必要ありません Amazon EventBridge。

指定するターゲットには、特定のアクセス許可または設定が必要になる場合があります。ターゲットに特定のサービスを使用する方法の詳細については、「Amazon EventBridge ユーザーガイド」の「Amazon EventBridge ターゲット」を参照してください。

その他の EventBridge リソース

EventBridge を使用してイベントを処理および管理する方法の詳細については、Amazon EventBridge 「 ユーザーガイド」の以下のトピックを参照してください。

- イベントバスの仕組みに関する詳細は、「Amazon EventBridge イベントバス」を参照してください。
- イベント構造については、「Amazon EventBridge イベント」を参照してください。
- ルールとイベントを照合するときに EventBridge が使用するイベントパターンの構築については、「イベントパターン」を参照してください。
- EventBridge が処理するイベントを指定するルールの作成方法については、「Amazon EventBridge ルール」を参照してください。
- が一致するイベント EventBridge を送信するサービスやその他の送信先を指定する方法については、「ターゲット」を参照してください。

Transfer Family イベント詳細リファレンス

AWS サービスからのすべてのイベントには、イベントに関するメタデータを含む共通のフィールドセットがあります。これらのメタデータには、イベントのソースである AWS サービス、イベントが生成された時刻、イベントが発生したアカウントとリージョンなどが含まれます。これらの一般的なフィールドの定義については、「Amazon EventBridge ユーザーガイド」の「イベント構造リファレンス」を参照してください。

さらに、各イベントには、その特定のイベントに固有のデータを含む `detail` フィールドがあります。次のリファレンスでは、さまざまな Transfer Family イベントの詳細フィールドを定義します。

EventBridge を使用して Transfer Family イベントを選択および管理する場合は、次の点を考慮してください。

- からのすべてのイベントの `source`フィールド Transfer Family は に設定されます`aws.transfer`。
- `detail-type` フィールドはイベントタイプを指定します。

例えば FTP File Server Download Completed です。

- detail フィールドには、その特定のイベントに固有のデータが含まれます。

Transfer Family イベントに一致するようにルールを有効化するイベントパターンの作成方法については、「Amazon EventBridge ユーザーガイド」の「Amazon EventBridge のイベントパターン」を参照してください。

イベントとその EventBridge 処理方法の詳細については、「 Amazon EventBridge ユーザーガイド」の「 Amazon EventBridge イベント」を参照してください。

トピック

- SFTP、FTPS、および FTP サーバーイベント
- SFTP コネクタイベント
- AS2 イベント

SFTP、FTPS、および FTP サーバーイベント

SFTP、FTPS、および FTP サーバーイベントの詳細フィールドは次のとおりです。

- FTP ファイルサーバーのダウンロードが完了しました
- FTP ファイルサーバーのダウンロードに失敗しました
- FTP ファイルサーバーのアップロードが完了しました
- FTP ファイルサーバーのアップロードに失敗しました
- FTPS ファイルサーバーのダウンロードが完了しました
- FTPS ファイルサーバーのダウンロードに失敗しました
- FTPS ファイルサーバーのアップロードが完了しました
- FTPS ファイルサーバーのアップロードに失敗しました
- SFTP サーバーファイルのダウンロードが完了しました
- SFTP サーバーファイルのダウンロードに失敗しました
- SFTP サーバーファイルのアップロードが完了しました
- SFTP サーバーファイルのアップロードに失敗しました

source および detail-typeフィールドには、 Transfer Family イベントの特定の値が含まれているため、以下が含まれます。すべてのイベントに含まれる他のメタデータフィールドの定義については、「 ユーザーガイド」の「イベント構造のリファレンスAmazon EventBridge 」を参照してください。

```
{
  . . .,
  "detail-type": "string",
  "source": "aws.transfer",
  . . .,
  "detail": {
    "failure-code" : "string",
    "status-code" : "string",
    "protocol" : "string",
    "bytes" : "number",
    "client-ip" : "string",
    "failure-message" : "string",
    "end-timestamp" : "string",
    "etag" : "string",
    "file-path" : "string",
    "server-id" : "string",
    "username" : "string",
    "session-id" : "string",
    "start-timestamp" : "string"
  }
}
```

`detail-type`

イベントのタイプを示します。

このイベントの場合、値は前述の SFTP、FTPS、または FTP サーバーイベント名のいずれかです。

`source`

イベントを発生させたサービスを識別します。Transfer Family イベントの場合、この値は ですaws.transfer。

`detail`

イベントに関する情報を含む JSON オブジェクト。このフィールドの内容は、イベントを生成するサービスによって決まります。

このイベントでは、データには以下が含まれます。

`failure-code`

転送が失敗した理由のカテゴリ。値: `PARTIAL_UPLOAD | PARTIAL_DOWNLOAD | UNKNOWN_ERROR`

`status-code`

転送が成功したかどうか。値: `COMPLETED | FAILED`。

`protocol`

転送に使用されるプロトコル。値: `SFTP | FTPS | FTP`

`bytes`

転送バイト数。

`client-ip`

転送に関係するクライアントの IP アドレス

`failure-message`

失敗した転送の場合、転送が失敗した理由の詳細。

`end-timestamp`

転送が成功した場合、ファイルの処理が終了する時刻のタイムスタンプ。

`etag`

エンティティタグ (Amazon S3 ファイルにのみ使用されます) 。

`file-path`

転送されるファイルへのパス。

`server-id`

Transfer Family サーバーの一意の ID。

`username`

転送を実行しているユーザー。

`session-id`

転送セッションの一意の識別子。

`start-timestamp`

転送が成功した場合、ファイル処理が開始される時刻のタイムスタンプ。

Example SFTP サーバーファイルのダウンロード失敗サンプルイベント

次の例は、SFTP サーバー (使用されているストレージAmazon EFS) でダウンロードが失敗するイベントを示しています。

```
{
    "version": "0",
    "id": "event-ID",
    "detail-type": "SFTP Server File Download Failed",
    "source": "aws.transfer",
    "account": "958412138249",
    "time": "2024-01-29T17:20:27Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:transfer:us-east-1:958412138249:server/s-1234abcd5678efghi"
    ],
    "detail": {
        "failure-code": "PARTIAL_DOWNLOAD",
        "status-code": "FAILED",
        "protocol": "SFTP",
        "bytes": 4100,
        "client-ip": "IP-address",
        "failure-message": "File was partially downloaded.",
        "end-timestamp": "2024-01-29T17:20:27.749749117Z",
        "file-path": "/fs-1234abcd5678efghi/user0/test-file",
        "server-id": "s-1234abcd5678efghi",
        "username": "test",
        "session-id": "session-ID",
        "start-timestamp": "2024-01-29T17:20:16.706282454Z"
    }
}
```

Example FTP ファイルサーバーのアップロード完了サンプルイベント

次の例は、FTP サーバー (使用されているストレージAmazon S3) でアップロードが正常に完了したイベントを示しています。

```
{
```

```
    "version": "0",
    "id": "event-ID",
    "detail-type": "FTP Server File Upload Completed",
    "source": "aws.transfer",
    "account": "958412138249",
    "time": "2024-01-29T16:31:43Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:transfer:us-east-1:958412138249:server/s-1111aaaa2222bbbb3"
    ],
    "detail": {
        "status-code": "COMPLETED",
        "protocol": "FTP",
        "bytes": 1048576,
        "client-ip": "10.0.0.141",
        "end-timestamp": "2024-01-29T16:31:43.311866408Z",
        "etag": "b6d81b360a5672d80c27430f39153e2c",
        "file-path": "/DOC-EXAMPLE-BUCKET/test/1mb_file",
        "server-id": "s-1111aaaa2222bbbb3",
        "username": "test",
        "session-id": "event-ID",
        "start-timestamp": "2024-01-29T16:31:42.462088327Z"
    }
}
```

SFTP コネクタイベント

SFTP コネクタイベントの詳細フィールドは次のとおりです。

- SFTP コネクタファイルの送信が完了しました
- SFTP コネクタファイルの送信に失敗しました
- SFTP コネクタファイルの取得が完了しました
- SFTP コネクタファイルの取得に失敗しました
- SFTP コネクタディレクトリの一覧表示が完了しました
- SFTP コネクタディレクトリの一覧表示に失敗しました

`source` および `detail-type`フィールドには、 Transfer Family イベントの特定の値が含まれているため、以下が含まれます。すべてのイベントに含まれる他のメタデータフィールドの定義については、「 ユーザーガイド」の「イベント構造のリファレンスAmazon EventBridge 」を参照してください。

```
{
  . . .,
  "detail-type": "string",
  "source": "aws.transfer",
  . . .,
  "detail": {
    "operation" : "string",
    "max-items" : "number",
    "connector-id" : "string",
    "output-directory-path" : "string",
    "listing-id" : "string",
    "transfer-id" : "string",
    "file-transfer-id" : "string",
    "url" : "string",
    "file-path" : "string",
    "status-code" : "string",
    "failure-code" : "string",
    "failure-message" : "string",
    "start-timestamp" : "string",
    "end-timestamp" : "string",
    "local-directory-path" : "string",
    "remote-directory-path" : "string"
    "item-count" : "number"
    "truncated" : "boolean"
    "bytes" : "number",
    "local-file-location" : {
      "domain" : "string",
      "bucket" : "string",
      "key" : "string"
    },
    "output-file-location" : {
      "domain" : "string",
      "bucket" : "string",
      "key" : "string"
    }
  }
}
```

`detail-type`

イベントのタイプを示します。

このイベントの場合、値は前述の SFTP コネクタイベント名の 1 つです。

source

イベントを発生させたサービスを識別します。イベントの場合 Transfer Family 、この値は ですaws.transfer。

detail

イベントに関する情報を含む JSON オブジェクト。イベントを生成するサービスによって、このフィールドの内容が決まります。

このイベントでは、データには以下が含まれます。

max-items

返されるディレクトリ/ファイル名の最大数。

operation

StartFileTransfer リクエストがファイルを送信または取得するかどうか。値: SEND| RETRIEVE。

connector-id

使用されている SFTP コネクタの一意の識別子。

output-directory-path

ファイル/ディレクトリのリストの結果を保存する Amazon S3 のパス (バケットとプレフィックス) 。

listing-id

StartDirectoryListing API コールの一意の識別子。この識別子を使用して、CloudWatch ログをチェックし、リストリクエストのステータスを確認できます。

transfer-id

転送イベント (StartFileTransferリクエスト) の一意の識別子。

file-transfer-id

転送されるファイルの一意の識別子。

url

パートナーの AS2 または SFTP エンドポイントの URL。

`file-path`

送信または取得される場所とファイル。

`status-code`

転送が成功したかどうか。値: `FAILED | COMPLETED`。

`failure-code`

失敗した転送の場合、転送が失敗した理由の理由コード。

`failure-message`

失敗した転送の場合、転送が失敗した理由の詳細。

`start-timestamp`

転送が成功した場合、ファイル処理が開始される時刻のタイムスタンプ。

`end-timestamp`

転送が成功した場合、ファイル処理が完了したときのタイムスタンプ。

`local-directory-path`

RETRIEVE リクエストの場合、取得したファイルを配置する場所。

`remote-directory-path`

SEND リクエストの場合、ファイルをパートナーの SFTP サーバーに配置するファイルディレクトリ。これは、`RemoteDirectoryPath`ユーザーが`StartFileTransfer`リクエストに渡した の値です。パートナーの SFTP サーバーでデフォルトのディレクトリを指定できます。その場合は、このフィールドは空です。

`item-count`

出品リクエストに対して返された項目 (ディレクトリとファイル) の数。

`truncated`

リスト出力にリモートディレクトリに含まれるすべての項目が含まれているかどうか。

`bytes`

転送されるバイト数。失敗した転送の値は 0 です。

`local-file-location`

このパラメータには、 AWS ストレージファイルの場所の詳細が含まれます。

`domain`

使用されているストレージ。現在、唯一の値は ですS3。

`bucket`

Amazon S3 内のオブジェクトのコンテナ。

`key`

Amazon S3 のオブジェクトに割り当てられた名前。

`output-file-location`

このパラメータには、ディレクトリリストの結果を AWS ストレージに保存する場所の詳細が含まれます。

`domain`

使用されているストレージ。現在、唯一の値は ですS3。

`bucket`

Amazon S3 内のオブジェクトのコンテナ。

`key`

Amazon S3 のオブジェクトに割り当てられた名前。

Example SFTP コネクタファイル送信失敗の例イベント

次の例は、リモート SFTP サーバーにファイルを送信しようとしたときに SFTP コネクタが失敗したイベントを示しています。

```
{
    "version": "0",
    "id": "event-ID",
    "detail-type": "SFTP Connector File Send Failed",
    "source": "aws.transfer",
    "account": "123456789012",
    "time": "2024-01-24T19:30:45Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:transfer:us-east-1:123456789012:connector/c-f1111aaaa2222bbbb3"
    ],
```

```
    "detail": {
        "operation": "SEND",
        "connector-id": "c-f1111aaaa2222bbbb3",
        "transfer-id": "transfer-ID",
        "file-transfer-id": "file-transfer-ID",
        "url": "sftp://s-21a23456789012a.server.transfer.us-east-1.amazonaws.com",
        "file-path": "/DOC-EXAMPLE-BUCKET/testfile.txt",
        "status-code": "FAILED",
        "failure-code": "CONNECTION_ERROR",
        "failure-message": "Unknown Host",
        "remote-directory-path": "",
        "bytes": 0,
        "start-timestamp": "2024-01-24T18:29:33.658729Z",
        "end-timestamp": "2024-01-24T18:29:33.993196Z",
        "local-file-location": {
            "domain": "S3",
            "bucket": "DOC-EXAMPLE-BUCKET",
            "key": "testfile.txt"
        }
    }
}
```

Example SFTP コネクタファイルの取得完了サンプルイベント

次の例は、SFTP コネクタがリモート SFTP サーバーから送信されたファイルを正常に取得するイベントを示しています。

```
{
    "version": "0",
    "id": "event-ID",
    "detail-type": "SFTP Connector File Retrieve Completed",
    "source": "aws.transfer",
    "account": "123456789012",
    "time": "2024-01-24T18:28:08Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:transfer:us-east-1:123456789012:connector/c-f1111aaaa2222bbbb3"
    ],
    "detail": {
        "operation": "RETRIEVE",
        "connector-id": "c-fc68000012345aa18",
        "transfer-id": "file-transfer-ID",
        "file-transfer-id": "file-transfer-ID",
```

```
        "url": "sftp://s-21a23456789012a.server.transfer.us-east-1.amazonaws.com",
        "file-path": "testfile.txt",
        "status-code": "COMPLETED",
        "local-directory-path": "/DOC-EXAMPLE-BUCKET",
        "bytes": 63533,
        "start-timestamp": "2024-01-24T18:28:07.632388Z",
        "end-timestamp": "2024-01-24T18:28:07.774898Z",
        "local-file-location": {
            "domain": "S3",
            "bucket": "DOC-EXAMPLE-BUCKET",
            "key": "testfile.txt"
        }
    }
}
```

Example SFTP Connector Directory Listing Completed サンプルイベント

次の例は、スタートディレクトリリスト呼び出しがリモート SFTP サーバーからリストファイルを取得するイベントを示しています。

```
{
    "version": "0",
    "id": "event-ID",
    "detail-type": "SFTP Connector Directory Listing Completed",
    "source": "aws.transfer",
    "account": "123456789012",
    "time": "2024-01-24T18:28:08Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:transfer:us-east-1:123456789012:connector/c-f1111aaaa2222bbbb3"
    ],
    "detail": {
        "max-items": 10000,
        "connector-id": "c-fc68000012345aa18",
        "output-directory-path": "/DOC-EXAMPLE-BUCKET/example/file-listing-output",
        "listing-id": "123456-23aa-7980-abc1-1a2b3c4d5e",
        "url": "sftp://s-21a23456789012a.server.transfer.us-east-1.amazonaws.com",

        "status-code": "COMPLETED",
        "remote-directory-path": "/home",
        "item-count": 10000,
        "truncated": true,
        "start-timestamp": "2024-01-24T18:28:07.632388Z",
```

```
        "end-timestamp": "2024-01-24T18:28:07.774898Z",
        "output-file-location": {
            "domain": "S3",
            "bucket": "DOC-EXAMPLE-BUCKET",
            "key": "c-fc1ab90fd0d047e7a-70987273-49nn-4006-bab1-1a7290cc412ba.json"
        }
    }
}
```

AS2 イベント

AS2 イベントの詳細フィールドは次のとおりです。

- AS2 ペイロード受信完了
- AS2 ペイロード受信失敗
- AS2 ペイロードの送信が完了しました
- AS2 ペイロードの送信に失敗しました
- AS2 MDN 受信完了
- AS2 MDN 受信失敗
- AS2 MDN 送信が完了しました
- AS2 MDN 送信失敗

`source` および `detail-type`フィールドには、 Transfer Family イベントの特定の値が含まれているため、以下が含まれます。すべてのイベントに含まれる他のメタデータフィールドの定義については、「 ユーザーガイド」の「イベント構造のリファレンスAmazon EventBridge 」を参照してください。

```
{
   . . .,
  "detail-type": "string",
  "source": "aws.transfer",
  . . .,
  "detail": {
    "s3-attributes" : {
      "file-bucket" : "string",
      "file-key" : "string",
      "json-bucket" : "string",
      "json-key" : "string",
```

```
      "mdn-bucket" : "string",
      "mdn-key" : "string"
      }
    "mdn-subject" : "string",
    "mdn-message-id" : "string",
    "disposition" : "string",
    "bytes" : "number",
    "as2-from" : "string",
    "as2-message-id" : "string",
    "as2-to" : "string",
    "connector-id" : "string",
    "client-ip" : "string",
    "agreement-id" : "string",
    "server-id" : "string",
    "requester-file-name" : "string",
    "message-subject" : "string",
    "start-timestamp" : "string",
    "end-timestamp" : "string",
    "status-code" : "string",
    "failure-code" : "string",
    "failure-message" : "string",
    "transfer-id" : "string"
  }
}
```

`detail-type`

イベントのタイプを示します。

このイベントの場合、値は前述の AS2 イベントの 1 つです。

`source`

イベントを発生させたサービスを識別します。イベントの場合 Transfer Family 、この値は です`aws.transfer`。

`detail`

イベントに関する情報を含む JSON オブジェクト。このフィールドの内容は、イベントを生成するサービスによって決まります。

`s3-attributes`

転送されるファイルの Amazon S3 バケットとキーを識別します。MDN イベントの場合、MDN ファイルのバケットとキーも識別されます。

`file-bucket`

Amazon S3 内のオブジェクトのコンテナ。

`file-key`

Amazon S3 のオブジェクトに割り当てられた名前。

`json-bucket`

COMPLETED 転送または FAILED 転送の場合、JSON ファイルのコンテナ。

`json-key`

COMPLETED 転送または FAILED 転送の場合、Amazon S3 の JSON ファイルに割り当てられた名前。

`mdn-bucket`

MDN イベントの場合、MDN ファイルのコンテナ。

`mdn-key`

MDN イベントの場合、Amazon S3 の MDN ファイルに割り当てられた名前。

`mdn-subject`

MDN イベントの場合、メッセージ処理に関するテキストの説明。

`mdn-message-id`

MDN イベントの場合、MDN メッセージの一意の ID。

`disposition`

MDN イベントの場合、処理のカテゴリ。

`bytes`

メッセージ内のバイト数。

`as2-from`

メッセージを送信している AS2 取引相手。

`as2-message-id`

転送される AS2 メッセージの一意の識別子。

`as2-to`

メッセージを受信している AS2 取引相手。

`connector-id`

Transfer Family サーバーから取引相手に送信される AS2 メッセージの場合、使用されている AS2 コネクタの一意の識別子。

`client-ip`

サーバーイベント (取引相手から Transfer Family サーバーへの転送) の場合、転送に関係するクライアントの IP アドレス。

`agreement-id`

サーバーイベントの場合、AS2 契約の一意の識別子。

`server-id`

サーバーイベントの場合、Transfer Family サーバーのみの一意の ID。

`requester-file-name`

ペイロードイベントの場合、転送中に受信したファイルの元の名前。

`message-subject`

メッセージの件名に関するテキストの説明。

`start-timestamp`

転送が成功した場合、ファイル処理が開始される時刻のタイムスタンプ。

`end-timestamp`

転送が成功した場合、ファイル処理が完了したときのタイムスタンプ。

`status-code`

AS2 メッセージ転送プロセスの状態に対応するコード。有効な値: `COMPLETED | FAILED | PROCESSING`。

`failure-code`

失敗した転送の場合、転送が失敗した理由のカテゴリ。

`failure-message`

失敗した転送の場合、転送が失敗した理由の詳細。

`transfer-id`

転送イベントの一意の識別子。

Example AS2 ペイロード受信完了サンプルイベント

```
{
    "version": "0",
     "id": "event-ID",
    "detail-type": "AS2 Payload Receive Completed",
    "source": "aws.transfer",
    "account": "076722215406",
    "time": "2024-02-07T06:47:05Z",
    "region": "us-east-1",
    "resources": ["arn:aws:transfer:us-east-1:076722215406:connector/
c-1111aaaa2222bbbb3"],
    "detail": {
        "s3-attributes": {
            "file-key": "/inbound/processed/testAs2Message.dat",
            "file-bucket": "DOC-EXAMPLE-BUCKET"
        },
        "client-ip": "client-IP-address",
        "requester-file-name": "testAs2MessageVerifyFile.dat",
        "end-timestamp": "2024-02-07T06:47:06.040031Z",
        "as2-from": "as2-from-ID",
        "as2-message-id": "as2-message-ID",
        "message-subject": "Message from AS2 tests",
        "start-timestamp": "2024-02-07T06:47:05.410Z",
        "status-code": "PROCESSING",
        "bytes": 63,
        "as2-to": "as2-to-ID",
        "agreement-id": "a-1111aaaa2222bbbb3",
        "server-id": "s-1234abcd5678efghi"
    }
}
```

Example AS2 MDN 受信失敗サンプルイベント

```
{
  "version": "0",
  "id": "event-ID",
  "detail-type": "AS2 MDN Receive Failed",
```

```
  "source": "aws.transfer",
  "account": "889901007463",
  "time": "2024-02-06T22:05:09Z",
  "region": "us-east-1",
  "resources": ["arn:aws:transfer:us-east-1:076722215406:server/s-1111aaaa2222bbbb3"],
  "detail": {
      "mdn-subject": "Your Requested MDN Response re: Test run from Id 123456789abcde
 to partner ijklmnop987654",
      "s3-attributes": {
          "json-bucket": "DOC-EXAMPLE-BUCKET1",
          "file-key": "/as2Integ/TestOutboundWrongCert.dat",
          "file-bucket": "DOC-EXAMPLE-BUCKET2",
          "json-key": "/as2Integ/failed/TestOutboundWrongCert.dat.json"
      },
      "mdn-message-id": "MDN-message-ID",
      "end-timestamp": "2024-02-06T22:05:09.479878Z",
      "as2-from": "PartnerA",
      "as2-message-id": "as2-message-ID",
      "connector-id": "c-1234abcd5678efghj",
      "message-subject": "Test run from Id 123456789abcde to partner ijklmnop987654",
      "start-timestamp": "2024-02-06T22:05:03Z",
      "failure-code": "VERIFICATION_FAILED_NO_MATCHING_KEY_FOUND",
      "status-code": "FAILED",
      "as2-to": "MyCompany",
      "failure-message": "No public certificate matching message signature could be
 found in profile: p-1234abcd5678efghj",
      "transfer-id": "transfer-ID"
  }
}
```

のセキュリティ AWS Transfer Family

のクラウドセキュリティが最優先事項 AWS です。お客様は AWS 、セキュリティを最も重視する組織の要件を満たすように構築されたデータセンターとネットワークアーキテクチャからメリットを得られます。

セキュリティは、 AWS とユーザーの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。

AWS のサービス が特定のコンプライアンスプログラムの範囲内にあるかどうかを確認するには、コンプライアンスプログラムAWS のサービス による対象範囲内のコンプライアンスプログラムを参照し、関心のあるコンプライアンスプログラムを選択します。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。

を使用して、サードパーティーの監査レポートをダウンロードできます AWS Artifact。詳細については、「 でのレポートのダウンロード AWS Artifact」の」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 のサービス は、お客様のデータの機密性、貴社のコンプライアンス目的、適用される法律および規制によって決まります。 は、コンプライアンスに役立つ以下のリソース AWS を提供しています。

- セキュリティとコンプライアンスのクイックスタートガイド – これらのデプロイガイドでは、アーキテクチャ上の考慮事項について説明し、セキュリティとコンプライアンスに重点を置いたベースライン環境 AWS を にデプロイする手順について説明します。
- アマゾン ウェブ サービスにおける HIPAA セキュリティとコンプライアンスのためのアーキテクチャ – このホワイトペーパーでは、企業が AWS を使用して HIPAA 対象アプリケーションを作成する方法について説明します。

> **Note**
>
> すべて AWS のサービス HIPAA の対象となるわけではありません。詳細については、「HIPAA 対応サービスのリファレンス」を参照してください。

- AWS コンプライアンスリソース – このワークブックとガイドのコレクションは、お客様の業界や地域に適用される場合があります。
- AWS カスタマーコンプライアンスガイド — コンプライアンスの観点から責任共有モデルを理解します。このガイドでは、ガイダンスを保護し AWS のサービス 、複数のフレームワーク (米国国

立標準技術研究所 (NIST)、Payment Card Industry Security Standards Council (PCI)、国際標準化機構 (ISO) を含む) のセキュリティコントロールにマッピングするためのベストプラクティスをまとめています。

- 「 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、および規制にどの程度準拠しているかを評価します。 AWS Config
- AWS Security Hub – これにより AWS のサービス 、 内のセキュリティ状態を包括的に確認できます AWS。Security Hub では、セキュリティコントロールを使用して AWS リソースを評価し、セキュリティ業界標準とベストプラクティスに対するコンプライアンスをチェックします。サポートされているサービスとコントロールのリストについては、「Security Hub のコントロールリファレンス」を参照してください。
- Amazon GuardDuty – これにより AWS アカウント、疑わしいアクティビティや悪意のあるアクティビティがないか環境を監視することで、、、ワークロード、コンテナ、データに対する潜在的な脅威 AWS のサービス を検出します。 GuardDuty は、特定のコンプライアンスフレームワークで義務付けられている侵入検知要件を満たすことで、PCI DSS などのさまざまなコンプライアンス要件への対応に役立ちます。
- AWS Audit Manager – これにより AWS のサービス 、 AWS 使用状況を継続的に監査し、リスクの管理方法と規制や業界標準への準拠を簡素化できます。

このドキュメントは、 を使用する際の責任共有モデルの適用方法を理解するのに役立ちます AWS Transfer Family。以下のトピックでは、セキュリティおよびコンプライアンスの目的を達成するために AWS Transfer Family を設定する方法を示します。また、 AWS Transfer Family リソースのモニタリングや保護に役立つ他の AWS のサービスの使用方法についても説明します。

既存のアプリケーションを変更したり、サーバーインフラストラクチャを管理 AWS したりすることなく、スケーラブルで安全なファイル転送アーキテクチャを で構築する方法に関する規範的なガイダンスと実践的なラボを提供するワークショップを提供しています。このワークショップの詳細については、「こちら」をご覧ください。

トピック

- AWS Transfer Family サーバーのセキュリティポリシー
- AWS Transfer Family SFTP コネクタのセキュリティポリシー
- AWS Transfer Familyでハイブリッドポストクォンタムキー交換の使用
- でのデータ保護 AWS Transfer Family
- の Identity and Access Management AWS Transfer Family

- のコンプライアンス検証 AWS Transfer Family
- の耐障害性 AWS Transfer Family
- のインフラストラクチャセキュリティ AWS Transfer Family
- ウェブアプリケーションファイアウォールを追加する
- サービス間の混乱した代理の防止
- AWSAWS Transfer Family の マネージドポリシー

AWS Transfer Family サーバーのセキュリティポリシー

のサーバーセキュリティポリシー AWS Transfer Family を使用すると、サーバーに関連付けられた一連の暗号化アルゴリズム (メッセージ認証コード (MACsキー交換 (KEXs)、暗号スイート) を制限できます。サポートされている暗号アルゴリズムのリストについては、「暗号アルゴリズム」を参照してください。サーバーホストキーとサービス管理ユーザーキーでの使用がサポートされているキーアルゴリズムのリストについては、ユーザーキーとサーバーキーでサポートされるアルゴリズム を参照してください。

> **Note**
>
> サーバーを最新のセキュリティポリシーに更新することを強くお勧めします。最新のセキュリティポリシーがデフォルトです。を使用して CloudFormation Transfer Family サーバーを作成し、デフォルトのセキュリティポリシーを受け入れるすべてのお客様には、最新のポリシーが自動的に割り当てられます。クライアントの互換性について懸念がある場合は、変更される可能性があるデフォルトポリシーを使用するのではなく、サーバーの作成または更新時に使用するセキュリティポリシーを肯定的に記述してください。
> サーバーのセキュリティポリシーを変更するには、「」を参照してくださいセキュリティポリシーを編集する。

Transfer Family のセキュリティの詳細については、ブログ記事「How Transfer Family can help you build a secure, compliant managed file transfer solution」を参照してください。

トピック

- 暗号アルゴリズム
- TransferSecurityPolicy-2024-01
- TransferSecurityPolicy-2023-05

- TransferSecurityPolicy-2022-03
- TransferSecurityPolicy-2020-06
- TransferSecurityPolicy-2018-11
- TransferSecurityPolicy-FIPS-2024-01/TransferSecurityPolicy-FIPS-2024-05
- TransferSecurityPolicy-FIPS-2023-05
- TransferSecurityPolicy-FIPS-2020-06
- ポスト・クアンタム・セキュリティ・ポリシー

> **Note**
>
> `TransferSecurityPolicy-2024-01` は、コンソール、API、または CLI を使用してサーバーを作成するときにサーバーにアタッチされるデフォルトのセキュリティポリシーです。

暗号アルゴリズム

ホストキーでは、次のアルゴリズムがサポートされています。

- `rsa-sha2-256`
- `rsa-sha2-512`
- `ecdsa-sha2-nistp256`
- `ecdsa-sha2-nistp384`
- `ecdsa-sha2-nistp521`
- `ssh-ed25519`

さらに、次のセキュリティポリシーで が許可されます`ssh-rsa`。

- TransferSecurityPolicy-2018-11
- TransferSecurityPolicy-2020-06
- TransferSecurityPolicy-FIPS-2020-06
- TransferSecurityPolicy-FIPS-2023-05
- TransferSecurityPolicy-FIPS-2024-01
- TransferSecurityPolicy-PQ-SSH-FIPS-Experimental-2023-04

> **Note**
>
> 常に である RSA キータイプssh-rsaと、サポートされているアルゴリズムのいずれかである RSA ホストキーアルゴリズムの違いを理解することが重要です。

以下は、各セキュリティポリシーでサポートされている暗号アルゴリズムの一覧です。

> **Note**
>
> 次の表とポリシーでは、アルゴリズムタイプの次の使用に注意してください。
>
> - SFTP サーバーは、SshCiphers、、SshKexsおよび SshMacsセクションのアルゴリズムのみを使用します。
> - FTPS サーバーは、 TlsCiphersセクションのアルゴリズムのみを使用します。
> - FTP サーバーは暗号化を使用しないため、これらのアルゴリズムは使用しません。
> - FIPS-2024-05 と FIPS-2024-01 のセキュリティポリシーは同じですが、FIPS-2024-05 はssh-rsaアルゴリズムをサポートしていません。

セキュリティポリシー	2024-01	2023-05	2022-03	2020-06	FIPS-2024-05 FIPS-2024-01	FIPS-2023-05	FIPS-2020-06	2018-11
SshCiphers								
aes128-ctr	♦			♦	♦		♦	♦
aes128-gcm@openssh.com	♦	♦	♦	♦	♦	♦	♦	♦
aes192-ctr	♦	♦	♦	♦	♦	♦	♦	♦

セキュリティポリシー	2024-01	2023-05	2022-03	2020-06	FIPS-2024-05 FIPS-2024-01	FIPS-2023-05	FIPS-2020-06	2018-11
aes256-ctr	♦	♦	♦	♦	♦	♦	♦	♦
aes256-gcm@openssh.com	♦	♦	♦	♦	♦	♦	♦	♦
chacha20-poly1305@openssh.com				♦				♦
SshKexs								
curve25519-sha256	♦	♦	♦					♦
curve25519-sha256@libssh.org	♦	♦	♦					♦
diffie-hellman-group14-sha1								♦

セキュリティポリシー	2024-01	2023-05	2022-03	2020-06	FIPS-2024-05 FIPS-2024-01	FIPS-2023-05	FIPS-2020-06	2018-11
diffie-hellman-group14-sha256				♦			♦	♦
diffie-hellman-group16-sha512	♦	♦	♦	♦	♦	♦	♦	♦
diffie-hellman-group18-sha512	♦	♦	♦	♦	♦	♦	♦	♦
diffie-hellman-group-exchange-sha256		♦	♦	♦		♦	♦	♦

セキュリティポリシー	2024-01	2023-05	2022-03	2020-06	FIPS-2024-05 FIPS-2024-01	FIPS-2023-05	FIPS-2020-06	2018-11
ecdh-nist p256-kyber-512r3-sha256-d00 @openquantumsafe.org	♦				♦			
ecdh-nist p384-kyber-768r3-sha384-d00 @openquantumsafe.org	♦				♦			

セキュリティポリシー	2024-01	2023-05	2022-03	2020-06	FIPS-2024-05 FIPS-2024-01	FIPS-2023-05	FIPS-2020-06	2018-11
ecdh-nistp521-kyber-1024r3-sha512-d00@openquantumsafe.org	♦				♦			
ecdh-sha2-nistp256	♦			♦	♦		♦	♦
ecdh-sha2-nistp384	♦			♦	♦		♦	♦
ecdh-sha2-nistp521	♦			♦	♦		♦	♦
x25519-kyber-512r3-sha256-d00@amazon.com	♦							

セキュリティポリシー	2024-01	2023-05	2022-03	2020-06	FIPS-2024-05 FIPS-2024-01	FIPS-2023-05	FIPS-2020-06	2018-11
SshMacs								
hmac-sha1								♦
hmac-sha1-etm@openssh.com								♦
hmac-sha2-256			♦	♦			♦	♦
hmac-sha2-256-etm@openssh.com	♦	♦	♦	♦	♦	♦	♦	♦
hmac-sha2-512			♦	♦			♦	♦
hmac-sha2-512-etm@openssh.com	♦	♦	♦	♦	♦	♦	♦	♦

セキュリティポリシー	2024-01	2023-05	2022-03	2020-06	FIPS-2024-05 FIPS-2024-01	FIPS-2023-05	FIPS-2020-06	2018-11
umac-128-etm@openssh.com				♦				♦
umac-128@openssh.com				♦				♦
umac-64-etm@openssh.com								♦
umac-64@openssh.com								♦
TlsCiphers								
TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256		♦	♦	♦	♦	♦	♦	♦
TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256		♦	♦	♦	♦	♦	♦	♦

セキュリティポリシー	2024-01	2023-05	2022-03	2020-06	FIPS-2024-05 FIPS-2024-01	FIPS-2023-05	FIPS-2020-06	2018-11
TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384	♦	♦	♦	♦	♦	♦	♦	♦
TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384	♦	♦	♦	♦	♦	♦	♦	♦
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256	♦	♦	♦	♦	♦	♦	♦	♦
TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256	♦	♦	♦	♦	♦	♦	♦	♦
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384	♦	♦	♦	♦	♦	♦	♦	♦

セキュリティポリシー	2024-01	2023-05	2022-03	2020-06	FIPS-2024-05 FIPS-2024-01	FIPS-2023-05	FIPS-2020-06	2018-11
TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384	♦	♦	♦	♦	♦	♦	♦	♦
TLS_RSA_WITH_AES_128_CBC_SHA256								♦
TLS_RSA_WITH_AES_256_CBC_SHA256								♦

TransferSecurityPolicy-2024-01

以下に、 TransferSecurityPolicy2024 年 1 月のセキュリティポリシーを示します。

```
{
    "SecurityPolicy": {
        "Fips": false,
        "SecurityPolicyName": "TransferSecurityPolicy-2024-01",
        "SshCiphers": [
            "aes128-gcm@openssh.com",
            "aes256-gcm@openssh.com",
            "aes128-ctr",
            "aes256-ctr",
            "aes192-ctr"
        ],
        "SshKexs": [
            "ecdh-nistp384-kyber-768r3-sha384-d00@openquantumsafe.org",
```

```
            "x25519-kyber-512r3-sha256-d00@amazon.com",
            "ecdh-nistp256-kyber-512r3-sha256-d00@openquantumsafe.org",
            "ecdh-nistp521-kyber-1024r3-sha512-d00@openquantumsafe.org",
            "ecdh-sha2-nistp256",
            "ecdh-sha2-nistp384",
            "ecdh-sha2-nistp521",
            "curve25519-sha256",
            "curve25519-sha256@libssh.org",
            "diffie-hellman-group18-sha512",
            "diffie-hellman-group16-sha512",
            "diffie-hellman-group-exchange-sha256"
        ],
        "SshMacs": [
            "hmac-sha2-256-etm@openssh.com",
            "hmac-sha2-512-etm@openssh.com"
        ],
        "TlsCiphers": [
            "TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256",
            "TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256",
            "TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256",
            "TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256",
            "TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384",
            "TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384",
            "TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384",
            "TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384"
        ]
    }
}
```

TransferSecurityPolicy-2023-05

以下に、-2023 TransferSecurityPolicy年 5 月のセキュリティポリシーを示します。

```
{
    "SecurityPolicy": {
        "Fips": false,
        "SecurityPolicyName": "TransferSecurityPolicy-2023-05",
        "SshCiphers": [
            "aes256-gcm@openssh.com",
            "aes128-gcm@openssh.com",
            "aes256-ctr",
            "aes192-ctr"
        ],
```

```
        "SshKexs": [
            "curve25519-sha256",
            "curve25519-sha256@libssh.org",
            "diffie-hellman-group16-sha512",
            "diffie-hellman-group18-sha512",
            "diffie-hellman-group-exchange-sha256"
        ],
        "SshMacs": [
            "hmac-sha2-512-etm@openssh.com",
            "hmac-sha2-256-etm@openssh.com"
        ],
        "TlsCiphers": [
            "TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256",
            "TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256",
            "TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256",
            "TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256",
            "TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384",
            "TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384",
            "TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384",
            "TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384"
        ]
    }
}
```

TransferSecurityPolicy-2022-03

以下に、 TransferSecurityPolicy-2022 年 3 月のセキュリティポリシーを示します。

```
{
  "SecurityPolicy": {
    "Fips": false,
    "SecurityPolicyName": "TransferSecurityPolicy-2022-03",
    "SshCiphers": [
      "aes256-gcm@openssh.com",
      "aes128-gcm@openssh.com",
      "aes256-ctr",
      "aes192-ctr"
    ],
    "SshKexs": [
      "curve25519-sha256",
      "curve25519-sha256@libssh.org",
      "diffie-hellman-group16-sha512",
      "diffie-hellman-group18-sha512",
```

```
      "diffie-hellman-group-exchange-sha256"
    ],
    "SshMacs": [
      "hmac-sha2-512-etm@openssh.com",
      "hmac-sha2-256-etm@openssh.com",
      "hmac-sha2-512",
      "hmac-sha2-256"
    ],
    "TlsCiphers": [
      "TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256",
      "TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256",
      "TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256",
      "TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256",
      "TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384",
      "TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384",
      "TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384",
      "TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384"
    ]
  }
}
```

TransferSecurityPolicy-2020-06

以下に、 TransferSecurityPolicy-2020 年 6 月のセキュリティポリシーを示します。

```
{
  "SecurityPolicy": {
    "Fips": false,
    "SecurityPolicyName": "TransferSecurityPolicy-2020-06",
    "SshCiphers": [
      "chacha20-poly1305@openssh.com",
      "aes128-ctr",
      "aes192-ctr",
      "aes256-ctr",
      "aes128-gcm@openssh.com",
      "aes256-gcm@openssh.com"
    ],
    "SshKexs": [
      "ecdh-sha2-nistp256",
      "ecdh-sha2-nistp384",
      "ecdh-sha2-nistp521",
      "diffie-hellman-group-exchange-sha256",
      "diffie-hellman-group16-sha512",
```

```
      "diffie-hellman-group18-sha512",
      "diffie-hellman-group14-sha256"
    ],
    "SshMacs": [
      "umac-128-etm@openssh.com",
      "hmac-sha2-256-etm@openssh.com",
      "hmac-sha2-512-etm@openssh.com",
      "umac-128@openssh.com",
      "hmac-sha2-256",
      "hmac-sha2-512"
    ],
    "TlsCiphers": [
      "TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256",
      "TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256",
      "TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256",
      "TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256",
      "TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384",
      "TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384",
      "TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384",
      "TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384"
    ]
  }
}
```

TransferSecurityPolicy-2018-11

以下に、2018 TransferSecurityPolicy年 11 月のセキュリティポリシーを示します。

```
{
  "SecurityPolicy": {
    "Fips": false,
    "SecurityPolicyName": "TransferSecurityPolicy-2018-11",
    "SshCiphers": [
      "chacha20-poly1305@openssh.com",
      "aes128-ctr",
      "aes192-ctr",
      "aes256-ctr",
      "aes128-gcm@openssh.com",
      "aes256-gcm@openssh.com"
    ],
    "SshKexs": [
      "curve25519-sha256",
      "curve25519-sha256@libssh.org",
```

```
      "ecdh-sha2-nistp256",
      "ecdh-sha2-nistp384",
      "ecdh-sha2-nistp521",
      "diffie-hellman-group-exchange-sha256",
      "diffie-hellman-group16-sha512",
      "diffie-hellman-group18-sha512",
      "diffie-hellman-group14-sha256",
      "diffie-hellman-group14-sha1"
    ],
    "SshMacs": [
      "umac-64-etm@openssh.com",
      "umac-128-etm@openssh.com",
      "hmac-sha2-256-etm@openssh.com",
      "hmac-sha2-512-etm@openssh.com",
      "hmac-sha1-etm@openssh.com",
      "umac-64@openssh.com",
      "umac-128@openssh.com",
      "hmac-sha2-256",
      "hmac-sha2-512",
      "hmac-sha1"
    ],
    "TlsCiphers": [
      "TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256",
      "TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256",
      "TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256",
      "TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256",
      "TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384",
      "TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384",
      "TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384",
      "TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384",
      "TLS_RSA_WITH_AES_128_CBC_SHA256",
      "TLS_RSA_WITH_AES_256_CBC_SHA256"
    ]
  }
}
```

TransferSecurityPolicy-FIPS-2024-01/TransferSecurityPolicy-FIPS-2024-05

以下に、 TransferSecurityPolicy-FIPS-2024-01 および TransferSecurityPolicy-FIPS-2024-05 セキュリティポリシーを示します。

> **Note**
>
> FIPS サービスエンドポイントと TransferSecurityPolicy-FIPS-2024-01 および TransferSecurityPolicy-FIPS-2024-05 セキュリティポリシーは、一部の AWS リージョンでのみ使用できます。詳細については、「AWS 全般のリファレンス」の「AWS Transfer Family エンドポイントとクォータ」を参照してください。
> これら 2 つのセキュリティポリシーの唯一の違いは、 TransferSecurityPolicy-FIPS-2024-01 が`ssh-rsa`アルゴリズムをサポートし、 TransferSecurityPolicy-FIPS-2024-05 がアルゴリズムをサポートしないことです。

```
{
    "SecurityPolicy": {
        "Fips": true,
        "SecurityPolicyName": "TransferSecurityPolicy-FIPS-2024-01",
        "SshCiphers": [
            "aes128-gcm@openssh.com",
            "aes256-gcm@openssh.com",
            "aes128-ctr",
            "aes256-ctr",
            "aes192-ctr"
        ],
        "SshKexs": [
            "ecdh-nistp384-kyber-768r3-sha384-d00@openquantumsafe.org",
            "ecdh-nistp256-kyber-512r3-sha256-d00@openquantumsafe.org",
            "ecdh-nistp521-kyber-1024r3-sha512-d00@openquantumsafe.org",
            "ecdh-sha2-nistp256",
            "ecdh-sha2-nistp384",
            "ecdh-sha2-nistp521",
            "diffie-hellman-group18-sha512",
            "diffie-hellman-group16-sha512",
            "diffie-hellman-group-exchange-sha256"
        ],
        "SshMacs": [
            "hmac-sha2-256-etm@openssh.com",
            "hmac-sha2-512-etm@openssh.com"
        ],
        "TlsCiphers": [
            "TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256",
            "TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256",
            "TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256",
```

```
            "TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256",
            "TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384",
            "TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384",
            "TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384",
            "TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384"
        ]
    }
}
```

TransferSecurityPolicy-FIPS-2023-05

の FIPS 証明書の詳細については AWS Transfer Family 、「」を参照してください。 https://csrc.nist.gov/projects/cryptographic-module-validation-program/validated-modules/search/all

以下に、- TransferSecurityPolicy-FIPS-2023-05 セキュリティポリシーを示します。

> **Note**
>
> FIPS サービスエンドポイントと TransferSecurityPolicy-FIPS-2023-05 セキュリティポリシーは、一部の AWS リージョンでのみ使用できます。詳細については、「AWS 全般のリファレンス」の「AWS Transfer Family エンドポイントとクォータ」を参照してください。

```
{
    "SecurityPolicy": {
        "Fips": true,
        "SecurityPolicyName": "TransferSecurityPolicy-FIPS-2023-05",
        "SshCiphers": [
            "aes256-gcm@openssh.com",
            "aes128-gcm@openssh.com",
            "aes256-ctr",
            "aes192-ctr"
        ],
        "SshKexs": [
            "diffie-hellman-group16-sha512",
            "diffie-hellman-group18-sha512",
            "diffie-hellman-group-exchange-sha256"
        ],
        "SshMacs": [
            "hmac-sha2-256-etm@openssh.com",
            "hmac-sha2-512-etm@openssh.com"
        ],
```

```
        "TlsCiphers": [
            "TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256",
            "TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256",
            "TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256",
            "TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256",
            "TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384",
            "TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384",
            "TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384",
            "TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384"
        ]
    }
}
```

TransferSecurityPolicy-FIPS-2020-06

の FIPS 証明書の詳細については AWS Transfer Family 、「」を参照してください。 https://csrc.nist.gov/projects/cryptographic-module-validation-program/validated-modules/search/all

以下に、- TransferSecurityPolicy-FIPS-2020-06 セキュリティポリシーを示します。

> **Note**
>
> FIPS サービスエンドポイントと TransferSecurityPolicy-FIPS-2020-06 セキュリティポリシーは、一部の AWS リージョンでのみ使用できます。詳細については、「AWS 全般のリファレンス」の「AWS Transfer Family エンドポイントとクォータ」を参照してください。

```
{
  "SecurityPolicy": {
    "Fips": true,
    "SecurityPolicyName": "TransferSecurityPolicy-FIPS-2020-06",
    "SshCiphers": [
      "aes128-ctr",
      "aes192-ctr",
      "aes256-ctr",
      "aes128-gcm@openssh.com",
      "aes256-gcm@openssh.com"
    ],
    "SshKexs": [
      "ecdh-sha2-nistp256",
      "ecdh-sha2-nistp384",
      "ecdh-sha2-nistp521",
```

```
      "diffie-hellman-group-exchange-sha256",
      "diffie-hellman-group16-sha512",
      "diffie-hellman-group18-sha512",
      "diffie-hellman-group14-sha256"
    ],
    "SshMacs": [
      "hmac-sha2-256-etm@openssh.com",
      "hmac-sha2-512-etm@openssh.com",
      "hmac-sha2-256",
      "hmac-sha2-512"
    ],
    "TlsCiphers": [
      "TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256",
      "TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256",
      "TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256",
      "TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256",
      "TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384",
      "TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384",
      "TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384",
      "TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384"
    ]
  }
}
```

ポスト・クアンタム・セキュリティ・ポリシー

この表は、TTransfer Family ilyのポスト・クォンタム・セキュリティ・ポリシーのアルゴリズムを示しています。これらのポリシーについては、AWS Transfer Familyでハイブリッドポストクォンタムキー交換の使用 で詳しく説明しています。

ポリシーの一覧は表の通りです。

セキュリティポリシー	TransferSecurityPolicy-PQ-SSH-Experimental-2023-04	TransferSecurityPolicy-PQ-SSH-FIPS-Experimental-2023-04
SSH ciphers		
aes128-ctr		♦
aes128-gcm@openssh.com	♦	♦

セキュリティポリシー	TransferSecurityPolicy-PQ-SSH-Experimental-2023-04	TransferSecurityPolicy-PQ-SSH-FIPS-Experimental-2023-04
aes192-ctr	♦	♦
aes256-ctr	♦	♦
aes256-gcm@openssh.com	♦	♦
KEXs		
ecdh-nistp256-kyber-512r3-sha256-d00@openquantumsafe.org	♦	♦
ecdh-nistp384-kyber-768r3-sha384-d00@openquantumsafe.org	♦	♦
ecdh-nistp521-kyber-1024r3-sha512-d00@openquantumsafe.org	♦	♦
x25519-kyber-512r3-sha256-d00@amazon.com	♦	
diffie-hellman-group14-sha256		♦
diffie-hellman-group16-sha512	♦	♦
diffie-hellman-group18-sha512	♦	♦
ecdh-sha2-nistp384		♦
ecdh-sha2-nistp521		♦
diffie-hellman-group-exchange-sha256	♦	♦
ecdh-sha2-nistp256		♦

セキュリティポリシー	TransferSecurityPolicy-PQ-SSH-Experimental-2023-04	TransferSecurityPolicy-PQ-SSH-FIPS-Experimental-2023-04
curve25519-sha256@libssh.org	♦	
curve25519-sha256	♦	
MACs		
hmac-sha2-256-etm@openssh.com	♦	♦
hmac-sha2-256	♦	♦
hmac-sha2-512-etm@openssh.com	♦	♦
hmac-sha2-512	♦	♦
TLS ciphers		
TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256	♦	♦
TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256	♦	♦
TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384	♦	♦
TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384	♦	♦
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256	♦	♦
TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256	♦	♦

セキュリティポリシー	TransferSecurityPolicy-PQ-SSH-Experimental-2023-04	TransferSecurityPolicy-PQ-SSH-FIPS-Experimental-2023-04
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384	♦	♦
TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384	♦	♦

TransferSecurityPolicy-PQ-SSH-Experimental-2023-04

以下は、 TransferSecurityPolicy-PQ-SSH-Experimental-2023-04 セキュリティポリシーを示しています。

```
{
    "SecurityPolicy": {
        "Fips": false,
        "SecurityPolicyName": "TransferSecurityPolicy-PQ-SSH-Experimental-2023-04",
        "SshCiphers": [
            "aes256-gcm@openssh.com",
            "aes128-gcm@openssh.com",
            "aes256-ctr",
            "aes192-ctr"
        ],
        "SshKexs": [
            "ecdh-nistp384-kyber-768r3-sha384-d00@openquantumsafe.org",
            "x25519-kyber-512r3-sha256-d00@amazon.com",
            "ecdh-nistp256-kyber-512r3-sha256-d00@openquantumsafe.org",
            "ecdh-nistp521-kyber-1024r3-sha512-d00@openquantumsafe.org",
            "curve25519-sha256",
            "curve25519-sha256@libssh.org",
            "diffie-hellman-group16-sha512",
            "diffie-hellman-group18-sha512",
            "diffie-hellman-group-exchange-sha256"
        ],
        "SshMacs": [
            "hmac-sha2-512-etm@openssh.com",
            "hmac-sha2-256-etm@openssh.com",
            "hmac-sha2-512",
            "hmac-sha2-256"
```

```
        ],
        "TlsCiphers": [
            "TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256",
            "TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256",
            "TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256",
            "TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256",
            "TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384",
            "TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384",
            "TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384",
            "TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384"
        ]
    }
}
```

TransferSecurityPolicy-PQ-SSH-FIPS-Experimental-2023-04

以下は、 TransferSecurityPolicy-PQ-SSH-FIPS-Experimental-2023-04 セキュリティポリシーを示しています。

```
{
    "SecurityPolicy": {
        "Fips": true,
        "SecurityPolicyName": "TransferSecurityPolicy-PQ-SSH-FIPS-
Experimental-2023-04",
        "SshCiphers": [
            "aes256-gcm@openssh.com",
            "aes128-gcm@openssh.com",
            "aes256-ctr",
            "aes192-ctr",
            "aes128-ctr"
        ],
        "SshKexs": [
            "ecdh-nistp384-kyber-768r3-sha384-d00@openquantumsafe.org",
            "ecdh-nistp256-kyber-512r3-sha256-d00@openquantumsafe.org",
            "ecdh-nistp521-kyber-1024r3-sha512-d00@openquantumsafe.org",
            "ecdh-sha2-nistp256",
            "ecdh-sha2-nistp384",
            "ecdh-sha2-nistp521",
            "diffie-hellman-group-exchange-sha256",
            "diffie-hellman-group16-sha512",
            "diffie-hellman-group18-sha512",
            "diffie-hellman-group14-sha256"
        ],
```

```
        "SshMacs": [
            "hmac-sha2-512-etm@openssh.com",
            "hmac-sha2-256-etm@openssh.com",
            "hmac-sha2-512",
            "hmac-sha2-256"
        ],
        "TlsCiphers": [
            "TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256",
            "TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256",
            "TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256",
            "TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256",
            "TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384",
            "TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384",
            "TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384",
            "TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384"
        ]
    }
}
```

AWS Transfer Family SFTP コネクタのセキュリティポリシー

の SFTP コネクタセキュリティポリシー AWS Transfer Family を使用すると、SFTP コネクタに関連付けられた暗号化アルゴリズム (メッセージ認証コード (MACsキー交換 (KEXs)、暗号スイート) のセットを制限できます。以下は、各 SFTP コネクタセキュリティポリシーでサポートされている暗号化アルゴリズムのリストです。

> **Note**
>
> `TransferSFTPConnectorSecurityPolicy-2024-03` は、SFTP コネクタに適用されるデフォルトのセキュリティポリシーです。

コネクタのセキュリティポリシーは変更できます。Transfer Family の左側のナビゲーションペインからコネクタを選択し、コネクタを選択します。次に、「Sftp 設定」セクションの「編集」を選択します。暗号化アルゴリズムオプションセクションで、セキュリティポリシーフィールドのドロップダウンリストから使用可能なセキュリティポリシーを選択します。

セキュリティポリシー	TransferSFTPConnectorSecurityPolicy-2024-03	TransferSFTPConnectorSecurityPolicy-2023-07
Ciphers		
aes128-ctr		♦
aes128-gcm@openssh.com	♦	♦
aes192-ctr	♦	♦
aes256-ctr	♦	♦
aes256-gcm@openssh.com	♦	♦
Kexs		
curve25519-sha256	♦	♦
curve25519-sha256@libssh.org	♦	♦
diffie-hellman-group14-sha1		♦
diffie-hellman-group16-sha512	♦	♦
diffie-hellman-group18-sha512	♦	♦
diffie-hellman-group-exchange-sha256	♦	♦
Macs		
hmac-sha2-512-etm@openssh.com	♦	♦
hmac-sha2-256-etm@openssh.com	♦	♦
hmac-sha2-512	♦	♦
hmac-sha2-256	♦	♦

セキュリティポリシー	TransferSFTPConnectorSecurityPolicy-2024-03	TransferSFTPConnectorSecurityPolicy-2023-07
hmac-sha1		♦
hmac-sha1-96		♦
Host Key Algorithms		
rsa-sha2-256	♦	♦
rsa-sha2-512	♦	♦
ecdsa-sha2-nistp256	♦	♦
ecdsa-sha2-nistp384	♦	♦
ecdsa-sha2-nistp521	♦	♦
ssh-rsa		♦

AWS Transfer Familyでハイブリッドポストクォンタムキー交換の使用

AWS Transfer Family は、Secure Shell (SSH) プロトコルのハイブリッドポスト量子キー確立オプションをサポートします。ポストクォンタムキー確立が必要なのは、ネットワークトラフィックを記録し、将来クォンタムコンピューターによって復号化できるように保存することがすでに可能であるためです。これは、「store-now-harvest-later」攻撃と呼ばれています。

Transfer Family に接続すると、Amazon Simple Storage Service (Amazon S3) ストレージまたは Amazon Elastic File System (Amazon EFS) との間で安全なファイル転送を行う場合にこのオプションを使用できます。SSH でのポストクォンタムハイブリッドキー確立には、従来のキー交換アルゴリズムと組み合わせて使用するポストクォンタムキー確立メカニズムが導入されています。従来の暗号スイートで作成された SSH 鍵は、現在の技術ではブルートフォース攻撃から安全です。ただし、future、大規模な量子コンピューティングが登場しても、従来の暗号化は安全性を維持できないと予想されます。

もし、組織がトランスファーファミリー接続でやり取りされるデータの長期的な機密性に依存しているのであれば、大規模な量子コンピュータが利用可能になる前に、ポスト量子暗号への移行計画を検討すべきです。

現在暗号化されているデータを将来の潜在的な攻撃から保護するために、 AWS は量子耐性アルゴリズムまたはポスト量子アルゴリズムの開発に暗号コミュニティに参加しています。Transfer Family にハイブリッドポストクォンタムキー交換暗号スイートを実装しました。

これらのハイブリッド暗号スイートは、ほとんどの AWS リージョンで本稼働ワークロードで使用できます。ただし、ハイブリッド暗号スイートのパフォーマンス特性と帯域幅要件は、古典的な鍵交換メカニズ ムとは異なるため、Transfer Family の接続でテストすることを推奨します。

「ポスト量子暗号技術」の詳細については、ポスト量子暗号のセキュリティに関するブログ記事をご覧ください。

目次

SSH におけるポスト量子ハイブリッドキー交換について

Transfer Familyは、従来の「楕円曲線ディフィー・ ヘルマン (ECDH)」鍵交換アルゴリズムと CRYSTALS「Kyber」の両方を使用するポスト量子ハイブリッドキー交換暗号スイートをサポートしています。Kyber は、「米国国立標準技術研究所 (NIST)」がポスト量子鍵合意アルゴリズムとして最初の標準ポスト量子鍵合意アルゴリズムとして指定した、ポスト量子公開鍵暗号化および鍵確立アルゴリズムです。

クライアントとサーバーは依然として ECDH 鍵交換を行います。さらに、サーバーはポスト量子共有シークレットをクライアントのポストクォンタム KEM 公開鍵にカプセル化します。この公開鍵は、クライアントの SSH 鍵交換メッセージでアドバタイズされます。この戦略では、従来の鍵交換

の高い保証と、提案されているポスト量子鍵交換のセキュリティを組み合わせて、ECDH またはポスト量子共有秘密が破られない限りハンドシェイクが確実に保護されるようにします。

TransferFamilyにおけるポスト量子ハイブリッド鍵確立の仕組み

AWS は最近、 での SFTP ファイル転送におけるポスト量子キー交換のサポートを発表しました AWS Transfer Family。Transfer Family は、SFTP やその他のプロトコルを使用して、 AWS ストレージサービスへの business-to-business ファイル転送を安全にスケーリングします。SFTP は、SSH 上で動作するファイル転送プロトコル (FTP) のより安全なバージョンです。TransTransfer Family ilyのポスト量子キー交換サポートにより、SFTPを介したデータ転送のセキュリティレベルが引き上げられます。

Transfer Familyのポスト量子ハイブリッドキー交換SFTPサポートには、ポスト量子アルゴリズムのKyber-512、Kyber-768、Kyber-1024と、P256、P384、P521、またはCurve25519曲線上のECDHを組み合わせることが含まれます。「ポスト・クォンタム・ハイブリッド SSH 鍵交換ドラフト」では、対応する SSH 鍵交換方法として以下のものが規定されています。

- `ecdh-nistp256-kyber-512r3-sha256-d00@openquantumsafe.org`
- `ecdh-nistp384-kyber-768r3-sha384-d00@openquantumsafe.org`
- `ecdh-nistp521-kyber-1024r3-sha512-d00@openquantumsafe.org`
- `x25519-kyber-512r3-sha256-d00@amazon.com`

> **Note**
>
> これらの新しい鍵交換方法は、草案が標準化に向けて進展したり、NIST が Kyber アルゴリズムを批准したりするにつれて、変更される可能性があります。

Kyber を使用する理由

AWS は、標準化された相互運用可能なアルゴリズムのサポートに取り組んでいます。Kyber は、「NIST ポスト量子暗号プロジェクト」によって標準化対象として選ばれた最初のポスト量子暗号化アルゴリズムです。一部の標準本文は、すでに Kyber をプロトコルに統合しています。 AWS は、一部の AWS API エンドポイントで TLS の Kyber を既にサポートしています。

このコミットメントの一環として、 AWS は Kyber と SSH 用の P256 のような NIST 承認の曲線を組み合わせたポスト量子暗号のドラフト提案を IETF に送信しました。お客様のセキュリティを強化

するために、SFTP と SSH でのポスト量子キー交換の AWS 実装はそのドラフトに従います。私たちの提案がIETFに採用され、標準になるまで、今後の更新をサポートする予定です。

新しい鍵交換方式 (セクション TransferFamilyにおけるポスト量子ハイブリッド鍵確立の仕組み) は、草案が標準化に向けて進展したり、NIST が Kyber アルゴリズムを承認したりする際に変更されるかもしれません。

> **Note**
>
> ポスト量子アルゴリズムのサポートは、現在、 AWS KMS （「 でのハイブリッドポスト量子 TLS の使用」を参照 ）、および API エンドポイントの TLS でのポスト量子ハイブリッド AWS KMSキー交換で利用できます。AWS Certificate Manager AWS Secrets Manager

ポスト量子ハイブリッドSSH鍵交換と暗号要件 (FIPS 140)

FIPS コンプライアンスを必要とするお客様向けに、Transfer Family は FIPS 140 認定のオープンソース暗号化ライブラリ AWS-LC を使用して SSH で AWS FIPS 承認の暗号化を提供します。Transfer Family の TransferSecurityPolicy-PQ-SSH-FIPS-Experimental-2023-04 でサポートされているポスト量子ハイブリッドキー交換方法は、NIST の SP 800-56Cr2 (セクション 2) に従って FIPS が承認されています。ドイツ連邦情報セキュリティ局 (BSI) とフランスの国家情報システムセキュリティ庁 (ANSSI) も、このようなポスト量子ハイブリッドキー交換方法を推奨しています。

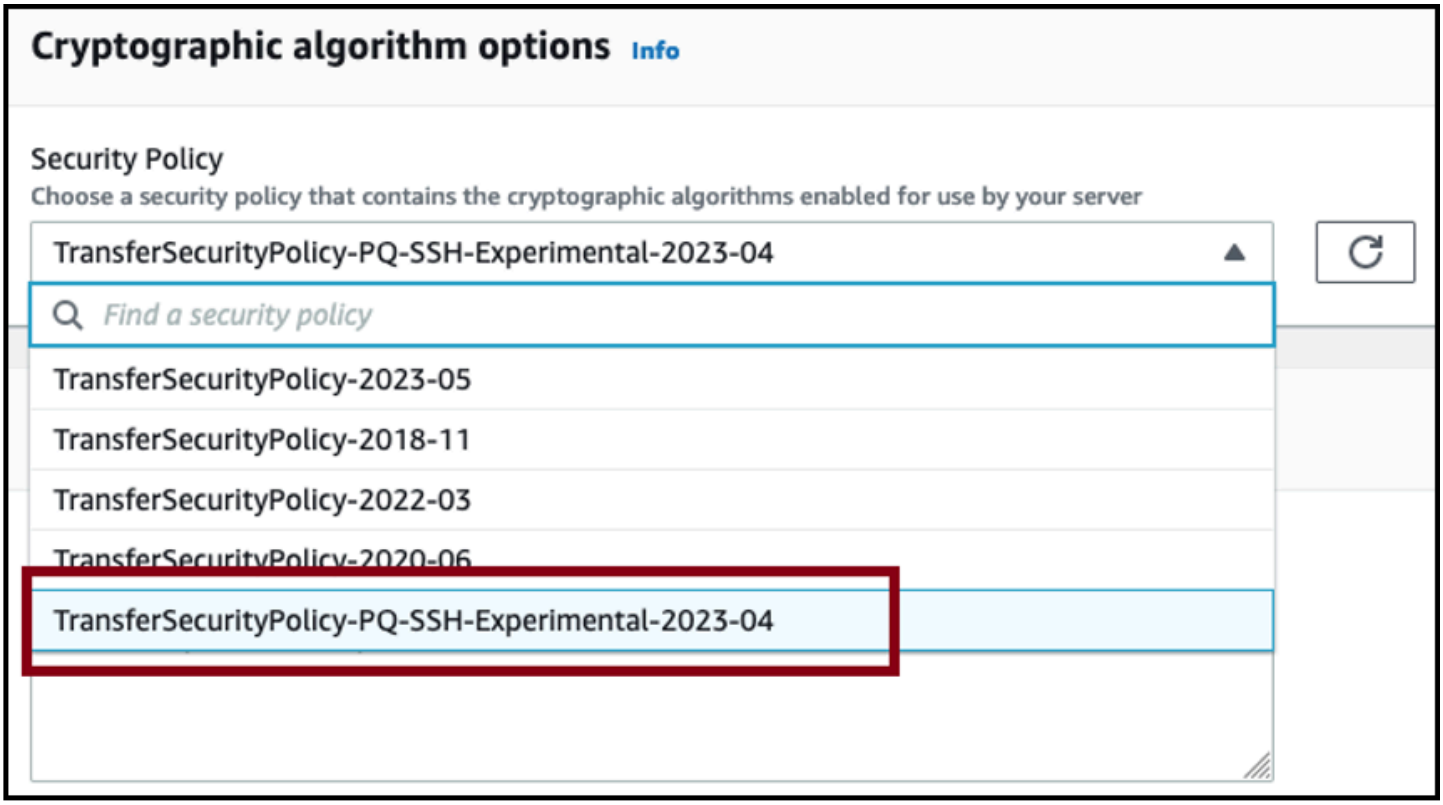
Transfer Familyにおけるポスト量子ハイブリッドキー交換のテスト

このセクションでは、ポスト量子ハイブリッドキー交換をテストする手順について説明します。

1. SFTP エンドポイントでポスト量子ハイブリッドキー交換を有効にします。.
2. 前述の仕様草案のガイダンスに従って、ポスト量子ハイブリッドキー交換をサポートする SFTP クライアント (ポスト量子ハイブリッドキー交換をサポートする SFTP クライアントをセットアップします。 など) を使用してください。
3. Transfer Family サーバーを使用してファイルを転送します。
4. SFTP でのポスト量子ハイブリッドキー交換の確認.

