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 1
の AWS Transfer Family 仕組み 3
Transfer Family に関連するブログ記事 4
前提条件 7
リージョン、エンドポイント、およびクォータ 7
にサインアップする AWS 7
ストレージを設定 8
Amazon S3 バケットを設定する 9
Amazon EFS ファイルシステムを設定する 13
IAM ロールとポリシーを作成する 16
ユーザーロールの作成 17
セッションポリシーの仕組み 21
読み書きアクセスポリシーの例 24
Transfer Family チュートリアル 28
サーバーエンドポイントの使用を開始する 28
前提条件 29
コンソールにサインインする 30
SFTP対応サーバーを作成する 30
サービスマネージドユーザーを追加する 31
クライアントを使用してファイルを転送する 33
復号ワークフローを作成する 35
ステップ 1 : 実行ロールを設定する 36
ステップ 2: マネージドワークフローを作成する 37
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 38
ステップ 4: PGPキーペアを作成する 40
ステップ 5: PGPプライベートキーを に保存する AWS Secrets Manager 41
ステップ 6: ファイルを暗号化する 42
ステップ 7: ワークフローを実行して結果を表示する 42
SFTP コネクタの作成と使用 43
ステップ 1: 必要なサポートリソースを作成する 45
ステップ 2: SFTP コネクタを作成してテストする 49
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 53
リモートサーバーとして使用する Transfer Family SFTPサーバーを作成する手順 57
カスタム ID プロバイダーを使用する 60

前提条件 60
ステップ 1: CloudFormation スタックを作成する 61
ステップ 2: サーバーの API Gateway メソッド設定を確認する 62
ステップ 3: Transfer Family サーバーの詳細を表示する 63
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 64
ステップ 5: SFTP接続とファイル転送をテストする 65
ステップ 6: バケットへのアクセスを制限する 65
Amazon を使用している場合は Lambda を更新する EFS 67
AS2 設定をセットアップする 68
ステップ 1: の証明書を作成する AS2 70
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 74
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 77
ステップ 4: 自分と取引相手のプロフィールを作成する 78
ステップ 5: 自分とパートナーとの間で契約を作成する 79
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 80
ステップ 7: Transfer Family を使用してファイル交換AS2をテストする 81
SFTP、、 の Transfer Family FTPS FTP 84
ID プロバイダーオプション 84
AWS Transfer Family エンドポイントタイプマトリックス 86
Transfer Family サーバーエンドポイントを設定する 90
SFTP対応サーバーを作成する 92
FTPS対応サーバーを作成する 104
FTP対応サーバーを作成する 116
でサーバーを作成する VPC 127
カスタムホスト名の使用 149
サーバーエンドポイント経由でファイルを転送する 152
使用可能なSFTP/FTPS/FTPコマンド 155
Amazon VPCエンドポイントを検索する 156
`setstat`エラーを回避する 158
Open を使用するSSH 34
Win を使用するSCP 160
Cyberduck を使用する 33
を使用する FileZilla 163
Perl クライアントを使用する 165
アップロード後の処理 165
ユーザーの管理 166

サービスマネージドユーザー 168
ディレクトリサービスのユーザー 178
カスタム ID プロバイダーユーザー 195
論理ディレクトリを使用する 224
論理ディレクトリを使用する際のルール 225
論理ディレクトリと の実装 chroot 227
論理ディレクトリの構成例 229
Amazon の論理ディレクトリを設定する EFS 230
カスタム AWS Lambda レスポンス 231
SFTP コネクタ 232
SFTP コネクタの設定 232
SFTP コネクタを作成する 233
SFTP コネクタアルゴリズム 241
SFTP コネクタで使用するシークレットを保存する 242
SFTP コネクタプライベートキーを生成してフォーマットする 243
SFTP コネクタをテストする 246
SFTP コネクタを使用してファイルを転送する 248
SFTP コネクタの管理 249
SFTP コネクタの更新 250
SFTP コネクタの詳細を表示する 250
SFTP コネクタのクォータ 252
の転送ファミリー AS2 254
AS2 ユースケース 255
設定 AS2 259
Transfer Family コンソールを使用してAS2サーバーを作成する 260
テンプレートを使用してAS2サーバーを作成する 263
AS2 設定 266
AS2 機能と機能 272
AS2 コネクタの設定 274
AS2 コネクタを作成する 274
AS2 コネクタアルゴリズム 277
AS2 コネクタの基本認証 278
AS2 コネクタの基本認証を有効にする 280
コネクタの詳細の表示 284
AS2 パートナーを管理する 285
AS2 証明書をインポートする 285

AS2 証明書のローテーション 287
AS2 プロファイルの作成 288
AS2 アグリーメントの作成 289
AS2 メッセージを転送する 290
AS2 メッセージの送信 291
AS2 メッセージを受信する 292
HTTPS に を設定する AS2 293
AS2 コネクタを使用してファイルを転送する 297
ファイル名と場所 298
ステータスコード 300
サンプルJSONファイル 301
AS2 のモニタリング 303
AS2 ステータスコード 305
AS2 エラーコード 305
ファイル処理ワークフローの管理 320
ワークフローを作成する 322
ワークフローの設定と実行 323
ワークフロー詳細の表示 326
事前定義されたステップを使用する 328
ファイルをコピーする 329
ファイルを復号化します。 334
タグファイル 340
ファイルを削除する 341
ワークフローの名前付き変数 342
タグ付けと削除のワークフローの例 342
カスタムのファイル処理ステップを使用してください。 347
複数の Lambda 関数を続けて使用する 349
カスタム処理後のファイルへのアクセス 349
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 350
カスタムワークフローステップの Lambda 関数の例 351
IAM カスタムステップのアクセス許可 352
IAM ワークフローの ポリシー 353
ワークフローの信頼関係 355
実行ロールの例:復号化、コピー、タグ付け 355
実行ロールの例:関数の実行と削除 357
ワークフローの例外処理 358

ワークフロー実行のモニタリング 359
CloudWatch ワークフローのログ記録 359
CloudWatch ワークフローのメトリクス 361
テンプレートからワークフローを作成する 362
Transfer Family サーバーからワークフローを削除する 365
制限事項と限度値 367
サーバーの管理 369
サーバーのリストを表示する 369
サーバーを削除する 369
SFTP サーバーの詳細を表示する 371
AS2 サーバーの詳細を表示する 372
サーバーの詳細を編集する 374
ファイル転送プロトコルを編集する 377
カスタム ID プロバイダーパラメーターを編集 380
サーバーエンドポイントを編集する 382
ロギングを編集 384
セキュリティポリシーを編集する 384
管理ワークフローを変更 386
SFTP サーバーのディスプレイバナーを変更します 387
サーバーのオンラインとオフラインを切り替える 388
サーバーのホストキーを管理 389
サーバーホストキーを追加する 390
サーバーのホストキーを削除する 392
サーバーのホストキーをローテーションする 392
サーバーホストキーの追加情報 394
コンソール内での使用状況のモニタリング 394
アクセスコントロールの管理 398
S3 バケットアクセスポリシーの作成 399
セッションポリシーの作成 400
ユーザーによる S3 mkdir バケットでの実行を無効にする 403
ログ記録 405
CloudTrail ログ記録 405
CloudTrail ログ記録の有効化 407
サーバーを作成するためのログエントリの例 407
CloudWatch ログ記録 409
サーバーのログ記録の作成 410

ワークフローのログ記録の管理 418
のロールの設定 CloudWatch 421
Transfer Family ログストリームの表示 423
Amazon CloudWatch アラームの作成 427
S3 アクセスログへの S3 API呼び出しのログ記録 427
混乱する代理問題の防止 428
CloudWatch Transfer Family の ログ構造 430
CloudWatch ログエントリの例 435
CloudWatch メトリクスの使用 439
ユーザー通知 442
を使用したイベントの管理 EventBridge 443
Transfer Family イベント 444
SFTP、FTPS、および FTPサーバーイベント 444
SFTP コネクタイベント 445
A2S イベント 445
Transfer Family イベントの送信 446
イベントパターンの作成 447
イベントの Transfer Family イベントパターンのテスト 448
アクセス許可 448
追加リソース 449
イベントの詳細リファレンス 449
サーバーイベント 450
コネクタイベント 454
AS2 イベント 459
セキュリティ 465
セキュリティポリシーの使用 466
暗号アルゴリズム 468
TransferSecurityPolicy2024 年 1 月 476
TransferSecurityPolicy2023 年 5 月 477
TransferSecurityPolicy2022 年 3 月 478
TransferSecurityPolicy2020 年 6 月 479
TransferSecurityPolicy2018 年 11 月 480
TransferSecurityPolicy-FIPS-2024 年 1 月 481
TransferSecurityPolicy-FIPS-2023 年 5 月 482
TransferSecurityPolicy-FIPS-2020 年 6 月 483
ポスト・クアンタム・セキュリティ・ポリシー 485

ポスト・クォンタム・セキュリティ・ポリシー 490
でのポスト量子ハイブリッドキー交換について SSH 491
使用方法 491
テスト方法 493
データ保護 496
データ暗号化 497
キーの管理 498
ID およびアクセス管理 515
対象者 515
アイデンティティを使用した認証 516
ポリシーを使用したアクセスの管理 519
の AWS Transfer Family 仕組み IAM 522
アイデンティティベースポリシーの例 527
タグベースのポリシーの例 530
ID とアクセスのトラブルシューティング 533
コンプライアンス検証 535
耐障害性 536
インフラストラクチャセキュリティ 537
ウェブアプリケーションファイアウォール 538
サービス間の混乱した代理の防止 539
Transfer Family ユーザーロール 541
Transfer Family ワークフローロール 543
トランスファーファミリーロギング/呼び出しロール 544
AWS マネージドポリシー 545
AWSTransferConsoleFullAccess 546
AWSTransferFullAccess 548
AWSTransferLoggingAccess 549
AWSTransferReadOnlyAccess 550
ポリシーの更新 551
Transfer Family のトラブルシューティング 553
サービス管理ユーザーのトラブルシューティング 553
Amazon EFSサービスマネージドユーザーのトラブルシューティング 554
公開鍵本文が長すぎる場合のトラブルシューティング 554
トラブルシューティングでSSHパブリックキーを追加できませんでした 555
Amazon API Gateway の問題のトラブルシューティング 555
認証失敗の回数が多すぎる 555

接続が終了する 557
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 557
認証の問題のトラブルシューティング 558
認証の失敗 —SSH/SFTP 558
マネージド AD のレルム不一致問題 559
その他の認証に関する問題 559
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 560
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 560
ワークフローコピーエラーのトラブルシューティング 562
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 562
Amazon EFSの問題のトラブルシューティング 565
欠落しているPOSIXプロファイルのトラブルシューティング 565
Amazon で論理ディレクトリをトラブルシューティングする EFS 566
ID プロバイダーのテストのトラブルシューティング 566
SFTP コネクタに信頼されたホストキーを追加するトラブルシューティング 567
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 568
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 568
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 568
ResourceNotFound 例外のトラブルシューティング 569
SFTP コネクタの問題のトラブルシューティング 569
キーネゴシエーションが失敗する 570
その他のSFTPコネクタの問題 570
AS2 問題のトラブルシューティング 571
API リファレンス 572
ようこそ 572
アクション 575
CreateAccess 578
CreateAgreement 585
CreateConnector 591
CreateProfile 598
CreateServer 603
CreateUser 616
CreateWorkflow 625
DeleteAccess 634
DeleteAgreement 637
DeleteCertificate 640

DeleteConnector 642
DeleteHostKey 644
DeleteProfile 647
DeleteServer 649
DeleteSshPublicKey 652
DeleteUser 655
DeleteWorkflow 658
DescribeAccess 660
DescribeAgreement 664
DescribeCertificate 667
DescribeConnector 670
DescribeExecution 673
DescribeHostKey 678
DescribeProfile 681
DescribeSecurityPolicy 684
DescribeServer 688
DescribeUser 693
DescribeWorkflow 698
ImportCertificate 703
ImportHostKey 708
ImportSshPublicKey 712
ListAccesses 717
ListAgreements 721
ListCertificates 725
ListConnectors 729
ListExecutions 732
ListHostKeys 737
ListProfiles 741
ListSecurityPolicies 745
ListServers 749
ListTagsForResource 753
ListUsers 758
ListWorkflows 763
SendWorkflowStepState 766
StartFileTransfer 770
StartServer 776

StopServer ... 779
TagResource ... 782
TestConnection ... 785
TestIdentityProvider ... 789
UntagResource ... 796
UpdateAccess ... 799
UpdateAgreement ... 806
UpdateCertificate ... 812
UpdateConnector ... 816
UpdateHostKey ... 821
UpdateProfile ... 825
UpdateServer ... 828
UpdateUser ... 841
データ型 ... 848
As2ConnectorConfig ... 851
CopyStepDetails ... 855
CustomStepDetails ... 858
DecryptStepDetails ... 860
DeleteStepDetails ... 863
DescribedAccess ... 865
DescribedAgreement ... 869
DescribedCertificate ... 873
DescribedConnector ... 877
DescribedExecution ... 881
DescribedHostKey ... 884
DescribedProfile ... 887
DescribedSecurityPolicy ... 890
DescribedServer ... 892
DescribedUser ... 901
DescribedWorkflow ... 905
EfsFileLocation ... 907
EndpointDetails ... 909
ExecutionError ... 913
ExecutionResults ... 915
ExecutionStepResult ... 916
FileLocation ... 918

HomeDirectoryMapEntry 919
IdentityProviderDetails 921
InputFileLocation 924
ListedAccess 925
ListedAgreement 928
ListedCertificate 931
ListedConnector 934
ListedExecution 936
ListedHostKey 938
ListedProfile 940
ListedServer 942
ListedUser 945
ListedWorkflow 948
LoggingConfiguration 950
PosixProfile 952
ProtocolDetails 954
S3FileLocation 958
S3InputFileLocation 960
S3StorageOptions 962
S3Tag 963
ServiceMetadata 964
SftpConnectorConfig 965
SshPublicKey 967
Tag 969
TagStepDetails 970
UserDetails 972
WorkflowDetail 974
WorkflowDetails 976
WorkflowStep 978
API リクエストを作成する 980
Transfer Family に必要なリクエストヘッダー 980
Transfer Family リクエストの入力と署名 982
エラーレスポンス 983
利用可能なライブラリ 985
共通パラメータ 985
共通エラー 988

ドキュメント履歴 990
AWS 用語集 1003
..... miv

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、、FTPS、、および Amazon S3 または Amazon AS2FTPへの直接転送と Amazon からのファイル転送に対するフルマネージドサポート AWS Transfer Family を提供しますEFS。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の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はありません。

詳細については、「 の使用開始 AWS」を参照してください。また、Amazon Web Services を使用してクラウドアプリケーションの構築を開始します。

AWS Transfer Family は、以下の AWS ストレージサービスとの間でデータの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 の詳細についてはEFS、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 Shell (SSH) File Transfer Protocol (SFTP): バージョン 3
- ファイル転送プロトコルセキュア (FTPS)
- ファイル転送プロトコル (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 がデータチャネルの確立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。

Amazon で Transfer Family を使用する際の一般的なユースケースを次に示しますEFS。

- 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

で Transfer Family を使用する際の一般的なユースケースを次に示しますAS2。

- 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
- Business-to-business (B2B) トランザクション
- エンタープライズリソースプランニング (ERP) 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、 の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を使用すると、ネイティブ AWS のサービス を使用して、処理、分析、レポート、監査、アーカイブの各機能を実行できます。
- Amazon をデータストアEFSとして使用すると、 AWS クラウド サービスやオンプレミスリソースで使用できるフルマネージド型のエラスティックファイルシステムが得られます。Amazon EFS は、アプリケーションを中断することなく、オンデマンドでペタバイトに拡張できるように構築されており、ファイルの追加や削除時に自動的に増減します。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、プロトコルが有効なさまざまなサーバーのセットアップ方法に関する詳細な手順、さまざまなタイプの ID プロバイダーの使用方法、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。

- の AWS Transfer Family 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

の AWS Transfer Family 仕組み

AWS Transfer Family は、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、以下のプロトコルを使用して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 Shell (SSH) File Transfer Protocol (SFTP): バージョン 3
- ファイル転送プロトコルセキュア (FTPS)
- ファイル転送プロトコル (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、接続リクエストと転送リクエストの自動スケーリング、冗長フリートによってサポートされています。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ための構築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、 を使用してアップロードされたファイルの処理を簡単にセットアップ、実行、自動化、モニタリングできる、フルマネージドのサーバーレスファイル転送ワークフローサービスです AWS Transfer Family。お客様は MFTW を使用して、Transfer Family を使用して転送されるcompressing/decompressing, and encrypting/decrypting データのコピー、タグ付け、スキャン、フィルタリングなど、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ています。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 — Macintosh および Linux コマンドラインユーティリティ。
- WinSCP – Windows のみのグラフィカルクライアント。
- 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 グラフィカルクライアント。

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。表は時系列の逆順にソートされており、最新の投稿が表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
Transfer Family が安全で準拠した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
を使用したマルウェアの脅威の検出 AWS Transfer Family	2023 年 7 月 20 日

ブログ投稿のタイトルとリンク	日付
を使用したSAPワークロードの拡張 AWS Transfer Family	2023 年 7 月 13 日
PGPおよび を使用してファイルを暗号化および復号化する AWS Transfer Family	2023 年 6 月 21 日
Azure Active Directory と AWS Transfer Family を使用した 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
A AWS Transfer Family と でユーザーセルフサービスキー管理を有効にします AWS Lambda。	2021 年 12 月 17 日
AWS Transfer Family と Amazon S3 を使用してデータアクセスコントロールを強化する	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 によるセキュリティ保護	2021 年 5 月 5 日
AWS Web Application Firewall と Amazon API Gateway AWS Transfer Family によるセキュリティ保護	2021 年 1 月 15 日

ブログ投稿のタイトルとリンク	日付
AWS Transfer Family Amazon Elastic File System のサポート	2021 年 1 月 7 日
AWS Transfer Family を使用するため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EFSの AWS Lambda での Amazon の使用	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
chroot ディレクトリと論理ディレクトリを使用して構造を簡素化する AWS SFTP	2019 年 9 月 26 日
で Okta を ID プロバイダーとして使用する 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 サービスの使用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 はユーザーのファイル転送リクエストを処理できるように、バケットにアクセスするIAMロールを引き受けることができます。

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 ロールとポリシーを作成する

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

Amazon Web Services (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 既にアカウントをお持ちの場合は、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/サインアップ 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、管理ユーザーに管理アクセスを割り当て、ルートユーザーアクセスが必要なタスクを実行する場合にのみ、ルートユーザーを使用してください。

Transfer Family の使用コストの見積り AWS 料金見積りツール を取得するために と を使用する料金の詳細については、AWS Transfer Family 「料金」を参照してください。

AWS リージョンの可用性の詳細については、「」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。

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EFSとして Amazon S3 または Amazon を使用できます。

目次

- Amazon S3 バケットを設定する
  - Amazon S3 アクセスポイント
  - Amazon S3 HeadObject の動作
    - ファイルの書き込みとリストのみの権限を付与します。
    - 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 EFS ファイルシステムを設定する
  - Amazon EFS ファイルの所有権
  - Transfer Family の Amazon EFS ユーザーを設定する
    - Amazon で Transfer Family ユーザーを設定する EFS

    - Amazon EFSルートユーザーを作成する
- 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 はユーザーの転送リクエストを処理するために Amazon S3 バケットにアクセスするため、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ときは、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 アクセスポイント をサポートしています。これは、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る Amazon S3 の機能です。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、 AWS Transfer Family および Amazon S3 のブログ記事でデータアクセスコントロールを強化する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。たとえば、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 渡します。`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、「Amazon Simple Storage

Service APIリファレンスHeadObject」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み (またはアップロード) および一覧表示するが、オブジェクトの読み取り (ダウンロード) は許可しない場合があります。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`と `PutObject` アクセス許可が必要です。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 アクセス拒否 というエラーが発生すると、この呼び出しには アクセスGetObject許可が必要なため、呼び出しは失敗します。

> **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 AWS Identity and Access Management （ IAM) で終わるオブジェクトにのみGetObjectアクセス許可を追加する () ポリシー条件を追加することで、アクセス許可を付与できます/。この条件は、ファイルへのGetObject呼び出し (読み取れないように) を防止しますが、ユーザーがフォルダを一覧表示してトラバースできるようにします。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 HeadObject は大量の呼び出しを発行し、処理が遅延する可能性があります。

この問題に対する考えられる解決策の 1 つは、ゼロバイトのオブジェクトをすべて削除することです。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。例えば、[/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。

考えられるもう 1 つの解決策は、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、呼び出しの数を減らすことです。`HeadObject`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。そのため、EFSファイル転送プロトコル対応サーバーのセットアップの一環として Amazon 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、AWS Organizations 「 ユーザーガイド」の「組織内の AWS アカウントの管理」を参照してください。
- ユーザーを設定するときは、各ユーザーにIAMロールを割り当てます。このロールは、Amazon EFS ファイルシステムへのアクセスレベルを決定します。
- 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 のEFS連携方法の詳細については、「Amazon Amazon Elastic File System ユーザーガイド」の「 を使用して Amazon EFS ファイルシステムのファイル AWS Transfer Family にアクセスする」を参照してください。

Amazon EFS ファイルの所有権

Amazon EFSは、ポータブルオペレーティングシステムインターフェイス (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。

ではPOSIX、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とディレクトリへのアクセス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。

アクセスPOSIX許可モデルでは、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) が前述の 3 つのアクセス許可セットに関連付けられます。Amazon EFS オブジェクトには、Unix スタイル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 スタイルのシステムでは、ユーザーとグループは、Amazon がファイルの所有権を表すためにEFS使用する数値識別子にマッピングされます。Amazon の場合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、マッピングされた数値IDsを使用して、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アクセス許可をチェックします。

Transfer Family の Amazon EFS ユーザーを設定する

Amazon EFS ユーザーを設定する前に、次のいずれかを実行できます。

- ユーザーを作成し、Amazon でホームフォルダを設定できますEFS。詳細については、「Amazon で Transfer Family ユーザーを設定する EFS」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ルートユーザーを作成する。

> **Note**
>
> Transfer Family サーバーは、アクセスPOSIX許可を設定する 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) では、POSIXアクセス許可を設定できます。これらのアクセス許可は、、、UIDGIDおよびセカンダリ に基づいて、きめ細かなアクセスのためにユーザーレベルで設定されますGIDs。

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

Transfer Family は、ユーザーを にUID/GID and directories you specify. If the UID/GID/directoriesまだ存在しない にマッピングします。次にEFS、Transfer でユーザーに割り当てる前に作成する必要があります。Amazon EFS ユーザーの作成の詳細については、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。

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 でユーザーの EFSUIDと をマッピングGIDします。
2. ログイン時に特定のフォルダからユーザーを起動する場合は、 `HomeDirectory`フィールドで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。とやり取りするLambda 関数の例についてはEFS、「サーバーレスアプリケーションでの Amazon EFS AWS Lambda の 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 の論理ディレクトリを設定する EFS 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ルートユーザーを作成する

組織がユーザーの設定のために SFTP/FTPS 経由でルートユーザーアクセスを有効にできる場合は、UIDおよび 0 (ルートユーザー) GID のユーザーを作成し、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 POSIX ID 所有者を割り当てます。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

Amazon EFS for では、次のコマンドがサポートされています AWS Transfer Family。

- `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 ロールとポリシーを作成する

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、ユーザーロールの作成プロセスを順を追って説明します。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 のコンテキストでこのロールを AWS Transfer Family 引き受けますARN。
- 「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、アクセスロールはアグリーメントに 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ARNに を使用します。

- 呼び出しロール – サーバーのカスタム ID プロバイダーとして Amazon API Gateway で使用します。Transfer Family は、Transfer Family サーバー のコンテキストでこのロールを引き受けますARN。
- 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 のコンテキストでこのロールを引き受けますARN。アウトバウンドAS2転送の場合、ログ記録ロールはコネクタ を使用しますARN。
- 「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 ワークフロー のコンテキストでこのロールを引き受けますARN。

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 セッションポリシーの仕組み で確認できます。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
- セッションポリシーの仕組み
- 読み書きアクセスポリシーの例

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。これらの決定には、ユーザーがアクセスできる Amazon S3 バケットまたは Amazon EFS ファイルシステム、各 Amazon S3 バケットのどの部分とファイルシステム内のどのファイルにアクセスできるか、およびユーザーが持つアクセス許可 (PUTまたは など) が含まれますGET。

アクセスを設定するには、そのアクセス情報を提供する ID 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、ファイルオペレーションのターゲットまたはソースである Amazon S3 バケットまたは Amazon ファイルEFSシステムへのアクセスをユーザーに付与します。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、新しいIAM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。

次の手順では、IAMポリシーと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. でIAMコンソールを開きますhttps://console.aws.amazon.com/iam/。
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. ポリシーの作成ページで、 JSONタブを選択します。
4. 表示されるエディタで、エディタの内容をIAMロールにアタッチするIAMポリシーに置き換えます。

   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、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所有者です。
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、各 を置き換えます。*user input placeholder* 独自の情報。

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 AWS Transfer Family 呼び出すIAMロールが作成されました。ユーザーにアクセスを許可するために作成したIAMポリシーをロールにアタッチしました。「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。

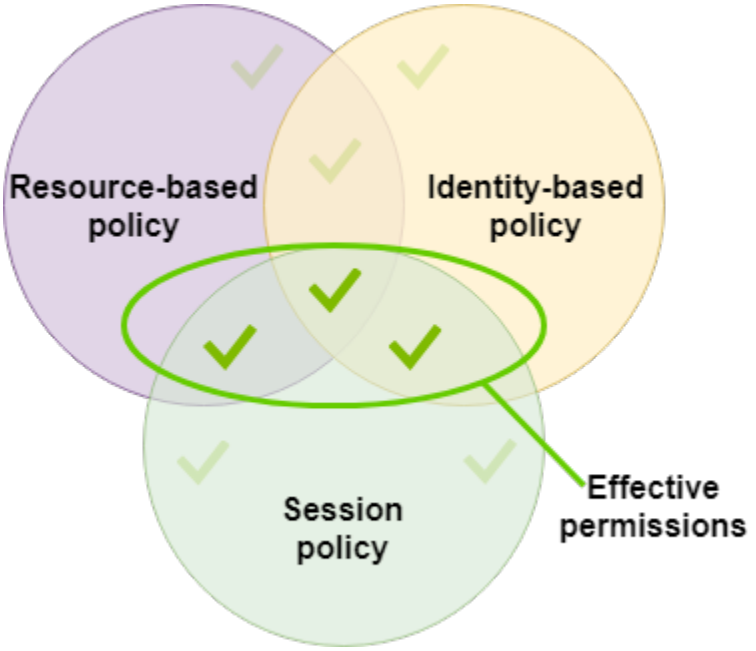
以下の資料も参照してください。

- IAM ロールの詳細については、IAM「 ユーザーガイド」のAWS 「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「 を使用してファイルシステムデータアクセスIAMを制御する」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を設定すると、セッションポリシー を使用して、結果のセッションのアクセス許可を減らすことができます。例えば、読み取り/書き込みアクセス で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URLする 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、「 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。 IAM

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リファレンスの「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。
- 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、ポリシーに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 メソッドの実装。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 パラメータには、Resourceステートメントの S3 Amazon リソースネーム (/) の一部として使用できるように、先頭のスラッシュ (ARN) が削除されました。

> **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 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 を許可するポリシーの例。

次の点に注意してください。

- 置換 *DOC-EXAMPLE-BUCKET* Amazon S3 バケットの名前。
- 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、「Unversioned」、「バージョニング対応」、「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 ファイルシステム内のファイルへのアクセスをEFSファイルシステムに許可する

> **Note**
>
> ポリシーに加えて、POSIXファイルアクセス許可が適切なアクセス許可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 File System ユーザーガイド」の「Network File System (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 ファイルシステム内のファイルへのルートEFSファイルシステムアクセスを許可します。

> **Note**
>
> 次の例では、 を置き換えます。*region* 自分のリージョンで、*account-id* ファイルが属するアカウント、および *file-system-id* Amazon Elastic File System (Amazon EFS) の ID。

```
{
    "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 ファイルシステム内のファイルへのアクセスをユーザーEFSファイルシステムに許可します。

```
{
    "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、SFTPTransfer Family サーバーとサービスマネージドユーザーの作成を順を追って説明し、クライアントを使用してファイルを転送する方法を示します。
- 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について説明します。
- Amazon API Gateway メソッドをカスタム ID プロバイダーとして設定する : 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説明します。
- 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設定する方法と、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ファイルを表示する方法について説明します。
- AS2 設定の設定: このチュートリアルでは、AS2Transfer Family サーバーの設定に必要なステップについて説明します。証明書のインポート、プロファイルとアグリーメントの作成、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
- 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
- SFTP コネクタのセットアップと使用
- Amazon API Gateway メソッドをカスタム ID プロバイダーとして設定する
- AS2 設定の設定

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に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を学びます。

トピック

- 前提条件
- ステップ 1: AWS Transfer Family コンソールにサインインする
- ステップ 2: SFTP対応サーバーを作成する
- ステップ 3: サービスマネージドユーザーを追加する
- 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

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、Transfer Family が使用する Amazon Simple Storage Service、Amazon Elastic File System AWS Certificate Manager、Amazon Route 53 などの他の 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 バケットを設定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 Transfer Family リソースを作成するアクセス許可をSFTPユーザーに付与します。
- IAMFullAccess (または特に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、サーバーにログインするユーザー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 と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 管理ポリシーであり、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、 で AWS Transfer Family コンソールを開きますhttps://console.aws.amazon.com/transfer/。
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 ユーザー名 には、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

Secure Shell (SSH) File Transfer Protocol (SFTP) は、インターネットを介したデータの安全な転送に使用されるネットワークプロトコルです。プロトコルは、 のセキュリティと認証機能をすべてサポートしていますSSH。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. プロトコルの選択 で、 を選択しSFTP、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. [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 をログに記録するには、新しいロールを作成する適切なアクセス許可がある限り、Transfer Family がIAMロールを自動的に作成できるようにする新しいロールを作成するを選択します。作成されるIAMロールは と呼ばれますAWSTransferLoggingAccess。
   b. [Cryptographic algorithm options] (暗号化アルゴリズムオプション) として、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。デフォルトセキュリティポリシーは TransferSecurityPolicy-2020-06 です。
   c. [Next (次へ)] を選択します。
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。

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. [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。
2. [Add user] (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 では、Amazon S3 バケットへのアクセスを提供する、以前に作成したIAMロールを選択します。

   このIAMロールは、 の手順を使用して作成しましたIAM ロールとポリシーを作成する。このIAMロールには、Amazon S3 バケットへのアクセスを提供するIAMポリシーが含まれています。また、別のIAMポリシーで定義されている AWS Transfer Family サービスとの信頼関係も含まれます。

> **Note**
>
> サービスマネージドユーザーのIAMロールには、目的のバケットにアクセスするためのアクセス許可が含まれている必要があります。目的のバケットにアクセスするアクセス許可は、S3FullAccess リソースに管理者レベルのアクセス許可を付与する S3 でカバーされます。

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 では、Amazon S3 バケットを選択して、 を使用して転送するデータを保存します AWS Transfer Family。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ロールがこの`root`ディレクトリへのアクセスを許可していることを確認してください。

   > **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を 含む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。

7. [Restricted] (制限) チェックボックスをオンにすると、ユーザーがそのフォルダの外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。さらにコントロールを適用する必要がある場合、セッションポリシーを使用します。

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。

> **Important**
>
> SSH パブリックキーの形式は ですssh-rsa *<string>*。SSH キーペアを生成する方法については、「」を参照してくださいサービスマネージドユーザーのSSHキーを生成する。

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 Open を使用する手順について説明しますSSH。

トピック

- Cyberduck を使用する
- Open を使用するSSH

Cyberduck を使用する

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 ダイアログボックスで (SFTP File Transfer Protocol) を選択しますSSH。
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、および FTPサーバーの詳細を表示する」を参照してください。

5. ポート番号 の場合は、 **22**に を入力しますSFTP。
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 プライベートキー の場合、SSHプライベートキー を選択または入力します。
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。

Open を使用するSSH

Open を使用してコマンドラインからファイルを転送するには、以下の手順に従いますSSH。

> **Note**
>
> このクライアントは、 SFTPが有効な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`

   前のコマンドでは、 `sftp_user` はユーザー名で、 `transfer-key`はSSHプライベートキーです。ここでは、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`service_endpoint`を示します。

   `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: `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、ファイルの暗号化と復号化、ファイルの暗号化と復号化を PGPと で AWS Transfer Family行う方法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
- ステップ 2: マネージドワークフローを作成する
- 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
- ステップ 4: PGPキーペアを作成する
- ステップ 5: PGPプライベートキーを に保存する AWS Secrets Manager
- ステップ 6: ファイルを暗号化する
- ステップ 7: ワークフローを実行して結果を表示する

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、IAM ワークフローの 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。

以下の実行ロールポリシーには、このチュートリアルで作成するワークフローを正常に実行するために必要なすべての権限が含まれています。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。*DOC-EXAMPLE-BUCKET*を暗号化ファイルをアップロードするAmazon S3バケット名に置き換えます。

> **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種類に基づいて権限を制限できます。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、IAM カスタムステップのアクセス許可 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. で AWS Transfer Family コンソールを開きますhttps://console.aws.amazon.com/transfer/。
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] 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。
- [復号されたファイルの宛先] には、Amazon S3 を選択します。
- 送信先バケット名 で、ステップ 1 で作成したIAMポリシー*DOC-EXAMPLE-BUCKET*で 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、*w-1234abcd5678efghi* をワークフロー ID の例として使用します。

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アタッチしたら、サーバーSFTPに を使用できるユーザーを作成し、ワークフローの実行をトリガー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. で AWS Transfer Family コンソールを開きますhttps://console.aws.amazon.com/transfer/。
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ます。
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 で、 で作成したIAMロールを選択しますステップ 1：実行ロールを設定する。
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. で AWS Transfer Family コンソールを開きますhttps://console.aws.amazon.com/transfer/。
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、所有しているプライベートキーに対応するパブリックキーに貼り付けます。詳細については、「サービスマネージドユーザーのSSHキーを生成する」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 1 つを使用して、PGPキーペアを生成します。このプロセスについては、PGP キーの生成 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、 を暗号化アルゴリズムRSA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。作成するキーのタイプの入力を求められたら、 RSAまたは を選択しますECC。ただし、 を選択した場合はECC、必ず次のいずれかを選択してください。NIST または BrainPool 楕円曲線。選択しない Curve 25519.

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プライベートキーを に保存する AWS Secrets Manager

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。

Secrets Manager にPGPプライベートキーを保存するには

1. にサインイン AWS Management Console し、 で AWS Secrets Manager コンソールを開きますhttps://console.aws.amazon.com/secretsmanager/。
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。
   - value – PGPキーペアを生成したときに使用したパスフレーズを に入力しますステップ 4: PGPキーペアを作成する。
7. [Next (次へ)] を選択します。
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。このチュートリアルでは、すべてのユーザーが使用できるデフォルトシークレットを作成できます。サーバー ID が であると仮定する *`s-11112222333344445`*、シークレットに名前を付けます**`aws/transfer/`*`s-11112222333344445`*`/@pgp-default`**。*`s-11112222333344445`*を Transfer Family サーバーの ID に置き換えます。シークレットの説明を入力します。

> **Note**
>
> 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。

Secrets Manager にPGPプライベートキーを追加する方法の詳細については、「キーの保存 AWS Secrets Manager に使用するPGP」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` 引数については、 をPGPキーペアの作成時に使用した E メールアドレス*marymajor@example.com*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、 をユーザーのSSHプライベートキー*transfer-key*に置き換えます。

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

例えば、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
```

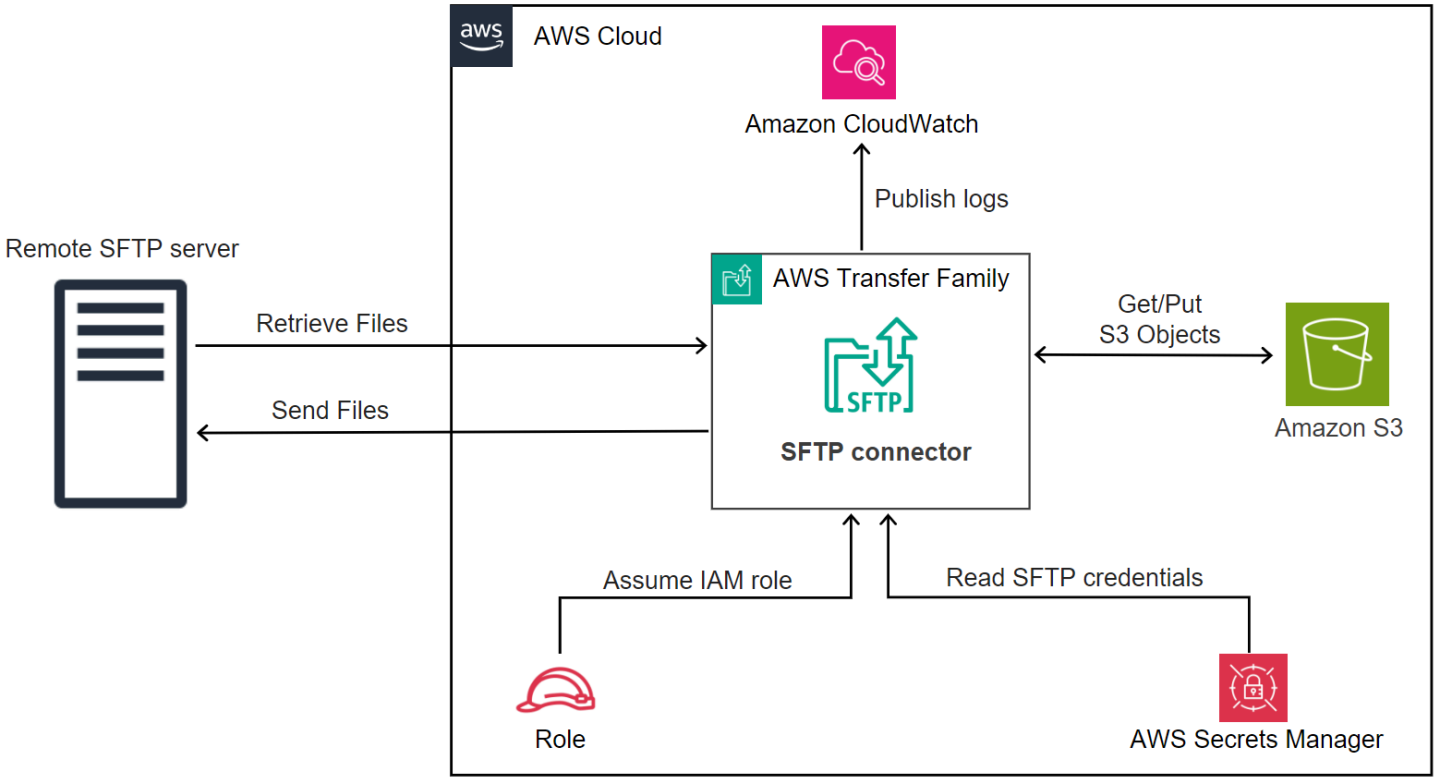
成功すると、lsコマンドはtestfile.txtファイルを一覧表示します。このファイルをダウンロードして、以前に暗号化した元のファイルと同じかどうかを確認できます。

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について説明します。

SFTP コネクタは からSFTP認証情報を取得し AWS Secrets Manager 、リモートSFTPサーバーに認証して接続を確立します。コネクタは、リモートサーバーに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。



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
- リモートサーバーとして使用する Transfer Family SFTP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 AWS Transfer Family サーバーとしてSFTP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。

- Amazon S3 バケットを作成して、 AWS 環境にファイルを保存し、リモートSFTPサーバーからファイルを送信および取得します。 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. で AWS Transfer Family コンソールにサインインしますhttps://console.aws.amazon.com/s3/。
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、 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。*DOC-EXAMPLE-BUCKET* Amazon S3、および Secrets Manager に保存されている指定されたシークレット。

```
{
  "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*、チュートリアルでは を使用します**s3-storage-east**。

- [*region*、チュートリアルでは を使用します**us-east-1**。
- [*account-id*、ID を使用します AWS アカウント 。
- [*SecretName-6RandomCharacters*、名前**using sftp-connector1**は 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する必要があります。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保存されているパブリックキーに対応する必要があります。

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 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、 で AWS Secrets Manager コンソールを開きますhttps://console.aws.amazon.com/secretsmanager/。
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. 行の追加 を選択し、 を入力します**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 (以前に保存) 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) 。

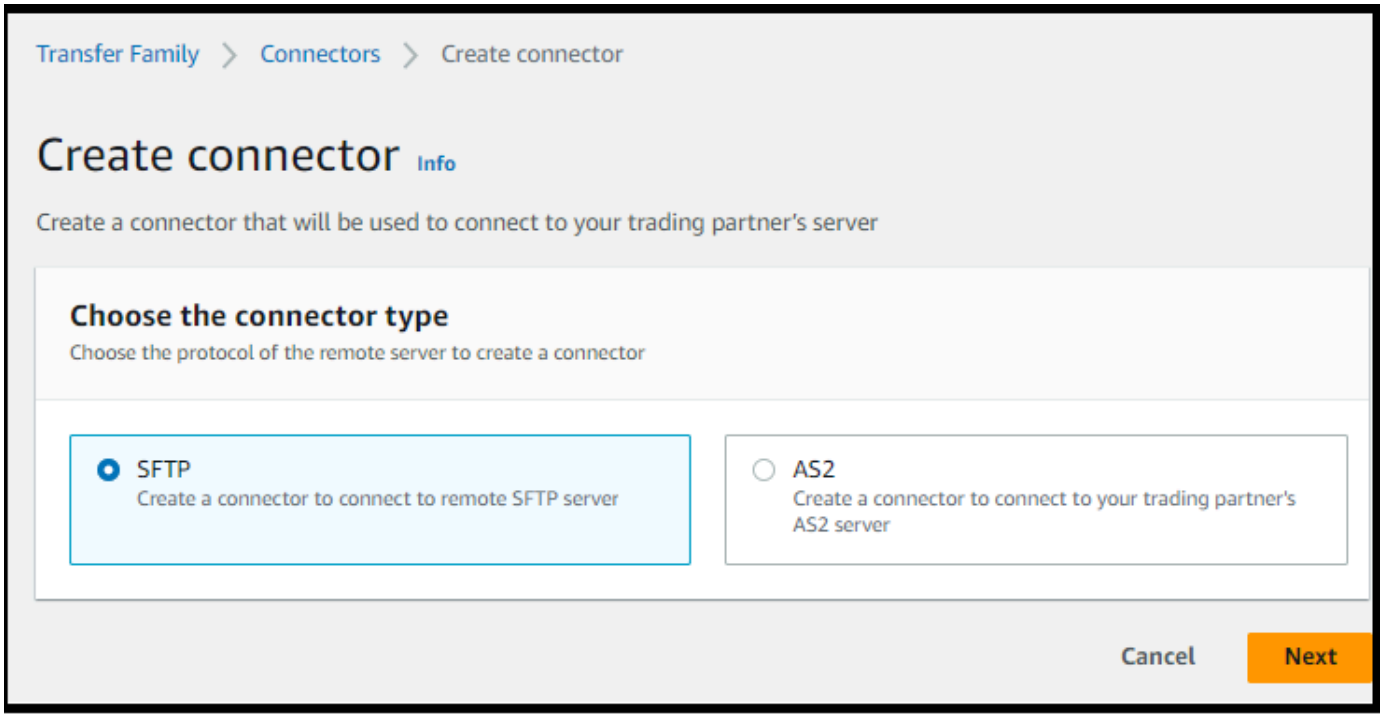
7. [Next (次へ)] を選択します。
8. シークレットの設定ページで、シークレットの名前を入力します。このチュートリアルでは、シークレット に名前を付けます**aws/transfer/sftp-connector1**。
9. [次へ] を選択し、[ローテーションの設定] ページのデフォルトを受け入れます。次いで、[次へ] を選択します。
10. [レビュー] ページで [ストア] を選択し、シークレットを作成して保存します。

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 コネクタの設定」を参照してください。

SFTP コネクタを作成するには

1. で AWS Transfer Family コンソールを開きますhttps://console.aws.amazon.com/transfer/。
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 コネクタタイプを選択してSFTPコネクタを作成し、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。

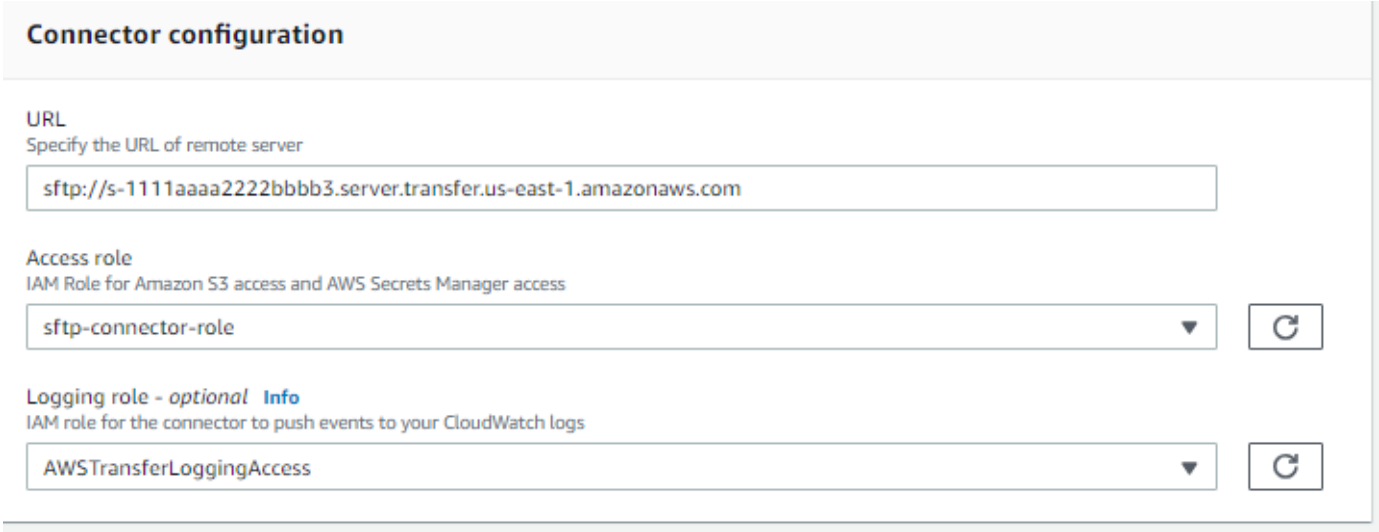
   - にURL、URLリモートSFTPサーバーの を入力します。このチュートリアルでは、リモートサーバーとして使用している URL Transfer Family SFTPサーバーの を入力します。

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

     置換 *1111aaaa2222bbbb3* Transfer Family サーバー ID を使用します。

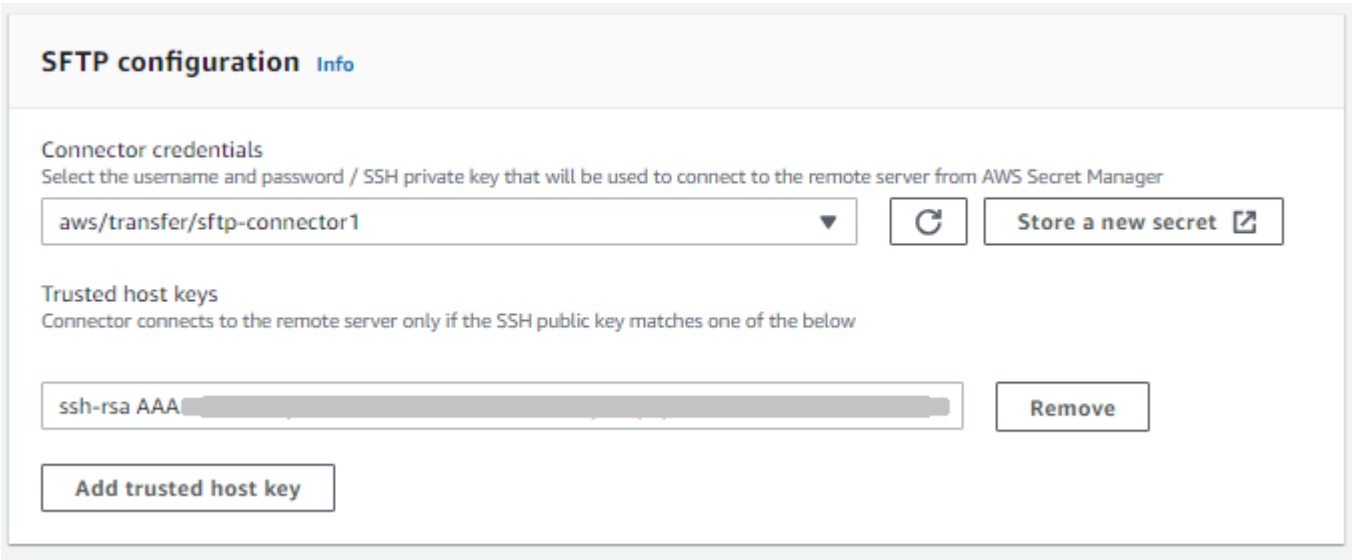
   - アクセスロール には、前に作成したロール を入力します**sftp-connector-role**。
   - ログ記録ロール で、 を選択します**AWSTransferLoggingAccess**。

     > **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 設定 セクションで、次の情報を指定します。

   - Connector 認証情報 では、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**。
   - 信頼されたホストキー の場合、ホストキーのパブリック部分に貼り付けます。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をフォーマットして保存する方法については、「」を参照してください。 SftpConnectorConfig データ型ドキュメント。



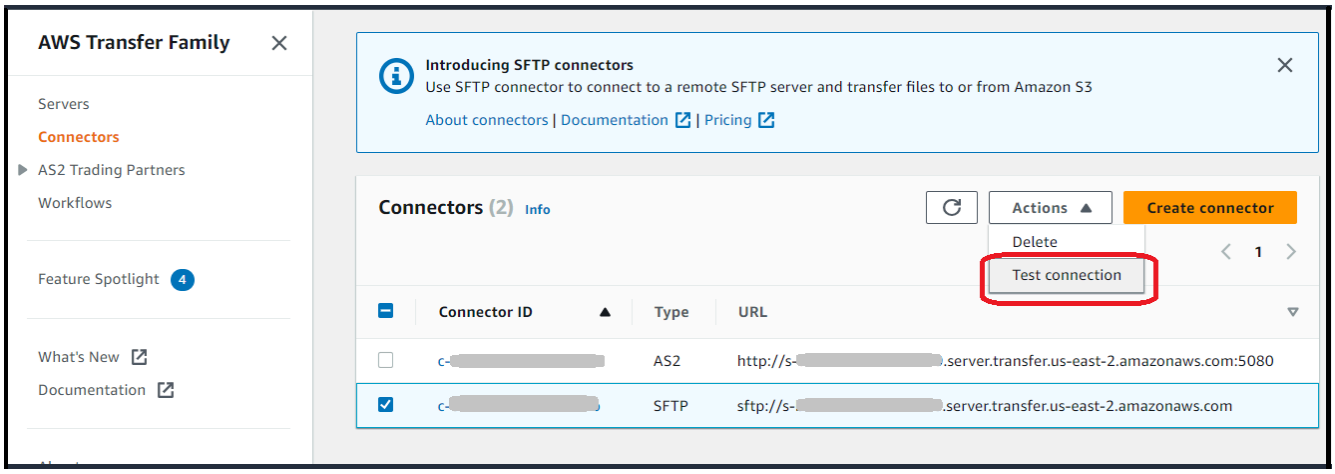
6. すべての設定を確認したら、コネクタの作成を選択してSFTPコネクタを作成します。

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

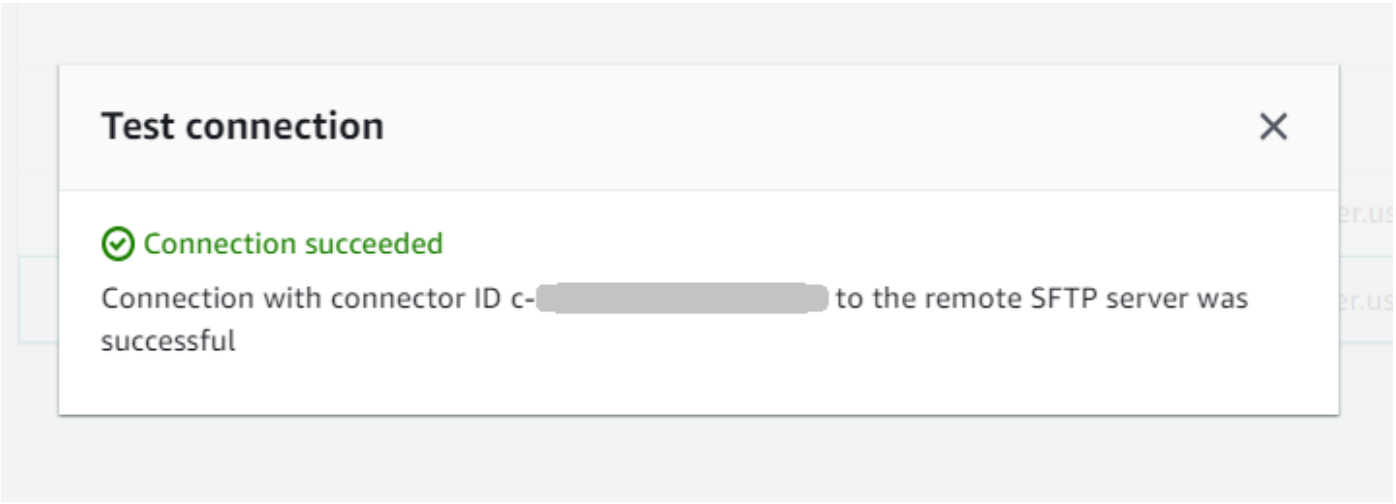
Test a connector using the console

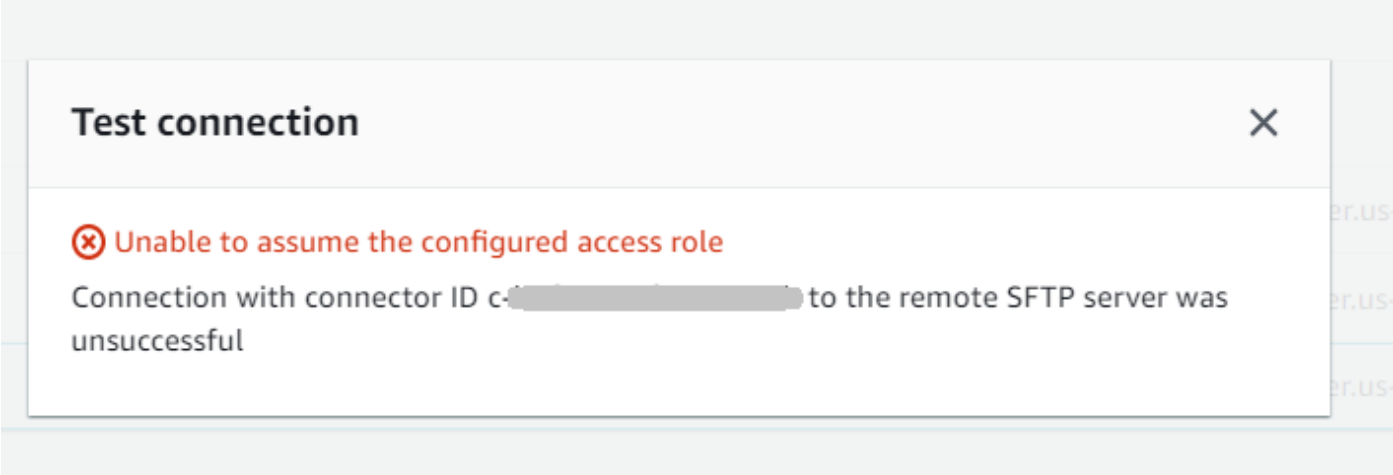
SFTP コネクタをテストするには

1. で AWS Transfer Family コンソールを開きますhttps://console.aws.amazon.com/transfer/。
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 (置き換える *connector-id* 実際のコネクタ ID を使用):

```
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"
}
```

テストが失敗した場合は、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

簡単にするために、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できるファイルの場所へのパスに置き換えることができます。

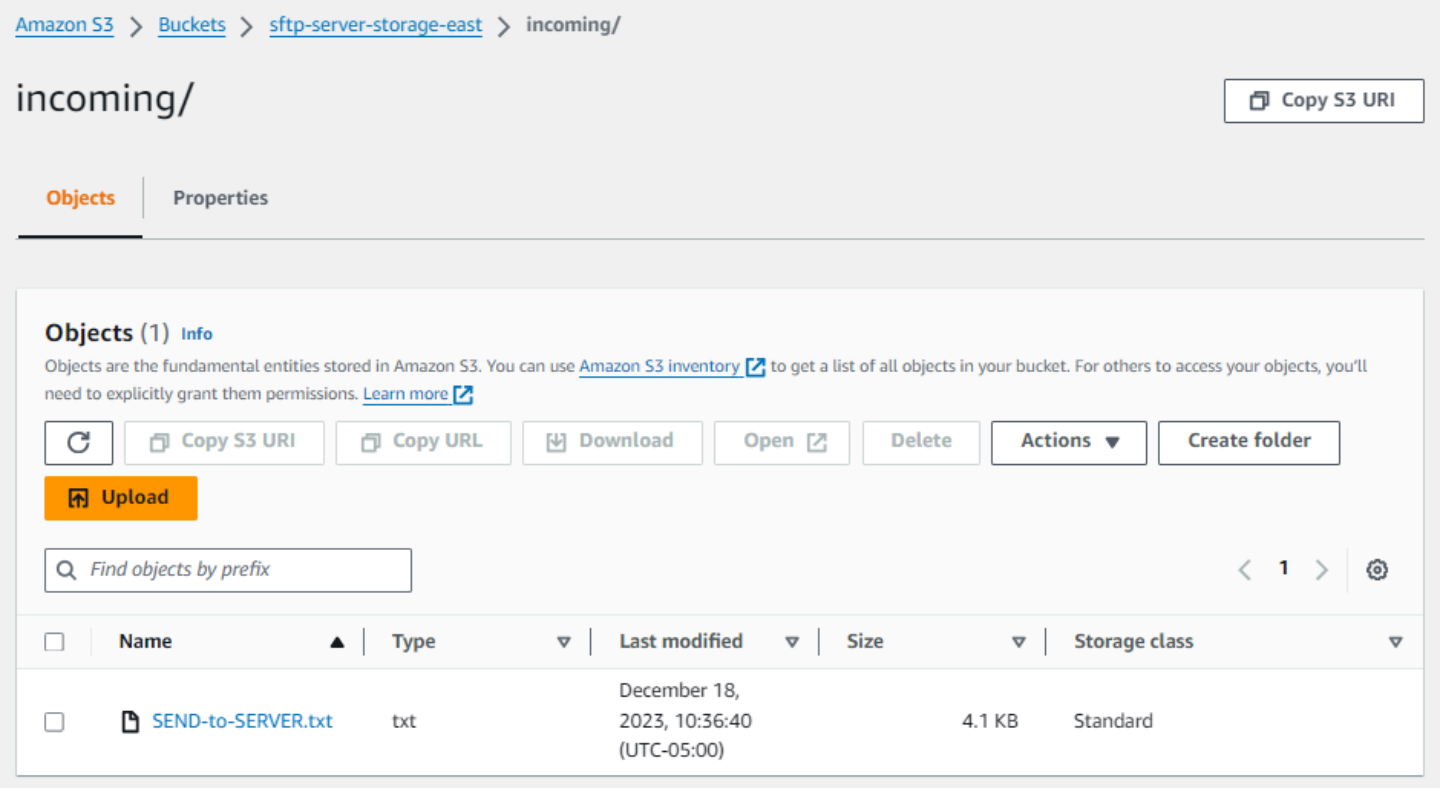
- Amazon S3 ストレージSEND-to-SERVER.txtから という名前のファイルをSFTPサーバーに送信します。
- SFTP サーバーRETRIEVE-to-S3.txtから Amazon S3 ストレージに という名前のファイルを取得します。

> **Note**
>
> 次のコマンドで、 を置き換えます。*connector-id* コネクタ ID を使用します。

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

成功すると、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",
```

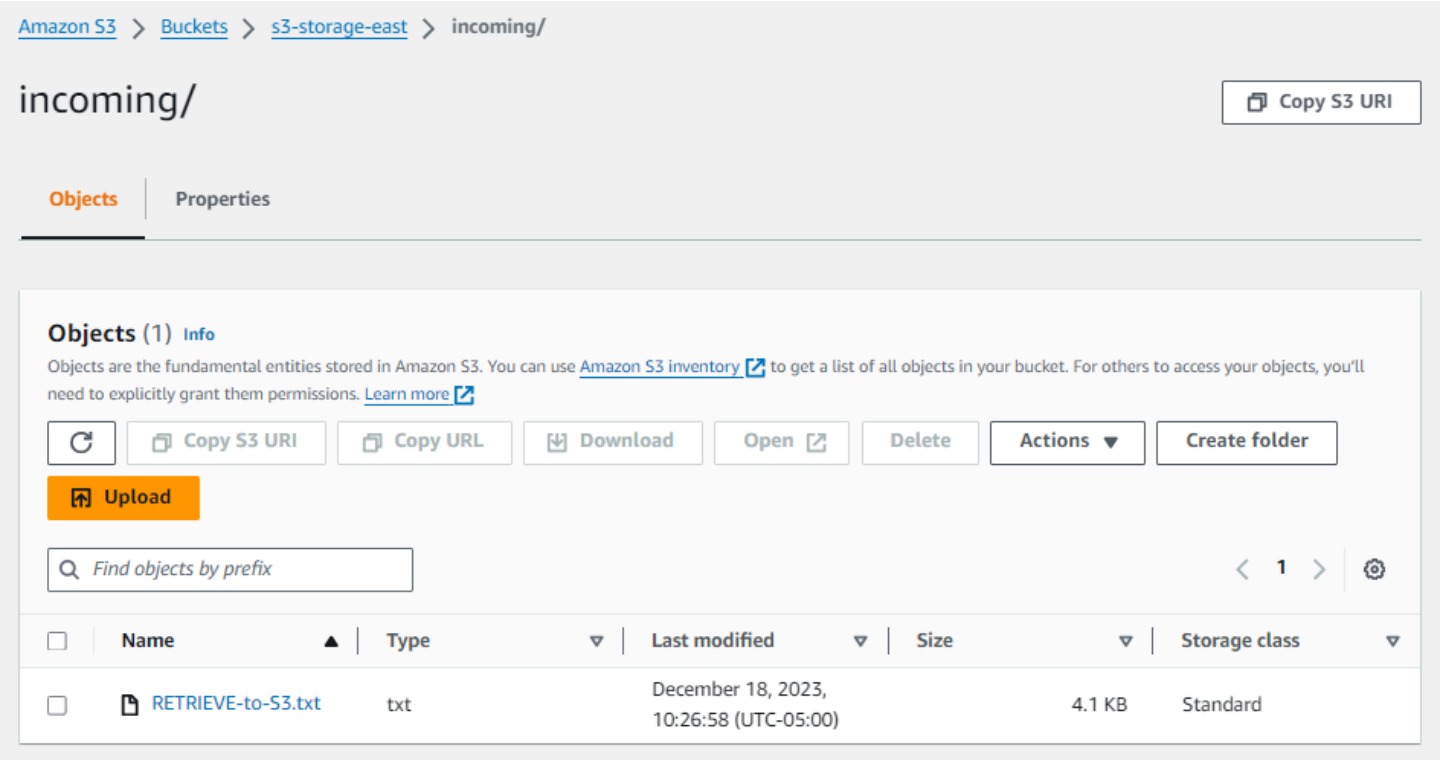
```
    "status-code": "COMPLETED",
    "start-time": "2023-12-18T15:26:56.915864Z",
    "end-time": "2023-12-18T15:26:57.298122Z",
    "account-id": "500655546075",
    "connector-arn": "arn:aws:transfer:us-east-1:500655546075:connector/connector-id",
    "remote-directory-path": "/sftp-server-storage-east/incoming"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

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、ここに示すように転送されたファイルが含まれます。



成功すると、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"
}
```

リモートサーバーとして使用する Transfer Family SFTP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サーバーとして機能する Transfer Family SFTP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、簡単にするために、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を、コネクタを使用するために必要なアクセス許可と組み合わせました。繰り返しになりますが、ほとんどのSFTPコネクタユースケースには、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。
- 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 番目のバケット 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、既にユーザーを持つSFTPサーバーがあり、このサーバーを使用してファイルを送受信できるため、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。ただし、このチュートリアルでは、簡単にするために、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。

で説明されている手順に従ってサーバーSFTP対応サーバーを作成するを作成し、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。その後、Secrets Manager でシークレットを作成するときは、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。チュートリアルでは、SFTPユーザーとSFTPコネクタの両方に同じIAMロールを使用しています。組織のコネクタを作成する場合、ユーザーロールとアクセスロールがそれぞれ異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、エンドポイントが の場合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
  ```

  ステップ 2: SFTP コネクタを作成してテストする プロシージャに貼り付ける必要があるため、このテキストを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

チュートリアルでは、単一の複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。次の例では、チュートリアルのタスクを実行する場合に備えて、このロールの詳細を示します。

次の例では、Amazon S3 の 2 つのバケットにアクセスするために必要なアクセス許可と、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。

Amazon API Gateway メソッドをカスタム ID プロバイダーとして設定する

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について説明します。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
- ステップ 1: CloudFormation スタックを作成する
- ステップ 2: サーバーの API Gateway メソッド設定を確認する
- ステップ 3: Transfer Family サーバーの詳細を表示する
- 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
- ステップ 5: SFTP接続とファイル転送をテストする
- ステップ 6: バケットへのアクセスを制限する
- Amazon を使用している場合は Lambda を更新する EFS

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 ロールとポリシーを作成する 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. AWS CloudFormation コンソールを https://console.aws.amazon.com/cloudformation で開きます。
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、ベーシックスタックテンプレート をコピーし、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 と のデフォルト値を置き換えますUserPassword。
   - にはUserHomeDirectory、以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、以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、「」を参照してくださいステップ 6: バケットへのアクセスを制限する。
   - 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。を使用してサーバーに初めて接続するときはSFTP、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**
>
> ページの下部の 機能 で、 が AWS CloudFormation IAMリソースを作成する可能性があることを確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**
>
> セキュリティを向上させるために、ウェブアプリケーションファイアウォールを設定できます。 AWS WAF は、Amazon API Gateway に転送される HTTPおよび HTTPSリクエストをモニタリングできるウェブアプリケーションファイアウォールです。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 メソッド設定を確認してデプロイするには

1. で API Gateway コンソールを開きますhttps://console.aws.amazon.com/apigateway/。
2. テンプレートが生成した Transfer Custom Identity Provider の基本API AWS CloudFormation テンプレートを選択します。
3. リソースペインで、 を選択しGET、メソッドリクエスト を選択します。
4. アクション で、デプロイ APIを選択します。[Deployment 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、ステージエディタセクションでそのパフォーマンスを表示します。

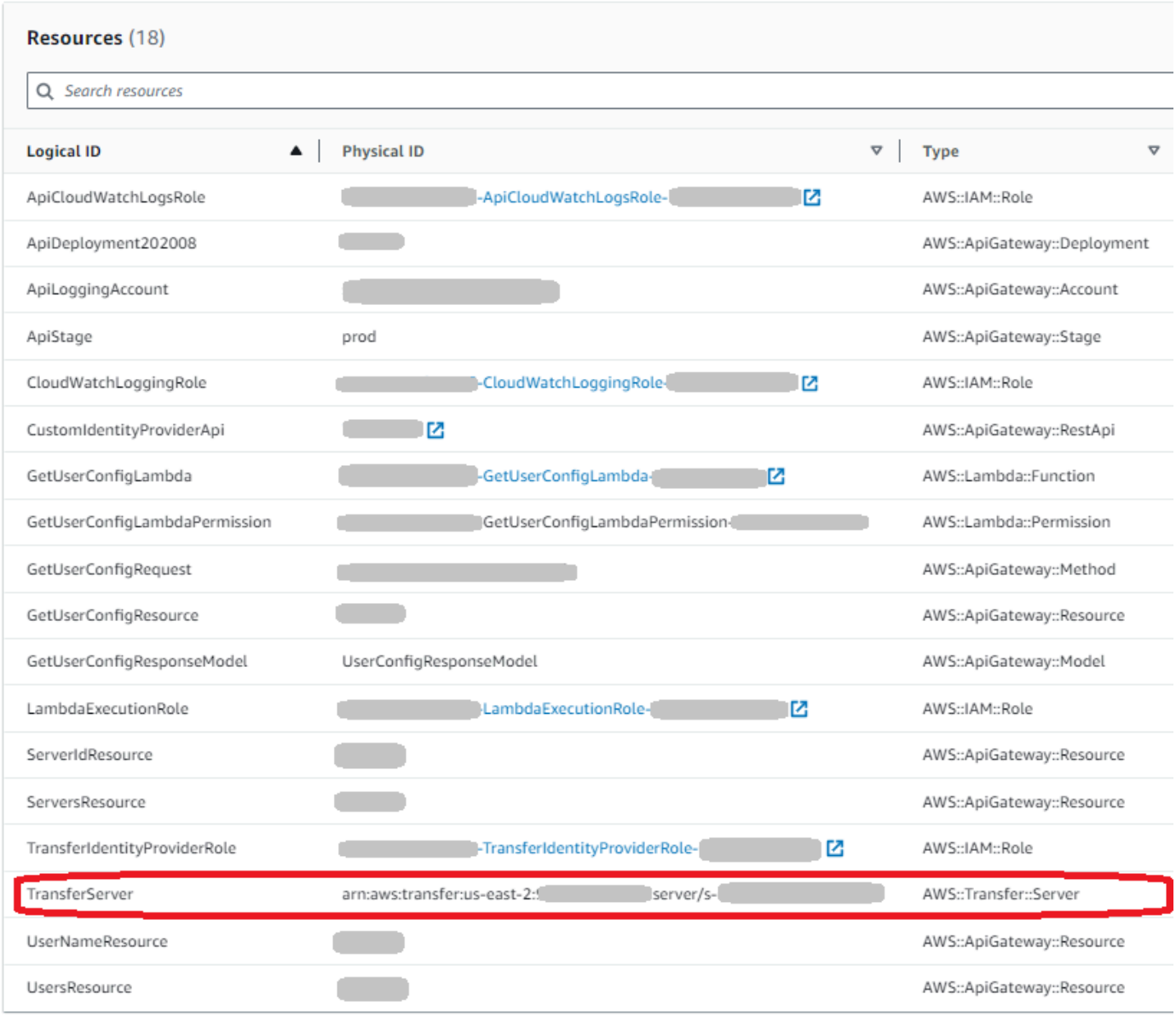
> **Note**
>
> ページの上部に表示される呼び出しURLアドレスをコピーします。次のステップで必要になります。

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. AWS CloudFormation コンソールを https://console.aws.amazon.com/cloudformation で開きます。
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、s-11112222333344445 などARN、 に含まれています。

4. で AWS Transfer Family コンソールを開きhttps://console.aws.amazon.com/transfer/、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. で AWS Transfer Family コンソールを開きますhttps://console.aws.amazon.com/transfer/。
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. サーバープロトコル で を選択しSFTP、ソース IP で と入力します**127.0.0.1**。
5. [テスト] を選択します。

   ユーザー認証が成功すると、テストはStatusCode: 200HTMLレスポンスと、ユーザーのロールとアクセス許可の詳細を含むJSONオブジェクトを返します。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ゲートウェイ AWS 管理ポリシーの 1 つを、 に使用しているロールに追加しますAPI。

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ときは、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/*"
        }
    ]
}
```

を使用してサーバーに接続するにはSFTP、プロンプトに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 を使用している場合は Lambda を更新する EFS

Transfer Family サーバーのストレージオプションEFSとして Amazon を選択した場合は、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. で Lambda コンソールを開きますhttps://console.aws.amazon.com/lambda/。
2. 先ほど作成した Lambda 関数を選択します。Lambda 関数の形式は です。 ***stack-name***-GetUserConfigLambda-***lambda-identifier***、ここで *stack-name* は CloudFormation スタック名で、*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},
   ```

   ここで、*uid-value* また、*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 設定の設定

このチュートリアルでは、 で Applicability Statement 2 (AS2) 設定を設定する方法について説明します AWS Transfer Family。ここで説明するステップを完了すると、サンプル取引パートナーからのAS2メッセージを受け入れる準備ができた AS2対応サーバーが完成します。また、サンプル取引先に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: の証明書を作成する AS2」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 を使用してファイル交換AS2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で有効な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 のリモート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: の証明書を作成する AS2

AS2 交換のどちらの当事者にも 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、少なくとも 2048 ビット、最大 4096 ビットである必要があります。

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**
>
> でキーを管理する場合はGUI、 Portecle は、使用できる 1 つのオプションです。

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、2 つのルート認証局 () を生成する必要はありませんCAs。

1. 次のコマンドを実行して、2048 ビット長のモジュラスを持つ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。
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。
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**
>
> Open などの一部の AS2対応サーバーではAS2、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、署名するルートキーの Certificate Signing Requests (CSRs) 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 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。

> **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 サーバーを作成する方法と同様に、 `create-server` AWS CLI コマンドの `--protocols AS2`パラメータを使用して AS2対応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 Amazon S3 ストレージのみをAS2プロトコルで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、AS2メッセージを受け入れることができるようにTCPポート 5080 を公開します。

VPC エンドポイントをインターネットに公開する場合は、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。

- の ID VPC (vpc-abcdef01 など ）。
- VPC サブネットIDsの （ 例えば、subnet-abcdef01、subnet-subnet-abcdef01、subnet-021345ab など ）。
- 取引相手からのTCPポート 5080 での受信トラフィックを許可する 1 つ以上のIDsセキュリティグループ (例: sg-1234567890abcdef0 および sg-abcdef01234567890 ）。
- （ オプション) VPCエンドポイントに関連付ける Elastic IP アドレス。

- 取引先が VPC経由で に接続されていない場合はVPN、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エンドポイントを公開できます。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-abcd012345cccccc
```

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

この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URLに提供する のは、次のとおりです。

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

3. AS2が有効なサーバーにアクセスできるかどうかをテストするには、次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 (Elastic IP アドレスをエンドポイントに関連付ける場合) からサーバーにアクセスできることを確認してください。

   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) レスポンスを受け取ります。

   - 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
     ```

   - 内で接続する場合はVPC、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
     ```

     このdescribe-serverコマンドは、 VpcEndpointIdパラメータでVPCエンドポイント IDを返します。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。次のコマンドでリージョンDNS名 (アベイラビリティーゾーンを含まない名前) 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、構造化されたJSON形式で送受信されたメッセージのステータスを 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許可するには、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 ロールとポリシーを作成する」を参照してください。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 証明書をインポートする” 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*をパートナーAS2の名前や証明書 など、独自の情報に置き換えますIDs。

```
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、*MYCORP* 組織の名前、および *PARTNER-COMPANY* と取引相手の組織の名前。

ステップ 5: 自分とパートナーとの間で契約を作成する

この手順では、 を使用してAS2アグリーメント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 アグリーメントの作成”を参照してください。

アグリーメントには、2 つのプロファイル (ローカルとパートナー) 、その証明書、および 2 つの当事者間のインバウンドAS2転送を許可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、バケットとの間で読み取り/書き込みアクセスが可能な 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SFTP、Transfer Family 、、および FTPSプロトコルの場合と同じでありFTP、 で利用できますIAM ロールとポリシーを作成する。

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)
- バケットにアクセスできるIAMロールARNの 。
- 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

この手順では、 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 Transfer Family コンソールを使用する場合は、the section called “AS2 コネクタの設定” を参照してください。

StartFileTransfer API オペレーションを使用して、コネクタを使用して Amazon S3 に保存されているファイルを取引先の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 を指定する必要がありますURL。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 を使用してファイル交換AS2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ると、Transfer Family はパブリック IP アドレスを含むDNS名前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。次の形式で、取引相手URLにサーバーを提供します。

```
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 の取引先HTTPPOSTからのAS2メッセージを受け入れることができる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 を返すとしますvpce-1234abcd5678efghi。次に、次のコマンドを使用してDNS名前を取得します。

```
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 名前は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VPC、取引相手が 内にある必要があります (AWS PrivateLink または などVPN)。VPC エンドポイントURLを次の形式でパートナーに提供します。

