



ユーザーガイド

Amazon Elastic File System



Amazon Elastic File System: ユーザーガイド

Copyright © 2025 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon の商標とトレードドレスは、Amazon 以外の製品またはサービスとの関連において、顧客に混乱を招いたり、Amazon の名誉または信用を毀損するような方法で使用することはできません。Amazon が所有していないその他のすべての商標は Amazon との提携、関連、支援関係の有無にかかわらず、それら該当する所有者の資産です。

Table of Contents

Amazon Elastic File System とは	1
Amazon EFS を初めてお使いになる方向けの情報	2
仕組み	3
Amazon EFS と Amazon EC2 の連携方法	5
Amazon EFS AWS Direct Connect と AWS Managed VPN の連携方法	7
Amazon EFS と の連携方法 AWS Backup	8
機能	9
認証とアクセスコントロール	9
Amazon EFS のデータ整合性	9
可用性と耐久性	10
レプリケーション	17
入門	18
前提条件	18
ファイルシステムの作成と EC2 インスタンスの起動	18
ファイルシステムへのファイルの転送	19
前提条件	20
リソースをクリーンアップする	20
EFS クライアントのインストール	22
EFS ツールの依存関係	23
サポートされているディストリビューション	23
EFS クライアントの手動インストール	25
Amazon EC2 Linux インスタンスでの Amazon EFS クライアントのインストール	25
他の Linux ディストリビューションで amazon-efs-utils パッケージをインストールする	26
EC2 Mac インスタンスへの EFS クライアントのインストール	27
EFS クライアントの自動インストール	27
インストール時に Amazon EFS クライアントが実行する処理	28
Systems Manager がサポートするオペレーティングシステム	28
EFS クライアントをインストール AWS Systems Manager するための の設定	30
botocore のインストールとアップグレード	31
stunnel のアップグレード	32
stunnel のインストールに関する問題の解決	34
FIPS モードの有効化	35
リソースの作成と管理	37
実装についての要約	38

リソース ID	40
作成トークンと冪等性	40
ファイルシステムの作成	41
ファイルシステムの作成に必要な IAM アクセス許可	41
ファイルシステムの設定オプション	42
ファイルシステムの削除	58
ファイルシステムポリシーの作成	59
アクセスポイントの作成	62
アクセスポイントの削除	65
リソースのタグ付け	66
タグの基本	66
タグの制限	66
アクセスコントロールにタグを使用する	67
リソースのタグ付け	67
チュートリアル: 書き込み可能なユーザーごとのサブディレクトリを作成する	68
ファイルシステムをマウントする	71
Linux でのマウントに関する考慮事項	71
EFS マウントヘルパーの使用	73
EFS マウントヘルパーで使用するマウント設定	75
サポートログの取得	76
前提条件	77
EC2 Linux にマウントする	78
EC2 Mac にマウントする	80
別のリージョンからのマウント	82
1 ゾーンファイルシステムをマウントする	83
IAM 認可を使用してマウントする	87
Amazon EFS アクセスポイントを使用してマウントする	88
複数の EC2 インスタンスのマウント	89
別のアカウントまたは VPC からマウントする	90
NFS の使用	94
前提条件	94
NFS サポート	95
NFS クライアントをインストールする	96
推奨される NFS マウント設定	98
DNS による EC2 へのマウント	100
IP アドレスを使用してマウントする	102

ファイルシステムの自動マウント	104
新しい EC2 Linux インスタンス	105
既存の EC2 Linux インスタンス	107
NFS を使用する Linux および Mac インスタンス	111
ファイルシステムをアンマウントする	113
チュートリアル: を使用してファイルシステムを作成してマウントする AWS CLI	114
前提条件	115
のセットアップ AWS CLI	116
ステップ 1: EC2 リソースを作成する	117
ステップ 2: EFS リソースを作成する	123
ステップ 3: ファイルシステムをマウントしてテストする	126
ステップ 4: クリーンアップする	129
チュートリアル: オンプレミスクライアントを使用してマウントする	131
前提条件	133
ステップ 1: EFS リソースを作成する	135
ステップ 2: NFS クライアントをインストールする	137
ステップ 3: オンプレミスクライアント Amazon EFS; ファイルシステムをマウントする ...	137
ステップ 4: リソースをクリーンアップし、AWS アカウントを保護する	139
オプション: 転送中のデータの暗号化	140
チュートリアル: 別の VPC からファイルシステムをマウントする	143
前提条件	144
ステップ 1: マウントターゲットの Availability ゾーンの ID を特定する	144
ステップ 2: マウントターゲットの IP アドレスを特定する	145
ステップ 3: マウントターゲットに Host エントリを追加する	146
ステップ 4: EFS マウントヘルパーを使用してファイルシステムをマウントする	147
ステップ 5: リソースをクリーンアップして AWS アカウントを保護する	148
マウントの問題のトラブルシューティング	149
Windows インスタンスでのファイルシステムのマウントが失敗する	149
サーバーによってアクセスが拒否されました	150
自動マウントが失敗してインスタンスがレスポンスしない	150
/etc/fstab での複数の Amazon EFS ファイルシステムのマウントが失敗する	150
エラーメッセージ「wrong fs type」でマウントコマンドが失敗する	152
エラーメッセージ「incorrect mount option」でマウントコマンドが失敗する	152
アクセスポイントでのマウントは失敗します。	153
ファイルシステムを作成した後すぐにファイルシステムのマウントが失敗する	153
ファイルシステムのマウントがハングした後、タイムアウトエラーで失敗する	153

DNS 名を使用した NFS によるファイルシステムのマウントが失敗する	154
「nfs が応答していません」が表示されてファイルシステムのマウントが失敗する	155
マウントターゲットのライフサイクル状態がスタックする	155
マウントターゲットのライフサイクルの状態にエラーが表示される	155
マウントが応答しない	156
マウントされたクライアントは切断されます。	157
新しくマウントされたファイルシステムでの操作が「bad file handle」エラーを返します ..	157
ファイルシステムのアンマウントが失敗する	158
データ転送	159
の使用 AWS DataSync	159
の使用 AWS Transfer Family	160
Amazon EFS AWS Transfer Family で 使用するための前提条件	161
AWS Transfer Family向けの EFS ファイルシステムの設定	161
ファイルシステムの管理	167
ファイルシステムのステータスについて	167
マウントターゲットの管理	168
マウントターゲットとアベイラビリティゾーン	169
マウントターゲットの作成	172
マウントターゲットの削除	178
マウントターゲット VPC の変更	180
マウントターゲットセキュリティグループの変更	180
スループットの管理	182
ストレージライフサイクルの管理	184
ライフサイクル管理のファイルシステムオペレーション	185
ライフサイクルポリシーの設定	185
暗号化されたファイルシステムへのアクセスの管理	188
EFS ファイルシステム用の KMS キーの管理	189
モニタリング	190
モニタリングツール	191
自動ツール	191
手動モニタリングツール	192
ファイルシステムの計測	192
計測オブジェクト	192
計測ファイルシステムサイズ	194
計測スループット	195
ストレージクラスサイズの表示	196

CloudWatch によるメトリクスのモニタリング	198
CloudWatch メトリクス	199
CloudWatch メトリクスへのアクセス	205
CloudWatch メトリクスの使用	207
CloudWatch メトリクスでの Metric Math の使用	208
マウント試行の成功と失敗のモニタリング	214
アラームの作成	216
CloudTrail による API コールのログ記録	218
CloudTrail 内の Amazon EFS 情報	218
Amazon EFS ログファイルエントリの概要	219
encrypted-at-rest ファイルシステムの Amazon EFS ログファイルエントリ	226
請求レポートと使用状況レポートの使用	227
コスト配分タグの使用	227
ユーザー定義のコスト配分タグ	228
請求レポート	228
使用状況レポート	229
請求および使用状況レポートの理解	230
使用状況レポートでオペレーションを追跡する	235
パフォーマンス仕様	236
パフォーマンスの概要	236
ストレージクラス	238
パフォーマンスモード	239
スループットモード	239
スループットモードの選択	240
エラスティックスループットモード	240
プロビジョニングされたスループット	241
スループットの切り替えとプロビジョニング量の変更に関する制限	244
パフォーマンスのヒント	244
平均 I/O サイズ	244
高いスループットと IOPS を必要とするワークロードの最適化	244
同時接続	245
リクエストモデル	245
NFS クライアントマウント設定	246
スモールファイルのパフォーマンスを最適化する	246
ディレクトリパフォーマンスの最適化	247
NFS を最適化する read_ahead_kb サイズ	247

パフォーマンスの問題のトラブルシューティング	248
EFS ファイルシステムを作成できない	249
NFS ファイルシステム上の許可されたファイルへのアクセスが拒否されました	250
Amazon EFS コンソールにアクセスするときにエラーが発生しました	250
Amazon EC2 インスタンスがハングする	251
大量のデータを書き込みしているアプリケーションがハングする	251
多くのファイルを並列に開くとパフォーマンスが低下する	251
カスタム NFS 設定による書き込みの遅延	252
Oracle Recovery Manager を使用したバックアップの作成に時間がかかる	253
AMI とカーネルの問題のトラブルシューティング	253
chown ができない	254
クライアントのバグのためにファイルシステムが繰り返し操作を実行し続ける	254
デッドロッククライアント	254
大きなディレクトリ内のファイルの一覧表示に時間がかかる	255
データのバックアップとレプリケーション	256
ファイルシステムのバックアップ	256
が Amazon EFS と AWS Backup 連携する方法	257
必要な IAM 許可	259
バックアップのパフォーマンス	259
自動バックアップの管理	260
ファイルシステムのレプリケート	261
コスト	263
レプリケーションのパフォーマンス	263
必要な IAM アクセス許可	263
新しいファイルシステムへのレプリケーションの設定	264
既存のファイルシステムへのレプリケーションの設定	269
AWS アカウント間のレプリケーション	275
レプリケーションの詳細の表示	280
レプリケーション設定の削除	284
レプリカの使用	286
データの保護	288
データの暗号化	289
AWS KMS	290
保管中のデータの暗号化	293
Encrypting data in transit	294
暗号化のトラブルシューティング	296

Identity and Access Management	298
対象者	299
アイデンティティを使用した認証	300
ポリシーを使用したアクセスの管理	303
Amazon Elastic File System で IAM が機能する仕組み	306
アイデンティティベースのポリシーの例	313
リソースベースのポリシーの例	317
AWS マネージドポリシー	322
Amazon EFS でのタグの使用	328
Amazon EFS のサービスリンクロールの使用	331
トラブルシューティング	335
ファイルシステムのデータアクセスの制御	337
デフォルトのファイルシステムポリシー	338
クライアントの EFS アクション	338
クライアントの EFS 条件キー	339
ファイルシステムポリシーの例	340
ネットワークアクセスの制御	340
VPC セキュリティグループの使用	340
VPCエンドポイントの使用	343
NFS レベルのユーザー、グループ、およびアクセス許可	345
ファイルおよびディレクトリのアクセス許可	346
EFS ファイルシステムのユースケースとアクセス許可の例	346
ファイルシステム内のファイルとディレクトリに対するユーザー ID およびグループ ID の アクセス許可	348
ルートスカッシュなし	349
アクセス許可のキャッシュ	352
ファイルシステムオブジェクトの所有権の変更	352
EFS アクセスポイント	352
アクセスポイントの使用	352
ユーザーアイデンティティ ID の強制	353
ルートディレクトリの強制	354
IAM ポリシーでのアクセスポイントの使用	356
ファイルシステムへのパブリックアクセスのブロック	358
AWS Transfer Familyを使用したパブリックアクセスのブロック	358
「パブリック」の意味	359
コンプライアンス検証	361

耐障害性	362
ネットワークの隔離	363
クォータ	364
引き上げることができる Amazon EFS のクォータ	364
クォータ引き上げのリクエスト	367
変更できない Amazon EFS リソースクォータ	367
NFS クライアントのクォータ	369
Amazon EFS ファイルシステムのクォータ	370
サポートされていない NFSv4.0 および 4.1 機能	371
追加の考慮事項	372
クォータ関連のファイル操作エラーのトラブルシューティング	373
「Disk quota exceeded」エラーでコマンドが失敗する	373
「I/O error」でコマンドが失敗する	373
「File name is too long」エラーでコマンドが失敗する	374
「File not found」エラーでコマンドが失敗する	374
「Too many links」エラーでコマンドが失敗する	375
「File too large」エラーでコマンドが失敗する	375
Amazon EFS API	376
API エンドポイント	376
IPv4 エンドポイント	377
デュアルスタック (IPv4 および IPv6) エンドポイント	377
エンドポイントの指定	378
API バージョン	378
関連トピック	378
Amazon EFS のクエリ API リクエスト率の使用	379
ポーリング	379
再試行またはバッチ処理	379
スリープ間隔の計算	379
アクション	379
CreateAccessPoint	382
CreateFileSystem	390
CreateMountTarget	406
CreateReplicationConfiguration	421
CreateTags	427
DeleteAccessPoint	430
DeleteFileSystem	432

DeleteFileSystemPolicy	436
DeleteMountTarget	439
DeleteReplicationConfiguration	443
DeleteTags	446
DescribeAccessPoints	449
DescribeAccountPreferences	454
DescribeBackupPolicy	457
DescribeFileSystemPolicy	460
DescribeFileSystems	464
DescribeLifecycleConfiguration	470
DescribeMountTargets	474
DescribeMountTargetSecurityGroups	480
DescribeReplicationConfigurations	484
DescribeTags	488
ListTagsForResource	493
ModifyMountTargetSecurityGroups	497
PutAccountPreferences	501
PutBackupPolicy	504
PutFileSystemPolicy	507
PutLifecycleConfiguration	513
TagResource	521
UntagResource	525
UpdateFileSystem	528
UpdateFileSystemProtection	536
データ型	540
AccessPointDescription	541
BackupPolicy	544
CreationInfo	545
Destination	547
DestinationToCreate	550
FileSystemDescription	554
FileSystemProtectionDescription	559
FileSystemSize	560
LifecyclePolicy	562
MountTargetDescription	564
PosixUser	568

ReplicationConfigurationDescription	570
ResourceIdPreference	572
RootDirectory	573
Tag	575
ドキュメント履歴	576
.....	dciv

Amazon Elastic File System とは

Amazon Elastic File System (Amazon EFS) は、サーバーレスで伸縮自在なファイルストレージを提供するため、ストレージ容量およびパフォーマンスのプロビジョニングや管理を行うことなくファイルデータを共有できます。Amazon EFS は、アプリケーションを中断することなく、ファイルの追加や削除に伴って自動的に伸縮し、ペタバイト規模までオンデマンドで拡張できるように構築されています。Amazon EFS はシンプルなウェブサービスインターフェイスを提供しているため、ファイルシステムをすばやく簡単に作成、設定できます。このサービスでは、ユーザーに代わってすべてのファイルストレージインフラストラクチャを管理するため、複雑なデプロイ、パッチ適用、および複雑なファイルシステム設定の保守を行う必要がありません。

Amazon EFS はネットワークファイルシステムバージョン 4 (NFSv4.1 および NFSv4.0) プロトコルをサポートするので、現在お使いのアプリケーションやツールも Amazon EFS とシームレスに動作します。Amazon EFS は、Amazon EC2、Amazon ECS、Amazon EKS、など AWS Lambda、ほとんどのタイプの Amazon Web Services コンピューティングインスタンスでアクセスできます AWS Fargate。

このサービスは拡張性と可用性に優れ、高い耐久性で設計されています。Amazon EFS では、可用性と耐久性の二重を満たすために、以下のファイルシステムのタイプを用意しています。

- リージョン (推奨) - リージョンファイルシステム (推奨) は、同じ AWS リージョン内で地理的に分離された複数のアベイラビリティゾーンにデータを冗長的に保存します。複数のアベイラビリティゾーンにデータを保存すると、の 1 つ以上のアベイラビリティゾーン AWS リージョンが使用できない場合でも、データを継続的に利用できます。
- 1 ゾーン - 1 ゾーンファイルシステムは、単一のアベイラビリティゾーン内にデータを保存します。1 つのアベイラビリティゾーンにデータを保存することで、データの継続的な可用性が提供されます。ただし、アベイラビリティゾーンの全体または一部に損失や破損が生じると、このタイプのファイルシステムに保存されているデータは失われる可能性があります。

ファイルシステムのタイプの詳細については、「[EFS ファイルシステムのタイプ](#)」を参照してください。

Amazon EFS は、さまざまなワークロードに必要なスループット、IOPS、低レイテンシーを実現するように設計されています。EFS ファイルシステムはペタバイト規模で拡張でき、高レベルのスループットを促進して、コンピューティング インスタンスからデータへの大規模な並列アクセスを可能にします。ほとんどのワークロードでは、デフォルトのモード (汎用パフォーマンスモードとエラスティックスループットモード) を使用することをお勧めします。

- 汎用 - 汎用パフォーマンスモードは、ウェブ配信環境、コンテンツ管理システム、ホームディレクトリ、一般的なファイルサービスなど、レイテンシーの影響を受けやすいアプリケーションに最適です。
- エラスティック - エラスティックスループットモードは、ワークロードアクティビティのニーズに合わせてスループットのパフォーマンスを自動的にスケールアップまたはスケールダウンするように設計されています。

EFS のパフォーマンスモードとスループットモードの詳細については、「[Amazon EFS のパフォーマンス仕様](#)」を参照してください。

Amazon EFS は、強いデータ整合性とファイルのロックなどのファイルシステムアクセスのセマンティクスが提供されます。詳細については、「[Amazon EFS のデータ整合性](#)」を参照してください。Amazon EFS では、Portable Operating System Interface (POSIX) アクセス許可を通じてファイルシステムへのアクセスを制御することができます。詳細については、「[Amazon EFS でのデータの保護](#)」を参照してください。

Amazon EFS は、セキュリティおよびコンプライアンス要件を満たすための認証、認可、および暗号化の各機能をサポートしています。Amazon EFS は、ファイルシステムの暗号化の方法として、転送時の暗号化と保管時の暗号化の 2 つをサポートしています。EFS ファイルシステムを作成するときに、保管時の暗号化を有効にできます。これを行うと、データとメタデータはすべて暗号化されます。伝送中の暗号化は、ファイルシステムをマウントする際に有効にできます。Amazon EFS への NFS クライアントアクセスは、AWS Identity and Access Management (IAM) ポリシーとセキュリティグループなどのネットワークセキュリティポリシーの両方によって制御されます。詳細については[Amazon EFS でのデータの暗号化](#)、[Amazon EFS のためのアイデンティティとアクセス管理](#)、および[NFS クライアントの EFS ファイルシステムへのネットワークアクセスの制御](#)を参照してください。

Note

Microsoft Windows ベースの Amazon EFS インスタンスでの Amazon EC2 の使用はサポートされていません。

Amazon EFS を初めてお使いになる方向けの情報

Amazon EFS を初めて使用する方には、次のセクションを順に読むことをお勧めします。

1. Amazon EFS の製品と料金の概要については、[Amazon EFS](#) を参照してください。

2. Amazon EFS に関する技術的な概要については、[Amazon EFS の仕組み](#) を参照してください。
3. 「[入門](#)」の演習を試してください。

Amazon EFS の詳細情報について、次のトピックでそのサービスを詳しく説明します。

- [EFS リソースの作成と管理](#)
- [EFS ファイルシステムの管理](#)
- [Amazon EFS API](#)

Amazon EFS の仕組み

Amazon Elastic File System (EFS) は、シンプルでサーバーレス、かつ全自動の伸縮自在なファイルシステムを提供します。Amazon EFS を使用すると、ファイルシステムを作成し、そのファイルシステムを Amazon EC2 インスタンスにマウントし、ファイルシステムとの間でデータの読取りおよび書き込みを行うことができます。ネットワークファイルシステムバージョン 4.0 および 4.1 (NFSv4) プロトコルを使用して、仮想プライベートクラウド (VPC) に EFS ファイルシステムをマウントできます。最新の Amazon Linux、Amazon Linux 2、Red Hat、Ubuntu、macOS Big Sur の AMI に搭載されている現行世代の Linux NFSv4.1 クライアントを、Amazon EFS マウントヘルパーと共に使用することをお勧めします。手順については、「[Amazon EFS クライアントの手動インストール](#)」を参照してください。

このプロトコルをサポートする Amazon EC2 Linux Amazon Machine Images (AMI) のリストについては、「[NFS サポート](#)」を参照してください。一部の AMI では、Amazon EC2 インスタンスにファイルシステムをマウントするために NFS クライアントをインストールする必要があります。手順については、「[NFS クライアントをインストールする](#)」を参照してください。

複数の NFS クライアントから EFS ファイルシステムに同時にアクセスできるため、単一の接続を超えてスケールするアプリケーションはファイルシステムにアクセスできます。同じ内の複数のアベイラビリティゾーンで実行されている Amazon EC2 およびその他の AWS コンピューティングインスタンス AWS リージョン は、ファイルシステムにアクセスできるため、多くのユーザーが共通のデータソースにアクセスして共有できます。

EFS ファイルシステムを作成 AWS リージョン できる のリストについては、「」を参照してください [Amazon Web Services 全般のリファレンス](#)。

VPC 内の EFS ファイルシステムにアクセスするには、VPC に 1 つ以上のマウントターゲットを作成します。マウントターゲットは、EFS ファイルシステムをマウントできる NFSv4 EFS エンドポ

イントの IP アドレスを提供します。Domain Name Service (DNS) 名を使用してファイルシステムをマウントします。そうすると EC2 インスタンスと同じアベイラビリティゾーンのエFS マウントターゲットの IP アドレスとして解決されます。AWS リージョンのアベイラビリティゾーンごとに 1 つのマウントターゲットを作成できます。VPC のアベイラビリティゾーンに複数のサブネットがある場合、サブネットの 1 つにマウントターゲットを作成します。次に、アベイラビリティゾーンのすべての EC2 インスタンスがそのマウントターゲットを共有します。

 Note

EFS ファイルシステムは、一度に 1 つの VPC にのみマウントターゲットを持つことができます。

マウントターゲットは高い可用性を実現できるように設計されています。高可用性と他のアベイラビリティゾーン (AZ) へのフェイルオーバーを設計する場合、各 AZ のマウントターゲットの IP アドレスと DNS は静的ですが、これらは複数のリソースによってバックアップされる冗長コンポーネントであることに注意してください。マウントターゲットの詳細については、「」を参照してください [マウントターゲットの管理](#)。

DNS 名を使用してファイルシステムをマウントしたら、このファイルシステムを他の POSIX 準拠のファイルシステムと同じように使用します。NFS レベルのアクセス権限と関連する考慮事項の詳細については、「[ネットワークファイルシステム \(NFS\) レベルのユーザー、グループ、およびアクセス許可](#)」を参照してください。

AWS Direct Connect または を使用して Amazon VPC に接続すると、EFS ファイルシステムをオンプレミスのデータセンターサーバーにマウントできます AWS VPN。EFS ファイルシステムをオンプレミスサーバーにマウントすると、データセットを EFS に移行したり、クラウドでのバーストシナリオを有効にしたり、オンプレミスデータを Amazon EFS にバックアップしたりできます。

以下では、Amazon EFS が他のサービスとどのように連携するかについて説明します。

トピック

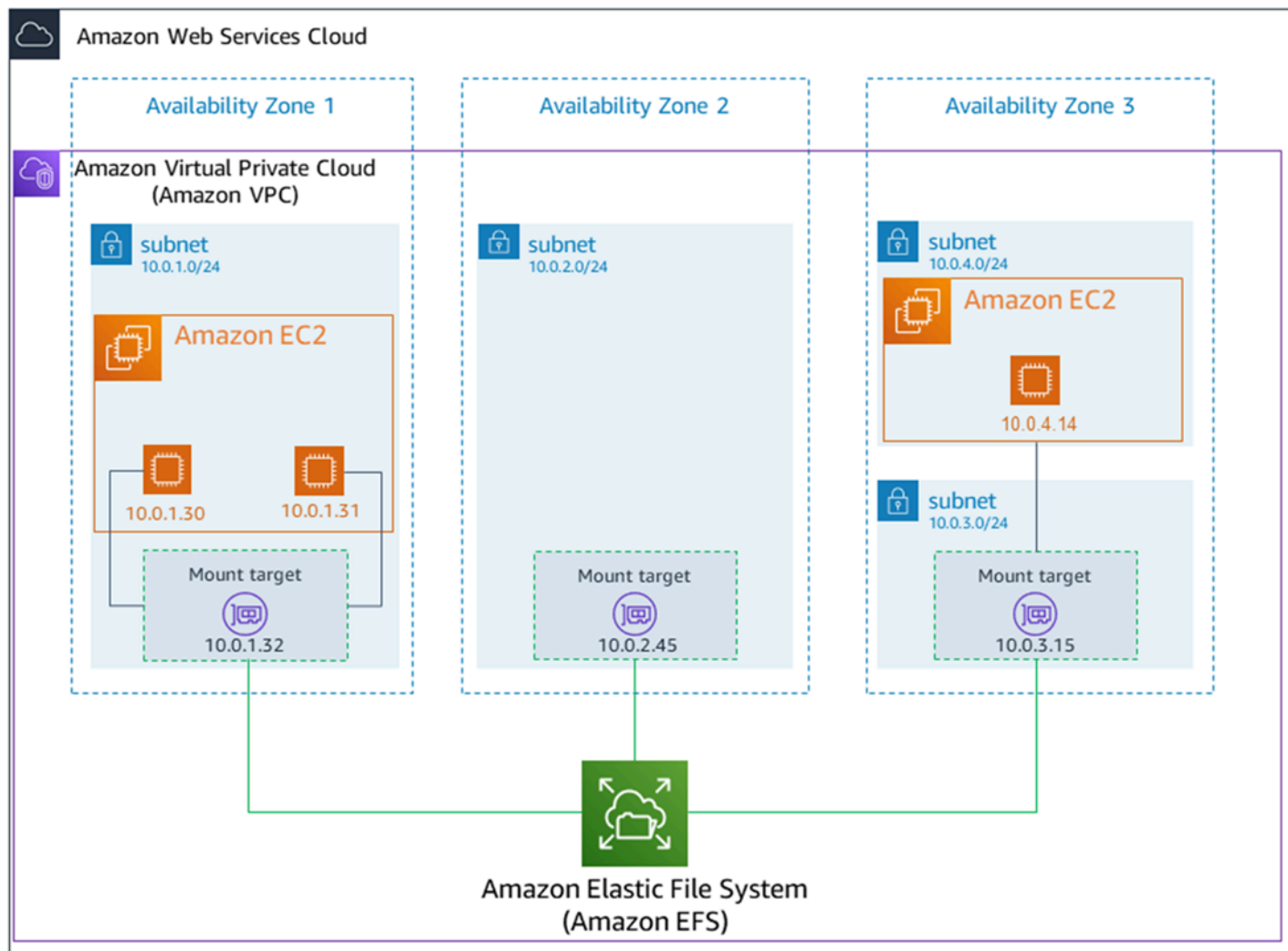
- [Amazon EFS と Amazon EC2 の連携方法](#)
- [Amazon EFS AWS Direct Connect と AWS Managed VPN の連携方法](#)
- [Amazon EFS と の連携方法 AWS Backup](#)

Amazon EFS と Amazon EC2 の連携方法

このセクションでは、Amazon EFS のリージョナルファイルシステムと 1 ザーンファイルシステムが Amazon VPC 内の EC2 インスタンスにマウントされる方法について説明します。

リージョン EFS ファイルシステム

次の図は、AWS リージョン内の複数のアベイラビリティゾーン用に設定された Amazon EFS ファイルシステムにアクセスする、複数の EC2 インスタンスを示しています。

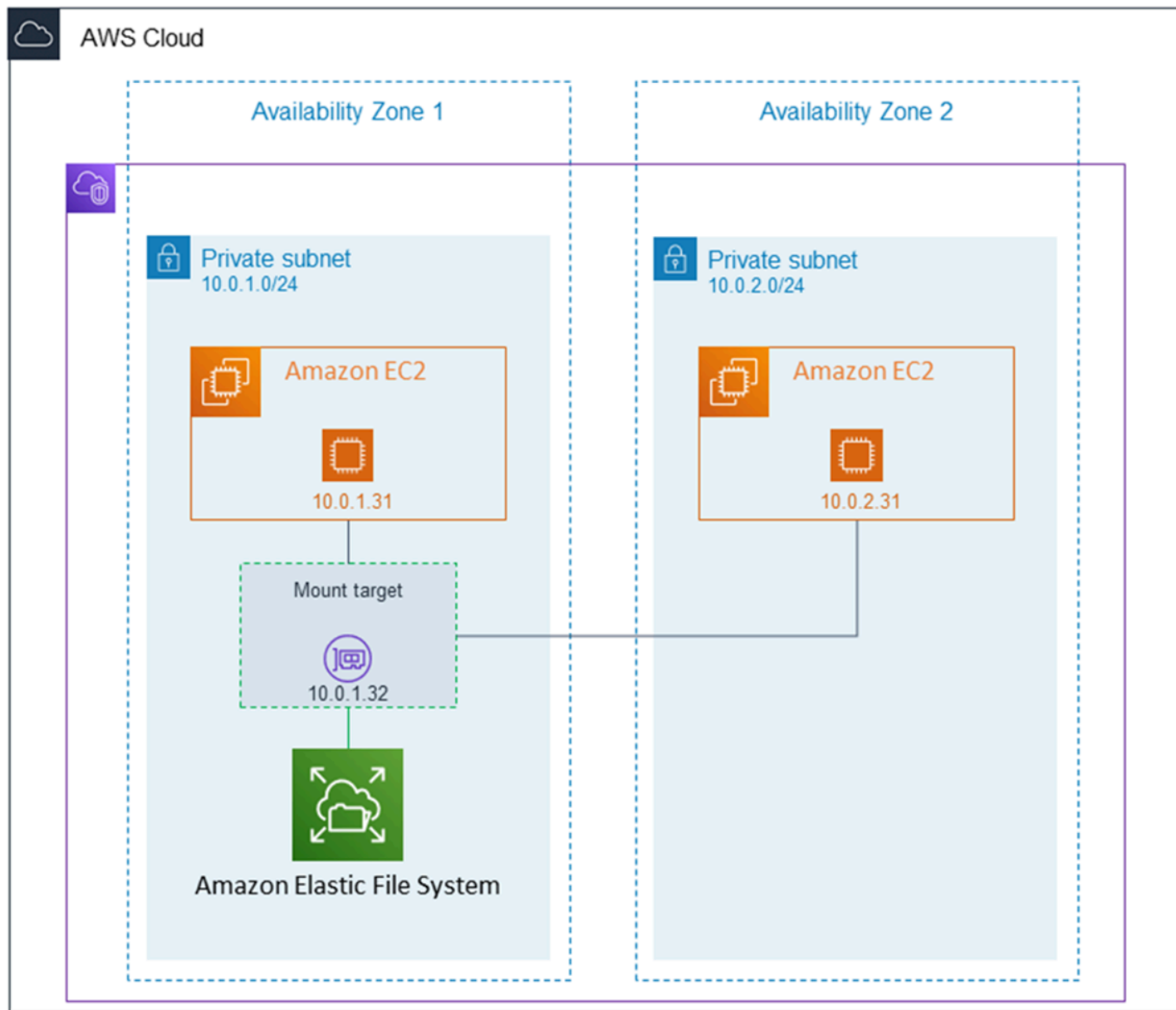


この図では、仮想プライベートクラウド (VPC) に 3 つのアベイラビリティゾーンがあります。ファイルシステムがリージョン別であるため、各アベイラビリティゾーンにマウントターゲットが作成されました。パフォーマンスとコストの観点から、同じアベイラビリティゾーンのマウントターゲットからファイルシステムにアクセスすることをお勧めします。アベイラビリティゾーンの 1 つには 2 つのサブネットがあります。ただし、マウントターゲットは 1 つのサブネットのみに作

成されます。詳細については、「[EFS マウントヘルパーを使用した EFS ファイルシステムのマウント](#)」を参照してください。

1 ゾーンファイルシステム

次の図は、単一の AWS リージョン内の異なるアベイラビリティゾーンから 1 ゾーンファイルシステムにアクセスする、複数の EC2 インスタンスを示しています。



この図では、VPC には 2 つのアベイラビリティゾーンがあり、それぞれに 1 つのサブネットがあります。ファイルシステムのタイプが 1 ゾーンなので、マウントターゲットを 1 つだけ持つことができます。パフォーマンスとコストを向上させるために、マウントする EC2 インスタンスと同じア

ベイラビリティーゾーンにあるマウントターゲットからファイルシステムにアクセスすることをお勧めします。

この例では、us-west-2c アベイラビリティーゾーンの EC2 インスタンスは、別のアベイラビリティーゾーンのマウントターゲットにアクセスするための EC2 データアクセス料金を支払います。詳細については、「[1 ゾーンファイルシステムをマウントする](#)」を参照してください。

Amazon EFS AWS Direct Connect と AWS Managed VPN の連携方法

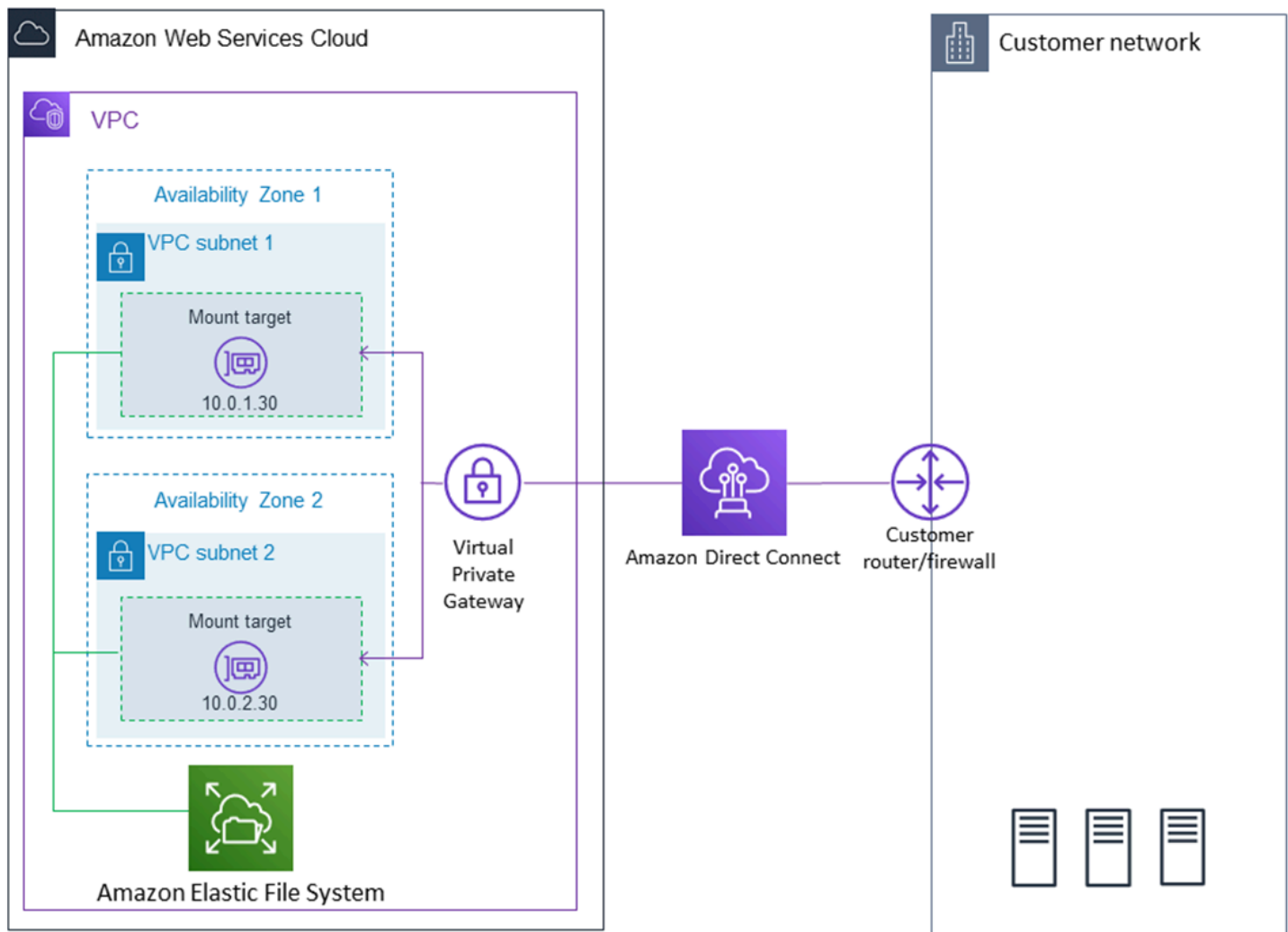
オンプレミスサーバーにマウントされた Amazon EFS ファイルシステムを使用することで、Amazon EFS ファイルシステムで AWS クラウド ホストされている にオンプレミスデータを移行できます。バーストを活用することもできます。つまり、オンプレミスサーバーから Amazon EFS にデータを移動し、Amazon VPC の Amazon EC2 インスタンスのフリート上で分析することができます。その後、結果をファイルシステムに永続的に保存するか、結果をオンプレミスサーバーに戻すことができます。

オンプレミスサーバーで Amazon EFS を使用する場合は、次の点に注意してください。

- オンプレミスサーバーには Linux ベースのオペレーティングシステムが必要です。Linux カーネルバージョン 4.0 以降をお勧めします。
- わかりやすいように、DNS 名ではなくマウントターゲットの IP アドレスを使用して、Amazon EFS ファイルシステムをオンプレミスサーバーにマウントすることをお勧めします。

Amazon EFS ファイルシステムのオンプレミスアクセスに、追加料金は必要ありません。Amazon VPC AWS Direct Connect への接続には料金がかかります。詳細については、[AWS Direct Connect 料金表](#)を参照してください。

次の図は、オンプレミスから Amazon EFS ファイルシステムにアクセスする方法 (オンプレミスサーバーにファイルシステムがマウントされている) の例を示しています。



オンプレミスサーバーと VPC 間の AWS Direct Connect 接続を使用して、そのマウントターゲットのサブネットに到達できる場合は、VPC 内の任意のマウントターゲットを使用できます。オンプレミスサーバーから Amazon EFS にアクセスするには、オンプレミスサーバーから NFS ポート (2049) へのインバウンドトラフィックを許可するために、マウントターゲットセキュリティグループにルールを追加する必要があります。詳しい手順などの詳細については、「[前提条件](#)」を参照してください。

Amazon EFS と の連携方法 AWS Backup

ファイルシステムの包括的なバックアップ実装のために、で Amazon EFS を使用できます AWS Backup。AWS Backup は、クラウドとオンプレミスのサービス間でデータバックアップを簡単に一元化および自動化できるフルマネージドバックアップ AWS サービスです。を使用すると AWS Backup、バックアップポリシーを一元的に設定し、AWS リソースのバックアップアクティビティをモニタリングできます。Amazon EFS は常にバックアップオペレーションよりもファイルシステ

ムオペレーションを優先します。を使用した EFS ファイルシステムのバックアップの詳細については AWS Backup、「」を参照してください[EFS ファイルシステムのバックアップ](#)。

Amazon EFS の機能

以下では、Amazon EFS の機能について説明します。

トピック

- [認証とアクセスコントロール](#)
- [Amazon EFS のデータ整合性](#)
- [EFS ファイルシステムの可用性と耐久性](#)
- [レプリケーション](#)

認証とアクセスコントロール

Amazon EFS コンソールを使用したり、ファイルシステムの作成などの Amazon EFS API リクエストを実行したりするには、有効な認証情報が必要です。さらに、他の EFS および AWS リソースを作成またはアクセスするためのアクセス許可も必要です。

AWS Identity and Access Management (IAM) で作成したユーザーとロールには、リソースを作成またはアクセスするためのアクセス許可を付与する必要があります。権限の詳細については、「[Amazon EFS のためのアイデンティティとアクセス管理](#)」を参照してください。

NFS クライアントの IAM 認可は、Amazon EFS の追加のセキュリティオプションであり、IAM を使用して、大規模なネットワークファイルシステム (NFS) クライアントのアクセス管理を簡素化できます。NFS クライアントの IAM 認可では、IAM を使用して、固有のスケラブルな方法で EFS ファイルシステムへのアクセスを管理できます。NFS クライアントの IAM 認可は、クラウド環境にも最適化されています。NFS クライアントの IAM 認可を使用する方法の詳細については、「[IAM を使用してファイルシステムへのアクセスを制御する](#)」を参照してください。

Amazon EFS のデータ整合性

Amazon EFS は、アプリケーションが NFS に求める close-to open 整合性セマンティクスを提供します。

Amazon EFS では、以下の状況において、リージョンファイルシステムの書き込みオペレーションがアベイラビリティゾーン間で永続的に保存されます。

- アプリケーションは、同期書き込みオペレーションを実行します (たとえば、open フラグで O_DIRECT Linux コマンドを使用するか、または fsync Linux コマンドを使用します)。
- アプリケーションがファイルを閉じます。

Amazon EFS は、アクセスパターンによっては、close-to-open セマンティクスよりも強力な整合性をもたらします。同期データアクセスを実行し、非追加書き込みを実行するアプリケーションは、データアクセスのための書き込み後の読み取り整合性を保ちます。

ファイルロック

NFS クライアントアプリケーションでは、Amazon EFS ファイルの読み取り/書き込み操作に NFS バージョン 4 のファイルロック (バイト範囲ロックを含む) を使用できます。

Amazon EFS がファイルをロックする方法については、次の点に注意してください。

- Amazon EFS はアドバイザリロックのみをサポートしており、読み取り/書き込みオペレーションでは実行前にロックの競合をチェックしません。例えば、アトミックオペレーションによるファイル同期の問題を回避するには、アプリケーションが NFS のセマンティクス (オープンに近い整合性など) を認識している必要があります。
- 1 つの特定のファイルが、接続されているすべてのインスタンスとそのファイルにアクセスしているユーザー全体で持つことのできるロックの数は、最大 512 個までです。

EFS ファイルシステムの可用性と耐久性

このセクションでは、Amazon Elastic File System (Amazon EFS) ファイルシステムにおけるファイルシステムのタイプとストレージクラスのオプションについて説明します。

EFS ファイルシステムのタイプ

Amazon EFS には、リージョンファイルシステムタイプと 1 ザーンファイルシステムタイプがあります。

- リージョン - リージョンファイルシステム (推奨) は、同じ AWS リージョン内で地理的に分離された複数のアベイラビリティゾーンにデータを冗長的に保存します。複数のアベイラビリティゾーンにデータを保存すると、の 1 つ以上のアベイラビリティゾーン AWS リージョン が使用できない場合でも、データの継続的な可用性が提供されます。
- 1 ザーン - 1 ザーンファイルシステムは、単一のアベイラビリティゾーン内にデータを保存します。1 つのアベイラビリティゾーンにデータを保存することで、データの継続的な可用性が提供