SFTP エンドポイントでポスト量子ハイブリッドキー交換を有効にします。

SSHポリシーは、Transfer Familyで新しいSFTPサーバーエンドポイントを作成するとき、または既存のSFTPエンドポイントの暗号化アルゴリズムオプションを編集するときに選択できます。次のスナップショットは、SSH ポリシーを更新する AWS Management Console の例です。



ポスト量子キー交換をサポートする SSH ポリシー名は、TransferSecurityPolicy-PQ-SSH-Experimental-2023-04および TransferSecurityPolicy-PQ-SSH-FIPS-Experimental-2023-04です。Transfer Familyポリシーの詳細については、AWS Transfer Family サーバーのセキュリティポリシー を参照してください。

ポスト量子ハイブリッドキー交換をサポートする SFTP クライアントをセットアップします。

SFTP Transfer Family エンドポイントで正しいポストクォンタム SSH ポリシーを選択したら、Transfer Family でポストクォンタム SFTP を試すことができます。前述のドラフト仕様のガイダンスに従うことで、ポスト量子ハイブリッドキー交換をサポートする SFTP クライアント (OQS OpenSSH など) を使用できます。

OQS OpenSSH は OpenSSH のオープンソースフォークで、liboqs を使用して SSH に量子安全暗号化を追加します。liboqs は、耐量子暗号アルゴリズムを実装したオープンソースの C ライブラリです。OQS OpenSSH と liboqs はオープンクアンタムセーフ (OQS) プロジェクトの一部です。

Transfer Family SFTP で OQS OpenSSH を使用してポスト量子ハイブリッドキー交換をテストするには、プロジェクトの「README」で説明されているように OQS OpenSSH をビルドする必要があります。OQS OpenSSH をビルドしたら、以下のコマンドに示すように、ポスト量子ハイブリッドキー交換方法を使用して、サンプル SFTP クライアントを実行して SFTP エンドポイント (例:s-1111aaaa2222bbbb3.server.transfer.us-west-2.amazonaws.com) に接続できます。

```
./sftp -S ./ssh -v -o \
   KexAlgorithms=ecdh-nistp384-kyber-768r3-sha384-d00@openquantumsafe.org \
   -i username_private_key_PEM_file \
   username@server-id.server.transfer.region-id.amazonaws.com
```

先のコマンドで、以下の項目を自分の情報に置き換える：

- 「*#####_########_PEM_####*」を SFTP ユーザーのプライベートキー PEM エンコードファイルに置き換えます。
- *username* はインスタンスのユーザー名に置き換えます。
- 「*server-id*」をTransfer Family サーバー ID に置き換えます
- 「*region-id*」をトランスポート・ファミリ・サーバが存在する実際の地域に置き換えます

SFTP でのポスト量子ハイブリッドキー交換の確認

SFTP から Transfer Family への SSH 接続中にポスト量子ハイブリッドキー交換が使用されたことを確認するには、クライアント出力を確認します。オプションで、パケットキャプチャプログラムを使用できます。Open Quantum Safe OpenSSH クライアントを使用する場合、出力は以下のようになります (簡潔にするために無関係な情報は省略)。

```
$./sftp -S ./ssh -v -o KexAlgorithms=ecdh-nistp384-kyber-768r3-sha384-
d00@openquantumsafe.org -
i username_private_key_PEM_file username@s-1111aaaa2222bbbb3.server.transfer.us-
west-2.amazonaws.com
OpenSSH_8.9-2022-01_p1, Open Quantum Safe 2022-08, OpenSSL 3.0.2 15 Mar 2022
debug1: Reading configuration data /home/lab/openssh/oqs-test/tmp/ssh_config
debug1: Authenticator provider $SSH_SK_PROVIDER did not resolve; disabling
```

```
debug1: Connecting to s-1111aaaa2222bbbb3.server.transfer.us-west-2.amazonaws.com
 [xx.yy.zz..12] port 22.
debug1: Connection established.
[...]
debug1: Local version string SSH-2.0-OpenSSH_8.9-2022-01_
debug1: Remote protocol version 2.0, remote software version AWS_SFTP_1.1
debug1: compat_banner: no match: AWS_SFTP_1.1
debug1: Authenticating to s-1111aaaa2222bbbb3.server.transfer.us-
west-2.amazonaws.com:22 as 'username'
debug1: load_hostkeys: fopen /home/lab/.ssh/known_hosts2: No such file or directory
[...]
debug1: SSH2_MSG_KEXINIT sent
debug1: SSH2_MSG_KEXINIT received
debug1: kex: algorithm: ecdh-nistp384-kyber-768r3-sha384-d00@openquantumsafe.org
debug1: kex: host key algorithm: ssh-ed25519
debug1: kex: server->client cipher: aes192-ctr MAC: hmac-sha2-256-etm@openssh.com
 compression: none
debug1: kex: client->server cipher: aes192-ctr MAC: hmac-sha2-256-etm@openssh.com
 compression: none
debug1: expecting SSH2_MSG_KEX_ECDH_REPLY
debug1: SSH2_MSG_KEX_ECDH_REPLY received
debug1: Server host key: ssh-ed25519 SHA256:e3b0c44298fc1c149afbf4c8996fb92427ae41e4649
[...]
debug1: rekey out after 4294967296 blocks
debug1: SSH2_MSG_NEWKEYS sent
debug1: expecting SSH2_MSG_NEWKEYS
debug1: SSH2_MSG_NEWKEYS received
debug1: rekey in after 4294967296 blocks
[...]
Authenticated to AWS.Tranfer.PQ.SFTP.test-endpoint.aws.com ([xx.yy.zz..12]:22) using
 "publickey".s
debug1: channel 0: new [client-session]
[...]
Connected to s-1111aaaa2222bbbb3.server.transfer.us-west-2.amazonaws.com.
sftp>
```

この出力は、ポスト・クアンタム・ハイブリッド ecdh-nistp384-kyber-768r3-sha384-d00@openquantumsafe.org メソッドを使用してクライアント・ネゴシエーションが行われ、SFTP セッションが正常に確立されたことを示しています。

でのデータ保護 AWS Transfer Family

責任 AWS 共有モデル、 AWS Transfer Family (Transfer Family) のデータ保護に適用されます。このモデルで説明されているように、 AWS はすべての AWS クラウドを実行するグローバルインフラストラクチャを保護する責任があります。お客様は、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があります。このコンテンツには、使用する AWS サービスのセキュリティ設定および管理タスクが含まれます。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」ブログを参照してください。

データ保護の目的で、 AWS アカウントの認証情報を保護し、 AWS Identity and Access Management (IAM) を使用して個々のユーザーアカウントを設定することをお勧めします。この方法により、それぞれのジョブを遂行するために必要な許可のみを各ユーザーに付与できます。また、次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 がサポートされています。
- を使用して API とユーザーアクティビティの口グ記録を設定します AWS CloudTrail。
- AWS 暗号化ソリューションと、 AWS サービス内のすべてのデフォルトのセキュリティコントロールを使用します。
- Amazon Macie などのアドバンストマネージドセキュリティサービスを使用します。これは、Amazon S3 に保存されている個人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ースまたは API を使用して AWS 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、「Federal information processing standard (FIPS) 140-2」(連邦情報処理規格 (FIPS) 140-2) を参照してください。

顧客のアカウント番号などの機密の識別情報は、[Name] (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、API、または SDK を使用して Transfer Family 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 Transfer Family または他のサービスに入力したデータはすべて、診断口グの内容として取得される可能性があります。外部サーバーへの URL を指定するときは、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 に含めないでください。

対照的に、Transfer Family サーバーへのアップロードおよびダウンロード操作のデータは完全にプライベートなものとして扱われ、SFTPまたはFTPS接続などの暗号化されたチャネルの外には存在しません。このデータには、権限のある人だけがアクセスできます。

トピック

- Amazon S3 でのデータ暗号化
- Transfer Family での SSH キーと PGP キーの管理

Amazon S3 でのデータ暗号化

AWS Transfer Family は、Amazon S3 バケットに設定したデフォルトの暗号化オプションを使用してデータを暗号化します。バケットで暗号化を有効にすると、バケットに保存されるすべてのオブジェクトが暗号化されます。オブジェクトは、Amazon S3 マネージドキー (SSE-S3) または AWS Key Management Service () マネージドキー (SSE-KMS AWS KMS) によるサーバー側の暗号化を使用して暗号化されます。サーバー側の暗号化の詳細については、Amazon Simple Storage Service ユーザーガイドの「サーバー側の暗号化を使用したデータの保護」を参照してください。

次の手順は、 でデータを暗号化する方法を示しています AWS Transfer Family。

で暗号化を許可するには AWS Transfer Family

1. Amazon S3 バケットのデフォルト暗号化を有効にします。詳細については、Amazon Simple Storage Service ユーザーガイドの「S3 バケットの Amazon S3 デフォルト暗号化」を参照してください。
2. ユーザーにアタッチされている AWS Identity and Access Management (IAM) ロールポリシーを更新して、必要な AWS Key Management Service （ AWS KMS) アクセス許可を付与します。
3. ユーザーのセッションポリシーを使用している場合、セッションポリシーは必要な AWS KMS アクセス許可を付与する必要があります。

次の例は、 AWS KMS 暗号化が有効になっている Amazon S3 バケット AWS Transfer Family で を使用する場合に必要な最小限のアクセス許可を付与する IAM ポリシーを示しています。 Amazon S3 このポリシー例は、IAM ロールポリシーおよびセッションポリシー (使用している場合) の両方に含めます。

```
{
 "Sid": "Stmt1544140969635",
 "Action": [
```

```
  "kms:Decrypt",
  "kms:Encrypt",
  "kms:GenerateDataKey"
 ],
 "Effect": "Allow",
 "Resource": "arn:aws:kms:region:account-id:key/kms-key-id"
}
```

> **Note**
>
> このポリシーで指定する KMS キー ID は、ステップ 1 でデフォルト暗号化のために指定したものと同じである必要があります。
> root、またはユーザーに使用される IAM ロールは、 AWS KMS キーポリシーで許可される必要があります。 AWS KMS キーポリシーの詳細については、「 デベロッパーガイド」の AWS 「KMS でのキーポリシーの使用」を参照してください。 AWS Key Management Service

Transfer Family での SSH キーと PGP キーの管理

このセクションでは、SSHキーの生成方法やローテーション方法など、SSHキーに関する情報を見ることができます。で Transfer Family を使用してキー AWS Lambda を管理する方法の詳細については、ブログ記事「A AWS Transfer Family と によるユーザーのセルフサービスキー管理の有効化 AWS Lambda」を参照してください。

> **Note**
>
> AWS Transfer Family は RSA、ECDSA、ED25519 キーを受け入れます。

このセクションでは、プリティ・ グッド・ プライバシー (PGP) キーを生成して管理する方法についても説明します。

トピック

- ユーザーキーとサーバーキーでサポートされるアルゴリズム
- サービス管理ユーザーの SSH キーの生成
- SSH キーのローテーション
- PGPキーの生成と管理

- サポートされている PGP クライアント

ユーザーキーとサーバーキーでサポートされるアルゴリズム

AWS Transfer Family内のユーザーキーペアとサーバーキーペアでは、以下のキーアルゴリズムがサポートされています。

> **Note**
>
> ワークフローで PGP 復号化に使用するアルゴリズムについては、「PGP キーペアでサポートされるアルゴリズム」を参照してください。

- ED25519 の場合:`ssh-ed25519`
- RSA の場合:
  - `rsa-sha2-256`
  - `rsa-sha2-512`
- ECDSA の場合:
  - `ecdsa-sha2-nistp256`
  - `ecdsa-sha2-nistp384`
  - `ecdsa-sha2-nistp521`

> **Note**
>
> 古いセキュリティポリシーはSHA1 を使用した `ssh-rsa` をサポートしています。詳細については、「暗号アルゴリズム」を参照してください。

サービス管理ユーザーの SSH キーの生成

サーバーをセットアップする際、サービスで管理された認証方法でユーザーを認証するよう設定できます。この方法では、ユーザー名と SSH キーは、サービスに保存されます。ユーザーのパブリック SSH キーは、ユーザーのプロパティとしてサーバーにアップロードされます。このキーは、キーベースの標準認証プロセスの一部としてサーバーによって使用されます。各ユーザーに対し、個々のサーバーに複数のパブリック SSH キーを登録できます。ユーザ 1 人あたりに保存できるキー数の制

限については、「Amazon Web Services 全般のリファレンス」の「AWS Transfer Family エンドポイントとクォータ」を参照してください。

サービスマネージド認証方法の代わりに、カスタム ID プロバイダー、または を使用してユーザーを認証できます AWS Directory Service for Microsoft Active Directory。詳細については、カスタム ID プロバイダーの使用またはAWS Directory Service ID プロバイダーの使用を参照してください。

サーバーがユーザー認証に使用できる方法は 1 つのみ (サービスマネージドまたはカスタム ID プロバイダー) であり、サーバーの作成後にそのメソッドを変更することはできません。

トピック

- macOS、Linux、または UNIX で SSH キーを作成する
- Microsoft Windows での SSH キーの作成
- SSH2 公開鍵を PEM 形式に変換します。

macOS、Linux、または UNIX で SSH キーを作成する

macOS、Linux、または UNIX のオペレーティングシステムでは、ssh-keygen コマンドを使用して、キーペトと呼ばれる SSH パブリックキーと SSH プライベートキーを作成します。

macOS、Linux、または UNIX のオペレーティングシステムで SSH キーを作成するには

1. macOS、Linux、または UNIX でコマンドターミナルを開きます。
2. AWS Transfer Family は、RSA、ECDSA、ED25519-formattedキーを受け入れます。生成するキーペアのタイプに基づいて適切なコマンドを選択してください。

> **Note**
>
> 以下の例では、パスフレーズは指定していません。この場合、ツールはパスフレーズの入力を要求し、確認のためもう一度入力するように求めます。パスフレーズを作成すると、秘密鍵の保護が強化され、システム全体のセキュリティも向上する可能性があります。パスフレーズは復元できません。忘れた場合は、新しいキーを作成する必要があります。
> ただし、サーバーホスト鍵を生成する場合、Transfer Family サーバーは起動時にパスワードを要求できないため、コマンドで `-N ""` オプションを指定する (またはプロンプトが表示されたら **Enter** を 2 回押す) ことで、空のパスフレーズを「必ず」指定する必要があります。

- 4096 ビット RSA キーペアを生成する

```
ssh-keygen -t rsa -b 4096 -f key_name
```

- ECDSA 521 ビットのキーペア (ECDSA のビットサイズは 256、384、521) を生成するには:

```
ssh-keygen -t ecdsa -b 521 -f key_name
```

- ED25519キーペアを生成するには：

```
ssh-keygen -t ed25519 -f key_name
```

> **Note**
>
> *key_name* は SSH キーペアファイル名です。

ssh-keygen 出力の例を以下に示します。

```
ssh-keygen -t rsa -b 4096 -f key_name
Generating public/private rsa key pair.

Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in key_name.
Your public key has been saved in key_name.pub.
The key fingerprint is:
SHA256:8tDDwPmanTFcEzjTwPGETVWOGW1nVz+gtCCE8hL7PrQ bob.amazon.com
The key's randomart image is:
+---[RSA 4096]----+
|    . ....E      |
| .   = ...       |
|. . . = ..o      |
| . o +  oo =     |
|  + =  .S.= *    |
| . o o ..B + o   |
|     .o.+.* .    |
|     =o*+*.      |
|    ..*o*+.      |
```

```
+----[SHA256]-----+
```

> **Note**
>
> 上記の ssh-keygen コマンドを実行すると、現在のディレクトリにパブリックキーとプライベートキーが作成されます。

これで、SSH キーペアを使用する準備が整いました。ステップ 3 と 4 に従って、サービスマネージドユーザーの SSH パブリックキーを保存します。これらのユーザーは、Transfer Family サーバーエンドポイントでファイルを転送するときに キーを使用します。

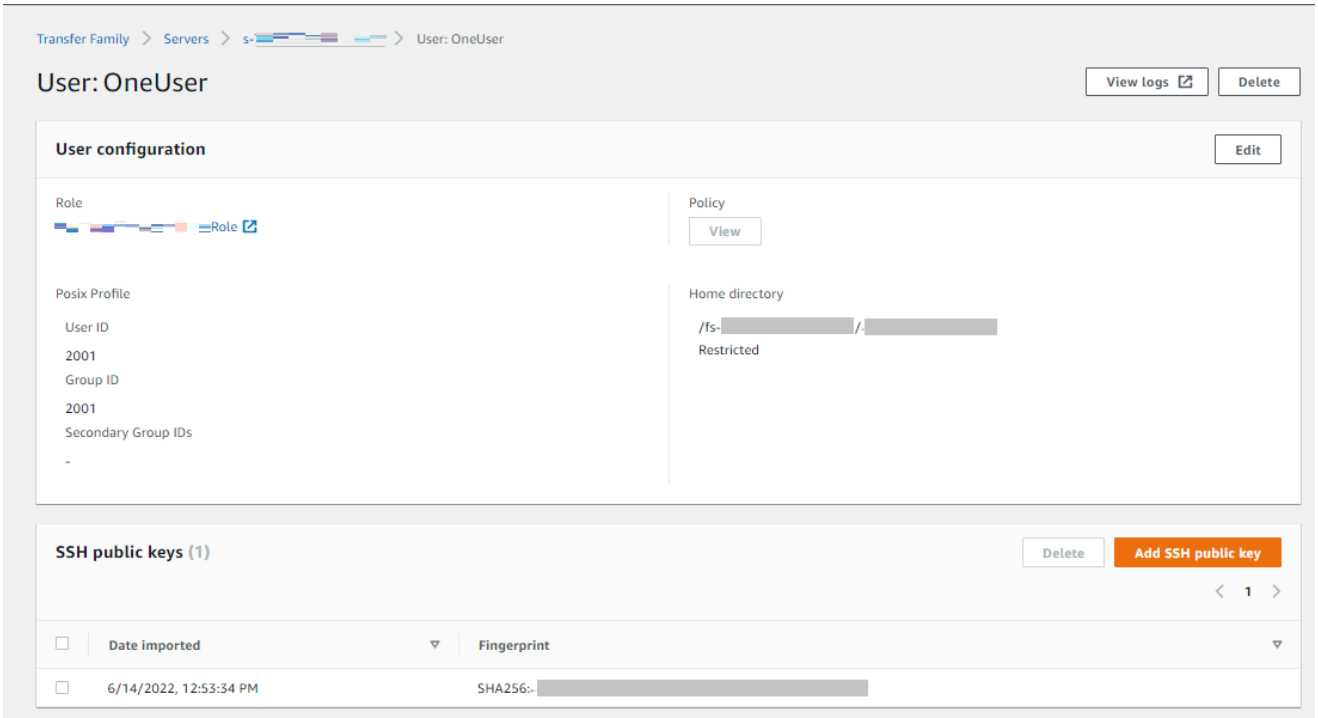
3. *key_name*.pub.ファイルを見つけて開きます。
4. テキストをコピーして、サービス管理対象ユーザーの「SSH 公開鍵」に貼り付けます。

   a. https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開き、ナビゲーションペインからサーバーを選択します。

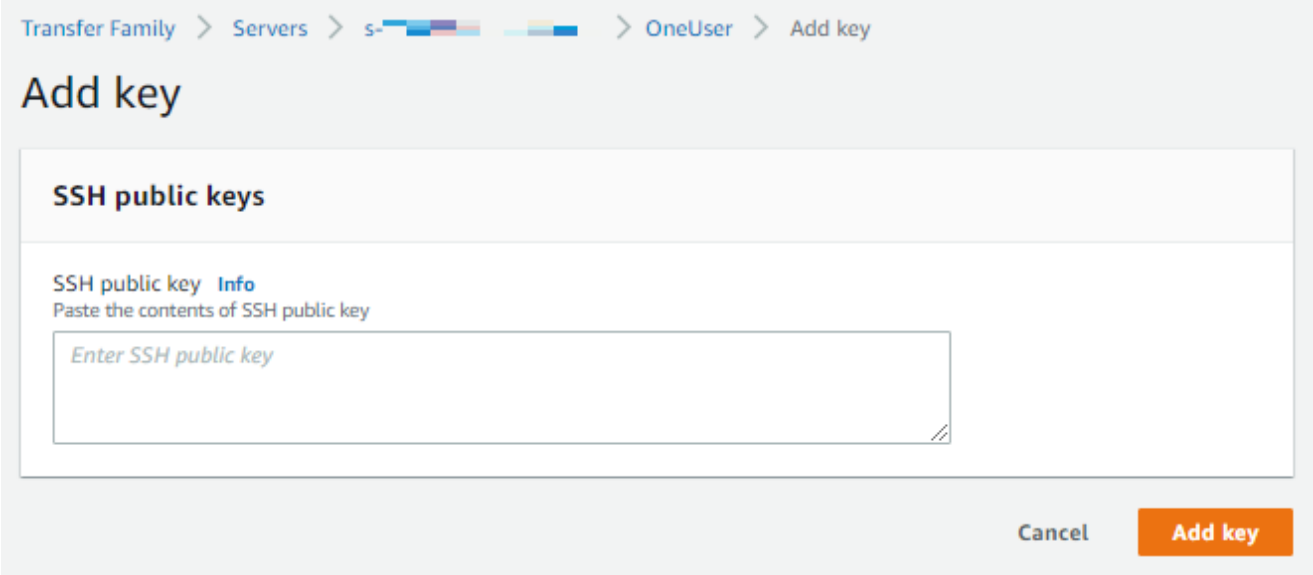
   b. 「サーバー」ページで、更新するユーザーを含むサーバーの「サーバー ID」を選択します。

   c. パブリックキーを追加するユーザーを選択します。

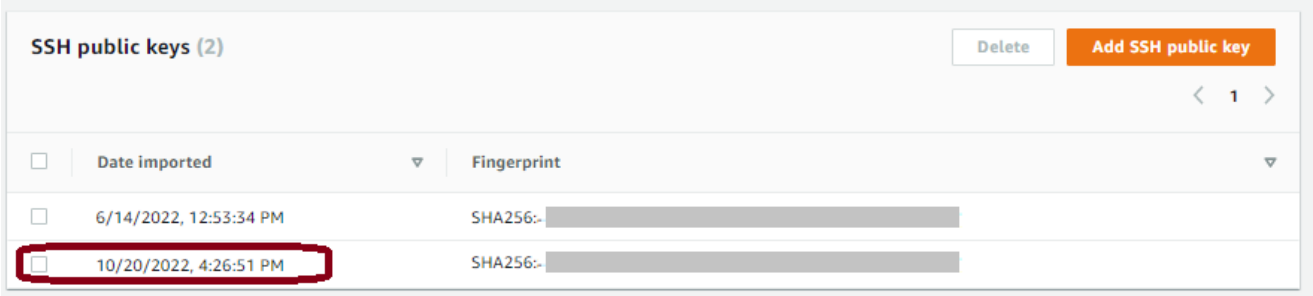
   d. 「SSH 公開鍵」ペインで、「SSH 公開鍵を追加」を選択します。



e. 生成した公開鍵のテキストを SSH 公開鍵テキストボックスに貼り付け、「鍵の追加」を選択します。



新しい鍵は SSH 公開鍵ペインに一覧表示されます。



Microsoft Windows での SSH キーの作成

Windows で使用される SSH キーペア形式は、わずかに異なります。パブリックキーは PUB 形式、プライベートキーは PPK 形式である必要があります。Windows では、PuTTYgen を使用すれば正しい形式の SSH キーペアが作成できます。PuTTYgen を使用して、ssh-keygen で生成されたプライベートキーを .ppk ファイルに変換することもできます。

> **Note**
>
> .ppk 形式でないプライベート・キー・ファイルを WinSCP に提示すると、WinSCP クライアントはキーを .ppk 形式に変換します。

Windows で PuTTYgen を使用して SSH キーを作成する方法のチュートリアルは、「SSH.com ウェブサイト」 にあります。

SSH2 公開鍵を PEM 形式に変換します。

AWS Transfer Family は PEM 形式のパブリックキーのみを受け入れます。SSH2 公開鍵がある場合は、それを変換する必要があります。SSH2 公開鍵は以下の形式になります。

```
---- BEGIN SSH2 PUBLIC KEY ----
Comment: "rsa-key-20160402"
AAAAB3NzaC1yc2EAAAABJQAAAgEAiL0jjDdFqK/kYThqKt7THrjABTPWvXmB3URI
:
:
---- END SSH2 PUBLIC KEY ----
```

PEM 公開鍵は以下の形式になります。

```
ssh-rsa AAAAB3NzaC1yc2EAAAABJQAAA...
```

次のコマンドを実行して、SSH2 形式の公開鍵を PEM 形式の公開鍵に変換します。「*ssh2-##*」は SSH2 キーの名前に置き換え、「*PEM ##*」は PEM キーの名前に置き換えてください。

```
ssh-keygen -i -f ssh2-key.pub > PEM-key.pub
```

SSH キーのローテーション

セキュリティ上、ベストプラクティスとして SSH キーのローテーションをお勧めします。通常、このローテーションは、セキュリティポリシーの一部として指定され、自動化した形で実装されます。セキュリティのレベルに応じて、機密性の高い通信では、SSH キーペアが 1 回しか使用されない場合もあります。これにより、キーを保存することのリスクがなくなります。ただし一般的には、SSH 認証情報を一定期間にわたって保存し、その期間をユーザーに過度の負荷がかからないような長さに設定するのが普通です。標準的な設定は 3 か月です。

SSH キーのローテーションを実行するには、2 つの方法があります。

- コンソールで新しい SSH パブリックキーをアップロードすると既存の SSH 公開キーを削除できます。
- API を使用すると、 DeleteSshPublicKey API を使用してユーザーの Secure Shell (SSH) パブリックキーを削除し、 ImportSshPublicKey API を使用してユーザーのアカウントに新しい Secure Shell (SSH) パブリックキーを追加することで、既存のユーザーを更新できます。

Console

コンソールでキーローテーションを実行するには

1. https://console.aws.amazon.com/transfer/ で AWS Transfer Family コンソールを開きます。
2. [Servers] (サーバー) ページに移動します。
3. [Server ID] (サーバー ID) 列で ID を選択すると、[Server Configuration] (サーバーの構成) ページが表示されます。
4. [Users] (ユーザー) の下で、SSH パブリックキーをローテーションするユーザーのチェックボックスをオンにし、[Actions] (アクション) を選択してから [Add key] (キーの追加) を選択すると [Add key] (キーの追加) ページが表示されます。

   または

   ユーザー名を選択してユーザーの詳細ページを表示し、SSHパブリックキーの追加を選択してキーの追加ページを表示する。
5. 新しい SSH パブリックキーを入力し、[Add key] (キーを追加する) を選択します。

   > **Important**
   > SSH 公開鍵の形式は、生成した鍵のタイプによって異なります。
   >
   > - RSA キーの場合、形式は `ssh-rsa` *`string`* です。
   > - ED25519 キーの場合、形式は `ssh-ed25519` *`string`* です。
   > - ECDSA キーの場合、生成したキーのサイズに応じてキーは`ecdsa-sha2-nistp256`、`ecdsa-sha2-nistp384`、または `ecdsa-sha2-nistp521` で始まります。他のキータイプと同様に、最初の文字列の後には *`string`* が続きます。

   [User details] (ユーザーの詳細) ページに戻ると、先ほど入力した SSH パブリックキーが [SSH public key] (SSH パブリックキー) セクションに表示されます。
6. 削除したい古いキーのチェックボックスをオンにしてから [Delete] (削除) を選択します。
7. `delete` という語を入力して削除オペレーションを確認してから [Delete] (削除) を選択します。

API

API を使用してキーローテーションを実行するには

1. macOS、Linux、または UNIX でコマンドターミナルを開きます。
2. 次のコマンドを入力することで、削除したい SSH キーを取得します。このコマンドを使用するには、*serverID* をTransfer Familyサーバーのサーバー ID に、*username* をユーザー名に置き換えます。

```
aws transfer describe-user --server-id='serverID' --user-name='username'
```

このコマンドは、ユーザーについての詳細を返します。"SshPublicKeyId": フィールドの内容をコピーします。この手順の後半で、この値を入力する必要があります。

```
"SshPublicKeys": [ { "SshPublicKeyBody": "public-key", "SshPublicKeyId":
 "keyID",
   "DateImported": 1621969331.072 } ],
```

3. 次に、ユーザー用の新しい SSH キーをインポートします。 プロンプトで次のコマンドを入力します。このコマンドを使用するには、*serverID* をTransfer Familyサーバーのサーバー ID に、*username* をユーザー名に、*public-key* を新しい公開鍵のフィンガープリントに置き換えます。

```
aws transfer import-ssh-public-key --server-id='serverID' --user-name='username'
   --ssh-public-key-body='public-key'
```

コマンドが成功した場合、出力は返りません。

4. 最後に、次のコマンドを実行して、古いキーを削除します。このコマンドを使用するには、*serverID* をTransfer Familyサーバーのサーバー ID に、*username* をユーザー名に、*keyID-from-step-2* をこの手順のステップ2でコピーしたキーID値に置き換えます。

```
aws transfer delete-ssh-public-key --server-id='serverID' --user-name='username'
   --ssh-public-key-id='keyID-from-step-2'
```

5. (オプション) 古いキーがもう存在しないことを確認するには、ステップ 2 を繰り返します。

PGPキーの生成と管理

Transfer Family がワークフローで処理するファイルで、Pretty Good Privacy (PGP) 復号化を使用できます。ワークフローステップで復号を使用するには、PGP キーを指定します。

AWS ストレージブログには、Transfer Family Managed ワークフロー、PGP および を使用したファイルの暗号化と復号化を使用してコードを記述せずにファイルを復号する方法 AWS Transfer Familyを説明する投稿があります。

PGP キーを生成する

PGP キーの生成に使用する演算子は、オペレーティングシステムと、使用しているキー生成ソフトウェアのバージョンによって異なります。

Linux または Unix を使用している場合は、パッケージインストーラーを使用して gpg をインストールします。お使いの Linux ディストリビューションに応じて、以下のコマンドのいずれかが動作するはずです。

```
sudo yum install gnupg
```

```
sudo apt-get install gnupg
```

Windows または macOS の場合は、「https://gnupg.org/download/」から必要なものをダウンロードできます。

PGP キージェネレータソフトウェアをインストールしたら、「`gpg --full-gen-key`」または「`gpg --gen-key`」コマンドを実行してキーペアを生成します。

> **Note**
>
> GnuPG バージョン 2.3.0 以降を使用している場合は、`gpg --full-gen-key` を実行する必要があります。作成する鍵の種類を求められたら、RSA または ECC を選択します。ただし、ECC を選択した場合は、必ず楕円曲線用に NIST または BrainPool を選択してください。Curve 25519 を「選びません」。

PGP キーペアでサポートされるアルゴリズム

PGP キーペアでは、次のアルゴリズムがサポートされています。

- RSA
- エルガマル
- ECC：
  - NIST
  - BrainPool

> **Note**
>
> cCurve25519 キーはサポートされていません。

役立つ**gpg** サブコマンド

以下は gpg に役立つサブコマンドです。

- `gpg --help` — このコマンドには、使用可能なオプションが一覧表示され、例もいくつか含まれている場合があります。
- `gpg --list-keys` – このコマンドは、作成したすべてのキーペアの詳細を一覧表示します。
- `gpg --fingerprint` – このコマンドは、各キーのフィンガープリントを含む、すべてのキーペアの詳細を一覧表示します。
- `gpg --export -a` *`user-name`* — このコマンドは、*`user-name`*キーの生成時に使用されたキーの公開キー部分をエクスポートします。

キーペアを管理する

PGP キーを管理するには、 を使用します AWS Secrets Manager。

> **Note**
>
> シークレットネームには、Transfer Familyのサーバー IDが含まれます。つまり、PGP キー情報を AWS Secrets Manager保存する「前」に、サーバーを特定または作成しておく必要があります。

1つのキーとパスフレーズをすべてのユーザーに使いたい場合は、PGPキーブロック情報を秘密名 `aws/transfer/`*`server-id`*`/@pgp-default`（ *`server-id`* はTransfer Familyサーバー のID ）の

下に保存することができます。Transfer Family は、 がワークフローを実行しているユーザー*user-name*と一致するキーがない場合、このデフォルトキーを使用します。

特定のユーザーのキーを作成できます。この場合、シークレット名の形式は です。 は`aws/transfer/`*server-id*`/`*user-name*、Transfer Family サーバーのワークフローを実行しているユーザー*user-name*と一致します。

> **Note**
>
> Transfer Family サーバー 1 台につき、1 ユーザーにつき最大 3 つの PGP 秘密鍵を保存できます。

PGP 鍵を復号化に使用するように設定するには

1. 使用している GPG のバージョンに応じて、次のいずれかのコマンドを実行して、Curve 25519 暗号化アルゴリズムを使用しない PGP キーペアを生成します。

   - **GnuPG** バージョン 2.3.0 以降を使用している場合は、次のコマンドを実行します。

     ```
     gpg --full-gen-key
     ```

     RSA を選ぶこともできるし、**ECC** を選んだ場合は楕円曲線に **NIST** か **BrainPool** を選ぶこともできます。代わりに `gpg --gen-key` を実行する場合は、ECC Curve 25519 暗号化アルゴリズムを使用するキーペアを作成します。このアルゴリズムは、現在 PGP キーではサポートされていません。

   - バージョン2.3.0より前の **GnuPG** では、RSAがデフォルトの暗号化タイプであるため、次のコマンドを使用できます。

     ```
     gpg --gen-key
     ```

     > **Important**
     >
     > キー生成プロセス中に、パスフレーズと E メールアドレスを指定する必要があります。これらの値は必ず書き留めておいてください。この手順の後半で AWS Secrets Manager に鍵の詳細を入力するときに、パスフレーズを提供する必要があります。また、次のス

テップでプライベートキーをエクスポートする場合も、同じメールアドレスを指定する必要があります。

2. 以下のコマンドを実行してプライベートキーをエクスポートします。このコマンドを使うには、*private.pgp* を秘密鍵ブロックを保存するファイル名に、*marymajor@example.com* をキーペアを生成したときに使った電子メールアドレスに置き換えます。

```
gpg --output private.pgp --armor --export-secret-key marymajor@example.com
```

3. AWS Secrets Manager を使用して PGP キーを保存します。

   a. にサインイン AWS Management Console し、https://console.aws.amazon.com/secretsmanager/ で AWS Secrets Manager コンソールを開きます。

   b. 左側のナビゲーションペインで [サーバー] を選択します。

   c. [シークレット]ページで、[新しいシークレットの保存]を選択します。

   d. [シークレットタイプの選択] ページの[シークレットタイプ] で[その他のシークレットタイプ] を選択します。

   e. [キー/値のペア] セクションで、[キー/値] タブを選択します。

   - キー — **PGPPrivateKey**と入力します。

   > **Note**
   >
   > **PGPPrivateKey** 文字列は正確に入力する必要があります。文字の前や間にスペースを入れないでください。

   - 「値」 — 秘密鍵のテキストを値フィールドに貼り付けます。プライベートキーのテキストは、この手順の前半でキーをエクスポートしたときに指定したファイル (private.pgp など) にあります。キーは `-----BEGIN PGP PRIVATE KEY BLOCK-----` で始まり、`-----END PGP PRIVATE KEY BLOCK-----` で終わります。

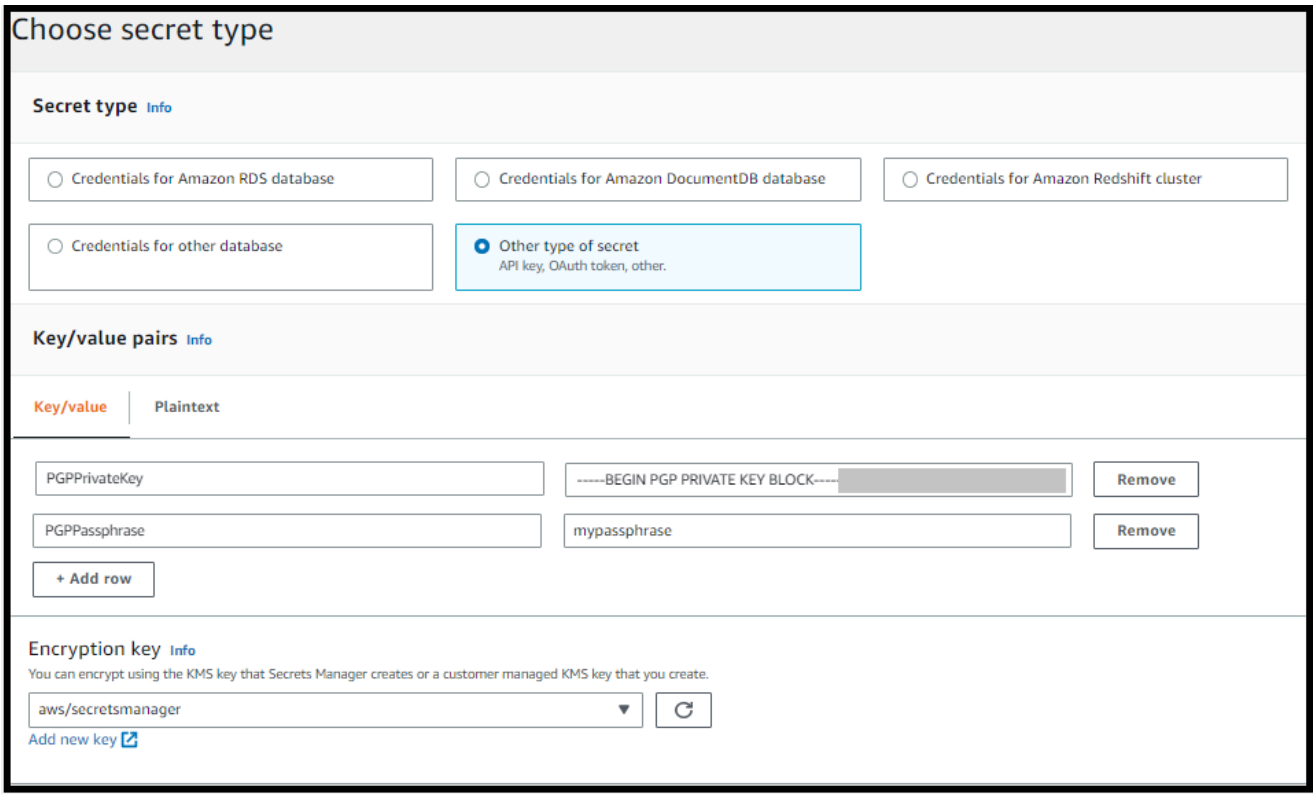
   > **Note**
   >
   > テキストブロックには秘密鍵のみが含まれ、公開鍵も含まれていないことを確認してください。

   f. 「行を追加」を選択し、「キー/値のペア」セクションで「キー/値」タブを選択します。

- キー — **PGPPassphrase**と入力します。

> **Note**
>
> **PGPPassphrase** 文字列は正確に入力する必要があります。文字の前や間にスペースを入れないでください。

- 「値」 — PGP キーペアを生成したときに使用したパスフレーズを入力します。



> **Note**
>
> 最大 3 セットのキーとパスフレーズを追加できます。2 つ目のセットを追加するには、2 つの新しい行を追加し、キーには **PGPPrivateKey2** と **PGPPassphrase2** を入力し、別のプライベートキーとパスフレーズを貼り付けます。3 番目のセットを追加するには、キー値は **PGPPrivateKey3** と **PGPPassphrase3** でなければなりません。

g. [次へ] をクリックします。

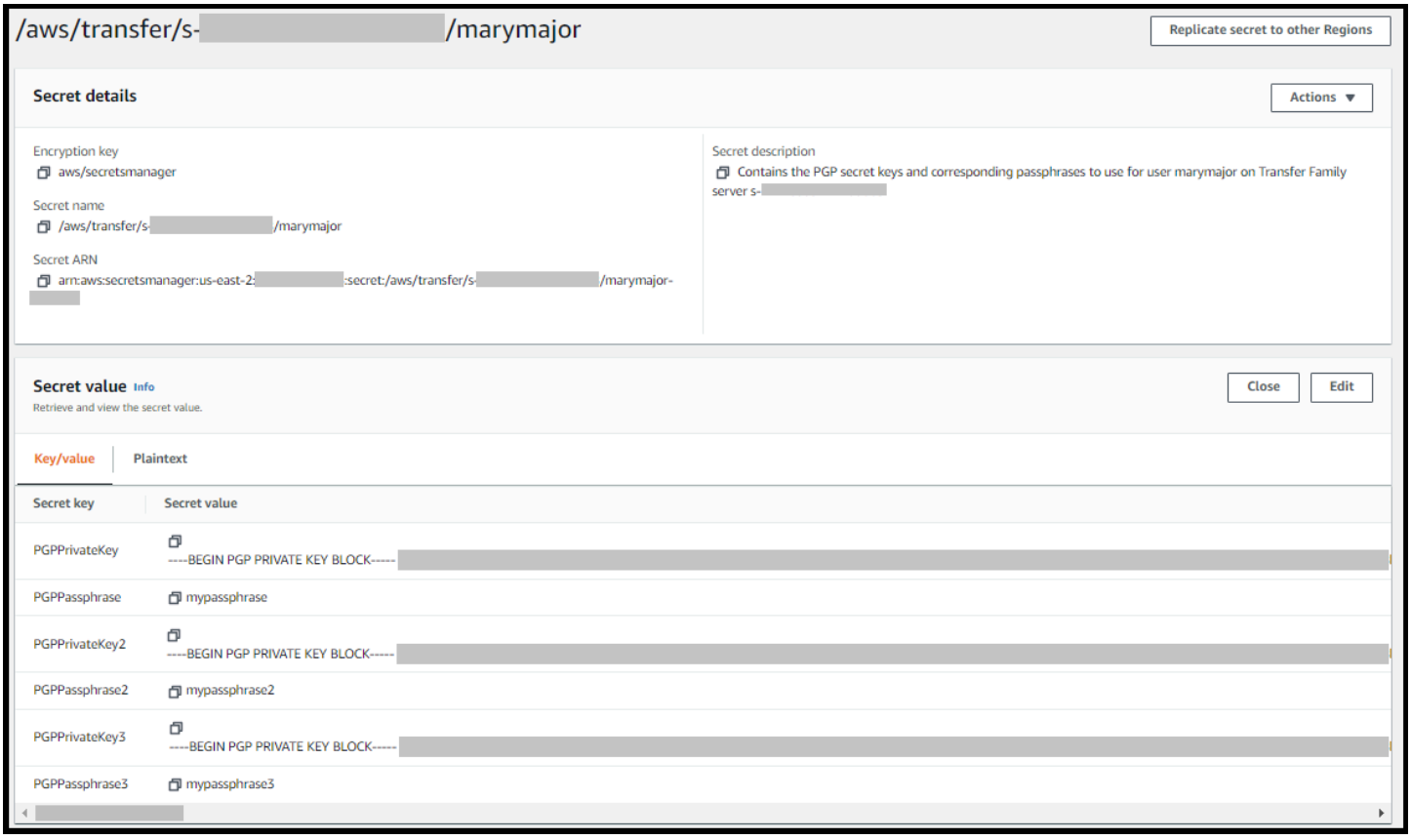
h. [シークレットの設定] ページで、シークレットの名前と説明を入力します。

- デフォルトキー、つまりTTransfer Family ilyのすべてのユーザーが使用できるキーを作成する場合は、**aws/transfer/*server-id*/@pgp-default** と入力します。*server-id*を、復号化ステップを持つワークフローを含むサーバーの ID に置き換えます。
- 特定のTransfer Family ユーザーが使用するキーを作成する場合は、**aws/transfer/*server-id*/*user-name*** と入力します。*server-id* を復号化ステップを持つワークフローを含むサーバーのIDに置き換え、*user-name* をワークフローを実行しているユーザー名に置き換えます。*user-name* は、Transfer Family サーバーが使用している ID プロバイダーに保存されます。

i. 「次へ」を選択し、「ローテーションの設定」ページのデフォルトを受け入れます。次いで、[次へ] を選択します。

j. [レビュー] ページで[ストア] を選択し、シークレットを作成して保存します。

次のスクリーンショットは、特定の Transfer Family サーバのユーザ **marymajor** の詳細を示している。この例では、3 つのキーとそれに対応するパスフレーズが表示されています。

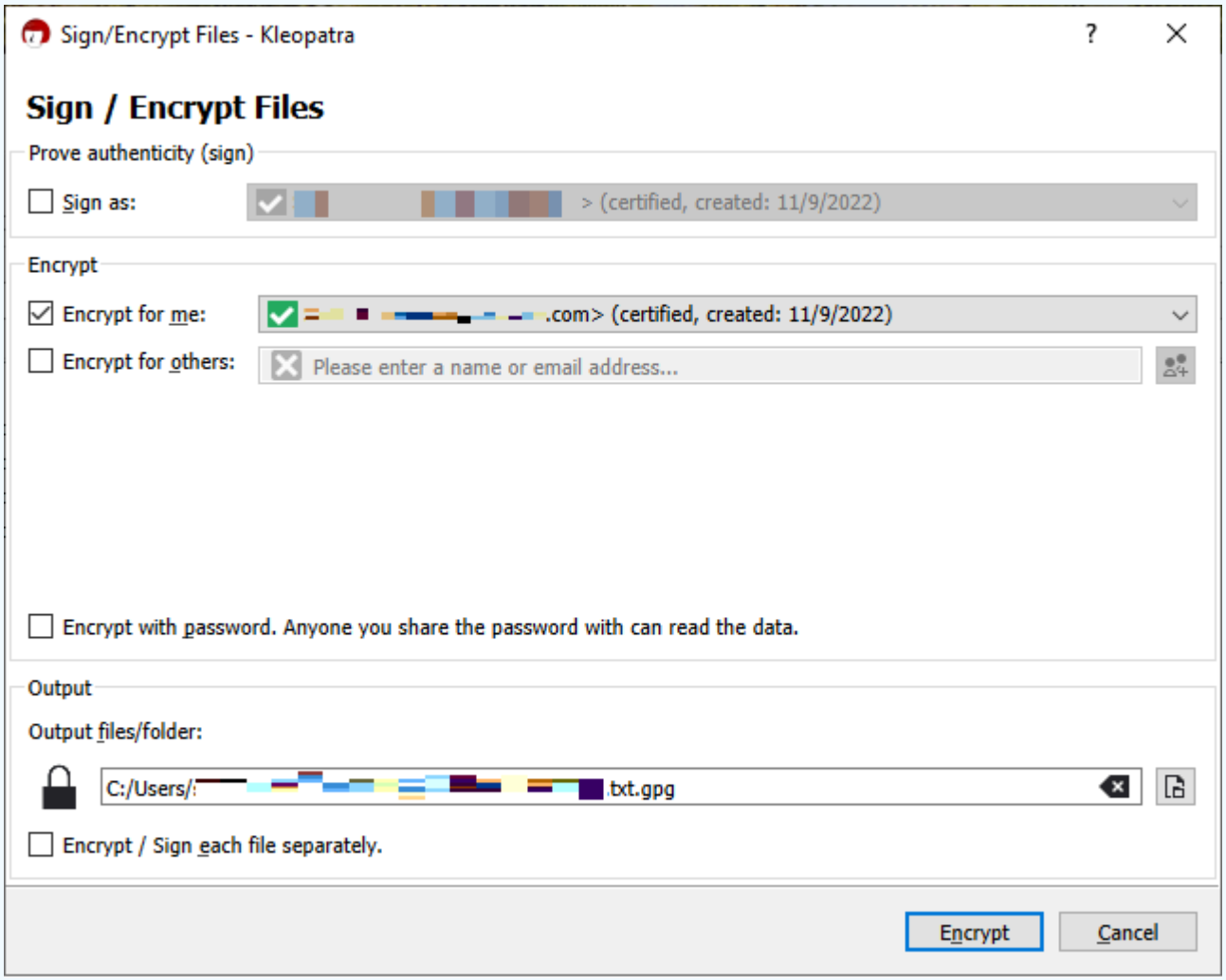


サポートされている PGP クライアント

以下のクライアントはTTransfer Family ilyでテストされており、PGPキーの生成や、ワークフローで復号化するファイルの暗号化に使用できます。

- GPG4Win + クレオパトラ。

> **Note**
>
> 「ファイルの署名/暗号化」を選択した場合は、「名前をつけて署名」の選択が解除されていることを確認してください。現在、暗号化されたファイルへの署名はサポートされていません。



暗号化されたファイルに署名し、復号ワークフローを使用して Transfer Family サーバーにアップロードしようとすると、次のエラーが表示されます。

```
Encrypted file with signed message unsupported
```

- 「GnuPG」の主要なバージョン:2.4、2.3、2.2、2.0 および 1.4。

他のPGPクライアントも同様に動作する可能性がありますが、TTransfer Family ilyでテストされたのはここに記載されているクライアントのみであることに注意してください。

の Identity and Access Management AWS Transfer Family

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 AWS Transfer Family 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
- が IAM と AWS Transfer Family 連携する方法
- AWS Transfer Family アイデンティティベースのポリシーの例
- AWS Transfer Family タグベースのポリシーの例
- AWS Transfer Family ID とアクセスのトラブルシューティング

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 AWS Transfer Family。

サービスユーザー – AWS Transfer Family サービスを使用してジョブを実行する場合、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 AWS Transfer Family 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すると、管理者から適切な権限をリクエストするのに役に立ちます。 AWS Transfer Family機能にアクセスできない場合は、「AWS Transfer Family ID 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 AWS Transfer Family 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 AWS Transfer Family。サービスユーザーがどの AWS Transfer Family 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。会社で IAM を で使用する方法の詳細については、 AWS Transfer Family「」を参照してくださいが IAM と AWS Transfer Family 連携する方法。

IAM 管理者 - 管理者は、 AWS Transfer Family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 AWS Transfer Family 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 Transfer Family アイデンティティベースのポリシーの例。

アイデンティティを使用した認証

認証とは、ID 認証情報 AWS を使用して にサインインする方法です。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 ロールを引き受けて認証（ にサインイン AWS) される必要があります。

ID ソースを介して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。 AWS IAM Identity Center（ 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、フェデレーション ID の例です。フェデレーティッド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 AWS を使用して にアクセスすると、間接的にロールを引き受けることになります。

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、「 ユーザーガイド」の「 にサインインする方法 AWS アカウントAWS サインイン 」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、リクエストに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 API AWS 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、追加セキュリティ情報の提供をリクエストされる場合もあります。例えば、 AWS 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向上させることをお勧めします。詳細については、『AWS IAM Identity Center ユーザーガイド』の「Multi-factor

authentication」(多要素認証) および『IAM ユーザーガイド』の「AWSにおける多要素認証 (MFA) の使用」を参照してください。

AWS アカウントのルートユーザー

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 ID 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は保護し、ルートユーザーでしか実行できないタスクを実行するときに使用します。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。

フェデレーティッドアイデンティティ

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、一時的な認証情報を使用して にアクセスするための ID プロバイダーとのフェデレーションの使用を要求 AWS のサービス します。

フェデレーティッド ID 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、アイデンティティセンターディレクトリ、または ID ソースを通じて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、独自の ID ソース内のユーザーとグループのセットに接続して同期して、すべての AWS アカウント とアプリケーションで使用することもできます。IAM Identity Center の詳細については、『AWS IAM Identity Center ユーザーガイド』の「What is IAM Identity Center?」(IAM Identity Center 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 AWS CLI オペレーションを呼び出すか、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。

IAM ロールと一時的な認証情報は、次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 – フェデレーティッド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。フェデレーティッド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) ポリシーをリソースに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 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、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 AWS のサービス、ダウンストリームサービス AWS のサービス へのリクエストのリクエストと組み合わせて使用します。FAS リクエストは、サービスが他の 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受け取っ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、サービスにリンクされた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 リクエストを行う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。 AWS ロールを EC2 インスタンスに割り当て、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 ユーザーではなく) IAM ロールをいつ作成したら良いのか?」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 AWS 、アイデンティティまたはリソースに関連付けられているときにアクセス許可を定義する のオブジェクトです。 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きに、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。

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。そ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、および条件をコントロールします。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、 内の複数のユーザー、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。マネージド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「マネージド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。ポリシーがアタッチされているリソースの場合、指定されたプリンシパルがそのリソースに対して実行できるアクションと条件は、ポリシーによって定義されます。リソースベースのポリシーでは、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は、IAM の AWS マネージドポリシーを使用できません。

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、 AWS WAF、および Amazon VPC は、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、一般的ではない追加の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。

- アクセス許可の境界 - アクセス許可の境界は、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。エンティティにアクセス許可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アクセス許可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、アクセス許可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、権限は無効になります。許可の境界の詳細については、「IAM ユーザーガイド」の「IAM エンティティの許可の境界」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) に対する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。Organizations 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、権限は無効になります。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与する場合にリクエストを許可するかどうか AWS を決定する方法については、IAM ユーザーガイドの「ポリシー評価ロジック」を参照してください。

が IAM と AWS Transfer Family 連携する方法

AWS Identity and Access Management (IAM) を使用して へのアクセスを管理する前に AWS Transfer Family、 で使用できる IAM 機能を理解しておく必要があります AWS Transfer Family。AWS Transfer Family およびその他の AWS のサービスが IAM 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「IAM AWS と連携する のサービス」を参照してください。

トピック

- AWS Transfer Family アイデンティティベースのポリシー
- AWS Transfer Family リソースベースのポリシー
- AWS Transfer Family タグに基づく認可
- AWS Transfer Family IAM ロール

AWS Transfer Family アイデンティティベースのポリシー

IAM アイデンティティベースのポリシーでは、許可または拒否されたアクションとリソースを指定でき、さらにアクションが許可または拒否された条件を指定できます。 AWS Transfer Family は、

特定のアクション、リソース、および条件キーをサポートします。JSON ポリシーで使用するすべての要素については、AWS Identity and Access Management ユーザーガイド の「IAM JSON ポリシーエレメントのリファレンス」を参照してください。

アクション

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、関連付けられた 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権限を付与するポリシーで使用されます。

のポリシーアクションは、アクションの前にプレフィックス AWS Transfer Family を使用しますtransfer:。たとえば、Transfer Family CreateServer API オペレーションを使用してサーバーを作成するアクセス許可を付与するには、ポリシーに transfer:CreateServer アクションを含めます。ポリシーステートメントには、Action 要素または NotAction 要素のいずれかを含める必要があります。 AWS Transfer Family は、このサービスで実行できるタスクを説明する独自の一連のアクションを定義します。

単一のステートメントに複数の アクションを指定するには、次のようにコンマで区切ります。

```
"Action": [
      "transfer:action1",
      "transfer:action2"
```

ワイルドカード *を使用して複数のアクションを指定することができます。例えば、Describe という単語で始まるすべてのアクションを指定するには、次のアクションを含めます。

```
"Action": "transfer:Describe*"
```

AWS Transfer Family 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「 で定義されるアクション AWS Transfer Family」を参照してください。

リソース

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どのような条件でアクションを実行できるかということです。

Resource JSON ポリシー要素は、アクションが適用されるオブジェクトを指定します。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、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。

オペレーションの一覧表示など、リソースレベルの許可をサポートしないアクションの場合は、ワイルドカード (*) を使用して、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します。

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

Transfer Family サーバーリソースには、次の ARN があります。

```
arn:aws:transfer:${Region}:${Account}:server/${ServerId}
```

たとえば、ステートメントで s-01234567890abcdef Transfer Family サーバーを指定するには、次の ARN を使用します。

```
"Resource": "arn:aws:transfer:us-east-1:123456789012:server/s-01234567890abcdef"
```

ARN の形式の詳細については、「サービス認証リファレンス」の「Amazon リソースネーム (ARN)」、または「IAM ユーザーガイド」の「IAM ARN」を参照してください。

特定のアカウントに属するすべてのインスタンスを指定するには、ワイルドカード *を使用します。

```
"Resource": "arn:aws:transfer:us-east-1:123456789012:server/*"
```

一部の AWS Transfer Family アクションは、IAM ポリシーで使用されるものなど、複数のリソースで実行されます。このような場合は、ワイルドカード *を使用する必要があります。

```
"Resource": "arn:aws:transfer:*:123456789012:server/*"
```

場合によっては、複数のタイプのリソースを指定する必要があります。たとえば、Transfer Family サーバーとユーザーへのアクセスを許可するポリシーを作成する場合などです。複数リソースを単一ステートメントで指定するには、ARN をカンマで区切ります。

```
"Resource": [
      "resource1",
      "resource2"
            ]
```

AWS Transfer Family リソースのリストを確認するには、「サービス認証リファレンス」の「 で定義されるリソースタイプ AWS Transfer Family」を参照してください。

条件キー

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。イコールや未満などの 条件演算子 を使用して条件式を作成することで、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 要素を指定するか、1 つの Condition 要素に複数のキーを指定すると、 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オペレーションを使用して条件 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。

AWS Transfer Family は独自の条件キーのセットを定義し、一部のグローバル条件キーの使用もサポートします。 AWS Transfer Family 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「 の条件キー AWS Transfer Family」を参照してください。

例

AWS Transfer Family 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 Transfer Family アイデンティティベースのポリシーの例。

AWS Transfer Family 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、指定されたプリンシパルが AWS Transfer Family リソースに対して実行できるアクションと条件を指定する JSON ポリシードキュメントです。Amazon S3 は、Amazon S3 ####に関するリソースベースのアクセス許可ポリシーをサポートします。リソースベースのポリシーでは、リソースごとに他の アカウントに使用許可を付与することができます。リソースベースのポリシーを使用して、 AWS サービスが Amazon S3 #### にアクセスすることを許可することもできます。

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 AWS アカウントにある場合は、プリンシパルエンティティにリソースへのアクセス許可も付与する必要があります。アクセス許可は、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、ID ベースのポリシーをさらに付与する必要はありません。詳細については、AWS Identity and Access Management 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。