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

例えば、前のコマンドのプレースホルダーのサンプル値を次URLに示します。

```
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 IDs、URLs、および Amazon リソースネーム (ARNs) 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、、、ロール、プロファイルURL、メッセージ処理通知 (MDNs) 、タグ、モニタリングメトリクスなど、コネクタのすべてのプロパティを返します。

パートナーが正常にファイルを受け取ったことを確認するには、 JSONおよび MDN ファイルを表示します。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 ログに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、SFTP、、および FTPプロトコルの 1 つまたは複数を使用する AWS Transfer Family サーバーエンドポイントの作成FTPSと使用に関する詳細について説明します。

トピック

- ID プロバイダーオプション
- AWS Transfer Family エンドポイントタイプマトリックス
- SFTP、FTPS、またはFTPサーバーエンドポイントの設定
- 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
- サーバーエンドポイントのユーザーの管理
- 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

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いくつかの方法があります。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	可能	いいえ	はい	可能
パスワード認証	不可	はい	はい	可能
AWS Identity and Access Managemen t （ IAM) と POSIX	あり	はい	はい	可能

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	あり	はい	はい	可能
パラメータ化されたアクセス (ユーザー名ベース)	あり	はい	はい	可能
アドホックアクセス構造	可能	いいえ	はい	はい
AWS WAF	いいえ	いいえ	はい	不可

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POSIX使用され、Amazon に使用されますEFS。
- 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 を使用した 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 サーバーをすばやく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。

次の手順では、 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対応サーバーを作成する
- FTPS対応サーバーを作成する
- FTP対応サーバーを作成する
- の設定 AS2

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 - 内部	VPC_Endpoint (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、 に特別な設定は必要ありません VPC。	インターネット経由、 AWS Direct Connect または 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VPN。	AWS Direct Connect または経由のオンプレミスデータセンターなど、 VPC および VPCに接続された環境内からVPN。	AWS Direct Connect または経由のオンプレミスデータセンターなど、 VPC および VPCに接続された環境内からVPN。
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 - 内部	VPC_Endpoint (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 - 内部	VPC_Endpoint (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ホストされたエンドポイントの場合、SFTPTransfer Family サーバーはポート 22 (デフォルト) またはポート 2222 で	ソース 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た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ACLsアタッチされたネットワークを使用できます。	ソース 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た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た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 ACLs) を使用できます。	ソース 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た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ACLsアタッチされたネットワークを使用できます。

特徴	Public	VPC - インターネット	VPC - 内部	VPC_Endpoint (非推奨)
	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DNSの名前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DNSの名前またはサーバーにアタッチされた Elastic IP アドレスを許可できます。	プライベート IP アドレスまたはエンドポイントDNSの名前を許可できます。	プライベート IP アドレスまたはエンドポイントDNSの名前を許可できます。

> **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。を使用する代わりにEndpointType=VPC_ENDPOINT、新しいVPCエンドポイントタイプ (EndpointType=VPC) を使用します。このタイプは、前の表で説明したように、内部またはインターネット対応 として使用できます。詳細については、「VPC_ の使用を中止するENDPOINT」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。

- 内部アクセスを持つVPCエンドポイントを使用して、サーバーが AWS Direct Connect または 経由でオンプレミスデータセンターなどの VPCまたは VPC接続環境内のクライアントにのみアクセスできるようにしますVPN。

- 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し、サーバーを保護するには、インターネット向けアクセスのVPCエンドポイントを使用します。次に、 VPCのセキュリティグループを変更して、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。Transfer Family SFTPサーバーで OS ネイティブコマンドを実行するために、基盤となるサーバーに直接アクセスすることはできません。
- Network Load Balancer は、内部アクセス権を持つVPCエンドポイントの前に使用します。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) – を介したファイル転送SSH。詳細については、「the section called “SFTP対応サーバーを作成する”」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 TypeScript、 GitHub ここで入手できます。

- ファイル転送プロトコルセキュア (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。

- ファイル転送プロトコル (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対応サーバーを作成する”」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 “設定 AS2”」を参照してください。ではAS2、デモ目的で 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デモ Transfer Family AS2スタックを作成する」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。ではFTP、 SFTPおよび とは別の認証情報を維持することをお勧めしますFTPS。これは、 SFTPや とは異なりFTPS、が認証情報をクリアテキストでFTP送信するためです。FTP 認証情報を SFTPまたは から分離することでFTPS、FTP認証情報が共有または公開された場合、 を使用するワークロード、SFTPまたは は安全FTPSになります。

サーバーを作成するときは、そのサーバーに割り当てられたユーザーのファイルオペレーションリクエスト AWS リージョン を実行する特定の を選択します。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 を使用して管理されるサービス。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 プロバイダーメソッドは 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ユーザーを認証および承認できるようにします。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 CloudFormation、Amazon API Gateway メソッドを使用して認証するテンプレートについて説明します。

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、任意のドメインネームシステム () サービスを使用して、サーバーにエンドポイントタイプ (パブリックアクセス可能またはVPCホストDNS) とホスト名を割り当てます。サーバーのホスト名は、作成された AWS リージョン 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、サーバーで使用可能な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、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り AWS 料金見積りツール を取得するために と を使用する料金の詳細については、AWS Transfer Family 「 の料金」を参照してください。

SFTP対応サーバーを作成する

Secure Shell (SSH) File Transfer Protocol (SFTP) は、インターネットを介したデータの安全な転送に使用されるネットワークプロトコルです。プロトコルは、 のセキュリティと認証機能をすべてサポートしていますSSH。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。

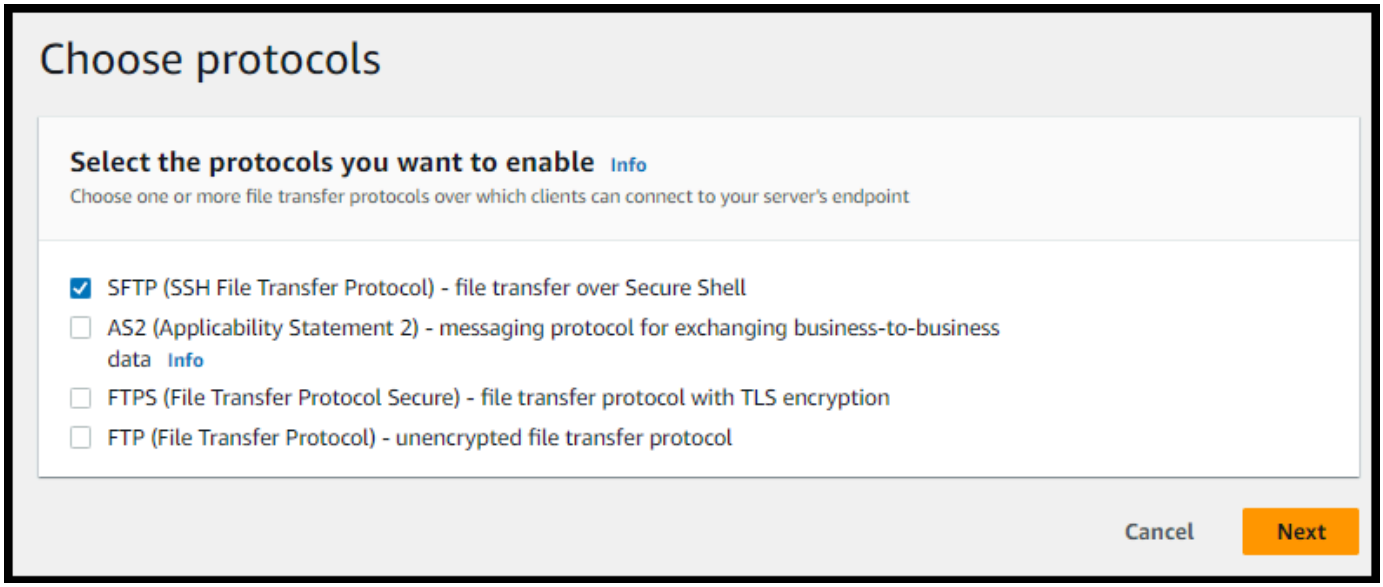
> **Note**
>
> SFTP Transfer Family の サーバーはポート 22 経由で動作します。VPCホストされたエンドポイントの場合、SFTPTransfer Family サーバーはポート 2222 経由で動作することもできます。詳細については、「Virtual Private Cloud でサーバーを作成する」を参照してください。

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 TypeScript、 GitHub ここで で利用できます。
- Transfer Family サーバーを 内にデプロイする手順についてはVPC、「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。

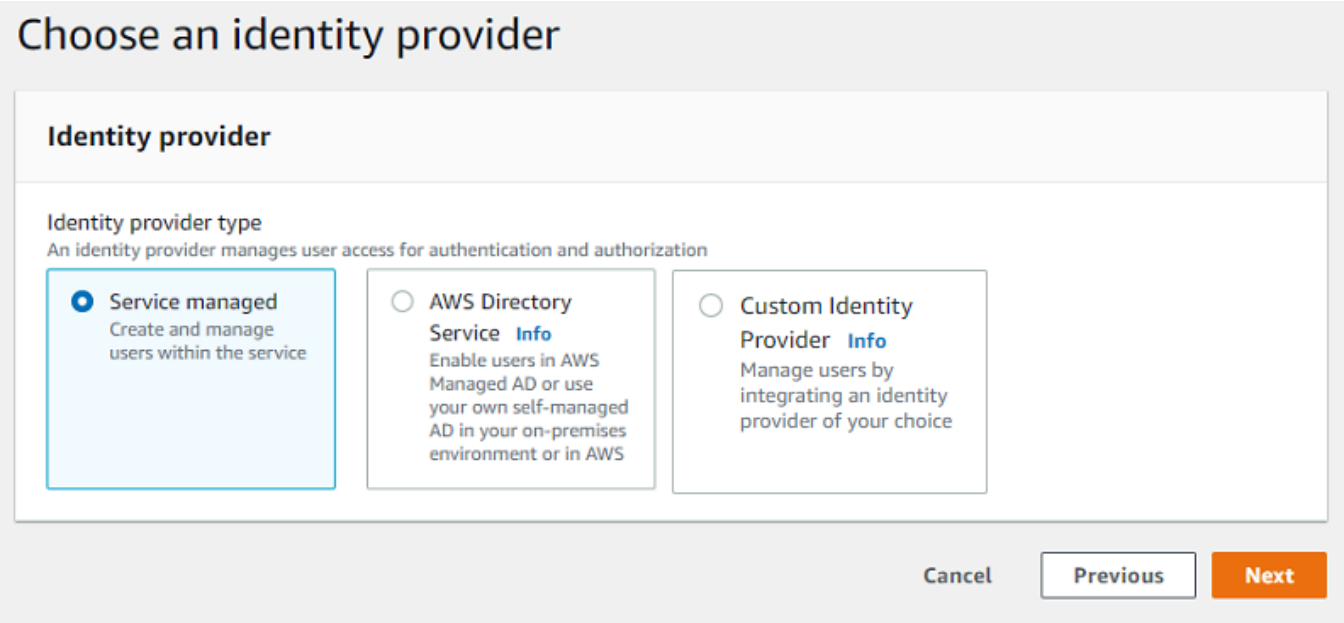
SFTP対応サーバーを作成するには

1. で AWS Transfer Family コンソールを開きhttps://console.aws.amazon.com/transfer/、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. プロトコルの選択 で、 を選択しSFTP、次へ を選択します。



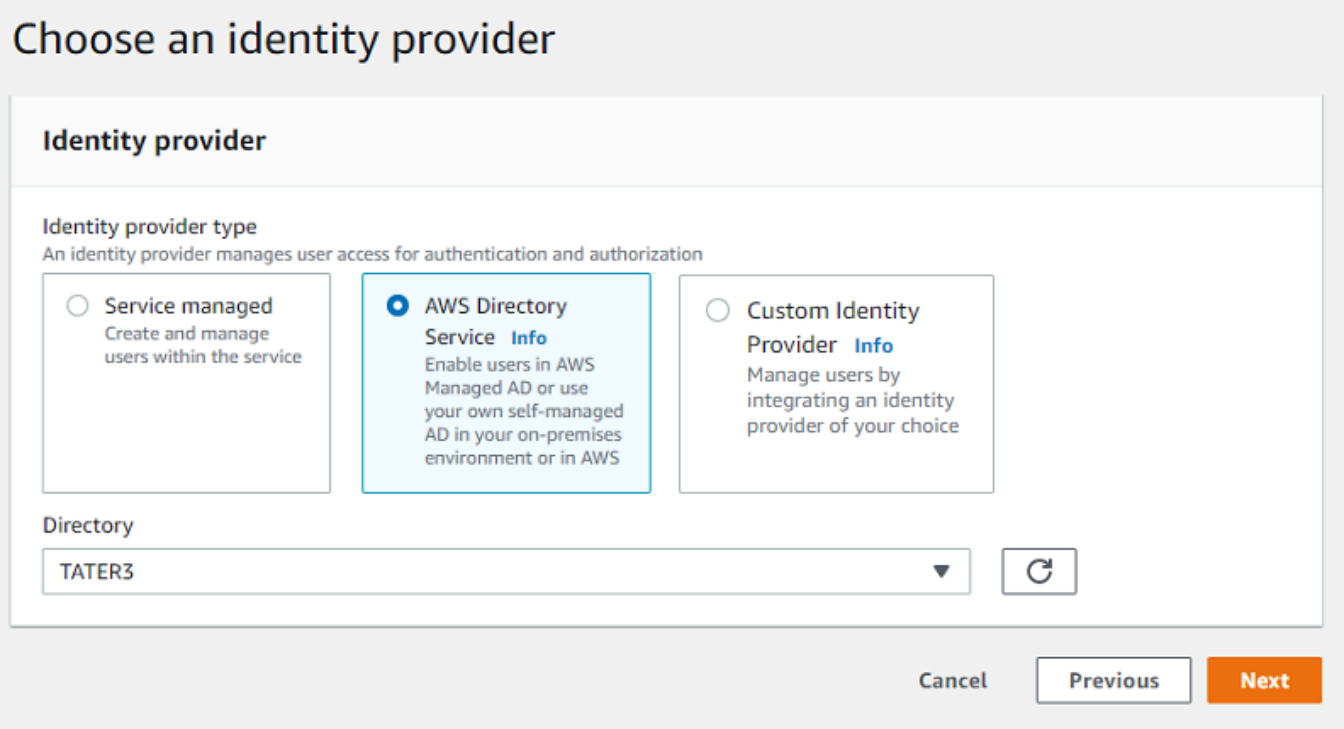
3. [Choose an identity provider] (ID プロバイダーの選択) で、ユーザーアクセスの管理に使用したい ID プロバイダーを選択します。次のオプションがあります。

   - サービスマネージド – ユーザー ID とキーは に保存されます AWS Transfer Family。



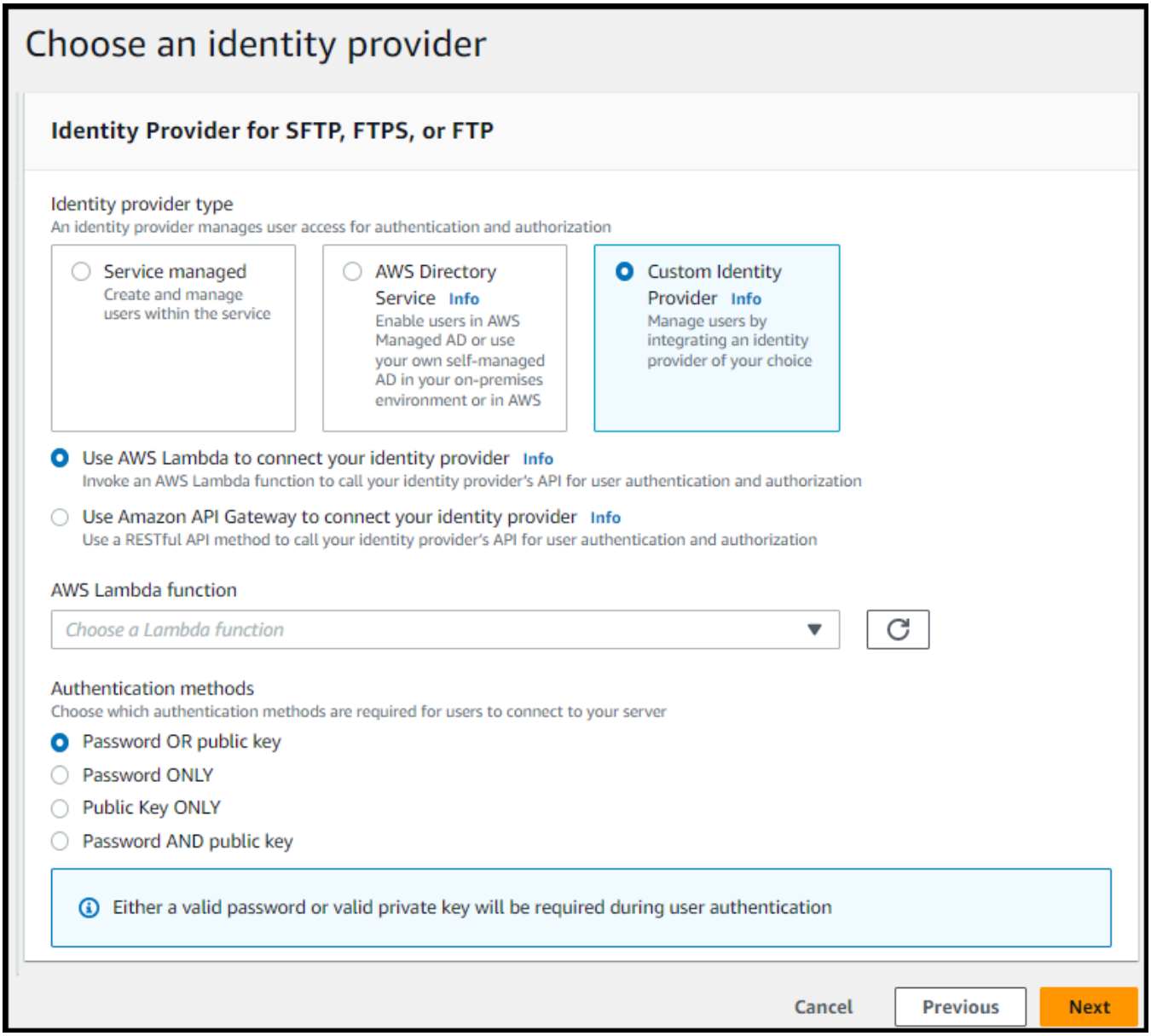
- AWS Directory Service for Microsoft Active Directory – エンドポイントにアクセスするための AWS Directory Service ディレクトリを指定します。そうすることで、Active Directory に保存されている認証情報を使用してユーザーを認証できるようになります。 AWS Managed Microsoft AD ID プロバイダーの操作の詳細については、「」を参照してくださいAWS Directory Service ID プロバイダーの使用。

> **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」 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 を使用して ID プロバイダーを接続する – Lambda 関数でバックアップされた API Gateway メソッドを作成して、ID プロバイダーとして使用できます。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 ONLY – ユーザーは接続するためにパスワードを指定する必要があります。
  - キー ONLY – 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ANDキー – 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. [Next (次へ)] を選択します。
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 ホストされたエンドポイントについては、「」を参照してくださいVirtual Private Cloud でサーバーを作成する。

   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。

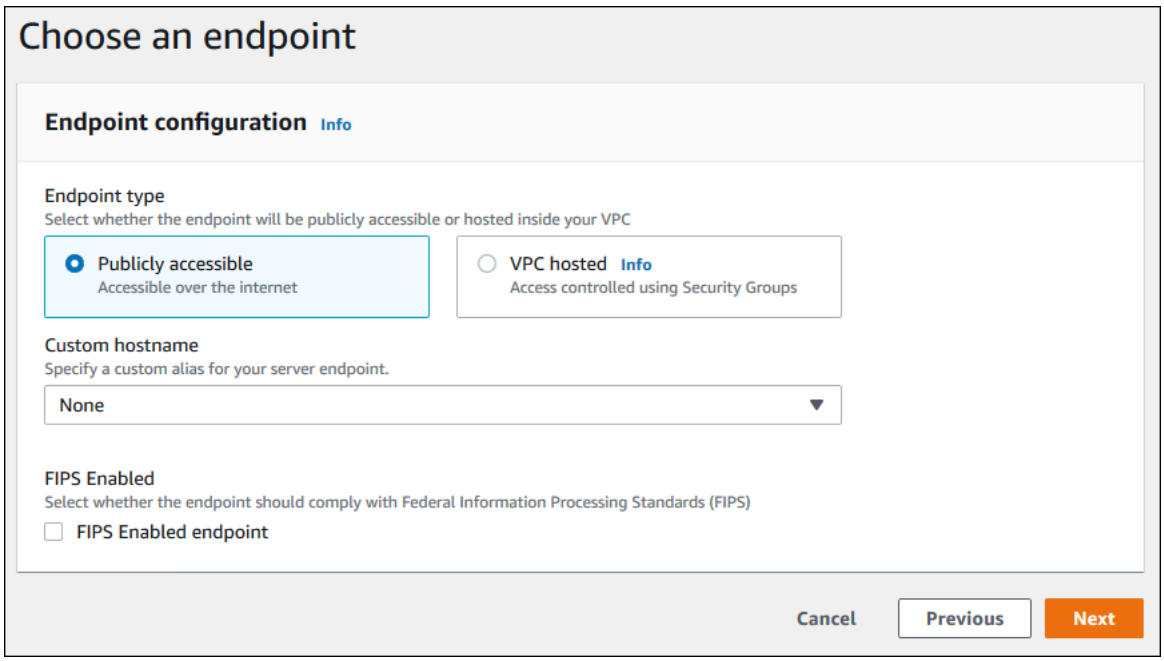
      によって提供されるサーバーホスト名を取得します AWS Transfer Family。サーバーホスト名の形式は `serverId.server.transfer.regionId.amazonaws.com` です。

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 有効 の場合、FIPS有効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 () に準拠していることを確認しますFIPS。

> **Note**
>
> FIPSが有効なエンドポイントは、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。の詳細については FIPS、連邦情報処理標準 (FIPS) 140-2 を参照してください。

d. [Next (次へ)] を選択します。



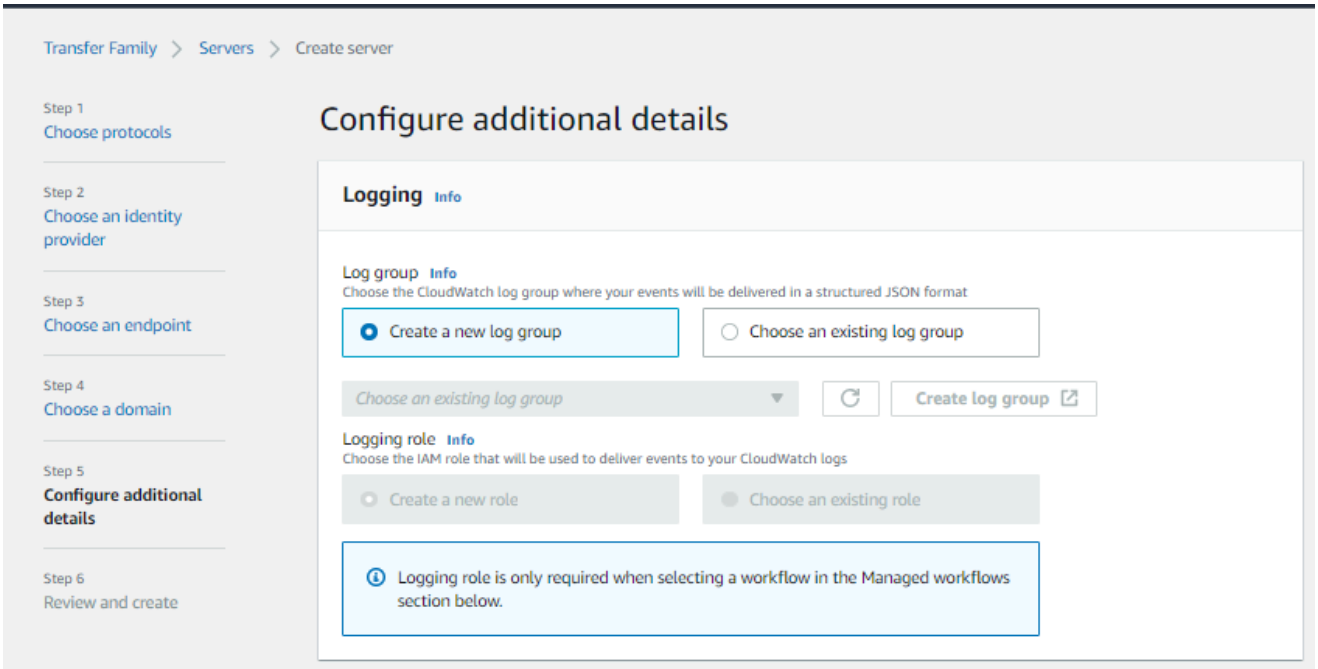
6. ドメインの選択ページで、選択したプロトコルを介してデータを保存およびアクセスするために使用する AWS ストレージサービスを選択します。

- 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 を選択して、選択したプロトコルを介して Amazon EFS ファイルシステムにファイルを保存してアクセスします。

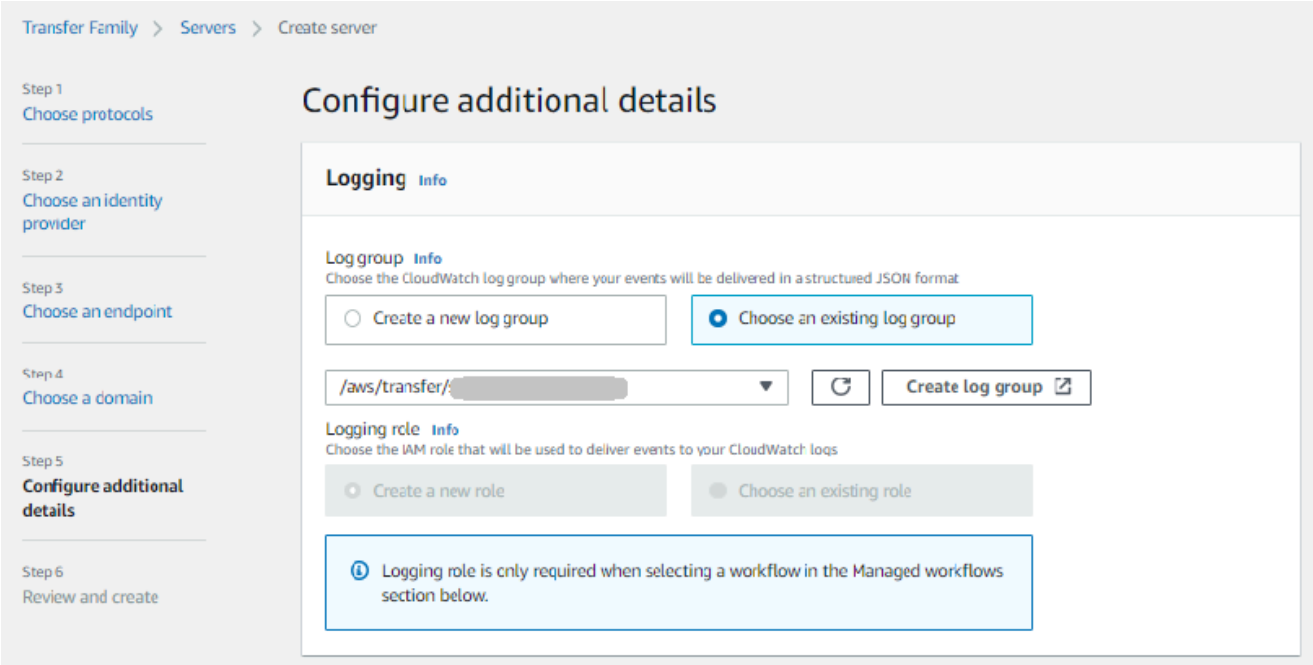
[Next (次へ)] を選択します。

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)。

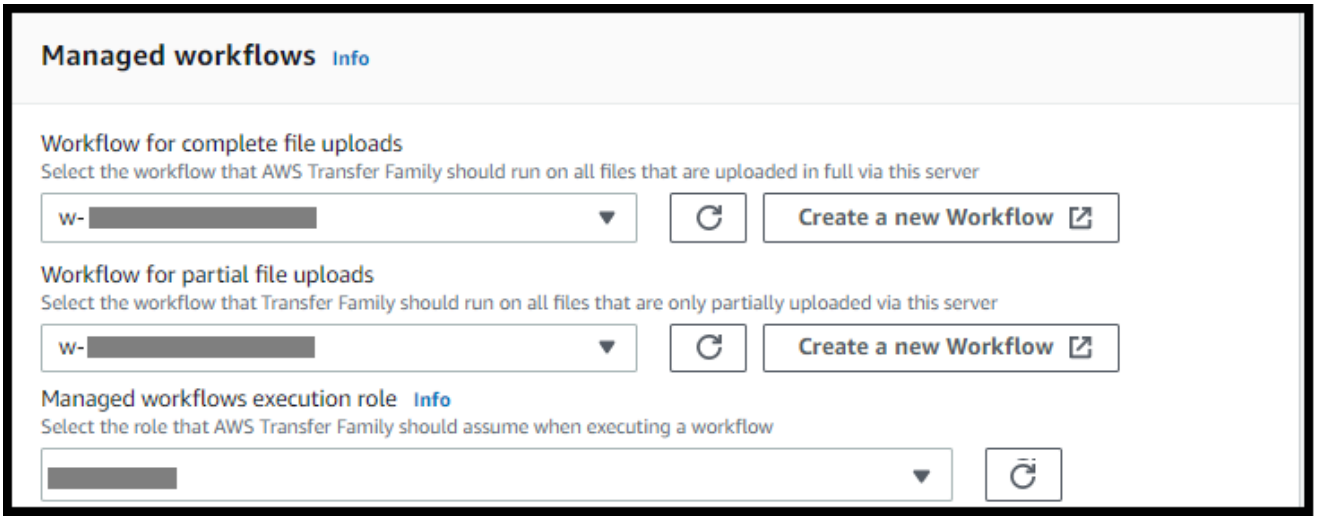


   既存のロググループを選択する場合は、 に関連付けられているロググループを選択する必要があります AWS アカウント。



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、 CloudWatch 「Logs でロググループを作成する」を参照してください。

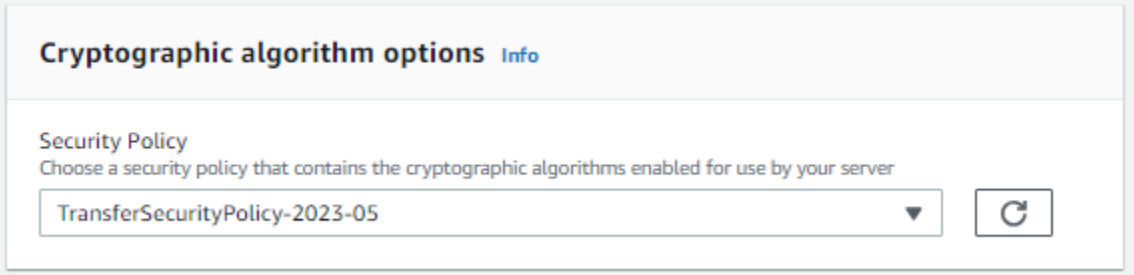
b. (オプション) マネージドワークフロー では、ワークフローの実行時に Transfer Family が引き受けるワークフロー IDs (および対応するロール) を選択します。完全なアップロード時に実行するワークフローと、部分的なアップロード時に実行するワークフローを選択できます。マネージドワークフローを使用したファイルの処理の詳細については、AWS Transfer Family マネージドワークフローを参照してください。



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**
>
> デフォルトでは、次にようになります。
>
> - FIPS 有効エンドポイントが選択されていない場合、TransferSecurityPolicy-2020-06セキュリティポリシーがサーバーにアタッチされます。
> - FIPS 有効エンドポイントが選択されている場合、TransferSecurityPolicy-FIPS-2020-06セキュリティポリシーがサーバーにアタッチされます。

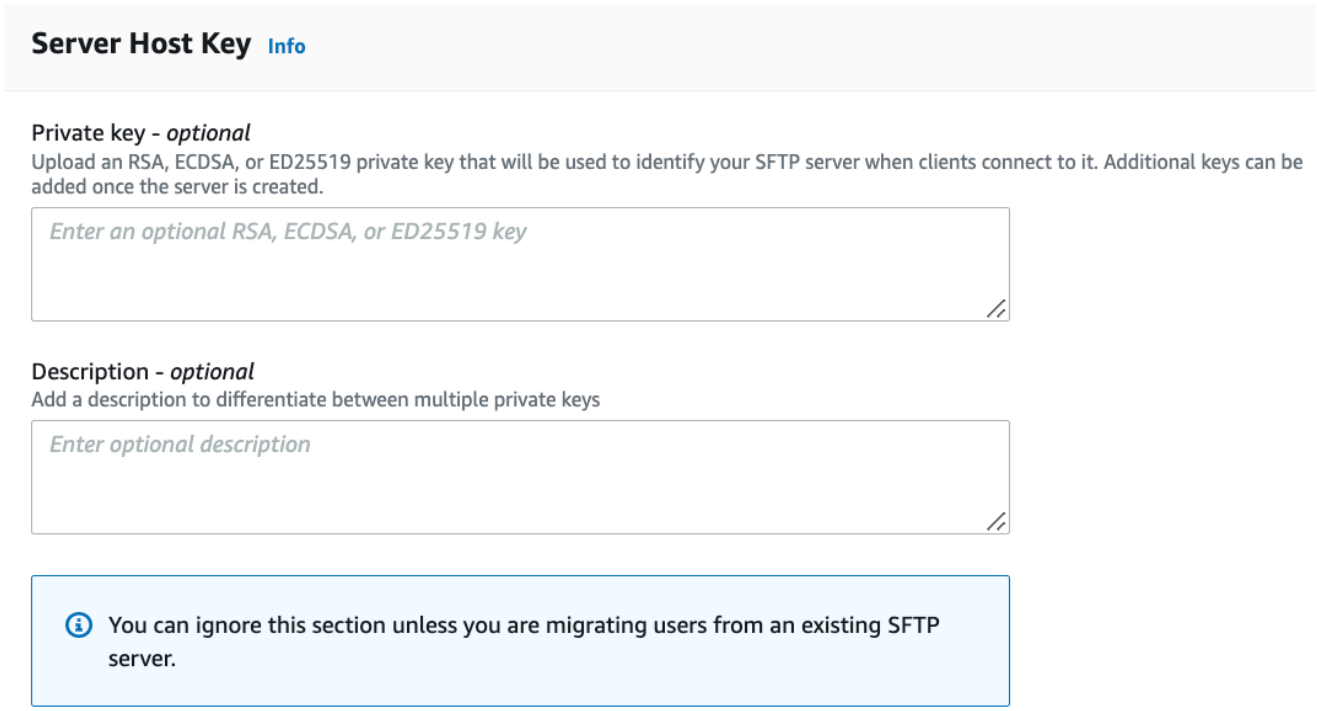
セキュリティポリシーの詳細については、「のセキュリティポリシー AWS Transfer Family」を参照してください。



d. （オプション) サーバーホストキー には、クライアントが 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SFTP。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。

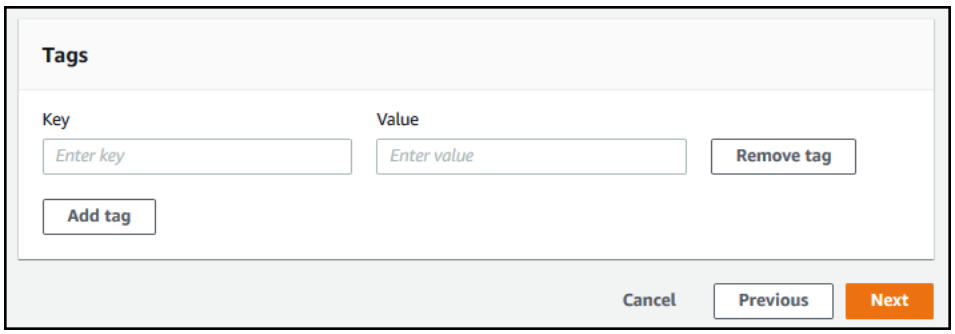
   サーバーを作成した後、追加のホスト キーを追加できます。複数のホストキーを持つことは、キーをローテーションする場合や、キーやキーなど、異なるタイプのRSAキーを持つ場合に便利ですECDSA。

> **Note**
>
> Server Host Key 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にのみ使用されます。

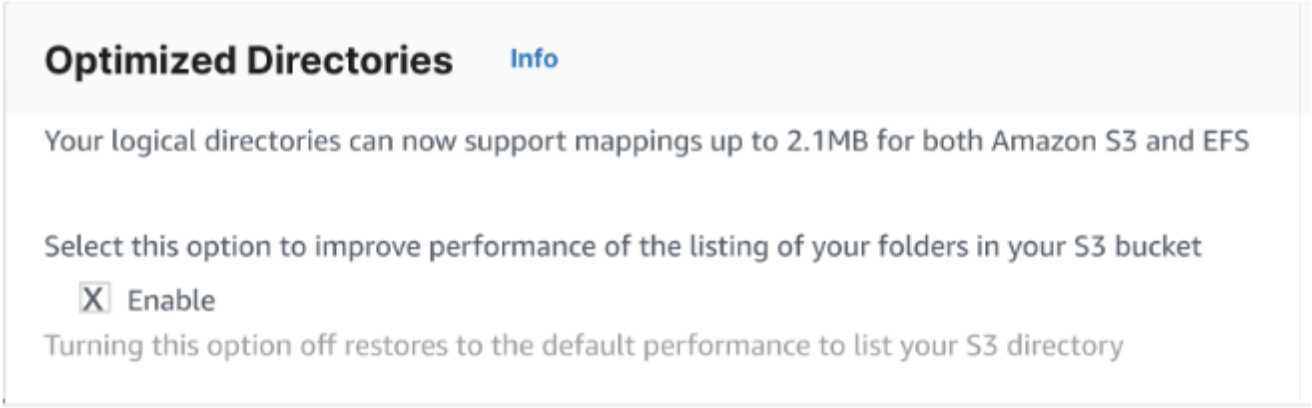


e. (オプション) [Tags] (タグ) の [Key] (キー) と [Value] (値) にキーバリューペアとして 1 つ以上のタグを入力して [Add tag] (タグの追加) を選択します。

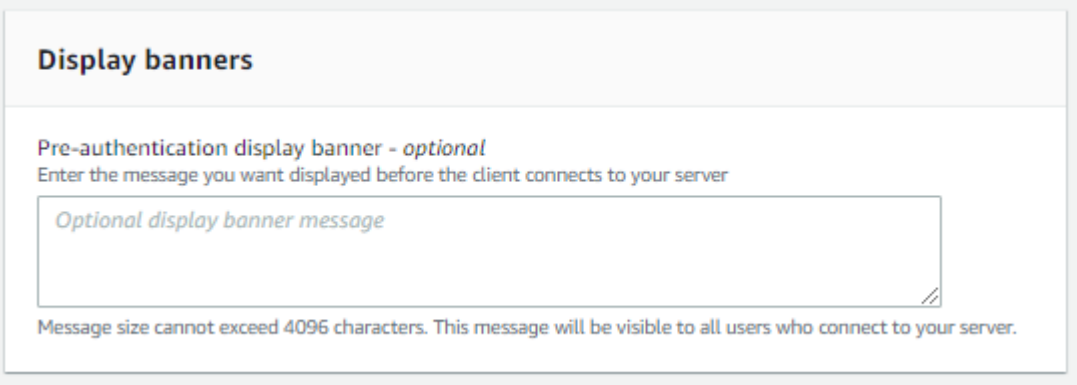
f. [Next (次へ)] を選択します。



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、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。

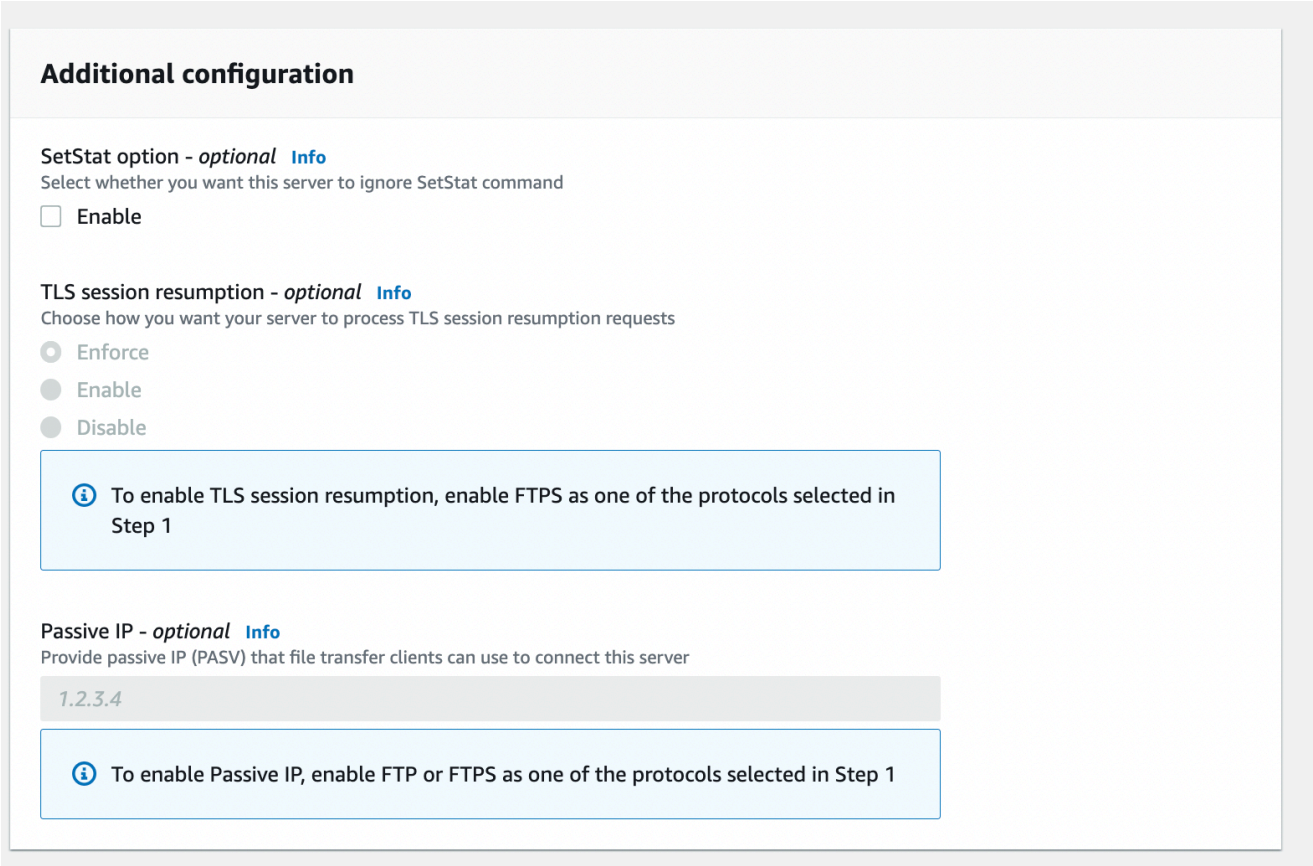


h. (オプション) 組織ポリシーや利用規約などのカスタマイズされたメッセージをエンドユーザーに表示するように AWS Transfer Family サーバーを設定します。[表示バナー] の [認証前表示バナー] テキスト ボックスに、ユーザーが認証する前に表示するテキスト メッセージを入力します。



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、「」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FTPSとして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FTPとして FTPS または を有効にしている場合にのみ使用できます。



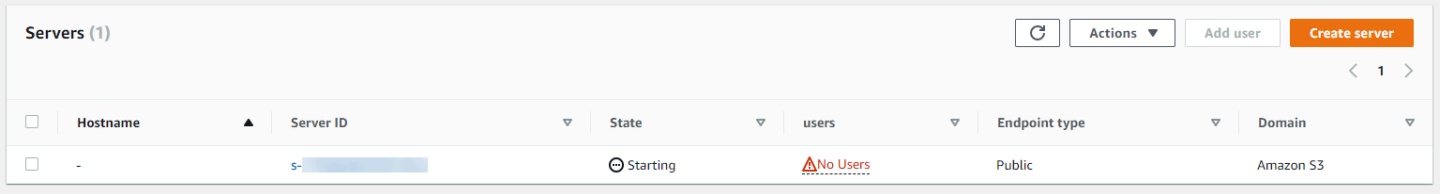
8. [Review and create] (確認と作成) で選択内容を確認します。

   - 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。



FTPS対応サーバーを作成する

SSL (FTPS) 経由のファイル転送プロトコルは、 の拡張機能ですFTP。Transport Layer Security (TLS) と Secure Sockets Layer (SSL) の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 では、制御チャネル接続とデータチャネル接続の両方を同時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. で AWS Transfer Family コンソールを開きhttps://console.aws.amazon.com/transfer/、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. プロトコルの選択 で、 を選択しますFTPS。

   サーバー証明書 では、 AWS Certificate Manager （ ACM) に保存されている証明書を選択します。この証明書はFTPS、クライアントが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。次に次へ 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 にインポートするにはACM、AWS Certificate Manager 「 ユーザーガイド」の「 に証明書をインポートACMする」を参照してください。

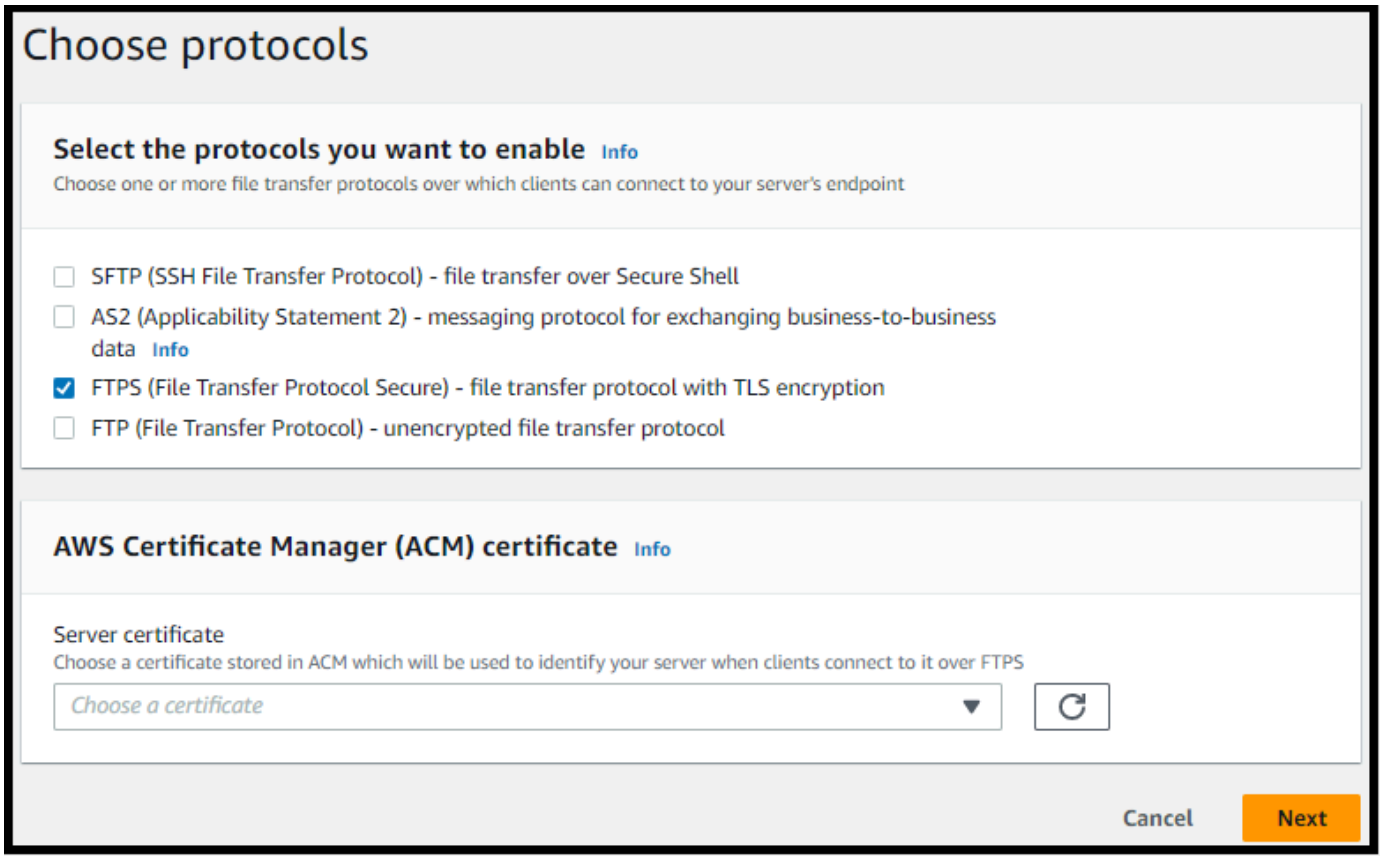
   プライベート IP アドレスFTPS経由でプライベート証明書を使用するようにリクエストするには、 AWS Certificate Manager ユーザーガイドの「プライベート証明書のリクエスト」を参照してください。

   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)

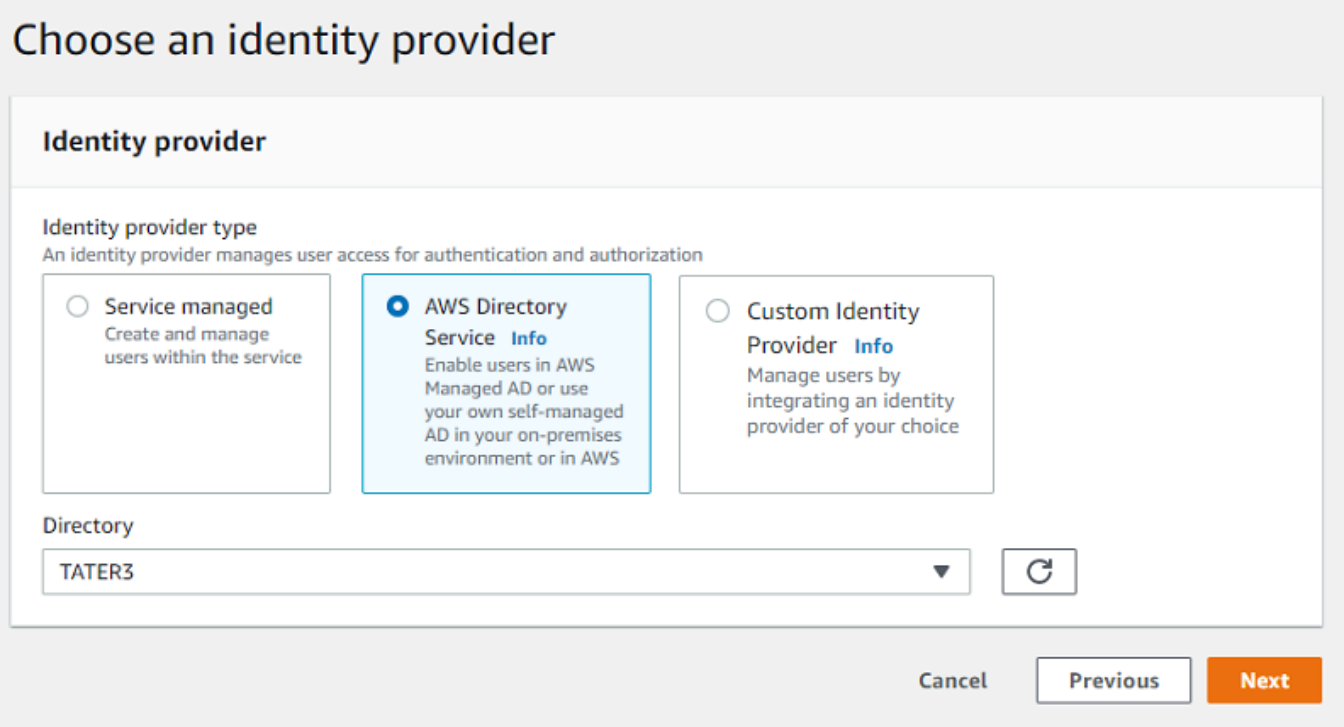
> **Note**
>
> 証明書は、 FQDNまたは IP アドレスを指定し、発行者に関する情報を含む有効な SSL/TLS X.509 バージョン 3 証明書である必要があります。



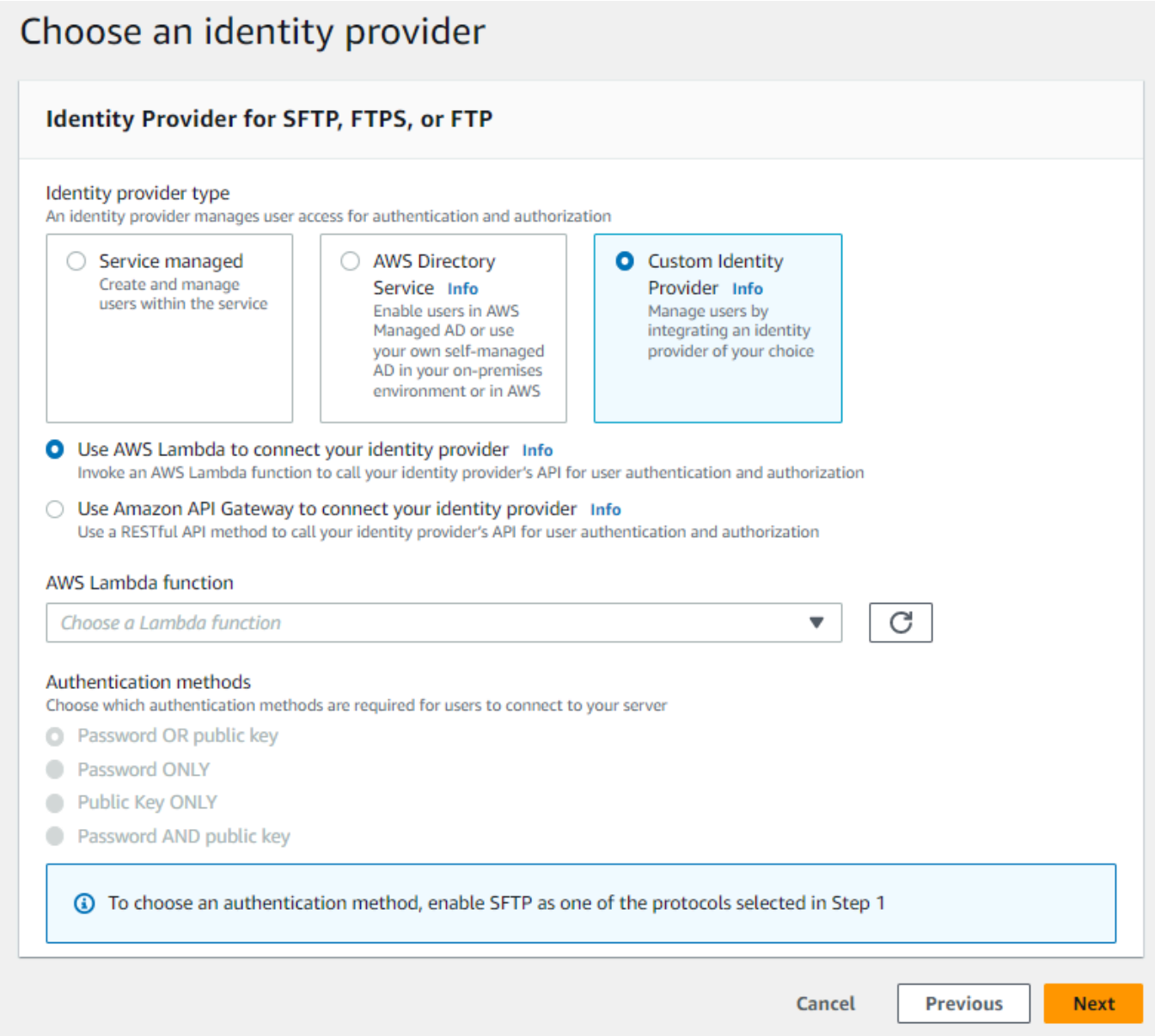
3. [Choose an identity provider] (ID プロバイダーの選択) で、ユーザーアクセスの管理に使用したい ID プロバイダーを選択します。次のオプションがあります。

   - 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 を使用して ID プロバイダーを接続する – Lambda 関数でバックアップされた API Gateway メソッドを作成して、ID プロバイダーとして使用できます。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. [Next (次へ)] を選択します。
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**
>
> FTPS Transfer Family の 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。

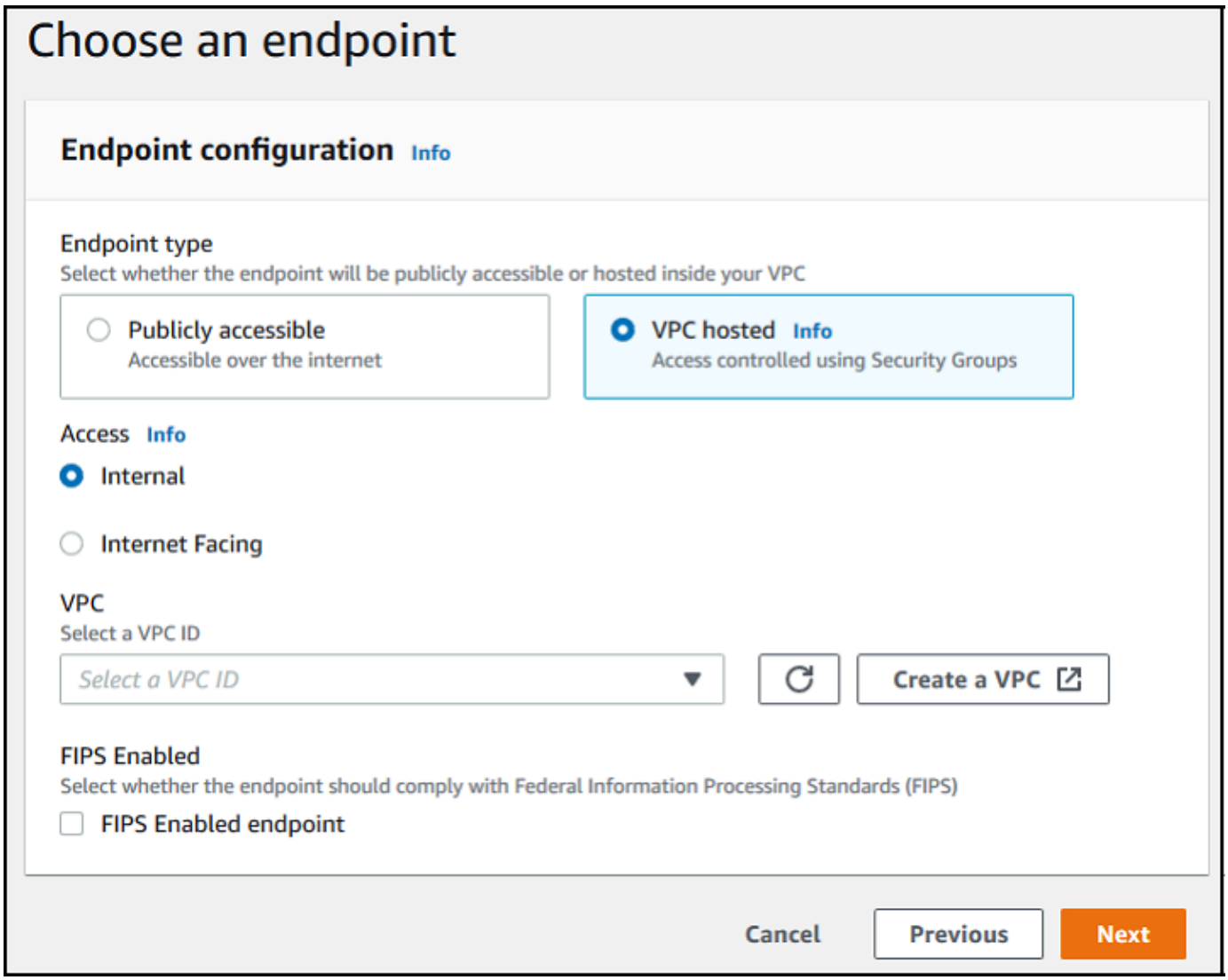
a. エンドポイントタイプ では、ホストVPCされたエンドポイントタイプを選択して、サーバーのエンドポイントをホストします。VPC ホストされたエンドポイントの設定については、「」を参照してくださいVirtual Private Cloud でサーバーを作成する。

> **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 有効 の場合、FIPS有効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 () に準拠していることを確認しますFIPS。

> **Note**
>
> FIPSが有効なエンドポイントは、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。の詳細についてはFIPS、連邦情報処理標準 (FIPS) 140-2 を参照してください。

c. [Next (次へ)] を選択します。



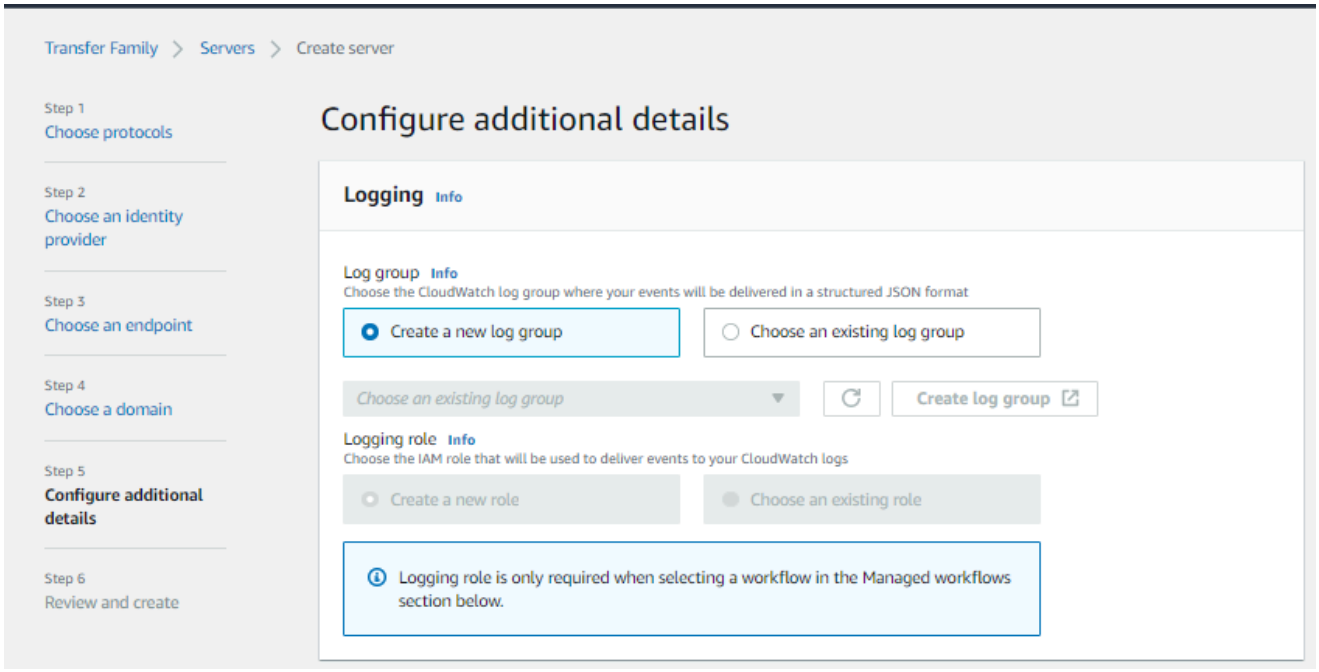
6. ドメインの選択ページで、選択したプロトコルを介してデータを保存およびアクセスするために使用する AWS ストレージサービスを選択します。

   - 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 を選択して、選択したプロトコルを介して Amazon EFS ファイルシステムにファイルを保存してアクセスします。

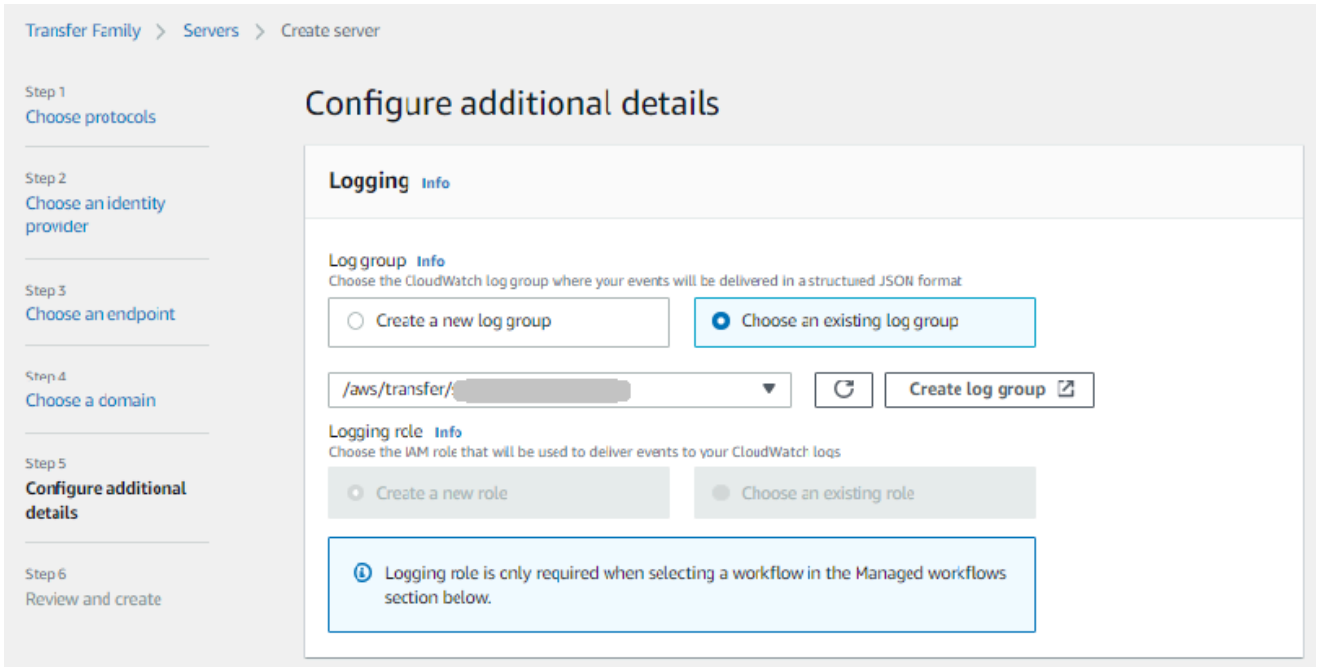
   [Next (次へ)] を選択します。

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)。

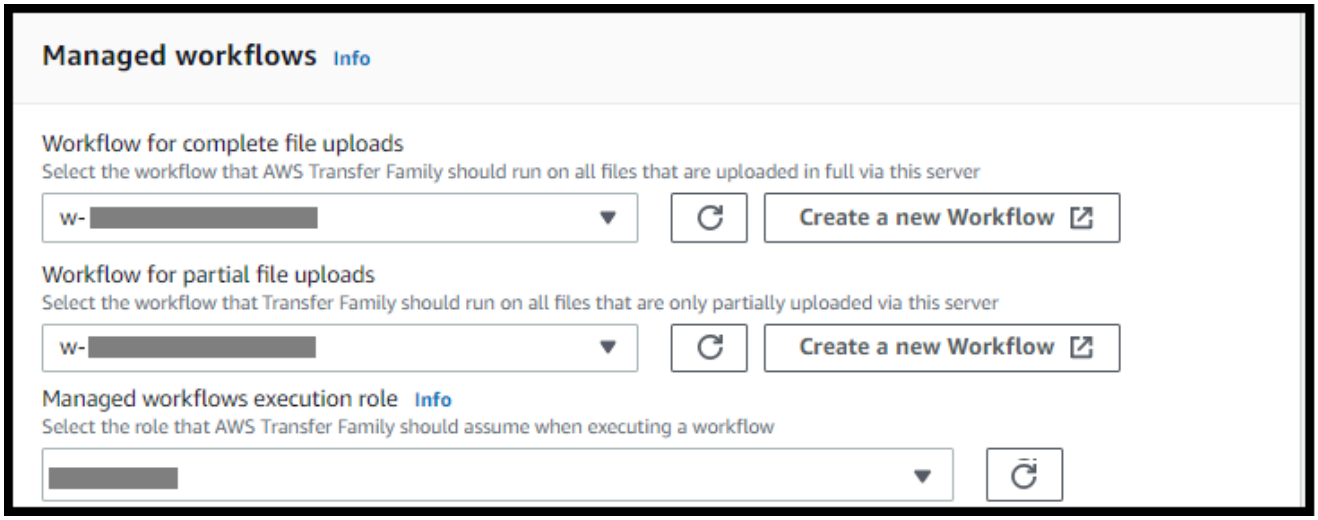


既存のロググループを選択する場合は、 に関連付けられているロググループを選択する必要があります AWS アカウント。



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、 CloudWatch 「 Logs でロググループを作成する」を参照してください。

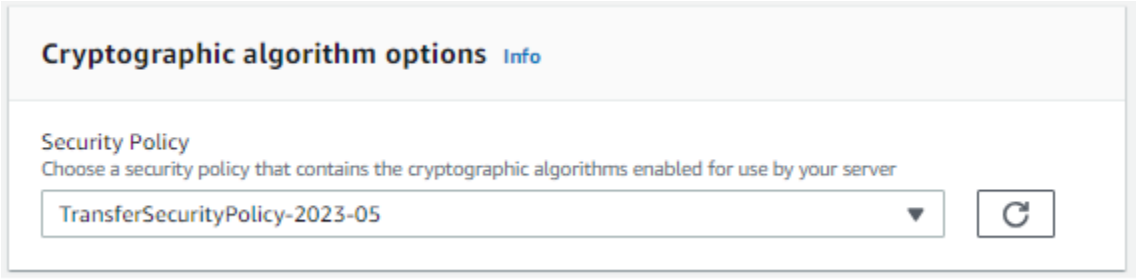
b. (オプション) マネージドワークフロー では、ワークフローの実行時に Transfer Family が引き受けるワークフロー IDs (および対応するロール) を選択します。完全なアップロード時に実行するワークフローと、部分的なアップロード時に実行するワークフローを選択できます。マネージドワークフローを使用したファイルの処理の詳細については、AWS Transfer Family マネージドワークフローを参照してください。



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。

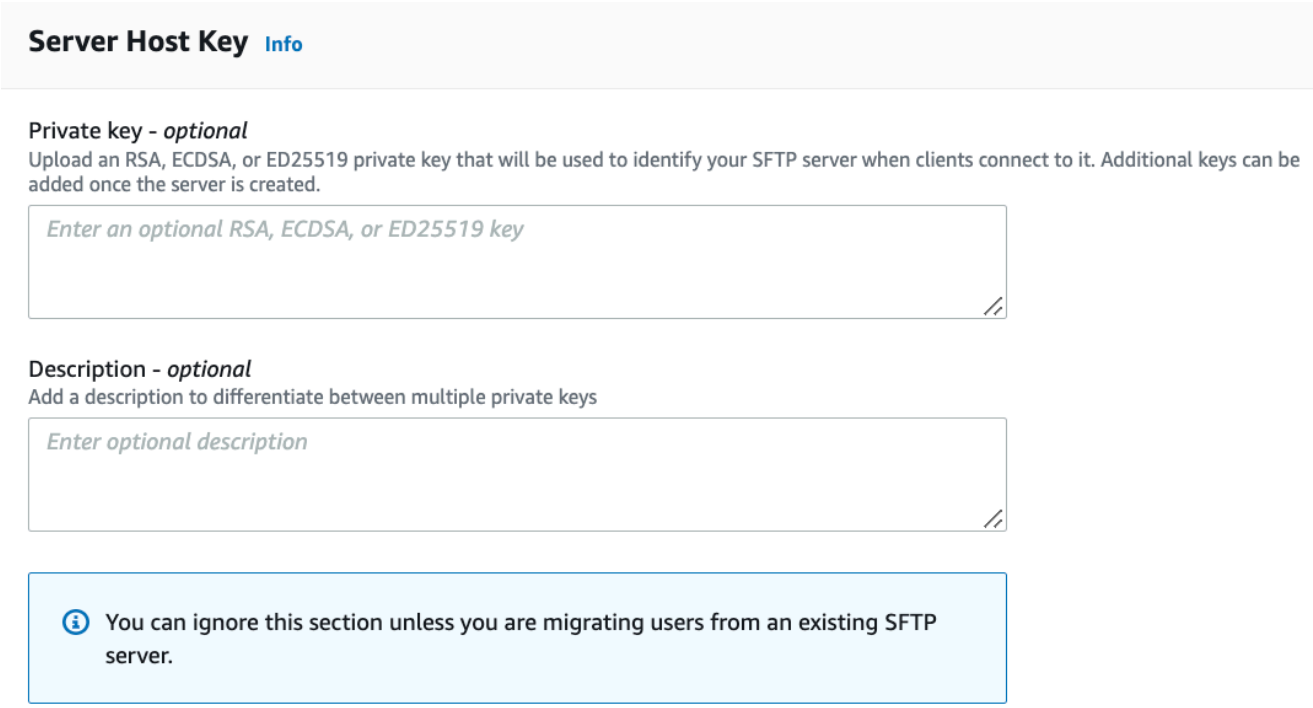
> **Note**
>
> デフォルトでは、次にようになります。
>
> - FIPS 有効エンドポイントが選択されていない場合、`TransferSecurityPolicy-2020-06`セキュリティポリシーがサーバーにアタッチされます。
> - FIPS 有効エンドポイントが選択されている場合、`TransferSecurityPolicy-FIPS-2020-06`セキュリティポリシーがサーバーにアタッチされます。

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



d. 「サーバーホストキー」は空白のままにしておきます。

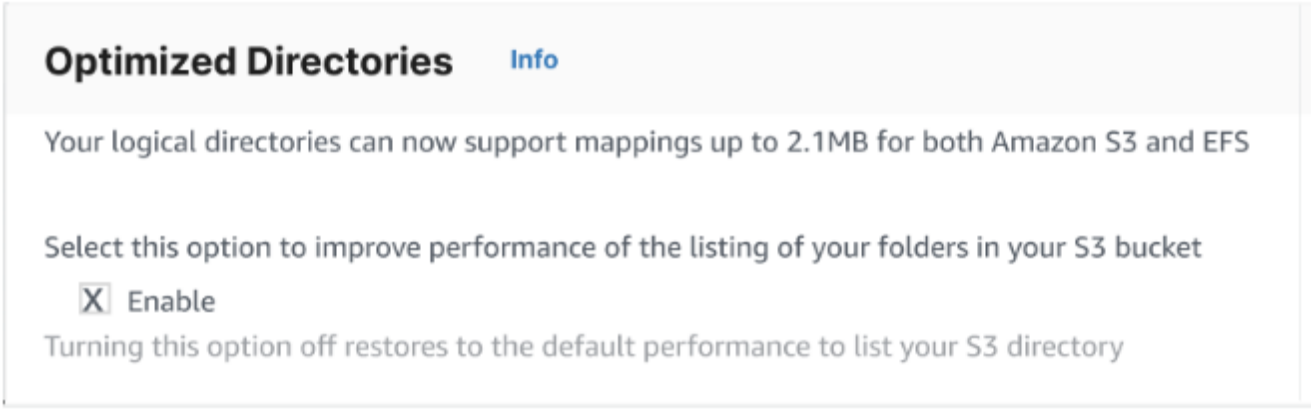
> **Note**
>
> Server Host Key セクションは、既存の SFTP対応サーバーからユーザーを移行する場合にのみ使用されます。



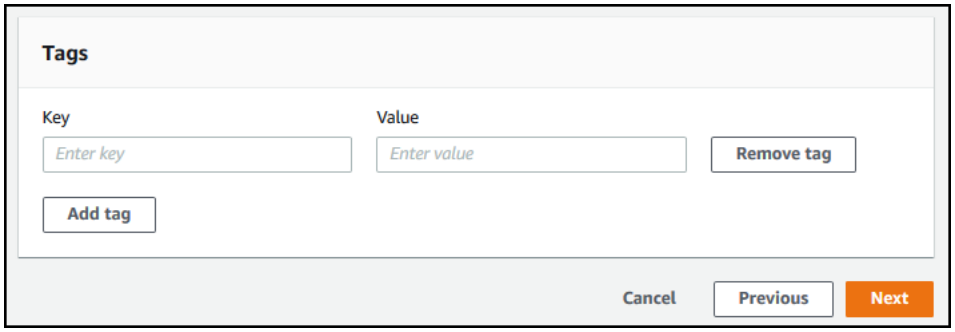
e. (オプション) [Tags] (タグ) の [Key] (キー) と [Value] (値) にキーバリューペアとして 1 つ以上のタグを入力して [Add tag] (タグの追加) を選択します。

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、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。

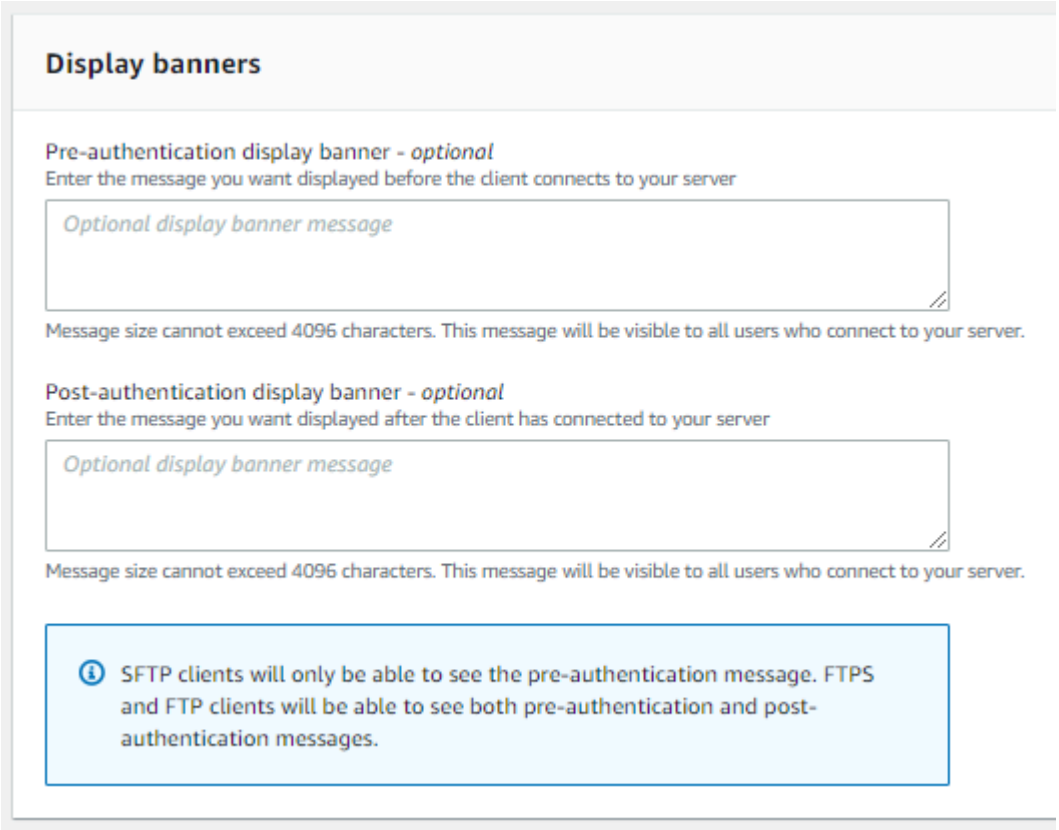


g. [Next (次へ)] を選択します。



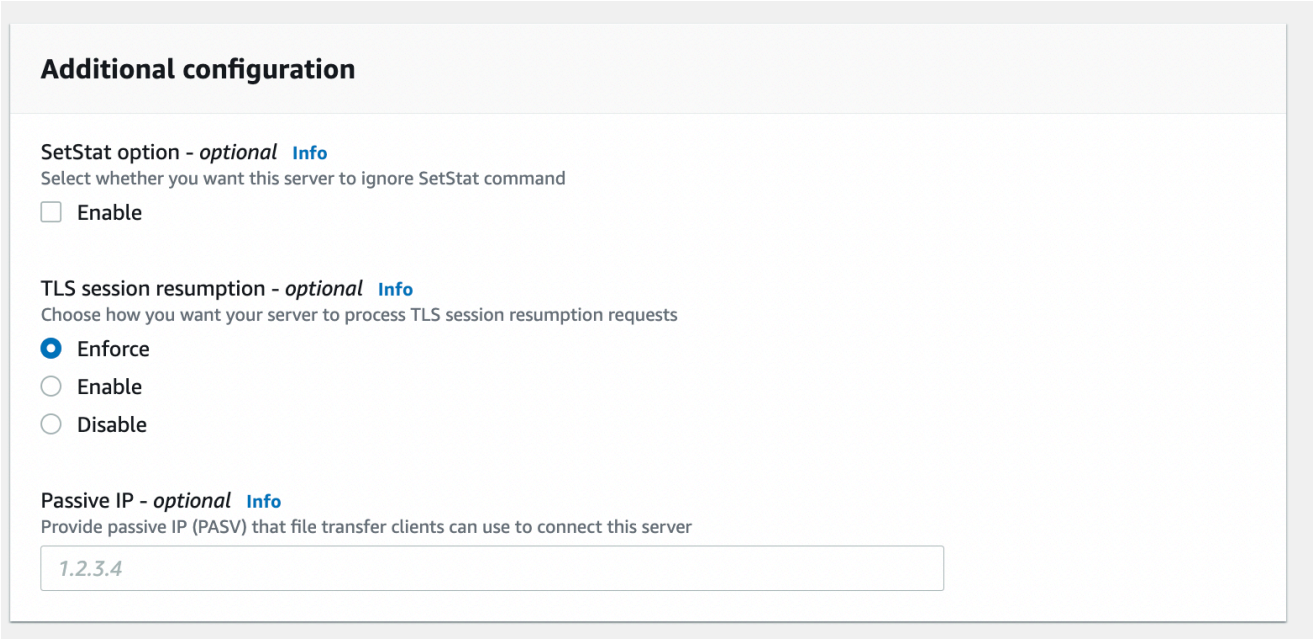
h. （オプション) AWS Transfer Family サーバーを設定して、組織のポリシーや利用規約などのカスタマイズされたメッセージをエンドユーザーに表示することができます。認証に成功したユーザーに、カスタマイズされた Message of the Day (MOTD) を表示することもできます。

[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セッションのコントロールとデータ接続の間でネゴシエートされたシークレット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。
- パッシブ IP : FTPと 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 IPv4、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、単一の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。



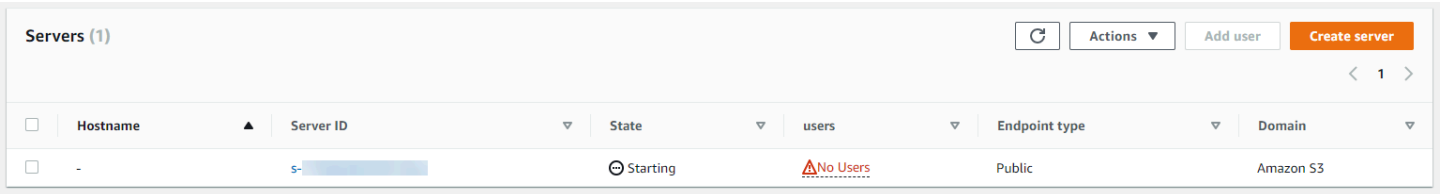
8. [Review and create] (確認と作成) で選択内容を確認します。

   - 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。

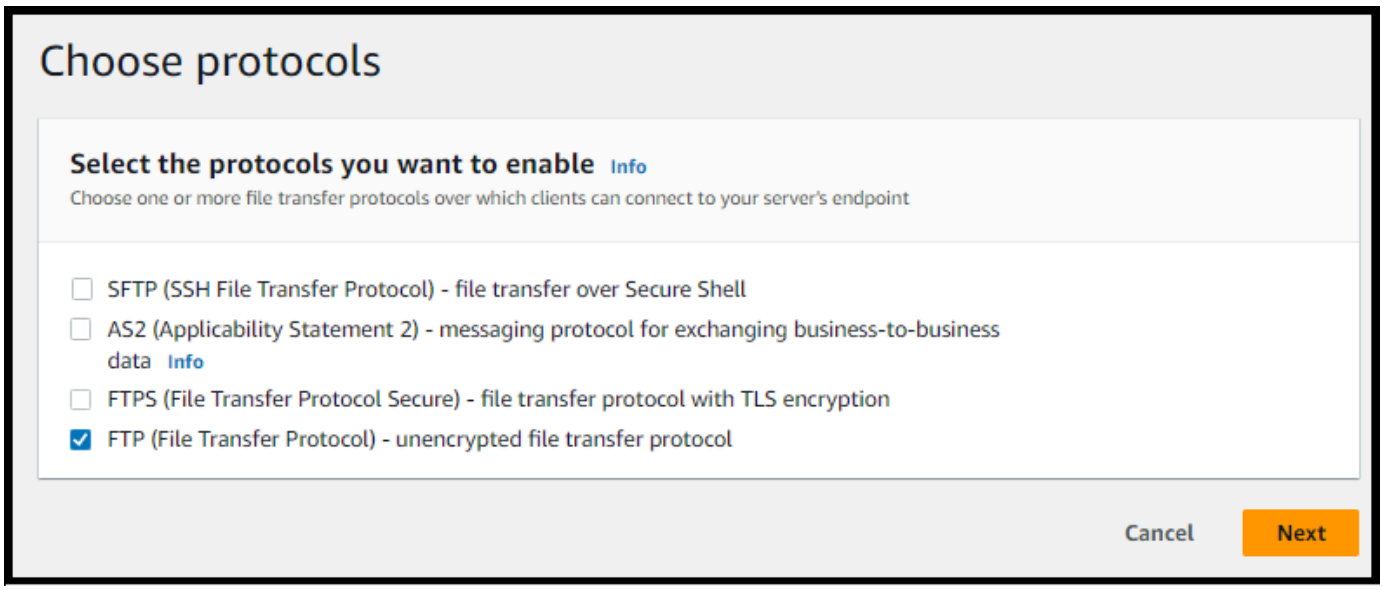
FTP対応サーバーを作成する

ファイル転送プロトコル (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 はクリアテキストを使用し、トラフィックの暗号化をサポートしていません。

> **Note**
>
> を有効にするときはFTP、VPCホストされた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。サーバーがパブリックネットワークをデータを横断する必要がある場合は、 SFTPや などの安全なプロトコルを使用する必要がありますFTPS。

FTP対応サーバーを作成するには

1. で AWS Transfer Family コンソールを開きhttps://console.aws.amazon.com/transfer/、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. プロトコルの選択 で、 を選択しFTP、次へ を選択します。

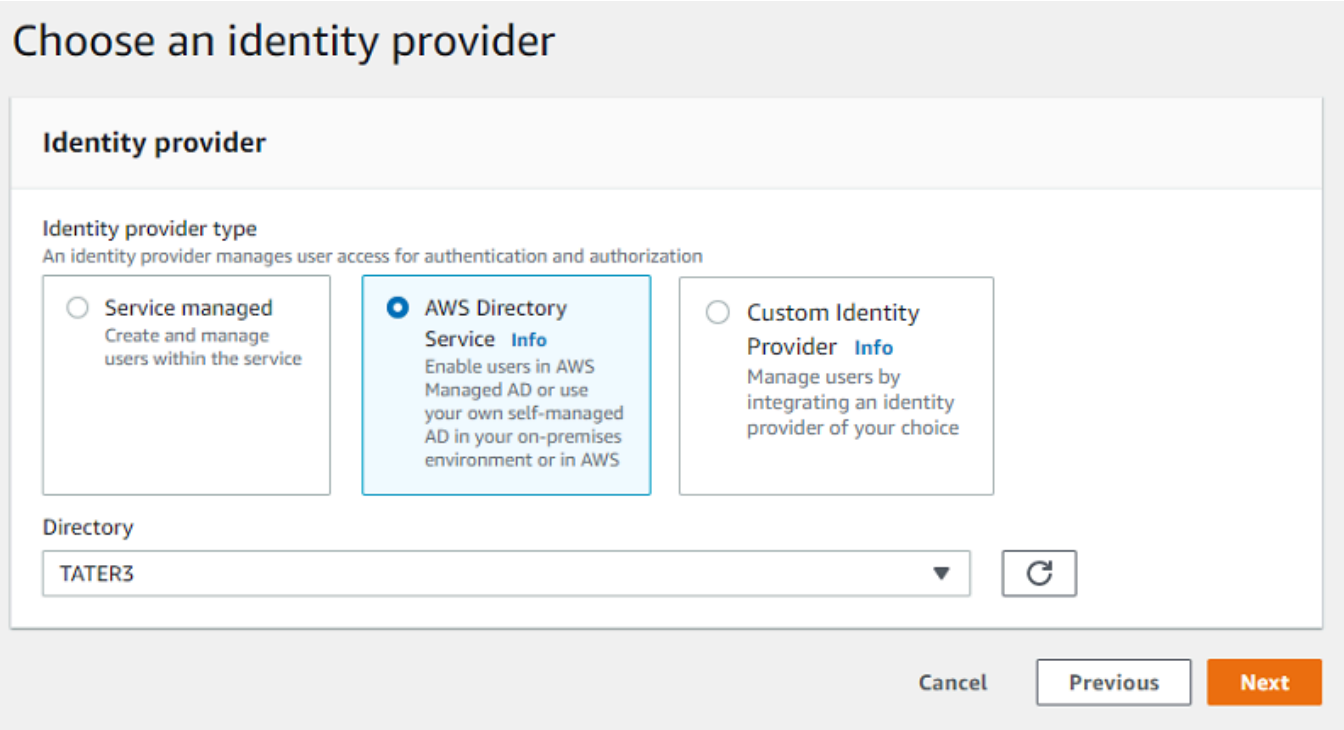


3. [Choose an identity provider] (ID プロバイダーの選択) で、ユーザーアクセスの管理に使用したい ID プロバイダーを選択します。次のオプションがあります。

   - 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

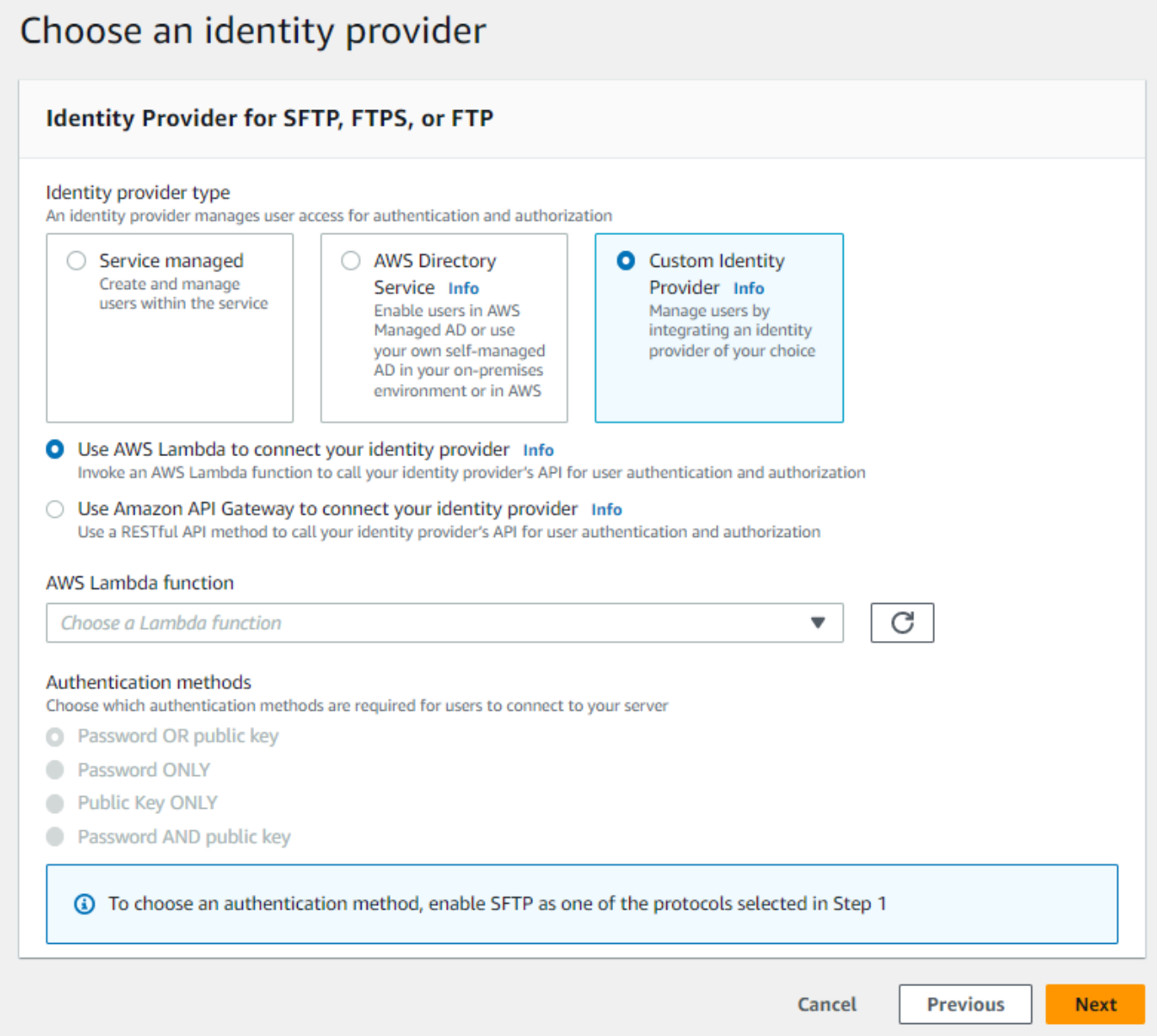
Microsoft AD ID プロバイダーの操作の詳細については、「」を参照してくださいAWS Directory Service ID プロバイダーの使用。

> **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 を使用して ID プロバイダーを接続する – Lambda 関数でバックアップされた API Gateway メソッドを作成して、ID プロバイダーとして使用でき

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. [Next (次へ)] を選択します。
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**
>
> FTP Transfer Family の 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。

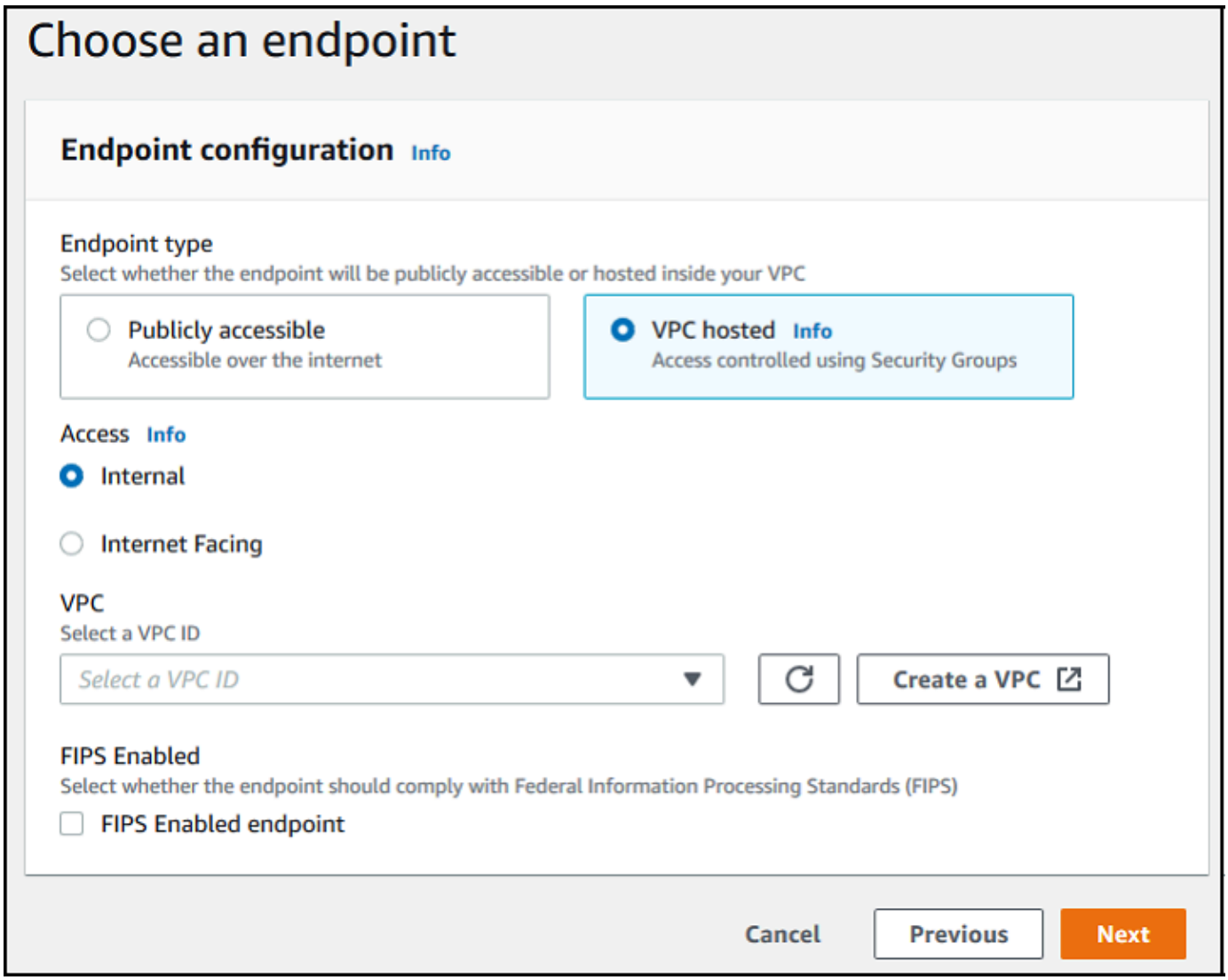
a. エンドポイントタイプ では、VPCホストを選択してサーバーのエンドポイントをホストします。VPC ホストされたエンドポイントの設定については、「」を参照してくださいVirtual Private Cloud でサーバーを作成する。

> **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 有効 の場合、FIPS有効エンドポイントチェックボックスはオフのままにします。

> **Note**
>
> FIPSが有効なエンドポイントはFTPサーバーではサポートされていません。

c. [Next (次へ)] を選択します。



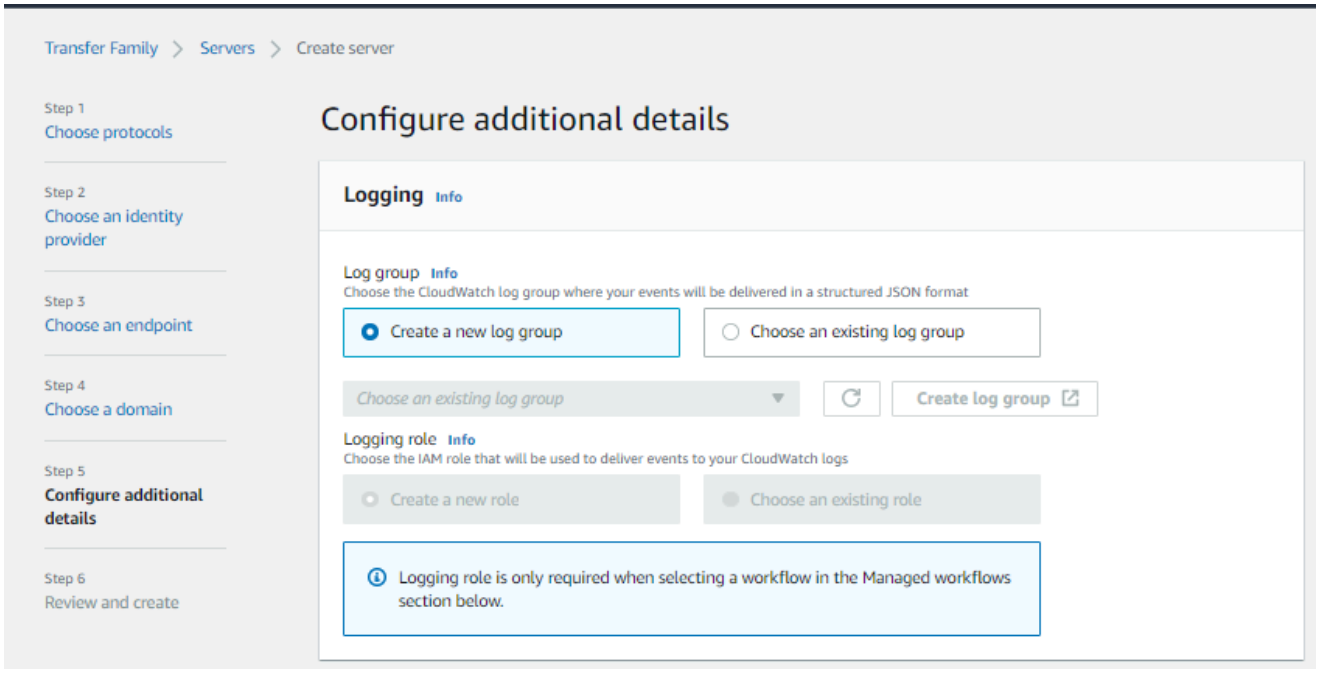
6. ドメインの選択ページで、選択したプロトコルを介してデータを保存およびアクセスするために使用する AWS ストレージサービスを選択します。

   - 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 を選択して、選択したプロトコルを介して Amazon EFS ファイルシステムにファイルを保存してアクセスします。

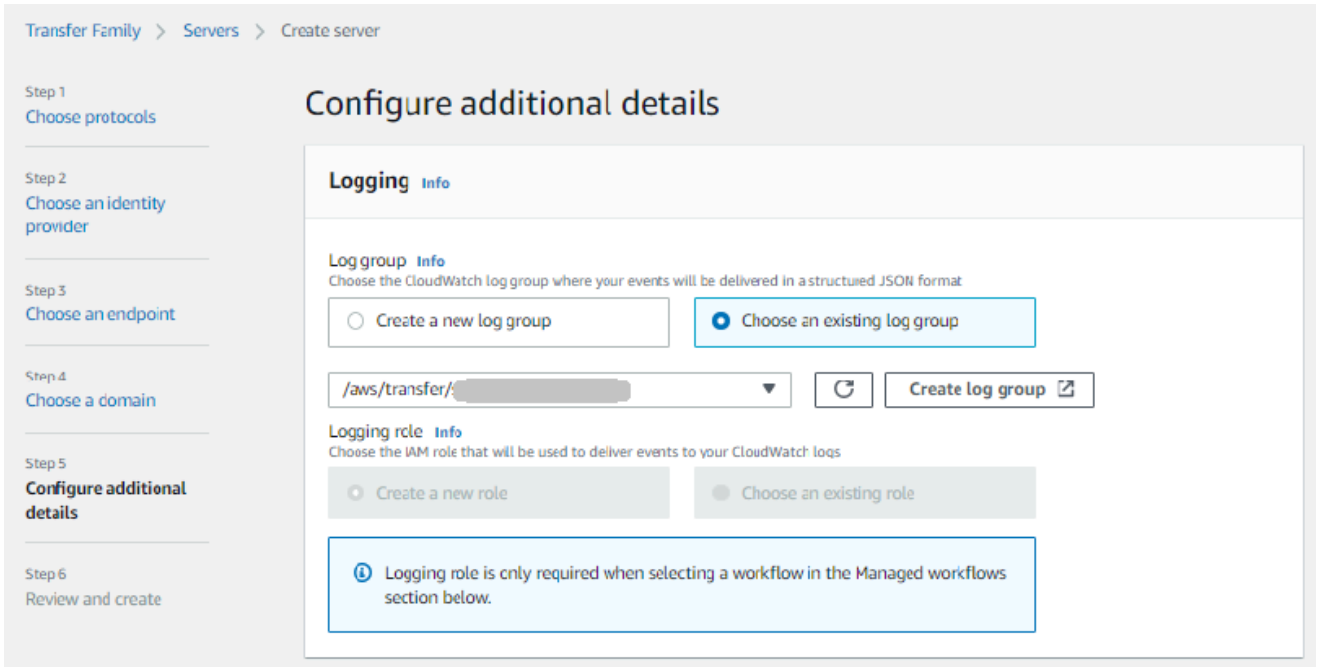
   [Next (次へ)] を選択します。

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)。

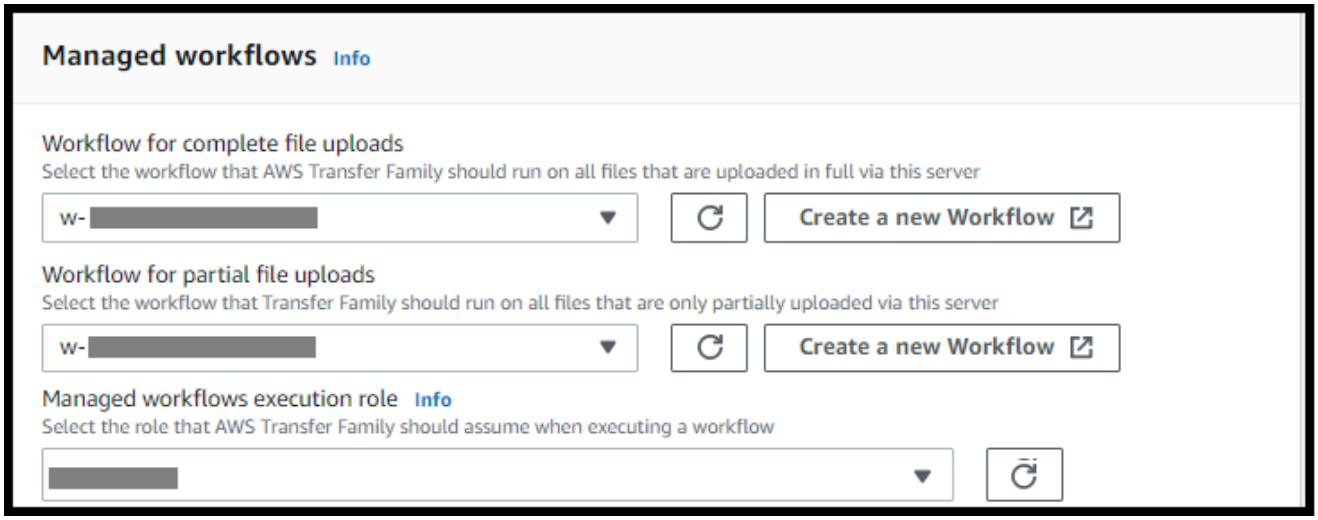


既存のロググループを選択する場合は、 に関連付けられているロググループを選択する必要があります AWS アカウント。



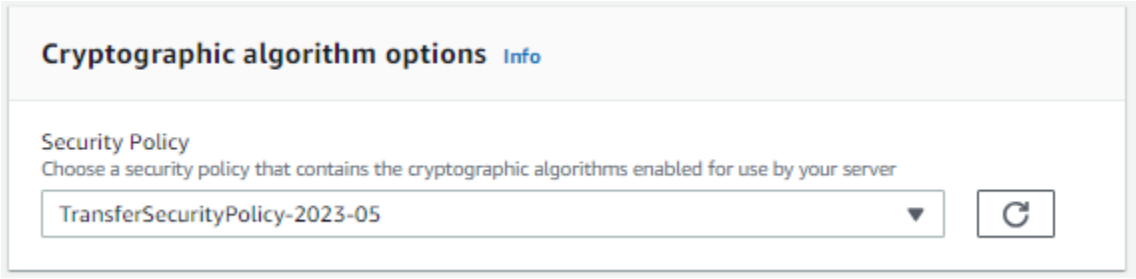
ロググループの作成 を選択すると、 CloudWatch コンソール (https://console.aws.amazon.com/cloudwatch/) がロググループの作成ページを開きます。詳細については、 CloudWatch 「 Logs でロググループを作成する」を参照してください。

b. (オプション) マネージドワークフロー では、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 IDs (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。



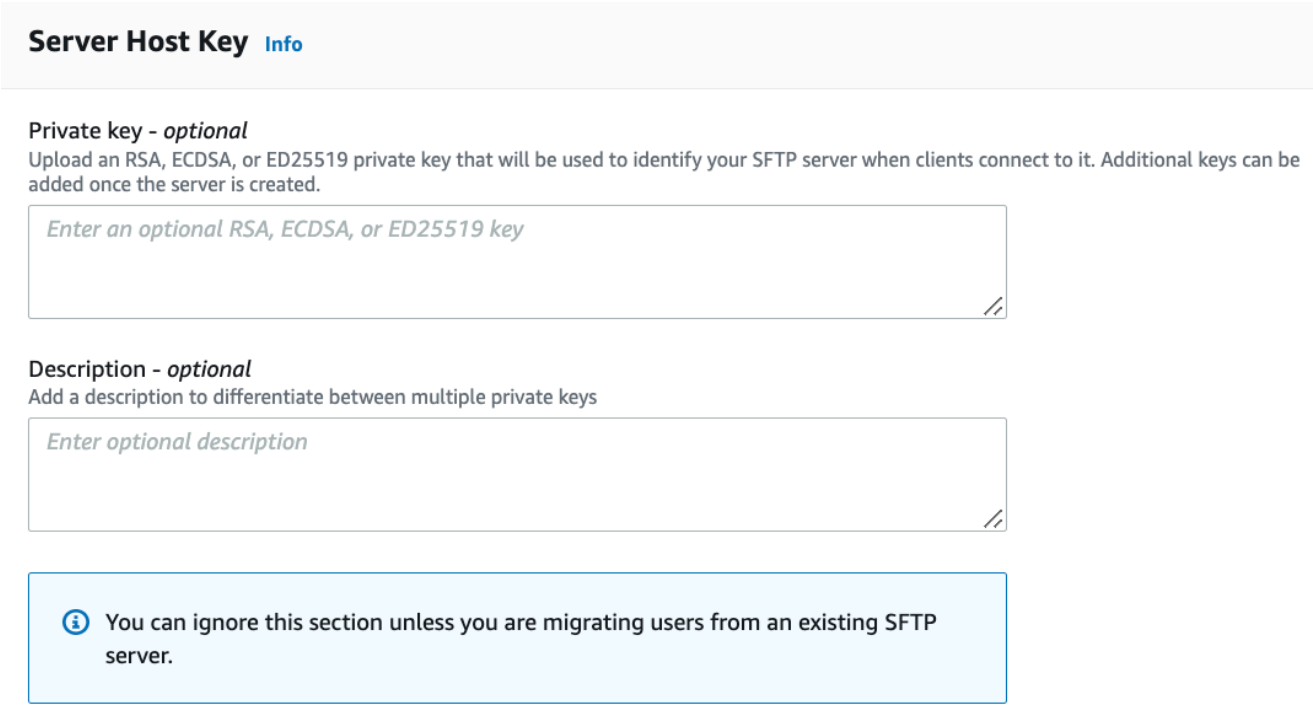
c. [Cryptographic algorithm options] (暗号化アルゴリズムオプション) で、サーバーで使用できる暗号化アルゴリズムを含むセキュリティポリシーを選択します。

> **Note**
>
> Transfer Family は、最新のセキュリティポリシーをFTPサーバーに割り当てます。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。

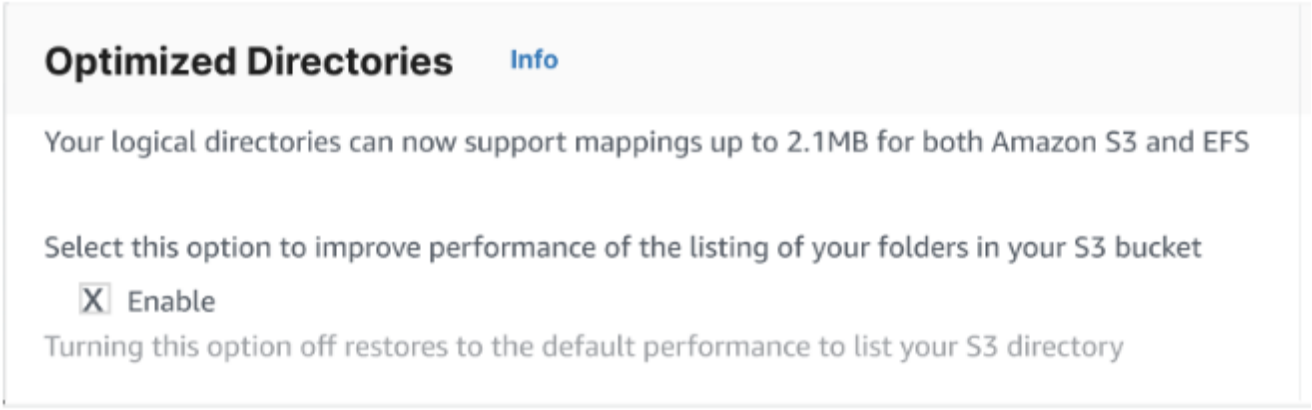
> **Note**
>
> Server Host Key セクションは、既存の SFTP対応サーバーからユーザーを移行する場合にのみ使用されます。



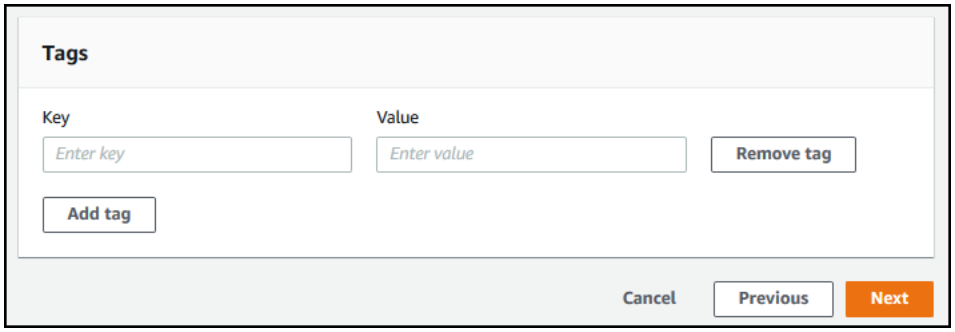
e. (オプション) [Tags] (タグ) の [Key] (キー) と [Value] (値) にキーバリューペアとして 1 つ以上のタグを入力して [Add tag] (タグの追加) を選択します。

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、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。

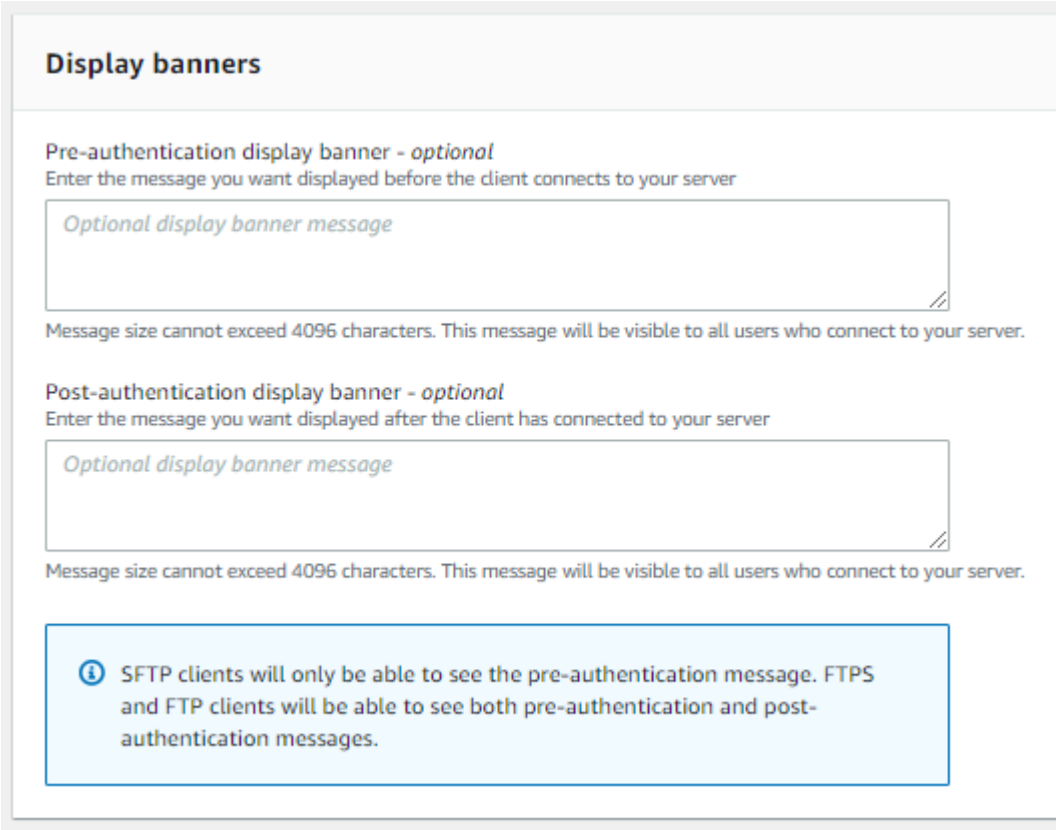


g. [Next (次へ)] を選択します。



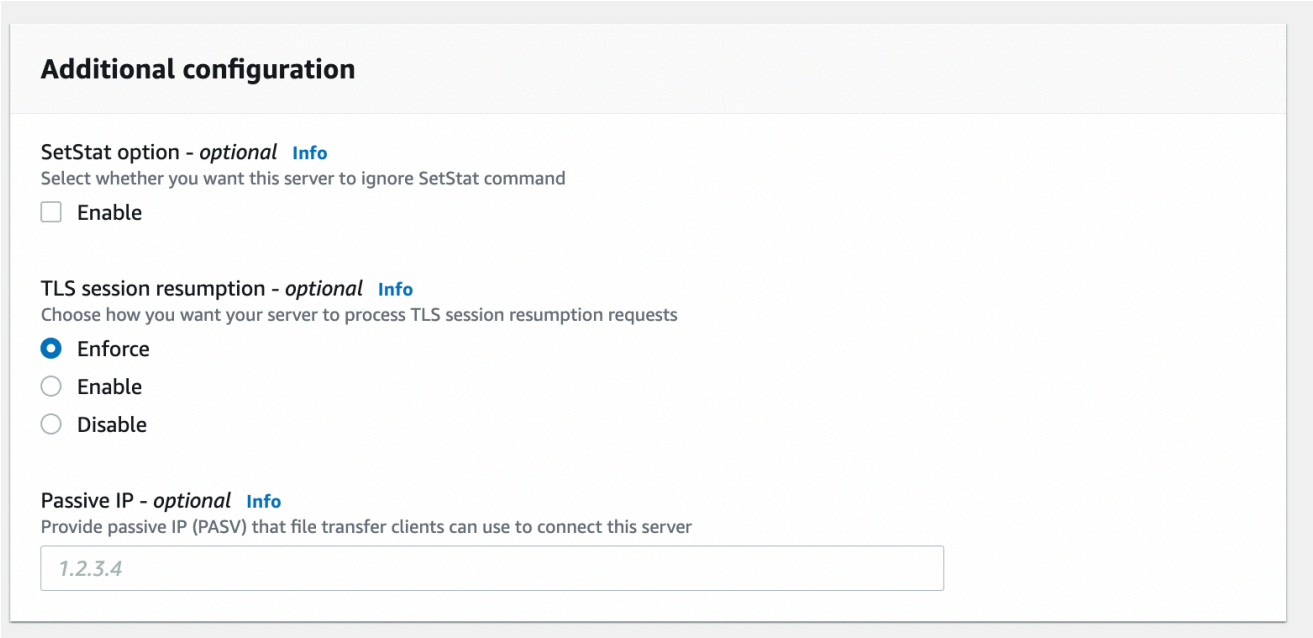
h. （オプション) AWS Transfer Family サーバーを設定して、組織のポリシーや利用規約などのカスタマイズされたメッセージをエンドユーザーに表示することができます。認証に成功したユーザーに、カスタマイズされた Message of the Day (MOTD) を表示することもできます。

[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セッションのコントロールとデータ接続の間でネゴシエートされたシークレット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。
- パッシブ 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IPv4、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、単一の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。



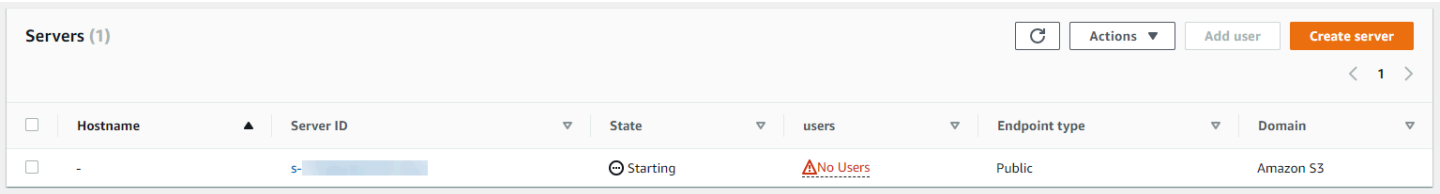
8. [Review and create] (確認と作成) で選択内容を確認します。

   - 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

サーバーのエンドポイントを仮想プライベートクラウド (VPC) 内にホストして、パブリックインターネットを経由することなく Amazon S3 バケットまたは 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 日以前にアカウントで を使用してサーバーを作成していない場合、 をアカウントEndpointType=VPC_ENDPOINTで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でを使用してサーバーを既に作成している場合、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、「the section called “VPC_ の使用を中止するENDPOINT”」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は、 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 を使用するとVPC、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。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VPC制御できます。の詳細についてはVPCs、「Amazon ユーザーガイド」の「Amazon とはVPC」を参照してください。 VPC

次のセクションでは、 を作成してVPCサーバーに接続する手順について説明します。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セットアップします。
2. VPC エンドポイントVPCを介して 内にあるクライアントを使用して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。これにより、 以外のクライアントがサーバーVPC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、リクエストが許可されたアドレスのみから送信された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

- 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 VPC
- 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_ の使用を中止するENDPOINT
- AWS Transfer Family サーバーエンドポイントタイプを VPC_ENDPOINT から 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 VPC

次の手順では、 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。

内にサーバーエンドポイントを作成するには VPC

1. で AWS Transfer Family コンソールを開きますhttps://console.aws.amazon.com/transfer/。
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する 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、次に次へ 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。カスタム を選択した場合は、Amazon API Gateway エンドポイントと、エンドポイントにアクセスするための AWS Identity and Access Management （ IAM) ロールを指定します。そうすることで、ディレクトリサービスを統合して SFTP ユーザーの認証と認可を実行できます。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**
>
> FTP Transfer Family の および 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. エンドポイントタイプ では、ホストVPCされたエンドポイントタイプを選択してサーバーのエンドポイントをホストします。

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。内部アクセスのみVPCのために 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ていません。

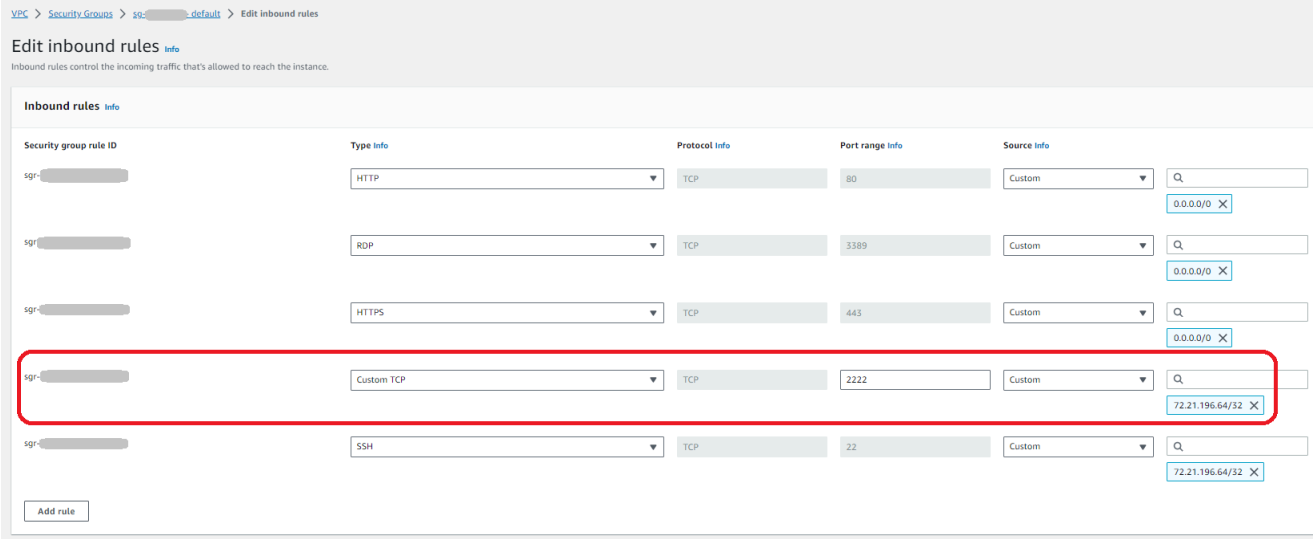
c. ではVPC、既存の VPC ID を選択するか、 VPCの作成 を選択して新しい を作成しますVPC。

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 セクションで、既存のセキュリティグループ ID を選択するIDsか、セキュリティグループの作成を選択して新しいセキュリティグループを作成します。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「 のセキュリティグループVPC」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**
>
> には、デフォルトのセキュリティグループVPCが自動的に付属しています。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、またはその両方を使用するように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 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 でカスタム TCPを選択し、ポート範囲 **2222**に を入力し、ソース でSSHポート 22 ルールと同じCIDR範囲を入力します。



f. (オプション) FIPS 有効 の場合、FIPS有効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 () に準拠していることを確認しますFIPS。

> **Note**
>
> FIPSが有効なエンドポイントは、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。の詳細については FIPS、連邦情報処理標準 (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。

- Transfer Family がロールを自動的に作成できるようにする新しいロールを作成します。ただし、新しいロールを作成するための適切なアクセス許可がある場合に限ります。IAM作成されるIAMロールは と呼ばれますAWSTransferLoggingAccess。

- 既存のロールを選択して、アカウントから既存のIAM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、 サービスが に設定されている信頼ポリシーを含める必要がありますtransfer.amazonaws.com。

  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、「のセキュリティポリシー AWS Transfer Family」を参照してください。

c. (オプション) サーバーホストキー には、クライアントが 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SFTP。

> **Note**
>
> このセクションでは、既存の SFTP対応サーバーからユーザーを移行する場合にのみ使用します。

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. [Next (次へ)] を選択します。

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。

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のデフォルトのセキュリティグループでソース IP アドレスが許可されているクライアントのみが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ホストエンドポイントで使用できるのFTPSは SFTPと 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. で AWS Transfer Family コンソールを開きますhttps://console.aws.amazon.com/transfer/。
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する 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。カスタム を選択した場合は、Amazon API Gateway エンドポイントと、エンドポイントにアクセスするための AWS Identity and Access Management （ IAM) ロールを指定します。そうすることで、ディレクトリサービスを統合して SFTP ユーザーの認証と認可を実行できます。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. エンドポイントタイプ では、ホストVPCされたエンドポイントタイプを選択してサーバーのエンドポイントをホストします。

   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、ユーザー AWS またはユーザーが所有できます。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) 必要なお客様は、Elastic IP アドレス経由で直接接続するか、 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EIP。 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、AWS GovCloud (US) 「 ユーザーガイド」のAWS GovCloud (US) 「リソースで Amazon Route 53 を設定する」を参照してください。

- 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。
- その他 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 。

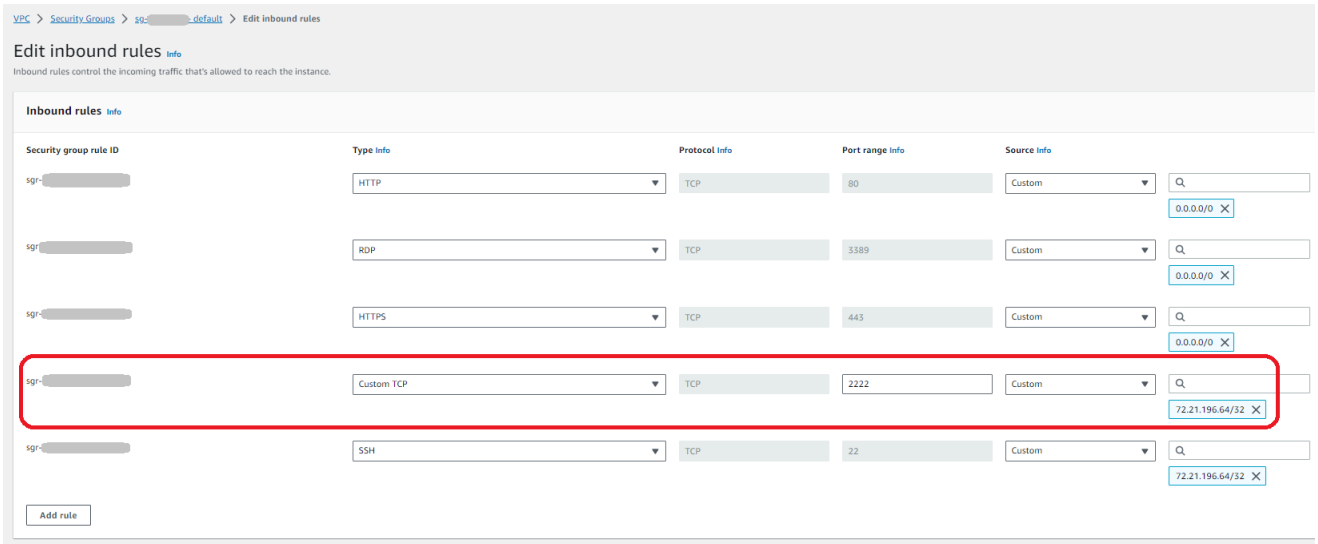
d. ではVPC、既存の VPC ID を選択するか、 VPCの作成 を選択して新しい を作成します VPC。

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 アドレス では、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 セクションで、既存のセキュリティグループ ID を選択するIDsか、セキュリティグループの作成を選択して新しいセキュリティグループを作成します。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「 のセキュリティグループVPC」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**
>
> には、デフォルトのセキュリティグループVPCが自動的に付属しています。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、またはその両方を使用するように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 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 にカスタム TCPを選択し、ポート範囲 **2222**に を入力し、ソースにSSHポート 22 ルールと同じCIDR範囲を入力します。



g. (オプション) FIPS 有効 の場合、FIPS有効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 () に準拠していることを確認しますFIPS。

> **Note**
>
> FIPSが有効なエンドポイントは、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。の詳細については FIPS、連邦情報処理標準 (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。

- Transfer Family がロールを自動的に作成できるようにする新しいロールを作成します。ただし、新しいロールを作成するための適切なアクセス許可がある場合に限ります。IAM作成されるIAMロールは と呼ばれますAWSTransferLoggingAccess。
- 既存のロールを選択して、アカウントから既存のIAM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、 サービスが に設定されている信頼ポリシーを含める必要がありますtransfer.amazonaws.com。

  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、「のセキュリティポリシー AWS Transfer Family」を参照してください。

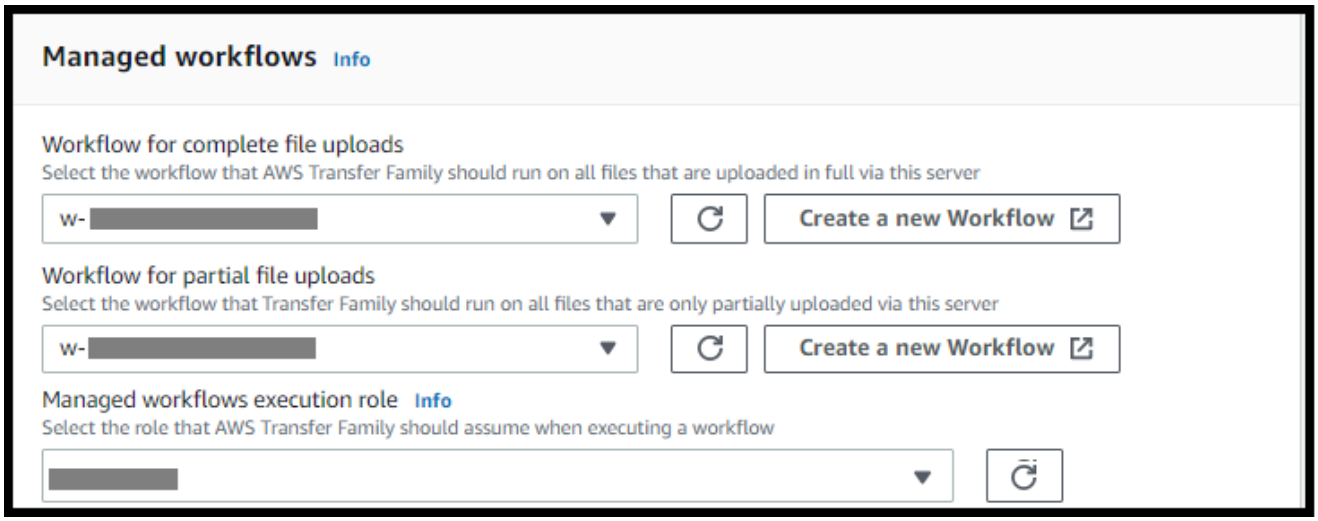
c. (オプション) サーバーホストキー には、クライアントが 経由でサーバーに接続するときにサーバーを識別するために使用される RSA、ED25519、またはECDSAプライベートキーを入力しますSFTP。

> **Note**
>
> このセクションでは、既存の SFTP対応サーバーからユーザーを移行する場合にのみ使用します。

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. [Next (次へ)] を選択します。

f. (オプション) マネージドワークフロー では、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 IDs (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。列パブリックIPv4アドレスが入力されると、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**
>
> のサーバーVPCがオンラインの場合、サブネットのみを変更でき、 UpdateServer を介してのみ変更できますAPI。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できる既存のサーバー (つまり、パブリックエンドポイントタイプ) がある場合は、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**
>
> に としてVPC表示される既存のサーバーがある場合はVPC_ENDPOINT、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す。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することもできます。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. で AWS Transfer Family コンソールを開きますhttps://console.aws.amazon.com/transfer/。
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. エンドポイントタイプの編集 で、VPCホストされた 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、ユーザー AWS またはユーザーが所有できます。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 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。

- その他 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. ではVPC、既存の VPC ID を選択するか、 VPCの作成 を選択して新しい を作成しますVPC。

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**
>
> 最大 3 つのアベイラビリティーゾーンが必要だが、十分な空きがない場合は、VPC コンソール () で作成しますhttps://console.aws.amazon.com/vpc/。
> 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、サーバーのエンドポイントタイプが に変更されていることに注意してくださいVPC。

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_ の使用を中止するENDPOINT

AWS Transfer Family は、EndpointType=VPC_ENDPOINT新しい AWS アカウントの でサーバーを作成する機能を停止しています。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 に更新VPC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 「」の「 はVPCセキュリティグループと Elastic IP アドレス SFTPをサポートします。この新しいエンドポイントは、機能豊富でコスト効率が高く、 PrivateLink 料金はかかりません。詳細については、AWS PrivateLink 「 の料金」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 の使用許可リスト」を参照して AWS Transfer Family 、 for SFTP Server のブログ記事を参照してください。

このエンドポイントを共有VPC環境でホストすることもできます。詳細については、「」でAWS Transfer Family 共有サービスVPC環境 がサポートされるようになりました。

に加えてSFTP、 VPC EndpointType を使用して FTPSと を有効にできますFTP。これらの機能とFTPS/FTP サポートを に追加する予定はありませんEndpointType=VPC_ENDPOINT。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、 AWS CLI、API、または を使用してSDKs、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 に更新する VPC」を参照してください。

質問がある場合は、 AWS サポート 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FTPSやFTPサポートを EndpointType=VPC_ に追加する予定はありません ENDPOINT。 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。

その他のご質問は、 AWS サポート 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 に更新する VPC

AWS Management Console、 AWS CloudFormation、または Transfer Family を使用して、サーバーの API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、以下のセクションで提供されているサンプルスクリプトを、変更を加えて使用して、更新する必要がある`VPC_ENDPOINT`タイプを使用して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
- EndpointType を使用したサーバーの更新 API

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. で AWS Transfer Family コンソールを開きますhttps://console.aws.amazon.com/transfer/。
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 を使用してサーバーを識別するには

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、次のサンプルスクリプトを変更して使用して、`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。この例では、Amazon EC2DescribeRegionsと Transfer Family のListServersAPI呼び出しを使用して、 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. で AWS Transfer Family コンソールを開きますhttps://console.aws.amazon.com/transfer/。
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. エンドポイントタイプ のVPCホストを選択します。
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. で AWS Transfer Family コンソールを開きますhttps://console.aws.amazon.com/transfer/。

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

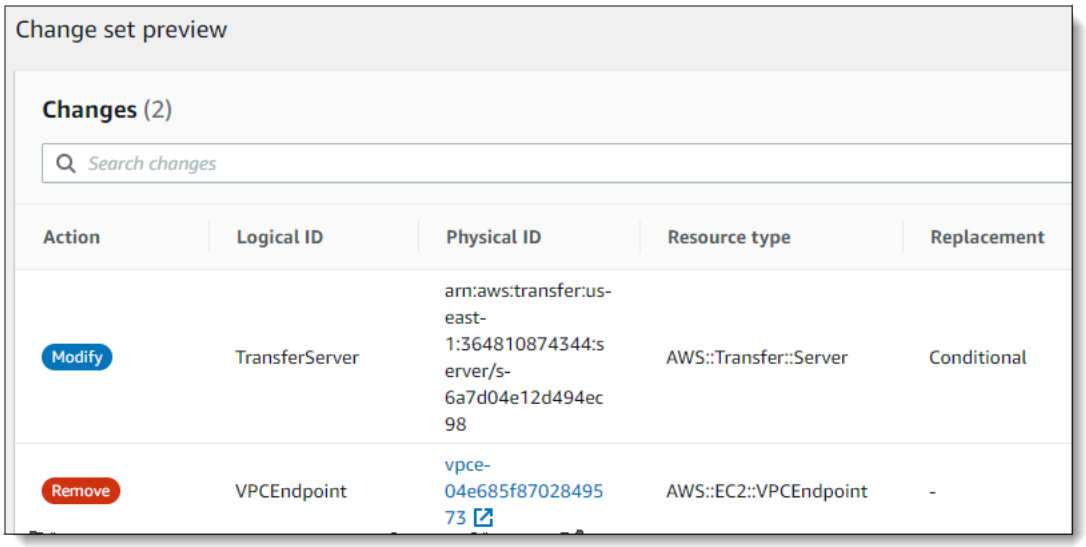
b. Transfer Family サーバーの作成に使用するスタックを選択します。

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、 の Transfer Family 管理コンソールに移動しますhttps://console.aws.amazon.com/transfer/。

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。

EndpointType を使用したサーバーの更新 API

describe-server AWS CLI コマンドまたは UpdateServer API コマンドを使用できます。次のスクリプト例では、Transfer Family サーバーを停止し、 を更新 EndpointTypeし、VPC_ を削除ENDPOINT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。

登録されたカスタム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できます。Route 53 は、 AWS Transfer Family ネイティブでサポートされる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する
- 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 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。
- その他 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する

サーバーを作成するときは、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 を初めて使用していて、ホストゾーンがない場合は、 は新しいホストゾーンとCNAMEレコード AWS Transfer Family を追加します。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。

> **Note**
>
> このステップを、サーバーを作成するワークフローの中で実行した場合、サーバーが正常に作成され、カスタムホスト名が [None] (なし) に設定されます。

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ときは、Amazon Route 53 以外のDNSプロバイダーを使用することもできます。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転送されていることを確認します。

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形式とは異なります。VPC エンドポイントを検索するには、VPCサーバーの詳細ページで 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、リストされているものDNSの名前です。

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 を使用してサーバーを作成するときにCLI、そのサーバーにカスタムホスト名を付ける場合は、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DNSレコードを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「レコードの操作」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。
- Route 53 をDNSプロバイダーとして使用している場合は、 を追加しますtransfer:route53HostedZoneId。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**
>
> パブリックゾーン、ホストゾーン、およびそれらのゾーンIDsは、Amazon Route 53 で利用できます。
> にサインイン AWS Management Console し、 で Route 53 コンソールを開きますhttps://console.aws.amazon.com/route53/。

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 をサポートしています。

- 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 と Linux)

> **Note**
>
> このクライアントは、Secure Shell (SSH) ファイル転送プロトコル () が有効になっているサーバーでのみ動作しますSFTP。

- WinSCP (Microsoft Windows のみ)
- Cyberduck (Windows、macOS、および Linux)
- FileZilla (Windows、macOSおよび Linux)

どのクライアントにも以下の制限が適用されます。

- 接続あたりの同時、多重化、SFTPセッションの最大数は 10 です。
- Amazon S3 と Amazon EFS (NFSv4プロトコルのため) では、ファイル名を UTF-8 エンコードにする必要があります。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。
- SSL (FTPS) 経由のファイル転送プロトコルでは、明示的モードのみがサポートされています。暗黙モードはサポートされません。
- ファイル転送プロトコル (FTP) と ではFTPS、パッシブモードのみがサポートされています。
- FTP および ではFTPS、 STREAM モードのみがサポートされています。
- FTP および ではFTPS、Image/Binary モードのみがサポートされています。
- FTP および の場合FTPS、データ接続TLSの PROT TLSC (保護されていない) がデフォルトですが、プロトコルでは AWS Transfer Family FTPS C PROT はサポートされていません。そのため、ではFTPS、データオペレーションを受け入れるために PROT P を発行する必要があります。
- 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。ストレージバックエンドEFSとして Amazon を使用している場合、 EFS は 1 回の転送で複数の接続をサポートします。

以下は、 FTPと で使用できるコマンドのリストですFTPS。

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
EPSV	MLSD	OPTS	QUIT	STAT	USER

> **Note**
>
> APPE はサポートされていません。

ではSFTP、Amazon Elastic File System (Amazon) 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しているユーザーに対して、現在次のオペレーションはサポートされていませんEFS。

サポートされていないSFTPコマンド			
SSH_FXP_READLINK	SSH_FXP_SYMLINK	SSH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、_FXP_STAT	SSHリクエストされたパスに symlink コンポーネントが含まれている場合、_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、「サービスマネージドユーザーのSSHキーを生成する」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
- Open を使用するSSH
- Win を使用するSCP
- 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

次の表は、、 AWS Transfer Family、FTPSおよび SFTPFTPプロトコルで使用可能な コマンドを示しています。

> **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

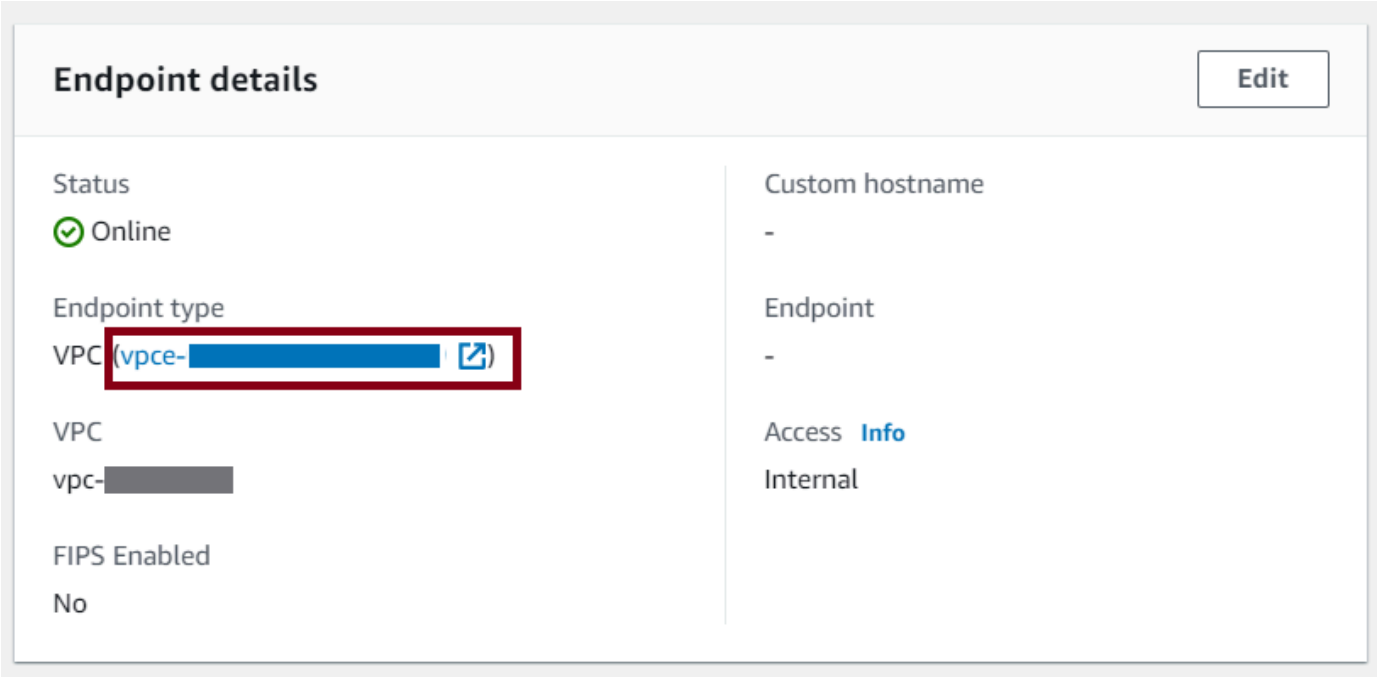
Command	Amazon S3	Amazon EFS
chown	サポートされていません	サポート (rootのみ)
get	サポート	サポート (シンボリックリンクの解決を含む)
ln -s	サポート外	サポート対象
ls/dir	サポート対象	サポート対象
mkdir	サポート対象	サポート対象
put	サポート対象	サポート対象
pwd	サポート対象	サポート
rename	ファイルでのみサポート	サポート **Note** 既存のファイルやディレクトリを上書きするような名前変更はサポートされていません。
rm	サポート	サポート
rmdir	サポート (空のディレクトリのみ)	サポート
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

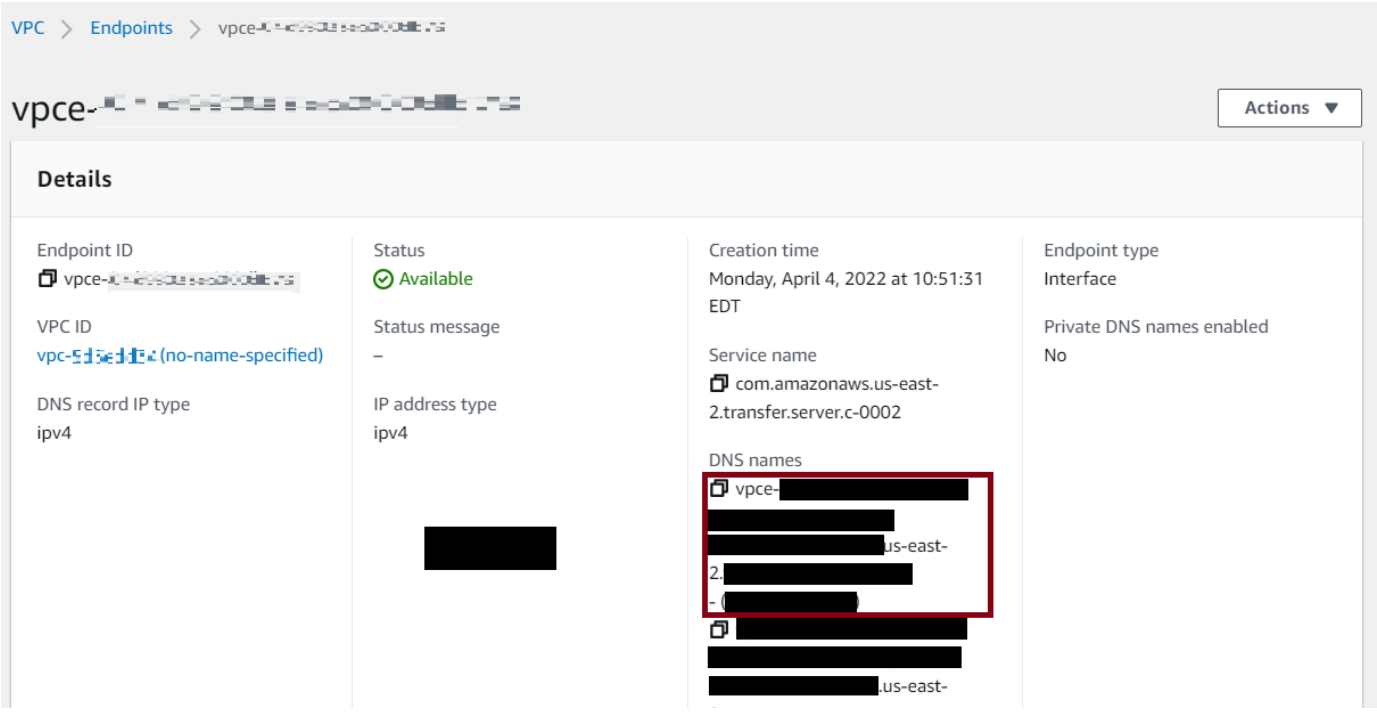
Transfer Family サーバーのエンドポイントタイプが の場合VPC、ファイルの転送に使用するエンドポイントを特定することは簡単ではありません。この場合、次の手順を使用して Amazon VPCエンドポイントを検索します。

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、 を選択しますVPC。



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 名前のリストでは、サーバーエンドポイントが最初にリストされます。



setstatエラーを回避する

一部のSFTPファイル転送クライアントは、ファイルのアップロードSETSTAT時などのコマンドを使用して、タイムスタンプやアクセス許可など、リモートファイルの属性を変更しようとします。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 `SetStatOption`を呼び出すときはUpdateServerAPI、`ProtocolDetails`オプションを使用して、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SETSTATで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`ENABLE_NO_OP`するように 値を 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します。
- `SetStatOption` `ENABLE_NO_OP` 設定ではエラーは無視されますが、 CloudWatch ログ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特定できます。

このオプションAPIの詳細については、「」を参照してくださいProtocolDetails。

Open を使用するSSH

Open を使用してコマンドラインからファイルを転送するには、以下の手順に従いますSSH。

> **Note**
>
> このクライアントは、 SFTPが有効な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
```