されます。ただし、アベイラビリティゾーン全体の全体または一部に損失や破損が生じると、このタイプのファイルシステムに保存されているデータは失われる可能性があります。

万一、AWS アベイラビリティゾーンの全部または一部に損失または損傷が発生した場合、1 ゾーンストレージクラスのデータは失われる可能性があります。例えば、火災や水害などの事象が発生すると、データが失われる可能性があります。このような事象は別として、1 ゾーンストレージクラスは、リージョン別ストレージクラスと同様のエンジニアリング設計を採用して、独立したディスク、ホスト、ラックレベルの障害からオブジェクトを保護し、それぞれ 99.999999999% のデータ耐久性を実現するように設計されています。

データ保護を強化するために、Amazon EFS は 1 ゾーンファイルシステムを で自動的にバックアップします AWS Backup。ファイルシステムのバックアップは、内の任意の運用アベイラビリティゾーンに復元することも AWS リージョン、別のアベイラビリティゾーンに復元することもできます AWS リージョン。を使用して作成および管理される EFS ファイルシステムのバックアップ AWS Backup は、3 つのアベイラビリティゾーンにレプリケートされ、耐久性を考慮して設計されています。詳細については、[「レジリエンス AWS Backup」](#)を参照してください。

Note

1 ゾーンファイルシステムは特定のアベイラビリティゾーンでのみ使用できます。1 ゾーンファイルシステムを使用できるアベイラビリティゾーンの一覧表については、[「1 ゾーンファイルシステムでサポートされるアベイラビリティゾーン」](#)を参照してください。

次の表では、可用性、耐久性、その他の考慮すべき事項を含めて、ファイルシステムのタイプを比較しています。

ファイルシステムのタイプ	対象	耐久性 (設計対象)	可用性	アベイラビリティゾーン	その他の考慮事項
リージョン別	高い耐久性と可用性を必要とするデータ。	99.999999999% (11 9s)	99.99%	>=3	なし

ファイルシステムのタイプ	対象	耐久性 (設計対象)	可用性	アベイラビリティゾーン	その他の考慮事項
1 ゾーン	高い耐久性と可用性を必要としないデータ。	99.999999 999% (11 9s)	99.99%	1	アベイラビリティゾーンの損失においては回復性なし

1 ゾーンファイルシステムでサポートされるアベイラビリティゾーン

1 ゾーンファイルシステムは特定のアベイラビリティゾーンでのみ使用できます。次の表に、1 ゾーンファイルシステムを使用できる各アベイラビリティゾーンの AWS リージョン と AZ IDs を示します。アカウントのアベイラビリティゾーンへの AZ IDs のマッピングを確認するには、AWS 「Resource Access Manager ユーザーガイド」の[AWS 「リソースのアベイラビリティゾーン IDs」](#)を参照してください。

1 ゾーンファイルシステムをサポートするアベイラビリティゾーン

AWS リージョン 名前	AWS リージョン コード	サポートされる AZ ID
米国東部 (オハイオ)	us-east-2	use2-az1、use2-az2、use2-az3
米国東部 (バージニア北部)	us-east-1	use1-az1、use1-az2、use1-az4、use1-az5、use1-az6
米国西部 (北カリフォルニア)	us-west-1	usw1-az1、usw1-az3
米国西部 (オレゴン)	us-west-2	usw2-az1、usw2-az2、usw2-az3、usw2-az4
アフリカ (ケープタウン)	af-south-1	afs1-az1、afs1-az2、afs1-az3

AWS リージョン 名前	AWS リージョン コード	サポートされる AZ ID
アジアパシフィック (香港)	ap-east-1	ape1-az1、ape1-az2、ape1-az3
アジアパシフィック (ムンバイ)	ap-south-1	aps1-az1、aps1-az2、aps1-az3
アジアパシフィック (大阪)	ap-northeast-3	apne3-az1、apne3-az2、apne3-az3
アジアパシフィック (ソウル)	ap-northeast-2	apne2-az1、apne2-az2、apne2-az3
アジアパシフィック (シンガポール)	ap-southeast-1	apse1-az1、apse1-az2
アジアパシフィック (シドニー)	ap-southeast-2	apse2-az1、apse2-az3、apse2-az2
アジアパシフィック (東京)	ap-northeast-1	apne1-az1、apne1-az4
カナダ (中部)	ca-central-1	cac1-az1、cac1-az2
中国 (北京)	cn-north-1	cnn1-az1、cnn1-az2
中国 (寧夏)	cn-northwest-1	cnnw1-az1、cnnw1-az2、cnnw1-az3
欧州 (フランクフルト)	eu-central-1	euc1-az1、euc1-az2、euc1-az3
欧州 (アイルランド)	eu-west-1	euw1-az1、euw1-az2、euw1-az3
欧州 (ロンドン)	eu-west-2	euw2-az1、euw2-az2
欧州 (ミラノ)	eu-south-1	eus1-az1、eus1-az2、eus1-az3

AWS リージョン 名前	AWS リージョン コード	サポートされる AZ ID
欧州 (パリ)	eu-west-3	euw3-az1、euw3-az3
欧州 (ストックホルム)	eu-north-1	eun1-az1、eun1-az2、eun1-az3
中東 (バーレーン)	me-south-1	mes1-az1、mes1-az2、mes1-az3
南米 (サンパウロ)	sa-east-1	sae1-az1、sae1-az2、sae1-az3
AWS GovCloud (米国東部)	us-gov-east-1	usge1-az1、usge1-az2、usge1-az3
AWS GovCloud (米国西部)	us-gov-west-1	usgw1-az1、usgw1-az2、usgw1-az3

EFS ストレージクラス

Amazon EFS では、ユースケースに応じて最も効果的なストレージになるように設計された、さまざまなストレージクラスが提供されています。

- **EFS 標準** — EFS 標準ストレージクラスは、ソリッドステートドライブ (SSD) ストレージを使用して、頻繁にアクセスされるファイルのレイテンシーを最小限に抑えます。新しいファイルシステムデータは、最初に EFS 標準ストレージクラスに書き込まれ、次にライフサイクル管理を使用して EFS 低頻度アクセスストレージクラスおよび EFS アーカイブストレージクラスに階層化できます。
- **EFS 低頻度アクセス (IA)** — コストが最適化されたストレージクラスで、四半期ごとに数回しかアクセスされないデータに向いています。
- **EFS アーカイブ** — コストが最適化されたストレージクラスで、1 年に数回しかアクセスされないデータに向いています。

EFS アーカイブストレージクラスは、エラスティックスループットの EFS ファイルシステムでサポートされます。ファイルシステムのデータがアーカイブストレージクラスに格納されたら、ファイルシステムのスループットをバーストまたはプロビジョンドに更新することはできません。

ストレージクラスを比較する

次の表ではストレージクラスの比較を示しています。各ストレージクラスのパフォーマンスの詳細については、「[Amazon EFS のパフォーマンス仕様](#)」を参照してください。

ストレージクラス	対象	最初のバイトの読み取りレイテンシー	耐久性 (設計対象) ¹	アベイラビリティ SLA	アベイラビリティゾーン	1 ファイルあたりの最低請求料金 ²	最小ストレージ期間
EFS スタンダード	ミリ秒未満の高速レイテンシーパフォーマンスを必要とするアクティブデータ	ミリ秒未満		99.99% (リージョン) 99.9% (1 ゾーン)	=>3 (リージョン)	該当しない	該当しない
EFS 低頻度アクセス	四半期ごとに数回しかアクセスされない非アクティブデータ。	数十ミリ秒	99.999999999% (11 9's)		1 (1 ゾーン)	128 KiB	該当しない
EFS アーカイブ	1 年に数回しかアクセスされない非アクティブデータ	数十ミリ秒		99.9% (リージョン)	=>3 (リージョン)	128 KiB	90 日間

¹ 1 ゾーンファイルシステムは 1 つの AWS アベイラビリティゾーンにデータを保存するため、アベイラビリティゾーン内のデータのすべてのコピーに影響する災害やその他の障害が発生した場合、またはアベイラビリティゾーンが破壊された場合に、これらのタイプのファイルシステムに保存されているデータは失われる可能性があります。

²2023 年 11 月 26 日午後 12 時以降に更新されたライフサイクルポリシーは、128 KiB 未満のファイルを IA クラスに階層化します。Amazon EFS が個々のファイルとメタデータを計測して請求する方法の詳細については、「[Amazon EFS がファイルシステムとオブジェクトのサイズをどのように報告するかについて説明します。](#)」を参照してください。

ストレージクラスの請求

各ストレージクラスでのデータの量に対して、請求が行われます。IA ストレージまたはアーカイブストレージ内のファイルが読み取られるときや、ライフサイクル管理を使用してストレージクラス間を移動するデータに対しても、データアクセス料金が請求されます。AWS の請求書には、各ストレージクラスの容量とファイルシステムのストレージクラスに対して計測されたアクセスが表示されます。詳細はこちら、「[Amazon EFS 料金表](#)」を参照してください。

また、低頻度アクセス (IA) ストレージクラスおよびアーカイブストレージクラスにおけるファイルあたりの最低請求額は 128 KiB です。128 KiB 未満のファイルのサポートは、2023 年 11 月 26 日午後 12:00 (太平洋時間) 以降に更新されたライフサイクルポリシーでのみ利用できます。Amazon EFS が個々のファイルとメタデータを計測して請求する方法の詳細については、「[Amazon EFS がファイルシステムとオブジェクトのサイズをどのように報告するかについて説明します。](#)」を参照してください。

スループットモードによっては、追加の請求に関する考慮事項が適用されます。

- Elastic スループットを使用するファイルシステムの場合、ストレージ料金とは無関係に、ファイルシステムに転送されたメタデータとデータの合計月額料金が請求されます。
- プロビジョニングされたスループットを使用しているファイルシステムの場合は、EFS 標準ストレージクラスにあるデータの量に基づいて、提供されているスループットを超えてプロビジョニングされたスループットの料金が請求されます。
- バーストスループットを使用するファイルシステムの場合、許容されるスループットは、EFS 標準ストレージクラスにのみ保存されているデータ量に基づいて決定されます。

EFS スループットモードの詳細については、「[スループットモード](#)」を参照してください。

Note

を使用してライフサイクル管理が有効な EFS ファイルシステムをバックアップする場合 AWS Backup、データアクセス料金は発生しません。Amazon EFS AWS Backup での詳細については、「」を参照してください[EFS ファイルシステムのバックアップ](#)。

ライフサイクル管理

ファイルシステムがライフサイクル全体にわたってコスト効率良く保存されるように管理するには、ライフサイクル管理を使用します。ライフサイクル管理は、ファイルシステムに定義されているライフサイクル設定に従って、ストレージクラス間でデータを自動的に移行します。ライフサイクル設定は、ファイルシステムデータを別のストレージクラスに移行するタイミングを定義する、一連のライフサイクルポリシーです。詳細については、「[ストレージライフサイクルの管理](#)」を参照してください。

レプリケーション

レプリケーションを使用して、AWS リージョン 任意の で Amazon EFS ファイルシステムのレプリカを作成できます。レプリケーションは、EFS ファイルシステム上のデータとメタデータを、AWS リージョン 選択した で作成された新しい送信先 EFS ファイルシステムに自動的にかつ透過的にレプリケートします。EFS は、ソースとデスティネーションのファイルシステムの同期を自動的に維持します。レプリケーションは継続的であり、分単位の目標復旧時点 (RPO) と目標復旧時間 (RTO) を実現するように設計されています。これらの機能は、コンプライアンスとビジネス継続性の目標を達成するのに役立ちます。詳細については、「[EFS ファイルシステムのレプリケート](#)」を参照してください。

Amazon EFS の開始方法

Amazon Elastic File System (Amazon EFS) を初めて使用する場合、最初の EFS ファイルシステムを作成するには、次の手順に従います。

1. [開始するための前提条件の確認](#)
2. [EFS ファイルシステムの作成と EC2 インスタンスの起動](#)
3. [を使用して EFS ファイルシステムにファイルを転送する AWS DataSync](#)
4. [リソースのクリーンアップと AWS アカウントの保護](#)

前提条件

開始手順を実行する前に、次の要件が満たされていることを確認してください。

- Amazon EC2 のセットアップが完了していて、EC2 インスタンスの起動方法を理解していること。AWS アカウント、管理アクセス権を持つユーザー、キーペア、およびセキュリティグループが必要です。詳細については、「Amazon EC2 ユーザーガイド」の「[Amazon EC2 Linux インスタンスの開始方法](#)」を参照してください。
- Amazon Virtual Private Cloud (Amazon VPC)、EC2、EFS リソースがすべて同じ AWS リージョンにあり、そのリージョン内にデフォルトの VPC があること。デフォルトの VPC がない場合や、新しい VPC から新規または既存のセキュリティグループでファイルシステムをマウントする場合は、「[VPC セキュリティグループの使用](#)」を参照してください。
- デフォルトのセキュリティグループのデフォルトのインバウンドアクセスルールを変更していないこと。

また、AWS Command Line Interface (AWS CLI) コマンドを使用して EFS API コールを行う同様の入門演習を実行することもできます。詳細については、「[チュートリアル: EFS ファイルシステムを作成し、を使用して EC2 インスタンスにマウントする AWS CLI](#)」を参照してください。

EFS ファイルシステムの作成と EC2 インスタンスの起動

この入門演習の前提条件が満たされていることを確認したら、EFS ファイルシステムを作成し、EC2 インスタンスを起動できます。最初の EFS ファイルシステムを開始するために必要なすべての手順を完了する最も簡単な方法は、インスタンスの起動時に EC2 の新しい起動ウィザードを使用することです。

 Note

Microsoft Windows ベースの EC2 インスタンスでは Amazon EFS を使用できません。

EC2 起動ウィザードを使用して EFS ファイルシステムを作成し、EC2 インスタンスを起動するには

EC2 インスタンス起動の作成時に EFS ファイルシステムを作成してマウントする手順については、「[Amazon EC2 ユーザーガイド](#)」の[Amazon EC2 で Amazon EFS](#) を使用する」を参照してください。 Amazon EC2

インスタンスの起動時に EFS ファイルシステムを作成する手順は次のとおりです。

1. 選択したキーペアとネットワーク設定を使用して、Linux オペレーティングシステムで実行される EC2 インスタンスを作成します。
2. 推奨設定を使用して、自動的に EC2 インスタンスにマウントされる共有 EFS ファイルシステムを作成します。
3. EC2 インスタンスを起動して、すぐに EFS ファイルシステムがファイル転送に対応できるようにします。

別の方法として、Amazon EFS コンソールを使用すると、推奨設定またはカスタム設定でファイルシステムを作成できます。AWS CLI と API を使用してファイルシステムを作成することもできます。ファイルシステムを作成できるすべての方法の詳細については、「[EFS ファイルシステムの作成](#)」を参照してください。

を使用して EFS ファイルシステムにファイルを転送する AWS DataSync

EFS ファイルシステムを作成したら、AWS DataSyncを使用して、そのファイルシステムに既存のファイルシステムからファイルを転送できます。DataSync は、インターネットまたは AWS Direct Connectを介して、オンプレミスのストレージシステムと AWS のストレージサービス間でのデータの移動とレプリケーションを簡素化、自動化、高速化するデータ転送サービスです。DataSync は、ファイルデータだけでなく、所有権、タイムスタンプ、アクセス許可などのファイルシステムメタデータも転送できます。

DataSync の詳細については、「[AWS DataSync](#)」を参照してください。

前提条件

EFS ファイルシステムにファイルを転送する前に、以下が用意されていることを確認してください。

- ファイルの転送元として使用できるソース NFS ファイルシステム。NFS バージョン 3、バージョン 4 または 4.1 経由で、この出典システムにアクセスする必要があります。ファイルシステムの例として、オンプレミスのデータセンターに配置されているファイルシステム、セルフマネージド型のクラウド内ファイルシステム、EFS ファイルシステムがあります。
- DataSync を使用する準備ができていること。詳細については、「[AWS DataSync ユーザーガイド](#)」の「[の開始方法 AWS DataSync](#)」を参照してください。

を使用して EFS ファイルシステムにファイルを転送するには AWS DataSync

DataSync を使用して EFS ファイルシステムにファイルを転送する手順については、「[AWS DataSync ユーザーガイド](#)」の「[AWS DataSyncを使ったデータ転送](#)」を参照してください。

DataSync を使用して EFS ファイルシステムにファイルを転送する手順は次のとおりです。

1. EC2 インスタンスに接続します。詳細については、「[Amazon EC2 ユーザーガイド](#)」の「[EC2 インスタンスに接続する](#)」を参照してください。
2. エージェントを環境にダウンロードし、デプロイしてアクティブ化します。
3. ソースと宛先の場所を作成して設定します。
4. タスクを作成し、設定します。
5. タスクを実行して、ソースから宛先にファイルを転送します。

リソースのクリーンアップと AWS アカウントの保護

この入門演習が完了したら、以下の手順を実行してリソースをクリーンアップし、AWS アカウントを保護します。

リソースをクリーンアップし、アカウントを保護するには

1. EC2 インスタンスに接続します。詳細については、「[Amazon EC2 ユーザーガイド](#)」の「[EC2 インスタンスに接続する](#)」を参照してください。
2. 次のコマンドで、EFS ファイルシステムをアンマウントします。


```
$ sudo umount efs
```

3. Amazon Elastic File System コンソールの <https://console.aws.amazon.com/efs/> を開いてください。
4. 入門演習の最初の手順で作成した EFS ファイルシステムを削除します。
 - a. ファイルシステムのリストから削除する EFS ファイルシステムを選択します。
 - b. [Actions] (アクション) で、[Delete file system] (ファイルシステムの削除) を選択します。
 - c. [ファイルシステムを完全に削除] ダイアログボックスで、削除する EFS ファイルシステムのファイルシステム ID を入力して、[ファイルシステムの削除] を選択します。
5. この入門演習で起動した EC2 インスタンスを終了します。手順については、「AWS IAM Identity Center ユーザーガイド」の「[EC2 インスタンスを終了する](#)」を参照してください。
6. この入門演習用にセキュリティグループを作成した場合は削除します。手順については、「AWS IAM Identity Center ユーザーガイド」の「[セキュリティグループを削除する](#)」を参照してください。

**Warning**

VPC のデフォルトのセキュリティグループを削除しないでください。

Amazon EFS クライアントの手動インストール

Amazon EFS クライアント (amazon-efs-utils) をインストールすることをお勧めします。これは、Amazon EFS 用のオープンソースのツールコレクションです。Amazon EFS クライアントには、EFS ファイルシステムのマウントを簡素化するために役立つプログラムであるマウントヘルパーが含まれています。クライアントでは、Amazon CloudWatch を使用して EFS ファイルシステムのマウントステータスをモニタリングすることもできます。さらに、Amazon EFS ファイルシステムで転送中のデータを簡単に暗号化できるようにするツールも含まれています。

[サポートされているディストリビューション](#) を実行している Amazon EC2 (EC2) インスタンスに Amazon EFS クライアントを手動でインストールできます。EC2 サポートされている特定のオペレーティングシステムでは、パッケージを自動的にインストールまたは更新 AWS Systems Manager するように設定することもできます。で利用できるディストリビューションのリストについては AWS Systems Manager、「」を参照してください [Systems Manager がサポートするオペレーティングシステム](#)。

Important

すべての機能へのアクセスを確保するために amazon-efs-utils、常に最新バージョンの使用することをお勧めします。例えば、IPv6 アドレスを使用したマウントはバージョン 2.3 以降ではサポートされていますが、以前のバージョンではサポートされていません。

トピック

- [EFS ツールの依存関係](#)
- [サポートされているディストリビューション](#)
- [Amazon EFS クライアントの手動インストール](#)
- [を使用して Amazon EFS クライアントを自動的にインストールまたは更新する AWS Systems Manager](#)
- [botocore のインストールとアップグレード](#)
- [stunnel のアップグレード](#)
- [FIPS モードの有効化](#)

EFS ツールの依存関係

amazon-efs-utils パッケージをインストールする場合、amazon-efs-utils に次の依存関係が存在し、インストールされています。

- NFS クライアント
 - nfs-utils RHEL、CentOS、Amazon Linux、および Fedora ディストリビューション用
 - Debian および Ubuntu ディストリビューション用 nfs-common
- ネットワークリレー (stunnel パッケージ、バージョン 4.56 以降)
- Python (バージョン 3.4 以降)
- OpenSSL 1.0.2 以降

Note

デフォルトでは、EFS マウントヘルパーで Transport Layer Security (TLS) を使用する場合、マウントヘルパーは証明書ホスト名のチェックを強制します。EFS マウントヘルパーは、stunnel プログラムを使用して TLS 機能を提供します。Linux のバージョンによっては、これらの TLS 機能をサポートする stunnel のバージョンがデフォルトに含まれていない場合があります。このような Linux バージョンのいずれかを使用している場合は、TLS を使用した EFS ファイルシステムのマウントに失敗します。

amazon-efs-utils パッケージのインストール後に stunnel をアップグレードしてください。
「[stunnel のアップグレード](#)」を参照してください。

AWS Systems Manager を使用して Amazon EFS クライアントを管理し、EC2 インスタンスに amazon-efs-utils パッケージをインストールまたは更新するために必要なタスクを自動化できます。詳細については、「[を使用して Amazon EFS クライアントを自動的にインストールまたは更新する AWS Systems Manager](#)」を参照してください。

暗号化の問題については、「[暗号化のトラブルシューティング](#)」を参照してください。

サポートされているディストリビューション

Amazon EFS クライアントは、以下の Linux および Mac ディストリビューションに対して検証されています。

ディストリビューション	パッケージタイプ	init system
Amazon Linux 2023 (AL2023)	rpm	systemd
Amazon Linux 2 (AL2)	rpm	systemd
CentOS 8	rpm	systemd
Amazon Linux 1 (AL1) 2017.09	rpm	upstart
 Note AL1 AMI は 2023 年 12 月 31 日にend-of-lifeとなり、2024 年 4 月以降にリリースされたamazon-efs-utils パッケージ (バージョン 2.0 以降) ではサポートされていません。		
Debian 11	deb	systemd
Fedora 29-32	rpm	systemd
macOS Big Sur		launchd
macOS Monterey		launchd
macOS Ventura		launchd
macOS Sonoma		launchd
OpenSUSE Leap、タンブルウィード	rpm	systemd
Oracle8	rpm	systemd
Red Hat Enterprise Linux (RHEL) 7、8、9	rpm	systemd

ディストリビューション	パッケージタイプ	init system
SUSE Linux Enterprise Server (SLES) 12, 15	rpm	systemd
Ubuntu 16.04 LTS、18.04 LTS、20.04 LTS、22.04 LTS	deb	systemd

パッケージが検証されたサポートされているディストリビューションの完全なリストについては、Github [amazon-efs-utils](#) [README](#) の「」を参照してください。

Amazon EFS クライアントの手動インストール

Amazon EFS クライアントは、Amazon EC2 (EC2) Linux インスタンスと、macOS Big Sur、macOS Monterey、macOS Ventura を実行している EC2 Mac インスタンスに手動でインストールできます。Amazon EFS クライアントをサポートするディストリビューションの一覧については、「[サポートされているディストリビューション](#)」を参照してください。

以下のセクションでは、サポートされているオペレーティングシステムでの手順を説明します。

トピック

- [Amazon EC2 Linux インスタンスでの Amazon EFS クライアントのインストール](#)
- [他の Linux ディストリビューションで amazon-efs-utils パッケージをインストールする](#)
- [macOS Big Sur、macOS Monterey、または macOS Ventura を実行している EC2 Mac インスタンスに Amazon EFS クライアントをインストールする](#)

Amazon EC2 Linux インスタンスでの Amazon EFS クライアントのインストール

Amazon EC2 Linux インスタンスでは、次の場所から amazon-efs-utils パッケージをインストールできます。

- Amazon Linux の Amazon マシンイメージ (AMI) パッケージリポジトリ。以下では、AMI パッケージリポジトリから amazon-efs-utils パッケージをインストールする手順を説明しています。

- AWS [efs-utils](#) GitHub リポジトリ。GitHub から amazon-efs-utils パッケージをインストールする方法の詳細については、「[他の Linux ディストリビューションで amazon-efs-utils パッケージをインストールする](#)」を参照してください。

Note

- を使用している場合は AWS Direct Connect、「」でインストール手順を確認できます[前提条件](#)。
- Amazon Linux 1 (AL1) AMI は 2023 年 12 月 31 日にend-of-lifeとなり、2024 年 4 月以降にリリースされたamazon-efs-utilsパッケージ (バージョン 2.0 以降) ではサポートされていません。アプリケーションを Amazon Linux 2023 (AL2023) にアップグレードすることをお勧めします。これには 2028 年までの長期サポートが含まれます。

EC2 Linux インスタンスに AMI **amazon-efs-utils**パッケージリポジトリからパッケージをインストールするには

1. AL2023 または Amazon Linux 2 (AL2) EC2 インスタンスが作成されていることを確認します。作成方法については、「[ステップ 1: インスタンスを起動する](#)」を参照してください。
2. Secure Shell (SSH) を介してインスタンスのターミナルにアクセスし、適切なユーザー名でログインします。詳細については、「Amazon EC2 ユーザーガイド」の「[EC2 インスタンスに接続する](#)」を参照してください。
3. 次のコマンドを実行して、amazon-efs-utils パッケージをインストールします。

```
sudo yum install -y amazon-efs-utils
```

他の Linux ディストリビューションで amazon-efs-utils パッケージをインストールする

amazon-efs-utils パッケージを Amazon Linux AMI のパッケージリポジトリから取得しない場合は、GitHub でも提供されています。

パッケージをクローンした後、Linux ディストリビューションでサポートされているパッケージタイプに応じて、次のいずれかの方法を使用して、amazon-efs-utils を構築、およびインストールできます。

- RPM – このパッケージタイプは、AL2023、Amazon Linux 2、Red Hat Linux、CentOS などでサポートされています。
- DEB – このパッケージタイプは Ubuntu、Debian などでサポートされています。

他の Linux ディストリビューション用の amazon-efs-utils パッケージをインストールする手順については、Github にある amazon-efs-utils README の「[On other Linux distributions](#)」を参照してください。

macOS Big Sur、macOS Monterey、または macOS Ventura を実行している EC2 Mac インスタンスに Amazon EFS クライアントをインストールする

amazon-efs-utils パッケージは、macOS Big Sur、macOS Monterey、または macOS Ventura を実行している EC2 Mac インスタンスにインストールできます。

Mac インスタンスに amazon-efs-utils パッケージをインストールする手順については、Github にある amazon-efs-utils README の「[On MacOS Big Sur, macOS Monterey, macOS Sonoma and macOS Ventura distribution](#)」を参照してください。

次のステップ

EC2 インスタンスに amazon-efs-utils をインストールしたら、次の手順に進んでファイルシステムをマウントします。

- [botocore をインストール](#)し、Amazon CloudWatch を使ってファイルシステムのマウント状態をモニタリングできるようにします。
- [stunnel を最新バージョンにアップグレード](#)すると、転送中のデータを暗号化できます。
- EFS マウントヘルパーを使用して[ファイルシステムをマウント](#)します。

を使用して Amazon EFS クライアントを自動的にインストールまたは更新する AWS Systems Manager

AWS Systems Manager を使用して、Amazon EFS クライアント () の管理を簡素化できます。amazon-efs-utils。AWS Systems Manager は、インフラストラクチャの表示と制御に使用で

きる AWS サービスです AWS。AWS Systems Manager を使用すると、Amazon EC2 (EC2) インスタンスに `amazon-efs-utils` パッケージをインストールまたは更新するために必要なタスクを自動化できます。ディストリビューターやステートマネージャなどの Systems Manager 機能を使用すると、次のプロセスを自動化できます。

- Amazon EFS クライアントのバージョンニング制御を維持する。
- Amazon EFS クライアントを Amazon EC2 インスタンスに集中的に保存し、体系的に配布します。
- EC2 インスタンスを定義された状態に保つプロセスを自動化します。

詳細については、「[AWS Systems Manager ユーザーガイド](#)」を参照してください。

インストール時に Amazon EFS クライアントが実行する処理

Amazon EFS クライアントを使用して、Amazon CloudWatch Logs のファイルシステムのマウントステータスのモニタリングを自動化し、選択した Linux ディストリビューションの最新バージョンに `stunnel` をアップグレードします。Systems Manager を使用して Amazon EC2 インスタンスに Amazon EFS クライアントをインストールすると、以下のアクションが実行されます。

- [botocore のインストールとアップグレード](#) で記述したのと同じ手順を使用して、`botocore` パッケージをインストールします。Amazon EFS クライアントでは `botocore` を使用して、EFS ファイルシステムのマウントステータスをモニタリングします。
- `efs-utils.conf` を更新することで、CloudWatch ログ内の EFS ファイルシステムのマウントステータスをモニタリングできます。詳細については、「[マウント試行の成功と失敗のモニタリング](#)」を参照してください。
- RHEL7 または CentOS7 を実行している EC2 インスタンスの場合、Amazon EFS クライアントは、[stunnel のアップグレード](#) で記述されているように `stunnel` を自動的にアップグレードします。TLS を使用して EFS ファイルシステムを正常にマウントするには、`stunnel` のアップグレードが必要です。RHEL7 と CentOS7 に付属の `stunnel` バージョンは Amazon EFS クライアント (`amazon-efs-utils`) をサポートしていません。

Systems Manager がサポートするオペレーティングシステム

で Amazon EFS クライアント AWS Systems Manager を自動的に更新またはインストールするには、EC2 インスタンスが次のいずれかのオペレーティングシステムを実行している必要があります。

プラットフォーム	プラットフォームバージョン	アーキテクチャ
Amazon Linux 2023 (AL2023)	AL2023	x86_64、arm64 (Graviton2 以降のプロセッサ)
Amazon Linux 2 (AL2)	2.0	x86_64, arm64 (Amazon Linux 2、A1 インスタンスタイプ)
Amazon Linux 1 (AL1)	2017 年 9 月、2018 年 3 月	x86_64
 Note Amazon Linux 1 (AL1) AMI は 2023 年 12 月 31 日にend-of-lifeとなり、2024 年 4 月以降にリリースされたamazon-efs-utilsパッケージ (バージョン 2.0 以降) ではサポートされていません。アプリケーションを Amazon Linux 2023 (AL2023) にアップグレードすることをお勧めします。これには 2028 年までの長期サポートが含まれます。		
CentOS	7、8	x86_64
Red Hat Enterprise Linux (RHEL)	7、8	x86_64, arm64 (RHEL 7.6 以降、A1 インスタンスタイプ)
SUSE Linux Enterprise Server (SLES)	12、15	x86_64

プラットフォーム	プラットフォームバージョン	アーキテクチャ
Ubuntu Server	16.04、18.04、20.04	x86_64, arm64 (Ubuntu Server 16 以降、A1 インスタスタ イプ)

EFS クライアントをインストール AWS Systems Manager するための の設定

amazon-efs-utils パッケージを自動的にインストールまたは更新するように Systems Manager をセットアップするには、2 つの 1 回限りの設定が必要です。

1. (AWS Identity and Access Management IAM) インスタンスプロファイルに必要なアクセス許可を設定します。
2. ステートマネージャーによるインストールまたは更新に使用される関連付け (スケジュールを含む) を設定します。

ステップ 1: 必要なアクセス許可を持つ IAM インスタンスプロファイルを設定する

デフォルトでは、Amazon EFS AWS Systems Manager クライアントを管理し、amazon-efs-utils パッケージをインストールまたは更新するアクセス許可はありません。AWS Identity and Access Management (IAM) インスタンスプロファイルを使用して Systems Manager へのアクセスを許可する必要があります。インスタンスプロファイルは、起動時に IAM ロール情報を Amazon EC2 (EC2) インスタンスに渡すコンテナです。

AmazonElasticFileSystemsUtils AWS 管理アクセス許可ポリシーを使用して、ロールに適切なアクセス許可を割り当てます。インスタンスプロファイルの新しいロールを作成するか、既存のロールに対する AmazonElasticFileSystemsUtils 権限ポリシーを追加できます。次に、このインスタンスプロファイルを使用して EC2 インスタンスを起動する必要があります。詳細については、「[Systems Manager に必要なインスタンスのアクセス許可を設定する](#)」を参照してください。

ステップ 2: State Manager で使用される関連付けを設定する

amazon-efs-utils パッケージはディストリビューターに含まれており、マネージド EC2 インスタンスにデプロイできます。インストールamazon-efs-utils可能な の最新バージョンを表示するには、AWS Systems Manager コンソールまたは任意の AWS コマンドラインツールを使用できます。Distributor にアクセスするには、<https://console.aws.amazon.com/systems-manager/>

を開き、左側のナビゲーションペインで Distributor を選択します。Amazonが所有 セクションの「AmazonEfsutils」に置きます。AmazonEfsutils を選択し、パッケージの詳細を確認します。詳細については、「[パッケージを表示する](#)」を参照してください。

ステートマネージャを使用すると、マネージド EC2 インスタンスを直ちに、またはスケジュールに従って amazon-efs-utils パッケージをインストールまたは更新できます。さらに、amazon-efs-utils が新しい EC2 インスタンスに自動的にインストールされることを確認できます。Distributor および State Manager を使用したパッケージのインストールまたは更新の詳細については、「[Distributor の使用](#)」を参照してください。

Systems Manager コンソールを使用して、インスタンスに amazon-efs-utils パッケージを自動的にインストールまたは更新するには、「[パッケージのインストールまたは更新のスケジュール \(コンソール\)](#)」を参照してください。これにより、一連のインスタンスに適用する状態を定義する State Manager の関連付けを作成するように求められます。関連付けを作成するときは、次の入力を使用します。

- パラメータでは、アクション > インストール および インストールのタイプ > インプレース更新を選択します。
- ターゲット では、推奨設定は すべてのインスタンスを選択する であり、自動的にインストールまたは AmazonEfsutils を更新するにはターゲットとして新規および既存の EC2 インスタンスを登録します。または、インスタスタグを指定したり、インスタンスを手動で選択したり、リソースグループを選択してインスタンスのサブセットに関連付けを適用することもできます。インスタスタグを指定する場合は、タグを使用して EC2 インスタンスを起動し、AWS Systems Manager が Amazon EFS クライアントを自動的にインストールまたは更新できるようにする必要があります。
- スケジュールの指定 では、AmazonEfsutils の推奨設定は30日ごとです。コントロールを使用して、cron または関連付けの rate スケジュールを作成できます。

を使用して複数の EC2 インスタンス AWS Systems Manager に EFS ファイルシステムをマウントするには、「」を参照してください[複数の EC2 インスタンスへの EFS のマウント](#)。

botocore のインストールとアップグレード

Amazon EFS クライアントは、を使用して他の AWS サービスとやり取りbotocoreします。これは、EFS ファイルシステムのマウント試行の成功または失敗を CloudWatch Logs でモニタリングする場合に必要です。詳細については、「[マウント試行の成功と失敗のモニタリング](#)」を参照してください。

botocore のインストールとアップグレードの手順については、Github にある `amazon-efs-utils` README の「[Installing botocore](#)」を参照してください。

stunnel のアップグレード

EFS マウントヘルパーによる転送中のデータの暗号化には、OpenSSL バージョン 1.0.2 以降、およびオンライン証明書ステータスプロトコル (OCSP) と証明書ホスト名チェックの両方をサポートするバージョンの `stunnel` が必要です。EFS マウントヘルパーは、`stunnel` プログラムを使用して TLS 機能を提供します。Linux のバージョンによっては、これらの TLS 機能をサポートする `stunnel` のバージョンがデフォルトで含まれていない場合があることに注意してください。このような Linux ディストリビューションのいずれかを使用している場合は、TLS を使用した EFS ファイルシステムのマウントに失敗します。

EFS マウントヘルパーをインストールした後、次の手順でシステムの `stunnel` のバージョンをアップグレードできます。

Amazon Linux、Amazon Linux 2、およびサポートされている Linux ディストリビューションで **stunnel** をアップグレードするには ([SLES 12](#) を除く)

1. ウェブブラウザで、`stunnel` ダウンロードページ <https://stunnel.org/downloads.html> に進みます。
2. `tar.gz` 形式で利用可能な最新 `stunnel` バージョンを検索します。ファイル名をメモしておきます。この名前は以降のステップで必要になります。
3. Linux クライアントでターミナルを開き、記載されている順にコマンドを実行します。

a. RPM の場合:

```
sudo yum install -y gcc openssl-devel tcp_wrappers-devel
```

DEB の場合:

```
sudo apt-get install build-essential libwrap0-dev libssl-dev
```

- b. ***latest-stunnel-version*** を、ステップ 2 で書き留めておいたファイル名に置き換えます。

```
sudo curl -o latest-stunnel-version.tar.gz https://www.stunnel.org/downloads/latest-stunnel-version.tar.gz
```

- c.

```
sudo tar xvfz latest-stunnel-version.tar.gz
```
- d.

```
cd latest-stunnel-version/
```
- e.

```
sudo ./configure
```
- f.

```
sudo make
```
- g. 現在の stunnel パッケージは bin/stunnel にインストールされています。したがって、新しいバージョンをインストールするには、次のコマンドを使用してそのディレクトリを削除します。

```
sudo rm /bin/stunnel
```

- h. 最新バージョンをインストールします。

```
sudo make install
```

- i. シンボリックリンクを作成します。

```
sudo ln -s /usr/local/bin/stunnel /bin/stunnel
```

macOS で stunnel をアップグレードするには

- EC2 Mac インスタンスでターミナルを開き、次のコマンドを実行して、stunnel の最新バージョンにアップグレードします。

```
brew upgrade stunnel
```

SLES 12 の stunnel のアップグレード

- 次のコマンドを実行し、zypper パッケージマネージャーの指示に従って、SLES12 を実行しているコンピューティングインスタンスで stunnel をアップグレードします。

```
sudo zypper addrepo https://download.opensuse.org/repositories/security:Stunnel/  
SLE_12_SP5/security:Stunnel.repo  
sudo zypper refresh
```

```
sudo zypper install -y stunnel
```

必要な機能のある stunnel のバージョンをインストールした後、推奨される設定で Amazon EFS と TLS を使用してファイルシステムをマウントできます。

stunnel のインストールに関する問題の解決

stunnel をインストールできない場合は、証明書ホスト名のチェックを無効にしてみてください。さらに、オンライン証明書ステータスプロトコル (OCSP) を有効にして、可能な限りセキュリティを強化してください。

トピック

- [証明書ホスト名のチェックの無効化](#)
- [オンライン証明書ステータスプロトコルの有効化](#)

証明書ホスト名のチェックの無効化

必要な依存関係をインストールできない場合、Amazon EFS マウントヘルパー設定で、必要に応じて証明書ホスト名チェックを無効にできます。実稼働環境でこれらの機能を無効にすることはお勧めしません。証明書ホスト名チェックを無効にするには、次の操作を行います。

1. 任意のテキストエディタを使用して、`/etc/amazon/efs/efs-utils.conf` ファイルを開きます。
2. `stunnel_check_cert_hostname` 値を `false` に設定します。
3. 変更をファイルに保存して閉じます。

転送中のデータの暗号化の使用の詳細については、[EFS ファイルシステムをマウントする](#) を参照してください。

オンライン証明書ステータスプロトコルの有効化

VPC から CA にアクセスできない場合のファイルシステムの可用性を最大化するため、転送中のデータの暗号化を選択した場合、オンライン証明書ステータスプロトコル (OCSP) はデフォルトでは有効になりません。Amazon EFS は [Amazon 認証機関](#) (CA) を使用して、TLS 証明書を発行して署名し、CA は OCSP を使用して失効した証明書をチェックするようにクライアントに指示します。OCSP エンドポイントは、証明書のステータスを確認するため、仮想プライベートクラウドが

らインターネット経由でアクセスできる必要があります。サービス内で、Amazon EFS は証明書のステータスを継続的にモニタリングし、検出した失効した証明書を置き換えるために新しい証明書を発行します。

可能な限り強力なセキュリティを提供するため、OCSP を有効にできます。これにより、Linux クライアントは失効した証明書を確認することができます。OCSP は失効した証明書の悪用を防止しますが、それが VPC 内部で発生する可能性はほとんどありません。EFS TLS 証明書が取り消された場合、Amazon はセキュリティ情報を発行し、取り消された証明書を拒否する新しいバージョンの EFS マウントヘルパーをリリースします。

以降のすべての EFS への TLS 接続において、Linux クライアントで OCSP を有効にするには

1. Linux クライアントのターミナルを開きます。
2. 任意のテキストエディタを使用して、`/etc/amazon/efs/efs-utils.conf` ファイルを開きます。
3. `stunnel_check_cert_validity` の値を `true` に設定します。
4. 変更をファイルに保存して閉じます。

mount コマンドの一部として OCSP を有効にするには

- 次のマウントコマンドを使用して、ファイルシステムをマウントするときに OCSP を有効にします。

```
$ sudo mount -t efs -o tls,ocsp fs-12345678:/ /mnt/efs
```

FIPS モードの有効化

ファイルシステムをマウントするときにオペレーティングシステムが連邦情報処理規格 (FIPS) エンドポイントを使用している場合は、Amazon EFS クライアントで FIPS モードを有効にする必要があります。FIPS モードを有効にするには、オペレーティングシステム上の `efs-utils.conf` ファイルを変更する必要があります。

 Note

FIPS モードでは、インストールされているバージョンの OpenSSL が FIPS でコンパイルされている必要があります。FIPS で OpenSSL を設定する方法の詳細については、[OpenSSL FIPS README](#) を参照してください。

Amazon EFS クライアントで FIPS モードを有効にするには

1. Secure Shell (SSH) を使用して Amazon EC2 インスタンスのターミナルにアクセスし、適切なユーザー名でログインします。詳細については、「Amazon Elastic Compute Cloud ユーザーガイド」の[EC2 インスタンスに接続する](#)を参照してください。
2. 任意のテキストエディタを使用して、`/etc/amazon/efs/efs-utils.conf` ファイルを開きます。
3. 次のテキストを含む行を見つけます。

```
"fips_mode_enabled = false"
```

4. テキストを次のように変更します。

```
"fips_mode_enabled = true"
```

5. 変更内容を保存します。

EFS リソースの作成と管理

Amazon EFS は、POSIX 準拠の伸縮自在な共有ファイルストレージを提供します。作成したファイルシステムは、複数の Amazon EC2 インスタンスからの同時書き込みおよび読み取りアクセスをサポートします。ファイルシステムは、AWS リージョン 作成された のすべてのアベイラビリティーゾーンからもアクセスできます。

ネットワークファイルシステム バージョン 4.0 および 4.1 プロトコル (NFSv4) を使用して、Amazon VPC に基づいて仮想プライベートクラウド (VPC) の EC2 インスタンスに Amazon EFS ファイルシステムをマウントできます。詳細については、「[Amazon EFS の仕組み](#)」を参照してください。