Amazon S3 サービスは、####ポリシーと呼ばれるリソースベースのポリシーの 1 つのタイプのみサポートし、それが####にアタッチされます。このポリシーは、どのプリンシパルエンティティ (アカウント、ユーザー、ロール、および連携ユーザー) がオブジェクトに対してアクションを実行できるかを定義します。

例

AWS Transfer Family リソース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 Transfer Family タグベースのポリシーの例。

AWS Transfer Family タグに基づく認可

AWS Transfer Family リソースにタグをアタッチしたり、 へのリクエストでタグを渡すことができます AWS Transfer Family。タグに基づいてアクセスを管理するには、`transfer: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。タグを使用してリソースへのアクセスを制御する方法については、 AWS Transfer Family 「」を参照してくださいAWS Transfer Family タグベースのポリシーの例。

AWS Transfer Family IAM ロール

IAM ロールは、特定のアクセス許可を持つ AWS アカウント内のエンティティです。

での一時的な認証情報の使用 AWS Transfer Family

一時的な認証情報を使用して、フェデレーションでサインインする、IAM ロールを引き受ける、またはクロスアカウントロールを引き受けることができます。一時的なセキュリティ認証情報を取得するには、 AssumeRoleやGetFederationトークン などの AWS STS API オペレーションを呼び出します。

AWS Transfer Family では、一時的な認証情報の使用がサポートされています。

AWS Transfer Family アイデンティティベースのポリシーの例

デフォルトでは、IAM ユーザーおよびロールには、 AWS Transfer Family リソースを作成または変更するアクセス許可はありません。また、、 AWS Management Console AWS CLI、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、ユーザーとロールに必要な、指定されたリソースで特定の API オペレーションを実行する権限をユーザーとロールに付与する IAM ポリシーを作成する必要があります。続いて、管理者はそれらの権限が必要な IAM ユーザーまたはグループにそのポリシーをアタッチする必要があります。

これらの JSON ポリシードキュメント例を使用して IAM の ID ベースのポリシーを作成する方法については、AWS Identity and Access Management ユーザーガイドの「JSON タブでのポリシーの作成」を参照してください。

トピック

- ポリシーのベストプラクティス
- AWS Transfer Family コンソールを使用する
- ユーザーが自分の許可を表示できるようにする

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 AWS Transfer Family 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、多くの一般的なユースケースにアクセス許可を付与する AWS 管理ポリシーを使用します。これらは で使用できます AWS アカウント。ユースケースに固有の AWS カスタマー管理ポリシーを定義して、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、最小特権権限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 を介してサービスアクションが使用される場合に AWS のサービス、サービスアクションへのアクセスを許可することもできます 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。

AWS Transfer Family コンソールを使用する

AWS Transfer Family コンソールにアクセスするには、最小限のアクセス許可のセットが必要です。これらのアクセス許可により、 AWS アカウント内の AWS Transfer Family リソースの詳細を一覧表示および表示できます。最小限必要な許可よりも厳しく制限されたアイデンティティベースポリシーを作成すると、そのポリシーを添付したエンティティ (IAM 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。詳細については、AWS Identity and Access Management ユーザーガイドの「ユーザーへのアクセス許可の追加」を参照してください。

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。

ユーザーが自分の許可を表示できるようにする

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

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

```
            "Resource": "*"
        }
    ]
}
```

AWS Transfer Family タグベースのポリシーの例

タグに基づいて AWS Transfer Family リソースへのアクセスを制御する方法の例を次に示します。

タグを使用した AWS Transfer Family リソースへのアクセスのコントロール

IAM ポリシーの条件は、 AWS Transfer Family リソースへのアクセス許可を指定するために使用する構文の一部です。 AWS Transfer Family リソース (ユーザー、サーバー、ロール、その他のエンティティなど) へのアクセスは、それらのリソースのタグに基づいて制御できます。タグはキーと値のペアです。リソースのタグ付けの詳細については、「」の「 AWS リソースのタグ付け」を参照してくださいAWS 全般のリファレンス。

では AWS Transfer Family、リソースにタグを付けることができ、一部のアクションにタグを含めることができます。IAM ポリシーを作成するときに、タグ条件キーを使用して以下をコントロールできます。

- AWS Transfer Family リソースのタグに基づいて、 リソースに対してアクションを実行できるユーザー。
- アクションのリクエストで渡すことができるタグ。
- リクエストで特定のタグキーを使用できるかどうか。

タグベースのアクセス制御を使用すると、API レベルよりも細かい制御を適用できます。また、リソースベースのアクセス制御を使用するよりも動的な制御を適用できます。リクエストで指定したタグ (リクエストタグ) に基づいて、オペレーションを許可または拒否する IAM ポリシーを作成できます。また、操作しようとするリソースのタグ (リソースタグ) に基づいて IAM ポリシーを作成することもできます。一般に、リソースタグはリソースに既に存在するタグ用であり、リクエストタグはリソースにタグを追加したり、リソースからタグを削除したりするときに使用します。

タグ条件キーの完全な構文とセマンティクスについては、IAM ユーザーガイドの「リソースタグを使用した AWS リソースへのアクセスコントロール」を参照してください。API Gateway を使用した IAM ポリシーの指定の詳細については、API Gateway デベロッパーガイドの「IAM アクセス許可により API へのアクセスを制御する」を参照してください。

例 1: リソースタグに基づいてアクションを拒否する

タグに基づいてリソースについて実行されるアクションを拒否できます。次の例では、ユーザーまたはサーバーのリソースがキー stage と値 prod でタグ付けされている場合にポリシーで TagResource、UntagResource、StartServer、StopServer、DescribeServer、および DescribeUser のオペレーションが拒否されます。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": [
                "transfer:TagResource",
                "transfer:UntagResource",
                "transfer:StartServer",
                "transfer:StopServer",
                "transfer:DescribeServer",
                "transfer:DescribeUser
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/stage": "prod"
                }
            }
        }
    ]
}
```

例 2: リソースタグに基づいてアクションを許可する

タグに基づいてリソースに対するアクションの実行を許可できます。次の例では、ユーザーまたはサーバーのリソースがキー TagResource と値 UntagResource でタグ付けされている場合にポリシーで StartServer、StopServer、DescribeServer、DescribeUser、stage、および prod のオペレーションが許可されます。

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

```
            "Action": [
                "transfer:TagResource",
                "transfer:UntagResource",
                "transfer:StartServer",
                "transfer:StopServer",
                "transfer:DescribeServer",
                "transfer:DescribeUser
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/stage": "prod"
                }
            }
        }
    ]
}
```

例 3: リクエストタグに基づくユーザーまたはサーバーの作成を拒否する

次のポリシー例には、2 つのステートメントが含まれています。1 つ目のステートメントでは、タグのコストセンターキーに値がない場合、すべてのリソースに対する CreateServer オペレーションが拒否されます。

2 つ目のステートメントは、タグのコストセンターキーに 1、2、または 3 以外の値が含まれている場合に CreateServer オペレーションが拒否されます。

> **Note**
>
> このポリシーでは、costcenter というキーと 1、2、または3 の値を含むリソースを作成または削除できます。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": [
                "transfer:CreateServer"
            ],
```

```
            "Resource": [
                "*"
            ],
            "Condition": {
                "Null":  {
                    "aws:RequestTag/costcenter": "true"
                }
            }
        },
        {
            "Effect": "Deny",
            "Action": "transfer:CreateServer",
            "Resource": [
                "*"
            ],
            "Condition": {
                "ForAnyValue:StringNotEquals": {
                    "aws:RequestTag/costcenter": [
                        "1",
                        "2",
                        "3"
                    ]
                }
            }
        }
    ]
}
```

AWS Transfer Family ID とアクセスのトラブルシューティング

次の情報は、 および IAM の使用時に発生する可能性がある一般的な問題の診断 AWS Transfer Family と修正に役立ちます。

トピック

- でアクションを実行する権限がない AWS Transfer Family
- iam を実行する権限がありません。PassRole
- 自分の AWS アカウント以外のユーザーに自分の AWS Transfer Family リソース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 AWS Transfer Family

からアクションを実行する権限がないと AWS Management Console 通知された場合は、管理者に連絡してサポートを依頼する必要があります。管理者とは、サインイン認証情報を提供した担当者です。

以下の例のエラーは、mateojackson IAM ユーザーがコンソールを使用して [######] の詳細を表示する際に、transfer:*GetWidget* 許可がない場合に発生します。

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

この場合、Mateo は、transfer;:*GetWidget* アクションを使用して *my-example-widget* リソースへのアクセスが許可されるように、管理者にポリシーの更新を依頼します。

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 AWS Transfer Family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、そのサービスに既存のロールを渡すことができます。そのためには、サービスにロールを渡す権限が必要です。

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 AWS Transfer Family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メアリーに iam:PassRole アクションの実行を許可する必要があります。

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

以下のサンプルポリシーには、ロールを AWS Transfer Familyに渡す権限が含まれています。

```
{
```

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

自分の AWS アカウント以外のユーザーに自分の AWS Transfer Family リソース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 AWS Transfer Family 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が IAM と AWS Transfer Family 連携する方法。
- 所有 AWS アカウント している のリソースへのアクセスを提供する方法については、IAM ユーザーガイドの「所有 AWS アカウント している別の の IAM ユーザーへのアクセスを提供する」を参照してください。
- リソースへのアクセスをサードパーティー に提供する方法については AWS アカウント、IAM ユーザーガイドの「サードパーティー AWS アカウント が所有する へのアクセスを提供する」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) へのアクセス権限」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。

のコンプライアンス検証 AWS Transfer Family

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

は、SOC、PCI、HIPAA などを含みます。詳細なリストについては、AWS「コンプライアンスプログラムによる対象範囲内のサービス」を参照してください。

特定のコンプライアンスプログラム AWS の対象となるサービスのリストについては、「コンプライアンスAWS プログラムによる対象範囲内の のサービス」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。

を使用して、サードパーティーの監査レポートをダウンロードできます AWS Artifact。詳細については、「 でのレポートのダウンロード AWS Artifact」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 Transfer Family は、お客様のデータの機密性、貴社のコンプライアンス目的、適用される法律および規制によって決まります。 AWS では、コンプライアンスに役立つ以下のリソースを提供しています。

- セキュリティとコンプライアンスのクイックスタートガイド – これらのデプロイガイドでは、アーキテクチャ上の考慮事項について説明し、セキュリティとコンプライアンスに重点を置いたベースライン環境を にデプロイする手順について説明します AWS。
- HIPAA セキュリティとコンプライアンスのアーキテクチャに関するホワイトペーパー — 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 コンプライアンスのリソース – 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。
- AWS Config – この AWS 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、および規制にどの程度準拠しているかを評価します。
- 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とベストプラクティスへの準拠を確認するのに役立ちます。

の耐障害性 AWS Transfer Family

AWS グローバルインフラストラクチャは、 AWS リージョンとアベイラビリティーゾーンを中心に構築されています。 AWS リージョンは、低レイテンシー、高スループット、および高度に冗長なネットワークで接続された、物理的に分離および分離された複数のアベイラビリティーゾーンを提供します。アベイラビリティーゾーンでは、アベイラビリティーゾーン間で中断せずに、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、耐障害性、およびスケーラビリティが優れています。

AWS Transfer Family は最大 3 つのアベイラビリティーゾーンをサポートし、自動スケーリング、接続および転送リクエスト用の冗長フリートによってバックアップされます。

次の点に注意してください。

- パブリックエンドポイントの場合:
  - アベイラビリティーゾーンレベルの冗長性がサービスに組み込まれています。
  - AZ ごとに冗長フリートがあります。
  - この冗長性は自動的に提供されます
- 仮想プライベートクラウド (VPC) 内のエンドポイントについては、Virtual Private Cloud でサーバーを作成する を参照してください。

以下の資料も参照してください。

- AWS リージョン および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。
- レイテンシーベースのルーティングを使用して冗長性を高め、ネットワークレイテンシーを最小限に抑える方法の例については、ブログ記事AWS Transfer Family 「サーバーでのネットワークレイテンシーを最小限に抑える」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 Transfer Family

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

が AWS 公開している API コールを使用して、ネットワーク AWS Transfer Family 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 は必須で TLS 1.3 がお勧めです。
- DHE (楕円ディフィー・ヘルマン鍵共有) や ECDHE (楕円曲線ディフィー・ヘルマン鍵共有) などの完全前方秘匿性 (PFS) による暗号スイート。これらのモードは、Java 7 以降など、ほとんどの最新システムでサポートされています。

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。

ウェブアプリケーションファイアウォールを追加する

AWS WAF は、ウェブアプリケーションと APIsから保護するのに役立つウェブアプリケーションファイアウォールです。定義したカスタマイズ可能なウェブセキュリティルールと条件に基づいて、ウェブリクエストを許可、ブロック、またはカウントする (ウェブアクセスコントロールリストまたはウェブ ACL と呼ばれる) 一連のルールを設定できます。詳細については、「 AWS WAF を使用して APIs」を参照してください。

追加するには AWS WAF

1. https://console.aws.amazon.com/apigateway で API Gateway コンソールを開きます。
2. API ナビゲーションペインでカスタム ID プロバイダーテンプレートを選択します。
3. [Stages] (ステージ) を選択します。
4. [Stages] (ステージ) ペインで、ステージの名前を選択します。
5. [Stage Editor] (ステージエディタ) ペインで [Settings] (設定) タブを選択します。
6. 以下のいずれかの操作をします。

   - [Web Application Firewall (WAF)] (Web アプリケーションファイアウォール (WAF)) の下にある [Web ACL] (ウェブ ACL) で、このステージに関連付けるウェブ ACL を選択します。
   - 必要なウェブ ACL が存在しない場合、次の手順に従ってウェブ ACL を作成する必要があります。
     1. [Create Web ACL] (ウェブ ACL を作成) を選択します。
     2. AWS WAF サービスのホームページで、ウェブ ACL の作成 を選択します。
     3. [Web ACL details] (ウェブ ACL の詳細) の [Name] (名前) にウェブ ACL の名前を入力します。
     4. [Rules] (ルール) で [Add rules] (ルールの追加) を選択してから [Add my own rules and rule groups] (独自のルールとルールグループの追加) を選択します。
     5. [Rule set] (ルールタイプ) で特定の IP アドレスのリストを識別する IP セットを選択します。
     6. [Rule] (ルール) にルールの名前を入力します。

7. [IP set] (IP セット) で既存の IP セットを選択します。IP セットを作成するには、「IP セットの作成」を参照してください。
8. [IP address to use as the originating address] (発信アドレスとして使用する IP アドレス) で [IP address in header] (ヘッダー内の IP アドレス) を選択します。
9. [Header field name] (ヘッダーフィールド名) に `SourceIP` 入力します。
10. [Position inside header] (ヘッダー内の位置) で [First IP address] (最初の IP アドレス) を選択します。
11. [Fallback for missing IP address] (IP アドレスが見つからない場合のフォールバック) で、ヘッダー内の無効な (または欠落している) IP アドレスの処理方法に応じて [Match] (一致) または [No Match] (一致なし) を選択します。
12. [Action] (アクション) で IP セットのアクションを選択します。
13. [Default web ACL action for requests that don't match any rules] (どのルールにも一致しないリクエストに対するデフォルトのウェブ ACL アクション) で [Allow] (許可) または [Block] (ブロック) を選択してから [Next] (次へ) を選択します。
14. ステップ 4 と 5 では [Next] (次へ) を選択します。
15. [Review and create] (確認と作成) で選択内容を見直してから[Create web ACL] (ウェブ ACL を作成) を選択します。

7. [Save Changes] (変更を保存) を選択します。
8. [Resources] (リソース) を選択します。
9. [Actions] (アクション) で [Deploy API] (API のデプロイ) を選択します。

AWS ウェブアプリケーションファイアウォール AWS Transfer Family によるセキュリティの詳細については、 AWS ストレージブログのAWS 「アプリケーションファイアウォールと Amazon API Gateway AWS Transfer Family による保護」を参照してください。

サービス間の混乱した代理の防止

混乱した代理問題は、アクションを実行するためのアクセス許可を持たないエンティティが、より特権のあるエンティティにアクションの実行を強制できてしまう場合に生じる、セキュリティ上の問題です。では AWS、サービス間のなりすましにより、混乱した代理問題が発生する可能性があります。サービス間でのなりすましは、あるサービス (呼び出し元サービス) が、別のサービス (呼び出し対象サービス) を呼び出すときに発生する可能性があります。呼び出す側サービスは、そのアクセス許可を使用して、他の顧客のリソースにアク セスするために、他の方法ではアクセス許可を持っていないはずの操作を行うことができます。これを防ぐために、 AWS には、アカウント内のリソー

スへのアクセス権が付与されたサービスプリンシパルですべてのサービスのデータを保護するために役立つツールが用意されています。この問題の詳細な説明については、「IAM ユーザーガイド」の「混乱した代理問題」を参照してください。

リソースポリシーで aws:SourceArnおよび aws:SourceAccount グローバル条件コンテキストキーを使用して、 AWS Transfer Family がリソースに対して持つアクセス許可を制限することをお勧めします。両方のグローバル条件コンテキストキーを同じポリシーステートメントで使用する場合は、aws:SourceAccount 値と、aws:SourceArn 値に含まれるアカウントが、同じアカウント ID を示している必要があります。

混乱した代理問題を回避するための最も効果的な方法は、許可するリソースに正確な Amazon リソースネーム (ARN) を使用することです。複数のリソースを指定する場合は、ARNの未知の部分にワイルドカード文字 (*) を使用したaws:SourceArnグローバルコンテキスト条件キーを使用する。例えば arn:aws:transfer::*region*::*account-id*:server/* です。

AWS Transfer Family は、次のタイプのロールを使用します。

- ユーザーロール – サービスマネージドユーザーが必要な Transfer Family リソースにアクセスできるようにします。 AWS Transfer Family は、Transfer Family ユーザー ARN のコンテキストでこのロールを引き受けます。
- 「アクセスロール」 — 転送中の Amazon S3 ファイルのみへのアクセスを提供します。インバウンド AS2 転送の場合、アクセスロールは契約の Amazon リソースネーム (ARN) を使用します。アウトバウンド AS2 転送の場合、アクセスロールはコネクタの ARN を使用します。
- 「呼び出しロール」 — Amazon API Gateway でサーバーのカスタム ID プロバイダーとして使用します。Transfer Family は、Transfer Family サーバー ARN のコンテキストでこの役割を引き受けます。
- ログ記録ロール — Amazon へのエントリのログ記録に使用されます CloudWatch。Transfer Family はこのロールを使用して、成功と失敗の詳細をファイル転送に関する情報とともに記録します。Transfer Family は、Transfer Family サーバー ARN のコンテキストでこの役割を引き受けます。アウトバウンドAS2転送では、ログロールはコネクタARNを使用する。
- 「実行ロール」 — Transfer Family ユーザーがワークフローを呼び出して起動できるようにします。Transfer Family は、Transfer Family のワークフローARNのコンテキストにおいて、この役割を引き受ける。

詳細については、「 IAM ユーザーガイド」の「IAM のポリシーとアクセス許可」を参照してください。

> **Note**
>
> 次の例では、*############*をユーザー自身の情報で置き換えます。

> **Note**
>
> この例では、ArnLike と ArnEquals の両方を使用しています。これらは機能的には同じなので、ポリシーを作成する際にはどちらを使用してもかまいません。Transfer Family ドキュメントでは、条件にワイルドカード文字が含まれる場合は ArnLikeを使用し、完全一致の条件を示す場合は ArnEquals を使用しています。

AWS Transfer Family ユーザーロールのサービス間の混乱した代理の防止

次のポリシーの例は、アカウント内の任意のサーバーのすべてのユーザーにロールを引き受けることを許可します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "Service": "transfer.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "account-id"
                },
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:transfer:region:account-id:user/*"
                }
            }
        }
    ]
}
```

次のポリシーの例は、特定のサーバーのすべてのユーザーにロールを引き受けることを許可します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "Service": "transfer.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "account-id"
                },
                "ArnEquals": {
                    "aws:SourceArn": "arn:aws:transfer:region:account-id:user/server-id/*"
                }
            }
        }
    ]
}
```

次のポリシーの例は、特定のサーバーの特定のユーザーにロールを引き受けることを許可します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "Service": "transfer.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:transfer:region:account-id:user/server-id/user-name"
                }
            }
```

```
        }
    ]
}
```

AWS Transfer Family ワークフローロールのサービス間の混乱した代理の防止

次のポリシーの例は、アカウント内のすべてのワークフローでロールを引き受けることを許可します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "Service": "transfer.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "account-id"
                },
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:transfer:region:account-id:workflow/*"
                }
            }
        }
    ]
}
```

次のポリシーの例は、特定のワークフローでロールを引き受けることを許可します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "Service": "transfer.amazonaws.com"
```

```
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:transfer:region:account-
id:workflow/workflow-id"
                }
            }
        }
    ]
}
```

AWS Transfer Family のログ記録と呼び出しロールのサービス間の混乱した代理の防止

> **Note**
>
> 以下の例は、ロギングロールと呼び出しロールの両方で使用できます。
> これらの例では、サーバーにワークフローがアタッチされていない場合、ワークフローの ARN の詳細を削除できます。

次のログ記録/呼び出しポリシーの例では、アカウント内のすべてのサーバー (およびワークフロー) がロールを引き受けることを許可します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAllServersWithWorkflowAttached",
            "Effect": "Allow",
            "Principal": {
                "Service": "transfer.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "account-id"
                },
                "ArnLike": {
                   "aws:SourceArn": [
```

```
                    "arn:aws:transfer:region:account-id:server/*",
                    "arn:aws:transfer:region:account-id:workflow/*"
                ]
            }
        }
    }
]
}
```

次の口グ記録/呼び出しポリシーの例では、特定のサーバー (およびワークフロー) がロールを引き受けることを許可します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowSpecificServerWithWorkflowAttached",
            "Effect": "Allow",
            "Principal": {
                "Service": "transfer.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "account-id"
                },
                "ArnEquals": {
                   "aws:SourceArn": [
                       "arn:aws:transfer:region:account-id:server/server-id",
                       "arn:aws:transfer:region:account-id:workflow/workflow-id"
                   ]
                }
            }
        }
    ]
}
```

AWSAWS Transfer Family の マネージドポリシー

ユーザー、グループ、ロールにアクセス許可を追加するには、自分でポリシーを記述するよりも、AWS 管理ポリシーを使用する方が簡単です。必要なアクセス許可のみをチームに提供する AWS Identity and Access Management (IAM) カスタマー管理ポリシーを作成するには、時間と専門知識が

必要です。すぐに開始するには、 AWS マネージドポリシーを使用できます。これらのポリシーは、一般的なユースケースをターゲット範囲に含めており、 AWS アカウントで利用できます。 AWS マネージドポリシーの詳細については、IAM ユーザーガイドの「AWS マネージドポリシー」を参照してください。すべての AWS 管理ポリシーの詳細なリストについては、AWS 「 管理ポリシーリファレンスガイド」を参照してください。

AWS サービスは、 AWS マネージドポリシーを維持および更新します。 AWS 管理ポリシーのアクセス許可は変更できません。サービスでは、新しい機能を利用できるようにするために、 AWS マネージドポリシーに権限が追加されることがあります。この種類の更新は、ポリシーがアタッチされている、すべてのアイデンティティ (ユーザー、グループおよびロール) に影響を与えます。新しい機能が立ち上げられた場合や、新しいオペレーションが使用可能になった場合に、各サービスが AWS マネージドポリシーを更新する可能性が最も高くなります。サービスは AWS マネージドポリシーからアクセス許可を削除しないため、ポリシーの更新によって既存のアクセス許可が中断されることはありません。

さらに、 は、複数の サービスにまたがる職務機能の マネージドポリシー AWS をサポートします。例えば、 ReadOnlyAccess AWS 管理ポリシーは、すべての AWS サービスとリソースへの読み取り専用アクセスを提供します。サービスが新機能を起動すると、 は新しいオペレーションとリソースの読み取り専用アクセス許可 AWS を追加します。ジョブ機能のポリシーの一覧および詳細については、「IAM ユーザーガイド」の「AWS のジョブ機能のマネージドポリシー」を参照してください。

AWS 管理ポリシー： AWSTransferConsoleFullAccess

このAWSTransferConsoleFullAccessポリシーは、 AWS マネジメントコンソールを介して Transfer Family へのフルアクセスを提供します。

アクセス許可の詳細

このポリシーには、以下のアクセス許可が含まれています。

- acm:ListCertificates- 証明書 Amazon Resource Names (ARN) のリストと各 ARN のドメイン名を取得する権限を付与する。
- ec2:DescribeAddresses- 1つ以上のElastic IPアドレスを記述する許可を与える。
- ec2:DescribeAvailabilityZones - 利用可能なアベイラビリティ・ ゾーンを1つ以上記述する許可を与える。
- ec2:DescribeNetworkInterfaces - 1つ以上のエラスティックネットワークインターフェースを記述する許可を与える。

- ec2:DescribeSecurityGroups—1 つ以上のセキュリティグループを記述する許可を付与
- ec2:DescribeSubnets—1 つ以上のサブネットを記述する許可を付与
- ec2:DescribeVpcs - 1つまたは複数の仮想プライベートクラウド (VPC) を記述する許可を与えます。
- ec2:DescribeVpcEndpoints - 1つ以上のVPCエンドポイントを記述する許可を与えます。
- health:DescribeEventAggregates—各イベントタイプ (発行、スケジュール変更、およびアカウント通知) のイベント数を返します。
- iam:GetPolicyVersion—指定の管理ポリシーのバージョンについて、ポリシードキュメントなどの情報を取得する許可を付与
- iam:ListPolicies—すべての管理ポリシーを一覧表示する許可を付与
- iam:ListRoles—指定されたパスのプレフィックスを持つ IAM ロールを一覧表示する許可を付与
- iam:PassRole — Transfer Family に IAM ロールを渡すための許可を付与します。詳細については、「 にロールを渡すアクセス許可をユーザーに付与する AWS のサービス」を参照してください。
- route53:ListHostedZones—現在の AWS アカウントに関連付けられたパブリックホストゾーンおよびプライベートホストゾーンのリストを取得するアクセス許可を付与
- s3:ListAllMyBuckets—リクエストの認証された送信者が所有するすべてのバケットを一覧表示するアクセス許可を付与します
- transfer:* — Transfer Family リソースへのアクセスを許可します。アスタリスク (*) はすべてのリソースへのアクセスを許可します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "transfer.amazonaws.com"
                }
            }
        },
        {
```

```
            "Effect": "Allow",
            "Action": [
                "acm:ListCertificates",
                "ec2:DescribeAddresses",
                "ec2:DescribeAvailabilityZones",
                "ec2:DescribeNetworkInterfaces",
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeSubnets",
                "ec2:DescribeVpcs",
                "ec2:DescribeVpcEndpoints",
                "health:DescribeEventAggregates",
                "iam:GetPolicyVersion",
                "iam:ListPolicies",
                "iam:ListRoles",
                "route53:ListHostedZones",
                "s3:ListAllMyBuckets",
                "transfer:*"
            ],
            "Resource": "*"
        }
    ]
}
```

AWS マネージドポリシー： AWSTransferFullAccess

この AWSTransferFullAccess ポリシーは、Transfer Family サービスへのフルアクセスを提供します

アクセス許可の詳細

このポリシーには、以下のアクセス許可が含まれています。

- transfer:* — TranTransfer Family ilyのリソースにアクセスする権限を付与します。アスタリスク (*) はすべてのリソースへのアクセスを許可します。
- iam:PassRole — Transfer Family に IAM ロールを渡すための許可を付与します。詳細については、「 にロールを渡すアクセス許可をユーザーに付与する AWS のサービス」を参照してください。
- ec2:DescribeAddresses- 1つ以上のElastic IPアドレスを記述する許可を与える。
- ec2:DescribeNetworkInterfaces—1 つ以上のネットワークインターフェイスを記述する許可を付与
- ec2:DescribeVpcEndpoints - 1つ以上のVPCエンドポイントを記述する許可を与えます。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "transfer:*",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "transfer.amazonaws.com"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeVpcEndpoints",
                "ec2:DescribeNetworkInterfaces",
                "ec2:DescribeAddresses"
            ],
            "Resource": "*"
        }
    ]
}
```

AWS マネージドポリシー： AWSTransferLoggingAccess

このAWSTransferLoggingAccessポリシーは、 AWS Transfer Family にログストリームとグループを作成し、アカウントにログイベントを配置するためのフルアクセスを付与します。

許可の詳細

このポリシーには、 に対する以下のアクセス許可が含まれています Amazon CloudWatch Logs。

- CreateLogStream — プリンシパルでログストリームを作成するアクセス許可を付与します。
- DescribeLogStreams - プリンシパルでロググループのログストリームを一覧表示できます。
- CreateLogGroup — プリンシパルでロググループを作成するアクセス許可を付与します。

- PutLogEvents — プリンシパルでログイベントのバッチを指定されたログストリームにアップロードできます。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogStream",
                "logs:DescribeLogStreams",
                "logs:CreateLogGroup",
                "logs:PutLogEvents"
            ],
            "Resource": "*"
        }
    ]
}
```

AWS マネージドポリシー： AWSTransferReadOnlyAccess

AWSTransferReadOnlyAccess ポリシーは、Transfer Familyサービスへの読み取り専用アクセスを提供します。

許可の詳細

このポリシーには Transfer Family に関する以下のアクセス許可が含まれています。

- DescribeUser — プリンシパルでユーザーの説明を表示できます。
- DescribeServer — プリンシパルでユーザーの説明を表示するためのアクセス許可を付与します。
- ListUsers - プリンシパルでユーザーの一覧を表示できます。
- ListServers — プリンシパルでアカウントのサーバーを一覧表示するアクセス許可を付与します。
- TestIdentityProvider - 構成された ID プロバイダーが正しく設定されているかどうかをプリンシパルでテストできます。
- ListTagsForResource - プリンシパルでリソースのタグを一覧表示するためのアクセス許可を付与します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "transfer:DescribeUser",
                "transfer:DescribeServer",
                "transfer:ListUsers",
                "transfer:ListServers",
                "transfer:TestIdentityProvider",
                "transfer:ListTagsForResource"
            ],
            "Resource": "*"
        }
    ]
}
```

AWSAWS マネージドポリシーへの Transfer Family の更新

AWS Transfer Family の AWS マネージドポリシーの更新に関する詳細を、このサービスがこれらの変更の追跡を開始した以降の分について表示します。このページへの変更に関する自動アラートについては、のドキュメント履歴 AWS Transfer Family ページの RSS フィードを購読してください。

変更	説明	日付
ドキュメントの更新	Transfer Familyの各管理ポリシーにセクションを追加しました。	2022 年 1 月 27 日
AWSTransferReadOnlyAccess – 既存ポリシーへの更新	AWS Transfer Family は、ポリシーが を読み取れるように新しいアクセス許可を追加しました AWS Managed Microsoft AD。	2021 年 9 月 30 日

変更	説明	日付
AWS Transfer Family が変更の追跡を開始しました	AWS Transfer Family が AWS マネージドポリシーの変更の追跡を開始しました。	2021 年 6 月 15 日

トラブルシューティング AWS Transfer Family

次の情報は、 の使用時に発生する可能性がある一般的な問題の診断と修正に役立ちます AWS Transfer Family。

Transfer Family の IAM に関する問題については、AWS Transfer Family ID 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。

トピック

- サービス管理ユーザーのトラブルシューティング
- Amazon API Gateway の問題をトラブルシューティングする
- 暗号化された Amazon S3 バケットのポリシーのトラブルシューティング
- 認証の問題のトラブルシューティング
- マネージド型ワークフローの問題のトラブルシューティング
- ワークフローの復号に関する問題のトラブルシューティング
- Amazon EFS の問題をトラブルシューティングする
- ID プロバイダーのテストのトラブルシューティング
- SFTP コネクターにトラステッドホストキーを追加する際のトラブルシューティングを行います。
- ファイルアップロードの問題のトラブルシューティング
- ResourceNotFound 例外のトラブルシューティング
- SFTP コネクタの問題のトラブルシューティング
- AS2 の問題をトラブルシューティングする

サービス管理ユーザーのトラブルシューティング

このセクションでは、以下の問題に対する可能な解決策を説明します。

トピック

- Amazon EFS サービス管理ユーザーのトラブルシューティング
- 公開鍵本文が長すぎる場合のトラブルシューティング
- SSH 公開鍵の追加に失敗した場合のトラブルシューティング

Amazon EFS サービス管理ユーザーのトラブルシューティング

説明

sftp コマンドを実行した場合にプロンプトの代わりに次のメッセージが表示されることがあります。

```
Couldn't canonicalize: Permission denied
           Need cwd
```

原因

AWS Identity and Access Management (IAM) ユーザーのロールには、Amazon Elastic File System (Amazon EFS) へのアクセス許可がありません。

対策

ユーザーのロールについてポリシーのアクセス権限を増やします。などの AWS 管理ポリシーを追加できますAmazonElasticFileSystemClientFullAccess。

公開鍵本文が長すぎる場合のトラブルシューティング

説明

サービス管理対象ユーザーを作成しようとすると、次のエラーが表示されます。

```
Failed to create user (1 validation error detected:
'sshPublicKeyBody' failed to satisfy constraint: Member must have length less than or
 equal to 2048)
```

原因

パブリックキー本文に PGP キーを入力する可能性があり、 AWS Transfer Family はサービスマネージドユーザーの PGP キーをサポートしていません。

解決策

PGP 鍵が RSA ベースの場合は、PEM 形式に変換できます。たとえば、Ubuntu は次のような変換ツールを提供しています:https://manpages.ubuntu.com/manpages/xenial/man1/openpgp2ssh.1.html

SSH 公開鍵の追加に失敗した場合のトラブルシューティング

説明

サービス管理ユーザーの公開鍵を追加しようとすると、次のエラーが表示されます。

```
Failed to add SSH public key (Unsupported or invalid SSH public key format)
```

原因

SSH2 形式のパブリックキーをインポートしようとしている可能性があり、 AWS Transfer Family はサービスマネージドユーザーの SSH2-formattedパブリックキーをサポートしていません。

解決策

キーを OpenSSH 形式に変換する必要があります。このプロセスは、「SSH2 公開鍵を PEM 形式に変換します。」で説明されています。

Amazon API Gateway の問題をトラブルシューティングする

このセクションでは、以下の API Gateway の問題に対して考えられる解決策について説明します。

トピック

- 認証失敗の回数が多すぎる
- 接続が終了する

認証失敗の回数が多すぎる

説明

Secure Shell (SSH) File Transfer Protocol (SFTP) を使用してサーバーに接続しようとすると、次のエラーが発生します。

```
Received disconnect from 3.15.127.197 port 22:2: Too many authentication failures
Authentication failed.
Couldn't read packet: Connection reset by peer
```

原因

入力したユーザーパスワードが正しない可能性があります。もう一度、正しいパスワードを入力してみてください。

パスワードが正しい場合、この問題はロールの Amazon リソースネーム (ARN) が有効でないことが原因かもしれません。それが原因であることを確認するには、サーバーの ID プロバイダをテストします。次のようなレスポンスが表示される場合、すべてがゼロのロール ID 値は、ロール ARN がプレースホルダーのみであることを示します。

```
{
    "Response": "{\"Role\": \"arn:aws:iam::000000000000:role/MyUserS3AccessRole\",
\"HomeDirectory\": \"/\"}",
    "StatusCode": 200,
    "Message": "",
    "Url": "https://api-gateway-ID.execute-api.us-east-1.amazonaws.com/prod/
servers/transfer-server-ID/users/myuser/config"
}
```

解決策

プレースホルダーロール ARN をサーバーへのアクセス権限がある実際のロールに置き換えます。

ロールを更新するには

1. https://console.aws.amazon.com/cloudformation で AWS CloudFormation コンソールを開きます。
2. 左のナビゲーションペインで [Stacks] (スタック) をクリックします。
3. 「スタック」リストでスタックを選択し、「パラメータ」タブを選択します。
4. [更新] を選択します。「スタックの更新」ページで、「現在のテンプレートを使用する」を選択し、「次へ」を選択します。
5. を、Transfer Family サーバーにアクセスするための十分なアクセス許可を持つロール ARN UserRoleArnに置き換えます。

   > **Note**
   >
   > 必要なアクセス許可を付与するには、ロールに AmazonAPIGatewayAdministrator および AmazonS3FullAccess マネージドポリシーを追加します。

6. 「次へ」を選択し、もう一度「次へ」を選択します。####の確認ページで、IAM リソース を作成する AWS CloudFormation 可能性があることを確認してから、スタックの更新 を選択します。

接続が終了する

説明

Secure Shell (SSH) File Transfer Protocol (SFTP) を使用してサーバーに接続しようとすると、次のエラーが発生します。

```
Connection closed
```

原因

この問題の考えられる原因の 1 つは、Amazon CloudWatch ログ記録ロールに Transfer Family との信頼関係がないことです。

解決策

サーバのロギングロールが Transfer Family と信頼関係にあることを確認します。詳細については、「信頼関係を確立するには」を参照してください。

暗号化された Amazon S3 バケットのポリシーのトラブルシューティング

説明

暗号化された Amazon S3 バケットがTransfer Family サーバーのストレージとして使用されています。サーバーにファイルをアップロードしようとすると、「`Couldn't close file: Permission denied`」エラーが表示されます。

そしてサーバーログを表示すると、次のエラーが表示されます。

```
ERROR Message="Access denied" Operation=CLOSE Path=/bucket/user/test.txt BytesIn=13
ERROR Message="Access denied"
```

原因

IAM ユーザーのポリシーには、暗号化されたバケットにアクセスする権限がありません。

解決策

必要な AWS Key Management Service (AWS KMS) アクセス権限を付与するには、ポリシーに追加のアクセス権限を指定する必要があります。詳細については、「Amazon S3 でのデータ暗号化」を参照してください。

認証の問題のトラブルシューティング

このセクションでは、以下の認証問題に対して考えられる解決策について説明します。

トピック

- 認証エラー-SSH/SFTP
- マネージド AD のレルム不一致問題
- その他の認証に関する問題

認証エラー-SSH/SFTP

説明

Secure Shell (SSH) File Transfer Protocol (SFTP) を使用してサーバーに接続しようとすると、次のようなメッセージが表示されます。

```
Received disconnect from 3.130.115.105 port 22:2: Too many authentication failures
  Authentication failed.
```

> **Note**
>
> API Gateway を使用していて、このエラーが表示される場合は、認証失敗の回数が多すぎるを参照してください。

原因

ユーザーの RSA キーペアを追加していないので、代わりにパスワードを使って認証する必要があります。

解決策

`sftp` コマンドを実行する際、`-o PubkeyAuthentication=no` オプションを指定します。このオプションは、システムにパスワードを要求させます。例:

```
sftp -o PubkeyAuthentication=no sftp-user@server-id.server.transfer.region-
id.amazonaws.com
```

マネージド AD のレルム不一致問題

説明

ユーザーのレルムとグループのレルムを一致する必要があります。両方ともデフォルトレルム内にあるか、または両方とも信頼されるレルムに属している必要があります。

原因

ユーザーとそのグループが一致しない場合、そのユーザーはTTransfer Family ilyによって認証されません。ユーザーの ID プロバイダーをテストすると、「ユーザーのグループに関連するアクセスが見つかりません」というエラーが表示されます。

解決策

グループレルム (デフォルトまたは信頼できる) と一致するユーザーのレルム内のグループを参照します。

その他の認証に関する問題

説明

認証エラーが発生し、他のトラブルシューティングは実行されません。

原因

先頭または末尾にスラッシュ (/) を含む論理ディレクトリのターゲットを指定した可能性があります。

解決策

論理ディレクトリのターゲットを更新して、先頭がスラッシュで、末尾にスラッシュが含まれていないことを確認してください。例えば、 /DOC-EXAMPLE-BUCKET/imagesは許容されますが、 DOC-EXAMPLE-BUCKET/imagesと /DOC-EXAMPLE-BUCKET/images/は許容されません。

マネージド型ワークフローの問題のトラブルシューティング

このセクションでは、以下のワークフロー問題に対して考えられる解決策について説明します。

トピック

Amazon を使用したワークフロー関連のエラーのトラブルシューティング CloudWatch

説明

ワークフローに問題がある場合は、Amazon CloudWatch を使用して原因を調査できます。

原因

いくつかの原因が考えられるので、Amazon CloudWatch Logs を使用して調査します。

解決策

Transfer Family は、ワークフロー実行ステータスを CloudWatch Logs に出力します。 CloudWatch ログには、次のタイプのワークフローエラーが表示されることがあります。

- `"type": "StepErrored"`
- `"type": "ExecutionErrored"`
- `"type": "ExecutionThrottled"`
- `"Service failure on starting workflow"`

ワークフローの実行ログは、異なるフィルターやパターン構文を使ってフィルターすることができます。例えば、ログに CloudWatch ログフィルターを作成して、ExecutionErroredメッセージを含むワークフロー実行ログをキャプチャできます。詳細については、「Amazon Logs ユーザーガイド」の「サブスクリプションを使用したログデータのリアルタイム処理」および「フィルターとパターン構文」を参照してください。 CloudWatch

StepErrored

```
2021-10-29T12:57:26.272-05:00
            {"type":"StepErrored","details":
{"errorType":"BAD_REQUEST","errorMessage":"Cannot
            tag Efs file","stepType":"TAG","stepName":"successful_tag_step"},
            "workflowId":"w-
abcdef01234567890","executionId":"1234abcd-56ef-78gh-90ij-1234klmno567",
```

```
            "transferDetails":
{"serverId":"s-1234567890abcdef0","username":"lhr","sessionId":"1234567890abcdef0"}
```

ここで、StepErrored はワークフロー内のステップでエラーを生成したことを示します。1 つのワークフローには複数のステップを設定できます。このエラーでは、どのステップでエラーが発生したかがエラーメッセージに表示されます。しかし、Amazon EFS ファイルシステム内のファイルへのタグ付けはサポートされていないため、このステップはエラーを生成しました。

ExecutionErrored

```
2021-10-29T12:57:26.618-05:00
            {"type":"ExecutionErrored","details":{},"workflowId":"w-w-
abcdef01234567890",
            "executionId":"1234abcd-56ef-78gh-90ij-1234klmno567","transferDetails":
{"serverId":"s-1234567890abcdef0",
            "username":"lhr","sessionId":"1234567890abcdef0"}}
```

ワークフローで実行できないステップがあると、「ExecutionErrored」というメッセージが表示されます。たとえば、特定のワークフローに設定したステップのうち 1 つのステップを実行できないと、ワークフロー全体が失敗します。

Executionthrottled

ワークフローが、システムがサポートできる速度よりも速い速度でトリガーされる場合、実行はスロットルされます。このログメッセージは、ワークフローの実行速度を落とす必要があることを示しています。ワークフロー実行率をスケールダウンできない場合は、 へのお問い合わせ AWS Support で AWSにお問い合わせください。

ワークフロー開始時に発生したサービス障害

サーバーからワークフローを削除し、新しいワークフローに置き換える場合、またはサーバー構成を更新する場合（ワークフローの実行ロールに影響する）、新しいワークフローを実行する前に約 10 分間待機する必要があります。Transfer Familyサーバーはワークフローの詳細をキャッシュし、サーバーがキャッシュを更新するのに10分かかります。

さらに、アクティブなSFTPセッションからログアウトし、10分間の待ち時間の後にログインし直すと、変更を確認できます。

ワークフローコピーエラーのトラブルシューティング

説明

アップロードされたファイルをコピーするステップを含むワークフローを実行している場合、以下のエラーが発生する可能性がある：

```
{
  "type": "StepErrored", "details": {
    "errorType": "BAD_REQUEST", "errorMessage": "Bad Request (Service: Amazon S3;
 Status Code: 400; Error Code: 400 Bad Request;
    Request ID: request-ID; S3 Extended Request ID: request-ID Proxy: null)",
 "stepType": "COPY", "stepName": "copy-step-name" },
  "workflowId": "workflow-ID",
  "executionId": "execution-ID",
  "transferDetails": {
     "serverId": "server-ID",
     "username": "user-name",
     "sessionId": "session-ID"
  }
}
```

原因

ソースファイルは、レプリケート先バケット AWS リージョン とは異なる にある Amazon S3 バケットにあります。

解決策

コピーステップを含むワークフローを実行する場合、コピー元とコピー先のバケットが同じ AWS リージョンにあることを確認します。

ワークフローの復号に関する問題のトラブルシューティング

このセクションでは、暗号化されたワークフローに関する以下の問題に対して考えられる解決策について説明します。

トピック

- 署名付き暗号化ファイルのエラーのトラブルシューティング
- FIPS アルゴリズムのエラーのトラブルシューティング

署名付き暗号化ファイルのエラーのトラブルシューティング

説明

復号ワークフローが失敗し、次のエラーが表示されます。

```
"Encrypted file with signed message unsupported"
```

原因

Transfer Family は現在、暗号化されたファイルの署名をサポートしていません。

解決策

PGP クライアントで、暗号化されたファイルに署名するオプションがある場合は、選択をクリアしてください。Transfer Family は現在、暗号化されたファイルの署名をサポートしていないためです。

FIPS アルゴリズムのエラーのトラブルシューティング

説明

復号化ワークフローが失敗し、ログメッセージが次のように表示されます。

```
{
   "type": "StepErrored",
   "details": {
      "errorType": "BAD_REQUEST",
      "errorMessage": "File encryption algorithm not supported with FIPS mode
 enabled.",
      "stepType": "DECRYPT",
      "stepName": "step-name"
   },
   "workflowId": "workflow-ID",
   "executionId": "execution-ID",
   "transferDetails": {
      "serverId": "server-ID",
      "username": "user-name",
      "sessionId": "session-ID"
   }
}
```

原因

Transfer Family サーバーでは FIPS モードが有効になっており、関連する復号化ワークフローステップがあります。Transfer Family サーバーにアップロードする前にファイルを暗号化すると、暗号化クライアントは FIPS で承認されていない対称暗号化アルゴリズムを使用する暗号化ファイルを生成

する可能性があります。このようなシナリオでは、ワークフローはファイルを復号化できません。次の例では、GnuPG バージョン 2.4.0 は OCB (非 FIPS ブロック暗号モード) を使用してファイルを暗号化しています。これによりワークフローが失敗します。

解決策

ファイルの暗号化に使用した GPG キーを編集し、再暗号化する必要があります。次の手順では、実行する必要のある手順について説明します。

PGP キーを編集するには

1. 次のコマンドを実行して、編集する必要のあるキーを特定します。`gpg --list-keys`

   これにより、キーのリストが返されます。各キーには次のような詳細が含まれます。

   ```
   pub   ed25519 2022-07-07 [SC]
         wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
   uid           [ultimate] Mary Major <marymajor@example.com>
   sub   cv25519 2022-07-07 [E]
   ```

2. 編集するキーを特定します。前の手順で示した例では、ID は`wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY`です。
3. `gpg --edit-key wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY` を実行します。

   システムから GnuPG プログラムと指定されたキーに関する詳細を応答します。
4. gpg>プロンプトで、`showpref`を入力します。以下の詳細情報が返されます。

   ```
   [ultimate] (1). Mary Major <marymajor@example.com>
      Cipher: AES256, AES192, AES, 3DES
      AEAD: OCB
      Digest: SHA512, SHA384, SHA256, SHA224, SHA1
      Compression: ZLIB, BZIP2, ZIP, Uncompressed
      Features: MDC, AEAD, Keyserver no-modify
   ```

   キーに保存されている推薦アルゴリズムが一覧表示されていることに注意してください。
5. OCB を除くすべてのアルゴリズムを保持するようにキーを編集したいと考えています。`setpref`保持するすべてのアルゴリズムを指定してコマンドを実行します。

   ```
   gpg> setpref AES256, AES192,AES,3DES,SHA512, SHA384, SHA256, SHA224, SHA1,ZLIB,
    BZIP2, ZIP, Uncompressed
   ```

これは以下の詳細を返す：

```
Set preference list to:
     Cipher: AES256, AES192, AES, 3DES
     AEAD:
     Digest: SHA512, SHA384, SHA256, SHA224, SHA1
     Compression: ZLIB, BZIP2, ZIP, Uncompressed
     Features: MDC, Keyserver no-modify
Really update the preferences? (y/N)
```

6. 「yupdate」と入力して更新し、変更を確認するプロンプトが表示されたらパスワードを入力します。
7. 変更を保存します。

```
gpg> save
```

復号化ワークフローを再実行する前に、編集したキーを使用してファイルを再暗号化する必要があります。

Amazon EFS の問題をトラブルシューティングする

このセクションでは、以下の Amazon EFS の問題に対して考えられる解決策について説明します。

トピック

- POSIX プロファイルが見つからない場合のトラブルシューティング
- Amazon EFS による論理ディレクトリのトラブルシューティング

POSIX プロファイルが見つからない場合のトラブルシューティング

説明

サーバーに Amazon EFS ストレージを使用していて、カスタム ID プロバイダーを使用している場合は、POSIX プロファイルを AWS Lambda 関数に提供する必要があります。

原因

考えられる原因のひとつは、 AWS Lambdaに対応した Amazon API Gateway のメソッドを作成するためのテンプレートが、現在 POSIX 情報を含んでいないことです。

POSIX 情報を提供した場合、POSIX 情報を提供するために使用したフォーマットが、Transfer Family によって正しく解析されていない可能性があります。

解決策

`PosixProfile` パラメータに JSON 要素を Transfer Family に提供していることを確認します。

例えば、Python を使用している場合、`PosixProfile` パラメータをパースする箇所に以下の行を追加することができる：

```
if PosixProfile:
        response_data["PosixProfile"] = json.loads(PosixProfile)
```

または、 で次の行 JavaScriptを追加できます。ここで、 *uid-value* と *gid-value*は、それぞれユーザー ID (UID) とグループ ID (GID) を表す整数 0 以上です。

```
PosixProfile: {"Uid": uid-value, "Gid": gid-value},
```

これらのコード例では、`PosixProfile` パラメータを文字列としてではなく、JSON オブジェクトとして Transfer Family に送信しています。

また、 内では AWS Secrets Manager、 `PosixProfile`パラメータを次のように保存する必要があります。*your-uid*と*your-gid*を GID と UID の実際の値に置き換えてください。

```
{"Uid": your-uid, "Gid": your-gid, "SecondaryGids": []}
```

Amazon EFS による論理ディレクトリのトラブルシューティング

説明

ユーザーのホームディレクトリが存在せず、ユーザーが `ls` コマンドを実行すると、システムは次のように応答します。

```
sftp> ls
remote readdir ("/"): No such file or directory
```

原因

Transfer Family サーバーが Amazon EFS を使用している場合は、ユーザーが論理ホームディレクトリで作業するには、ユーザーのホームディレクトリを読み取りおよび書き込みアクセスで作成する必要があります。ユーザーは自分の論理ホームディレクトリに対する mkdir の権限がないため、このディレクトリを自分で作成することはできません。

解決策

親ディレクトリへの管理アクセス権を持つユーザーは、ユーザーの論理ホームディレクトリを作成する必要があります。

ID プロバイダーのテストのトラブルシューティング

説明

コンソールまたは TestIdentityProvider API コールを使用して ID プロバイダーをテストする場合、Responseフィールドは空です。例:

```
{
    "Response": "{}",
    "StatusCode": 200,
    "Message": ""
}
```

原因

最も考えられる原因は、ユーザー名やパスワードが間違っているために認証が失敗したことです。

解決策

ユーザーの認証情報が正しいことを確認し、必要に応じてユーザー名またはパスワードを更新してください。

SFTP コネクターにトラステッドホストキーを追加する際のトラブルシューティングを行います。

説明

SFTP コネクタを作成または編集しているときに、トラステッドホストキーを追加しようとすると、次のエラーが表示されます。Failed to edit connector details (Invalid host key format.)

原因

正しいパブリックキーを貼り付けた場合、問題は key. AWS Transfer Family does の comment部分が現在キーのコメント部分を受け入れていないことである可能性があります。

解決策

キーのコメント部分をテキストフィールドに貼り付けると、そのキーのコメント部分が削除されます。例えば、あなたのキーが以下のようなものだとする：

```
ssh-rsa AAAA...== marymajor@dev-dsk-marymajor-1d-c1234567.us-east-1.amazon.com
```

==文字の後に続くテキストを削除し、キーの最初の部分と==を含む部分だけを貼り付けます。

```
ssh-rsa AAAA...==
```

ファイルアップロードの問題のトラブルシューティング

このセクションでは、次のファイルアップロードの問題の考えられる解決策について説明します。

トピック

- Amazon S3 ファイルアップロードエラーのトラブルシューティング
- 読み取り不可能なファイル名のトラブルシューティング

Amazon S3 ファイルアップロードエラーのトラブルシューティング

説明

Transfer Family を使用して Amazon S3 ストレージにファイルをアップロードしようとすると、次のエラーメッセージが表示されます。AWS Transfer は S3 オブジェクトへのランダムアクセス書き込みをサポートしていません。

原因

サーバーのストレージに Amazon S3 を使用している場合、Transfer Family は 1 回の転送で複数の接続をサポートしていません。

解決策

Transfer Family サーバーのストレージに Amazon S3 を使用している場合は、1 回の転送に複数の接続を使用することを記載しているクライアントソフトウェアのオプションをすべて無効にします。

読み取り不可能なファイル名のトラブルシューティング

説明

アップロードしたファイルの一部に壊れたファイル名が表示される。FTP や SFTP での転送で、ウムラウト、アクセント付きの文字、中国語やアラビア語などの特定の文字など、ファイル名の特定の文字が文字化けする問題が発生することがあります。

原因

FTP プロトコルと SFTP プロトコルではファイル名の文字エンコーディングをクライアントがネゴシエートできますが、Amazon S3 と Amazon EFS ではできません。代わりに、UTF-8 文字エンコードが必要です。その結果、特定の文字が正しく表示されません。

解決策

この問題を解決するには、クライアントアプリケーションでファイル名の文字エンコーディングを確認し、UTF-8 に設定されていることを確認します。

`ResourceNotFound` 例外のトラブルシューティング

説明

リソースが見つからないというエラーが表示されます。たとえば、`UpdateServer`を実行すると、エラーが返されます。

```
An error occurred (ResourceNotFoundException) when calling the UpdateServer operation:
 Unknown server
```

原因

ResourceNotFoundException メッセージを受信する理由はいくつかあります。ほとんどの場合、API コマンドで指定したリソースは存在しません。既存のリソースを指定した場合、最も可能性の高い原因は、デフォルトリージョンがリソースのリージョンと異なることです。たとえば、デフォルトのリージョンが us-east-1 で、Transfer Family サーバーが us-east-2 にある場合、「未知リソース」例外が発生します。

デフォルトリージョンの設定について詳しくは、「によるクイック設定」を参照してください。aws configure

解決策

API コマンドにリージョンパラメータを追加して、特定のリソースの場所を明示的に指定します。

```
aws transfer -describe-server --server-id server-id --region us-east-2
```

SFTP コネクタの問題のトラブルシューティング

このセクションでは、以下の SFTP コネクタの問題に対して考えられる解決策について説明します。

トピック

- キーネゴシエーションが失敗する
- その他の SFTP コネクタの問題

キーネゴシエーションが失敗する

説明

キー交換ネゴシエーションが失敗するエラーが表示されます。例:

```
Key exchange negotiation failed due to incompatible host key algorithms.
Client offered: [ecdsa-sha2-nistp256, ecdsa-sha2-nistp384,
ecdsa-sha2-nistp521, rsa-sha2-512, rsa-sha2-256] Server offered: [ssh-rsa]
```

原因

このエラーは、サーバーがサポートするホストキーアルゴリズムとコネクタがサポートするホストキーアルゴリズムが重複していないためです。

解決策

リモートサーバーが、エラーメッセージに記載されているクライアントホストキーアルゴリズムの少なくとも 1 つをサポートしていることを確認してください。サポートされているアルゴリズムのリストについては、「AWS Transfer Family SFTP コネクタのセキュリティポリシー」を参照してください。

その他の SFTP コネクタの問題

説明

の実行後にエラーが表示されますが`StartFileTransfer`、問題の原因は不明であり、API コール後にコネクタ ID のみが返されます。

原因

このエラーにはいくつかの原因が考えられます。トラブルシューティングを行うには、コネクタをテストし、 CloudWatch ログを検索することをお勧めします。

解決策

- コネクタをテストする：「」を参照してくださいSFTP コネクタをテストします。。テストに失敗した場合、システムは失敗の理由に基づいたエラーメッセージを提供します。このセクションでは、コンソールまたは TestConnection API コマンドを使用してコネクタをテストする方法について説明します。
- コネクタの CloudWatch ログを表示する：「」を参照してくださいSFTP コネクタのログエントリの例。このトピックでは、SFTP コネクタログエントリの例と、適切なログを見つけるのに役立つ命名規則について説明します。

AS2 の問題をトラブルシューティングする

適用宣言書 2 (AS2) 対応サーバーのエラーメッセージとトラブルシューティングのヒントについては、以下を参照してください。AS2 エラーコード

API リファレンス

以下のセクションでは、 AWS Transfer Family API サービスの呼び出し、データ型、パラメータ、エラーについて説明します。

トピック

AWS Transfer Family API へようこそ

AWS Transfer Family は、以下のプロトコルを介して Amazon Simple Storage Service (Amazon S3) ストレージとの間でファイルを転送するために使用できる安全な転送サービスです。

- Secure Shell (SSH) File Transfer Protocol (SFTP)
- File Transfer Protocol Secure (FTPS)
- File Transfer Protocol (FTP)
- 適用性ステートメント 2 (AS2)

ファイル転送プロトコルは、金融サービス、医療、広告、小売など、さまざまな業界のデータ交換ワークフローで使用されます。 AWS Transfer Family は、ファイル転送ワークフローの への移行を簡素化します AWS。

AWS Transfer Family サービスを使用するには、選択した AWS リージョンでサーバーをインスタンス化します。サーバーを作成して、使用可能なサーバーを一覧表示し、サーバーを更新および削除できます。サーバーは、 からファイルオペレーションをリクエストするエンティティです AWS Transfer Family。サーバーには、多くの重要なプロパティが含まれています。サーバーは、システムにより割り当てられた `ServerId` 識別子によって識別される名前付きインスタンスです。必要に応じて、ホスト名や、カスタムホスト名もサーバーに割り当てることができます。サービスの使用料

は、すべてのインスタンス化されたサーバー (OFFLINE も含む) および転送されたデータの量に従って請求されます。

ファイルオペレーションをリクエストするサーバーがユーザーを認識している必要があります。ユーザー名によって識別されるユーザーがサーバーに割り当てられます。ユーザー名は、リクエストの認証に使用されます。サーバーの認証方法となりうるのは、AWS_DIRECTORY_SERVICE、SERVICE_MANAGED、AWS_LAMBDA、または API_GATEWAY のいずれかです。

次のいずれかの ID プロバイダーのタイプを使用して、ユーザーを認証できます。

- SERVICE_MANAGED の場合は、SSH パブリックキーはサーバーにユーザーのプロパティとともに保存されます。ユーザーは、SERVICE_MANAGED 認証方法に 1 つ以上の SSH パブリックキーを登録できます。クライアントが SERVICE_MANAGED メソッドのファイルオペレーションをリクエストする場合、クライアントがユーザー名とSSH プライベートキーを提供することで、認証され、アクセスが提供されます。
- AWS_DIRECTORY_SERVICE 認証方法を選択することで、Microsoft Active Directory グループでのユーザー認証とアクセスを管理できます。
- を使用してカスタム ID プロバイダーに接続できます AWS Lambda。AWS_LAMBDA 認証方法を選択します。
- ユーザー認証とアクセスの両方を提供するカスタム認証方法を使用すると、ユーザーリクエストも認証できます。このメソッドは Amazon API Gateway に依存しており、ID プロバイダーからの API コールを使用して、ユーザーリクエストを検証します。このメソッドは、API コールでは API_GATEWAY と呼ばれ、コンソールでは [Custom] と呼ばれます。このカスタムメソッドを使用して、ディレクトリサービス、データベース名とパスワードのペア、またはその他のメカニズムに対してユーザーを認証できる場合があります。

ユーザーには、ユーザー自身と Amazon S3 バケットの間の信頼関係が含まれたポリシーが割り当てられます。SFTP ユーザーは、バケットのすべてまたは一部にアクセスできる可能性があります。サーバーがユーザーに代わって機能するためには、サーバーがユーザーから信頼関係を継承する必要があります。信頼関係を含んでいる AWS Identity and Access Management (IAM) ロールが作成され、そのロールが AssumeRole アクションに割り当てられます。サーバーは、ユーザーであるかのようにファイルオペレーションを実行できるようになります。

home ディレクトリのプロパティセットを持つユーザーの場合、そのディレクトリ (またはフォルダ) はファイルオペレーションのターゲットおよびソースとして機能します。home ディレクトリが設定されていない場合は、バケットの root ディレクトリがランディングディレクトリになります。

サーバー、ユーザー、およびロールはすべて、Amazon リソースネーム (ARN) で識別されます。ARN が含まれているエンティティにキーバリューペアであるタグを割り当てることができます。タグは、これらのエンティティのグループ化または検索に使用できるメタデータです。タグが便利な例としては、経理上の処理が挙げられます。

AWS Transfer Family ID 形式では、次の規則が守られています。

- `ServerId` 値の形式は `s-01234567890abcdef` です。
- `SshPublicKeyId` 値の形式は `key-01234567890abcdef` です。

Amazon リソースネーム (ARN) は次のような形式です。

- ユーザーの場合、ARN の形式は `arn:aws:transfer:`*`region`*`:`*`account-id`*`:server/`*`server-id`* です。

  サーバー ARN の例: `arn:aws:transfer:us-east-1:123456789012:server/s-01234567890abcdef`
- ユーザーの場合、ARN の形式は `arn:aws:transfer:`*`region`*`:`*`account-id`*`:user/`*`server-id`*`/`*`username`* です。

  例は `arn:aws:transfer:us-east-1:123456789012:user/s-01234567890abcdef/user1` です。

使用する DNS エントリ (エンドポイント) は次のとおりです。

- API エンドポイントの形式は `transfer.`*`region`*`.amazonaws.com` です。
- サーバーエンドポイントの形式は `server.transfer.`*`region`*`.amazonaws.com` です。

AWS リージョン別の Transfer Family エンドポイントのリストについては、「」のAWS Transfer Family「 エンドポイントとクォータ」を参照してくださいAWS 全般のリファレンス。

この の API インターフェイスリファレンスには、 の管理に使用できるプログラミングインターフェイスのドキュメント AWS Transfer Family が含まれています AWS Transfer Family。リファレンス構造は次のとおりです。

- API アクションのアルファベット順リストについては、「Actions」を参照してください。
- データ型のアルファベット順リストについては、「Data Types」を参照してください。

- 共通クエリパラメータのリストについては、「共通パラメータ」を参照してください。
- エラーコードの説明については、「共通エラー」を参照してください。

> **Tip**
>
> 実際にコマンドを実行する代わりに、任意の API コールで `--generate-cli-skeleton` パラメータを使用してパラメータテンプレートを生成して表示できます。生成されたテンプレートを使用してカスタマイズし、後のコマンドの入力として使用できます。詳細については、「パラメータスケルトンファイルを生成して使用するには」を参照してください。

アクション

以下のアクションがサポートされています:

- CreateAccess
- CreateAgreement
- CreateConnector
- CreateProfile
- CreateServer
- CreateUser
- CreateWorkflow
- DeleteAccess
- DeleteAgreement
- DeleteCertificate
- DeleteConnector
- DeleteHostKey
- DeleteProfile
- DeleteServer
- DeleteSshPublicKey
- DeleteUser
- DeleteWorkflow
- DescribeAccess

- DescribeAgreement
- DescribeCertificate
- DescribeConnector
- DescribeExecution
- DescribeHostKey
- DescribeProfile
- DescribeSecurityPolicy
- DescribeServer
- DescribeUser
- DescribeWorkflow
- ImportCertificate
- ImportHostKey
- ImportSshPublicKey
- ListAccesses
- ListAgreements
- ListCertificates
- ListConnectors
- ListExecutions
- ListHostKeys
- ListProfiles
- ListSecurityPolicies
- ListServers
- ListTagsForResource
- ListUsers
- ListWorkflows
- SendWorkflowStepState
- StartDirectoryListing
- StartFileTransfer
- StartServer
- StopServer

- TagResource
- TestConnection
- TestIdentityProvider
- UntagResource
- UpdateAccess
- UpdateAgreement
- UpdateCertificate
- UpdateConnector
- UpdateHostKey
- UpdateProfile
- UpdateServer
- UpdateUser

CreateAccess

管理者が、 を使用して、有効なプロトコル経由でファイルをアップロードおよびダウンロードするためのアクセス権をディレクトリ内のどのグループが持つかを選択するために使用します AWS Transfer Family。たとえば、Microsoft Active Directory に 50,000 ユーザーが含まれる可能性があってもサーバーにファイルを転送する能力を必要とするのはそのうちのごく一部のみです。管理者は `CreateAccess` を使用することで、この機能を必要とする適切なユーザーのセットにアクセス権を限定することができます。

リクエストの構文