前のコマンドでは、 *sftp_user* はユーザー名で、 *transfer-key*はSSHプライベートキーです。ここでは、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*service_endpoint*を示します。

> **Note**
>
> このコマンドは、デフォルト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た場合を除き、 はポート 22 SFTPを使用します。次のように、 コマンドに**-P**フラグを追加して、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 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、サーバーエンドポイントのホスト名が環境内の DNSサービスによって解決されるまでに数分かかることがあります。

Win を使用するSCP

Win を使用してコマンドラインからファイルを転送するには、以下の手順に従いますSCP。

> **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。

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

1. WinSCP クライアントを開きます。
2. ログインダイアログボックスの ファイルプロトコル で、プロトコル SFTPまたは を選択しますFTP。

   暗号化 FTPで を選択した場合は、次のいずれかを選択します。

   - の暗号化なし FTP
   - TLS/SSL の明示的暗号化 FTPS
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、および 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、「」を参照してくださいAmazon VPCエンドポイントを検索する。
4. [Port number] (ポート番号) に次の値を入力します。

   - SFTP に関する **22**

- **21** FTP/ 用FTPS

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 つ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、認証 を選択します。
7. プライベートキーファイル については、ファイルシステムからSSHプライベートキーファイルを参照して選択します。

> **Note**
>
> WinSCP がSSHプライベートキーを PPK形式に変換することを提案する場合は、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。

    メソッドを使用して drag-and-drop、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、Win 内の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SCP。

> **Note**
>
> この注意は、Amazon をストレージEFSに使用する場合には適用されません。
> 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。したがって、Amazon S3 をストレージに使用する場合、ファイル転送を実行する前に、WinSCP タイムスタンプ設定を無効にする (または で説明`SetStatOption`されているように を使用する`setstat`エラーを回避する) 必要があります。これを行うには、WinSCP Transfer 設定ダイアログボックスで、アクセス許可のアップロード設定オプションとタイムスタンプの共通保存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 ダイアログボックスで、プロトコル 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、および 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、「」を参照してくださいAmazon VPCエンドポイントを検索する。

5. [Port number] (ポート番号) に次の値を入力します。
   - SFTP に関する **22**
   - **21** FTP/ 用FTPS
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 を選択した場合は、SSHプライベートキー でプライベートSSH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 用に を設定するには

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. プロトコル の全般タブで、プロトコル SFTPまたは を選択しますFTP。

   暗号化 FTPで を選択した場合は、次のいずれかを選択します。

   - プレーン FTP (安全でない) のみを使用する – 用 FTP
   - 利用可能なTLS場合は FTP を明示的に使用する - 用 FTPS
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、および 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、「」を参照してくださいAmazon VPCエンドポイントを検索する。

   - を使用している場合はSFTP、次のように入力します。 `sftp://hostname`
   - を使用している場合はFTPS、次のように入力します。 `ftps://hostname`

   必ず を置き換えてください *hostname* 実際のサーバーエンドポイントを使用します。
6. [Port number] (ポート番号) に次の値を入力します。

   - SFTP に関する **22**
   - **21** FTP/ 用FTPS
7. SFTP を選択した場合、ログオンタイプ でキーファイル を選択します。

   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場合は、Amazon S3 バケットに部分的なオブジェクトを書き込む AWS Transfer Family ことができます。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。

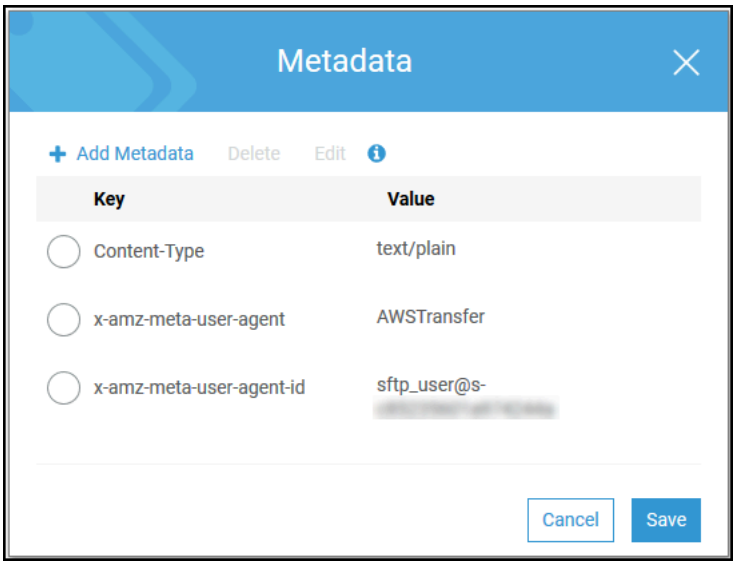
トピック

- 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 オブジェクトの 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
```

上記のリクエスタフィールドには、 というIAMロールが表示されます`IamRoleName`。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関する情報を示します。

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リクエストを送信すると、サーバーはまず、ユーザーが関連付けられたSSHプライベート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、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、SFTPサーバーへの認証から 1 文字または 2 文字の長さのユーザー名をブロックします。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

各ストレージオプションの特性:

- アクセスを制限するには: Amazon S3 はセッションポリシーをサポートします。Amazon はPOSIXユーザー、グループ、セカンダリグループEFSをサポートします。 IDs
- 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 の場合、論理ディレクトリのサポートのほとんどは API/ 経由ですCLI。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 の例を参照してください API。

> **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は 2,000 件に制限されます。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 Amazon S3 バケットを設定する場合は、このナレッジセンターの記事で説明されているステップに従います。別の AWS アカウントにある Amazon Simple Storage Service バケットを使用するように AWS Transfer Family サーバーを設定するにはどうすればよいですか？

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. で AWS Transfer Family コンソールを開きhttps://console.aws.amazon.com/transfer/、ナビゲーションペインから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. アクセス では、Amazon S3 バケットへのアクセスを提供する、以前に作成したIAMロールを選択します。

   このIAMロールは、 の手順を使用して作成しましたIAM ロールとポリシーを作成する。このIAMロールには、Amazon S3 バケットへのアクセスを提供するIAMポリシーが含まれています。また、別のIAMポリシーで定義されている 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なアクセスコントロールが必要な場合は、「 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。
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)
   - からポリシーを選択IAM: 既存のセッションポリシーを選択できます。表示を選択して、ポリシーの詳細を含むJSONオブジェクトを表示します。

- [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。表示を選択して、ポリシーの詳細を含むJSONオブジェクトを表示します。

> **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 ロールとポリシーを作成する」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. ホームディレクトリ では、Amazon S3 バケットを選択して、 を使用して転送するデータを保存します AWS Transfer Family。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ロールがこの`root`ディレクトリへのアクセスを許可していることを確認してください。

> **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 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方法については、「」を参照してくださいサービスマネージドユーザーのSSHキーを生成する。

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、ポータブルオペレーティングシステムインターフェイス (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 ユーザーのディレクトリの設定の詳細については、「」を参照してくださいTransfer Family の Amazon EFS ユーザーを設定する。

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. で AWS Transfer Family コンソールを開きhttps://console.aws.amazon.com/transfer/、ナビゲーションペインからサーバーを選択します。
2. サーバーページで、ユーザーを追加する Amazon EFSサーバーを選択します。

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 のユーザー管理者権限が付与されますEFS。
      - 追加のユーザーの場合は、ユーザーのPOSIXユーザー ID とグループ ID を入力します。これらはIDs、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) セカンダリグループ IDsには、ユーザーIDsごとに 1 つ以上の追加のPOSIXグループをカンマで区切って入力します。

   d. Access では、次のIAMロールを選択します。

      - ユーザーがアクセスする Amazon EFSリソース (ファイルシステム) のみにアクセスできるようにします。
      - 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。

      マウントアクセスおよび読み取り/書き込みアクセス許可を持つ Amazon EFS ファイルシステムの選択には、 IAMロールを使用することをお勧めします。例えば、次の 2 つの AWS 管理ポリシーの組み合わせは、かなり許可されますが、 は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。

- を使用して転送するデータの保存に使用する Amazon EFS ファイルシステムを選択します AWS Transfer Family。
- 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 のEFSファイルシステム名 (fs-xxxxxxx) を表示できません。

> **Note**
>
> 制限付きオプションを選択すると、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ロールがこのルートディレクトリへのアクセスを許可していることを確認してください。

5. 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方法については、「」を参照してくださいサービスマネージドユーザーのSSHキーを生成する。

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。

Transfer Family サーバーSFTPに初めてアクセスしたときに発生する可能性のある問題：

- 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。などの AWS マネージドポリシーを追加できますAmazonElasticFileSystemClientFullAccess。
- sftp プロンプトpwdで 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、次の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、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

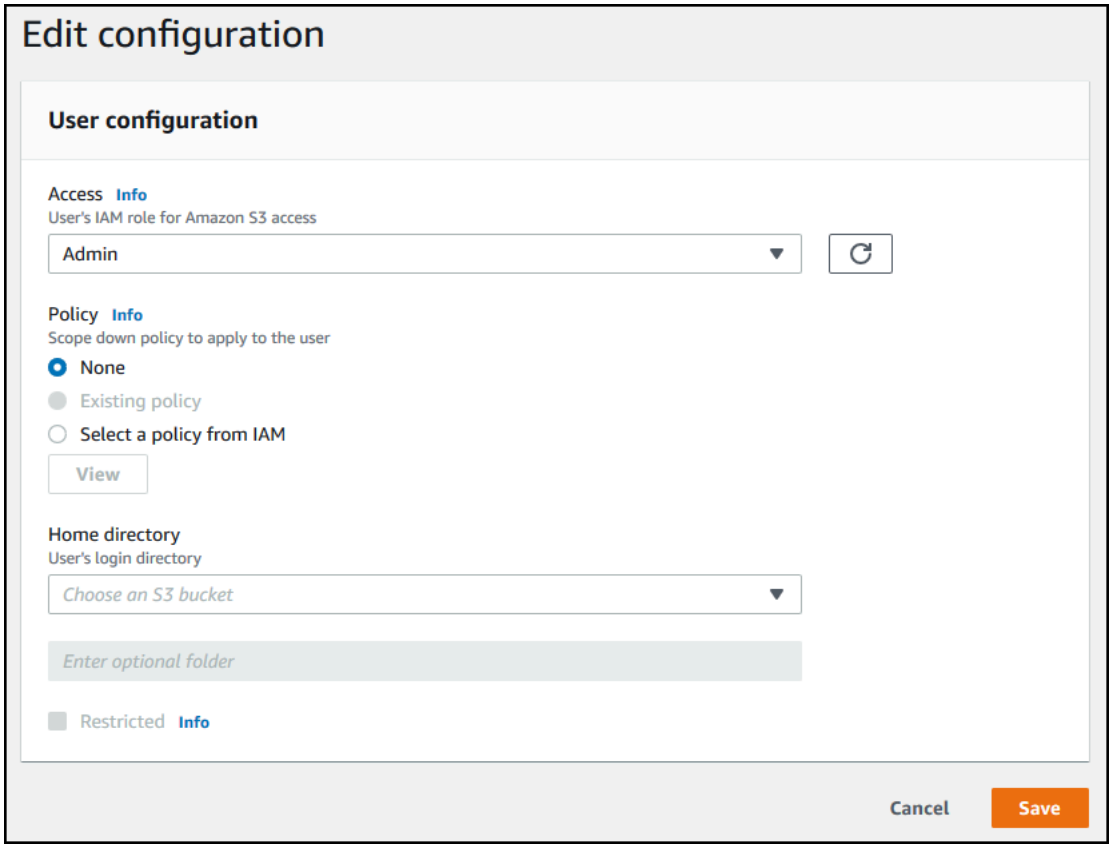
1. で AWS Transfer Family コンソールを開きますhttps://console.aws.amazon.com/transfer/。
2. ナビゲーションペインで [Servers] (サーバー) を選択して [Servers] (サーバー) ページを表示します。

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. で AWS Transfer Family コンソールを開きますhttps://console.aws.amazon.com/transfer/。
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. 設定の編集ページの アクセス で、Amazon S3 バケットへのアクセスを提供する、以前に作成したIAMロールを選択します。

このIAMロールは、 の手順を使用して作成しましたIAM ロールとポリシーを作成する。このIAMロールには、Amazon S3 バケットへのアクセスを提供するIAMポリシーが含まれています。また、別のIAMポリシーで定義されている 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)
   - からポリシーIAMを選択して、既存のポリシーを選択します。表示を選択して、ポリシーの詳細を含むJSONオブジェクトを表示します。

   セッションポリシーの詳細については、「IAM ロールとポリシーを作成する」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. ホームディレクトリ では、Amazon S3 バケットを選択して、 を使用して転送するデータを保存します AWS Transfer Family。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ロールがこの`root`ディレクトリへのアクセスを許可していることを確認してください。

   > **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. で AWS Transfer Family コンソールを開きますhttps://console.aws.amazon.com/transfer/。
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. で AWS Transfer Family コンソールを開きますhttps://console.aws.amazon.com/transfer/。
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. 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) ファイル転送プロトコル () が有効になっているサーバーでのみ使用されますSFTP。SSH キーペアを生成する方法については、「」を参照してくださいサービスマネージドユーザーのSSHキーを生成する。

6. SSH パブリックキー には、キーペアのSSHパブリックSSH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 ですssh-rsa *string*。SSH キーペアを生成するには、「」を参照してくださいサービスマネージドユーザーのSSHキーを生成する。

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

AWS Transfer Family を使用して、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) に保存されているデータFTPに対してSFTP、、FTPS、および に対するアクセスを AWS Directory Service ユーザーとグループに安全に提供できます

EFS。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。

Active Directory コネクタを使用して、オンプレミス環境または AWS クラウド AWS Managed Microsoft AD 内の の Active Directory グループへのアクセスを提供できます。クラウド AWS またはオンプレミスネットワークのいずれかで Microsoft Windows 環境で既に設定されているユーザーに、アイデンティティ AWS Managed Microsoft AD に を使用する 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、 AWS Managed Microsoft AD または AD Connector を使用して既存の RADIUSベースのMFAインフラストラクチャの多要素認証 (MFA) を有効にすることはできません。
> - 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、次の手順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 グループからの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
- Active Directoryレルムでの作業
- ID プロバイダー AWS Managed Microsoft AD としての選択

- グループへのアクセス権の付与
- ユーザーのテスト
- グループのサーバーアクセスの削除
- (SSHSecure Shell) を使用してサーバーに接続する
- 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

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。これを行うには、グループごとにセキュリティ識別子 (SID) を使用できます。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を介して 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 を取得し、 を置き換えます SID。*YourGroupName* グループの名前。

```
Get-ADGroup -Filter {samAccountName -like "YourGroupName*"} -Properties * | Select
 SamAccountName,ObjectSid
```

> **Note**
>
> ID プロバイダー AWS Directory Service として を使用し、 `userPrincipalName` との値`SamAccountName`が異なる場合、 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。

ロールへのアクセス AWS Directory Service 許可を追加する

また、 を ID プロバイダー AWS Directory Service として使用するアクセス許可も必要です AWS Directory Service API。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。これらのアクセス許可の詳細については、AWS Directory Service API「アクセス許可: アクション、リソース、および条件」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。

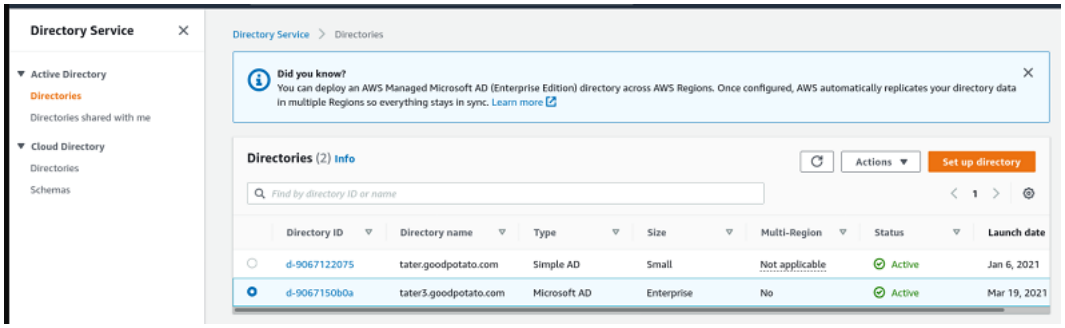
ID プロバイダー AWS Managed Microsoft AD としての選択

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 で使用するには

1. にサインイン AWS Management Console し、 で AWS Directory Service コンソールを開きますhttps://console.aws.amazon.com/directoryservicev2/。

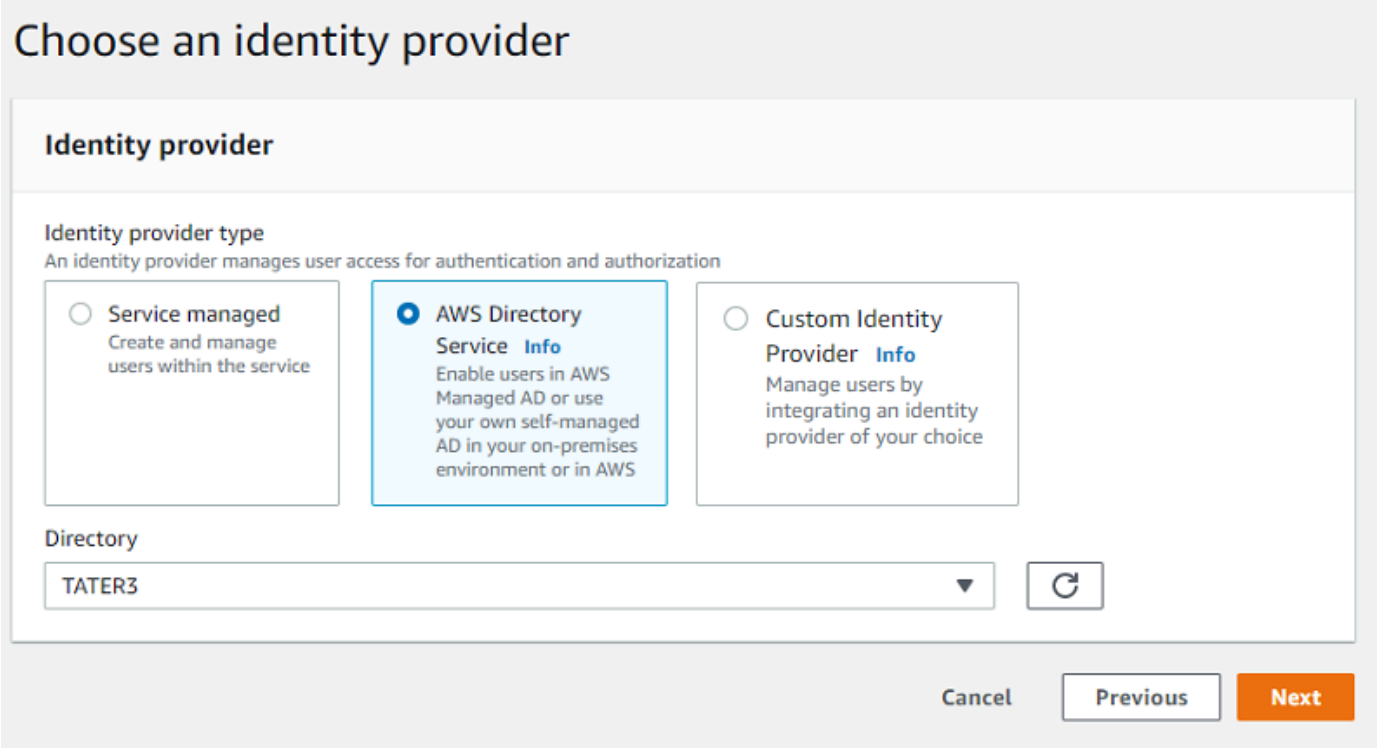
   AWS Directory Service コンソールを使用して、1 つ以上のマネージドディレクトリを設定します。詳細については、 AWS Directory Service 管理者ガイドの AWS Managed Microsoft AD を参照してください。



2. で AWS Transfer Family コンソールを開きhttps://console.aws.amazon.com/transfer/、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**
>
> を選択した場合はFTPS、証明書を指定 AWS Certificate Manager 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対応サーバーを作成する
   - FTPS対応サーバーを作成する
   - FTP対応サーバーを作成する

   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、 ディレクトリ内のどのグループが、 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アクセス権を持つ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. で AWS Transfer Family コンソールを開きますhttps://console.aws.amazon.com/transfer/。
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. このサーバーにアクセスする AWS Managed Microsoft AD ディレクトリSIDの を入力します。

   > **Note**
   >
   > グループの を検索する方法については、SID「」を参照してくださいthe section called “の使用を開始する前に AWS Directory Service for Microsoft Active Directory”。

5. アクセス では、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 。

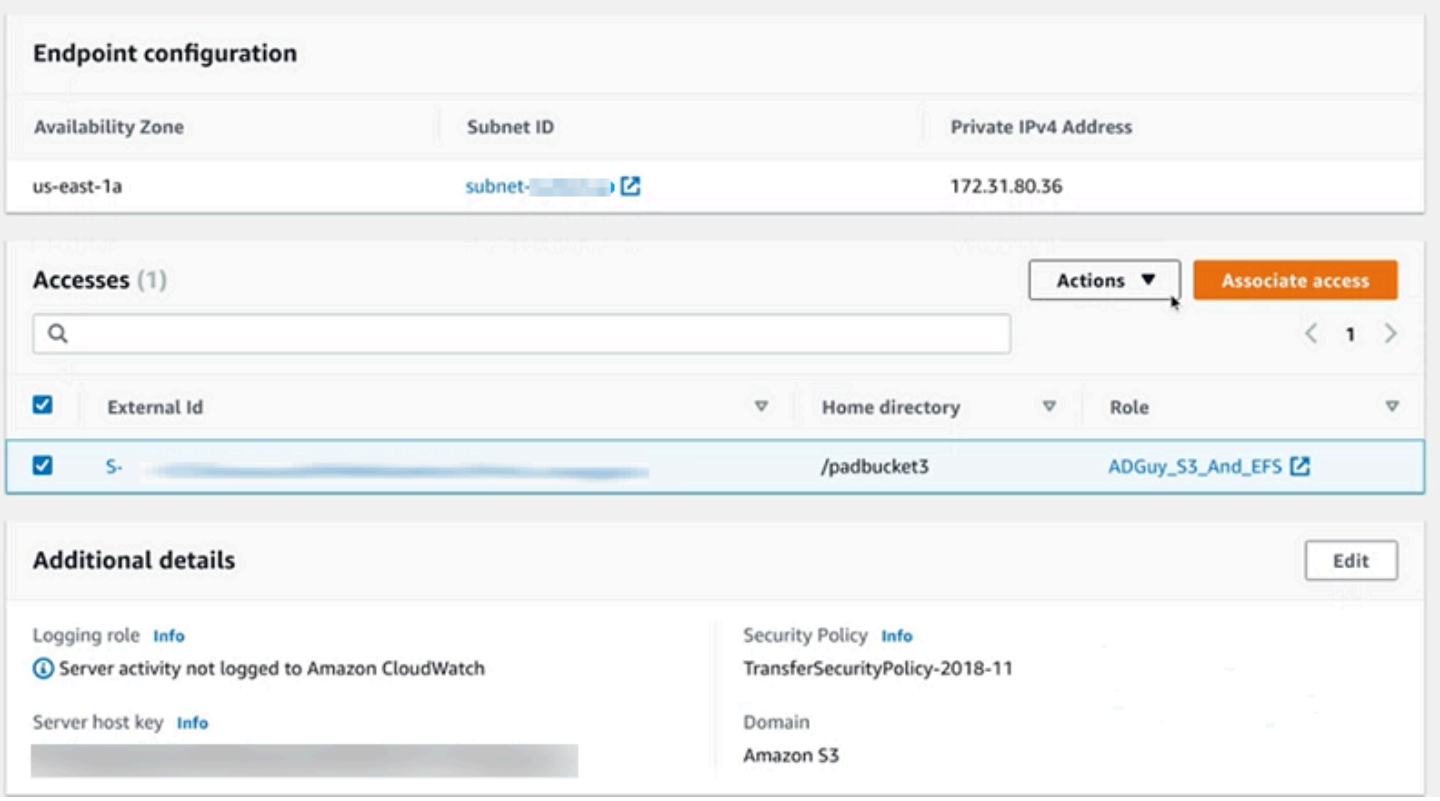
8. [Add] (追加) をクリックして関連付けを作成します。
9. サーバーを選択します。
10. 「アクセスの追加」を選択します。

    - グループの SIDを入力します。

> **Note**
>
> を検索する方法についてはSID、「」を参照してくださいthe section called “の使用を開始する前に AWS Directory Service for Microsoft Active Directory”。

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。

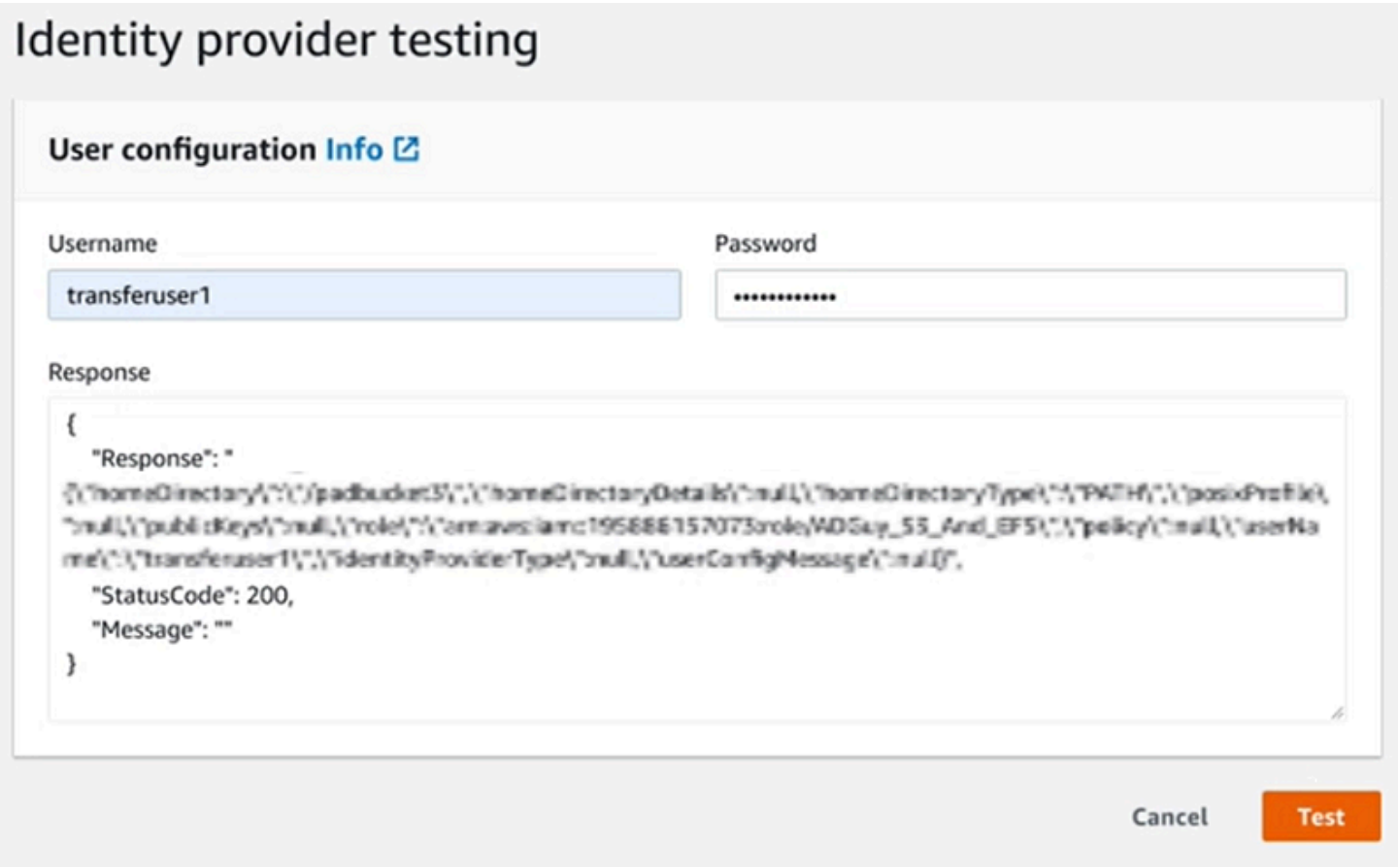
> **Note**
>
> ユーザーは、[Endpoint configuration] (エンドポイント設定) ページの [Access] (アクセス) セクションのリストにあるいずれかのグループ (外部 ID) に属している必要があります。ユー

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。

(SSHSecure 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、 を使用してサーバーに接続SSH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用できます。

```
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 次のオプションを使用できます。

- 一方向フォレスト信頼 (オンプレミス Active Directory の送受信) は AWS Managed Microsoft AD 、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 Active Directory 間の双方向信頼を使用する
  - 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

- 既存の Active Directory フォレストを SFTP Transfer のニーズに活用するには、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 ドメインサービスの使用を開始する前に
- ステップ 1: Azure Active Directory ドメイン サービスを追加する
- ステップ 2 : サービスアカウントの作成
- ステップ 3: AD Connector を使用して AWS ディレクトリを設定する
- ステップ 4: AWS Transfer Family サーバーのセットアップ
- ステップ5 : グループへのアクセス権の付与
- ステップ 6: ユーザーのテスト

AWS Directory Service for Azure Active Directory ドメインサービスの使用を開始する前に

の場合 AWS、以下が必要です。

- Transfer Family サーバーを使用している AWS リージョンの仮想プライベートクラウド (VPC)

- 内の少なくとも 2 つのプライベートサブネット VPC
- にはインターネット接続VPCが必要です
- Microsoft Azure VPNに接続するための site-to-site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

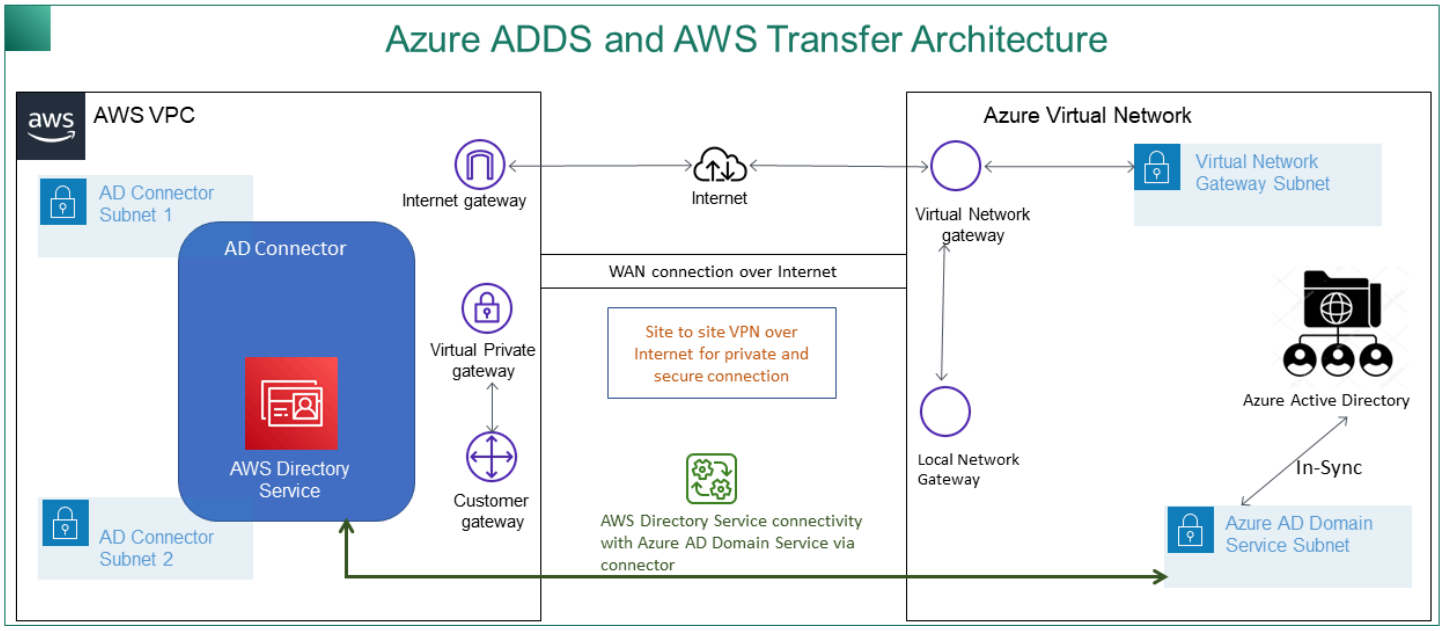
Microsoft Azure の場合は、次のものが必要です

- 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
- VPN Amazon VPCと Azure リソースグループ間の接続

> **Note**
>
> これは、ネイティブIPSECトンネルまたは VPN アプライアンスを介して行うことができます。このトピックでは、Azure Virtual Network Gateway とローカルネットワークゲートウェイ間の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 AWS VPC。

- Microsoft Azure VPNに接続するための site-to-site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。

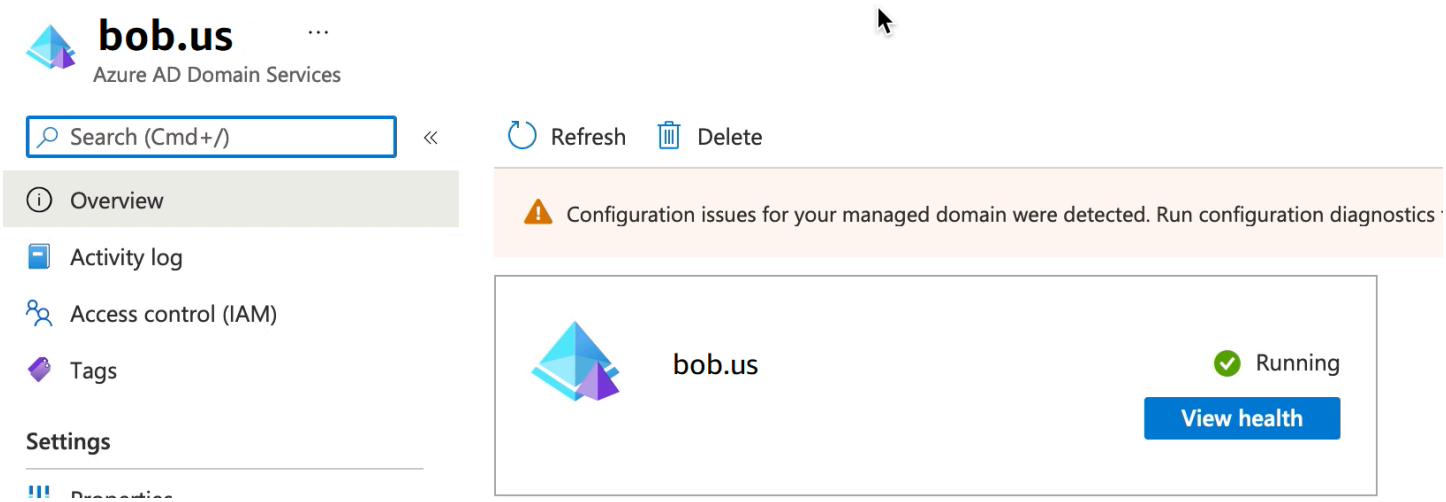


ステップ 1: Azure Active Directory ドメイン サービスを追加する

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 Transfer Server 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。

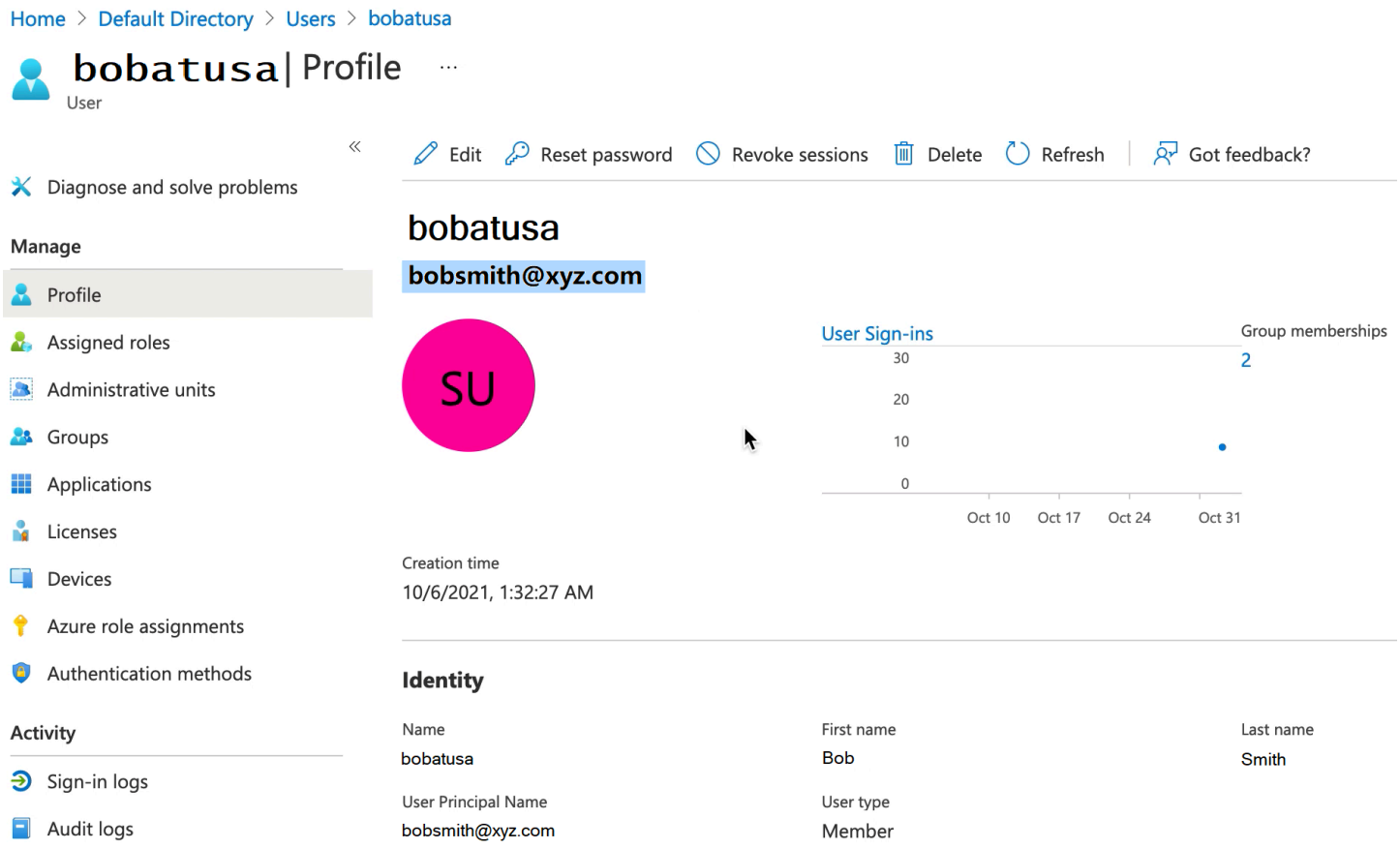
Azure Active Directory ドメインサービス (Azure ADDS) を有効にする方法については、「チュートリアル: Azure Active Directory ドメインサービスマネージドドメイン の作成と設定」を参照してください。

> **Note**
>
> Azure を有効にするときはADDS、SFTPTransfer Server を接続するリソースグループと Azure AD ドメイン用に設定されていることを確認します。



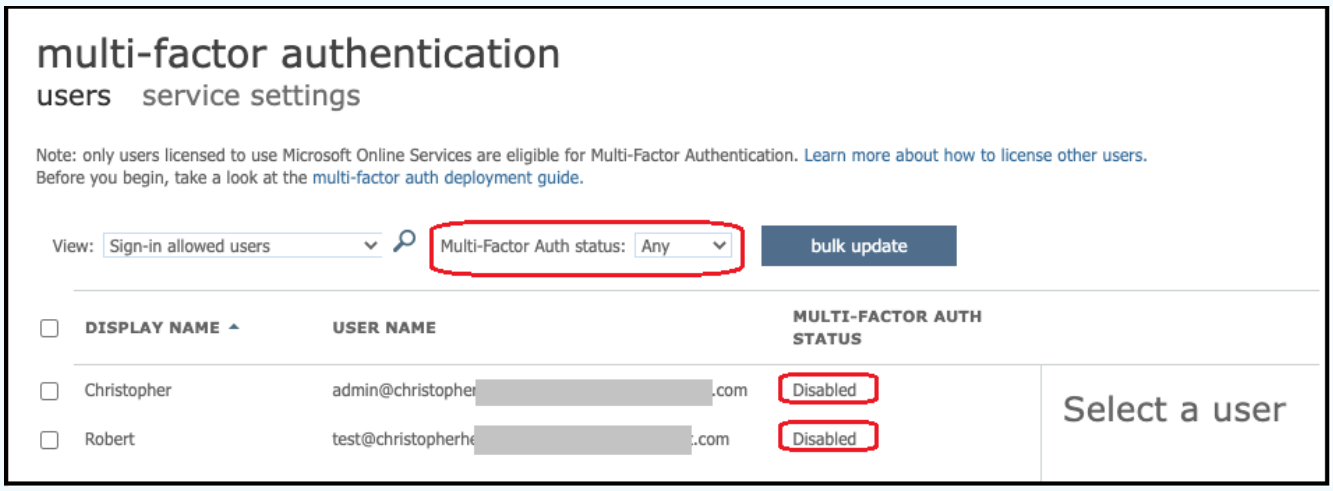
ステップ 2：サービスアカウントの作成

Azure AD には、Azure の管理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ADDS。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 と同期していることを確認しますADDS。



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 を認証した後にMFAトークンを提供できませんSFTP。接続を試みるMFA前に、必ず 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 を設定しADDS、 と Azure Virtual Network の間にIPSECVPN AWS VPC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 を送信して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。