たとえば、VPC に 1 つ以上の EC2 インスタンスが起動されているとします。これらのインスタンスでファイルシステムを作成して使用することにします。VPC で Amazon EFS ファイルシステムを使用するには、以下の一般的なステップを実行する必要があります。

- Amazon EFS ファイルシステムの作成 — ファイルシステムを作成するときは、名前 タグを使用することをお勧めします。名前 タグの値はコンソールに表示され、ファイルシステムを簡単に識別できます。その他のオプションのタグをファイルシステムに追加することもできます。
- ファイルシステムのマウントターゲットを作成する – VPC のファイルシステムにアクセスし Amazon EC2 インスタンスにファイルシステムをマウントするには、VPC サブネット内にマウントターゲットを作成する必要があります。
- セキュリティグループを作成する – Amazon EC2 インスタンスとマウントターゲットの両方とも、関連付けられたセキュリティグループを持っている必要があります。これらのセキュリティグループは相互のトラフィックを制御する仮想ファイアウォールとして機能します。マウントターゲットに関連付けたセキュリティグループを使用して、ファイルシステムへのインバウンドトラフィックを制御できます。これを行うには、特定の EC2 インスタンスからのアクセスを許可するインバウンドルールをマウントターゲットのセキュリティグループに追加します。その後、その EC2 インスタンスでのみファイルシステムをマウントできます。

トピック

- [実装についての要約](#)
- [リソース ID](#)
- [作成トークンと冪等性](#)
- [EFS ファイルシステムの作成](#)

- [EFS ファイルシステムの削除](#)
- [ファイルシステムポリシーの作成](#)
- [アクセスポイントの作成](#)
- [アクセスポイントの削除](#)
- [EFS リソースのタグ付け](#)
- [チュートリアル: 書き込み可能なユーザーごとのサブディレクトリを作成する](#)

実装についての要約

Amazon EFS では、ファイルシステムはプライマリリソースです。各ファイルシステムには、ID、作成トークン、作成時刻、バイト単位でのファイルシステムサイズ、ファイルシステム用に作成されたマウントターゲットの数、ファイルシステムのライフサイクルポリシーなどのプロパティがあります。

Amazon EFS は、プライマリリソースの設定のための他のリソースもサポートします。これには、マウントターゲットやアクセスポイントが含まれます。

- マウントターゲット - ファイルシステムにアクセスするには、VPC にマウントターゲットを作成する必要があります。各マウントターゲットには、プロパティとしてマウントターゲット ID、マウントターゲットが作成されたサブネットの ID、マウントターゲットが作成されたファイルシステムの ID、ファイルシステムのマウント先の IP アドレス、VPC セキュリティグループ、マウントターゲット状態などがあります。mount コマンドで DNS 名または IP アドレスが使用できます。

各ファイルシステムの DNS 名は次の形式になります。

```
file-system-id.efs.aws-region.amazonaws.com
```

この DNS 名を mount コマンドで指定して、Amazon EFS ファイルシステムをマウントできます。EC2 インスタンスまたはオンプレミスサーバーに efs-mount-point サブディレクトリを作成するとします。次に、mount コマンドを使用してファイルシステムをマウントできます。たとえば、Amazon Linux AMI では、以下の mount コマンドを使用できます。

```
$ sudo mount -t nfs -o  
nfsvers=4.1,rsiz=1048576,wsiz=1048576,hard,timeo=600,retrans=2,noresvport file-  
system-DNS-name:/ ~/efs-mount-point
```

詳細については、「[マウントターゲットの管理](#)」を参照してください。

- アクセスポイント - アクセスポイントは、アクセスポイントを介したすべてのファイルシステム要求に対して、オペレーティングシステムのユーザー、グループ、およびファイルシステムのパスを適用します。アクセスポイントのオペレーティングシステムのユーザーおよびグループは、NFS クライアントから提供されるすべての ID 情報を上書きします。ファイルシステムのパスは、アクセスポイントのルートディレクトリとしてクライアントに公開されます。これにより、各アプリケーションは共有ファイルベースのデータセットにアクセスするときに、常に正しいオペレーティングシステム ID と正しいディレクトリを使用できます。アクセスポイントを使用するアプリケーションは、それ自体のディレクトリ以下のデータにのみアクセスできます。詳細については、「[アクセスポイントの使用](#)」を参照してください。

マウントターゲットとタグは、ファイルシステムに関連付けられたサブリソースです。これらは、既存のファイルシステムのコンテキスト内でのみ作成できます。

Amazon EFS では、API オペレーションでこれらのリソースの作成および管理ができます。各リソースの作成および削除オペレーションに加えて、Amazon EFS はリソース情報を取得する記述オペレーションをサポートしています。これらのリソースの作成および管理には、以下のオプションがあります。

- Amazon EFS コンソールの使用 - 使用例については、「[入門](#)」を参照してください。
- Amazon EFS コマンドラインインターフェース (CLI) の使用 - 例については、「[チュートリアル: EFS ファイルシステムを作成し、を使用して EC2 インスタンスにマウントする AWS CLI](#)」を参照してください。
- 以下のように、これらのリソースをプログラムで管理することもできます。
 - AWS SDKs を使用する - AWS SDKs 基盤となる Amazon EFS API をラップすることで、プログラミングタスクを簡素化します。SDK クライアントは、アクセスキーを使用してリクエストの認証も行います。詳細については、「[サンプルコードとライブラリ](#)」を参照してください。
 - アプリケーションから直接 API を呼び出す - 何らかの理由で SDK を使用できない場合、Amazon EFS API コールをアプリケーションから直接呼び出せます。ただし、このオプションを使用する場合、リクエストを認証するために必要なコードを記述する必要があります。Amazon EFS API の詳細については、「[Amazon EFS API](#)」を参照してください。

リソース ID

Amazon EFS は、作成時に一意のリソース識別子 (ID) をすべての EFS リソースに割り当てます。すべての EFS リソース ID は、リソース識別子と、数字 0~9 と小文字の a~f の組み合わせで構成されます。

2021 年 10 月以前は、新しく作成されたファイルシステムおよびマウントターゲットリソースに割り当てられた ID が、ハイフンの後に 8 文字を使用していました (例: fs-12345678)。2021 年 5 月から 2021 年 10 月にかけて、これらのリソースタイプの ID は、ハイフンの後に 17 文字を使用するように変更されました (例: fs-1234567890abcdef0)。アカウントの作成時期によっては、ファイルシステムやマウントターゲットのリソースに短い ID が付いている場合がありますが、これらのタイプの新しいリソースには長い ID が付けられます。リソース ID が変更されることはありません。

作成トークンと冪等性

冪等性は、API リクエストが 1 回だけ完了することを保証します。冪等性リクエストでは、最初のリクエストが正常に完了した場合、その後のリクエストには追加効果はありません。これは、Amazon EFS API を操作するときに重複するジョブが作成されるのを防ぐのに役立ちます。

Amazon EFS API は、クライアントリクエストトークンとの冪等性をサポートします。クライアントのリクエストトークンは、ジョブリクエストを行うときに指定する一意の文字列です。

クライアントのリクエストトークンには、64 文字の ASCII 文字を含む任意の文字列を指定できます。リクエストが成功してから 1 分以内にクライアントリクエストトークンを再利用すると、API は最初のリクエストのジョブ詳細を返します。

コンソールを使用する場合、トークンが生成されます。コンソールでカスタム作成フローを使用した場合、ユーザーに対して生成される作成トークンの形式は次のとおりです。

```
"CreationToken": "console-d215fa78-1f83-4651-b026-facafd8a7da7"
```

クイック作成を使用してサービス推奨設定でファイルシステムを作成する場合、作成トークンの形式は次のとおりです。

```
"CreationToken": "quickCreated-d7f56c5f-e433-41ca-8307-9d9c0f8a77a2"
```

EFS ファイルシステムの作成

「開始方法」の[演習](#)で説明されているように、新しい EC2 起動インスタンスを作成するときにファイルシステムを作成できます。ただし、Amazon EFS コンソール、AWS Command Line Interface (AWS CLI)、または Amazon EFS API を使用してファイルシステムを作成することもできます。

Amazon EFS コンソールを使用してファイルシステムを作成する場合、ファイルシステムで推奨設定を使用するか、設定をカスタマイズするかに応じて、さまざまなオプションがあります。

- クイック作成を使用すると、次の推奨設定でファイルシステムをすばやく作成できます。
 - リージョナルな可用性
 - ファイルシステムを 30 日後に EFS 低頻度アクセス (IA) ストレージに移行し、90 日後に EFS Archive ストレージに移行し、EFS 標準ストレージに移行しないライフサイクルポリシー
 - 保管時のデータの暗号化が有効
 - Elastic スループットモード
 - [General Purpose(汎用)]パフォーマンスモード。
 - 指定されたサブネットで使用可能な IPv4 を使用してファイルシステムが作成される AWS リージョンの各アベイラビリティゾーンで設定されたマウントターゲット
- カスタマイズを使用して、選択した設定でファイルシステムを作成できます。

ファイルシステム設定と推奨値を一覧表示する表については、「」を参照してください[ファイルシステムの設定オプション](#)。

ファイルシステムの作成に必要な IAM アクセス許可

ファイルシステムやアクセスポイントなどの EFS リソースを作成するには、対応する API オペレーションとリソースに対する AWS Identity and Access Management (IAM) アクセス許可が必要です。

アカウントに IAM ユーザーを作成し、ユーザーポリシーを使用して Amazon EFS のアクションのアクセス許可をユーザーに付与します。また、ロールを使用して、クロスアカウントのアクセス権限を付与できます。Amazon Elastic File System は、AWS のサービス ユーザーに代わって他の を呼び出すために必要なアクセス許可を含む IAM サービスにリンクされたロールも使用します。API オペレーションのアクセス権限を管理する方法の詳細については、「[Amazon EFS のためのアイデンティティとアクセス管理](#)」を参照してください。

ファイルシステムの設定オプション

EFS ファイルシステムは、次の表に示す設定で設定されています。

- クイック作成を使用してファイルシステムを作成する場合、ファイルシステムは 設定に指定された推奨値で作成されます。
- カスタマイズを使用してカスタムファイルシステムを作成する場合は、設定の推奨値を変更できます。

ファイルシステムを作成したら、ファイルシステムタイプ、暗号化、パフォーマンスモードを除き、ファイルシステムの設定をカスタマイズできます。

EFS ファイルシステム設定

設定	説明
ファイルシステムタイプ	推奨: リージョン ファイルシステムタイプは、EFS ファイルシステムが 内にデータを保存する 可用性と耐久性 を決定します AWS リージョン。 • リージョンファイルシステムは、内のすべてのアベイラビリティーゾーンにデータとメタデータを冗長的に保存します AWS リージョン。AWS リージョンのアベイラビリティーゾーンごとに マウントターゲットを作成できます。Regional(リージョン別)は、最高レベルの可用性と耐久性を実現します。 • 1 ゾーンファイルシステムは、データとメタデータを 1 つのアベイラビリティーゾーン内に冗長的に保存します。1 ゾーンファイルシステムは、1 つのマウントターゲットのみを持つことができます。このマウントターゲットは、ファイルシステムの作成場所と同じアベイラビリティーゾーンにある必要があります。

設定	説明
	 Note 1 ゾーンファイルシステムは特定の アベイラビリティーゾーンでのみ使 用できます。 ファイルシステムのタイプの詳細については、 「EFS ファイルシステムの可用性と耐久性」 を参照してください。

設定	説明
ライフサイクル管理	推奨: 30 日後に IA ストレージに移行し、90 日後にアーカイブストレージに移行します。標準ストレージに移行しないでください。 ライフサイクル管理では、ライフサイクルポリシーを使用して、アクセスパターンに基づいて、低コストの低頻度アクセス (IA) ストレージクラスとの間でファイルを自動的に移動します。を使用してファイルシステムを作成すると AWS Management Console、ファイルシステムのライフサイクルポリシーは次のデフォルト設定で設定されます。 • [IA へ移行] は [前回のアクセスから 30 日間] に設定されています。 • アーカイブへの移行は、前回のアクセスから 90 日間に設定されています。 • [標準への移行] は [なし] に設定されています。 AWS CLI、Amazon EFS API、または AWS SDKs を使用してファイルシステムを作成する場合、ライフサイクルポリシーを同時に設定することはできません。ファイルシステムが作成されるのを待ってから、PutLifecycleConfiguration API オペレーションを使ってライフサイクルポリシーを更新する必要があります。 ライフサイクル管理の詳細については、「ストレージライフサイクルの管理」を参照してください。

設定	説明
保管時の暗号化	推奨: 有効 Amazon EFS は、デフォルトで保管中のデータを暗号化するために AWS Key Management Service、(AWS KMS) EFS サービスキー (aws/elasticfilesystem) を使用します。暗号化では、そこに保存されているすべてのデータとメタデータが暗号化されます。EFS ファイルシステムを作成した後、暗号化設定を変更することはできません。つまり、暗号化されていないファイルシステムを暗号化するように変更することはできません。代わりに、暗号化されたファイルシステムを新たに作成する必要があります。 Amazon EFS の暗号化の詳細については、「Amazon EFS でのデータの暗号化」を参照してください。

設定	説明
スループットモード	推奨: Elastic 次のスループットモードから選択できます。 • Elastic – ワークロードのパフォーマンスニーズを満たすために、リアルタイムで自動的にスケールアップおよびスケールダウンするスループットを提供します。 • プロビジョニング済み – ファイルシステムのサイズに関係なく、必要なスループット量を指定します。 • バースト – 標準ストレージ内のデータ量に応じてスケールリングするスループットを提供します。  Note エラスティックスループットおよびプロビジョニングされたスループットの使用には追加料金に関連付けられます。詳細については、「Amazon EFS 料金表」を参照してください。 スループットモードの詳細については、「」を参照してください スループットモード。

設定	説明
パフォーマンスモード	推奨: 汎用 汎用パフォーマンスモードは、オペレーションごとのレイテンシーが最も低く、すべてのファイルシステムに推奨されます。 最大 I/O モードは前世代のパフォーマンスタイプで、汎用モードよりも高いレイテンシーに耐えられる高度に並列化されたワークロード向けに設計されています。最大 I/O モードは、1 ゾーンファイルシステムまたはエラスティックスループットを使用するファイルシステムではサポートされません。  Important 最大 I/O ではオペレーションごとのレイテンシーが高くなるため、すべてのファイルシステムに汎用パフォーマンスモードを使用することをお勧めします。 詳細については、「パフォーマンスモード」を参照してください。

設定	説明
ネットワークアクセス	推奨: ファイルシステムが使用可能な各アベイラビリティゾーンに作成されたマウントターゲット マウントターゲットの設定は次のとおりです。 • VPC – ファイルシステムのマウントターゲットは、一度に 1 つの VPC に作成できます。デフォルトでは、EC2 インスタンスのデフォルトサブネットが選択されます。別の VPC からファイルシステムにアクセスする必要がある場合は、VPC を変更します。 ファイルシステムの後に を作成した後に VPC を変更する場合は、まずマウントターゲットを現在の VPC から削除する必要があります。詳細については、「マウントターゲット VPC の変更」を参照してください。 • アベイラビリティゾーンとサブネット ID – マウントターゲットのアベイラビリティゾーンまたはサブネットを変更できます。 • IP アドレスタイプ – デフォルトでは、Amazon EFS はマウントターゲットを作成する IPv4 アドレスを使用します。代わりに、IPv6 アドレスのみをサポートするように IPv6 を選択するか、IPv4 アドレスと IPv6 アドレスの両方をサポートするようにデュアルスタックを選択できます。  Note IP アドレスタイプは、サブネットの IP タイプと一致する必要があります。さらに、IP アドレスタイプはサブネットの IP アドレス属性を上書

設定	説明
	きします。例えば、IP アドレスタイプが IPv4-only で、サブネットに対して IPv6 アドレス属性が有効になっている場合、サブネットで作成されたネットワークインターフェイスはサブネットの範囲から IPv4 アドレスを受け取ります。詳細については「サブネットの IP アドレス指定属性を変更する」を参照してください。 • IP アドレス – デフォルトでは、Amazon EFS は指定されたサブネットの使用可能なアドレスにマウントターゲットを作成します。または、マウントターゲットの IP アドレスを指定することもできます。 マウントターゲットの作成後に IP アドレスを変更することはできません。マウントターゲットを削除し、新しいアドレスで新しいターゲットを作成する必要があります。 • セキュリティグループ – VPC ターゲットに割り当てられたセキュリティグループは、デフォルトで使用されます。セキュリティグループを追加または削除できます。セキュリティグループの詳細については、マウントターゲットセキュリティグループの変更 を参照してください。 AWS CLI、Amazon EFS API、または AWS SDKs を使用してファイルシステムを作成する場合、マウントターゲットを同時に作成することはできません。ファイルシステムが作成されるまで待ってから、CreateMountTarget API オペレーションを使用してマウントターゲットを作成する必要があります。

設定	説明
	マウントターゲットの詳細については、「」を参照してください マウントターゲットの管理 。

コンソールを使用したクイック作成

Amazon EFS コンソールを使用して、推奨設定の Amazon EFS ファイルシステムを作成します。カスタマイズされた構成でファイルシステムを作成する場合、「[コンソールを使用したカスタム作成](#)」を参照してください。

推奨設定を持つ Amazon EFS ファイルシステムをすばやく作成するには

1. にサインイン AWS Management Console し、<https://console.aws.amazon.com/efs/> で Amazon EFS コンソールを開きます。
2. [Create file system (ファイルシステムの作成)] を選択して、ファイルシステム作成 ダイアログを開きます。
3. (オプション) ファイルシステムに [名前] を入力します。
4. Virtual Private Cloud (VPC) では、VPC を選択するか、デフォルトの VPC に設定したままにします。
5. [作成] を選択し、以下のサービス推奨設定を使用するファイルシステムを作成します。
 - リージョナルな可用性
 - ファイルシステムを 30 日後に EFS 低頻度アクセス (IA) ストレージに移行し、90 日後に EFS Archive ストレージに移行し、EFS 標準ストレージに移行しないライフサイクルポリシー
 - 保管時のデータの暗号化が有効
 - Elastic スループットモード
 - [General Purpose(汎用)]パフォーマンスモード。
 - 指定されたサブネットで使用可能な IPv4 を使用して、ファイルシステムが作成される AWS リージョン の各アベイラビリティゾーンに設定されたマウントターゲット

ファイルシステム ページが表示され、作成したファイルシステムのステータスを示すバナーが上部に表示されます。ファイルシステムが使用可能になると、バナーにファイルシステムの詳細ページにアクセスするためのリンクが表示されます。

ファイルシステムの状態の詳細については、「[ファイルシステムのステータスについて](#)」を参照してください。

コンソールを使用したカスタム作成

このセクションでは、Amazon EFS コンソールを使用して、サービスの推奨設定を使用する代わりに、カスタマイズされた設定で EFS ファイルシステムを作成するプロセスについて説明します。推奨設定を使用したファイルシステムの作成の詳細については、「[コンソールを使用したクイック作成](#)」を参照してください。

コンソールを使用してカスタム設定で EFS ファイルシステムを作成するプロセスは、4 つのステップから構成されます。

- ステップ 1 — ストレージクラスやスループットモードなど、ファイルシステムの一般的な設定を行います。
- ステップ 2 - 仮想プライベートクラウド (VPC) およびマウントターゲットを含むファイルシステムのネットワーク設定を構成します。マウントターゲットごとに、アベイラビリティーゾーン、サブネット、IP アドレス、およびセキュリティグループを設定します。
- ステップ 3 - (オプション) ファイルシステムへの NFS クライアントアクセスを制御するファイルシステムポリシーを作成します。
- ステップ 4 - ファイルシステムの設定を確認し、変更を加えてから、ファイルシステムを作成します。

ステップ 1: ファイルシステム設定を構成する

1. にサインイン AWS Management Console し、<https://console.aws.amazon.com/efs/> で Amazon EFS コンソールを開きます。
2. [ファイルシステムの作成] を選択して、[ファイルシステムの作成] ダイアログボックスを開きます。
3. カスタマイズを選択して、推奨設定を使用してファイルシステムを作成する代わりに、カスタマイズされたファイルシステムを作成します。[ファイルシステム設定] ページが開きます。
4. [全般] 設定で、次のように入力します。
 - a. (オプション) ファイルシステムに[Name (名前)] を入力します。
 - b. ファイルシステムタイプでは、デフォルトでリージョンが選択されます。ファイルシステムデータとメタデータを 1 つのアベイラビリティーゾーン内に冗長的に保存するファイルシ

システムを作成する場合は、1 ゾーンを選択します。[1 ゾーン] を選択した場合は、ファイルシステムを作成する [アベイラビリティーゾーン] を選択するか、デフォルト値のままにします。

c. [ライフサイクル管理] では、必要に応じてライフサイクルポリシーを変更します。

- [IA へ移行] — 標準ストレージでファイルに最後にアクセスした日時に基づいて、ファイルを低頻度アクセス (IA) ストレージクラスに移行するタイミングを選択します。
- [アーカイブへの移行] — 標準ストレージでファイルに最後にアクセスした日時に基づいて、ファイルをアーカイブストレージクラスに移行するタイミングを選択します。
- [標準への移行] — ファイルシステムをこのストレージクラスに移行するかどうかを選択します。

ライフサイクルポリシーの詳細については、「[ストレージライフサイクルの管理](#)」を参照してください。

d. Amazon EFS は、デフォルトで保管中のデータを暗号化するために AWS Key Management Service、(AWS KMS) EFS サービスキー (aws/elasticfilesystem) を使用します。暗号化に使用する別の KMS キーを選択するには、暗号化設定のカスタマイズを展開し、リストからキーを選択します。または、使用する KMS キーの KMS キー ID または Amazon リソースネーム (ARN) を入力します。

新しいキーを作成する必要がある場合は、を作成 AWS KMS key を選択して AWS KMS コンソールを起動し、新しいキーを作成します。

このチェックボックスをオフにすると、保存中のデータの暗号化をオフにすることができません。

ファイルシステムの作成後に暗号化設定を変更することはできません。詳細については、「[Amazon EFS でのデータの暗号化](#)」を参照してください。

5. [パフォーマンス] 設定では、次のオペレーションを行います。

a. [スループットモード] では、[伸縮自在] モードがデフォルトで選択されています。

- プロビジョニングされたスループットを使用するには、[プロビジョニング済み] モードを選択し、[プロビジョニングされたスループット (MiB/秒)] に、ファイルシステム要求に対してプロビジョニングするスループットの量を入力します。[最大読み込みスループット] の量は、入力したスループットの 3 倍の量で表示されます。
- バーストスループットを使用するには、[バースト] を選択します。

スループットモードを選択すると、ファイルシステムの月額コストの見積もりが表示されます。ファイルシステムが利用可能になった後に、スループットモードを変更できます。

パフォーマンスのニーズに合った適切なスループットモードを選択する方法の詳細については、「[Amazon EFS でのデータの暗号化](#)」を参照してください。

- b. パフォーマンスモードでは、デフォルトでは「汎用」です。パフォーマンスモードを変更するには、[その他の設定] を展開し、[最大 I/O] を選択します。

ファイルシステムが利用可能になった後は、パフォーマンスモードを変更できません。詳細については、「[パフォーマンスモード](#)」を参照してください。

 Important

最大 I/O ではオペレーションごとのレイテンシーが高くなるため、すべてのファイルシステムに汎用パフォーマンスモードを使用することをお勧めします。

6. (オプション) タグのキーバリューのペアをファイルシステムに追加します。
7. [次へ] を選択して、ファイルシステムのネットワークアクセスを設定します。

ステップ 2: ネットワークアクセスの設定

ステップ 2 では、VPC およびマウントターゲットなど、ファイルシステムのネットワーク設定を構成します。

1. EC2 インスタンスをファイルシステムに接続する[Virtual Private Cloud(仮想プライベートクラウド) (VPC)]を選択します。詳細については、「[マウントターゲットの管理](#)」を参照してください。
2. マウントターゲットでは、ファイルシステム用の 1 つ以上のマウントターゲットを作成します。マウントターゲットごとに、次のプロパティを設定します。
 - アベイラビリティーゾーン — デフォルトでは、マウントターゲットは、AWS リージョンの各アベイラビリティーゾーンで構成されます。特定のアベイラビリティーゾーンにマウントターゲットが必要ない場合は、[Remove(削除)]を選択してそのゾーンのマウントターゲットを削除します。ファイルシステムにアクセスする予定のすべてのアベイラビリティーゾーンにマウントターゲットを作成します。これにはコストはかかりません。

- サブネット ID — アベイラビリティーゾーンで使用可能なサブネットから選択します。デフォルトのサブネットが事前に選択されています。
 - IP アドレスタイプ — IPv4 アドレスのみをサポートする場合は IPv4、IPv6 アドレスのみをサポートする場合は IPv6、IPv4 アドレスと IPv6 アドレスの両方をサポートする場合はデュアルスタックを選択します。
 - IPv4 または IPv6 アドレス — マウントターゲットを配置する IP アドレスがわかっている場合は、IP アドレスタイプに一致する IP アドレスボックスに入力します。値を指定しない場合、Amazon EFS は指定されたサブネットから未使用の IP アドレスを選択します。
 - セキュリティグループ — デフォルトでは、Amazon EFS は VPC のデフォルトのセキュリティグループを選択します。セキュリティグループを変更するには、割り当てられたグループを削除し、セキュリティグループの選択リストからグループを選択します。マウントターゲットに 1 つ以上のセキュリティグループを指定できます。詳細については、「[VPC セキュリティグループの使用](#)」を参照してください。
3. [Add mount target(マウントターゲットの追加)]を選択して、アベイラビリティーゾーンが存在しないアベイラビリティーゾーンのマウントターゲットを作成します。マウントターゲットが各アベイラビリティーゾーンに対して構成されている場合、この選択は使用できません。
 4. [次へ] を選択して、ファイルシステムポリシーを設定します。

ステップ 3: ファイルシステムポリシーを作成する (オプション)

必要に応じて、ファイルシステムのファイルシステムポリシーを作成できます。EFS ファイルシステムポリシーは、NFS クライアントのファイルシステムへのアクセスを制御するために使用する IAM リソースポリシーです。詳細については、「[IAM を使用してファイルシステムへのアクセスを制御する](#)」を参照してください。

1. ポリシーのオプションの場合は、設定済みのファイルシステムポリシーの任意の組み合わせを選択できます。
 - デフォルトでルートアクセスを防止する - このオプションは、許可された EFS アクションの設定から ClientRootAccess を削除します。
 - デフォルトで読み取り専用アクセスを強制する - このオプションは、許可された EFS アクションの設定から ClientWriteAccess を削除します。
 - 匿名アクセスを防止する - このオプションは、許可された EFS アクションの設定から ClientMount を削除します。

- すべてのクライアントに転送中の暗号化を強制する - このオプションは、暗号化されていないクライアントへのアクセスを拒否します。

事前設定されたポリシーを選択すると、ポリシーの JSON オブジェクトが[Policy editor(ポリシーエディター)]ペインに表示されます。

2. 追加のアクセス許可を付与を使用して、別のプリンシパルを含む追加の IAM プリンシパルにファイルシステムアクセス許可を付与します AWS アカウント。[Add(追加)] を選択し、アクセス許可を付与するエンティティのプリンシパル ARN を入力します。付与する[Permissions(アクセス許可)]を選択します。追加のアクセス許可は、[Policy editor(ポリシーエディター)]で表示されます。
3. [Policy editor(ポリシーエディター)]を使用して、事前に設定されたポリシーをカスタマイズしたり、独自のファイルシステムポリシーを作成したりできます。エディタを使用すると、事前設定されたポリシーオプションは使用できなくなります。現在のファイルシステムポリシーをクリアして新しいポリシーの作成を開始するには、[Clear(クリア)]を選択します。
4. [次へ] を選択して、ファイルシステムを確認、作成します。

ステップ 4: 確認して作成する

1. 各ファイルシステム構成グループを確認します。この時点で[Edit(編集)]を選択すると、各グループに変更を加えることができます。
2. ファイルシステムの作成を選択してファイルシステムを作成し、ファイルシステムページに戻ります。

上部にあるバナーは、新しいファイルシステムが作成されていることを示します。ファイルシステムが使用可能になると、新しいファイルシステムの詳細ページにアクセスするためのリンクがバナーに表示されます。

を使用して を作成する AWS CLI

を使用している場合は AWS CLI、これらのリソースを順番に作成します。まず、ファイルシステムを作成します。次に、対応する AWS CLI コマンドを使用して、ファイルシステムのマウントターゲットと追加のオプションタグを作成できます。

以下の例では、adminuser を --profile パラメータの値として使用しています。適切なユーザープロファイルを使用して認証情報を提供する必要があります。詳細については、「AWS Command Line Interface ユーザーガイド」の「[AWS CLIを使用するための前提条件](#)」を参照してください。

- 自動バックアップを有効にして暗号化されたファイルシステムを作成するには、次に示すように、Amazon EFS create-file-system CLI コマンド (対応するオペレーションは [CreateFileSystem](#)) を使用します。

```
aws efs create-file-system \  
--creation-token creation-token \  
--encrypted \  
--backup \  
--performance-mode generalPurpose \  
--throughput-mode elastic \  
--region aws-region \  
--tags Key=key,Value=value Key=key1,Value=value1 \  
--profile adminuser
```

たとえば、次のcreate-file-systemコマンドは、で Elastic スループットを使用してファイルシステムを作成しますus-west-2AWS リージョン。コマンドは作成トークンとして MyFirstFS を指定します。Amazon EFS ファイルシステムを作成 AWS リージョン できる のリストについては、の「[Amazon EFS 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。

```
aws efs create-file-system \  
--creation-token MyFirstFS \  
--backup \  
--encrypted \  
--performance-mode generalPurpose \  
--throughput-mode elastic \  
--region us-west-2 \  
--tags Key=Name,Value="Test File System" Key=developer,Value=rhoward \  
--profile adminuser
```

次の例に示すように、ファイルシステムが正常に作成されると、Amazon EFS はファイルシステムの説明を JSON として返します。

```
{  
  "OwnerId": "123456789abcd",  
  "CreationToken": "MyFirstFS",  
  "Encrypted": true,  
  "FileSystemId": "fs-c7a0456e",  
  "CreationTime": 1422823614.0,  
  "LifecycleState": "creating",
```

```

    "Name": "Test File System",
    "NumberOfMountTargets": 0,
    "SizeInBytes": {
        "Value": 6144,
        "ValueInIA": 0,
        "ValueInStandard": 6144,
        "ValueInArchive": 0
    },
    "PerformanceMode": "generalPurpose",
    "ThroughputMode": "elastic",
    "Tags": [
        {
            "Key": "Name",
            "Value": "Test File System"
        }
    ]
}

```

- 次の例では、availability-zone-nameプロパティを使用して、us-west-2aアベイラビリティゾーンでバーストスループットを使用するファイルシステムを作成します。

```

aws efs create-file-system \
--creation-token MyFirstFS \
--availability-zone-name us-west-2a \
--backup \
--encrypted \
--performance-mode generalPurpose \
--throughput-mode bursting \
--region us-west-2 \
--tags Key=Name,Value="Test File System" Key=developer,Value=rhoward \
--profile adminuser

```

次の例に示すように、ファイルシステムが正常に作成されると、Amazon EFS はファイルシステムの説明を JSON として返します。

```

{
    "AvailabilityZoneId": "usw-az1",
    "AvailabilityZoneName": "us-west-2a",
    "OwnerId": "123456789abcd",
    "CreationToken": "MyFirstFS",
    "Encrypted": true,
    "FileSystemId": "fs-c7a0456e",

```

```
"CreationTime": 1422823614.0,
"LifecycleState": "creating",
"Name": "Test File System",
"NumberOfMountTargets": 0,
"SizeInBytes": {
  "Value": 6144,
  "ValueInIA": 0,
  "ValueInStandard": 6144
  "ValueInArchive": 0
},
"PerformanceMode": "generalPurpose",
"ThroughputMode": "bursting",
"Tags": [
  {
    "Key": "Name",
    "Value": "Test File System"
  }
]
```

また、以下に示すように、Amazon EFS はアカウントのファイルシステムを取得するために使用できる `describe-file-systems` CLI コマンド (対応する API オペレーションは [DescribeFileSystems](#)) を提供します。

```
aws efs describe-file-systems \
--region aws-region \
--profile adminuser
```

Amazon EFS は、指定されたリージョンで AWS アカウント 作成された 内のファイルシステムのリストを返します。

EFS ファイルシステムの削除

ファイルシステムの削除は、元に戻すことのできない破壊的なアクションです。ファイルシステムおよび含まれるデータをすべて失います。ファイルシステムから削除したすべてのデータは削除され、データを復元することはできません。ユーザーがファイルシステムからデータを削除すると、そのデータはすぐに使用できなくなくなります。EFS によって、最終的にデータが上書きされます。

Note

レプリケーション設定の一部であるファイルシステムを削除することはできません。最初にレプリケーション設定を削除する必要があります。詳細については、「[レプリケーション設定の削除](#)」を参照してください。

Important

削除する前にファイルシステムを必ずアンマウントする必要があります。

コンソールを使用する

ファイルシステムを削除するには

1. Amazon Elastic File System コンソール の<https://console.aws.amazon.com/efs/> を開いてください。
2. ファイルシステムページで、削除する ファイルシステムを選択します。
3. [削除] を選択します。
4. [ファイルシステムの削除] ダイアログボックスで、表示されているファイルシステム ID を入力し、[確認] を選択して削除を確認します。

コンソールにより、ファイルシステムの削除が簡素化されます。まず、関連付けられているマウントターゲットが削除され、次にファイルシステムが削除されます。

の使用 AWS CLI

AWS CLI コマンドを使用してファイルシステムを削除する前に、ファイルシステム用に作成されたマウントターゲットとアクセスポイントをすべて削除する必要があります。

AWS CLI コマンドの例については、「」を参照してください[ステップ 4: クリーンアップする](#)。

ファイルシステムポリシーの作成

Amazon EFS コンソールまたは AWS CLIを使用してファイルシステムポリシーを作成できます。AWS SDKs または Amazon EFS API を直接使用して、プログラムでファイルシステムポリシーを作成することもできます。EFS ファイルシステムポリシーには 20,000 文字の制限があります。EFS

ファイルシステムポリシーの使用および例の詳細については、「[IAM を使用してファイルシステムへのアクセスを制御する](#)」を参照してください。

Note

Amazon EFS ファイルシステムポリシーの変更が有効になるまでに数分かかる場合があります。

コンソールを使用する

1. Amazon Elastic File System コンソールの <https://console.aws.amazon.com/efs/> を開いてください。
 2. [File Systems (ファイルシステム)] を選択します。
 3. [File systems (ファイルシステム)] ページで、ファイルシステムポリシーを編集または作成する対象のファイルシステムを選択します。
 4. [File system policy (ファイルシステムポリシー)]、[編集] の順に選択します。
 5. ポリシーのオプションの場合は、設定済みのファイルシステムポリシーの任意の組み合わせを選択できます。
 - デフォルトでルートアクセスを防止する - このオプションは、許可された EFS アクションの設定から ClientRootAccess を削除します。
 - デフォルトで読み取り専用アクセスを強制する - このオプションは、許可された EFS アクションの設定から ClientWriteAccess を削除します。
 - 匿名アクセスを防止する - このオプションは、許可された EFS アクションの設定から ClientMount を削除します。
 - すべてのクライアントに転送中の暗号化を強制する - このオプションは、暗号化されていないクライアントへのアクセスを拒否します。
- 事前設定されたポリシーを選択すると、ポリシーの JSON オブジェクトが[Policy editor(ポリシーエディター)]ペインに表示されます。
6. 追加のアクセス許可を付与を使用して、別のプリンシパルを含む追加の IAM プリンシパルにファイルシステムアクセス許可を付与します AWS アカウント。[Add(追加)] を選択し、アクセス許可を付与するエンティティのプリンシパル ARN を入力します。付与する[Permissions(アクセス許可)]を選択します。追加のアクセス許可は、[Policy editor(ポリシーエディター)]で表示されます。

7. [Policy editor(ポリシーエディター)]を使用して、事前に設定されたポリシーをカスタマイズしたり、独自のファイルシステムポリシーを作成したりできます。エディタを使用すると、事前設定されたポリシーオプションは使用できなくなります。現在のファイルシステムポリシーをクリアして新しいポリシーの作成を開始するには、[Clear(クリア)]を選択します。

エディタをクリアすると、事前設定されたポリシーが再度使用可能になります。

8. ポリシーの編集が完了したら、[Save(保存)]を選択します。

の使用 AWS CLI

次の例では、[put-file-system-policy](#) CLI コマンドは、指定された EFS ファイルシステムへの AWS アカウント 読み取り専用アクセスを許可するファイルシステムポリシーを作成します。同等の API コマンドは [PutFileSystemPolicy](#) です。

```
aws efs put-file-system-policy --file-system-id fs-01234567 --policy '{
  "Id": "1",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "elasticfilesystem:ClientMount"
      ],
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:root"
      }
    }
  ]
}'
```

```
{
  "FileSystemId": "fs-01234567",
  "Policy": "{
    \"Version\" : \"2012-10-17\",
    \"Id\" : \"1\",
    \"Statement\" : [
      {
        \"Sid\" : \"efs-statement-7c8d8687-1c94-4fdc-98b7-555555555555\",
        \"Effect\" : \"Allow\",
        \"Principal\" : {
          \"AWS\" : \"arn:aws:iam::111122223333:root\"
        }
      }
    ]
  }"
```

```
    },  
    "Action" : [  
        "elasticfilesystem:ClientMount"  
    ],  
    "Resource" : "arn:aws:elasticfilesystem:us-east-2:555555555555:file-system/  
fs-01234567"  
    }  
  ]  
}
```

アクセスポイントの作成

Amazon EFS アクセスポイントを作成および削除するには、AWS Command Line Interface (AWS CLI)、AWS Management Console、Amazon EFS API および SDKsを使用します。アクセスポイントは一度作成すると変更できません。ファイルシステムは、引き上げをリクエストしない限り、最大 10,000 個のアクセスポイントを持つことができます。

Note

同じファイルシステムにアクセスポイントを作成する複数のリクエストが連続して送信され、ファイルシステムがアクセスポイントの制限に近づいている場合、これらのリクエストに対してスロットリングレスポンスが発生する可能性があります。これは、ファイルシステムが指定されたアクセスポイントのクォータを超えないようにするためです。

EFS アクセスポイントの詳細については、「[アクセスポイントの使用](#)」を参照してください。

コンソールを使用する

1. Amazon Elastic File System コンソール の<https://console.aws.amazon.com/efs/> を開いてください。
2. [アクセスポイント]を選択して、[アクセスポイント] ウィンドウを開きます。
3. [アクセスポイントの作成] を選択して、[アクセスポイントの作成] ページを表示します。

また、[ファイルシステム] を選択して、[アクセスポイントの作成] ページを開くこともできます。[ファイルシステム名]または[ファイルシステムID]を選択し、その後[アクセスポイント]および[アクセスポイントの作成]を選択して、そのファイルシステム用のアクセスポイントを作成します。

- a. [Details (詳細)] パネルに、次の情報を入力します。
- ファイルシステム - ファイルシステム名または ID を入力し、一致するファイルシステムを選択します。入力フィールドを選択したときに表示されるリストからファイルシステムを選択することもできます。
 - (オプション) [Name (名前)] - アクセスポイントの名前を入力します。
 - (オプション)[Root directory path(ルートディレクトリパス)]— アクセスポイントのルートディレクトリを指定できます。デフォルトのアクセスポイントのルートは / です。ルートディレクトリパスを入力するには、次の形式 /foo/bar を使用します。詳細については、「[アクセスポイントを使用したルートディレクトリの適用](#)」を参照してください。
- b. (オプション)[POSIX ユーザー] パネルでは、アクセスポイントを使用する NFS クライアントによるすべてのファイル操作に対して、ユーザーおよびグループ情報を適用するために使用している、完全な [POSIX ID] を指定できます。詳細については、「[アクセスポイントを使用したユーザー ID の適用](#)」を参照してください。
- ユーザー ID - ユーザーの POSIX ユーザー ID を数値で入力します。
 - グループ ID - ユーザーの POSIX グループ ID を数値で入力します。
 - セカンダリグループ ID— セカンダリグループ ID のカンマ区切りリスト (オプション) を入力します。
- c. (オプション) ルートディレクトリの作成許可では、Amazon EFS がルートディレクトリパスを作成するときに使用するアクセス許可を指定できます (指定がされており、ルートディレクトリがまだ存在しない場合)。詳細については、「[アクセスポイントを使用したルートディレクトリの適用](#)」を参照してください。

 Note

ルートディレクトリの所有権とアクセス許可を指定せず、ルートディレクトリがまだ存在しない場合、EFS はルートディレクトリを作成しません。アクセスポイントを使用してファイルシステムをマウントしようとすると、失敗します。

- 所有者ユーザー ID - ルートディレクトリの所有者として使用する POSIX ユーザー ID を数値で入力します。
- 所有者グループ ID - ルートディレクトリの所有者グループとして使用する POSIX グループ ID を数値で入力します。

- アクセス許可 - ディレクトリの Unix モードを入力します。一般的な設定は 755 です。アクセスポイントユーザーがマウントできるように、実行ビットが設定されていることを確認します。

4. [アクセスポイントの作成] を選択して、この設定を使用したアクセスポイントを作成します。

の使用 AWS CLI

以下の例では、create-access-point CLI コマンドを使用して EFS ファイルシステムのアクセスポイントを作成します。同等の API コマンドは [CreateAccessPoint](#) です。

```
aws efs create-access-point --file-system-id fs-abcdef0123456789a --client-token
010102020-3 \
--root-directory "Path=/efs/mobileapp/
east,CreationInfo={OwnerId=0,OwnerGid=11,Permissions=775}" \
--posix-user "Uid=22,Gid=4" \
--tags Key=Name,Value=east-users
```

リクエストが成功すると、CLI はアクセスポイントの記述で応答します。

```
{
  "ClientToken": "010102020-3",
  "Name": "east-users",
  "AccessPointId": "fsap-abcd1234ef5678901",
  "AccessPointArn": "arn:aws:elasticfilesystem:us-east-2:111122223333:access-point/
fsap-abcd1234ef5678901",
  "FileSystemId": "fs-01234567",
  "LifecycleState": "creating",
  "OwnerId": "111122223333",
  "PosixUser": {
    "Gid": 4,
    "Uid": 22
  },
  "RootDirectory": {
    "CreationInfo": {
      "OwnerGid": 0,
      "OwnerId": 11,
      "Permissions": "775"
    },
    "Path": "/efs/mobileapp/east",
  },
  "Tags": []
}
```

}

 Note

同じファイルシステムにアクセスポイントを作成する複数のリクエストが連続して送信され、ファイルシステムがアクセスポイントの制限に近づいている場合、これらのリクエストに対してスロットリングレスポンスが発生する可能性があります。これは、ファイルシステムが指定されたアクセスポイントのクォータを超えないようにするためです。