```
{
   "ExternalId": "string",
   "HomeDirectory": "string",
   "HomeDirectoryMappings": [
      {
         "Entry": "string",
         "Target": "string",
         "Type": "string"
      }
   ],
   "HomeDirectoryType": "string",
   "Policy": "string",
   "PosixProfile": {
      "Gid": number,
      "SecondaryGids": [ number ],
      "Uid": number
   },
   "Role": "string",
   "ServerId": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

ExternalId

ディレクトリ内の特定のグループを識別するために必要な一意の識別子。関連付けるグループのユーザーは、 を使用して、有効なプロトコル経由で Amazon S3 または Amazon EFS リソースにアクセスできます AWS Transfer Family。グループ名がわかっている場合は、Windows を使用して次のコマンドを実行して SID 値を表示できます PowerShell。

```
Get-ADGroup -Filter {samAccountName -like "YourGroupName*"} -Properties
* | Select SamAccountName,ObjectSid
```

そのコマンドで、 を Active Directory グループの名前YourGroupNameに置き換えます。

このパラメータの検証に使用される正規表現は、スペースを含まない大文字と小文字の英数字からなる文字列です。下線文字 (_) や =,.@:/- の文字を含めることもできます。

型: 文字列

長さの制限：最小長は 1 です。最大長は 256 です。

パターン：`S-1-[\d-]+`

必須：はい

HomeDirectory

ユーザーがクライアントを使用してサーバーにログインするときの、ユーザーのランディングディレクトリ (フォルダ)。

`HomeDirectory` の例は `/bucket_name/home/mydirectory` です。

> **Note**
>
> `HomeDirectory` パラメータは、`HomeDirectoryType` が `PATH` に設定されている場合のみ使用されます。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 1,024 です。

パターン: `(|/.*)`

必須: いいえ

HomeDirectoryMappings

ユーザーに表示する Amazon S3 または Amazon EFS のパスとキー、およびそれらをどのように表示するかを指定する論理ディレクトリマッピング。Entry と Target のペアを指定する必要があり、Entry はパスの表示方法を示し、Target は実際の Amazon S3 または Amazon EFS のパスです。ターゲットを指定しただけの場合は、そのまま表示されます。また、 AWS Identity and Access Management (IAM) ロールが のパスへのアクセスを許可していることを確認する必要がありますTarget。この値は、HomeDirectoryType がLOGICALに設定されている場合にのみ設定できます。

以下は Entry と Target のペアの例です。

```
[ { "Entry": "/directory1", "Target": "/bucket_name/home/
mydirectory" } ]
```

ほとんどの場合、セッションポリシーの代わりにこの値を使用することで、指定されたホームディレクトリ (chroot) にユーザーをロックダウンできます。そのためには、Entry を/ に設定し、Target を HomeDirectory パラメータ値に設定します。

以下は、chroot のための Entry と Target のペアの例です。

```
[ { "Entry": "/", "Target": "/bucket_name/home/mydirectory" } ]
```

型: HomeDirectoryMapEntry オブジェクトの配列

配列メンバー : 最小数は 1 項目です。最大数は 50,000 項目です。

必須: いいえ

HomeDirectoryType

ユーザーがサーバーにログインするときにホームディレクトリにするランディングディレクトリ (フォルダ) のタイプ。これを PATH に設定した場合、ユーザーには、ファイル転送プロトコルクライアントに、絶対的な Amazon S3 バケットまたは現状の Amazon EFS パスが表示されます。これを LOGICAL に設定した場合、Amazon S3 または Amazon EFS パスをユーザーに表示する方法に関して、HomeDirectoryMappings でマッピングを指定する必要があります。

> **Note**
>
> HomeDirectoryType が LOGICAL の場合は、HomeDirectoryMappings パラメータを使用してマッピングを指定する必要があります。一方、HomeDirectoryType が

PATH の場合は、HomeDirectory パラメータを使用して絶対パスを指定します。テンプレートに HomeDirectory と HomeDirectoryMappings の両方を含めることはできません。

型: 文字列

有効な値：PATH | LOGICAL

必須：いいえ

Policy

複数のユーザーで同じ AWS Identity and Access Management (IAM) ロールを使用できるようにするためのユーザーのセッションポリシー。このポリシーは、ユーザーアクセスのスコープを Amazon S3 バケットの一部に絞り込みます。このポリシー内に使用できる変数には、${Transfer:UserName}、${Transfer:HomeDirectory}、${Transfer:HomeBucket} があります。

> **Note**
>
> このポリシーは、ServerId ドメインが Amazon S3 の場合にのみ適用されます。Amazon EFS はセッション· ポリシーを使用しません。
> セッションポリシーの場合、 はポリシーの Amazon リソースネーム (ARN) ではなく、ポリシーを JSON BLOB として AWS Transfer Family 保存します。JSON blob としてポリシーを保存し、Policy 因数に渡します。
> セッションポリシーの例については、「セッションポリシーの例」を参照してください。
> 詳細については、 API リファレンスAssumeRoleの「」を参照してください。 AWS Security Token Service

型: 文字列

長さの制限: 最小長は 0 です。最大長は 2048 です。

必須: いいえ

PosixProfile

ユーザーの Amazon EFS ファイルシステムへのアクセスを制御する、ユーザー ID (Uid Gid)、グループ ID (SecondaryGids)、およびセカンダリグループ ID () を含む完全な POSIX ID。ファ

イルシステム内のファイルとディレクトリに設定される POSIX アクセス許可によって、Amazon EFS ファイルシステムとの間でファイルを転送するときにユーザーが得るアクセスのレベルが決まります。

型: PosixProfile オブジェクト

必須: いいえ

Role

Amazon S3 バケットまたは Amazon EFS ファイルシステムへのユーザーのアクセスを制御する AWS Identity and Access Management (IAM) ロールの Amazon リソースネーム (ARN)。Amazon S3 このロールにアタッチされたポリシーにより、ファイルを Amazon S3 バケットまたは Amazon EFS ファイルシステム間で転送する際の、ユーザーに付与するアクセスレベルが決定されます。IAM ロールには、ユーザーの転送リクエストを処理する際に、サーバーによるリソースへのアクセスを許可する信頼関係も含まれる必要があります。

型: 文字列

長さの制限: 最小長は 20 です。最大長は 2,048 です。

Pattern: `arn:.*role/\S+`

必須：はい

ServerId

サーバーインスタンスにシステムで割り当てられた一意の識別子。これはユーザーを追加したサーバーに固有です。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `s-([0-9a-f]{17})`

必須：はい

レスポンスの構文

```
{
   "ExternalId": "string",
```

```
    "ServerId": "string"
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

ExternalId

を使用して、有効なプロトコルを介して Amazon S3 または Amazon EFS リソースにユーザーがアクセスできるグループの外部識別子 AWS Transfer Family。

型: 文字列

長さの制限：最小長は 1 です。最大長は 256 です。

パターン：`S-1-[\d-]+`

ServerId

ユーザーが接続しているサーバーの識別子。

型: 文字列

長さの制限: 固定長は 19 です。

パターン：`s-([0-9a-f]{17})`

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceExistsException

要求されたリソースは存在しないか、コマンドに指定されたリージョン以外のリージョンに存在します。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

CreateAgreement

契約を作成します。契約とは、 AWS Transfer Family サーバーと AS2 プロセスの間の二者間取引相手契約またはパートナーシップです。この契約は、サーバーと AS2 プロセス間のファイルおよびメッセージの転送の関係を定義します。Transfer Family は、サーバー、ローカルプロファイル、パートナープロファイル、証明書、その他の属性を組み合わせて、契約を定義します。

パートナーは `PartnerProfileId` で識別され、AS2 プロセスは `LocalProfileId` で識別されます。

リクエストの構文

```
{
   "AccessRole": "string",
   "BaseDirectory": "string",
   "Description": "string",
   "LocalProfileId": "string",
   "PartnerProfileId": "string",
   "ServerId": "string",
   "Status": "string",
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
      }
   ]
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

AccessRole

コネクターは、AS2 または SFTP プロトコルを使用してファイルを送信するために使用されます。アクセスロールには、使用する AWS Identity and Access Management ロールの Amazon リソースネーム (ARN) を指定します。

AS2 コネクタ用

AS2 では、StartFileTransfer を呼び出してリクエストパラメータ にファイルパスを指定することでファイルを送信できます。SendFilePathsファイルの親ディレクトリ (例えば --send-file-paths /bucket/dir/file.txt の場合、親ディレクトリは /bucket/dir/) を使用して、処理済みの AS2 メッセージファイルを一時的に保存し、パートナーから受け取った MDN を保存して、送信の関連メタデータを含む最終的な JSON ファイルを記述します。そのため、AccessRole は、StartFileTransfer リクエストで使用されるファイルの場所の親ディレクトリに対する読み取り/書き込みアクセスを提供する必要があります。さらに、StartFileTransfer で送信するファイルの親ディレクトリに対する読み取り/書き込みアクセスを提供する必要があります。

AS2 コネクタに Basic 認証を使用している場合、アクセスロールにはシークレットの secretsmanager:GetSecretValue 権限が必要です。シークレットが Secrets Manager のマネージドキーではなく、カスタマー AWS マネージドキーを使用して暗号化されている場合、ロールにはそのキーに対する アクセスkms:Decrypt許可も必要です。

SFTP コネクタ用

アクセスロールが、StartFileTransfer リクエストで使用されるファイルロケーションの親ディレクトリへの読み取りおよび書き込みアクセスを提供していることを確認します。さらに、ロールが にアクセスsecretsmanager:GetSecretValue許可を付与していることを確認します AWS Secrets Manager。

型: 文字列

長さの制限: 最小長は 20 です。最大長は 2,048 です。

Pattern: arn:.*role/\S+

必須：はい

BaseDirectory

AS2 プロトコルを使用して転送されるファイルのランディングディレクトリ (フォルダ) 。

BaseDirectory の例は /DOC-EXAMPLE-BUCKET/home/mydirectory です。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 1,024 です。

Pattern: (|/.*)

必須：はい

Description

契約を識別するための名前または短い説明。

型: 文字列

長さの制限: 最小長は 1 です。最大長は 200 です。

Pattern: `[\p{Graph}]+`

必須: いいえ

LocalProfileId

AS2 ローカルプロファイルの一意の識別子です。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `p-([0-9a-f]{17})`

必須：はい

PartnerProfileId

契約で使用されるパートナープロファイルの一意の識別子です。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `p-([0-9a-f]{17})`

必須：はい

ServerId

サーバーインスタンスにシステムで割り当てられた一意の識別子。これは、契約が使用する特定のサーバーです。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `s-([0-9a-f]{17})`

必須：はい

Status

契約のステータス。合意内容は `ACTIVE` または `INACTIVE` のいずれかになります。

型: 文字列

有効な値：`ACTIVE | INACTIVE`

必須：いいえ

Tags

契約のグループ化および検索に使用できるキーと値のペアです。

型: Tag オブジェクトの配列

配列メンバー：最小数は 1 項目です。最大数は 50 項目です。

必須：いいえ

レスポンスの構文

```
{
   "AgreementId": "string"
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

AgreementId

契約の一意の識別子。この ID は、契約を削除または更新する場合だけでなく、契約 ID を指定する必要があるその他の API 呼び出しでも使用できます。

型: 文字列

長さの制限: 固定長は 19 です。

パターン：a-([0-9a-f]{17})

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceExistsException

要求されたリソースは存在しないか、コマンドに指定されたリージョン以外のリージョンに存在します。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

ThrottlingException

リクエストのスロットリングにより、リクエストが拒否されました。

HTTP ステータスコード：400

例

例

次の例では、契約を作成し、契約 ID を返します。

```
aws transfer create-agreement --server-id s-021345abcdef6789 --local-profile-
id p-1234567890abcdef0 --partner-profile-id p-abcdef01234567890 --base-folder /DOC-
EXAMPLE-BUCKET/AS2-files --access-role arn:aws:iam::111122223333:role/AS2-role
```

レスポンス例

API コールは、新しい契約の契約 ID を返します。

```
{
    "AgreementId": "a-11112222333344444"
}
```

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

CreateConnector

AS2 または SFTP プロトコルの接続パラメータをキャプチャするコネクタを作成します。AS2 の場合、コネクタは、外部でホストされている AS2 サーバーにファイルを送信するために必要です。SFTP の場合、SFTP サーバーにファイルを送信するとき、または SFTP サーバーからファイルを受信するときにコネクタが必要です。コネクタの詳細については、AS2 コネクタの設定」および「SFTP コネクタの作成」を参照してください。

> **Note**
>
> AS2 (`As2Config`) または SFTP (`SftpConfig`) のいずれかに対して、構成オブジェクトを 1 つだけ指定する必要があります。

リクエストの構文

```
{
   "AccessRole": "string",
   "As2Config": {
      "BasicAuthSecretId": "string",
      "Compression": "string",
      "EncryptionAlgorithm": "string",
      "LocalProfileId": "string",
      "MdnResponse": "string",
      "MdnSigningAlgorithm": "string",
      "MessageSubject": "string",
      "PartnerProfileId": "string",
      "SigningAlgorithm": "string"
   },
   "LoggingRole": "string",
   "SecurityPolicyName": "string",
   "SftpConfig": {
      "TrustedHostKeys": [ "string" ],
      "UserSecretId": "string"
   },
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
      }
   ],
```

```
   "Url": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

AccessRole

コネクターは、AS2 または SFTP プロトコルを使用してファイルを送信するために使用されます。アクセスロールには、使用する AWS Identity and Access Management ロールの Amazon リソースネーム (ARN) を指定します。

AS2 コネクタ用

AS2 では、`StartFileTransfer` を呼び出してリクエストパラメータ にファイルパスを指定することでファイルを送信できます。`SendFilePaths`ファイルの親ディレクトリ (例えば `--send-file-paths /bucket/dir/file.txt` の場合、親ディレクトリは `/bucket/dir/`) を使用して、処理済みの AS2 メッセージファイルを一時的に保存し、パートナーから受け取った MDN を保存して、送信の関連メタデータを含む最終的な JSON ファイルを記述します。そのため、`AccessRole` は、`StartFileTransfer` リクエストで使用されるファイルの場所の親ディレクトリに対する読み取り/書き込みアクセスを提供する必要があります。さらに、`StartFileTransfer` で送信するファイルの親ディレクトリに対する読み取り/書き込みアクセスを提供する必要があります。

AS2 コネクタに Basic 認証を使用している場合、アクセスロールにはシークレットの `secretsmanager:GetSecretValue` 権限が必要です。シークレットが Secrets Manager のマネージドキーではなく、カスタマー AWS マネージドキーを使用して暗号化されている場合、ロールにはそのキーに対する アクセス`kms:Decrypt`許可も必要です。

SFTP コネクタ用

アクセスロールが、`StartFileTransfer` リクエストで使用されるファイルロケーションの親ディレクトリへの読み取りおよび書き込みアクセスを提供していることを確認します。さらに、ロールが にアクセス`secretsmanager:GetSecretValue`許可を付与していることを確認します AWS Secrets Manager。

型: 文字列

長さの制限: 最小長は 20 です。最大長は 2,048 です。

Pattern: `arn:.*role/\S+`

必須：はい

As2Config

AS2 コネクタオブジェクトのパラメータを含む構造体。

タイプ：As2ConnectorConfig オブジェクト

必須: いいえ

LoggingRole

コネクタが Amazon S3 イベントの CloudWatch ログ記録をオンにできるようにする (IAM) ロールの Amazon リソースネーム AWS Identity and Access Management (ARN)。 Amazon S3 設定すると、 CloudWatch ログにコネクタアクティビティを表示できます。

型: 文字列

長さの制限: 最小長は 20 です。最大長は 2,048 です。

パターン: `arn:.*role/\S+`

必須: いいえ

SecurityPolicyName

コネクタのセキュリティポリシーの名前を指定します。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 100 です。

Pattern: `TransferSFTPConnectorSecurityPolicy-[A-Za-z0-9-]+`

必須: いいえ

SftpConfig

SFTP コネクタオブジェクトのパラメータを含む構造体。

タイプ：SftpConnectorConfig オブジェクト

必須: いいえ

Tags

コネクタのグループ化および検索に使用できるキーと値のペアです。タグは、あらゆる目的でコネクターに付けられるメタデータです。

型: Tag オブジェクトの配列

配列メンバー：最小数は 1 項目です。最大数は 50 項目です。

必須: いいえ

Url

パートナーの AS2 または SFTP エンドポイントの URL。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 255 です。

必須: はい

レスポンスの構文

```
{
   "ConnectorId": "string"
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

ConnectorId

API コールが成功した後に返されるコネクタの一意の識別子。

型: 文字列

長さの制限: 固定長は 19 です。

パターン：`c-([0-9a-f]{17})`

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceExistsException

要求されたリソースは存在しないか、コマンドに指定されたリージョン以外のリージョンに存在します。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

ThrottlingException

リクエストのスロットリングにより、リクエストが拒否されました。

HTTP ステータスコード：400

例

例

次の例では、AS2 コネクタを作成します。コマンド内の項目を次のように置き換えます。

- `url`: 取引パートナーの AS2 サーバーの URL を指定します。
- `your-IAM-role-for-bucket-access`: ファイルの保存に使用している Amazon S3 バケットにアクセスできる IAM ロール。
- AWS アカウント ID を含むログ記録ロールの ARN を使用します。
- AS2 コネクタ構成パラメータを含むファイルへのパスを指定します。AS2 コネクタ設定オブジェクトについては、As2ConnectorConfig」で説明されています。

```
// Listing for testAs2Config.json
{
  "LocalProfileId": "your-profile-id",
  "PartnerProfileId": "partner-profile-id",
  "MdnResponse": "SYNC",
  "Compression": "ZLIB",
  "EncryptionAlgorithm": "AES256_CBC",
  "SigningAlgorithm": "SHA256",
  "MdnSigningAlgorithm": "DEFAULT",
  "MessageSubject": "Your Message Subject"
}
```

```
aws transfer create-connector --url "http://partner-as2-server-url" \
          --access-role your-IAM-role-for-bucket-access \
          --logging-role arn:aws:iam::your-account-id:role/service-role/
AWSTransferLoggingAccess \
          --as2-config file://path/to/testAS2Config.json
```

例

次の例では、SFTP コネクタを作成します。コマンド内の項目を次のように置き換えます。

- `sftp-server-url`: ファイルを交換する SFTP サーバーの URL を指定します。
- `your-IAM-role-for-bucket-access`: ファイルの保存に使用している Amazon S3 バケットにアクセスできる IAM ロール。
- AWS アカウント ID を含むログ記録ロールの ARN を使用します。
- SFTP コネクタ構成パラメータを含むファイルへのパスを指定します。SFTP コネクタ設定オブジェクトについては、SftpConnector「Config 」で説明されています。

```
// Listing for testSFTPConfig.json
{
   "UserSecretId": "arn:aws:secretsmanager:us-east-2:123456789012:secret:aws/transfer/
example-username-key",
   "TrustedHostKeys": [
      "sftp.example.com ssh-rsa AAAAbbbb...EEEE="
   ]
}
```

```
aws transfer create-connector --url "sftp://sftp-server-url" \
--access-role your-IAM-role-for-bucket-access \
--logging-role arn:aws:iam::your-account-id:role/service-role/AWSTransferLoggingAccess
 \
--sftp-config file://path/to/testSFTPConfig.json
```

例

API コールは、新しいコネクタのコネクタ ID を返します。

レスポンス例

```
{
    "ConnectorId": "a-11112222333344444"
}
```

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3

- AWS SDK for Python
- AWS SDK for Ruby V3

CreateProfile

AS2 転送に使用するローカルプロファイルまたはパートナープロファイルを作成します。

リクエストの構文

```
{
   "As2Id": "string",
   "CertificateIds": [ "string" ],
   "ProfileType": "string",
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
      }
   ]
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

As2Id

As2Id は、RFC 4130 で定義されている AS2-name です。インバウンド転送の場合、これはパートナーから送信される AS2 メッセージの AS2-From ヘッダーです。アウトバウンドコネクタの場合、これは StartFileTransfer API オペレーションを使用してパートナーに送信される AS2 メッセージの AS2-To ヘッダーです。この ID にスペースを含めることはできません。

型: 文字列

長さの制限: 最小長は 1 です。最大長は 128 です。

パターン: [\p{Print}\s]*

必須：はい

CertificateIds

インポートされた証明書の識別子の配列です。この識別子は、プロファイルやパートナープロファイルの操作に使用します。

タイプ：文字列の配列

長さの制限: 22 の固定長

Pattern: `cert-([0-9a-f]{17})`

必須: いいえ

ProfileType

作成するプロファイルのタイプを決定します。

- ローカルプロファイルを作成する場合にLOCAL指定します。ローカルプロファイルは、AS2対応のTransTransfer Family yサーバーの組織またはパーティーを表します。
- パートナー プロファイルを作成するようにPARTNER指定します。パートナープロフィールは、Transfer Familyの外部にあるリモート組織を表します。

型: 文字列

有効な値：`LOCAL | PARTNER`

必須: はい

Tags

AS2 プロファイルのグループ化と検索に使用できるキーと値のペア。

型: Tag オブジェクトの配列

配列メンバー：最小数は 1 項目です。最大数は 50 項目です。

必須：いいえ

レスポンスの構文

```
{
   "ProfileId": "string"
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

ProfileId

API コールが成功すると返される AS2 プロファイルの一意の識別子です。

型: 文字列

長さの制限: 固定長は 19 です。

パターン：`p-([0-9a-f]{17})`

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

ThrottlingException

リクエストのスロットリングにより、リクエストが拒否されました。

HTTP ステータスコード：400

例

例

次の例では、プロファイルを作成します。プロファイルIDを返します。

証明書 ID はimport-certificateの実行時に作成され、1 つは署名証明書用、もう 1 つは暗号化証明書用です。

```
aws transfer create-profile --as2-id MYCORP --certificate-ids c-abcdefg123456hijk
               c-987654aaaa321bbbb
```

レスポンス例

API コールは、新しいプロファイルのプロファイル ID を返します。

```
{
    "ProfileId": "p-11112222333344444"
}
```

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3

- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

CreateServer

AWSで選択したファイル転送プロトコルに基づいて、Auto Scaling 仮想サーバーをインスタンス化します。ファイル転送プロトコル対応サーバーの更新や、ユーザーの操作を行う際、新しく作成されたサーバーに割り当てられた、サービスによって生成された ServerId プロパティを使用します。

リクエストの構文

```
{
   "Certificate": "string",
   "Domain": "string",
   "EndpointDetails": {
      "AddressAllocationIds": [ "string" ],
      "SecurityGroupIds": [ "string" ],
      "SubnetIds": [ "string" ],
      "VpcEndpointId": "string",
      "VpcId": "string"
   },
   "EndpointType": "string",
   "HostKey": "string",
   "IdentityProviderDetails": {
      "DirectoryId": "string",
      "Function": "string",
      "InvocationRole": "string",
      "SftpAuthenticationMethods": "string",
      "Url": "string"
   },
   "IdentityProviderType": "string",
   "LoggingRole": "string",
   "PostAuthenticationLoginBanner": "string",
   "PreAuthenticationLoginBanner": "string",
   "ProtocolDetails": {
      "As2Transports": [ "string" ],
      "PassiveIp": "string",
      "SetStatOption": "string",
      "TlsSessionResumptionMode": "string"
   },
   "Protocols": [ "string" ],
   "S3StorageOptions": {
      "DirectoryListingOptimization": "string"
   },
   "SecurityPolicyName": "string",
   "StructuredLogDestinations": [ "string" ],
```

```
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
      }
   ],
   "WorkflowDetails": {
      "OnPartialUpload": [
         {
            "ExecutionRole": "string",
            "WorkflowId": "string"
         }
      ],
      "OnUpload": [
         {
            "ExecutionRole": "string",
            "WorkflowId": "string"
         }
      ]
   }
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

Certificate

AWS Certificate Manager (ACM) 証明書の Amazon リソースネーム (ARN)。`Protocols` が FTPS に設定されている場合は必須です。

新しいパブリック証明書をリクエストするには、「 ユーザーガイド」の「パブリック証明書のリクエスト AWS Certificate Manager 」を参照してください。

既存の証明書を ACM にインポートするには、「 AWS Certificate Manager ユーザーガイド」の「ACM への証明書のインポート」を参照してください。

プライベート IP アドレスを介して FTPS を使用するようにプライベート証明書をリクエストするには、「 ユーザーガイド」の「プライベート証明書のリクエスト AWS Certificate Manager 」を参照してください。

以下の暗号化アルゴリズムとキーサイズの証明書がサポートされています。

- 2048 ビット RSA (RSA_2048)
- 4096 ビット RSA (RSA_4096)
- 楕円素数曲線 256 ビット (EC_Prime256v1)
- 楕円素数曲線 384 ビット (EC_secp384R1)
- 楕円素数曲線 521 ビット (EC_secp521R1)

> **Note**
>
> 証明書は、FQDN または IP アドレスが指定され、発行者に関する情報が記載された有効な SSL/TLS X.509 バージョン 3 の証明書である必要があります。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 1600 です。

必須: いいえ

Domain

ファイル転送に使用されるストレージシステムのドメイン。使用できるドメインは、Amazon Simple Storage Service (Amazon S3) と Amazon Elastic File System (Amazon EFS) です。デフォルト値は S3 です。

> **Note**
>
> サーバーの作成後にサーバー名を変更することはできません。

型: 文字列

有効な値：`S3 | EFS`

必須：いいえ

EndpointDetails

サーバーに設定された Virtual Private Cloud (VPC) エンドポイント設定。VPC 内でエンドポイントをホストする場合、VPC 内のリソースにのみアクセスできるようにすることも、Elastic IP ア

ドレスをアタッチしてインターネット経由でクライアントにアクセスできるようにすることもできます。VPC のデフォルトのセキュリティグループは、エンドポイントに自動的に割り当てられます。

型: EndpointDetails オブジェクト

必須: いいえ

EndpointType

サーバーで使用したいエンドポイントのタイプ。サーバーのエンドポイントをパブリックアクセス (PUBLIC) にするか、VPC 内でホストするかを選択できます。VPC でホストされているエンドポイントを使用すると、サーバーとリソースへのアクセスを VPC 内のみに制限したり、Elastic IP アドレスを直接アタッチしてインターネット向けにするといった選択ができます。

> **Note**
>
> 2021 年 5 月 19 日以降、2021 年 5 月 19 日より前にアカウントで を使用してサーバーを作成 AWS アカウント していない場合、 EndpointType=VPC_ENDPOINTでサーバーを作成することはできません。2021 年 5 月 19 日 AWS アカウント 以前に EndpointType=VPC_ENDPOINTで を使用して既にサーバーを作成している場合は、影響を受けません。この日付を過ぎたら EndpointType=VPC を使用します。
> 詳細については、「VPC_ENDPOINT のサポート終了」を参照してください。
> VPC を EndpointType として使用することをお勧めします。このエンドポイントタイプでは、最大 3 つの Elastic IPv4 アドレス (BYO IP を含む) をサーバーのエンドポイントに直接関連付けて、VPC セキュリティグループを使用してクライアントのパブリック IP アドレスでトラフィックを制限できます。EndpointType を VPC_ENDPOINT に設定した場合、これは不可能です。

型: 文字列

有効な値：PUBLIC | VPC | VPC_ENDPOINT

必須：いいえ

HostKey

SFTP 対応サーバーで使用する RSA、ECDSA、または ED25519 プライベートキー。キーをローテーションしたい場合や、異なるアルゴリズムを使用するアクティブキーのセットが必要な場合に備えて、複数のホストキーを追加できます。

パスフレーズなしで RSA 2048 ビットキーを生成するには、以下のコマンドを使用する：

```
ssh-keygen -t rsa -b 2048 -N "" -m PEM -f my-new-server-key.
```

-b オプションには最小値 2048 を使用してください。3072 または 4096 を使用すると、より強力なキーを作成できます。

パスフレーズなしで ECDSA 256 ビット鍵を生成するには、以下のコマンドを使用する：

```
ssh-keygen -t ecdsa -b 256 -N "" -m PEM -f my-new-server-key.
```

ECDSA の -b オプションの有効値は 256、384、521 です。

パスフレーズなしで ED25519 鍵を生成するには、以下のコマンドを使用する：

```
ssh-keygen -t ed25519 -N "" -f my-new-server-key.
```

これらのコマンドのすべてについて、 を任意の文字列my-new-server-keyに置き換えることができます。

> **Important**
>
> 既存のユーザーを既存のSFTP 対応サーバーから新しいサーバーに移行する計画がない場合、ホストキーを更新しないでください。サーバーのホストキーを誤って変更することは、破壊的な操作になり得えます。

詳細については、「 ユーザーガイド」の「SFTP 対応サーバーのホストキーの更新 AWS Transfer Family 」を参照してください。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 4,096 です。

必須: いいえ

IdentityProviderDetails

IdentityProviderType が AWS_DIRECTORY_SERVICE、 AWS_LAMBDA または API_GATEWAY に設定されている場合に必要です。AWS_DIRECTORY_SERVICE でディレクトリを使用したり、API Gateway URL を含むカスタマー提供の認証 API を呼び出したりするた

めに必要な、すべての情報が含まれている配列を受け入れます。IdentityProviderType が SERVICE_MANAGED に設定されている場合、必須ではありません。

型: IdentityProviderDetails オブジェクト

必須: いいえ

IdentityProviderType

サーバーの認証のモード。デフォルト値は です。これによりSERVICE_MANAGED、 AWS Transfer Family サービス内でユーザー認証情報を保存してアクセスできます。

AWS_DIRECTORY_SERVICE を使用して、オンプレミス環境の AWS Directory Service for Microsoft Active Directory または Microsoft Active Directory、または AD Connector AWS を使用する の Active Directory グループへのアクセスを提供します。このオプションでは、IdentityProviderDetails パラメータを使用してディレクトリ ID を指定する必要もあります。

この API_GATEWAY 値を使用して、選択した ID プロバイダーと統合します。API_GATEWAY 設定では、IdentityProviderDetails パラメータを使用して認証を呼び出すには Amazon API Gateway エンドポイントの URL を指定する必要があります。

AWS_LAMBDA 値を使用して、 AWS Lambda 関数を ID プロバイダーとして直接使用します。この値を選択した場合、Function パラメータで、IdentityProviderDetails データ型の Lambda 関数の ARN を指定する必要があります。

型: 文字列

有効な値：SERVICE_MANAGED | API_GATEWAY | AWS_DIRECTORY_SERVICE | AWS_LAMBDA

必須：いいえ

LoggingRole

サーバーが Amazon S3 または Amazon EFSevents の Amazon CloudWatch ログ記録をオンにできるようにする AWS Identity and Access Management (IAM) ロールの Amazon リソースネーム (ARN)。設定すると、 CloudWatch ログにユーザーアクティビティを表示できます。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 2,048 です。

パターン: `(|arn:.*role/\S+)`

必須: いいえ

PostAuthenticationLoginBanner

ユーザーがサーバーに接続するときに表示する文字列を指定します。この文字列はユーザーが認証した後で表示されます。

> **Note**
>
> SFTP プロトコルは認証後の表示バナーをサポートしていません。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 4096 です。

パターン: `[\x09-\x0D\x20-\x7E]*`

必須: いいえ

PreAuthenticationLoginBanner

ユーザーがサーバーに接続するときに表示する文字列を指定します。この文字列はユーザーが認証される前に表示されます。例えば、次のバナーはシステムの使用に関する詳細を表示します。

```
This system is for the use of authorized users only. Individuals using
this computer system without authority, or in excess of their authority,
are subject to having all of their activities on this system monitored
and recorded by system personnel.
```

型: 文字列

長さの制限: 最小長は 0 です。最大長は 4096 です。

パターン: `[\x09-\x0D\x20-\x7E]*`

必須: いいえ

ProtocolDetails

サーバー用に構成されたプロトコル設定。

- パッシブモード (FTP および FTPS プロトコル用) を示すには、PassiveIp パラメータを使用します。ファイアウォール、ルーター、ロードバランサーの外部 IP アドレスなど、クアッドドット形式の単一 IPv4 アドレスを入力します。
- S3 バケットにアップロードしているファイルに対して、クライアントが SETSTAT コマンドの使用を試みた時に発生するエラーを無視するには、SetStatOption パラメータを使用します。SFTP クライアントを変更せずに AWS Transfer Family サーバーでSETSTATコマンドを無視してファイルをアップロードするには、値を に設定しますENABLE_NO_OP。SetStatOption パラメータを に設定するとENABLE_NO_OP、Transfer Family は Amazon CloudWatch Logs にログエントリを生成し、クライアントがいつSETSTAT呼び出しを行っているかを特定できるようにします。
- AWS Transfer Family サーバーが一意のセッション ID を介して最近ネゴシエートされたセッションを再開するかどうかを確認するには、 TlsSessionResumptionModeパラメータを使用します。
- As2Transports は、AS2 メッセージの転送方法を示します。現在は、HTTP のみがサポートされます。

タイプ：ProtocolDetails オブジェクト

必須: いいえ

Protocols

ファイル転送プロトコルクライアントがサーバーのエンドポイントに接続できる 1 つまたは複数のファイル転送プロトコルを指定します。使用可能なプロトコルは次のとおりです。

- SFTP(Secure Shell (SSH) File Transfer Protocol): SSH 経由のファイル転送
- FTPS (File Transfer Protocol Secure): TLS 暗号化によるファイル転送
- FTP (File Transfer Protocol): 暗号化されていないファイル転送
- AS2 (適用性ステートメント 2): 構造化 business-to-business データの転送に使用されます

> **Note**
>
> - を選択した場合はFTPS、 AWS Certificate Manager (ACM) に保存されている証明書を選択する必要があります。この証明書は、クライアントが FTPS 経由でサーバーに接続するときにサーバーを識別するために使用されます。
> - Protocol に FTP または FTPS が含まれる場合、EndpointType は VPC でなければならず、IdentityProviderType は AWS_DIRECTORY_SERVICE、AWS_LAMBDA または API_GATEWAY でなければなりません。

- Protocol に FTPが含まれる場合、AddressAllocationIds は関連付けられません。
- Protocol が SFTP のみに設定されている場合、EndpointType は PUBLIC に設定でき、IdentityProviderType はサポートされている ID タイプ (SERVICE_MANAGED、AWS_DIRECTORY_SERVICE、AWS_LAMBDA、または API_GATEWAY) のいずれかに設定できます。
- Protocol が AS2 を含む場合、EndpointType は VPC でなければならず、ドメインは、Amazon S3 でなければなりません。

タイプ：文字列の配列

配列メンバー：最小数は 1 項目です。最大数は 4 項目です。

有効な値：SFTP | FTP | FTPS | AS2

必須：いいえ

S3StorageOptions

Amazon S3 ディレクトリのパフォーマンスを最適化するかどうかを指定します。これはデフォルトでは無効になっています。

デフォルトでは、ホームディレクトリマッピングの は TYPEですDIRECTORY。
このオプションを有効にすると、マッピングにファイルターゲットを設定するHomeDirectoryMapEntryTypeFILE場合は、 を明示的に に設定する必要があります。

タイプ：S3StorageOptions オブジェクト

必須: いいえ

SecurityPolicyName

サーバーのセキュリティポリシーの名前を指定します。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 100 です。

Pattern: Transfer[A-Za-z0-9]*SecurityPolicy-[A-Za-z0-9-]+

必須: いいえ

StructuredLogDestinations

サーバーログの送信先となるロググループを指定します。

ロググループを指定するには、既存のロググループの ARN を指定する必要があります。この場合、ロググループの形式は次のようになります。

`arn:aws:logs:region-name:amazon-account-id:log-group:log-group-name:*`

例えば、次のようになります: `arn:aws:logs:us-east-1:111122223333:log-group:mytestgroup:*`

以前にサーバーのロググループを指定したことがある場合は、`update-server` 呼び出し時にこのパラメータに空の値を指定することで、そのロググループをクリアし、構造化ロギングを事実上無効にすることができます。例:

```
update-server --server-id s-1234567890abcdef0 --structured-log-destinations
```

タイプ：文字列の配列

配列メンバー: 最小数は 0 項目です。最大数は 1 項目です。

長さの制限: 最小長は 20 です。最大長は 1600 です。

Pattern: `arn:\S+`

必須: いいえ

Tags

サーバーのグループ化および検索に使用できるキーバリューペア。

型: Tag オブジェクトの配列

配列メンバー：最小数は 1 項目です。最大数は 50 項目です。

必須: いいえ

WorkflowDetails

割り当てるワークフローのワークフロー ID とワークフローの実行に使用する実行ロールを指定します。

ファイルのアップロード完了時に実行するワークフローに加えて、部分的なアップロードで実行するワークフローのワークフロー ID (および実行ロール) も WorkflowDetails に含めることができます。部分的なアップロードは、ファイルのアップロード中にサーバーセッションが切断されたときに発生します。

タイプ：WorkflowDetails オブジェクト

必須: いいえ

レスポンスの構文

```
{
   "ServerId": "string"
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

ServerId

作成されるサーバーのサービス割り当て識別子。

型: 文字列

長さの制限: 固定長は 19 です。

パターン：s-([0-9a-f]{17})

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

AccessDeniedException

このアクションを実行する十分なアクセス権限がありません。

HTTP ステータスコード：400

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceExistsException

要求されたリソースは存在しないか、コマンドに指定されたリージョン以外のリージョンに存在します。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

ThrottlingException

リクエストのスロットリングにより、リクエストが拒否されました。

HTTP ステータスコード：400

例

例

次の例では、VPC_ENDPOINT を使用して新しいサーバーを作成します。

リクエスト例

```
{
```

```
   "EndpointType": "VPC",
   "EndpointDetails":...,
   "HostKey": "Your RSA private key",
   "IdentityProviderDetails": "IdentityProvider",
   "IdentityProviderType": "SERVICE_MANAGED",
   "LoggingRole": "CloudWatchLoggingRole",
   "Tags": [
      {
         "Key": "Name",
         "Value": "MyServer"
      }
   ]
}
```

例

これは、この API コールに対するレスポンスのサンプルです。

レスポンス例

```
{
   "ServerId": "s-01234567890abcdef"
}
```

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

CreateUser

ユーザーを作成し、既存のファイル転送プロトコル対応サーバーと関連付けます。IdentityProviderType が SERVICE_MANAGED に設定されたサーバーのみを作成し、ユーザーを関連付けることができます。のパラメータを使用してCreateUser、ユーザー名の指定、ホームディレクトリの設定、ユーザーのパブリックキーの保存、ユーザーの AWS Identity and Access Management (IAM) ロールの割り当てを行うことができます。オプションで、セッションポリシーの追加、ユーザーのグループ化や検索に使用できるタグがあるメタデータの割り当てが可能です。

リクエストの構文

```
{
   "HomeDirectory": "string",
   "HomeDirectoryMappings": [
      {
         "Entry": "string",
         "Target": "string",
         "Type": "string"
      }
   ],
   "HomeDirectoryType": "string",
   "Policy": "string",
   "PosixProfile": {
      "Gid": number,
      "SecondaryGids": [ number ],
      "Uid": number
   },
   "Role": "string",
   "ServerId": "string",
   "SshPublicKeyBody": "string",
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
      }
   ],
   "UserName": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

HomeDirectory

ユーザーがクライアントを使用してサーバーにログインするときの、ユーザーのランディングディレクトリ (フォルダ)。

HomeDirectory の例は /bucket_name/home/mydirectory です。

> **Note**
>
> HomeDirectory パラメータは、HomeDirectoryType が PATH に設定されている場合のみ使用されます。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 1,024 です。

パターン: (|/.*)

必須: いいえ

HomeDirectoryMappings

ユーザーに表示する Amazon S3 または Amazon EFS のパスとキー、およびそれらをどのように表示するかを指定する論理ディレクトリマッピング。Entry と Target のペアを指定する必要があり、Entry はパスの表示方法を示し、Target は実際の Amazon S3 または Amazon EFS のパスです。ターゲットを指定しただけの場合は、そのまま表示されます。また、 AWS Identity and Access Management (IAM) ロールが のパスへのアクセスを提供することを確認する必要がありますTarget。この値は、HomeDirectoryType がLOGICALに設定されている場合にのみ設定できます。

以下は Entry と Target のペアの例です。

```
[ { "Entry": "/directory1", "Target": "/bucket_name/home/
mydirectory" } ]
```

ほとんどの場合、セッションポリシーの代わりにこの値を使用することで、指定されたホームディレクトリ (「chroot」) にユーザーをロックダウンできます。これを行うには、Entry を / に設定し、Target をユーザーがログインしたときに表示されるホームディレクトリの値に設定すればよいです。

以下は、chroot についての Entry と Target のペアの例です。

```
[ { "Entry": "/", "Target": "/bucket_name/home/mydirectory" } ]
```

型: HomeDirectoryMapEntry オブジェクトの配列

配列メンバー：最小数は 1 項目です。最大数は 50,000 項目です。

必須: いいえ

HomeDirectoryType

ユーザーがサーバーにログインするときにホームディレクトリにするランディングディレクトリ (フォルダ) のタイプ。これを PATH に設定した場合、ユーザーには、ファイル転送プロトコルクライアントに、絶対的な Amazon S3 バケットまたは現状の Amazon EFS パスが表示されます。これを LOGICAL に設定した場合、Amazon S3 または Amazon EFS パスをユーザーに表示する方法に関して、HomeDirectoryMappings でマッピングを指定する必要があります。

> **Note**
>
> HomeDirectoryType が LOGICAL の場合は、HomeDirectoryMappings パラメータを使用してマッピングを指定する必要があります。一方、HomeDirectoryType が PATH の場合は、HomeDirectory パラメータを使用して絶対パスを指定します。テンプレートに HomeDirectory と HomeDirectoryMappings の両方を含めることはできません。

型: 文字列

有効な値：PATH | LOGICAL

必須：いいえ

Policy

複数のユーザーで同じ AWS Identity and Access Management (IAM) ロールを使用できるようにするためのユーザーのセッションポリシー。このポリシーは、ユーザーアクセスのス

コープを Amazon S3 バケットの一部に絞り込みます。このポリシー内に使用できる変数には、`${Transfer:UserName}`、`${Transfer:HomeDirectory}`、`${Transfer:HomeBucket}` があります。

> **Note**
>
> このポリシーは、`ServerId` ドメインが Amazon S3 の場合にのみ適用されます。Amazon EFS はセッション・ポリシーを使用しません。
> セッションポリシーの場合、 はポリシーの Amazon リソースネーム (ARN) ではなく、ポリシーを JSON BLOB として AWS Transfer Family 保存します。JSON blob としてポリシーを保存し、`Policy` 因数に渡します。
> セッションポリシーの例については、「セッションポリシーの例」を参照してください。
> 詳細については、 AWS 「 Security Token Service API リファレンスAssumeRole」の「」を参照してください。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 2048 です。

必須: いいえ

PosixProfile

ユーザーの Amazon EFS ファイルシステムへのアクセスを制御する、ユーザー ID (`Uid`)、グループ ID (`Gid`)、およびセカンダリグループ ID (`SecondaryGids`) を含む完全 POSIX ID を指定します。Amazon EFS 内のファイルとディレクトリに設定される POSIX アクセス許可によって、Amazon EFS ファイルシステムとの間でファイルを転送するときにユーザーが得るアクセスのレベルが決まります。

型: PosixProfile オブジェクト

必須: いいえ

Role

Amazon S3 バケットまたは Amazon EFS ファイルシステムへのユーザーのアクセスを制御する AWS Identity and Access Management (IAM) ロールの Amazon リソースネーム (ARN)。このロールにアタッチされたポリシーにより、ファイルを Amazon S3 バケットまたは Amazon EFS ファイルシステム間で転送する際の、ユーザーに付与するアクセスレベルが決定されます。IAM

ロールには、ユーザーの転送リクエストを処理する際に、サーバーによるリソースへのアクセスを許可する信頼関係も含まれる必要があります。

型: 文字列

長さの制限: 最小長は 20 です。最大長は 2,048 です。

Pattern: `arn:.*role/\S+`

必須：はい

ServerId

サーバーインスタンスにシステムで割り当てられた一意の識別子。これはユーザーを追加したサーバーに固有です。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `s-([0-9a-f]{17})`

必須：はい

SshPublicKeyBody

ユーザーをサーバーに認証する際に使用される Secure Shell (SSH) キーのパブリック部分。

SSH パブリックキーフォーマットの標準要素は `<key type>`、`<body base64>`、そしてオプションの `<comment>` の 3 つで、各要素の間にスペースがあります。

AWS Transfer Family は、RSA、ECDSA、ED25519 キーを受け入れます。

- RSA キーの場合、キータイプは `ssh-rsa` です。
- ED25519 キーの場合、キータイプは `ssh-ed25519` です。
- ECDSA キーの場合、キータイプは、生成したキーのサイズに応じて `ecdsa-sha2-nistp256`、`ecdsa-sha2-nistp384` または `ecdsa-sha2-nistp521` のいずれかになります。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 2048 です。

必須: いいえ

Tags

ユーザーのグループ化および検索に使用できるキーと値のペア。タグは、あらゆる目的でユーザーにアタッチされるメタデータです。

型: Tag オブジェクトの配列

配列メンバー：最小数は 1 項目です。最大数は 50 項目です。

必須: いいえ

UserName

ユーザーを識別する一意の文字列であり、ServerId に関連付けられています。このユーザー名は、最小 3 文字、最大 100 文字にする必要があります。有効な文字は a~z、A~Z、0~9、アンダースコア (_)、ハイフン (-)、ピリオド (。)、アットマーク (@) です。ユーザー名をハイフン、ピリオド、アットマークで始めることはできません。

型: 文字列

長さの制限: 最小長は 3 です。最大長は 100 です。

パターン: `[\w][\w@.-]{2,99}`

必須：はい

レスポンスの構文

```
{
   "ServerId": "string",
   "UserName": "string"
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

ServerId

ユーザーが接続しているサーバーの識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `s-([0-9a-f]{17})`

UserName

Transfer Family ユーザーを識別する一意の文字列。

型: 文字列

長さの制限: 最小長は 3 です。最大長は 100 です。

パターン：`[\w][\w@.-]{2,99}`

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceExistsException

要求されたリソースは存在しないか、コマンドに指定されたリージョン以外のリージョンに存在します。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

例

例

ユーザを作成するには、まずパラメータを createUserParameters などのJSONファイルに保存してから、create-user APIコマンドを実行します。

```
{
    "HomeDirectory": "/DOC-EXAMPLE-BUCKET",
    "HomeDirectoryType": "PATH",
    "Role": "arn:aws:iam::111122223333:role/bob-role",
    "ServerId": "s-1111aaaa2222bbbb3",
    "SshPublicKeyBody": "ecdsa-sha2-nistp521 AAAAE2VjZHNhLXNoYTItbmlzdHA...
 bobusa@mycomputer.us-east-1.amazon.com",
    "UserName": "bobusa-API"
}
```

リクエスト例

```
aws transfer create-user --cli-input-json file://createUserParameters
```

レスポンス例

```
{
    "ServerId": ""s-1111aaaa2222bbbb3",
    "UserName": "bobusa-API"
}
```

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス

- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

CreateWorkflow

ファイル転送完了後にワークフローが呼び出す、指定されたステップとステップの詳細に従って、ワークフローを作成します。ワークフローの作成後に、CreateServer および UpdateServer のオペレーションで workflow-details フィールドを指定することで、作成したワークフローを任意の転送サーバーに関連付けることができます。

リクエストの構文

```
{
   "Description": "string",
   "OnExceptionSteps": [
      {
         "CopyStepDetails": {
            "DestinationFileLocation": {
               "EfsFileLocation": {
                  "FileSystemId": "string",
                  "Path": "string"
               },
               "S3FileLocation": {
                  "Bucket": "string",
                  "Key": "string"
               }
            },
            "Name": "string",
            "OverwriteExisting": "string",
            "SourceFileLocation": "string"
         },
         "CustomStepDetails": {
            "Name": "string",
            "SourceFileLocation": "string",
            "Target": "string",
            "TimeoutSeconds": number
         },
         "DecryptStepDetails": {
            "DestinationFileLocation": {
               "EfsFileLocation": {
                  "FileSystemId": "string",
                  "Path": "string"
               },
               "S3FileLocation": {
                  "Bucket": "string",
                  "Key": "string"
```

```
            }
         },
         "Name": "string",
         "OverwriteExisting": "string",
         "SourceFileLocation": "string",
         "Type": "string"
      },
      "DeleteStepDetails": {
         "Name": "string",
         "SourceFileLocation": "string"
      },
      "TagStepDetails": {
         "Name": "string",
         "SourceFileLocation": "string",
         "Tags": [
            {
               "Key": "string",
               "Value": "string"
            }
         ]
      },
      "Type": "string"
   }
],
"Steps": [
   {
      "CopyStepDetails": {
         "DestinationFileLocation": {
            "EfsFileLocation": {
               "FileSystemId": "string",
               "Path": "string"
            },
            "S3FileLocation": {
               "Bucket": "string",
               "Key": "string"
            }
         },
         "Name": "string",
         "OverwriteExisting": "string",
         "SourceFileLocation": "string"
      },
      "CustomStepDetails": {
         "Name": "string",
         "SourceFileLocation": "string",
```

```
            "Target": "string",
            "TimeoutSeconds": number
         },
         "DecryptStepDetails": {
            "DestinationFileLocation": {
               "EfsFileLocation": {
                  "FileSystemId": "string",
                  "Path": "string"
               },
               "S3FileLocation": {
                  "Bucket": "string",
                  "Key": "string"
               }
            },
            "Name": "string",
            "OverwriteExisting": "string",
            "SourceFileLocation": "string",
            "Type": "string"
         },
         "DeleteStepDetails": {
            "Name": "string",
            "SourceFileLocation": "string"
         },
         "TagStepDetails": {
            "Name": "string",
            "SourceFileLocation": "string",
            "Tags": [
               {
                  "Key": "string",
                  "Value": "string"
               }
            ]
         },
         "Type": "string"
      }
   ],
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
      }
   ]
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

Description

ワークフローの説明テキスト。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 256 です。

パターン：`[\w- ]*`

必須: いいえ

OnExceptionSteps

ワークフローの実行中にエラーが発生した場合に実行する手順 (アクション) を指定します。

> **Note**
> カスタムステップの場合、Lambda 関数で FAILURE をコールバック API に送信して例外ステップを開始する必要があります。さらに、タイムアウトになる前に Lambda が SUCCESS を送信しない場合、例外ステップが実行されます。

タイプ：WorkflowStep オブジェクトの配列

配列メンバー：最小数は 0 項目です。最大 8 項目。

必須: いいえ

Steps

指定したワークフローに含まれるステップの詳細を指定します。

TYPE では、このステップで実行するものを以下のアクションから指定します。

- **COPY** - ファイルを別の場所にコピーします。

- **CUSTOM** - AWS Lambda 関数ターゲットを使用してカスタムステップを実行します。
- **DECRYPT** - アップロード前に暗号化されていたファイルを復号化します。
- **DELETE** - ファイルを削除します。
- **TAG** - ファイルにタグを追加します。

> **Note**
>
> 現在、コピーとタグ付けは S3 でのみサポートされています。

ファイルの場所として、Amazon S3 バケットとキー、または Amazon EFS ファイルシステム ID とパスのいずれかを指定します。

タイプ：WorkflowStep オブジェクトの配列

配列メンバー：最小数は 0 項目です。最大 8 項目。

必須: はい

Tags

ワークフローのグループ化および検索に使用できるキーバリューペア。タグとは、任意の目的でユーザーにアタッチされるメタデータです。

型: Tag オブジェクトの配列

配列メンバー：最小数は 1 項目です。最大数は 50 項目です。

必須：いいえ

レスポンスの構文

```
{
   "WorkflowId": "string"
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

WorkflowId

ワークフローの一意の識別子。

型: 文字列

長さの制限: 固定長は 19 です。

パターン：`w-([a-z0-9]{17})`

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

AccessDeniedException

このアクションを実行する十分なアクセス権限がありません。

HTTP ステータスコード：400

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceExistsException

要求されたリソースは存在しないか、コマンドに指定されたリージョン以外のリージョンに存在します。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

ThrottlingException

リクエストのスロットリングにより、リクエストが拒否されました。

HTTP ステータスコード：400

例

例

次の例のように、ワークフローステップ情報をテキストファイルに保存し、そのファイルを使用してワークフローを作成できます。次の例では、ワークフローステップを *example-file.json* (コマンドを実行した場所と同じフォルダ内) に保存し、ワークフローをバージニア北部 (us-east-1) リージョンで作成することを前提とします。

```
aws transfer create-workflow --description "example workflow from a file" --steps
 file://example-file.json --region us-east-1
```

```
// Example file containing workflow steps
[
   {
     "Type": "TAG",
     "TagStepDetails": {
        "Name": "TagStep",
        "Tags": [
          {
            "Key": "name",
            "Value": "testTag"
          }
        ]
     }
   },
   {
     "Type": "COPY",
     "CopyStepDetails": {
        "Name": "CopyStep",
        "DestinationFileLocation": {
        "S3FileLocation": {
           "Bucket": "DOC-EXAMPLE-BUCKET",
```

```
            "Key": "DOC-EXAMPLE-KEY/"
                   }
        },
        "OverwriteExisting": "TRUE",
        "SourceFileLocation": "${original.file}"
      }
    },
    {
      "Type": "DELETE",
      "DeleteStepDetails":{
      "Name":"DeleteStep",
      "SourceFileLocation": "${original.file}"
      }
    }
]
```

例

CreateWorkflow コールにより、新しいワークフローのワークフロー ID が返されます。

レスポンス例

```
{
    "WorkflowId": "w-1234abcd5678efghi"
}
```

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python

- AWS SDK for Ruby V3

DeleteAccess

ServerID および ExternalID のパラメータで指定されたアクセス権を削除できます。

リクエストの構文

```
{
   "ExternalId": "string",
   "ServerId": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

ExternalId

ディレクトリ内の特定のグループを識別するために必要な一意の識別子。関連付けるグループのユーザーは、 を使用して、有効なプロトコル経由で Amazon S3 または Amazon EFS リソースにアクセスできます AWS Transfer Family。グループ名がわかっている場合は、Windows を使用して次のコマンドを実行して SID 値を表示できます PowerShell。

```
Get-ADGroup -Filter {samAccountName -like "YourGroupName*"} -Properties
* | Select SamAccountName,ObjectSid
```

そのコマンドで、 を Active Directory グループの名前YourGroupNameに置き換えます。

このパラメータの検証に使用される正規表現は、スペースを含まない大文字と小文字の英数字からなる文字列です。下線文字 (_) や =,.@:/- の文字を含めることもできます。

型: 文字列

長さの制限：最小長は 1 です。最大長は 256 です。

パターン：S-1-[\d-]+

必須：はい

ServerId

このユーザーが割り当てられたサーバーにシステムで割り当てられた一意の識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `s-([0-9a-f]{17})`

必須：はい

レスポンス要素

アクションが成功した場合、サービスは空の HTTP 本文を持つ HTTP 200 レスポンスを返します。

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DeleteAgreement

提供されている AgreementId の契約書を削除します。

リクエストの構文

```
{
   "AgreementId": "string",
   "ServerId": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

AgreementId

契約の一意の識別子。この識別子は、契約を作成するときに返されます。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: a-([0-9a-f]{17})

必須：はい

ServerId

削除する契約に関連付けられているサーバー識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: s-([0-9a-f]{17})

必須：はい

レスポンス要素

アクションが成功した場合、サービスは空の HTTP 本文を持つ HTTP 200 レスポンスを返します。

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2

- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DeleteCertificate

CertificateId パラメータで指定された証明書を削除します。

リクエストの構文

```
{
   "CertificateId": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

CertificateId

削除する証明書オブジェクトの識別子。

型: 文字列

長さの制限: 22 の固定長

Pattern: cert-([0-9a-f]{17})

必須：はい

レスポンス要素

アクションが成功した場合、サービスは空の HTTP 本文を持つ HTTP 200 レスポンスを返します。

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DeleteConnector

提供された ConnectorId で指定されたコネクタを削除します。

リクエストの構文

```
{
   "ConnectorId": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

ConnectorId

コネクタの一意の識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: c-([0-9a-f]{17})

必須：はい

レスポンス要素

アクションが成功した場合、サービスは空の HTTP 本文を持つ HTTP 200 レスポンスを返します。

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DeleteHostKey

HostKeyId パラメータで指定されたホストキーを削除します。

リクエストの構文

```
{
   "HostKeyId": "string",
   "ServerId": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

HostKeyId

削除するホストキーの識別子。

型: 文字列

長さの制限: 固定長は 25 です。

Pattern: hostkey-[0-9a-f]{17}

必須：はい

ServerId

削除するホストキーを含むサーバーの識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: s-([0-9a-f]{17})

必須：はい

レスポンス要素

アクションが成功した場合、サービスは空の HTTP 本文を持つ HTTP 200 レスポンスを返します。

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

ThrottlingException

リクエストのスロットリングにより、リクエストが拒否されました。

HTTP ステータスコード：400

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス

- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DeleteProfile

ProfileIdパラメータで指定されたプロファイルを削除します。

リクエストの構文

```
{
   "ProfileId": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

ProfileId

削除するプロファイルの ID。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: p-([0-9a-f]{17})

必須：はい

レスポンス要素

アクションが成功した場合、サービスは空の HTTP 本文を持つ HTTP 200 レスポンスを返します。

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DeleteServer

指定したファイル転送プロトコル対応サーバーを削除します。

このオペレーションからレスポンスは返りません。

リクエストの構文

```
{
   "ServerId": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

ServerId

サーバーインスタンスにシステムで割り当てられた一意の識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `s-([0-9a-f]{17})`

必須：はい

レスポンス要素

アクションが成功した場合、サービスは空の HTTP 本文を持つ HTTP 200 レスポンスを返します。

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

AccessDeniedException

このアクションを実行する十分なアクセス権限がありません。

HTTP ステータスコード：400

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

例

例

次の例では、サーバーを削除します。

リクエスト例

```
{
   "ServerId": "s-01234567890abcdef"
}
```

例

成功した場合、何も返されません。

レスポンス例

```
{
}
```

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DeleteSshPublicKey

ユーザーの Secure Shell (SSH) パブリックキーを削除します。

リクエストの構文

```
{
   "ServerId": "string",
   "SshPublicKeyId": "string",
   "UserName": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

ServerId

ファイル転送プロトコル対応サーバーインスタンスについてシステムで割り当てられた一意の識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `s-([0-9a-f]{17})`

必須：はい

SshPublicKeyId

ユーザーの特定の SSH キーを参照するために使用される一意の識別子。

型: 文字列

長さの制限: 固定長は 21 です。

Pattern: `key-[0-9a-f]{17}`

必須：はい

UserName

パブリックキーが削除されたユーザーを識別する一意の文字列。

型: 文字列

長さの制限: 最小長は 3 です。最大長は 100 です。

パターン: `[\w][\w@.-]{2,99}`

必須：はい

レスポンス要素

アクションが成功した場合、サービスは空の HTTP 本文を持つ HTTP 200 レスポンスを返します。

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

ThrottlingException

リクエストのスロットリングにより、リクエストが拒否されました。

HTTP ステータスコード：400

例

例

次の例では、ユーザーの SSH パブリックキーを削除します。

リクエスト例