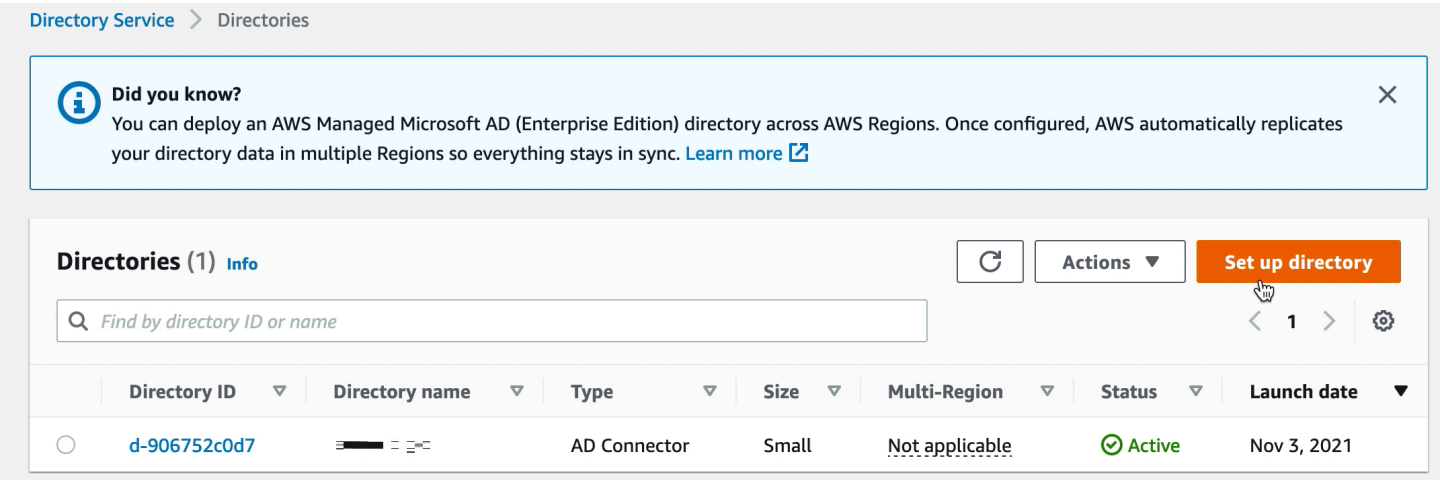
AD Connector を使用して AWS ディレクトリを設定するには

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、 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 に使用しているドメイン名を入力しますADDS。
   - 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、ディレクトリステータスはアクティブ になり、SFTPTransfer サーバーで使用できるようになります。



ステップ 4: AWS Transfer Family サーバーのセットアップ

SFTP プロトコル、および AWS Directory Service 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。Directory ドロップダウンリストから、AD Connector を使用してステップ 3: Setup AWS Directory 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 ディレクトリサービスで 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

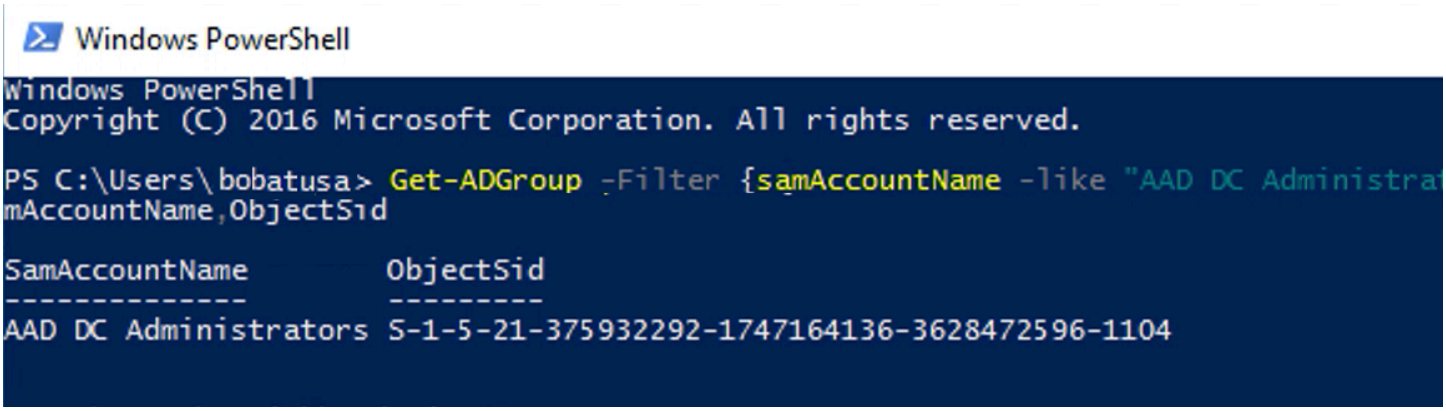
サーバーを作成したら、 ディレクトリ内のどのグループが、 を使用して有効なプロトコル経由でファイルをアップロードおよびダウンロードするアクセス権を持つかを選択する必要があります AWS Transfer Family。そのためには、アクセス権を作成します。

> **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、SIDグループの 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 を取得し、 を置き換えます SID。*YourGroupName* グループの名前。

```
Get-ADGroup -Filter {samAccountName -like "YourGroupName*"} -Properties * | Select
 SamAccountName,ObjectSid
```

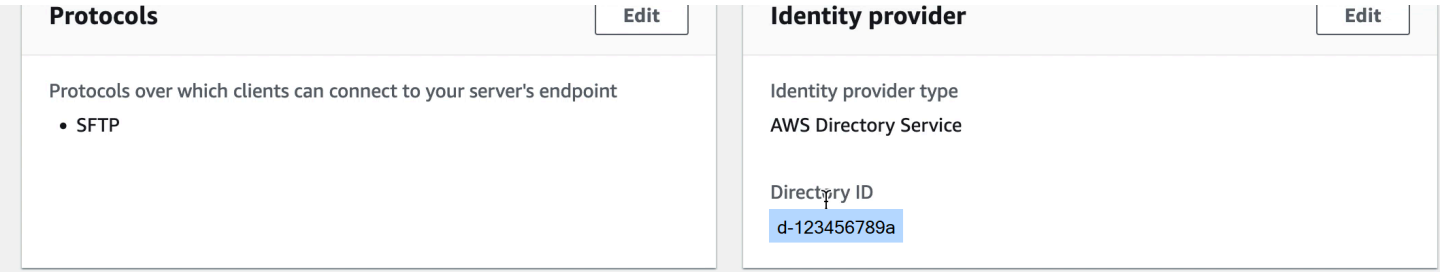


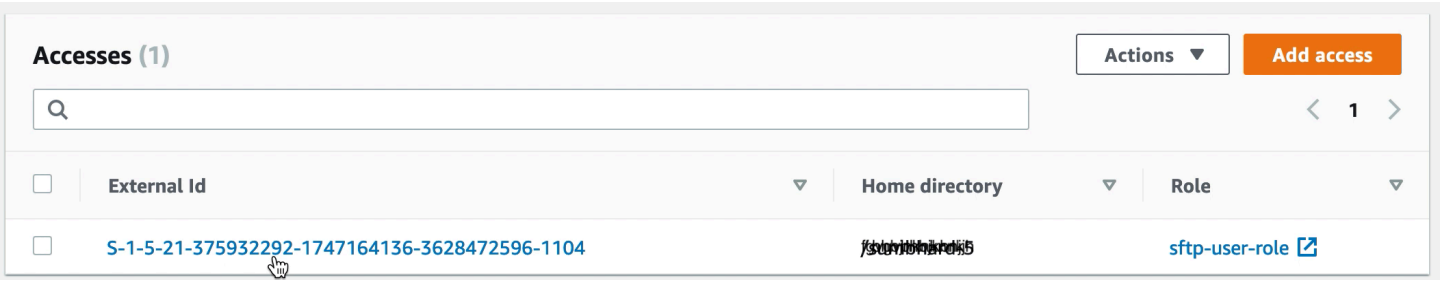
グループへのアクセスを許可する

1. を開きますhttps://console.aws.amazon.com/transfer/。
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SID受け取った を入力します。
4. アクセス では、グループ AWS Identity and Access Management の役割を選択します。
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、 で既存の ID プロバイダーを使用できます AWS Transfer Family。ID プロバイダーは、Amazon S3 または Amazon Elastic File System (Amazon) 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EFS。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。 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、一元管理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 つの画面が提供されます。

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ます。ID プロバイダーは、Amazon S3 または Amazon へのアクセスをユーザーに認証および許可しますEFS。アイデ

ンティティプロバイダーを統合するRESTfulAPIために が必要な場合、または AWS WAF を使用してジオブロッキングまたはレート制限リクエストにその機能を活用する場合は、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 を使用して新しいサーバーを作成できます。 CreateServer API オペレーション。

> **Note**
>
> 参加できるワークショップを開催しています。このワークショップでは、ファイル転送ソリューションを構築できます。このソリューションは、 マネージド SFTP/FTPS エンドポイントと Amazon Cognito と DynamoDB AWS Transfer Family をユーザー管理に活用します。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。

AWS Transfer Family には、カスタム ID プロバイダーを操作するための次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 を使用して ID プロバイダーを接続する – Lambda 関数でバックアップされた API Gateway メソッドを作成して、ID プロバイダーとして使用できます。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 ONLY – ユーザーは接続するためにパスワードを指定する必要があります。
- キー ONLY – 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ANDキー – 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場合ANDに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、その効果は組み合わされます。

両方のリクエストは、認証の完了を許可する正しい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、必要なすべての要素 (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) が含まれます。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ことも必要です。

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AND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で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リクエストします。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

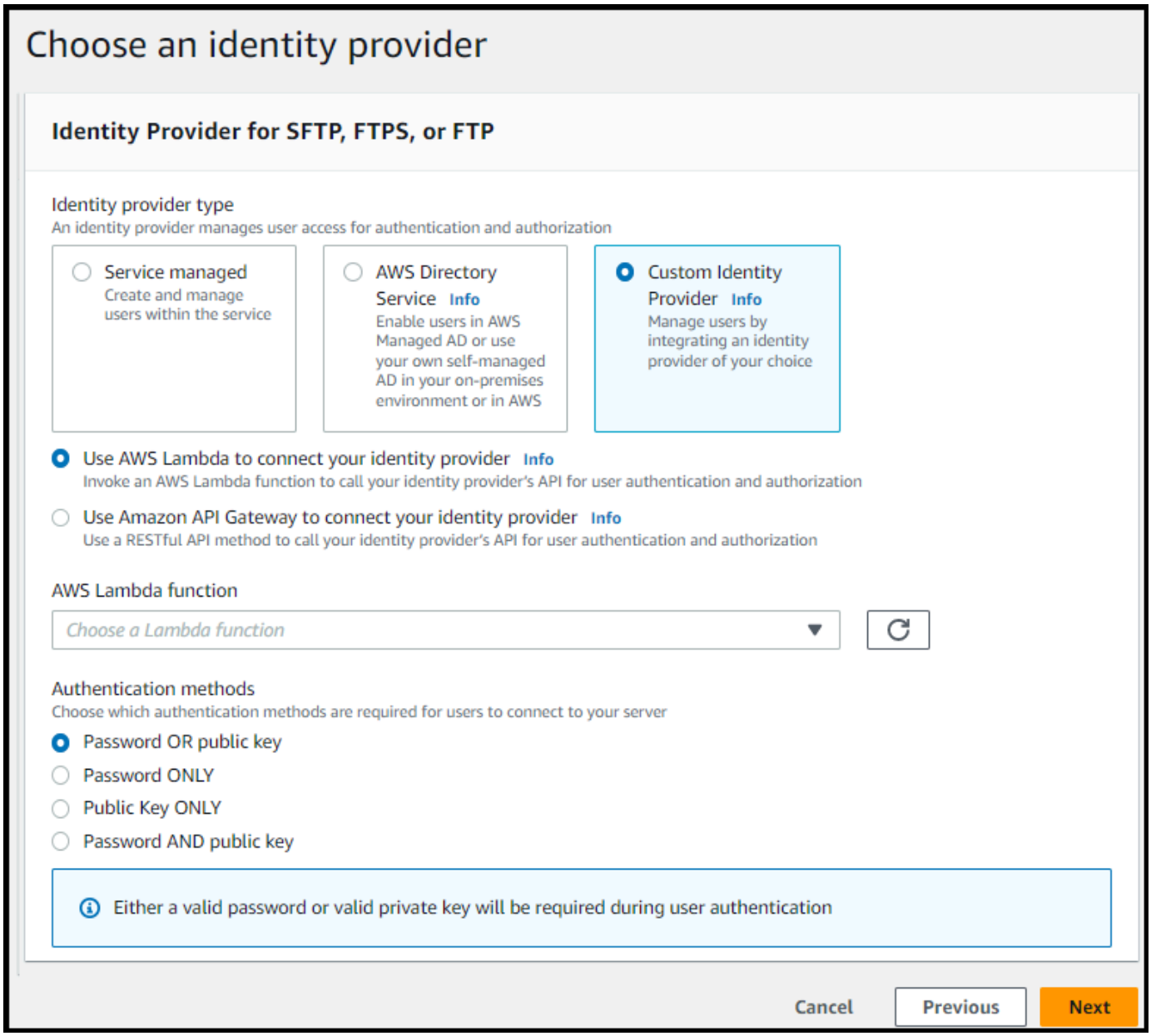
- AWS Lambda を使用して ID プロバイダーを統合する
- Amazon API Gateway を使用して ID プロバイダーを統合する

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、、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**
>
> 認証方法の選択は、Transfer Family サーバーのプロトコルの 1 つSFTPとして を有効にする場合にのみ使用できます。

3. デフォルト値 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 を参照するポリシーが必要ですARNs。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。ポリシーは文字列JSON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、各 を置き換えます。*user input placeholder* 独自の情報。

イベントメッセージの構造

カスタムのオーソライザー Lambda 関数に送信されるSFTPサーバーからのイベントメッセージ構造IDP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'
}
```

イベント構造は FTPと で似ていますFTPS。唯一の違いは、これらの値が ではなく protocolパラメータに使用されることですSFTP。

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認可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、 FIPSを通じて 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、のセキュリティポリシー AWS Transfer Family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. AWS CloudFormation コンソールを https://console.aws.amazon.com/cloudformation で開きます。
2. 「 ユーザーガイド」の AWS CloudFormation 「スタックテンプレートの選択」の「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。 AWS CloudFormation
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 プロバイダーとして使用する AWS Lambda 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

  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で、Secrets Manager を ID プロバイダーとして統合します。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 : を 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 で使用する基本テンプレートで、カスタム ID プロバイダーとして Okta を MultiFactor Authentication と統合します。
- 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 と AWS Transfer Family による の認証 AWS Lambda」で説明されています。

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。

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	必須

値	説明	必須
	システムへのユーザーのアクセスを制御するIAMロールの Amazon リソースネーム (ARN) を指定します。Amazon S3 このロールにアタッチされたポリシーは、Amazon S3 または Amazon EFS ファイルシステムとの間でファイルを転送する際にユーザーに提供するアクセスレベルを決定します。IAM ロールには、ユーザーの転送リクエストを処理するときにサーバーがリソースにアクセスできるようにする信頼関係も含める必要があります。 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 ID (`Uid`)、グループ ID (`Gid`)、および 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任意のセカンダリグループ IDs (`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たPOSIXアクセス許可により、Amazon EFS ファイルシステムとの間でファイルを転送する際にユーザーが得るアクセスレベルが決まります。	Amazon EFS バッキングストレージに必要です

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	複数のユーザーで同じ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。	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。 • に設定するとPATH、ユーザーはファイル転送プロトコルクライアントに絶対Amazon S3 バケットまたはAmazon EFSパスをそのまま表示します。 • に設定する場合はLOGICAL、 HomeDirectoryDetails パラメータにマッピングを指定してAmazon S3または Amazon EFSパスをユーザーに表示する必要があります。	オプションです。

値	説明	必須
HomeDirectoryDetails	ユーザーに表示させるAmazon S3 または Amazon EFSパスとキーと、表示方法を指定する論理ディレクトリマッピング。Entry と Targetペアを指定する必要があります。 はパスがどのように表示されるかEntryを示し、 Target は実際のAmazon S3 または Amazon EFS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 はマップの文字列表現ですJSON。これは、実際のJSONマップオブジェクトでありPosixProfile、文字列のJSON配列PublicKeysである 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がすべて例であり、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 スタックをデプロイするときに設定する場合は、サーバープロトコルを選択し、ソース IP の IP アドレスを入力します。

CLI

を使用して設定をテストするには AWS CLI

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するときに設定する場合は、サーバープロトコルとソース IP アドレスを入力します。

ユーザー認証が成功すると、テストはStatusCode: 200HTTPレスポンス、空の文字列 Message: "" (それ以外の場合は失敗の理由を含む)、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」 (プログラム内で使用できるフラットJSON文字列に変換) されたJSONオブジェクトであり、ユーザーのロールとアクセス許可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。ID プロバイダーを統合するRESTfulAPIために が必要な場合、または AWS WAF を使用してジオブロッキングまたはレート制限リクエストにその機能を活用する場合は、このオプションを使用します。

API Gateway を使用して ID プロバイダーを統合する場合の制限

- この構成はカスタムドメインをサポートしていません。
- この設定はプライベートAPIゲートウェイ をサポートしていませんURL。

これらのいずれかが必要な場合は、APIゲートウェイを使用せずに、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、 を作成して提供するための非常に安全な方法を提供しますAPIs。API Gateway を使用すると、エンドポイントを作成して、HTTPSすべての着信APIコールがセキュリティを強化して送信されるようにできます。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 は、 という名前の認証方法を提供します。これによりAWS_IAM、 が内部 AWS 的に使用する AWS Identity and Access Management （ IAM) に基づく認証と同じ認証が提供されます。で認証を有効にするとAWS_IAM、 を呼び出す明示的な権限を持つ発信者のみが、その APIのAPIゲートウェイメソッドに到達APIできます。

Transfer Family のカスタム ID プロバイダーとして API Gateway メソッドを使用するには、APIGateway メソッドIAMの を有効にします。このプロセスの一環として、Transfer Family がゲートウェイを使用するためのアクセス許可をIAMロールに付与します。

> **Note**
>
> セキュリティを向上させるために、ウェブアプリケーションファイアウォールを設定できます。 AWS WAF は、Amazon API Gateway に転送される HTTPおよび HTTPSリクエストをモニタリングできるウェブアプリケーションファイアウォールです。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 でのカスタム認証に 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はBASE64、エンコードされたパスワードを使用するように更新されました。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. AWS CloudFormation コンソールを https://console.aws.amazon.com/cloudformation で開きます。

b. 「 ユーザーガイド」の「スタックテンプレートの選択」の「既存のテンプレート AWS CloudFormation からスタックをデプロイする手順に従います。 https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/cfn-using-console-create-stack-template.html AWS CloudFormation

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、APIGateway 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、単一のサーバー内の単一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、APIGateway メソッドは、Secrets Manager の 形式のエントリに対して認証します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。詳細については、ブログ記事「 AWS Transfer Family を使用するためのパスワード認証を有効にする 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ゲートウェイに基づくAPIメソッドをサポートします。その後、Transfer Family のカスタム 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。これらの詳細には、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ARNの 、新しいゲートウェイURLの が含まれます。

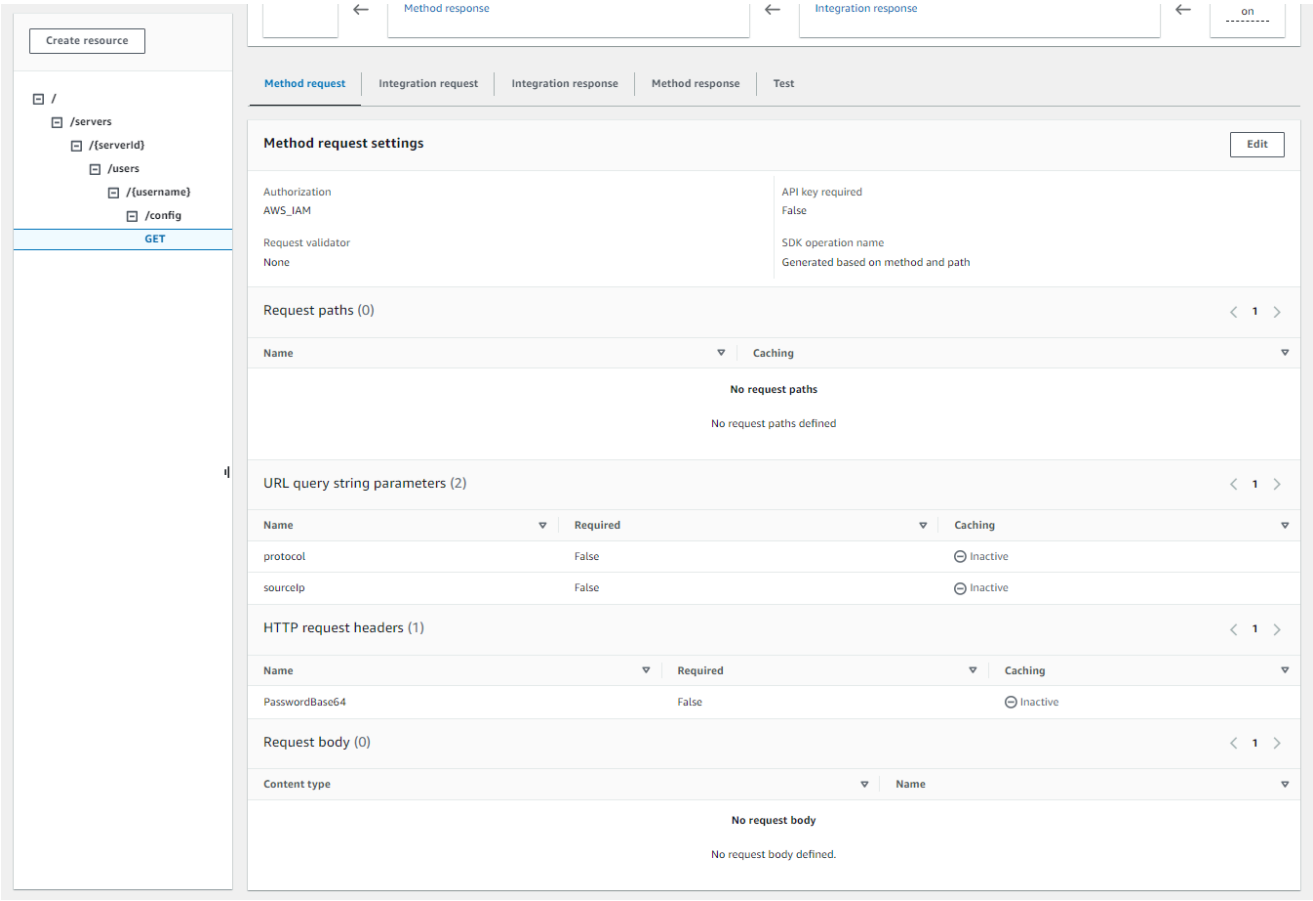
> **Note**
>
> カスタム ID プロバイダーオプションを使用してユーザーのパスワードベースの認証を有効にし、APIGateway が提供するリクエストとレスポンスのログ記録を有効にすると、APIGateway はユーザーのパスワードを Amazon CloudWatch Logs に記録します。本稼働環境でこのログを使用することは推奨されません。詳細については、API「ゲートウェイデベロッパーガイド」の「ゲートウェイでの CloudWatch APIログ記録の設定」を参照してください。 API

2. サーバーの API Gateway メソッド設定を確認します。これを実行するには:

   a. で API Gateway コンソールを開きますhttps://console.aws.amazon.com/apigateway/。

   b. テンプレートが生成した Transfer Custom Identity Provider の基本API AWS CloudFormation テンプレートを選択します。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ります。

   c. リソースペインで、 を選択しますGET。次の画面例は、正しいメソッド設定を示しています。

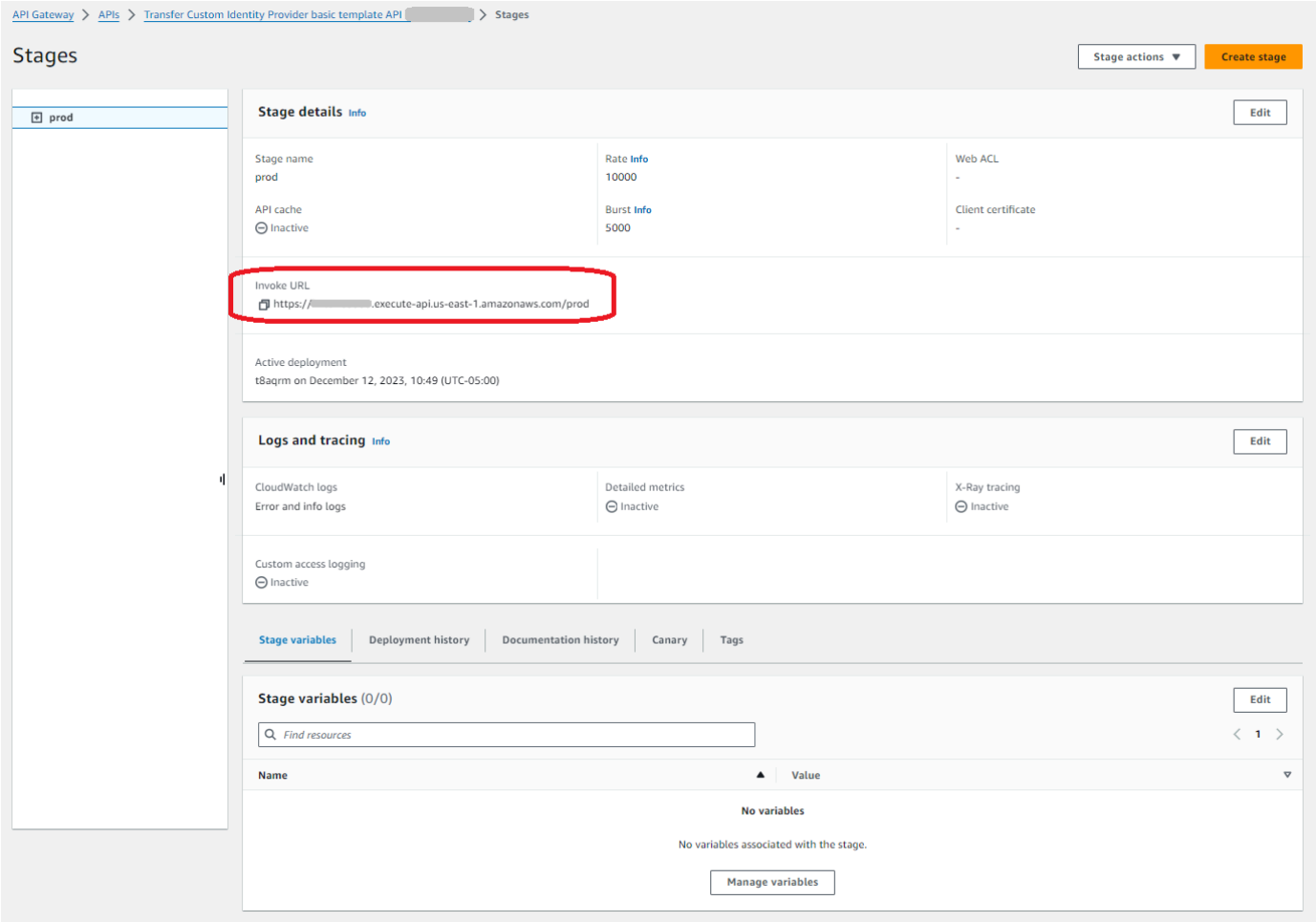


この時点で、APIゲートウェイをデプロイする準備が整います。

3. アクション で、デプロイ APIを選択します。[Deployment 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。

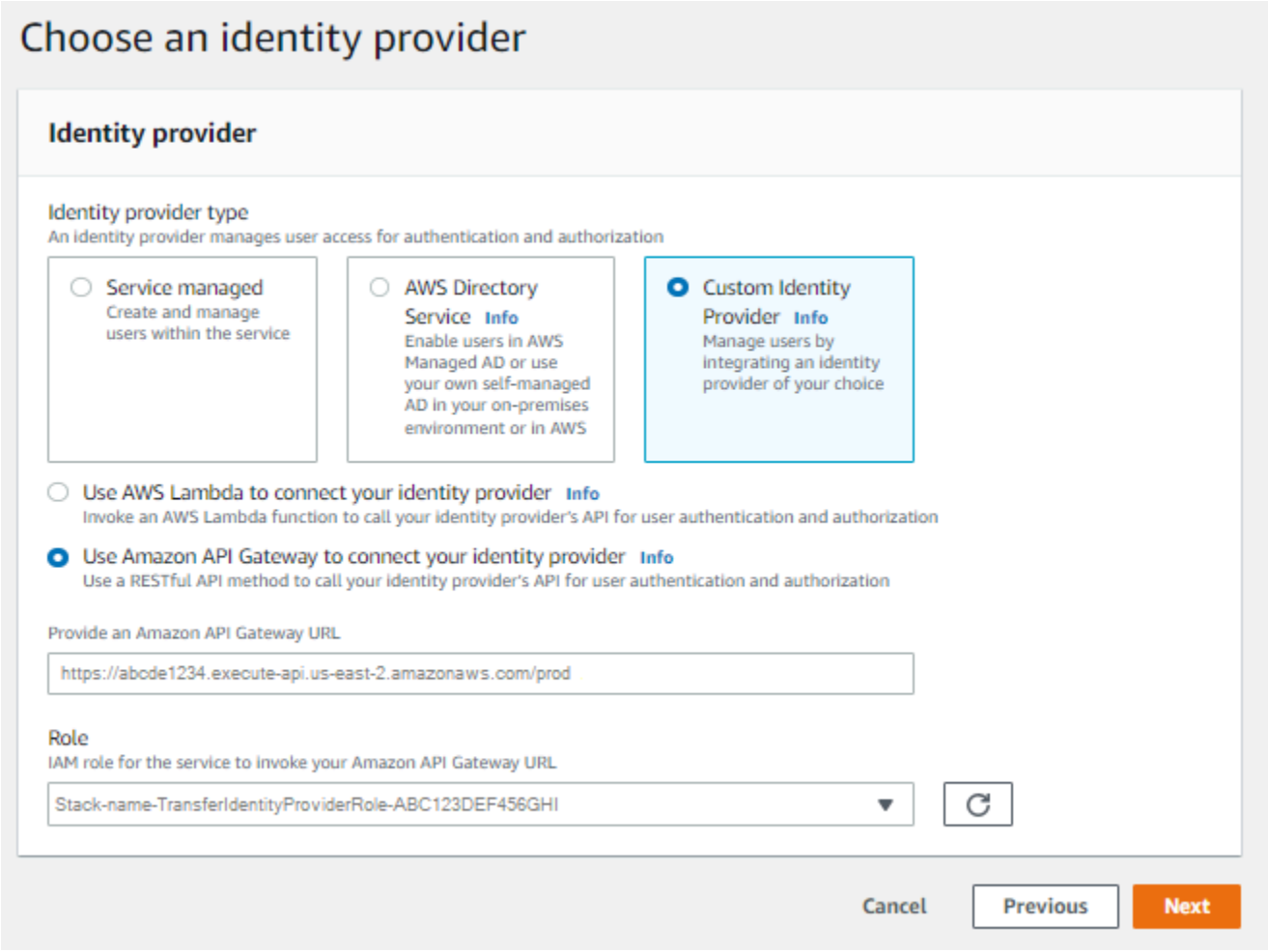
   Gateway APIメソッドが正常にデプロイされたら、次のスクリーンショットに示すように、そのパフォーマンスをステージ > ステージの詳細 で表示します。

> **Note**
>
> 画面の上部に表示される呼び出し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. で AWS Transfer Family コンソールを開きますhttps://console.aws.amazon.com/transfer/。
5. スタックの作成時に、Transfer Family が作成されているはずです。そうでない場合は、次のステップ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。ID プロバイダーを選択する では、次のスクリーンショットに示すように、カスタム を選択し、Amazon API Gateway を使用して ID プロバイダー に接続します。



b. Amazon API Gateway URLを提供するテキストボックスに、この手順のステップ 3 で作成した API Gateway エンドポイントの呼び出しURLアドレスを貼り付けます。

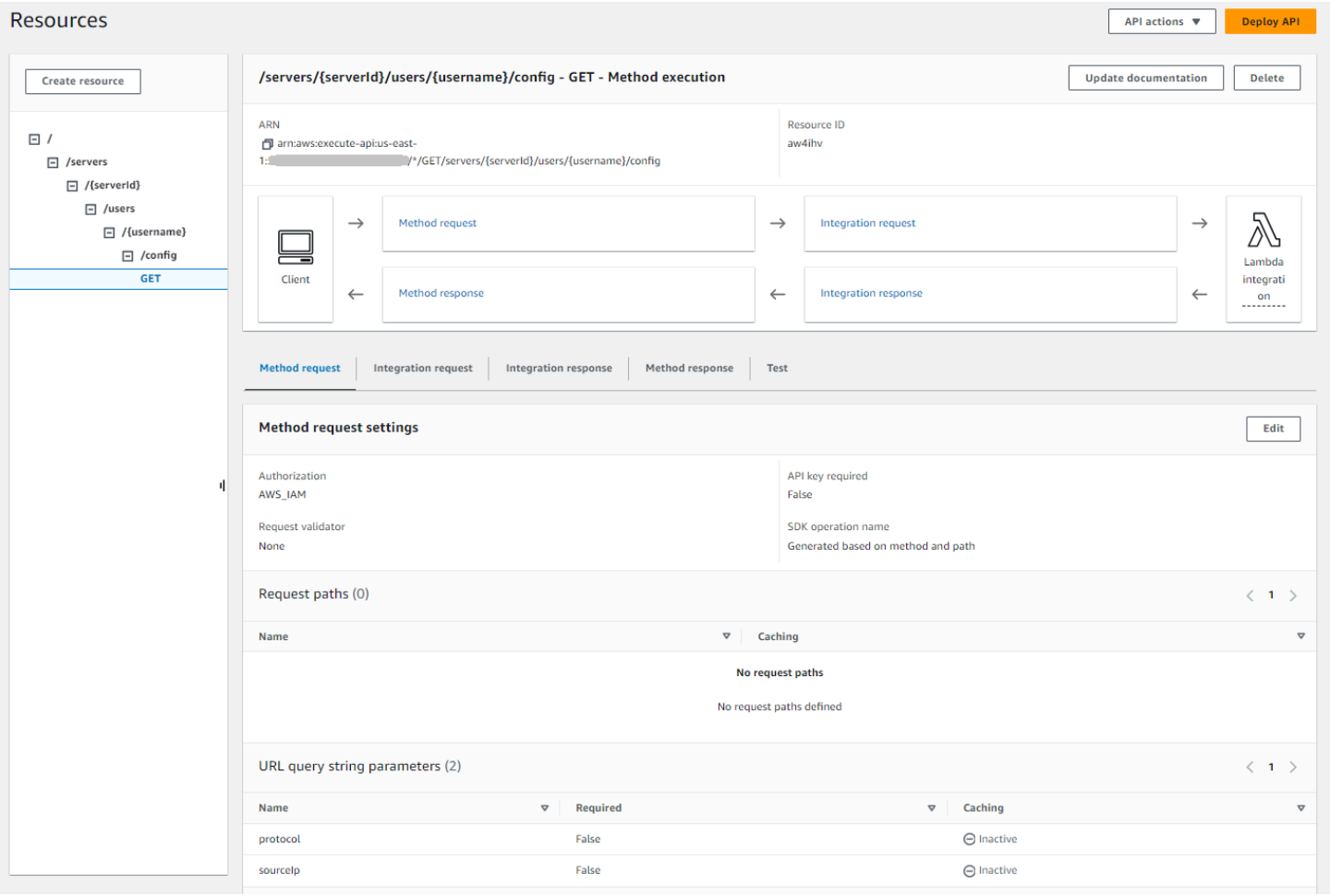
c. ロール では、 AWS CloudFormation テンプレートによって作成されたIAMロールを選択します。このロールにより、Transfer Family はAPIゲートウェイメソッドを呼び出すことができます。

   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 メソッドの実装

Transfer Family のカスタム ID プロバイダーを作成するには、APIGateway メソッドでリソースパスが の単一のメソッドを実装する必要があります`/servers/`*`serverId`*`/users/`*`username`*`/config`。*`serverId`* および *`username`*値はRESTfulリソースパスから取得されます。また、次の図に示すように、メソッドリクエスト に `sourceIp`と をURLクエリ文字列パラメータ`protocol`として追加します。



> **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認可に 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 を返す必要があります`200`。その他のHTTPステータスコードは、 へのアクセス中にエラーが発生しましたAPI。

Amazon S3 のレスポンスの例

レスポンス本文の例は、Amazon S3 の次の形式のJSONドキュメントです。

```
{
 "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**
>
> ポリシーは文字列JSON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 のレスポンスEFS例

レスポンス本文の例は、Amazon の次の形式のJSONドキュメントですEFS。

```
{
 "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:ヘッダーを指定する場合) 、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は、空のJSONオブジェクトです。

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 関数には、 JSON形式のユーザーポリシーを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。

ファイル転送プロトコル (FTP) では、Secure Shell (SSH) ファイル転送プロトコル () とファイル転送プロトコル (SFTP) SSL とは別の認証情報を維持することをお勧めします FTPS。SFTP と とは異なり、 は認証情報をプレーンテキストでFTP送信FTPするため FTPS、 には別の認証情報を維持することをお勧めします。SFTP または からFTP認証情報を分離することでFTPS、FTP認証情報が共有または公開された場合、 を使用するワークロード、SFTPまたは は安全FTPSになります。

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、 の値によって異なります。 HomeDirectoryType フィールド。

- HomeDirectoryType に設定 – PATHHomeDirectoryフィールドは、ユーザーに表示される絶対 Amazon S3 バケット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、 で説明されている値と同様に、目的のエントリ/ターゲットマッピングを提供する HomeDirectoryDetails フィールドを設定します。 HomeDirectoryDetails サービスマネージドユーザーの パラメータ。

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

ID プロバイダー AWS Secrets Manager として を使用するには、サンプル AWS CloudFormation テンプレートの 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 インターフェイスが、公開 CloudFormationされたテンプレートに改善されました。テンプレートでは、APIゲートウェイでBASE64エンコード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の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RequestParameeters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. AWS CloudFormation コンソールを https://console.aws.amazon.com/cloudformation で開きます。
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. 以下を確認します。
   - 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 スタックを更新する方法の詳細については、AWS CloudFormation「 ユーザーガイド」の「スタックテンプレートの変更」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には、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するわかりやすい名前を使用する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、およびEFSファイルシステム名をエンドユーザーに開示することは避けることができます。

> **Note**
>
> セッションポリシーやその他の内部要件の使用によって異なりますが、通常、ユーザーが意図したファイルのみにアクセスできるようにするために、セッションポリシーと論理ディレクトリの両方は必要ありません。論理ディレクトリを設定する場合は、セッションポリシーも作成しないでください。両方があると、アクセス許可が拒否されるエラーが発生する可能性があります。

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できたはずです。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場合は、明示的に Typeを 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 を呼び出しAPI、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。マッピングが多い場合は、次のようにマッピングのサイズを計算できます。
  1. 一般的なマッピングを 形式で書き出します。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 (1) を追加します。
  3. その数に、サーバー用に持っているマッピングのおおよその数を掛けます。

ステップ 3 で推定した数が 2.1 MB 未満の場合、マッピングは許容制限内です。

- 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) がこ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` パラメータの値を使用している場合、この値は`EntryTarget`ペアによって暗黙的に暗示されるため、`LOGICAL`を`HomeDirectoryType`パラメータを指定しないでください。
- ターゲットはスラッシュ (/) 文字で始ま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した場合、ストレージに Amazon S3 を使用している場合、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 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。
- ユーザーに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、 に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)、ルートディレクトリは起動時に存在する必要があります。Amazon S3 の場合、ルートフォルダを作成するには、スラッシュ (/) で終わるゼロバイ

トオブジェクトを既に作成し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では、ユーザーのIAMロールマッピングにそれらにアクセスするアクセス許可がある限り、複数のバケットまたはEFSファイルシステムを含む S3 バケットまたはファイルシステムの任意の場所にターゲットを持つ複数のEntryTargetペアリングを作成できます。

次の仮想構造の例では、ユーザーが にログインすると AWS SFTP、 /pics、、/doc、 /reporting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)、ユーザーまたは管理者がディレクトリが存在しない場合は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 の場合EFS、管理者は論理マッピングまたは / ディレクトリを作成する必要があります。

```
[
{"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chrootの設定の詳細については、 AWS ストレージブログの「chroot と論理ディレクトリで構造を簡素化する AWS SFTP」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 の論理ディレクトリを設定する EFS

Transfer Family サーバーが Amazon を使用している場合EFS、ユーザーが論理ホームディレクトリで作業する前に、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": 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との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可能なエンドポイントを提供するリモートSFTPサーバーへの接続にのみ使用できます。

Transfer Family AWS Transfer Family SFTP コネクタの概要についてはSFTP、コネクタを参照してください。

トピック

- SFTP コネクタの設定
- SFTP コネクタを使用してファイルを送信および取得する
- SFTP コネクタの管理

SFTP コネクタの設定

このトピックでは、SFTPコネクタを作成する方法、コネクタ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シークレットを保存する方法、プライベートキーのフォーマットの詳細、コネクタをテストする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 コネクタアルゴリズム
- 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を生成してフォーマットする
- SFTP コネクタをテストする

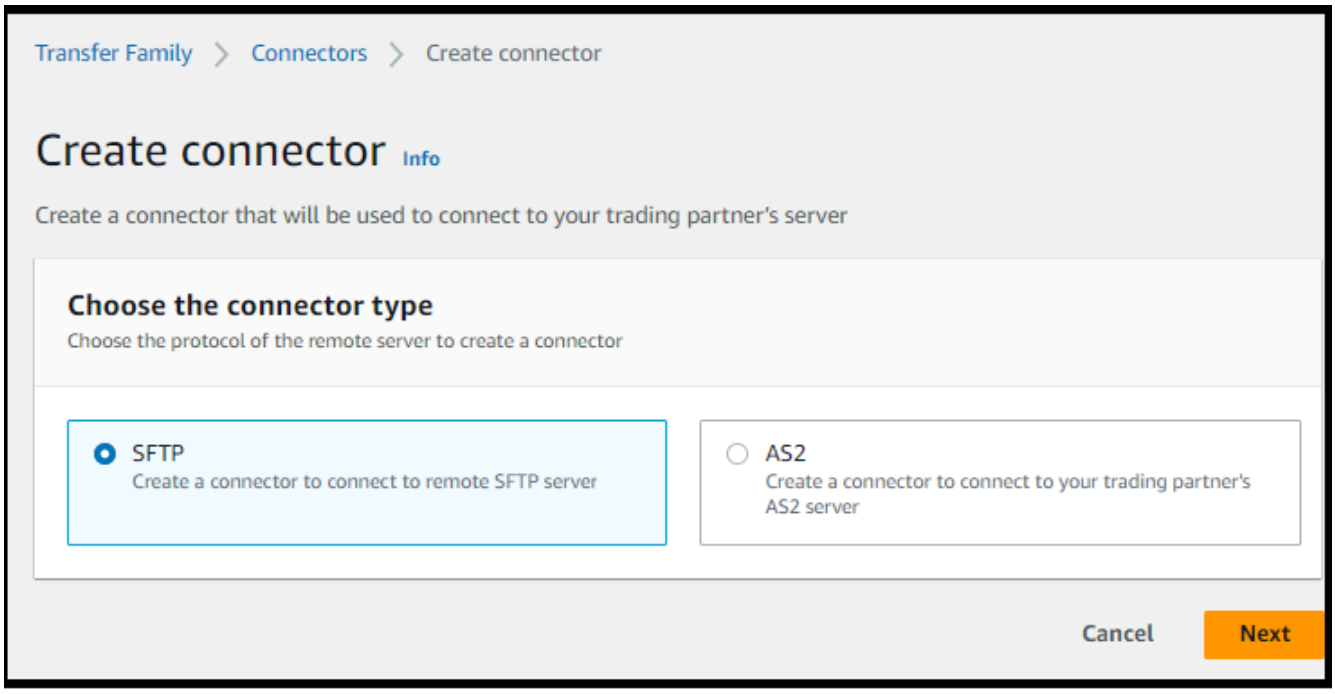
SFTP コネクタを作成する

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. で AWS Transfer Family コンソールを開きますhttps://console.aws.amazon.com/transfer/。
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. コネクタタイプSFTPを選択してSFTPコネクタを作成し、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。

   - にはURL、URLリモートSFTPサーバーの を入力します。これは`sftp://`*`partner-SFTP-server-url`*、 などの としてフォーマットURLする必要がありますsftp://AnyCompany.com。

> **Note**
>
> 必要に応じて、 にポート番号を指定できますURL。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 (ARN) ロールの Amazon リソースネーム AWS Identity and Access Management （ IAM) 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ARNの を指定する必要があります。にはシークレット名ARNが含まれていますが、名前に 6 つのランダムな英数字を追加します。シークレットARNの には、次の形式があります。
    >
    > ```
    > 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、 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。*DOC-EXAMPLE-BUCKET* Amazon S3 で、指定されたシークレットが Secrets Manager に保存されている。

```
{
  "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できます。複数のユーザーとシークレットに同じIAMロールを再利用する場合は、作業を保存できます。例えば、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 に保存することもできます AWS アカウント。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。

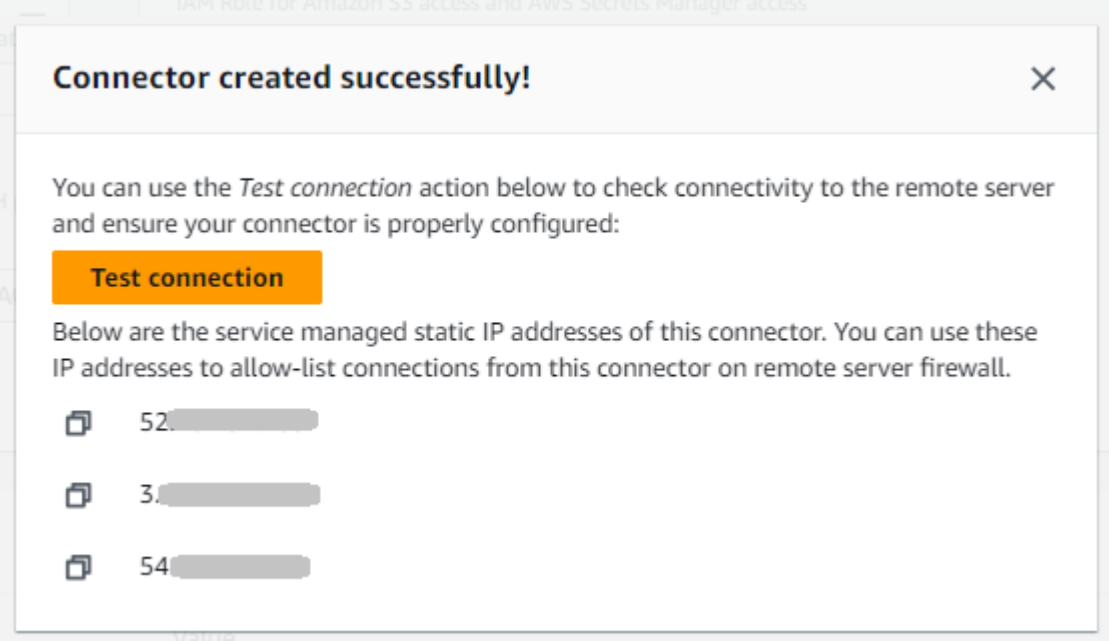
```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Sid": "SFTPConnectorPermissions",
        "Effect": "Allow",
        "Action": [
            "logs:CreateLogStream",
            "logs:DescribeLogStreams",
            "logs:CreateLogGroup",
            "logs:PutLogEvents"
        ],
        "Resource": [
            "arn:aws:logs:*:*:log-group:/aws/transfer/*"
        ]
    }]
}
```

5. SFTP 設定 セクションで、次の情報を指定します。

   - Connector 認証情報 の場合、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] — 外部サーバーの識別に使用されるホストキーの公開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 がサポートする信頼された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. （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。

7. 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を含む画面が表示されます。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。



Connectors ページが表示され、新しいSFTPコネクタの ID がリストに追加されます。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する を参照してください。

CLI

は、 create-connector コネクタを作成するコマンド。このコマンドを使用してSFTPコネクタを作成するには、次の情報を提供する必要があります。

- URL リモートSFTPサーバーの 。これは`sftp://`*`partner-SFTP-server-url`*、 などの としてフォーマットURLする必要がありますsftp://AnyCompany.com。
- アクセスロール 使用する (ARN) ロールの Amazon リソースネーム AWS Identity and Access Management （ IAM) 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ARNの を指定する必要があります。にはシークレット名ARNが含まれていますが、名前に 6 つのランダムな英数字を追加します。シークレットARNの には、次の形式があります。
>
> ```
> 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、 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。*DOC-EXAMPLE-BUCKET* Amazon S3 で、指定されたシークレットが Secrets Manager に保存されている。

```
{
  "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できます。複数のユーザーとシークレットに同じIAMロールを再利用する場合は、作業を保存できます。例えば、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 に保存することもできます AWS アカウント。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設定情報を入力します。
  - 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 ARNのシークレット。
  - 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、それをファイルに保存することです。例えば、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="
   ]
}
```

> **Note**
>
> SecretId は、シークレット全体ARNまたは名前 (*example-username-key* 前のリスト) 。

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
```

SFTP コネクタアルゴリズム

SFTP 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
SSH 暗号	aes256-gcm@openssh.com aes128-gcm@openssh.com aes256-ctr aes192-ctr
SSH キー交換方法 (KEX)	curve25519-sha256 curve25519-sha256@libssh.org diffie-hellman-group16-sha512 diffie-hellman-group18-sha512 diffie-hellman-group-exchange-sha256
SSH MAC	hmac-sha2-512-etm@openssh.com hmac-sha2-256-etm@openssh.com hmac-sha2-512 hmac-sha2-256
SSH ホストキー	ecdsa-sha2-nistp256 ecdsa-sha2-nistp384

タイプ	アルゴリズム
	ecdsa-sha2-nistp521 rsa-sha2-512 rsa-sha2-256

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認証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 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、 で AWS Secrets Manager コンソールを開きますhttps://console.aws.amazon.com/secretsmanager/。
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を生成してフォーマットする 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、このユーザー用にリモートSFTPサーバーに保存されているパブリックキーに対応する必要があります。

8. [Next (次へ)] を選択します。
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を生成してフォーマットする

パブリック/プライベートキー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 でのSSHキーの作成。

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 (置き換える *key_name* キーペアの実際のファイル名を含む):

```
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、RSAキーサイズが 4096 ビットの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、SFTPコネクタシークレットで使用するために Transfer Family が必要

とするものです。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埋め込み改行文字 (「\n」) でフォーマットされていることを確認する必要がありますJSON。

コマンドを使用して、既存のプライベートキーを正しい形式に変換します。形式JSONは改行文字が埋め込まれています。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

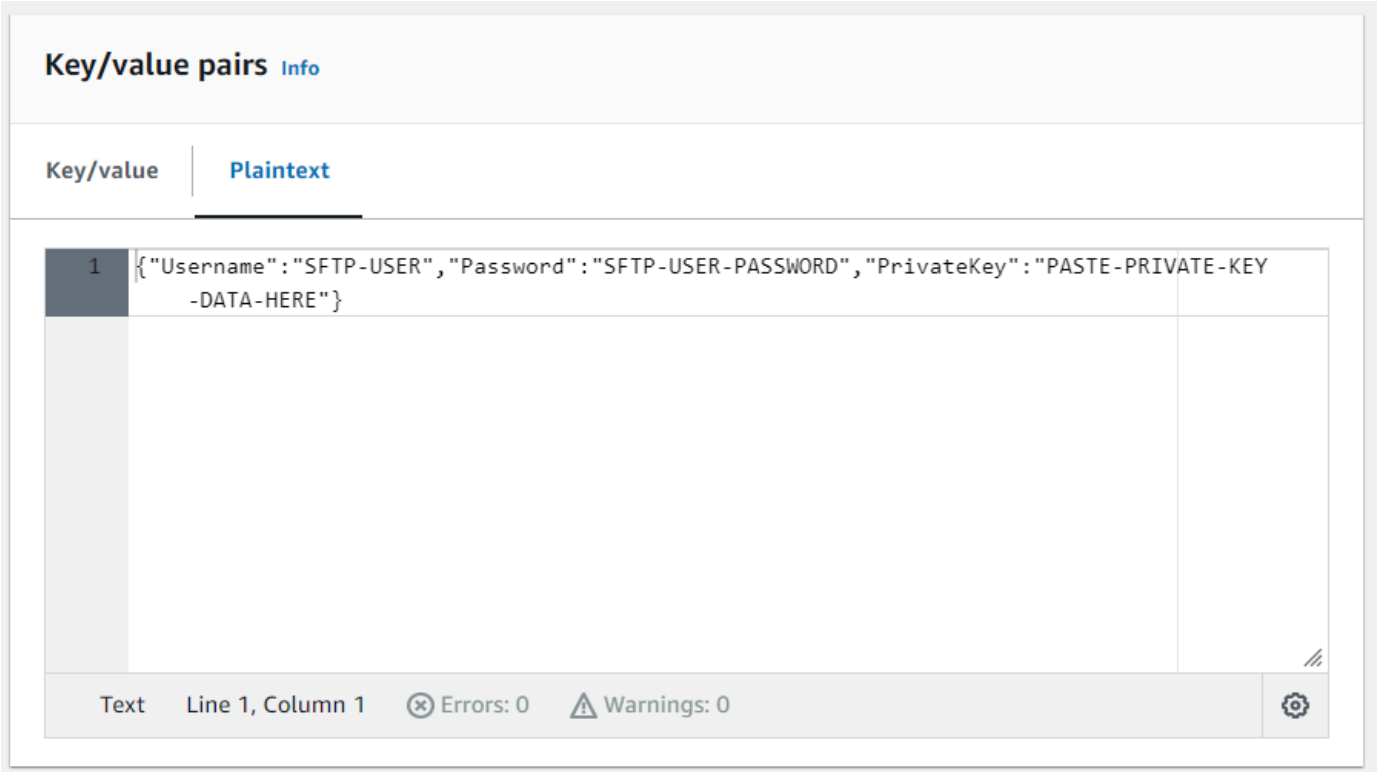
Windows を使用している場合は、 PowerShell を使用してキーを正しい形式に変換できます。次の Powershell コマンドは、プライベートキーを正しい形式に変換します。

```
Get-Content -Raw path-to-private-key-file | ConvertTo-Json
```

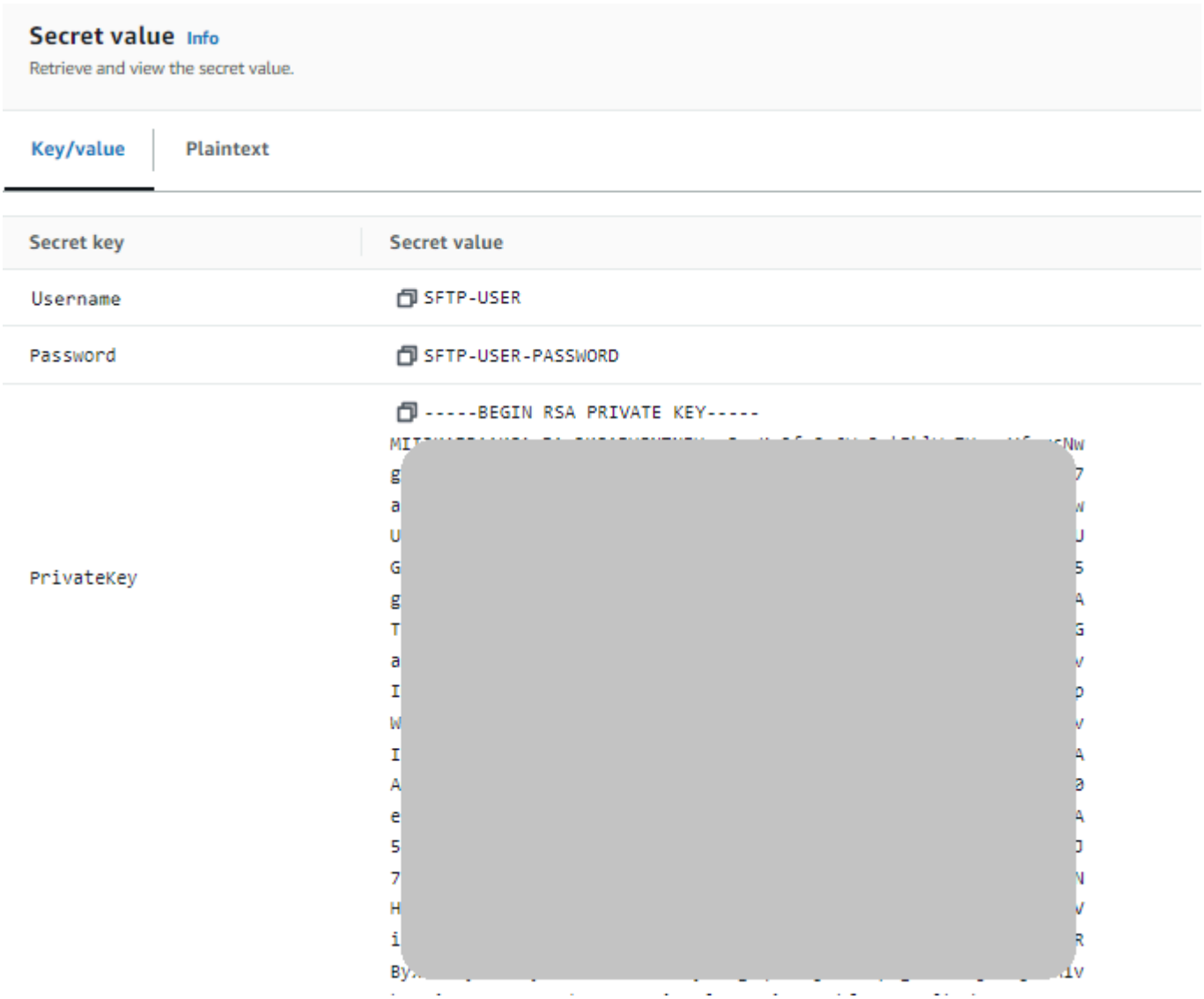
SFTP コネクタで使用するプライベートキーデータをシークレットに追加するには

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。

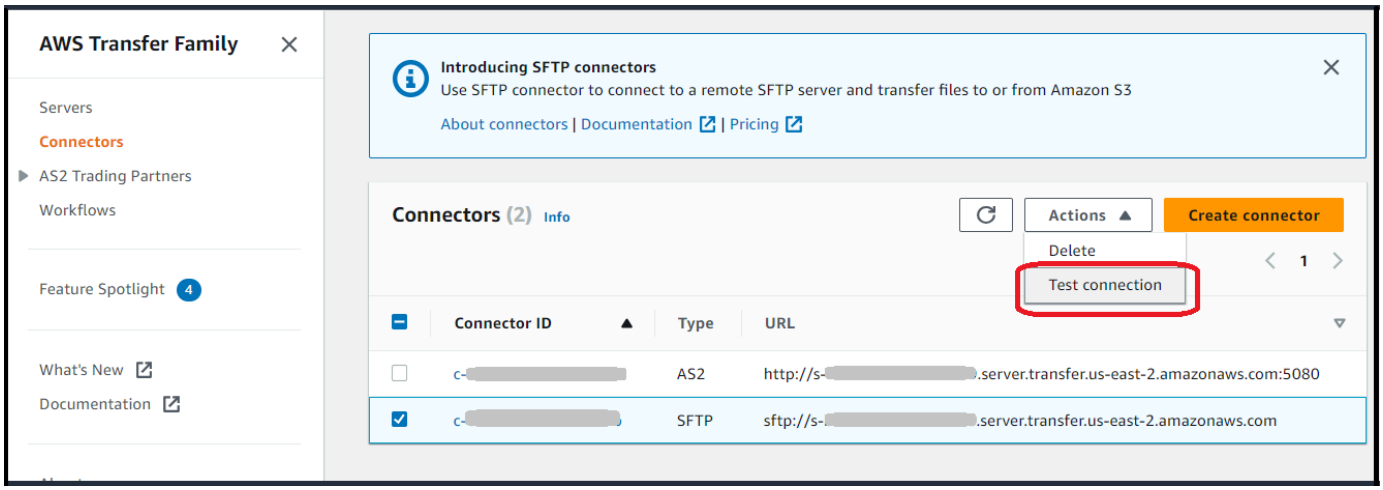
SFTP コネクタをテストする

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

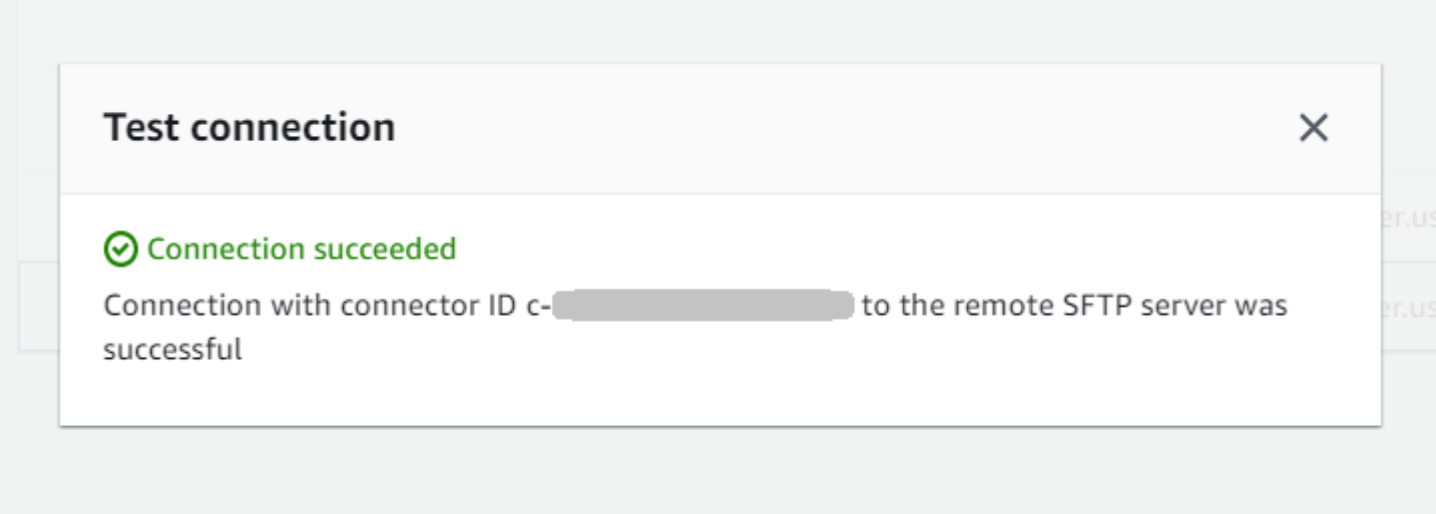
SFTP コネクタをテストするには

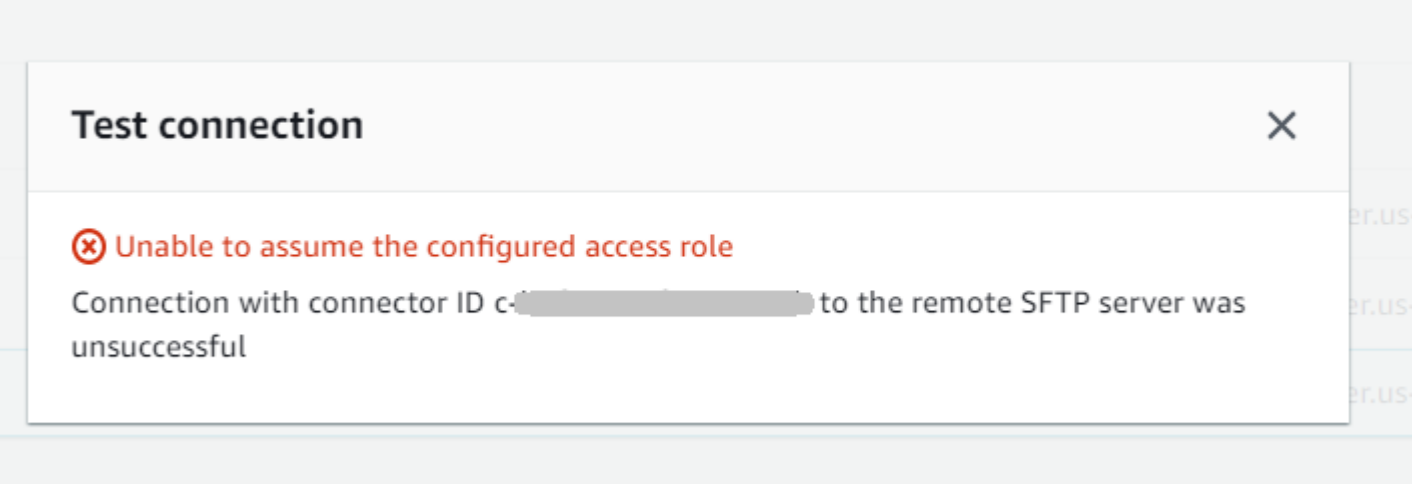
1. で AWS Transfer Family コンソールを開きますhttps://console.aws.amazon.com/transfer/。
2. 左側のナビゲーション ウィンドウで、[コネクタ] を選択し、コネクタを選択します。

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**
>
> を使用してコネクタAPIをテストするには、「」を参照してください。 TestConnection API ドキュメント。

SFTP コネクタを使用してファイルを送信および取得する

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 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、 を使用します。 StartFileTransfer API オペレーション。コネクタを使用して転送SFTPを実行します。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 コネクタの設定 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、お客様のSFTPサーバー上の に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。次のコマンド例では/*DOC-EXAMPLE-SOURCE-BUCKET*/file2.txt、Amazon S3 ストレージにある /*DOC-EXAMPLE-SOURCE-BUCKET*/file1.txtと 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 / に配置します。*DOC-EXAMPLE-BUCKET*/*prefix*。 このコマンド例を使用するには、 *user input placeholders*を独自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ます。相対パス、つまりSFTPユーザーのホームディレクトリを基準とするパスを使用することもできます。例えば、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 の場合/users/marymajor/、次のコマンドは を に送信/*DOC-EXAMPLE-SOURCE-BUCKET*/file1.txtします。 /users/marymajor/test-connectors/file1.txt

```
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
```

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。

トピック

- SFTP コネクタの更新
- SFTP コネクタの詳細を表示する
- SFTP コネクタのクォータ

SFTP コネクタの更新

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
```

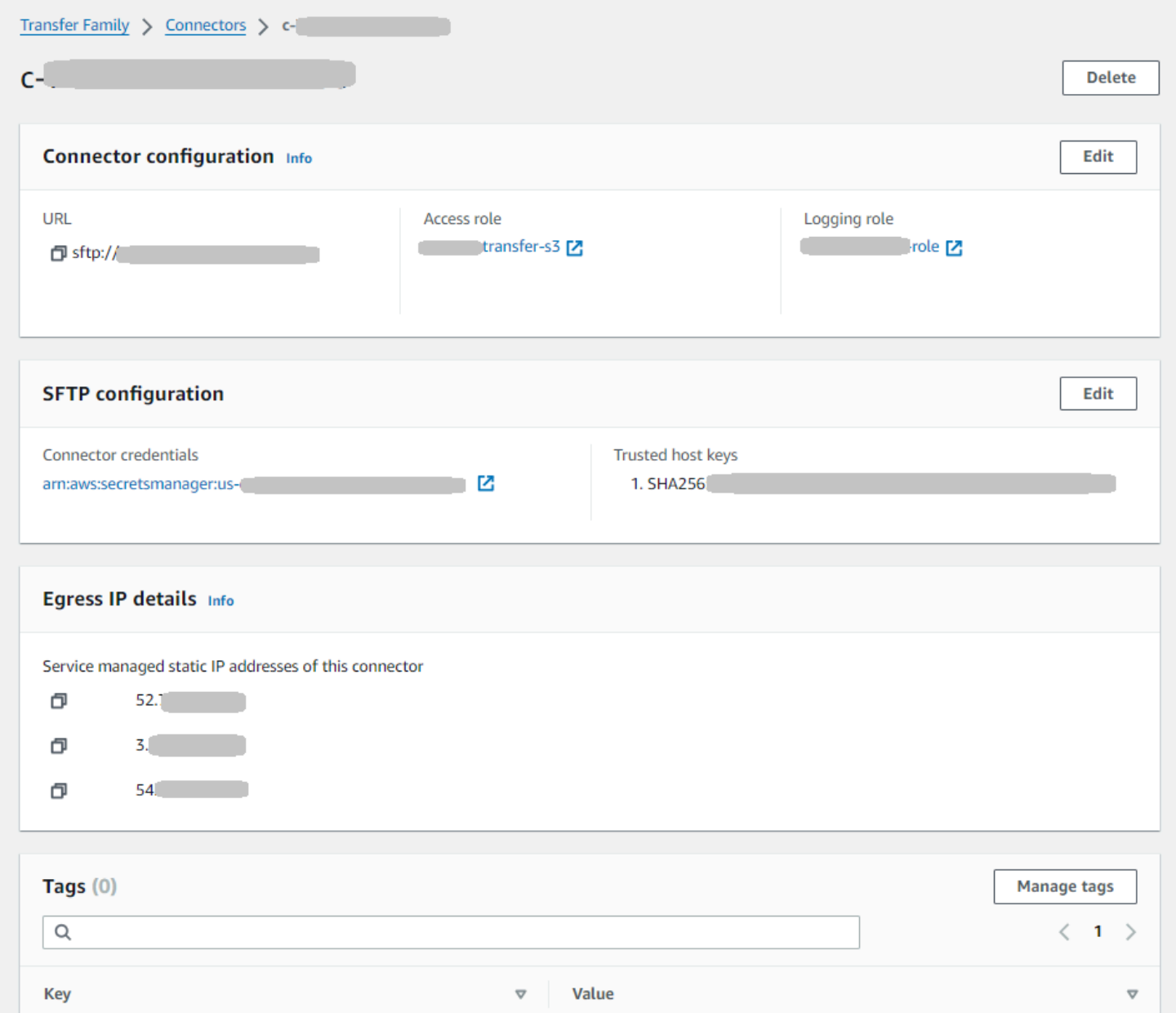
SFTP コネクタの詳細を表示する

SFTP 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。

コネクタの詳細を表示するには

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

コネクタのプロパティを変更するには、SFTPコネクタの詳細ページで編集を選択します。



> **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、「」を参照してくださいDescribeConnector API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あります。AS2 コネクタのクォータについては、「」を参照してくださいAS2 クォータと制限。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。

SFTP コネクタクォータ

名前	デフォルト	引き上げ可能
1 秒あたりのテスト接続トランザクションの最大数 (TPS)	1 アカウントにつき毎秒 1 リクエスト	不可
最大 StartFileTransfer TPS	1 アカウントにつき毎秒 3 リクエスト	可能
保留中のファイル転送の最大キューサイズ	1,000	不可
最大ファイルサイズ	50 ギビバイト (GiB)	不可
1 ファイルあたりの最大転送時間	6 時間	不可
1 ファイルあたりの最大リクエスト待機時間	6 時間	不可
最低AccessRole または LoggingRole セッション継続時間	60 分	不可
最大同時ファイル転送数	1コネクターあたりの1回の同時ファイル転送	不可
アカウントあたり、1 秒あたりの最大ファイル転送リクエスト数	3	可能
アカウントあたりのコネクタの最大数 (この数には SFTP	100	可能

名前	デフォルト	引き上げ可能
と AS2 コネクタの両方が寄与します)		
アカウントあたりのコネクタの最大帯域幅 (SFTPと AS2 コネクタの両方がこの値に寄与します)	50 MBps	不可

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場合、複数の目的でこれらのクォータが発生することがあります。

- 単一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 の AS2

適用性ステートメント 2 (AS2) は、強力なメッセージ保護と検証メカニズムを含む RFC定義済みのファイル送信仕様です。AS2 プロトコルは、プロトコルに組み込まれたデータ保護とセキュリティ機能に依存するコンプライアンス要件を持つワークフローにとって重要です。

> **Note**
>
> AS2 Transfer Family の は Drummond 認定です。

小売、ライフサイエンス、製造、金融サービス、ユーティリティなどの業界の顧客は、サプライチェーン、物流、支払いワークフローAS2に頼る 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、 で実行されるエンタープライズリソースプランニング (ERP) および顧客関係管理 (CRM) システムとの統合にも利用できます AWS。を使用するとAS2、既存のビジネスパートナーの統合とコンプライアンスを維持しながら AWS 、 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 のお客様が、 AS2が有効なサーバーを設定しているパートナーとファイルを交換する場合、セットアップには暗号化用に 1 つのパブリック/プライベートキーペアを生成し、パートナーとパブリックキーに署名して交換するための 1 つのパブリック/プライベートキーペアを生成します。

転送中のAS2ペイロードを保護するには、通常、暗号化メッセージ構文 (CMS) を使用し、一般的に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提供します。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、正常に復号されたことを検証 (否認なし) します。

これらのCMSペイロードとMDNレスポンスの転送は 経由で行われますHTTP。

> **Note**
>
> HTTPS AS2 サーバーエンドポイントは現在サポートされていません。TLS 現在、終了はお客様の責任となります。

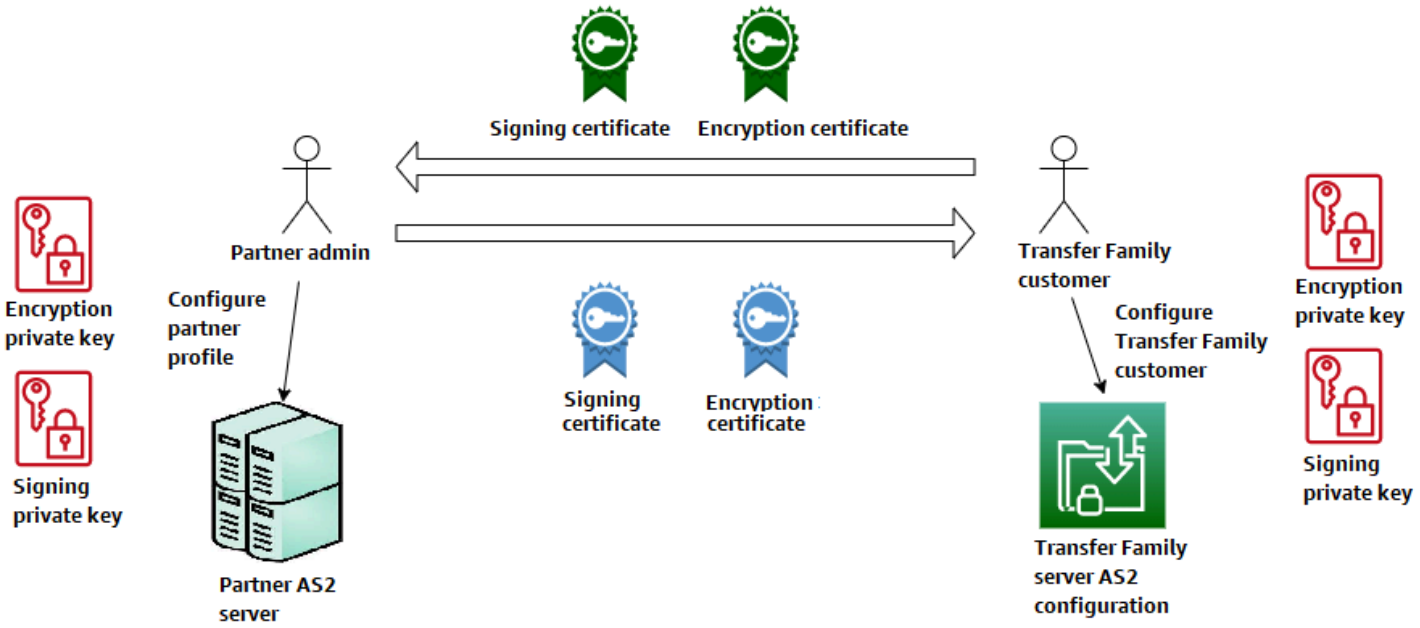
適用可能性ステートメント 2 (AS2) 設定の設定の詳細なチュートリアルについては、 step-by-step チュートリアル「」を参照してくださいAS2 設定の設定。

トピック

- AS2 ユースケース
- の設定 AS2
- AS2 コネクタの設定
- AS2 パートナーを管理する
- AS2 メッセージの送受信
- AS2 使用状況のモニタリング

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を交換する AWS Transfer Family 顧客の場合、セットアップの最も複雑な部分には、暗号化用に 1 つのパブリック/プライベートキーペアを生成し、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。



AWS Transfer Family を で使用するには、次のバリエーションを考慮してくださいAS2。

> **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ているパートナーです。

次の表MDNの のすべての記述は、署名付き を前提としていますMDNs。

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メッセージを取引先から 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. 取引先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メッセージを取引先から 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をインポートします (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) 。
- 暗号化されたAS2メッセージを取引先から 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、インバウンド転送のみを実行していますが、現在は署名付きペイロードを受け取るだけでなく、取引相手は署名付きMDNレスポンスを受け取ることを希望しています。

  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先に送信します。

アウトバウンドのみのユースケース

- Transfer Family サーバーから取引先に暗号化されたAS2メッセージを転送します。

このケースは、インバウンド専用転送のユースケースに似ています。ただし、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点が異なります。この場合、取引先のパブリックキーをプロファイルにインポートします。

- Transfer Family サーバーから取引先に暗号化されたAS2メッセージを転送し、署名を追加します。

  まだアウトバウンド転送のみを実行していますが、取引先から送信したメッセージに署名を求められたところです。

  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。
- Transfer Family サーバーから取引先に暗号化されたAS2メッセージを転送し、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実行していますが、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ことができるようになりました。

  1. 取引相手からパブリック署名キーが送信されます。
  2. 取引相手のパブリックキーをインポートします (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)。

インバウンドとアウトバウンドのユースケース

- Transfer Family サーバーと取引先間で、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. 取引先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. 取引先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して保存できます。
- Transfer Family サーバーと取引パートナー間で暗号化された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. 取引先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と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し、自分と取引相手の両方がMDNレスポンスを求めます。

  1. 取引相手からパブリック署名キーが送信されます。
  2. 取引相手のパブリックキーをインポートします (パートナープロファイルの署名証明書として)。
  3. 公開キーを取引先に送信します。

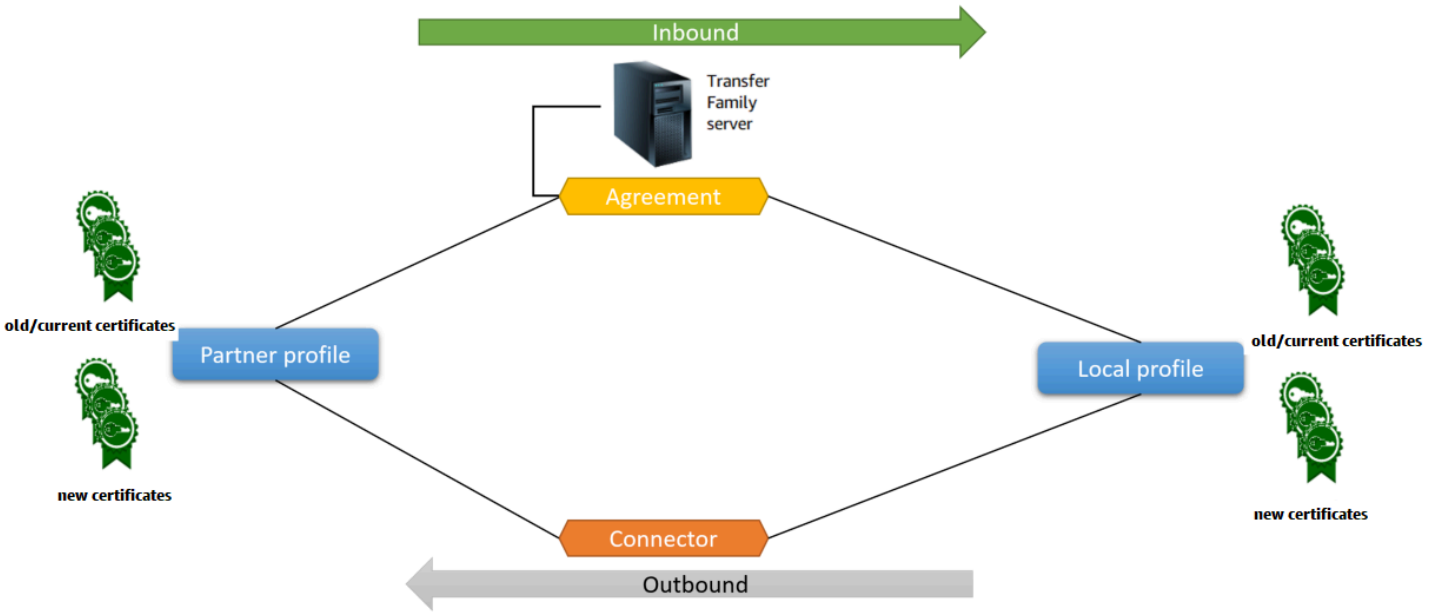
の設定 AS2

AS2対応サーバーを作成するには、次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) を交換する二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は契約を使用します。
- 証明書 – パブリックキー (X.509) 証明書は、メッセージの暗号化と検証のためにAS2通信に使用されます。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有効な Transfer Family 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、Transfer Family 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が有効なサーバーで必須ではありませんが、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、お客様の外部の非 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与するAS2オブジェクトの関係を示しています。



AS2 設定例については、 end-to-end「」を参照してくださいAS2 設定の設定。

トピック

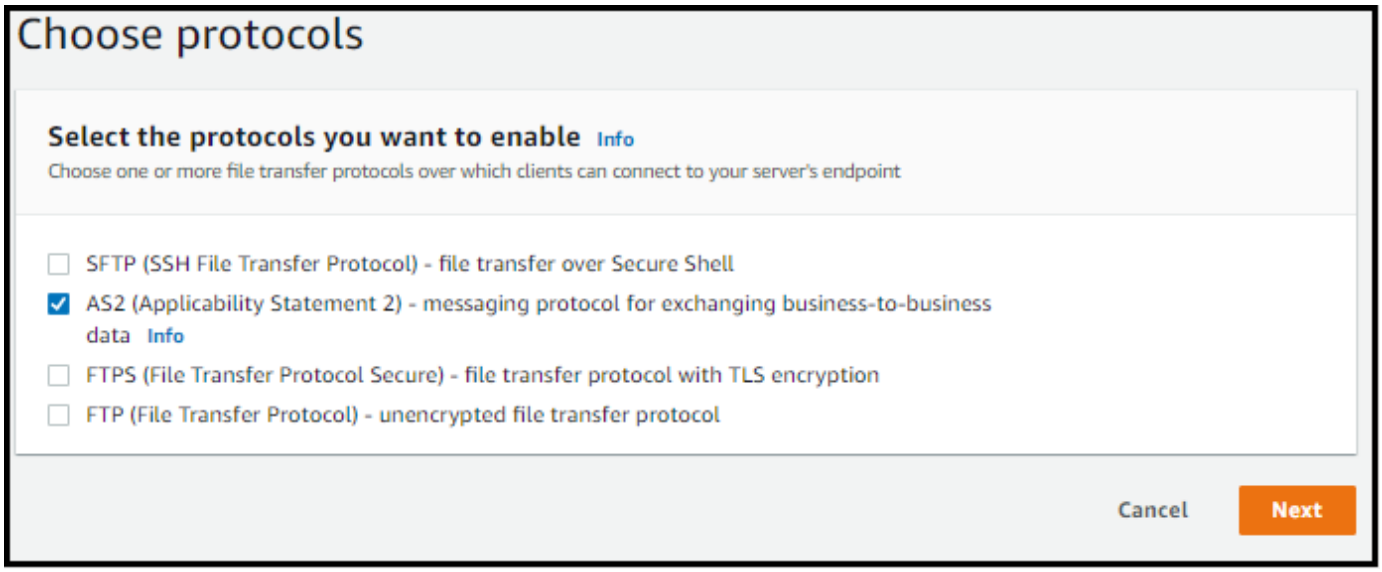
- Transfer Family コンソールを使用してAS2サーバーを作成する
- テンプレートを使用してデモ Transfer Family AS2スタックを作成する
- AS2 設定と制限
- AS2 機能と機能

Transfer Family コンソールを使用してAS2サーバーを作成する

この手順では、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

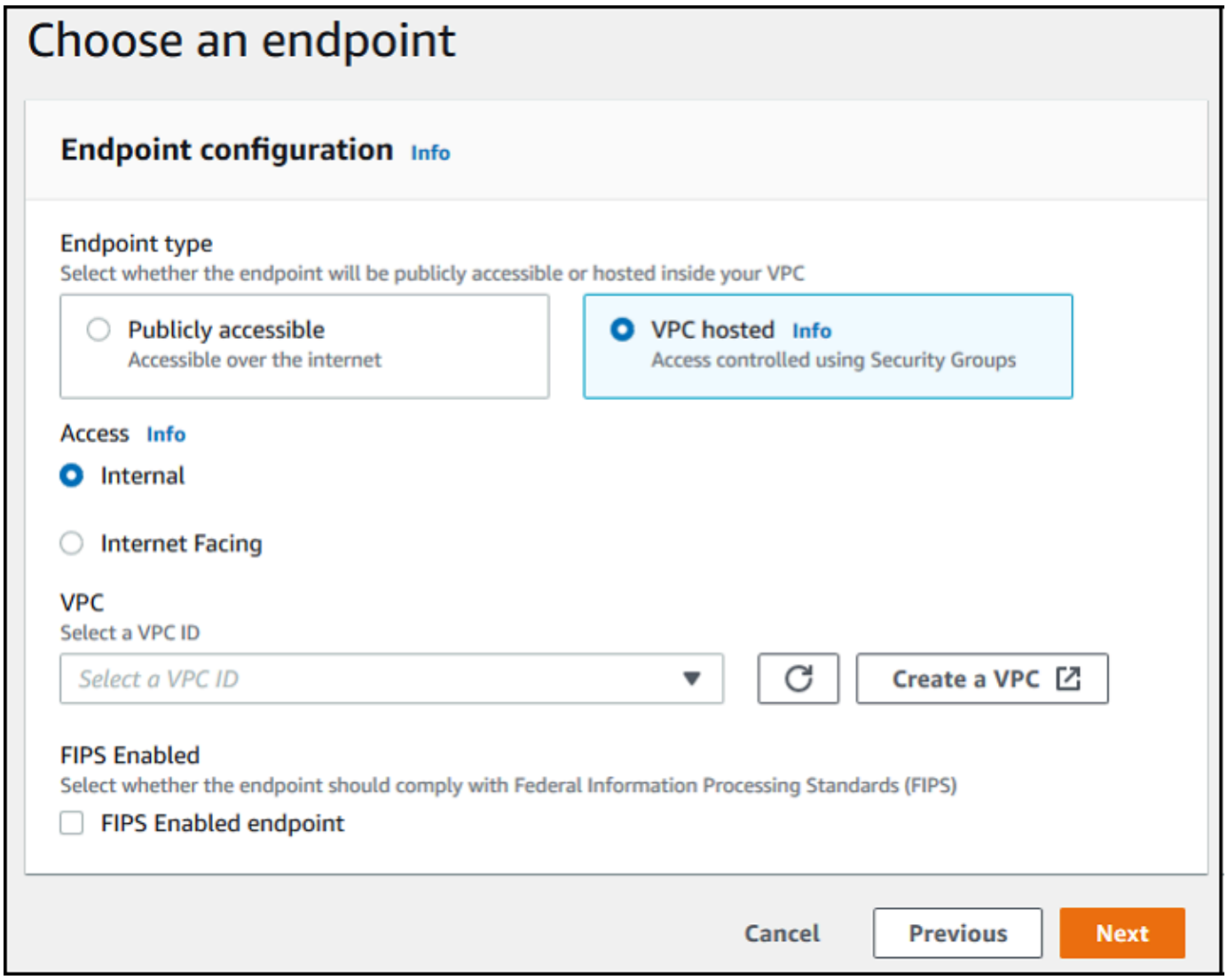
1. で AWS Transfer Family コンソールを開きますhttps://console.aws.amazon.com/transfer/。
2. 左のナビゲーションペインで「サーバー」を選択し、「サーバーの作成」を選択します。
3. プロトコルの選択ページで (AS2適用性ステートメント 2) を選択し、次へ を選択します。



4. [ID プロバイダーの選択] ページで、[次へ] を選択します。

> **Note**
>
> ではAS2、基本認証がAS2プロトコルで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. エンドポイントタイプ では、VPCホストを選択してサーバーのエンドポイントをホストします。ホストされたエンドポイントの設定については、VPC「」を参照してくださいVirtual Private Cloud でサーバーを作成する。

> **Note**
>
> パブリックにアクセス可能なエンドポイントは、AS2プロトコルではサポートされていません。インターネット経由でVPCエンドポイントにアクセスできるようにするには、 アクセス でインターネットフェイシングを選択し、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、 または 経由で AWS Direct Connect オンプレミスデータセンターなど、 VPCおよび VPC接続された環境内からアクセスできるようになりますVPN。
- インターネットと向き合う – このオプションを選択すると、インターネット経由で、AWS Direct Connect または 経由のオンプレミスデータセンターなど、 VPCおよび VPC接続された環境内からアクセスできますVPN。

  [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. ではVPC、既存の を選択するVPCか、作成VPCを選択して新しい を作成しますVPC。

d. FIPS 有効 の場合、FIPS有効エンドポイントチェックボックスはオフのままにします。

> **Note**
>
> FIPSが有効なエンドポイントは、 AS2 プロトコルではサポートされていません。

e. [Next (次へ)] を選択します。

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。

   [Next (次へ)] を選択します。

7. [追加詳細の設定] ページで、必要な設定を選択します。

> **Note**
>
> とともに他のプロトコルを設定する場合はAS2、すべての追加の詳細設定が適用されます。ただし、AS2プロトコルでは、適用される設定はCloudWatch ログ記録とタグセクションの設定のみです。

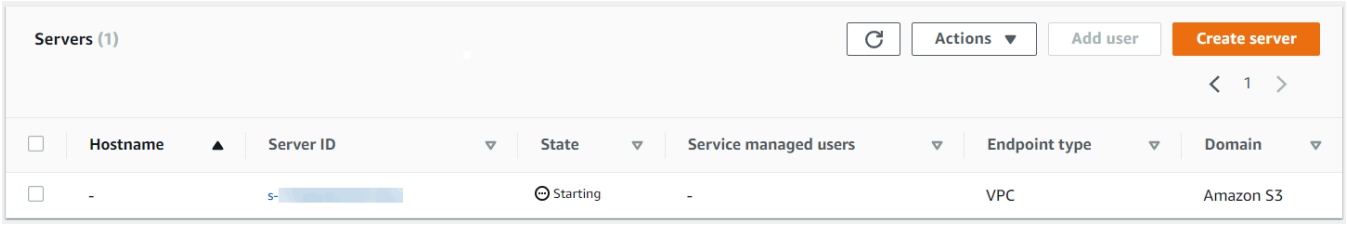
CloudWatch ログ記録ロールの設定はオプションですが、メッセージのステータスを確認し、設定に関する問題のトラブルシューティングができるように、設定することを強くお勧めします。