アクセスポイントの削除

アクセスポイントを削除すると、アクセスポイントを使用しているすべてのクライアントは、そのアクセスポイントが設定した Amazon EFS ファイルシステムへのアクセスを失います。

コンソールを使用する

1. Amazon Elastic File System コンソール の <https://console.aws.amazon.com/efs/> を開いてください。
2. 左のナビゲーションペインで [Access points(アクセスポイント)] を選択して、[Access points(アクセスポイント)] ページを開きます。
3. 削除するアクセスポイントを選択します。
4. [Delete(削除)] を選択します。
5. [Confirm(確認)]を選択してアクションを確認し、アクセスポイントを削除します。

の使用 AWS CLI

次の例では、delete-access-point CLI コマンドは、指定したアクセスポイントを削除します。同等の API コマンドは [DeleteAccessPoint](#) です。コマンドが成功した場合、サービスは空の HTTP 本文を持つ HTTP 204 レスポンスを返します。

```
aws efs delete-access-point --access-point-id fsap-092e9f80b3fb5e6f3 --client-token 010102020-3
```

EFS リソースのタグ付け

EFS リソースを管理しやすくするために、各リソースに独自のメタデータをタグの形で割り当てることができます。タグを使用すると、目的、所有者、環境など、さまざまな方法で AWS リソースを分類できます。これは同じ型のリソースが多い場合に役立ちます。割り当てたタグに基づいて特定のリソースをすばやく識別できます。ここではタグとその作成方法について説明します。

タグの基本

タグは、AWS リソースに割り当てるラベルです。タグはそれぞれ、1 つのキーとオプションの 1 つの値で設定されており、どちらもお客様側が定義します。

タグを使用すると、目的、所有者、環境など、さまざまな方法で AWS リソースを分類できます。たとえば、各ファイルシステムの所有者を追跡しやすくするため、アカウントの Amazon EFS ファイルシステムに対して一連のタグを定義できます。

各リソースタイプのニーズを満たす一連のタグキーを考案することをお勧めします。一貫性のある一連のタグキーを使用することで、リソースの管理が容易になります。追加したタグに基づいてリソースを検索およびフィルタリングできます。

タグには、Amazon EFS に関連する意味はなく、完全に文字列として解釈されます。また、タグは自動的にリソースに割り当てられます。タグのキーと値は編集でき、タグはリソースからいつでも削除できます。タグの値を空の文字列に設定することはできますが、タグの値を null に設定することはできません。特定のリソースについて既存のタグと同じキーを持つタグを追加した場合、以前の値は新しい値によって上書きされます。リソースを削除すると、リソースのタグも削除されます。

タグの制限

タグには以下のような基本制限があります。

- リソースあたりのタグの最大数 - 50 件
- タグキーはリソースごとにそれぞれ一意である必要があります。また、各タグキーに設定できる値は 1 つのみです。
- キーの最大長 - UTF-8 の 128 Unicode 文字
- 値の最大長 - UTF-8 の 256 Unicode 文字
- Amazon EFS ではタグ内に任意の文字を使用できますが、他のサービスでは制限があります。すべてのサービスで使用できる文字は、UTF-8 で表現できる文字、数字、およびスペースに加えて、+ - = . _ : / @ です。

- タグのキーと値は大文字と小文字が区別されます。
- `aws:` プレフィックスは AWS 使用のために予約されています。タグにこのプレフィックスが付いたタグキーがある場合、タグのキーまたは値を編集、削除することはできません。`aws:` プレフィックスを持つタグはリソースあたりのタグ数の制限時には計算されません。

タグのみに基づいてリソースを更新・削除することはできないので、リソース識別子を指定する必要があります。例えば、DeleteMe というタグキーを使用してタグ付けしたファイルシステムを削除するには、`fs-1234567890abcdef0` のようなファイルシステムのリソース識別子で DeleteFileSystem アクションを使用する必要があります。

公開リソースまたは共有リソースにタグを付けると、割り当てたタグは、AWS アカウントのみが使用できます。他の AWS アカウント はこれらのタグにアクセスできません。共有リソースへのタグベースのアクセスコントロールの場合、各 リソースへのアクセスを制御するために独自のタグセットを割り当てる AWS アカウント 必要があります。

Amazon EFS ファイルシステムおよびアクセスポイントリソースにタグ付けすることができます。

アクセスコントロールにタグを使用する

タグを使用すると、Amazon EFS リソースへのアクセスをコントロールしたり、属性ベースのアクセスコントロール (ABAC) を実装したりできます。

Note

レプリケーションは、属性ベースのアクセス制御 (ABAC) でのタグの使用をサポートしていません。

リソースのタグ付け

アカウントにすでに存在する Amazon EFS ファイルシステムおよびアクセスポイントリソースにタグ付けできます。

コンソールを使用する

- Amazon EFS コンソールでは、リソースの詳細画面の [Tags(タグ)] タブを使って、既存のリソースにタグを適用することができます。Amazon EFS コンソールでは、リソースの作成時にリソースのタグを指定することができます。たとえば、Name キーと指定した値を含むタグを追加できます。

ほとんどの場合、リソースの作成後すぐに (リソースの作成時ではなく) コンソールによりタグが適用されます。コンソールではリソースを Name タグに応じて整理できますが、このタグには Amazon EFS サービスに対する意味論的意味はありません。

の使用 AWS CLI

- Amazon EFS API、AWS CLI、または AWS SDK を使用している場合は、TagResourceEFS API アクションを使用して既存のリソースにタグを適用できます。さらに、リソース作成アクションによってはリソースの作成時にリソースのタグを指定できます。

タグを管理するための AWS CLI コマンド、および同等の Amazon EFS API アクションを次の表に示します。

CLI コマンド	説明	同等の API オペレーション
tag-resource	新しいタグを追加する、または既存のタグを更新する	TagResource
list-tags-for-resource	既存のタグの取得	ListTagsForResource
untag-resource	既存のタグの削除	UntagResource

チュートリアル: 書き込み可能なユーザーごとのサブディレクトリを作成する

EFS ファイルシステムを作成し、Amazon EC2 (EC2) インスタンスにローカルにマウントすると、**#####**と呼ばれる空のディレクトリが公開されます。このファイルシステムルートディレクトリの一般的なユースケースの 1 つは、EC2 インスタンスに作成したユーザーごとに「書き込み可能」なサブディレクトリを作成し、そのサブディレクトリをユーザーのホームディレクトリにマウントすることです。これにより、ユーザーがホームディレクトリに作成したすべてのファイルとサブディレクトリは、EFS ファイルシステム上に作成されます。

 Note

「[入門](#)」の演習に従うと、EC2 インスタンスに EFS ファイルシステムを作成してマウントできます。

以下の手順では、ユーザーを作成し、ユーザー用のサブディレクトリを作成して、ユーザーをサブディレクトリの所有者にします。次に、Amazon EFS サブディレクトリをユーザーのホームディレクトリにマウントします。

1. ユーザー mike の作成:

- EC2 インスタンスにログインします。ルート権限を使用して (この場合は `sudo` コマンドを使用して)、ユーザーを作成し、パスワードを割り当てます。

例えば、次のコマンドはユーザー `mike` を作成します。

```
$ sudo useradd -c "Mike Smith" mike
$ sudo passwd mike
```

ユーザーのホームディレクトリも作成されます。例えば、`/home/mike`。

2. `EFSroot` の下にユーザー用のサブディレクトリを作成します。

例えば、次のコマンドは、`EFSroot` の下に `mike` サブディレクトリを作成します。

```
$ sudo mkdir /EFSroot/mike
```

`EFSroot` をローカルディレクトリ名と置き換える必要があります。

- #### 3. サブディレクトリの所有者は、ルートユーザーとルートグループになります (`ls -l` コマンドを使用してこれを確認できます)。このサブディレクトリに対する完全なアクセス許可をユーザーに与えるには、ディレクトリの所有権をユーザーに付与します。

例 :

```
$ sudo chown mike:mike /EFSroot/mike
```

- #### 4. `mount` コマンドを使用して、サブディレクトリをユーザーのホームディレクトリにマウントします。

例：

```
$ sudo mount -t nfs -o  
nfsvers=4.1,rsiz=1048576,wsiz=1048576,hard,timeo=600,retrans=2,noresvport mount-  
target-DNS:/mike /home/mike
```

mount-target-DNS は、リモート EFS ファイルシステムのルートを識別するアドレスです。

このマウントターゲットをアンマウントすると、再マウントしない限り、ユーザーはディレクトリにアクセスできなくなります。再マウントにはルート権限が要求されます。

EFS ファイルシステムをマウントする

EFS ファイルシステムをマウントするには、EFS マウントヘルパーを使用することをお勧めします。EFS マウントヘルパーを使用すると、サポートされているディストリビューションを実行している EC2 Linux および Mac インスタンスに EFS ファイルシステムをマウントできます。マウントヘルパーは、Amazon EFS クライアント (amazon-efs-utils) のインストール時にインストールされるオープンソースのツールコレクションの一部です。Amazon EFS クライアントとサポートされているディストリビューションの詳細については、「[Amazon EFS クライアントの手動インストール](#)」を参照してください。

または、標準の Linux NFS クライアントを使用して EFS ファイルシステムを手動でマウントすることもできます。Amazon EFS は、ファイルシステムを Amazon EC2 インスタンスにマウントするときに、[Network File System(ネットワークファイルシステム)] バージョン 4.0 と 4.1 (NFSv4) プロトコルをサポートします。

さらに、EFS マウントヘルパーまたは NFS を使用して EC2 インスタンスを設定することで、インスタンスの起動時に自動的に EFS ファイルシステムをマウントできます。

トピック

- [Linux でのマウントに関する考慮事項](#)
- [EFS マウントヘルパーを使用した EFS ファイルシステムのマウント](#)
- [ネットワークファイルシステムを使用した EFS ファイルシステムのマウント](#)
- [EFS ファイルシステムの自動マウント](#)
- [ファイルシステムをアンマウントする](#)
- [チュートリアル: EFS ファイルシステムを作成し、を使用して EC2 インスタンスにマウントする AWS CLI](#)
- [チュートリアル: オンプレミス Linux クライアントを使用してマウントする](#)
- [チュートリアル: 別の VPC からファイルシステムをマウントする](#)
- [マウントの問題のトラブルシューティング](#)

Linux でのマウントに関する考慮事項

Linux の マウントオプションには次の値をお勧めします。

- `rsz=1048576` - 各ネットワーク読み取りリクエストに対して NFS クライアントが受信できるデータの最大バイト数を設定します。この値は、EFS ファイルシステム上のファイルからデータを読み取る際に適用されます。パフォーマンスが低下しないように、可能な限り大きいサイズ (最大 1048576) を使用することをお勧めします。
- `wsz=1048576` - 各ネットワーク書き込みリクエストに対して NFS クライアントが送信できるデータの最大バイト数を設定します。この値は、EFS ファイルシステム上のファイルにデータを書き込む際に適用されます。パフォーマンスが低下しないように、可能な限り大きいサイズ (最大 1048576) を使用することをお勧めします。
- `hard` - NFS リクエストがタイムアウトした後の NFS クライアントのリカバリ動作を設定します。これにより、NFS リクエストは、サーバーが応答するまで無期限に再試行されます。データの整合性を確保できるように、ハードマウントオプション (`hard`) を使用することをお勧めします。`soft` マウントを使用する場合は、`timeo` パラメータを 150 デシ秒 (15 秒) 以上に設定してください。これにより、ソフトマウントに固有のデータ破損が生じるリスクを最小限に抑えることができます。
- `timeo=600` - NFS クライアントがレスポンスを待機するのに要するタイムアウト値を設定してから、NFS リクエストを 600 デシ秒 (60 秒) に設定します。タイムアウトパラメータ (`timeo`) を変更する必要がある場合は、少なくとも 150 の値を使用することをお勧めします。これは 15 秒に相当します。これにより、パフォーマンスの低下を避けることができます。
- `retrans=2` - NFS クライアントがさらなるリカバリアクションを試行する前にリクエストを再試行する回数を 2 回に設定します。
- `noresvport` - ネットワーク接続が再確立された時に、新しい非特権の Transmission Control Protocol (TCP) 送信元ポートを使用するように、NFS クライアントに指示します。これにより、ネットワーク復旧イベント後、EFS ファイルシステムでの中断のない可用性が保証されます。
- `_netdev` - `/etc/fstab` に存在する場合、クライアントは、ネットワークが有効になるまで、EFS ファイルシステムをマウントすることはできません。

一般に、デフォルトとは異なる他のマウントオプションを設定しないでください。パフォーマンスが低下し、別の問題が生じる可能性があります。前述のデフォルト設定を使用しない場合は、以下の点に注意してください。

- 読み取りまたは書き込みバッファサイズを変更したり、属性のキャッシュを無効にしたりするとパフォーマンスが損なわれる可能性があります。
- Amazon EFS は送信元ポートを無視します。Amazon EFS 送信元ポートを変更した場合、変更は無効になります。

- Amazon EFS では Kerberos セキュリティバリエーションをサポートしていません。たとえば、以下のマウントコマンドは失敗します。

```
$ mount -t nfs4 -o krb5p <DNS_NAME>:/ /efs/
```

- ファイルシステムは、DNS 名を使用してマウントすることをお勧めします。Amazon EFS は、外部リソースを呼び出すことなく、この名前を Amazon EC2 インスタンスと同じアベイラビリティゾーン内の EFS マウントターゲットの IP アドレスに解決します。ご使用の EC2 インスタンスとは異なるアベイラビリティゾーンでマウントターゲットを使用すると、アベイラビリティゾーンから送信されるデータに対して EC2 の標準料金がかかります。また、ファイルシステムオペレーションのレイテンシーが増加することがあります。
- 他のマウントオプションや、デフォルト設定の詳細な説明については、Linux のドキュメントを参照してください。

Note

マウントされた EFS ファイルシステムのステータスに関係なく EC2 インスタンスを起動する必要がある場合は、`/etc/fstab` ファイルのファイルシステムのエントリに `nofail` オプションを追加します。

EFS マウントヘルパーを使用した EFS ファイルシステムのマウント

Amazon EFS クライアント (`amazon-efs-utils`) をインストールしたら、EFS マウントヘルパーを使用して、[サポートされているディストリビューション](#)を実行している EC2 Linux および Mac インスタンスに EFS ファイルシステムをマウントできます。Amazon EFS は EC2 Windows インスタンスからのマウントをサポートしていません。

Important

マウントを成功させる `amazon-efs-utils` には、常に最新バージョンのを使用することをお勧めします。例えば、2.3 `amazon-efs-utils` より前のバージョンのでは、IPv6 アドレスを使用したマウントはサポートされていません。

ファイルシステムをマウントするとき、マウントヘルパーは `efs` という新しいネットワークファイルシステムタイプを定義します。これには Linux の標準的な `mount` コマンドと完全な互換性があります。マウントヘルパーは、EC2 Linux インスタンス/`etc/fstab` の設定ファイル内のエントリを使用して、インスタンスの起動時に EFS ファイルシステムを自動的にマウントすることもできます。

⚠ Warning

ファイルシステムを自動的にマウントする場合、ネットワークファイルシステムを識別するために使用された `_netdev` オプションを使用します。`_netdev` が見つからない場合、EC2 インスタンスはレスポンスを停止する可能性があります。この結果は、コンピューティングインスタンスがネットワークを開始後、ネットワークファイルシステムを初期化する必要があります。詳細については、「[自動マウントが失敗してインスタンスがレスポンスしない](#)」を参照してください。

ファイルシステムをマウントするには、次のいずれかのプロパティを指定します。

- ファイルシステムの DNS 名 – ファイルシステムの DNS 名を使用し、マウントヘルパーがそれを解決できない場合 (たとえば、ファイルシステムを別の VPC にマウントする場合など)、マウントターゲットの IP アドレスを使用するようにフォールバックします。詳細については、「[別の AWS アカウント または VPC からの EFS ファイルシステムのマウント](#)」を参照してください。
- ファイルシステム ID – ファイルシステム ID を使用する場合、マウントヘルパーは、外部リソースを呼び出さずに、マウントターゲットの Elastic Network Interface (ENI) のローカル IP アドレスへ解決します。
- マウントターゲット IP アドレス – ファイルシステムのマウントターゲットのいずれかの IP アドレスを使用できます。

Amazon EFS コンソールで、これらのプロパティの値をすべて確認できます。ファイルシステムの DNS 名は、アタッチ 画面にあります。

転送中のデータの暗号化が EFS ファイルシステムのマウントオプションとして宣言されると、マウントヘルパーはクライアント `stunnel` プロセスと と呼ばれるスーパーバイザープロセスを初期化します `amazon-efs-mount-watchdog`。 `amazon-efs-mount-watchdog` プロセスは TLS マウントの状態を監視し、EFS ファイルシステムが初めて TLS にマウントされたときに自動的に開始されます。クライアントが Linux 上で動作している場合、このプロセスは Linux ディストリビューションによって、`upstart` または `systemd` のどちらか一方により管理されます。サポートされている macOS 上で動作するクライアントの場合は、`launchd` によって管理されます。

Stunnel はオープンソースの多目的ネットワークリレーです。クライアント stunnel プロセスはインバウンドトラフィックのローカルポートをリッスンし、マウントヘルパーは NFS クライアントトラフィックをこのローカルポートにリダイレクトします。

マウントヘルパーはファイルシステムとの通信に TLS バージョン 1.2 を使用します。TLS の使用には、信頼された Amazon 認証機関によって署名された証明書が必要です。暗号化の動作の詳細については、[Amazon EFS でのデータの暗号化](#) を参照してください。

トピック

- [EFS マウントヘルパーで使用されるマウント設定](#)
- [サポートログの取得](#)
- [EFS マウントヘルパーを使用するための前提条件](#)
- [EFS マウントヘルパーを使用した EC2 Linux インスタンスへのマウント EFS](#)
- [EFS マウントヘルパーを使用した EC2 Mac インスタンスへのマウント](#)
- [別の から EFS ファイルシステムをマウントする AWS リージョン](#)
- [1 ゾーンファイルシステムをマウントする](#)
- [IAM 認可を使用してマウントする](#)
- [Amazon EFS アクセスポイントを使用してマウントする](#)
- [複数の EC2 インスタンスへの EFS のマウント](#)
- [別の AWS アカウント または VPC からの EFS ファイルシステムのマウント](#)

EFS マウントヘルパーで使用されるマウント設定

Amazon EFS マウントヘルパークライアントは、Amazon EFS 用に最適化された次のマウントオプションを使用します。

- `nfsvers=4.1`— EC2 Linux インスタンスへのマウント時に使用

`nfsvers=4.0` — macOS Big Sur、Monterey、Ventura を実行している、サポートされている EC2 Mac インスタンスにマウントするときに使用されます
- `rsiz=1048576` - NFS クライアントが各ネットワーク書き込みリクエストで送信できるデータの最大バイト数を 1048576 に設定し、利用可能な最大値にすることで、パフォーマンスの低下を防ぎます。

- `wsiz=1048576` - NFS クライアントが各ネットワーク書き込みリクエストで送信できるデータの最大バイト数を 1048576 に設定し、利用可能な最大値にすることで、パフォーマンスの低下を防ぎます。
- `hard` - NFS リクエストがタイムアウトした後の NFS クライアントのリカバリ動作を設定し、サーバーが応答するまで NFS リクエストを無期限に再試行し、データの整合性を確保します。
- `timeo=600` - NFS クライアントがレスポンスを待機するのに要するタイムアウト値を設定してから、NFS リクエストを 600 デシ秒 (60 秒) に設定し、パフォーマンスの低下を防ぎます。
- `retrans=2` - NFS クライアントがさらなるリカバリアクションを試行する前にリクエストを再試行する回数を 2 回に設定します。
- `noresvport` - ネットワーク接続が再確立された時に、新しい非特権の Transmission Control Protocol (TCP) 送信元ポートを使用するように、NFS クライアントに指示します。`noresvport` オプションを使用すると、再接続やネットワーク回復イベントの後、EFS ファイルシステムの可用性が中断されることはありません。
- `mountport=2049`— macOS Big Sur、Monterey および Ventura を実行している EC2 Mac インスタンスにマウントするときのみ使用されます。

サポートログの取得

EFS マウントヘルパーには EFS ファイルシステムのログ記録が組み込まれています。これらのログは、トラブルシューティングの目的で AWS サポートと共有できます。EFS マウントヘルパーを使用して、クライアントの `/var/log/amazon/efs` に保存されているログを検索できます。これらのログは、EFS マウントヘルパー、`stunnel` プロセス (デフォルトでは無効)、`stunnel` プロセスをモニタリングする `amazon-efs-mount-watchdog` プロセスのためのものです。

Note

`amazon-efs-mount-watchdog` このプロセスにより、各マウントの `stunnel` プロセスが実行され、EFS ファイルシステムがアンマウントされると `stunnel` プロセスが停止します。何らかの理由で、`stunnel` プロセスが予期せず終了した場合、ウォッチドッグプロセスにより再開されます。

`/etc/amazon/efs/efs-utils.conf` でログの設定を変更できます。ログの変更を有効にするには、EFS マウントヘルパーを使用してファイルシステムをアンマウントし、再マウントする必要があります。マウントヘルパーおよびウォッチドッグログのログ容量は 20 MiB に制限されています。`stunnel` プロセスのログはデフォルトでは無効になっています。

⚠ Important

stunnel プロセスログのログ記録を有効にすることができます。ただし、stunnel ログを有効にするとファイルシステムのいくつかの容量が使用されます。

EFS マウントヘルパーを使用するための前提条件

Amazon EFS マウントヘルパーを使用して、EFS ファイルシステムを Amazon EC2 EFS インスタンスにマウントできます。マウントヘルパーを使用するには、以下が必要になります。

- マウントするファイルシステムのファイルシステム ID – EFS マウントヘルパーは、外部リソースを呼び出さずに、ファイルシステム ID をマウントターゲット Elastic Network Interface (ENI) のローカル IP アドレスを解決します。
- EFS マウントターゲット – Virtual Private Cloud (VPC) にマウントターゲットを作成します。サービス推奨設定を使用してコンソールでファイルシステムを作成すると、ファイルシステム AWS リージョン がある の各アベイラビリティゾーンにマウントターゲットが作成されます。マウントターゲットの作成手順については、「[マウントターゲットの管理](#)」を参照してください。

i Note

新しく作成したマウントターゲットのライフサイクルステータスが利用可能になってから 60 秒待ってから DNS 経由でファイルシステムをマウントすることをお勧めします。この待機により、DNS レコード AWS リージョン はファイルシステムが存在する に完全に伝播されます。

ご使用の EC2 インスタンスとは異なるアベイラビリティゾーンでマウントターゲットを使用すると、アベイラビリティゾーンから送信されるデータに対して EC2 の標準料金がかかります。また、ファイルシステムオペレーションのレイテンシーが増加することがあります。

- 別のアベイラビリティゾーンから 1 ゾーンファイルシステムをマウントするには:
 - ファイルシステムのアベイラビリティゾーンの名前 - EC2 インスタンスとは異なるアベイラビリティゾーンにある EFS 1 ゾーンファイルシステムをマウントする場合。
 - マウントターゲット DNS 名 – または、アベイラビリティゾーンの代わりにマウントターゲットの DNS 名を指定することもできます。

- サポートされている Linux または macOS ディストリビューションのいずれかを実行している EC2 インスタンス – マウントヘルパーを使用してファイルシステムをマウントするためにサポートされているディストリビューションは次のとおりです。
 - Amazon Linux 2
 - Amazon Linux 2023
 - Amazon Linux 2017.09 以降
 - macOS Big Sur
 - Red Hat Enterprise Linux (および、その派生物 CentOS など) バージョン 7 以降
 - Ubuntu 16.04 LTS 以降

 Note

macOS ビッグサーを実行している EC2 Mac インスタンスは NFS 4.0 のみをサポートします。

- EFS マウントヘルパーは EC2 インスタンスにインストールされます – マウントヘルパーはユーティリティの `amazon-efs-utils` パッケージのツールです。 `amazon-efs-utils` のインストールについては、「[Amazon EFS クライアントの手動インストール](#)」を参照してください。
- EC2 インスタンスは VPC 内にあります – T接続する EC2 インスタンスは、Amazon VPC サービスに基づいて Virtual Private Cloud (VPC) で作成する必要があります。また、 が提供する DNS サーバーを使用するように設定する必要があります AWS。 Amazon DNS サーバーの詳細については、「[Amazon VPC ユーザーガイド](#)」の「[Amazon VPC の DHCP オプションセット](#)」を参照してください。
- VPC が DNS のホスト名を有効にしました – 接続する EC2 インスタンスの VPC で DNS ホスト名が有効になっている必要があります。詳細については、「Amazon [VPC ユーザーガイド](#)」の「[VPC の DNS 属性](#)」を参照してください。
- 異なる の EC2 インスタンスとファイルシステム AWS リージョンの場合 – マウントする EC2 インスタンスとファイルシステムが異なる には AWS リージョン、 `efs-utils.conf` ファイル内の `region` プロパティを編集する必要があります。詳細については、「[別の から EFS ファイルシステムをマウントする AWS リージョン](#)」を参照してください。

EFS マウントヘルパーを使用した EC2 Linux インスタンスへのマウント EFS

この手順には、以下について示します。

- Amazon EC2 インスタンスに `amazon-efs-utils` パッケージをインストールしました。詳細については、「[Amazon EFS クライアントの手動インストール](#)」を参照してください。
- ファイルシステムにはマウントターゲットが作成されました。詳細については、「[マウントターゲットの管理](#)」を参照してください。

EC2 Linux インスタンスでマウントヘルパーを使用して EFS ファイルシステムをマウントするには

1. Secure Shell (SSH) を介して EC2 Mac インスタンスのターミナルにアクセスし、適切なユーザー名でログインします。詳細については、「Amazon EC2 ユーザーガイド」の「[EC2 インスタンスに接続する](#)」を参照してください。
2. 以下のコマンドを使用して、ファイルシステムのマウントポイントとして使用するディレクトリ `efs` を作成します。

```
sudo mkdir efs
```

3. 以下のコマンドの 1 つを使用してファイルシステムをマウントします。

 Note

EC2 インスタンスとマウントするファイルシステムが異なる AWS リージョンにある場合、[別の から EFS ファイルシステムをマウントする AWS リージョン](#) 参照して `efs-utils.conf` ファイルの `region` プロパティを編集してください。

- ファイルシステム ID を使用してマウントするには：

```
sudo mount -t efs file-system-id efs-mount-point/
```

efs-mount-point の代わりに *file-system-id* と `efs` にマウントするファイルシステムの ID を使用してください。

```
sudo mount -t efs fs-abcd123456789ef0 efs/
```

または、転送時にデータの暗号化を使用する場合、次のコマンドを使用してファイルシステムをマウントできます。

```
sudo mount -t efs -o tls fs-abcd123456789ef0:/ efs/
```

- ファイルシステム DNS 名を使用してマウントするには

```
sudo mount -t efs -o tls file-system-dns-name efs-mount-point/
```

```
sudo mount -t efs -o tls fs-abcd123456789ef0.efs.us-east-2.amazonaws.com efs/
```

- マウントターゲット IP アドレスを使用してマウントするには :

```
sudo mount -t efs -o tls,mounttargetip=mount-target-ip file-system-id efs-mount-point/
```

```
sudo mount -t efs -o tls,mounttargetip=192.0.2.0 fs-abcd123456789ef0 efs/
```

アタッチ ダイアログボックスにあるファイルシステムをマウントするために、正確なコマンドを表示およびコピーできます。

- a. Amazon EFS コンソールで、マウントするファイルシステムを選択して詳細ページを表示します。
- b. このファイルシステムで使用するマウントコマンドを表示するには、右上の [アタッチ] を選択します。

[アタッチ] 画面には、ファイルシステムをマウントするために使用するコマンドが次のように表示されます。

- (DNS 経由でマウントする) EFS マウントヘルパーまたは NFS クライアントでファイルシステムの DNS 名を使用します。
- (IP 経由でマウント) 選択したアベイラビリティゾーンのマウントターゲットの IP アドレスを NFS クライアントで使用します。

EFS マウントヘルパーを使用した EC2 Mac インスタンスへのマウント

この手順には、以下について示します。

- Amazon EC2 Mac インスタンスに `amazon-efs-utils` パッケージをインストールしました。詳細については、「[macOS Big Sur、macOS Monterey、または macOS Ventura を実行している EC2 Mac インスタンスに Amazon EFS クライアントをインストールする](#)」を参照してください。

- ファイルシステムにはマウントターゲットが作成されました。ファイルシステムの作成時にマウントターゲットを作成し、既存のファイルシステムに追加できます。詳細については、「[マウントターゲットの管理](#)」を参照してください。
- macOS Big Sur、Monterey、または Ventura を実行している EC2 Mac インスタンスにファイルシステムをマウントしています。他の macOS バージョンはサポートされていません。

Note

macOS Big Sur、Monterey、および Ventura を実行している EC2 Mac インスタンスのみがサポートされています。その他の macOS バージョンは Amazon EFS での使用はサポートされていません。

macOS Big Sur、Monterey、または Ventura を実行している EC2 Mac インスタンスに EFS マウントヘルパーを使用して EFS ファイルシステムをマウントするには

1. Secure Shell (SSH) を介して EC2 Mac インスタンスのターミナルにアクセスし、適切なユーザー名でログインします。詳細については、「Amazon EC2 ユーザーガイド」の「[EC2 インスタンスに接続する](#)」を参照してください。
2. 次のコマンドを使用して、ファイルシステムのマウントポイントとして使用するためのディレクトリを作成します。

```
sudo mkdir efs
```

3. 以下のコマンドを実行してファイルシステムをマウントします。

Note

マウントコマンドの `tls` オプションを使用しなくても、デフォルトでは、EFS マウントヘルパーは、EC2 Mac インスタンスへのマウント時に、転送中に暗号化を使用し

```
sudo mount -t efs file-system-id efs-mount-point/
```

```
sudo mount -t efs fs-abcd123456789ef0 efs/
```

また、マウントするときに `tls` オプションを使用できます。

```
sudo mount -t efs -o tls fs-abcd123456789ef0:/ efs
```

転送中に暗号化を使用せずに EC2 Mac インスタンスにファイルシステムをマウントするには、`notls` オプションを使用します (次のコマンドを参照)。

```
sudo mount -t efs -o notls file-system-id efs-mount-point/
```

以下に説明するように、アタッチ ダイアログボックスにある管理コンソールでファイルシステムをマウントするために、正確なコマンドを表示およびコピーできます。

- Amazon EFS コンソールで、マウントするファイルシステムを選択して詳細ページを表示します。
- このファイルシステムで使用するマウントコマンドを表示するには、右上の [アタッチ] を選択します。

[アタッチ] 画面には、ファイルシステムをマウントするために使用するコマンドが次のように表示されます。

- (DNS 経由でマウントする) EFS マウントヘルパーまたは NFS クライアントでファイルシステムの DNS 名を使用します。
- (IP 経由でマウント) 選択したアベイラビリティゾーンのマウントターゲットの IP アドレスを NFS クライアントで使用します。

別の から EFS ファイルシステムをマウントする AWS リージョン

ファイルシステム AWS リージョン とは異なる にある EC2 インスタンスから EFS ファイルシステムをマウントするには、`efs-utils.conf` ファイル内の `region` プロパティ値を編集する必要があります。

efs-utils.conf の **region** プロパティを編集するには

- Secure Shell (SSH) を介して EC2 インスタンスのターミナルにアクセスし、適切なユーザー名でログインします。詳細については、「Amazon EC2 ユーザーガイド」の「[EC2 インスタンスに接続する](#)」を参照してください。
- 任意のエディタを使用して `efs-utils.conf` ファイルを検索して開きます。

3. 以下の行を見つけます。

```
#region = us-east-1
```

- a. 行のコメントを解除します。
 - b. ファイルシステムが us-east-1 リージョンにある場合、us-east-1 とファイルシステムが配置されているリージョンの ID を置換してください。
 - c. 変更を保存します。
4. クロスリージョンマウント用のホストエントリを追加します。方法の詳細については、「[ステップ 3: マウントターゲットにホストエントリを追加する](#)」を参照してください。
 5. [Linux](#) または [Mac](#) インスタンス用の EFS マウントヘルパーを使用してファイルシステムをマウントします。

1 ザーンファイルシステムをマウントする

EFS 1 ザーンファイルシステムは、ファイルシステムと同じアベイラビリティゾーンにある 1 つのマウントターゲットのみをサポートします。マウントターゲットは追加できません。このセクションでは、1 ザーンファイルシステムをマウントする際に考慮すべき事項について説明します。

アベイラビリティゾーン間のデータ転送料金を回避し、ファイルシステムのマウントターゲットと同じアベイラビリティゾーンにある Amazon EC2 コンピューティングインスタンスを使用して EFS ファイルシステムにアクセスすることで、パフォーマンスを向上させることができます。

このセクションの手順では、次が必要となります。

- EC2 インスタンスに `amazon-efs-utils` package をインストールしました。詳細については、「[Amazon EFS クライアントの手動インストール](#)」を参照してください。
- ファイルシステムにマウントターゲットを作成しました。詳細については、「[マウントターゲットの管理](#)」を参照してください。

別のアベイラビリティゾーンの EC2 で 1 ザーンファイルシステムをマウントする

別のアベイラビリティゾーンにある Amazon EC2 インスタンスに 1 ザーンファイルシステムをマウントする場合は、マウントヘルパーマウントコマンドでファイルシステムのアベイラビリティゾーン名またはファイルシステムのマウントターゲットの DNS 名を指定する必要があります。

次のコマンドを使用して、ファイルシステムマウントポイントとして使用する `efs` というディレクトリを作成します。

```
sudo mkdir efs
```

EFS マウントヘルパーを使用してファイルシステムをマウントするには、次のコマンドを使用します。コマンドは、ファイルシステムのアベイラビリティゾーン名を指定します。

```
sudo mount -t efs -o az=availability-zone-name,tls file-system-id mount-point/
```

これは、サンプル値を含むコマンドです。

```
sudo mount -t efs -o az=us-east-1a,tls fs-abcd1234567890ef efs/
```

次のコマンドは、ファイルシステムをマウントし、マウントターゲットの DNS 名を指定します。

```
sudo mount -t efs -o tls mount-target-dns-name mount-point/
```

これは、マウントターゲット DNS 名の例を含むコマンドです。

```
sudo mount -t efs -o tls us-east-1a.fs-abcd1234567890ef9.efs.us-east-1.amazonaws.com  
efs/
```

EFS マウントヘルパーを使用して、1 ゾーンファイルシステムを別のアベイラビリティゾーンに自動的にマウントする

別のアベイラビリティゾーンに配置されている EC2 インスタンスに EFS 1 ゾーンファイルシステムをマウントするために `/etc/fstab` を使用する場合は、ファイルシステムのアベイラビリティゾーン名またはファイルシステムのマウントターゲットの DNS 名を `/etc/fstab` エントリに指定する必要があります。

```
availability-zone-name.file-system-id.efs.aws-region.amazonaws.com:/ efs-mount-point  
efs defaults,_netdev,noresvport,tls 0 0
```

```
us-east-1a.fs-abc123def456a7890.efs.us-east-1.amazonaws.com:/ efs-one-zone efs  
defaults,_netdev,noresvport,tls 0 0
```


1 ゾーンファイルシステムを自動的にマウントする

1 ゾーンストレージを使用して EC2 インスタンスを別のアベイラビリティゾーンにある EFS ファイルシステムをマウントするために `/etc/fstab` を使用の場合は、ファイルシステムのアベイラビリティゾーン名とファイルシステムの DNS 名を `/etc/fstab` エントリに指定する必要があります。

```
availability-zone-name.file-system-id.efs.aws-region.amazonaws.com:/ efs-mount-point  
nfs4  
nfsvers=4.1,rsize=1048576,wsiz=1048576,hard,timeo=600,retrans=2,noresvport,_netdev 0  
0
```

```
us-east-1a.fs-abc123def456a7890.efs.us-east-1.amazonaws.com:/ efs-one-zone nfs4  
nfsvers=4.1,rsize=1048576,wsiz=1048576,hard,timeo=600,retrans=2,noresvport,_netdev 0  
0
```

`/etc/fstab` ファイルの編集方法と、このコマンドで使用される値の詳細については、「[EFS ファイルシステムの自動マウント](#)」を参照してください。

1 ゾーンファイルシステムを他の AWS コンピューティングインスタンスにマウントする

Amazon Elastic Container Service、Amazon Elastic Kubernetes Service、またはで 1 ゾーンファイルシステムを使用する場合は AWS Lambda、EFS ファイルシステムが存在するのと同じアベイラビリティゾーンを使用するようにサービスを設定する必要があります。以下に示し、以下のセクションで説明します。



Amazon Elastic Container サービスからの接続

Amazon ECS で EFS ファイルシステムを使用すると、コンテナインスタンスのフリート間でファイルシステムデータを共有できるため、タスクが配置されているインスタンスに関係なく、同じ永続的ストレージにアクセスできます。Amazon ECS で EFS One Zone ファイルシステムを使用するには、タスクの起動時にファイルシステムと同じアベイラビリティゾーンにあるサブネットのみを選択する必要があります。詳細については、Amazon Elastic Container Service デベロッパーガイドの「[Amazon EFS ボリューム](#)」を参照してください。

Amazon Elastic Kubernetes Service からの接続

Amazon EKS から 1 ゾーンファイルシステムをマウントする場合、EFS アクセスポイントをサポートする Amazon EFS [Container Storage Interface](#) (CSI) ドライバーを使用して、Amazon EKS またはセルフマネージド Kubernetes クラスター内の複数のポッド間でファイルシステムを共有できます。Amazon EFS CSI ドライバーは Fargate スタックにインストールされています。EFS 1 ゾーンファイルシステムで Amazon EFS CSI ドライバーを使用する場合、ポッドを起動するときに `nodeSelector` オプションを使用して、ファイルシステムと同じアベイラビリティゾーン内でスケジューラれるようにできます。EFS

からの接続 AWS Lambda

で Amazon EFS を使用すると AWS Lambda、関数の呼び出し間でデータを共有したり、大きな参照データファイルを読み取り、関数出力を永続的な共有ストアに書き込むことができます。Lambda は、関数インスタンスを同じアベイラビリティーゾーンとサブネットにある EFS マウントターゲットに安全に接続します。1 ゾーンファイルシステムで Lambda を使用する場合は、ファイルシステムと同じアベイラビリティーゾーンにあるサブネットへの呼び出しのみを起動するように関数を設定します。

IAM 認可を使用してマウントする

AWS Identity and Access Management (IAM) 認可を使用して EFS ファイルシステムを Linux インスタンスにマウントするには、EFS マウントヘルパーを使用します。NFS クライアントの IAM 認可の詳細については、「[IAM を使用してファイルシステムへのアクセスを制御する](#)」を参照してください。

以下のセクションの手順に従うには、ファイルシステムのマウントポイントとして使用するディレクトリを作成する必要があります。次のコマンドを使用して、マウントポイントディレクトリ `efs` を作成できます。

```
sudo mkdir efs
```

その後、*efs-mount-point* のインスタンスを `efs` に置き換えることができます。

EC2 インスタンスプロファイルを使用した IAM によるマウント

インスタンスプロファイル持つ Amazon EC2 インスタンスに対して IAM 認可によるマウントを行う場合は、次に示すように、マウントオプションとして `tls` と `iam` を使用します。

```
$ sudo mount -t efs -o tls,iam file-system-id efs-mount-point/
```

インスタンスプロファイルを持つ EC2 インスタンスに IAM 認可を使用して自動的にマウントするには、EC2 インスタンスの `/etc/fstab` ファイルに次の行を追加します。

```
file-system-id:/ efs-mount-point efs _netdev,tls,iam 0 0
```

名前付きプロファイルを使用した IAM によるマウント

認証情報 AWS CLI ファイルにある IAM 認証情報、または AWS CLI 設定ファイルを使用して `~/.aws/credentials`、IAM 認可を使用してマウントできます `~/.aws/`

config。"awsprofile" が指定されていない場合は、"デフォルト" のプロファイルが使用されます。

認証情報ファイルを使用して Linux インスタンスに対して IAM 認可によるマウントを行うには、次に示すように `tls`、`awsprofile`、`iam` マウントオプションを使用します。

```
$ sudo mount -t efs -o tls,iam,awsprofile=namedprofile file-system-id efs-mount-point/
```

認証情報ファイルを使用して Linux インスタンスに対して IAM 認可によるマウントを自動的に行うには、EC2 インスタンスの `/etc/fstab` ファイルに次の行を追加します。

```
file-system-id:/ efs-mount-point efs _netdev,tls,iam,awsprofile=namedprofile 0 0
```

Amazon EFS アクセスポイントを使用してマウントする

EFS マウントヘルパーを使用することによってのみ、EFS アクセスポイントを使用して EFS ファイルシステムをマウントできます。

Note

EFS アクセスポイントを使用してファイルシステムをマウントする場合、ファイルシステム用の 1 つ以上のマウントターゲットを設定する必要があります。

アクセスポイントを使用してファイルシステムをマウントする場合、`mount` コマンドには、通常のマウントオプションに加えて、`access-point-id` および `tls` マウントオプションが含まれます。以下に例を示します。

```
$ sudo mount -t efs -o tls,accesspoint=access-point-id file-system-id efs-mount-point
```

アクセスポイントを使用してファイルシステムを自動的にマウントするには、EC2 インスタンスの `/etc/fstab` ファイルに次の行を追加します。

```
file-system-id efs-mount-point efs _netdev,tls,accesspoint=access-point-id 0 0
```

EFS アクセスポイントの詳細については、「[アクセスポイントの使用](#)」を参照してください。

複数の EC2 インスタンスへの EFS のマウント

AWS Systems Manager Run Command を使用すると、各インスタンスにログインしなくても、リモートから安全に EFS ファイルシステムを複数の Amazon EC2 インスタンスにマウントできます。AWS Systems Manager Run Command の詳細については、AWS Systems Manager 「ユーザーガイド」の[AWS Systems Manager 「Run Command」](#)を参照してください。この方法を使用して EFS ファイルシステムをマウントする前に、次の前提条件が必要です。

1. EC2 インスタンスは、AmazonElasticFileSystemsUtils アクセス許可ポリシーを含むインスタンスプロファイルを使用して起動されます。詳細については、「[ステップ 1: 必要なアクセス許可を持つ IAM インスタンスプロファイルを設定する](#)」を参照してください。
2. バージョン 1.28.1 以降の Amazon EFS クライアント (amazon-efs-utils パッケージ) が EC2 インスタンスにインストールされています。AWS Systems Manager を使用して、インスタンスにパッケージを自動的にインストールできます。詳細については、「[ステップ 2: State Manager で使用される関連付けを設定する](#)」を参照してください。