```
{
   "ServerId": "s-01234567890abcdef",
   "SshPublicKeyId": "MyPublicKey",
   "UserName": "my_user"
}
```

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DeleteUser

指定したファイル転送プロトコル対応サーバーに属するユーザーを削除します。

このオペレーションからレスポンスは返りません。

> **Note**
>
> サーバーからユーザーを削除すると、ユーザーの情報は失われます。

リクエストの構文

```
{
   "ServerId": "string",
   "UserName": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

ServerId

ユーザーの割り当て先サーバーインスタンスにシステムで割り当てられた一意の識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `s-([0-9a-f]{17})`

必須：はい

UserName

サーバーから削除されるユーザーを識別する一意の文字列。

型: 文字列

長さの制限: 最小長は 3 です。最大長は 100 です。

パターン: `[\w][\w@.-]{2,99}`

必須：はい

レスポンス要素

アクションが成功した場合、サービスは空の HTTP 本文を持つ HTTP 200 レスポンスを返します。

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

例

例

次の例では、Transfer Family ユーザーを削除します。

リクエスト例

```
{
   "ServerId": "s-01234567890abcdef",
   "UserNames": "my_user"
}
```

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DeleteWorkflow

指定されたワークフローを削除します。

リクエストの構文

```
{
   "WorkflowId": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

WorkflowId

ワークフローの一意の識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `w-([a-z0-9]{17})`

必須：はい

レスポンス要素

アクションが成功した場合、サービスは空の HTTP 本文を持つ HTTP 200 レスポンスを返します。

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

AccessDeniedException

このアクションを実行する十分なアクセス権限がありません。

HTTP ステータスコード：400

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DescribeAccess

特定のファイル転送プロトコル対応サーバー(その ServerId プロパティとその ExternalId で識別) に割り当てられたアクセス権について説明します。

この呼び出しからのレスポンスは、指定した ServerId に関連付けられたアクセス権のプロパティを返します。

リクエストの構文

```
{
   "ExternalId": "string",
   "ServerId": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

ExternalId

ディレクトリ内の特定のグループを識別するために必要な一意の識別子。関連付けるグループのユーザーは、 を使用して、有効なプロトコル経由で Amazon S3 または Amazon EFS リソースにアクセスできます AWS Transfer Family。グループ名がわかっている場合は、Windows を使用して次のコマンドを実行して SID 値を表示できます PowerShell。

```
Get-ADGroup -Filter {samAccountName -like "YourGroupName*"} -Properties
* | Select SamAccountName,ObjectSid
```

そのコマンドで、 を Active Directory グループの名前YourGroupNameに置き換えます。

このパラメータの検証に使用される正規表現は、スペースを含まない大文字と小文字の英数字からなる文字列です。下線文字 (_) や =,.@:/- の文字を含めることもできます。

型: 文字列

長さの制限：最小長は 1 です。最大長は 256 です。

パターン：`S-1-[\d-]+`

必須：はい

ServerId

このアクセス権が割り当てられたサーバーにシステムで割り当てられた一意の識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `s-([0-9a-f]{17})`

必須：はい

レスポンスの構文

```
{
   "Access": {
      "ExternalId": "string",
      "HomeDirectory": "string",
      "HomeDirectoryMappings": [
         {
            "Entry": "string",
            "Target": "string",
            "Type": "string"
         }
      ],
      "HomeDirectoryType": "string",
      "Policy": "string",
      "PosixProfile": {
         "Gid": number,
         "SecondaryGids": [ number ],
         "Uid": number
      },
      "Role": "string"
   },
   "ServerId": "string"
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

Access

アクセスが接続されているサーバーの外部識別子。

型: DescribedAccess オブジェクト

ServerId

このアクセス権が割り当てられたサーバーにシステムで割り当てられた一意の識別子。

型: 文字列

長さの制限: 固定長は 19 です。

パターン：`s-([0-9a-f]{17})`

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DescribeAgreement

AgreementId によって識別される契約について説明します。

リクエストの構文

```
{
   "AgreementId": "string",
   "ServerId": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

AgreementId

契約の一意の識別子。この識別子は、契約を作成するときに返されます。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: a-([0-9a-f]{17})

必須：はい

ServerId

契約に関連付けられたサーバー識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: s-([0-9a-f]{17})

必須：はい

レスポンスの構文

```
{
   "Agreement": {
      "AccessRole": "string",
      "AgreementId": "string",
      "Arn": "string",
      "BaseDirectory": "string",
      "Description": "string",
      "LocalProfileId": "string",
      "PartnerProfileId": "string",
      "ServerId": "string",
      "Status": "string",
      "Tags": [
         {
            "Key": "string",
            "Value": "string"
         }
      ]
   }
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

Agreement

指定された契約の詳細は、DescribedAgreement オブジェクトとして返されます。

型: DescribedAgreement オブジェクト

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DescribeCertificate

`CertificateId` によって識別される証明書について説明します。

リクエストの構文

```
{
   "CertificateId": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

CertificateId

インポートされた証明書の識別子の配列です。この識別子は、プロファイルやパートナープロファイルの操作に使用します。

型: 文字列

長さの制限: 22 の固定長

Pattern: `cert-([0-9a-f]{17})`

必須：はい

レスポンスの構文

```
{
   "Certificate": {
      "ActiveDate": number,
      "Arn": "string",
      "Certificate": "string",
      "CertificateChain": "string",
      "CertificateId": "string",
      "Description": "string",
```

```
      "InactiveDate": number,
      "NotAfterDate": number,
      "NotBeforeDate": number,
      "Serial": "string",
      "Status": "string",
      "Tags": [
         {
            "Key": "string",
            "Value": "string"
         }
      ],
      "Type": "string",
      "Usage": "string"
   }
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

Certificate

指定された証明書の詳細は、オブジェクトとして返されます。

型: DescribedCertificate オブジェクト

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DescribeConnector

ConnectorId. で識別されるコネクタについて説明します。

リクエストの構文

```
{
   "ConnectorId": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

ConnectorId

コネクタの一意の識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: c-([0-9a-f]{17})

必須：はい

レスポンスの構文

```
{
   "Connector": {
      "AccessRole": "string",
      "Arn": "string",
      "As2Config": {
         "BasicAuthSecretId": "string",
         "Compression": "string",
         "EncryptionAlgorithm": "string",
         "LocalProfileId": "string",
         "MdnResponse": "string",
         "MdnSigningAlgorithm": "string",
```

```
         "MessageSubject": "string",
         "PartnerProfileId": "string",
         "SigningAlgorithm": "string"
      },
      "ConnectorId": "string",
      "LoggingRole": "string",
      "SecurityPolicyName": "string",
      "ServiceManagedEgressIpAddresses": [ "string" ],
      "SftpConfig": {
         "TrustedHostKeys": [ "string" ],
         "UserSecretId": "string"
      },
      "Tags": [
         {
            "Key": "string",
            "Value": "string"
         }
      ],
      "Url": "string"
   }
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

Connector

コネクターの詳細を含む構造体。

型: DescribedConnector オブジェクト

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DescribeExecution

DescribeExecution を使用して、指定したワークフローの実行の詳細を確認します。

> **Note**
>
> この API コールは、進行中のワークフローの詳細のみを返します。
> 実行中でない実行に対してIDを提供するか、実行が指定したワークフローIDと一致しない場合、ResourceNotFound 例外が発生します。

リクエストの構文

```
{
   "ExecutionId": "string",
   "WorkflowId": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

ExecutionId

ワークフローの一意の識別子。

型: 文字列

長さの制限: 最大長は 36 です。

パターン: [0-9a-fA-F]{8}\-[0-9a-fA-F]{4}\-[0-9a-fA-F]{4}\-[0-9a-fA-F]{4}\-[0-9a-fA-F]{12}

必須：はい

WorkflowId

ワークフローの一意の識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `w-([a-z0-9]{17})`

必須：はい

レスポンスの構文

```
{
   "Execution": {
      "ExecutionId": "string",
      "ExecutionRole": "string",
      "InitialFileLocation": {
         "EfsFileLocation": {
            "FileSystemId": "string",
            "Path": "string"
         },
         "S3FileLocation": {
            "Bucket": "string",
            "Etag": "string",
            "Key": "string",
            "VersionId": "string"
         }
      },
      "LoggingConfiguration": {
         "LoggingRole": "string",
         "LogGroupName": "string"
      },
      "PosixProfile": {
         "Gid": number,
         "SecondaryGids": [ number ],
         "Uid": number
      },
      "Results": {
         "OnExceptionSteps": [
            {
               "Error": {
                  "Message": "string",
                  "Type": "string"
               },
               "Outputs": "string",
```

```
                "StepType": "string"
            }
         ],
         "Steps": [
            {
               "Error": {
                  "Message": "string",
                  "Type": "string"
               },
               "Outputs": "string",
               "StepType": "string"
            }
         ]
      },
      "ServiceMetadata": {
         "UserDetails": {
            "ServerId": "string",
            "SessionId": "string",
            "UserName": "string"
         }
      },
      "Status": "string"
   },
   "WorkflowId": "string"
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

Execution

ワークフローの実行の詳細を含む構造。

型: DescribedExecution オブジェクト

WorkflowId

ワークフローの一意の識別子。

型: 文字列

長さの制限: 固定長は 19 です。

パターン：w-([a-z0-9]{17})

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2

- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DescribeHostKey

HostKeyId と ServerId で指定されたホスト キーの詳細を返します。

リクエストの構文

```
{
   "HostKeyId": "string",
   "ServerId": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

HostKeyId

説明するホストキーの識別子。

型: 文字列

長さの制限: 固定長は 25 です。

Pattern: hostkey-[0-9a-f]{17}

必須：はい

ServerId

説明するホストキーを含むサーバーの識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: s-([0-9a-f]{17})

必須：はい

レスポンスの構文

```
{
   "HostKey": {
      "Arn": "string",
      "DateImported": number,
      "Description": "string",
      "HostKeyFingerprint": "string",
      "HostKeyId": "string",
      "Tags": [
         {
            "Key": "string",
            "Value": "string"
         }
      ],
      "Type": "string"
   }
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

HostKey

指定されたホストキーの詳細を返します。

型: DescribedHostKey オブジェクト

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DescribeProfile

ProfileId で指定されたプロファイルの詳細を返します。

リクエストの構文

```
{
   "ProfileId": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

ProfileId

説明するプロファイルの識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: p-([0-9a-f]{17})

必須：はい

レスポンスの構文

```
{
   "Profile": {
      "Arn": "string",
      "As2Id": "string",
      "CertificateIds": [ "string" ],
      "ProfileId": "string",
      "ProfileType": "string",
      "Tags": [
         {
            "Key": "string",
```

```
            "Value": "string"
         }
      ]
   }
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

Profile

指定されたプロファイルの詳細。オブジェクトとして返されます。

型: DescribedProfile オブジェクト

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DescribeSecurityPolicy

サーバーまたは SFTP コネクタにアタッチされているセキュリティポリシーについて説明します。レスポンスには、セキュリティポリシーのプロパティの説明が含まれます。セキュリティポリシーの詳細については、「サーバーのセキュリティポリシーの使用」または「SFTP コネクタのセキュリティポリシーの使用」を参照してください。

リクエストの構文

```
{
   "SecurityPolicyName": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

SecurityPolicyName

詳細を取得するセキュリティポリシーのテキスト名を指定します。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 100 です。

パターン: `Transfer[A-Za-z0-9]*SecurityPolicy-[A-Za-z0-9-]+`

必須：はい

レスポンスの構文

```
{
   "SecurityPolicy": {
      "Fips": boolean,
      "Protocols": [ "string" ],
      "SecurityPolicyName": "string",
      "SshCiphers": [ "string" ],
      "SshHostKeyAlgorithms": [ "string" ],
```

```
        "SshKexs": [ "string" ],
        "SshMacs": [ "string" ],
        "TlsCiphers": [ "string" ],
        "Type": "string"
    }
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

SecurityPolicy

セキュリティポリシーのプロパティを含む配列。

型: DescribedSecurityPolicy オブジェクト

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

例

例

次のコマンド例は、セキュリティポリシー名を引数として受け取り、指定されたセキュリティポリシーのアルゴリズムを返します。

リクエスト例

```
aws transfer describe-security-policy --security-policy-name "TransferSecurityPolicy-FIPS-2023-05"
```

レスポンス例

```
{
    "SecurityPolicy": {
        "Fips": true,
        "SecurityPolicyName": "TransferSecurityPolicy-FIPS-2023-05",
        "SshCiphers": [
            "aes256-gcm@openssh.com",
            "aes128-gcm@openssh.com",
            "aes256-ctr",
            "aes192-ctr"
        ],
        "SshKexs": [
            "diffie-hellman-group16-sha512",
            "diffie-hellman-group18-sha512",
            "diffie-hellman-group-exchange-sha256"
        ],
        "SshMacs": [
            "hmac-sha2-256-etm@openssh.com",
            "hmac-sha2-512-etm@openssh.com"
        ],
        "TlsCiphers": [
            "TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256",
            "TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256",
            "TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256",
            "TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256",
            "TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384",
            "TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384",
```

```
            "TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384",
            "TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384"
        ]
    }
}
```

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DescribeServer

ServerId パラメータを渡すことで指定したファイル転送プロトコル対応サーバーについて説明します。

レスポンスには、サーバーのプロパティの説明が含まれます。EndpointType を VPCに設定すると、レスポンスに EndpointDetails が含まれます。

リクエストの構文

```
{
   "ServerId": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

ServerId

システムでサーバーに割り当てられた一意の識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: s-([0-9a-f]{17})

必須：はい

レスポンスの構文

```
{
   "Server": {
      "Arn": "string",
      "As2ServiceManagedEgressIpAddresses": [ "string" ],
      "Certificate": "string",
      "Domain": "string",
      "EndpointDetails": {
```

```
      "AddressAllocationIds": [ "string" ],
      "SecurityGroupIds": [ "string" ],
      "SubnetIds": [ "string" ],
      "VpcEndpointId": "string",
      "VpcId": "string"
   },
   "EndpointType": "string",
   "HostKeyFingerprint": "string",
   "IdentityProviderDetails": {
      "DirectoryId": "string",
      "Function": "string",
      "InvocationRole": "string",
      "SftpAuthenticationMethods": "string",
      "Url": "string"
   },
   "IdentityProviderType": "string",
   "LoggingRole": "string",
   "PostAuthenticationLoginBanner": "string",
   "PreAuthenticationLoginBanner": "string",
   "ProtocolDetails": {
      "As2Transports": [ "string" ],
      "PassiveIp": "string",
      "SetStatOption": "string",
      "TlsSessionResumptionMode": "string"
   },
   "Protocols": [ "string" ],
   "S3StorageOptions": {
      "DirectoryListingOptimization": "string"
   },
   "SecurityPolicyName": "string",
   "ServerId": "string",
   "State": "string",
   "StructuredLogDestinations": [ "string" ],
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
      }
   ],
   "UserCount": number,
   "WorkflowDetails": {
      "OnPartialUpload": [
         {
            "ExecutionRole": "string",
```

```
                "WorkflowId": "string"
            }
         ],
         "OnUpload": [
            {
               "ExecutionRole": "string",
               "WorkflowId": "string"
            }
         ]
      }
   }
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

Server

指定した ServerID を含むサーバーのプロパティを含む配列。

型: DescribedServer オブジェクト

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

例

例

次の例では、サーバーに割り当てられたプロパティを返します。

リクエスト例

```
{
   "ServerId": "s-01234567890abcdef"
}
```

例

この例では、 の使用例を示します DescribeServer。

レスポンス例

```
{
    "Server": {
        "Arn": "arn:aws:transfer:us-east-1:176354371281:server/s-01234567890abcdef",
        "EndpointDetails": {
            "AddressAllocationIds": [
                "eipalloc-01a2eabe3c04d5678",
                "eipalloc-102345be"
            ],
            "SubnetIds": [
                "subnet-047eaa7f0187a7cde",
                "subnet-0a2d0f474daffde18"
            ],
            "VpcEndpointId": "vpce-03fe0080e7cb008b8",
            "VpcId": "vpc-09047a51f1c8e1634"
        },
```

```
        "EndpointType": "VPC",
        "HostKeyFingerprint": "your host key,
        "IdentityProviderType": "SERVICE_MANAGED",
        "ServerId": "s-01234567890abcdef",
        "State": "ONLINE",
        "Tags": [],
        "UserCount": 0
    }
}
```

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DescribeUser

特定のファイル転送プロトコル対応サーバー(その ServerId プロパティで識別) に割り当てられたユーザーについて説明します。

この呼び出しからのレスポンスは、指定した ServerId に関連付けられたユーザーのプロパティを返します。

リクエストの構文

```
{
   "ServerId": "string",
   "UserName": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

ServerId

このユーザーが割り当てられたサーバーにシステムで割り当てられた一意の識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `s-([0-9a-f]{17})`

必須：はい

UserName

1 台以上のサーバーに割り当てられているユーザーの名前。ユーザー名は、 AWS Transfer Family サービスを使用してファイル転送タスクを実行するためのサインイン認証情報の一部です。

型: 文字列

長さの制限: 最小長は 3 です。最大長は 100 です。

パターン: `[\w][\w@.-]{2,99}`

必須：はい

レスポンスの構文

```
{
   "ServerId": "string",
   "User": {
      "Arn": "string",
      "HomeDirectory": "string",
      "HomeDirectoryMappings": [
         {
            "Entry": "string",
            "Target": "string",
            "Type": "string"
         }
      ],
      "HomeDirectoryType": "string",
      "Policy": "string",
      "PosixProfile": {
         "Gid": number,
         "SecondaryGids": [ number ],
         "Uid": number
      },
      "Role": "string",
      "SshPublicKeys": [
         {
            "DateImported": number,
            "SshPublicKeyBody": "string",
            "SshPublicKeyId": "string"
         }
      ],
      "Tags": [
         {
            "Key": "string",
            "Value": "string"
         }
      ],
      "UserName": "string"
   }
```

```
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

ServerId

このユーザーが割り当てられたサーバーにシステムで割り当てられた一意の識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `s-([0-9a-f]{17})`

User

指定した `ServerID` 値の Transfer Family ユーザーのプロパティを含む配列。

型: DescribedUser オブジェクト

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

例

例

次の例は、既存のユーザーの詳細を示したものです。

リクエスト例

```
aws transfer describe-user --server-id s-1111aaaa2222bbbb3 --user-name bob-test
```

レスポンス例

```
{
    "ServerId": "s-1111aaaa2222bbbb3",
    "User": {
        "Arn": "arn:aws:transfer:us-east-1:111122223333:user/s-1111aaaa2222bbbb3/bob-
test",
        "HomeDirectory": "/DOC-EXAMPLE-BUCKET",
        "HomeDirectoryType": "PATH",
        "Role": "arn:aws:iam::111122223333:role/bob-role",
        "SshPublicKeys": [
            {
                "DateImported": "2022-03-31T12:27:52.614000-04:00",
                "SshPublicKeyBody": "ssh-rsa AAAAB3NzaC1yc..... bobusa@mycomputer.us-
east-1.amaazon.com",
                "SshPublicKeyId": "key-abcde12345fghik67"
            }
        ],
        "Tags": [],
        "UserName": "bob-test"
    }
}
```

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DescribeWorkflow

指定されたワークフローについて説明します。

リクエストの構文

```
{
   "WorkflowId": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

WorkflowId

ワークフローの一意の識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `w-([a-z0-9]{17})`

必須：はい

レスポンスの構文

```
{
   "Workflow": {
      "Arn": "string",
      "Description": "string",
      "OnExceptionSteps": [
         {
            "CopyStepDetails": {
               "DestinationFileLocation": {
                  "EfsFileLocation": {
                     "FileSystemId": "string",
                     "Path": "string"
                  },
```

```
            "S3FileLocation": {
               "Bucket": "string",
               "Key": "string"
            }
         },
         "Name": "string",
         "OverwriteExisting": "string",
         "SourceFileLocation": "string"
      },
      "CustomStepDetails": {
         "Name": "string",
         "SourceFileLocation": "string",
         "Target": "string",
         "TimeoutSeconds": number
      },
      "DecryptStepDetails": {
         "DestinationFileLocation": {
            "EfsFileLocation": {
               "FileSystemId": "string",
               "Path": "string"
            },
            "S3FileLocation": {
               "Bucket": "string",
               "Key": "string"
            }
         },
         "Name": "string",
         "OverwriteExisting": "string",
         "SourceFileLocation": "string",
         "Type": "string"
      },
      "DeleteStepDetails": {
         "Name": "string",
         "SourceFileLocation": "string"
      },
      "TagStepDetails": {
         "Name": "string",
         "SourceFileLocation": "string",
         "Tags": [
            {
               "Key": "string",
               "Value": "string"
            }
         ]
```

```
            },
            "Type": "string"
         }
      ],
      "Steps": [
         {
            "CopyStepDetails": {
               "DestinationFileLocation": {
                  "EfsFileLocation": {
                     "FileSystemId": "string",
                     "Path": "string"
                  },
                  "S3FileLocation": {
                     "Bucket": "string",
                     "Key": "string"
                  }
               },
               "Name": "string",
               "OverwriteExisting": "string",
               "SourceFileLocation": "string"
            },
            "CustomStepDetails": {
               "Name": "string",
               "SourceFileLocation": "string",
               "Target": "string",
               "TimeoutSeconds": number
            },
            "DecryptStepDetails": {
               "DestinationFileLocation": {
                  "EfsFileLocation": {
                     "FileSystemId": "string",
                     "Path": "string"
                  },
                  "S3FileLocation": {
                     "Bucket": "string",
                     "Key": "string"
                  }
               },
               "Name": "string",
               "OverwriteExisting": "string",
               "SourceFileLocation": "string",
               "Type": "string"
            },
            "DeleteStepDetails": {
```

```
                "Name": "string",
                "SourceFileLocation": "string"
            },
            "TagStepDetails": {
                "Name": "string",
                "SourceFileLocation": "string",
                "Tags": [
                    {
                        "Key": "string",
                        "Value": "string"
                    }
                ]
            },
            "Type": "string"
        }
    ],
    "Tags": [
        {
            "Key": "string",
            "Value": "string"
        }
    ],
    "WorkflowId": "string"
  }
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

Workflow

ワークフローの詳細を含む構造。

型: DescribedWorkflow オブジェクト

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

ImportCertificate

ローカル (AS2) プロファイルとパートナープロファイルの作成に必要な署名証明書と暗号化証明書をインポートします。

リクエストの構文

```
{
   "ActiveDate": number,
   "Certificate": "string",
   "CertificateChain": "string",
   "Description": "string",
   "InactiveDate": number,
   "PrivateKey": "string",
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
      }
   ],
   "Usage": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

ActiveDate

証明書がいつアクティブになるかを指定するオプションの日付です。

型: タイムスタンプ

必須: いいえ

Certificate

- CLIの場合、証明書のファイルパスを URI 形式で指定します。例えば `--certificate file://encryption-cert.pem` です。または、未加工のコンテンツを使用できます。

- SDK には、証明書ファイルの未加工コンテンツを指定します。例えば `--certificate "`cat encryption-cert.pem`"` です。

タイプ: 文字列

長さの制限: 最小長は 1 です。最大長は 16384 です。

Pattern: `[\u0009\u000A\u000D\u0020-\u00FF]*`

必須：はい

CertificateChain

インポートされる証明書のチェーンを構成する証明書のオプションのリストです。

型: 文字列

長さの制限：最小長は 1 です。最大長は 2097152 です。

Pattern: `[\u0009\u000A\u000D\u0020-\u00FF]*`

必須: いいえ

Description

証明書の識別に役立つ簡単な説明。

型: 文字列

長さの制限: 最小長は 1 です。最大長は 200 です。

Pattern: `[\p{Graph}]+`

必須: いいえ

InactiveDate

証明書がいつ非アクティブになるかを指定するオプションの日付です。

型: タイムスタンプ

必須: いいえ

PrivateKey

- CLIの場合、秘密鍵のファイルパスを URI 形式で指定します。たとえば、`--private-key file://encryption-key.pem`。または、秘密鍵ファイルの未加工の内容を指定することもできます。

- SDK には、秘密鍵ファイルの未加工の内容を指定します。例えば、次のようになります: `--private-key "`cat encryption-key.pem`"`

型: 文字列

長さの制限: 最小長は 1 です。最大長は 16384 です。

Pattern: `[\u0009\u000A\u000D\u0020-\u00FF]*`

必須: いいえ

Tags

証明書のグループ化および検索に使用できるキーと値のペアです。

型: Tag オブジェクトの配列

配列メンバー : 最小数は 1 項目です。最大数は 50 項目です。

必須: いいえ

Usage

この証明書の使用方法を指定します。これは、次の方法で使用できます。

- SIGNING: AS2 メッセージの署名用
- ENCRYPTION: AS2 メッセージの暗号化用
- TLS: HTTPS 経由で送信される AS2 通信を保護する場合

型: 文字列

有効な値 : `SIGNING | ENCRYPTION`

必須: はい

レスポンスの構文

```
{
   "CertificateId": "string"
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

CertificateId

インポートされた証明書の識別子の配列です。この識別子は、プロファイルやパートナープロファイルの操作に使用します。

型: 文字列

長さの制限: 22 の固定長

パターン：`cert-([0-9a-f]{17})`

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

例

例

次の例では、暗号化に使用する証明書をインポートします。最初のコマンドでは、証明書と証明書チェーンファイルの内容を指定します。SDK コマンドにはこの形式を使用してください。

```
aws transfer import-certificate --usage ENCRYPTION --certificate "`cat encryption-
cert.pem`" \
    --private-key "`cat encryption-key.pem`" --certificate-chain "`cat root-ca.pem`"
```

例

次の例は、プライベートキー、証明書、証明書チェーンファイルのファイルの場所を指定することを除いて、前のコマンドと同じです。SDK を使用している場合、このバージョンのコマンドは機能しません。

```
aws transfer import-certificate --usage ENCRYPTION --certificate file://encryption-
cert.pem \
    --private-key file://encryption-key.pem --certificate-chain file://root-ca.pem
```

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

ImportHostKey

ServerIdパラメータで指定されたホストキーをサーバーに追加します。

リクエストの構文

```
{
   "Description": "string",
   "HostKeyBody": "string",
   "ServerId": "string",
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
      }
   ]
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

Description

このホストキーを識別するテキスト説明。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 200 です。

Pattern: [\p{Print}]*

必須: いいえ

HostKeyBody

SSH キーペアのプライベートキー部分。

AWS Transfer Family は、RSA、ECDSA、ED25519 キーを受け入れます。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 4,096 です。

必須: はい

ServerId

インポートするホストキーを含むサーバーの識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `s-([0-9a-f]{17})`

必須：はい

Tags

ホストキーのグループ化および検索に使用できるキーバリューペア。

型: Tag オブジェクトの配列

配列メンバー：最小数は 1 項目です。最大数は 50 項目です。

必須：いいえ

レスポンスの構文

```
{
   "HostKeyId": "string",
   "ServerId": "string"
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

HostKeyId

インポートされたキーのホストキー識別子を返します。

型: 文字列

長さの制限: 固定長は 25 です。

Pattern: `hostkey-[0-9a-f]{17}`

ServerId

インポートされたキーを含むサーバー ID を返します。

型: 文字列

長さの制限: 固定長は 19 です。

パターン：`s-([0-9a-f]{17})`

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceExistsException

要求されたリソースは存在しないか、コマンドに指定されたリージョン以外のリージョンに存在します。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

ThrottlingException

リクエストのスロットリングにより、リクエストが拒否されました。

HTTP ステータスコード：400

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

ImportSshPublicKey

ServerId で識別される、特定のファイル転送プロトコル対応サーバーに割り当てられた UserName 値で識別される Transfer Family ユーザーに、Secure Shell (SSH) パブリックキーを追加します。

レスポンスは UserName 値、ServerId 値、および SshPublicKeyId の名前を返します。

リクエストの構文

```
{
   "ServerId": "string",
   "SshPublicKeyBody": "string",
   "UserName": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

ServerId

システムでサーバーに割り当てられた一意の識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: s-([0-9a-f]{17})

必須：はい

SshPublicKeyBody

SSH キーペアのパブリックキー部分。

AWS Transfer Family は、RSA、ECDSA、ED25519 キーを受け入れます。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 2,048 です。

必須: はい

UserName

1 つ以上のサーバーに割り当てられている Transfer Family ユーザーの名前。

型: 文字列

長さの制限: 最小長は 3 です。最大長は 100 です。

パターン: `[\w][\w@.-]{2,99}`

必須 : はい

レスポンスの構文

```
{
   "ServerId": "string",
   "SshPublicKeyId": "string",
   "UserName": "string"
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

ServerId

サーバーにシステムで割り当てられた一意の識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `s-([0-9a-f]{17})`

SshPublicKeyId

インポートされたシステムによってパブリックキーに与えられた名前。

型: 文字列

長さの制限: 固定長は 21 です。

Pattern: `key-[0-9a-f]{17}`

UserName

指定した `ServerID` 値に割り当てられたユーザー名。

型: 文字列

長さの制限: 最小長は 3 です。最大長は 100 です。

パターン：`[\w][\w@.-]{2,99}`

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceExistsException

要求されたリソースは存在しないか、コマンドに指定されたリージョン以外のリージョンに存在します。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

ThrottlingException

リクエストのスロットリングにより、リクエストが拒否されました。

HTTP ステータスコード：400

例

例

このコマンドは、id_ecdsa.pubファイルに保存されている ECDSA キーをインポートします。

```
aws transfer import-ssh-public-key --server-id s-021345abcdef6789 --ssh-public-key-body
 file://id_ecdsa.pub --user-name jane-doe
```

例

先のコマンドを実行すると、システムは以下の情報を返します。

```
{
    "ServerId": "s-021345abcdef6789",
   "SshPublicKeyId": "key-1234567890abcdef0",
   "UserName": "jane-doe"
}
```

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2

- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

ListAccesses

サーバー上で持っているすべてのアクセス権の詳細を一覧表示します。

リクエストの構文

```
{
   "MaxResults": number,
   "NextToken": "string",
   "ServerId": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

MaxResults

返されるアクセス SID の最大数を指定します。

型: 整数

有効範囲: 最小値 は 1 です。最大値は 1000 です。

必須: いいえ

NextToken

ListAccesses からさらに結果を得られる場合、出力で NextToken パラメータが返されます。以降のコマンドで NextToken パラメータを渡すことで、追加のアクセス権を引き続き一覧表示できます。

型: 文字列

長さの制限：最小長は 1 です。最大長は 6,144 です。

必須: いいえ

ServerId

ユーザーが割り当てられたサーバーにシステムで割り当てられた一意の識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `s-([0-9a-f]{17})`

必須：はい

レスポンスの構文

```
{
   "Accesses": [
      {
         "ExternalId": "string",
         "HomeDirectory": "string",
         "HomeDirectoryType": "string",
         "Role": "string"
      }
   ],
   "NextToken": "string",
   "ServerId": "string"
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

Accesses

指定する ServerId 値 についてアクセス権とそのプロパティを返します。

型: ListedAccess オブジェクトの配列

NextToken

ListAccesses からさらに結果を得られる場合、出力で NextToken パラメータが返されます。以降のコマンドで NextToken パラメータを渡すことで、追加のアクセス権を引き続き一覧表示できます。

型: 文字列

長さの制限：最小長は 1 です。最大長は 6,144 です。

ServerId

ユーザーが割り当てられたサーバーにシステムで割り当てられた一意の識別子。

型: 文字列

長さの制限: 固定長は 19 です。

パターン：`s-([0-9a-f]{17})`

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidNextTokenException

渡された `NextToken` パラメータが無効です。

HTTP ステータスコード：400

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

ListAgreements

指定した内容でServerIdによって識別されるサーバーの契約書のリストを返します。結果を特定の数に制限したい場合は、MaxResults パラメータに値を指定します。以前にコマンドを実行してNextTokenの値を受け取った場合は、その値を指定して、中断したところから契約書を一覧表示できます。

リクエストの構文

```
{
   "MaxResults": number,
   "NextToken": "string",
   "ServerId": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

MaxResults

返される契約の最大数。

タイプ：整数

有効範囲: 最小値 は 1 です。最大値は 1000 です。

必須: いいえ

NextToken

ListAgreements からさらに結果を得られる場合、出力で NextToken パラメータが返されます。その後、NextToken パラメータに後続のコマンドを渡すことで、追加のアグリーメントをリストアップし続けることができます。

型: 文字列

長さの制限：最小長は 1 です。最大長は 6,144 です。

必須: いいえ

ServerId

契約書のリストを取得したいサーバーの識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `s-([0-9a-f]{17})`

必須：はい

レスポンスの構文

```
{
   "Agreements": [
      {
         "AgreementId": "string",
         "Arn": "string",
         "Description": "string",
         "LocalProfileId": "string",
         "PartnerProfileId": "string",
         "ServerId": "string",
         "Status": "string"
      }
   ],
   "NextToken": "string"
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

Agreements

各項目に契約書の詳細を含む配列を返します。

型: ListedAgreement オブジェクトの配列

NextToken

トークンを返すと、ListAgreements このトークンを使用して再度呼び出し、追加の結果があれば、その結果を受け取ることができます。

型: 文字列

長さの制限：最小長は 1 です。最大長は 6,144 です。

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidNextTokenException

渡された NextToken パラメータが無効です。

HTTP ステータスコード：400

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

ListCertificates

にインポートされた現在の証明書のリストを返します AWS Transfer Family。結果を特定の数に制限したい場合は、MaxResults パラメータに値を指定します。以前にコマンドを実行して NextToken パラメーターに値が返された場合、その値を提供して、中断した場所から証明書のリストを続けることができます。

リクエストの構文

```
{
   "MaxResults": number,
   "NextToken": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

MaxResults

返却する証明書の最大数。

タイプ：整数

有効範囲: 最小値 は 1 です。最大値は 1000 です。

必須: いいえ

NextToken

ListCertificates からさらに結果を得られる場合、出力で NextToken パラメータが返されます。その後、NextToken パラメータに後続のコマンドを渡すことで、さらに証明書をリストアップし続けることができます。

型: 文字列

長さの制限：最小長は 1 です。最大長は 6,144 です。

必須: いいえ

レスポンスの構文

```
{
   "Certificates": [
      {
         "ActiveDate": number,
         "Arn": "string",
         "CertificateId": "string",
         "Description": "string",
         "InactiveDate": number,
         "Status": "string",
         "Type": "string",
         "Usage": "string"
      }
   ],
   "NextToken": "string"
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

Certificates

ListCertificates 呼び出しで指定された証明書の配列を返します。

型: ListedCertificate オブジェクトの配列

NextToken

次の証明書をリストするために使用できる次のトークンを返します。

型: 文字列

長さの制限：最小長は 1 です。最大長は 6,144 です。

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidNextTokenException

渡された NextToken パラメータが無効です。

HTTP ステータスコード：400

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3

- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

ListConnectors

指定したリージョンのコネクタを一覧表示します。

リクエストの構文

```
{
   "MaxResults": number,
   "NextToken": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

MaxResults

返されるコネクターの最大数。

タイプ：整数

有効範囲: 最小値 は 1 です。最大値は 1000 です。

必須: いいえ

NextToken

ListConnectors からさらに結果を得られる場合、出力で NextToken パラメータが返されます。その後、NextToken パラメータに後続のコマンドを渡すことで、さらにコネクターをリストアップし続けることができます。

型: 文字列

長さの制限：最小長は 1 です。最大長は 6,144 です。

必須: いいえ

レスポンスの構文

```
{
```

```
   "Connectors": [
      {
         "Arn": "string",
         "ConnectorId": "string",
         "Url": "string"
      }
   ],
   "NextToken": "string"
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

Connectors

各項目にコネクタの詳細を含む配列を返します。

型: ListedConnector オブジェクトの配列

NextToken

トークンを返すと、ListConnectors このトークンを使用して再度呼び出し、追加の結果があれば、その結果を受け取ることができます。

型: 文字列

長さの制限：最小長は 1 です。最大長は 6,144 です。

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidNextTokenException

渡された NextToken パラメータが無効です。

HTTP ステータスコード：400

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

ListExecutions

指定されたワークフローについて、進行中のすべての実行を一覧表示します。

> **Note**
>
> 指定されたワークフロー ID が見つからない場合、ListExecutions は ResourceNotFound 例外を返します。

リクエストの構文

```
{
   "MaxResults": number,
   "NextToken": "string",
   "WorkflowId": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

MaxResults

返される最大実行回数を指定します。

タイプ : 整数

有効範囲: 最小値 は 1 です。最大値は 1000 です。

必須: いいえ

NextToken

ListExecutions は出力で NextToken を返します。以降のコマンドで NextToken パラメータを渡すことで、追加の実行を引き続き一覧表示できます。

これはページ処理などに役立ちます。1 つのワークフローを 100 回実行する場合、最初の 10 回のみを一覧表示することをお勧めします。もしそうなら、max-results を指定して API を呼び出す :

```
aws transfer list-executions --max-results 10
```

これは最初の 10 回の実行の詳細ならびに 11 回目を指すポインタ (NextToken) を返します。これで、受け取った NextToken 値を指定して、再度APIを呼び出すことができる：

```
aws transfer list-executions --max-results 10 --next-token
$somePointerReturnedFromPreviousListResult
```

この呼び出しは、次の 10 回の実行、11 回目 ~ 20 回目の実行を返します。次いで、100 回すべての実行の詳細が返るまでこの呼び出しを繰り返すことができます。

型: 文字列

長さの制限：最小長は 1 です。最大長は 6,144 です。

必須: いいえ

WorkflowId

ワークフローの一意の識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `w-([a-z0-9]{17})`

必須：はい

レスポンスの構文

```
{
   "Executions": [
      {
         "ExecutionId": "string",
         "InitialFileLocation": {
            "EfsFileLocation": {
               "FileSystemId": "string",
               "Path": "string"
            },
            "S3FileLocation": {
               "Bucket": "string",
               "Etag": "string",
```

```
                "Key": "string",
                "VersionId": "string"
             }
          },
          "ServiceMetadata": {
             "UserDetails": {
                "ServerId": "string",
                "SessionId": "string",
                "UserName": "string"
             }
          },
          "Status": "string"
       }
    ],
    "NextToken": "string",
    "WorkflowId": "string"
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

Executions

各実行の詳細を ListedExecution 配列で返します。

型: ListedExecution オブジェクトの配列

NextToken

ListExecutions は出力で NextToken を返します。以降のコマンドで NextToken パラメータを渡すことで、追加の実行を引き続き一覧表示できます。

型: 文字列

長さの制限：最小長は 1 です。最大長は 6,144 です。

WorkflowId

ワークフローの一意の識別子。

型: 文字列

長さの制限: 固定長は 19 です。

パターン：`w-([a-z0-9]{17})`

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidNextTokenException

渡された `NextToken` パラメータが無効です。

HTTP ステータスコード：400

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

ListHostKeys

ServerIdパラメータで指定されたサーバーのホストキーのリストを返します。

リクエストの構文

```
{
   "MaxResults": number,
   "NextToken": "string",
   "ServerId": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

MaxResults

返されるホストキーの最大数。

タイプ：整数

有効範囲: 最小値 は 1 です。最大値は 1000 です。

必須: いいえ

NextToken

返されなかった結果が他にもある場合は、NextTokenパラメーターが返されます。ListHostKeysの値を以降の呼び出しに使用して、結果を一覧表示し続けることができます。

型: 文字列

長さの制限：最小長は 1 です。最大長は 6,144 です。

必須: いいえ

ServerId

表示するホストキーが含まれているサーバーの識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `s-([0-9a-f]{17})`

必須：はい

レスポンスの構文

```
{
   "HostKeys": [
      {
         "Arn": "string",
         "DateImported": number,
         "Description": "string",
         "Fingerprint": "string",
         "HostKeyId": "string",
         "Type": "string"
      }
   ],
   "NextToken": "string",
   "ServerId": "string"
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

HostKeys

各項目にホストキーの詳細が含まれる配列を返します。

型: ListedHostKey オブジェクトの配列

NextToken

トークンを返すと、ListHostKeys このトークンを使用して再度呼び出し、追加の結果があれば、その結果を受け取ることができます。

型: 文字列

長さの制限：最小長は 1 です。最大長は 6,144 です。

ServerId

リストされたホストキーを含むサーバー ID を返します。

型: 文字列

長さの制限: 固定長は 19 です。

パターン：`s-([0-9a-f]{17})`

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidNextTokenException

渡された `NextToken` パラメータが無効です。

HTTP ステータスコード：400

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

ListProfiles

システムのプロファイルのリストを返します。結果を特定の数に制限したい場合は、MaxResults パラメータに値を指定します。以前にコマンドを実行して NextToken の値を受け取った場合は、その値を入力することで、前回の続きからプロファイルをリストアップすることができます。

リクエストの構文

```
{
   "MaxResults": number,
   "NextToken": "string",
   "ProfileType": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

MaxResults

返されるプロファイルの最大数。

タイプ：整数

有効範囲: 最小値 は 1 です。最大値は 1000 です。

必須: いいえ

NextToken

返されなかった結果が他にもある場合は、NextTokenパラメーターが返されます。ListProfilesの値を以降の呼び出しに使用して、結果を一覧表示し続けることができます。

型: 文字列

長さの制限：最小長は 1 です。最大長は 6,144 です。

必須: いいえ

ProfileType

LOCAL タイププロファイルのみを一覧表示するか、PARTNER タイププロファイルのみを一覧表示するかを示します。リクエストで指定されていない場合、コマンドはすべてのタイプのプロファイルを一覧表示します。

型: 文字列

有効な値：LOCAL | PARTNER

必須：いいえ

レスポンスの構文

```
{
   "NextToken": "string",
   "Profiles": [
      {
         "Arn": "string",
         "As2Id": "string",
         "ProfileId": "string",
         "ProfileType": "string"
      }
   ]
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

NextToken

トークンを返すと、ListProfiles このトークンを使用して再度呼び出し、追加の結果があれば、その結果を受け取ることができます。

型: 文字列

長さの制限：最小長は 1 です。最大長は 6,144 です。

Profiles

配列を返します。各項目にはプロファイルの詳細が含まれます。

型: ListedProfile オブジェクトの配列

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidNextTokenException

渡された `NextToken` パラメータが無効です。

HTTP ステータスコード：400

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

ListSecurityPolicies

サーバーと SFTP コネクタにアタッチされているセキュリティポリシーを一覧表示します。セキュリティポリシーの詳細については、「サーバーのセキュリティポリシーの使用」または「SFTP コネクタのセキュリティポリシーの使用」を参照してください。

リクエストの構文

```
{
   "MaxResults": number,
   "NextToken": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

MaxResults

ListSecurityPolicies クエリへのレスポンスとして返すセキュリティポリシーの数を指定します。

型: 整数

有効範囲: 最小値 は 1 です。最大値は 1000 です。

必須: いいえ

NextToken

ListSecurityPolicies コマンドから付加的な結果を取得する際には、出力で NextToken パラメータが返されます。以降のコマンドで NextToken パラメータを渡すことで、追加のセキュリティポリシーを引き続き一覧表示できます。

型: 文字列

長さの制限：最小長は 1 です。最大長は 6,144 です。

必須: いいえ

レスポンスの構文

```
{
   "NextToken": "string",
   "SecurityPolicyNames": [ "string" ]
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

NextToken

ListSecurityPolicies オペレーションから付加的な結果を得られる場合、出力で NextToken パラメータが返されます。次のコマンドでは、NextToken パラメータを渡すことでセキュリティポリシーを引き続き一覧表示します。

型: 文字列

長さの制限：最小長は 1 です。最大長は 6,144 です。

SecurityPolicyNames

一覧表示されたセキュリティポリシーの配列。

タイプ：文字列の配列

長さの制限: 最小長は 0 です。最大長は 100 です。

パターン：Transfer[A-Za-z0-9]*SecurityPolicy-[A-Za-z0-9-]+

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidNextTokenException

渡された NextToken パラメータが無効です。

HTTP ステータスコード：400

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

例

例

次の例では、使用可能なすべてのセキュリティポリシーの名前を一覧表示します。

リクエスト例

```
aws transfer list-security-policies
```

レスポンス例

```
{
    "SecurityPolicyNames": [
        "TransferSecurityPolicy-2023-05",
        "TransferSecurityPolicy-2022-03",
        "TransferSecurityPolicy-FIPS-2024-01",
        "TransferSecurityPolicy-2024-01",
        "TransferSecurityPolicy-PQ-SSH-FIPS-Experimental-2023-04",
        "TransferSecurityPolicy-PQ-SSH-Experimental-2023-04",
        "TransferSecurityPolicy-FIPS-2020-06",
        "TransferSecurityPolicy-2020-06",
        "TransferSecurityPolicy-2018-11",
        "TransferSecurityPolicy-FIPS-2023-05"
    ]
```

```
}
```

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

ListServers

AWS アカウントに関連付けられているファイル転送プロトコル対応サーバーを一覧表示します。

リクエストの構文

```
{
   "MaxResults": number,
   "NextToken": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

MaxResults

ListServers クエリへのレスポンスとして返すサーバーの数を指定します。

型: 整数

有効範囲: 最小値 は 1 です。最大値は 1000 です。

必須: いいえ

NextToken

ListServers コマンドから付加的な結果を取得する際には、出力で NextToken パラメータが返されます。以降のコマンドで NextToken パラメータを渡すことで、追加のサーバーを引き続き一覧表示できます。

型: 文字列

長さの制限：最小長は 1 です。最大長は 6,144 です。

必須: いいえ

レスポンスの構文

```
{
```

```
   "NextToken": "string",
   "Servers": [
      {
         "Arn": "string",
         "Domain": "string",
         "EndpointType": "string",
         "IdentityProviderType": "string",
         "LoggingRole": "string",
         "ServerId": "string",
         "State": "string",
         "UserCount": number
      }
   ]
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

NextToken

ListServers オペレーションから付加的な結果を得られる場合、出力で NextToken パラメータが返されます。次のコマンドでは、NextToken パラメータを渡すことで追加のサーバーを引き続き一覧表示できます。

型: 文字列

長さの制限：最小長は 1 です。最大長は 6,144 です。

Servers

一覧表示されたサーバーの配列。

型: ListedServer オブジェクトの配列

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidNextTokenException

渡された NextToken パラメータが無効です。

HTTP ステータスコード：400

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

例

例

次の例では、 に存在するサーバーを一覧表示します AWS アカウント。

例の NextToken 値は実際のものではないことに注意してください。これらはパラメータの使用方法を示すためのものです。

リクエスト例

```
{
   "MaxResults": 1,
   "NextToken": "token-from-previous-API-call"
}
```

レスポンス例

```
{
   "NextToken": "another-token-to-continue-listing",
   "Servers": [
      {
         "Arn": "arn:aws:transfer:us-east-1:111112222222:server/s-01234567890abcdef",
         "Domain": "S3",
```

```
            "IdentityProviderType": "SERVICE_MANAGED",
            "EndpointType": "PUBLIC",
            "LoggingRole": "arn:aws:iam::111112222222:role/my-role",
            "ServerId": "s-01234567890abcdef",
            "State": "ONLINE",
            "UserCount": 3
        }
    ]
}
```

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

ListTagsForResource

指定した Amazon リソースネーム (ARN) に関連付けられているすべてのタグを一覧表示します。リソースは、ユーザー、サーバー、またはロールのいずれかです。

リクエストの構文

```
{
   "Arn": "string",
   "MaxResults": number,
   "NextToken": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

Arn

特定の Amazon リソースネーム (ARN) に関連付けられているタグをリクエストします。ARN は、サーバー、ユーザー、ロールなど、特定の AWS リソースの識別子です。

型: 文字列

長さの制限: 最小長は 20 です。最大長は 1600 です。

Pattern: `arn:\S+`

必須：はい

MaxResults

`ListTagsForResource` クエリへのレスポンスとして返すタグの数を指定します。

型: 整数

有効範囲: 最小値 は 1 です。最大値は 1000 です。

必須: いいえ

NextToken

ListTagsForResource オペレーションからの付加的な結果をリクエストすると、出力で NextToken パラメータが返されます。以降のコマンドで NextToken パラメータを渡すことで、追加のタグを引き続き一覧表示できます。

型: 文字列

長さの制限：最小長は 1 です。最大長は 6,144 です。

必須: いいえ

レスポンスの構文

```
{
   "Arn": "string",
   "NextToken": "string",
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
      }
   ]
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

Arn

タグを一覧表示するために指定した ARN。

型: 文字列

長さの制限: 最小長は 20 です。最大長は 1600 です。

Pattern: arn:\S+

NextToken

ListTagsForResource からさらに結果を得られる場合、出力で NextToken パラメータが返されます。以降のコマンドで NextToken パラメータを渡すことで、追加のタグを引き続き一覧表示できます。

型: 文字列

長さの制限：最小長は 1 です。最大長は 6,144 です。

Tags

リソースに割り当てられるキーバリューペアであり、通常は項目をグループ化または検索する目的で使用されます。タグとは、ユーザーが定義するメタデータです。

型: Tag オブジェクトの配列

配列メンバー：最小数は 1 項目です。最大数は 50 項目です。

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidNextTokenException

渡された NextToken パラメータが無効です。

HTTP ステータスコード：400

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

例

例

次の例では、指定した ARN を持つリソースのタグを一覧表示します。

リクエスト例

```
{
   "Arn": "arn:aws:transfer:us-east-1:176354371281:server/s-01234567890abcdef"
}
```

例

この例では、 の使用例を示します ListTagsForResource。

レスポンス例

```
{
   "Tags": [
      {
         "Key": "Name",
         "Value": "MyServer"
      }
   ]
}
```

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2

- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

ListUsers

ServerId パラメータを渡すことで指定したファイル転送プロトコル対応サーバーのユーザーを一覧表示します。

リクエストの構文

```
{
   "MaxResults": number,
   "NextToken": "string",
   "ServerId": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

MaxResults

ListUsers クエリへのレスポンスとして返すユーザーの数を指定します。

型: 整数

有効範囲: 最小値 は 1 です。最大値は 1000 です。

必須: いいえ

NextToken

ListUsers コールからさらに結果を得られる場合、出力で NextToken パラメータが返されます。以降の ListUsers コマンドで NextToken を渡すことで、追加のユーザーを引き続き一覧表示できます。

型: 文字列

長さの制限：最小長は 1 です。最大長は 6,144 です。

必須: いいえ

ServerId

ユーザーが割り当てられたサーバーにシステムで割り当てられた一意の識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `s-([0-9a-f]{17})`

必須：はい

レスポンスの構文

```
{
   "NextToken": "string",
   "ServerId": "string",
   "Users": [
      {
         "Arn": "string",
         "HomeDirectory": "string",
         "HomeDirectoryType": "string",
         "Role": "string",
         "SshPublicKeyCount": number,
         "UserName": "string"
      }
   ]
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

NextToken

`ListUsers` からさらに結果を得られる場合、出力で `NextToken` パラメータが返されます。以降のコマンドで `NextToken` パラメータを渡すことで、追加のユーザーを引き続き一覧表示できます。

型: 文字列

長さの制限：最小長は 1 です。最大長は 6,144 です。

ServerId

ユーザーの割り当て先サーバーにシステムで割り当てられた一意の識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `s-([0-9a-f]{17})`

Users

指定する `ServerId` 値 について Transfer Family ユーザーとそのプロパティを返します。

型: ListedUser オブジェクトの配列

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidNextTokenException

渡された `NextToken` パラメータが無効です。

HTTP ステータスコード：400

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

例

例

ListUsers API コールは、指定したサーバーに関連付けられたユーザーのリストを返します。

リクエスト例

```
        {
         "MaxResults": 100,
         "NextToken": "eyJNYXJrZXIiOiBudWxsLCAiYm90b1X0cnVuU2F0ZV9hbW91bnQiOiAyfQ==",
         "ServerId": "s-01234567890abcdef"
         }
```

例

これは、この API コールに対するレスポンスのサンプルです。

レスポンス例

```
{
   "NextToken": "eyJNYXJrZXIiOiBudWxsLCAiYm90b1X0cnVuU2F0ZV9hbW91bnQiOiAyfQ==",
   "ServerId": "s-01234567890abcdef",
   "Users": [
      {
         "Arn": "arn:aws:transfer:us-east-1:176354371281:user/s-01234567890abcdef/
charlie",
         "HomeDirectory": "/tests/home/charlie",
         "SshPublicKeyCount": 1,
         "Role": "arn:aws:iam::176354371281:role/transfer-role1",
         "Tags": [
            {
               "Key": "Name",
               "Value": "user1"
```

```
            }
         ],
         "UserName": "my_user"
      }
   ]
}
```

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

ListWorkflows

現在のリージョン AWS アカウント の に関連付けられているすべてのワークフローを一覧表示します。

リクエストの構文

```
{
   "MaxResults": number,
   "NextToken": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

MaxResults

返されるワークフローの最大数を指定します。

型: 整数

有効範囲: 最小値 は 1 です。最大値は 1000 です。

必須: いいえ

NextToken

ListWorkflows は出力で NextToken を返します。以降のコマンドで NextToken パラメータを渡すことで、追加のワークフローを引き続き一覧表示できます。

型: 文字列

長さの制限：最小長は 1 です。最大長は 6,144 です。

必須: いいえ

レスポンスの構文

```
{
```

```
   "NextToken": "string",
   "Workflows": [
      {
         "Arn": "string",
         "Description": "string",
         "WorkflowId": "string"
      }
   ]
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

NextToken

ListWorkflows は出力で NextToken を返します。以降のコマンドで NextToken パラメータを渡すことで、追加のワークフローを引き続き一覧表示できます。

型: 文字列

長さの制限：最小長は 1 です。最大長は 6,144 です。

Workflows

各ワークフローについて Arn、WorkflowId、およびDescription を返します。

型: ListedWorkflow オブジェクトの配列

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidNextTokenException

渡された NextToken パラメータが無効です。

HTTP ステータスコード：400

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

SendWorkflowStepState

非同期カスタムステップ向けにコールバックを送信します。

ワークフローのカスタムステップの実行中に ExecutionId、WorkflowId、およびToken がターゲットリソースに渡されます。ステータスの提供するだけでなく、コールバック付きでそれらを含める必要があります。

リクエストの構文

```
{
   "ExecutionId": "string",
   "Status": "string",
   "Token": "string",
   "WorkflowId": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

ExecutionId

ワークフローの一意の識別子。

型: 文字列

長さの制限: 最大長は 36 です。

パターン: [0-9a-fA-F]{8}\-[0-9a-fA-F]{4}\-[0-9a-fA-F]{4}\-[0-9a-fA-F]{4}\-[0-9a-fA-F]{12}

必須：はい

Status

指定したステップの成否を示します。

型: 文字列

有効な値：SUCCESS | FAILURE

必須: はい

Token

同じ実行内で複数の Lambda ステップについて複数のコールバックを区別するために使用されます。

型: 文字列

長さの制限：最小長は 1 です。最大長は 64 文字です。

パターン: \w+

必須：はい

WorkflowId

ワークフローの一意の識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: w-([a-z0-9]{17})

必須：はい

レスポンス要素

アクションが成功した場合、サービスは空の HTTP 本文を持つ HTTP 200 レスポンスを返します。

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

AccessDeniedException

このアクションを実行する十分なアクセス権限がありません。

HTTP ステータスコード：400

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

ThrottlingException

リクエストのスロットリングにより、リクエストが拒否されました。

HTTP ステータスコード：400

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

StartDirectoryListing

リモート SFTP サーバーからディレクトリの内容のリストを取得します。コネクタ ID、出力パス、リモートディレクトリパスを指定します。オプションMaxItemsの値を指定して、リモートディレクトリからリストされる項目の最大数を制御することもできます。この API は、リモートディレクトリ内のすべてのファイルとディレクトリのリスト (最大値まで) を返しますが、サブディレクトリ内のファイルやフォルダは返しません。つまり、ファイルとディレクトリのリストを 1 レベルの深さでのみ返します。

出品ファイルを受け取ったら、StartFileTransferAPI コールの RetrieveFilePathsパラメータに転送するファイルを指定できます。

出力ファイルの命名規則は です *connector-ID-listing-ID*.json。出力ファイルには、次の情報が含まれています。

- filePath: リモートサーバー上の SFTP コネクタのリストリクエストのディレクトリを基準とした、リモートファイルの完全なパス。
- modifiedTimestamp: ファイルが最後に変更された時刻を UTC 時間形式で表します。このフィールドはオプションです。リモートファイル属性にタイムスタンプが含まれていない場合、ファイルリストから除外されます。
- size: ファイルのサイズ、バイト単位。このフィールドはオプションです。リモートファイル属性にファイルサイズが含まれていない場合は、ファイルリストから省略されます。
- path: リモートサーバー上の SFTP コネクタのリストリクエストのディレクトリに対するリモートディレクトリの完全なパス。
- truncated: リスト出力にリモートディレクトリに含まれるすべての項目が含まれているかどうかを示すフラグ。Truncated 出力値が true の場合、オプションのmax-items入力属性で指定された値を増やして、より多くの項目を一覧表示できます (最大許容リストサイズは 10,000 項目まで)。

リクエストの構文

```
{
   "ConnectorId": "string",
   "MaxItems": number,
   "OutputDirectoryPath": "string",
   "RemoteDirectoryPath": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

ConnectorId

コネクタの一意の識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `c-([0-9a-f]{17})`

必須：はい

MaxItems

取得するファイル/ディレクトリ名の最大数を指定できるオプションのパラメータ。デフォルト値は 1,000 です。

タイプ：整数

値の範囲: 最小値 は 1 です。最大値は 10,000 です。

必須: いいえ

OutputDirectoryPath

ディレクトリリストの結果を保存する Amazon S3 ストレージのパス (バケットとプレフィックス) を指定します。

型: 文字列

長さの制限: 最小長は 1 です。最大長は 1,024 です。

Pattern: `(.)+`

必須：はい

RemoteDirectoryPath

コンテンツを一覧表示するリモート SFTP サーバーのディレクトリを指定します。

型: 文字列

長さの制限: 最小長は 1 です。最大長は 1,024 です。

Pattern: `(.)+`

必須：はい

レスポンスの構文

```
{
   "ListingId": "string",
   "OutputFileName": "string"
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

ListingId

ディレクトリリスト呼び出しの一意の識別子を返します。

型: 文字列

長さの制限：最小長は 1 です。最大長は 512 です。

Pattern: `[0-9a-zA-Z./-]+`

OutputFileName

結果が保存されているファイル名を返します。これは、コネクタ ID とリスト ID の組み合わせです`<connector-id>-<listing-id>.json`。

型: 文字列

長さ制限: 最小長は 26 です。最大長は 537 です。

パターン：`c-([0-9a-f]{17})-[0-9a-zA-Z./-]+.json`

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

ThrottlingException

リクエストのスロットリングにより、リクエストが拒否されました。

HTTP ステータスコード：400

例

例

次の例では、指定されたコネクタによって識別されるリモート SFTP サーバー上の homeフォルダの内容を一覧表示します。結果は、Amazon S3 の場所 、/DOC-EXAMPLE-BUCKET/connector-filesおよび という名前のファイルに配置されますc-AAAA1111BBBB2222C-6666abcd-11aa-22bb-cc33-0000aaaa3333.json。

リクエスト例

```
{
    "ConnectorId": "c-AAAA1111BBBB2222C",
    "MaxItems": "10",
    "OutputDirectoryPath": "/DOC-EXAMPLE-BUCKET/connector-files",
    "RemoteDirectoryPath": "/home"
}
```

レスポンス例

```
{
    "ListingId": "6666abcd-11aa-22bb-cc33-0000aaaa3333",
    "OutputFileName": "c-AAAA1111BBBB2222C-6666abcd-11aa-22bb-cc33-0000aaaa3333.json"
}
```

```
// under bucket "DOC-EXAMPLE-BUCKET"
connector-files/c-AAAA1111BBBB2222C-6666abcd-11aa-22bb-cc33-0000aaaa3333.json
{
    "files": [
        {
            "filePath": "/home/what.txt",
            "modifiedTimestamp": "2024-01-30T20:34:54Z",
            "size" : 2323
        },
        {
            "filePath": "/home/how.pgp",
            "modifiedTimestamp": "2024-01-30T20:34:54Z",
            "size" : 51238
        }
    ],
    "paths": [
        {
            "path": "/home/magic"
        },
        {
            "path": "/home/aws"
        },
    ],
    "truncated": false
}
```

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

StartFileTransfer

ローカル AWS ストレージとリモート AS2 または SFTP サーバー間のファイル転送を開始します。

- AS2 コネクタでは、転送するファイルを識別するために ConnectorId と 1 つ以上の SendFilePaths を指定します。
- SFTP コネクタの場合、ファイル転送はアウトバウンドでもインバウンドでもかまいません。いずれの場合も、ConnectorId を指定します。転送方向に応じて、次の項目も指定します。
  - パートナーの SFTP サーバーから Amazon Web Services ストレージにファイルを転送する場合は、転送するファイルを識別するために 1 つ以上の RetrieveFilePaths を指定し、宛先フォルダを指定する LocalDirectoryPath を指定します。
  - AWS ストレージからパートナーの SFTP サーバーにファイルを転送する場合は、転送するファイルを識別するために 1 つ以上の SendFilePaths を指定し、宛先フォルダを指定する RemoteDirectoryPath を指定します。

リクエストの構文

```
{
   "ConnectorId": "string",
   "LocalDirectoryPath": "string",
   "RemoteDirectoryPath": "string",
   "RetrieveFilePaths": [ "string" ],
   "SendFilePaths": [ "string" ]
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

ConnectorId

コネクタの一意の識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `c-([0-9a-f]{17})`

必須：はい

LocalDirectoryPath

インバウンド転送の場合、`LocaDirectoryPath` はパートナーの SFTP サーバーから転送される 1 つ以上のファイルの宛先を指定します。

型: 文字列

長さの制限: 最小長は 1 です。最大長は 1,024 です。

パターン: `(.)+`

必須: いいえ

RemoteDirectoryPath

アウトバウンド転送の場合、`RemoteDirectoryPath` はパートナーの SFTP サーバーに転送される 1 つ以上のファイルの宛先を指定します。`RemoteDirectoryPath` を指定しない場合、ファイルの送信先は SFTP ユーザーのホームディレクトリです。

型: 文字列

長さの制限: 最小長は 1 です。最大長は 1,024 です。

パターン: `(.)+`

必須: いいえ

RetrieveFilePaths

パートナーの SFTP サーバーの 1 つ以上のソースパス。各文字列は、1 回のインバウンドファイル転送のソースファイルパスを表します。

タイプ：文字列の配列

配列メンバー：最小数は 1 項目です。最大数は 10 項目です。

長さの制限: 最小長は 1 です。最大長は 1,024 です。

パターン: `(.)+`

必須: いいえ

SendFilePaths

Amazon S3 ストレージの 1 つ以上のソースパス。各文字列は、1 回のアウトバウンドファイル転送のソースファイルパスを表します。例えば *DOC-EXAMPLE-BUCKET/myfile.txt* です。

> **Note**
>
> *DOC-EXAMPLE-BUCKET* を実際のバケットの 1 つに置き換えてください。

タイプ：文字列の配列

配列メンバー：最小数は 1 項目です。最大数は 10 項目です。

長さの制限: 最小長は 1 です。最大長は 1,024 です。

パターン: (.)+

必須: いいえ

レスポンスの構文