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デモ Transfer Family AS2スタックを作成する

がAS2有効な 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ます。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、本番稼働AS2用サーバーで使用する証明書については、Secrets Manager を使用して、保存された証明書を管理および定期的にローテーションすることもできます。

- 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対応の Transfer Family サーバーを作成するには

1. AWS CloudFormation コンソールを https://console.aws.amazon.com/cloudformation で開きます。
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、AS2デモテンプレート をコピーし、Amazon S3 URLフィールドに貼り付けます。
6. [Next (次へ)] を選択します。
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。

   - の下にAS2、ローカル AS2 ID とパートナー AS2 ID の値を入力するか、それぞれデフォルト `local`と を受け入れます`partner`。
   - 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 を受け入れます`0.0.0.0/0`。

     > **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. [Next (次へ)] を選択します。[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**
>
> ページの下部の 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。

スタックを作成したら、 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 コマンドは、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 Outputs 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JSONファイルとMDNレスポンスを記述する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、それ自体に `failed/`、`processed/`、および `processing/`フォルダが含まれます。*`/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 および MDN ファイルが含まれます。
- `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 設定と制限

このトピックでは、承認された暗号とダイジェストを含む、適用可能性ステートメント 2 (AS2) プロトコルを使用する転送でサポートされている設定、機能、および機能について説明します。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

- AS2 サポートされている設定
- AS2 クォータと制限

AS2 サポートされている設定

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 (HTTP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) や `X-Forwarded-Proto: https`ヘッダーなどのTLS終了プロキシによって転送された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 のみですZLIB)
- メッセージ処理通知 (MDN) – オプション

暗号

インバウンド転送とアウトバウンド転送の両方で、次の暗号が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、次のような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唯一の転送HTTPであり、明示的に指定する必要があります。

> **Note**
>
> インバウンド転送HTTPSに を使用する必要がある場合は、Application Load Balancer または Network Load Balancer TLSで終了できます。これについては、「経由でAS2メッセージを受信する HTTPS」で説明しています。

- アウトバウンド転送 - を指定する場合はHTTPURL、暗号化アルゴリズムも指定する必要があります。を指定する場合はHTTPSURL、暗号化アルゴリズムNONEに を指定するオプションがあります。

AS2 クォータと制限

このセクションでは、 のクォータと制限について説明します。 AS2

トピック

- AS2 クォータ
- シークレットの処理クォータ
- 既知の制限事項

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
サーバーあたりのインバウンドAS2リクエスト	1 秒あたり 25	不可
サーバーあたりのインバウンドAS2リクエストの進行中	100	不可
ファイル転送リクエストあたりの最大ファイル数	10	不可

名前	デフォルト	引き上げ可能
コネクタごとに進行中のアウトバウンドAS2リクエスト	100	不可
最大ファイルサイズ (圧縮または非圧縮)	50 MiB	可能
非アクティブタイムアウト	350 秒	不可
アカウントあたりのパートナープロファイルの最大数	1000 (パートナープロファイルごとに最大 10 個の証明書: 調整不可)	可能
アカウントあたりの証明書の最大数	1,000	可能
アカウントあたり、1 秒あたりの最大ファイル転送リクエスト数	3	可能
アカウントあたりのコネクタの最大数 (この数には SFTP と AS2 コネクタの両方が寄与します)	100	可能
アカウントあたりのコネクタの最大帯域幅 (SFTPと AS2 コネクタの両方がこの値に寄与します)	50 MBps	不可
サーバーあたりの契約の最大数	100	可能

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顧客 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、 AWS アカウント内のすべてのサービスで共有されます。

Secrets Manager の場合`GetSecretValue`、適用されるクォータは、AWS Secrets Manager クォータ で説明されているように、 DescribeSecret および GetSecretValue API リクエストの複合レートです。

Secrets Manager **`GetSecretValue`**

名前	値	説明
DescribeSecret と 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 および GetSecretValue APIリクエストの合計 1 秒あたりの最大トランザクション数。

の場合 AWS KMS、次のクォータが に適用されます`Decrypt`。詳細については、「各 AWS KMS APIオペレーションのクォータのリクエスト」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

クォータ名	デフォルト値 (1 秒あたりのリクエスト)
	• アジアパシフィック (東京)、ap-northeast-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 keep-alive 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。
- サービスがアクティブな契約を受け入れ、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。
- for からメッセージを受信するサーバーは、RFC6211 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 AWS Transfer Family AS2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。
- メッセージが重複IDsすると、処理済み/警告: ドキュメントメッセージが重複します。
- AS2 証明書のキーの長さは、少なくとも 2048 ビット、最大 4096 ビットである必要があります。
- メッセージAS2または非同期MDNsを取引先のHTTPSエンドポイントに送信する場合、メッセージまたは MDNs は、パブリックに信頼されたSSL認証局 (CA) によって署名された有効な証明書を使用する必要があります。自己署名証明書は現在、サポートされていません。

- 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 (で説明) のセキュリティポリシー AWS Transfer Family。
- 相互 TLS (m TLS) は現在サポートされていません。
- AS2 バージョン 1.2 の複数のアタッチメントと証明書交換メッセージング (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 機能と機能

次の表に、 を使用する Transfer Family リソースで使用できる機能と機能を示しますAS2。

AS2 機能

Transfer Family では、 に次の機能が用意されていますAS2。

機能	でサポート AWS Transfer Family
Drummond 認証	可能
AWS CloudFormation のサポート	可能
Amazon CloudWatch メトリクス	可能
SHA-2 暗号化アルゴリズム	可能
Amazon S3 のサポート	可能
Amazon のサポート EFS	不可
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	不可
Certificate Exchange Messaging (CEM)	不可
相互 TLS (m TLS)	不可

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スケジュール済みメッセージ

AS2 送受信機能

次の表に、送受信機能のリストを示します AWS Transfer Family AS2。

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 暗号化されたトランスポート (HTTPS)	はい [1]	可能
非TLSトランスポート (HTTP)	可能	はい [2]
同期 MDN	あり	可能
メッセージ圧縮	あり	可能
非同期 MDN	可能	不可
静的 IP アドレス	あり	可能
独自の IP アドレスの取得	可能	不可
複数のファイルアタッチメント	不可	なし
基本認証	不可	可能
AS2 再起動	該当しない	不可
メッセージあたりのカスタム件名	該当しない	不可

1. Network Load Balancer TLS で利用可能なインバウンド暗号化トランスポート (NLB)

2. アウトバウンド非トランスポート -TLS 暗号化が有効になっている場合にのみ使用可能

AS2 コネクタの設定

コネクタの目的は、Transfer Family サーバーから外部のパートナー所有の宛先にAS2ファイルを送信するアウトバウンド転送の取引パートナー間の関係を確立することです。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。

> **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、送信前にAS2メッセージがファイルをエンベロープにラップするため発生します。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。

適用性ステートメント 2 (AS2) コネクタは、アウトバウンド転送の取引パートナー間の関係を確立します。AS2 コネクタの詳細については、「」を参照してくださいAS2 コネクタの設定。

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 コネクタを作成するには

1. で AWS Transfer Family コンソールを開きますhttps://console.aws.amazon.com/transfer/。
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URLの を入力します。
   - アクセスロール – 使用する (ARN) ロールの Amazon リソースネーム AWS Identity and Access Management （ IAM) 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 の代わりに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の`kms:Decrypt`アクセス許可も必要です。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、必要なレガシークライアントをサポートする必要がある場合`DES_EDE3_CBC`を除いて、 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メッセージで`subject`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、 をプロトコル`HTTPS`として指定する必要があります。

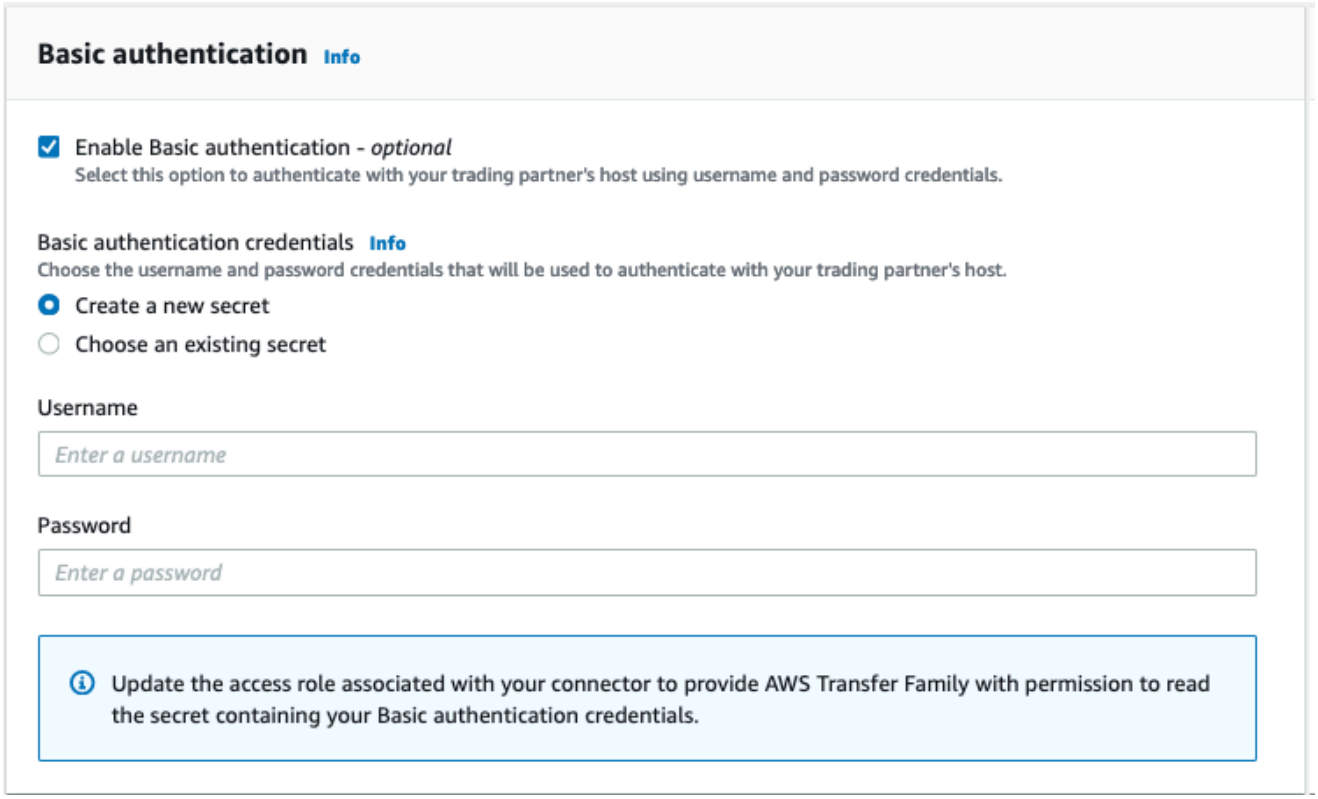
5. MDN 設定セクションで、次の情報を指定します。

   - リクエスト MDN – 経由でメッセージが正常に受信されたMDN後、取引相手に の送信を要求するオプションがありますAS2。
   - 署名済み MDN – MDNs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、リクエスト MDNを選択した場合のみ使用できます。

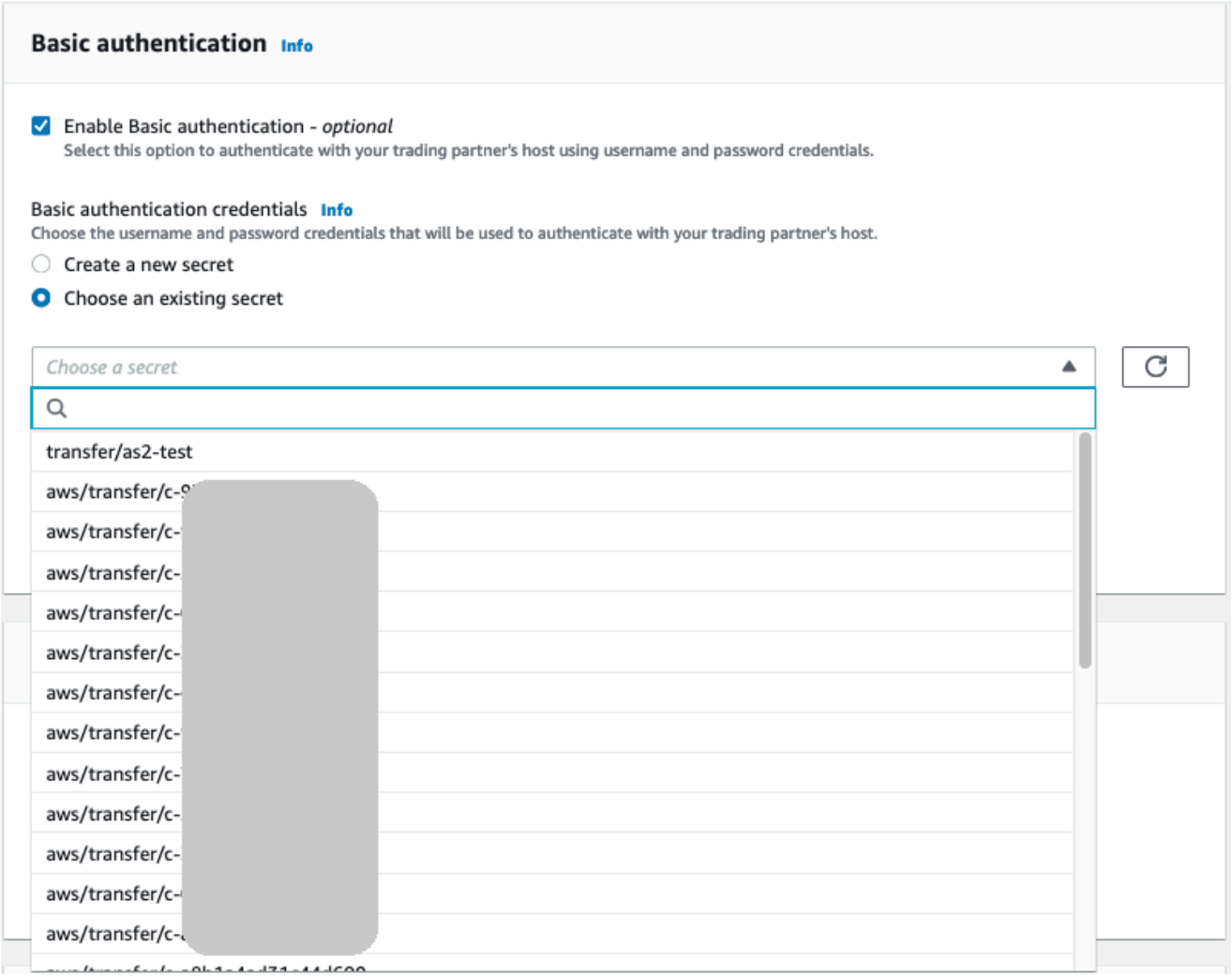
6. [基本認証] セクションで、以下の情報を指定します。

   - 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する 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 コネクタの詳細を表示する 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_ECDHEECDSA_WITH_AES_128_GCM_SHA256

タイプ	アルゴリズム
	TLS_ECDHERSA_WITH_AES_128_GCM_SHA256 TLS_ECDHEECDSA_WITH_AES_128_CBC_SHA256 TLS_ECDHERSA_WITH_AES_128_CBC_SHA256 TLS_ECDHEECDSA_WITH_AES_256_GCM_SHA384 TLS_ECDHERSA_WITH_AES_256_GCM_SHA384 TLS_ECDHEECDSA_WITH_AES_256_CBC_SHA384 TLS_ECDHERSA_WITH_AES_256_CBC_SHA384

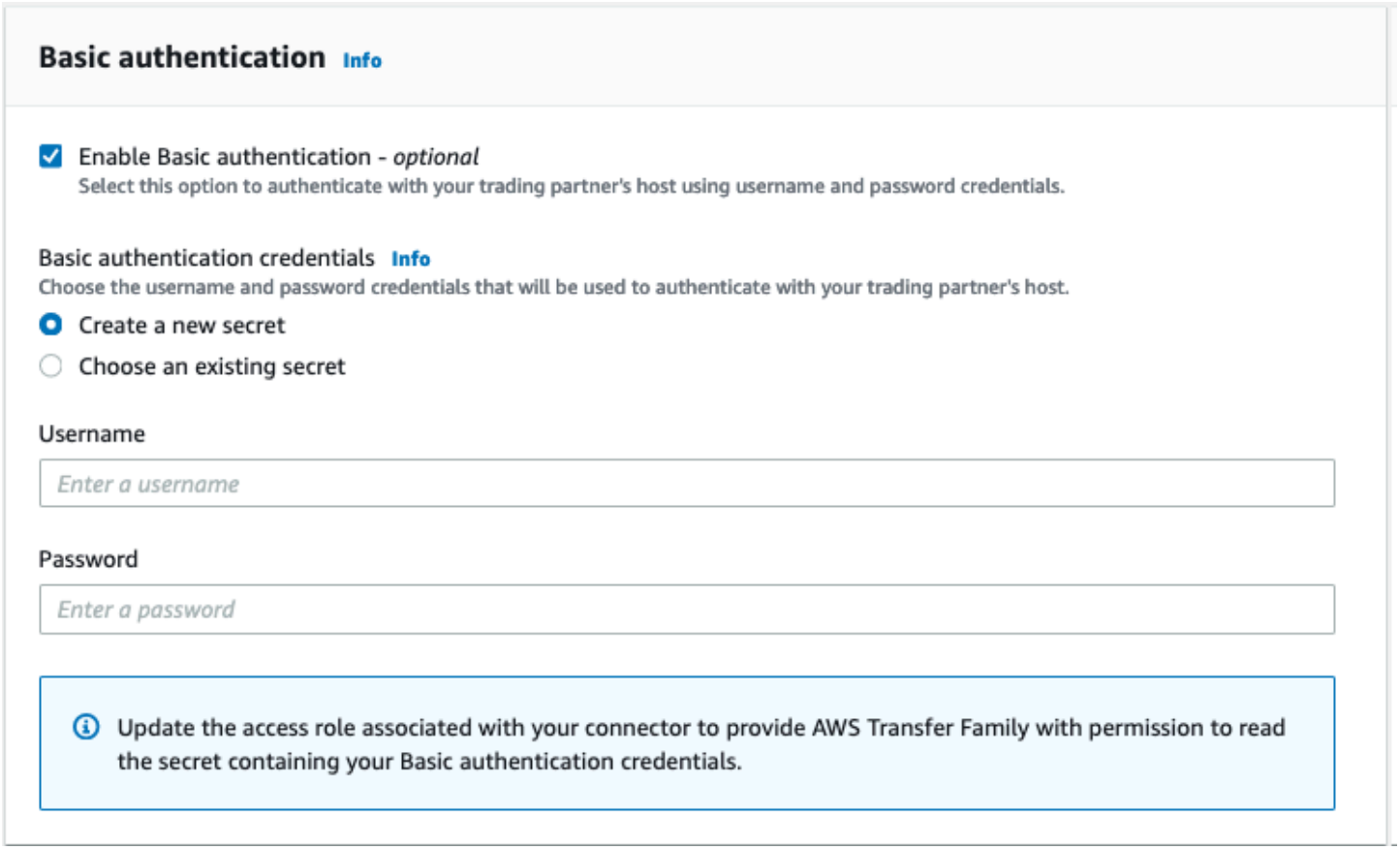
AS2 コネクタの基本認証

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きは、アウトバウンドメッセージの基本的な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。

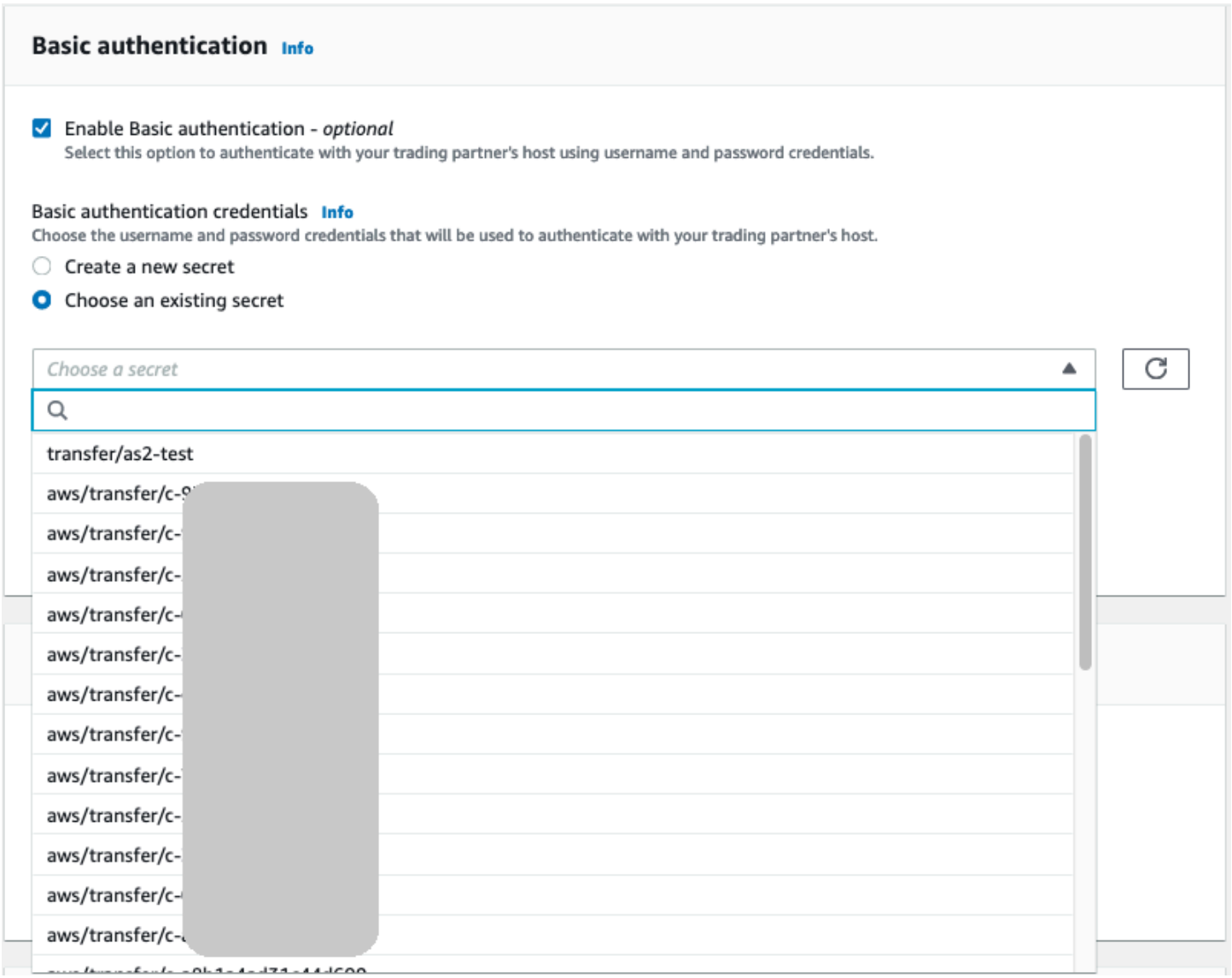
> **Note**
>
> 基本認証は、 を使用している場合にのみ使用できますHTTPS。

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する 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する

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か、 で作成したシークレットを使用できます AWS Secrets Manager。いずれの場合も、シークレットは Secrets Manager に保存されます。

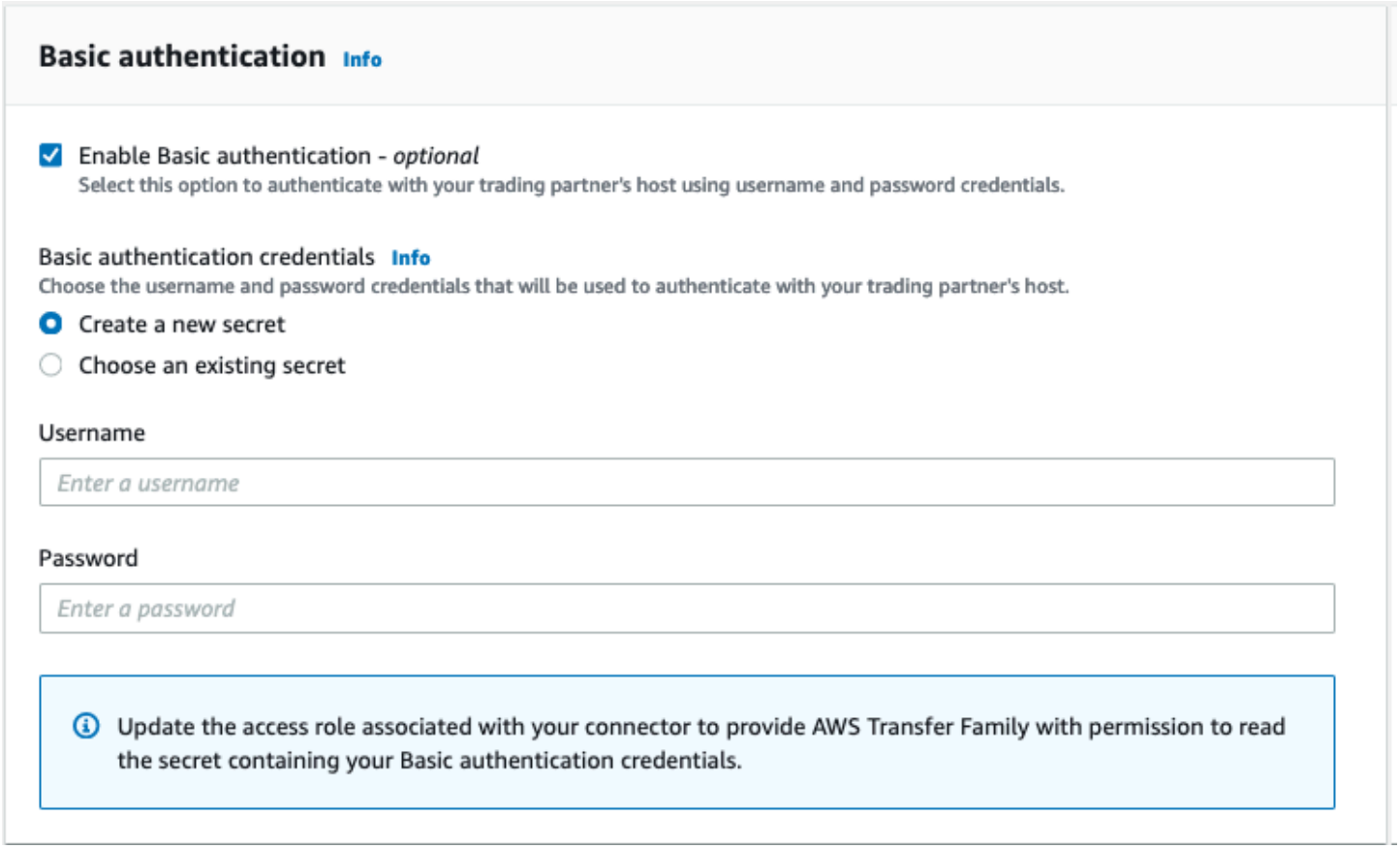
トピック

- 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。

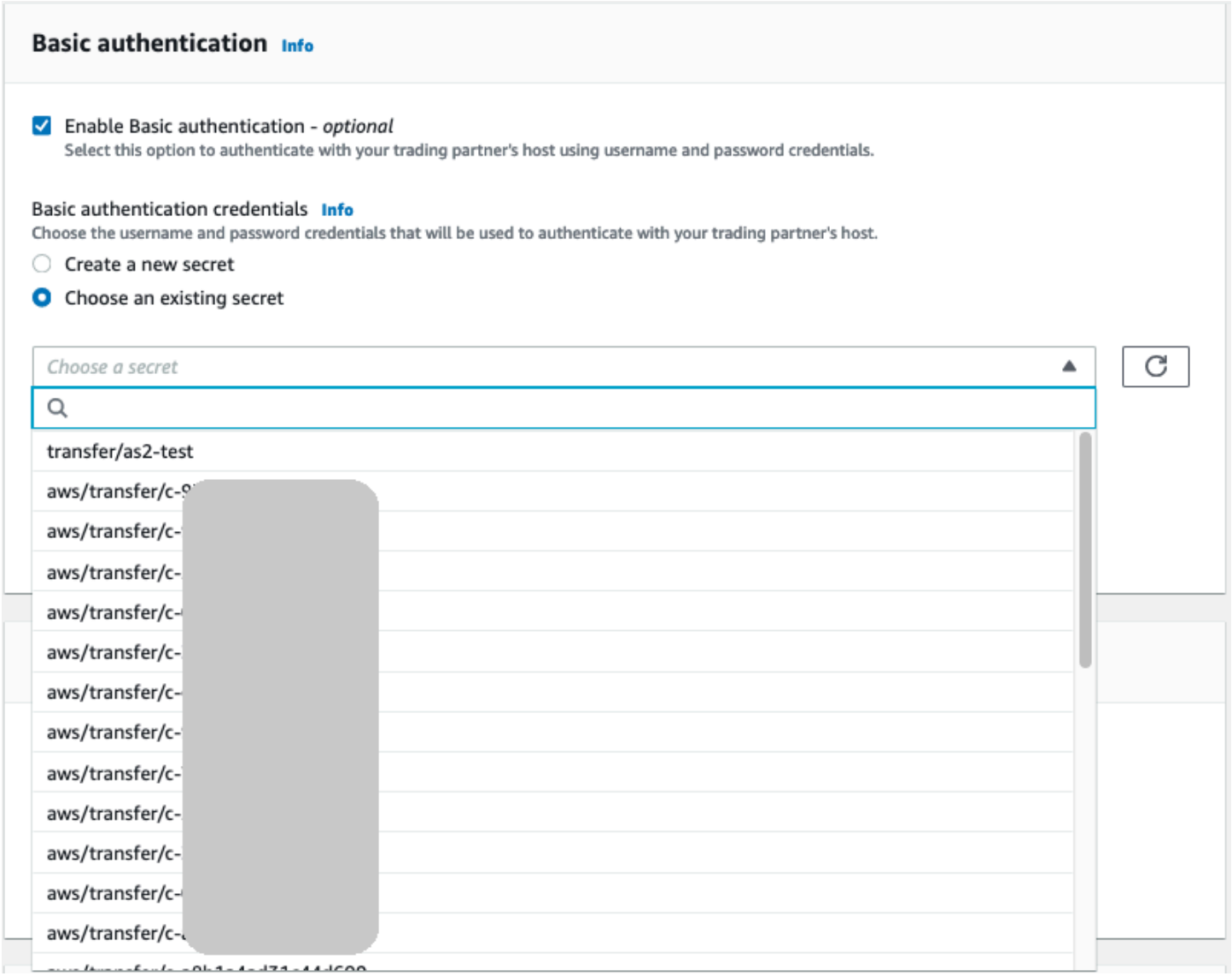


> **Note**
>
> コンソールで新しいシークレットを作成する場合、シークレットの名前は次の命名規則に従います: **`/aws/transfer/`*`connector-id`***、ここで *`connector-id`* は、作成するコネクタの ID です。 AWS Secrets Managerでシークレットを探す際には、この点を考慮してください。

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、 を使用している場合にのみ使用できますHTTPS。

AS2 基本認証用に 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、 で AWS Secrets Manager コンソールを開きますhttps://console.aws.amazon.com/secretsmanager/。
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、このユーザー用にリモートサーバーに保存されているパブリックキーに対応する必要があります。
8. [Next (次へ)] を選択します。
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 コネクタの設定 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 コネクタの詳細を表示する

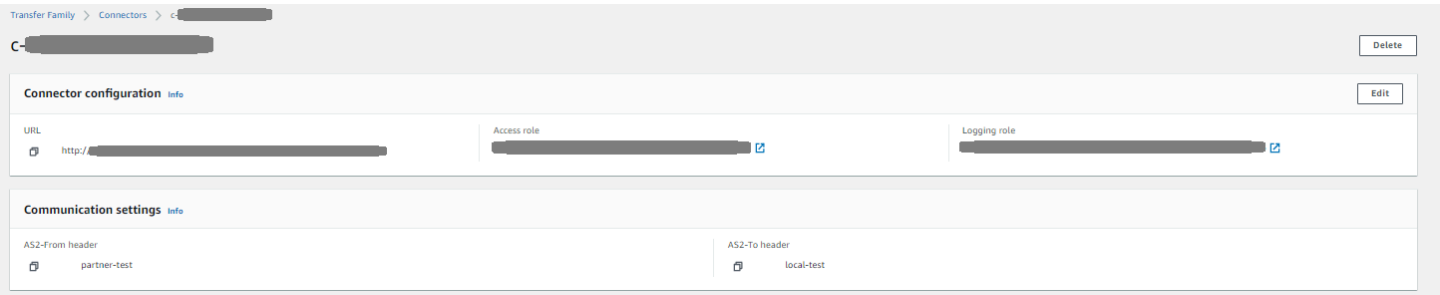
AS2 AWS Transfer Family コネクタの詳細とプロパティのリストは、 AWS Transfer Family コンソールで確認できます。AS2 コネクタのプロパティにはURL、、ロール、プロファイル、MDNs、タグ、モニタリングメトリクスが含まれます。

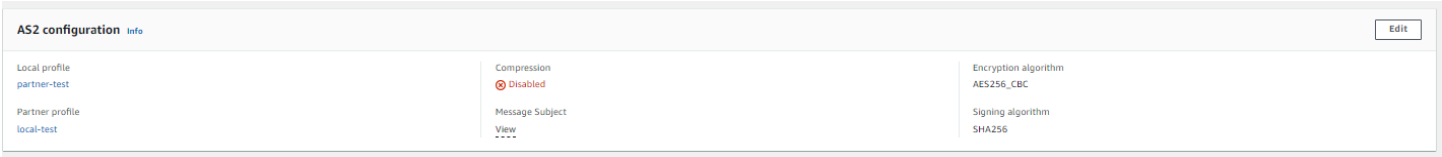
これはコネクタの詳細を表示する手順です。

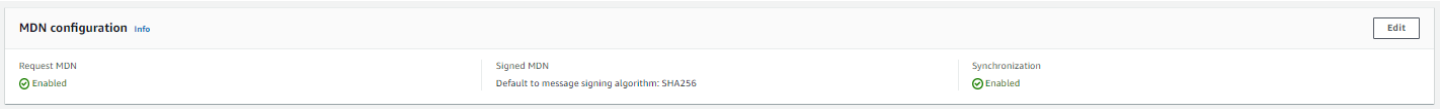
コネクタの詳細を表示するには

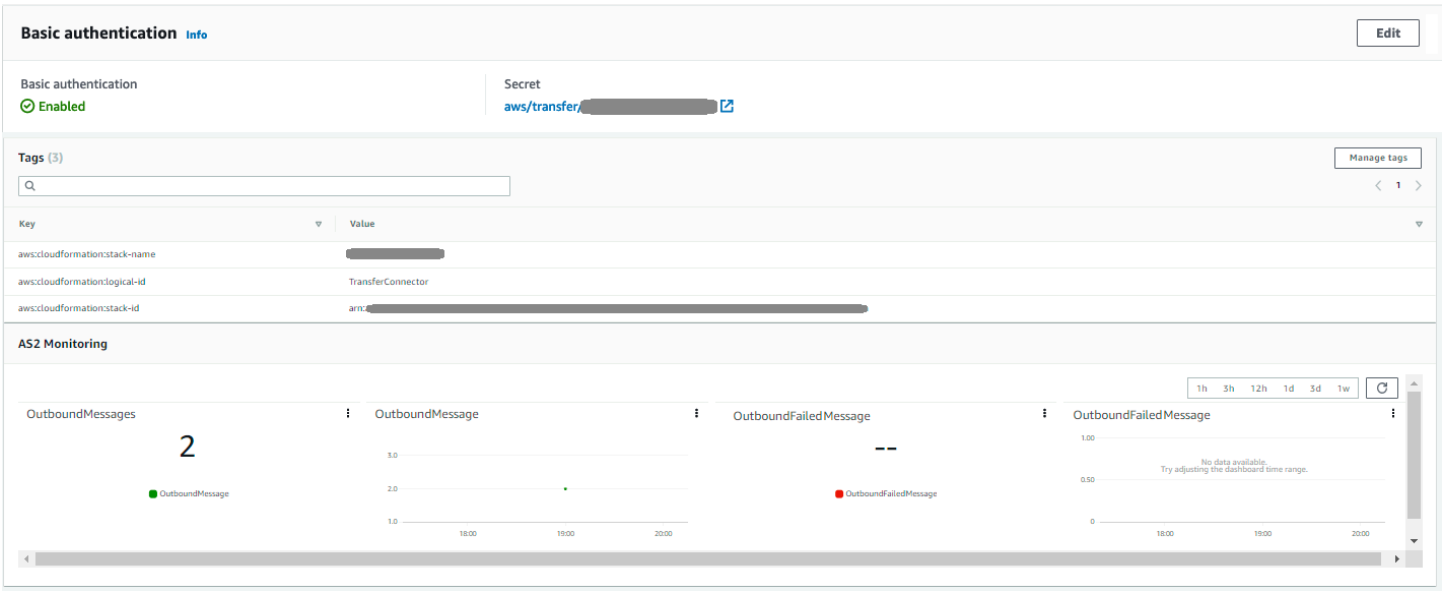
1. で AWS Transfer Family コンソールを開きますhttps://console.aws.amazon.com/transfer/。
2. 左側のナビゲーションペインで、[Connectors (コネクタ)] を選択します。
3. [コネクタ ID] 列の識別子を選択すると、選択したコネクタの詳細ページが表示されます。

コネクタの詳細ページでAS2コネクタのプロパティを変更するには、編集 を選択します。









> **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、「」を参照してくださいDescribeConnector API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、およびアグリーメントを管理する方法について説明します。

AS2 証明書をインポートする

Transfer Family AS2プロセスは、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 証明書のキーの長さは、少なくとも 2048 ビット、最大 4096 ビットである必要があります。

以下のポイントは、プロセス中にAS2証明書がどのように使用されるかについて詳しく説明します。

- 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。

証明書を作成する方法については、the section called “ステップ 1: の証明書を作成する AS2” 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有効な証明書を指定するには

1. で AWS Transfer Family コンソールを開きますhttps://console.aws.amazon.com/transfer/。
2. 左側のナビゲーションペインのAS2取引パートナー で、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。

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（認証局によって管理）	ACTIVE DATE (Transfer Family によって設定）	INACTIVE DATE (Transfer Family によって設定）	NOT AFTER（認証局によって設定）
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。この手順では、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. で AWS Transfer Family コンソールを開きますhttps://console.aws.amazon.com/transfer/。
2. 左側のナビゲーションペインのAS2取引パートナー で、プロファイル を選択し、プロファイルの作成 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ヘッダーas2-fromと、使用する証明書を決定する取引パートナーシップas2-toを識別するために使用されます。
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 証明書をインポートする 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。

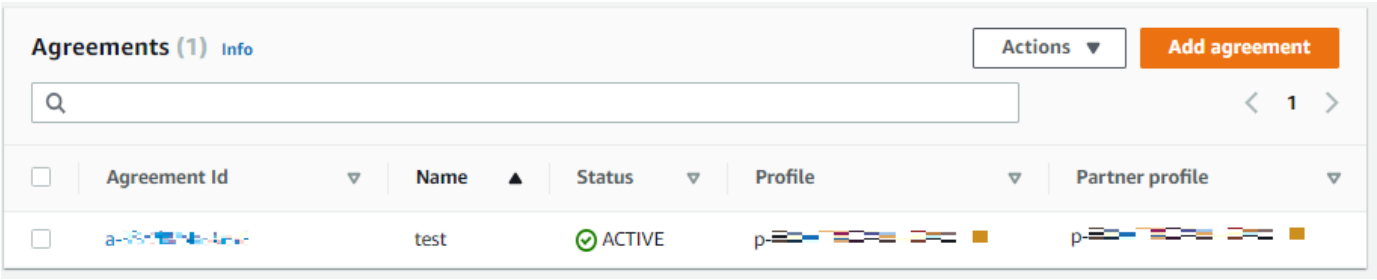
AS2 アグリーメントの作成

契約はTransfer Family サーバーに関連付けられています。これらは、AS2プロトコルを使用してメッセージまたはファイルを交換するために Transfer Family を使用して、インバウンド転送のために、外部のパートナー所有のソースから Transfer Family サーバーにAS2ファイルを送信する取引相手の詳細を指定します。

この手順では、Transfer Family コンソールを使用してAS2アグリーメント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. で AWS Transfer Family コンソールを開きますhttps://console.aws.amazon.com/transfer/。
2. 左側のナビゲーションペインで、サーバー 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. 受信トレイ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メッセージの送受信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。

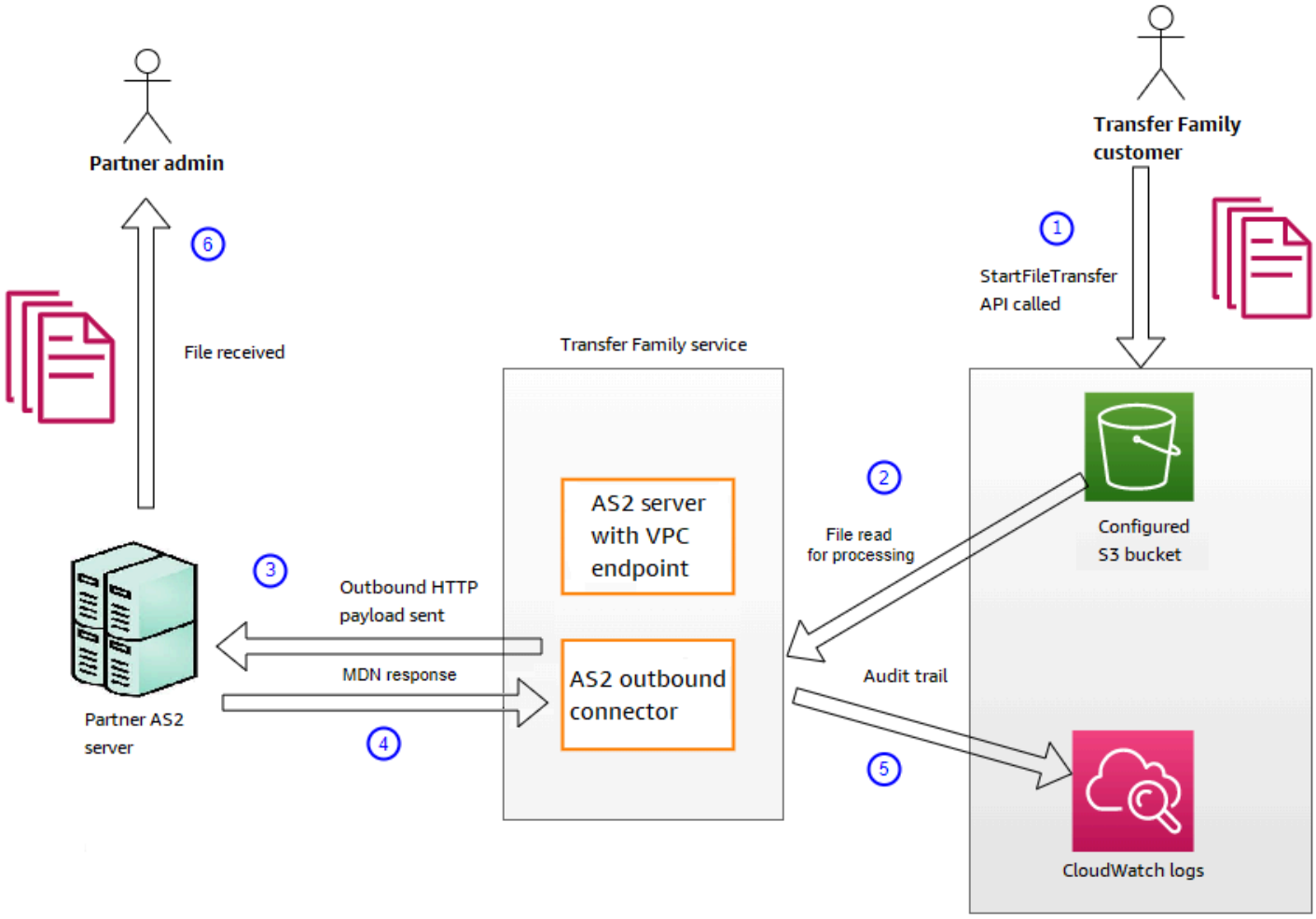
トピック

- AS2 メッセージ送信プロセス
- AS2 メッセージ受信プロセス
- 経由でAS2のメッセージの送受信 HTTPS
- 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
- サンプルJSONファイル

AS2 メッセージ送信プロセス

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、ペイロードをパートナーのAS2サーバーに送信するHTTPPOSTリクエストを実行します。
4. プロセスは、署名付きMDNレスポンスをHTTPレスポンス (同期) とインラインで返しますMDN。
5. ファイルが送信の異なるステージ間を移動すると、プロセスはMDNレスポンス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、復号化および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。



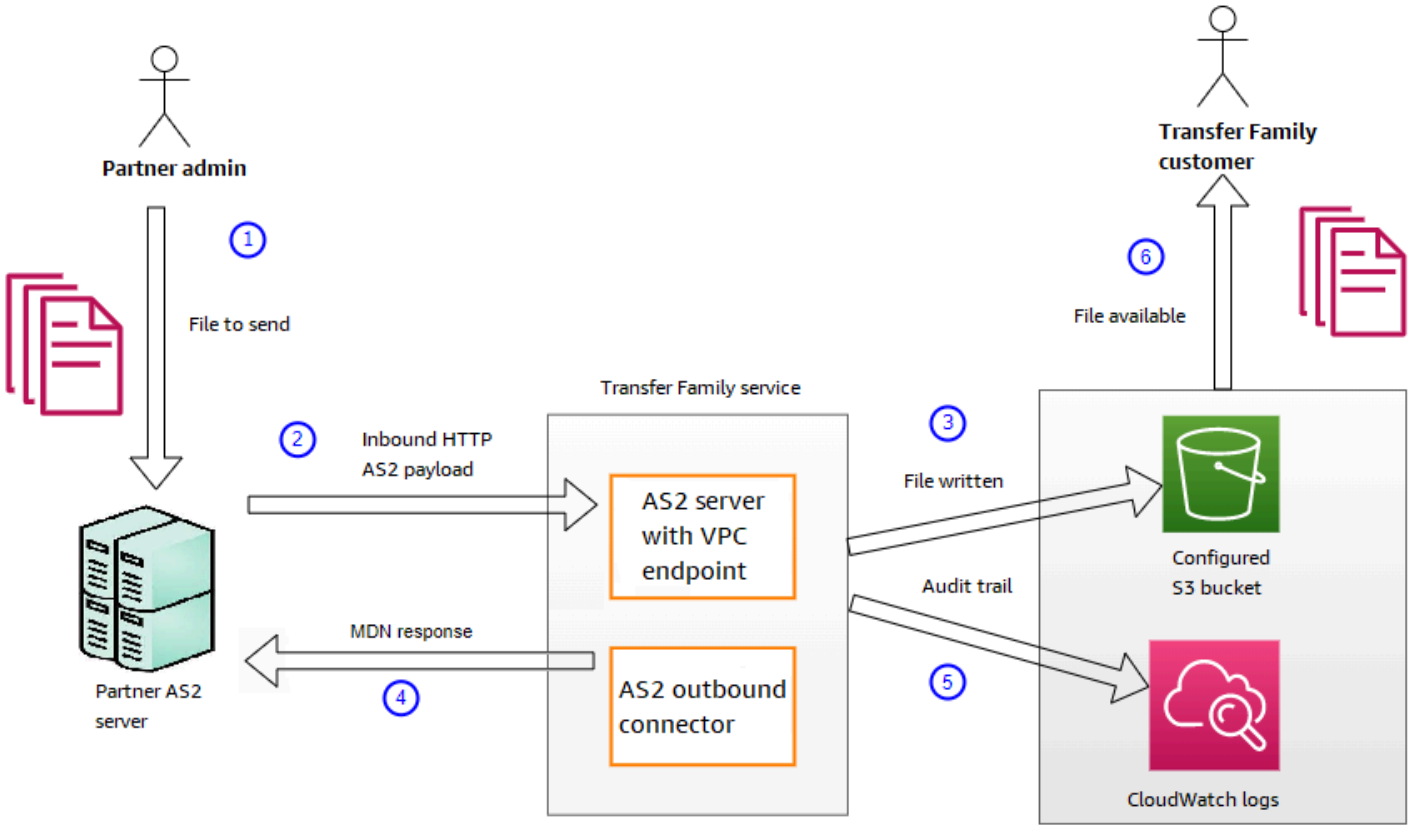
AS2 処理は、一般的なユースケースと既存の AS2対応サーバー実装との統合に焦点を当て、多くの RFC 4130 プロトコルをサポートしています。サポートされている構成の詳細については、AS2 サポートされている設定 を参照してください。

AS2 メッセージ受信プロセス

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. 管理者または自動プロセスは、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コンテンツ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HTTPPOSTリクエストを送信します。

3. Transfer Family は、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ペイAS2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. 署名付きMDNレスポンスは、HTTPレスポンスとインラインで返されるか、別のHTTPPOSTリクエストによって非同期的に送信元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。



経由でAS2のメッセージの送受信 HTTPS

このセクションでは、AS2プロトコルを使用して 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HTTPS。

経由でAS2メッセージを送信する HTTPS

を使用してAS2メッセージを送信するにはHTTPS、次の情報を含むコネクタを作成します。

- にURL、 を指定します。 HTTPS URL

- 暗号化アルゴリズムには、NONE を指定します。
- 「AS2 コネクタの設定」の説明に従って、コネクタの残りの値を指定します。

経由でAS2メッセージを受信する HTTPS

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、Transfer Family サーバーVPCエンドポイントの前にあるロードバランサーTLSで終了するには、選択したポートと証明書を使用します。このアプローチでは、受信AS2メッセージにを使用させることができますHTTPS。

前提条件

- は Transfer Family サーバー AWS リージョン と同じ にあるVPC必要があります。
- 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VPC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 () を選択することもできますBYOIP。

> **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 Balancer を設定する

でインターネット向け Network Load Balancer (NLB) を設定しますVPC。

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. で Amazon Elastic Compute Cloud コンソールを開きますhttps://console.aws.amazon.com/ec2/。

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 address type] (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。

   - でVPC、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VPCで使用するのと同じ で使用できるパブリックサブネットに関連付けられた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アドレスで、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 では、 を選択しますTLS。
   - [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. Secure Listener 設定セクションで、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 を選択して を作成しますNLB。
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 で終了するため、このステップをお勧めしますNLB。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NLBの外部 IP アドレスではなく、 のプライベート IP アドレスです。

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 では、 を選択しますTCP。
   - [Port (ポート)] に「**5080**」と入力します。
   - [IP address type] (IP アドレスタイプ) で、IPv4 を選択します。
   - ではVPC、Transfer Family AS2サーバー用にVPC作成した 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 TCPを選択します。
4. [Next (次へ)] を選択します。
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 の場合、Transfer Family AS2サーバー用にVPC作成した が指定されていることを確認します。
   - IPv4 アドレス には、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。

     サーバーに複数のエンドポイントがある場合は、IPv4アドレスの追加を選択して別の行を追加して別のIPv4アドレスを入力します。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、前の手順に戻って を作成しNLB、指定された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 DNSの名前を使用して、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ACLs) を使用して、クライアント IP アドレスからのサーバーへのアクセスを管理します。ネットワークACL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してい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 サーバーから外部のパートナー所有の宛先に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 IDs、URLs、および Amazon リソースネーム (ARNs) 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、、、ロール、プロファイルURL、メッセージ処理通知 (MDNs) 、タグ、モニタリングメトリクスなど、コネクタのすべてのプロパティを返します。

パートナーが正常にファイルを受け取ったことを確認するには、 JSONおよび MDN ファイルを表示します。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 ログに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 コネクタの詳細を表示するには、「」を参照してくださいAS2 コネクタの詳細を表示する。AS2 コネクタの作成の詳細については、「」を参照してくださいAS2 コネクタの設定。

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 と指定します。
- 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 と指定します。

  JSON ファイルには次のフィールドが含まれます。

  - 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フォルダに保存されます。例えば、/DOC-EXAMPLE-BUCKET/AS2-folder/failed と指定します。
- 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 ファイルが作成され、 として保存されます*original_filename*.*messageId*.*original_extension*.json。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、 として保存されます*original_filename*.*messageId*.*original_extension*.mdn。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、StartFileTransferAPIオペレーションによって返されます (または別のプロセスまたはスクリプトがこのオペレーションを呼び出す場合) 。

- 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、StartFileTransferAPIオペレーションまたはstart-file-transfer AWS CLI コマンドで指定したパスです。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MDN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 (転送が失敗した場合) に保存されます。
- JSON ファイルが作成され、 として保存されますoriginal_filename.transferId.messageId.original_extension.json。
- MDN ファイルが作成され、 として保存されますoriginal_filename.transferId.messageId.original_extension.mdn。
- 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

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 と FAILEDの状態は、処理の最終ステップを表し、 JSON ファイルに表示されます。

コード	説明	処理が完了しましたか？
PROCESSING	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。	不可
MDN_TRANSMIT	メッセージ処理はMDNレスポンスを送信しています。	不可
MDN_RECEIVE	メッセージ処理がMDN応答を受信しています。	不可
COMPLETED	メッセージ処理が正常に完了しました。この状態には、MDNが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MDNの検証のために送信される場合が含まれます。	可能
FAILED	メッセージ処理に失敗しました。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。	可能

サンプルJSONファイル

このセクションでは、インバウンド転送とアウトバウンド転送の両方のサンプルJSONファイルを一覧表示します。これには、転送が成功し、失敗した転送のサンプルファイルが含まれます。

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 ロギング記録 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。 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
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ログ記録用のロールを引き受けられない場合にも発生します。 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。

Code	エラー	説明と解決策
		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する」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。

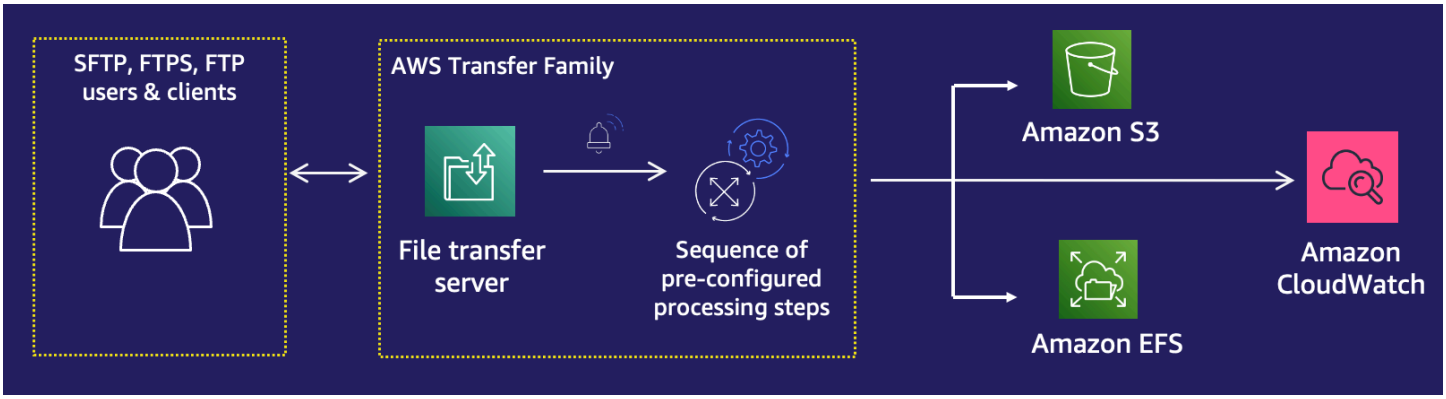
Code	エラー	説明と解決策
SIGNING_FAILED_NO_SIGNING_KEY_FOUND	プロファイル: *local-profile-ID* の証明書が見つかりません。	アウトバウンドメッセージを署名しようとしていますが、ローカルプロファイルの署名用証明書が見つかりません。 解決オプション: • ローカルプロファイルには、署名用の証明書とプライベートキーが添付されていることを確認してください。 • 署名用証明書が現在有効であることを確認してください。
UNABLE_RESOLVE_HOST_TO_IP_ADDRESS	ホスト名を IP アドレスに変換できません。	Transfer 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。マネージドワークフローでは、、SFTP、FTPSまたは 経由でファイルが転送された後にワークフローを開始できますFTP。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コスト効率良く満たすことができます。さらに、監査と可視性のメリット end-to-end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。(このユースケースの詳細については、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」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、「IAM ワークフローの 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。
- 参加できるワークショップを開催しています。このワークショップでは、ファイル転送ソリューションを構築できます。このソリューションは、 マネージド SFTP/FTPS エンドポイント、Amazon Cognito と DynamoDB AWS Transfer Family をユーザー管理に活用します。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。
- IAM ワークフローの 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、公称ステップが失敗したり、検証エラーが発生した場合に 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. で AWS Transfer Family コンソールを開きますhttps://console.aws.amazon.com/transfer/。
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. で AWS Transfer Family コンソールを開きますhttps://console.aws.amazon.com/transfer/。
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。

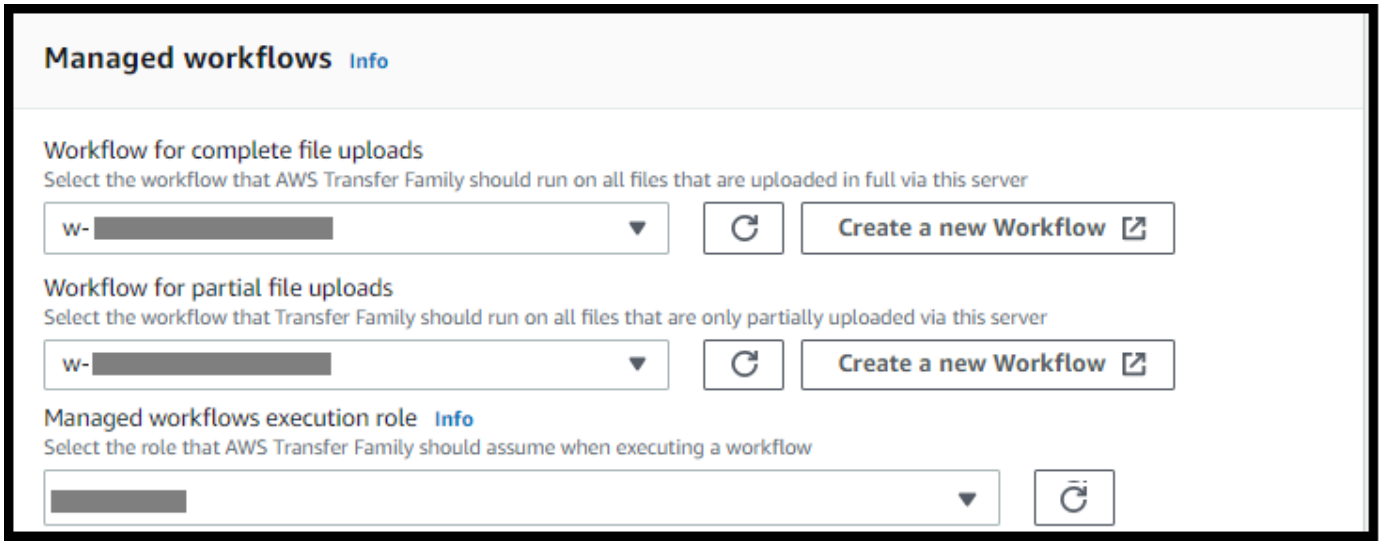
> **Note**
>
> デフォルトでは、サーバーに関連付けられたワークフローはありません。[Additional details] (その他の詳細) セクションを使用して、選択したサーバーにワークフローを関連付けます。

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、「IAM ワークフローの ポリシー」を参照してください。[Save] 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 分間の待機期間後に再度ログインして変更を確認する必要があります。

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

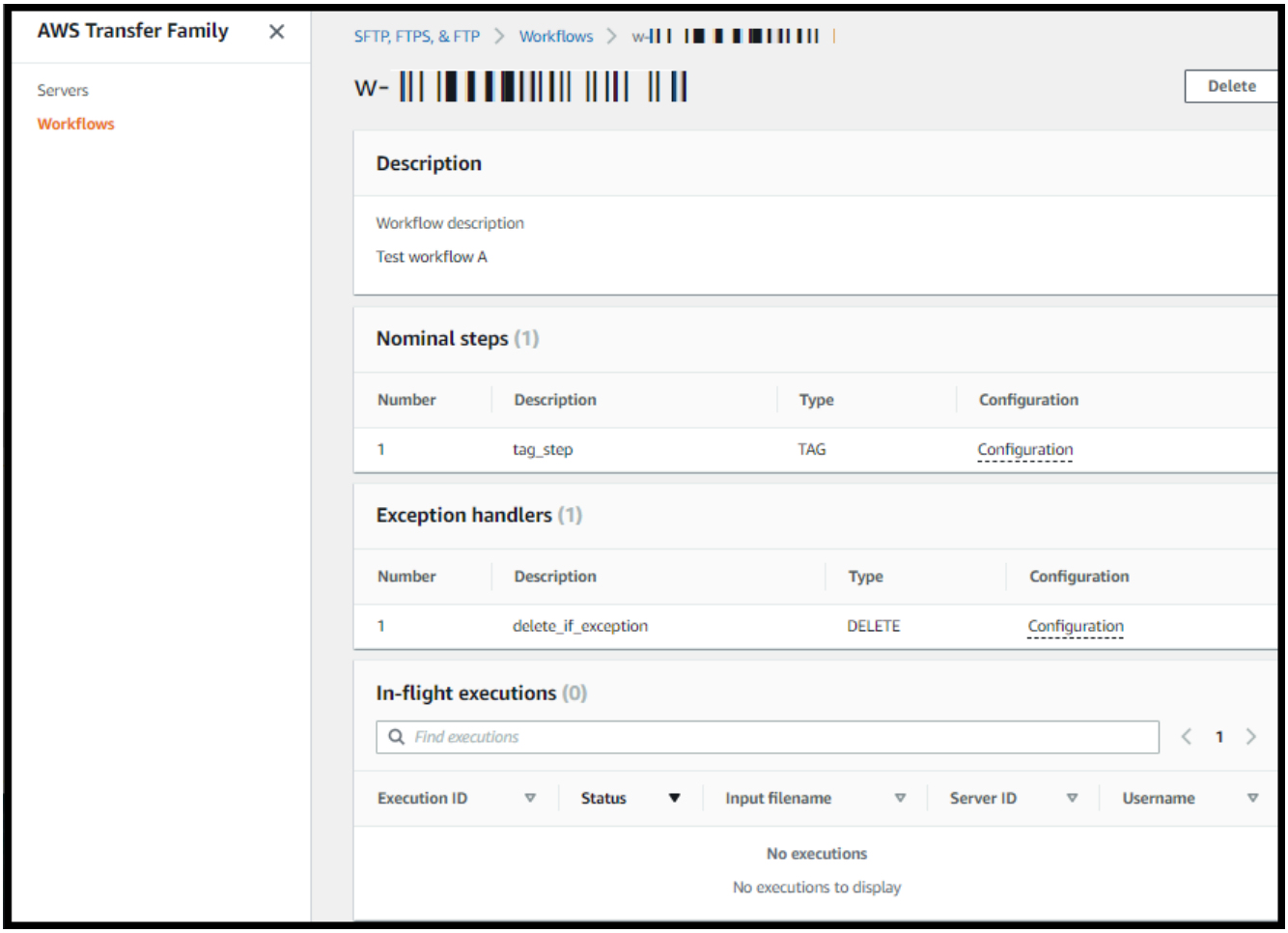
以前に作成したワークフローの詳細や、ワークフローの実行状況を見ることができます。これらの詳細を表示するには、 コンソールまたは AWS Command Line Interface () を使用しますAWS CLI。

Console

ワークフロー詳細の表示

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

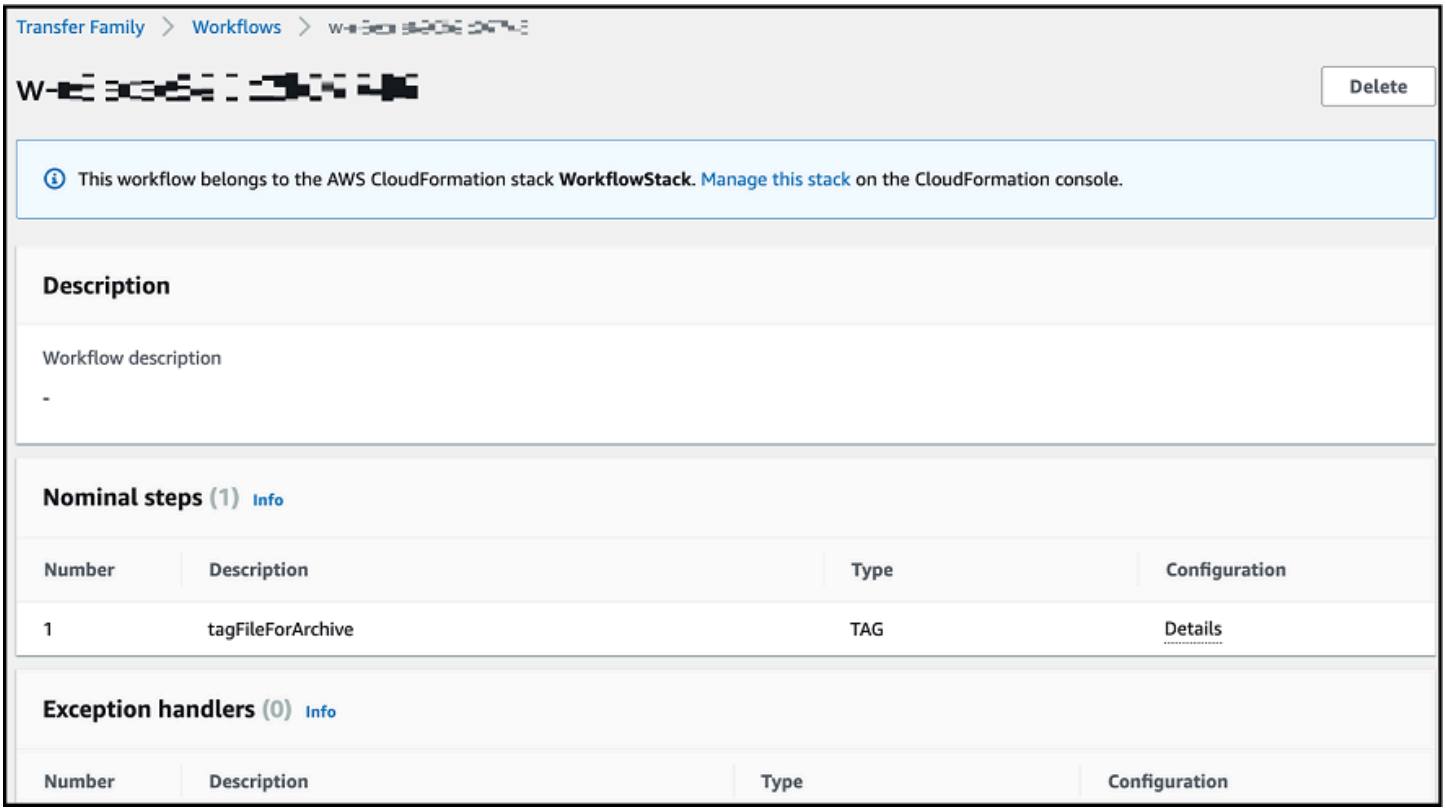


CLI

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 describe-workflow CLI 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

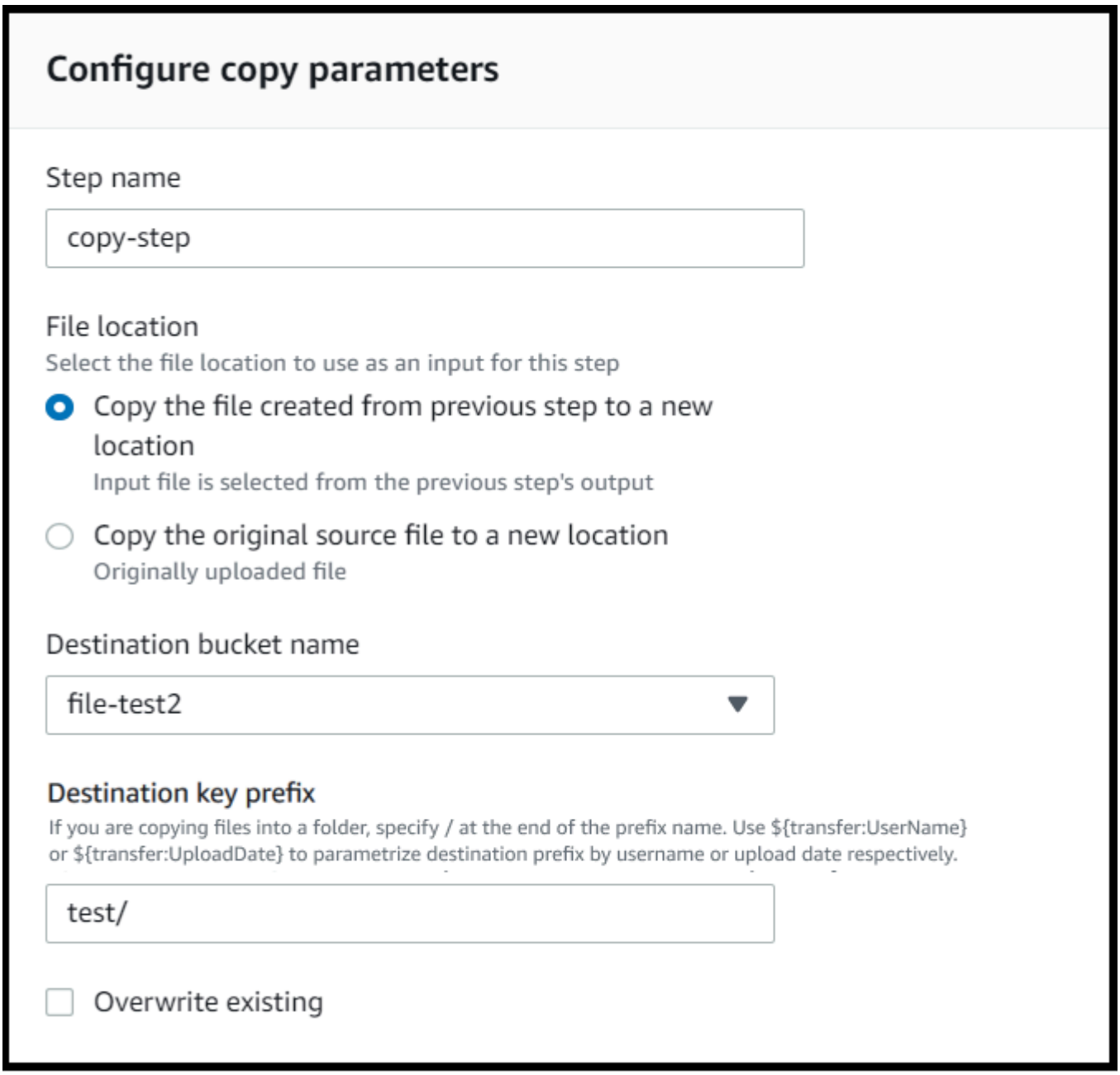
- ファイルをコピーする
- ファイルを復号化します。
- タグファイル
- ファイルを削除する
- ワークフローの名前付き変数
- タグ付けと削除のワークフローの例

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。

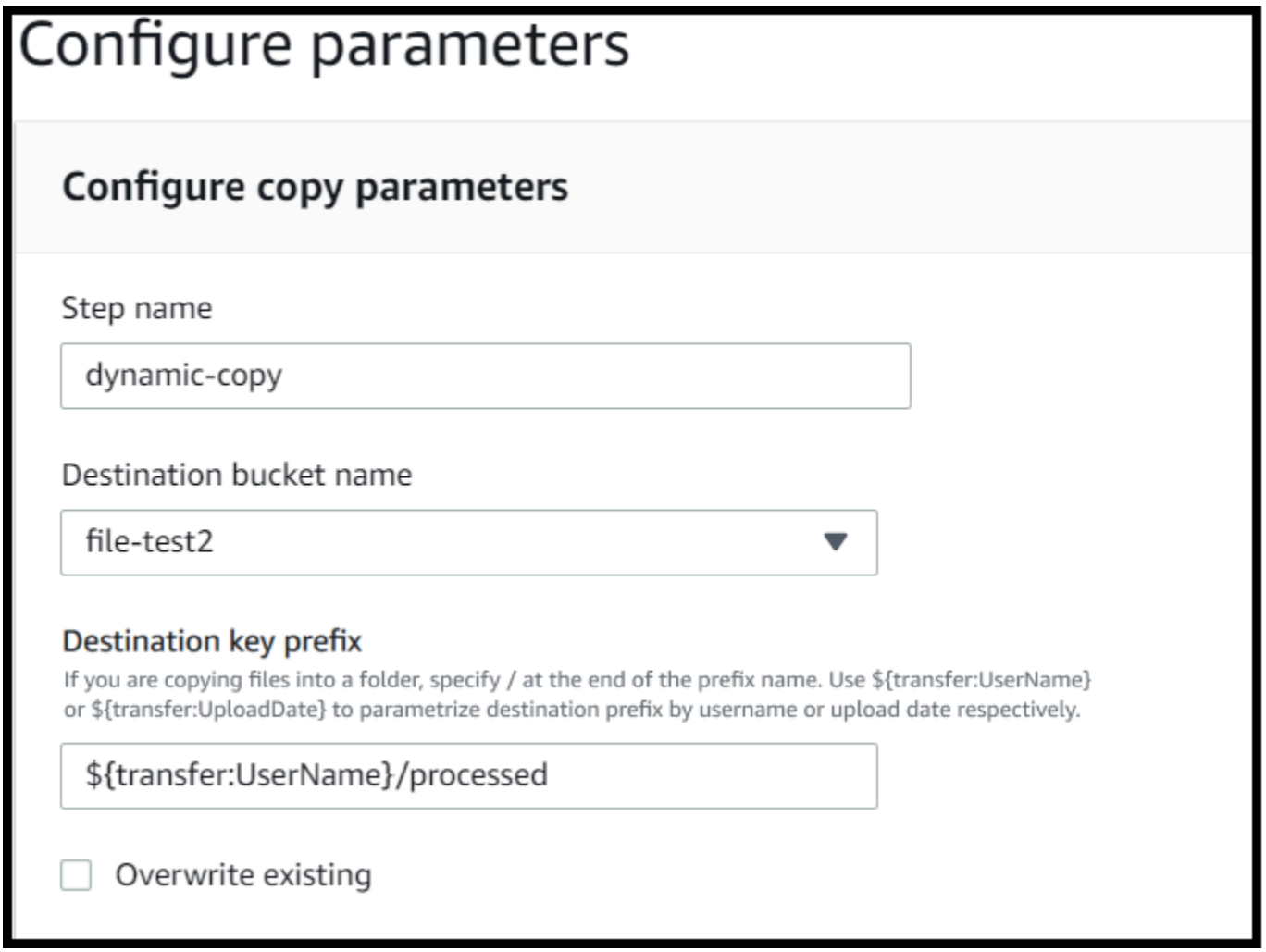
たとえば、キー値が `test/` であれば、アップロードされたファイルは `test` フォルダにコピーされます。キーの値が `test/today` で、「Overwrite existing」が選択されている場合、アップロードしたファイルはすべて `test` フォルダー内の `today` という名前のファイルにコピーされ、後続のファイルはそれぞれ前のファイルが上書きされます。

> **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。

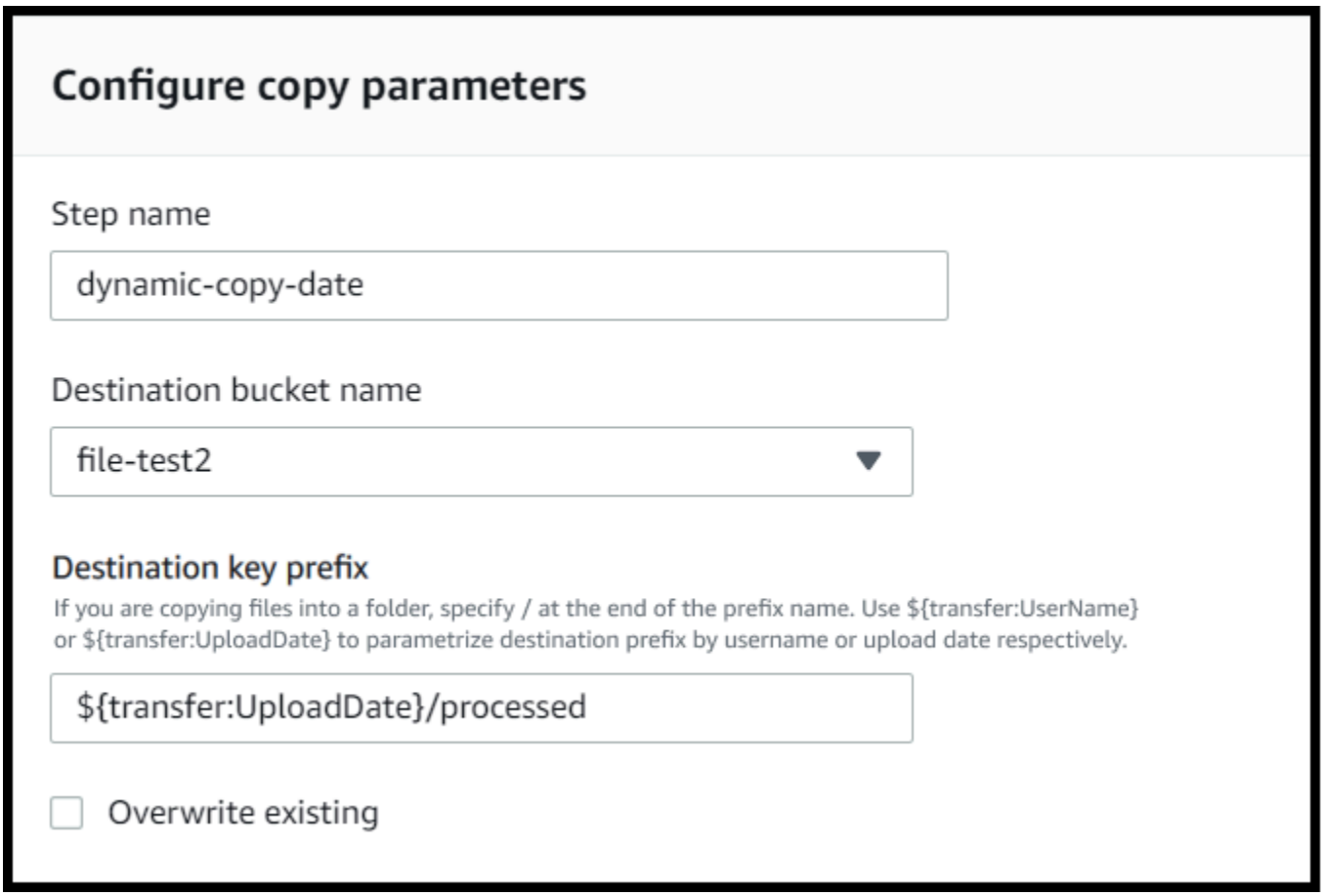
ファイルのコピーステップでは名前付き変数を使用します。

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。

IAM コピーステップのアクセス許可

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 つのアクセス許可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、 PGPおよび を使用してファイルの暗号化と復号、ファイルの暗号化と復号 AWS Transfer Familyを行う方法について説明します。

ワークフローでPGP復号を使用する

Transfer Family には、Pretty Good Privacy (PGP) 復号のサポートが組み込まれています。SFTP、、FTPSまたは FTP Amazon Simple Storage Service (Amazon S3) または Amazon PGP Amazon Elastic File System) にアップロードされたファイルでは、復号を使用できますEFS。

PGP 復号を使用するには、ファイルの復号に使用される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。その後、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、対応する暗号化キーを使用してファイルをPGP暗号化できます。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。

ワークフローでPGP復号を使用するには

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、「PGP キーの管理」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、 をPGPキーペアの作成に使用した E メールアドレス*username@example.com*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

復号ステップは、ワークフローEFSの一部として Amazon S3 または Amazon にアップロードされた暗号化されたファイルを復号します。復号化の設定の詳細については、ワークフローでPGP復号を使用する 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 を使用して、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 復号化パラメータの設定セクションの値を入力します。

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。

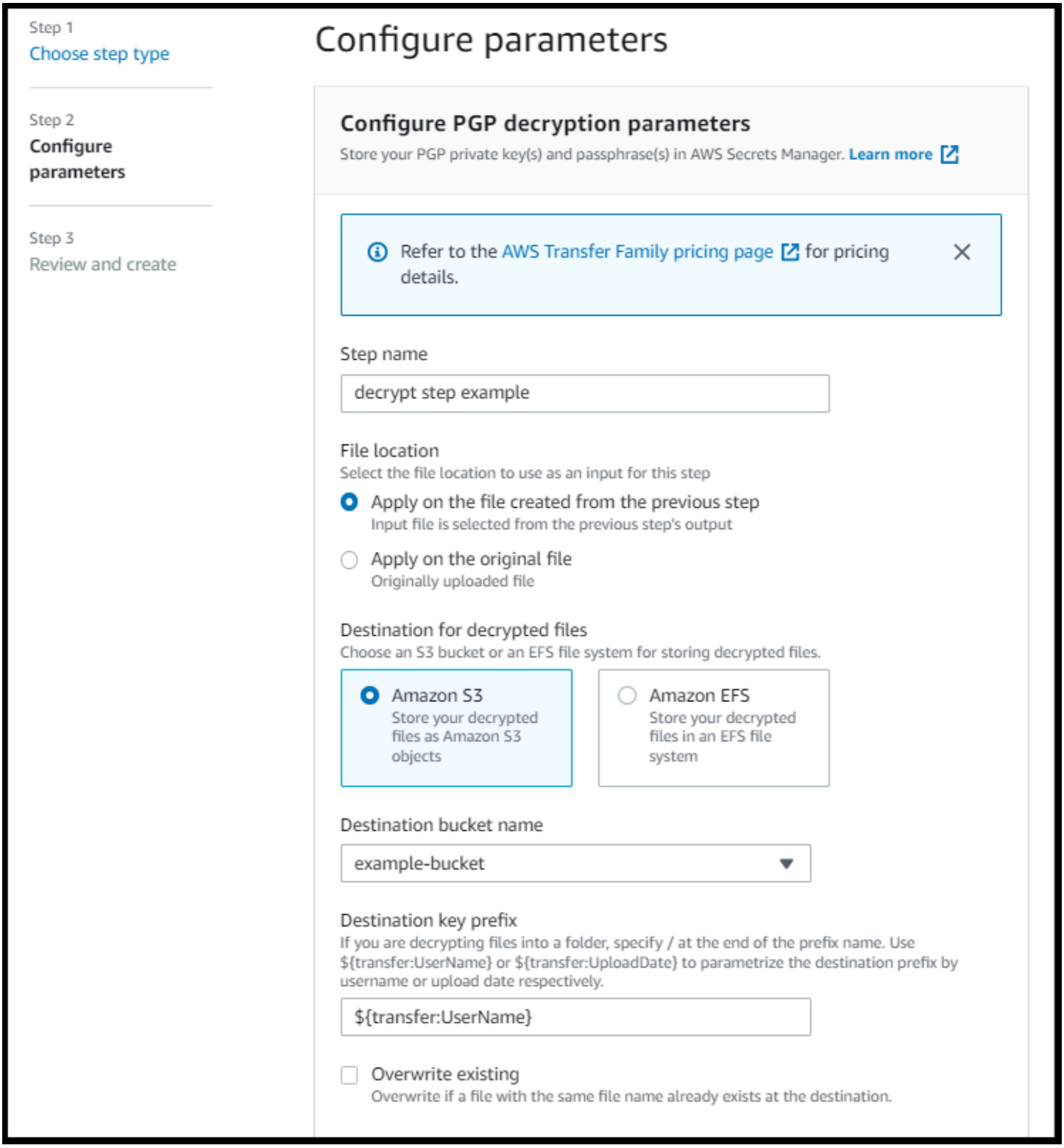
- 復号化されたファイルの宛先 — 復号されたEFSファイルの宛先として Amazon S3 バケットまたは Amazon ファイルシステムを選択します。
  - Amazon S3 を選択した場合、宛先バケット名と宛先キープレフィックスを指定する必要があります。宛先キープレフィックスをユーザー名でパラメータ化するには、「宛先キープレフィックス」に**`${transfer:UserName}`**と入力します。同様に、アップロード日ごとに宛先キープレフィックスをパラメータ化するには、「宛先キープレフィックス」に**`${Transfer:UploadDate}`**と入力します。
  - Amazon を選択した場合はEFS、送信先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.