コンソールを使用して複数の EFS ファイルシステムを複数の EC2 インスタンスにマウントするには

1. <https://console.aws.amazon.com/systems-manager/> で AWS Systems Manager コンソールを開きます。
2. ナビゲーションペインで、[Run Command] を選択します。
3. [Run a command] を選択します。
4. コマンド検索フィールドに「**AWS-RunShellScript**」と入力します。
5. AWS-RunShellScript を選択します。
6. コマンドパラメータに、マウントする EFS ファイルシステムごとに使用するマウントコマンドを入力します。例:

```
sudo mount -t efs -o tls fs-12345678:/ /mnt/efs
sudo mount -t efs -o tls,accesspoint=fsap-12345678 fs-01233210 /mnt/efs
```

Amazon EFS クライアントを使用した EFS マウントコマンドの詳細については、「[EFS マウントヘルパーを使用した EC2 Linux インスタンスへのマウント EFS](#)」または「[EFS マウントヘルパーを使用した EC2 Mac インスタンスへのマウント](#)」を参照してください。

7. コマンドを実行するターゲット AWS Systems Manager マネージド EC2 インスタンスを選択します。

8. その他の追加設定を行ってください。Run (実行) を選択し、コマンドで指定された EFS ファイルシステムをマウントします。

コマンドを実行すると、そのステータスがコマンド履歴に表示されます。

別の AWS アカウント または VPC からの EFS ファイルシステムのマウント

EFS マウントヘルパーを使用して、NFS クライアントと EFS アクセスポイントの IAM 認可を使用して EFS ファイルシステムをマウントできます。既定では、EFS マウントヘルパーは、ドメインネームサービス (DNS) を使用して、EFS マウントターゲットの IP アドレスを解決します。別のアカウントまたは Virtual Private Cloud (VPC) からファイルシステムをマウントする場合は、EFS マウントターゲットを手動で解決する必要があります。

次に、NFS クライアントに使用する正しい EFS マウントターゲットの IP アドレスを決定する手順を示します。また、その IP アドレスを使用して EFS ファイルシステムがマウントされるようにクライアントを設定する方法についても説明します。

トピック

- [別の から EFS ファイルシステムをマウントする AWS アカウント](#)
- [別の VPC からの EFS ファイルシステムのマウント](#)

別の から EFS ファイルシステムをマウントする AWS アカウント

共有 VPCs を使用すると、別の が所有する Amazon EC2 インスタンス AWS アカウント から 1 つが所有する EFS ファイルシステムをマウントできます AWS アカウント。VPC サブネット共有の詳細については、「Amazon VPC ピアリングガイド」の「[VPC を他のアカウントと共有する](#)」を参照してください。

VPC 共有をセットアップしたら、EC2 インスタンスで、ドメインネームシステム (DNS) 名前解決または EFS マウントヘルパーを使用して EFS ファイルシステムをマウントできるようになります。EFS マウントヘルパーを使用して、EFS ファイルシステムをマウントすることをお勧めします。

別の VPC からの EFS ファイルシステムのマウント

VPC ピアリング接続またはトランジットゲートウェイを使用して VPC に接続する場合、ある VPC 上の Amazon EC2 インスタンスから別の VPC の EFS ファイルシステムにアクセスすることができます。VPC 同士が異なるアカウントに属していても可能です。

別の VPC で EFS マウントポイントに DNS 名前解決を使用することはできません。EFS ファイルシステムをマウントするには、対応する可用性ゾーンのマウントポイントの IP アドレスを使用します。

または、DNS サービスとして Amazon Route 53 を使用することもできます。Route 53 では、プライベートホストゾーンとリソースレコードセットを作成することで、別の VPC から EFS マウントターゲット IP アドレスを解決できます。これを行う方法の詳細については、「[Amazon Route 53 デベロッパーガイド](#)」の「[プライベートホストゾーンの使用](#)」を参照してください。

前提条件

ここに示す手順を使用する前に、以下を行う必要があります：

- EFS ファイルシステムをマウントするコンピューティングインスタンス上のユーティリティのセットである amazon-efs-utils の一部である、Amazon EFS クライアントをインストールします。EFS マウントヘルパーを使用します。これは amazon-efs-utils に含まれ、ファイルシステムをマウントします。amazon-efs-utils のインストール手順については、「[Amazon EFS クライアントの手動インストール](#)」を参照してください。
- ec2:DescribeAvailabilityZones に、インスタンスにアタッチした IAM ロールの IAM ポリシー内のアクションを許可します。AWS 管理ポリシーを IAM エンティティ AmazonElasticFileSystemsUtils にアタッチして、エンティティに必要なアクセス許可を付与することをお勧めします。
- 別の からマウントする場合は AWS アカウント、ファイルシステムリソースポリシーを更新して、他の のプリンシパル ARN の elasticfilesystem:DescribeMountTarget アクションを許可します AWS アカウント。例:

```
{
  "Id": "access-point-example03",
  "Statement": [
    {
      "Sid": "access-point-statement-example03",
      "Effect": "Allow",
      "Principal": {"AWS": "arn:aws:iam::555555555555:root"},

```



```
    "Action": "elasticfilesystem:DescribeMountTargets",
    "Resource": "arn:aws:elasticfilesystem:us-east-2:111122223333:file-
system/fs-12345678"
  }
]
```

EFS ファイルシステムリソースポリシーの詳細については、「[Amazon EFS 内のリソーススペースのポリシー](#)」を参照してください。

- botocone をインストールしてください。EFS クライアントは、ファイルシステムを別の VPC にマウントするときにファイルシステムの DNS 名を解決できない場合に、ボトコアを使用してマウントターゲット IP アドレスを取得します。詳細については、「amazon-efs-utils README」ファイルの「[botocone のインストール](#)」を参照してください。
- VPC ピアリング接続または VPC トランジットゲートウェイを設定します。

クライアントの VPC と EFS ファイルシステムの VPC を接続するには、VPC ピアリング接続または VPC トランジットゲートウェイを使用します。VPC ピアリング接続またはトランジットゲートウェイを使用して VPC に接続する場合、ある VPC 上の Amazon EC2 インスタンスから別の VPC の EFS ファイルシステムにアクセスすることができます。VPC 同士が異なるアカウントに属していても可能です。

トランジットゲートウェイは、VPC とオンプレミスネットワークを相互接続するために使用できるネットワークの中継ハブです。Transit Gateways の詳細については、Amazon VPC Transit Gateways Guide の [Transit Gateway で始める](#) を参照してください。

VPC ピアリング接続は、2 つの VPC 間のネットワーク接続です。このタイプの接続では、インターネットプロトコルバージョン 4 (IPv4) またはインターネットプロトコルバージョン 6 (IPv6) のプライベートアドレスを使用して、2 つの VPC 間でトラフィックをルーティングできます。VPC ピアリングを使用して、同じ AWS リージョン または 間で VPCs を接続できます AWS リージョン。VPC ピアリング詳細については、Amazon VPC Peering Guide の「[VPC ピアリングとは](#)」を参照してください。

ファイルシステムの高可用性を確保するために、NFS クライアントと同じアベイラビリティーゾーン (AZ) にある EFS マウントターゲットの IP アドレスを常に変更することをお勧めします。別のアカウントにある EFS ファイルシステムをマウントする場合は、NFS クライアントと EFS マウントターゲットが同じ アベイラビリティーゾーン ID にあることを確認します。この要件が適用されるのは、AZ 名がアカウントによって異なる可能性があるためです。

IAM またはアクセスポイントを使用して EFS ファイルシステムを別の VPC にマウントするには

1. EC2 インスタンスに接続します。詳細については、「Amazon EC2 ユーザーガイド」の「[EC2 インスタンスに接続する](#)」を参照してください。
2. 次のコマンドを使用して、ファイルシステムをマウントするためのディレクトリを作成します。

```
$ sudo mkdir /mnt/efs
```

3. IAM 認可を使用してファイルシステムをマウントするには、次のコマンドを使用します:

```
$ sudo mount -t efs -o tls,iam file-system-dns-name /mnt/efs/
```

EFS を使用した IAM 認可の詳細については、「[IAM を使用してファイルシステムへのアクセスを制御する](#)」を参照してください。

EFS アクセスポイントを使用してファイルシステムをマウントするには、次のコマンドを使用します:

```
$ sudo mount -t efs -o tls,accesspoint=access-point-id file-system-dns-name /mnt/efs/
```

EFS アクセスポイントの詳細については、「[アクセスポイントの使用](#)」を参照してください。

別の AWS リージョンからの EFS ファイルシステムのマウント

ファイルシステム AWS リージョン とは異なる にある別の VPC から EFS ファイルシステムをマウントする場合は、efs-utils.conf ファイルを編集する必要があります。/dist/efs-utils.conf で、次の行を見つけます。

```
#region = us-east-1
```

行のコメントを解除し、us-east-1 にない場合、ファイルシステムが配置されているリージョンの ID の値を置換します。

ネットワークファイルシステムを使用した EFS ファイルシステムのマウント

以下では、ネットワーク ファイルシステム (NFS) クライアントをインストールする方法と、Amazon EFS ファイルシステムを Amazon EC2 インスタンスにマウントする方法について説明します。また、mount コマンドの説明と、mount コマンドでファイルシステムの ドメインネームシステム (DNS) 名を指定するために使用可能なオプションについても説明します。加えて、システムの再起動後に fstab ファイルを使用してファイルシステムを自動的に再マウントする方法を説明します。

Note

このセクションでは、amazon-efs-utils パッケージを使用せずに、Amazon EFS ファイルシステムをマウントする方法について説明します。ファイルシステムを使用して、転送時のデータの暗号化を使用するには、Transport Layer Security (TLS) を使用して、ファイルシステムをマウントする必要があります。これを行うには、amazon-efs-utils パッケージを使用することをお勧めします。詳細については、「[Amazon EFS クライアントの手動インストール](#)」を参照してください。

トピック

- [前提条件](#)
- [NFS サポート](#)
- [NFS クライアントをインストールする](#)
- [推奨される NFS マウント設定](#)
- [DNS 名を使用して Amazon EC2 にマウントする](#)
- [IP アドレスを使用してマウントする](#)

前提条件

ファイルシステムをマウントする前に、次の要件が満たされていることを確認してください。

- 関連する AWS リソースを作成、設定、起動します。手順については、「[Amazon EFS の開始方法](#)」を参照してください。

- Amazon EC2 インスタンスとマウントターゲットに、必要なインバウンドおよびアウトバウンドアクセス権を持つ VPC セキュリティグループを作成します。詳細については、「[VPC セキュリティグループの使用](#)」を参照してください。

NFS サポート

Amazon EFS は、ファイルシステムを Amazon EC2 インスタンスにマウントするときに、[Network File System(ネットワークファイルシステム)] バージョン 4.0 と 4.1 (NFSv4) プロトコルをサポートします。NFSv4.0 はサポートされていますが、NFSv4.1 を使用することをお勧めします。Amazon EC2 インスタンスに Amazon EFS ファイルシステムをマウントするには、選択した NFSv4 プロトコルをサポートする NFS クライアントも必要になります。macOS Big Sur を実行している Amazon EC2 Mac インスタンスは、NFS v4.0 のみをサポートします。

Amazon EFS は、nconnect マウントオプションをサポートしていません。

Note

Linux カーネルバージョン5.4.*では、Linux NFS クライアントは、デフォルトの `readahead_kb` の値である 128KB を使用します。この値を 15 MB に増やすことをお勧めします。詳細については、「[NFS を最適化する readahead_kb サイズ](#)」を参照してください。

最適なパフォーマンスのため、また、既知のさまざまな NFS クライアントのバグを避けるため、最新の Linux カーネルの使用をお勧めします。エンタープライズ Linux ディストリビューションを使用している場合は、以下をお勧めします。

- Amazon Linux 2
- Amazon Linux 2017.09 以降
- Red Hat Enterprise Linux (および、その派生物 CentOS など) バージョン 7 以降
- Ubuntu 16.04 LTS 以降
- SLES 12 Sp2 以降

別のディストリビューションまたはカスタムカーネルを使用している場合は、カーネルバージョン 4.3 以降をお勧めします。EC2 インスタンスから Amazon EFS を使用する場合は、特定の AMI またはカーネルバージョンに関連する問題のトラブルシューティングを行うには [AMI とカーネルの問題のトラブルシューティング](#) を参照してください。

 Note

Microsoft Windows を実行している Amazon EC2 インスタンスで EFS ファイルシステムをマウントすることはサポートされていません。

NFS クライアントをインストールする

Amazon EC2 インスタンスに EFS ファイルシステムをマウントするには、まず NFS クライアントをインストールする必要があります。EC2 インスタンスに接続して NFS クライアントをインストールするには、EC2 インスタンスの公開 DNS 名とユーザー名でログインする必要があります。インスタンスのユーザー名は通常 `ec2-user` です。

EC2 インスタンスに接続し、NFS クライアントをインストールするには

1. EC2 インスタンスに接続します。詳細については、「Amazon EC2 ユーザーガイド」の「[EC2 インスタンスに接続する](#)」を参照してください。

キーファイルは SSH 用に公開できません。これらの許可を設定するには、`chmod 400 filename.pem` コマンドを使用できます。詳細については、「[Amazon EC2 インスタンスのキーペアを作成する](#)」を参照してください。

2. (オプション) アップデートを入手して再起動します。

```
$ sudo yum -y update
$ sudo reboot
```

3. 再起動後、EC2 インスタンスに再接続します。
4. NFS クライアントをインストールします。

Amazon Linux AMI または Red Hat Linux AMI を使用している場合は、次のコマンドを使用して NFS クライアントをインストールします。

```
$ sudo yum -y install nfs-utils
```

Ubuntu Amazon EC2 AMI を使用している場合は、次のコマンドを使用して NFS クライアントをインストールします。

```
$ sudo apt-get -y install nfs-common
```

5. 次のコマンドを使用して NFS サービスを起動します。RHEL 7 の場合:

```
$ sudo service nfs start
```

RHEL 8 の場合:

```
$ sudo service nfs-server start
```

6. 次のように、NFS サービスが起動したことを確認します。

```
$ sudo service nfs status
Redirecting to /bin/systemctl status nfs.service
# nfs-server.service - NFS server and services
   Loaded: loaded (/usr/lib/systemd/system/nfs-server.service; disabled; vendor
   preset: disabled)
   Active: active (exited) since Wed 2019-10-30 16:13:44 UTC; 5s ago
   Process: 29446 ExecStart=/usr/sbin/rpc.nfsd $RPCNFSDARGS (code=exited, status=0/SUCCESS)
   Process: 29441 ExecStartPre=/bin/sh -c /bin/kill -HUP `cat /run/gssproxy.pid` (code=exited, status=0/SUCCESS)
   Process: 29439 ExecStartPre=/usr/sbin/exportfs -r (code=exited, status=0/SUCCESS)
  Main PID: 29446 (code=exited, status=0/SUCCESS)
   CGroup: /system.slice/nfs-server.service
```

カスタムカーネル (カスタム AMI を構築する場合) を使用する場合は、少なくとも NFSv4.1 クライアントカーネルモジュールと適切な NFS4 ユーザースペースマウントヘルパーを含める必要があります。

Note

Amazon EC2 インスタンスの起動時に [Amazon Linux AMI 2016.03.0] または [Amazon Linux AMI 2016.09.0] を選択する場合、nfs-utils はデフォルトですでに AMI に含まれているため、インストールする必要はありません。

次の手順: ファイルシステムをマウントする

以下のいずれかの手順を実行し、ファイルシステムをマウントします。

- [DNS 名を使用して Amazon EC2 にマウントする](#)

- [IP アドレスを使用してマウントする](#)
- [EFS ファイルシステムの自動マウント](#)

推奨される NFS マウント設定

Linux の マウントオプションには次の値をお勧めします。

- `noresvport` - ネットワーク接続が再確立された時に、新しい非特権の Transmission Control Protocol (TCP) 送信元ポートを使用するように、NFS クライアントに指示します。古いバージョンの Linux カーネル (バージョン v5.4 以下) に含まれている NFS クライアントソフトウェアには、切断時に NFS クライアントが同じ TCP ソースポートで再接続を試みるという動作が含まれています。この動作は TCP RFC に準拠していないため、これらのクライアントが EFS ファイルシステムへの接続をすぐに再確立できなくなる可能性があります。

`noresvport` オプションを使用すると、NFS クライアントが EFS ファイルシステムに透過的に再接続できるようになり、ネットワーク回復イベント後に再接続しても中断されない可用性が維持されます。

Important

再接続やネットワーク復旧イベントの後も EFS ファイルシステムの可用性が中断されないように、`noresvport` マウントオプションを使用することを強くお勧めします。

[EFS マウントヘルパー](#)を使用して、ファイルシステムをマウントすることを検討します。EFS マウントヘルパーは、Amazon EFS ファイルシステム用に最適化された NFS マウントオプションを使用します。

- `rsz=1048576` - 各ネットワーク読み取りリクエストに対して NFS クライアントが受信できるデータの最大バイト数を設定します。この値は、EFS ファイルシステム上のファイルからデータを読み取る際に適用されます。パフォーマンスが低下しないように、可能な限り大きいサイズ (最大 1048576) を使用することをお勧めします。
- `wsz=1048576` - 各ネットワーク書き込みリクエストに対して NFS クライアントが送信できるデータの最大バイト数を設定します。この値は、EFS ファイルシステム上のファイルにデータを書き込む際に適用されます。パフォーマンスが低下しないように、可能な限り大きいサイズ (最大 1048576) を使用することをお勧めします。
- `hard` - NFS リクエストがタイムアウトした後の NFS クライアントのリカバリ動作を設定します。これにより、NFS リクエストは、サーバーが応答するまで無期限に再試行されます。データの整合性を確保できるように、ハードマウントオプション (`hard`) を使用することをお勧めしま

す。soft マウントを使用する場合は、timeo パラメータを 150 デシ秒 (15 秒) 以上に設定してください。これにより、ソフトマウントに固有のデータ破損が生じるリスクを最小限に抑えることができます。

- timeo=600 - NFS クライアントがレスポンスを待機するのに要するタイムアウト値を設定してから、NFS リクエストを 600 デシ秒 (60 秒) に設定します。タイムアウトパラメータ (timeo) を変更する必要がある場合は、少なくとも 150 の値を使用することをお勧めします。これは 15 秒に相当します。これにより、パフォーマンスの低下を避けることができます。
- retrans=2 - NFS クライアントでリカバリアクションを試行する前に、そのアクションのリクエスト試行回数を 2 回に設定します。
- _netdev - /etc/fstab に存在する場合、クライアントは、ネットワークが有効になるまで、EFS ファイルシステムをマウントすることはできません。
- nofail - マウントされた EFS ファイルシステムのステータスに関係なく EC2 インスタンスを起動する必要がある場合は、/etc/fstab ファイルのファイルシステムのエントリに nofail オプションを追加します。

前述のデフォルト設定を使用しない場合は、以下の点に注意してください。

- 一般に、デフォルトとは異なる他のマウントオプションを設定しないでください。パフォーマンスが低下し、別の問題が生じる可能性があります。たとえば、読み取りまたは書き込みバッファサイズを変更したり、属性のキャッシュを無効にしたりするとパフォーマンスが損なわれる可能性があります。
- Amazon EFS は送信元ポートを無視します。Amazon EFS 送信元ポートを変更した場合、変更は無効になります。
- Amazon EFS では、nconnect マウントオプションをサポートしていません。
- Amazon EFS では Kerberos セキュリティバリエーションをサポートしていません。たとえば、以下のマウントコマンドは失敗します。

```
$ mount -t nfs4 -o krb5p <DNS_NAME>:/ /efs/
```

- ファイルシステムは、DNS 名を使用してマウントすることをお勧めします。この名前は、Amazon EC2 インスタンスと同じアベイラビリティーゾーンの Amazon EFS マウントターゲットの IP アドレスに解決されます。ご使用の Amazon EC2 インスタンスとは異なるアベイラビリティーゾーンでマウントターゲットを使用すると、アベイラビリティーゾーンから送信されるデータに対して EC2 の標準料金がかかります。また、ファイルシステムオペレーションのレイテンシーが増加することがあります。

- 他のマウントオプションや、デフォルト設定の詳細な説明については、Linux のドキュメントを参照してください。

DNS 名を使用して Amazon EC2 にマウントする

Note

ファイルシステムをマウントする前に、EC2 セキュリティグループからのインバウンドの NFS アクセスを許可するルールをマウントターゲットのセキュリティグループに追加する必要があります。詳細については、「[VPC セキュリティグループの使用](#)」を参照してください。

- ファイルシステムの DNS 名 – ファイルシステムの DNS 名を使用するのがもっとも簡単なマウントオプションです。ファイルシステムの DNS 名は、接続する Amazon EC2 インスタンスのアベイラビリティゾーン内のマウントターゲットの IP アドレスを自動的に解決します。この DNS 名はコンソールから取得できます。ファイルシステム ID がある場合は、次の規則を使用してコンストラクトすることができます。

```
file-system-id.efs.aws-region.amazonaws.com
```

Note

ファイルシステムの DNS 名の DNS 解決には、Amazon EFS ファイルシステムが、クライアント インスタンスと同じアベイラビリティゾーンにマウントターゲットがある必要があります。

- ファイルシステムの DNS 名を使用して、Amazon EC2 Linux インスタンスにファイルシステムをマウントするには次のコマンドを使用します。

```
sudo mount -t nfs -o  
nfsvers=4.1,rsiz=1048576,wsiz=1048576,hard,timeo=600,retrans=2,noresvport file-  
system-id.efs.aws-region.amazonaws.com:/ /efs-mount-point
```


- ファイルシステムの DNS 名を使用して、macOS バージョン (Big Sur、Monterey、Ventura) をサポートしている Amazon EC2 Mac インスタンスにファイルシステムをマウントするには、次のコマンドを使用します。

```
sudo mount -t nfs -o  
nfsvers=4.0,rsiz=65536,wsiz=65536,hard,timeo=600,retrans=2,noresvport,mountport=2049 file-system-id.efs.aws-region.amazonaws.com:/ /efs
```

Important

macOS バージョンをサポートしている EC2 Mac インスタンスにマウントするときに EFS ファイルシステムに正常に接続するには、`mountport=2049` を使用する必要があります。

- マウントターゲットの DNS 名 – 2016 年 12 月、ファイルシステムの DNS 名を導入しました。各アベイラビリティゾーンマウントターゲットに DNS 名を引き続き提供し、下位互換性を確保します。一般的な形式のマウントターゲット DNS 名は次のようになります。

```
availability-zone.file-system-id.efs.aws-region.amazonaws.com
```

Note

アベイラビリティゾーン間でのマウントターゲット DNS 名解決がサポートされています。

場合によっては、マウントターゲットを削除して同じアベイラビリティゾーンに新たに作成します。このような場合は、そのアベイラビリティゾーンの新しいマウントターゲットの DNS 名は、古いマウントターゲットの DNS 名と同じになります。

ファイルシステムをマウントするための正確なコマンドを「Attach(添付)」ダイアログボックスで表示し、コピーすることができます。

ファイルシステムのマウントコマンドを表示するには

1. Amazon EFS コンソールで、マウントするファイルシステムを選択して詳細ページを表示します。

2. このファイルシステムで使用するマウントコマンドを表示するには、右上の [アタッチ] を選択します。

「Attach(添付)」画面には、ファイルシステムのマウントに使用する正確なコマンドが表示されます。

3. デフォルトの [Mount via DNS (DNS 経由でマウントする)] ビューでは、EFS マウントヘルパーまたは NFS クライアントでマウントする際に、ファイルシステムの DNS 名を使用してファイルシステムをマウントするコマンドが表示されます。

Amazon EFS をサポートする AWS リージョンのリストについては、の [Amazon Elastic File System](#) を参照してください AWS 全般のリファレンス。

mount コマンドで DNS 名を使用するには、以下の条件が満たされている必要があります。

- 接続する EC2 インスタンスは VPC 内にあり、Amazon が提供する DNS サーバーを使用するように設定されている必要があります。Amazon DNS サーバーの詳細については、「Amazon VPC ユーザーガイド」の「[Amazon VPC の DHCP オプションセット](#)」を参照してください。
- 接続する EC2 インスタンスの VPC で DNS 解決と DNS ホスト名の利用が有効になっている必要があります。詳細については、「Amazon [VPC ユーザーガイド](#)」の「[VPC の DNS 属性](#)」を参照してください。
- 接続する EC2 インスタンスは、EFS ファイルシステムと同じ VPC 内にいる必要があります。別の場所または別の VPC からの、ファイルシステムへのアクセスおよびマウントの詳細については、[前提条件](#) および [チュートリアル: 別の VPC からファイルシステムをマウントする](#) を参照してください。

Note

ファイルシステムをマウントする前に、マウントターゲットを作成してから 90 秒待機することをお勧めします。この待機により、DNS レコードはファイルシステム AWS リージョンがある で完全に伝播されます。

IP アドレスを使用してマウントする

Amazon EFS ファイルシステムを DNS 名でマウントする代わりに、Amazon EC2 インスタンスはマウントターゲットの IP アドレスを使ってファイルシステムをマウントすることができます。IP アド

レスによるマウントは、DNS ホスト 名が無効化されている VPC などの、DNS が無効な環境で機能します。

デフォルトで DNS 名を使用してファイルシステムをマウントするように設定されたアプリケーションのフォールバックオプションとして、マウントターゲット IP アドレスを使用してファイルシステムのマウントを設定することもできます。マウントターゲット IP アドレスに接続する場合、EC2 インスタンスは接続先インスタンスと同じアベイラビリティーゾーンのマウントターゲット IP アドレスを使用してマウントする必要があります。

ファイルシステムをマウントするための正確なコマンドを「Attach(添付)」ダイアログボックスで表示し、コピーすることができます。

 Note

ファイルシステムをマウントする前に、EC2 セキュリティグループからのインバウンドの NFS アクセスを許可するルールをマウントターゲットのセキュリティグループに追加する必要があります。詳細については、「[VPC セキュリティグループの使用](#)」を参照してください。

マウントターゲット IP アドレスを使用して EFS ファイルシステムをマウントするための正確なコマンドを表示およびコピーするには

1. Amazon Elastic File System コンソールの <https://console.aws.amazon.com/efs/> を開いてください。
2. Amazon EFS コンソールで、マウントするファイルシステムを選択して詳細ページを表示します。
3. このファイルシステムで使用するマウントコマンドを表示するには、右上の [アタッチ] を選択します。
4. 「Attach(添付)」画面には、ファイルシステムのマウントに使用する正確なコマンドが表示されます。

[Mount via IP(IP経由でマウント)]を選択して、NFSクライアントで選択したアベイラビリティーゾーンのマウントターゲットIPアドレスを使用してファイルシステムをマウントするコマンドを表示します。

- mount コマンドでマウントターゲットの IP アドレスを使用して、次のコマンドで Amazon EC2 Linux インスタンスにファイルシステムをマウントすることができます。

```
sudo mount -t nfs -o  
nfsvers=4.1,rsize=1048576,wsiz=1048576,hard,timeo=600,retrans=2,noresvport mount-  
target-IP:/ /efs
```

- mount コマンドでマウントターゲットの IP アドレスを使用して、次のコマンドで macOS Big Sur を実行している Amazon EC2 Mac インスタンスにファイルシステムをマウントすることができます。

```
sudo mount -t nfs -o  
nfsvers=4.0,rsize=65536,wsiz=65536,hard,timeo=600,retrans=2,noresvport,mountport=2049 mount-  
target-IP:/ /efs
```

Important

macOS Big Sur を実行している EC2 Mac インスタンスにマウントするときに EFS ファイルシステムに正常に接続するには、mountport=2049 を使用する必要があります。

での IP アドレスを使用したマウント AWS CloudFormation

AWS CloudFormation テンプレートの IP アドレスを使用してファイルシステムをマウントすることもできます。詳細については、GitHub のコミュニティが提供している設定ファイルの [awsdocs/elastic-beanstalk-samples](#) リポジトリの [storage-efs-mountfilesystem-ip-addr.config](#) を参照してください。

EFS ファイルシステムの自動マウント

EFS マウントヘルパーまたは NFS を使用して、インスタンスの起動時に EFS ファイルシステムを自動的にマウントするように Amazon EC2 インスタンスを設定できます。

- EFS マウントヘルパーの使用
 - EC2 インスタンス起動ウィザードを使用して新しい EC2 Linux インスタンスを作成するときに EFS ファイルシステムをアタッチします。EC2
 - EFS ファイルシステムのエントリで EC2 の /etc/fstab ファイルを更新する。
- EC2 Linux および Mac インスタンスを対象に、[NFS を使用して EFS マウントヘルパーなしで](#) EC2 の /etc/fstab ファイルを更新します。

Note

EFS マウントヘルパーは、macOS Big Sur または Monterey を実行している EC2 Mac インスタンスへの自動マウントをサポートしていません。代わりに、[NFS を使用して EC2 Mac インスタンスの /etc/fstab ファイルを設定](#)し、EFS ファイルシステムを自動的にマウントすることができます。

トピック

- [新しい EC2 Linux インスタンスでの自動マウントの有効化](#)
- [既存の EC2 Linux インスタンスでの自動マウントの有効化](#)
- [NFS を使用した EC2 Linux または Mac インスタンスでの自動マウントの有効化](#)

新しい EC2 Linux インスタンスでの自動マウントの有効化

Amazon EC2 インスタンス起動ウィザードを使用して新しい EC2 Linux インスタンスを作成する場合、Amazon EFS ファイルシステムを自動的にマウントするように設定できます。Amazon EC2 EC2 インスタンスでは、インスタンスが最初に起動されたときと再起動ごとに、ファイルシステムが自動的にマウントされます。

この方法では、EFS マウントヘルパーを使用してファイルシステムをマウントし、EC2 インスタンス上で /etc/fstab ファイルを更新します。マウントヘルパーは、[amazon-efs-utils](#) というツールセットの一部です。

Note

EFS ファイルシステムは、インスタンスの起動時に macOS Big Sur または Monterey を実行している EC2 Mac インスタンスへのマウントをサポートしていません。

Note

Microsoft Windows ベースの EC2 インスタンスで Amazon EFS を使用することはできません。

EC2 インスタンスを起動して接続する前に、キーペアを作成する必要があります。詳細については、「Amazon EC2 ユーザーガイド」の「[Amazon EC2 のキーペアと Amazon EC2 インスタンス](#)」を参照してください。

起動時に自動的に EFS ファイルシステムをマウントするよう EC2 インスタンスを設定するには

1. Amazon EC2 コンソール (<https://console.aws.amazon.com/ec2/>) を開きます。
2. [インスタンスの作成] を選択してください。
3. [ステップ 1: Amazon マシンイメージ (AMI) の選択] で、リストの一番上にある Amazon Linux AMI を見つけて [選択] を選択します。
4. [ステップ 2: インスタンスタイプの選択] で、[次へ: インスタンスの詳細の設定] を選択します。
5. [ステップ 3: インスタンスの詳細の設定] で、以下の情報を入力します。
 - [ネットワーク] で、マウントする EFS ファイルシステムと同じ VPC のエントリを選択します。
 - [サブネット] で、任意のアベイラビリティーゾーンのデフォルトのサブネットを選択します。
 - [ファイルシステム] で、マウントする EFS ファイルシステムを選択します。ファイルシステム ID の横に表示されるパスは、EC2 インスタンスが使用するマウントポイントです。このマウントポイントは変更できます。
 - [Advanced Details (高度な詳細)] の [User data (ユーザーデータ)] で、ユーザーデータが自動的に生成されます。このデータには、[File systems (ファイルシステム)] で指定した EFS ファイルシステムをマウントするために必要なコマンドが含まれます。
6. [Next: Add Storage] (次の手順: ストレージの追加) を選択します。
7. [次の手順: タグの追加] を選択します。
8. インスタンスに名前を付け、[次へ: セキュリティグループの設定] を選択します。
9. [ステップ 6: セキュリティグループの設定] で、[セキュリティグループの割り当て] を [既存のセキュリティグループを選択する] に設定します。デフォルトのセキュリティグループを選択して、EFS ファイルシステムにアクセスできることを確認します。

このセキュリティグループを使用して、Secure Shell (SSH) で EC2 インスタンスにアクセスすることはできません。SSH によるアクセスについては、後でデフォルトのセキュリティを編集し、SSH を許可するルールまたは SSH を許可する新しいセキュリティグループを追加できます。以下の設定を使用できます。

- タイプ: SSH
- [Protocol]: TCP

- ポート範囲: 22
- 出典: Anywhere 0.0.0.0/0

10. [Review and Launch (確認と作成)] を選択します。

11. [起動] を選択します。

12. 作成したキーペアのチェックボックスを選択した後、[インスタンスの起動] を選択します。

これで、EC2 インスタンスは、起動時および再起動されるたびに EFS ファイルシステムをマウントするよう設定されました。

既存の EC2 Linux インスタンスでの自動マウントの有効化

/etc/fstab ファイルには、ファイルシステムに関する情報が含まれています。インスタンスの起動中に実行される `mount -a` コマンドは、/etc/fstab に示されているすべてのファイルシステムをマウントします。この手順では、インスタンスの再起動時にインスタンスが EFS マウントヘルパーを使用して EFS ファイルシステムを自動的に再マウントするように、Amazon EC2 Linux EFS インスタンス/etc/fstabの を手動で更新します。

Note

EFS ファイルシステムは、macOS Big Sur または Monterey を実行している EC2 Mac インスタンスでEFS マウントヘルパー/etc/fstabと を使用した自動マウントをサポートしていません。代わりに、[NFS と /etc/fstab](#)を使用して、macOS Big Sur と Monterey を実行している EC2 Mac インスタンスにファイルシステムを自動的にマウントできます。

この方法では、EFS マウントヘルパーを使用してファイルシステムをマウントします。マウントヘルパーは、amazon-efs-utils というツールセットの一部です。

これらの amazon-efs-utils ツールは、Amazon Linux および Amazon Linux 2 の Amazon マシンイメージ (AMI) にインストールできます。amazon-efs-utils の詳細については、「[Amazon EFS クライアントの手動インストール](#)」を参照してください。Red Hat Enterprise Linux (RHEL) など、他の Linux ディストリビューションを使用している場合は、amazon-efs-utils を手動でビルドおよびインストールします。詳細については、「[他の Linux ディストリビューションで amazon-efs-utils パッケージをインストールする](#)」を参照してください。

前提条件

この手順を正常に実装するには、以下の要件を満たす必要があります。

- 自動的に再マウントする EFS ファイルシステムが既に作成されています。詳細については、「[コンソールを使用したクイック作成](#)」を参照してください。
- EFS ファイルシステムを自動的に再マウントするように設定する EC2 Linux インスタンスは既に作成されています。
- EFS マウントヘルパーは EC2 Linux インスタンスにインストールされます。詳細については、「[Amazon EFS クライアントの手動インストール](#)」を参照してください。

/etc/fstab ファイルの更新

EC2 Linux インスタンスの /etc/fstab を更新すると、インスタンスの再起動時に、インスタンスで EFS マウントヘルパーを使用して自動的に EFS ファイルシステムを再マウントできます。この設定を行うには、次の手順を実行します。

EC2 インスタンスで /etc/fstab ファイルを更新するには

1. EC2 インスタンスに接続します。詳細については、「Amazon EC2 ユーザーガイド」の「[EC2 インスタンスに接続する](#)」を参照してください。
2. エディタで /etc/fstab ファイルを開きます。
3. IAM 認可または EFS アクセスポイントを使用した自動マウントの場合:
 - インスタンスプロファイルを持つ EC2 インスタンスに IAM 認可を使用して自動的にマウントするには、次の行を /etc/fstab ファイルに追加します。

```
file-system-id:/ efs-mount-point efs _netdev,noresvport,tls,iam 0 0
```

- 認証情報ファイルを使用して Linux インスタンスに対して IAM 認可によるマウントを自動的に行うには、/etc/fstab ファイルに次の行を追加します。

```
file-system-id:/ efs-mount-point efs  
_netdev,noresvport,tls,iam,awsprofile=namedprofile 0 0
```

- EFS アクセスポイントを使用してファイルシステムを自動的にマウントするには、/etc/fstab ファイルに次の行を追加します。


```
file-system-id:/ efs-mount-point efs _netdev,noresvport,tls,accesspoint=access-point-id 0 0
```

Warning

ファイルシステムを自動的にマウントする場合、ネットワークファイルシステムを識別するために使用された `_netdev` オプションを使用します。`_netdev` が見つからない場合、EC2 インスタンスはレスポンスを停止する可能性があります。この結果は、コンピューティングインスタンスがネットワークを開始後、ネットワークファイルシステムを初期化する必要があるためです。詳細については、「[自動マウントが失敗してインスタンスがレスポンスしない](#)」を参照してください。

詳細については、「[IAM 認可を使用してマウントする](#)」および「[Amazon EFS アクセスポイントを使用してマウントする](#)」を参照してください。

4. 変更をファイルに保存します。

Note

マウントされた EFS ファイルシステムのステータスに関係なく、EC2 インスタンスを起動する必要がある場合があります。そのような場合は、`/etc/fstab` ファイルに記載されているファイルシステムのエントリに `nofail` オプションを追加します。

`/etc/fstab` ファイルに追加したコードの行は以下のようになります。

フィールド	説明
<i>file-system-id</i> :/	EFS ファイルシステムの ID。この ID は、コンソールから取得することも、CLI または AWS SDK からプログラムで取得することもできます。
<i>efs-mount-point</i>	EC2 インスタンス上の EFS ファイルシステムのマウントポイントです。

フィールド	説明
efs	ファイルシステムのタイプ。マウントヘルパーを使用している場合、このタイプは常に efs です。
mount options	ファイルシステムのマウントオプション。以下はオプションのカンマ区切りのリストです。 • <code>_netdev</code> – このオプションは、ネットワークアクセスを必要とするデバイスにファイルシステムが存在することをオペレーティングシステムに通知します。このオプションは、クライアント上でネットワークが有効になるまで、インスタンスがファイルシステムをマウントするのを防ぎます。 • <code>noresvport</code> – ネットワーク接続が再確立された時に、新しい Transmission Control Protocol (TCP) 送信元ポートを使用するように、NFS クライアントに指示します。これにより、ネットワーク復旧イベント後、EFS ファイルシステムでの中断のない可用性が保証されます。 • <code>tls</code> – 転送時のデータの暗号化を可能にします。 • <code>iam</code> – このオプションを使用して、インスタンスプロファイルを持つ EC2 インスタンスに IAM 認可でマウントします。iam マウントオプションを使用するには、tls オプションも使用する必要があります。詳細については、「IAM を使用してファイルシステムへのアクセスを制御する」を参照してください。 • <code>awsprofile= <i>namedprofile</i></code> – 認証情報ファイルを使用して Linux インスタンスに対して IAM 認可によるマウントを行うには、このオプションを iam および tls オプションと共に使用します。EFS アクセスポイントの詳細については、「IAM を使用してファイルシステムへのアクセスを制御する」を参照してください。 • <code>accesspoint= <i>access-point-id</i></code> – このオプションを tls オプションと共に指定して、EFS アクセスポイントを使用したマウントを行います。EFS アクセスポイントの詳細については、「アクセスポイントの使用」を参照してください。

フィールド	説明
0	ゼロ以外の値の場合、ファイルシステムを dump でバックアップする必要があることを示します。EFS の場合、この値は 0 になっている必要があります。
0	起動時に fsck がファイルシステムをチェックする順序。EFS ファイルシステムの場合、起動時に fsck を実行すべきでないことを示すにはこの値を 0 にします

NFS を使用した EC2 Linux または Mac インスタンスでの自動マウントの有効化

EC2 Amazon EC2 /etc/fstab ファイルを更新します。

EC2 インスタンスで **/etc/fstab** ファイルを更新するには

1. EC2 インスタンスに接続します。詳細については、「Amazon EC2 ユーザーガイド」の「[EC2 インスタンスに接続する](#)」を参照してください。
2. エディタで /etc/fstab ファイルを開きます。
3. EFS マウントヘルパーの代わりに NFS を使って使用してファイルシステムを自動的にマウントするには、/etc/fstab ファイルに次の行を追加します。
 - *file_system_id* を、マウントするファイルシステムの ID に置き換えます。
 - *aws-region* を、などのファイルシステム AWS リージョン が存在する に置き換えます us-east-1。
 - *mount_point* をファイルシステムのマウントポイントに置き換えます。

```
file_system_id.efs.aws-region.amazonaws.com:/ mount_point nfs4
nfsvers=4.1,rsiz=1048576,wsiz=1048576,hard,timeo=600,retrans=2,noresvport,_netdev
0 0
```

/etc/fstab ファイルに追加したコードの行は以下のようになります。

フィールド	説明
<i>file-system-id</i> :/	EFS ファイルシステムの ID。この ID は、コンソールから取得することも、CLI または AWS SDK からプログラムで取得することもできます。
<i>efs-mount-point</i>	EC2 インスタンス上の EFS ファイルシステムのマウントポイントです。
nfs4	ファイルシステムのタイプを指定します。
mount options	ファイルシステムのマウント・オプションのカンマ区切りリスト: • nfsvers=4.1 — NFS v4.1 の使用を指定します。 • rsize=1048576 - パフォーマンスを向上させるために、EFS ファイルシステム上のファイルからデータを読み取る際に、各ネットワーク読み取りリクエストに対して NFS クライアントが受信できるデータの最大バイト数を設定します。1048576 は可能な最大サイズです。 • wsize=1048576 - パフォーマンスを向上させるために、EFS ファイルシステム上のファイルにデータを書き出します。各ネットワーク書き込みリクエストに対して NFS クライアントが送信できるデータの最大バイト数を設定します。1048576 は可能な最大サイズです。 • hard - NFS リクエストがタイムアウトした後の NFS クライアントのリカバリ動作を設定します。これにより、NFS リクエストは、サーバーが応答するまで無期限に再試行されます。データの整合性を確保できるように、ハードマウントオプション (hard) を使用することをお勧めします。soft マウントを使用する場合は、timeo パラメータを 150 デシ秒 (15 秒) 以上に設定してください。これにより、ソフトマウントに固有のデータ破損が生じるリスクを最小限に抑えることができます。 • timeo=600 - NFS クライアントがレスポンスを待機するのに要するタイムアウト値を設定してから、リクエストを 600 デシ秒 (60 秒) に設定します。タイムアウトパラメータ (timeo) を変更する必要がある場合は、少なくとも 150 の値を使用することをお勧めしま

フィールド	説明
	す。これは 15 秒に相当します。これにより、パフォーマンスの低下を避けることができます。 • <code>retrans=2</code> - NFS クライアントでリカバリアクションを試行する前に、そのアクションのリクエスト試行回数を 2 回に設定します。 • <code>noresvport</code> - ネットワーク接続が再確立された時に、新しい Transmission Control Protocol (TCP) 送信元ポートを使用するように、NFS クライアントに指示します。これにより、ネットワーク復旧イベント後、EFS ファイルシステムでの中断のない可用性が保証されます。 • <code>_netdev</code> - クライアントは、ネットワークが有効になるまで、EFS ファイルシステムをマウントしようとするのを防ぎます。
0	dump 値を指定し、0 がファイルシステムをバックアップしないように dump ユーティリティに指示します。
0	起動時に実行しないように fsck ユーティリティに指示します。