```
{
   "TransferId": "string"
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

TransferId

ファイル転送の一意の識別子を返します。

型: 文字列

長さの制限：最小長は 1 です。最大長は 512 です。

パターン：[0-9a-zA-Z./-]+

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

ThrottlingException

リクエストのスロットリングにより、リクエストが拒否されました。

HTTP ステータスコード：400

例

例

次の例では、Transfer Family サーバーからリモートの取引相手のエンドポイントへの AS2 ファイル転送を開始します。 *DOC-EXAMPLE-BUCKET* を実際のバケットの 1 つに置き換えてください。

リクエスト例

```
{
```

```
    "ConnectorId": "c-AAAA1111BBBB2222C",
    "SendFilePaths": [
        "/DOC-EXAMPLE-BUCKET/myfile-1.txt",
        "/DOC-EXAMPLE-BUCKET/myfile-2.txt",
        "/DOC-EXAMPLE-BUCKET/myfile-3.txt"
    ]
}
```

レスポンス例

```
{
   "TransferId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
}
```

例

次の例では、ローカル AWS ストレージからリモート SFTP サーバーへのファイル転送を開始します。

リクエスト例

```
{
    "ConnectorId": "c-01234567890abcdef",
    "SendFilePaths": [
        "/DOC-EXAMPLE-BUCKET/myfile-1.txt",
        "/DOC-EXAMPLE-BUCKET/myfile-2.txt",
        "/DOC-EXAMPLE-BUCKET/myfile-3.txt"
    ],
    "RemoteDirectoryPath": "/MySFTPRootFolder/fromTranferFamilyServer"
}
```

レスポンス例

```
{
   "TransferId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222"
}
```

例

次の例では、リモート SFTP サーバーからローカル AWS ストレージへのファイル転送を開始します。

リクエスト例

```
{
    "ConnectorId": "c-111122223333AAAAA",
    "RetrieveFilePaths": [
        "/MySFTPFolder/toTranferFamily/myfile-1.txt",
        "/MySFTPFolder/toTranferFamily/myfile-2.txt",
        "/MySFTPFolder/toTranferFamily/myfile-3.txt"
    ],
    "LocalDirectoryPath": "/DOC-EXAMPLE-BUCKET/mySourceFiles"
}
```

レスポンス例

```
{
   "TransferId": "a1b2c3d4-5678-90ab-cdef-EXAMPLEaaaaa"
}
```

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

StartServer

ファイル転送プロトコル対応サーバーの状態を OFFLINE から ONLINE に変更します。既に ONLINE であるサーバーには影響しません。ONLINE サーバーはファイル転送ジョブを受け入れて処理できます。

STARTING のステータスは、サーバーが中間状態である (完全にレスポンス可能ではないか完全なオフラインではない) ことを示します。START_FAILED の値は、エラー条件を示す可能性あります。

この呼び出しからレスポンスは返りません。

リクエストの構文

```
{
   "ServerId": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

ServerId

起動するサーバーにシステムで割り当てられた一意の識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `s-([0-9a-f]{17})`

必須：はい

レスポンス要素

アクションが成功した場合、サービスは空の HTTP 本文を持つ HTTP 200 レスポンスを返します。

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

ThrottlingException

リクエストのスロットリングにより、リクエストが拒否されました。

HTTP ステータスコード：400

例

例

次の例では サーバーを起動します。

リクエスト例

```
{
   "ServerId": "s-01234567890abcdef"
```

```
}
```

例

これは、この API コールに対するレスポンスのサンプルです。

レスポンス例

```
{
   "ServerId": "s-01234567890abcdef"
}
```

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

StopServer

ファイル転送プロトコル対応サーバーの状態を ONLINE から OFFLINE に変更します。OFFLINE サーバーはファイル転送ジョブを受け入れて処理することができません。サーバーやユーザーのプロパティなど、サーバーに関連付けられている情報は、サーバーを停止しても影響を受けません。

> **Note**
>
> サーバーを停止しても、ファイル転送プロトコルのエンドポイント課金は減らないし、影響もありません。

STOPPING のステータスは、サーバーが中間状態である (完全なレスポンスができないか完全にオフラインでない) ことを示します。STOP_FAILED の値は、エラー条件を示す可能性あります。

この呼び出しからレスポンスは返りません。

リクエストの構文

```
{
   "ServerId": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

ServerId

停止したサーバーにシステムで割り当てられた一意の識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `s-([0-9a-f]{17})`

必須：はい

レスポンス要素

アクションが成功した場合、サービスは空の HTTP 本文を持つ HTTP 200 レスポンスを返します。

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

ThrottlingException

リクエストのスロットリングにより、リクエストが拒否されました。

HTTP ステータスコード：400

例

例

次の例では サーバーを停止します。

リクエスト例

```
{
   "ServerId": "s-01234567890abcdef"
}
```

例

これは、この API コールに対するレスポンスのサンプルです。

レスポンス例

```
{
   "ServerId": "s-01234567890abcdef"
}
```

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

TagResource

Amazon リソースネーム (ARN) によって識別されるようなキーバリューペアをリソースにアタッチします。リソースは、ユーザー、サーバー、ロール、その他のエンティティです。

この呼び出しから返るレスポンスはありません。

リクエストの構文

```
{
   "Arn": "string",
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
      }
   ]
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

Arn

サーバー、ユーザー、ロールなど、特定のリソースの Amazon AWS リソースネーム (ARN)。

型: 文字列

長さの制限: 最小長は 20 です。最大長は 1600 です。

Pattern: `arn:\S+`

必須：はい

Tags

ARN に割り当てられたキーバリューペアであり、これを使用するとリソースをタイプ別にグループ化したり検索することができます。このメタデータは、あらゆる目的でリソース (サーバー、ユーザー、ワークフローなど) に添付できます。

型: Tag オブジェクトの配列

配列メンバー：最小数は 1 項目です。最大数は 50 項目です。

必須: はい

レスポンス要素

アクションが成功した場合、サービスは空の HTTP 本文を持つ HTTP 200 レスポンスを返します。

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

例

例

次の例では、ファイル転送プロトコル対応サーバーにタグを追加します。

リクエスト例

```
{
   "Arn": "arn:aws:transfer:us-east-1:176354371281:server/s-01234567890abcdef",
   "Tags": [
      {
         "Key": "Group",
         "Value": "Europe"
      }
   ]
}
```

例

この例では、 の使用例を示します TagResource。

レスポンス例

```
         HTTP 200 response with an empty HTTP body.
```

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

TestConnection

SFTP コネクタが正常にセットアップされているかどうかをテストします。このオペレーションを呼び出して、ローカル AWS ストレージと取引相手の SFTP サーバー間でファイルを転送できるかどうかをテストすることを強くお勧めします。

リクエストの構文

```
{
   "ConnectorId": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

ConnectorId

コネクタの一意の識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `c-([0-9a-f]{17})`

必須：はい

レスポンスの構文

```
{
   "ConnectorId": "string",
   "Status": "string",
   "StatusMessage": "string"
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

ConnectorId

テストしているコネクタオブジェクトの識別子を返します。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `c-([0-9a-f]{17})`

Status

テストに成功した場合はOKを、テストに失敗した場合はERRORを返します。

型: 文字列

StatusMessage

テストが成功した場合は `Connection succeeded` を返します。または、テストが失敗した場合は説明的なエラーメッセージを返します。以下のリストは、表示されるエラーメッセージに応じて、トラブルシューティングの詳細を示しています。

- シークレット名がロールの転送権限のシークレット名と一致していることを確認します。
- コネクタ構成内のサーバー URL を確認し、ログイン認証情報がコネクタ外でも正常に機能することを確認します。
- シークレットが存在し、正しい形式になっていることを確認します。
- コネクタ設定のトラステッドホストキーがssh-keyscan出力と一致することを確認します。

型: 文字列

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

例

例

次の例では、リモートサーバーへの接続をテストします。

```
aws transfer test-connection --connector-id c-abcd1234567890fff
```

レスポンス例

API コールが成功すると、以下の詳細が返されます。

```
{

    "Status": "OK",

    "StatusMessage": "Connection succeeded"

}
```

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

TestIdentityProvider

ファイル転送プロトコル対応サーバーの IdentityProviderType が AWS_DIRECTORY_SERVICE または API_Gateway の場合、ID プロバイダーが正常に設定されたかどうかをテストします。サーバーを作成したら認証方法をテストするために、すぐにこのオペレーションを呼び出すことをお勧めします。そうすることで、ID プロバイダーの統合に関する問題を解決でき、ユーザーがサービスを正常に使用できるようになります。

ServerId および UserName パラメータが必要です。ServerProtocol、SourceIp、および UserPassword はすべてオプションです。

次の点に注意してください。

- サーバーの IdentityProviderType が SERVICE_MANAGED である場合、TestIdentityProvider を使用できません。
- TestIdentityProviderはキーでは動作しません。パスワードのみを受け付けます。
- TestIdentityProviderは、キーとパスワードを処理するカスタム ID プロバイダのパスワード操作をテストできます。
- パラメータに正しくない値を指定すると、Response フィールドは空になります。
- サービスマネージドユーザーを使用するサーバーのサーバー ID を指定すると、エラーが発生します。

  ```
  An error occurred (InvalidRequestException) when calling the
  TestIdentityProvider operation: s-server-ID not configured for external
  auth
  ```

- --server-id パラメータのサーバー ID を入力した場合、実際の転送サーバーを特定しないパラメータを使用すると、次のエラーが表示されます。

  ```
  An error occurred (ResourceNotFoundException) when calling the
  TestIdentityProvider operation: Unknown server.
  ```

  サーバーが別の地域にある可能性があります。地域を指定するには、--region region-code(--region us-east-2 など)を追加して、「米国東部 (オハイオ)」のサーバーを指定します。

リクエストの構文

```
{
   "ServerId": "string",
   "ServerProtocol": "string",
   "SourceIp": "string",
   "UserName": "string",
   "UserPassword": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

ServerId

特定のサーバーにシステムで割り当てられた識別子。そのサーバーのユーザー認証方法は、ユーザー名とパスワードを使用してテストされます。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `s-([0-9a-f]{17})`

必須：はい

ServerProtocol

テストするファイル転送プロトコルのタイプ。

使用可能なプロトコルは次のとおりです。

- Secure Shell (SSH) File Transfer Protocol (SFTP)
- File Transfer Protocol Secure (FTPS)
- File Transfer Protocol (FTP)
- 適用性ステートメント 2 (AS2)

型: 文字列

有効な値：SFTP | FTP | FTPS | AS2

必須：いいえ

SourceIp

テスト対象のアカウントのソース IP アドレス。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 32 です。

パターン: \d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}

必須: いいえ

UserName

テスト対象のアカウント名。

型: 文字列

長さの制限: 最小長は 3 です。最大長は 100 です。

パターン: [\w][\w@.-]{2,99}

必須：はい

UserPassword

テスト対象のアカウントのパスワード。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 1,024 です。

必須: いいえ

レスポンスの構文

```
{
   "Message": "string",
   "Response": "string",
   "StatusCode": number,
   "Url": "string"
```

```
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

Message

テストの成否を示すメッセージ。

> **Note**
> 空の文字列が返された場合は、ユーザー名またはパスワードが間違っていたために認証に失敗した可能性が高いです。

型: 文字列

Response

API Gateway または Lambda 関数から返されるレスポンス。

型: 文字列

StatusCode

API Gateway または Lambda 関数からのレスポンスであるHTTPステータスコード。

タイプ: 整数

Url

ユーザーの認証に使用されるサービスのエンドポイント。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 255 です。

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

例

例

次のリクエストは、ユーザー名とパスワードの組み合わせが で使用する有効な ID であるというメッセージを ID プロバイダーから返します AWS Transfer Family。

リクエスト例

```
{
   "ServerID": "s-01234567890abcdef",
   "UserName": "my_user",
   "UserPassword": "MyPassword-1"
}
```

例

次のレスポンスは、テストが成功した場合のレスポンス例を示します。

レスポンス例

```
      "Response":"
      {\"homeDirectory\":\"/mybucket001\",\"homeDirectoryDetails\":null,
\"homeDirectoryType\":\"PATH\",\"posixProfile\":null,
      \"publicKeys\":\"[ssh-rsa-key]\",\"role\":\"arn:aws:iam::123456789012:role/
my_role\",\"policy\":null,\"username\":\"transferuser002\",
      \"identityProviderType\":null,\"userConfigMessage\":null)"}
      "StatusCode": "200",
      "Message": ""
```

例

以下のレスポンスは、ユーザーがアクセス権のある複数のグループに属していることを示します。

```
         "Response":"",
         "StatusCode":200,
         "Message":"More than one associated access found for user's groups."
```

例

API Gateway を使用してカスタム ID プロバイダーを作成および設定した場合、次のコマンドを入力するとユーザーをテストできます。

`aws transfer test-identity-provider --server-id` *`s-0123456789abcdefg`* `--user-name` *`myuser`*

ここで s-0123456789abcdefg は転送サーバー、myuser はカスタムユーザーのユーザー名です。

コマンドが正常に完了した場合、レスポンスは次のようになります。

- AWS アカウント ID は 012345678901
- ユーザーロールは user-role-api-gateway
- ホームディレクトリは myuser-bucket
- パブリックキーは public-key
- 呼び出し URL は invocation-URL です。

```
{
    "Response": "{\"Role\": \"arn:aws:iam::012345678901:role/user-role-api-gateway\",
\"HomeDirectory\": \"/myuser-bucket\",\"PublicKeys\": \"[public-key]\"}",
    "StatusCode": 200,
    "Message": "",
    "Url": "https://invocation-URL/servers/s-0123456789abcdefg/users/myuser/config"
}
```

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

UntagResource

Amazon リソースネーム (ARN) によって識別されるようなキーバリューペアをリソースからデタッチします。リソースは、ユーザー、サーバー、ロール、その他のエンティティです。

この呼び出しからレスポンスは返りません。

リクエストの構文

```
{
   "Arn": "string",
   "TagKeys": [ "string" ]
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

Arn

削除されるタグを含んでいるリソースの値。Amazon リソースネーム (ARN) は、サーバー、ユーザー、ロールなど、特定の AWS リソースの識別子です。

型: 文字列

長さの制限: 最小長は 20 です。最大長は 1600 です。

Pattern: `arn:\S+`

必須：はい

TagKeys

TagKeys は ARNsで、リソースをタイプ別にグループ化および検索するために使用できます。このメタデータは、任意の目的でリソースにアタッチできます。

型: 文字列の配列

配列メンバー：最小数は 1 項目です。最大数は 50 項目です。

長さの制限: 最小長は 0 です。最大長は 128 です。

必須: はい

レスポンス要素

アクションが成功した場合、サービスは空の HTTP 本文を持つ HTTP 200 レスポンスを返します。

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

例

例

次の例では、ファイル転送プロトコル対応サーバーのタグを削除します。

リクエスト例

```
{
   "Arn": "arn:aws:transfer:us-east-1:176354371281:server/s-01234567890abcdef",
   "TagKeys": "Europe" ]
}
```

例

この例では、 の使用例を示します UntagResource。

レスポンス例

```
        HTTP 200 response with an empty HTTP body.
```

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

UpdateAccess

ServerID および ExternalID パラメータで指定されたアクセス権についてパラメータを更新できます。

リクエストの構文

```
{
   "ExternalId": "string",
   "HomeDirectory": "string",
   "HomeDirectoryMappings": [
      {
         "Entry": "string",
         "Target": "string",
         "Type": "string"
      }
   ],
   "HomeDirectoryType": "string",
   "Policy": "string",
   "PosixProfile": {
      "Gid": number,
      "SecondaryGids": [ number ],
      "Uid": number
   },
   "Role": "string",
   "ServerId": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

ExternalId

ディレクトリ内の特定のグループを識別するために必要な一意の識別子。関連付けるグループのユーザーは、 を使用して、有効なプロトコル経由で Amazon S3 または Amazon EFS リソースにアクセスできます AWS Transfer Family。グループ名がわかっている場合は、Windows を使用して次のコマンドを実行して SID 値を表示できます PowerShell。

```
Get-ADGroup -Filter {samAccountName -like "YourGroupName*"} -Properties
* | Select SamAccountName,ObjectSid
```

そのコマンドで、 を Active Directory グループの名前YourGroupNameに置き換えます。

このパラメータの検証に使用される正規表現は、スペースを含まない大文字と小文字の英数字からなる文字列です。下線文字 (_) や =,.@:/- の文字を含めることもできます。

型: 文字列

長さの制限：最小長は 1 です。最大長は 256 です。

パターン：S-1-[\d-]+

必須：はい

HomeDirectory

ユーザーがクライアントを使用してサーバーにログインするときの、ユーザーのランディングディレクトリ (フォルダ)。

HomeDirectory の例は /bucket_name/home/mydirectory です。

> **Note**
>
> HomeDirectory パラメータは、HomeDirectoryType が PATH に設定されている場合のみ使用されます。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 1,024 です。

パターン: (|/.*)

必須: いいえ

HomeDirectoryMappings

ユーザーに表示する Amazon S3 または Amazon EFS のパスとキー、およびそれらをどのように表示するかを指定する論理ディレクトリマッピング。Entry と Target のペアを指定する必要があり、Entry はパスの表示方法を示し、Target は実際の Amazon S3 または Amazon EFS のパスです。ターゲットを指定しただけの場合は、そのまま表示されます。また、 AWS Identity and

Access Management (IAM) ロールが のパスへのアクセスを許可していることを確認する必要がありますTarget。この値は、HomeDirectoryType がLOGICALに設定されている場合にのみ設定できます。

以下は Entry と Target のペアの例です。

```
[ { "Entry": "/directory1", "Target": "/bucket_name/home/
mydirectory" } ]
```

ほとんどの場合、セッションポリシーの代わりにこの値を使用することで、指定されたホームディレクトリ (chroot) にユーザーをロックダウンできます。そのためには、Entry を/ に設定し、Target を HomeDirectory パラメータ値に設定します。

以下は、chroot のための Entry と Target のペアの例です。

```
[ { "Entry": "/", "Target": "/bucket_name/home/mydirectory" } ]
```

型: HomeDirectoryMapEntry オブジェクトの配列

配列メンバー : 最小数は 1 項目です。最大数は 50,000 項目です。

必須: いいえ

HomeDirectoryType

ユーザーがサーバーにログインするときにホームディレクトリにするランディングディレクトリ (フォルダ) のタイプ。これを PATH に設定した場合、ユーザーには、ファイル転送プロトコルクライアントに、絶対的な Amazon S3 バケットまたは現状の Amazon EFS パスが表示されます。これを LOGICAL に設定した場合、Amazon S3 または Amazon EFS パスをユーザーに表示する方法に関して、HomeDirectoryMappings でマッピングを指定する必要があります。

> **Note**
>
> HomeDirectoryType が LOGICAL の場合は、HomeDirectoryMappings パラメータを使用してマッピングを指定する必要があります。一方、HomeDirectoryType が PATH の場合は、HomeDirectory パラメータを使用して絶対パスを指定します。テンプレートに HomeDirectory と HomeDirectoryMappings の両方を含めることはできません。

型: 文字列

有効な値：PATH | LOGICAL

必須：いいえ

Policy

複数のユーザーで同じ AWS Identity and Access Management (IAM) ロールを使用できるようにするためのユーザーのセッションポリシー。このポリシーは、ユーザーアクセスのスコープを Amazon S3 バケットの一部に絞り込みます。このポリシー内に使用できる変数には、${Transfer:UserName}、${Transfer:HomeDirectory}、${Transfer:HomeBucket} があります。

> **Note**
>
> このポリシーは、ServerId ドメインが Amazon S3 の場合にのみ適用されます。Amazon EFS はセッション· ポリシーを使用しません。
> セッションポリシーの場合、 はポリシーの Amazon リソースネーム (ARN) ではなく、ポリシーを JSON BLOB として AWS Transfer Family 保存します。JSON blob としてポリシーを保存し、Policy 因数に渡します。
> セッションポリシーの例については、「セッションポリシーの例」を参照してください。
> 詳細については、 AWS「 Security Token Service API リファレンスAssumeRole」の「」を参照してください。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 2048 です。

必須: いいえ

PosixProfile

ユーザーの Amazon EFS ファイルシステムへのアクセスを制御する、ユーザー ID (Uid Gid)、グループ ID (SecondaryGids)、およびセカンダリグループ ID () を含む完全な POSIX ID。ファイルシステム内のファイルとディレクトリに設定される POSIX アクセス許可によって、Amazon EFS ファイルシステムとの間でファイルを転送するときにユーザーが得るアクセスのレベルが決まります。

型: PosixProfile オブジェクト

必須: いいえ

Role

Amazon S3 バケットまたは Amazon EFS ファイルシステムへのユーザーのアクセスを制御する AWS Identity and Access Management (IAM) ロールの Amazon リソースネーム (ARN)。Amazon S3 このロールにアタッチされたポリシーにより、ファイルを Amazon S3 バケットまたは Amazon EFS ファイルシステム間で転送する際の、ユーザーに付与するアクセスレベルが決定されます。IAM ロールには、ユーザーの転送リクエストを処理する際に、サーバーによるリソースへのアクセスを許可する信頼関係も含まれる必要があります。

型: 文字列

長さの制限: 最小長は 20 です。最大長は 2,048 です。

パターン: `arn:.*role/\S+`

必須: いいえ

ServerId

サーバーインスタンスにシステムで割り当てられた一意の識別子。これはユーザーを追加したサーバーに固有です。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `s-([0-9a-f]{17})`

必須：はい

レスポンスの構文

```
{
   "ExternalId": "string",
   "ServerId": "string"
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

ExternalId

AWS Transfer Family を使用して、有効になっているプロトコルを介して Amazon S3 または Amazon EFS リソースにユーザーがアクセスできるグループの外部識別子。

型: 文字列

長さの制限：最小長は 1 です。最大長は 256 です。

パターン：`S-1-[\d-]+`

ServerId

ユーザーが接続しているサーバーの識別子。

型: 文字列

長さの制限: 固定長は 19 です。

パターン：`s-([0-9a-f]{17})`

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceExistsException

要求されたリソースは存在しないか、コマンドに指定されたリージョン以外のリージョンに存在します。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

ThrottlingException

リクエストのスロットリングにより、リクエストが拒否されました。

HTTP ステータスコード：400

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

UpdateAgreement

既存の契約の一部のパラメータを更新します。更新する契約書のAgreementIdとServerIdを、更新するパラメータの新しい値を指定します。

リクエストの構文

```
{
   "AccessRole": "string",
   "AgreementId": "string",
   "BaseDirectory": "string",
   "Description": "string",
   "LocalProfileId": "string",
   "PartnerProfileId": "string",
   "ServerId": "string",
   "Status": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

AccessRole

コネクターは、AS2 または SFTP プロトコルを使用してファイルを送信するために使用されます。アクセスロールには、使用する AWS Identity and Access Management ロールの Amazon リソースネーム (ARN) を指定します。

AS2 コネクタ用

AS2 では、StartFileTransfer を呼び出してリクエストパラメータ にファイルパスを指定することでファイルを送信できます。SendFilePathsファイルの親ディレクトリ (例えば --send-file-paths /bucket/dir/file.txt の場合、親ディレクトリは /bucket/dir/) を使用して、処理済みの AS2 メッセージファイルを一時的に保存し、パートナーから受け取った MDN を保存して、送信の関連メタデータを含む最終的な JSON ファイルを記述します。そのため、AccessRole は、StartFileTransfer リクエストで使用されるファイルの場所の親ディレクトリに対する読み取り/書き込みアクセスを提供する必要があります。さら

に、StartFileTransfer で送信するファイルの親ディレクトリに対する読み取り/書き込みアクセスを提供する必要があります。

AS2 コネクタに Basic 認証を使用している場合、アクセスロールにはシークレットの secretsmanager:GetSecretValue 権限が必要です。Secrets Manager の マネージドキーではなく、カスタマー AWS マネージドキーを使用してシークレットが暗号化されている場合、ロールにはそのキーに対する kms:Decrypt アクセス許可も必要です。

SFTP コネクタ用

アクセスロールが、StartFileTransfer リクエストで使用されるファイルロケーションの親ディレクトリへの読み取りおよび書き込みアクセスを提供していることを確認します。さらに、ロールが にアクセスsecretsmanager:GetSecretValue許可を付与していることを確認します AWS Secrets Manager。

型: 文字列

長さの制限: 最小長は 20 です。最大長は 2,048 です。

パターン: arn:.*role/\S+

必須: いいえ

AgreementId

契約の一意の識別子。この識別子は、契約を作成するときに返されます。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: a-([0-9a-f]{17})

必須：はい

BaseDirectory

転送されるファイルのランディングディレクトリ (フォルダ) を変更するには、使用したいバケットフォルダを指定します。例えば、*/DOC-EXAMPLE-BUCKET/home/mydirectory* 。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 1,024 です。

パターン: (|/.*)

必須: いいえ

Description

既存の説明を置き換えるには、契約の簡単な説明を入力します。

型: 文字列

長さの制限: 最小長は 1 です。最大長は 200 です。

Pattern: `[\p{Graph}]+`

必須: いいえ

LocalProfileId

AS2 ローカルプロファイルの一意の識別子です。

ローカルプロファイル識別子を変更するには、ここに新しい値を入力します。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `p-([0-9a-f]{17})`

必須: いいえ

PartnerProfileId

パートナープロファイルの一意の識別子。パートナープロファイル ID を変更するには、ここに新しい値を入力します。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `p-([0-9a-f]{17})`

必須: いいえ

ServerId

サーバーインスタンスにシステムで割り当てられた一意の識別子。これは、契約が使用する特定のサーバーです。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `s-([0-9a-f]{17})`

必須：はい

Status

契約のステータスを更新できます。無効になっている契約を有効化することも、その逆を行うこともできます。

型: 文字列

有効な値：`ACTIVE | INACTIVE`

必須：いいえ

レスポンスの構文

```
{
   "AgreementId": "string"
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

AgreementId

契約の一意の識別子。この識別子は、契約を作成するときに返されます。

型: 文字列

長さの制限: 固定長は 19 です。

パターン：`a-([0-9a-f]{17})`

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceExistsException

要求されたリソースは存在しないか、コマンドに指定されたリージョン以外のリージョンに存在します。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

ThrottlingException

リクエストのスロットリングにより、リクエストが拒否されました。

HTTP ステータスコード：400

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET

- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

UpdateCertificate

証明書の有効および無効な日付を更新します。

リクエストの構文

```
{
   "ActiveDate": number,
   "CertificateId": "string",
   "Description": "string",
   "InactiveDate": number
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

ActiveDate

証明書がいつアクティブになるかを指定するオプションの日付です。

型: タイムスタンプ

必須: いいえ

CertificateId

更新している証明書オブジェクトの識別子。

型: 文字列

長さの制限: 22 の固定長

Pattern: `cert-([0-9a-f]{17})`

必須：はい

Description

証明書を識別するための簡単な説明。

型: 文字列

長さの制限: 最小長は 1 です。最大長は 200 です。

Pattern: `[\p{Graph}]+`

必須: いいえ

InactiveDate

証明書がいつ非アクティブになるかを指定するオプションの日付です。

型: タイムスタンプ

必須: いいえ

レスポンスの構文

```
{
   "CertificateId": "string"
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

CertificateId

更新している証明書オブジェクトの識別子を返します。

型: 文字列

長さの制限: 22 の固定長

パターン：`cert-([0-9a-f]{17})`

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

ThrottlingException

リクエストのスロットリングにより、リクエストが拒否されました。

HTTP ステータスコード：400

例

例

以下は、証明書の有効な日付を更新する例であり、有効な日付を2022年1月16日のUTC時刻16:12:07に設定しています（UTC -5時間）。

リクエスト例

```
aws transfer update-certificate --certificate-id c-abcdefg123456hijk --active-date
 2022-01-16T16:12:07-05:00
```

例

以下は、この API コールに対するレスポンスのサンプルです。

レスポンス例

```
"CertificateId": "c-abcdefg123456hijk"
```

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

UpdateConnector

既存のコネクタのパラメータの一部を更新します。更新するコネクタの ConnectorId を、更新するパラメータの新しい値とともに指定します。

リクエストの構文

```
{
   "AccessRole": "string",
   "As2Config": {
      "BasicAuthSecretId": "string",
      "Compression": "string",
      "EncryptionAlgorithm": "string",
      "LocalProfileId": "string",
      "MdnResponse": "string",
      "MdnSigningAlgorithm": "string",
      "MessageSubject": "string",
      "PartnerProfileId": "string",
      "SigningAlgorithm": "string"
   },
   "ConnectorId": "string",
   "LoggingRole": "string",
   "SecurityPolicyName": "string",
   "SftpConfig": {
      "TrustedHostKeys": [ "string" ],
      "UserSecretId": "string"
   },
   "Url": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

AccessRole

コネクターは、AS2 または SFTP プロトコルを使用してファイルを送信するために使用されます。アクセスロールには、使用する AWS Identity and Access Management ロールの Amazon リソースネーム (ARN) を指定します。

AS2 コネクタ用

AS2 では、StartFileTransfer を呼び出してリクエストパラメータ にファイルパスを指定することでファイルを送信できます。SendFilePathsファイルの親ディレクトリ (例えば --send-file-paths /bucket/dir/file.txt の場合、親ディレクトリは /bucket/dir/) を使用して、処理済みの AS2 メッセージファイルを一時的に保存し、パートナーから受け取った MDN を保存して、送信の関連メタデータを含む最終的な JSON ファイルを記述します。そのため、AccessRole は、StartFileTransfer リクエストで使用されるファイルの場所の親ディレクトリに対する読み取り/書き込みアクセスを提供する必要があります。さらに、StartFileTransfer で送信するファイルの親ディレクトリに対する読み取り/書き込みアクセスを提供する必要があります。

AS2 コネクタに Basic 認証を使用している場合、アクセスロールにはシークレットの secretsmanager:GetSecretValue 権限が必要です。Secrets Manager の マネージドキーではなく、カスタマー AWS マネージドキーを使用してシークレットが暗号化されている場合、ロールにはそのキーに対する アクセスkms:Decrypt許可も必要です。

SFTP コネクタ用

アクセスロールが、StartFileTransfer リクエストで使用されるファイルロケーションの親ディレクトリへの読み取りおよび書き込みアクセスを提供していることを確認します。さらに、ロールが にアクセスsecretsmanager:GetSecretValue許可を付与していることを確認します AWS Secrets Manager。

型: 文字列

長さの制限: 最小長は 20 です。最大長は 2,048 です。

パターン: arn:.*role/\S+

必須: いいえ

As2Config

AS2 コネクタオブジェクトのパラメータを含む構造体。

タイプ : As2ConnectorConfig オブジェクト

必須: いいえ

ConnectorId

コネクタの一意の識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `c-([0-9a-f]{17})`

必須：はい

LoggingRole

コネクタが Amazon S3 イベントの CloudWatch ログ記録をオンにできるようにする (IAM) ロールの Amazon リソースネーム AWS Identity and Access Management (ARN)。 Amazon S3 設定すると、 CloudWatch ログにコネクタアクティビティを表示できます。

型: 文字列

長さの制限: 最小長は 20 です。最大長は 2,048 です。

パターン: `arn:.*role/\S+`

必須: いいえ

SecurityPolicyName

コネクタのセキュリティポリシーの名前を指定します。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 100 です。

Pattern: `TransferSFTPConnectorSecurityPolicy-[A-Za-z0-9-]+`

必須: いいえ

SftpConfig

SFTP コネクタオブジェクトのパラメータを含む構造体。

タイプ：SftpConnectorConfig オブジェクト

必須: いいえ

Url

パートナーの AS2 または SFTP エンドポイントの URL。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 255 です。

必須: いいえ

レスポンスの構文

```
{
   "ConnectorId": "string"
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

ConnectorId

更新しているコネクタ オブジェクトの識別子を返します。

型: 文字列

長さの制限: 固定長は 19 です。

パターン：`c-([0-9a-f]{17})`

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceExistsException

要求されたリソースは存在しないか、コマンドに指定されたリージョン以外のリージョンに存在します。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

ThrottlingException

リクエストのスロットリングにより、リクエストが拒否されました。

HTTP ステータスコード：400

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python

- AWS SDK for Ruby V3

UpdateHostKey

パラメータ ServerId と HostKeyId で指定されたホストキーの説明を更新します。

リクエストの構文

```
{
   "Description": "string",
   "HostKeyId": "string",
   "ServerId": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

Description

ホストキーの説明が更新されました。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 200 です。

Pattern: [\p{Print}]*

必須：はい

HostKeyId

更新するホストキーの識別子。

型: 文字列

長さの制限: 固定長は 25 です。

Pattern: hostkey-[0-9a-f]{17}

必須：はい

ServerId

更新するホストキーを含むサーバーの識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `s-([0-9a-f]{17})`

必須：はい

レスポンスの構文

```
{
   "HostKeyId": "string",
   "ServerId": "string"
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

HostKeyId

更新されたホストキーのホストキー識別子を返します。

型: 文字列

長さの制限: 固定長は 25 です。

Pattern: `hostkey-[0-9a-f]{17}`

ServerId

更新されたホストキーを含むサーバーのサーバー識別子を返します。

型: 文字列

長さの制限: 固定長は 19 です。

パターン：s-([0-9a-f]{17})

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

ThrottlingException

リクエストのスロットリングにより、リクエストが拒否されました。

HTTP ステータスコード：400

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET

- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

UpdateProfile

既存のプロファイルのパラメータの一部を更新します。更新するプロファイルの ProfileId と、更新するパラメータの新しい値を指定します。

リクエストの構文

```
{
   "CertificateIds": [ "string" ],
   "ProfileId": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

CertificateIds

インポートされた証明書の識別子の配列です。この識別子は、プロファイルやパートナープロファイルの操作に使用します。

タイプ：文字列の配列

長さの制限: 22 の固定長

Pattern: cert-([0-9a-f]{17})

必須: いいえ

ProfileId

更新するプロファイル オブジェクトの識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: p-([0-9a-f]{17})

必須：はい

レスポンスの構文

```
{
   "ProfileId": "string"
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

ProfileId

更新中のプロファイルの識別子を返します。

型: 文字列

長さの制限: 固定長は 19 です。

パターン：`p-([0-9a-f]{17})`

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

ThrottlingException

リクエストのスロットリングにより、リクエストが拒否されました。

HTTP ステータスコード：400

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

UpdateServer

ファイル転送プロトコル対応サーバーの作成後にそのサーバーのプロパティを更新します。

UpdateServer コールは更新されたサーバーの ServerId を返します。

リクエストの構文

```
{
   "Certificate": "string",
   "EndpointDetails": {
      "AddressAllocationIds": [ "string" ],
      "SecurityGroupIds": [ "string" ],
      "SubnetIds": [ "string" ],
      "VpcEndpointId": "string",
      "VpcId": "string"
   },
   "EndpointType": "string",
   "HostKey": "string",
   "IdentityProviderDetails": {
      "DirectoryId": "string",
      "Function": "string",
      "InvocationRole": "string",
      "SftpAuthenticationMethods": "string",
      "Url": "string"
   },
   "LoggingRole": "string",
   "PostAuthenticationLoginBanner": "string",
   "PreAuthenticationLoginBanner": "string",
   "ProtocolDetails": {
      "As2Transports": [ "string" ],
      "PassiveIp": "string",
      "SetStatOption": "string",
      "TlsSessionResumptionMode": "string"
   },
   "Protocols": [ "string" ],
   "S3StorageOptions": {
      "DirectoryListingOptimization": "string"
   },
   "SecurityPolicyName": "string",
   "ServerId": "string",
   "StructuredLogDestinations": [ "string" ],
   "WorkflowDetails": {
```

```
      "OnPartialUpload": [
         {
            "ExecutionRole": "string",
            "WorkflowId": "string"
         }
      ],
      "OnUpload": [
         {
            "ExecutionRole": "string",
            "WorkflowId": "string"
         }
      ]
   }
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

Certificate

AWS Certificate Manager (ACM) 証明書の Amazon リソースネーム (ARN)。`Protocols` が FTPS に設定されている場合は必須です。

新しいパブリック証明書をリクエストするには、 AWS Certificate Manager ユーザーガイドの「パブリック証明書のリクエスト」を参照してください。

既存の証明書を ACM にインポートするには、 AWS Certificate Manager ユーザーガイドの「ACM への証明書のインポート」を参照してください。

プライベート IP アドレス経由で FTPS を使用できるようにプライベート証明書をリクエストするには、 AWS Certificate Manager ユーザーガイドの「プライベート証明書をリクエストする」を参照してください。

以下の暗号化アルゴリズムとキーサイズの証明書がサポートされています。

- 2048 ビット RSA (RSA_2048)
- 4096 ビット RSA (RSA_4096)

- 楕円素数曲線 256 ビット (EC_Prime256v1)
- 楕円素数曲線 384 ビット (EC_secp384R1)
- 楕円素数曲線 521 ビット (EC_secp521R1)

> **Note**
>
> 証明書は、FQDN または IP アドレスが指定され、発行者に関する情報が記載された有効な SSL/TLS X.509 バージョン 3 の証明書である必要があります。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 1600 です。

必須: いいえ

EndpointDetails

サーバーに設定された Virtual Private Cloud (VPC) エンドポイント設定。VPC 内でエンドポイントをホストする場合、VPC 内のリソースにのみアクセスできるようにすることも、Elastic IP アドレスをアタッチしてインターネット経由でクライアントにアクセスできるようにすることもできます。VPC のデフォルトのセキュリティグループは、エンドポイントに自動的に割り当てられます。

型: EndpointDetails オブジェクト

必須: いいえ

EndpointType

サーバーで使用したいエンドポイントのタイプ。サーバーのエンドポイントをパブリックアクセス (PUBLIC) にするか、VPC 内でホストするかを選択できます。VPC でホストされているエンドポイントを使用すると、サーバーとリソースへのアクセスを VPC 内のみに制限したり、Elastic IP アドレスを直接アタッチしてインターネット向けにするといった選択ができます。

> **Note**
>
> 2021 年 5 月 19 日以降、 AWS 2021 年 5 月 19 日より前にアカウントEndpointType=VPC_ENDPOINTで を使用してサーバーを作成していない場合、そのサーバーを作成することはできません。2021 年 5 月 19 日 AWS以前にアカウントに

EndpointType=VPC_ENDPOINTで既にサーバーを作成している場合、影響を受けません。この日付を過ぎたら EndpointType=VPC を使用します。
詳細については、「VPC_ENDPOINT のサポート終了」を参照してください。
VPC を EndpointType として使用することをお勧めします。このエンドポイントタイプでは、最大 3 つの Elastic IPv4 アドレス (BYO IP を含む) をサーバーのエンドポイントに直接関連付けて、VPC セキュリティグループを使用してクライアントのパブリック IP アドレスでトラフィックを制限できます。EndpointType を VPC_ENDPOINT に設定した場合、これは不可能です。

型: 文字列

有効な値：PUBLIC | VPC | VPC_ENDPOINT

必須：いいえ

HostKey

SFTP 対応サーバーで使用する RSA、ECDSA、または ED25519 プライベートキー。キーをローテーションしたい場合や、異なるアルゴリズムを使用するアクティブキーのセットが必要な場合に備えて、複数のホストキーを追加できます。

パスフレーズなしで RSA 2048 ビットキーを生成するには、以下のコマンドを使用する：

```
ssh-keygen -t rsa -b 2048 -N "" -m PEM -f my-new-server-key.
```

-b オプションには最小値 2048 を使用してください。3072 または 4096 を使用すると、より強力なキーを作成できます。

パスフレーズなしで ECDSA 256 ビット鍵を生成するには、以下のコマンドを使用する：

```
ssh-keygen -t ecdsa -b 256 -N "" -m PEM -f my-new-server-key.
```

ECDSA の -b オプションの有効値は 256、384、521 です。

パスフレーズなしで ED25519 鍵を生成するには、以下のコマンドを使用する：

```
ssh-keygen -t ed25519 -N "" -f my-new-server-key.
```

これらのコマンドのすべてについて、 を任意の文字列my-new-server-keyに置き換えることができます。

> **Important**
>
> 既存のユーザーを既存のSFTP 対応サーバーから新しいサーバーに移行する計画がない場合、ホストキーを更新しないでください。サーバーのホストキーを誤って変更することは、破壊的な操作になり得えます。

詳細については、「 ユーザーガイド」の「SFTP 対応サーバーのホストキーの更新 AWS Transfer Family 」を参照してください。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 4,096 です。

必須: いいえ

IdentityProviderDetails

カスタマーの認証 API メソッドを呼び出す際に必要なすべての情報を含む配列。

型: IdentityProviderDetails オブジェクト

必須: いいえ

LoggingRole

サーバーが Amazon S3 または Amazon EFSevents の Amazon CloudWatch ログ記録をオンにできるようにする AWS Identity and Access Management (IAM) ロールの Amazon リソースネーム (ARN)。設定すると、 CloudWatch ログにユーザーアクティビティを表示できます。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 2,048 です。

パターン: `(|arn:.*role/\S+)`

必須: いいえ

PostAuthenticationLoginBanner

ユーザーがサーバーに接続するときに表示する文字列を指定します。この文字列はユーザーが認証した後で表示されます。

> **Note**
>
> SFTP プロトコルは認証後の表示バナーをサポートしていません。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 4096 です。

パターン: `[\x09-\x0D\x20-\x7E]*`

必須: いいえ

PreAuthenticationLoginBanner

ユーザーがサーバーに接続するときに表示する文字列を指定します。この文字列はユーザーが認証される前に表示されます。例えば、次のバナーはシステムの使用に関する詳細を表示します。

```
This system is for the use of authorized users only. Individuals using
this computer system without authority, or in excess of their authority,
are subject to having all of their activities on this system monitored
and recorded by system personnel.
```

型: 文字列

長さの制限: 最小長は 0 です。最大長は 4096 です。

パターン: `[\x09-\x0D\x20-\x7E]*`

必須: いいえ

ProtocolDetails

サーバー用に構成されたプロトコル設定。

- パッシブモード (FTP および FTPS プロトコル用) を示すには、`PassiveIp` パラメータを使用します。ファイアウォール、ルーター、ロードバランサーの外部 IP アドレスなど、クアッドドット形式の単一 IPv4 アドレスを入力します。
- S3 バケットにアップロードしているファイルに対して、クライアントが `SETSTAT` コマンドの使用を試みた時に発生するエラーを無視するには、`SetStatOption` パラメータを使用します。SFTP クライアントを変更せずに AWS Transfer Family サーバー

でSETSTATコマンドを無視してファイルをアップロードするには、値を に設定しますENABLE_NO_OP。SetStatOption パラメータを に設定するとENABLE_NO_OP、Transfer Family は Amazon CloudWatch Logs にログエントリを生成し、クライアントがいつSETSTAT呼び出しを行っているかを判断できるようにします。

- AWS Transfer Family サーバーが一意のセッション ID を介して最近ネゴシエートされたセッションを再開するかどうかを確認するには、 TlsSessionResumptionModeパラメータを使用します。
- As2Transports は、AS2 メッセージの転送方法を示します。現在は、HTTP のみがサポートされます。

タイプ : ProtocolDetails オブジェクト

必須: いいえ

Protocols

ファイル転送プロトコルクライアントがサーバーのエンドポイントに接続できる 1 つまたは複数のファイル転送プロトコルを指定します。使用可能なプロトコルは次のとおりです。

- SFTP(Secure Shell (SSH) File Transfer Protocol): SSH 経由のファイル転送
- FTPS (File Transfer Protocol Secure): TLS 暗号化によるファイル転送
- FTP (File Transfer Protocol): 暗号化されていないファイル転送
- AS2 (適用性ステートメント 2): 構造化 business-to-business データの転送に使用されます

> **Note**
>
> - を選択した場合はFTPS、 AWS Certificate Manager (ACM) に保存されている証明書を選択する必要があります。この証明書は、クライアントが FTPS 経由でサーバーに接続するときにサーバーを識別するために使用されます。
> - Protocol に FTP または FTPS が含まれる場合、EndpointType は VPC でなければならず、IdentityProviderType は AWS_DIRECTORY_SERVICE、AWS_LAMBDA または API_GATEWAY でなければなりません。
> - Protocol に FTPが含まれる場合、AddressAllocationIds は関連付けられません。
> - Protocol が SFTP のみに設定されている場合、EndpointType は PUBLIC に設定でき、IdentityProviderType はサポートされている ID タイプ (SERVICE_MANAGED、AWS_DIRECTORY_SERVICE、AWS_LAMBDA、または API_GATEWAY) のいずれかに設定できます。

- Protocol が AS2 を含む場合、EndpointType は VPC でなければならず、ドメインは、Amazon S3 でなければなりません。

タイプ：文字列の配列

配列メンバー：最小数は 1 項目です。最大数は 4 項目です。

有効な値：SFTP | FTP | FTPS | AS2

必須：いいえ

S3StorageOptions

Amazon S3 ディレクトリのパフォーマンスを最適化するかどうかを指定します。これはデフォルトでは無効になっています。

デフォルトでは、ホームディレクトリマッピングの は TYPEですDIRECTORY。
このオプションを有効にすると、マッピングにファイルターゲットを設定するHomeDirectoryMapEntryTypeFILE場合は、 を明示的に に設定する必要があります。

タイプ：S3StorageOptions オブジェクト

必須: いいえ

SecurityPolicyName

サーバーのセキュリティポリシーの名前を指定します。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 100 です。

Pattern: Transfer[A-Za-z0-9]*SecurityPolicy-[A-Za-z0-9-]+

必須: いいえ

ServerId

Transfer Family ユーザーが割り当てられているサーバーインスタンスのシステム割り当て一意識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `s-([0-9a-f]{17})`

必須：はい

StructuredLogDestinations

サーバーログの送信先となるロググループを指定します。

ロググループを指定するには、既存のロググループの ARN を指定する必要があります。この場合、ロググループの形式は次のようになります。

`arn:aws:logs:region-name:amazon-account-id:log-group:log-group-name:*`

例えば、次のようになります: `arn:aws:logs:us-east-1:111122223333:log-group:mytestgroup:*`

以前にサーバーのロググループを指定したことがある場合は、`update-server` 呼び出し時にこのパラメータに空の値を指定することで、そのロググループをクリアし、構造化ロギングを事実上無効にすることができます。例:

```
update-server --server-id s-1234567890abcdef0 --structured-log-destinations
```

タイプ：文字列の配列

配列メンバー: 最小数は 0 項目です。最大数は 1 項目です。

長さの制限: 最小長は 20 です。最大長は 1600 です。

Pattern: `arn:\S+`

必須: いいえ

WorkflowDetails

割り当てるワークフローのワークフロー ID とワークフローの実行に使用する実行ロールを指定します。

ファイルのアップロード完了時に実行するワークフローに加えて、部分的なアップロードで実行するワークフローのワークフロー ID (および実行ロール) も `WorkflowDetails` に含めることができます。部分的なアップロードは、ファイルのアップロード中にサーバーセッションが切断されたときに発生します。

関連するワークフローをサーバーから削除するには、以下の例に示すように、空の OnUpload オブジェクトを提供します。

```
aws transfer update-server --server-id s-01234567890abcdef --workflow-details '{"OnUpload":[]}'
```

タイプ：WorkflowDetails オブジェクト

必須: いいえ

レスポンスの構文

```
{
   "ServerId": "string"
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

ServerId

Transfer Family ユーザーが割り当てられているサーバーの、システムから割り当てられた一意の識別子。

型: 文字列

長さの制限: 固定長は 19 です。

パターン：`s-([0-9a-f]{17})`

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

AccessDeniedException

このアクションを実行する十分なアクセス権限がありません。

HTTP ステータスコード：400

ConflictException

この例外は、エンドポイントタイプが VPC であるファイル転送プロトコル対応サーバーの場合に `UpdateServer` が呼び出され、サーバーの `VpcEndpointID` が利用可能な状態でない場合にスローされます。

HTTP ステータスコード：400

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceExistsException

要求されたリソースは存在しないか、コマンドに指定されたリージョン以外のリージョンに存在します。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

ThrottlingException

リクエストのスロットリングにより、リクエストが拒否されました。

HTTP ステータスコード：400

例

例

次の例では、サーバーのロールを更新します。

リクエスト例

```
{
   "EndpointDetails": {
   "VpcEndpointId": "vpce-01234f056f3g13",
   "LoggingRole": "CloudWatchS3Events",
   "ServerId": "s-01234567890abcdef"
   }
}
```

例

以下の例では、サーバーから関連するワークフローを削除します。

リクエスト例

```
aws transfer update-server --server-id s-01234567890abcdef --workflow-details
 '{"OnUpload":[]}'
```

例

これは、この API コールに対するレスポンスのサンプルです。

レスポンス例

```
{
   "ServerId": "s-01234567890abcdef"
}
```

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス

- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

UpdateUser

新しいプロパティをユーザーに割り当てます。渡されたパラメータは、指定する UserName および ServerId のホームディレクトリ、ロール、およびポリシーのいずれかまたはすべてを変更します。

レスポンスは更新されたユーザーの ServerId と UserName を返します。

コンソールでは、ユーザーを作成または更新するときに制限付き を選択できます。これにより、ユーザーはホームディレクトリ外のものにアクセスできなくなります。この動作を設定するプログラム的な方法は、ユーザーを更新することです。を HomeDirectoryTypeに設定しLOGICAL、HomeDirectoryMappingsをルート (/) Entryとして指定し、 をホームディレクトリTargetとして指定します。

例えば、ユーザーのホームディレクトリが の場合/test/admin-user、次のコマンドはユーザーを更新して、コンソールの設定で制限付きフラグが選択されていることを表示します。

```
aws transfer update-user --server-id <server-id> --user-name admin-user --
home-directory-type LOGICAL --home-directory-mappings "[{\"Entry\":\"/\",
\"Target\":\"/test/admin-user\"}]"
```

リクエストの構文

```
{
   "HomeDirectory": "string",
   "HomeDirectoryMappings": [
      {
         "Entry": "string",
         "Target": "string",
         "Type": "string"
      }
   ],
   "HomeDirectoryType": "string",
   "Policy": "string",
   "PosixProfile": {
      "Gid": number,
      "SecondaryGids": [ number ],
      "Uid": number
   },
   "Role": "string",
   "ServerId": "string",
```

```
    "UserName": "string"
}
```

リクエストパラメータ

すべてのアクションに共通のパラメータの詳細については、「共通パラメータ」を参照してください。

リクエストは以下の JSON 形式のデータを受け入れます。

HomeDirectory

ユーザーがクライアントを使用してサーバーにログインするときの、ユーザーのランディングディレクトリ (フォルダ)。

HomeDirectory の例は /bucket_name/home/mydirectory です。

> **Note**
>
> HomeDirectory パラメータは、HomeDirectoryType が PATH に設定されている場合のみ使用されます。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 1,024 です。

パターン: (|/.*)

必須: いいえ

HomeDirectoryMappings

ユーザーに表示する Amazon S3 または Amazon EFS のパスとキー、およびそれらをどのように表示するかを指定する論理ディレクトリマッピング。Entry と Target のペアを指定する必要があり、Entry はパスの表示方法を示し、Target は実際の Amazon S3 または Amazon EFS のパスです。ターゲットを指定しただけの場合は、そのまま表示されます。また、 AWS Identity and Access Management (IAM) ロールが のパスへのアクセスを提供することを確認する必要がありますTarget。この値は、HomeDirectoryType がLOGICALに設定されている場合にのみ設定できます。

以下は Entry と Target のペアの例です。

```
[ { "Entry": "/directory1", "Target": "/bucket_name/home/
mydirectory" } ]
```

ほとんどの場合、セッションポリシーの代わりにこの値を使用することで、指定されたホームディレクトリ (chroot) にユーザーをロックダウンできます。これを行うには、 Entryを '/' に設定し、 Target を HomeDirectory パラメータ値に設定します。

以下は、chroot についての Entry と Target のペアの例です。

```
[ { "Entry": "/", "Target": "/bucket_name/home/mydirectory" } ]
```

型: HomeDirectoryMapEntry オブジェクトの配列

配列メンバー：最小数は 1 項目です。最大数は 50,000 項目です。

必須: いいえ

HomeDirectoryType

ユーザーがサーバーにログインするときにホームディレクトリにするランディングディレクトリ (フォルダ) のタイプ。これを PATH に設定した場合、ユーザーには、ファイル転送プロトコルクライアントに、絶対的な Amazon S3 バケットまたは現状の Amazon EFS パスが表示されます。これを LOGICAL に設定した場合、Amazon S3 または Amazon EFS パスをユーザーに表示する方法に関して、HomeDirectoryMappings でマッピングを指定する必要があります。

> **Note**
>
> HomeDirectoryType が LOGICAL の場合は、HomeDirectoryMappings パラメータを使用してマッピングを指定する必要があります。一方、HomeDirectoryType が PATH の場合は、HomeDirectory パラメータを使用して絶対パスを指定します。テンプレートに HomeDirectory と HomeDirectoryMappings の両方を含めることはできません。

型: 文字列

有効な値：PATH | LOGICAL

必須：いいえ

Policy

複数のユーザーで同じ AWS Identity and Access Management (IAM) ロールを使用できるように、ユーザーのセッションポリシー。このポリシーは、ユーザーアクセスのスコープを Amazon S3 バケットの一部に絞り込みます。このポリシー内に使用できる変数には、`${Transfer:UserName}`、`${Transfer:HomeDirectory}`、`${Transfer:HomeBucket}` があります。

> **Note**
>
> このポリシーは、`ServerId` ドメインが Amazon S3 の場合にのみ適用されます。Amazon EFS はセッション・ポリシーを使用しません。
> セッションポリシーの場合、 はポリシーの Amazon リソースネーム (ARN) ではなく、ポリシーを JSON BLOB として AWS Transfer Family 保存します。JSON blob としてポリシーを保存し、`Policy` 因数に渡します。
> セッションポリシーの例については、「セッションポリシーの例」を参照してください。
> 詳細については、 AWS 「 Security Token Service API リファレンスAssumeRole」の「」を参照してください。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 2048 です。

必須: いいえ

PosixProfile

Amazon Elastic File System (Amazon EFS) ファイルシステムへのユーザーのアクセスを制御する、ユーザー ID (`Uid`)、グループ ID (`Gid`)、およびセカンダリグループ ID (`SecondaryGids`) を含む、完全な POSIX ID を指定します。ファイルシステム内のファイルとディレクトリに設定される POSIX アクセス許可によって、Amazon EFS ファイルシステムとの間でファイルを転送するときにユーザーが得るアクセスのレベルが決まります。

型: PosixProfile オブジェクト

必須: いいえ

Role

Amazon S3 バケットまたは Amazon EFS ファイルシステムへのユーザーのアクセスを制御する AWS Identity and Access Management (IAM) ロールの Amazon リソースネーム (ARN)。この

ロールにアタッチされたポリシーにより、ファイルを Amazon S3 バケットまたは Amazon EFS ファイルシステム間で転送する際の、ユーザーに付与するアクセスレベルが決定されます。IAM ロールには、ユーザーの転送リクエストを処理する際に、サーバーによるリソースへのアクセスを許可する信頼関係も含まれる必要があります。

型: 文字列

長さの制限: 最小長は 20 です。最大長は 2,048 です。

パターン: `arn:.*role/\S+`

必須: いいえ

ServerId

ユーザーが割り当てられている Transfer Family サーバーインスタンスの、システムから割り当てられた一意の識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `s-([0-9a-f]{17})`

必須：はい

UserName

ユーザーを識別する一意の文字列で、`ServerId` による指定に従いサーバーに関連付けられています。このユーザー名は、最小 3 文字、最大 100 文字にする必要があります。有効な文字は a ~ z、A ~ Z、0 ~ 9、アンダースコア (_)、ハイフン (-)、ピリオド (。)、アットマーク (@) です。ユーザー名をハイフン、ピリオド、アットマークで始めることはできません。

型: 文字列

長さの制限: 最小長は 3 です。最大長は 100 です。

パターン: `[\w][\w@.-]{2,99}`

必須：はい

レスポンスの構文

```
{
```

```
    "ServerId": "string",
    "UserName": "string"
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

ServerId

アカウントが割り当てられている Transfer Family サーバーインスタンスの、システムから割り当てられた一意の識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `s-([0-9a-f]{17})`

UserName

リクエストで指定されたサーバーインスタンスに割り当てられているユーザーの一意の識別子。

型: 文字列

長さの制限: 最小長は 3 です。最大長は 100 です。

パターン：`[\w][\w@.-]{2,99}`

エラー

すべてのアクションに共通のエラーについては、「共通エラー」を参照してください。

InternalServiceError

この例外は、 AWS Transfer Family サービスでエラーが発生した場合にスローされます。

HTTP ステータスコード：500

InvalidRequestException

この例外は、クライアントが不正な形式のリクエストを送信した場合にスローされます。

HTTP ステータスコード：400

ResourceNotFoundException

この例外は、 AWS Transfer Family サービスによってリソースが見つからない場合にスローされます。

HTTP ステータスコード：400

ServiceUnavailableException

AWS Transfer Family サービスが利用できないため、リクエストは失敗しました。

HTTP ステータスコード：500

ThrottlingException

リクエストのスロットリングにより、リクエストが拒否されました。

HTTP ステータスコード：400

例

例

次の例では、Transfer Family ユーザーを更新します。

リクエスト例

```
{
   "HomeDirectory": "/bucket2/documentation",
   "HomeDirectoryMappings": [
      {
         "Entry": "/directory1",
         "Target": "/bucket_name/home/mydirectory"
      }
   ],
   "HomeDirectoryType:" "PATH",
   "Role": "AssumeRole",
   "ServerId": "s-01234567890abcdef",
   "UserName": "my_user"
}
```

例

これは、この API コールに対するレスポンスのサンプルです。

レスポンス例

```
{
   "ServerId": "s-01234567890abcdef",
   "UserName": "my_user"
}
```

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

データ型

以下のデータ型 (タイプ) がサポートされています。

- As2ConnectorConfig
- CopyStepDetails
- CustomStepDetails
- DecryptStepDetails
- DeleteStepDetails
- DescribedAccess

- DescribedAgreement
- DescribedCertificate
- DescribedConnector
- DescribedExecution
- DescribedHostKey
- DescribedProfile
- DescribedSecurityPolicy
- DescribedServer
- DescribedUser
- DescribedWorkflow
- EfsFileLocation
- EndpointDetails
- ExecutionError
- ExecutionResults
- ExecutionStepResult
- FileLocation
- HomeDirectoryMapEntry
- IdentityProviderDetails
- InputFileLocation
- ListedAccess
- ListedAgreement
- ListedCertificate
- ListedConnector
- ListedExecution
- ListedHostKey
- ListedProfile
- ListedServer
- ListedUser
- ListedWorkflow
- LoggingConfiguration

- PosixProfile
- ProtocolDetails
- S3FileLocation
- S3InputFileLocation
- S3StorageOptions
- S3Tag
- ServiceMetadata
- SftpConnectorConfig
- SshPublicKey
- Tag
- TagStepDetails
- UserDetails
- WorkflowDetail
- WorkflowDetails
- WorkflowStep

As2ConnectorConfig

AS2 コネクタオブジェクトの詳細が含まれます。コネクタオブジェクトは AS2 アウトバウンドプロセスに使用され、 AWS Transfer Family 顧客を取引相手に接続します。

内容

BasicAuthSecretId

AS2 コネクタ API に基本認証サポートを提供します。基本認証を使用するには、シークレットの名前または AWS Secrets Managerでのシークレットの Amazon リソースネーム (ARN) を指定する必要があります。

このパラメータのデフォルト値はnullです。これは、コネクタに対して基本認証が有効になっていないことを示します。

コネクタが基本認証を使用する場合、シークレットは次の形式である必要があります。

```
{ "Username": "user-name", "Password": "user-password" }
```

user-nameとuser-passwordを、認証される実際のユーザーの認証情報に置き換えてください。

次の点に注意してください。

- これらの認証情報は、この API に直接渡すのではなく、Secrets Manager に保存しています。
- API、SDKs、または を使用してコネクタ CloudFormation を設定する場合は、基本認証を有効にする前にシークレットを作成する必要があります。ただし、 AWS マネジメントコンソールを使用している場合は、システムにシークレットを作成させることができます。

以前にコネクタの基本認証を有効にしたことがある場合は、UpdateConnector API コールを使用して無効にできます。例えば、CLI を使用している場合は、次のコマンドを実行して基本認証を削除できます。