IAM 復号ステップのアクセス許可

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。

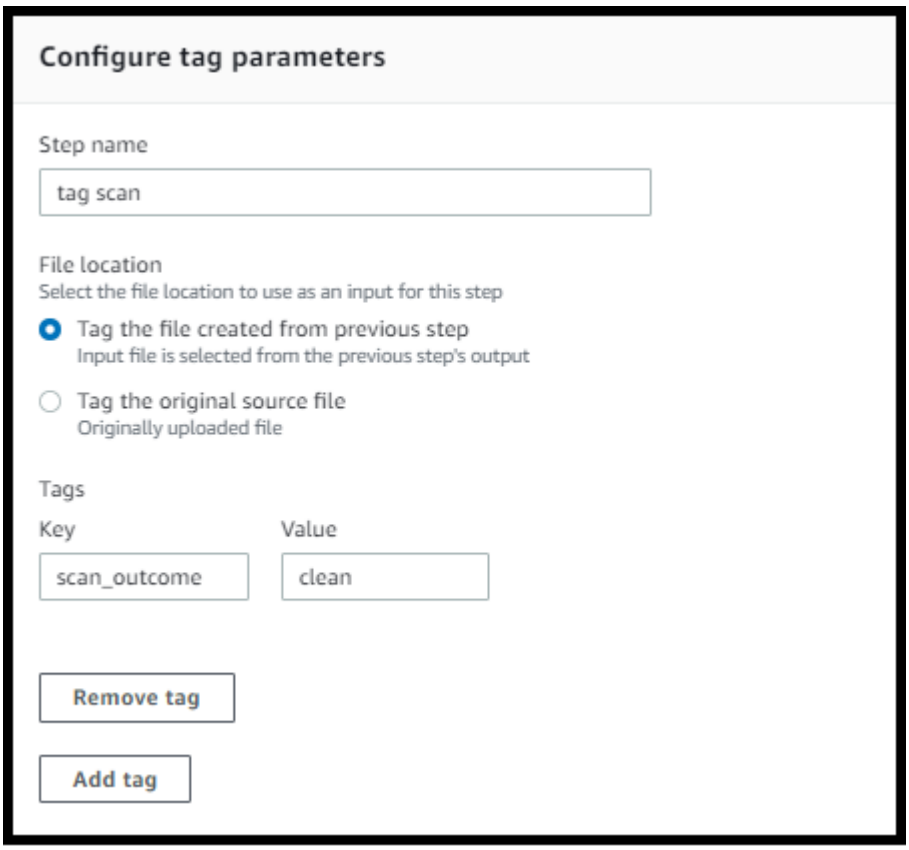
Amazon S3 ファイルにタグがある場合は、IAMポリシーに 1 つまたは 2 つのアクセス許可を追加する必要があります。

- 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。



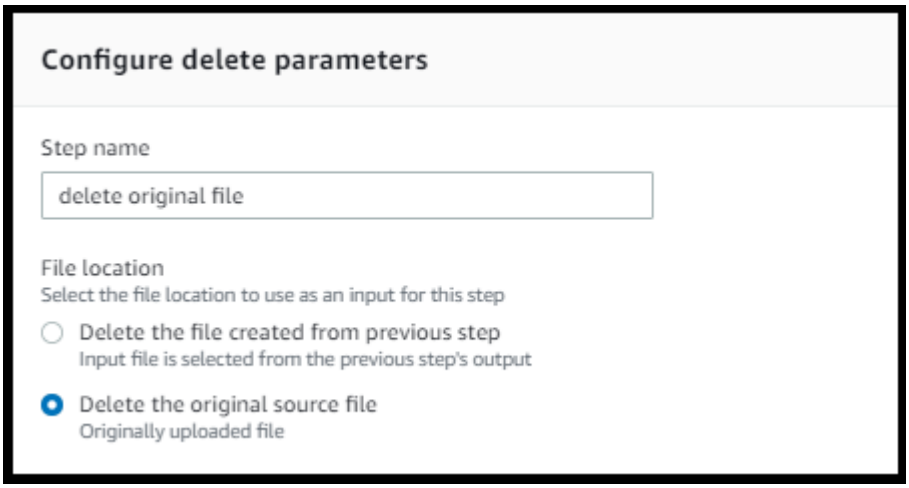
タグステップを成功させるには、ワークフローの実行ロールに次の権限が含まれていることを確認してください。

```
{
            "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 つのアクセス許可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

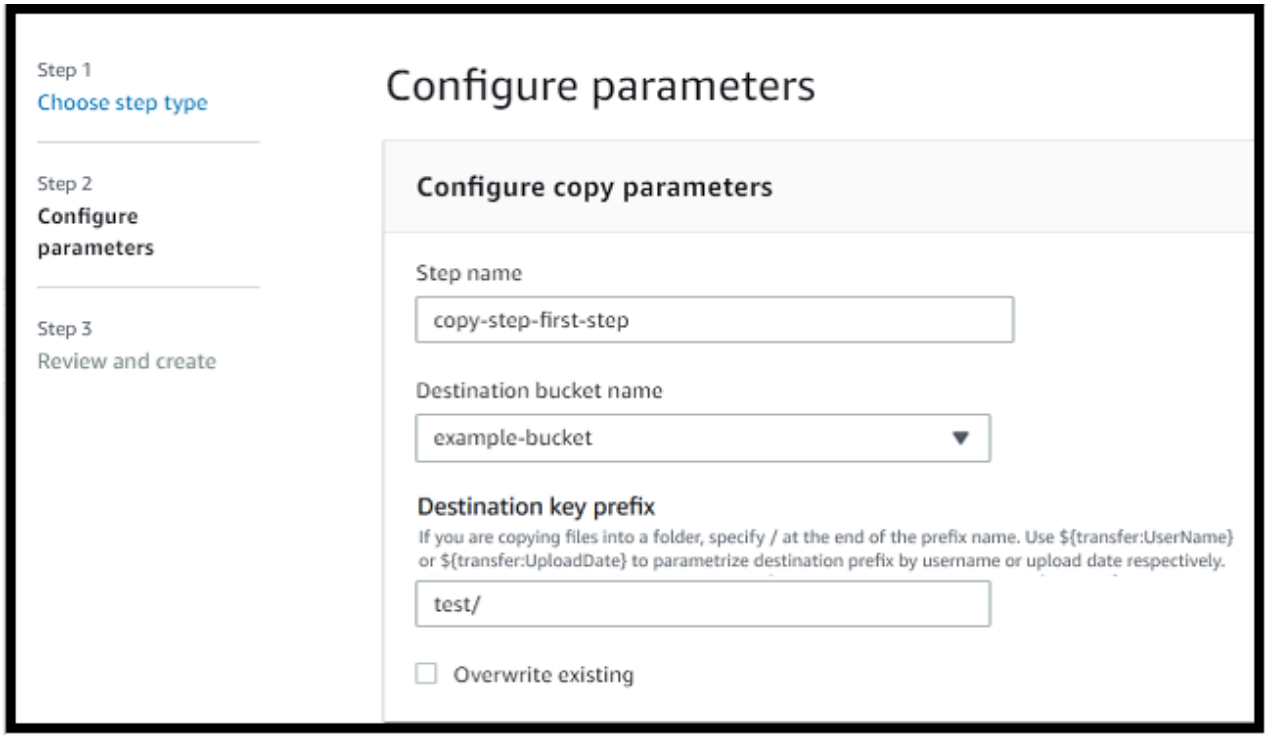
次の例は、データ分析プラットフォームなどのダウンストリームアプリケーションで処理する必要がある受信ファイルにタグを付けるワークフローを示しています。受信ファイルにタグを付けた後、ワークフローはストレージコストを節約するために最初にアップロードされたファイルを削除します。

Console

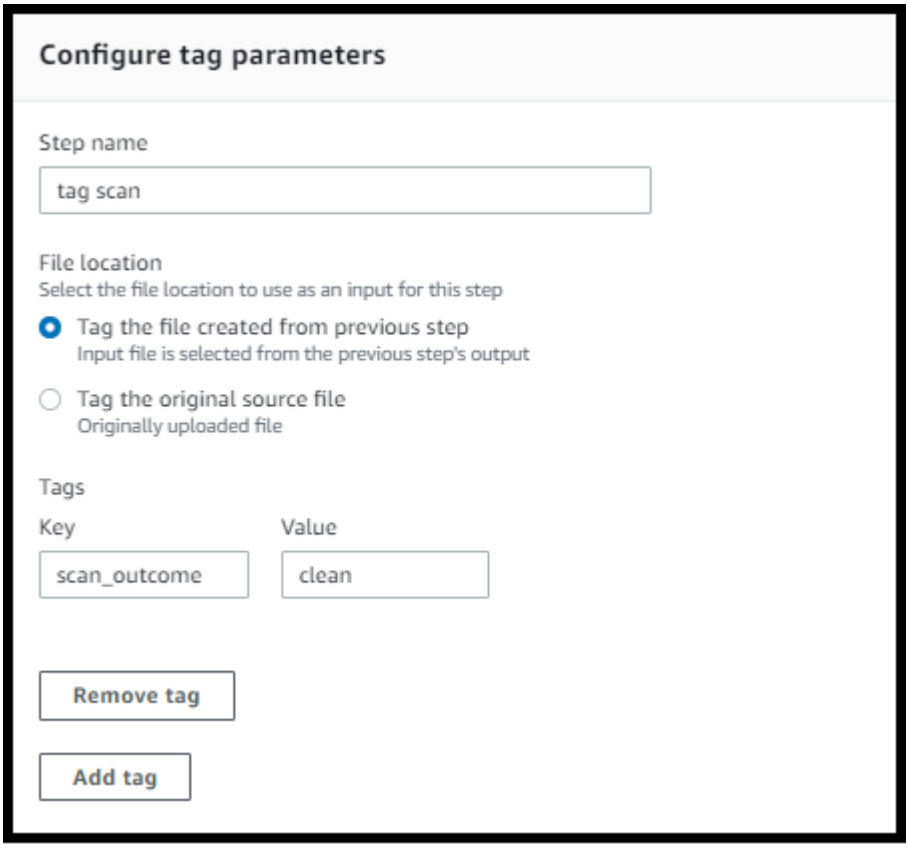
例:タグ付けして移動するワークフロー

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



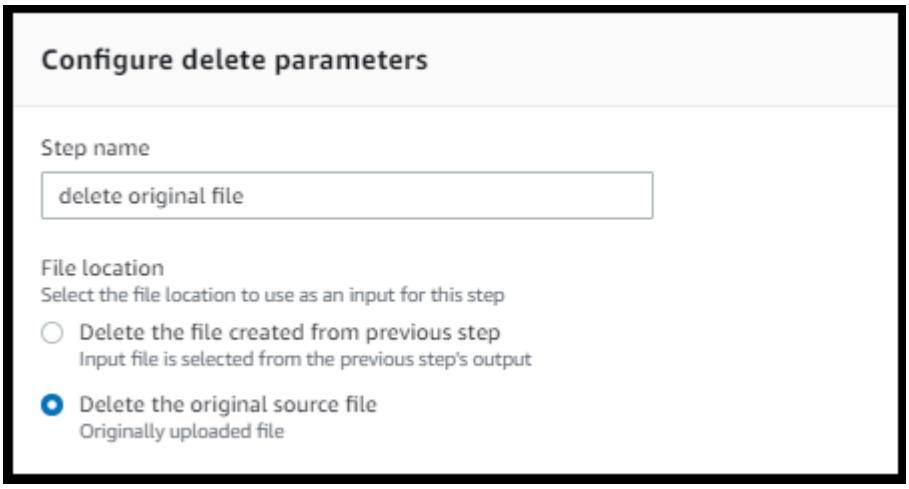
f. [Next] (次へ) を選択してからステップの詳細を見直します。

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
```

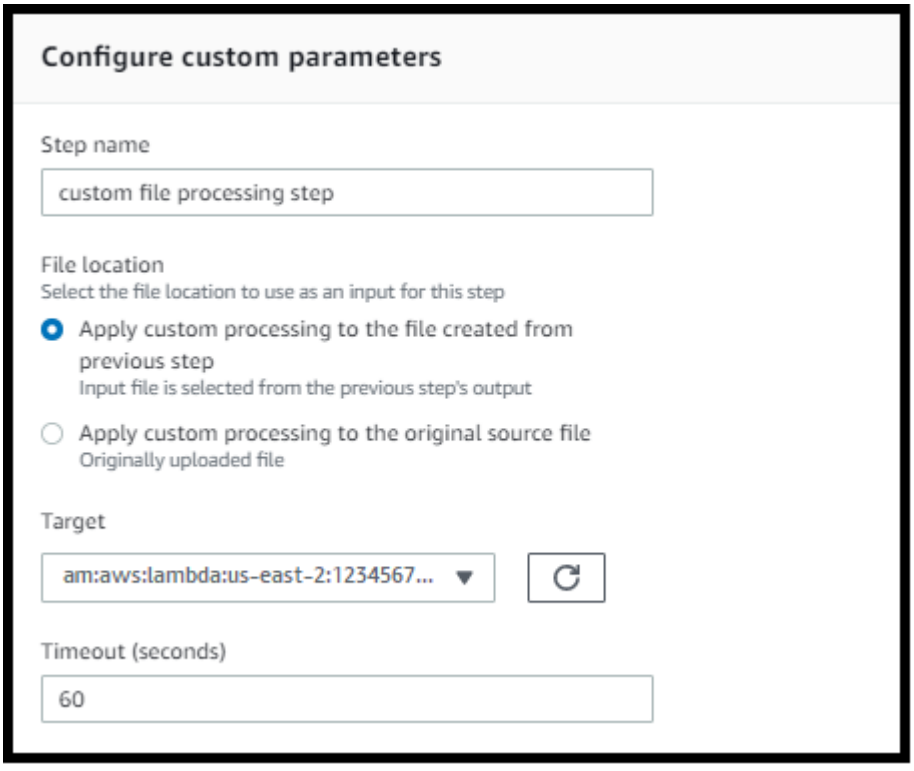
```
  --workflow-details '{"OnUpload":[{ "WorkflowId": "workflow-
ID","ExecutionRole": "execution-role-ARN"}]}'
```

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、SendWorkflowStepStateAPI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。SendWorkflowStepState API オペレーションのステータスは、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公称ステップを呼び出します。

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ときは、次のパラメータ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、Managed Workflows が導入された後に発行された のバージョン AWS SDK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、カスタムステップが のエンティティタグ (ETag) を変更`example-file.txt`して、元のファイルと一致しないために発生します。

> **Note**
>
> Amazon はエンティティタグを使用してファイルを識別しないEFSため、Amazon を使用している場合EFS、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 つの例では、ドメインが Amazon を使用する Transfer Family サーバーを使用していますEFS。

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、 SendWorkflowStepStateAPIオペレーションを呼び出して、 SUCCESSまたは のいずれかのステータスをステップのワークフローに返します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てアクションを実行するように 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)
    }
```

IAM カスタムステップのアクセス許可

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"
            ]
        }
```

IAM ワークフローの 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 つのアクセス許可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ロールの作成の詳細については、「」を参照してくださいthe section called “IAM ロールとポリシーを作成する”。
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、 AWS は、ワークフローに対してリソースへのアクセスを可能な限り制限することをお勧めします。

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に対応するための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": "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"
            ]
```

```
        }
    ]
}
```

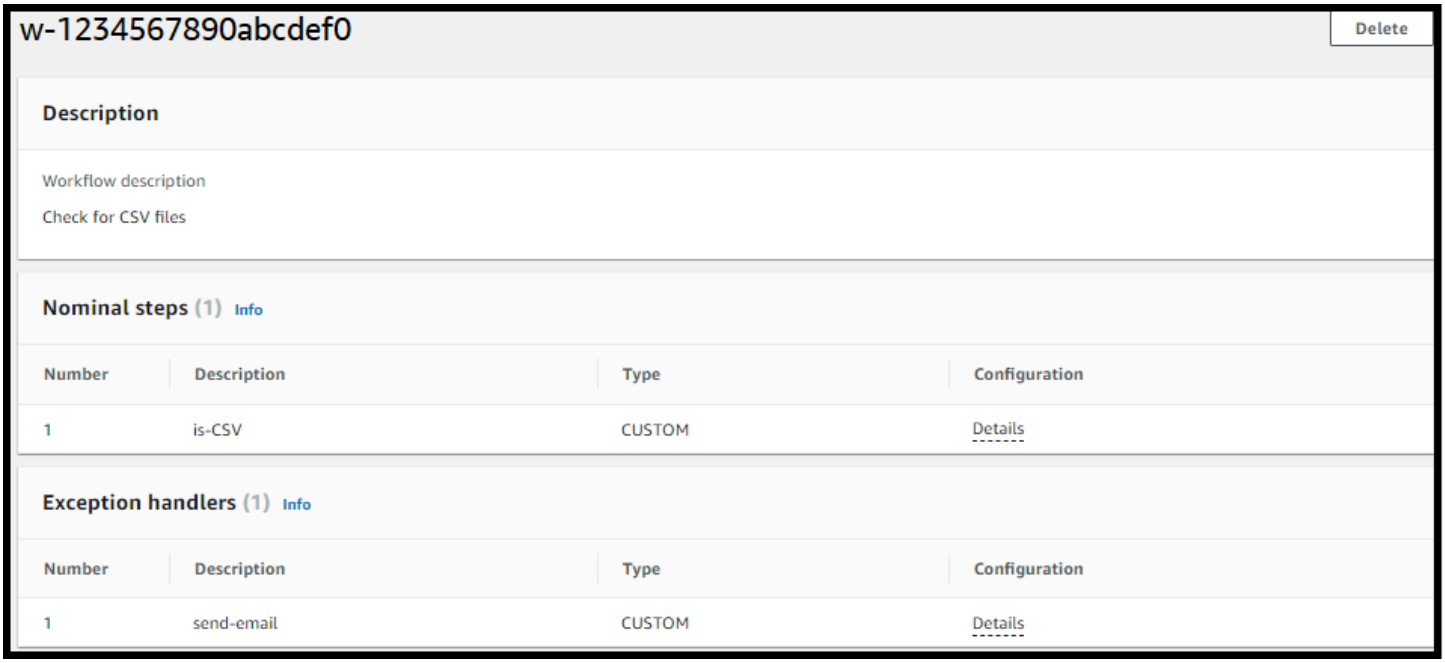
ワークフローの例外処理

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形式であるかどうかをチェックする 1 つの名目上のステップ
- アップロードされたファイルがCSVフォーマットされておらず、公称ステップが失敗した場合に E メールを送信する例外処理ステップ

例外処理ステップを開始するには、公称ステップの 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、 AWS クラウド で実行する AWS リソースとアプリケーションをリアルタイムで 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を CloudWatch 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に関する統合された監査とログ記録を提供します。

ワークフローの Amazon CloudWatch ログを表示する

1. で Amazon CloudWatch コンソールを開きますhttps://console.aws.amazon.com/cloudwatch/。
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) ファイル転送プロトコル (SFTP) セッションごとに 1 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、ワークフローのいくつか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、「」を参照してください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. AWS CloudFormation コンソールを https://console.aws.amazon.com/cloudformation で開きます。
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。

   - 置換 *workflow-execution-role-arn* を実際のワークフロー実行ロールARNの で使用します。例えば、arn:aws:transfer:us-east-2:111122223333:workflow/w-1234567890abcdef0
   - Lambda 関数ARNの を arn:aws:lambda:*region*:*account-id*:function:*function-name*に置き換えます。例えば、arn:aws:lambda:us-east-2:123456789012:function:example-lambda-idp と指定します。

4. 「 ユーザーガイド」の AWS CloudFormation 「スタックテンプレートの選択」の「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。 AWS CloudFormation

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

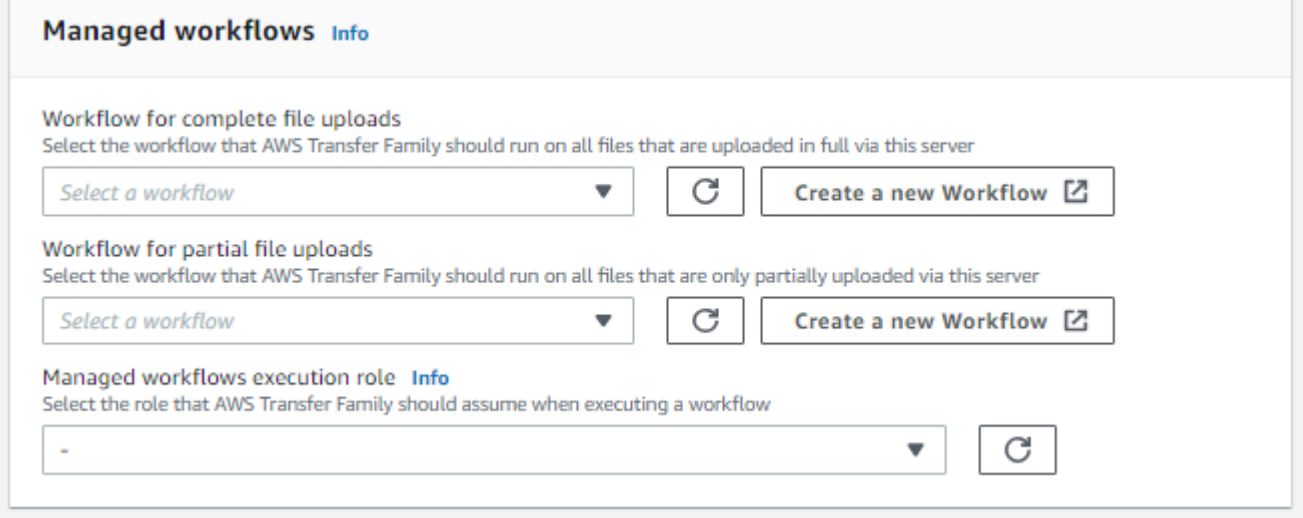
1. で AWS Transfer Family コンソールを開きますhttps://console.aws.amazon.com/transfer/。
2. 左側のナビゲーションペインで [Servers] (サーバー) を選択します。
3. 「サーバー ID」欄でサーバーの識別子を選択します。

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 `UpdateServer`の場合API) 呼び出しを使用し、 `OnUpload` および `OnPartialUpload`パラメータに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。例えば、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 (例えば、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 サーバーごとにユーザーごとに最大 3 つのPGPプライベートキーを保存できます。
- 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 分間の待機期間後に再度ログインして変更を確認する必要があります。

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が有効なサーバーのホストキーを変更する方法に関する情報を確認できます。

トピック

- サーバーのリストを表示する
- サーバーを削除する
- SFTP、FTPS、および FTPサーバーの詳細を表示する
- AS2 サーバーの詳細を表示する
- サーバーの詳細を編集する
- SFTP対応サーバーのホストキーを管理する
- コンソールで使用状況をモニターする

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

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

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

を使用してサーバーを削除するには CLI

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。

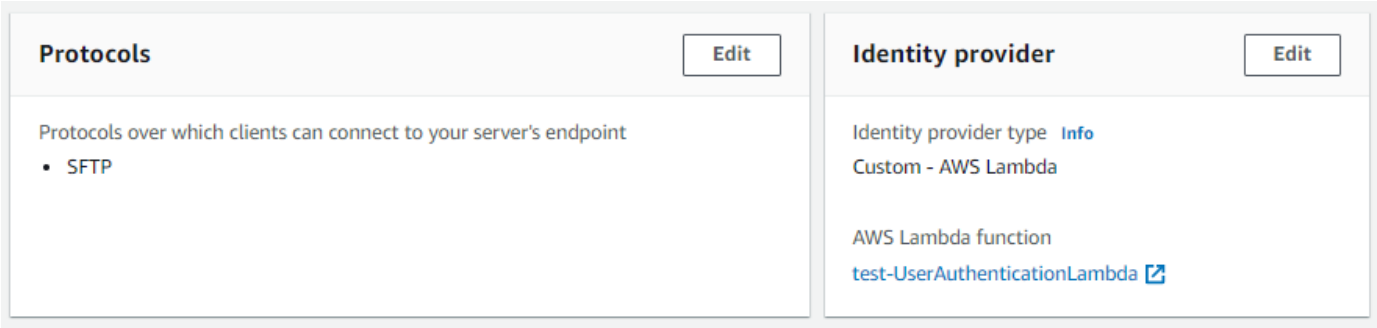
SFTP、FTPS、および FTPサーバーの詳細を表示する

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

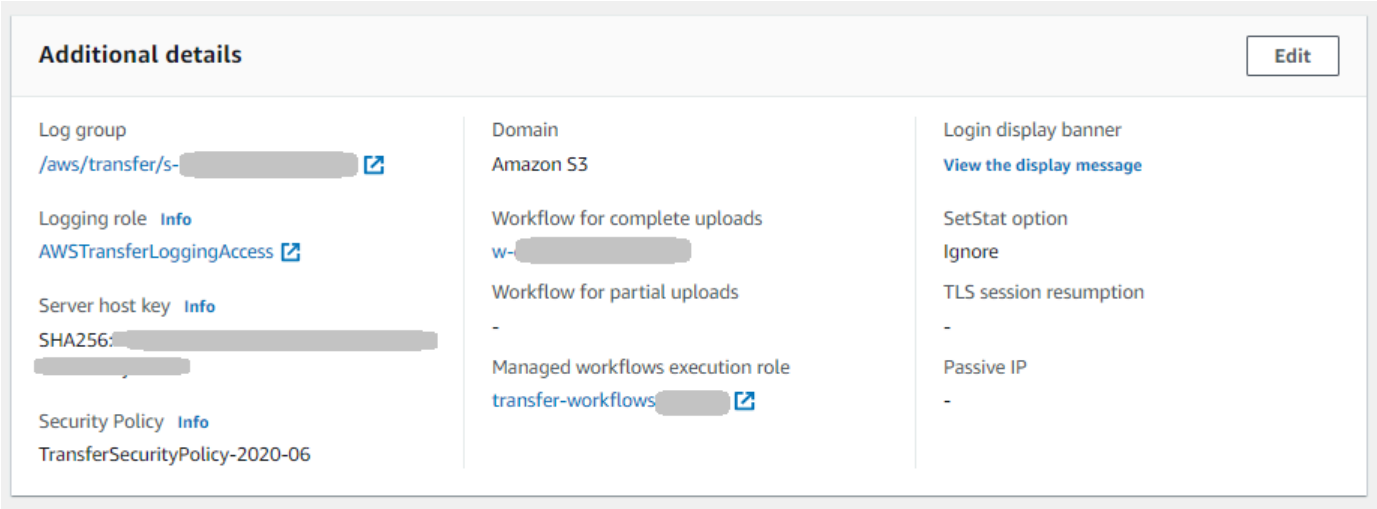
1. で AWS Transfer Family コンソールを開きますhttps://console.aws.amazon.com/transfer/。
2. ナビゲーションペインで、[Servers] (サーバー) を選択します。
3. [Server ID] (サーバー ID) 列で ID を選択すると、次のように [Server Configuration] (サーバーの構成) ページが開きます。

   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若干異なります。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。



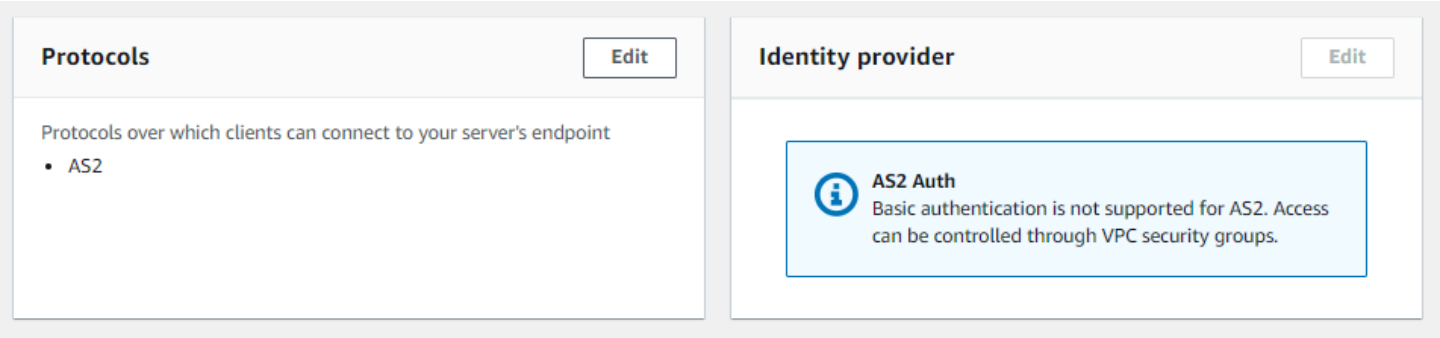
> **Note**
>
> サーバーホストキーの [説明] と [インポートされた日付] の値は、2022 年 9 月時点で新しくなりました。これらの値は、複数のホストキー特徴量をサポートするために導入さ

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。

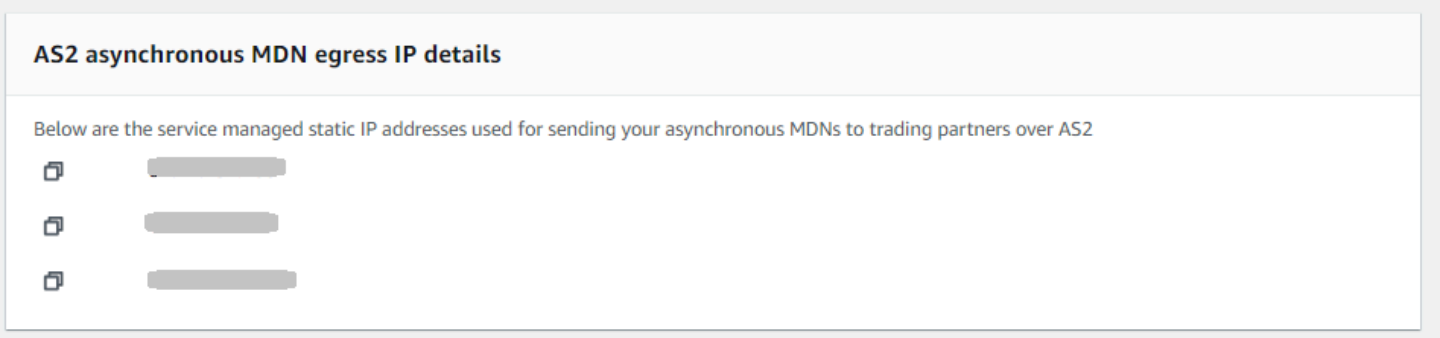


AS2 サーバーの詳細を表示する

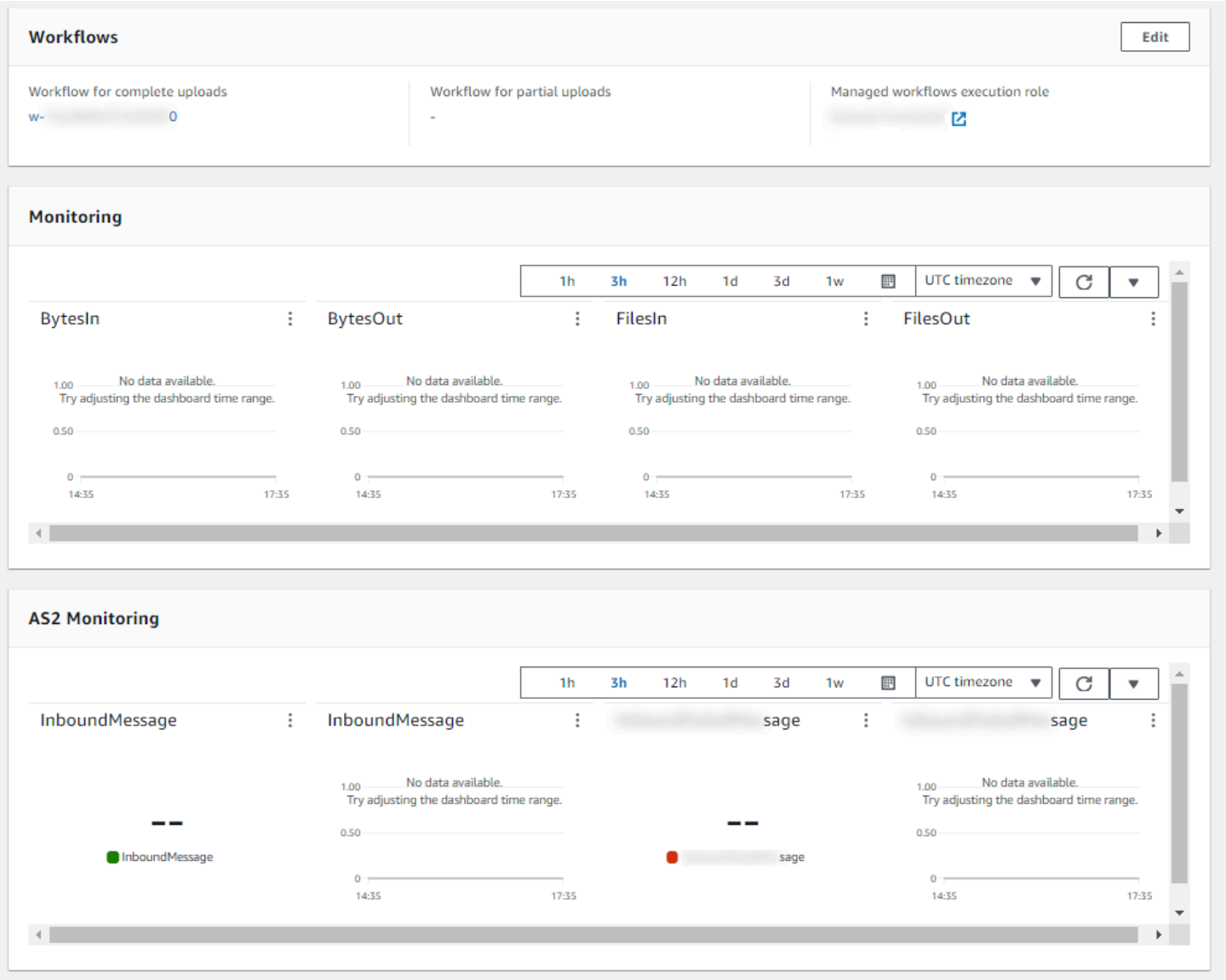
個々の AWS Transfer Family サーバーの詳細とプロパティのリストを確認できます。サーバープロパティには、プロトコル、ステータスなどが含まれます。AS2 サーバーの場合、AS2非同期MDN出力 IP アドレスを表示することもできます。



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、経由で取引先MDNsに非同期を送信しますAS2。



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の詳細と、モニタリングとタグ付けに関する情報が含まれています。



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

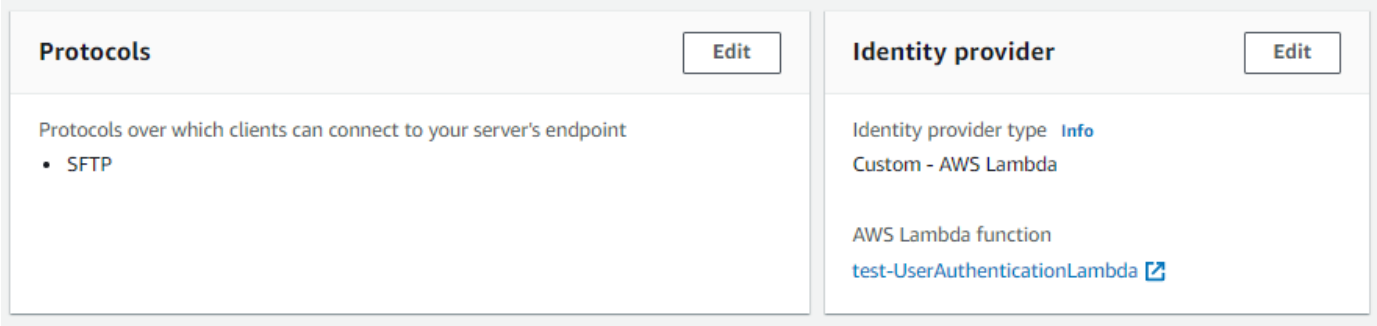
   このページで [Edit] (編集) を選択して、サーバーのプロパティを変更できます。

   - 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。

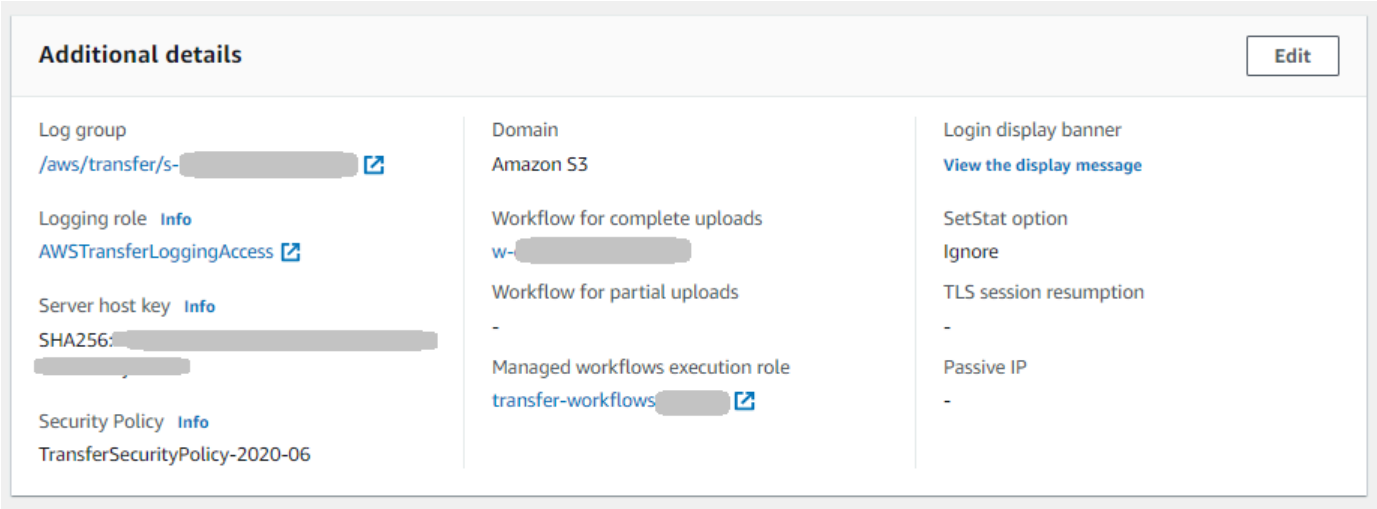
     > **Note**
     >
     > サーバーがカスタム ID プロバイダーを使用している場合は、一部のプロパティを編集できます。詳細については、「カスタム ID プロバイダーパラメーターを編集」を参照してください。

   - 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 をプロトコルAS2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する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する」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セッションのコントロールとデータ接続の間でネゴシエートされたシークレット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。
  - パッシブ IP : FTPと 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 IPv4、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、単一の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、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。

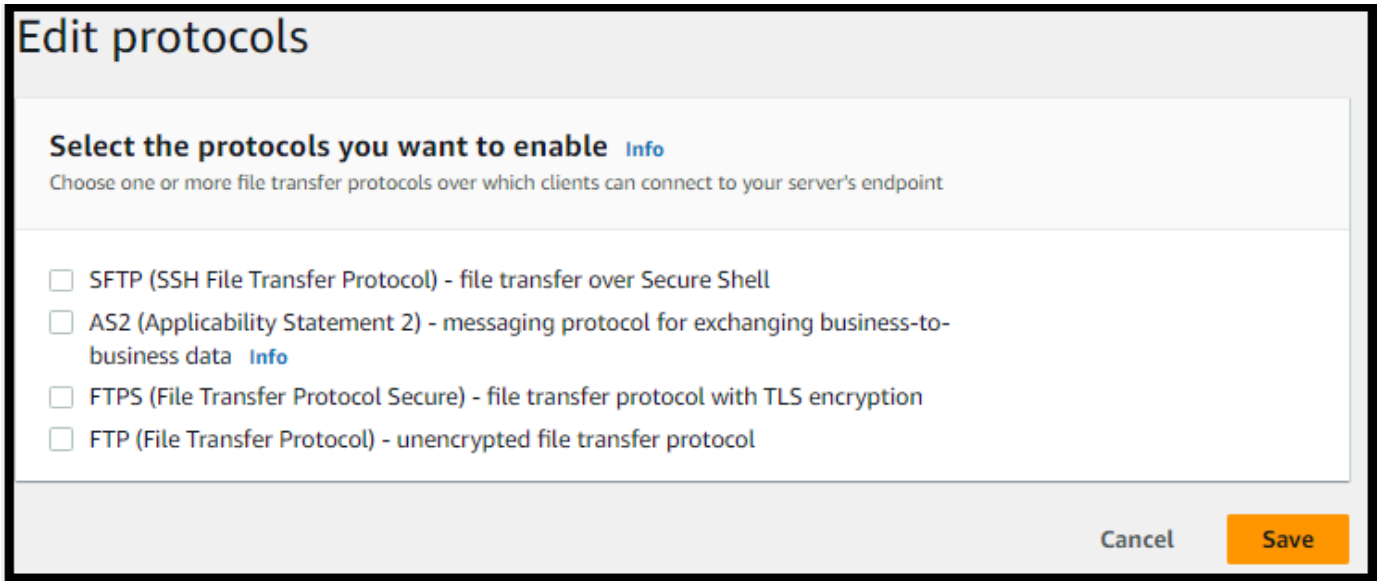
   - Secure Shell (SSH) File Transfer Protocol (SFTP) – ファイル転送 SSH

     の詳細についてはSFTP、「」を参照してくださいSFTP対応サーバーを作成する。
   - ファイル転送プロトコルセキュア (FTPS) – TLS暗号化によるファイル転送

     の詳細についてはFTP、「」を参照してくださいFTPS対応サーバーを作成する。
   - ファイル転送プロトコル (FTP) – 暗号化されていないファイル転送

     の詳細についてはFTPS、「」を参照してくださいFTP対応サーバーを作成する。

> **Note**
>
> に対してのみ既存のサーバーを有効にしていてSFTP、 FTPS と を追加する場合は FTP、 FTPS と 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あることを確認する必要がありますFTP。



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 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FTPS。

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。

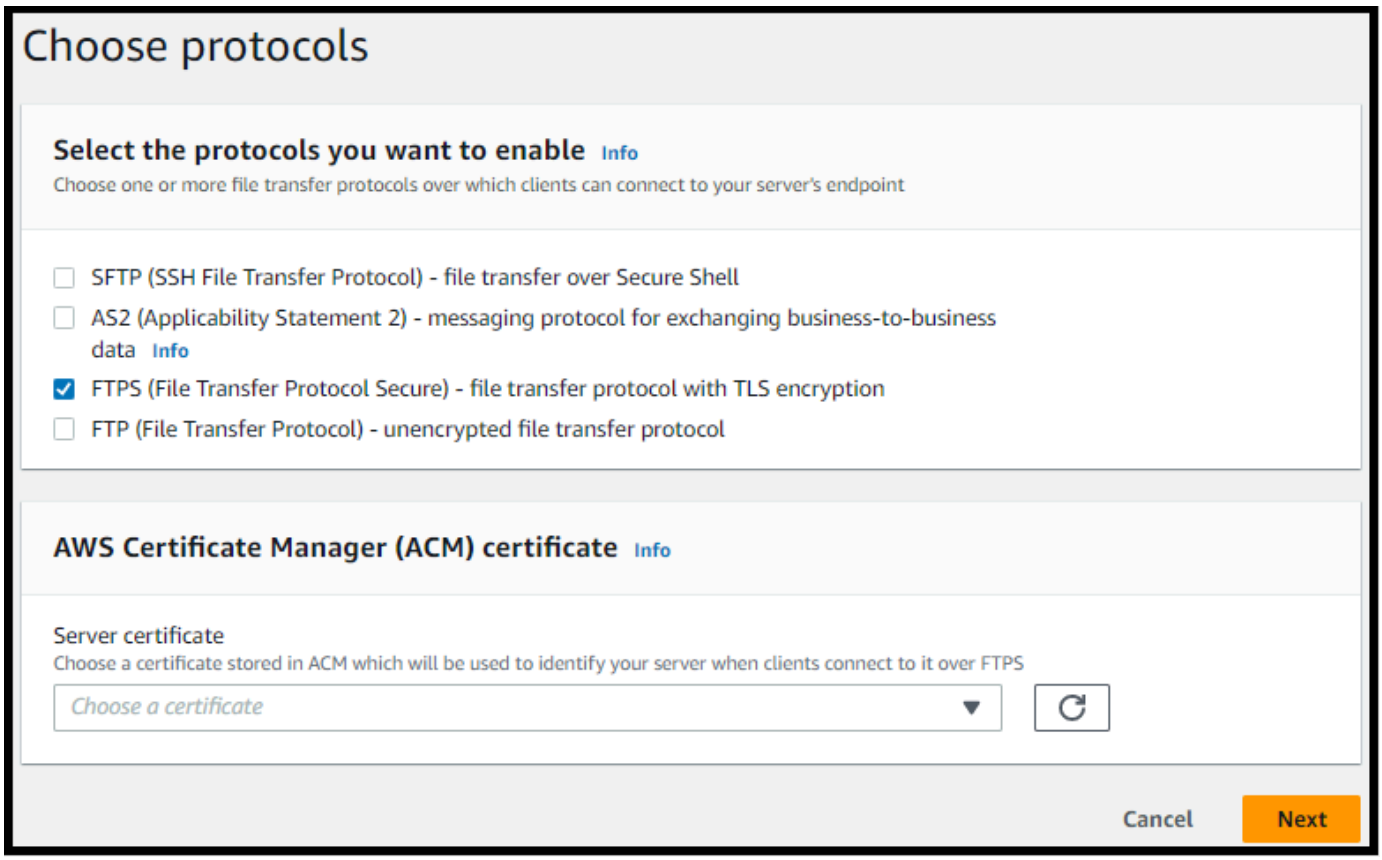
既存の証明書を にインポートするにはACM、AWS Certificate Manager 「 ユーザーガイド」の「 に証明書をインポートACMする」を参照してください。

プライベート IP アドレスFTPSを介してプライベート証明書を使用するように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 アドレスを指定し、発行者に関する情報を含む有効な SSL/TLS X.509 バージョン 3 証明書である必要があります。

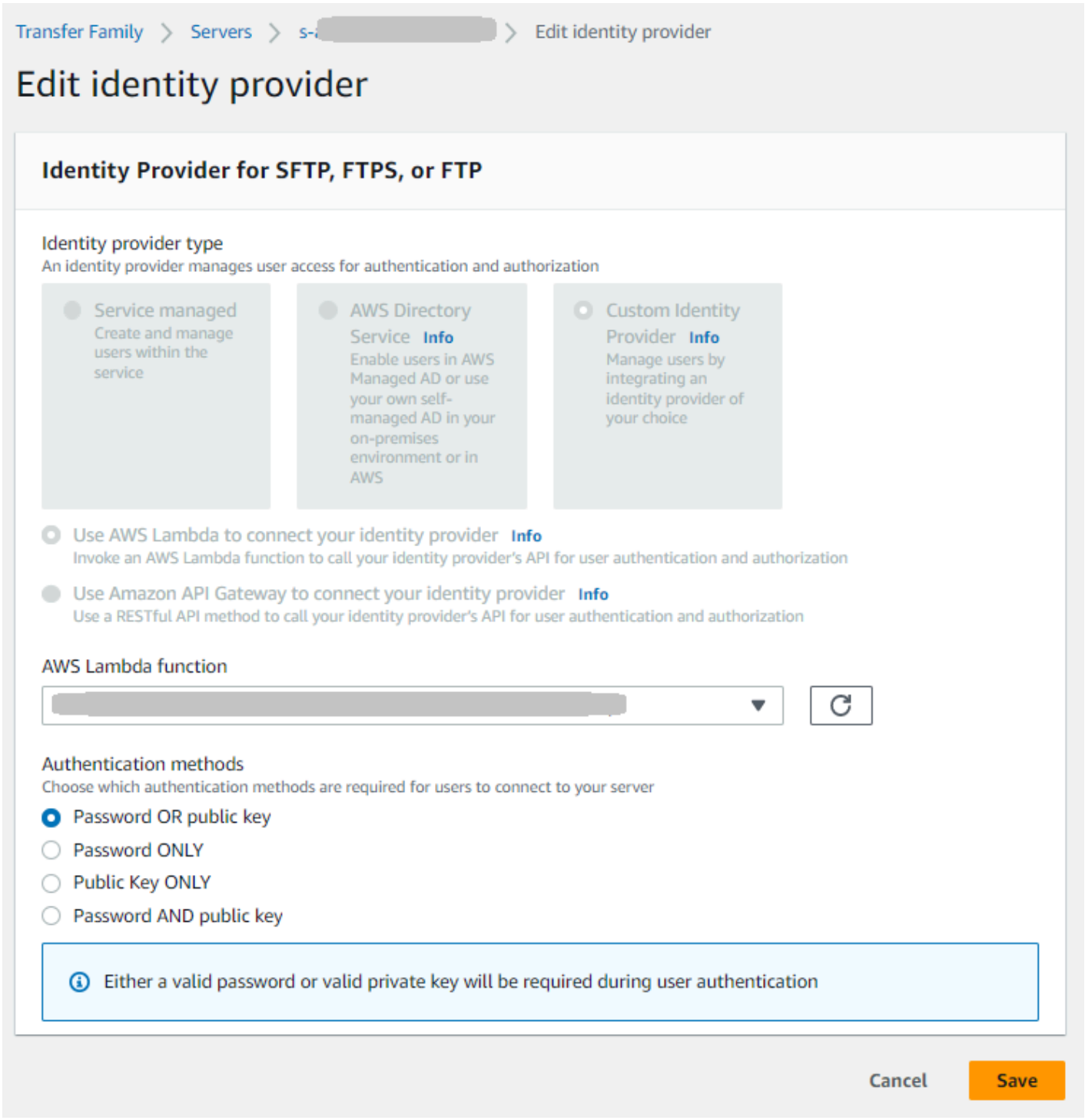


3. [Save] を選択します。[Server details] (サーバーの詳細) ページが再表示されます。

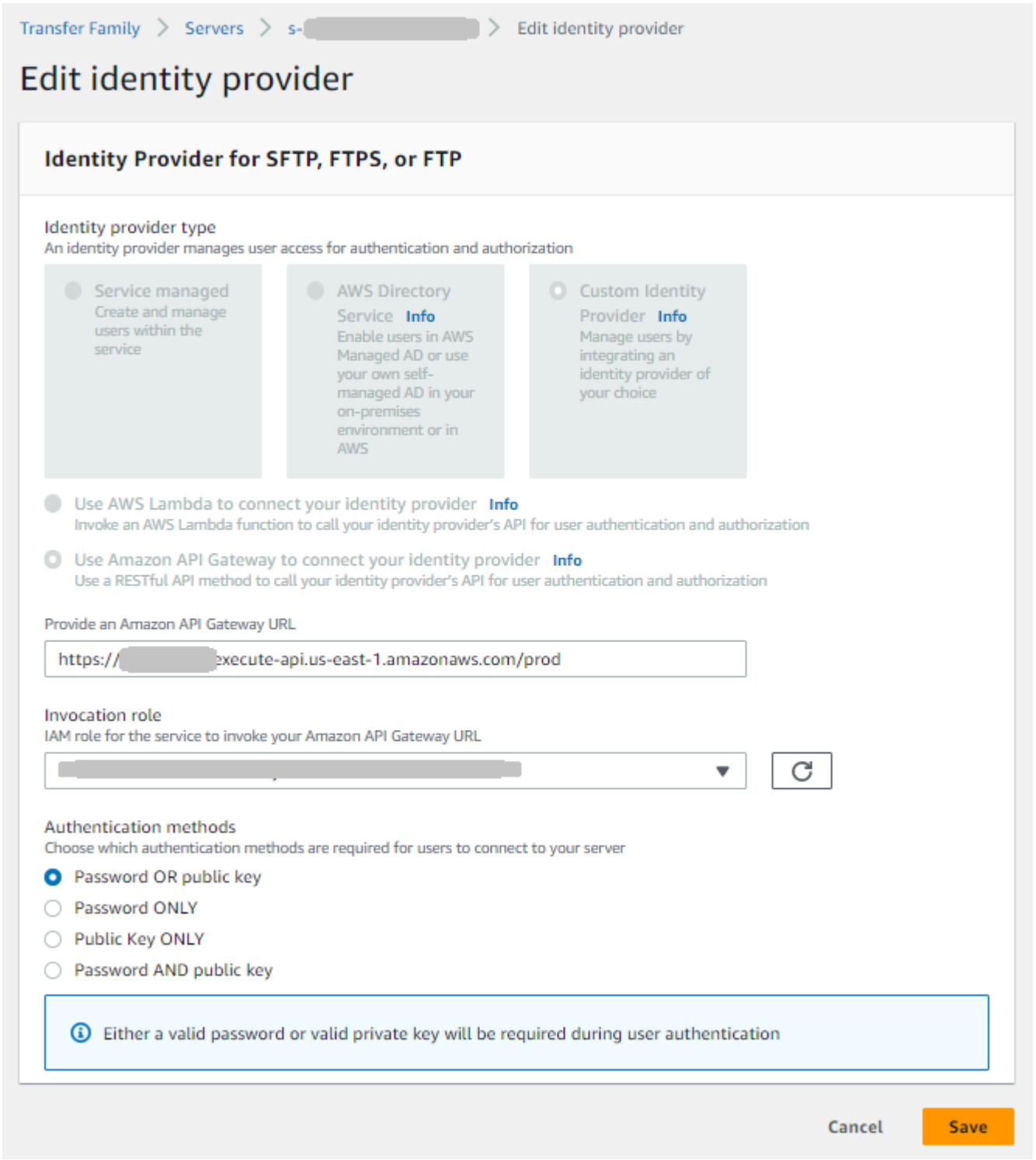
カスタム ID プロバイダーパラメーターを編集

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 Gateway 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、また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。

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. [Edit endpoint configuration] (エンドポイント設定の編集) ページで [Endpoint type] (エンドポイントタイプ) について次のいずれかを選択します。

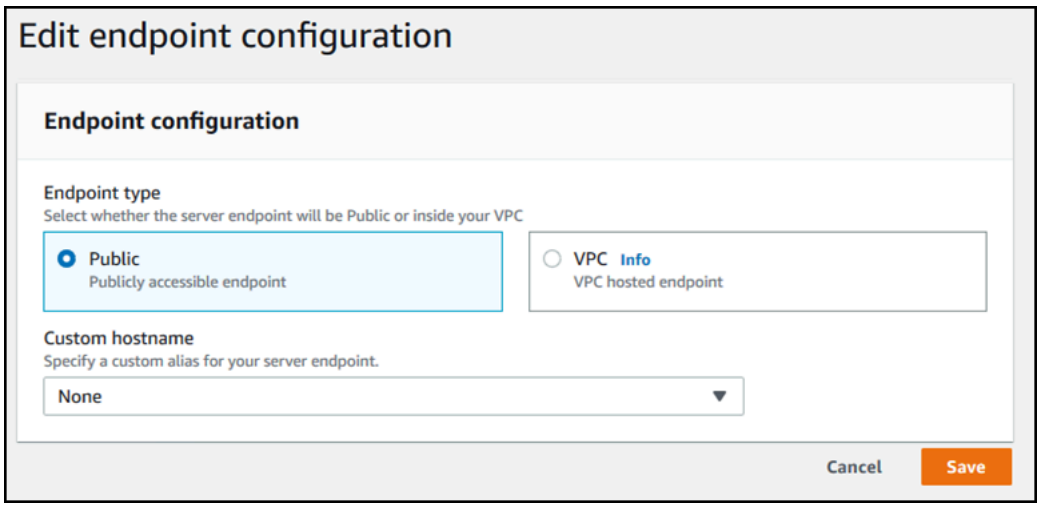
   - [公開] ー このオプションは、インターネット経由でサーバーにアクセスできるようにします。
   - VPC – このオプションにより、仮想プライベートクラウド () でサーバーにアクセスできるようになりますVPC。の詳細についてはVPC、「」を参照してくださいVirtual Private Cloud でサーバーを作成する。
3. [Custom hostname] (カスタムホスト名) の場合、次のいずれかを選択します。

   - [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 – 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。
   - その他 DNS – 外部DNSサービスで既に所有しているホスト名を使用するには、その他 DNSを選択します。

   Amazon Route 53 DNS エイリアスまたはその他DNSを選択すると、サーバーのエンドポイントに関連付ける名前解決方法を指定します。

   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、指定した名前とDNSサービスが解決できるDNS名前を使用します。Route 53 をDNSリゾルバーとして使用することも、独自のDNSサービスプロバイダーを使用することもできます。 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. [Save] 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、その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、ログ記録ロール、構造化JSONログ記録、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。

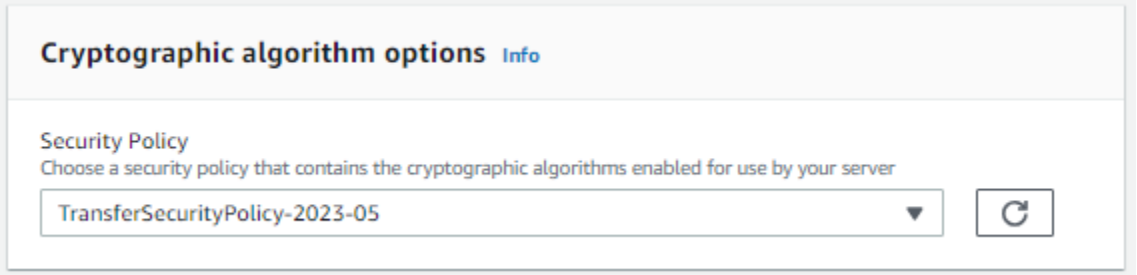
> **Note**
>
> エンドポイントが有効になっている場合FIPS、FIPSセキュリティポリシーをセキュリティFIPSポリシー以外のポリシーに変更することはできません。

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、「のセキュリティポリシー AWS Transfer Family」を参照してください。

3. [Save] 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を確認できます。

AWS CLI

を使用してセキュリティポリシーを編集するには CLI

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 すTransferSecurityPolicy-2018-11。

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のセキュリティポリシー AWS Transfer Family。

成功すると、 コマンドは次のコードを返します。また、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マネージドワークフローを変更できます。

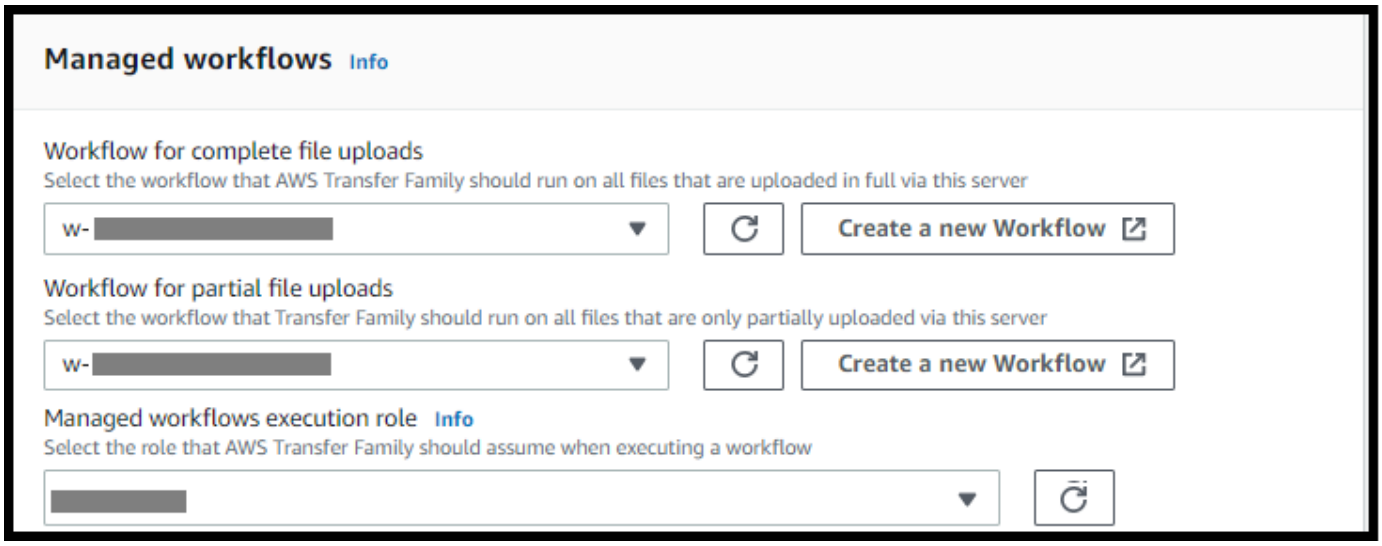
管理対象ワークフローを変更するには

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、「IAM ワークフローの ポリシー」を参照してください。[Save] を選択します。



3. [Save] 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] 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か、オフラインにすることができます。

サーバーをオンラインにするには

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

エラー条件`START_FAILED`または が見つかった場合は`STOP_FAILED`、 AWS サポート 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する

> **Important**
>
> 既存のユーザーを既存の SFTP対応サーバーから新しい SFTP対応サーバーに移行する予定がない場合は、このセクションを無視してください。
> 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、信頼されたホストキーが存在しないというメッセージが表示されてすぐに失敗したり、脅威のある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。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場合にのみ、これを行います。

ユーザーが SFTP対応サーバーの真正性を再度検証するように求めら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攻撃に関する警告を受けることもできなくなります。

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、すべてのサーバーに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にのみ役立ちます。

トピック

- サーバーホストキーを追加する
- サーバーのホストキーを削除する
- 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の追加情報

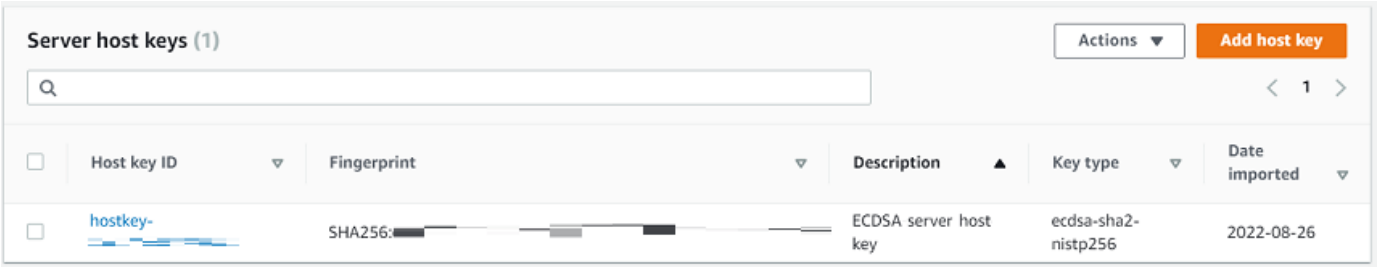
サーバーホストキーを追加する

AWS Transfer Family コンソールで、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。例えば、元のキーがRSAキーである場合は、追加のECDSAキーを追加できます。

> **Note**
>
> SFTP クライアントは、アクティブなサーバーキーの 1 つと照合できる最初のパブリックキー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

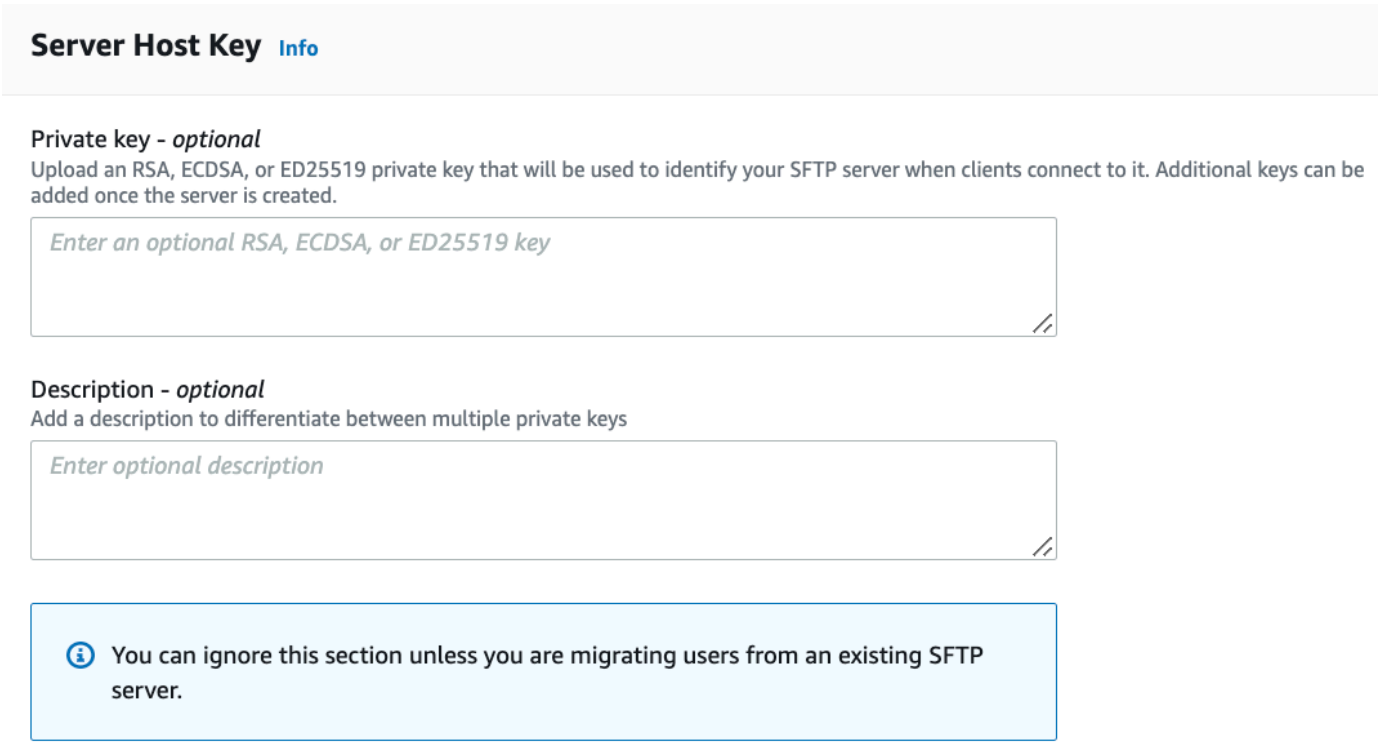
1. で AWS Transfer Family コンソールを開きますhttps://console.aws.amazon.com/transfer/。
2. 左側のナビゲーションペインで、サーバー を選択し、SFTPプロトコルを使用するサーバーを選択します。
3. サーバーの詳細ページで [サーバーホストキー] セクションまでスクロールします。



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で、クライアントが SFTP対応サーバー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、RSAECDSA、、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 でのSSHキーの作成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 SFTP対応サーバーを作成する場合は、 the section called “CreateServer”APIオペレーションでホストキーをパラメータとして指定します。 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。

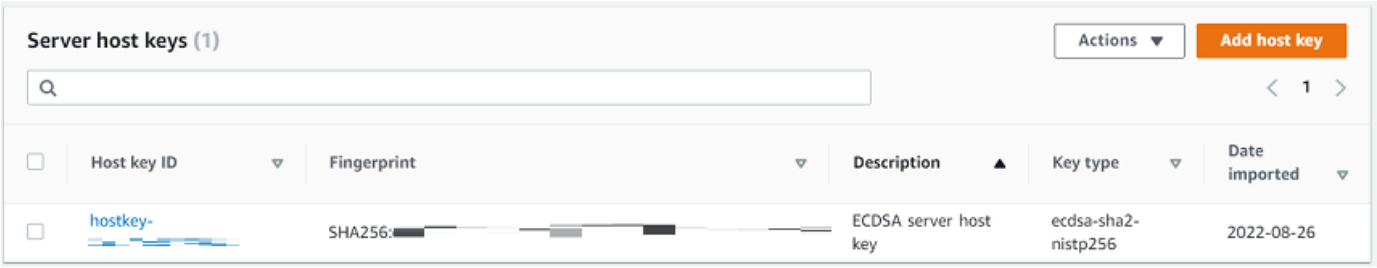
```
aws transfer import-host-key --description key-description --server-id your-server-id
 --host-key-body file://my-host-key
```

サーバーのホストキーを削除する

AWS Transfer Family コンソールで、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. で AWS Transfer Family コンソールを開きますhttps://console.aws.amazon.com/transfer/。
2. 左側のナビゲーションペインで、サーバー を選択し、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。

> **Note**
>
> Transfer Familyは、各アルゴリズムに最初に追加されたキーをアクティブなホストキーとして使用します。SFTP サーバーごとに最大 10 個のホストキーを関連付けることができますが、アルゴリズムごとにアクティブなキーは 1 回に 1 つだけです。

たとえば、次のサーバーホストキーセットをサーバーに追加したとします。

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	アクティブ
RSA	2020 年 4 月 1 日	不可
ECDSA	2020 年 2 月 1 日	不可
ED25519	2019年12月1日	不可
RSA	2019 年 10 月 1 日	可能
ECDSA	2019 年 6 月 1 日	可能
ED25519	2019 年 3 月 1 日	可能

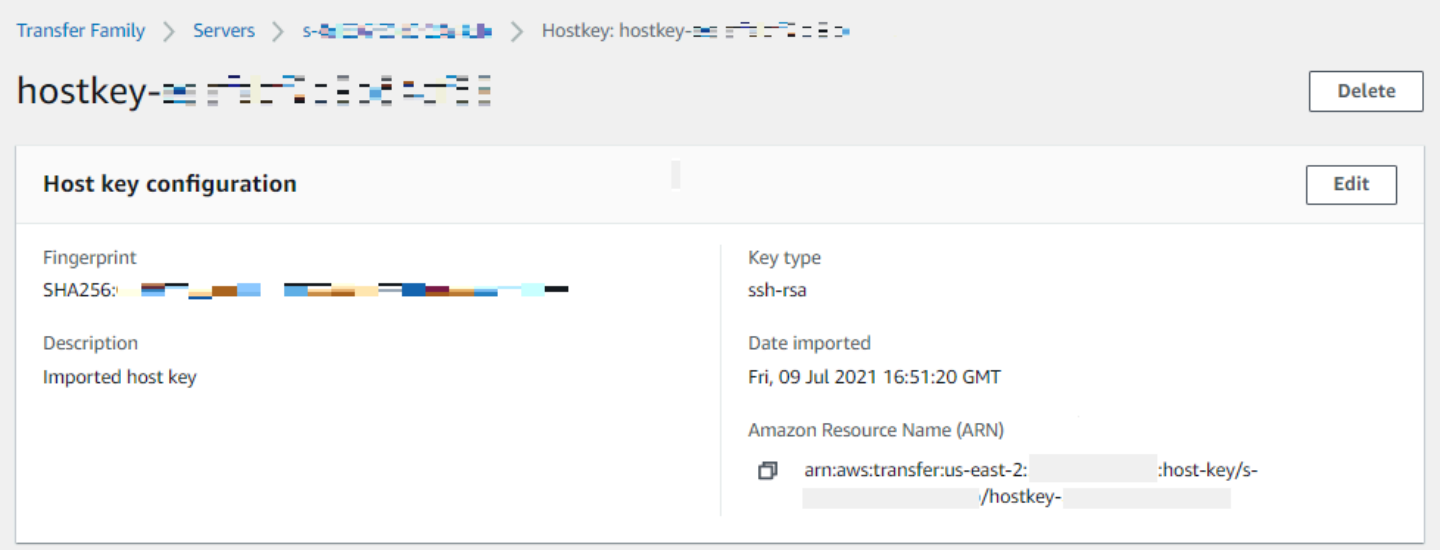
各アルゴリズムの最も古いキーがアクティブになります。2019 年 10 月 1 日に追加したRSAキーを削除すると、2020 年 4 月 1 日に追加したRSAキーがアクティブになります。

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. 同じタイプの最も早く残っているキーが、アクティブにしたいキーであることを確認してください。

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

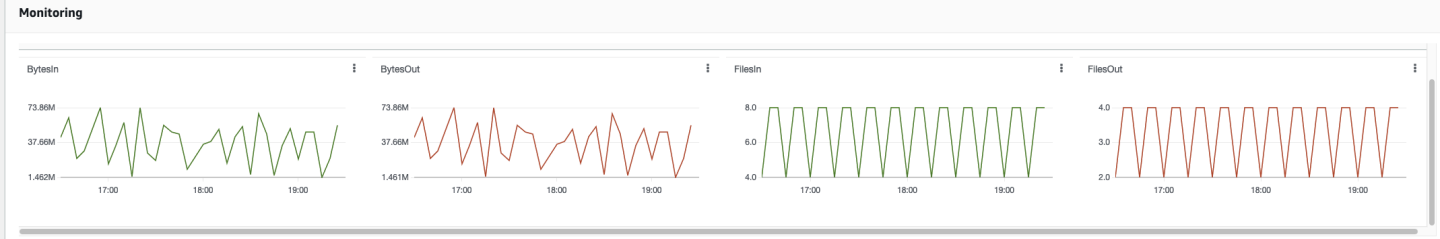
[Server details] (サーバーの詳細) ページでサーバーのメトリクスに関する情報を取得できます。これにより、ファイル転送のワークロードを 1 か所でモニタリングできます。一元化されたダッシュボードを使用して、パートナーと交換したファイルの数を追跡し、その詳細な使用状況を追跡できます。詳細については、「SFTP、FTPS、および FTPサーバーの詳細を表示する」を参照してください。Transfer Family で使用できるメトリクスの説明を次の表に示します。

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。

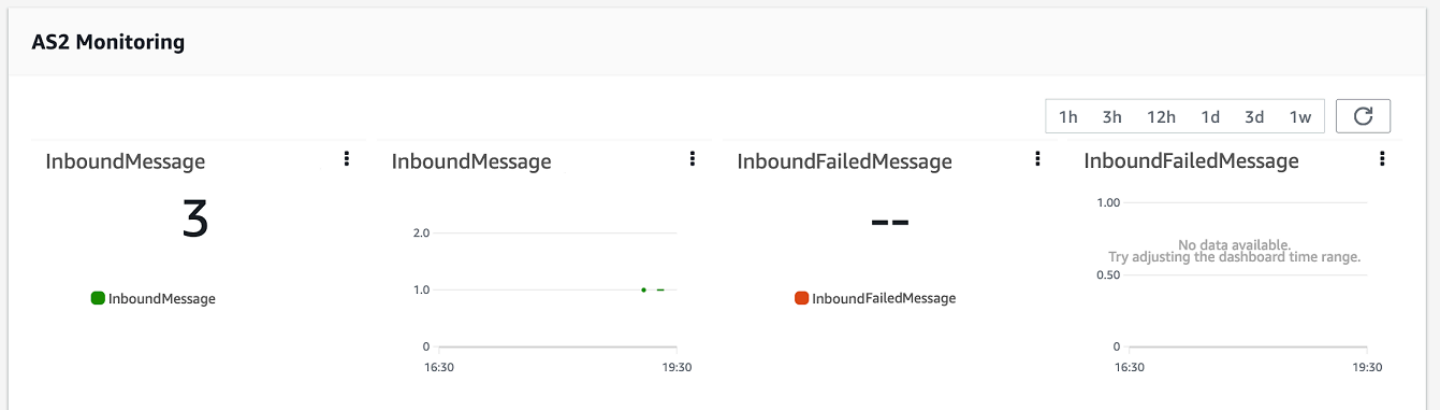
名前空間	メトリクス	説明
		単位: カウント 期間: 5 分
	BytesOut	サーバーから転送された総バイト数。 単位: 個 期間: 5 分
	FilesIn	サーバーに転送されたファイルの合計数。 AS2 プロトコルを使用するサーバーの場合、このメトリクスは受信したメッセージの数を表します。 単位: カウント 期間: 5 分
	FilesOut	サーバーから転送されたファイルの合計数。 単位: カウント 期間: 5 分
	InboundMessage	取引先から正常に受信したAS2メッセージの合計数。 単位: カウント 期間: 5 分
	InboundFailedMessage	取引先から受信できなかったAS2メッセージの合計数。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
	OnPartialUploadExecutionsStarted	サーバーで開始されたワークフロー実行の合計数 on-partial-upload。 単位: カウント 期間 = 1 分
	OnPartialUploadExecutionsSuccess	サーバーで成功 on-partial-uploadしたワークフロー実行の合計数。 単位: カウント 期間 = 1 分
	OnPartialUploadExecutionsFailed	サーバーで失敗した on-partial-uploadワークフロー実行の合計数。 単位: カウント 期間 = 1 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

- [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、AS2モニタリング情報の下にモニタリング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」 ー 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」 ー 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、ユーザーの AWS Transfer Family リソースへのアクセスを AWS Identity and Access Management 制御できます。IAM ポリシーは、通常 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します。IAM ポリシーを使用して、ユーザーに実行を許可するファイルオペレーションと実行しないファイルオペレーションを定義します。IAM ポリシーを使用して、ユーザーにアクセス権を付与する Amazon S3 バケットを定義することもできます。ユーザーに対してこれらのポリシーを指定するには、IAMポリシーと信頼関係が関連付け AWS Transfer Family られている IAM ロールを作成します。

各ユーザーにIAMロールが割り当てられます。 AWS Transfer Family が使用する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 AWS Transfer Family を想定します。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 「サービスへの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 Storage Blog には、最小特権アクセスを設定する方法を説明する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、ポリシーに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、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 を参照してください。

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および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、そのユーザーに指定された Amazon S3 バケットへの読み取り/書き込みアクセスが付与されます。

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であるため、異なる Amazon リソースネーム () を使用して指定されますARNs。

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 ではEFS、POSIXファイルアクセス許可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リファレンスの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、ポリシーで次のIAM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。また、IAM ロール定義でポリシー変数として使用することはできません。これらの変数はIAMポリシーで作成し、ユーザーを設定するときに直接指定します。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、 に必要なIAMユーザー名ではなくユーザー名を指します AWS Transfer Family。

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 メソッドの実装。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 パラメータには、Resourceステートメントの S3 Amazon リソースネーム (/) の一部として使用できるように、先頭のスラッシュ (ARN) が削除されました。

> **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。このアプローチにより、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IAMロールとポリシーの管理のオーバーヘッドが軽減されます。

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のアクセス許可」および 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) ではなくJSON、ポリシー を保存します。そのため、IAMコンソールでポリシーを変更するときは、 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。これを行うには、`s3:PutObject`アクションを許可するIAMポリシーを作成しますが、キーが「/」 (スラッシュ)

で終わると拒否します。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。

のログ記録 AWS Transfer Family

AWS Transfer Family は、 AWS CloudTrail と Amazon の両方と統合 CloudTrail され、異なるが補完的な目的 CloudWatch を果たし CloudWatchます。

- CloudTrail は、 内で実行されたアクションの記録を作成する AWS 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 APIコールなどのアクティビティの呼び出しを継続的にモニタリングおよび記録します。これにより、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、誰がどのようなアクションを実行したか、いつ、どこからどのようなアクションを取ったかをログに記録できます AWS 。監査、アクセス管理、規制コンプライアンス CloudTrail に役立ちます。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、全体的なシステムヘルスを可視化します。 は、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運用タスク CloudWatch に役立ちます。詳細については、「Amazon CloudWatch ユーザーガイド」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
- サーバーを作成するためのログエントリの例

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。また、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してコストを削減することもできます。

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
- S3 アクセスログへの Amazon S3 API呼び出しのログ記録
- 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

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 (サーバーにアタッチされたワークフローのログ記録LoggingRoleに使用) または のいずれかを選択できますStructuredLogDestinations。StructuredLogDestinations を使用すると、次のような利点があります。

- 構造化JSONされた形式でログを受信します。
- Amazon CloudWatch Logs Insights JSON を使用してログをクエリします。これにより、フォーマットされたフィールドが自動的に検出されます。
- AWS Transfer Family リソース間でロググループを共有することで、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まとめることができるため、モニタリング設定とログ保持設定を簡単に管理できます。
- 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、 UpdateServerAPI呼び出しを使用してログ記録設定を変更できます。詳細については、「」を参照してくださいStructuredLogDestinations。

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。

- ワークフローをサーバーに関連付ける場合、 CreateServerまたは UpdateServerAPI呼び出しを使用しても、システムは自動的にログ記録ロールを作成しません。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

- コンソールを介し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JSONログ記録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。
- 既存のサーバーは、コンソールを介していつでも構造化JSONログ記録を有効にできます。
- コンソールで構造化JSONログ記録を有効にすると、既存のログ記録メソッドが無効になり、顧客への二重請求がなくなります。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ログ記録を有効にすると、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ログ記録を有効にすると、コンソールからいつでもロググループの宛先を変更できます。
- 構造化JSONログ記録を有効にすると、 で両方のログ記録タイプを有効にしている場合、コンソールでログ記録ロールを編集することはできませんAPI。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 を使用してサーバーを作成および編集する際のログ記録に関する考慮事項 SDK

- を使用して新しいサーバーを作成する場合はAPI、どちらかまたは両方のタイプのログ記録を設定するか、ログ記録を選択しないでください。
- 既存のサーバーでは、構造化JSONログ記録をいつでも有効または無効にします。
- ロググループは、 を通じてAPIいつでも変更できます。
- ログ記録ロールは、 を通じてAPI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`

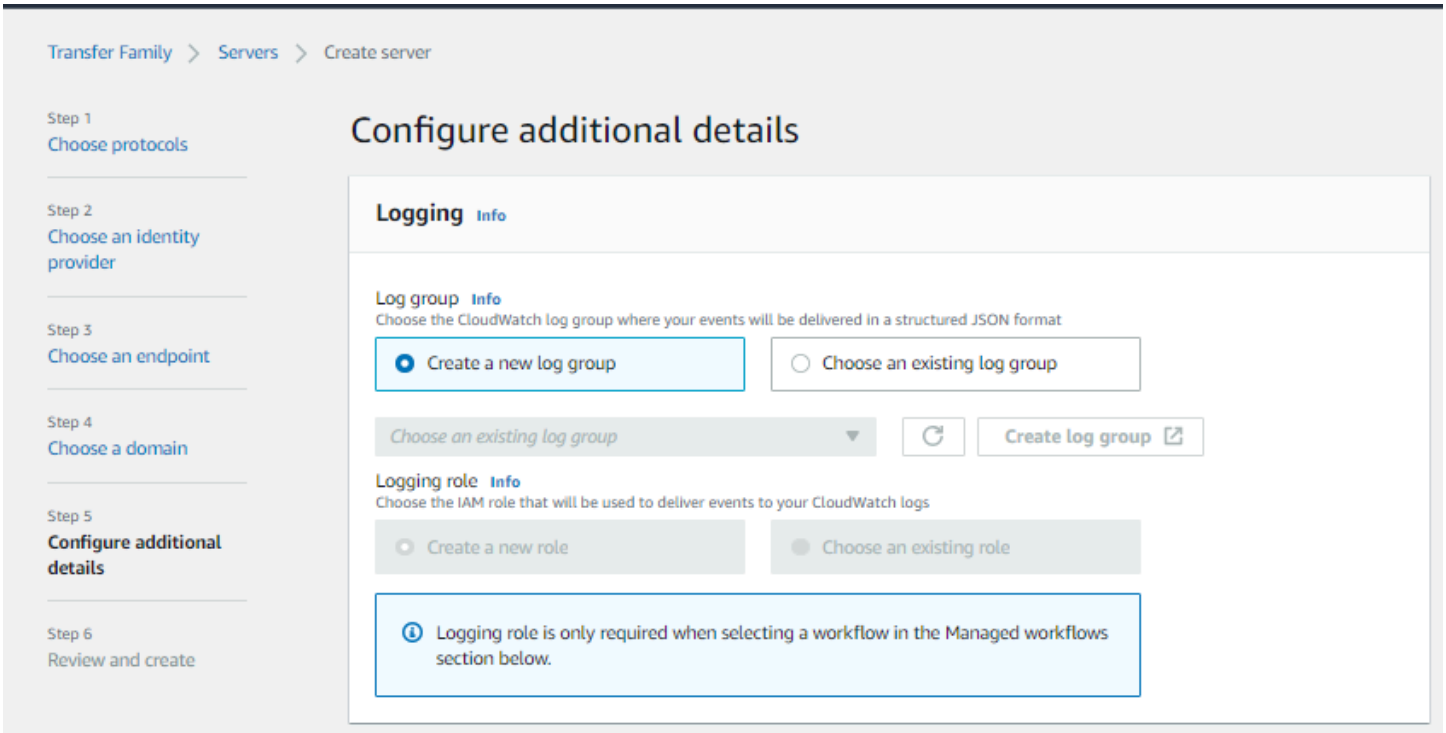
ポリシーの例は、 セクションで入手できます CloudWatch ログ記録ロールを設定する。

トピック

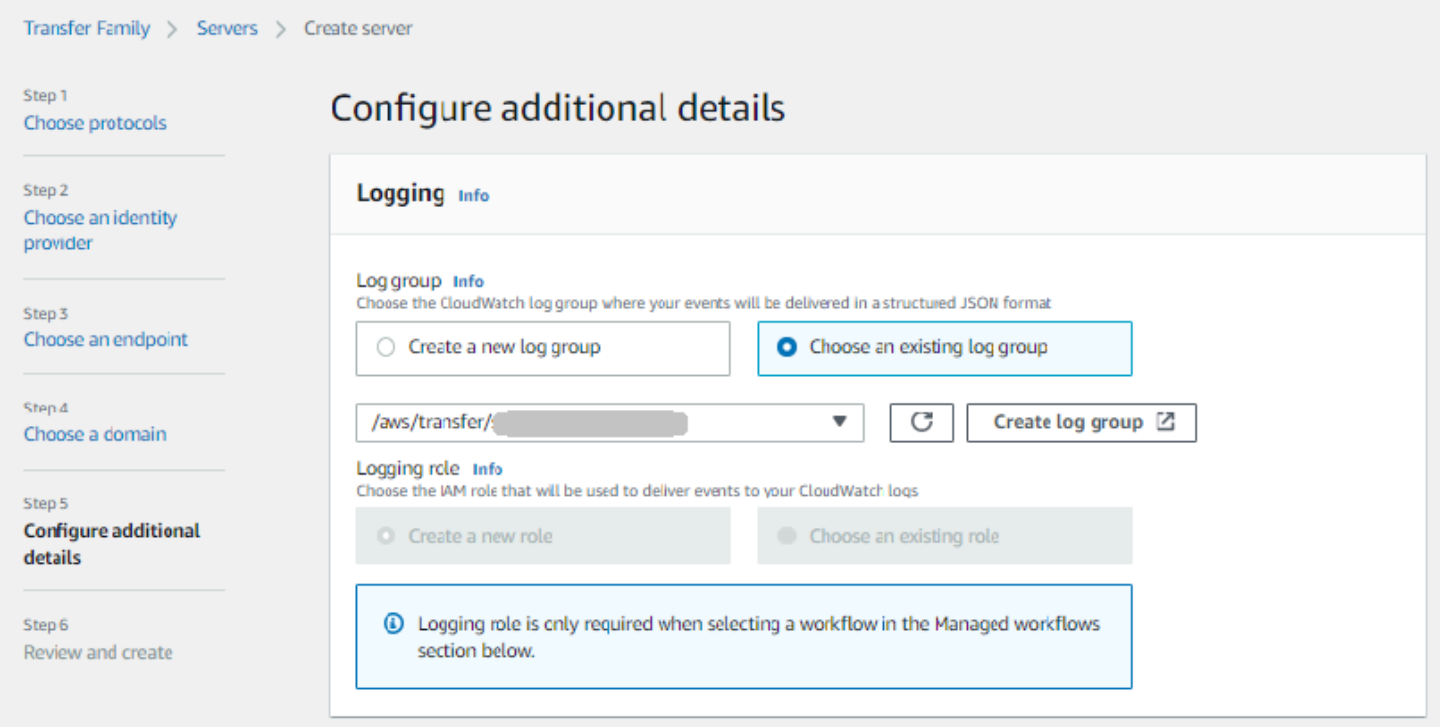
- サーバーのログ記録の作成
- サーバーのロギングを更新します。
- サーバー設定の表示

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。



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、 CloudWatch 「 Logs でロググループを作成する」を参照してください。

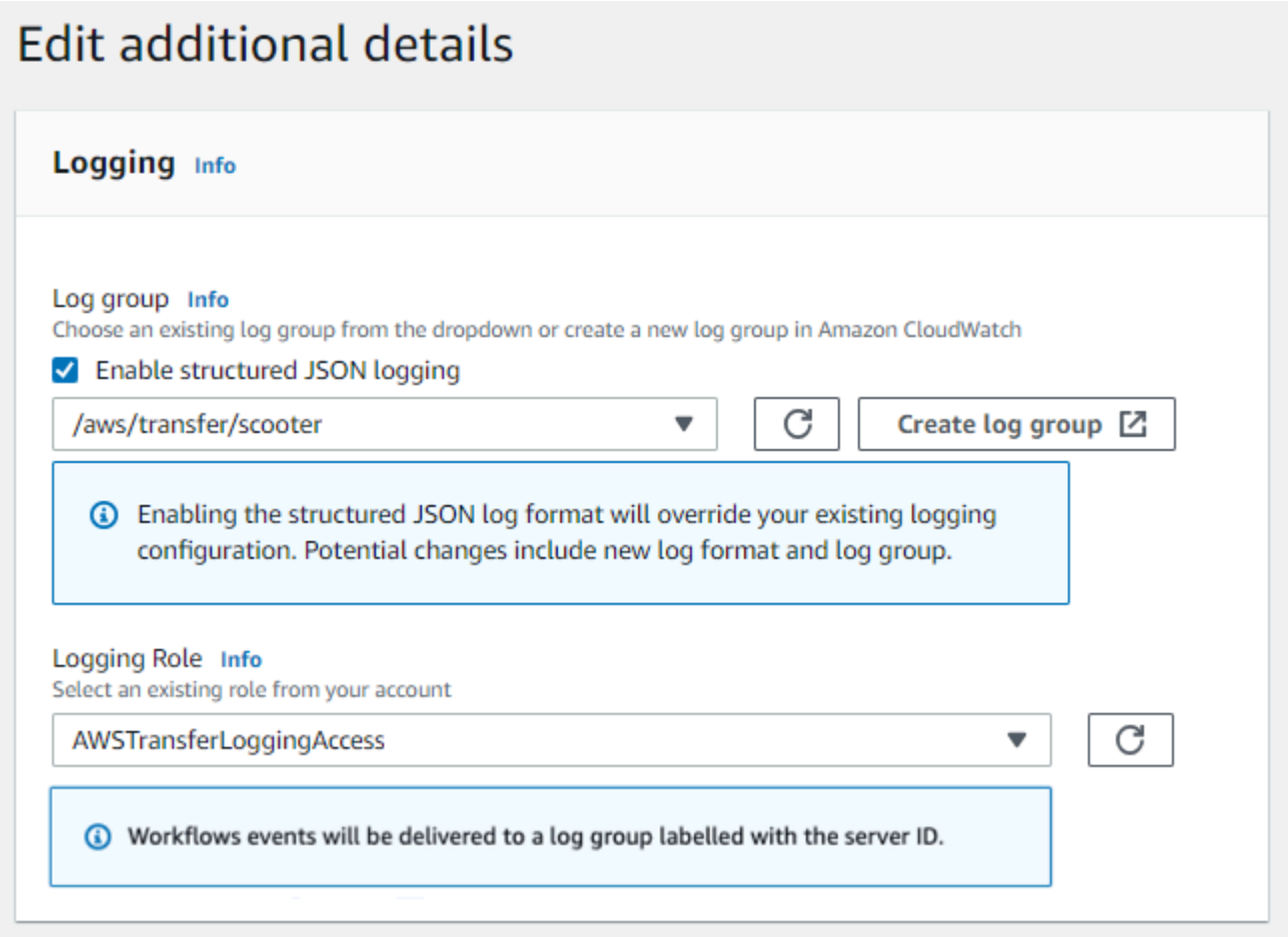
サーバーのロギングを更新します。

ロギングの詳細は、更新のシナリオによって異なります。

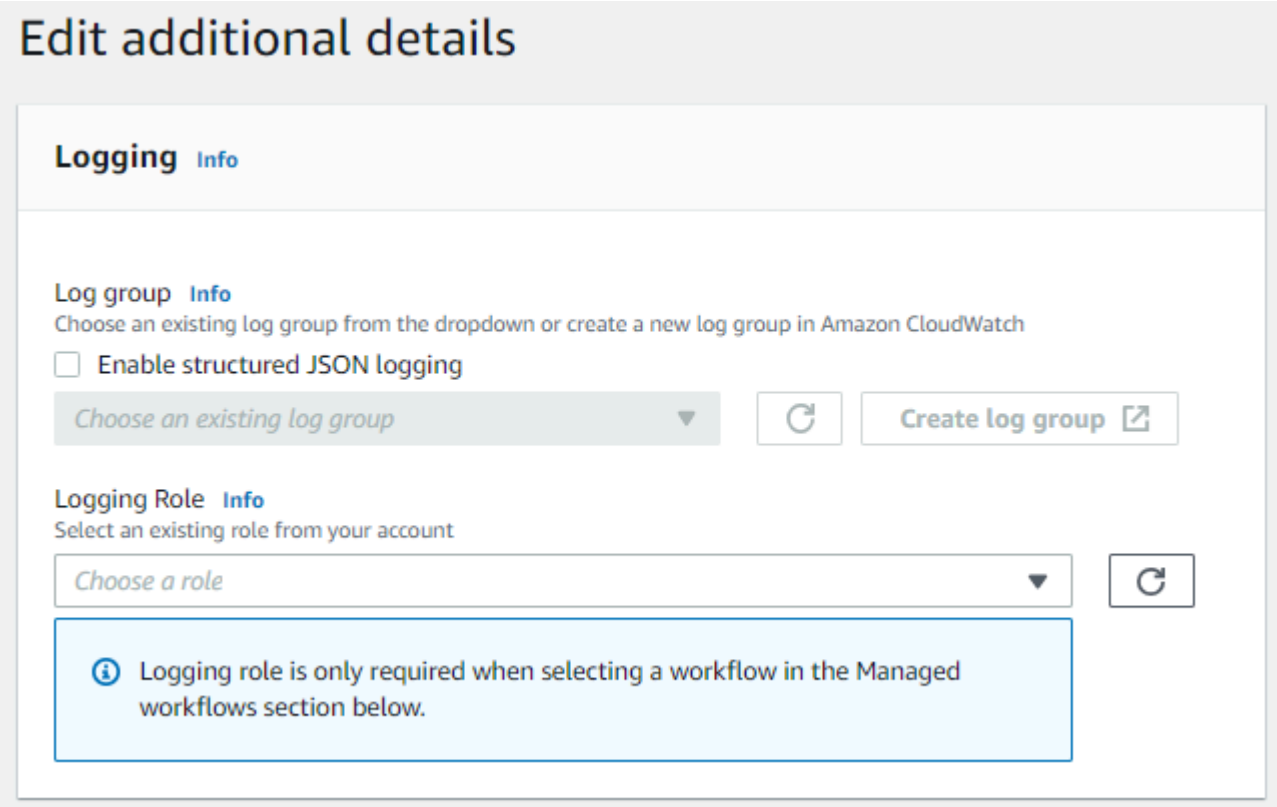
> **Note**
>
> 構造化JSONログ記録にオプトインすると、Transfer Family が古い形式でのログ記録を停止するが、新しいJSON形式でのログ記録を開始するまでに時間がかかることがまれにあります。その結果、イベントが記録されないことがあります。サービスが中断されることはありませんが、ログ記録方法を変更してから最初の 1 時間はログが削除される可能性があるため、ファイルの転送には注意が必要です。

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。

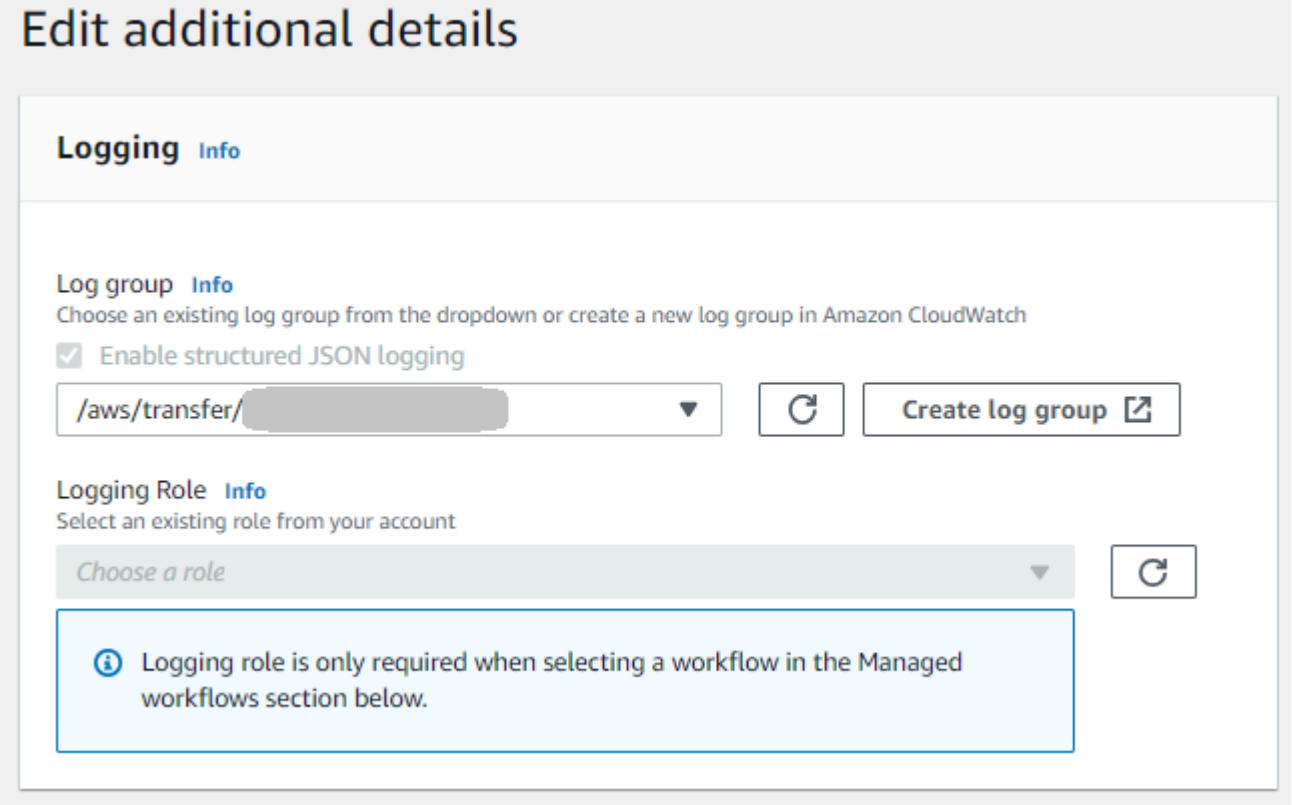
- サーバーではすでにログ記録ロールが有効になっていますが、構造化JSONログ記録が有効になっていません。



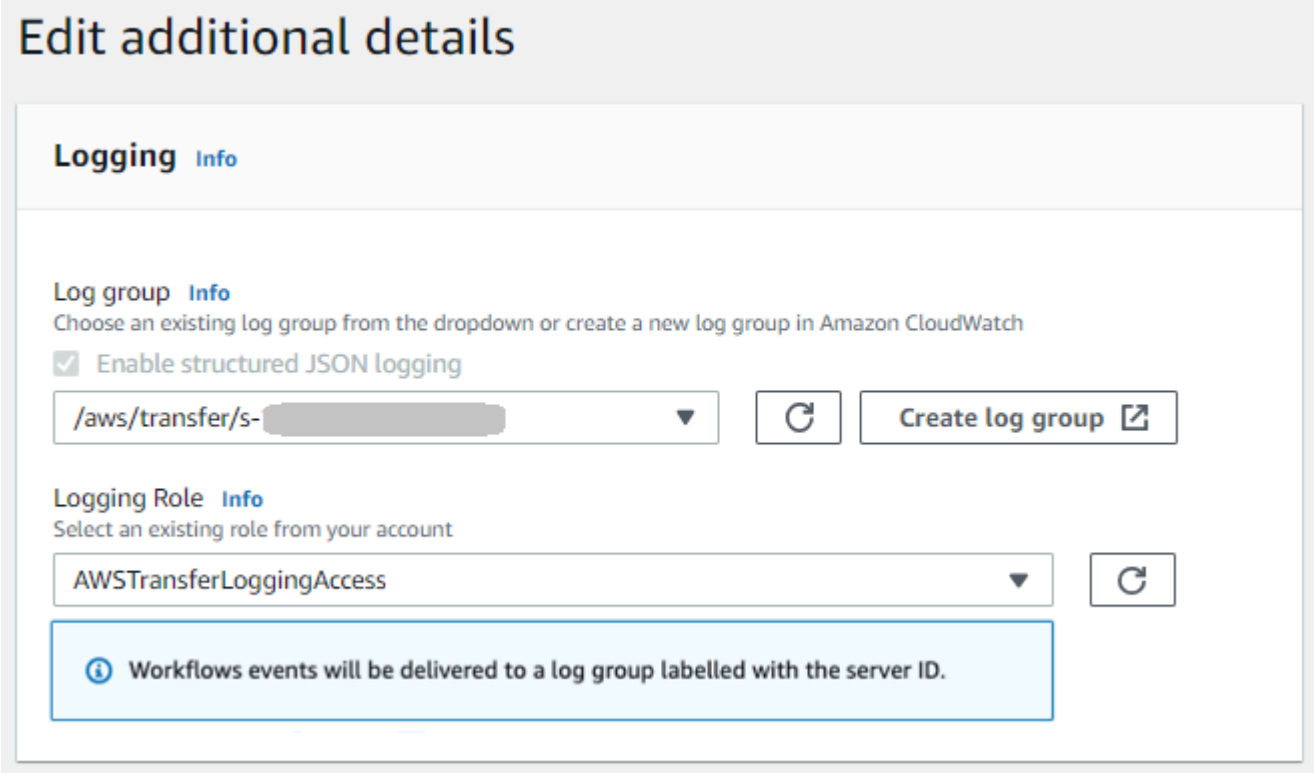
- サーバーでログ記録が有効になっていません。



- サーバーで構造化JSONログ記録が既に有効になっていますが、ログ記録ロールが指定されていません。



- サーバーでは、構造化JSONログ記録が既に有効になっており、ログ記録ロールも指定されています。

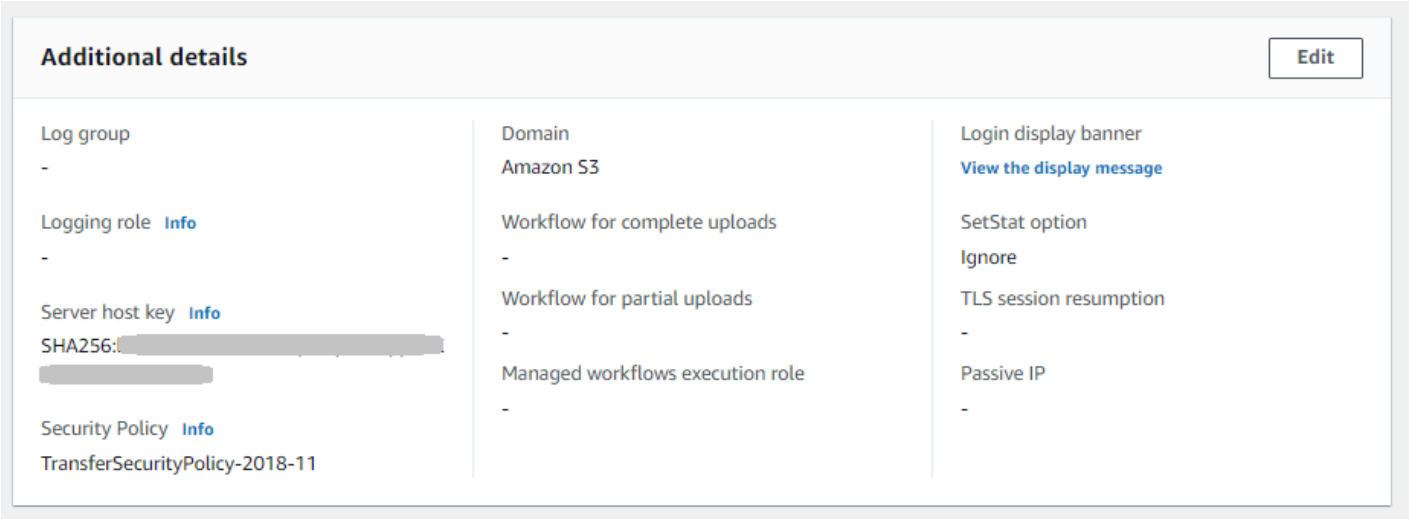


サーバー設定の表示

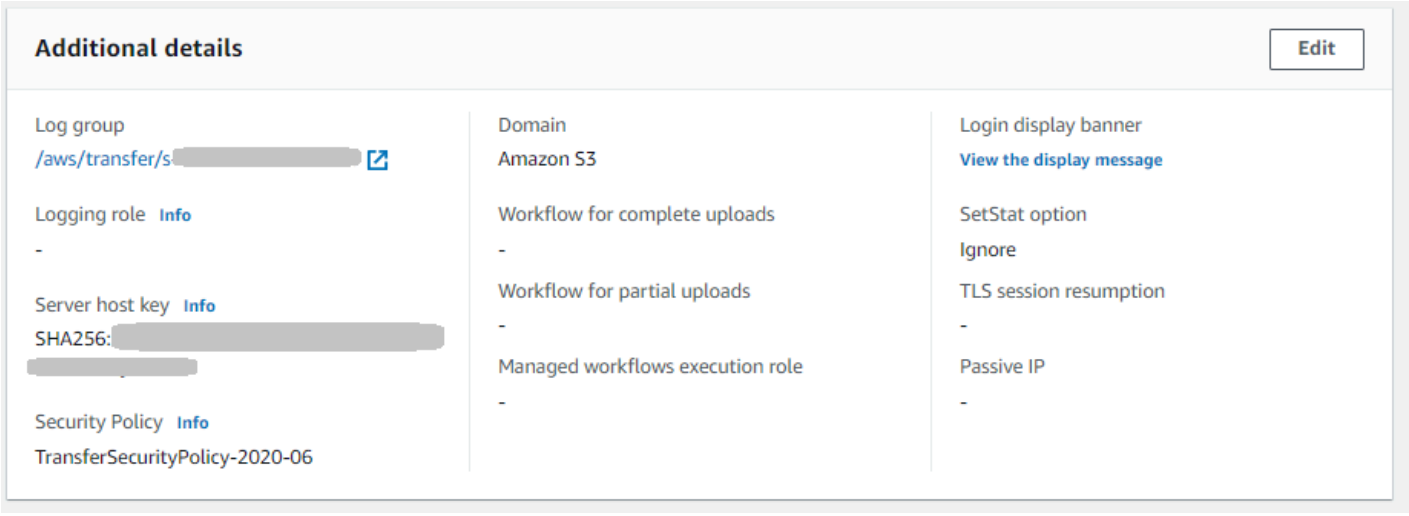
サーバー設定ページの詳細は、シナリオによって異なります。

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。

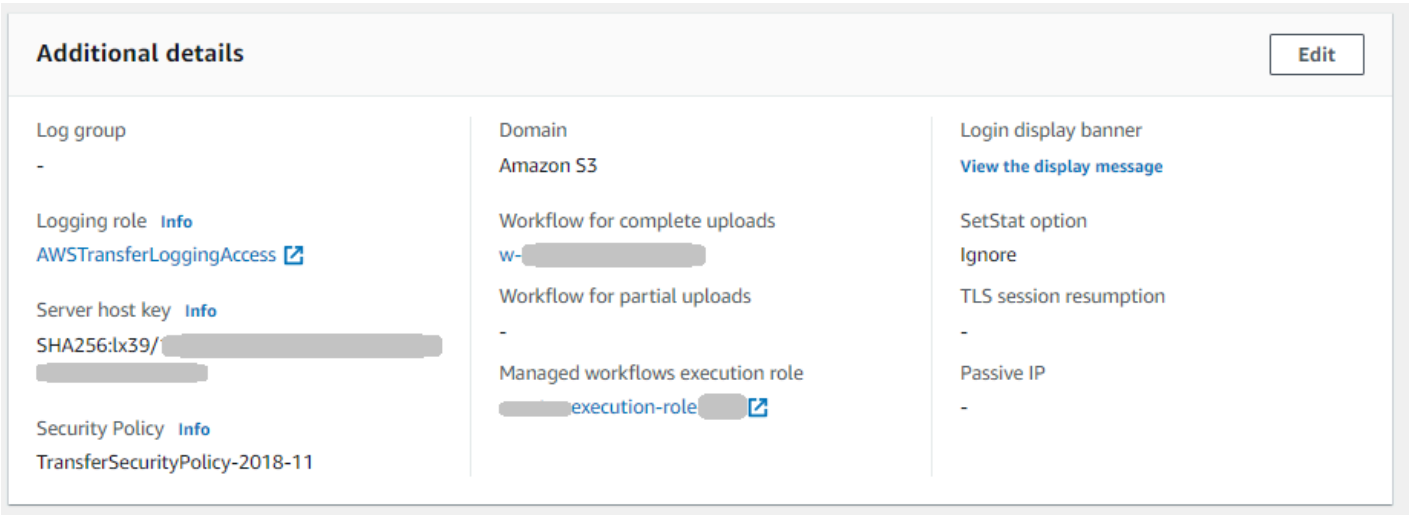
- ログは有効になっていません。



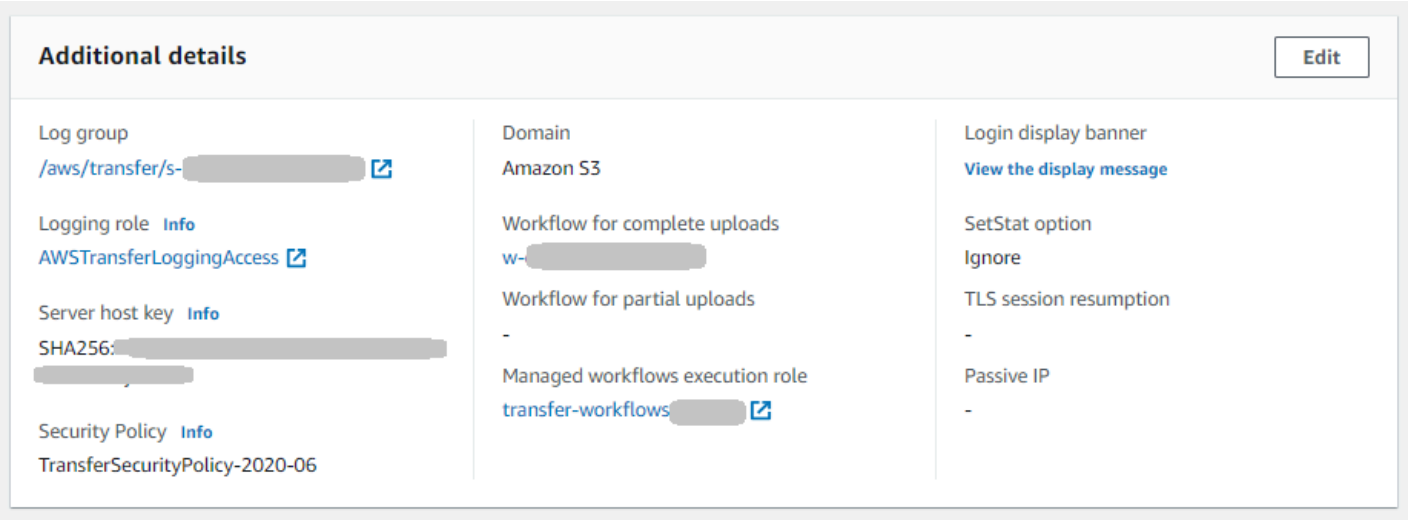
- 構造化JSONログ記録が有効になっています。



- ログ記録ロールは有効になっていますが、構造化JSONログ記録は有効になっていません。



- 両方のタイプのログ記録 (ログ記録ロールと構造化JSONログ記録) が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に関する統合された監査とログ記録を提供します。さらに、 はワークフローのいくつかのメトリクス 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、「」を参照してください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. で Amazon CloudWatch コンソールを開きますhttps://console.aws.amazon.com/cloudwatch/。
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) ファイル転送プロトコル (SFTP) セッションごとに 1 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

アクセスを設定するには、リソースベースのIAMポリシーと、その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IAM CloudWatchポリシーを作成します。次に、IAMロールを作成し、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 とは CloudWatch」および CloudWatch「Amazon ログとは」を参照してください。 CloudWatch

CloudWatch ログ記録を許可するには、次のIAMポリシー例を使用します。

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/*"
        }
    ]
}
```