ファイルシステムをアンマウントする

ファイルシステムを削除する前に、接続しているすべての Amazon EC2 インスタンスからアンマウントすることをお勧めします。インスタンス自体で `umount` コマンドを実行することで、Amazon EC2 インスタンスのファイルシステムをアンマウントできます。EFS ファイルシステムは、AWS CLI、AWS Management Console または AWS SDKs を使用してアンマウントすることはできません。Linux を実行している EC2 インスタンスに接続された EFS ファイルシステムをアンマウントするには、次のように `umount` コマンドを使用します。

```
umount /mnt/efs
```

他の `umount` オプションを指定しないことをお勧めします。デフォルトと異なる `umount` オプションを設定しないでください。

`df` コマンドを実行して EFS ファイルシステムがアンマウントされたことを確認できます。このコマンドを実行すると、Linux ベースの Amazon EC2 インスタンスに現在マウントされているファイルシステムのディスク使用状況の統計情報が表示されます。アンマウントする EFS ファイルシステム

がdfコマンド出力に表示されていない場合は、ファイルシステムがアンマウントされていることを意味します。

Example – EFS ファイルシステムのマウントステータスを特定し、アンマウントします。

```
$ df -T
Filesystem Type 1K-blocks Used Available Use% Mounted on
/dev/sda1 ext4 8123812 1138920 6884644 15% /
availability-zone.file-system-id.efs.aws-region.amazonaws.com :/ nfs4 9007199254740992
0 9007199254740992 0% /mnt/efs
```

```
$ umount /mnt/efs
```

```
$ df -T
```

```
Filesystem Type 1K-blocks Used Available Use% Mounted on
/dev/sda1 ext4 8123812 1138920 6884644 15% /
```

チュートリアル: EFS ファイルシステムを作成し、 を使用して EC2 インスタンスにマウントする AWS CLI

暗号化された EFS ファイルシステムを作成し、VPC 内の EC2 インスタンスにマウントし、AWS CLIを使用してセットアップをテストします。

Note

「[入門](#)」チュートリアルでは、コンソールを使用して Amazon EC2 および EFS リソースを作成します。このチュートリアルでは、 を使用して同じ AWS CLI ことを行います。主に Amazon EFS API に慣れます。

このチュートリアルでは、アカウントに次の AWS リソースを作成します。

- Amazon EC2 リソース。
 - 2つのセキュリティグループ (EC2 インスタンスおよび EFS ファイルシステム用)。

セキュリティグループにルールを追加して、適切なインバウンド/アウトバウンドアクセスを許可します。これにより、EC2 インスタンスは、標準の NFSv4.1 TCP ポートを使用して、マウントターゲットを通じてファイルシステムに接続できます。

- VPC 内の EC2 インスタンス。
- Amazon EFS のリソース
 - ファイルシステム。
 - ファイルシステムのマウントターゲット。

ファイルシステムを EC2 インスタンスにマウントするには、VPC にマウントターゲットを作成する必要があります。VPC 内の各アベイラビリティゾーンに 1 つのマウントターゲットを作成できます。詳細については、「[Amazon EFS の仕組み](#)」を参照してください。

次に、EC2 インスタンス上のファイルシステムをテストします。チュートリアル最後のクリーンアップの手順には、これらのリソースを削除するための情報が記載されています。

このチュートリアルでは、これらのリソースをすべて米国西部 (オレゴン) リージョン (us-west-2) に作成します。どちら AWS リージョン を使用する場合でも、常に使用してください。VPC、EC2 リソース、EFS リソースなどのすべてのリソースは、同じ AWS リージョン内になければなりません。

トピック

- [前提条件](#)
- [のセットアップ AWS CLI](#)
- [ステップ 1: EC2 リソースを作成する](#)
- [ステップ 2: EFS リソースを作成する](#)
- [ステップ 3: ファイルシステムを EC2 インスタンスにマウントしてテストする](#)
- [ステップ 4: クリーンアップする](#)

前提条件

- のルート認証情報を使用して コンソールに AWS アカウント サインインし、入門演習を試すことができます。ただし、AWS Identity and Access Management (IAM) では、 のルート認証情報を使用しないことをお勧めします AWS アカウント。代わりに、アカウントに管理者ユーザーを作成し、それらの認証情報を使用してアカウントのリソースを管理します。代わりに、アカウントに

管理者ユーザーを作成し、それらの認証情報を使用してアカウントのリソースを管理します。詳細については、「[AWS IAM Identity Center ユーザーガイド](#)」の「[IAM Identity Center ユーザーの AWS アカウント アクセスの割り当て](#)」を参照してください。

- アカウントで作成したデフォルトの VPC またはカスタム VPC を使用できます。このウォークスルーでは、デフォルトの VPC 設定が機能します。ただし、カスタム VPC を使用する場合は、次の点を確認してください。
- DNS ホスト名は有効です。詳細については、「Amazon VPC ユーザーガイド」の「[DNS attributes in your VPC](#)」(VPC の DNS 属性) を参照してください。
- インターネットゲートウェイが VPC にアタッチされています。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。
- VPC サブネットは、VPC サブネットで起動されたインスタンスのパブリック IP アドレスを要求するように設定されています。詳細については、「Amazon VPC ユーザーガイド」の「[VPC とサブネットの IP アドレス指定](#)」を参照してください。
- VPC ルートテーブルには、インターネット経由のすべてのトラフィックをインターネットゲートウェイに送信するルールが含まれています。
- をセットアップ AWS CLI し、管理者ユーザープロファイルを追加する必要があります。

のセットアップ AWS CLI

以下の手順を使用して、AWS CLI および ユーザープロファイルを設定します。

をセットアップするには AWS CLI

1. AWS CLIをダウンロードして設定します。手順については、「AWS Command Line Interface ユーザーガイド」の「[AWS CLIの開始方法](#)」を参照してください。
2. プロファイルを設定します。

ユーザー認証情報を AWS CLI config ファイルに保存します。このチュートリアル of the CLI コマンドの例では、adminuser プロファイルを指定します。configuser ファイルを config ファイルに作成します。以下のように、管理者ユーザープロファイルを config ファイルのデフォルトとして設定することもできます。

```
[profile adminuser]
aws_access_key_id = admin user access key ID
aws_secret_access_key = admin user secret access key
```



```
region = us-west-2

[default]
aws_access_key_id = admin user access key ID
aws_secret_access_key = admin user secret access key
region = us-west-2
```

前述のプロファイルもデフォルトを設定します AWS リージョン。CLI コマンドでリージョンを指定しない場合、us-west-2 リージョンが使用されます。

3. コマンドプロンプトで以下のコマンドを入力して、セットアップを確認します。これらのコマンドの両方は、いずれも認証情報を明示的に提供しないため、デフォルトプロファイルの認証情報が使用されます。

- help コマンドを試してみます。

--profile パラメータを追加することで、明示的にユーザープロファイルを指定することもできます。

```
aws help
```

```
aws help \  
--profile adminuser
```

ステップ 1: EC2 リソースを作成する

このステップでは、次の作業を行います。

- 2 つの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にルールを追加して、追加のアクセスを許可します。
- EC2 インスタンスを起動します。その次の手順で EFS ファイルシステムを作成し、このインスタンス上にマウントします。

ステップ 1.1: 2 つのセキュリティグループを作成する

このセクションでは、EC2 インスタンスと EFS マウントターゲットの VPC にセキュリティグループを作成します。チュートリアル後の手順で、これらのセキュリティグループを EC2 インスタン

スと EFS マウントターゲットに割り当てます。セキュリティグループについては、「[Linux インスタンス用の Amazon EC2 セキュリティグループ](#)」を参照してください。

セキュリティグループを作成するには

1. create-security-group CLI コマンドを使用して、2 つのセキュリティグループを作成します。
 - a. EC2 インスタンス用のセキュリティグループ (efs-walkthrough1-ec2-sg) を作成し、VPC ID を指定します。

```
$ aws ec2 create-security-group \  
--region us-west-2 \  
--group-name efs-walkthrough1-ec2-sg \  
--description "Amazon EFS walkthrough 1, SG for EC2 instance" \  
--vpc-id vpc-id-in-us-west-2 \  
--profile adminuser
```

セキュリティグループ ID を書き留めます。以下に、応答の例を示します。

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

次のコマンドを使用して、VPC ID を見つけることができます。

```
$ aws ec2 describe-vpcs
```

- b. EFS マウントターゲット用のセキュリティグループ (efs-walkthrough1-mt-sg) を作成します。VPC ID を指定する必要があります。

```
$ aws ec2 create-security-group \  
--region us-west-2 \  
--group-name efs-walkthrough1-mt-sg \  
--description "Amazon EFS walkthrough 1, SG for mount target" \  
--vpc-id vpc-id-in-us-west-2 \  
--profile adminuser
```

セキュリティグループ ID を書き留めます。以下に、応答の例を示します。

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

2. セキュリティグループを確認します。

```
aws ec2 describe-security-groups \  
--group-ids list of security group IDs separated by space \  
--profile adminuser \  
--region us-west-2
```

両方には、すべてのトラフィックを残すことができるアウトバウンドルールが 1 つだけ必要です。

次のセクションでは、以下を可能にする追加アクセスを承認します。

- EC2 インスタンスに接続できるようにします。
- EC2 インスタンスと EFS マウントターゲットの間のトラフィックを有効にします (このチュートリアルの手順で、これらにセキュリティグループを関連付けます)。

ステップ 1.2: セキュリティグループにルールを追加してインバウンド/アウトバウンドアクセスを承認する

このステップでは、セキュリティグループにルールを追加して、インバウンド/アウトバウンドアクセスを許可します。

ルールを追加するには

1. 任意のホストからの Secure Shell (SSH) を使用して EC2 インスタンスに接続できるように、EC2 インスタンス (efs-walkthrough1-ec2-sg) のセキュリティグループへの受信 SSH 接続を認可します。

```
$ aws ec2 authorize-security-group-ingress \  
--group-id id of the security group created for EC2 instance \  
--protocol tcp \  
--port 22 \  
--cidr 0.0.0.0/0 \  
--profile adminuser \  

```

```
--region us-west-2
```

セキュリティグループに追加したインバウンドおよびアウトバウンドのルールがあることを確認します。

```
aws ec2 describe-security-groups \  
--region us-west-2 \  
--profile adminuser \  
--group-id security-group-id
```

2. EFS マウントターゲット (efs-walkthrough1-mt-sg) のセキュリティグループへのインバウンドアクセスを許可します。

コマンドプロンプトで、adminuser プロファイルを使用して次の AWS CLI authorize-security-group-ingress コマンドを実行し、インバウンドルールを追加します。

```
$ aws ec2 authorize-security-group-ingress \  
--group-id ID of the security group created for Amazon EFS mount target \  
--protocol tcp \  
--port 2049 \  
--source-group ID of the security group created for EC2 instance \  
--profile adminuser \  
--region us-west-2
```

3. 両方のセキュリティグループがインバウンドアクセスを許可することを確認します。

```
aws ec2 describe-security-groups \  
--group-names efs-walkthrough1-ec2-sg efs-walkthrough1-mt-sg \  
--profile adminuser \  
--region us-west-2
```

ステップ 1.3: EC2 インスタンスを起動する

このステップでは、EC2 インスタンスを起動します。

EC2 インスタンスを起動するには

1. EC2 インスタンスを起動するときに必要な次の情報を収集します。

- [キーペア名]。キーペアを作成する手順については、「Amazon EC2 ユーザーガイド」の「[Amazon EC2 インスタンスのキーペアを作成する](#)」を参照してください。
- インスタンスの起動に使用する Amazon マシンイメージ (AMI) の ID。

EC2 インスタンスを起動するために使用する AWS CLI コマンドには、パラメータとしてデプロイする Amazon マシンイメージ (AMI) の ID が必要です。この演習では、Amazon Linux HVM AMI を使用しています。

 Note

ほとんどの汎用の Linux ベースの AMI を使用することができます。別の Linux AMI を使用する場合は、必ずご使用のディストリビューションのパッケージマネージャーを使用して、インスタンスに NFS クライアントをインストールします。また、必要に応じてソフトウェアパッケージを追加する必要がある場合もあります。

Amazon Linux HVM AMI では、[Amazon Linux AMI](#) で最新の ID を見つけることができます。Amazon Linux AMI ID テーブルから次のように ID 値を選択します。

- [US West Oregon (米国西部オレゴン)] リージョンを選択します。このチュートリアルでは、米国西部 (オレゴン) リージョン (us-west-2) 内にすべてのリソースを作成していることを前提としています。
- [EBS-backed HVM 64-bit] タイプを選択します (CLI コマンドでインスタンスストアをサポートしない t2.micro インスタンスタイプを指定するため)。
- EC2 インスタンス用に作成したセキュリティグループの ID。
- AWS リージョン。このチュートリアルでは、us-west-2 リージョンを使用しています。
- インスタンスを起動する VPC サブネット ID。サブネットのリストを取得するには、describe-subnets コマンドを使用します。

```
$ aws ec2 describe-subnets \  
--region us-west-2 \  
--filters "Name=vpc-id,Values=vpc-id" \  
--profile adminuser
```

サブネット ID を選択した後、describe-subnets の結果から次の値を書き留めます。

- サブネット ID – マウントターゲットを作成するときにこの値が必要です。この練習では、EC2 インスタンスを起動するのと同じサブネットにマウントターゲットを作成します。
- サブネットのアベイラビリティゾーン - この値は、マウントターゲットの DNS 名を構成するために必要です。この名前は、EC2 インスタンスにファイルシステムをマウントするために使用します。

2. 次の AWS CLI `run-instances` コマンドを実行して EC2 インスタンスを起動します。

```
$ aws ec2 run-instances \  
--image-id AMI ID \  
--count 1 \  
--instance-type t2.micro \  
--associate-public-ip-address \  
--key-name key-pair-name \  
--security-group-ids ID of the security group created for EC2 instance \  
--subnet-id VPC subnet ID \  
--region us-west-2 \  
--profile adminuser
```

3. `run-instances` コマンドによって返されるインスタンス ID を書き留めます。
4. 作成した EC2 インスタンスには、EC2 インスタンスに接続してファイルシステムをマウントするために使用するパブリック DNS 名が必要です。パブリック DNS 名の形式は次のとおりです。

```
ec2-xx-xx-xx-xxx.compute-1.amazonaws.com
```

次の CLI コマンドを実行して、パブリック DNS 名を書き留めます。

```
aws ec2 describe-instances \  
--instance-ids EC2 instance ID \  
--region us-west-2 \  
--profile adminuser
```

パブリック DNS 名が見つからない場合は、EC2 インスタンスを起動した VPC の設定を確認してください。詳細については、「[前提条件](#)」を参照してください。

5. (オプション) 作成した EC2 インスタンスに名前を割り当てます。これを行うには、インスタンスに割り当てる名前にキー名と値を設定したタグを追加します。これを行うには、次の AWS CLI `create-tags` コマンドを実行します。

```
$ aws ec2 create-tags \  
--resources EC2-instance-ID \  
--tags Key=Name,Value=Provide-instance-name \  
--region us-west-2 \  
--profile adminuser
```

ステップ 2: EFS リソースを作成する

このステップでは、次の作業を行います。

- 暗号化された EFS ファイルシステムを作成します。
- ライフサイクル管理を有効にします。
- EFS インスタンスを起動したアベイラビリティゾーンにマウントターゲットを作成します。

ステップ 2.1: EFS ファイルシステムを作成する

このステップでは、EFS ファイルシステムを作成します。次のステップで、ファイルシステムのマウントターゲットを作成するとき使用する FileSystemId を書き留めます。

ファイルシステムを作成するには

- オプションの Name タグを使用してファイルシステムを作成します。
 - a. コマンドプロンプトで、次の AWS CLI create-file-system コマンドを実行します。

```
$ aws efs create-file-system \  
--encrypted \  
--creation-token FileSystemForWalkthrough1 \  
--tags Key=Name,Value=SomeExampleNameValue \  
--region us-west-2 \  
--profile adminuser
```

次のレスポンスが返されます。

```
{  
  "OwnerId": "111122223333",  
  "CreationToken": "FileSystemForWalkthrough1",  
  "FileSystemId": "fs-c657c8bf",
```

```
"CreationTime": 1548950706.0,
"LifecycleState": "creating",
"NumberOfMountTargets": 0,
"SizeInBytes": {
  "Value": 0,
  "ValueInIA": 0,
  "ValueInStandard": 0
},
"PerformanceMode": "generalPurpose",
"Encrypted": true,
"KmsKeyId": "arn:aws:kms:us-west-2:111122223333:a5c11222-7a99-43c8-9dcc-
abcdef123456",
"ThroughputMode": "bursting",
"Tags": [
  {
    "Key": "Name",
    "Value": "SomeExampleNameValue"
  }
]
}
```

- b. [FileSystemId] の値を書き留めます。「[ステップ2.3: マウントターゲットを作成する](#)」でこのファイルシステムのマウントターゲットを作成するときに、この値が必要になります。

ステップ 2.2: ライフサイクル管理を有効にする

このステップでは、EFS 低頻度アクセス (IA) ストレージクラスを使用するために、ファイルシステムでライフサイクル管理を有効にします。詳細については、「[ストレージライフサイクルの管理](#)」および「[EFS ストレージクラス](#)」を参照してください。

ライフサイクル管理を有効にするには

- コマンドプロンプトで、次の AWS CLI `put-lifecycle-configuration` コマンドを実行します。

```
$ aws efs put-lifecycle-configuration \
--file-system-id fs-c657c8bf \
--lifecycle-policies TransitionToIA=AFTER_30_DAYS \
--region us-west-2 \
--profile adminuser
```


次のレスポンスが返されます。

```
{
  "LifecyclePolicies": [
    {
      "TransitionToIA": "AFTER_30_DAYS"
    }
  ]
}
```

ステップ2.3: マウントターゲットを作成する

このステップでは、EC2 インスタンスを起動したアベイラビリティゾーンにファイルシステムのマウントターゲットを作成します。

1. 以下の情報があることを確認してください。

- マウント対象を作成するファイルシステムの ID (例: fs-example)。
- [ステップ 1: EC2 リソースを作成する](#) で EC2 インスタンスを起動した VPC サブネット ID。

このチュートリアルでは、EC2 インスタンスを起動したサブネットと同じサブネットにマウントターゲットを作成するため、サブネット ID (例えば subnet-example) が必要です。

- 前のステップでマウントターゲット用に作成したセキュリティグループの ID。
2. コマンドプロンプトで、次の AWS CLI create-mount-target コマンドを実行します。

```
$ aws efs create-mount-target \
--file-system-id file-system-id \
--subnet-id subnet-id \
--security-group ID-of-the security-group-created-for-mount-target \
--region us-west-2 \
--profile adminuser
```

次のレスポンスが返されます。

```
{
  "MountTargetId": "fsmt-example",
  "NetworkInterfaceId": "eni-example",
  "FileSystemId": "fs-example",
```

```
"PerformanceMode" : "generalPurpose",  
"LifecycleState": "available",  
"SubnetId": "fs-subnet-example",  
"OwnerId": "account-id",  
"IpAddress": "xxx.xx.xx.xxx"  
}
```

3. また、describe-mount-targets コマンドを使用して、ファイルシステムで作成したマウントターゲットの説明を取得することもできます。

```
$ aws efs describe-mount-targets \  
--file-system-id file-system-id \  
--region us-west-2 \  
--profile adminuser
```

ステップ 3: ファイルシステムを EC2 インスタンスにマウントしてテストする

このステップでは、次の作業を行います。

- 必要な情報を収集します。
- EC2 インスタンスに NFS クライアントをインストール。
- EC2 インスタンスにファイルシステムをマウントしてテストします。

トピック

- [ステップ 3.1: 情報を収集する](#)
- [ステップ 3.2: EC2 インスタンスに NFS クライアントをインストールする](#)
- [ステップ 3.3: ファイルシステムを EC2 インスタンスにマウントしてテストする](#)

ステップ 3.1: 情報を収集する

このセクションの手順に従って、次の情報を確認してください。

- EC2 インスタンスのパブリック DNS 名の形式は以下のとおりです。

```
ec2-xx-xxx-xxx-xx.aws-region.compute.amazonaws.com
```

- ファイルシステムの DNS 名。この DNS 名は、次の一般的な形式を使用して作成できます。

```
file-system-id.efs.aws-region.amazonaws.com
```

マウントターゲットを使用してファイルシステムをマウントする EC2 インスタンスは、ファイルシステムの DNS 名をマウントターゲットの IP アドレスに解決できます。

Note

Amazon EFS は、EC2 インスタンスがパブリック IP アドレスまたはパブリック DNS 名を持つことを要求するものではありません。前述の要件は、SSH を使用して VPC 外のインスタンスに接続するための、このチュートリアルでの一例です。

ステップ 3.2: EC2 インスタンスに NFS クライアントをインストールする

Windows または Linux、macOS X、またはその他の Unix バリエーションを実行しているコンピュータから、EC2 インスタンスに接続できます。

NFS クライアントをインストールするには

1. EC2 インスタンスに接続します。詳細については、「Amazon EC2 ユーザーガイド」の「[EC2 インスタンスに接続する](#)」を参照してください。
2. SSH セッションを使用して、EC2 インスタンスで次のコマンドを実行します。
 - a. (オプション) アップデートを入手して再起動します。

```
$ sudo yum -y update
$ sudo reboot
```

再起動後、EC2 インスタンスに再接続します。

- b. NFS クライアントをインストールします。

```
$ sudo yum -y install nfs-utils
```

 Note

EC2 インスタンスの起動時に Amazon Linux AMI 2016.03.0 という Amazon Linux AMI を選択した場合、`nfs-utils` はデフォルトで既に AMI に組み込まれているため、インストールする必要はありません。

ステップ 3.3: ファイルシステムを EC2 インスタンスにマウントしてテストする

これで、EC2 インスタンスにファイルシステムをマウントします。

1. ディレクトリを作成します (「efs-mount-point」)。

```
$ mkdir ~/efs-mount-point
```

2. EFS ファイルシステムをマウントします。

```
$ sudo mount -t nfs -o  
nfsvers=4.1,rsiz=1048576,wsiz=1048576,hard,timeo=600,retrans=2,noresvport mount-  
target-DNS:/ ~/efs-mount-point
```

EC2 インスタンスは、マウントターゲットの DNS 名を IP アドレスに解決できます。オプションでマウントターゲットの IP アドレスを直接指定することもできます。

```
$ sudo mount -t nfs -o  
nfsvers=4.1,rsiz=1048576,wsiz=1048576,hard,timeo=600,retrans=2,noresvport mount-  
target-ip:/ ~/efs-mount-point
```

3. これで EC2 インスタンスに EFS ファイルシステムがマウントされたので、ファイルを作成することができます。

- a. ディレクトリを変更します。

```
$ cd ~/efs-mount-point
```

- b. ディレクトリの内容を一覧表示します。

```
$ ls -al
```

これは空である必要があります。

```
drwxr-xr-x 2 root      root      4096 Dec 29 22:33 .  
drwx----- 4 ec2-user ec2-user 4096 Dec 29 22:54 ..
```

- c. ファイルシステムのルートディレクトリは、作成時に root ユーザーが所有し、root ユーザーによって書き込みが可能なため、ファイルを追加する権限を変更する必要があります。

```
$ sudo chmod go+rw .
```

ls -al コマンドを実行すると、権限が変更されたことがわかります。

```
drwxrwxrwx 2 root      root      4096 Dec 29 22:33 .  
drwx----- 4 ec2-user ec2-user 4096 Dec 29 22:54 ..
```

- d. テキストファイルを作成します。

```
$ touch test-file.txt
```

- e. ディレクトリのコンテンツを一覧表示します。

```
$ ls -l
```

これで、EFS ファイルシステムが正常に作成され、VPC 内の EC2 インスタンスにマウントされました。

マウントしたファイルシステムは、再起動後も保持されません。ディレクトリを自動的に再マウントするには、fstab ファイルを使用します。Auto Scaling グループを使用して EC2 インスタンスを起動する場合は、起動設定でスクリプトを設定することもできます。

ステップ 4: クリーンアップする

作成したリソースが不要になった場合は、削除する必要があります。これは CLI で行うことができます。

- EC2 リソース (EC2 インスタンスと 2 つのセキュリティグループ) を削除します。マウントターゲットを削除すると、Amazon EFS はネットワークインタフェースを削除します。
- EFS リソース (ファイルシステム、マウントターゲット) を削除します。

このチュートリアルで作成した AWS リソースを削除するには

1. このチュートリアルのために作成した EC2 インスタンスを終了します。

```
$ aws ec2 terminate-instances \
--instance-ids instance-id \
--profile adminuser
```

コンソールを使用して EC2 リソースを削除することもできます。手順については、[「インスタンスの終了」](#)を参照してください。

2. マウントターゲットを削除します。

ファイルシステムを削除する前に、そのファイルシステム用に作成されたマウントターゲットを削除する必要があります。マウントターゲットのリストを取得するには、describe-mount-targets CLI コマンドを使用します。

```
$ aws efs describe-mount-targets \
--file-system-id file-system-ID \
--profile adminuser \
--region aws-region
```

次に、delete-mount-target CLI コマンドを使用してマウントターゲットを削除します。

```
$ aws efs delete-mount-target \
--mount-target-id ID-of-mount-target-to-delete \
--profile adminuser \
--region aws-region
```

3. (オプション) 作成した 2 つのセキュリティグループを削除します。セキュリティグループを作成するための料金はかかりません。

EC2 インスタンスのセキュリティグループを削除する前に、まずマウントターゲットのセキュリティグループを削除する必要があります。マウントターゲットのセキュリティグループには、EC2 セキュリティグループを参照するルールがあります。したがって、EC2 インスタンスのセキュリティグループを最初に削除することはできません。

手順については、「Amazon EC2 ユーザーガイド」の[「セキュリティグループを削除する」](#)を参照してください。

4. `delete-file-system` CLI コマンドを使用してファイルシステムを削除します。`describe-file-systems` CLI コマンドを使用すると、ファイルシステムのリストを取得できます。レスポンスからファイルシステム ID を取得できます。

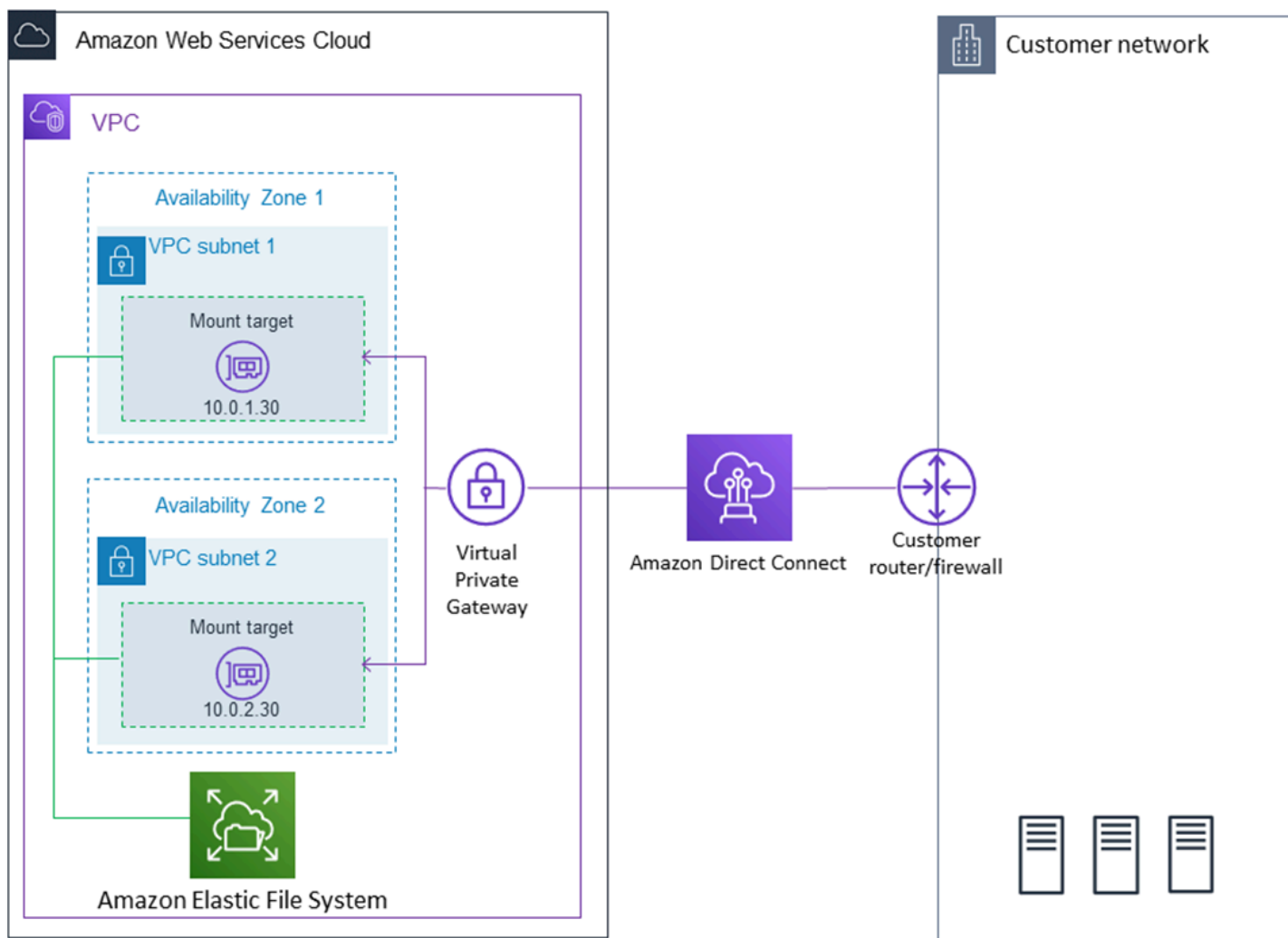
```
aws efs describe-file-systems \  
--profile adminuser \  
--region aws-region
```

ファイルシステム ID を指定してファイルシステムを削除します。

```
$ aws efs delete-file-system \  
--file-system-id ID-of-file-system-to-delete \  
--region aws-region \  
--profile adminuser
```

チュートリアル: オンプレミス Linux クライアントを使用してマウントする

AWS Direct Connect または VPN を使用して Amazon VPC に接続すると、Amazon EFS ファイルシステムをオンプレミスのデータセンターサーバーにマウントできます。次の図は、オンプレミスから Amazon EFS ファイルシステムをマウント AWS のサービス するために必要な の概略図を示しています。

**Note**

Microsoft Windows ベースのクライアントでの Amazon EFS; の使用はサポートされていません。

トピック

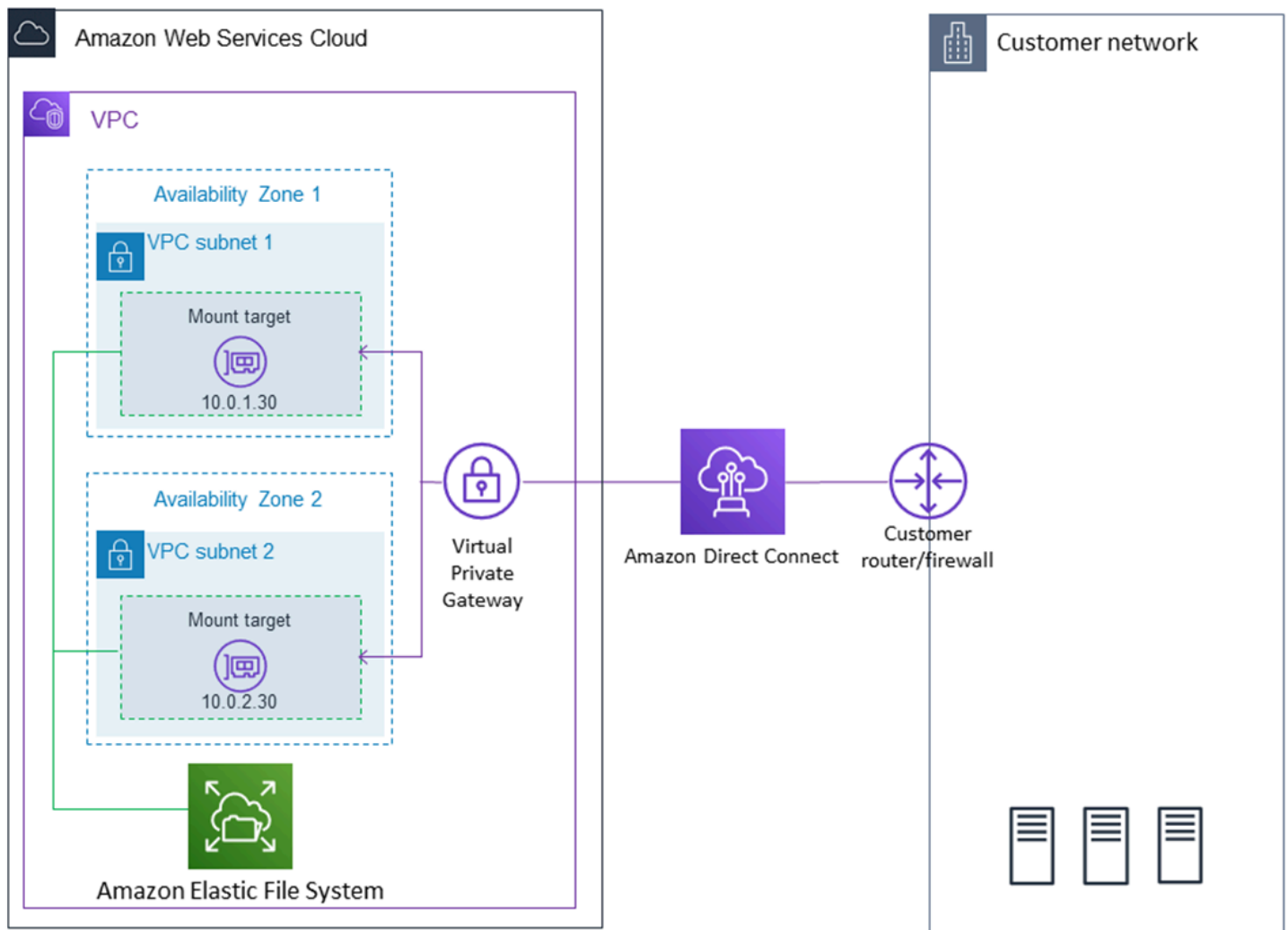
- [前提条件](#)
- [ステップ 1: EFS リソースを作成する](#)
- [ステップ 2: NFS クライアントをインストールする](#)
- [ステップ 3: オンプレミスクライアント Amazon EFS; ファイルシステムをマウントする](#)
- [ステップ 4: リソースをクリーンアップし、AWS アカウントを保護する](#)
- [オプション: 転送中のデータの暗号化](#)

前提条件

AWS Direct Connect または VPN 接続が既にあることを確認します。AWS Direct Connectの詳細については、「[AWS Direct Connect ユーザーガイド](#)」を参照してください。VPN 接続の設定の詳細については、「Amazon [VPC ユーザーガイド](#)」の「[を使用して VPC をリモートネットワークに接続する AWS Virtual Private Network](#)」を参照してください。

AWS Direct Connect または VPN 接続が完了したら、EFS ファイルシステムとマウントターゲットを Amazon VPC に作成します。その後、amazon-efs-utils ツールをダウンロードしてインストールします。次に、オンプレミスクライアントからファイルシステムをテストします。最後に、ウォークスルーの最後のクリーンアップのステップでは、これらのリソースを削除するための情報が提供されます。

このチュートリアルにより、リージョン () にこれらのリソースがすべて作成されます。どちら AWS リージョン を使用する場合でも、常に使用してください。次の図に示すように、VPC、マウントターゲット、Amazon EFS ファイルシステムなど AWS リージョン、すべてのリソースが同じにある必要があります。



Note

場合によっては、ローカルアプリケーションが EFS ファイルシステムが使用可能かどうかを知る必要があります。このような場合、最初のマウントポイントを一時的に使用できなくなった場合、アプリケーションは別のマウントポイントの IP アドレスを指すことができます。このシナリオでは、可用性を高めるために、2 つのオンプレミスクライアントを異なるアベイラビリティゾーン (AZ) を介してファイルシステムに接続することをお勧めします。

のルート認証情報を使用して AWS アカウント コンソールにサインインし、この演習を試すことができます。ただし、AWS Identity and Access Management (IAM) のベストプラクティスでは、のルート認証情報を使用しないことをお勧めします AWS アカウント。代わりに、アカウントに管理者ユーザーを作成し、それらの認証情報を使用してアカウントのリソースを管理します。詳細について

は、AWS IAM Identity Center ユーザーガイドの「[へのシングルサインオンアクセス AWS アカウント](#)」を参照してください。

アカウントで作成したデフォルトの VPC またはカスタム VPC を使用できます。このウォークスルーでは、デフォルトの VPC 設定が機能します。ただし、カスタム VPC を使用する場合は、次の点を確認してください。

- インターネットゲートウェイが VPC にアタッチされています。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 VPC のインターネットアクセスを有効にする](#)」を参照してください。
- VPC ルートテーブルには、インターネット経由のすべてのトラフィックをインターネットゲートウェイに送信するルールが含まれています。

ステップ 1: EFS リソースを作成する

このステップでは、EFS ファイルシステムとマウントターゲットを作成します。

EFS ファイルシステムを作成するには

1. Amazon Elastic File System コンソール (<https://console.aws.amazon.com/efs/>) を開きます。
2. [ファイルシステムの作成] を選択します。
3. [VPC] リストからデフォルトの VPC を選択します。
4. すべてのアベイラビリティゾーンのチェックボックスをオンにします。それらがすべてデフォルトのサブネット、自動 IP アドレス、および選択済みのデフォルトのセキュリティグループを持っていることを確認します。これらが、マウントターゲットです。詳細については、「[マウントターゲットの管理](#)」を参照してください。
5. [Next Step(次のステップ)] をクリックします。
6. ファイルシステムに名前を付け、デフォルトのパフォーマンスモードとして [general purpose (汎用)] を選択したまま [Next Step (次のステップ)] を選択します。
7. [ファイルシステムの作成] を選択します。
8. リストからファイルシステムを選択し、[Security group (セキュリティグループ)] 値を書き留めます。この値は次のステップで必要になります。

作成したファイルシステムにはマウントターゲットがあります。各マウントターゲットには、関連するセキュリティグループがあります。セキュリティグループは、ネットワークトラフィックを制御する仮想ファイアウォールとして機能します。マウントターゲットの作成時にセキュリティグループを

指定しなかった場合、Amazon EFSはVPCのデフォルトのセキュリティグループをそれに関連付けます。上記のステップに正確に従った場合、マウントターゲットはデフォルトのセキュリティグループを使用します。

次に、マウントターゲットのセキュリティグループにルールを追加して、ネットワークファイルシステム (NFS) ポート (2049) へのインバウンドトラフィックを許可します。を使用して AWS Management Console、VPC 内のマウントターゲットのセキュリティグループにルールを追加できます。

NFS ポートへのインバウンドトラフィックを許可するには

1. にサインイン AWS Management Console し、<https://console.aws.amazon.com/ec2/> で Amazon EC2 コンソールを開きます。
2. [NETWORK & SECURITY (ネットワークとセキュリティ)] で、[Security Groups (セキュリティグループ)] を選択します。
3. ファイルシステムに関連付けられているセキュリティグループを選択します。[ステップ 1: EFS リソースを作成する](#) の最後にこれを書き留めました。
4. セキュリティグループのリストの下に表示されるタブ付きペインで、[Inbound (インバウンド)] タブを選択します。
5. [編集] を選択します。
6. [Add Rule (ルールの追加)] を選択し、以下のタイプのルールを選択します。
 - タイプ – [NFS]
 - ソース – [Anywhere (任意の場所)]

テストには [Anywhere (任意の場所)] ソースのみを使用することをお勧めします。オンプレミスクライアントの IP アドレスに設定されたカスタムソースを作成するか、クライアント自体のコンソールを使用して [My IP (マイ IP)] を選択するかを選ぶことができます。

 Note

アウトバウンドルールを追加する必要はありません。これは、デフォルトのアウトバウンドルールですべてのトラフィックを残すことができるためです。このデフォルトのアウトバウンドルールがない場合は、アウトバウンドルールを追加して NFS ポート上の TCP 接続を開き、マウントターゲットのセキュリティグループを送信先として識別します。

ステップ 2: NFS クライアントをインストールする

このステップでは、NFS クライアントをインストールします。

オンプレミスサーバーに NFS クライアントをインストール

Note

転送中のデータを暗号化する必要がある場合は、NFS クライアントの代わりに Amazon EFS マウントヘルパー `amazon-efs-utils` を使用します。`amazon-efs-utils` の詳細については、「オプション: 転送中のデータの暗号化」セクションを参照してください。

1. オンプレミスクライアントのターミナルにアクセスします。
2. NFS をインストールします。

Red Hat Linux を使用している場合は、次のコマンドを使用して NFS をインストールします。

```
$ sudo yum -y install nfs-utils
```

Ubuntu を使用している場合は、次のコマンドを使用して NFS をインストールします。

```
$ sudo apt-get -y install nfs-common
```

ステップ 3: オンプレミスクライアント Amazon EFS; ファイルシステムをマウントする

マウントのディレクトリを作成するには

1. 次のコマンドを使用して、マウントポイントのディレクトリを作成します。

Example

```
mkdir ~/efs
```

2. アベイラビリティゾーンで、目的のマウントターゲットの IP アドレスを選択します。オンプレミス Linux クライアントからレイテンシーを測定できます。これを行うには、異なるアベイラ

ビリティーゾーンの EC2 インスタンスの IP アドレスに対して ping のようなターミナルベースのツールを実行して、レイテンシーが最も短いものを探します。

- mount コマンドを実行して、マウントターゲットの IP アドレスを使用してファイルシステムをマウントします。

```
$ sudo mount -t nfs -o  
nfsvers=4.1,rsize=1048576,wsiz=1048576,hard,timeo=600,retrans=2,noresvport mount-  
target-IP:/ ~/efs
```

Amazon EFS ファイルシステムをマウントしたので、次の手順でテストできます。

Amazon EFS ファイルシステム接続をテストするには

1. ディレクトリを、次のコマンドで、作成した新しいディレクトリに変更します。

```
$ cd ~/efs
```

2. サブディレクトリを作成し、そのサブディレクトリの所有権を EC2 インスタンスユーザーに変更します。次に、以下のコマンドを使用して、その新しいディレクトリに移動します。

```
$ sudo mkdir getting-started  
$ sudo chown ec2-user getting-started  
$ cd getting-started
```