```
update-connector --connector-id my-connector-id --as2-config
'BasicAuthSecretId=""'
```

型: 文字列

長さの制限: 最小長は 0 です。最大長は 2048 です。

必須: いいえ

Compression

AS2 ファイルが圧縮されているかどうかを指定します。

型: 文字列

有効な値：ZLIB | DISABLED

必須：いいえ

EncryptionAlgorithm

ファイルの暗号化に使用されるアルゴリズム。

次の点に注意してください。

- アルゴリズムは弱い暗号化DES_EDE3_CBCアルゴリズムであるため、それを必要とするレガシークライアントをサポートする必要がある場合を除き、アルゴリズムを使用しないでください。
- コネクタの URL が HTTPS NONE を使用するかどうかのみ指定できます。HTTPS を使用すると、トラフィックがクリアテキストで送信されることはありません。

型: 文字列

有効な値：AES128_CBC | AES192_CBC | AES256_CBC | DES_EDE3_CBC | NONE

必須：いいえ

LocalProfileId

AS2 ローカルプロファイルの一意の識別子です。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: p-([0-9a-f]{17})

必須: いいえ

MdnResponse

転送に対するパートナーの応答が同期か非同期かを判断するために、(AWS Transfer Family サーバーからパートナー AS2 サーバーへの) アウトバウンドリクエストに使用されます。以下のいずれかの値を指定する：

- SYNC: システムは、ファイルが正常に転送された (または失敗した) ことを確認する MDN の同期応答を期待しています。
- NONE: MDN レスポンスがが必要ないことを指定します。

型: 文字列

有効な値：SYNC | NONE

必須：いいえ

MdnSigningAlgorithm

MDN レスポンスの署名アルゴリズム。

> **Note**
>
> DEFAULT に設定された (またはまったく設定されていない) 場合は、SigningAlgorithmの値が使用されます。

型: 文字列

有効な値：SHA256 | SHA384 | SHA512 | SHA1 | NONE | DEFAULT

必須：いいえ

MessageSubject

コネクタで送信される AS2 メッセージの Subject HTTP ヘッダー属性として使用されます。

型: 文字列

長さの制限: 最小長は 1 です。最大長は 1,024 です。

パターン: [\p{Print}\p{Blank}]+

必須: いいえ

PartnerProfileId

コネクタのパートナープロファイルの一意の識別子です。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `p-([0-9a-f]{17})`

必須: いいえ

SigningAlgorithm

コネクタで送信される AS2 メッセージに署名するために使用されるアルゴリズム。

型: 文字列

有効な値：`SHA256 | SHA384 | SHA512 | SHA1 | NONE`

必須：いいえ

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

CopyStepDetails

各ステップタイプに独自の `StepDetails` 構造があります。

内容

DestinationFileLocation

コピーしようとするファイルの場所を指定します。このフィールドで`${Transfer:UserName}`または`${Transfer:UploadDate}`を使用して、宛先プレフィックスをユーザー名またはアップロード日でパラメータ化します。

- 値を`DestinationFileLocation`から`${Transfer:UserName}`に設定すると、アップロードされたファイルを、ファイルをアップロードした Transfer Family ユーザーの名前のプレフィックスが付いた Amazon S3 バケットにコピーされます。
- 値を`DestinationFileLocation`から`${Transfer:UploadDate}`に設定すると、アップロードされたファイルを、アップロード日のプレフィックスが付いた Amazon S3 バケットにコピーします。

> **Note**
>
> システムは、ファイルが UTC `UploadDate` でアップロードされた日付に基づいて YYYY-MM-DD の日付形式に解決します。

タイプ：InputFileLocation オブジェクト

必須: いいえ

Name

識別子として使用されるステップの名前。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 30 です。

Pattern: `[\w-]*`

必須: いいえ

OverwriteExisting

同じ名前の既存のファイルを上書きするかを示すフラグです。デフォルトは `FALSE` です。

ワークフローが既存のファイルと同じ名前のファイルを処理している場合、動作は次のようになります。

- OverwriteExistingがTRUEの場合、既存のファイルは処理中のファイルに置き換えられます。
- OverwriteExistingがFALSEの場合、何も起こらず、ワークフロー処理は停止します。

型: 文字列

有効な値：TRUE | FALSE

必須：いいえ

SourceFileLocation

ワークフローステップへの入力として使用するファイル (前のステップからの出力、またはワークフロー用に最初にアップロードされたファイル) を指定します。

- 前のファイルを入力として使用するには、${previous.file} と入力します。この場合、このワークフローステップは前のワークフローステップの出力ファイルを入力として使用します。これは、デフォルト値です。
- 最初にアップロードされたファイルの場所をこのステップの入力として使用するには、${original.file} と入力します。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 256 です。

パターン：\$\{(\w+.)+\w+\}

必須: いいえ

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

CustomStepDetails

各ステップタイプに独自の StepDetails 構造があります。

内容

Name

識別子として使用されるステップの名前。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 30 です。

Pattern: [\w-]*

必須: いいえ

SourceFileLocation

ワークフローステップへの入力として使用するファイル (前のステップからの出力、またはワークフロー用に最初にアップロードされたファイル) を指定します。

- 前のファイルを入力として使用するには、${previous.file} と入力します。この場合、このワークフローステップは前のワークフローステップの出力ファイルを入力として使用します。これは、デフォルト値です。
- 最初にアップロードされたファイルの場所をこのステップの入力として使用するには、${original.file} と入力します。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 256 です。

パターン：\$\{(\w+.)+\w+\}

必須: いいえ

Target

呼び出される Lambda 関数の ARN。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 170 です。

パターン: `arn:[a-z-]+:lambda:.*`

必須: いいえ

TimeoutSeconds

ステップのタイムアウト値 (秒単位)。

型: 整数

有効範囲: 最小値 は 1 です。最大値は 1,800 です。

必須 : いいえ

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

DecryptStepDetails

各ステップタイプに独自の StepDetails 構造があります。

内容

DestinationFileLocation

復号化されるファイルの場所を指定します。このフィールドで${Transfer:UserName}または${Transfer:UploadDate}を使用して、宛先プレフィックスをユーザー名またはアップロード日でパラメータ化します。

- ファイルをアップロードした Transfer Family ユーザーの名前のプレフィックスが付いた Amazon S3 バケットにアップロードされたファイルを復号化するには、の値をDestinationFileLocationto${Transfer:UserName}に設定します。
- 値をDestinationFileLocationから${Transfer:UploadDate}に設定すると、アップロードされたファイルを、アップロード日のプレフィックスが付いた Amazon S3 バケットに復号化します。

> **Note**
>
> システムは、ファイルが UTC UploadDate でアップロードされた日付に基づいて YYYY-MM-DD の日付形式に解決します。

型: InputFileLocation オブジェクト

必須: はい

Type

使用される暗号化のタイプ。現在のところ、この値は PGP である必要があります。

型: 文字列

有効な値：PGP

必須: はい

Name

識別子として使用されるステップの名前。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 30 です。

Pattern: [\w-]*

必須: いいえ

OverwriteExisting

同じ名前の既存のファイルを上書きするかを示すフラグです。デフォルトは FALSE です。

ワークフローが既存のファイルと同じ名前のファイルを処理している場合、動作は次のようになります。

- OverwriteExistingがTRUEの場合、既存のファイルは処理中のファイルに置き換えられます。
- OverwriteExistingがFALSEの場合、何も起こらず、ワークフロー処理は停止します。

型: 文字列

有効な値：TRUE | FALSE

必須：いいえ

SourceFileLocation

ワークフローステップへの入力として使用するファイル (前のステップからの出力、またはワークフロー用に最初にアップロードされたファイル) を指定します。

- 前のファイルを入力として使用するには、${previous.file} と入力します。この場合、このワークフローステップは前のワークフローステップの出力ファイルを入力として使用します。これは、デフォルト値です。
- 最初にアップロードされたファイルの場所をこのステップの入力として使用するには、${original.file} と入力します。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 256 です。

パターン：\$\{(\w+.)+\w+\}

必須: いいえ

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

DeleteStepDetails

削除ステップを識別するために使用されるステップの名前。

内容

Name

識別子として使用されるステップの名前。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 30 です。

Pattern: `[\w-]*`

必須: いいえ

SourceFileLocation

ワークフローステップへの入力として使用するファイル (前のステップからの出力、またはワークフロー用に最初にアップロードされたファイル) を指定します。

- 前のファイルを入力として使用するには、`${previous.file}` と入力します。この場合、このワークフローステップは前のワークフローステップの出力ファイルを入力として使用します。これは、デフォルト値です。
- 最初にアップロードされたファイルの場所をこのステップの入力として使用するには、`${original.file}` と入力します。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 256 です。

パターン : `\$\{(\w+.)+\w+\}`

必須: いいえ

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++

- AWS SDK for Java V2
- AWS SDK for Ruby V3

DescribedAccess

指定されたアクセスのプロパティを記述します。

内容

ExternalId

ディレクトリ内の特定のグループを識別するために必要な一意の識別子。関連付けるグループのユーザーは、 を使用して、有効なプロトコル経由で Amazon S3 または Amazon EFS リソースにアクセスできます AWS Transfer Family。グループ名がわかっている場合は、Windows を使用して次のコマンドを実行して SID 値を表示できます PowerShell。

```
Get-ADGroup -Filter {samAccountName -like "YourGroupName*"} -Properties
* | Select SamAccountName,ObjectSid
```

そのコマンドで、 を Active Directory グループの名前YourGroupNameに置き換えます。

このパラメータの検証に使用される正規表現は、スペースを含まない大文字と小文字の英数字からなる文字列です。下線文字 (_) や =,.@:/- の文字を含めることもできます。

型: 文字列

長さの制限：最小長は 1 です。最大長は 256 です。

パターン：`S-1-[\d-]+`

必須: いいえ

HomeDirectory

ユーザーがクライアントを使用してサーバーにログインするときの、ユーザーのランディングディレクトリ (フォルダ)。

`HomeDirectory` の例は `/bucket_name/home/mydirectory` です。

> **Note**
>
> `HomeDirectory` パラメータは、`HomeDirectoryType` が `PATH` に設定されている場合のみ使用されます。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 1,024 です。

パターン: (|/.*)

必須: いいえ

HomeDirectoryMappings

ユーザーに表示する Amazon S3 または Amazon EFS のパスとキー、およびそれらをどのように表示するかを指定する論理ディレクトリマッピング。Entry と Target のペアを指定する必要があり、Entry はパスの表示方法を示し、Target は実際の Amazon S3 または Amazon EFS のパスです。ターゲットを指定しただけの場合は、そのまま表示されます。また、 AWS Identity and Access Management (IAM) ロールが のパスへのアクセスを許可していることを確認する必要がありますTarget。この値は、HomeDirectoryType がLOGICALに設定されている場合にのみ設定できます。

ほとんどの場合、セッションポリシーの代わりにこの値を使用することで、関連付けられたアクセスを指定されたホームディレクトリ (「chroot」) にロックダウンできます。そのためには、Entry をスラッシュ (/) に設定し、Target を HomeDirectory パラメータ値に設定します。

型: HomeDirectoryMapEntry オブジェクトの配列

配列メンバー : 最小数は 1 項目です。最大数は 50,000 項目です。

必須: いいえ

HomeDirectoryType

ユーザーがサーバーにログインするときにホームディレクトリにするランディングディレクトリ (フォルダ) のタイプ。これを PATH に設定した場合、ユーザーには、ファイル転送プロトコルクライアントに、絶対的な Amazon S3 バケットまたは現状の Amazon EFS パスが表示されます。これを LOGICAL に設定した場合、Amazon S3 または Amazon EFS パスをユーザーに表示する方法に関して、HomeDirectoryMappings でマッピングを指定する必要があります。

> **Note**
>
> HomeDirectoryType が LOGICAL の場合は、HomeDirectoryMappings パラメータを使用してマッピングを指定する必要があります。一方、HomeDirectoryType が PATH の場合は、HomeDirectory パラメータを使用して絶対パスを指定します。テンプ

レートに HomeDirectory と HomeDirectoryMappings の両方を含めることはできません。

型: 文字列

有効な値：PATH | LOGICAL

必須：いいえ

Policy

複数のユーザーで同じ AWS Identity and Access Management (IAM) ロールを使用できるようにするためのユーザーのセッションポリシー。このポリシーは、ユーザーアクセスのスコープを Amazon S3 バケットの一部に絞り込みます。このポリシー内に使用できる変数には、${Transfer:UserName}、${Transfer:HomeDirectory}、${Transfer:HomeBucket} があります。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 2048 です。

必須: いいえ

PosixProfile

ユーザーの Amazon EFS ファイルシステムへのアクセスを制御する、ユーザー ID (Uid Gid)、グループ ID (SecondaryGids)、およびセカンダリグループ ID () を含む完全な POSIX ID。ファイルシステム内のファイルとディレクトリに設定される POSIX アクセス許可によって、Amazon EFS ファイルシステムとの間でファイルを転送するときにユーザーが得るアクセスのレベルが決まります。

型: PosixProfile オブジェクト

必須: いいえ

Role

Amazon S3 バケットまたは Amazon EFS ファイルシステムへのユーザーのアクセスを制御する AWS Identity and Access Management (IAM) ロールの Amazon リソースネーム (ARN)。Amazon S3 このロールにアタッチされたポリシーにより、ファイルを Amazon S3 バケットまたは Amazon EFS ファイルシステム間で転送する際の、ユーザーに付与するアクセスレベルが決定

されます。IAM ロールには、ユーザーの転送リクエストを処理する際に、サーバーによるリソースへのアクセスを許可する信頼関係も含まれる必要があります。

型: 文字列

長さの制限: 最小長は 20 です。最大長は 2,048 です。

パターン: `arn:.*role/\S+`

必須: いいえ

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

DescribedAgreement

契約のプロパティを説明します。

内容

Arn

契約の一意のAmazon リソースネーム (ARN)。

型: 文字列

長さの制限: 最小長は 20 です。最大長は 1600 です。

Pattern: `arn:\S+`

必須：はい

AccessRole

コネクターは、AS2 または SFTP プロトコルを使用してファイルを送信するために使用されます。アクセスロールには、使用する AWS Identity and Access Management ロールの Amazon リソースネーム (ARN) を指定します。

AS2 コネクタ用

AS2 では、`StartFileTransfer` を呼び出してリクエストパラメータ にファイルパスを指定することでファイルを送信できます。`SendFilePaths`ファイルの親ディレクトリ (例えば `--send-file-paths /bucket/dir/file.txt` の場合、親ディレクトリは `/bucket/dir/`) を使用して、処理済みの AS2 メッセージファイルを一時的に保存し、パートナーから受け取った MDN を保存して、送信の関連メタデータを含む最終的な JSON ファイルを記述します。そのため、`AccessRole` は、`StartFileTransfer` リクエストで使用されるファイルの場所の親ディレクトリに対する読み取り/書き込みアクセスを提供する必要があります。さらに、`StartFileTransfer` で送信するファイルの親ディレクトリに対する読み取り/書き込みアクセスを提供する必要があります。

AS2 コネクタに Basic 認証を使用している場合、アクセスロールにはシークレットの `secretsmanager:GetSecretValue` 権限が必要です。Secrets Manager の マネージドキーではなく、カスタマー AWS マネージドキーを使用してシークレットが暗号化されている場合、ロールにはそのキーに対する `kms:Decrypt` アクセス許可も必要です。

SFTP コネクタ用

アクセスロールが、StartFileTransfer リクエストで使用されるファイルロケーションの親ディレクトリへの読み取りおよび書き込みアクセスを提供していることを確認します。さらに、ロールが にアクセスsecretsmanager:GetSecretValue許可を付与していることを確認します AWS Secrets Manager。

型: 文字列

長さの制限: 最小長は 20 です。最大長は 2,048 です。

パターン: arn:.*role/\S+

必須: いいえ

AgreementId

契約の一意の識別子。この識別子は、契約を作成するときに返されます。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: a-([0-9a-f]{17})

必須: いいえ

BaseDirectory

AS2 プロトコルを使用して転送されるファイルのランディングディレクトリ (フォルダ) です。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 1,024 です。

パターン: (|/.*)

必須: いいえ

Description

契約書を識別するために使用される名前または簡単な説明です。

型: 文字列

長さの制限: 最小長は 1 です。最大長は 200 です。

Pattern: [\p{Graph}]+

必須: いいえ

LocalProfileId

AS2 ローカルプロファイルの一意の識別子です。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `p-([0-9a-f]{17})`

必須: いいえ

PartnerProfileId

契約で使用されるパートナープロファイルの一意の識別子です。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `p-([0-9a-f]{17})`

必須: いいえ

ServerId

サーバーインスタンスにシステムで割り当てられた一意の識別子。この識別子は、契約が使用する特定のサーバーを示します。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `s-([0-9a-f]{17})`

必須: いいえ

Status

契約の現在のステータス (`ACTIVE` または `INACTIVE`) です。

型: 文字列

有効な値：`ACTIVE | INACTIVE`

必須：いいえ

Tags

契約のグループ化および検索に使用できるキーと値のペアです。

型: Tag オブジェクトの配列

配列メンバー：最小数は 1 項目です。最大数は 50 項目です。

必須：いいえ

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

DescribedCertificate

証明書のプロパティについて記述します。

内容

Arn

証明書の一意の Amazon リソースネーム (ARN) です。

型: 文字列

長さの制限: 最小長は 20 です。最大長は 1600 です。

Pattern: `arn:\S+`

必須：はい

ActiveDate

証明書がいつアクティブになるかを指定するオプションの日付です。

型: タイムスタンプ

必須: いいえ

Certificate

証明書のファイル名です。

型: 文字列

長さの制限: 最小長は 1 です。最大長は 16384 です。

Pattern: `[\u0009\u000A\u000D\u0020-\u00FF]*`

必須: いいえ

CertificateChain

証明書のチェーンを構成する証明書のリストです。

型: 文字列

長さの制限：最小長は 1 です。最大長は 2097152 です。

Pattern: [\u0009\u000A\u000D\u0020-\u00FF]*

必須: いいえ

CertificateId

インポートされた証明書の識別子の配列です。この識別子は、プロファイルやパートナープロファイルの操作に使用します。

型: 文字列

長さの制限: 22 の固定長

Pattern: cert-([0-9a-f]{17})

必須: いいえ

Description

証明書を識別するために使用される名前または説明です。

型: 文字列

長さの制限: 最小長は 1 です。最大長は 200 です。

Pattern: [\p{Graph}]+

必須: いいえ

InactiveDate

証明書がいつ非アクティブになるかを指定するオプションの日付です。

型: タイムスタンプ

必須: いいえ

NotAfterDate

証明書が有効である最終日です。

型: タイムスタンプ

必須: いいえ

NotBeforeDate

証明書が有効となる初日です。

型: タイムスタンプ

必須: いいえ

Serial

証明書のシリアル番号です。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 48 です。

Pattern: `[\p{XDigit}{2}:?]*`

必須: いいえ

Status

証明書は、ACTIVE、PENDING_ROTATION、INACTIVE のいずれかになります。PENDING_ROTATION は、現在の証明書の有効期限が切れると、この証明書が置き換えられることを意味します。

型: 文字列

有効な値：`ACTIVE | PENDING_ROTATION | INACTIVE`

必須：いいえ

Tags

証明書のグループ化および検索に使用できるキーと値のペアです。

型: Tag オブジェクトの配列

配列メンバー：最小数は 1 項目です。最大数は 50 項目です。

必須: いいえ

Type

証明書にプライベートキーが指定されている場合、そのタイプは CERTIFICATE_WITH_PRIVATE_KEY です。プライベートキーがない場合、タイプは CERTIFICATE です。

型: 文字列

有効な値：CERTIFICATE | CERTIFICATE_WITH_PRIVATE_KEY

必須：いいえ

Usage

この証明書の使用方法を指定します。これは、次の方法で使用できます。

- SIGNING: AS2 メッセージの署名用
- ENCRYPTION: AS2 メッセージの暗号化用
- TLS: HTTPS 経由で送信される AS2 通信を保護する場合

型: 文字列

有効な値：SIGNING | ENCRYPTION

必須：いいえ

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

DescribedConnector

ConnectorId で識別されるコネクタのパラメータを記述します。

内容

Arn

コネクタの一意の Amazon リソース名（ ARN ）。

型: 文字列

長さの制限: 最小長は 20 です。最大長は 1600 です。

Pattern: arn:\S+

必須：はい

AccessRole

コネクターは、AS2 または SFTP プロトコルを使用してファイルを送信するために使用されます。アクセスロールには、使用する AWS Identity and Access Management ロールの Amazon リソースネーム (ARN) を指定します。

AS2 コネクタ用

AS2 では、StartFileTransfer を呼び出してリクエストパラメータ にファイルパスを指定することでファイルを送信できます。SendFilePathsファイルの親ディレクトリ (例えば --send-file-paths /bucket/dir/file.txt の場合、親ディレクトリは /bucket/dir/) を使用して、処理済みの AS2 メッセージファイルを一時的に保存し、パートナーから受け取った MDN を保存して、送信の関連メタデータを含む最終的な JSON ファイルを記述します。そのため、AccessRole は、StartFileTransfer リクエストで使用されるファイルの場所の親ディレクトリに対する読み取り/書き込みアクセスを提供する必要があります。さらに、StartFileTransfer で送信するファイルの親ディレクトリに対する読み取り/書き込みアクセスを提供する必要があります。

AS2 コネクタに Basic 認証を使用している場合、アクセスロールにはシークレットの secretsmanager:GetSecretValue 権限が必要です。シークレットが Secrets Manager のマネージドキーではなく、カスタマー AWS マネージドキーを使用して暗号化されている場合、ロールにはそのキーに対する アクセスkms:Decrypt許可も必要です。

SFTP コネクタ用

アクセスロールが、StartFileTransfer リクエストで使用されるファイルロケーションの親ディレクトリへの読み取りおよび書き込みアクセスを提供していることを確認します。さらに、ロールが にアクセスsecretsmanager:GetSecretValue許可を付与していることを確認します AWS Secrets Manager。

型: 文字列

長さの制限: 最小長は 20 です。最大長は 2,048 です。

パターン: arn:.*role/\S+

必須: いいえ

As2Config

AS2 コネクタオブジェクトのパラメータを含む構造体。

タイプ : As2ConnectorConfig オブジェクト

必須: いいえ

ConnectorId

コネクタの一意の識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: c-([0-9a-f]{17})

必須: いいえ

LoggingRole

コネクタが Amazon S3 イベントの CloudWatch ログ記録をオンにできるようにする (IAM) ロールの Amazon リソースネーム AWS Identity and Access Management (ARN)。 Amazon S3 設定すると、 CloudWatch ログにコネクタアクティビティを表示できます。

型: 文字列

長さの制限: 最小長は 20 です。最大長は 2,048 です。

パターン: arn:.*role/\S+

必須: いいえ

SecurityPolicyName

指定されたコネクタのセキュリティポリシーのテキスト名。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 100 です。

Pattern: `TransferSFTPConnectorSecurityPolicy-[A-Za-z0-9-]+`

必須: いいえ

ServiceManagedEgressIpAddresses

このコネクタの出力 IP アドレスのリスト。これらの IP アドレスは、コネクタの作成時に自動的に割り当てられます。

タイプ：文字列の配列

Pattern: `\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}`

必須: いいえ

SftpConfig

SFTP コネクタオブジェクトのパラメータを含む構造体。

タイプ：SftpConnectorConfig オブジェクト

必須: いいえ

Tags

コネクタのグループ化および検索に使用できるキーと値のペアです。

型: Tag オブジェクトの配列

配列メンバー：最小数は 1 項目です。最大数は 50 項目です。

必須: いいえ

Url

パートナーの AS2 または SFTP エンドポイントの URL。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 255 です。

必須：いいえ

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

DescribedExecution

実行オブジェクトの詳細。

内容

ExecutionId

ワークフローの一意の識別子。

型: 文字列

長さの制限: 最大長は 36 です。

Pattern: `[0-9a-fA-F]{8}\-[0-9a-fA-F]{4}\-[0-9a-fA-F]{4}\-[0-9a-fA-F]{4}\-[0-9a-fA-F]{12}`

必須: いいえ

ExecutionRole

実行に関連付けられた IAM ロール。

型: 文字列

長さの制限: 最小長は 20 です。最大長は 2,048 です。

パターン: `arn:.*role/\S+`

必須: いいえ

InitialFileLocation

Amazon S3 または EFS ファイルの場所を記述する構造。これは実行開始時のファイルの場所であり、ファイルをコピーしようとする場合にはファイルの (コピー先ではなく) 初期の場所です。

型: FileLocation オブジェクト

必須: いいえ

LoggingConfiguration

実行に関連付けられた IAM ログ記録ロール。

型: LoggingConfiguration オブジェクト

必須: いいえ

PosixProfile

ユーザーの Amazon EFS ファイルシステムへのアクセスを制御する、ユーザー ID (Uid Gid)、グループ ID (SecondaryGids)、およびセカンダリグループ ID () を含む完全な POSIX ID。ファイルシステム内のファイルとディレクトリに設定される POSIX アクセス許可によって、Amazon EFS ファイルシステムとの間でファイルを転送するときにユーザーが得るアクセスのレベルが決まります。

型: PosixProfile オブジェクト

必須: いいえ

Results

実行結果を記述する構造。具体的には、ステップの一覧と各ステップの詳細、エラータイプとメッセージ (存在する場合) 、および OnExceptionSteps 構造です。

型: ExecutionResults オブジェクト

必須: いいえ

ServiceMetadata

ワークフローに関連付けられたセッション詳細のコンテナオブジェクト。

タイプ：ServiceMetadata オブジェクト

必須: いいえ

Status

ステータスは実行のいずれかです。進行中、完了、例外発生、または例外処理中の可能性があります。

型: 文字列

有効な値：IN_PROGRESS | COMPLETED | EXCEPTION | HANDLING_EXCEPTION

必須：いいえ

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

DescribedHostKey

サーバーホストキーの詳細。

内容

Arn

ホストキーの一意の Amazon リソースネーム (ARN)。

型: 文字列

長さの制限: 最小長は 20 です。最大長は 1600 です。

Pattern: `arn:\S+`

必須：はい

DateImported

ホストキーがサーバーに追加された日付。

型: タイムスタンプ

必須: いいえ

Description

このホストキーのテキスト説明。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 200 です。

Pattern: `[\p{Print}]*`

必須: いいえ

HostKeyFingerprint

パブリックキーのフィンガープリントは、長い方のパブリックキーを識別するために使用される短いバイト列です。

タイプ: 文字列

必須: いいえ

HostKeyId

ホストキーの一意の識別子。

型: 文字列

長さの制限: 固定長は 25 です。

Pattern: `hostkey-[0-9a-f]{17}`

必須: いいえ

Tags

ホストキーのグループ化および検索に使用できるキーバリューペア。

型: Tag オブジェクトの配列

配列メンバー：最小数は 1 項目です。最大数は 50 項目です。

必須: いいえ

Type

ホストキーに使用される暗号化アルゴリズムの種類。Type パラメータは、次のいずれかの値を使用して指定します。

- `ssh-rsa`
- `ssh-ed25519`
- `ecdsa-sha2-nistp256`
- `ecdsa-sha2-nistp384`
- `ecdsa-sha2-nistp521`

タイプ: 文字列

必須: いいえ

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

DescribedProfile

ローカルまたはパートナーの AS2 プロファイルの詳細。

内容

Arn

プロファイルの一意の Amazon リソースネーム (ARN)。

型: 文字列

長さの制限: 最小長は 20 です。最大長は 1600 です。

Pattern: `arn:\S+`

必須：はい

As2Id

`As2Id` は、RFC 4130 で定義されている AS2-name です。インバウンド転送の場合、これはパートナーから送信される AS2 メッセージの `AS2-From` ヘッダーです。アウトバウンドコネクタの場合、これは `StartFileTransfer` API オペレーションを使用してパートナーに送信される AS2 メッセージの `AS2-To` ヘッダーです。この ID にスペースを含めることはできません。

型: 文字列

長さの制限: 最小長は 1 です。最大長は 128 です。

Pattern: `[\p{Print}\s]*`

必須: いいえ

CertificateIds

インポートされた証明書の識別子の配列です。この識別子は、プロファイルやパートナープロファイルの操作に使用します。

タイプ：文字列の配列

長さの制限: 22 の固定長

Pattern: `cert-([0-9a-f]{17})`

必須: いいえ

ProfileId

ローカルまたはパートナー AS2 プロファイルの一意の識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `p-([0-9a-f]{17})`

必須: いいえ

ProfileType

LOCAL タイププロファイルのみを一覧表示するか、PARTNER タイププロファイルのみを一覧表示するかを示します。リクエストで指定されていない場合、コマンドはすべてのタイプのプロファイルを一覧表示します。

型: 文字列

有効な値：`LOCAL | PARTNER`

必須：いいえ

Tags

プロファイルのグループ化および検索に使用できるキーと値のペアです。

型: Tag オブジェクトの配列

配列メンバー：最小数は 1 項目です。最大数は 50 項目です。

必須：いいえ

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2

- AWS SDK for Ruby V3

DescribedSecurityPolicy

指定したセキュリティポリシーのプロパティについて説明します。セキュリティポリシーの詳細については、「サーバーのセキュリティポリシーの使用」または「SFTP コネクタのセキュリティポリシーの使用」を参照してください。

内容

SecurityPolicyName

指定されたセキュリティポリシーのテキスト名。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 100 です。

パターン: `Transfer[A-Za-z0-9]*SecurityPolicy-[A-Za-z0-9-]+`

必須：はい

Fips

このポリシーで連邦情報処理規格 (FIPS) を有効にするかどうかを指定します。このパラメータは、サーバーとコネクタの両方のセキュリティポリシーに適用されます。

型: ブール値

必須: いいえ

Protocols

セキュリティポリシーが適用されるファイル転送プロトコルを一覧表示します。

タイプ：文字列の配列

配列メンバー: 最小数は 1 項目です。最大数は 5 項目です。

有効な値：`SFTP | FTPS`

必須：いいえ

SshCiphers

サーバーまたはコネクタにアタッチされているセキュリティポリシーで有効な Secure Shell (SSH) 暗号暗号化アルゴリズムを一覧表示します。このパラメータは、サーバーとコネクタの両方のセキュリティポリシーに適用されます。

タイプ：文字列の配列

長さの制限: 最小長は 0 です。最大長は 50 です。

必須: いいえ

SshHostKeyAlgorithms

セキュリティポリシーのホストキーアルゴリズムを一覧表示します。

> **Note**
> このパラメータは、コネクタのセキュリティポリシーにのみ適用されます。

タイプ：文字列の配列

長さの制限: 最小長は 0 です。最大長は 50 です。

必須: いいえ

SshKexs

サーバーまたはコネクタにアタッチされているセキュリティポリシー内の有効な SSH キー交換 (KEX) 暗号化アルゴリズムを一覧表示します。このパラメータは、サーバーとコネクタの両方のセキュリティポリシーに適用されます。

タイプ：文字列の配列

長さの制限: 最小長は 0 です。最大長は 50 です。

必須: いいえ

SshMacs

サーバーまたはコネクタにアタッチされているセキュリティポリシー内の有効な SSH メッセージ認証コード (MAC) 暗号化アルゴリズムを一覧表示します。このパラメータは、サーバーとコネクタの両方のセキュリティポリシーに適用されます。

タイプ：文字列の配列

長さの制限: 最小長は 0 です。最大長は 50 です。

必須: いいえ

TlsCiphers

サーバーにアタッチされているセキュリティポリシーで有効な Transport Layer Security (TLS) 暗号暗号化アルゴリズムを一覧表示します。

> **Note**
> このパラメータは、 サーバーのセキュリティポリシーにのみ適用されます。

タイプ：文字列の配列

長さの制限: 最小長は 0 です。最大長は 50 です。

必須: いいえ

Type

セキュリティポリシーが適用されるリソースタイプ。サーバーまたはコネクタ。

型: 文字列

有効な値：`SERVER | CONNECTOR`

必須：いいえ

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

DescribedServer

指定されたファイル転送プロトコル対応サーバーのプロパティについて記述します。

内容

Arn

サーバーの一意の Amazon リソースネーム (ARN) を指定します。

型: 文字列

長さの制限: 最小長は 20 です。最大長は 1600 です。

Pattern: `arn:\S+`

必須：はい

As2ServiceManagedEgressIpAddresses

このサーバーの出力 IP アドレスのリスト。これらの IP アドレスは、AS2 プロトコルを使用するサーバーにのみ関係します。非同期 MDNsの送信に使用されます。

これらの IP アドレスは、AS2 サーバーを作成するときに自動的に割り当てられます。さらに、既存のサーバーを更新して AS2 プロトコルを追加すると、静的 IP アドレスも割り当てられます。

タイプ：文字列の配列

Pattern: `\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}`

必須: いいえ

Certificate

AWS Certificate Manager (ACM) 証明書の ARN を指定します。`Protocols` が `FTPS` に設定されている場合は必須です。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 1600 です。

必須: いいえ

Domain

ファイル転送に使用されるストレージシステムのドメインを指定します。使用できるドメインは、Amazon Simple Storage Service (Amazon S3) と Amazon Elastic File System (Amazon EFS) です。デフォルト値は S3 です。

型: 文字列

有効な値：`S3 | EFS`

必須：いいえ

EndpointDetails

サーバーに設定された Virtual Private Cloud (VPC) エンドポイント設定。VPC 内でエンドポイントをホストする場合、VPC 内のリソースにのみアクセスできるようにすることも、Elastic IP アドレスをアタッチしてインターネット経由でクライアントにアクセスできるようにすることもできます。VPC のデフォルトのセキュリティグループは、エンドポイントに自動的に割り当てられます。

型: EndpointDetails オブジェクト

必須: いいえ

EndpointType

サーバーの接続先になるエンドポイントのタイプ。サーバーが VPC エンドポイントに接続している場合、パブリックインターネット経由で はサーバーにはアクセスできません。

型: 文字列

有効な値：`PUBLIC | VPC | VPC_ENDPOINT`

必須：いいえ

HostKeyFingerprint

サーバーのホストキーの Base64 エンコード SHA256 フィンガープリントを指定します。この値は、`ssh-keygen -l -f my-new-server-key` コマンドの出力と等価です。

タイプ: 文字列

必須: いいえ

IdentityProviderDetails

カスタマー提供の認証 API を呼び出すための情報を指定します。サーバーの IdentityProviderType が AWS_DIRECTORY_SERVICE または SERVICE_MANAGED の場合、このフィールドには自動機な値が入力されます。

型: IdentityProviderDetails オブジェクト

必須: いいえ

IdentityProviderType

サーバーの認証のモード。デフォルト値は です。これによりSERVICE_MANAGED、 AWS Transfer Family サービス内でユーザー認証情報を保存してアクセスできます。

AWS_DIRECTORY_SERVICE を使用して、オンプレミス環境の AWS Directory Service for Microsoft Active Directory または Microsoft Active Directory、または AD Connector AWS を使用する の Active Directory グループへのアクセスを提供します。このオプションでは、IdentityProviderDetails パラメータを使用してディレクトリ ID を指定する必要もあります。

この API_GATEWAY 値を使用して、選択した ID プロバイダーと統合します。API_GATEWAY 設定では、IdentityProviderDetails パラメータを使用して認証を呼び出すには Amazon API Gateway エンドポイントの URL を指定する必要があります。

AWS_LAMBDA 値を使用して、 AWS Lambda 関数を ID プロバイダーとして直接使用します。この値を選択した場合、Function パラメータで、IdentityProviderDetails データ型の Lambda 関数の ARN を指定する必要があります。

型: 文字列

有効な値：SERVICE_MANAGED | API_GATEWAY | AWS_DIRECTORY_SERVICE | AWS_LAMBDA

必須：いいえ

LoggingRole

サーバーが Amazon S3 または Amazon EFSevents の Amazon CloudWatch ログ記録をオンにできるようにする AWS Identity and Access Management (IAM) ロールの Amazon リソースネーム (ARN)。設定すると、 CloudWatch ログにユーザーアクティビティを表示できます。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 2,048 です。

パターン: `(|arn:.*role/\S+)`

必須: いいえ

PostAuthenticationLoginBanner

ユーザーがサーバーに接続するときに表示する文字列を指定します。この文字列はユーザーが認証した後で表示されます。

> **Note**
>
> SFTP プロトコルは認証後の表示バナーをサポートしていません。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 4096 です。

パターン: `[\x09-\x0D\x20-\x7E]*`

必須: いいえ

PreAuthenticationLoginBanner

ユーザーがサーバーに接続するときに表示する文字列を指定します。この文字列はユーザーが認証される前に表示されます。例えば、次のバナーはシステムの使用に関する詳細を表示します。

```
This system is for the use of authorized users only. Individuals using
this computer system without authority, or in excess of their authority,
are subject to having all of their activities on this system monitored
and recorded by system personnel.
```

型: 文字列

長さの制限: 最小長は 0 です。最大長は 4096 です。

パターン: `[\x09-\x0D\x20-\x7E]*`

必須: いいえ

ProtocolDetails

サーバー用に構成されたプロトコル設定。

- パッシブモード (FTP および FTPS プロトコル用) を示すには、PassiveIp パラメータを使用します。ファイアウォール、ルーター、ロードバランサーの外部 IP アドレスなど、クアッドドット形式の単一 IPv4 アドレスを入力します。
- S3 バケットにアップロードしているファイルに対して、クライアントが SETSTAT コマンドの使用を試みた時に発生するエラーを無視するには、SetStatOption パラメータを使用します。SFTP クライアントを変更せずに AWS Transfer Family サーバーでSETSTATコマンドを無視してファイルをアップロードするには、値を に設定しますENABLE_NO_OP。SetStatOption パラメータを に設定するとENABLE_NO_OP、Transfer Family は Amazon CloudWatch Logs にログエントリを生成し、クライアントがいつSETSTAT呼び出しを行っているかを特定できるようにします。
- AWS Transfer Family サーバーが一意のセッション ID を介して最近ネゴシエートされたセッションを再開するかどうかを確認するには、 TlsSessionResumptionModeパラメータを使用します。
- As2Transports は、AS2 メッセージの転送方法を示します。現在は、HTTP のみがサポートされます。

タイプ：ProtocolDetails オブジェクト

必須: いいえ

Protocols

ファイル転送プロトコルクライアントがサーバーのエンドポイントに接続できる 1 つまたは複数のファイル転送プロトコルを指定します。使用可能なプロトコルは次のとおりです。

- SFTP(Secure Shell (SSH) File Transfer Protocol): SSH 経由のファイル転送
- FTPS (File Transfer Protocol Secure): TLS 暗号化によるファイル転送
- FTP (File Transfer Protocol): 暗号化されていないファイル転送
- AS2 (適用性ステートメント 2): 構造化 business-to-business データの転送に使用されます

> **Note**
>
> - を選択した場合はFTPS、 AWS Certificate Manager (ACM) に保存されている証明書を選択する必要があります。この証明書は、クライアントが FTPS 経由でサーバーに接続するときにサーバーを識別するために使用されます。
> - Protocol に FTP または FTPS が含まれる場合、EndpointType は VPC でなければならず、IdentityProviderType は AWS_DIRECTORY_SERVICE、AWS_LAMBDA または API_GATEWAY でなければなりません。

- Protocol に FTPが含まれる場合、AddressAllocationIds は関連付けられません。
- Protocol が SFTP のみに設定されている場合、EndpointType は PUBLIC に設定でき、IdentityProviderType はサポートされている ID タイプ (SERVICE_MANAGED、AWS_DIRECTORY_SERVICE、AWS_LAMBDA、または API_GATEWAY) のいずれかに設定できます。
- Protocol が AS2 を含む場合、EndpointType は VPC でなければならず、ドメインは、Amazon S3 でなければなりません。

タイプ：文字列の配列

配列メンバー：最小数は 1 項目です。最大数は 4 項目です。

有効な値：SFTP | FTP | FTPS | AS2

必須：いいえ

S3StorageOptions

Amazon S3 ディレクトリのパフォーマンスを最適化するかどうかを指定します。これはデフォルトでは無効になっています。

デフォルトでは、ホームディレクトリマッピングの は TYPEですDIRECTORY。このオプションを有効にすると、マッピングにファイルターゲットを設定するHomeDirectoryMapEntryTypeFILE場合は、 を明示的に に設定する必要があります。

タイプ：S3StorageOptions オブジェクト

必須: いいえ

SecurityPolicyName

サーバーのセキュリティポリシーの名前を指定します。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 100 です。

Pattern: Transfer[A-Za-z0-9]*SecurityPolicy-[A-Za-z0-9-]+

必須: いいえ

ServerId

インスタンス化するサーバーの一意のシステム割り当て識別子を指定します。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `s-([0-9a-f]{17})`

必須: いいえ

State

記述されたサーバーの状態です。ONLINE の値は、サーバーがジョブを受け入れてファイルを転送できることを示します。OFFLINE の State 値は、サーバーがファイルオペレーションを実行できないことを意味します。

STARTING と STOPPING のステータスは、サーバーが中間状態である (完全に応答できないか完全にオフラインでない) ことを示します。START_FAILED または STOP_FAILED の値は、エラー条件を示すことができます。

型: 文字列

有効な値 : `OFFLINE | ONLINE | STARTING | STOPPING | START_FAILED | STOP_FAILED`

必須 : いいえ

StructuredLogDestinations

サーバーログの送信先となるロググループを指定します。

ロググループを指定するには、既存のロググループの ARN を指定する必要があります。この場合、ロググループの形式は次のようになります。

`arn:aws:logs:region-name:amazon-account-id:log-group:log-group-name:*`

例えば、次のようになります: `arn:aws:logs:us-east-1:111122223333:log-group:mytestgroup:*`

以前にサーバーのロググループを指定したことがある場合は、update-server 呼び出し時にこのパラメータに空の値を指定することで、そのロググループをクリアし、構造化ロギングを事実上無効にすることができます。例:

update-server --server-id s-1234567890abcdef0 --structured-log-destinations

タイプ：文字列の配列

配列メンバー: 最小数は 0 項目です。最大数は 1 項目です。

長さの制限: 最小長は 20 です。最大長は 1600 です。

Pattern: arn:\S+

必須: いいえ

Tags

記述したサーバーに割り当てられているサーバーを検索およびグループ化するために使用できるキーバリューペアを指定します。

型: Tag オブジェクトの配列

配列メンバー：最小数は 1 項目です。最大数は 50 項目です。

必須: いいえ

UserCount

ServerId で指定したサーバーに割り当てられるユーザーの数を指定します。

型: 整数

必須: いいえ

WorkflowDetails

割り当てるワークフローのワークフロー ID とワークフローの実行に使用する実行ロールを指定します。

ファイルのアップロード完了時に実行するワークフローに加えて、部分的なアップロードで実行するワークフローのワークフロー ID (および実行ロール) も WorkflowDetails に含めることができます。部分的なアップロードは、ファイルのアップロード中にサーバーセッションが切断されたときに発生します。

タイプ：WorkflowDetails オブジェクト

必須: いいえ

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

DescribedUser

指定されたユーザーのプロパティについて記述します。

内容

Arn

記述を要求されたユーザーの一意の Amazon リソースネーム (ARN) を指定します。

型: 文字列

長さの制限: 最小長は 20 です。最大長は 1600 です。

Pattern: `arn:\S+`

必須：はい

HomeDirectory

ユーザーがクライアントを使用してサーバーにログインするときの、ユーザーのランディングディレクトリ (フォルダ)。

`HomeDirectory` の例は `/bucket_name/home/mydirectory` です。

> **Note**
>
> `HomeDirectory` パラメータは、`HomeDirectoryType` が `PATH` に設定されている場合のみ使用されます。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 1,024 です。

パターン: `(|/.*)`

必須: いいえ

HomeDirectoryMappings

ユーザーに表示する Amazon S3 または Amazon EFS のパスとキー、およびそれらをどのように表示するかを指定する論理ディレクトリマッピング。`Entry` と `Target` のペアを指定する必要があり、`Entry` はパスの表示方法を示し、`Target` は実際の Amazon S3 または Amazon EFS のパ

スです。ターゲットを指定しただけの場合は、そのまま表示されます。また、 AWS Identity and Access Management (IAM) ロールが のパスへのアクセスを許可していることを確認する必要がありますTarget。この値は、HomeDirectoryType がLOGICALに設定されている場合にのみ設定できます。

ほとんどの場合、セッションポリシーの代わりにこの値を使用することで、指定されたホームディレクトリ (「chroot」) にユーザーをロックダウンできます。これを行うには、 Entryを '/' に設定し、 Target を HomeDirectory パラメータ値に設定します。

型: HomeDirectoryMapEntry オブジェクトの配列

配列メンバー：最小数は 1 項目です。最大数は 50,000 項目です。

必須: いいえ

HomeDirectoryType

ユーザーがサーバーにログインするときにホームディレクトリにするランディングディレクトリ (フォルダ) のタイプ。これを PATH に設定した場合、ユーザーには、ファイル転送プロトコルクライアントに、絶対的な Amazon S3 バケットまたは現状の Amazon EFS パスが表示されます。これを LOGICAL に設定した場合、Amazon S3 または Amazon EFS パスをユーザーに表示する方法に関して、HomeDirectoryMappings でマッピングを指定する必要があります。

> **Note**
>
> HomeDirectoryType が LOGICAL の場合は、HomeDirectoryMappings パラメータを使用してマッピングを指定する必要があります。一方、HomeDirectoryType が PATH の場合は、HomeDirectory パラメータを使用して絶対パスを指定します。テンプレートに HomeDirectory と HomeDirectoryMappings の両方を含めることはできません。

型: 文字列

有効な値：PATH | LOGICAL

必須：いいえ

Policy

複数のユーザーで同じ AWS Identity and Access Management (IAM) ロールを使用できるようにするためのユーザーのセッションポリシー。このポリシーは、ユーザーアクセスのス

コープを Amazon S3 バケットの一部に絞り込みます。このポリシー内に使用できる変数には、`${Transfer:UserName}`、`${Transfer:HomeDirectory}`、`${Transfer:HomeBucket}` があります。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 2048 です。

必須: いいえ

PosixProfile

Amazon Elastic File System (Amazon EFS) ファイルシステムへのユーザーのアクセスを制御する、ユーザー ID (`Uid`)、グループ ID (`Gid`)、およびセカンダリグループ ID (`SecondaryGids`) を含む、完全な POSIX ID を指定します。ファイルシステム内のファイルとディレクトリに設定される POSIX アクセス許可によって、Amazon EFS ファイルシステムとの間でファイルを転送するときにユーザーが得るアクセスのレベルが決まります。

型: PosixProfile オブジェクト

必須: いいえ

Role

Amazon S3 バケットまたは Amazon EFS ファイルシステムへのユーザーのアクセスを制御する AWS Identity and Access Management (IAM) ロールの Amazon リソースネーム (ARN)。Amazon S3 このロールにアタッチされたポリシーにより、ファイルを Amazon S3 バケットまたは Amazon EFS ファイルシステム間で転送する際の、ユーザーに付与するアクセスレベルが決定されます。IAM ロールには、ユーザーの転送リクエストを処理する際に、サーバーによるリソースへのアクセスを許可する信頼関係も含まれる必要があります。

型: 文字列

長さの制限: 最小長は 20 です。最大長は 2,048 です。

パターン: `arn:.*role/\S+`

必須: いいえ

SshPublicKeys

記述されたユーザー用に保存された Secure Shell (SSH) のパブリックキー部分を指定します。

タイプ：SshPublicKey オブジェクトの配列

配列メンバー：最小数は 0 項目です。最大数は 5 項目です。

必須: いいえ

Tags

リクエストされたユーザーのキーバリューペアを指定します。タグは、さまざまな目的でユーザーを検索やグループ化する際に使用できます。

型: Tag オブジェクトの配列

配列メンバー：最小数は 1 項目です。最大数は 50 項目です。

必須: いいえ

UserName

記述をリクエストされたユーザーの名前を指定します。ユーザー名は、認証の目的で使用されます。これは、ユーザーがサーバーにログインするときに使用される文字列です。

型: 文字列

長さの制限: 最小長は 3 です。最大長は 100 です。

Pattern: `[\w][\w@.-]{2,99}`

必須: いいえ

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

DescribedWorkflow

指定されたワークフローのプロパティを記述します。

内容

Arn

ワークフローの一意の Amazon リソースネーム (ARN) を指定します。

型: 文字列

長さの制限: 最小長は 20 です。最大長は 1600 です。

Pattern: `arn:\S+`

必須：はい

Description

ワークフローのテキスト説明を指定します。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 256 です。

パターン：`[\w- ]*`

必須: いいえ

OnExceptionSteps

ワークフローの実行中にエラーが発生した場合に実行する手順 (アクション) を指定します。

タイプ：WorkflowStep オブジェクトの配列

配列メンバー：最小数は 0 項目です。最大 8 項目。

必須: いいえ

Steps

指定したワークフロー内にあるステップの詳細を指定します。

タイプ：WorkflowStep オブジェクトの配列

配列メンバー：最小数は 0 項目です。最大 8 項目。

必須: いいえ

Tags

ワークフローのグループ化および検索に使用できるキーと値のペア。タグとは、任意の目的でユーザーにアタッチされるメタデータです。

型: Tag オブジェクトの配列

配列メンバー：最小数は 1 項目です。最大数は 50 項目です。

必須: いいえ

WorkflowId

ワークフローの一意の識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `w-([a-z0-9]{17})`

必須: いいえ

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

EfsFileLocation

ワークフローで使用するファイルの場所の詳細を指定します。ストレージに Amazon Elastic File Systems (Amazon EFS) を使用している場合にのみ適用されます。

内容

FileSystemId

Amazon EFS によって割り当てられたファイルシステムの識別子。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 128 です。

Pattern: `(arn:aws[-a-z]*:elasticfilesystem:[0-9a-z-:]+:(access-point/fsap|file-system/fs)-[0-9a-f]{8,40}|fs(ap)?-[0-9a-f]{8,40})`

必須: いいえ

Path

ワークフローで使用されているフォルダのパス名。

型: 文字列

長さの制限：最小長は 1 です。最大長は 65,536 です。

Pattern: `[^\x00]+`

必須: いいえ

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

EndpointDetails

ファイル転送プロトコル対応サーバー向けに構成された Virtual Private Cloud (VPC) エンドポイントの設定。VPC エンドポイントを使用すると、サーバーとリソースへのアクセスを VPC 内のみに制限できます。着信インターネットトラフィックを制御するには、`UpdateServer` API を実行し、Elastic IP アドレスをサーバーのエンドポイントにアタッチします。

> **Note**
>
> 2021 年 5 月 19 日以降、 AWS 2021 年 5 月 19 日より前にアカウントで `EndpointType=VPC_ENDPOINT` を使用してサーバーを作成していない場合、そのサーバーを作成することはできません。2021 年 5 月 19 日 AWS以前にアカウントに `EndpointType=VPC_ENDPOINT`で既にサーバーを作成している場合、影響を受けません。この日付を過ぎたら `EndpointType=VPC` を使用します。
> 詳細については、「VPC_ENDPOINT のサポート終了」を参照してください。

内容

AddressAllocationIds

Elastic IP アドレスをサーバーのエンドポイントにアタッチするために必要なアドレス割り当て ID のリスト。

アドレス割り当て ID は、Elastic IP アドレスの割り当て ID に対応します。この値は、Amazon EC2 アドレスデータ型から `allocationId`フィールドから取得できます。この値を取得する 1 つの方法は、EC2 DescribeAddresses API を呼び出すことです。

このパラメータはオプションです。VPC エンドポイントをパブリックにする場合は、このパラメータを設定します。詳細については、「サーバー用のインターネット向けエンドポイントを作成する」を参照してください。

> **Note**
>
> このプロパティは次のようにのみ設定できます。
>
> - `EndpointType` を に設定する必要があります VPC
> - Transfer Family サーバーはオフラインである必要があります。
> - FTP プロトコルを使用する Transfer Family サーバーでは、このパラメータを設定できません。

- サーバーには既に入力されている必要があります (SubnetIds と SubnetIds を同時に更新AddressAllocationIdsすることはできません)。
- AddressAllocationIds は重複を含めることはできません。また、 の長さは と同じである必要がありますSubnetIds。例えば、3 つのサブネット IDsつのアドレス割り当て IDsも指定する必要があります。
- UpdateServer API を呼び出して、このパラメータを設定または変更します。

タイプ：文字列の配列

必須: いいえ

SecurityGroupIds

サーバーのエンドポイントにアタッチできるセキュリティグループ ID のリスト。

> **Note**
> このプロパティは、EndpointType が VPC に設定されている場合にのみ設定できます。UpdateServer API で SecurityGroupIdsプロパティを編集できるのは、 を PUBLICまたは EndpointTypeから VPC_ENDPOINT に変更する場合のみですVPC。作成後にサーバーの VPC エンドポイントに関連付けられたセキュリティグループを変更するには、Amazon EC2 ModifyVpcEndpoint API を使用します。

タイプ：文字列の配列

長さの制限: 最小長は 11 です。最大長は 20 です。

Pattern: sg-[0-9a-f]{8,17}

必須: いいえ

SubnetIds

VPC でサーバーのエンドポイントをホストするために必要なサブネット ID のリスト。

> **Note**
> このプロパティは、EndpointType が VPC に設定されている場合にのみ設定できます。

型: 文字列の配列

必須: いいえ

VpcEndpointId

VPC エンドポイントの識別子。

> **Note**
>
> このプロパティは、EndpointType が VPC_ENDPOINT に設定されている場合にのみ設定できます。
> 詳細については、「VPC_ENDPOINT のサポート終了」を参照してください。

型: 文字列

長さの制限: 22 の固定長

Pattern: `vpce-[0-9a-f]{17}`

必須: いいえ

VpcId

サーバーのエンドポイントがホストされる VPC の識別子。

> **Note**
>
> このプロパティは、EndpointType が VPC に設定されている場合にのみ設定できます。

タイプ: 文字列

必須: いいえ

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++

- AWS SDK for Java V2
- AWS SDK for Ruby V3

ExecutionError

ワークフローの実行中に発生するエラーについて、エラーメッセージとタイプを指定します。

内容

Message

`ErrorType` に対応する記述的なメッセージを指定します。

型: 文字列

必須: はい

Type

エラータイプを指定します。

- `ALREADY_EXISTS`: 上書きオプションが選択されておらず、同じファイルが既にコピー先のインスタンスに同じファイルが存在している場合、これはコピーステップで発生します。
- `BAD_REQUEST`: 一般的な不正なリクエスト:たとえば、EFS ファイルにタグを付けようとするステップでは、S3ファイルにしかタグを付けることができないため、`BAD_REQUEST`が返されます。
- `CUSTOM_STEP_FAILED`: カスタムステップが失敗を示すコールバックを提供したときに発生します。
- `INTERNAL_SERVER_ERROR`: さまざまな理由で発生する可能性があります。
- `NOT_FOUND`: コピーステップのソースファイルなど、要求されたエンティティが存在しない場合に発生します。
- `PERMISSION_DENIED`: ワークフローの 1 つ以上のステップを完了するための正しいアクセス許可がポリシーに含まれていない場合に発生します。
- `TIMEOUT`: 実行がタイムアウトになったときに発生します。

  > **Note**
  >
  > 1 秒から 1,800 秒 (30 分) までのカスタムステップの`TimeoutSeconds`を設定することができます。

- `THROTTLED`: 1 秒あたり 1 つのワークフローという新しい実行リフィル率を超えた場合に発生します。

型: 文字列

有効な値：PERMISSION_DENIED | CUSTOM_STEP_FAILED | THROTTLED | ALREADY_EXISTS | NOT_FOUND | BAD_REQUEST | TIMEOUT | INTERNAL_SERVER_ERROR

必須：はい

以下の資料も参照してください。

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

ExecutionResults

ワークフロー内のステップならびにワークフローの実行中にエラーが発生した場合に実行するステップを指定します。

内容

OnExceptionSteps

ワークフローの実行中にエラーが発生した場合に実行する手順 (アクション) を指定します。

型: ExecutionStepResult オブジェクトの配列

配列メンバー：最小数は 1 項目です。最大数は 50 項目です。

必須: いいえ

Steps

指定したワークフロー内にあるステップの詳細を指定します。

型: ExecutionStepResult オブジェクトの配列

配列メンバー：最小数は 1 項目です。最大数は 50 項目です。

必須：いいえ

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

ExecutionStepResult

次のステップの詳細、つまりエラー (存在する場合)、出力 (存在する場合)、およびステップタイプを指定します。

内容

Error

指定されたワークフローステップの実行中に発生したエラーの詳細を指定します。

タイプ : ExecutionError オブジェクト

必須: いいえ

Outputs

ファイルにタグとして適用されるキーバリューペアの値。ステップタイプが TAG の場合にのみ適用されます。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 65,536 です。

必須: いいえ

StepType

利用可能なステップタイプのいずれか。

- **COPY** - ファイルを別の場所にコピーします。
- **CUSTOM** - AWS Lambda 関数ターゲットを使用してカスタムステップを実行します。
- **DECRYPT** - アップロード前に暗号化されていたファイルを復号化します。
- **DELETE** - ファイルを削除します。
- **TAG** - ファイルにタグを追加します。

型: 文字列

有効な値 : COPY | CUSTOM | TAG | DELETE | DECRYPT

必須 : いいえ

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

FileLocation

ステップで使用する Amazon S3 または EFS ファイルの詳細を指定します。

内容

EfsFileLocation

使用するファイルの Amazon EFS 識別子とパスを指定します。

タイプ：EfsFileLocation オブジェクト

必須: いいえ

S3FileLocation

バケット、Etag など、S3 で使用されるファイルの詳細を指定します。

型: S3FileLocation オブジェクト

必須: いいえ

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

HomeDirectoryMapEntry

HomeDirectoryMappings のエントリとターゲットを含むオブジェクトを表します。

以下は、chroot についての Entry と Target のペアの例です。

```
[ { "Entry": "/", "Target": "/bucket_name/home/mydirectory" } ]
```

内容

Entry

HomeDirectoryMappings のエントリを表します。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 1,024 です。

Pattern: `/.*`

必須：はい

Target

HomeDirectoryMapEntry で使用されるマップターゲットを表します。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 1,024 です。

Pattern: `/.*`

必須：はい

Type

マッピングのタイプを指定します。マッピングがファイルを指すFILE場合は タイプを に設定し、ディレクトリDIRECTORYがディレクトリを指す場合は に設定します。

> **Note**
>
> デフォルトでは、Transfer Family サーバーを作成するDIRECTORYとき、ホームディレクトリマッピングには Type の があります。マッピングにファイルターゲットを設定するFILE場合は、明示的に を Typeに設定する必要があります。

型: 文字列

有効な値：`FILE | DIRECTORY`

必須：いいえ

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

IdentityProviderDetails

ファイル転送プロトコル対応サーバーのユーザーに使用されているユーザー認証のタイプに関する情報を返します。サーバーの認証方法となりうるのは 1 つのみです。

内容

DirectoryId

ID プロバイダーとして使用する AWS Directory Service ディレクトリの識別子。

型: 文字列

長さの制限: 固定長は 12 です。

Pattern: `d-[0-9a-f]{10}`

必須: いいえ

Function

Identity プロバイダに使用する Lambda 関数の ARN。

型: 文字列

長さの制限: 最小長は 1 です。最大長は 170 です。

パターン: `arn:[a-z-]+:lambda:.*`

必須: いいえ

InvocationRole

このパラメータは、`IdentityProviderType` が `API_GATEWAY` の場合にのみ適用されます。ユーザーアカウントを認証するために使用される `InvocationRole` のタイプを提供します。

型: 文字列

長さの制限: 最小長は 20 です。最大長は 2,048 です。

パターン: `arn:.*role/\S+`

必須: いいえ

SftpAuthenticationMethods

SFTPが有効なサーバーおよびカスタムアイデンティティプロバイダーの場合のみ、パスワード、SSHキーペア、またはその両方を使用して認証するかどうかを指定できます。

- PASSWORD - ユーザーは接続するためにパスワードを提供する必要があります。
- PUBLIC_KEY - 接続するにはユーザーがプライベートキーを入力する必要があります。
- PUBLIC_KEY_OR_PASSWORD - ユーザーは、パスワードまたはキーのいずれかで認証できます。これは、デフォルト値です。
- PUBLIC_KEY_AND_PASSWORD - 接続するには、ユーザーはプライベートキーとパスワードの両方を入力する必要があります。サーバーは最初にキーを確認し、キーが有効な場合はパスワードの入力を求めます。提供されたプライベートキーが保存されたパブリックキーと一致しない場合、認証は失敗します。

型: 文字列

有効な値：PASSWORD | PUBLIC_KEY | PUBLIC_KEY_OR_PASSWORD | PUBLIC_KEY_AND_PASSWORD

必須：いいえ

Url

ユーザーを認証するために使用されるサービスエンドポイントの場所を提供します。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 255 です。

必須：いいえ

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

InputFileLocation

処理中のファイルの場所を指定します。

内容

EfsFileLocation

復号化される Amazon Elastic File System (Amazon EFS) ファイルの詳細を指定します。

タイプ：EfsFileLocation オブジェクト

必須: いいえ

S3FileLocation

コピーまたは復号化される Amazon S3 ファイルの詳細を指定します。

タイプ：S3InputFileLocation オブジェクト

必須: いいえ

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

ListedAccess

指定され関連付けられたアクセス権のプロパティを一覧表示します。

内容

ExternalId

ディレクトリ内の特定のグループを識別するために必要な一意の識別子。関連付けるグループのユーザーは、 を使用して、有効なプロトコル経由で Amazon S3 または Amazon EFS リソースにアクセスできます AWS Transfer Family。グループ名がわかっている場合は、Windows を使用して次のコマンドを実行することで SID 値を表示できます PowerShell。