前述のポリシー例では、 で**Resource**、 を置き換えます。*region-id* また、*AWS #####* 値を使用します。例えば、**"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 ロールを作成して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ロール間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

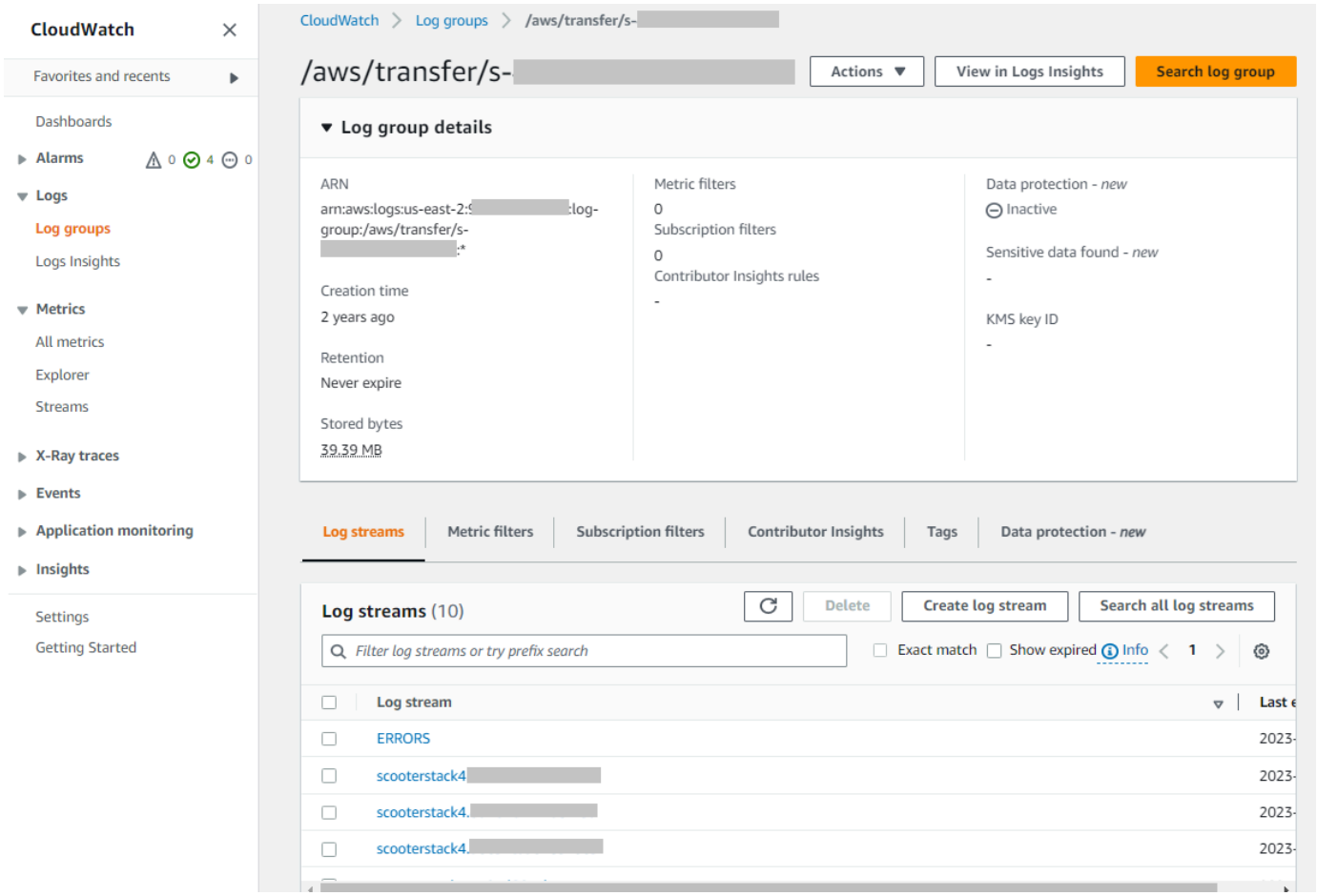
3. アクセス許可ポリシーのアタッチセクションで、先ほど作成した CloudWatch ログポリシーを見つけて選択し、次へ: タグ を選択します。
4. (オプション) タグのキーと値を入力して [Next: Review] (次へ: レビュー) を選択します。
5. [Review] (確認) ページで新しいロールの名前と説明を入力してから [Create role] (ロールの作成) を選択します。
6. ログを表示するには、[Server ID] (サーバー ID) を選択してサーバー設定ページを開き、[View logs] (ログの表示) を選択します。 CloudWatch コンソールにリダイレクトされ、ログストリームが表示されます。

サーバーの CloudWatch 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

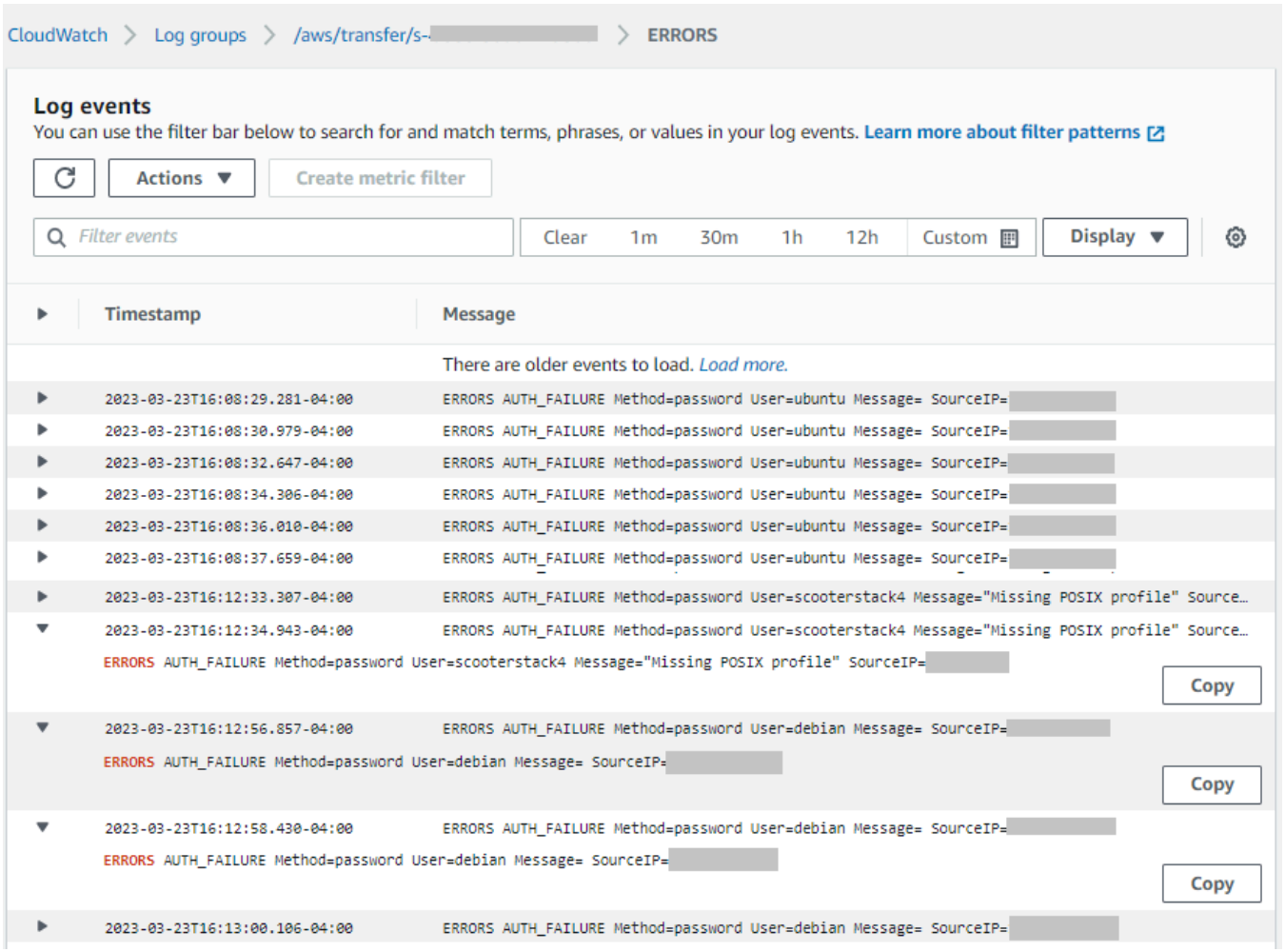
Transfer Familyのサーバーログを表示するには

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。

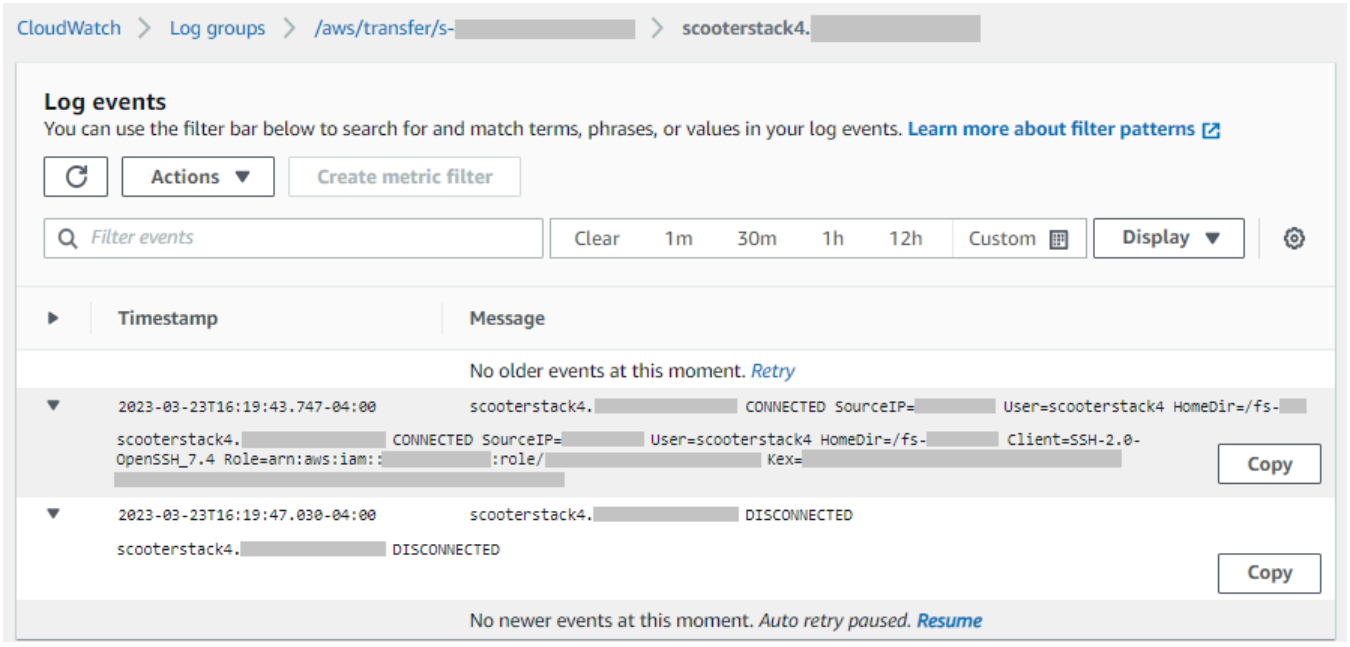


4. ログストリームを選択すると、そのストリームの詳細と個々のエントリを表示できます。

   - のリストがある場合はERRORS、サーバーの最新エラーの詳細を表示するように選択できます。

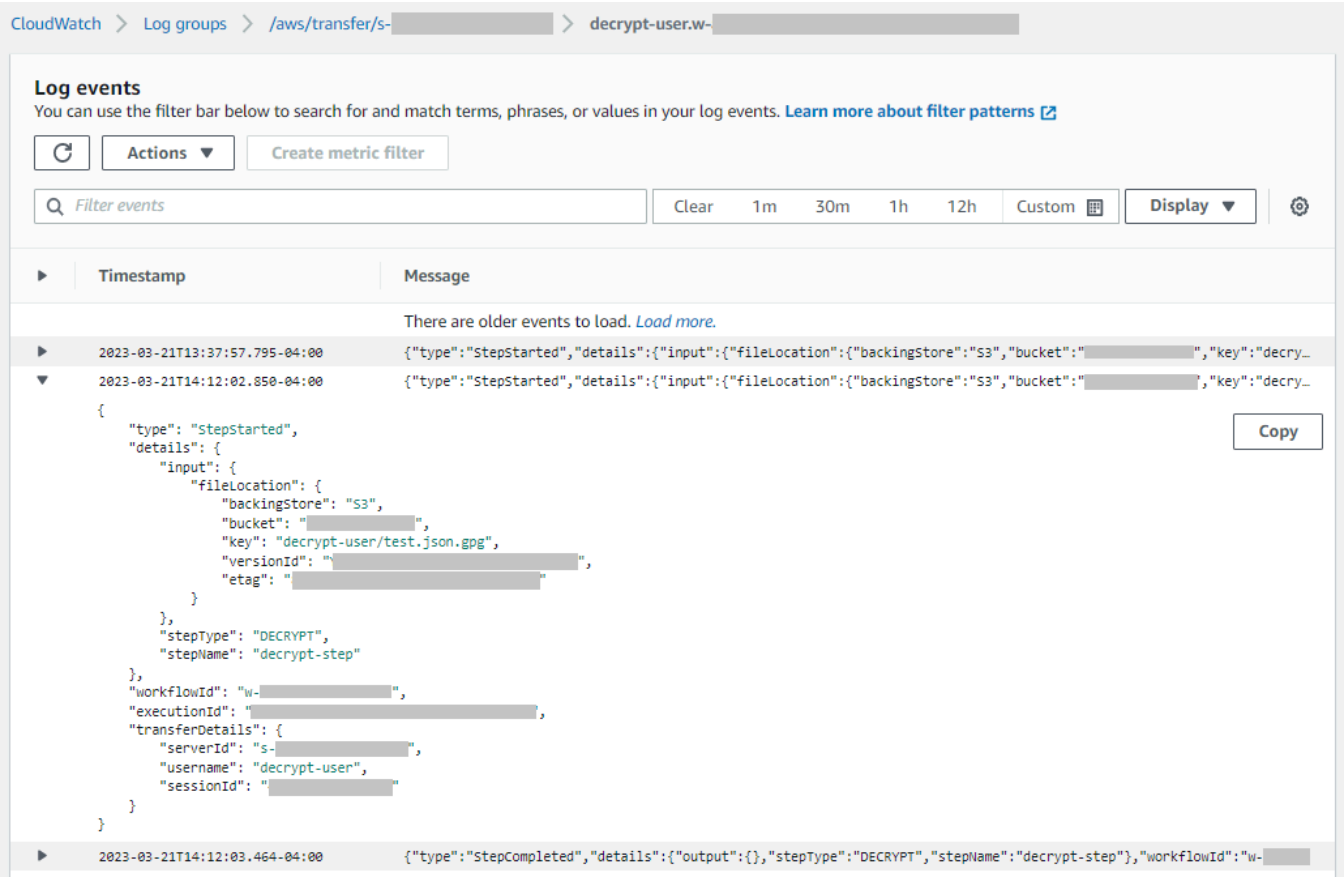


- 他のエントリを選択すると、ログストリームの例が表示されます。



- 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。例えば、「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
```

S3 アクセスログへの Amazon S3 API呼び出しのログ記録

Amazon S3 アクセスログを使用して、ファイル転送ユーザーに代わって行われた S3 リクエストを識別する場合、 RoleSessionName はファイル転送のサービスとして引き受けられたIAM

ロールを表示するために使用されます。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
```

上記のリクエスタフィールドには、 というIAMロールが表示されます`IamRoleName`。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、各 を置き換えます。*`user input placeholder`* 独自の情報。
> 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
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	/user-home-bucket/test

フィールド	説明	エントリ例
	have a logical home directory, this value is always /	
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
user	The end user's username	myname192

フィールド	説明	エントリ例
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

- 転送セッションのログエントリの例
- 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

転送セッションのログエントリの例

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反映しています。まず、クライアントはバケットへの接続を終了し、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/aws/transfer/*connector-id*されます。*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コマンドを実行したとき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"
}
```

このログエントリは、タイムアウトした転送のため、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"
}
```

前のログ例のいくつかのキーフィールドの説明。

- timestamp は、ログが に追加されるタイミングを表start-timeし CloudWatch、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-transferAPI呼び出しで複数のファイルパスを渡すと、すべてのファイルは同じ 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。の初期部分は 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。これらは、構造化ERRORSログの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-99999999999999999",
    "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

次の表は、Transfer Family の CloudWatch メトリクスを示しています。

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。

名前空間	メトリクス	説明
		AS2 プロトコルを使用するサーバーの場合、このメトリクス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	取引先から正常に受信したAS2メッセージの合計数。 単位: カウント 期間: 5 分
	InboundFailedMessage	取引先から受信できなかったAS2メッセージの合計数。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
	OnPartialUploadExecutionsStarted	サーバーで開始されたワークフロー実行の合計数 on-partial-upload。 単位: カウント 期間 = 1 分

名前空間	メトリクス	説明
	OnPartialUploadExecutionsSuccess	サーバーで成功 on-partial-uploadしたワークフロー実行の合計数。 単位: カウント 期間 = 1 分
	OnPartialUploadExecutionsFailed	サーバーで失敗した on-partial-uploadワークフロー実行の合計数。 単位: カウント 期間 = 1 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。

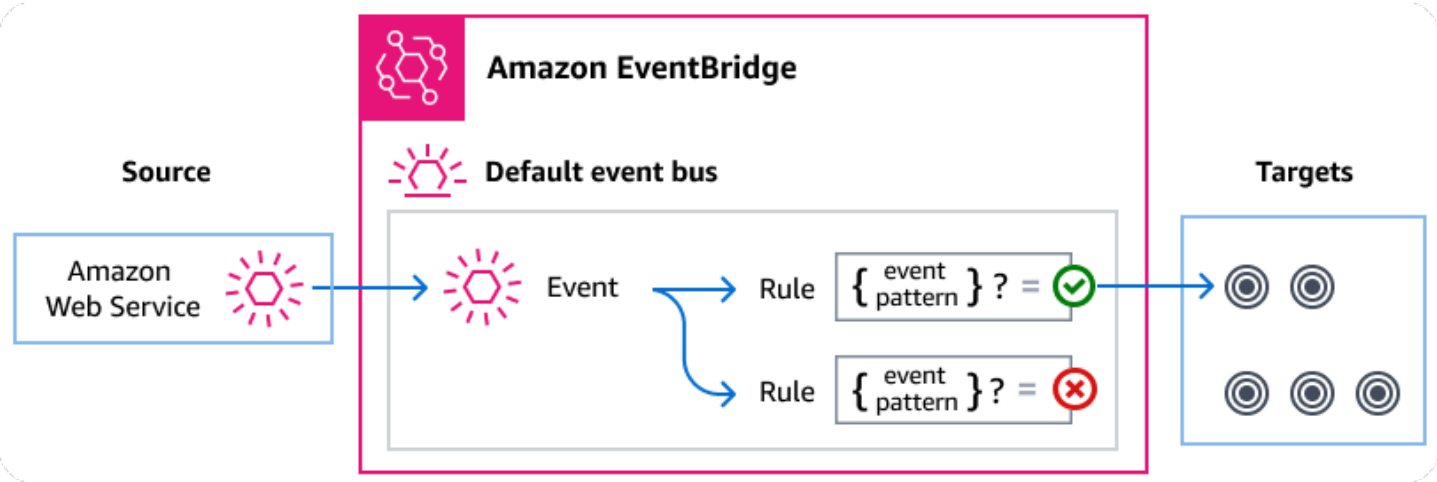
イベントの通知は、E メール、チャットアプリケーションでの Amazon Q Developer チャット通知、AWS Console Mobile Application プッシュ通知など、複数のチャネルを通じて受け取ることができます。コンソール通知センター で通知を確認することもできます。 User Notifications は集約をサポートしているため、特定のイベント中に受信する通知の数を減らすことができます。

詳細については、AWS User Notifications 「 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、および「 とは AWS User Notifications」を参照してください。

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、イベントを発信して応答することで連携する、疎結合された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。イベントが一致した場合、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
- 追加 EventBridge リソース
- Transfer Family イベントの詳細リファレンス

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序どおりに配信されない場合があります。

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」を参照してください。 Amazon EventBridge

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 プロトコルの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 プロトコルのファイル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 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 プロトコルの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のアップロードに失敗しました。

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 プロトコルの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 プロトコルのファイルのアップロードに失敗しました。

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	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に失敗しました。

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を受信しました。
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。

イベントの詳細のタイプ	説明
AS2 ペイロード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	メッセージのメッセージ処理通知を受信AS2しました。
AS2 MDN 受信失敗	メッセージのメッセージ処理通知が受信AS2されていません。
AS2 MDN 送信完了	メッセージのメッセージ処理通知AS2が正常に送信されました。
AS2 MDN 送信失敗	メッセージのメッセージ処理通知の送信AS2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がターゲットに Transfer Family イベントを送信する場合は、目的の Transfer Family イベントのデータに一致する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、次の一般的なステップを実行します。

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 EventBridge を配信するターゲットを指定します。

ターゲットは、他の AWS 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

がイベントをデフォルトのイベントバスに Transfer Family 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かどうかを決定します。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、 のすべてのイベントと一致します Transfer Family。

```
{
  "source": ["aws.transfer"]
}
```

次のイベントパターンの例は、すべてのSFTPコネクタイベントと一致します。

```
{
  "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例は、Transfer Family で失敗したすべてのイベントと一致します。

```
{
```

```
  "source": ["aws.transfer"],
  "detail-type": [{"wildcard", "*Failed"}]
}
```

次のイベントパターンの例は、ユーザーの正常なSFTPダウンロードと一致します *username*:

```
{
  "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、EventBridge 「 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」を参照してください。

Amazon EventBridge アクセス許可

Transfer Family は、イベントを に配信するための追加のアクセス許可を必要としません Amazon EventBridge。

指定したターゲットには、特定のアクセス許可または設定が必要な場合があります。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。

追加 EventBridge リソース

を使用してイベント EventBridge 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 EventBridge を照合するときに使用する の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 イベントの詳細リファレンス

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フィールドの共通セットがあります。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を含めることができます。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。以下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` と指定します。
- `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、「 ユーザーガイド」のAmazon EventBridge 「イベント」を参照してください。 Amazon EventBridge

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

以下は、SFTP、、FTPSおよび FTPサーバーイベントの詳細フィールドです。

- 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、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。

```
{
```

```
  . . .,
  "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 `aws.transfer`です。

`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`

転送に関与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 ファイルでのみ使用されます) 。

`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たときのタイムスタンプ。

Example SFTP サーバーファイルのダウンロードに失敗した例のイベント

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した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の例イベント

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

`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、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。


```
{
  . . .,
  "detail-type": "string",
  "source": "aws.transfer",
  . . .,
  "detail": {
```

```
    "operation" : "string",
    "connector-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"
    "bytes" : "number",
    "local-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は、このフィールドの内容を決定します。

このイベントでは、データには以下が含まれます。

`operation`

StartFileTransfer リクエストがファイルを送信または取得しているかどうか。値: SEND|RETRIEVE。

`connector-id`

使用されているSFTPコネクタの一意の識別子。

`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AS2またはSFTPエンドポイントURLの 。

`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たときのタイムスタンプ。

`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、ユーザーがStartFileTransferリクエストに
渡RemoteDirectoryPath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、このフィールドは空です。

`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 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 コネクタファイルの送信に失敗した例のイベント

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例イベント

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した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"
        }
    }
}
```

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 送信完了
- 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、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。

```
{
   . . .,
  "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`

転送するファイルの 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 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 または FAILED転送の場合、JSONファイルのコンテナ。

`json-key`

COMPLETED 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、メッセージ処理のテキスト説明。

`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`

メッセージを送信するAS2取引先。

`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取引先。

`connector-id`

Transfer Family サーバーから取引先に送信される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 (取引先から Transfer Family サーバーへの転送) の場合、転送に関与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 サーバー専用の一意の 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たときのタイムスタンプ。

`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 失敗したサンプルイベントの受信

```
{
  "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 、最もセキュリティに敏感な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の恩恵を受けることができます。

セキュリティは、 AWS とユーザーの間で責任を共有します。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、AWS のサービス「コンプライアンスプログラムによるスコープ」の」の「」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、「」の 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 のサービス は、データの機密性、会社のコンプライアンス目的、および適用される法律と規制によって決まります。 は、コンプライアンスに役立つ以下のリソース AWS を提供します。

- セキュリティとコンプライアンスのクイックスタートガイド – これらのデプロイガイドでは、アーキテクチャ上の考慮事項について説明し、セキュリティとコンプライアンスに重点を置いたベースライン環境 AWS を にデプロイする手順について説明します。
- Amazon Web Services HIPAA の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**
  >
  > すべての AWS のサービス がHIPAA対象となるわけではありません。詳細については、HIPAA「対象サービスリファレンス」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、お客様の業界とロケーション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 ()、国際標準化機構

(ISO) などPCI) 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。

- AWS Config デベロッパーガイドの ルールによるリソースの評価 – この AWS Config サービスは、リソース設定が内部プラクティス、業界ガイドライン、および規制にどの程度準拠しているかを評価します。
- AWS 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。
- 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、リスクと規制や業界標準へのコンプライアンスの管理を簡素化できます。

このドキュメントは、 を使用する際に責任共有モデルを適用する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、セキュリティとコンプライアンスの目標を達成 AWS Transfer Family するために 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と保護に役立つ他の AWS サービスの使用方法についても説明します。

トピック

- のセキュリティポリシー AWS Transfer Family
- 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
- のアイデンティティとアクセスの管理 AWS Transfer Family
- のコンプライアンス検証 AWS Transfer Family
- でのレジリエンス 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

のセキュリティポリシー AWS Transfer Family

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。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が心配な場合は、変更される可能性のあるデフォルトポリシーを使用するのではなく、サーバーを作成または更新するとき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
- TransferSecurityPolicy2024 年 1 月
- TransferSecurityPolicy2023 年 5 月
- TransferSecurityPolicy2022 年 3 月
- TransferSecurityPolicy2020 年 6 月
- TransferSecurityPolicy2018 年 11 月
- TransferSecurityPolicy-FIPS-2024 年 1 月
- TransferSecurityPolicy-FIPS-2023 年 5 月
- TransferSecurityPolicy-FIPS-2020 年 6 月
- 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 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CLI。

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-sha-256
- rsa-sha-512
- ecdsa-sha2-nistp256
- ecdsa-sha2-nistp384
- ecdsa-sha2-nistp512
- ssh-ed25519

さらに、2018 年と 2020 年のセキュリティポリシーでは が許可されていますssh-rsa。

> **Note**
>
> 常にキータイプであるRSAキータイプssh-rsaと、サポートされている任意のアルゴリズムである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セクションでのみアルゴリズムを使用します。
> - 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。

セキュリティポリシー	2024-01	2023-05	2022-03	2020-06	FIPS2024年1月	FIPS2023年5月	FIPS2020年6月	2018-11
SshCiphers								
aes128-ctr	♦			♦	♦		♦	♦
aes128-gcm@openssh.com	♦	♦	♦	♦	♦	♦	♦	♦
aes192-ctr	♦	♦	♦	♦	♦	♦	♦	♦
aes256-ctr	♦	♦	♦	♦	♦	♦	♦	♦
aes256-gcm@openssh.com	♦	♦	♦	♦	♦	♦	♦	♦
chacha20-poly1305@openssh.com				♦				♦
SshKexs								
curve25519-sha256	♦	♦	♦					♦
curve25519-	♦	♦	♦					♦

セキュリティポリシー	2024-01	2023-05	2022-03	2020-06	FIPS2024年1月	FIPS2023年5月	FIPS2020年6月	2018-11
sha256@libssh.org								
diffie-hellman-group14-sha1								♦
diffie-hellman-group14-sha256				♦			♦	♦
diffie-hellman-group16-sha512	♦	♦	♦	♦	♦	♦	♦	♦
diffie-hellman-group18-sha512	♦	♦	♦	♦	♦	♦	♦	♦

セキュリティポリシー	2024-01	2023-05	2022-03	2020-06	FIPS2024 年 1 月	FIPS2023 年 5 月	FIPS2020 年 6 月	2018-11
diffie-hellman-group-exchange-sha256		♦	♦	♦		♦	♦	♦
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	FIPS2024年1月	FIPS2023年5月	FIPS2020年6月	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	♦							

SshMacs

セキュリティポリシー	2024-01	2023-05	2022-03	2020-06	FIPS2024年1月	FIPS2023年5月	FIPS2020年6月	2018-11
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	♦	♦	♦	♦	♦	♦	♦	♦
umac-128-etm@openssh.com				♦				♦

セキュリティポリシー	2024-01	2023-05	2022-03	2020-06	FIPS2024 年 1 月	FIPS2023 年 5 月	FIPS2020 年 6 月	2018-11
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	♦	♦	♦	♦	♦	♦	♦	♦
TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384	♦	♦	♦	♦	♦	♦	♦	♦

セキュリティポリシー	2024-01	2023-05	2022-03	2020-06	FIPS2024年1月	FIPS2023年5月	FIPS2020年6月	2018-11
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	♦	♦	♦	♦	♦	♦	♦	♦
TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384	♦	♦	♦	♦	♦	♦	♦	♦
TLS_RSA_WITHAES_128_CBC_SHA256								♦

セキュリティポリシー	2024-01	2023-05	2022-03	2020-06	FIPS2024年1月	FIPS2023年5月	FIPS2020年6月	2018-11
TLS_RSA_WITHAES_256_CBC_SHA256								◆

TransferSecurityPolicy2024 年 1 月

2024 TransferSecurityPolicy年 1 月のセキュリティポリシーを以下に示します。

```
{
    "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"
        ]
    }
}
```

TransferSecurityPolicy2023 年 5 月

以下は、 TransferSecurityPolicy-2023-05 セキュリティポリシーを示しています。

```
{
    "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"
        ]
    }
}
```

TransferSecurityPolicy2022 年 3 月

以下は、 TransferSecurityPolicy-2022-03 セキュリティポリシーを示しています。

```
{
  "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"
    ]
  }
}
```

TransferSecurityPolicy2020 年 6 月

TransferSecurityPolicy2020 年 6 月のセキュリティポリシーを以下に示します。

```
{
  "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"
    ]
  }
}
```

TransferSecurityPolicy2018 年 11 月

2018 TransferSecurityPolicy年 11 月のセキュリティポリシーを以下に示します。

```
{
  "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 年 1 月

以下は、 TransferSecurityPolicy-FIPS-2024-01 セキュリティポリシーを示しています。

> **Note**
>
> FIPS サービスエンドポイントと TransferSecurityPolicy-FIPS-2024-01 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-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 年 5 月

のFIPS証明書の詳細は AWS Transfer Family 、 で確認できます。 https://csrc.nist.gov/projects/cryptographic-module-validation-program/validated-modules/search/all

以下は、 TransferSecurityPolicy-FIPS-2023-05 セキュリティポリシーを示しています。

> **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 年 6 月

のFIPS証明書の詳細は AWS Transfer Family 、 で確認できます。 https://csrc.nist.gov/projects/cryptographic-module-validation-program/validated-modules/search/all

以下は、 TransferSecurityPolicy-FIPS-2020-06 セキュリティポリシーを示しています。

> **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-SSHFIPS-Experimental-2023-04 月
SSH ciphers		
aes128-ctr		♦
aes128-gcm@openssh.com	♦	♦
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	♦	♦

セキュリティポリシー	TransferSecurityPolicy-PQ-SSH-Experimental-2023-04	TransferSecurityPolicy-PQ-SSHFIPS-Experimental-2023-04 月
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		♦
curve25519-sha256@libssh.org	♦	
curve25519-sha256	♦	
MACs		
hmac-sha2-256-etm@openssh.com	♦	♦
hmac-sha2-256	♦	♦
hmac-sha2-512-etm@openssh.com	♦	♦

セキュリティポリシー	TransferSecurityPolicy-PQ-SSH-Experimental-2023-04	TransferSecurityPolicy-PQ-SSHFIPS-Experimental-2023-04 月
hmac-sha2-512	♦	♦
TLS ciphers		
TLS_ECDHEECDSA_WITH_AES_128_CBC_SHA256	♦	♦
TLS_ECDHEECDSA_WITH_AES_128_GCM_SHA256	♦	♦
TLS_ECDHEECDSA_WITH_AES_256_CBC_SHA384	♦	♦
TLS_ECDHEECDSA_WITH_AES_256_GCM_SHA384	♦	♦
TLS_ECDHERSA_WITH_AES_128_CBC_SHA256	♦	♦
TLS_ECDHERSA_WITH_AES_128_GCM_SHA256	♦	♦
TLS_ECDHERSA_WITH_AES_256_CBC_SHA384	♦	♦
TLS_ECDHERSA_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-SSHFIPS-Experimental-2023-04 月

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。量子後キーの確立は、store-now-harvest-laterネットワークトラフィックを記録し、攻撃と呼ばれる量子コンピュータによって将来的に復号するために保存できるため、必要です。

このオプションは、Transfer Family に接続して、Amazon Simple Storage Service (Amazon S3) ストレージまたは Amazon Elastic File System (Amazon) との間で安全なファイル転送を行う場合に使用できますEFS。の量子後ハイブリッドキーの確立には、古典的なキー交換アルゴリズムと組み合わせて使用する量子後キー確立メカニズムSSHが導入されています。SSH クラシック暗号スイートで作成されたキーは、現在のテクノロジーによる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 は量子耐性または量子後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

- でのポスト量子ハイブリッドキー交換について SSH
- TransferFamilyにおけるポスト量子ハイブリッド鍵確立の仕組み
  - Kyber を使用する理由
  - 量子後のハイブリッドSSHキー交換と暗号化の要件 (FIPS 140)
- 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

- 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する
- ポスト量子ハイブリッドキー交換をサポートするSFTPクライアントを設定する
- で量子後のハイブリッドキー交換を確認する SFTP

でのポスト量子ハイブリッドキー交換について SSH

Transfer Family は、古典的な楕円曲線 Diffie-Hellman (ECDH) キー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、国立標準技術研究所 (NIST) が最初の標準ポスト量子キー契約アルゴリズムとして指定したポスト量子パブリックキー暗号化およびキー確立アルゴリズムです。

クライアントとサーバーは引き続きECDHキー交換を行います。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量子KEMパブリックキーにカプセル化し、クライアントのSSHキー交換メッセージにアドバタイズされます。この戦略は、古典的なキー交換の高い保証と提案された量子後キー交換のセキュリティを組み合わせて、 ECDHまたは量子後共有シークレットを壊すことができない限り、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 business-to-business、 SFTP および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ファイル転送を安全にスケーリングします。SFTP は、 経由で実行され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SSH。Transfer Family のポスト量子キー交換サポートにより、 を介したデータ転送のセキュリティバーが引き上げられますSFTP。

Transfer Family でのポスト量子ハイブリッドキー交換SFTPのサポートにはKyber-512, Kyber-768、および Kyber-1024 と ECDH P256, P384, P521、または Curve25519 曲線の組み合わせが含まれます。以下の対応するSSHキー交換方法は、ポスト量子ハイブリッドSSHキー交換ドラフト で指定されています。

- `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**
>
> これらの新しいキー交換方法は、ドラフトが標準化に向けて進化するにつれて、または が Kyber アルゴリズムをNIST批准するときに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、Post-Quantum NIST Cryptography プロジェクト によって標準化のために選択された最初のポスト量子暗号化アルゴリズムです。一部の標準化団体は、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TLSで で 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 IETFの P256 のようなNIST承認済み曲線を組み合わせた量子後暗号化の提案案を に送信しましたSSH。お客様のセキュリティを強化するために、 および での量子後キー交換 AWS を実装SFTPし、そのドラフトSSHに従います。提案が によって採用IETF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。

新しいキー交換方法 (セクション に記載TransferFamilyにおけるポスト量子ハイブリッド鍵確立の仕組み) は、ドラフトが標準化に向けて進化したり、 が Kyber NIST アルゴリズムを批准したりすると変更される可能性があります。

> **Note**
>
> 量子後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の使用」を参照) AWS Certificate Manager、および AWS Secrets Manager APIエンドポイントTLSの で量子後ハイブリッドキー交換に使用できます。 TLS AWS KMS

量子後のハイブリッドSSHキー交換と暗号化の要件 (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 140 認定のオープンソース暗号化ライブラリである AWS-LC SSHを使用して AWS FIPS、 FIPSで承認済み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量子後の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 Agence nationale de la sécurité des systèmes d'information (ANSSI) も、このような量子後のハイブリッドキー交換方法を推奨しています。

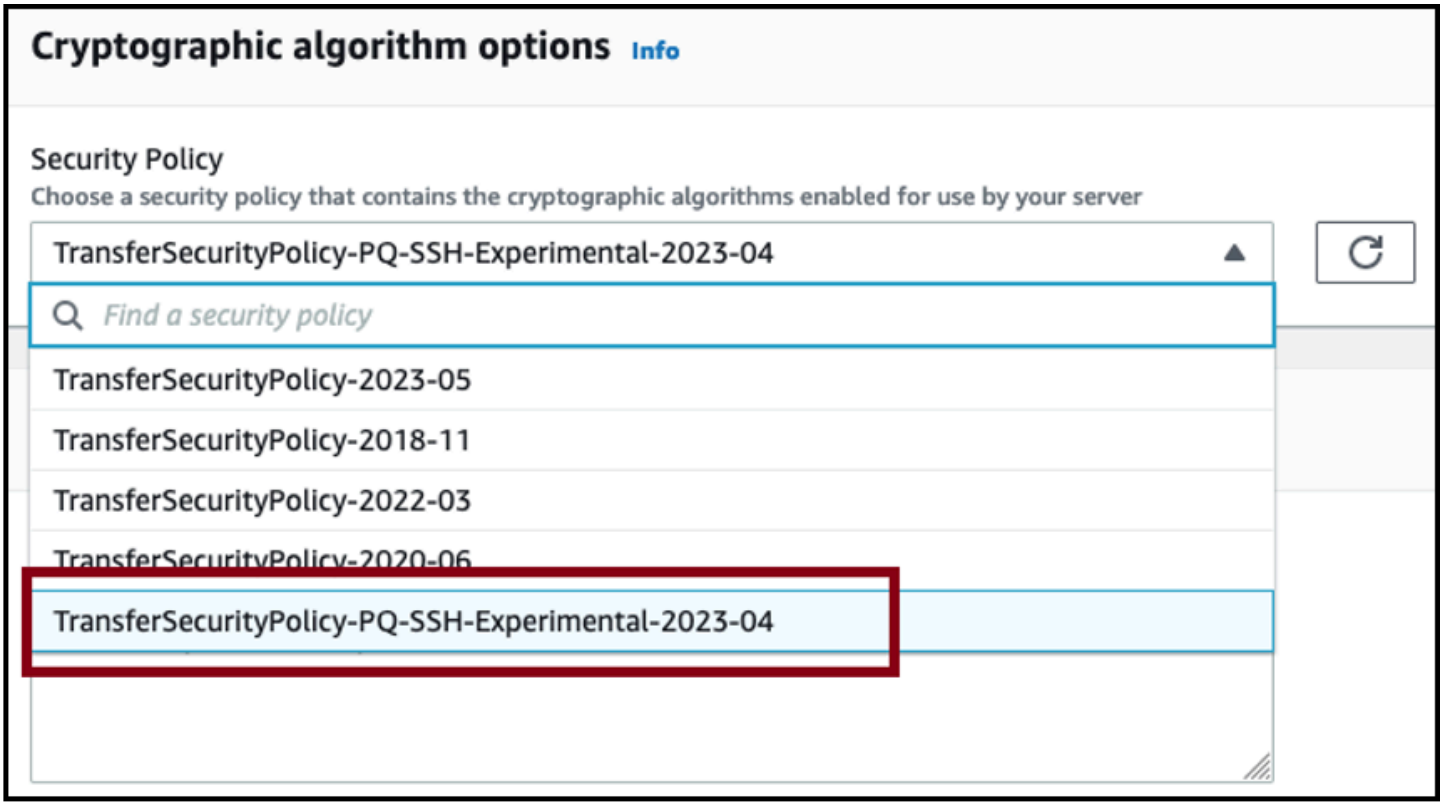
Transfer Family におけるポスト量子ハイブリッドキー交換のテスト

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する.
2. 上記のドラフト仕様のガイダンスに従って、量子後ハイブリッドキー交換をサポートするSFTPクライアント (などポスト量子ハイブリッドキー交換をサポートするSFTPクライアントを設定する) を使用します。
3. Transfer Family サーバーを使用してファイルを転送します。
4. で量子後のハイブリッドキー交換を確認する SFTP.

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する

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エンドポイントで暗号化アルゴリズムオプションを編集することで、SSHポリシーを選択できます。次のスナップショットは、SSHポリシーを更新する AWS Management Console の例を示しています。



量子後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-SSHFIPS-Experimental-2023-04 で

す。Transfer Familyポリシーの詳細については、のセキュリティポリシー AWS Transfer Family 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クライアントを設定する

SFTP Transfer Family エンドポイントで正しいポスト量子SSHポリシーを選択したら、Transfer Family SFTPでポスト量子を試すことができます。上記のドラフト仕様のガイダンスに従うことで、量子後のハイブリッドキー交換をサポートするSFTPクライアント (OQSOpen SSHなど) i を使用できます。

OQS OpenSSH は、 SSHを使用して量子安全暗号化を追加する OpenSSH のオープンソースフォークですliboqs。 liboqsは、量子耐性の暗号化アルゴリズムを実装するオープンソース C ライブラリです。OQS OpenSSH および liboqsは Open Quantum Safe (OQS) プロジェクトの一部です。

Transfer Family SFTP with OQS Open でポスト量子ハイブリッドキー交換をテストするにはSSH、プロジェクトの で説明されているように OQS OpenSSH を構築する必要がありますREADME。OQS Open を構築したらSSH、次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メソッドを使用して、サンプルSFTPクライアントを実行してSFTPエンドポイント (など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：

- 置換 *username_private_key_PEM_file* SFTP ユーザーのプライベートキーPEMでエンコードされたファイルを使用する
- 置換 *username* SFTP ユーザー名
- 置換 *server-id* Transfer Family サーバー ID を使用する
- 置換 *region-id* Transfer Family サーバーがある実際のリージョン

で量子後のハイブリッドキー交換を確認する SFTP

Transfer Family SFTPへの SSHの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、出力は次のようになります (簡潔にするために無関係な情報は省略します)。

```
$./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>
```

出力は、ポスト量子ハイブリッド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発生し、SFTPセッションが正常に確立されたことを示します。

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FAQ」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」のAWS 「責任共有モデル」とGDPR「ブログ記事」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 () 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IAM。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。1.2 TLS をサポートしています。
- で APIとユーザーアクティビティのログ記録を設定します AWS CloudTrail。
- 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イスまたは を介して にアクセスする AWS 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API、FIPSエンドポイントを使用します。利用可能なFIPSエンドポイントの詳細については、連邦情報処理標準 (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、[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。これには、コンソール、、または API AWS CLIを使用して Transfer Family やその他の AWS サービスを使用する場合も含まれます 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、診断ログの内容として取得される可能性があります。URL を外部サーバーに提供するときは、そのサーバーへのリクエストを検証URLするために認証情報を に含めないでください。

これとは対照的に、Transfer Family サーバーへのアップロードおよびダウンロードオペレーションからのデータは完全にプライベートとして扱われ、 SFTP や FTPS 接続などの暗号化されたチャネルの外部には存在しません。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
- キーの管理

Amazon S3 でのデータ暗号化

AWS Transfer Family は、Amazon S3 バケット用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（ AWS KMS) マネージドキー (SSE-) でサーバー側の暗号化を使用して暗号化されますKMS。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。

次の手順では、 でデータを暗号化する方法を示します 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. ユーザーに添付されている 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。このポリシーの例を、ユーザーIAMロールポリシーとセッションポリシーの両方に含めます。

```
{
 "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 でデフォルトの暗号化で指定されたキー ID と同じである必要があります。
> ルート、またはユーザーに使用されるIAMロールは、キーポリシーで AWS KMS 許可する必要があります。 AWS KMS キーポリシーの詳細については、「 AWS Key Management Service デベロッパーガイド」の「 でキーポリシーを使用する AWS KMS」を参照してください。

キーの管理

このセクションでは、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確認できます。 AWS Lambda で Transfer Family を使用してキー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 を使用したユーザー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、および ED25519キーを受け入れます。

このセクションでは、Pretty Good Privacy (PGP) キーを生成および管理する方法も説明します。

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
- サービスマネージドユーザーのSSHキーを生成する

- 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**
>
> ワークフローのPGP復号に使用するアルゴリズムについては、PGP「キーペア でサポートされているアルゴリズム」を参照してください。

- の場合ED25519: `ssh-ed25519`
- の場合RSA：
  - `rsa-sha2-256`
  - `rsa-sha2-512`
- の場合ECDSA：
  - `ecdsa-sha2-nistp256`
  - `ecdsa-sha2-nistp384`
  - `ecdsa-sha2-nistp521`

> **Note**
>
> 古いセキュリティポリシーSHA1では`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。

サービスマネージドユーザーのSSHキーを生成する

サービスマネージド認証方法を使用してユーザーを認証するようにサーバーを設定できます。この方法では、ユーザー名とSSHキーがサービス内に保存されます。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。各ユーザーは、個々のサーバーで複数のパブリックSSHキーをファイルに保存できます。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 でのSSHキーの作成
- Microsoft Windows でのSSHキーの作成
- SSH2 パブリックキーをPEM形式に変換する

macOS 、Linux、または Unix でのSSHキーの作成

macOS 、Linux、または Unix 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SSHパブリックキーとSSHプライベートキーを作成し、キーペアとも呼ばれます。

macOS 、Linux、または Unix 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-、および ED25519形式の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。

- RSA 4096 ビットキーペアを生成するには：

```
ssh-keygen -t rsa -b 4096 -f key_name
```

- ECDSA 521 ビットキーペアを生成するには (ECDSA のビットサイズは 256、384、521 です) 。

```
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。

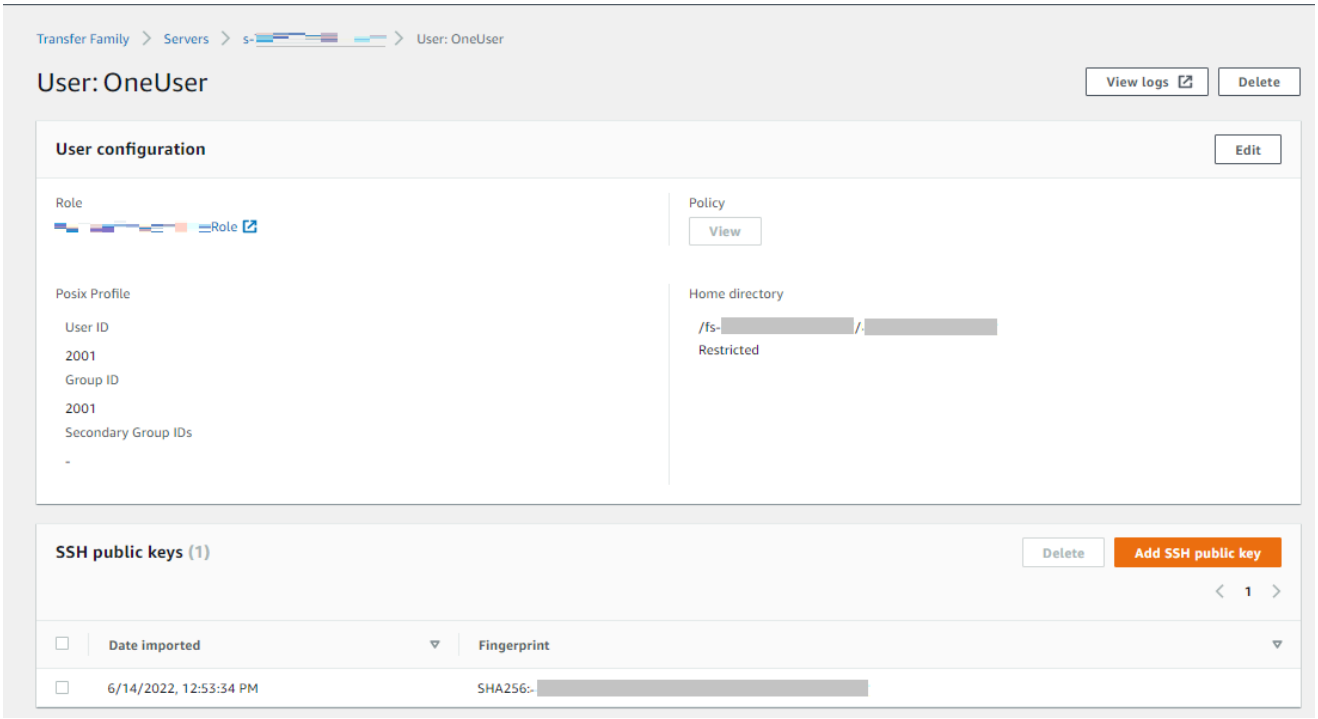
3. *key_name*.pub.ファイルを見つけて開きます。
4. テキストをコピーし、サービスマネージドユーザーのSSHパブリックキーに貼り付けます。

   a. で AWS Transfer Family コンソールを開きhttps://console.aws.amazon.com/transfer/、ナビゲーションペインからサーバーを選択します。

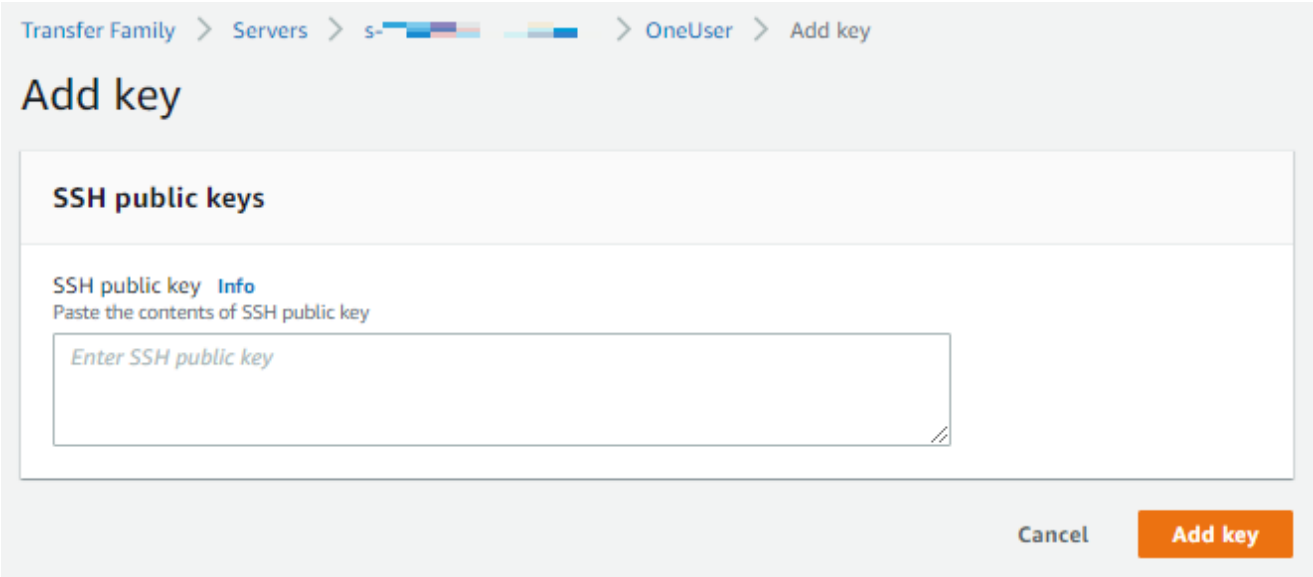
   b. [サーバー] ページで、更新するユーザーを含むサーバーの [サーバー ID] を選択します。

   c. パブリックキーを追加するユーザーを選択します。

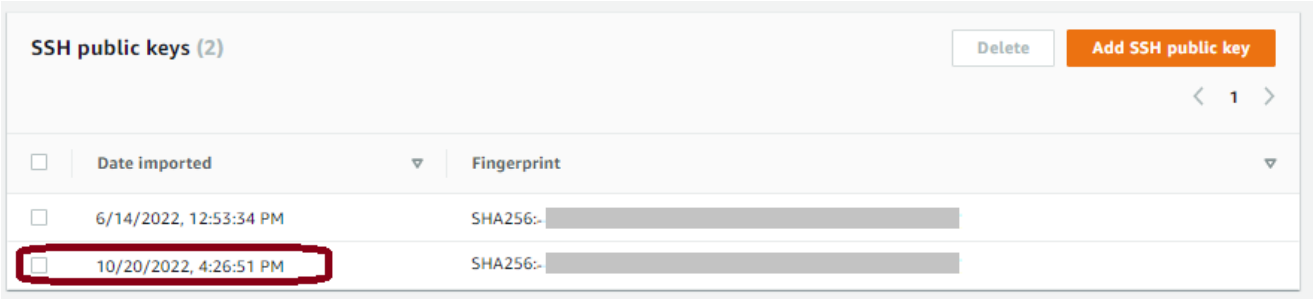
   d. SSH パブリックキーペインで、SSHパブリックキーの追加 を選択します。



e. 生成したパブリックキーのテキストをSSHパブリックキーテキストボックスに貼り付け、キーの追加 を選択します。



新しいキーはSSHパブリックキーペインに表示されます。



Microsoft Windows でのSSHキーの作成

Windows では、若干異なるSSHキーペア形式が使用されます。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、P uTTYgen を使用して適切な形式でSSHキーペアを作成できます。P を使用して、 を使用して生成されたプライベートキーssh-keygenを .ppk ファイルuTTYgen に変換することもできます。

> **Note**
>
> プライベートキーファイルを .ppk 形式ではない状態で WinSCP を提示すると、そのクライアントはキーを .ppk形式に変換することを提案します。

Windows uTTYgen で P を使用してSSHキーを作成するチュートリアルについては、SSH.com ウェブサイト を参照してください。

SSH2 パブリックキーをPEM形式に変換する

AWS Transfer Family は、フォーマットPEMされたパブリックキーのみを受け入れます。SSH2 パブリックキーがある場合は、変換する必要があります。SSH2 パブリックキーの形式は次のとおりです。

```
---- BEGIN SSH2 PUBLIC KEY ----
Comment: "rsa-key-20160402"
AAAAB3NzaC1yc2EAAAABJQAAAgEAiL0jjDdFqK/kYThqKt7THrjABTPWvXmB3URI
:
:
---- END SSH2 PUBLIC KEY ----
```

PEM パブリックキーの形式は次のとおりです。

```
ssh-rsa AAAAB3NzaC1yc2EAAAABJQAAA...
```

次のコマンドを実行して、SSH2フォーマットされたパブリックキーをPEMフォーマットされたパブリックキーに変換します。置換 *ssh2-key* SSH2 キーの名前、および *PEM-key* PEM キーの名前。

```
ssh-keygen -i -f ssh2-key.pub > PEM-key.pub
```

SSH キーのローテーション

セキュリティのために、SSHキーをローテーションするベストプラクティスをお勧めします。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よっては、機密性の高い通信では、SSHキーペアを 1 回だけ使用できます。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、SSH認証情報を一定期間保存し、ユーザーに過度の負担をかけない間隔を設定する方がはるかに一般的です。標準的な設定は 3 か月です。

SSH キーローテーションを実行するには、次の 2 つの方法を使用します。

- コンソールで、新しいSSHパブリックキーをアップロードし、既存のSSHパブリックキーを削除できます。
- を使用してAPI、 を使用してユーザーの Secure Shell (SSH) パブリックキーDeleteSshPublicKeyAPIを削除し、 ImportSshPublicKeyAPI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. で AWS Transfer Family コンソールを開きますhttps://console.aws.amazon.com/transfer/。
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. ユーザー 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選択し、アクション を選択し、キーの追加 を選択してキーの追加ページを表示します。

   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します。

5. 新しいSSHパブリックキーを入力し、キーの追加 を選択します。

> **Important**
>
> SSH パブリックキーの形式は、生成したキーのタイプによって異なります。
>
> - RSA キーの場合、形式は ですssh-rsa *string*。
> - ED25519 キーの場合、形式は ですssh-ed25519 *string*。
> - ECDSA キーの場合、キーは、生成したキーのサイズに応じてecdsa-sha2-nistp256ecdsa-sha2-nistp384、ecdsa-sha2-nistp521、、または 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。

ユーザーの詳細ページに戻り、先ほど入力した新しいSSHパブリックキーが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

を使用してキーローテーションを実行するには API

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して、削除する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は、Pretty Good Privacy (PGP) 復号を使用できます。ワークフローステップで復号を使用するには、 PGP キーを指定する必要があります。

AWS ストレージブログには、ファイルの暗号化と復号化、ファイルの暗号化と復号化を PGPと で AWS Transfer Family行う方法を説明する投稿があります。

PGP キーの生成

PGP キーの生成に使用する方法は、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 キージェネレーター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。作成するキーのタイプの入力を求められたら、 RSAまたは を選択しますECC。ただし、 を選択した場合はECC、必ず次のいずれかを選択してください。NIST または BrainPool 楕円曲線。選択しない Curve 25519.

PGPキーペアでサポートされているアルゴリズム

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**
>
> Curve25519 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。

PGP キーの管理

PGP キーを管理するには、 を使用する必要があります 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 に保存する前に、サーバーを既に識別または作成しておく必要があります AWS Secrets Manager。

すべてのユーザーに 1 つのキーとパスフレーズを使用する場合は、PGPキーブロック情報をシークレット名 に保存できます。ここで`aws/transfer/`*`server-id`*`/@pgp-default`、 *`server-id`*は Transfer Family サーバーの ID です。このデフォルト・キーは、*`user-name`* がワークフローを実行するユーザと一致するキーがない場合に使用されます。

または、特定のユーザーのキーを作成できます。この場合、シークレットネームのフォーマットは `aws/transfer/`*`server-id`*`/`*`user-name`* となり、*`user-name`* は Transfer Family サーバーのワークフローを実行しているユーザーと一致します。

> **Note**
>
> Transfer Family サーバーごとに、ユーザーごとに最大 3 つのPGPプライベートキーを保存できます。

復号で使用するPGPキーを設定するには

1. GPG 使用している 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。gpg --gen-key 代わりに を実行する場合は、ECCCurve 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。現在、PGPキーはサポートされていません。

- 2.3.0 **GnuPG**より前のバージョンでは、 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、 で AWS Secrets Manager コンソールを開きますhttps://console.aws.amazon.com/secretsmanager/。

   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。

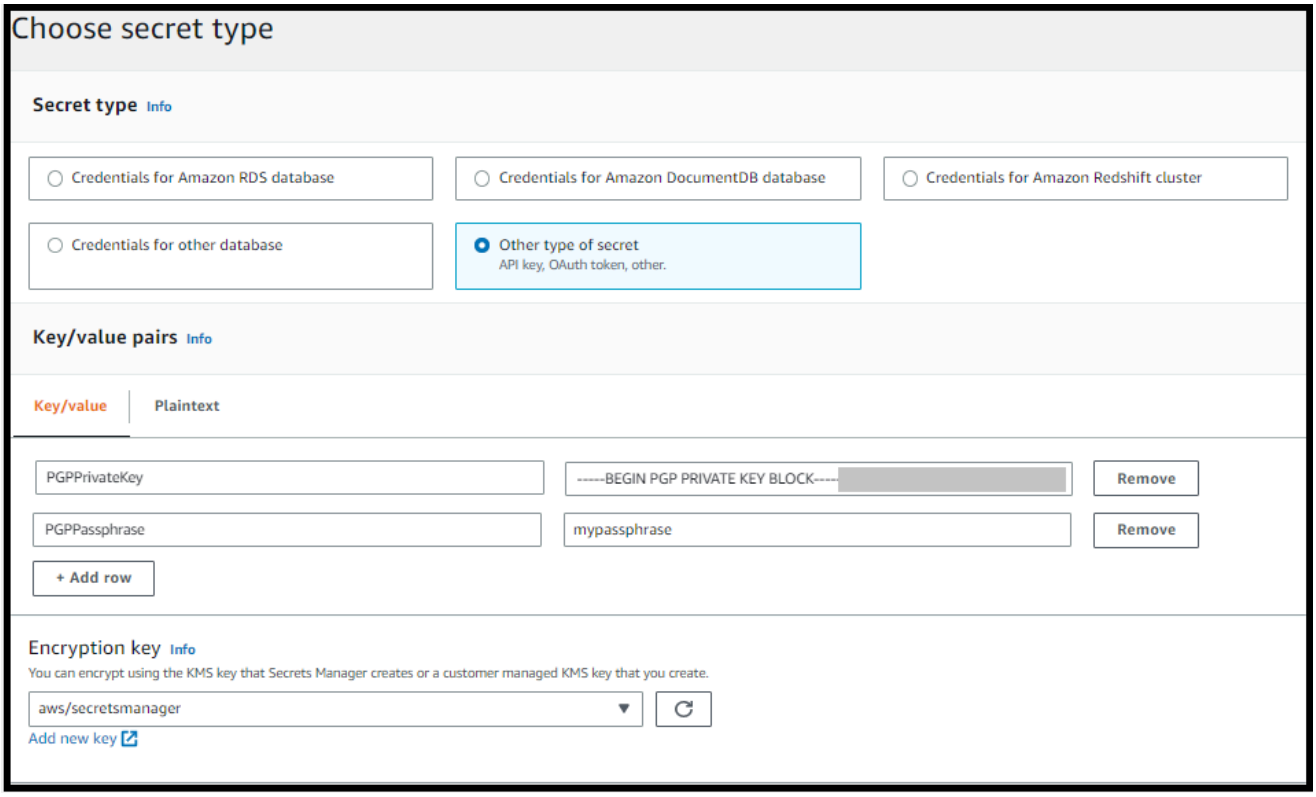
> **Note**
>
> テキストブロックには秘密鍵のみが含まれ、公開鍵も含まれていないことを確認してください。

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。

- value – PGPキーペアの生成時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. [Next (次へ)] を選択します。

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。

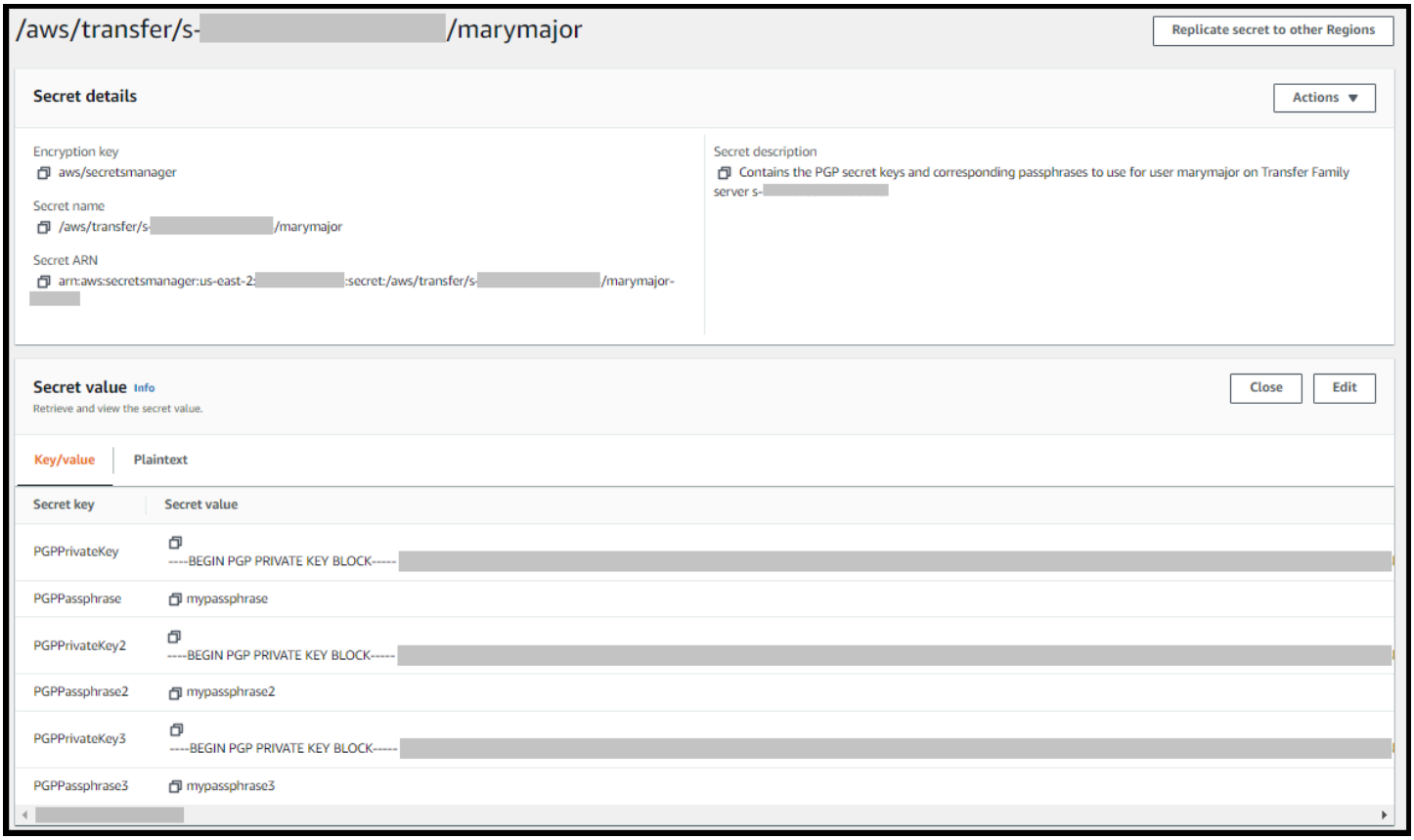
- デフォルトキー、つまり Transfer Family のすべてのユーザーが使用できるキーを作成する場合は、**aws/transfer/*server-id*/@pgp-default** と入力します。*server-id*を、復号ステップを持つワークフローを含むサーバーの ID に置き換えます。
- 特定の Transfer Family ユーザーが使用するキーを作成する場合は、**aws/transfer/*server-id*/*user-name*** と入力します。*server-id* を復号ステップを持つワークフローを含むサーバーの ID に置き換え、*user-name* 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。

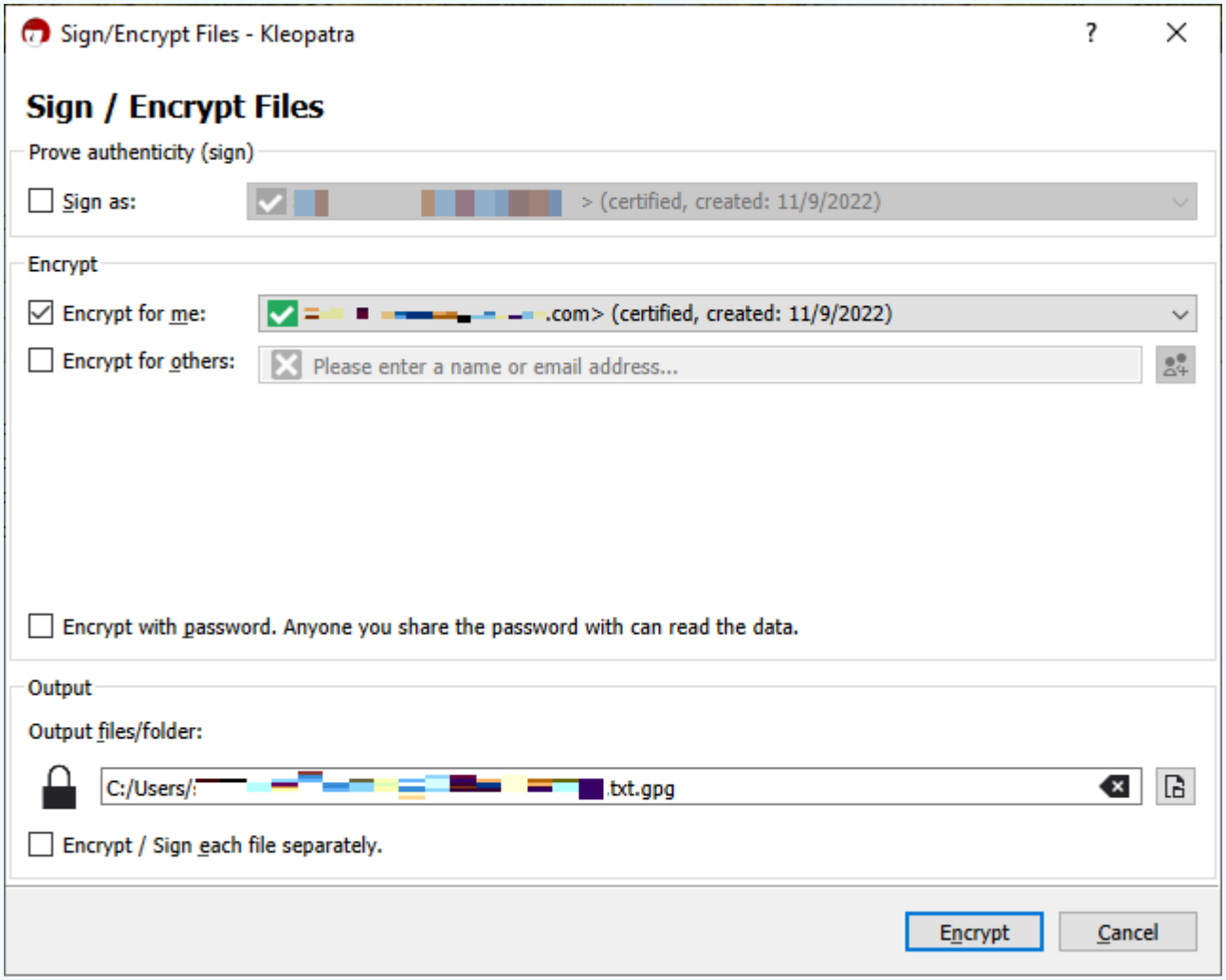


サポートされているPGPクライアント

次のクライアントは Transfer Family 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。



- 「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クライアントも機能する可能性がありますが、ここで言及されているクライアントのみが Transfer Family でテストされていることに注意してください。

のアイデンティティとアクセスの管理 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 リソースの使用を許可する (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
- の AWS Transfer Family 仕組み IAM
- AWS Transfer Family ID 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、管理者は必要な認証情報とアクセス許可を提供します。より多くの AWS Transfer Family 機能を使用して作業を行う際には、追加のアクセス許可が必要になる場合があります。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、 へのフルアクセスがある可能性があります 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必要があるかを判断するのはお客様の仕事です。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アクセス許可を変更する必要があります。このページの情報を確認して、 の基本概念を理解しますIAM。会社IAMで を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の AWS Transfer Family 仕組み IAM。

IAM 管理者 – IAM管理者の場合は、 へのアクセスを管理するためのポリシーの作成方法の詳細を知りたい場合があります AWS Transfer Family。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IAM、「」を参照してくださいAWS Transfer Family ID 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

認証は、アイデンティティ認証情報 AWS を使用して にサインインする方法です。として、IAMユーザーとして AWS アカウントのルートユーザー、またはIAMロールを引き受けることで、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 認証情報は、フェデレーティッド ID の例です。フェデレーティッド ID としてサインインすると、管理者は以前にIAMロールを使用して ID フェデレーションをセットアップしていました。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、 AWS サインイン ユーザーガイドの「 へのサインイン方法 AWS アカウント」を参照してください。

AWS プログラムで にアクセスする場合、 はソフトウェア開発キット (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化して署名します。 AWS ツールを使用しない場合は、自分でリクエストに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「リクエストの署名 AWS API」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、 では、アカウントのセキュリティを高めるために多要素認証 (MFA) を使用する AWS 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「多要素認証の使用」および「 ユーザーガイド」の「多要素認証の使用 (MFA ） AWS」を参照してください。 IAM

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 および リソースへの AWS のサービス 完全なアクセス権を持つ 1 つのサインイン ID 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してアクセスします。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「ルートユーザーの認証情報を必要とするタスク」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、 では、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アクセスするために ID プロバイダーとのフェデレーション AWS のサービス の使用を要求します。

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、ウェブアイデンティティプロバイダー、 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 を使用してアクセスするすべてのユーザーのユーザーです。フェデレーティッド ID が にアクセスすると AWS アカウント、それらはロールを引き受け、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したり、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 ユーザーとは、1 人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 ID 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持つIAMユーザーを作成する代わりに、一時的な認証情報に依存することをお勧めします。ただし、IAMユーザーとの長期的な認証情報を必要とする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ユーザーのコレクションを指定する ID 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、という名前のグループがありIAMAdmins、そのグループにIAMリソースを管理するアクセス許可を付与できます。

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 (ロールではなく) IAM ユーザーを作成するタイミング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 ID です。ユーザーと似ていますがIAM、特定の人物には関連付けられていません。IAM ロール を切り替える AWS Management Console ことで、 でロールを一時的に引き受けることができます。または オペレーションを AWS CLI AWS API呼び出すか、カスタム を使用してロールを引き受けることができますURL。ロールの使用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 一時的な認証情報を持つ ロールは、以下の状況で役立ちます。

- フェデレーションユーザーアクセス – フェデレーティッド ID に許可を割り当てるには、ロールを作成してそのロールの許可を定義します。フェデレーティッド ID が認証されると、その ID はロールに関連付けられ、ロールで定義されている許可が付与されます。フェデレーションのロールの詳細については、 IAM ユーザーガイドの「サードパーティー ID プロバイダーのロールの作成」を参照してください。IAM Identity Center を使用する場合は、アクセス許可セットを設定します。ID が認証された後にアクセスできるものを制御するために、IAMIdentity Center はアクセス許可セットを のロールに関連付けますIAM。アクセス許可セットの詳細については、「AWS IAM Identity Center ユーザーガイド」の「アクセス許可セット」を参照してください。
- 一時的なIAMユーザーアクセス許可 – IAM ユーザーまたはロールは、特定のタスクに対して異なるアクセス許可を一時的に引き受けるIAMロールを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、別のアカウントの誰か (信頼できるプリンシパル) が自分のアカウントのリソースにアクセスすることを許可できます。クロスアカウントアクセスを許可する主な方法は、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、 (プロキシとしてロールを使用する代わりに) リソースに直接ポリシーをアタッチできます。クロスアカウントアクセスの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の違い」を参照してください。
- クロスサービスアクセス — 他の の機能 AWS のサービス を使用するものもあります AWS のサービス。例えば、 サービスで呼び出しを行う場合、そのサービスが Amazon でアプリケーションを実行EC2したりAmazon S3にオブジェクトを保存したりするのが一般的です。サービスでは、呼び出し元プリンシパルの許可、サービスロール、またはサービスリンク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と AWS、プリンシパルと見なされます。一部のサービスを使用する際に、アクションを

実行することで、別のサービスの別のアクションがトリガーされることがあります。FAS は、を呼び出すプリンシパルのアクセス許可と AWS のサービス、ダウンストリームサービス AWS のサービス へのリクエストのリクエストを使用します。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権限が必要です。FAS リクエストを行う際のポリシーの詳細については、「アクセスセッションの転送」を参照してください。

- サービスロール – サービスロールは、ユーザーに代わってアクションを実行するためにサービスが引き受けるIAMロールです。IAM 管理者は、 内からサービスロールを作成、変更、削除できますIAM。詳細については、IAM「 ユーザーガイド」の「 へのアクセス許可を委任するロールの作成 AWS のサービス」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アクセス許可を表示することはできますが、編集することはできません。

- Amazon で実行されているアプリケーション EC2 – 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インスタンス内にアクセスキーを保存するよりも望ましいです。EC2インスタンスに AWS ロールを割り当て、そのすべてのアプリケーションで使用できるようにするには、インスタンスにアタッチされたインスタンスプロファイルを作成します。インスタンスプロファイルには ロールが含まれており、EC2インスタンスで実行されているプログラムが一時的な認証情報を取得できるようにします。詳細については、IAM「 ユーザーガイド」のIAM「ロールを使用して Amazon EC2インスタンスで実行されているアプリケーションにアクセス許可を付与する」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「 (ユーザーではなく) IAMロールを作成するタイミング」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 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、「 ユーザーガイド」のJSON「ポリシーの概要」を参照してください。 IAM

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

デフォルトでは、ユーザーやロールに権限はありません。必要なリソースに対してアクションを実行するアクセス許可をユーザーに付与するには、IAM管理者はIAMポリシーを作成できます。その後、管理者はIAMポリシーをロールに追加し、ユーザーはロールを引き受けることができます。

IAM ポリシーは、 オペレーションの実行に使用する方法に関係なく、アクションのアクセス許可を定義します。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 からロール情報を取得できますAPI。

アイデンティティベースのポリシー

ID ベースのポリシーは、IAMユーザー、ユーザーのグループ、ロールなどの ID にアタッチできるJSONアクセス許可ポリシードキュメントです。これらのポリシーは、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。ID ベースのポリシーを作成する方法については、IAM「 ユーザーガイド」のIAM「ポリシーの作成」を参照してください。

アイデンティティベースのポリシーは、さらにインラインポリシーまたはマネージドポリシーに分類できます。インラインポリシーは、単一のユーザー、グループ、またはロールに直接埋め込まれています。マネージドポリシーは、 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選択」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、リソースにアタッチする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の例としては、IAMロール信頼ポリシー と Amazon S3 バケットポリシー があります。リソースベースのポリシーをサポートするサービスでは、サービス管理者はポリシーを使用して特定のリソースへのアクセスをコントロールできます。ポリシーがアタッチされているリソースの場合、指定されたプリンシパルがそのリソースに対して実行できるアクションと条件は、ポリシーによって定義されます。リソースベースのポリシーでは、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IAMでは、 から AWS 管理ポリシーを使用することはできません。

アクセスコントロールリスト (ACLs)

アクセスコントロールリスト (ACLs) 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アクセス許可を持っているかを制御します。ACLs 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ていません。

Amazon S3、および Amazon VPCは AWS WAF、 をサポートするサービスの例ですACLs。の詳細についてはACLs、「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) 概要」を参照してください。

その他のポリシータイプ

AWS は、追加の低頻度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がIAMエンティティ (IAMユーザーまたはロール) に付与できる最大アクセス許可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。アクセス許可の境界の詳細については、IAM「 ユーザーガイド」のIAM「エンティティのアクセス許可の境界」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) の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、ビジネスが所有する複数の をグループ化して一元管理するためのサービス AWS アカウント 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 (SCPs) をアカウントの一部またはすべてに適用できます。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SCPを制限します AWS アカウントのルートユーザー。Organizations と の詳細についてはSCPs、AWS Organizations 「 ユーザーガイド」のSCPs「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、「 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。 IAM

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。が複数のポリシータイプが関与する場合にリクエストを許可するかどうか AWS を決定する方法については、IAM「 ユーザーガイド」の「ポリシー評価ロジック」を参照してください。

の AWS Transfer Family 仕組み IAM

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 とどのように連携するかの概要を確認するにはIAM、 IAM ユーザーガイドのAWS 「 と連携する のサービスIAM」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 ID 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、およびアクションが許可または拒否される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IAMJSON「ポリシー要素リファレンス」を参照してください。

アクション

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

JSON ポリシーの Action要素は、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を記述します。ポリシーアクションは通常、関連付けられた AWS APIオペレーションと同じ名前です。一致するAPIオペレーションがないアクセス許可のみのアクションなど、いくつかの例外があります。また、ポリシーに複数の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、アクションの前に次のプレフィックス AWS Transfer Family を使用します。 transfer:例えば、Transfer Family CreateServerAPI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ユーザーに付与するには、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

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

Resource JSON ポリシー要素は、アクションが適用されるオブジェクトを指定します。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、リソースレベルの許可と呼ばれる特定のリソースタイプをサポートするアクションに対して実行できます。

オペレーションのリスト化など、リソースレベルの権限をサポートしないアクションの場合は、ステートメントがすべてのリソースに適用されることを示すために、ワイルドカード (*) を使用します。

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

Transfer Family サーバーリソースには、次の がありますARN。

```
arn:aws:transfer:${Region}:${Account}:server/${ServerId}
```

例えば、s-01234567890abcdefTransfer Family サーバーをステートメントで指定するには、次のを使用しますARN。

```
"Resource": "arn:aws:transfer:us-east-1:123456789012:server/s-01234567890abcdef"
```

の形式の詳細についてはARNs、「サービス認証リファレンス」の「Amazon リソースネーム (ARNs)」、またはIAMARNsIAM「 ユーザーガイド」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、複数のリソースに対して実行されます。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。1 つのステートメントで複数のリソースを指定するには、 をカンマARNs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

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

Condition 要素 (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 要素を指定する場合、または 1 つの Condition 要素に複数のキーを指定する場合、 AWS では 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ユーザー名でタグ付けされている場合にのみ、リソースにアクセスする許可をIAMユーザーに付与できます。詳細については、「 ユーザーガイド」のIAM「ポリシー要素: 変数とタグ」を参照してください。 IAM

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、 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。 IAM

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 ID 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 ID 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 が Amazon S3 のリソースベースのアクセス許可ポリシーをサポート *buckets*。 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できます。リソースベースのポリシーを使用して、Amazon S3 へのアクセスを AWS サービスに許可することもできます。*buckets*.