3. 次のコマンドを使用してテキストファイルを作成します。

```
$ touch test-file.txt
```

4. 次のコマンドを使用して、ディレクトリの内容を一覧表示します。

```
$ ls -al
```

その結果、次のファイルが作成されます。

```
-rw-rw-r-- 1 username username 0 Nov 15 15:32 test-file.txt
```

⚠ Warning

ファイルシステムを自動的にマウントする場合、ネットワークファイルシステムを識別するために使用された `_netdev` オプションを使用します。`_netdev` が見つからない場合、EC2 インスタンスはレスポンスを停止する可能性があります。この結果は、コンピューティングインスタンスがネットワークを開始後、ネットワークファイルシステムを初期化する必要があるためです。詳細については、「[自動マウントが失敗してインスタンスがレスポンスしない](#)」を参照してください。

ステップ 4: リソースをクリーンアップし、AWS アカウントを保護する

このウォークスルーが完了したら、またはウォークスルーを調べない場合は、以下の手順に従ってリソースをクリーンアップし、AWS アカウントを保護する必要があります。

リソースをクリーンアップして を保護するには AWS アカウント

1. Amazon EFSを開始するための3番目の最後のステップ。

```
$ sudo umount ~/efs
```

2. Amazon EFS コンソールで [Amazon EFS コンソール](#)を開きます。
3. ファイルシステムのリストから削除する Amazon EFSファイルシステムを選択します。
4. [Actions (アクション)] で、[Delete file system(ファイルシステムの削除)] を選択します。
5. [ファイルシステムの完全削除]ダイアログボックスで、削除するAmazon EFSファイルシステムのファイルシステムIDを入力し、[ファイルシステムの削除]を選択します。
6. Amazon EC2 コンソールの <https://console.aws.amazon.com/ec2/> を開いてください。
7. ナビゲーションペインで、[Security Groups] を選択してください。
8. このウォークスルーのルールを追加したセキュリティグループの名前を選択します。

⚠ Warning

VPC のデフォルトのセキュリティグループを削除しないでください。

9. [アクション] メニューで、[Edit inbound rules] (インバウンドルールの編集) を選択します。
10. 追加したインバウンドルールの最後に [X] を選択し、[Save (保存)] を選択します。

オプション: 転送中のデータの暗号化

転送中のデータを暗号化する必要がある場合は、NFS クライアントの代わりに、マウントヘルパー、amazon-efs-utils を使用します

amazon-efs-utils パッケージは、ツールのオープンソースのコレクションです。amazon-efs-utils コレクションには、マウントヘルパーおよびの転送時のデータ暗号化の実行を簡単にするツールが付属しています。このパッケージの詳細については、「[Amazon EFS クライアントの手動インストール](#)」を参照してください。このパッケージは、パッケージのリポジトリを複製することで GitHub から無料でダウンロードできます。

GitHub から amazon-efs-utils のクローンを作成するには

1. オンプレミスクライアントのターミナルにアクセスします。
2. ターミナルから以下のコマンドで、GitHub の amazon-efs-utils ツールを任意のディレクトリにクローンします。

```
git clone https://github.com/aws/efs-utils
```

パッケージが手に入ったため、それをインストールすることができます。このインストールは、オンプレミスクライアントの Linux ディストリビューションによって異なります。以下のディストリビューションがサポートされています。

- Amazon Linux 2
- Amazon Linux
- Red Hat Enterprise Linux (および、その派生物 CentOS など) バージョン 7 以降
- Ubuntu 16.04 LTS 以降

amazon-efs-utils を RPM パッケージとしてビルドおよびインストールするには

1. クライアント上のターミナルを開き、GitHub からクローンされた amazon-efs-utils パッケージのあるディレクトリに移動します。
2. 次のコマンドを使用して、パッケージをビルドします。

```
make rpm
```


 Note

rpm-builder パッケージをまだインストールしていない場合は、次のコマンドを使用してインストールします。

```
sudo yum -y install rpm-build
```

3. 次のコマンドでパッケージをインストールします。

```
sudo yum -y install build/amazon-efs-utils*rpm
```

amazon-efs-utils を deb パッケージとしてビルドおよびインストールするには

1. クライアント上のターミナルを開き、GitHub からクローンされた amazon-efs-utils パッケージのあるディレクトリに移動します。
2. 次のコマンドを使用して、パッケージをビルドします。

```
./build-deb.sh
```

3. 次のコマンドでパッケージをインストールします。

```
sudo apt-get install build/amazon-efs-utils*deb
```

パッケージをインストールしたら、AWS Direct Connect または VPN AWS リージョン で使用するために amazon-efs-utils を設定します。

で使用する amazon-efs-utils を設定するには AWS リージョン

1. 任意のテキストエディタを使用して、編集のために /etc/amazon/efs/efs-utils.conf ファイルを開きます。
2. “dns_name_format = {fs_id}.efs.{region}.amazonaws.com” という行を探します。
3. us-west-2 のように、{region} を AWS リージョンの ID に変更します。

オンプレミスのクライアントに EFS ファイルシステムをマウントするには、まず、オンプレミス Linux クライアントでターミナルを開きます。システムをマウントするには、ファイルシステム ID、

ユーザーのマウントターゲット内のマウントターゲット IP アドレス、ファイルシステムの AWS リージョンリージョンが必要です。ファイルシステムに複数のマウントターゲットを作成した場合は、これらのいずれかを選択できます。

その情報がある場合は、次の 3 つのステップで、ファイルシステムをマウントできます。

マウントのディレクトリを作成するには

1. 次のコマンドを使用して、マウントポイントのディレクトリを作成します。

Example

```
mkdir ~/efs
```

2. アベイラビリティーゾーンで、目的のマウントターゲットの IP アドレスを選択します。オンプレミス Linux クライアントからレイテンシーを測定できます。これを行うには、異なるアベイラビリティーゾーンの EC2 インスタンスの IP アドレスに対して ping のようなターミナルベースのツールを実行して、レイテンシーが最も短いものを探します。

/etc/hosts を更新するには

- ローカルの `/etc/hosts` ファイルに、ファイルシステム ID とマウントターゲットの IP アドレスを含むエントリを次の形式で追加します。

```
mount-target-IP-Address file-system-ID.efs.region.amazonaws.com
```

Example

```
192.0.2.0 fs-12345678.efs.us-west-2.amazonaws.com
```

マウントのディレクトリを作成するには

1. 次のコマンドを使用して、マウントポイントのディレクトリを作成します。

Example

```
mkdir ~/efs
```

2. マウントコマンドを使用してファイルシステムをマウントします。

Example

```
sudo mount -t efs fs-12345678 ~/efs
```

転送時にデータの暗号化を使用する場合、マウントコマンドは次のようになります。

Example

```
sudo mount -t efs -o tls fs-12345678 ~/efs
```

チュートリアル: 別の VPC からファイルシステムをマウントする

このチュートリアルでは、EC2 インスタンスを設定して、別の仮想プライベートクラウド (VPC) にある EFS ファイルシステムをマウントします。これは、EFS マウントヘルパーを使用して行います。マウントヘルパーは、amazon-efs-utils というツールセットの一部です。amazon-efs-utils の詳細については、「[Amazon EFS クライアントの手動インストール](#)」を参照してください。

クライアントの VPC と EFS ファイルシステムの VPC を接続するには、VPC ピアリング接続または VPC トランジットゲートウェイを使用する必要があります。VPC ピアリング接続またはトランジットゲートウェイを使用して VPC に接続すると、ある VPC 上の EC2 インスタンスから別の VPC の EFS ファイルシステムにアクセスすることができます。VPC 同士が異なるアカウントに属していても可能です。

Note

Microsoft Windows ベースのクライアントでの Amazon EFS; の使用はサポートされていません。

トピック

- [前提条件](#)
- [ステップ 1: マウントターゲットのアベイラビリティーゾーンの ID を特定する](#)
- [ステップ 2: マウントターゲットの IP アドレスを特定する](#)
- [ステップ 3: マウントターゲットにホストエントリを追加する](#)
- [ステップ 4: EFS マウントヘルパーを使用してファイルシステムをマウントする](#)

• [ステップ 5: リソースをクリーンアップして AWS アカウントを保護する](#)

前提条件

このチュートリアルを完了するには、以下が必要です。

- この手順を使用する前に、amazon-efs-utils ツールセットが EC2 インスタンスにインストールされていること。amazon-efs-utils のインストール手順については、「[Amazon EFS クライアントの手動インストール](#)」を参照してください。
- 次のいずれかです:
 - EFS ファイルシステムが存在する VPC と EC2 インスタンスが存在する VPC との間の VPC ピアリング接続。VPC ピアリング接続は、2 つの VPC 間のネットワーク接続です。このタイプの接続では、インターネットプロトコルバージョン 4 (IPv4) またはインターネットプロトコルバージョン 6 (IPv6) のプライベートアドレスを使用して、2 つの VPC 間でトラフィックをルーティングできます。VPC ピアリングを使用して、同じ 内 AWS リージョン または 間で VPCs を接続できます AWS リージョン。詳細については、Amazon VPC ピアリングガイドの「[VPC ピア接続の作成と承認](#)」を参照してください。
 - EFS ファイルシステムが存在する VPC と EC2 インスタンスが存在する VPC を接続するトランジットゲートウェイ。トランジットゲートウェイは、VPC とオンプレミスネットワークを相互接続するために使用できるネットワークの中継ハブです。詳細については、「[Amazon VPC Transit Gateways Guide](#)」の「[Getting Started with using Amazon VPC Transit Gateways](#)」を参照してください。

ステップ 1: マウントターゲットのアベイラビリティーゾーンの ID を特定する

ファイルシステムの高可用性を確保するために、NFS クライアントと同じアベイラビリティーゾーンにある EC2 マウントターゲットの IP アドレスを常に使用することをお勧めします。別のアカウントにある EFS ファイルシステムをマウントする場合は、NFS クライアントと EFS マウントターゲットが同じアベイラビリティーゾーン ID にあることを確認します。この要件が適用されるのは、アベイラビリティーゾーン名がアカウントによって異なる可能性があるためです。

EC2 インスタンスのアベイラビリティーゾーン ID を決定するには

1. EC2 インスタンスに接続します。詳細については、「Amazon EC2 ユーザーガイド」の「[EC2 インスタンスに接続する](#)」を参照してください。

2. EC2 インスタンスがある アベイラビリティーゾーン ID を特定するには、次のように `describe-availability-zones` CLI コマンドを使用します。

```
[ec2-user@ip-10.0.0.1] $ aws ec2 describe-availability-zones --zone-name
{
  "AvailabilityZones": [
    {
      "State": "available",
      "ZoneName": "us-east-2b",
      "Messages": [],
      "ZoneId": "use2-az2",
      "RegionName": "us-east-2"
    }
  ]
}
```

このアベイラビリティーゾーン ID は、`ZoneId` プロパティ、`use2-az2` に返されます。

ステップ 2: マウントターゲットの IP アドレスを特定する

EC2 インスタンスの アベイラビリティーゾーン ID を特定したら、同じ アベイラビリティーゾーン ID にあるマウントターゲットの IP アドレスを取得します。

同じ アベイラビリティーゾーン ID のマウントターゲットの IP アドレスを特定するには

- 次のように CLI コマンド `describe-mount-targets` を使用して、AZ ID `use2-az2` 内のファイルシステムに対するマウントターゲットの IP アドレスを取得します。

```
$ aws efs describe-mount-targets --file-system-id file_system_id
{
  "MountTargets": [
    {
      "OwnerId": "111122223333",
      "MountTargetId": "fsmt-11223344",
      "AvailabilityZoneId": "use2-az2",
      "NetworkInterfaceId": "eni-048c09a306023eeec",
      "AvailabilityZoneName": "us-east-2b",
      "FileSystemId": "fs-01234567",
      "LifecycleState": "available",
      "SubnetId": "subnet-06eb0da37ee82a64f",
      "OwnerId": "958322738406",
      =====>
```

```
=====>  "IpAddress": "10.0.2.153"
        },
    ...
    {
        "OwnerId": "111122223333",
        "MountTargetId": "fsmt-667788aa",
        "AvailabilityZoneId": "use2-az3",
        "NetworkInterfaceId": "eni-0edb579d21ed39261",
        "AvailabilityZoneName": "us-east-2c",
        "FileSystemId": "fs-01234567",
        "LifecycleState": "available",
        "SubnetId": "subnet-0ee85556822c441af",
        "OwnerId": "958322738406",
        "IpAddress": "10.0.3.107"
    }
]
}
```

アベイラビリティゾーン ID use2-az2 内のマウントターゲットの IP アドレスは 10.0.2.153 です。

ステップ 3: マウントターゲットにホストエントリを追加する

次は、EC2 インスタンスの `/etc/hosts` ファイルに、マウントターゲット IP アドレスを EFS ファイルシステムのホスト名にマッピングするエントリを作成します。

マウントターゲットのホストエントリを追加するには

1. EC2 インスタンスの `/etc/hosts` ファイルに、マウントターゲットの IP アドレスの行を追加します。このエントリには、`mount-target-IP-Address file-system-ID.efs.region.amazonaws.com` という形式を使用します。ファイルに行を追加するには、次のコマンドを使用します。

```
echo "10.0.2.153 fs-01234567.efs.us-east-2.amazonaws.com" | sudo tee -a /etc/hosts
```

2. EC2 インスタンスとマウントターゲットの VPC セキュリティグループに、必要に応じて EFS ファイルシステムへのアクセスを許可するルールがあることを確認してください。詳細については、「[VPC セキュリティグループの使用](#)」を参照してください。

ステップ 4: EFS マウントヘルパーを使用してファイルシステムをマウントする

EFS ファイルシステムをマウントするには、まず EC2 インスタンスにマウントディレクトリを作成します。次に、EFS マウントヘルパーを使用して、AWS Identity and Access Management (IAM) 認可または EFS アクセスポイントを使用してファイルシステムをマウントできます。詳細については、「[IAM を使用してファイルシステムへのアクセスを制御する](#)」および「[アクセスポイントの使用](#)」を参照してください。

マウントのディレクトリを作成するには

- 次のコマンドを使用して、ファイルシステムをマウントするためのディレクトリを作成します。

```
$ sudo mkdir /mnt/efs/
```

IAM 認可を使用してファイルシステムをマウントするには

- IAM 認可を使用してファイルシステムをマウントするには、次のコマンドを使用します。

```
$ sudo mount -t efs -o tls,iam file-system-id /mnt/efs/
```

EFS アクセスポイントを使用してファイルシステムをマウントするには

- EFS アクセスポイントを使用してファイルシステムをマウントするには、次のコマンドを使用します。

```
$ sudo mount -t efs -o tls,accesspoint=access-point-id file-system-id /mnt/efs/
```

これで EFS ファイルシステムがマウントされたので、次の手順でテストすることができます。

EFS ファイルシステムの接続をテストするには

- ディレクトリを、次のコマンドで、作成した新しいディレクトリに変更します。

```
$ cd ~/mnt/efs
```

- サブディレクトリを作成し、そのサブディレクトリの所有権を EC2 インスタンスユーザーに変更します。次に、以下のコマンドを使用して、その新しいディレクトリに移動します。

```
$ sudo mkdir getting-started
$ sudo chown ec2-user getting-started
$ cd getting-started
```

- 次のコマンドを使用してテキストファイルを作成します。

```
$ touch test-file.txt
```

- 次のコマンドを使用して、ディレクトリの内容を一覧表示します。

```
$ ls -al
```

その結果、次のファイルが作成されます。

```
-rw-rw-r-- 1 username username 0 Nov 15 15:32 test-file.txt
```

/etc/fstab ファイルにエントリを追加することで、自動的にファイルシステムをマウントすることもできます。詳細については、「[既存の EC2 Linux インスタンスでの自動マウントの有効化](#)」を参照してください。

Warning

ファイルシステムを自動的にマウントする場合、ネットワークファイルシステムを識別するために使用された `_netdev` オプションを使用します。`_netdev` が見つからない場合、EC2 インスタンスはレスポンスを停止する可能性があります。この結果は、コンピューティングインスタンスがネットワークを開始後、ネットワークファイルシステムを初期化する必要がありますためです。詳細については、「[自動マウントが失敗してインスタンスがレスポンスしない](#)」を参照してください。

ステップ 5: リソースをクリーンアップして AWS アカウントを保護する

このチュートリアルを完了したら、次のステップを実行してリソースをクリーンアップし、を保護します AWS アカウント。

リソースをクリーンアップして を保護するには AWS アカウント

1. 次のコマンドで、EFS ファイルシステムをアンマウントします。

```
$ sudo umount ~/efs
```

2. Amazon Elastic File System コンソール の <https://console.aws.amazon.com/efs/> を開いてください。
3. ファイルシステムのリストから削除する EFS ファイルシステムを選択します。
4. [Actions] (アクション) で、[Delete file system] (ファイルシステムの削除) を選択します。
5. [ファイルシステムを完全に削除] ダイアログボックスで、削除する EFS ファイルシステムのファイルシステム ID を入力して、[ファイルシステムの削除] を選択します。
6. Amazon EC2 コンソールの <https://console.aws.amazon.com/ec2/> を開いてください。
7. ナビゲーションペインで、[Security Groups] を選択してください。
8. このチュートリアルでルールを追加したセキュリティグループの名前を選択します。

 Warning

VPC のデフォルトのセキュリティグループを削除しないでください。

9. [アクション] メニューで、[Edit inbound rules] (インバウンドルールの編集) を選択します。
10. 追加したインバウンドルールの最後に [X] を選択し、[Save (保存)] を選択します。

マウントの問題のトラブルシューティング

以下では、EFS ファイルシステムのマウントに関する問題のトラブルシューティングについて説明します。

Windows インスタンスでのファイルシステムのマウントが失敗する

Microsoft Windows の Amazon EC2 インスタンスで、ファイルシステムのマウントが失敗します。

実行するアクション

Amazon EFS を Windows EC2 インスタンスで使用しないでください。これはサポートされていません。

サーバーによってアクセスが拒否されました

ファイルシステムのマウントが失敗し、次のメッセージが表示されます。

```
/efs mount.nfs4: access denied by server while mounting 127.0.0.1:/
```

この問題は、ファイルシステムをマウントするためのアクセス許可が NFS クライアントにない場合に発生することがあります。

実行するアクション

IAM を使用してファイルシステムをマウントする場合は、マウントコマンドで `-o iam` または `-o tls` オプションを使用していることを確認してください。これにより、EFS マウントヘルパーから EFS マウントターゲットに認証情報を渡すように指示します。それでもアクセスできない場合は、ファイルシステムポリシーと ID ポリシーをチェックして、接続に DENY 句が適用されていないこと、および接続に ALLOW 句が少なくとも 1 つ適用されていることを確認します。詳細については、「[IAM を使用してファイルシステムへのアクセスを制御する](#)」および「[ファイルシステムポリシーの作成](#)」を参照してください。

自動マウントが失敗してインスタンスがレスポンスしない

この問題は、ファイルシステムがインスタンスで自動的にマウントされ、`_netdev` オプションが宣言されていない場合に発生することがあります。`_netdev` が見つからない場合、EC2 インスタンスはレスポンスを停止する可能性があります。この結果は、コンピューティングインスタンスがネットワークを開始後、ネットワークファイルシステムを初期化する必要があるためです。

実行するアクション

この問題が発生した場合は、AWS サポートにお問い合わせください。

/etc/fstab での複数の Amazon EFS ファイルシステムのマウントが失敗する

/etc/fstab で 2 つ以上の Amazon EFS エントリを使用して systemd init システムを使用するインスタンスの場合、これらのエントリの一部またはすべてがマウントされないことがあります。この場合、`dmesg` 出力で以下のような 1 つ以上の行が表示されます。

```
NFS: nfs4_discover_server_trunking unhandled error -512. Exiting with error EIO
```

実行するアクション

この場合、`/etc/systemd/system/mount-nfs-sequentially.service` に新しい `systemd` サービスファイルを作成することをお勧めします。ファイルに含めるコードは、ファイルシステムを手動でマウントするか、Amazon EFS マウントヘルパーを使用するかによって異なります。

- ファイルシステムを手動でマウントする場合、`ExecStart` コマンドはネットワークファイルシステム (NFS4) を指している必要があります。ファイルに次のコードを含めます。

```
[Unit]
Description=Workaround for mounting NFS file systems sequentially at boot time
After=remote-fs.target

[Service]
Type=oneshot
ExecStart=/bin/mount -avt nfs4
RemainAfterExit=yes

[Install]
WantedBy=multi-user.target
```

- Amazon EFS マウントヘルパーを使用している場合、Transport Layer Security (TLS) を使用するためには、`ExecStart` コマンドは NFS4 ではなく EFS を指している必要があります。ファイルに次のコードを含めます。

```
[Unit]
Description=Workaround for mounting NFS file systems sequentially at boot time
After=remote-fs.target

[Service]
Type=oneshot
ExecStart=/bin/mount -avt efs
RemainAfterExit=yes

[Install]
WantedBy=multi-user.target
```

ファイル作成後、次の 2 つのコマンドを実行します。

- `sudo systemctl daemon-reload`
- `sudo systemctl enable mount-nfs-sequentially.service`

次に、Amazon EC2 インスタンスを再起動します。通常 1 秒以内にオンデマンドでファイルシステムがマウントされます。

エラーメッセージ「wrong fs type」でマウントコマンドが失敗する

マウントコマンドが失敗し、次のエラーメッセージが表示されます。

```
mount: wrong fs type, bad option, bad superblock on 10.1.25.30:/,
missing codepage or helper program, or other error (for several filesystems
(e.g. nfs, cifs) you might need a /sbin/mount.<type> helper program)
In some cases useful info is found in syslog - try dmesg | tail or so.
```

実行するアクション

このメッセージが表示された場合、nfs-utils (または Ubuntu では nfs-common) パッケージをインストールします。詳細については、「[NFS クライアントをインストールする](#)」を参照してください。

エラーメッセージ「incorrect mount option」でマウントコマンドが失敗する

マウントコマンドが失敗し、次のエラーメッセージが表示されます。

```
mount.nfs: an incorrect mount option was specified
```

実行するアクション

このエラーメッセージは、ほとんどの場合、ご利用の Linux ディストリビューションが Network File System バージョン 4.0 および 4.1 (NFSv4) をサポートしていないことを意味します。これが該当するかどうかを確認するには、次のコマンドを実行します。

```
$ grep CONFIG_NFS_V4_1 /boot/config*
```

前述のコマンドが # CONFIG_NFS_V4_1 is not set を返すなら、ご利用の Linux ディストリビューションで NFSv4.1 がサポートされていません。NFSv4.1 をサポートする Amazon Elastic Compute Cloud (Amazon EC2) の Amazon マシンイメージ (AMI) の一覧については、「[NFS サポート](#)」を参照してください。

アクセスポイントでのマウントは失敗します。

アクセスポイントでマウントすると、マウントコマンドが失敗し、次のエラーメッセージが表示されます。

```
mount.nfs4: mounting access_point failed, reason given by server: No such file or directory
```

実行するアクション

このエラーメッセージは、指定した EFS パスが存在しないことを示します。アクセスポイントのルートディレクトリの所有権とアクセス許可を必ず指定してください。この情報がないと EFS はルートディレクトリを作成しません。詳細については、「[アクセスポイントの使用](#)」を参照してください。

ルートディレクトリの所有権とアクセス許可を指定せず、ルートディレクトリがまだ存在しない場合、EFS はルートディレクトリを作成しません。この場合、アクセスポイントを使用してファイルシステムをマウントしようとするすると失敗します。

ファイルシステムを作成した後すぐにファイルシステムのマウントが失敗する

マウントターゲットを作成した後、ドメインネームサービス (DNS) のレコードが AWS リージョンに完全に伝達するには、最大 90 秒かかります。

実行するアクション

たとえば、AWS CloudFormation テンプレートを使用してファイルシステムをプログラムで作成およびマウントする場合は、待機条件を実装することをお勧めします。

ファイルシステムのマウントがハングした後、タイムアウトエラーで失敗する

ファイルシステムのマウントコマンドが 1、2 分間ハングし、タイムアウトエラーで失敗します。次のコードは例を示しています。

```
$ sudo mount -t nfs -o  
nfsvers=4.1,rsiz=1048576,wsiz=1048576,hard,timeo=600,retrans=2,noresvport mount-  
target-ip:/ mnt
```

```
[2+ minute wait here]
mount.nfs: Connection timed out
$
```

実行するアクション

このエラーは、Amazon EC2 インスタンスまたはマウントターゲットのセキュリティグループが正しく設定されていないために発生することがあります。マウントターゲットセキュリティグループに、EC2 セキュリティグループから NFS へのアクセスを許可するインバウンドルールがあることを確認します。詳細については、「[VPC セキュリティグループの使用](#)」を参照してください。

指定したマウントターゲットの IP アドレスが有効であることを確認します。間違った IP アドレスを指定し、その IP アドレスにマウントを拒否するものが他に何も無い場合、この問題が生じることがあります。

DNS 名を使用した NFS によるファイルシステムのマウントが失敗する

次の例に示すように、ファイルシステムの DNS 名を使用して (amazon-efs-utils クライアントを使用せずに) NFS クライアントを使用してファイルシステムをマウントしようとすると失敗します。

```
$ sudo mount -t nfs -o
nfsvers=4.1,rsz=1048576,wsz=1048576,hard,timeo=600,retrans=2,noresvport file-
system-id.efs.aws-region.amazonaws.com:/ mnt
mount.nfs: Failed to resolve server file-system-id.efs.aws-region.amazonaws.com:
Name or service not known.

$
```

実行するアクション

VPC 設定を確認します。カスタム VPC を使用している場合は、DNS 設定が有効であることを確認します。詳細については Amazon VPC ユーザーガイドの「[DNS attributes for your VPC](#)」(VPC の DNS 属性) を参照してください。また、ファイルシステムおよびマウントターゲット DNS 名は、存在している VPC 外部から解決できません。

mount コマンドの DNS 名を使用してファイルシステムをマウントするには、次の作業を行う必要があります。

- Amazon EC2 インスタンスと同じアベイラビリティゾーンに Amazon EFS マウントターゲットがあることを確認します。

- Amazon EC2 インスタンスと同じ VPC にマウントターゲットがあることを確認します。そうしないと、別の VPC で EFS マウントターゲットに DNS 名前解決を使用することができません。詳細については、「[別の AWS アカウント または VPC からの EFS ファイルシステムのマウント](#)」を参照してください。
- Amazon が提供する DNS サーバーを使用するように設定された Amazon VPC 内で Amazon EC2 インスタンスを接続します。詳細については、「Amazon VPC ユーザーガイド」の「[Amazon VPC の DHCP オプションセット](#)」を参照してください。
- 接続する Amazon EC2 インスタンスの Amazon VPC で、DNS ホスト名が有効であることを確認します。詳細については、「Amazon VPC ユーザーガイド」の「[DNS attributes in your VPC](#)」(VPC の DNS 属性) を参照してください。

「nfs が応答していません」が表示されてファイルシステムのマウントが失敗する

"nfs: server_name still not responding" との Transmission Control Protocol (TCP) 再接続イベントで Amazon EFS ファイルシステムのマウントが失敗します。

実行するアクション

ネットワーク接続が再確立されたときに、NFS クライアントが新しい TCP ソースポートを使用することを確認するには、noresvport マウントオプションを使用します。これにより、ネットワーク復旧イベント後の中断のない可用性が保証されます。

マウントターゲットのライフサイクル状態がスタックする

マウントターゲットのライフサイクル状態が [creating (作成中)] または [deleting (削除中)] の状態でスタックします。

実行するアクション

CreateMountTarget または DeleteMountTarget の呼び出しを再試行します。

マウントターゲットのライフサイクルの状態にエラーが表示される

マウントターゲットのライフサイクルステータスが「エラー」と表示されます。

実行するアクション

仮想プライベートクラウド (VPC) に競合するホストゾーンがある場合、Amazon EFS は新しいファイルシステムのマウントターゲットに必要なドメインネームシステム (DNS) レコードを作成できません。Amazon EFS は、カスタマー所有のホストゾーン内に新しいレコードを作成することはできません。efs.<region>.amazonaws.com DNS 範囲が競合するホストゾーンを維持する必要がある場合は、別の VPC にホストゾーンを作成します。VPC の DNS に関する考慮事項に関する詳細については、「[VPC の DNS 属性](#)」を参照してください。

この問題を解決するには、VPC から競合している efs.<region>.amazonaws.com ホストを削除して、マウントターゲットを再作成してください。マウントターゲットの削除について詳細は、「[マウントターゲットの管理](#)」を参照してください。

マウントが応答しない

Amazon EFS マウントが応答していないようにみえます。たとえば、ls のようなコマンドがハングします。

実行するアクション

別のアプリケーションが大量のデータをファイルシステムに書き込んでいる際に、このエラーが発生することがあります。オペレーションが完了するまで、書き込み中のファイルへのアクセスがブロックされている可能性があります。一般的に、書き込まれているファイルにアクセスしようとするコマンドまたはアプリケーションはハングしているように見えます。たとえば、ls コマンドは、書き込み中のファイルに実行された場合、ハングする場合があります。この結果は、一部の Linux ディストリビューションでは、ls コマンドに別名を付け、ディレクトリの内容を一覧表示するだけでなく、ファイルの属性を取得するためです。

この問題を解決するには、以下の例のように、別のアプリケーションが Amazon EFS マウントにファイルを書き込んでいること、また、それが Uninterruptible sleep (D) 状態であることを確認します。

```
$ ps aux | grep large_io.py
root 33253 0.5 0.0 126652 5020 pts/3 D+ 18:22 0:00 python large_io.py /efs/large_file
```

このような場合には、他の書き込み操作が完了するのを待つか、回避策を実装して問題を解決できます。ls の例では、エイリアスの代わりに直接 /bin/ls コマンドを使用できます。これにより、書き込まれているファイルにハングすることなくコマンドを続行できます。一般的に、データを書き込むアプリケーションが (おそらく fsync(2) を使用して) 定期的にデータを強制的にフラッシュする場合、そうすることで他のアプリケーションへのファイルシステムの応答性が向上する可能性があります。

ます。ただし、この改善は、アプリケーションがデータを書き込むときのパフォーマンスを犠牲にする可能性があります。

マウントされたクライアントは切断されます。

Amazon EFS ファイルシステムにマウントされたクライアントは、さまざまな原因により切断されることがあります。NFS クライアントは、日常的な切断がアプリケーションのパフォーマンスと可用性に与える影響を最小限に抑えるため、中断した場合は自動的に再接続するように設計されています。ほとんどの場合、クライアントは数秒以内に透過的に再接続します。

ただし、古いバージョンの Linux カーネル (バージョン v5.4 以下) に含まれている NFS クライアントソフトウェアには、切断時に NFS クライアントが同じ TCP ソースポートに再接続を試みるという動作が含まれています。この動作は TCP RFC に準拠していないため、これらのクライアントが NFS サーバー (この場合は EFS ファイルシステム) への接続をすぐに再確立できなくなる可能性があります。

この問題を解決するために、Amazon EFS マウントヘルパーを使用して、EFS ファイルシステムをマウントすることを強くお勧めします。EFS マウントヘルパーは、Amazon EFS ファイルシステム用に最適化されたマウント設定を使用します。EFS クライアントおよびマウントヘルパーに関する詳細については、「[Amazon EFS クライアントの手動インストール](#)」を参照してください。

EFS マウントヘルパーを使用できない場合は、この問題を回避するために、新しい TCP ソースポートを使用して接続を再確立するよう NFS クライアントに指示する `noresvport` NFS マウントオプションの使用を強くお勧めします。詳細については、「[推奨される NFS マウント設定](#)」を参照してください。

新しくマウントされたファイルシステムでの操作が「bad file handle」エラーを返します

新しくマウントされたファイルシステムで実行された操作が `bad file handle` エラーを返します。

このエラーは、Amazon EC2 インスタンスが 1 つのファイルシステムおよび 1 つのマウントターゲットに指定された IP アドレスで接続され、そのファイルシステムとマウントターゲットが削除された場合に発生します。同じマウントターゲットの IP アドレスを持つ Amazon EC2 インスタンスに接続するために新しいファイルシステムとマウントターゲットを作成すると、この問題が発生する可能性があります。

実行するアクション

このエラーは、ファイルシステムをアンマウントし、Amazon EC2 インスタンス上のファイルシステムを再マウントすることで解決できます。Amazon EFS ファイルシステムのアンマウントの詳細については、[ファイルシステムをアンマウントする](#) を参照してください。

ファイルシステムのアンマウントが失敗する

ファイルシステムが使用中の場合、アンマウントすることはできません。

実行するアクション

この問題は以下の方法で解決できます。

- 遅延アンマウント、`umount -l` を使用します。これは、実行時にファイルシステム階層からファイルシステムを切り離し、ビジー状態でなくなるとすぐにファイルシステムへのすべてのリファレンスをクリーンアップします。
- すべての読み取りおよび書き込み操作が終了するまで待機してから、`umount` コマンドを再実行してください。
- `umount -f` コマンドを使用して強制的にアンマウントします。

Warning

強制的なアンマウントにより、ファイルシステムで現在処理中の、すべてのデータ読み込みまたは書き込み操作が中断されます。このオプションを使用する場合の詳細とガイダンスについては、[umount man page](#) を参照してください。

Amazon EFS との間でのデータの転送

AWS DataSync とを使用して AWS Transfer Family、Amazon EFS ファイルシステムとの間でデータを転送できます。AWS DataSync は、ネットワークファイルシステム (NFS)、サーバーメッセージブロック (SMB) ファイルサーバー、セルフマネージドオブジェクトストレージ、および サービス間で AWS データをコピーできるオンラインデータ転送サービスです。Amazon EFS での DataSync の使用の詳細については、「[AWS DataSync を使用したデータ転送](#)」を参照してください。

AWS Transfer Family は、Secure File Transfer Protocol (SFTP)、File Transfer Protocol (FTP)、および Secure Sockets Layer (FTPS) プロトコルを介した FTP を介して、Amazon EFS ファイルシステムとの間でファイルを転送するために使用できるフルマネージド AWS サービスです。Transfer Family を使用すると、データ配信、サプライチェーン、コンテンツ管理、ウェブサービスアプリケーションなどのユースケースで、Amazon EFS ファイルシステムに格納されているファイルへのアクセスをビジネスパートナーに提供できます。Amazon EFS での Transfer Family の使用方法については、「[AWS Transfer Family を使用したデータ転送](#)」を参照してください。

トピック

- [AWS DataSync を使用したデータ転送](#)
- [AWS Transfer Family を使用したデータ転送](#)

AWS DataSync を使用したデータ転送

AWS DataSync は、オンプレミスのストレージシステム間、およびストレージサービス間の AWS データの移動とレプリケーションを簡素化、自動化、および高速化するオンラインデータ転送サービスです。DataSync は、ネットワークファイルシステム (NFS)、サーバーメッセージブロック (SMB) ファイルサーバー、セルフマネージドオブジェクトストレージ AWS Snowball Edge、Amazon S3 バケット、EFS ファイルシステム、FSx for Windows File Server ファイルシステム間でデータをコピーできます。

DataSync を使用して、異なる のファイルシステムや異なる が所有するファイルシステムなど、2 AWS リージョンつの EFS ファイルシステム間でファイルを転送することもできます AWS アカウント。DataSync を使用して EFS ファイルシステム間でデータをコピーすると、1 回限りのデータ移行、分散ワークロード用の定期的なデータ取り込み、およびデータ保護と回復のためのレプリケーションの自動化を実行できます。

詳細については、[Amazon EFS の開始方法](#) および [AWS DataSync ユーザーガイド](#)を参照してください。

AWS Transfer Family を使用したデータ転送

AWS Transfer Family は、次のプロトコルを使用して Amazon EFS ファイルシステムとの間でファイルを転送するために使用できるフルマネージド AWS サービスです。

- Secure Shell (SSH) File Transfer Protocol (SFTP) (AWS Transfer for SFTP)
- File Transfer Protocol Secure (FTPS) (AWS Transfer for FTPS)
- ファイル転送プロトコル (FTP) (AWS Transfer for FTP)

Transfer Familyを使用すると、インフラストラクチャを管理する必要なく、ベンダー、パートナー、顧客などのサードパーティーが、サポートされているプロトコルを使ってグローバル規模で安全にファイルにアクセスできるようになります。さらに、SFTP、FTPS、および FTP クライアントを使用して、Windows、macOS、Linux 環境から EFS ファイルシステムに簡単にアクセスできるようになりました。これにより、NFS クライアントやアクセスポイント以外にも、複数の環境にわたるユーザーへのデータのアクセシビリティが拡張されます。

Transfer Family を使用して Amazon EFS ファイルシステム内のデータを転送することは、他のクライアントの使用法と同じ方法で説明できます。詳細については、「[スループットモード](#)」および「[Amazon EFS のクォータ](#)」を参照してください。

詳細については AWS Transfer Family、[AWS Transfer Family ユーザーガイド](#)を参照してください。

Note

Amazon EFS での Transfer Family の使用は、2021 AWS アカウント年 1 月 6 日より前に作成されたパブリックアクセスを許可するポリシーを持つ Amazon EFS ファイルシステムを持つでは、デフォルトで無効になっています。Transfer Family を使用してファイルシステムにアクセスできるようにするには、[お問い合わせ](#)ください サポート。

トピック

- [Amazon EFS AWS Transfer Family で 使用するための前提条件](#)
- [AWS Transfer Family向けの EFS ファイルシステムの設定](#)

Amazon EFS AWS Transfer Family で 使用するための前提条件

Transfer Family を使用して Amazon EFS ファイルシステムのファイルにアクセスするには、設定が次の条件を満たしている必要があります。

- Transfer Family サーバーと Amazon EFS ファイルシステムが、同じ AWS リージョンにあること。
- IAM ポリシーが、Transfer Family で使用される IAM ロールへのアクセスを有効にするように設定されていること。詳細については、AWS Transfer Family ユーザーガイドの「[IAM ロールとポリシーを作成する](#)」を参照してください。
- (オプション) Transfer Family サーバーが別のアカウントによって所有されている場合は、クロスアカウントアクセスが有効になります。
 - ファイルシステムポリシーでパブリックアクセスを許可していないことを確認します。詳細については、「[EFS ファイルシステムへのパブリックアクセスのブロック](#)」を参照してください。
 - クロスアカウントアクセスを有効にするようにファイルシステムポリシーを変更します。詳細については、「[Transfer Family クロスアカウントアクセスの設定](#)」を参照してください。

AWS Transfer Family向けの EFS ファイルシステムの設定

Amazon EFS ファイルシステムを Transfer Family と連携するように設定するには、以下の手順が必要です。

- ステップ 1. Transfer Family ユーザーに割り当てられている POSIX ID のリストを取得します。
- ステップ 2. Transfer Family ユーザーに割り当てられた POSIX ID を使用して、Transfer Family ユーザーがファイルシステムのディレクトリにアクセスできることを確認します。
- ステップ 3. Transfer Family で使用される IAM ロールへのアクセスを有効にするように IAM を設定します。

Transfer Family ユーザーに対するファイルおよびディレクトリへのアクセス許可の設定

Transfer Family ユーザーが、EFS ファイルシステム上の必要なファイルとディレクトリにアクセスできることを確認します。Transfer Family ユーザーに割り当てられた POSIX ID のリストを使用して、ディレクトリへのアクセス権限を割り当てます。この例では、ユーザーは Amazon EFS マウントポイントtransferFamの下に という名前のディレクトリを作成します。ディレクトリの作成はオ

プシオンで、ユースケース次第です。必要に応じて、EFS ファイルシステム上の名前と場所を選択できます。

Transfer Family の POSIX ユーザーにファイルとディレクトリのアクセス許可を割り当てるには

1. Amazon EC2 インスタンスに接続します。Amazon EFS は、Linux ベースの EC2 インスタンスによるマウントのみをサポートします。
2. EFS ファイルシステムがまだ EC2 インスタンスにマウントされていない場合は、マウントします。詳細については、「[EFS ファイルシステムをマウントする](#)」を参照してください。
3. 次の例では、EFS ファイルシステム上にディレクトリを作成し、そのグループを Transfer Family ユーザーの POSIX グループ ID に変更します。この例では 1101 です。
 - a. 次の `efs/transferFam` コマンドを使用してディレクトリを作成します。実際には、選択したファイルシステム上の名前と場所を使用できます。

```
[ec2-user@ip-192-0-2-0 ~]$ ls
efs  efs-mount-point  efs-mount-point2
[ec2-user@ip-192-0-2-0 ~]$ ls efs
[ec2-user@ip-192-0-2-0 ~]$ sudo mkdir efs/transferFam
[ec2-user@ip-192-0-2-0 ~]$ ls -l efs
total 0
drwxr-xr-x 2 root root 6 Jan  6 15:58 transferFam
```

- b. 次のコマンドを使用して、`efs/transferFam` のグループを Transfer Family ユーザーに割り当てられた POSIX GID に変更します。

```
[ec2-user@ip-192-0-2-0 ~]$ sudo chown :1101 efs/transferFam/
```

- c. 変更を確認します。

```
[ec2-user@ip-192-0-2-0 ~]$ ls -l efs
total 0
drwxr-xr-x 2 root 1101 6 Jan  6 15:58 transferFam
```

Transfer Family で使用される IAM ロールへのアクセスを有効にする

Transfer Family では、EFS ファイルシステムへのユーザーアクセスを定義するリソースベースの IAM ポリシーと IAM ロールを作成します。詳細については、AWS Transfer Family ユーザーガイドの「[IAM ロールとポリシーを作成する](#)」を参照してください。IAM アイデンティティポリシーま

またはファイルシステムポリシーを使用して、その Transfer Family IAM ロールに EFS ファイルシステムへのアクセス権を付与する必要があります。

次に示すのは、ClientMount(読み取り)とClientWriteIAM ロールへのアクセス権EFS-role-for-transferを付与するファイルシステムポリシーの例です。

```
{
  "Version": "2012-10-17",
  "Id": "efs-policy-wizard-8698b356-4212-4d30-901e-ad2030b57762",
  "Statement": [
    {
      "Sid": "Grant-transfer-role-access",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/EFS-role-for-transfer"
      },
      "Action": [
        "elasticfilesystem:ClientWrite",
        "elasticfilesystem:ClientMount"
      ]
    }
  ]
}
```

ファイルシステムの作成ポリシーの詳細については、「[ファイルシステムポリシーの作成](#)」を参照してください。EFS リソースへのアクセス権を管理するためにアイデンティティベースの IAM ポリシーを使用する方法の詳細については、「[Amazon EFS のアイデンティティベースの ポリシー](#)」を参照してください。