```
Get-ADGroup -Filter {samAccountName -like "YourGroupName*"} -Properties
* | Select SamAccountName,ObjectSid
```

そのコマンドで、 を Active Directory グループの名前YourGroupNameに置き換えます。

このパラメータの検証に使用される正規表現は、スペースを含まない大文字と小文字の英数字からなる文字列です。下線文字 (_) や =,.@:/- の文字を含めることもできます。

型: 文字列

長さの制限：最小長は 1 です。最大長は 256 です。

パターン：`S-1-[\d-]+`

必須: いいえ

HomeDirectory

ユーザーがクライアントを使用してサーバーにログインするときの、ユーザーのランディングディレクトリ (フォルダ)。

`HomeDirectory` の例は `/bucket_name/home/mydirectory` です。

> **Note**
>
> `HomeDirectory` パラメータは、`HomeDirectoryType` が `PATH` に設定されている場合のみ使用されます。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 1,024 です。

パターン: `(|/.*)`

必須: いいえ

HomeDirectoryType

ユーザーがサーバーにログインするときにホームディレクトリにするランディングディレクトリ (フォルダ) のタイプ。これを `PATH` に設定した場合、ユーザーには、ファイル転送プロトコルクライアントに、絶対的な Amazon S3 バケットまたは現状の Amazon EFS パスが表示されます。これを `LOGICAL` に設定した場合、Amazon S3 または Amazon EFS パスをユーザーに表示する方法に関して、`HomeDirectoryMappings` でマッピングを指定する必要があります。

> **Note**
>
> `HomeDirectoryType` が `LOGICAL` の場合は、`HomeDirectoryMappings` パラメータを使用してマッピングを指定する必要があります。一方、`HomeDirectoryType` が `PATH` の場合は、`HomeDirectory` パラメータを使用して絶対パスを指定します。テンプレートに `HomeDirectory` と `HomeDirectoryMappings` の両方を含めることはできません。

型: 文字列

有効な値：`PATH | LOGICAL`

必須：いいえ

Role

Amazon S3 バケットまたは Amazon EFS ファイルシステムへのユーザーのアクセスを制御する AWS Identity and Access Management (IAM) ロールの Amazon リソースネーム (ARN)。このロールにアタッチされたポリシーにより、ファイルを Amazon S3 バケットまたは Amazon EFS ファイルシステム間で転送する際の、ユーザーに付与するアクセスレベルが決定されます。IAM ロールには、ユーザーの転送リクエストを処理する際に、サーバーによるリソースへのアクセスを許可する信頼関係も含まれる必要があります。

型: 文字列

長さの制限: 最小長は 20 です。最大長は 2,048 です。

パターン: `arn:.*role/\S+`

必須: いいえ

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

ListedAgreement

契約のプロパティを説明します。

内容

AgreementId

契約の一意の識別子。この識別子は、契約を作成するときに返されます。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `a-([0-9a-f]{17})`

必須: いいえ

Arn

指定された契約の Amazon リソースネーム (ARN)。

型: 文字列

長さの制限: 最小長は 20 です。最大長は 1600 です。

Pattern: `arn:\S+`

必須: いいえ

Description

契約の現在の説明。`UpdateAgreement` 操作を呼び出して新しい説明を指定することで変更できます。

型: 文字列

長さの制限: 最小長は 1 です。最大長は 200 です。

Pattern: `[\p{Graph}]+`

必須: いいえ

LocalProfileId

AS2 ローカルプロファイルの一意の識別子です。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `p-([0-9a-f]{17})`

必須: いいえ

PartnerProfileId

パートナープロファイルの一意の識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `p-([0-9a-f]{17})`

必須: いいえ

ServerId

契約の一意の識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `s-([0-9a-f]{17})`

必須: いいえ

Status

合意内容は `ACTIVE` または `INACTIVE` のいずれかになります。

型: 文字列

有効な値：`ACTIVE | INACTIVE`

必須：いいえ

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

ListedCertificate

証明書のプロパティについて記述します。

内容

ActiveDate

証明書がいつアクティブになるかを指定するオプションの日付です。

型: タイムスタンプ

必須: いいえ

Arn

指定された証明書の Amazon リソースネーム (ARN)

型: 文字列

長さの制限: 最小長は 20 です。最大長は 1600 です。

Pattern: `arn:\S+`

必須: いいえ

CertificateId

インポートされた証明書の識別子の配列です。この識別子は、プロファイルやパートナープロファイルの操作に使用します。

型: 文字列

長さの制限: 22 の固定長

Pattern: `cert-([0-9a-f]{17})`

必須: いいえ

Description

証明書を識別するための名前または短い説明。

型: 文字列

長さの制限: 最小長は 1 です。最大長は 200 です。

Pattern: `[\p{Graph}]+`

必須: いいえ

InactiveDate

証明書がいつ非アクティブになるかを指定するオプションの日付です。

型: タイムスタンプ

必須: いいえ

Status

証明書は、ACTIVE、PENDING_ROTATION、INACTIVE のいずれかになります。PENDING_ROTATION は、現在の証明書の有効期限が切れると、この証明書が置き換えられることを意味します。

型: 文字列

有効な値 : `ACTIVE | PENDING_ROTATION | INACTIVE`

必須 : いいえ

Type

証明書のタイプ。証明書にプライベートキーが指定されている場合、そのタイプは CERTIFICATE_WITH_PRIVATE_KEY です。プライベートキーがない場合、タイプは CERTIFICATE です。

型: 文字列

有効な値 : `CERTIFICATE | CERTIFICATE_WITH_PRIVATE_KEY`

必須 : いいえ

Usage

この証明書の使用方法を指定します。これは、次の方法で使用できます。

- SIGNING: AS2 メッセージの署名用
- ENCRYPTION: AS2 メッセージの暗号化用
- TLS: HTTPS 経由で送信される AS2 通信を保護する場合

型: 文字列

有効な値：`SIGNING | ENCRYPTION`

必須：いいえ

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

ListedConnector

指定されたコネクタの詳細を返します。

内容

Arn

指定されたコネクタの Amazon リソースネーム (ARN)。

型: 文字列

長さの制限: 最小長は 20 です。最大長は 1600 です。

Pattern: `arn:\S+`

必須: いいえ

ConnectorId

コネクタの一意の識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `c-([0-9a-f]{17})`

必須: いいえ

Url

パートナーの AS2 または SFTP エンドポイントの URL。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 255 です。

必須：いいえ

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

ListedExecution

指定された実行のプロパティを返します。

内容

ExecutionId

ワークフローの一意の識別子。

型: 文字列

長さの制限: 最大長は 36 です。

Pattern: `[0-9a-fA-F]{8}\-[0-9a-fA-F]{4}\-[0-9a-fA-F]{4}\-[0-9a-fA-F]{4}\-[0-9a-fA-F]{12}`

必須: いいえ

InitialFileLocation

Amazon S3 または EFS ファイルの場所を記述する構造。これは実行開始時のファイルの場所であり、ファイルをコピーしようとする場合にはファイルの (コピー先ではなく) 初期の場所です。

型: FileLocation オブジェクト

必須: いいえ

ServiceMetadata

ワークフローに関連付けられたセッション詳細のコンテナオブジェクト。

タイプ : ServiceMetadata オブジェクト

必須: いいえ

Status

ステータスは実行のいずれかです。進行中、完了、例外発生、または例外処理中の可能性があります。

型: 文字列

有効な値 : `IN_PROGRESS | COMPLETED | EXCEPTION | HANDLING_EXCEPTION`

必須：いいえ

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

ListedHostKey

指定されたホストキーのプロパティを返します。

内容

Arn

ホストキーの一意の Amazon リソースネーム（ARN）。

型: 文字列

長さの制限: 最小長は 20 です。最大長は 1600 です。

Pattern: `arn:\S+`

必須：はい

DateImported

ホストキーがサーバーに追加された日付。

型: タイムスタンプ

必須: いいえ

Description

ホストキーの現在の説明。`UpdateHostKey` 操作を呼び出して新しい説明を指定することで変更できます。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 200 です。

Pattern: `[\p{Print}]*`

必須: いいえ

Fingerprint

パブリックキーのフィンガープリントは、長い方のパブリックキーを識別するために使用される短いバイト列です。

タイプ: 文字列

必須: いいえ

HostKeyId

ホストキーの一意の識別子。

型: 文字列

長さの制限: 固定長は 25 です。

Pattern: `hostkey-[0-9a-f]{17}`

必須: いいえ

Type

ホストキーに使用される暗号化アルゴリズムの種類。Type パラメータは、次のいずれかの値を使用して指定します。

- `ssh-rsa`
- `ssh-ed25519`
- `ecdsa-sha2-nistp256`
- `ecdsa-sha2-nistp384`
- `ecdsa-sha2-nistp521`

タイプ: 文字列

必須: いいえ

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

ListedProfile

指定されたプロファイルのプロパティを返します。

内容

Arn

指定されたプロファイルの Amazon リソースネーム (ARN)。

型: 文字列

長さの制限: 最小長は 20 です。最大長は 1600 です。

Pattern: `arn:\S+`

必須: いいえ

As2Id

`As2Id` は、RFC 4130 で定義されている AS2-name です。インバウンド転送の場合、これはパートナーから送信される AS2 メッセージの `AS2-From` ヘッダーです。アウトバウンドコネクタの場合、これは `StartFileTransfer` API オペレーションを使用してパートナーに送信される AS2 メッセージの `AS2-To` ヘッダーです。この ID にスペースを含めることはできません。

型: 文字列

長さの制限: 最小長は 1 です。最大長は 128 です。

Pattern: `[\p{Print}\s]*`

必須: いいえ

ProfileId

ローカルまたはパートナー AS2 プロファイルの一意の識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `p-([0-9a-f]{17})`

必須: いいえ

ProfileType

LOCAL タイププロファイルのみを一覧表示するか、PARTNER タイププロファイルのみを一覧表示するかを示します。リクエストで指定されていない場合、コマンドはすべてのタイプのプロファイルを一覧表示します。

型: 文字列

有効な値：LOCAL | PARTNER

必須：いいえ

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

ListedServer

指定されたファイル転送プロトコル対応サーバーのプロパティを返します。

内容

Arn

一覧表示するサーバーについて一意の Amazon リソースネーム (ARN) を指定します。

型: 文字列

長さの制限: 最小長は 20 です。最大長は 1600 です。

Pattern: `arn:\S+`

必須：はい

Domain

ファイル転送に使用されるストレージシステムのドメインを指定します。使用できるドメインは、Amazon Simple Storage Service (Amazon S3) と Amazon Elastic File System (Amazon EFS) です。デフォルト値は S3 です。

型: 文字列

有効な値：`S3 | EFS`

必須：いいえ

EndpointType

サーバーの接続先になる VPC エンドポイントのタイプを指定します。サーバーが VPC エンドポイントに接続している場合、パブリックインターネット経由で はサーバーにはアクセスできません。

型: 文字列

有効な値：`PUBLIC | VPC | VPC_ENDPOINT`

必須：いいえ

IdentityProviderType

サーバーの認証のモード。デフォルト値は です。これによりSERVICE_MANAGED、 AWS Transfer Family サービス内でユーザー認証情報を保存してアクセスできます。

AWS_DIRECTORY_SERVICE を使用して、オンプレミス環境の AWS Directory Service for Microsoft Active Directory または Microsoft Active Directory、または AD Connector AWS を使用する の Active Directory グループへのアクセスを提供します。このオプションでは、IdentityProviderDetails パラメータを使用してディレクトリ ID を指定する必要もあります。

この API_GATEWAY 値を使用して、選択した ID プロバイダーと統合します。API_GATEWAY 設定では、IdentityProviderDetails パラメータを使用して認証を呼び出すには Amazon API Gateway エンドポイントの URL を指定する必要があります。

AWS_LAMBDA 値を使用して、 AWS Lambda 関数を ID プロバイダーとして直接使用します。この値を選択した場合、Function パラメータで、IdentityProviderDetails データ型の Lambda 関数の ARN を指定する必要があります。

型: 文字列

有効な値：SERVICE_MANAGED | API_GATEWAY | AWS_DIRECTORY_SERVICE | AWS_LAMBDA

必須：いいえ

LoggingRole

サーバーが Amazon S3 または Amazon EFSevents の Amazon CloudWatch ログ記録をオンにできるようにする AWS Identity and Access Management (IAM) ロールの Amazon リソースネーム (ARN)。設定すると、 CloudWatch ログにユーザーアクティビティを表示できます。

型: 文字列

長さの制限: 最小長は 20 です。最大長は 2,048 です。

パターン: arn:.*role/\S+

必須: いいえ

ServerId

一覧表示されたサーバーについて一意のシステム割り当て識別子を指定します。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `s-([0-9a-f]{17})`

必須: いいえ

State

記述されたサーバーの状態です。ONLINE の値は、サーバーがジョブを受け入れてファイルを転送できることを示します。OFFLINE の State 値は、サーバーがファイルオペレーションを実行できないことを意味します。

STARTING と STOPPING のステータスは、サーバーが中間状態である (完全に応答できないか完全にオフラインでない) ことを示します。START_FAILED または STOP_FAILED の値は、エラー条件を示すことができます。

型: 文字列

有効な値 : `OFFLINE | ONLINE | STARTING | STOPPING | START_FAILED | STOP_FAILED`

必須 : いいえ

UserCount

ServerId で指定したサーバーに割り当てられるユーザーの数を指定します。

型: 整数

必須 : いいえ

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

ListedUser

指定するユーザーのプロパティを返します。

内容

Arn

知りたいユーザーの一意の Amazon リソースネーム (ARN) を指定します。

型: 文字列

長さの制限: 最小長は 20 です。最大長は 1600 です。

Pattern: `arn:\S+`

必須：はい

HomeDirectory

ユーザーがクライアントを使用してサーバーにログインするときの、ユーザーのランディングディレクトリ (フォルダ)。

`HomeDirectory` の例は `/bucket_name/home/mydirectory` です。

> **Note**
>
> `HomeDirectory` パラメータは、`HomeDirectoryType` が `PATH` に設定されている場合のみ使用されます。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 1,024 です。

パターン: `(|/.*)`

必須: いいえ

HomeDirectoryType

ユーザーがサーバーにログインするときにホームディレクトリにするランディングディレクトリ (フォルダ) のタイプ。これを `PATH` に設定した場合、ユーザーには、ファイル転送プロトコルクライアントに、絶対的な Amazon S3 バケットまたは現状の Amazon EFS パスが表示されます。

これを LOGICAL に設定した場合、Amazon S3 または Amazon EFS パスをユーザーに表示する方法に関して、HomeDirectoryMappings でマッピングを指定する必要があります。

> **Note**
>
> HomeDirectoryType が LOGICAL の場合は、HomeDirectoryMappings パラメータを使用してマッピングを指定する必要があります。一方、HomeDirectoryType が PATH の場合は、HomeDirectory パラメータを使用して絶対パスを指定します。テンプレートに HomeDirectory と HomeDirectoryMappings の両方を含めることはできません。

型: 文字列

有効な値：PATH | LOGICAL

必須：いいえ

Role

Amazon S3 バケットまたは Amazon EFS ファイルシステムへのユーザーのアクセスを制御する AWS Identity and Access Management (IAM) ロールの Amazon リソースネーム (ARN)。このロールにアタッチされたポリシーにより、ファイルを Amazon S3 バケットまたは Amazon EFS ファイルシステム間で転送する際の、ユーザーに付与するアクセスレベルが決定されます。IAM ロールには、ユーザーの転送リクエストを処理する際に、サーバーによるリソースへのアクセスを許可する信頼関係も含まれる必要があります。

> **Note**
>
> Domain=S3 を持つサーバー用の Amazon S3 バケットまたは Domain=EFS を持つサーバー用の EFS ファイルシステムへのユーザーのアクセスを制御する IAM ロール、。このロールにアタッチされたポリシーにより、ファイルを S3 バケットまたは EFS ファイルシステム間で転送する際の、ユーザーに付与するアクセスのレベルが決定されます。

型: 文字列

長さの制限: 最小長は 20 です。最大長は 2,048 です。

パターン: arn:.*role/\S+

必須: いいえ

SshPublicKeyCount

指定したユーザー用に保存される SSH 公開キーの数を指定します。

型: 整数

必須: いいえ

UserName

ARN が指定されているユーザーの名前を指定します。ユーザー名は、認証の目的で使用されます。

型: 文字列

長さの制限: 最小長は 3 です。最大長は 100 です。

Pattern: `[\w][\w@.-]{2,99}`

必須: いいえ

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

ListedWorkflow

ワークフローの識別子、テキスト記述、および Amazon リソースネーム (ARN) が含まれます。

内容

Arn

ワークフローの一意の Amazon リソースネーム (ARN) を指定します。

型: 文字列

長さの制限: 最小長は 20 です。最大長は 1600 です。

Pattern: `arn:\S+`

必須: いいえ

Description

ワークフローのテキスト説明を指定します。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 256 です。

パターン：`[\w- ]*`

必須: いいえ

WorkflowId

ワークフローの一意の識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `w-([a-z0-9]{17})`

必須: いいえ

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

LoggingConfiguration

ログ記録ロールとロググループ名で構成されます。

内容

LoggingRole

サーバーが Amazon S3 または Amazon EFSevents の Amazon CloudWatch ログ記録をオンにできるようにする AWS Identity and Access Management (IAM) ロールの Amazon リソースネーム (ARN)。設定すると、 CloudWatch ログにユーザーアクティビティを表示できます。

型: 文字列

長さの制限: 最小長は 20 です。最大長は 2,048 です。

パターン: `arn:.*role/\S+`

必須: いいえ

LogGroupName

このワークフローが属する AWS Transfer Family サーバーの CloudWatch ログ記録グループの名前。

型: 文字列

長さの制限：最小長は 1 です。最大長は 512 です。

Pattern: `[\.\-_/#A-Za-z0-9]*`

必須: いいえ

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

PosixProfile

ユーザーの Amazon EFS ファイルシステムへのアクセスを制御する、ユーザー ID (Uid Gid)、グループ ID (SecondaryGids)、およびセカンダリグループ ID () を含む完全な POSIX ID。ファイルシステム内のファイルとディレクトリに設定される POSIX アクセス許可によって、Amazon EFS ファイルシステムとの間でファイルを転送するときにユーザーが得るアクセスのレベルが決まります。

内容

Gid

このユーザーがすべての EFS 操作で使用する POSIX グループ ID。

型: 長整数

有効な範囲: 最小値 は 0 です。最大値は 4,294,967,295 です。

必須: はい

Uid

このユーザーのすべての EFS 操作で使用する POSIX ユーザー ID。

型: 長整数

有効な範囲: 最小値 は 0 です。最大値は 4,294,967,295 です。

必須: はい

SecondaryGids

このユーザーのすべての EFS 操作に使用するセカンダリ POSIX グループ ID。

型: 長整数

配列メンバー：最小数は 0 項目です。最大数は 16 項目です。

有効な範囲：最小値 は 0 です。最大値は 4,294,967,295 です。

必須：いいえ

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

ProtocolDetails

サーバー用に構成されたプロトコル設定。

内容

As2Transports

AS2 メッセージの転送方法を示します。現在は、HTTP のみがサポートされます。

タイプ：文字列の配列

配列メンバー: 定数は 1 項目です。

有効な値：HTTP

必須：いいえ

PassiveIp

FTP および FTP プロトコルのパッシブモードを示します。ファイアウォール、ルーター、ロードバランサーのパブリック IP アドレスなど、単一 IPv4 アドレスを入力します。例:

```
aws transfer update-server --protocol-details PassiveIp=0.0.0.0
```

上の例では、`0.0.0.0` を実際に使用する IP アドレスに置き換えます。

> **Note**
>
> `PassiveIp` 値を変更した場合、変更を反映するには、転送ファミリーサーバーを停止してから再起動する必要があります。NAT 環境でパッシブモード (PASV) を使用する方法の詳細については、「ファイアウォールの背後にある FTPS サーバーの設定」または「 で NAT を設定する AWS Transfer Family」を参照してください。

特殊な値

`AUTO` と `0.0.0.0` は、`PassiveIp` パラメータの特殊な値です。FTP および FTPS タイプのサーバーにデフォルトで値 `PassiveIp=AUTO` が割り当てられます。この場合、サーバーは自動的に PASV 応答内のエンドポイント IP の 1 つで応答します。`PassiveIp=0.0.0.0` にはその使い方についてよりユニークなアプリケーションがあります。たとえば、サブネットが 3 つある高可用性 (HA) Network Load Balancer (NLB) 環境では、`PassiveIp` パラメータを使用して単一の IP アドレスのみを指定できます。これにより、高可用性の有効性が低下します。この場

合、PassiveIp=0.0.0.0 を指定することができます。これにより、クライアントはコントロール接続と同じ IP アドレスを使用し、接続にはすべての AZ を使用するように指示されます。ただし、すべての FTP クライアントが PassiveIp=0.0.0.0 response をサポートしているわけではありません。 FileZilla WinSCP がそれをサポートしていることに注意してください。他のクライアントを使用している場合は、そのクライアントが PassiveIp=0.0.0.0 応答をサポートしているかどうかを確認してください。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 15 です。

必須: いいえ

SetStatOption

SetStatOption を使用して、S3 バケットにアップロードしているファイルに対して、クライアントが SETSTAT の使用を試みたときに生成されるエラーを無視します。

一部の SFTP ファイル転送クライアントは、ファイルをアップロードする場合に、SETSTAT などコマンドを使用して、タイムスタンプやアクセス権限などを含んだリモートファイルの属性を変更しようとすることがあります。ただし、これらのコマンドは Amazon S3 などのオブジェクトストレージシステムと互換性がありません。この非互換性が原因で、これらのクライアントからファイルをアップロードする際に、ファイルが正常にアップロードされた場合でもエラーが発生する可能性があります。

Transfer Family サーバーが SETSTAT コマンドを無視するように値を ENABLE_NO_OP に設定し、SFTP クライアントに変更を加えることなくファイルをアップロードできます。SetStatOption ENABLE_NO_OP 設定ではエラーは無視されますが、Amazon CloudWatch Logs にログエントリが生成されるため、クライアントがSETSTATいつ呼び出しを行っているかを判断できます。

> **Note**
>
> ファイルの元のタイムスタンプを保持し、SETSTAT を使用して他のファイル属性を変更する場合は、Transfer Family のバックエンドストレージとして Amazon EFS を使用できます。

型: 文字列

有効な値：DEFAULT | ENABLE_NO_OP

必須：いいえ

TlsSessionResumptionMode

FTPS プロトコルを使用する Transfer Family サーバーで使用されるプロパティ。TLS セッション再開は、FTPS セッションの制御接続とデータ接続の間でネゴシエートされた秘密鍵を再開または共有するメカニズムを提供します。TlsSessionResumptionMode サーバーが一意のセッション ID を使用して最近ネゴシエートされたセッションを再開するかどうかを決定します。このプロパティは、CreateServer の間および UpdateServer の呼び出し中に使用可能です。TlsSessionResumptionMode の値が CreateServer の間に指定されていない場合、デフォルトで ENFORCED に設定されます。

- DISABLED: サーバーは TLS セッション再開クライアント要求を処理せず、要求ごとに新しい TLS セッションを作成します。
- ENABLED: サーバーは TLS セッション再開を実行しているクライアントを処理して受け入れます。サーバーは、TLS セッション再開クライアント処理を実行しないクライアントデータ接続を拒否しません。
- ENFORCED: サーバーは TLS セッション再開を実行しているクライアントを処理して受け入れます。サーバーは、TLS セッションの再開クライアント処理を実行しないクライアントデータ接続を拒否します。値を ENFORCED に設定する前に、クライアントをテストします。

> **Note**
>
> すべての FTPS クライアントが TLS セッションの再開を実行するわけではありません。そのため、TLS セッションの再開を強制することを選択した場合、プロトコルネゴシエーションを実行しない FTPS クライアントからの接続は禁止されます。ENFORCED の値を使用できるかどうかを判断するには、クライアントをテストする必要があります。

型: 文字列

有効な値：DISABLED | ENABLED | ENFORCED

必須：いいえ

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

S3FileLocation

ワークフローで使用するファイルの場所の詳細を指定します。S3 ストレージを使用している場合にのみ適用されます。

内容

Bucket

使用されるファイルを含む S3 バケットを指定します。

型: 文字列

長さの制限: 最小長は 3 です。最大長は 63 です。

パターン: `[a-z0-9][\.\-a-z0-9]{1,61}[a-z0-9]`

必須: いいえ

Etag

エンティティタグは、オブジェクトのハッシュです。ETag は、オブジェクトのコンテンツに加えた変更のみを反映し、メタデータに加えた変更は反映しません。

型: 文字列

長さの制限：最小長は 1 です。最大長は 65,536 です。

Pattern: `.+`

必須: いいえ

Key

Amazon S3 でファイルの作成時にファイルに割り当てられた名前。オブジェクトを取得するには、オブジェクトキーを使用します。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 1,024 です。

パターン: `[\P{M}\p{M}]*`

必須: いいえ

VersionId

ファイルのバージョンを指定します。

型: 文字列

長さの制限: 最小長は 1 です。最大長は 1,024 です。

パターン: .+

必須: いいえ

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

S3InputFileLocation

カスタマー入力 Amazon S3ファイルの場所を指定します。copyStepDetails.DestinationFileLocation の内部で使用する場合、S3 コピー先にしてください。

バケットとキーを提供する必要があります。キーは、パスまたはファイルのいずれかを表すことができます。これは、キー値をスラッシュ (/) の文字で終了するかどうかによって決まります。最後の文字が「/」の場合、ファイルはフォルダにコピーされ、その名前は変わりません。最後の文字が英数字の場合は、アップロードしたファイルの名前が path 値に変更されます。その名前のファイルが既に存在する場合、ファイルは上書きされます。

たとえば、パスが shared-files/bob/ である場合、アップロードしたファイルは shared-files/bob/ フォルダにコピーされます。当該パスが shared-files/today である場合、アップロードした各ファイルは shared-files フォルダにtoday という名前でコピーされ、アップロードのたびに以前のバージョンの Bob ファイルが上書きされます。

内容

Bucket

カスタマー入力ファイルの S3 バケットを指定します。

型: 文字列

長さの制限: 最小長は 3 です。最大長は 63 です。

パターン: [a-z0-9][\.\-a-z0-9]{1,61}[a-z0-9]

必須: いいえ

Key

Amazon S3 でファイルの作成時にファイルに割り当てられた名前。オブジェクトを取得するには、オブジェクトキーを使用します。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 1,024 です。

パターン: [\P{M}\p{M}]*

必須: いいえ

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

S3StorageOptions

サーバー用に設定された Amazon S3 ストレージオプション。

内容

DirectoryListingOptimization

Amazon S3 ディレクトリのパフォーマンスを最適化するかどうかを指定します。これはデフォルトでは無効になっています。

デフォルトでは、ホームディレクトリマッピングの は TYPEですDIRECTORY。
このオプションを有効にすると、マッピングにファイルターゲットを設定するHomeDirectoryMapEntryTypeFILE場合は、 を明示的に に設定する必要があります。

型: 文字列

有効な値：ENABLED | DISABLED

必須：いいえ

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

S3Tag

タグ付けステップの実行中にファイルに割り当てられるキーバリューペアを指定します。

内容

Key

作成するタグに割り当てた名前。

型: 文字列

長さの制限: 最小長は 1 です。最大長は 128 です。

パターン: `([\p{L}\p{Z}\p{N}_.:/=+\-@]*)`

必須：はい

Value

キーに対応する値。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 256 です。

パターン：`([\p{L}\p{Z}\p{N}_.:/=+\-@]*)`

必須：はい

以下の資料も参照してください。

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

ServiceMetadata

ワークフローに関連付けられたセッション詳細のコンテナオブジェクト。

内容

UserDetails

サーバー ID (`ServerId`)、セッション ID (`SessionId`)、ユーザー (`UserName`) で `UserDetails` を構成します。

型: UserDetails オブジェクト

必須：はい

以下の資料も参照してください。

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

SftpConnectorConfig

SFTP コネクタオブジェクトの詳細が含まれます。コネクタオブジェクトは、パートナーの SFTP サーバーとの間でファイルを転送するために使用されます。

> **Note**
>
> SftpConnectorConfig データ型は SFTP コネクタとそのパラメータの作成と更新の両方に使用されるため、 TrustedHostKeys および UserSecretIdは必須ではないとマークされます。これは、既存の SFTP コネクタを更新するときに必要ではなく、新しい SFTP コネクタを作成するときに必要になるため、少し誤解を招くものです。

内容

TrustedHostKeys

接続先の外部サーバーを識別するために使用されるホストキー (1 つまたは複数) のパブリック部分。SFTP サーバーに対して ssh-keyscan コマンドを使用して、必要なキーを取得できます。

SSH パブリックキーフォーマットの標準要素は <key type>、<body base64>、そしてオプションの <comment> の 3 つで、各要素の間にスペースがあります。<key type> と <body base64> だけを指定し、鍵の <comment> 部分は入力しないでください。

信頼できるホストキーの場合、 は RSA キーと ECDSA キー AWS Transfer Family を受け入れます。

- RSA キーの場合、<key type> 文字列は ssh-rsa です。
- ECDSA キーの場合、生成したキーのサイズに応じて、<key type> 文字列は ecdsa-sha2-nistp256、ecdsa-sha2-nistp384、または ecdsa-sha2-nistp521 のいずれかになります。

このコマンドを実行して、SFTP サーバーのホストキーを取得します。SFTP サーバー名は ですftp.host.com。

```
ssh-keyscan ftp.host.com
```

これにより、パブリックホストキーが標準出力に出力されます。

```
ftp.host.com ssh-rsa AAAAB3Nza...<long-string-for-public-key
```

この文字列をコピーして、 `create-connector` コマンドの `TrustedHostKeys`フィールドまたはコンソールの信頼されたホストキー フィールドに貼り付けます。

タイプ：文字列の配列

配列メンバー：最小数は 1 項目です。最大数は 10 項目です。

長さの制限：最小長は 1 です。最大長は 2,048 です。

必須: いいえ

UserSecretId

SFTP ユーザーのプライベートキー、パスワード、またはその両方を含むシークレットの識別子 (AWS Secrets Manager 内) 。識別子はシークレットの Amazon リソースネーム (ARN) である必要があります。

型: 文字列

長さの制限：最小長は 1 です。最大長は 2,048 です。

必須：いいえ

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

SshPublicKey

特定のファイル転送プロトコル対応サーバー（`ServerId` によって識別される）の Transfer Family ユーザーに関連付けられた公開 Secure Shell（SSH）キーに関する情報を提供します。返される情報には、キーがインポートされた日付、パブリックキーの内容、パブリックキー ID が含まれます。ユーザーは、特定のサーバーで自分のユーザー名に関連付けられている複数の SSH パブリックキーを保存できます。

内容

DateImported

パブリックキーが Transfer Family ユーザーに追加された日付を指定します。

型: タイムスタンプ

必須: はい

SshPublicKeyBody

`PublicKeyId` で指定されるとおりに SSH パブリックキーの内容を指定します。

AWS Transfer Family は、RSA、ECDSA、ED25519 キーを受け入れます。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 2,048 です。

必須: はい

SshPublicKeyId

パブリックキーの識別子が含まれる `SshPublicKeyId` パラメータを指定します。

型: 文字列

長さの制限: 固定長は 21 です。

Pattern: `key-[0-9a-f]{17}`

必須：はい

以下の資料も参照してください。

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

Tag

特定のリソースに対して 1 つのキーバリューペアを作成します。タグは、さまざまな目的でリソースを検索してグループ化するために使用できるメタデータです。サーバー、ユーザー、およびロールにタグを適用できます。タグキーには複数の値を設定することもできます。たとえば、経理上の理由でサーバーをグループ化するには、`Group` というタグを作成してそのグループに `Research`および `Accounting` の値を割り当てます。

内容

Key

作成するタグに割り当てた名前。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 128 です。

必須: はい

Value

作成したキー名に割り当てた 1 つ以上の値が含まれます。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 256 です。

必須：はい

以下の資料も参照してください。

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

TagStepDetails

各ステップタイプに独自の `StepDetails` 構造があります。

ワークフローステップの実行中にファイルにタグを付けるために使用されるキーバリューペア。

内容

Name

識別子として使用されるステップの名前。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 30 です。

Pattern: `[\w-]*`

必須: いいえ

SourceFileLocation

ワークフローステップへの入力として使用するファイル (前のステップからの出力、またはワークフロー用に最初にアップロードされたファイル) を指定します。

- 前のファイルを入力として使用するには、`${previous.file}` と入力します。この場合、このワークフローステップは前のワークフローステップの出力ファイルを入力として使用します。これは、デフォルト値です。
- 最初にアップロードされたファイルの場所をこのステップの入力として使用するには、`${original.file}` と入力します。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 256 です。

パターン：`\$\{(\w+.)+\w+\}`

必須: いいえ

Tags

1～10 のキーバリューペアを含む配列。

型: S3Tag オブジェクトの配列

配列メンバー：最小数は 1 項目です。最大数は 10 項目です。

必須：いいえ

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

UserDetails

ワークフローのユーザー名、サーバー ID、セッション ID を指定します。

内容

ServerId

システムによって割り当てられたサーバーインスタンスの一意の識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `s-([0-9a-f]{17})`

必須：はい

UserName

サーバーに関連付けられている Transfer Family を識別する一意の文字列。

型: 文字列

長さの制限: 最小長は 3 です。最大長は 100 です。

パターン: `[\w][\w@.-]{2,99}`

必須：はい

SessionId

ワークフローに対応するセッションに対して、システムによって割り当てられた、一意の識別子。

型: 文字列

長さの制限: 最小長は 3 です。最大長は 32 です。

パターン: `[\w-]*`

必須: いいえ

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

WorkflowDetail

割り当てるワークフローのワークフロー ID とワークフローの実行に使用する実行ロールを指定します。

ファイルのアップロード完了時に実行するワークフローに加えて、部分的なアップロードで実行するワークフローのワークフロー ID (および実行ロール) も `WorkflowDetails` に含めることができます。部分的なアップロードは、ファイルのアップロード中にサーバーセッションが切断されたときに発生します。

内容

ExecutionRole

Transfer で想定できる S3、EFS、および Lambda オペレーションに必要なアクセス許可が含まれ、すべてのワークフローステップで必要なリソースを操作できます。

型: 文字列

長さの制限: 最小長は 20 です。最大長は 2,048 です。

Pattern: `arn:.*role/\S+`

必須：はい

WorkflowId

ワークフローの一意の識別子。

型: 文字列

長さの制限: 固定長は 19 です。

Pattern: `w-([a-z0-9]{17})`

必須：はい

以下の資料も参照してください。

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

WorkflowDetails

`WorkflowDetail` データ型のコンテナ。ワークフローの実行開始をトリガーするアクションによって使用されます。

内容

OnPartialUpload

ファイルの部分的アップロードのみでワークフローを開始するトリガー。部分的アップロードがあるたびに実行されるサーバーにワークフローをアタッチできます。

部分的なアップロードは、セッションが切断されたときにファイルが開いている場合に発生します。

> **Note**
>
> `OnPartialUpload` には、最大 1 つの`WorkflowDetail`オブジェクトを含めることができます。

タイプ : WorkflowDetail オブジェクトの配列

配列メンバー: 最小数は 0 項目です。最大数は 1 項目です。

必須: いいえ

OnUpload

ワークフローを開始するトリガー: ファイルのアップロード後にワークフローの実行が開始されます。

関連するワークフローをサーバーから削除するには、以下の例に示すように、空の `OnUpload` オブジェクトを提供します。

```
aws transfer update-server --server-id s-01234567890abcdef --workflow-details '{"OnUpload":[]}'
```

> **Note**
>
> `OnUpload` には、最大 1 つの`WorkflowDetail`オブジェクトを含めることができます。

タイプ：WorkflowDetail オブジェクトの配列

配列メンバー: 最小数は 0 項目です。最大数は 1 項目です。

必須：いいえ

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

WorkflowStep

ワークフローのベーシックな構築ブロック。

内容

CopyStepDetails

ファイルコピーを実行するステップの詳細。

以下の値で構成されます。

- 記述
- ファイルのコピー先である Amazon S3 の場所。
- 同じ名前の既存のファイルを上書きするかを示すフラグです。デフォルトは FALSE です。

型: CopyStepDetails オブジェクト

必須: いいえ

CustomStepDetails

AWS Lambda 関数を呼び出すステップの詳細。

Lambda 関数名、ターゲット、タイムアウト値 (秒単位) で構成されます。

タイプ：CustomStepDetails オブジェクト

必須: いいえ

DecryptStepDetails

暗号化されたファイルを復号するステップの詳細。

以下の値で構成されます。

- わかりやすい名前
- ソースファイルの復号に使用されるAmazon S3 Amazon Elastic File System (Amazon EFS) の場所。
- ファイルの復号化先の S3 または Amazon EFS の場所。
- 同じ名前の既存のファイルを上書きするかを示すフラグです。デフォルトは FALSE です。
- 使用される暗号化のタイプ。現在、PGP 暗号化のみがサポートされています。

タイプ：DecryptStepDetails オブジェクト

必須: いいえ

DeleteStepDetails

ファイルを削除するステップの詳細。

型: DeleteStepDetails オブジェクト

必須: いいえ

TagStepDetails

1 つ以上のタグを作成するステップの詳細。

複数のタグを指定できます。タグは、キーと値のペアを含みます。

タイプ：TagStepDetails オブジェクト

必須: いいえ

Type

現時点で次のステップタイプはサポートされていません。

- **COPY** - ファイルを別の場所にコピーします。
- **CUSTOM** - AWS Lambda 関数ターゲットを使用してカスタムステップを実行します。
- **DECRYPT** - アップロード前に暗号化されていたファイルを復号化します。
- **DELETE** - ファイルを削除します。
- **TAG** - ファイルにタグを追加します。

型: 文字列

有効な値：COPY | CUSTOM | TAG | DELETE | DECRYPT

必須：いいえ

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

API リクエストを作成する

コンソールを使うだけでなく、AWS Transfer Family APIを使ってプログラムでサーバーを設定・管理することもできます。このセクションでは、AWS Transfer Family のオペレーション、認証のための署名要求、エラー処理について説明します。トランスファーファミリーで利用可能なリージョンとエンドポイントについては、「AWS 全般のリファレンス」の「AWS Transfer Family エンドポイントとクォータ」を参照してください。

> **Note**
>
> また、AWS SDKは Transfer Family でアプリケーションを開発する際にも使用できます。Java、.NET、PHP 用の AWS SDK は、基盤となる Transfer Family API をラップし、プログラミング作業を簡素化します。SDK ライブラリのダウンロードについては、「サンプルコードライブラリ」を参照してください。

トピック

Transfer Family に必要なリクエストヘッダー

このセクションでは、AWS Transfer Family へのすべての POST リクエストで送信しなければならない必須ヘッダーについて説明します。HTTP ヘッダーでは、呼び出すオペレーション、リクエストの日付、リクエストの送信者として認可されていることを示す情報など、リクエストに関する重要な情報を特定します。ヘッダーは大文字と小文字を区別されず、ヘッダーの順序は重要ではありません。

次の例は、「ListServers」操作で使用されるヘッダーを示しています。

```
POST / HTTP/1.1
Host: transfer.us-east-1.amazonaws.com
x-amz-target: TransferService.ListServers
x-amz-date: 20220507T012034Z
Authorization: AWS4-HMAC-SHA256 Credential=AKIDEXAMPLE/20220507/us-east-1/transfer/
aws4_request,
```

```
    SignedHeaders=content-type;host;x-amz-date;x-amz-target,
    Signature=13550350a8681c84c861aac2e5b440161c2b33a3e4f302ac680ca5b686de48de
Content-Type: application/x-amz-json-1.1
Content-Length: 17

{"MaxResults":10}
```

以下は、Transfer Family への POST リクエストに含めなければならないヘッダーです。以下に示す「x-amz」で始まるヘッダーは、AWS 固有のものです。それ以外のヘッダーはすべて、HTTP トランザクションで使用される共通のヘッダーです。

[Header] (ヘッダー)	説明
`Authorization`	認証ヘッダーが必要です。形式は標準の Sigv4 リクエスト署名です。これについては、「AWSAPI リクエストの署名」で説明されています。
`Content-Type`	Transfer Familyへのすべてのリクエストのコンテンツタイプとして `application/x-amz-json-1.1` を使用します。 `Content-Type: application/x-amz-json-1.1`
`Host`	リクエストを送る Transfer Family のエンドポイントを指定するには、ホストヘッダーを使用します。例えば `transfer.us-east-1.amazonaws.com` は、米国東部 (オハイオ) リージョンのエンドポイントを表します。Transfer Family で利用可能なエンドポイントの詳細については、「AWS 全般のリファレンス」の「AWS Transfer Family エンドポイントとクォータ」を参照してください。 `Host: transfer.` *`region`*`.amazonaws.com`
`x-amz-date`	HTTP `Date` ヘッダーまたは AWS `x-amz-date` ヘッダーでタイムスタンプを指定する必要があります。(一部の HTTP クライアントライブラリでは、`Date` ヘッダーを設定することができません)。`x-amz-date` ヘッダーが存在する場合、Transfer Family はリクエスト認証中にいかなる `Date` ヘッダーも無視します。`x-amz-date` 形式は ISO8601 で、YYYYMMDD'T'HHMMSS'Z' 形式でなければなりません。

[Header] (ヘッダー)	説明
	`x-amz-date: YYYYMMDD'T'HHMMSS'Z'`
`x-amz-target`	このヘッダーでは、API のバージョンおよびリクエストするオペレーションを指定します。ターゲットヘッダーの値を作成するには、API のバージョンと API の名前を次のような形式で連結します。 `x-amz-target: TransferService. operationName` 「operationName」の値 (たとえばListServers) は API リストの「ListServers」から見つけることができます。
`x-amz-security-token`	このヘッダーは、リクエストの署名に使用される認証情報が一時的な認証情報またはセッション認証情報である場合に必要となります(詳細は、「IAM ユーザーガイド」の「AWS リソースで一時的な認証情報を使用する」を参照する)。詳細については、Amazon Web Services 全般のリファレンス の「HTTP リクエストに署名を追加する」を参照します。

Transfer Family リクエストの入力と署名

すべてのリクエスト入力は、リクエスト本文の JSON ペイロードの一部として送信する必要があります。すべてのリクエスト· フィールドがオプショナルであるアクション (例えば ListServers) では、リクエスト· ボディに空のJSONオブジェクト ({} など) を提供する必要があります。Transfer Family ペイロードのリクエスト/レスポンスの構造は、たとえば、「DescribeServer」のような既存の API リファレンスに記載されています。

転送ファミリーは、AWS 署名バージョン 4 を使用した認証をサポートしています。詳細については、「AWSAPI リクエストへの署名」を参照してください。

エラーレスポンス

エラーが発生した場合、レスポンスヘッダー情報には、以下の項目が含まれています。

- Content-Type : `application/x-amz-json-1.1`

- 適切な 4xx または 5xx HTTP ステータスコード

エラーレスポンスの本文には、発生したエラーに関する情報が含まれています。次のサンプルエラーは、すべてのエラーレスポンスに共通する、レスポンスエレメントの出力構文を示します。

```
{
    "__type": "String",
    "Message": "String", <!-- Message is lowercase in some instances -->
    "Resource": String,
    "ResourceType": String
    "RetryAfterSeconds": String
}
```

次の表では、前述の構文で表示される JSON エラーレスポンスフィールドを説明します。

__type

Transfer Family API コールの例外の 1 つ。

タイプ: 文字列

「メッセージ」または「メッセージ」

オペレーションエラーコードメッセージの 1 つ 。

> **Note**
>
> messageを使用する例外もあれば、Messageを使用する例外もあります。インターフェイスのコードを確認して、適切なケースを判断できます。あるいは、各オプションをテストして、どれが機能するかを確認することもできます。

タイプ: 文字列

[Resource] (リソース)

エラーが発生したリソース。たとえば、すでに存在するユーザーを作成しようとすると、Resourceは既存のユーザーのユーザー名になります。

タイプ: 文字列

ResourceType

エラーが発生したリソースタイプ。たとえば、すでに存在するユーザーを作成しようとすると、ResourceTypeはUserになります。

タイプ: 文字列

秒後に再試行

コマンドを再試行する前に待つ秒数。

タイプ: 文字列

エラーレスポンスの例

DescribeServerAPI を呼び出し、存在しないサーバーを指定すると、次の JSON 本文が返されます。

```
{
  "__type": "ResourceNotFoundException",
  "Message": "Unknown server",
  "Resource": "s-11112222333344444",
  "ResourceType": "Server"
}
```

API を実行してスロットリングが発生すると、次の JSON 本文が返されます。

```
{
   "__type":"ThrottlingException",
   "RetryAfterSeconds":"1"
}
```

CreateServerAPI を使用していて、Transfer Family サーバーを作成するための十分な権限がない場合は、次の JSON 本文が返されます。

```
{
  "__type": "AccessDeniedException",
  "Message": "You do not have sufficient access to perform this action."
}
```

CreateUserAPI を使用し、すでに存在するユーザーを指定すると、次の JSON 本文が返されます。

```
{
  "__type": "ResourceExistsException",
  "Message": "User already exists",
  "Resource": "Alejandro-Rosalez",
  "ResourceType": "User"
}
```

利用可能なライブラリ

AWS は、コマンドラインツールや Query API ではなく、言語固有の API を使用してアプリケーションを構築することを好むソフトウェア開発者のために、ライブラリ、サンプルコード、チュートリアル、その他のリソースを提供しています。これらのライブラリは、リクエスト認証、リクエストの再試行、エラー処理などの基本的な機能 (API には含まれていない) を提供するため、簡単に使い始めることができます。「AWSを構築するためのツール」を参照してください。

すべての言語のライブラリとサンプルコードについては、「サンプルコードとライブラリ」を参照してください。

共通パラメータ

次のリストには、すべてのアクションが署名バージョン 4 リクエストにクエリ文字列で署名するために使用するパラメータを示します。アクション固有のパラメータは、アクションのトピックに示されています。Signature Version 4 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。

Action

実行するアクション。

型: 文字列

必須: はい

Version

リクエストが想定している API バージョンである、YYYY-MM-DD 形式で表示されます。

型: 文字列

必須: はい

X-Amz-Algorithm

リクエストの署名を作成するのに使用したハッシュアルゴリズム。

条件: HTTP 認証ヘッダーではなくクエリ文字列に認証情報を含める場合は、このパラメータを指定します。

型: 文字列

有効な値: `AWS4-HMAC-SHA256`

必須: 条件による

X-Amz-Credential

認証情報スコープの値で、アクセスキー、日付、対象とするリージョン、リクエストしているサービス、および終了文字列 ("aws4_request") を含む文字列です。値は次の形式で表現されます。[access_key/YYYYYYYYMMDD/リージョン/サービス/aws4_request]

詳細については、「IAM ユーザーガイド」の「署名付きAWS API リクエストの作成」を参照してください。

条件: HTTP 認証ヘッダーではなくクエリ文字列に認証情報を含める場合は、このパラメータを指定します。

型: 文字列

必須: 条件による

X-Amz-Date

署名を作成するときに使用する日付です。形式は ISO 8601 基本形式の YYYYMMDD'T'HHMMSS'Z' でなければなりません。例えば、日付 `20120325T120000Z` は、有効な X-Amz-Date の値です。

条件: X-Amz-Date はすべてのリクエストに対してオプションです。署名リクエストで使用する日付よりも優先される日付として使用できます。ISO 8601 ベーシック形式で日付ヘッダーが指定されている場合、X-Amz-Date は必要ありません。X-Amz-Date を使用すると、常に Date ヘッダーの値よりも優先されます。詳細については、「IAM ユーザーガイド」の「AWS API リクエスト署名の要素」を参照してください。

タイプ: 文字列

必須: 条件による

X-Amz-Security-Token

AWS Security Token Service (AWS STS) へのを呼び出しで取得された一時的なセキュリティトークン。AWS STS の一時的なセキュリティ認証情報をサポートするサービスのリストについては、「IAM ユーザーガイド」の「IAM と連携するAWS のサービス」を参照してください。

条件:AWS STS の一時的なセキュリティ認証情報を使用する場合、セキュリティトークンを含める必要があります。

タイプ: 文字列

必須: 条件による

X-Amz-Signature

署名する文字列と派生署名キーから計算された 16 進符号化署名を指定します。

条件: HTTP 認証ヘッダーではなくクエリ文字列に認証情報を含める場合は、このパラメータを指定します。

型: 文字列

必須: 条件による

X-Amz-SignedHeaders

正規リクエストの一部として含まれていたすべての HTTP ヘッダーを指定します。署名付きヘッダーの指定に関する詳細については、「IAM ユーザーガイド」の「署名付き AWS API リクエストの作成」を参照してください。

条件: HTTP 認証ヘッダーではなくクエリ文字列に認証情報を含める場合は、このパラメータを指定します。

型: 文字列

必須: 条件による

共通エラー

このセクションでは、AWS のすべてのサービスの API アクションに共通の工ラーを一覧表示しています。このサービスの API アクションに固有のエラーについては、その API アクションのトピックを参照してください。

AccessDeniedException

このアクションを実行する十分なアクセス権限がありません。

HTTP ステータスコード: 400

IncompleteSignature

リクエストの署名が AWS 基準に適合しません。

HTTP ステータスコード: 400

InternalFailure

リクエストの処理が、不明なエラー、例外、または障害により実行できませんでした。

HTTP ステータスコード: 500

InvalidAction

リクエストされたアクション、またはオペレーションは無効です。アクションが正しく入力されていることを確認します。

HTTP ステータスコード: 400

InvalidClientTokenId

指定された x.509 証明書、または AWS アクセスキー ID が見つかりません。

HTTP ステータスコード: 403

NotAuthorized

このアクションを実行するにはアクセス許可が必要です。

HTTP ステータスコード: 400

OptInRequired

サービスを利用するためには、AWS アクセスキー ID を取得する必要があります。

HTTP ステータスコード: 403

RequestExpired

リクエストの日付スタンプの 15 分以上後またはリクエストの有効期限 (署名付き URL の場合など) の 15 分以上後に、リクエストが到着しました。または、リクエストの日付スタンプが現在より 15 分以上先です。

HTTP ステータスコード: 400

ServiceUnavailable

リクエストは、サーバーの一時的障害のために実行に失敗しました。

HTTP ステータスコード: 503

ThrottlingException

リクエストは、制限が必要なために実行が拒否されました。

HTTP ステータスコード: 400

ValidationError

入力が、AWS サービスで指定された制約を満たしていません。

HTTP ステータスコード: 400

のドキュメント履歴 AWS Transfer Family

次の表は、 のこのリリースのドキュメントを示しています AWS Transfer Family。

- API バージョン: transfer-2018-11-05
- ドキュメントの最終更新日： 2024 年 4 月 23 日

変更	説明	日付
SFTP コネクタがリモートファイルとディレクトリを一覧表示する機能	Transfer Family では、お客様が SFTP コネクタを使用してリモート SFTP サーバーに保存されているファイルを一覧表示する機能が追加されました。詳細については、「リモートディレクトリの内容を一覧表示する」を参照してください。	2024 年 4 月 23 日
AS2 メッセージ交換で取引相手の自己署名 TLS 証明書を使用する機能	AWS Transfer Family は、HTTPS 経由で サーバーに Applicability Statement 2 (AS2) メッセージを送信するために、取引相手のパブリックで自己署名された TLS 証明書をインポートして使用するオプションを追加しました。	2024 年 4 月 12 日
SFTP コネクタのセキュリティポリシーの追加	AWS Transfer Family は、SFTP コネクタで使用するセキュリティポリシーを追加しました。詳細については、「AWS Transfer Family SFTP コネクタのセキュリティポリシー」を参照してください。	2024 年 4 月 5 日

変更	説明	日付
Amazon との統合 EventBridge	AWS Transfer Family は、すべてのファイル転送オペレーション EventBridge のイベントを Amazon に自動的に発行するようになりました。詳細については、「を使用した Transfer Family イベントの管理 Amazon EventBridge」を参照してください。	2024 年 2 月 8 日
新しいセキュリティポリシーの追加	AWS Transfer Family は、新しい FIPS および非 FIPS セキュリティポリシーを追加しました。また、サーバーに割り当てられたデフォルトのセキュリティポリシーは常に最新のセキュリティポリシーです。詳細については、「AWS Transfer Family サーバーのセキュリティポリシー」を参照してください。	2024 年 2 月 5 日
SFTP コネクタと AS2 の静的 IP アドレスのサポート	Transfer Family は、SFTP コネクタと AS2 の静的 IP アドレスを提供するようになりました。これにより、IP 許可リストのコントロールによって保護されているリモート SFTP サーバーとの接続が可能になります。AS2 では、AS2 サーバーからの非同期 MDN 応答用の静的 IP アドレスを導入します。	2024 年 1 月 16 日

変更	説明	日付
ユーザーガイドは、最新バージョンの とより密接に連携するように再編成されましたAWS Transfer Family。	Transfer Family は、ガイドの作成以降に複数の機能を追加し、ガイドの再構築を必要としています。	2024 年 1 月 3 日
論理ディレクトリマッピングの機能強化 Amazon S3 リストのパフォーマンス最適化	Transfer Family は、最大 2.1 MB の論理ディレクトリマッピングをサポートするようになりました。また、ユーザーマッピングが ファイルに対するものであるかどうかを宣言できるようになりました。詳細については、「論理ディレクトリを使用する際のルール」を参照してください。 ストレージに Amazon S3 を使用するサーバーを作成または更新するときに、S3 ディレクトリ (またはフォルダ) の一覧表示のパフォーマンスを最適化できるようになりました。詳細については、「SFTP、FTPS、または FTP サーバーエンドポイントの設定」を参照してください。	2023 年 11 月 17 日
Virtual Private Cloud (VPC) エンドポイントを使用する SFTP サーバーの代替ポート	VPC エンドポイントを持つ SFTP Transfer Family サーバーで、標準以外の代替ポートを有効にできるようになりました。詳細については、「Virtual Private Cloud でサーバーを作成する」を参照してください。	2023 年 11 月 17 日

変更	説明	日付
SFTP コネクタをサポート	SFTP コネクタは、クラウドとオンプレミスの両方でリモートサーバーと通信 AWS Transfer Family するための機能を拡張します。詳細については、「SFTP コネクタを使用してファイルを送受信します。」を参照してください。	2023 年 7 月 25 日
AS2 基本認証をサポート	Transfer Family は、Applicability Statement 2 (AS2) プロトコルを使用するサーバー向けに基本認証の使用をサポートするようになりました。詳細については、「AS2 コネクタの基本認証」を参照してください。	2023 年 6 月 30 日
構造化された JSON ロギングをサポート	Transfer Family は、構造化された JSON ログの Amazon への配信 CloudWatch、ログストリームのカスタムロググループへのグループ化、プロトコル間の一般的なログクエリの実行をサポートするようになりました。詳細については、「の Amazon CloudWatch ログ記録 AWS Transfer Family」を参照してください。	2023 年 6 月 24 日

変更	説明	日付
複数の認証方法をサポート	Transfer Family は、パスワード、公開鍵と秘密key pair、またはその両方による認証をサポートしています。これはSFTP プロトコルを使用するサーバーとカスタム ID プロバイダーで使用できます。詳細については、「SFTP 対応サーバーの作成」を参照してください。	2023 年 5 月 17 日
PGP (プリティグッドプライバシー) の復号化をサポートし、ワークフローでファミリーのプロセスを転送します。	Transfer Family には完璧なプライバシー (PGP) 復号化のサポートが組み込まれています。SFTP、FTPS、またはFTP経由でAmazon Simple Storage Service (Amazon S3) またはAmazon Elastic File System (Amazon EFS) にアップロードされたファイルでPGP復号化を使用できます。詳細については、「PGP キーの生成と管理」および「PGP 復号化をワークフローで使用してください。」を参照してください。	2022 年 12 月 21 日
Transfer Family ServerによるApplicability Statement 2 (AS2) ファイル転送プロトコルのフルマネージドサポート	AS2 プロトコルを使用して、AWS 環境内外の取引相手との間で情報を送受信するサーバーを作成できます。詳細については、「AS2 の設定」を参照してください。	2022 年 7 月 25 日

変更	説明	日付
サーバー作成時のディスプレイバナーをサポート	サーバーを作成するときに、カスタマイズしたメッセージを追加できます。認証前メッセージ (全プロトコル) と認証後メッセージ (FTP および FTPS サーバー用) を表示できます。詳細については「SFTP 対応サーバーの作成」、「FTPS 対応サーバーを作成する」または「FTP 対応サーバーの作成」を参照してください。	2022 年 2 月 17 日
ID プロバイダー AWS Lambda としての のサポート	Transfer Family サーバー AWS Lambda で を使用してカスタム ID プロバイダーに接続できるようになりました。以前は、カスタム ID プロバイダーを統合するには Amazon API Gateway URL を指定する必要がありました。詳細については、「AWS Lambda を使用して ID プロバイダーを統合する」を参照してください。	2021 年 11 月 16 日
マネージドファイル転送ワークフローの サポート	マネージドファイル転送ワークフローは、現在手動で実行している共通タスクについてアップロード後の処理の抽象化を提供します。詳細については、「AWS Transfer Family マネージドワークフロー」を参照してください。	2021 年 9 月 2 日

変更	説明	日付
のサポート AWS Directory Service for Microsoft Active Directory	サービスマネージド ID プロバイダーとカスタム ID プロバイダーに加えて、 AWS Directory Service for Microsoft Active Directory を使用して、認証と承認のためのユーザーアクセスを管理できるようになりました。詳細については、「AWS Directory Service ID プロバイダーの使用」を参照してください。	2021 年 5 月 24 日
新規 AWS リージョン	AWS Transfer Family がアフリカ (ケープタウン) リージョンで利用可能になりました。トランスファーファミリーのエンドポイントについては、AWS 全般のリファレンスの「AWS Transfer Family エンドポイントとクォータ」を参照してください。	2021 年 2 月 24 日
新規 AWS リージョン	AWS Transfer Family が、アジアパシフィック (香港) および中東 (バーレーン) リージョンで利用可能になりました。トランスファーファミリーのエンドポイントについては、AWS 全般のリファレンスの「AWS Transfer Family エンドポイントとクォータ」を参照してください。	2021 年 2 月 17 日

変更	説明	日付
データストアとしての Amazon EFS のサポート	Transfer Family で Amazon Elastic File System (Amazon EFS) との間のファイル転送をサポートするようになりました。Amazon EFS は、シンプルでスケーラブル、かつ伸縮自在なフルマネージド型の NFS ファイルシステムです。詳細については、「Amazon EFS ファイルシステムを設定する」を参照してください。	2021 年 1 月 6 日
のサポート AWS WAF	Transfer Family は AWS WAF、ウェブアプリケーションと API オペレーションを攻撃から保護するウェブアプリケーションファイアウォールである をサポートするようになりました。詳細については、「ウェブアプリケーションファイアウォールを追加する」を参照してください。	2020 年 11 月 24 日
Virtual Private Cloud (VPC) での複数セキュリティグループのサポート	VPC で 1 台のサーバーに複数のセキュリティグループを接続できるようになりました。詳細については、「Virtual Private Cloud でサーバーを作成する」を参照してください。	2020 年 10 月 15 日

変更	説明	日付
新規 AWS リージョン	Transfer Family が AWS GovCloud (US) リージョンで利用可能になりました。AWS GovCloud (US) リージョンの Transfer Family エンドポイントの詳細については、「」のAWS Transfer Family「エンドポイントとクォータ」を参照してくださいAWS 全般のリファレンス。 AWS GovCloud (US) リージョンで Transfer Family を使用する方法については、AWS Transfer FamilyAWS GovCloud (US)「ユーザーガイド」の「」を参照してください。	2020 年 9 月 30 日
サポートされる暗号化アルゴリズムでセキュリティポリシーをサーバーにアタッチできます。	サポートされる一連の暗号化アルゴリズムでセキュリティポリシーをサーバーに接続できるようになりました。詳細については、「AWS Transfer Family サーバーのセキュリティポリシー」を参照してください。	2020 年 8 月 12 日

変更	説明	日付
連邦情報処理規格 (FIPS) エンドポイントのサポート	FIPS 対応エンドポイントが北米の AWS リージョンで利用可能になりました。利用可能なリージョンについては、「AWS 全般のリファレンス」の「AWS Transfer Family エンドポイントとクォータ」を参照してください。SFTP 対応サーバーエンドポイントの FIPS を有効にするには、「SFTP 対応サーバーの作成」を参照してください。FTPS 対応サーバーエンドポイントの FIPS を有効にするには、「FTPS 対応サーバーを作成する」を参照してください。FTP 対応サーバーエンドポイントの FIPS を有効にするには、「FTP 対応サーバーの作成」を参照してください。	2020 年 8 月 12 日
ユーザー名の文字長増加および許容文字数の追加	ユーザー名にはアットマーク (@) とピリオド (.) を使用できるようになり、最大長は 100 文字になりました。ユーザーを追加するには、「サーバーエンドポイントのユーザーの管理」を参照してください。	2020 年 8 月 12 日

変更	説明	日付
Amazon CloudWatch ログ記録 AWS Identity and Access Management (IAM) ロールの自動作成のサポート	Transfer Family は、エンドユーザーのアクティビティを表示するための CloudWatch ログ記録 IAM ロールの自動作成をサポートするようになりました。詳細については「SFTP 対応サーバーの作成」、「FTPS 対応サーバーを作成する」または「FTP 対応サーバーの作成」を参照してください。	2020 年 7 月 30 日
AWS Transfer Family は、承認の要素としてソース IP をサポートするようになりました。	Transfer Family では、エンドユーザーの送信元 IP アドレスを認証の要素として使用できるようになり、Secure File Transfer Protocol (SFTP)、File Transfer Protocol over SSL (FTPS)、または File Transfer Protocol (FTP) を経由してアクセスを認証する際に、さらなるセキュリティレイヤーの適用が可能になりました。詳細については、「カスタム IDプロバイダーの使用」を参照してください。	2020 年 6 月 9 日

変更	説明	日付
AWS Transfer for SFTP が AWS Transfer Family になり、FTP と FTPS のサポートが追加されました。	ユーザーのファイル転送に File Transfer Protocol Secure (FTPS) と File Transfer Protocol (FTP) という 2 つの追加ファイル転送プロトコルを使用できるようになりました。ユーザーは、既存の Secure File Transfer Protocol (SFTP) サポートに加えて、FTP over SSL (FTPS) およびプレーンテキスト FTP ベースのワークフローを で移動、実行 AWS、保護、統合できます。	2020 年 4 月 23 日
Virtual Private Cloud (VPC) セキュリティグループと Elastic IP アドレスのサポート	セキュリティグループを使って、着信 IP アドレスの許可リストを作成することができるようになり、サーバーにさらなるセキュリティレイヤーを提供できるようになりました。Elastic IP アドレスをサーバーのエンドポイントに関連付けることもできます。そうすることで、ファイアウォールの背後にいるユーザーがそのエンドポイントにアクセスできるようになります。詳細については、「Virtual Private Cloud でサーバーを作成する」を参照してください。	2020 年 1 月 10 日

変更	説明	日付
VPC 使用のサポート	VPC でサーバーを作成できるようになりました。パブリックなインターネットを経由することなく、サーバーを使用してクライアント経由で Amazon S3 バケットとの間でデータを転送できます。詳細については、「Virtual Private Cloud でサーバーを作成する」を参照してください。	2019 年 3 月 27 日
の最初のバージョンが AWS Transfer Family リリースされました。	この初回リリースでは、セットアップ手順を指示し、使用を開始する方法について説明するとともに、クライアント設定、ユーザー設定、およびモニタリング活動に関する情報を提供しています。	2018 年 11 月 25 日

AWS 用語集

AWS の最新の用語については、「AWS の用語集リファレンス」の「AWS 用語集」を参照してください。

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。