クロスアカウントアクセスを有効にするには、リソースベースのポリシー のプリンシパルとして、別のアカウントのアカウントまたはIAMエンティティ全体を指定できます。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の違い」を参照してください。

Amazon S3 サービスは、 *bucket* ポリシー 。これは にアタッチされます。*bucket*。 このポリシーは、オブジェクトに対してアクションを実行できるプリンシパルエンティティ (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) 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 AWS Transfer Family リソースへのアクセスを制御する方法については、「」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 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したり、IAMロールを引き受けたり、クロスアカウントロールを引き受けたりすることができます。AssumeRole や などのオペレーションを呼び出す AWS STS APIことで、一時的なセキュリティ認証情報を取得しますGetFederationToken。

AWS Transfer Family は一時的な認証情報の使用をサポートしています。

AWS Transfer Family ID ベースのポリシーの例

デフォルトでは、IAMユーザーとロールには AWS Transfer Family リソースを作成または変更するアクセス許可がありません。また、 AWS Management Console、 AWS CLI、または AWS を使用してタスクを実行することはできませんAPI。IAM 管理者は、ユーザーとロールに必要な特定のリソースに対して特定のAPIオペレーションを実行するアクセス許可を付与するIAMポリシーを作成する必要があります。その後、管理者は、これらのアクセス許可を必要とするIAMユーザーまたはグループにこれらのポリシーをアタッチする必要があります。

これらのポリシードキュメント例を使用して IAM ID ベースのJSON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
- 自分の権限の表示をユーザーに許可する

ポリシーのベストプラクティス

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

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

- IAM ポリシーの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、アクションと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 を使用して送信する必要があることを指定できますSSL。また、 などの特定のを通じてサービスアクションが使用されている場合は AWS のサービス、 条件を使用して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IAMJSON「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、安全で機能的なアクセス許可を確保する – IAM Access Analyzer は、ポリシーがポリシー言語 (JSON) とIAMベストプラクティスに準拠するように、新規および既存のIAMポリシーを検証します。IAM Access Analyzer には、安全で機能的なポリシーの作成に役立つ 100 を超えるポリシーチェックと実用的なレコメンデーションが用意されています。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。
- 多要素認証が必要 (MFA) – 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、 をオンにMFAしてセキュリティを強化します。API オペレーションが呼び出されるMFAタイミングを要求するには、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。

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「 のセキュリティのベストプラクティスIAM」を参照してください。 IAM

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。最低限必要なアクセス許可よりも制限の厳しいアイデンティティベースのポリシーを作成すると、そのポリシーを持つエンティティ (IAMユーザーまたはロール) に対してコンソールは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 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 AWS API。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を許可します。

自分の権限の表示をユーザーに許可する

この例では、IAMユーザーがユーザー ID にアタッチされているインラインポリシーとマネージドポリシーを表示できるようにするポリシーを作成する方法を示します。このポリシーには、コンソールで、または AWS CLI または 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 AWS API。

```
{
    "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

以下は、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です。

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、リソースにタグを含めることができ、一部のアクションには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は、タグ条件キーを使用して以下を制御できます。

- 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コントロールを使用すると、 APIレベルよりも詳細なコントロールを適用できます。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。リクエストで指定されたタグ (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、 で運用されているリソースのタグ (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AWS 「リソースタグを使用したリソースへのアクセスの制御」を参照してください。Gateway でIAMポリシーを指定する方法の詳細については、「APIGateway APIデベロッパーガイド」の「アクセスIAM許可APIを持つ 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

AWS Transfer Family 以下の情報は、 および の使用時に発生する可能性のある一般的な問題を診断して修正するのに役立ちますIAM。

トピック

- でアクションを実行する権限がありません 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ありません AWS Transfer Family

がアクションを実行する権限がないと AWS Management Console 指示した場合は、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。

次のエラー例は、mateojacksonIAMユーザーがコンソールを使用して の詳細を表示しようとすると発生します。*widget* ただし、 にはtransfer:*GetWidget*アクセス許可がありません。

```
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。

次のエラー例は、 という名前のIAMユーザーがコンソールを使用して marymajor でアクションを実行しようとすると発生します AWS Transfer Family。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。メアリーには、ロールをサービスに渡す許可がありません。

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

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

サポートが必要な場合は、 AWS 管理者にお問い合わせください。サインイン認証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 (ACLs) をサポートするサービスでは、これらのポリシーを使用して、リソースへのアクセスをユーザーに許可できます。

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については、「」を参照してくださいの AWS Transfer Family 仕組み IAM。
- 所有 AWS アカウント している 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。
- サードパーティー にリソースへのアクセスを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「外部認証されたユーザーへのアクセスを提供する (ID フェデレーション)」を参照してください。
- クロスアカウントアクセスにロールとリソースベースのポリシーを使用する方法の違いについては、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の違い」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。これにはSOC、、PCIHIPAA、 などが含まれます。完全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 の範囲内のサービスのリストについては、AWS コンプライアンスプログラムの範囲内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、「」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、データの機密性、会社のコンプライアンス目的、および適用される法律と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に準拠した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、リソース設定が内部プラクティス、業界ガイドライン、および規制にどの程度準拠しているかを評価します。
- AWS 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。

でのレジリエンス 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、および冗長性の高い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、接続リクエストと転送リクエストの自動スケーリング、冗長フリートによってサポートされています。

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
- Virtual Private Cloud (VPC) のエンドポイントについては、「」を参照してくださいVirtual Private Cloud でサーバーを作成する。

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ための構築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として、 AWS Transfer Family は AWS グローバルネットワークセキュリティによって保護されています。 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 「 Cloud Security 」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。

AWS 公開されたAPI呼び出しを使用して、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) 。1.2 が必要でTLS、1.3 TLS をお勧めします。
- （ DHEエフェメラルディフィ-ヘルマンPFS) や (エリプティックカーブエフェメラルディフィ-ヘルマン) など、完全なフォワードシークレット ECDHE () を持つ暗号スイート。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。

さらに、リクエストは、アクセスキー ID とプリンシIAM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 を攻撃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。これを使用して、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 (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、「 を使用して を保護する AWS WAF APIs」を参照してください。

追加するには AWS WAF

1. で API Gateway コンソールを開きますhttps://console.aws.amazon.com/apigateway/。
2. APIs 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。

   - ウェブアプリケーションファイアウォール (WAF) で、ウェブ ACLで、このステージACLに関連付けるウェブを選択します。
   - ACL 必要なウェブが存在しない場合は、以下を実行してウェブを作成する必要があります。

     1. Web の作成 ACLを選択します。
     2. サービスのホームページで AWS WAF、ウェブの作成 ACLを選択します。
     3. ウェブACLの詳細 で、名前 にウェブ の名前を入力しますACL。
     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. ルール に一致しないリクエストのデフォルトウェブACLアクションでは、許可またはブロック を選択し、次へ をクリックします。
14. ステップ 4 と 5 では [Next] (次へ) を選択します。
15. の確認と作成で、選択を確認してから、ウェブの作成 ACLを選択します。

7. [Save Changes] を選択します。
8. [Resources] (リソース) を選択します。
9. アクション で、デプロイ API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 での安全性については、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 によるセキュリティ保護」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、 AWS では、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を提供しています。この問題の詳細については、 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。

Transfer AWS Family がリソースに対して持つアクセス許可を制限するには、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。

混乱した代理問題から保護する最も効果的な方法は、許可するリソースの正確な 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、 の未知の部分にワイルドカード文字 (*) を含むaws:SourceArnグローバルコンテキスト条件キーを使用しますARN。例えば、arn:aws:transfer::*region*::*account-id*:server/* と指定します。

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。 AWS Transfer Family は、Transfer Family ユーザー のコンテキストでこのロールを引き受けますARN。
- 「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、アクセスロールはアグリーメントに 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ARNに を使用します。
- 呼び出しロール - サーバーのカスタム ID プロバイダーとして Amazon API Gateway で使用します。Transfer Family は、Transfer Family サーバー のコンテキストでこのロールを引き受けますARN。
- 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 のコンテキストでこのロールを引き受けますARN。アウトバウンドAS2転送の場合、ログ記録ロールはコネクタ を使用しますARN。
- 「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 ワークフロー のコンテキストでこのロールを引き受けますARN。

詳細については、「 ユーザーガイド」の「 のポリシーとアクセス許可IAM」を参照してください。IAM

> **Note**
>
> 次の例では、各 を置き換えます。*user input placeholder* 独自の情報。

> **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 ユーザーロールのクロスサービス混乱した代理防止

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"
                   ]
                }
            }
        }
    ]
}
```

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 管理ポリシーの詳細については、「 ユーザーガイド」のAWS 「 管理ポリシー」を参照してく

ださい。 IAM 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 マネージド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 管理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損なわ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ジョブ関数の マネージドポリシー AWS をサポートしています。例えば、 ReadOnlyAccess AWS マネージドポリシーは、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。ジョブ関数ポリシーのリストと説明については、「 ユーザーガイド」のAWS 「ジョブ関数の管理ポリシー」を参照してください。 IAM

AWS マネージドポリシー： 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 – 証明書の Amazon リソースネーム (ARNs) と各 のドメイン名のリストを取得するアクセス許可を付与しますARN。
- 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 つ以上の仮想プライベートクラウド () を記述するアクセス許可を付与しますVPCs。
- ec2:DescribeVpcEndpoints – 1 つ以上のVPCエンドポイントを記述するアクセス許可を付与します。
- health:DescribeEventAggregates一各イベントタイプ (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一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一すべての管理ポリシーを一覧表示する許可を付与
- iam:ListRoles – 指定されたパスプレフィックスを持つIAMロールを一覧表示するアクセス許可を付与します。
- iam:PassRole – Transfer Family にIAMロールを渡すアクセス許可を付与します。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一現在の 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一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:* — Transfer Family 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ロールを渡すアクセス許可を付与します。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アクセス許可を付与します。

```
{
    "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許可します。

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 ファミリーの更新を管理ポリシーに転送する

このサービスがこれらの変更の追跡を開始してからの AWS Transfer Family の AWS マネージドポリシーの更新に関する詳細を表示します。このページ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ページのの RSSフィードにサブスクライブしてください。

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、ポリシーに の読み取りを許可する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 は AWS 、管理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

以下の情報を使用して、 の使用時に発生する可能性のある一般的な問題を診断し、修正します AWS Transfer Family。

Transfer Family IAMの に関する問題については、「」を参照してくださいAWS Transfer Family IDとアクセスのトラブルシューティング。

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 の問題のトラブルシューティング
- 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の問題のトラブルシューティング
- ID プロバイダーのテストのトラブルシューティング
- SFTP コネクタに信頼されたホストキーを追加するトラブルシューティング
- 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 問題のトラブルシューティング

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サービスマネージド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
- トラブルシューティングでSSHパブリックキーを追加できませんでした

Amazon EFSサービスマネージド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) にアクセスするアクセス許可がありませんEFS。

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 マネージド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PGPキー本文に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 キーが RSAベースの場合は、PEM形式に変換できます。例えば、Ubuntu は https://manpages.ubuntu.com/manpages/xenial/man1/openpgp2ssh.1.html という変換ツールを提供しています。

トラブルシューティングでSSHパブリックキーを追加できませんでした

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、サービスマネージドユーザーの SSH2形式のパブリックキーをサポート AWS Transfer Family 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 パブリックキーをPEM形式に変換する」で説明されています。

Amazon API Gateway 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APIゲートウェイ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 () を使用してサーバーに接続しようとするSFTP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、問題は、無効なロール Amazon リソースネーム (ARN) が原因である可能性があります。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、ロールARNはすべてのゼロのロール ID 値で示されるように、プレースホルダーのみです。

```
{
    "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をARN、サーバーへのアクセス許可を持つ実際のロールに置き換えます。

ロールを更新するには

1. AWS CloudFormation コンソールを https://console.aws.amazon.com/cloudformation で開きます。
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. [Update] (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ARNを持つロール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。レビュー時 ***stack*** 「」ページを選択し、が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 () を使用してサーバーに接続しようとするSFTP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 ログ記録ロールが Transfer Family と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アクセス許可がありません。

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

- 認証の失敗 —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

認証の失敗 —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**
>
> Gateway を使用していて、このエラーが表示された場合は、API「」を参照してください認証失敗の回数が多すぎる。

原因

ユーザーのRSAキーペアを追加していないため、代わりにパスワードを使用し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

- 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
- ワークフローコピーエラーのトラブルシューティング

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、 CloudWatch ログに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。この特定の例では、ステップはファイルにタグ付けするように設定されていますが、Amazon EFS ファイルシステム内のファイルのタグ付けはサポートされていないため、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、 連絡先 AWS AWS サポート 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 分間の待機期間後に再度ログインして変更を確認する必要があります。

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、送信先バケット AWS リージョン とは異なる 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

説明

復号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場合、暗号化クライアントは、FIPS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ことがあります。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 (ブロックFIPS以外の暗号モード) 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キーを編集してから、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. キーを編集して、 を除くすべてのアルゴリズムを保持しますOCB。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の問題のトラブルシューティング

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

- 欠落しているPOSIXプロファイルのトラブルシューティング
- Amazon で論理ディレクトリをトラブルシューティングする EFS

欠落しているPOSIXプロファイル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 1 つは、 AWS Lambdaでバックアップされた Amazon API Gateway メソッドを作成するために指定したテンプレートに現在POSIX情報が含まれていないことです。

POSIX 情報を提供した場合、POSIX情報提供に使用した形式が Transfer Family によって正しく解析されない可能性があります。

解決策

`PosixProfile` パラメータの Transfer Family に JSON要素を指定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。 *`uid-value`*および *`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},
```

これらのコード例は、文字列としてではなくJSONオブジェクトとして Transfer Family に PosixProfileパラメータを送信します。

また、 内には AWS Secrets Manager、次のように PosixProfileパラメータを保存する必要があります。*your-uid* と を*your-gid*、 GIDと の実際の値に置き換えますUID。

```
{"Uid": your-uid, "Gid": your-gid, "SecondaryGids": []}
```

Amazon で論理ディレクトリをトラブルシューティングする EFS

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 を使用している場合EFS、ユーザーが論理ホームディレクトリで作業する前に、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TestIdentityProviderAPI呼び出しを使用して 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 コネクタに信頼されたホストキーを追加するトラブルシューティング

説明

SFTP コネクタを作成または編集していて、信頼できるホストキーを追加すると、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に貼り付けた場合、問題は `comment` key. AWS Transfer Family does currently accept the comment portion of the key.

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、以下のファイルアップロード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

- 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。ユーザー名、アクセント付き文字、中国語FTPやアラビア語などの特定のスクリプトなど、ファイル名に特定の文字を書き込む問題やSFTP転送が発生することがあります。

原因

FTP および SFTPプロトコルでは、クライアントがファイル名の文字エンコードをネゴシエートできますが、Amazon S3 と Amazon EFSはネゴシエートできません。代わりに、UTF-8 文字の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文字エンコーディングを確認し、それが 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。例えば、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。例えば、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、「SFTP コネクタアルゴリズム」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発生しますがStartFileTransfer、問題の原因は不明であり、API呼び出し後にコネクタ 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 コネクタをテストする。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 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。 TestConnection API コマンド。

- 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 問題のトラブルシューティング

Applicability Statement 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、「」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

- へようこそ AWS Transfer Family API
- アクション
- データ型
- API リクエストを作成する
- 共通パラメータ
- 共通エラー

へようこそ AWS Transfer Family API

AWS Transfer Family は、次のプロトコルを使用して 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) ファイル転送プロトコル (SFTP)
- ファイル転送プロトコルセキュア (FTPS)
- ファイル転送プロトコル (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。

- の場合SERVICE_MANAGED、SSHパブリックキーはサーバーのユーザーのプロパティに保存されます。ユーザーは、SERVICE_MANAGED認証メソッド用に 1 つ以上のSSHパブリックキーをファイルに保持できます。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と、クライアントはユーザー名とSSHプライベートキーを提供し、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。この方法は、Amazon API Gateway を使用して ID プロバイダーからのAPI呼び出し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呼び出しAPI_GATEWAYでは と呼ばれ、コンソールではカスタムと呼ばれます。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。信頼関係を含む AWS Identity and Access Management （ IAM) ロールが作成され、そのロールにAssumeRoleアクションが割り当てられます。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、ロールはすべて、Amazon リソースネーム () で識別されますARN。キーと値のペアであるタグを、 を持つエンティティに割り当てることができますARN。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。

ID AWS Transfer Family 形式では、次の規則が適用されます。

- `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) 形式は、次の形式になります。

- サーバーの場合は、 の形式ARNsを使用しますarn:aws:transfer:*region*:*account-id*:server/*server-id*。

  サーバーの例は、 ARNですarn:aws:transfer:us-east-1:123456789012:server/s-01234567890abcdef。
- ユーザーの場合は、 の形式ARNsを使用しますarn:aws:transfer:*region*:*account-id*:user/*server-id*/*username*。

  例は `arn:aws:transfer:us-east-1:123456789012:user/s-01234567890abcdef/user1` です。

DNS 使用中のエントリ (エンドポイント) は次のとおりです。

- API エンドポイントは の形式になりますtransfer.*region*.amazonaws.com。
- 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 AWS Transfer Family 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のリストについては、「」を参照してください。 Actions.
- データ型のアルファベット順のリストについては、「」を参照してください。 Data Types.

- 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**
>
> コマンドを実際に実行するのではなく、任意のAPI呼び出しで `--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および表示できます。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
- StartFileTransfer
- StartServer
- StopServer

- TagResource
- TestConnection
- TestIdentityProvider
- UntagResource
- UpdateAccess
- UpdateAgreement
- UpdateCertificate
- UpdateConnector
- UpdateHostKey
- UpdateProfile
- UpdateServer
- UpdateUser

CreateAccess

管理者が、 ディレクトリ内のどのグループが 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アクセス権限を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、次の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を介して 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。

Pattern: `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

Amazon S3 または Amazon EFSパスとキーをユーザーに表示する方法と表示方法を指定する論理ディレクトリマッピング。Entry と Targetペアを指定する必要があります。 はパスがどのように表示されるかEntryを示し、 Target は実際の Amazon S3 または Amazon EFS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ようにする必要がありますTarget。この値は、 HomeDirectoryType が に設定されている場合にのみ設定できますLOGICAL。

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。に設定するとPATH、ユーザーはファイル転送プロトコルクライアントに絶対 Amazon S3 バケットまたは Amazon EFSパスをそのまま表示します。に設定する場合はLOGICAL、Amazon S3 または Amazon EFSパスをユーザーに表示させる方法HomeDirectoryMappingsのマッピングを に提供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 BLOB JSON として AWS Transfer Family 保存します。ポリシーを BLOB JSON として保存し、Policy引数に渡します。
> 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、「 リファレンスAssumeRole」の「」を参照してください。 AWS Security Token Service API

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 ID (Uid) 、グループ ID (Gid) 、および 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セカンダリグループ IDs (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たPOSIXアクセス許可により、Amazon EFS ファイルシステムとの間でファイルを転送する際にユーザーが得るアクセス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 () ロールARNの Amazon リソースネーム AWS Identity and Access Management （ IAM ）。このロールにアタッチされたポリシーは、Amazon S3 バケットまたは Amazon EFS ファイルシステムとの間でファイルを転送する際にユーザーに提供するアクセスレベルを決定します。IAM ロールには、ユーザーの転送リクエストを処理するときにサーバーがリソースにアクセスできるようにする信頼関係も含める必要があります。

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 200 HTTP レスポンスを送り返します。

次のデータは、 サービスによって JSON 形式で返されます。

ExternalId

ユーザーが 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。

Pattern: `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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

CreateAgreement

契約を作成します。契約は、 AWS Transfer Family サーバーとAS2プロセス間の二者間取引パートナー契約またはパートナーシップです。この契約では、サーバーとAS2プロセス間のファイルとメッセージの転送関係を定義します。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 で識別されPartnerProfileId、AS2プロセスは で識別されますLocalProfileId。

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、次のJSON形式のデータを受け入れます。

AccessRole

コネクタは、 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

ではAS2、 を呼び出しStartFileTransferて、リクエストパラメータ で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 (の場合、親ディレクトリは など/bucket/dir/) を使用して--send-file-paths /bucket/dir/file.txt、処理されたAS2メッセージファイルを一時的に保存し、パートナーから受信したMDNときに を保存し、送信の関連するメタデータを含む最終JSONファイルを書き込みます。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 コネクタに基本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アクセス許可が必要です。Secrets Manager の AWS マネージドキーではなく、カスタマー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の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 へのアクセス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。

パターン: `[\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。

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 200 HTTP レスポンスを送り返します。

次のデータは、 サービスによって JSON 形式で返されます。

AgreementId

契約の一意の識別子。この ID は、契約の削除や更新、および契約 ID の指定を必要とするその他のAPI呼び出しに使用します。

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 呼び出しは、新しい契約の契約 ID を返します。

```
{
    "AgreementId": "a-11112222333344444"
}
```

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ の場合
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

CreateConnector

コネクタを作成します。コネクタは、 AS2または SFTPプロトコルの接続のパラメータをキャプチャします。の場合AS2、外部ホストAS2サーバーにファイルを送信するにはコネクタが必要です。の場合SFTP、サーバーにファイルを送信したり、SFTPサーバーからファイルを受信したりするときにコネクタが必要ですSFTP。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**
>
> (As2Config) または AS2 () のいずれかに、1 つの設定オブジェクトのみを指定する必要がありますSFTPSftpConfig。

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",
   "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、次のJSON形式のデータを受け入れます。

AccessRole

コネクタは、 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

ではAS2、 を呼び出しStartFileTransferてリクエストパラメータ で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、の場合--send-file-paths /bucket/dir/file.txt、親ディレクトリは /bucket/dir/) を使用して、処理された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ときに を保存し、送信の関連するメタデータを含む最終JSONファイルを書き込みます。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 コネクタに基本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アクセス許可が必要です。Secrets Manager の AWS マネージドキーではなく、カスタマー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の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 へのアクセス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。

型: As2ConnectorConfig オブジェクト

必須：いいえ

LoggingRole

（ARN) ロールの Amazon リソースネーム AWS Identity and Access Management （IAM）。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します。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

SftpConfig

SFTP コネクタオブジェクトのパラメータを含む構造。

型: 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AS2またはSFTPエンドポイントURLの 。

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 200 HTTP レスポンスを送り返します。

次のデータは、 サービスによって JSON 形式で返されます。

ConnectorId

API 呼び出し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`: 取引相手のAS2サーバーURLに を指定します。
- `your-IAM-role-for-bucket-access`: ファイルの保存に使用する Amazon S3 バケットにアクセスできるIAMロール。
- ID を含むログ記録ロールARNには を使用します AWS アカウント 。
- AS2 コネクタ設定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 で説明します。

```
// 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URLSFTPサーバーの を指定します。
- `your-IAM-role-for-bucket-access`: ファイルの保存に使用する Amazon S3 バケットにアクセスできるIAMロール。
- ID を含むログ記録ロールARNには を使用します AWS アカウント 。
- SFTP コネクタ設定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、「」を参照してくださいSftpConnectorConfig。

```
// 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 呼び出しは、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"
}
```

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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、次のJSON形式のデータを受け入れます。

As2Id

As2Id は、4130 で定義されている AS2-name です。 RFC 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た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 StartFileTransferAPI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：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。

型: 文字列の配列

長さの制限: 22 の固定長

パターン: `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有効な Transfer Family サーバー組織またはパーティを表します。
- 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 200 HTTP レスポンスを送り返します。

次のデータは、 サービスによって JSON 形式で返されます。

ProfileId

API 呼び出しが成功した後に返される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。

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IDsはimport-certificate、 を実行するときに作成されます。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 呼び出しは、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"
}
```

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3

- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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、次のJSON形式のデータを受け入れます。

Certificate

（ARN) 証明書の Amazon リソースネーム AWS Certificate Manager （ ACM ）。`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「パブリック証明書をリクエストする」を参照してください。

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

プライベート IP アドレスFTPSを介してプライベート証明書を使用するように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 アドレスを指定し、発行者に関する情報を含む有効な SSL/TLS X.509 バージョン 3 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。Amazon Simple Storage Service (Amazon S3) と Amazon Elastic File System (Amazon) の 2 つのドメインを使用できますEFS。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

サーバー用に設定された仮想プライベートクラウド (VPC) エンドポイント設定。内でエンドポイントをホストする場合VPC、エンドポイントを 内のリソースにのみアクセスできるようにするか

VPC、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します。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にアクセス可能 (PUBLIC) にするか、 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ではVPC、 内でのみサーバーとリソースへのアクセスを制限したり、Elastic IP アドレスを直接アタッチしてVPCインターネットに接続したりできます。

> **Note**
>
> 2021 年 5 月 19 日以降、 AWS アカウント アカウントが 2021 年 5 月 19 日より前に を使用してサーバーを作成していない場合、 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、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_ の使用を中止するENDPOINT」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、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 アドレスによってトラフィックを制限するオプションがあります。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が有効なサーバーに使用する 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。

次のコマンドを使用して、パスフレーズのない 2048 RSA ビットキーを生成します。

```
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。

次のコマンドを使用して、パスフレーズのない 256 ECDSA ビットキーを生成します。

```
ssh-keygen -t ecdsa -b 256 -N "" -m PEM -f my-new-server-key.
```

-b のオプションに有効な値は、256、384、および 521 ECDSAです。

次のコマンドを使用して、パスフレーズのないED25519キーを生成します。

```
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予定がない場合は、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、 AWS Transfer Family 「 ユーザーガイド」のSFTP「 対応サーバーのホストキーの更新」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。API ゲートウェイ を含む、 でディレクトリを使用するAWS_DIRECTORY_SERVICEかAPI、カスタマー提供の認証 を呼び出すために必要なすべ

ての情報を含む配列を受け入れますURL。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 サービス内にユーザー認証情報を保存してアクセスできます。

を使用してAWS_DIRECTORY_SERVICE、オンプレミス環境の AWS Directory Service for Microsoft Active Directory または Microsoft Active Directory の Active Directory グループ、または AD Connector AWS を使用する 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 この設定では、 IdentityProviderDetailsパラメータを使用して認証を呼びURL出す Amazon API Gateway エンドポイント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。この値を選択する場合は、IdentityProviderDetailsデータ型の Functionパラメータで ARN Lambda 関数の 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 の Amazon CloudWatch ログ記録を有効にすることを許可する () ロールARNの Amazon リソースネーム AWS Identity and Access Management （ IAM ）EFSevents。 Amazon S3 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 IPv4 アドレスなど、単一のドットクワッド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 サーバーがクライアントを変更せずにSETSTATコマンド AWS Transfer Family を無視してファイルをアップロードできるようにするには、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SETSTATいつ電話を発信しているかを判断できます。
- 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メッセージのトランスポート方法を示します。現在は、HTTP のみがサポートされます。

型: 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) ファイル転送プロトコル): ファイル転送 SSH
- FTPS (ファイル転送プロトコルセキュア): 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 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FTPS。
> - 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。

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 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には TYPEの がありますDIRECTORY。このオプションを有効にすると、マッピングHomeDirectoryMapEntryTypeにファイルターゲットを含めるFILE場合は、 を明示的に に設定する必要があります。

型: S3StorageOptions オブジェクト

必須 : いいえ

SecurityPolicyName

サーバーにアタッチする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。

パターン: TransferSecurityPolicy-.+

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、ARN既存のロググループの 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、`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
```

型: 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。

パターン: `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。

型: 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 200 HTTP レスポンスを送り返します。

次のデータは、 サービスによって 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呼び出しのサンプルレスポンスです。

レスポンス例

```
{
   "ServerId": "s-01234567890abcdef"
}
```

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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、次の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

Amazon S3 または Amazon EFSパスとキーをユーザーに表示する方法と表示方法を指定する論理ディレクトリマッピング。Entry と Targetペアを指定する必要があります。 はパスがどのように表示されるかEntryを示し、 Target は実際の Amazon S3 または Amazon EFS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ようにする必要がありますTarget。この値は、 HomeDirectoryType が に設定されている場合にのみ設定できますLOGICAL。

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。に設定するとPATH、ユーザーはファイル転送プロトコルクライアントに絶対 Amazon S3 バケットまたは Amazon EFSパスをそのまま表示します。に設定する場合はLOGICAL、Amazon S3 または Amazon EFSパスをユーザーに表示させる方法HomeDirectoryMappingsのマッピングを に提供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 BLOB JSON として AWS Transfer Family 保存します。ポリシーを BLOB JSON として保存し、`Policy`引数に渡します。
> 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 「 セキュリティトークンサービス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 ID (`Uid`) 、グループ ID (`Gid`) 、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任意のセカンダリグループ IDs (`SecondaryGids`) を含む完全な POSIX ID を指定します。Amazon のファイルとディレクトリに設定されたPOSIXアクセス許可は、Amazon EFS ファイルシステムとの間でファイルを転送する際にユーザーが得るアクセスレベルEFSを決定します。

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 () ロールARNの Amazon リソースネーム AWS Identity and Access Management （ IAM ）。このロールにアタッチされたポリシーは、Amazon S3 バケットまたは Amazon EFS ファイルシステムとの間でファイルを転送する際にユーザーに提供するアクセスレベルを決定します。IAM ロー

ルには、ユーザーの転送リクエストを処理するときにサーバーがリソースにアクセスできるようにする信頼関係も含める必要があります。

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

サーバーへのユーザーの認証に使用される Secure Shell (SSH) キーのパブリック部分。

3 つの標準SSHパブリックキー形式要素は`<key type>`、、`<body base64>`、およびオプションの で`<comment>`、各要素間にスペースがあります。

AWS Transfer Family は、RSA、ECDSA、および ED25519キーを受け入れます。

- RSA キーの場合、キータイプは ですssh-rsa。
- ED25519 キーの場合、キータイプは ですssh-ed25519。
- ECDSA キーの場合、キータイプはecdsa-sha2-nistp256、生成したキーのサイズに応じてecdsa-sha2-nistp384、ecdsa-sha2-nistp521、、または のいずれかです。

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。

Pattern: `[\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 200 HTTP レスポンスを送り返します。

次のデータは、 サービスによって 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。

パターン: `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。

Pattern: `[\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

ユーザーを作成するには、まず などのJSONファイルにパラメータを保存しcreateUserParameters、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"
}
```

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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、次の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。

Pattern: `[\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 関数は例外ステップを開始APIするためにコールバックFAILUREに送信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。

型: 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。

ファイルの場所には、Amazon S3 バケットとキー、または Amazon EFS ファイルシステム ID とパスを指定します。

型: 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 200 HTTP レスポンスを送り返します。

次のデータは、 サービスによって 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"
}
```

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用

- AWS SDK Ruby V3 用

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、次の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を介して 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
```

このコマンドで、 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。

Pattern: 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

アクションが成功すると、サービスは空のHTTP本文で 200 HTTP レスポンスを送り返します。

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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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、次の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

アクションが成功すると、サービスは空のHTTP本文で 200 HTTP レスポンスを送り返します。

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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go

- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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、次の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

アクションが成功すると、サービスは空のHTTP本文で 200 HTTP レスポンスを送り返します。

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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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、次の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

アクションが成功すると、サービスは空のHTTP本文で 200 HTTP レスポンスを送り返します。

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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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、次の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

アクションが成功すると、サービスは空のHTTP本文で 200 HTTP レスポンスを送り返します。

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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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、次の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

アクションが成功すると、サービスは空のHTTP本文で 200 HTTP レスポンスを送り返します。

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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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、次の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

アクションが成功すると、サービスは空のHTTP本文で 200 HTTP レスポンスを送り返します。

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

```
{
}
```

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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、次の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

ユーザー固有の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。

Pattern: `[\w][\w@.-]{2,99}`

必須：はい

レスポンス要素

アクションが成功すると、サービスは空のHTTP本文で 200 HTTP レスポンスを送り返します。

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"
}
```

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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、次の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。

Pattern: `[\w][\w@.-]{2,99}`

必須：はい

レスポンス要素

アクションが成功すると、サービスは空のHTTP本文で 200 HTTP レスポンスを送り返します。

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"
}
```

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の場合。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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、次の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

アクションが成功すると、サービスは空のHTTP本文で 200 HTTP レスポンスを送り返します。

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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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、次の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を介して 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
```

このコマンドで、 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。

Pattern: 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 200 HTTP レスポンスを送り返します。

次のデータは、 サービスによって 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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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、次の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 200 HTTP レスポンスを送り返します。

次のデータは、 サービスによって 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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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、次の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 200 HTTP レスポンスを送り返します。

次のデータは、 サービスによって 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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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、次の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",
      "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 200 HTTP レスポンスを送り返します。

次のデータは、 サービスによって 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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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呼び出しは、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、次の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。

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：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 200 HTTP レスポンスを送り返します。

次のデータは、 サービスによって 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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用

- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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、次の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 200 HTTP レスポンスを送り返します。

次のデータは、 サービスによって 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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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、次の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 200 HTTP レスポンスを送り返します。

次のデータは、 サービスによって 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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

DescribeSecurityPolicy

ファイル転送プロトコル対応サーバーに適用され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。DB セキュリティポリシーの詳細については、「DB 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、次のJSON形式のデータを受け入れます。

SecurityPolicyName

サーバーにアタッチする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: `TransferSecurityPolicy-.+`

必須：はい

レスポンスの構文

```
{
   "SecurityPolicy": {
      "Fips": boolean,
      "SecurityPolicyName": "string",
      "SshCiphers": [ "string" ],
      "SshKexs": [ "string" ],
      "SshMacs": [ "string" ],
      "TlsCiphers": [ "string" ]
```

```
    }
}
```

レスポンス要素

アクションが成功すると、サービスは 200 HTTP レスポンスを送り返します。

次のデータは、 サービスによって 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"
        ]
```

```
    }
}
```

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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 に設定すると VPC、レスポンスには が含まれますEndpointDetails。

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、次の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 200 HTTP レスポンスを送り返します。

次のデータは、 サービスによって 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 1 つの使用状況を示しています 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
    }
}
```

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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、次の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、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。

Pattern: `[\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 200 HTTP レスポンスを送り返します。

次のデータは、 サービスによって 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。

パターン: `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"
    }
}
```

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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、次の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 200 HTTP レスポンスを送り返します。

次のデータは、 サービスによって 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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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、次の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

- にはCLI、 URI 形式の証明書のファイルパスを指定します。例えば、`--certificate file://encryption-cert.pem` と指定します。または、未加工のコンテンツを使用できます。

- にSDK、証明書ファイルの raw コンテンツを指定します。例えば、`--certificate "`cat encryption-cert.pem`"` と指定します。

型: 文字列

長さの制限: 最小長は 1 です。最大長は 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。

パターン: `[\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。

パターン: `[\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

- にはCLI、プライベートキーのファイルパスを URI 形式で指定します。例えば、 です`--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。

- にはSDK、プライベートキーファイルの raw コンテンツを指定します。例えば、`--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。

パターン: `[\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

この証明書を署名に使用するか、暗号化に使用するかを指定します。

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 200 HTTP レスポンスを送り返します。

次のデータは、 サービスによって 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、この形式を使用します。

```
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。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SDK。

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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

ImportHostKey

ServerId パラメータで指定されたサーバーにホストキーを追加します。

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、次の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。

パターン: [\p{Print}]*

必須: いいえ

HostKeyBody

キーペアのプライベートSSHキー部分。

AWS Transfer Family は、RSA、ECDSA、および 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 200 HTTP レスポンスを送り返します。

次のデータは、 サービスによって 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。

パターン: `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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

ImportSshPublicKey

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値によって識別される Transfer Family ユーザーに、 によって識別される Secure Shell (SSH) パブリックキーを追加しますServerId。

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、次の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

キーペアのパブリックSSHキー部分。

AWS Transfer Family は、RSA、ECDSA、および 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。

Pattern: `[\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 200 HTTP レスポンスを送り返します。

次のデータは、 サービスによって JSON 形式で返されます。

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。

パターン: `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。

パターン: `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。

Pattern: `[\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"
}
```

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go

- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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、次のJSON形式のデータを受け入れます。

MaxResults

返SIDsされるアクセス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 200 HTTP レスポンスを送り返します。

次のデータは、 サービスによって 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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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、次の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。

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

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 200 HTTP レスポンスを送り返します。

次のデータは、 サービスによって 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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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、次の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。

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

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 200 HTTP レスポンスを送り返します。

次のデータは、 サービスによって 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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3

- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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、次の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。

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

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 200 HTTP レスポンスを送り返します。

次のデータは、 サービスによって 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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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、次の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。

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

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。その場合は、 を指定APIして を呼び出しますmax-results。

```
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API、再度 を呼び出し、受け取ったNextToken値を指定できるようになりました。

```
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 200 HTTP レスポンスを送り返します。

次のデータは、 サービスによって 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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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、次の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。

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

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 200 HTTP レスポンスを送り返します。

次のデータは、 サービスによって 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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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、次の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。

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

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 200 HTTP レスポンスを送り返します。

次のデータは、 サービスによって 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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

ListSecurityPolicies

ファイル転送プロトコル対応サーバーに適用されているセキュリティポリシ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、次の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 200 HTTP レスポンスを送り返します。

次のデータは、 サービスによって 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。

型: 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。

Pattern: TransferSecurityPolicy-.+

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-2022-03",
        "TransferSecurityPolicy-FIPS-2020-06",
        "TransferSecurityPolicy-2018-11",
        "TransferSecurityPolicy-2023-05",
        "TransferSecurityPolicy-FIPS-2023-05",
        "TransferSecurityPolicy-2020-06"
    ]
}
```

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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、次の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 200 HTTP レスポンスを送り返します。

次のデータは、 サービスによって 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
        }
    ]
}
```

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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、次の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 () に関連付けられたタグをリクエストしますARN。ARN は、サーバー、ユーザー、ロールなどの特定の 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 200 HTTP レスポンスを送り返します。

次のデータは、 サービスによって JSON 形式で返されます。

Arn

のタグを一覧表示するためにARN指定した 。

型: 文字列

長さの制限: 最小長は 20 です。最大長は 1600 です。

パターン: 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 を持つリソースのタグを一覧表示ARN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 1 つの使用状況を示しています ListTagsForResource。

レスポンス例

```
{
   "Tags": [
      {
         "Key": "Name",
         "Value": "MyServer"
      }
   ]
}
```

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go

- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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、次の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 200 HTTP レスポンスを送り返します。

次のデータは、 サービスによって 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。

パターン: `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 呼び出しは、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呼び出しのサンプルレスポンスです。

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"
      }
   ]
}
```

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

ListWorkflows

現在のリージョン AWS アカウント の に関連付けられた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、次の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 200 HTTP レスポンスを送り返します。

次のデータは、 サービスによって 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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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、次の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。

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：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。

Pattern: \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

アクションが成功すると、サービスは空のHTTP本文で 200 HTTP レスポンスを送り返します。

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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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 コネクタには、 ConnectorIdと 1 つ以上の を指定SendFilePathsして、転送するファイルを識別します。
- SFTP コネクタの場合、ファイル転送はアウトバウンドまたはインバウンドのいずれかになります。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 RetreiveFilePaths 1 つ以上の を指定し、LocalDirectoryPathを使用して宛先フォルダを指定します。
  - ストレージから AWS 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 SendFilePaths 1 つ以上の を指定し、 RemoteDirectoryPathを使用して送信先フォルダを指定します。

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、次の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、 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LocaDirectoryPath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、 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RemoteDirectoryPathを指定します。を指定しない場合RemoteDirectoryPath、転送されたファイルの宛先は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。

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 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* と指定します。

> **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。

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 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 200 HTTP レスポンスを送り返します。

次のデータは、 サービスによって 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取引パートナー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"
}
```

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK Go の場合
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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、次の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

アクションが成功すると、サービスは空のHTTP本文で 200 HTTP レスポンスを送り返します。

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呼び出しのサンプルレスポンスです。

レスポンス例

```
{
   "ServerId": "s-01234567890abcdef"
}
```

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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、次の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

アクションが成功すると、サービスは空のHTTP本文で 200 HTTP レスポンスを送り返します。

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呼び出しのサンプルレスポンスです。

レスポンス例

```
{
   "ServerId": "s-01234567890abcdef"
}
```

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

TagResource

Amazon リソースネーム () で識別されるように、キーと値のペアをリソースにアタッチしますARN。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、次の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 AWS リソースの 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

Tags

に割り当てられたキーと値のペアARNs。 を使用して、タイプ別にリソースをグループ化および検索できます。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

アクションが成功すると、サービスは空のHTTP本文で 200 HTTP レスポンスを送り返します。

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 1 つの使用状況を示しています TagResource。

レスポンス例

```
        HTTP 200 response with an empty HTTP body.
```

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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 ストレージと取引先の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、次の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 200 HTTP レスポンスを送り返します。

次のデータは、 サービスによって 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。

パターン: `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。
- コネクタ設定 URLのサーバーを確認し、ログイン認証情報がコネクタの外部で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

成功すると、API呼び出しは次の詳細を返します。

```
{

    "Status": "OK",

    "StatusMessage": "Connection succeeded"

}
```

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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、次の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) ファイル転送プロトコル (SFTP)
- ファイル転送プロトコルセキュア (FTPS)
- ファイル転送プロトコル (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。

Pattern: [\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 200 HTTP レスポンスを送り返します。

次のデータは、 サービスによって 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

Gateway API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。

型: 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 ID プロバイダーからのメッセージを返します 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は、次のコマンドを入力して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 です
- ユーザーロールは です 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は呼び出し -URL

```
{
    "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"
}
```

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

UntagResource

Amazon リソースネーム () で識別されるように、リソースからキーと値のペアをデタッチしますARN。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、次の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、ロールなどの特定の 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、 に割り当てられたキーと値のペアARNsで、 タイプ別にリソースを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

アクションが成功すると、サービスは空のHTTP本文で 200 HTTP レスポンスを送り返します。

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 1 つの使用状況を示しています UntagResource。

レスポンス例

```
        HTTP 200 response with an empty HTTP body.
```

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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、次の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を介して 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
```

このコマンドで、 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。

Pattern: `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

Amazon S3 または Amazon EFSパスとキーをユーザーに表示する方法と表示方法を指定する論理ディレクトリマッピング。`Entry` と `Target`ペアを指定する必要があります。 はパスの表示方法`Entry`を示し、 `Target` は実際の Amazon S3 または Amazon EFS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ようにする必要がありますTarget。この値は、HomeDirectoryType が に設定されている場合にのみ設定できますLOGICAL。

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。に設定するとPATH、ファイル転送プロトコルクライアントに、絶対 Amazon S3 バケットまたは Amazon EFSパスがそのまま表示されます。に設定する場合はLOGICAL、Amazon S3 または Amazon EFSパスをユーザーに表示させる方法HomeDirectoryMappingsのマッピングを に提供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 BLOB JSON として AWS Transfer Family 保存します。ポリシーを BLOB JSON として保存し、引Policy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「 セキュリティトークンサービス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 ID (Uid)、グループ ID (Gid)、および 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セカンダリグループ IDs (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たPOSIXアクセス許可により、Amazon EFS ファイルシステムとの間でファイルを転送する際にユーザーが得るアクセス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 () ロールARNの Amazon リソースネーム AWS Identity and Access Management （ IAM ）。このロールにアタッチされたポリシーは、Amazon S3 バケットまたは Amazon EFS ファイルシステムとの間でファイルを転送する際にユーザーに提供するアクセスレベルを決定します。IAM ロールには、ユーザーの転送リクエストを処理するときにサーバーがリソースにアクセスできるようにする信頼関係も含める必要があります。

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 200 HTTP レスポンスを送り返します。

次のデータは、 サービスによって 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。

Pattern: `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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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、次のJSON形式のデータを受け入れます。

AccessRole

コネクタは、 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

ではAS2、 を呼び出しStartFileTransferて、リクエストパラメータ で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 (の場合、親ディレクトリは など/bucket/dir/) を使用して--send-file-paths /bucket/dir/file.txt、処理されたAS2メッセージファイルを一時的に保存し、パートナーから受信したMDNときに を保存し、送信の関連するメタデータを含む最終JSONファイルを書き込みます。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 コネクタに基本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アクセス許可が必要です。Secrets Manager の AWS マネージドキーではなく、カスタマー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の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 へのアクセス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。

パターン: `[\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。

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。

パターン: `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。

パターン: `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 200 HTTP レスポンスを送り返します。

次のデータは、 サービスによって 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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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、次の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。

パターン: `[\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 200 HTTP レスポンスを送り返します。

次のデータは、 サービスによって 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

次の例では、証明書のアクティブな日付を更新し、アクティブな日付を 2022 年 1 月 16 日 16:12:07 UTC -5 時間に設定します。

リクエスト例

```
aws transfer update-certificate --certificate-id c-abcdefg123456hijk --active-date
 2022-01-16T16:12:07-05:00
```

例

このAPI呼び出しの応答例を次に示します。

レスポンス例

```
"CertificateId": "c-abcdefg123456hijk"
```

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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",
   "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、次のJSON形式のデータを受け入れます。

AccessRole

コネクタは、 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

ではAS2、 を呼び出しStartFileTransferて、リクエストパラメータ で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 (の場合、親ディレクトリは など/bucket/dir/) を使用して--send-file-paths /bucket/dir/file.txt、処理されたAS2メッセージファイルを一時的に保存し、パートナーから受信したMDNときに を保存し、送信の関連するメタデータを含む最終JSONファイルを書き込みます。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 コネクタに基本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アクセス許可が必要です。Secrets Manager の AWS マネージドキーではなく、カスタマー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の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 へのアクセス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。

型: 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

（ARN) ロールの Amazon リソースネーム AWS Identity and Access Management （IAM）。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します。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

SftpConfig

SFTP コネクタオブジェクトのパラメータを含む構造。

型: SftpConnectorConfig オブジェクト

必須：いいえ

Url

パートナーAS2またはSFTPエンドポイントURLの 。

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 200 HTTP レスポンスを送り返します。

次のデータは、 サービスによって 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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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、次の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 200 HTTP レスポンスを送り返します。

次のデータは、 サービスによって 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。

パターン: `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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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、次の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。

型: 文字列の配列

長さの制限: 22 の固定長

パターン: 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 200 HTTP レスポンスを送り返します。

次のデータは、 サービスによって 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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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、次のJSON形式のデータを受け入れます。

Certificate

Certificate Manager (ARN) AWS証明書の Amazon リソースネーム (ACM) 。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 にインポートするにはACM、 AWS「 Certificate Manager ユーザーガイド」の「 に証明書をインポートACMする」を参照してください。

プライベート IP アドレスFTPSを介してプライベート証明書を使用するように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 アドレスを指定し、発行者に関する情報を含む有効な SSL/TLS X.509 バージョン 3 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

サーバー用に設定された仮想プライベートクラウド (VPC) エンドポイント設定。内でエンドポイントをホストする場合VPC、エンドポイントを 内のリソースにのみアクセスできるようにするかVPC、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します。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にアクセス可能 (PUBLIC) にするか、 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ではVPC、 内でのみサーバーとリソースへのアクセスを制限VPC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にフェイシングしたりすることができます。

> **Note**
>
> 2021 年 5 月 19 日以降、 AWS 2021 年 5 月 19 日以前にアカウントで を使用してサーバーを作成していない場合、 をアカウントEndpointType=VPC_ENDPOINTで作成することはできません。2021 年 5 月 19 日以前に AWSアカウン

トEndpointType=VPC_ENDPOINTで を使用してサーバーを既に作成している場合、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_ の使用を中止するENDPOINT」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、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 アドレスによってトラフィックを制限するオプションがあります。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が有効なサーバーに使用する 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。

次のコマンドを使用して、パスフレーズのない 2048 RSA ビットキーを生成します。

```
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。

次のコマンドを使用して、パスフレーズのない 256 ECDSA ビットキーを生成します。

```
ssh-keygen -t ecdsa -b 256 -N "" -m PEM -f my-new-server-key.
```

-b のオプションに有効な値は、256、384、および 521 ECDSAです。

次のコマンドを使用して、パスフレーズのないED25519キーを生成します。

```
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予定がない場合は、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、 AWS Transfer Family 「 ユーザーガイド」のSFTP「 対応サーバーのホストキーの更新」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

顧客の認証API方法を呼び出すため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 の Amazon CloudWatch ログ記録を有効にすることを許可する () ロールARNの Amazon リソースネーム AWS Identity and Access Management （ IAM ）EFSevents。 Amazon S3 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 IPv4 アドレスなど、単一のドットクワッド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。 AWS Transfer Family サーバーにSETSTATコマンドを無視さ

せ、SFTPクライアントを変更せずにファイルをアップロードさせるには、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SETSTATいつ呼び出しを行っているかを特定できます。

- 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メッセージのトランスポート方法を示します。現在は、HTTP のみがサポートされます。

型: 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) ファイル転送プロトコル): ファイル転送 SSH
- FTPS (ファイル転送プロトコルセキュア): 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 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FTPS。
> - 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。

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 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には TYPEの がありますDIRECTORY。このオプションを有効にすると、マッピングHomeDirectoryMapEntryTypeにファイルターゲットを設定するFILE場合は、 を明示的に に設定する必要があります。

型: S3StorageOptions オブジェクト

必須：いいえ

SecurityPolicyName

サーバーにアタッチする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。

パターン: TransferSecurityPolicy-.+

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、ARN既存のロググループの 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、`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
```

型: 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。

パターン: `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":[]}'
```

型: 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 200 HTTP レスポンスを送り返します。

次のデータは、 サービスによって 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、 UpdateServer がエンドポイントタイプVPCとして を持つファイル転送プロトコル対応サーバーに対して呼び出され、サーバーの VpcEndpointID が使用可能な状態になってい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呼び出しのサンプルレスポンスです。

レスポンス例

```
{
   "ServerId": "s-01234567890abcdef"
}
```

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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、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以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による方法は、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、 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、コンソールの設定に選択した制限付きフラグが表示されるようにします。

```
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、次の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

Amazon S3 または Amazon EFSパスとキーをユーザーに表示する方法と表示方法を指定する論理ディレクトリマッピング。Entry と Targetペアを指定する必要があります。 はパスの表示方法Entryを示し、 Target は実際の Amazon S3 または Amazon EFS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ようにする必要がありますTarget。この値は、HomeDirectoryType が に設定されている場合にのみ設定できますLOGICAL。

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パラメータ HomeDirectory値に設定できます。

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。に設定するとPATH、ファイル転送プロトコルクライアントに、絶対 Amazon S3 バケットまたは Amazon EFSパスがそのまま表示されます。に設定する場合はLOGICAL、Amazon S3 または Amazon EFSパスをユーザーに表示させる方法HomeDirectoryMappingsのマッピングを に提供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 BLOB JSON として AWS Transfer Family 保存します。ポリシーを BLOB JSON として保存し、引`Policy`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 「 セキュリティトークンサービス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 ID (`Uid`) 、グループ ID (`Gid`) 、Amazon Elastic File Systems IDs (Amazon) へのユーザーのアクセスを制御するセカンダリグループ (`SecondaryGids`) など、完全な POSIX ID を指定しますEFS。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たPOSIXアクセス許可により、Amazon EFS ファイルシステムとの間でファイルを転送する際にユーザーが得るアクセス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 () ロールARNの Amazon リソースネーム AWS Identity and Access Management （ IAM ）。このロールにアタッチされたポリシーは、Amazon S3 バケットまたは Amazon EFS ファイルシステムとの間でファイルを転送する際にユーザーに提供するアクセスレベルを決定します。IAM ロー

ルには、ユーザーの転送リクエストを処理するときにサーバーがリソースにアクセスできるようにする信頼関係も含める必要があります。

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。

Pattern: `[\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 200 HTTP レスポンスを送り返します。

次のデータは、 サービスによって 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。

パターン: `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。

Pattern: `[\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呼び出しのサンプルレスポンスです。

レスポンス例

```
{
   "ServerId": "s-01234567890abcdef",
   "UserName": "my_user"
}
```

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、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 の 。NET
- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK for JavaScript V3
- AWS SDK PHP V3 用
- AWS SDK Python 用
- AWS SDK Ruby V3 用

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 顧客を取引先と接続します。

内容

BasicAuthSecretId

AS2 Connectors に基本的な認証サポートを提供しますAPI。基本認証を使用するには、 でシークレットの名前または Amazon リソースネーム (ARN) を指定する必要があります AWS Secrets Manager。

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 Secrets Manager に保存されており、この に直接渡すことはありませんAPI。
- 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 管理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。

コネクタの基本認証を以前に有効にした場合は、 UpdateConnectorAPI呼び出しを使用して無効にできます。例えば、 を使用している場合CLI、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URLの NONEが を使用している場合のみ指定できますHTTPS。を使用すると HTTPS、トラフィックがクリアテキストで送信されません。

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。

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})`

必須: いいえ

MdnResponse

アウトバウンドリクエスト (AWS Transfer Family サーバーからパートナーAS2サーバーへ) で、転送のパートナーレスポンスが同期的か非同期かを判断するために使用されます。以下のいずれかの値を指定する：

- SYNC: システムは同期MDNレスポンスを期待し、ファイルが正常に転送された (または転送されなかった) ことを確認します。
- NONE: MDNレスポンスが不要であることを指定します。

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**
>
> を に設定する DEFAULT (またはまったく設定しない) と、 の値SigningAlgorithm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SubjectHTTPヘッダー属性として使用します。

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: [\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。

パターン: `p-([0-9a-f]{17})`

必須: いいえ

SigningAlgorithm

コネクタで送信された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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

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 UploadDateにアップロードされた日付に基づいて、YYYY-MM-DD の日付形式に解決されますUTC。

型: 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。

パターン: [\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: FALSE。

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。

Pattern: \$\{(\w+.)+\w+\}

必須: いいえ

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

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。

パターン: [\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。

Pattern: \$\{(\w+.)+\w+\}

必須: いいえ

Target

呼び出されている Lambda 関数ARNの 。

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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

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 バケットに復号します。

> **Note**
>
> システムは、ファイルが UploadDateにアップロードされた日付に基づいて、YYYY-MM-DD の日付形式に解決されますUTC。

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。

パターン: [\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: FALSE。

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。

Pattern: \$\{(\w+.)+\w+\}

必須: いいえ

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

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。

パターン: `[\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。

Pattern: `\$\{(\w+.)+\w+\}`

必須: いいえ

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用

- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

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を介して 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
```

このコマンドで、 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。

Pattern: `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

Amazon S3 または Amazon EFSパスとキーをユーザーに表示する方法と表示方法を指定する論理ディレクトリマッピング。Entry と Targetペアを指定する必要があります。 はパスがどのように表示されるかEntryを示し、 Target は実際の Amazon S3 または Amazon EFS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ようにする必要がありますTarget。この値は、 HomeDirectoryType が に設定されている場合にのみ設定できますLOGICAL。

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。に設定するとPATH、ユーザーはファイル転送プロトコルクライアントに絶対 Amazon S3 バケットまたは Amazon EFSパスをそのまま表示します。に設定する場合はLOGICAL、Amazon S3 または Amazon EFSパスをユーザーに表示させる方法HomeDirectoryMappingsのマッピングを に提供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。

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 ID (Uid)、グループ ID (Gid)、および 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セカンダリグループ IDs (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たPOSIXアクセス許可により、Amazon EFS ファイルシステムとの間でファイルを転送する際にユーザーが得るアクセス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() ロールARNの Amazon リソースネーム AWS Identity and Access Management（IAM）。このロールにアタッチされたポリシーは、Amazon S3 バケットまたは Amazon EFS ファイルシステムとの間でファイルを転送する際にユーザーに提供するアクセスレベルを決定します。IAM ロー

ルには、ユーザーの転送リクエストを処理するときにサーバーがリソースにアクセスできるようにする信頼関係も含める必要があります。

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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

DescribedAgreement

契約のプロパティを説明します。

内容

Arn

アグリーメントの一意の 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

ではAS2、 を呼び出しStartFileTransferてリクエストパラメータ で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、の場合--send-file-paths /bucket/dir/file.txt、親ディレクトリは /bucket/dir/) を使用して、処理された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ときに を保存し、送信の関連するメタデータを含む最終JSONファイルを書き込みます。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 コネクタに基本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アクセス許可が必要です。Secrets Manager の AWS マネージドキーではなく、カスタマー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の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 へのアクセス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。

パターン: 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) 。

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。

パターン: [\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。

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})`

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。

パターン: `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。

パターン: `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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

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) 。

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。

パターン: `[\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。

パターン: [\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

パターン: 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。

パターン: [\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。

パターン: `[\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

この証明書を署名に使用するか、暗号化に使用するかを指定します。

型: 文字列

有効な値: `SIGNING | ENCRYPTION`

必須：いいえ

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

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

ではAS2、 を呼び出しStartFileTransferて、リクエストパラメータ で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 (の場合、親ディレクトリは など/bucket/dir/) を使用して--send-file-paths /bucket/dir/file.txt、処理されたAS2メッセージファイルを一時的に保存し、パートナーから受信したMDNときに を保存し、送信の関連するメタデータを含む最終JSONファイルを書き込みます。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 コネクタに基本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アクセス許可が必要です。Secrets Manager の AWS マネージドキーではなく、カスタマー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の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 へのアクセス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。

型: 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。

パターン: c-([0-9a-f]{17})

必須: いいえ

LoggingRole

（ARN) ロールの Amazon リソースネーム AWS Identity and Access Management （IAM）。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します。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

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。

型: 文字列の配列

パターン: `\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。

型: 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AS2またはSFTPエンドポイントURLの 。

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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

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。

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: 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ファイルの場所を説明する構造。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 ID (Uid) 、グループ ID (Gid) 、および 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任意のセカンダリグループ IDs (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たPOSIXアクセス許可により、Amazon EFS ファイルシステムとの間でファイルを転送する際にユーザーが得るアクセス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。

型: 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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

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。

パターン: `[\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。

型: 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。

パターン: `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`

型: 文字列

必須: いいえ

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

DescribedProfile

ローカルプロファイルまたはパートナー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、4130 で定義されている AS2-name です。 RFC 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た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。

型: 文字列の配列

長さの制限: 22 の固定長

パターン: `cert-([0-9a-f]{17})`

必須: いいえ

ProfileId

ローカルプロファイ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。

パターン: `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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go

- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

DescribedSecurityPolicy

指定されたセキュリティポリシーのプロパティについて記述します。DB セキュリティポリシーの詳細については、「DB 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

サーバーにアタッチする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: `TransferSecurityPolicy-.+`

必須：はい

Fips

このポリシーが連邦情報処理標準 () を有効にするかどうかを指定しますFIPS。

型: ブール値

必須：いいえ

SshCiphers

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 Secure Shell (SSH) 暗号暗号化アルゴリズムを指定します。

型: 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

サーバーにアタッチされているセキュリティポリシーで有効なSSHキー交換 (KEX) 暗号化アルゴリズムを指定します。

型: 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

サーバーにアタッチされているセキュリティポリシーで有効なSSHメッセージ認証コード (MAC) 暗号化アルゴリズムを指定します。

型: 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指定します。

型: 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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

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連します。非同期 の送信に使用されますMDNs。

これらの IP アドレスは、AS2サーバーの作成時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。

型: 文字列の配列

パターン: `\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ARNの 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。

型: 文字列

有効な値: `S3 | EFS`

必須：いいえ

EndpointDetails

サーバー用に設定された仮想プライベートクラウド (VPC) エンドポイント設定。内でエンドポイントをホストする場合VPC、エンドポイントを 内のリソースにのみアクセスできるようにするかVPC、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します。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されている場合、サーバーはパブリックインターネット経由で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-encodedされた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。

型: 文字列

必須: いいえ

IdentityProviderDetails

お客様が用意した認証 を呼び出す情報を指定しますAPI。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 サービス内にユーザー認証情報を保存してアクセスできます。

を使用してAWS_DIRECTORY_SERVICE、オンプレミス環境の AWS Directory Service for Microsoft Active Directory または Microsoft Active Directory の Active Directory グループ、または AD Connector AWS を使用した 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 この設定では、 IdentityProviderDetailsパラメータを使用して認証を呼びURL出す Amazon API Gateway エンドポイント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。この値を選択する場合は、IdentityProviderDetailsデータ型の Functionパラメータで ARN Lambda 関数の 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 の Amazon CloudWatch ログ記録を有効にすることを許可する () ロールARNの Amazon リソースネーム AWS Identity and Access Management （ IAM ）EFSevents。 Amazon S3 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) を指定するには、 `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 IPv4 アドレスなど、単一のドットクワッド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 サーバーがクライアントを変更せずにSETSTATコマンド AWS Transfer Family を無視し、ファイルをアップロードできるようにするには、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SETSTATいつ通話を行っているかを特定できます。
- 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。

型: 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) ファイル転送プロトコル): ファイル転送 SSH
- FTPS (ファイル転送プロトコルセキュア): 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 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FTPS。
> - 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。

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 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には TYPEの がありますDIRECTORY。このオプションを有効にすると、マッピングHomeDirectoryMapEntryTypeにファイルターゲットを設定するFILE場合は、 を明示的に に設定する必要があります。

型: S3StorageOptions オブジェクト

必須：いいえ

SecurityPolicyName

サーバーにアタッチする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。

パターン: TransferSecurityPolicy-.+

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。

パターン: 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、ARN既存のロググループの 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、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

型: 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。

パターン: `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。

型: WorkflowDetails オブジェクト

必須: いいえ

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

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

説明をリクエスト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

Amazon S3 または Amazon EFSパスとキーをユーザーに表示する方法と表示方法を指定する論理ディレクトリマッピング。`Entry` と `Target`ペアを指定する必要があります。 はパスの表示

方法Entryを示し、 Target は実際の Amazon S3 または Amazon EFS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ようにする必要がありますTarget。この値は、HomeDirectoryType が に設定されている場合にのみ設定できますLOGICAL。

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パラメータ HomeDirectory値に設定できます。

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。に設定するとPATH、ファイル転送プロトコルクライアントに、絶対 Amazon S3 バケットまたは Amazon EFSパスがそのまま表示されます。に設定する場合はLOGICAL、Amazon S3 または Amazon EFSパスをユーザーに表示させる方法HomeDirectoryMappingsのマッピングを に提供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。

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 ID (`Uid`)、グループ ID (`Gid`)、および Amazon Elastic File System IDs (Amazon `SecondaryGids`) ファイルシステムへのユーザーのアクセスを制御するセカンダリグループ (EFS) を含む完全な POSIX ID を指定します。 Amazon Elastic File System 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たPOSIXアクセス許可により、Amazon EFS ファイルシステムとの間でファイルを転送する際にユーザーが得るアクセス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 () ロールARNの Amazon リソースネーム AWS Identity and Access Management（IAM）。このロールにアタッチされたポリシーは、Amazon S3 バケットまたは Amazon EFS ファイルシステムとの間でファイルを転送する際にユーザーに提供するアクセスレベルを決定します。IAM ロールには、ユーザーの転送リクエストを処理するときにサーバーがリソースにアクセスできるようにする信頼関係も含める必要があります。

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

説明されたユーザーに保存されている Secure Shell (SSH) キーのパブリックキー部分を指定します。

型: 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。

パターン: `[\w][\w@.-]{2,99}`

必須: いいえ

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

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。

Pattern: `[\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。

型: 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。

型: 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。

パターン: `w-([a-z0-9]{17})`

必須: いいえ

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

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 によって割り当てられたファイルシステムの識別子EFS。

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。

パターン: `[^\x00]+`

必須: いいえ

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

EndpointDetails

ファイル転送プロトコル対応サーバー用に設定された仮想プライベートクラウド (VPC) エンドポイント設定。VPC エンドポイントを使用すると、 内のサーバーとリソースへのアクセスを制限できますVPC。受信インターネットトラフィックを制御するには、 を呼び出しUpdateServerAPI、サーバーのエンドポイントに Elastic IP アドレスをアタッチします。

> **Note**
>
> 2021 年 5 月 19 日以降、 AWS 2021 年 5 月 19 日以前にアカウントで を使用してサーバーを作成していない場合、 をアカウントEndpointType=VPC_ENDPOINTで作成することはできません。2021 年 5 月 19 日以前に AWSアカウントEndpointType=VPC_ENDPOINTでを使用してサーバーを既に作成している場合、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_ の使用を中止するENDPOINT」を参照してください。

内容

AddressAllocationIds

Elastic IP アドレスIDs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 Address データ型から allocationIdフィールドから取得できます。この値を取得する方法の 1 つは、 EC2 DescribeAddresses を呼び出すことですAPI。

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。
> - このパラメータは、FTPプロトコルを使用する Transfer Family サーバーには設定できません。

- サーバーには既にSubnetIds入力されている必要があります (SubnetIds と を同時に更新AddressAllocationIdsすることはできません)。
- AddressAllocationIds に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、サブネット が 3 つある場合はIDs、3 つのアドレス割り当て も指定する必要がありますIDs。
- このパラメータを設定または変更UpdateServerAPIするには、 を呼び出します。

型: 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IDsできるセキュリティグループのリスト。

> **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。SecurityGroupIds プロパティは、 を EndpointType PUBLICまたは から に変更するUpdateServerAPI場合にのみVPC_ENDPOINT、 で編集できます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 を使用しますAPI。

型: 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。

パターン: sg-[0-9a-f]{8,17}

必須: いいえ

SubnetIds

でサーバーエンドポイントIDsをホストするために必要なサブネットのリストVPC。

> **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_ の使用を中止するENDPOINT」を参照してください。

型: 文字列

長さの制限: 22 の固定長

パターン: 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VPC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。

型: 文字列

必須: いいえ

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ の場合

- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

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: 例えば、EFSファイルにタグ付けしようとするステップは `BAD_REQUEST`を返します。S3 ファイルのみがタグ付けできるためです。
- `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 のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

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。

型: 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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

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

Amazon EFS識別子と、使用するファイルのパスを指定します。

型: EfsFileLocation オブジェクト

必須：いいえ

S3FileLocation

バケット、 など、使用されているファイルの S3 の詳細を指定しますETag。

型: S3FileLocation オブジェクト

必須: いいえ

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

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場合、またはディレクトリDIRECTORYがディレクトリを指すFILE場合は、 タイプを 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ときに、ホームディレクトリマッピングには Typeが になります。マッピングにファイルターゲットを含めるFILE場合は、明示的に Typeを 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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

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。

パターン: `d-[0-9a-f]{10}`

必須: いいえ

Function

ID プロバイダーに使用する ARN Lambda 関数の 。

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対応サーバーの場合、およびカスタム ID プロバイダーのみ の場合、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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

InputFileLocation

処理中のファイルの場所を指定します。

内容

EfsFileLocation

復号される Amazon Elastic File System (Amazon EFS) ファイルの詳細を指定します。

型: 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。

型: S3InputFileLocation オブジェクト

必須: いいえ

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

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を介して 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
```

このコマンドで、 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。

Pattern: `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。に設定するとPATH、ユーザーはファイル転送プロトコルクライアントに絶対 Amazon S3 バケットまたは Amazon EFSパスをそのまま表示します。に設定する場合はLOGICAL、Amazon S3 または Amazon EFSパスをユーザーに表示させる方法HomeDirectoryMappingsのマッピングを に提供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 () ロールARNの Amazon リソースネーム AWS Identity and Access Management （ IAM ）。このロールにアタッチされたポリシーは、Amazon S3 バケットまたは Amazon EFS ファイルシステムとの間でファイルを転送する際にユーザーに提供するアクセスレベルを決定します。IAM ロールには、ユーザーの転送リクエストを処理するときにサーバーがリソースにアクセスできるようにする信頼関係も含める必要があります。

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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

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。

パターン: `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。

パターン: `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。

パターン: `[\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。

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})`

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。

パターン: `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。

パターン: `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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

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。

パターン: `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

パターン: `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。

パターン: `[\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

この証明書を署名に使用するか、暗号化に使用するかを指定します。

型: 文字列

有効な値: `SIGNING | ENCRYPTION`

必須：いいえ

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

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。

パターン: `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。

パターン: `c-([0-9a-f]{17})`

必須: いいえ

Url

パートナーAS2またはSFTPエンドポイントURLの 。

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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

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。

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: いいえ

InitialFileLocation

Amazon S3 またはEFSファイルの場所を説明する構造。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。

型: 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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

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。

パターン: `[\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。

型: 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。

パターン: `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`

型: 文字列

必須: いいえ

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

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。

パターン: `arn:\S+`

必須: いいえ

As2Id

`As2Id` は、4130 で定義されている AS2-name です。 RFC 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た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

ローカルプロファイ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。

パターン: `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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

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

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。

型: 文字列

有効な値: `S3 | EFS`

必須：いいえ

EndpointType

サーバーが接続されている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されている場合、サーバーはパブリックインターネット経由で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 サービス内にユーザー認証情報を保存してアクセスできます。

を使用してAWS_DIRECTORY_SERVICE、オンプレミス環境の AWS Directory Service for Microsoft Active Directory または Microsoft Active Directory の Active Directory グループ、または AD Connector AWS を使用した 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 この設定では、 IdentityProviderDetailsパラメータを使用して認証を呼びURL出す Amazon API Gateway エンドポイント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。この値を選択する場合は、IdentityProviderDetailsデータ型の Functionパラメータで ARN Lambda 関数の 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 の Amazon CloudWatch ログ記録を有効にすることを許可する () ロールARNの Amazon リソースネーム AWS Identity and Access Management （ IAM ）EFSevents。 Amazon S3 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。

パターン: `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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

ListedUser

指定するユーザーのプロパティを返します。

内容

Arn

学習するユーザーの一意の Amazon リソースネーム (ARN) 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 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。に設定すると`PATH`、ファイル転送プロトコルクライアントに、絶対 Amazon S3 バケットまたは Amazon EFSパスがそのまま表示されます。に設

定する場合はLOGICAL、Amazon S3 または Amazon EFSパスをユーザーに表示させる方法HomeDirectoryMappingsのマッピングを に提供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 () ロールARNの Amazon リソースネーム AWS Identity and Access Management （ IAM ）。このロールにアタッチされたポリシーは、Amazon S3 バケットまたは Amazon EFS ファイルシステムとの間でファイルを転送する際にユーザーに提供するアクセスレベルを決定します。IAM ロールには、ユーザーの転送リクエストを処理するときにサーバーがリソースにアクセスできるようにする信頼関係も含める必要があります。

> **Note**
>
> を持つサーバーの場合は Amazon S3 バケットへのユーザーのアクセスを制御するIAM ロールDomain=S3。 を持つサーバーの場合はEFSファイルシステム。 Domain=EFS このロールにアタッチされたポリシーは、S3 バケットまたはEFSファイルシステムとの間でファイルを転送する際にユーザーに提供するアクセスレベルを決定します。

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

SshPublicKeyCount

指定したユーザーに保存されているSSHパブリックキーの数を指定します。

タイプ: 整数

必須：いいえ

UserName

ARN 指定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。

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: いいえ

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

ListedWorkflow

ワークフローの識別子、テキストの説明、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。

パターン: `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。

Pattern: `[\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。

パターン: `w-([a-z0-9]{17})`

必須: いいえ

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

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 の Amazon CloudWatch ログ記録を有効にすることを許可する () ロールARNの Amazon リソースネーム AWS Identity and Access Management （ IAM ）EFSevents。 Amazon S3 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。

パターン: `[\.\-_/#A-Za-z0-9]*`

必須: いいえ

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用

- AWS SDK Ruby V3 用

PosixProfile

ユーザー ID (`Uid`)、グループ ID (`Gid`)、および 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セカンダリグループ IDs (`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たPOSIXアクセス許可により、Amazon EFS ファイルシステムとの間でファイルを転送する際にユーザーが得るアクセスレベルが決まります。

内容

Gid

このユーザーがすべてのEFSオペレーションに使用する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POSIXユーザーがすべてのEFSオペレーションに使用するユーザー 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

このユーザーがすべてのEFSオペレーションIDsに使用するセカンダリPOSIXグループ。

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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

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。

型: 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 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 IPv4 アドレスなど、単一の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NAT 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。値はデフォルトで FTPおよびFTPSタイプサーバーに`PassiveIp=AUTO`割り当てられます。この場合、サーバーはレスポンスIPs内のエンドポイントの 1 つで自動的にPASV応答します。 `PassiveIp=0.0.0.0`には、その使用に対してより一意のアプリケーションがあります。例えば、3 つのサブネットがある高可用性 (HA) Network Load Balancer (NLB) 環境がある場合は、 `PassiveIp`パラメータを使用して

1 つの 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 Control 接続と同じ IP アドレスを使用し、その接続AZsにすべて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 レスポンスをサポートしているわけではありません。 FileZilla また、WinSCP が対応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、ファイルのアップロードSETSTAT時などのコマンドを使用して、タイムスタンプやアクセス許可など、リモートファイルの属性を変更しようとします。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ENABLE_NO_OPするように 値をに設定し、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します。SetStatOption ENABLE_NO_OP 設定ではエラーは無視されますが、Amazon CloudWatch Logs でログエントリが生成されるため、クライアントがSETSTATいつ電話をかけ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、 を使用して他のファイル属性を変更する場合はSETSTAT、Transfer Family で Amazon をバックエンドストレージEFSとして使用できます。

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セッションのコントロールとデータ接続の間でネゴシエートされたシークレットキーを再開または共有するメカニズムを提供します。 は、サーバーが一意のセッション ID を介して最近のネゴシエートされたセッションを再開するかどうかTlsSessionResumptionMode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リクエストを処理しず、リクエストごとに新しいTLSセッションを作成します。
- ENABLED: サーバーは、TLSセッションの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セッションの再開を実行しているクライアントを処理し、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。したがって、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防止されます。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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

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。は、メタデータではなく、オブジェクトの内容にのみ変更ETagを反映します。

型: 文字列

長さの制限：最小長は 1 です。最大長は 65,536 です。

パターン: `.+`

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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

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には TYPEの がありますDIRECTORY。このオプションを有効にすると、マッピングHomeDirectoryMapEntryTypeにファイルターゲットを含めるFILE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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

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。

Pattern: `([\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。

Pattern: `([\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 のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

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 のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

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します。

> **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であるため、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。

3 つの標準SSHパブリックキー形式要素は<key type>、、<body base64>、およびオプションの で<comment>、各要素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。

信頼されたホストキーの場合、 AWS Transfer Family と RSAECDSAキーを受け入れます。

- RSA キーの場合、<key type>文字列は ですssh-rsa。
- ECDSA キーの場合、<key type>文字列はecdsa-sha2-nistp256、生成したキーのサイズに応じてecdsa-sha2-nistp384、ecdsa-sha2-nistp521、、または のいずれかになります。

このコマンドを実行して、SFTPサーバー名の であるSFTPサーバーホストキーを取得します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。

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 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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

SshPublicKey

特定のファイル転送プロトコル対応サーバー (で識別) の Transfer Family ユーザーに関連付けられているパブリック Secure Shell (SSH) キーに関する情報を提供しますServerId。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、ユーザー名に関連付けられた複数のSSHパブリックキーを特定のサーバーに保存できます。

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

で指定されたSSHパブリックキーの内容を指定しますPublicKeyId。

AWS Transfer Family は、RSA、ECDSA、および 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 のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

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 のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

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。

パターン: `[\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。

Pattern: `\$\{(\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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

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。

Pattern: `[\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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

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 が引き受けることができる 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ているため、すべてのワークフローステップが必要なリソースで操作できます。

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 のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

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。

型: 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":[]}'
```

型: 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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用
- AWS SDK Ruby V3 用

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: FALSE。

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。

型: 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
- 復号するソースファイルの Amazon S3 または Amazon Elastic File System (Amazon EFS) の場所。
- ファイル復号の送信先の 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: FALSE。
- 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。

型: 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。

型: 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

以下の資料も参照してください。

言語固有の のいずれかAPIでこれを使用する方法の詳細については AWS SDKs、以下を参照してください。

- AWS SDK C++ 用
- AWS SDK for Go
- AWS SDK Java V2 用

- AWS SDK Ruby V3 用

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
- Transfer Family リクエストの入力と署名
- エラーレスポンス
- 利用可能なライブラリ

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

[Header] (ヘッダー)	説明
	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。 `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エ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 1 月 16 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 のためにイベントを 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、新しいセキュリティポリシー FIPSと非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、「のセキュリティポリシー AWS Transfer Family」を参照してください。	2024 年 2 月 5 日
SFTP コネクタと の静的 IP アドレスのサポート AS2	Transfer Family がSFTPコネクタと の静的 IP アドレスを提供するようになりましたAS2。これにより、IP 許可リス	2024 年 1 月 16 日

変更	説明	日付
	トのコントロールで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。ではAS2、AS2サーバーからの非同期MDNレスポンスに静的 IP アドレスを導入します。	
ユーザーガイドは、 の最新バージョン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、ガイドの発行以降、複数の機能を追加しており、ガイドの再編が必要です。	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の最適化	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。ユーザーマッピングが ファイルへの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) を一覧表示する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

変更	説明	日付
仮想プライベートクラウド (VPC) エンドポイントを持つ 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、代替の非標準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
SFTP コネクタのサポート	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 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信および取得する」を参照してください。	2023 年 7 月 25 日
AS2 基本認証のサポート	Transfer Family は、適用可能性ステートメント 2 (AS2) プロトコルを使用するサーバーの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

変更	説明	日付
構造化JSONログ記録のサポート	Transfer Family では、構造化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、プロトコル間での一般的なログクエリの実行がサポートされ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
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プロトコルとカスタム ID プロバイダーを使用するサーバーで使用できます。詳細については、「SFTP対応サーバーを作成する」を参照してください。	2023 年 5 月 17 日

変更	説明	日付
Transfer Family がワークフローで処理するファイルによる Pretty Good Privacy (PGP) 復号のサポート	Transfer Family には、Pretty Good Privacy (PGP) 復号のサポートが組み込まれています。SFTP、、FTPSまたはFTP Amazon Simple Storage Service (Amazon S3) またはAmazon PGP Amazon Elastic File System) にアップロードされたファイルでは、復号を使用できますEFS。詳細については、「PGP キーの生成と管理」および「ワークフローでPGP復号を使用する」を参照してください。	2022 年 12 月 21 日
Transfer Family サーバーを使用した 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	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AS2送受信するサーバーを作成できます。詳細については、「の設定 AS2」を参照してください。	2022 年 7 月 25 日
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 (すべてのプロトコル) 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対応サーバーを作成する」、「FTPS対応サーバーを作成する」または「FTP対応サーバーを作成する」を参照してください。	2022 年 2 月 17 日

変更	説明	日付
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 Amazon API Gateway URLために 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
のサポート AWS Directory Service for Microsoft Active Directory	マネージドサービスおよびカスタム ID プロバイダーのほかに、 AWS Directory Service for Microsoft Active Directory を使用して認証や認可の対象として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

変更	説明	日付
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
データストアEFSとしてのAmazon のサポート	Transfer Family は、Amazon Elastic File System (Amazon) との間でのファイル転送をサポートするようになりましたEFS。Amazon EFSは、シンプルでスケーラブル、フルマネージド型のエラスティック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

変更	説明	日付
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
仮想プライベートクラウド内の複数のセキュリティグループのサポート (VPC)	のサーバーに複数のセキュリティグループをアタッチできるようになりましたVPC。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 GovCloud (US)「 ユーザーガイドAWS Transfer Family」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、「のセキュリティポリシー AWS Transfer Family」を参照してください。	2020 年 8 月 12 日

変更	説明	日付
連邦情報処理標準 (FIPS) エンドポイントのサポート	FIPSが有効なエンドポイントが北米 で利用可能になりました AWS リージョン。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が有効なサーバーエンドポイントFIPSで を有効にするには、「」を参照してくださいSFTP対応サーバーを作成する。FTPSが有効なサーバーエンドポイントFIPSで を有効にするには、「」を参照してくださいFTPS対応サーバーを作成する。FTPが有効なサーバーエンドポイントFIPSで を有効にするには、「」を参照してくださいFTP対応サーバーを作成する。	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 では、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ロールの自動作成がサポートされるようになりました。詳細については「SFTP対応サーバーを作成する」、「FTPS対応サーバーを作成する」または「FTP対応サーバーを作成する」を参照してください。	2020 年 7 月 30 日
AWS Transfer Family は、認証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 は、エンドユーザーのソース IP アドレスを認可の要素として使用するためのサポートを追加し、Secure File Transfer Protocol ()、File Transfer Protocol over SSL (SFTP ）、または File Transfer Protocol (FTPS) 経由でアクセスを承認するときにセキュリティレイヤーを追加で適用できるようにしますFTP。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 になり AWS Transfer Family 、FTP と のサポートが追加されましたFTPS。	ユーザーのファイル転送には、ファイル転送プロトコルセキュア (FTPS) とファイル転送プロトコル () の 2 つの追加プロトコルを使用できるようになりましたFTP。ユーザーは、既存の Secure File Transfer Protocol SSL (FTPS) SFTPサポートに加えて、 () およびプレーンテキストFTPベースのワークフローFTPをで移動、実行、保護 AWS、統合できます。	2020 年 4 月 23 日
仮想プライベートクラウド (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
での作業のサポート VPC	でサーバーを作成できるようになりましたVPC。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。