Transfer Family クロスアカウントアクセスの設定

ファイルシステムへのアクセスに使用される Transfer Family サーバーが別の AWS アカウントに属している場合は、AWS アカウント、そのアカウントにファイルシステムへのアクセスを許可する必要があります。また、ファイルシステムポリシーはパブリックではない必要があります。ファイルへのパブリックアクセスのブロックの詳細については、「[EFS ファイルシステムへのパブリックアクセスのブロック](#)」を参照してください。

ファイルシステムポリシーで、ファイルシステムへの別の AWS アカウント アクセスを許可できます。Amazon EFS コンソールで、ファイルシステムポリシーエディタの追加のアクセス許可を付与セクションを使用して、AWS アカウント と付与するファイルシステムアクセスのレベルを指定

します。ファイルシステムポリシーの作成または編集の詳細については、「[ファイルシステムポリシーの作成](#)」を参照してください。

アカウント ID またはアカウント Amazon リソースネーム (ARN) を使用してアカウントを指定できます。ARN の詳細については、IAM ユーザーガイドの「[IAM ARN](#)」を参照してください。

次の例は、ファイルシステムへのクロスアカウントアクセスを許可する非パブリックファイルシステムポリシーです。次の 2 つのステートメントがあります。

1. 最初のステートメント、NFS-client-read-write-via-fsmtでは、ファイルシステムマウントターゲットを使用してファイルシステムにアクセスする NFS クライアントに、読み取り、書き込み、ルート権限を付与します。
2. 2 番目のステートメントはGrant-cross-account-access、にのみ読み取りおよび書き込み権限を付与します。AWS アカウント 111122223333 は、アカウント内のこの EFS ファイルシステムにアクセスする必要がある Transfer Family サーバーを所有するアカウントです。

```
{
  "Statement": [
    {
      "Sid": "NFS-client-read-write-via-fsmt",
      "Effect": "Allow",
      "Action": [
        "elasticfilesystem:ClientRootAccess",
        "elasticfilesystem:ClientWrite",
        "elasticfilesystem:ClientMount"
      ],
      "Condition": {
        "Bool": {
          "elasticfilesystem:AccessedViaMountTarget": "true"
        }
      }
    },
    {
      "Sid": "Grant-cross-account-access",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:root"
      },
      "Action": [
        "elasticfilesystem:ClientWrite",
        "elasticfilesystem:ClientMount"
      ]
    }
  ]
}
```



```

    ]
  }
]
}

```

次のファイルシステムポリシーは、Transfer Family で使用される IAM ロールへのアクセスを許可するステートメントを追加します。

```

{
  "Statement": [
    {
      "Sid": "NFS-client-read-write-via-fsmt",
      "Effect": "Allow",
      "Action": [
        "elasticfilesystem:ClientRootAccess",
        "elasticfilesystem:ClientWrite",
        "elasticfilesystem:ClientMount"
      ],
      "Condition": {
        "Bool": {
          "elasticfilesystem:AccessedViaMountTarget": "true"
        }
      }
    },
    {
      "Sid": "Grant-cross-account-access",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:root"
      },
      "Action": [
        "elasticfilesystem:ClientWrite",
        "elasticfilesystem:ClientMount"
      ]
    },
    {
      "Sid": "Grant-transfer-role-access",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/EFS-role-for-transfer"
      },
      "Action": [
        "elasticfilesystem:ClientWrite",

```

```
    "elasticfilesystem:ClientMount"  
  ]  
}  
]
```

EFS ファイルシステムの管理

ファイルシステム管理タスクには、マウントターゲットによるファイルシステムのネットワークアクセスビリティの管理、スループットモードの変更、ライフサイクルポリシーの更新、暗号化の管理が含まれます。

これらのファイルシステム管理タスクは AWS Management Console、以下のセクションで説明するように、を使用するか、(AWS CLI) または API を使用して AWS Command Line Interface プログラムで実行できます。

トピック

- [ファイルシステムのステータスについて](#)
- [マウントターゲットの管理](#)
- [ファイルシステムのスループットの管理](#)
- [ストレージライフサイクルの管理](#)
- [暗号化されたファイルシステムへのアクセスの管理](#)

ファイルシステムのステータスについて

Amazon EFS コンソールまたは AWS CLI を使用して、Amazon EFS ファイルシステムのステータスを表示できます。Amazon EFS ファイルシステムは、次の表で説明されているステータス値のいずれかを持つことができます。

ファイルシステムの状態	説明
AVAILABLE (利用可能)	ファイルシステムは正常な状態にあり、到達可能であり、使用可能です。
CREATING (作成)	Amazon EFS は新しいファイルシステムを作成中です。
DELETING	Amazon EFS は、ユーザーが開始する削除リクエストに応答して、ファイルシステムを削除しています。詳細については、「 EFS ファイルシステムの削除 」を参照してください。

ファイルシステムの状態	説明
DELETED	Amazon EFS は、ユーザーが開始する削除リクエストに応答して、ファイルシステムを削除しました。詳細については、「 EFS ファイルシステムの削除 」を参照してください。
UPDATING	ファイルシステムは、ユーザーが開始する更新要求に応答して更新を受けています。
ERROR	レプリケーション設定内のファイルシステムを含む、1 ゾーンファイルシステムに適用されます。 ファイルシステムに障害が発生しており、回復不可能な状態です。ファイルシステムデータにアクセスするには、このファイルシステムのバックアップを新しいファイルシステムに復元します。詳細については、Amazon EFS のデータのバックアップとレプリケーションを参照してください。

マウントターゲットの管理

ファイルシステム用に作成した 1 つ以上のマウントターゲットを使用して、仮想プライベートクラウド (VPC) の Amazon EC2 または他の AWS コンピューティングインスタンスにファイルシステムをマウントします。マウントターゲットは、ファイルシステムの作成時またはファイルシステムの作成後に作成できます。

ファイルシステムのマウントターゲットを作成したら、追加のマウントターゲットの作成、マウントターゲットの削除、マウントターゲットのセキュリティグループの変更を行うことができます。マウントターゲットの VPC を変更する場合は、まず既存のマウントターゲットを削除する必要があります。

Note

既存のマウントターゲットの IP アドレスを変更することはできません。IP アドレスを変更するには、マウントターゲットを削除し、新しいアドレスで新しいターゲットを作成する必要があります。

トピック

- [マウントターゲットとアベイラビリティーゾーン](#)
- [マウントターゲットの作成](#)
- [マウントターゲットの削除](#)
- [マウントターゲット VPC の変更](#)
- [マウントターゲットセキュリティグループの変更](#)

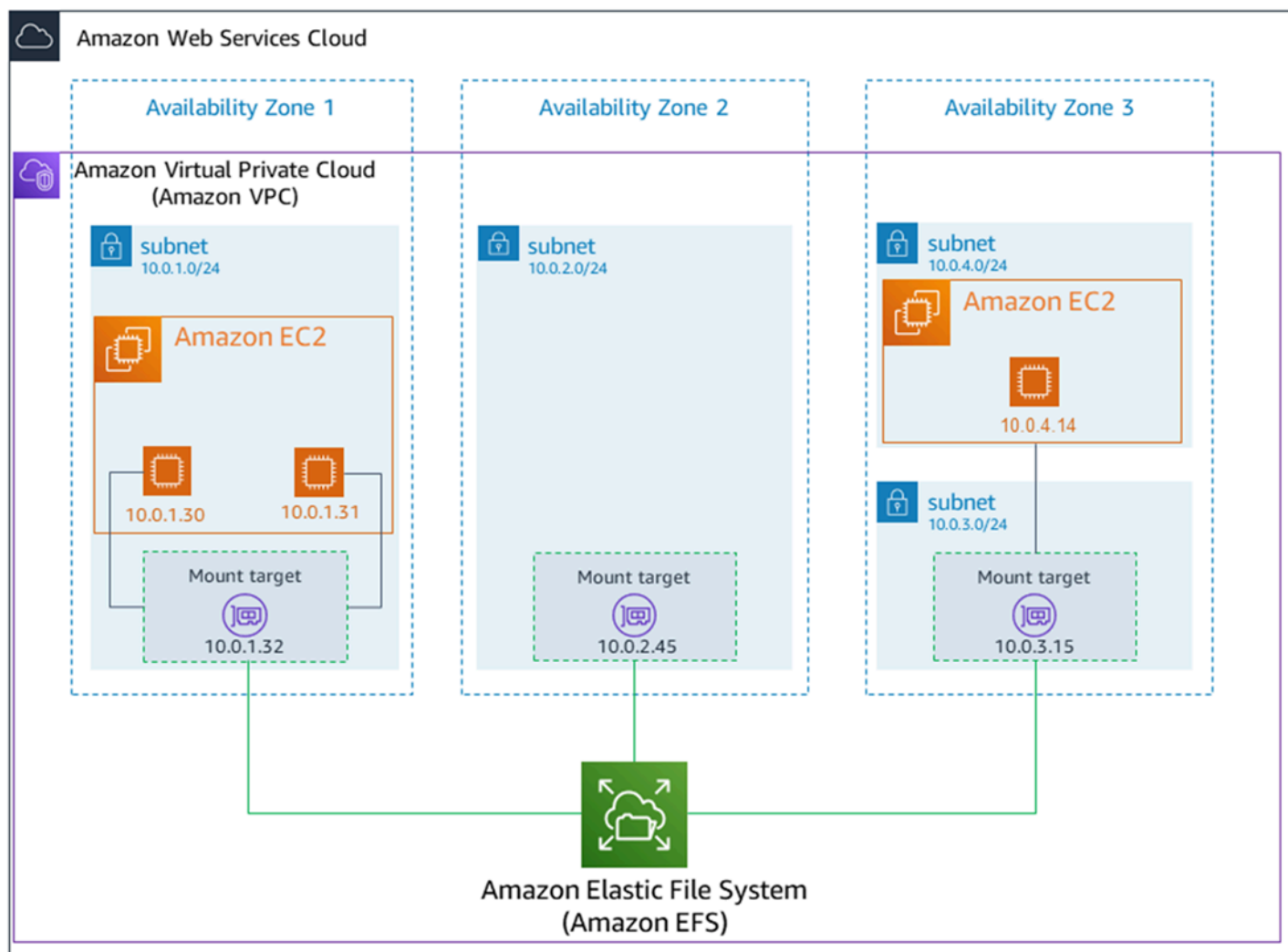
マウントターゲットとアベイラビリティーゾーン

リージョンストレージクラスを使用する EFS ファイルシステムでは、 の各アベイラビリティーゾーンにマウントターゲットを作成できます AWS リージョン。

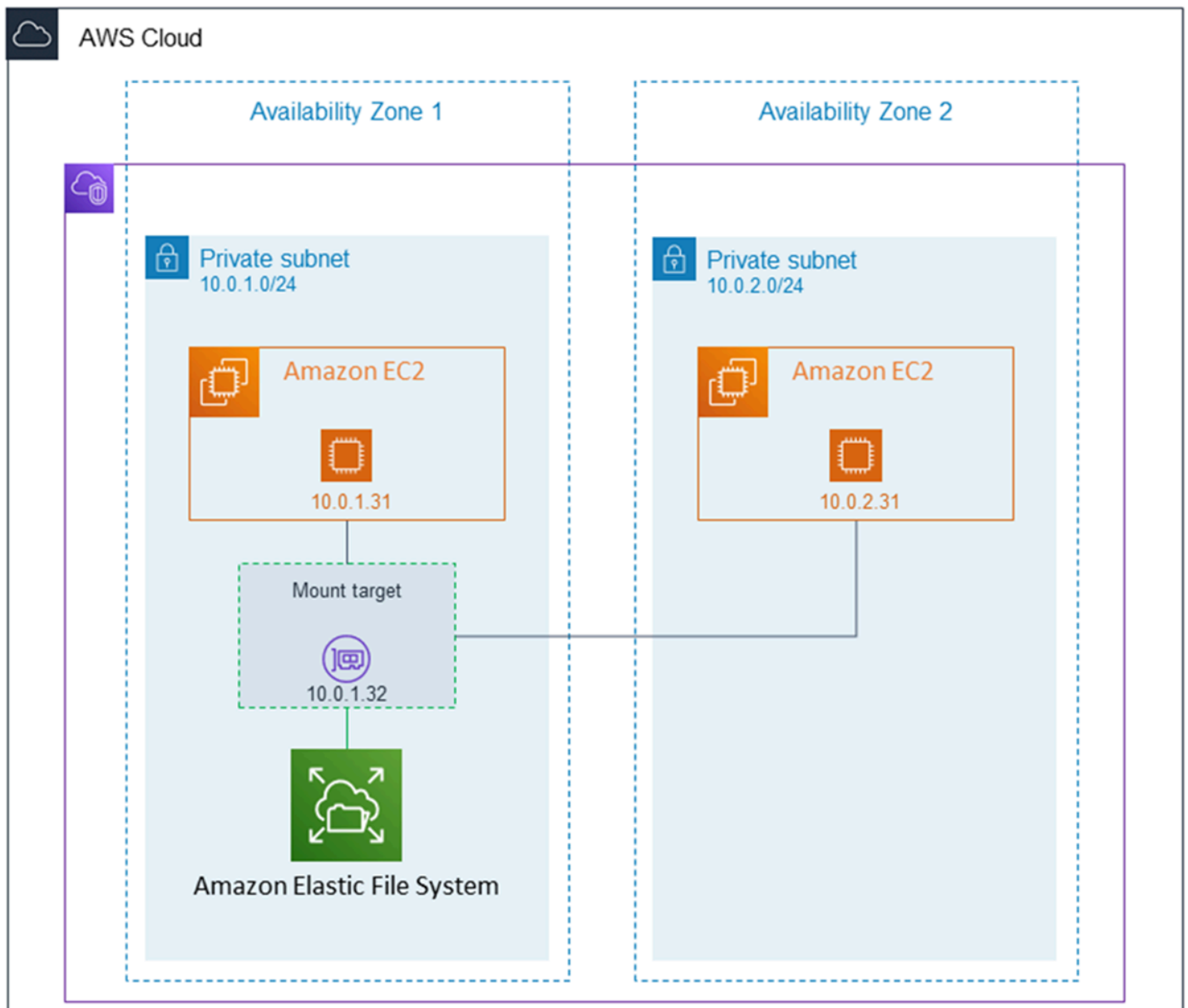
1 ゾーンファイルシステムの場合は、ファイルシステムと同じアベイラビリティーゾーンにマウントターゲットを 1 つだけ作成できます。その後、Amazon EC2、Amazon ECS、Virtual Private Cloud (VPC) などのコンピューティングインスタンス AWS Lambda にファイルシステムをマウントできます。

次の図は、VPC 内のすべてのアベイラビリティーゾーンにマウントターゲットが作成されたリージョンファイルシステムを示しています。この図は、EFS ファイルシステムにアクセスする異なる VPC サブネットで起動された 3 つの EC2 インスタンスを示しています。EFS また、図には、各アベイラビリティーゾーンに 1 つのマウントターゲットが表示されています (各アベイラビリティーゾーンのサブネットの数に関係なく)。

アベイラビリティーゾーンごとに作成できるマウントターゲットは 1 つだけです。アベイラビリティーゾーンに複数のサブネットがある場合は、図のゾーンの 1 つに示されているように、サブネットの 1 つのみにマウントターゲットを作成します。アベイラビリティーゾーンに 1 つのマウントターゲットがある限り、そのいずれかのサブネットで起動された EC2 インスタンスは同じマウントターゲットを共有できます。



次の図は、1つのマウントターゲットがファイルシステムと同じアベイラビリティーゾーンに作成された、1ゾーンファイルシステムを示しています。us-west2c アベイラビリティーゾーンの EC2 インスタンスを使用してファイルシステムにアクセスすると、マウントターゲットとは異なるアベイラビリティーゾーンにあるため、データアクセス料金が発生します。



マウントターゲットのセキュリティグループはトラフィックを制御する仮想ファイアウォールとして機能します。たとえば、どのクライアントがファイルシステムにアクセスできるかを決定します。このセクションでは次の項目について説明します。

- マウントターゲットのセキュリティグループの管理、およびトラフィックを有効化。
- クライアントにファイルシステムをマウントします。
- NFS レベルのアクセス権限を考慮します。

最初は、Amazon EC2 インスタンスのルートユーザーだけがファイルシステムに対して読み取り/書き込みの実行権限を持っています。このトピックでは、NFS レベルのアクセス権限について説

明し、一般的なシナリオでアクセス権限を付与する方法を示す例を提供しています。詳細については、「[ネットワークファイルシステム \(NFS\) レベルのユーザー、グループ、およびアクセス許可](#)」を参照してください。

マウントターゲットの作成

VPC 内の EFS ファイルシステムにアクセスするには、ファイルシステムのマウントターゲットを作成する必要があります。

EFS ファイルシステムの場合、以下が当てはまります。

- ファイルシステムのマウントターゲットは、一度に 1 つの VPC に作成できます。別の VPC からファイルシステムにアクセスする場合は、現在の VPC からマウントターゲットを削除し、他の VPC に新しいマウントターゲットを作成する必要があります。詳細については、「[マウントターゲット VPC の変更](#)」を参照してください。
- VPC のアベイラビリティーゾーンに複数のサブネットがある場合、それらのサブネットの 1 つのみにマウントターゲットを作成できます。アベイラビリティーゾーンのすべての EC2 インスタンスは、1 つのマウントターゲットを共有できます。
- 少なくとも、ファイルシステムにアクセスする各アベイラビリティーゾーンにマウントターゲットを作成する必要があります。

Note

別のアベイラビリティーゾーンで作成されたマウントターゲットを使用して、アベイラビリティーゾーンの EC2 インスタンスにファイルシステムをマウントする場合は、コストを考慮する必要があります。詳細については、「[Amazon EFS 料金表](#)」を参照してください。さらに、インスタンスのアベイラビリティーゾーンにローカルなマウントターゲットを常に使用することで、部分的な障害の発生を避けられます。マウントターゲットのゾーンに障害が発生した場合、そのマウントターゲットからファイルシステムにアクセスすることはできません。

ファイルシステムのマウントターゲットは、を使用するか AWS Management Console AWS CLI、AWS SDKs を使用してプログラムで作成できます。コンソールでは、ファイルシステムの作成時またはファイルシステムの作成後にマウントターゲットを作成できます。ファイルシステムの作成時に

マウントターゲットを作成する手順については、「」を参照してください [コンソールを使用したカスタム作成](#)。

コンソールを使用する

既存の EFS ファイルシステムにマウントターゲットを追加するには、次の手順に従います。

EFS ファイルシステムにマウントターゲットを作成するには

1. Amazon Elastic File System コンソール の <https://console.aws.amazon.com/efs/> を開いてください。
2. 左側のナビゲーションペインで、ファイルシステムを選択し、VPC を変更するファイルシステムを選択します。
3. Network を選択し、Manage を選択してファイルシステムのマウントターゲットを表示します。
4. マウントターゲットを追加するファイルシステムを選択するには、その名前またはファイルシステム ID を選択します。

Note

1 ザーンファイルシステムの場合は、ファイルシステムと同じアベイラビリティーゾーンにあるマウントターゲットを 1 つだけ作成できます。

5. EFS リージョンストレージクラスを使用するファイルシステムの場合は、ファイルシステム用に作成するマウントターゲットごとにマウントターゲットを追加を選択します。
6. マウントターゲット設定を定義します。
 - a. マウントターゲットのアベイラビリティーゾーンとサブネット ID を選択します。
 - b. IP アドレスタイプでは、IPv4 アドレスのみをサポートする IPv4、IPv6 アドレスのみをサポートする IPv6、または IPv4 アドレスと IPv6 アドレスの両方をサポートするデュアルスタックを選択します。

Note

IP アドレスタイプは、サブネットの IP タイプと一致する必要があります。さらに、IP アドレスタイプはサブネットの IP アドレス属性を上書きします。例えば、IP アドレスタイプが IPv4-only で、サブネットに対して IPv6 アドレス属性が有効になっている場合、サブネットで作成されたネットワークインターフェイスは、

サブネットの範囲から IPv4 アドレスを受け取ります。詳細については「[サブネットの IP アドレス指定属性を変更する](#)」を参照してください。

- c. マウントターゲットを配置する IP アドレスがわかっている場合は、IP アドレスタイプに一致する IP アドレスボックスにその IP アドレスを入力します。値を指定しない場合、Amazon EFS は指定されたサブネットから未使用の IP アドレスを選択します。

Note

マウントターゲットの作成後に IP アドレスを変更することはできません。IP アドレスを変更するには、マウントターゲットを削除し、新しいアドレスで新しいターゲットを作成する必要があります。

7. マウントターゲットに関連付けるセキュリティグループを少なくとも 1 つ選択します。[セキュリティグループは後で変更できます](#)。
8. [保存] を選択します。

の使用 AWS CLI

このセクションでは、create-mount-target コマンドを使用して AWS CLI でマウントターゲットを作成する例を示します。同等の API コマンドは [CreateMountTarget](#) です。

- マウントターゲットの IP アドレスタイプを指定しない場合、IPv4-only が使用されます。
- マウントターゲットの IP アドレスを指定しない場合、Amazon EFS は指定されたサブネットに使用可能なアドレスを割り当てます。
- IP アドレスタイプは、サブネットの IP アドレス属性を上書きします。例えば、IP アドレスタイプが IPv4-only で、サブネットに対して IPv6 アドレス属性が有効になっている場合、サブネットで作成されたネットワークインターフェイスは、サブネットの範囲から IPv4 アドレスを受け取ります。詳細については「[サブネットの IP アドレス指定属性を変更する](#)」を参照してください。

Note

1 ゾーンファイルシステムの場合は、ファイルシステムと同じアベイラビリティゾーンにあるマウントターゲットを 1 つだけ作成できます。

例: サブネットで使用可能な IPv4 アドレスにマウントターゲットを作成する

次のコマンドは、マウントターゲットのファイルシステム、サブネット、およびセキュリティグループを指定します。ターゲットは、指定されたサブネットで使用可能な IPv4 アドレスに作成されます。

```
$ aws efs create-mount-target \  
--file-system-id file-system-id \  
--subnet-id subnet-id \  
--security-group ID-of-the-security-group-created-for-mount-target \  
--region aws-region \  

```

次の例は、このコマンドのサデータデータを示しています。

```
$ aws efs create-mount-target \  
--file-system-id fs-0123456789abcdef1 \  
--subnet-id subnet-b3983dc4 \  
--security-group sg-01234567 \  
--region us-east-2 \  

```

次の例に示すように、マウントターゲットを正常に作成すると、Amazon EFS は JSON としてマウントターゲットの説明を返します。

```
{  
  "OwnerId": "111122223333"  
  "MountTargetId": "fsmt-f9a14450",  
  "FileSystemId": "fs-0123456789abcdef1",  
  "SubnetId": "subnet-b3983dc4",  
  "LifecycleState": "available",  
  "IpAddress": "10.0.1.24",  
  "NetworkInterfaceId": "eni-3851ec4e",  
  "AvailabilityZoneId": "use2-az1",  
  "AvailabilityZoneName": "us-east-2a",  
  "VpcId": "vpc-3c39ef57"  
}
```

例: 特定の IPv4 アドレスにマウントターゲットを作成する

次のコマンドは、マウントターゲットに使用するファイルシステム、サブネット、セキュリティグループ、および IPv4 アドレスを指定します。ターゲットは、指定されたサブネットの指定された IPv4 アドレスに作成されます。

```
$ aws efs create-mount-target \  
--file-system-id file-system-id \  
--subnet-id subnet-id \  
--security-group ID-of-the-security-group-created-for-mount-target \  
--ip-address IPv4-address \  
--region aws-region \  

```

次の例は、このコマンドのサデータデータを示しています。

```
$ aws efs create-mount-target \  
--file-system-id fs-0123456789abcdef1 \  
--subnet-id subnet-b3983dc4 \  
--security-group sg-01234567 \  
--ip-address 10.0.1.24 \  
--region us-east-2 \  

```

次の例に示すように、マウントターゲットを正常に作成すると、Amazon EFS は JSON としてマウントターゲットの説明を返します。

```
{  
  "OwnerId": "111122223333"  
  "MountTargetId": "fsm-t-f9a14450",  
  "FileSystemId": "fs-0123456789abcdef1",  
  "SubnetId": "subnet-b3983dc4",  
  "LifeCycleState": "available",  
  "IpAddress": "10.0.1.24",  
  "NetworkInterfaceId": "eni-3851ec4e",  
  "AvailabilityZoneId": "use2-az1",  
  "AvailabilityZoneName": "us-east-2a",  
  "VpcId": "vpc-3c39ef57"  
}
```

例: 特定の IPv6 アドレスにマウントターゲットを作成する

次のコマンドは、マウントターゲットに使用するファイルシステム、サブネット、セキュリティグループ、および IPv6 アドレスを指定します。ターゲットは、指定されたサブネットの指定された IPv6 アドレスに作成されます。

```
$ aws efs create-mount-target \  
--file-system-id file-system-id \  
--subnet-id subnet-id \  

```

```
--security-group ID-of-the-security-group-created-for-mount-target \  
--ip-address-type IP-address-type \  
--ipv6-address IPv6-address \  
--region aws-region \
```

次の例は、このコマンドのサデータデータを示しています。

```
$ aws efs create-mount-target \  
--file-system-id fs-0123456789abcdef1 \  
--subnet-id subnet-b3983dc4 \  
--security-group sg-01234567 \  
--ip-address-type IPV6_ONLY \  
--ipv6-address 2001:0db8:85a3:0000:0000:8a2e:0370:7334 \  
--region us-east-2 \
```

次の例に示すように、マウントターゲットを正常に作成すると、Amazon EFS は JSON としてマウントターゲットの説明を返します。

```
{  
  "OwnerId": "111122223333"  
  "MountTargetId": "fsmt-f9a14450",  
  "FileSystemId": "fs-0123456789abcdef1",  
  "SubnetId": "subnet-b3983dc4",  
  "LifecycleState": "available",  
  "Ipv6Address": "2001:0db8:85a3:0000:0000:8a2e:0370:7334",  
  "NetworkInterfaceId": "eni-3851ec4e",  
  "AvailabilityZoneId": "use2-az1",  
  "AvailabilityZoneName": "us-east-2a",  
  "VpcId": "vpc-3c39ef57"  
}
```

例: デュアルスタックサブネットで使用可能な IPv4 アドレスにマウントターゲットを作成する

コマンドは、マウントターゲットのファイルシステム、サブネット、セキュリティグループ、デュアルスタック IP アドレスタイプ、および IPv6 アドレスを指定します。ターゲットは、デュアルスタックサブネットで使用可能な IPv4 アドレスと指定された IPv6 アドレスに作成されます。

```
$ aws efs create-mount-target \  
--file-system-id file-system-id \  
--subnet-id subnet-id \  
--security-group ID-of-the-security-group-created-for-mount-target \
```

```
--ip-address-type IP-address-type
--ipv6-address IPv6-address \
--region aws-region \
```

次の例は、このコマンドのサデータデータを示しています。

```
$ aws efs create-mount-target \
--file-system-id fs-0123456789abcdef1 \
--subnet-id subnet-b3983dc4 \
--security-group sg-01234567 \
--ip-address-type DUAL_STACK \
--ipv6-address 2001:0db8:85a3:0000:0000:8a2e:0370:7334 \
--region us-east-2 \
```

次の例に示すように、マウントターゲットを正常に作成すると、Amazon EFS は JSON としてマウントターゲットの説明を返します。

```
{
  "OwnerId": "111122223333"
  "MountTargetId": "fsm-t-f9a14450",
  "FileSystemId": "fs-0123456789abcdef1",
  "SubnetId": "subnet-b3983dc4",
  "LifecycleState": "available",
  "IpAddress": "10.0.1.24",
  "Ipv6Address": "2001:0db8:85a3:0000:0000:8a2e:0370:7334",
  "NetworkInterfaceId": "eni-3851ec4e",
  "AvailabilityZoneId": "us-east-2a",
  "AvailabilityZoneName": "us-east-2a",
  "VpcId": "vpc-3c39ef57"
}
```

マウントターゲットの削除

マウントターゲットを削除すると、オペレーションによってファイルシステムのマウントが強制的に切断され、それらのマウントを使用するインスタンスまたはアプリケーションが中断される可能性があります。アプリケーションの中断を避けるには、マウントターゲットを削除する前に、アプリケーションを停止してファイルシステムをアンマウントします。詳細については、「[ファイルシステムをアンマウントする](#)」を参照してください。

ファイルシステムのマウントターゲットは AWS Management Console、 を使用するか AWS CLI、AWS SDKs を使用してプログラムで削除できます。

コンソールを使用する

既存の EFS ファイルシステムのマウントターゲットを削除するには、次の手順に従います。

EFS ファイルシステムのマウントターゲットを削除するには

1. ファイルシステムをアンマウントします。手順については、「[ファイルシステムをアンマウントする](#)」を参照してください。
2. Amazon Elastic File System コンソールの <https://console.aws.amazon.com/efs/> を開いてください。
3. 左側のナビゲーションペインで、ファイルシステムを選択し、マウントターゲットを削除するファイルシステムを選択します。
4. Network を選択し、Manage を選択してファイルシステムのマウントターゲットを表示します。
5. 削除するマウントターゲットごとに、削除を選択します。
6. [保存] を選択します。

の使用 AWS CLI

既存のマウントターゲットを削除するには、次に示すように delete-mount-target AWS CLI コマンド (対応するオペレーションは [DeleteMountTarget](#)) を使用します。

Note

マウントターゲットを削除する前に、ファイルシステムをアンマウントします。

```
$ aws efs delete-mount-target \  
--mount-target-id mount-target-ID-to-delete \  
--region aws-region-where-mount-target-exists
```

以下に、サンプルデータを含む例を示します。

```
$ aws efs delete-mount-target \  
--mount-target-id fsmt-5751852e \  
--region us-east-2 \  

```

マウントターゲット VPC の変更

Amazon VPC サービスに基づいて、一度に 1 つの VPC で EFS ファイルシステムを使用できます。つまり、ファイルシステムの VPC にマウントターゲットを作成し、それらのマウントターゲットを使用して、ファイルシステムにアクセスできます。

これらのターゲットから EFS ファイルシステムをマウントできます。

- 同じ VPC 内の Amazon EC2 インスタンス
- VPC ピアリングにより接続された VPC 内の EC2 インスタンス
- を使用したオンプレミスサーバー AWS Direct Connect
- Amazon VPC を使用した AWS 仮想プライベートネットワーク (VPN) 経由のオンプレミスサーバー

VPC ピアリング接続は、2 つの VPC 間でトラフィックのルーティングを可能にするネットワーク接続です。接続では、プライベートインターネットプロトコルバージョン 4 (IPv4) またはバージョン 6 (IPv6) を使用できます。Amazon EFS での VPC ピアリング接続の詳細については、「[別の AWS アカウント または VPC からの EFS ファイルシステムのマウント](#)」を参照してください。

ファイルシステムの VPC を変更するには

以下は、EFS ファイルシステムのネットワーク設定の VPC を変更するために実行するステップです。

1. ファイルシステムに割り当てられた各マウントターゲットを削除します。手順については、「[マウントターゲットの削除](#)」を参照してください。
2. 各マウントターゲットのマウントターゲットステータスが削除されたら、新しい VPC を割り当て、ファイルシステムの新しいマウントターゲットを作成します。手順については、「[マウントターゲットの作成](#)」を参照してください。

マウントターゲットセキュリティグループの変更

セキュリティグループによって、インバウンドおよびアウトバウンドアクセスが定義されます。マウントターゲットに関連付けられたセキュリティグループを変更する場合は、必要なインバウンド/アウトバウンドアクセスを許可するようにしてください。これにより、EC2 インスタンスはファイルシステムと通信できるようになります。セキュリティグループの詳細については、[VPC セキュリティグループの使用](#) を参照してください。

ファイルシステムのマウントターゲットのセキュリティグループを追加または削除するには AWS Management Console、AWS CLI、またはプログラムで AWS SDKs を使用します。

コンソールを使用する

マウントターゲットのセキュリティグループを変更するには

既存の EFS ファイルシステムのマウントターゲットセキュリティグループを追加または削除するには、次の手順に従います。

1. Amazon Elastic File System コンソールの <https://console.aws.amazon.com/efs/> を開いてください。
2. 左側のナビゲーションペインで、ファイルシステムを選択し、マウントターゲットを管理するファイルシステムを選択します。
3. Network を選択し、Manage を選択してファイルシステムのマウントターゲットを表示します。
4. マウント・ターゲットからセキュリティ・グループを削除するには、セキュリティグループ ID の横にある「X」を選択します。
5. セキュリティグループをマウントターゲットに追加するには、セキュリティグループリストからセキュリティを選択します。
6. [保存] を選択します。

の使用 AWS CLI

マウントターゲットに対して有効なセキュリティグループを変更するには、次に示すように、`modify-mount-target-security-group` AWS CLI コマンド (対応するオペレーションは [ModifyMountTargetSecurityGroups](#)) を使用して既存のセキュリティグループを置き換えます。

```
$ aws efs modify-mount-target-security-groups \  
--mount-target-id mount-target-ID-whose-configuration-to-update \  
--security-groups security-group-ids-separated-by-space \  
--region aws-region-where-mount-target-exists \  
--profile adminuser
```

以下に、サンプルデータを含む例を示します。

```
$ aws efs modify-mount-target-security-groups \  
--mount-target-id fsmt-5751852e \  

```

```
--security-groups sg-1004395a sg-1114433a \  
--region us-east-2
```

ファイルシステムのスループットの管理

エラスティックスループットは Amazon EFS コンソールのデフォルトのスループットモードであり、ほとんどのユースケースで推奨されます。エラスティックスループットでは、ワークロードアクティビティのニーズに合わせてパフォーマンスを自動的にスケールアップまたはスケールダウンします。ただし、ワークロードの特定のアクセスパターン (スループット、レイテンシー、ストレージのニーズなど) がわかっている場合は、スループットモードを変更することもできます。

Note

Elastic スループットはスループットに合わせて伸縮自在にスケールするように設計されていますが、運用のベストプラクティスの一環として、CloudWatch (MeteredIOBytes) と使用状況アラートを使用してメトリクスをモニタリングし、適切なガバナンスを実装することをお勧めします。これにより、最適なリソース使用率を維持し、計画された運用パラメータを維持できます。詳細については、「[Amazon CloudWatch によるメトリクスのモニタリング](#)」を参照してください。

その他に選択できるスループットモードには以下があります。

- プロビジョニングされたスループット — ファイルシステムのサイズやバーストクレジットバランスとは無関係に、ファイルシステムが処理できるスループットのレベルを指定します。
- バーストスループット — スループットはファイルシステムのストレージ容量に応じて調整され、1 日あたり最大 12 時間のより高いレベルへのバーストをサポートします。

Amazon EFS スループットモードの詳細については、「[スループットモード](#)」を参照してください。

Note

ファイルシステムが利用可能になった後は、スループットモードとプロビジョニングされたスループット量を変更できます。スループットモードを変更しても、アプリケーションのダウンタイムは発生しません。ただし、ファイルシステムをプロビジョニングされたスループットに変更したり、プロビジョニングされたスループット量を増やしたりするたびに、ス

スループットモードを再度変更したり、プロビジョニングされたスループット量を減らしたりできるようになるまで、少なくとも 24 時間待つ必要があります。

ファイルシステムのスループットモードは、Amazon EFS コンソール、AWS Command Line Interface (AWS CLI)、および Amazon EFS API を使用して管理できます。

コンソールを使用する

ファイルシステムのスループットを管理するには

1. Amazon Elastic File System コンソールの <https://console.aws.amazon.com/efs/> を開いてください。
 2. 左のナビゲーション画面で [ファイルシステム] を選択して、アカウント内の EFS ファイルシステムのリストを表示します。
 3. スループットモードを変更するファイルシステムを選択します。
 4. ファイルシステムの詳細ページで、[全般] セクションの [編集] を選択します。[編集] ページが表示されます。
 5. スループットモードの設定を変更します。
- エラスティックスループットまたはプロビジョニングされたスループットを使用するには、[拡張] を選択し、[伸縮自在] または [プロビジョニング済み] を選択します。

[プロビジョニング済み] を選択した場合は、[プロビジョニングされたスループット (MiB/秒)] に、ファイルシステム要求に対してプロビジョニングするスループットの量を入力します。[最大読み取りスループット] の量は、入力したスループットの 3 倍の量で表示されます。EFS ファイルシステムは、他のリクエストの 3 分の 1 の割合で読み取り要求を測定します。スループットを入力すると、ファイルシステムの月次コストの見積もりが表示されます。

Note

ファイルシステムが利用可能になった後は、スループットモードとプロビジョニングされたスループット量を変更できます。ただし、ファイルシステムのスループットをプロビジョンドに変更したり、プロビジョンドスループットの量を増やしたりした場合は、再びスループットモードを変更したり、プロビジョンドスループットの量を減らしたりできるようになるまで、少なくとも 24 時間待つ必要があります。

- バーストスループットを使用するには、[バースト] を選択します。

パフォーマンスのニーズに合った適切なスループットモードを選択する方法の詳細については、「[スループットモード](#)」を参照してください。

6. [変更の保存] を選択して、変更を保存します。

の使用 AWS CLI

[update-file-system](#) CLI コマンドまたは [UpdateFileSystem](#) API アクションを使用して、ファイルシステムのスループットモードを変更します。

ストレージライフサイクルの管理

ファイルシステムは、そのライフサイクルにわたってコスト効率の高いストレージを維持できます。ライフサイクル管理を使用すると、ファイルシステムのライフサイクル設定に従って、ストレージクラス間でデータを自動的に移行できます。ライフサイクル設定は、ファイルシステムに設定した 3 つのライフサイクルポリシーで構成されます。

ライフサイクルポリシーは、EFS 低頻度アクセス (IA) ストレージクラスや EFS アーカイブストレージクラスとの間でファイルを移行するタイミングをライフサイクル管理に指示します。移行時間は、標準ストレージクラスでファイルが最後にアクセスされた日時に基づいています。標準ストレージクラスでの最終アクセス日時を確認するために、内部タイマーが、公開されている POSIX ファイルシステムの属性ではなく、ファイルが最後にアクセスされた日時を追跡します。標準ストレージ内のファイルにアクセスがあると、そのたびにライフサイクル管理タイマーはリセットされます。

ライフサイクルポリシーは、EFS ファイルシステム全体に適用されます。

EFS ライフサイクルポリシーは以下のとおりです。

- [IA へ移行] - ファイルを低頻度アクセスストレージに移動するタイミングをライフサイクル管理に指示します。低頻度アクセスストレージは、四半期に数回しかアクセスされないデータ向けにコストが最適化されています。デフォルトでは、標準ストレージで 30 日間アクセスされなかったファイルは IA に移行されます。
- [アーカイブへの移行] - ファイルをアーカイブストレージクラスに移動するタイミングをライフサイクル管理に指示します。アーカイブストレージクラスは、年に数回しかアクセスされないデータ向けにコストが最適化されています。デフォルトでは、標準ストレージで 90 日間アクセスされなかったファイルはアーカイブに移行されます。

- [標準への移行] - IA ストレージまたはアーカイブストレージでファイルがアクセスされたときに、ファイルを IA またはアーカイブから標準ストレージに戻すかどうかをライフサイクル管理に指示します。デフォルトでは、ファイルは標準ストレージに戻されず、アクセス時に IA ストレージクラスまたはアーカイブストレージクラスに残ります。

最速のレイテンシーパフォーマンスを要するパフォーマンス重視のユースケース (大量の小さなファイル进行处理するアプリケーションなど) では、[初回アクセス時] にファイルを標準ストレージに移行することを選択します。

ファイルシステムのライフサイクルポリシーの設定の詳細については、「[ライフサイクルポリシーの設定](#)」を参照してください。

ライフサイクル管理のファイルシステムオペレーション

ライフサイクル管理のためのファイルシステムオペレーションは、EFS ファイルシステムのワークロードに対するオペレーションよりも優先度が低くなります。IA ストレージとアーカイブストレージとの間でファイルを移行するために必要な時間は、ファイルサイズとファイルシステムのワークロードによって異なります。例えば、数百万の小さなファイルの移行は、同じ合計ストレージサイズのより大きなファイルの移行よりも時間がかかる場合があります。

ファイル名、所有権情報、ファイルシステムのディレクトリ構造などのファイルメタデータは、常に標準のストレージに格納され、一貫したメタデータパフォーマンスの確保に役立ちます。

ディレクトリの内容を一覧表示するなど、IA またはアーカイブストレージのファイルシステムに対するメタデータオペレーションは、ファイルアクセスとしてカウントされません。ファイルの内容を IA ストレージクラスまたはアーカイブストレージクラスのいずれかに移行するプロセスの間、ファイルは標準ストレージクラスに保存され、そのストレージ料金で請求されます。

ファイルシステムの IA ストレージクラスまたはアーカイブストレージクラス内のファイルへの書き込みオペレーションはすべて、まず標準ストレージクラスに書き込まれ、24 時間後に該当するストレージクラスに移行する資格が与えられます。

ライフサイクルポリシーの設定

を使用して推奨設定を持つ EFS ファイルシステムを作成すると AWS Management Console、ファイルシステムは次のデフォルトのライフサイクル設定で自動的に設定されます。

- IA への移行は、前回のアクセスから 30 日間 に設定されます。
- [アーカイブへの移行] は、[前回のアクセスから 90 日間] に設定されています。

- [標準への移行] は [なし] に設定されています。

を使用してカスタマイズされた設定でファイルシステムを作成するとき、AWS Management Console または を使用してファイルシステムを作成するときに、デフォルトのライフサイクルポリシーを変更できます AWS CLI。または、次の手順で説明するように、ファイルシステムの作成後にポリシーを変更することもできます。

コンソールを使用する

を使用して AWS Management Console 、既存のファイルシステムのライフサイクルポリシーを設定できます。

1. にサインイン AWS Management Console し、 <https://console.aws.amazon.com/efs/> で Amazon EFS コンソールを開きます。
2. ファイルシステムを選択して、アカウント内のファイルシステムのリストを表示します。
3. ライフサイクルポリシーを変更するファイルシステムを選択します。
4. ファイルシステムの詳細ページで、[全般] セクションの [編集] を選択します。[編集] ページが表示されます。
5. [ライフサイクル管理] で、ライフサイクルポリシーを設定します。
 - [IA へ移行] を、選択できるオプションのいずれかに設定します。IAストレージへのファイルの移動を停止するには、なしを選択します。
 - [アーカイブへの移行] を、選択できるオプションのいずれかに設定します。アーカイブストレージへのファイルの移動を停止するには、[なし] を選択します。
 - IA ストレージ内のファイルがメタデータ以外のオペレーションでアクセスされたときに標準ストレージに移動されるようにするには、[標準への移行] を [初回アクセス時] に設定します。

初回アクセス時にファイルが IA ストレージまたはアーカイブストレージから標準ストレージに移動されないようにするには、[なし] を選択します。
6. [変更を保存] を選択して、変更を保存します。

の使用 AWS CLI

を使用して AWS CLI 、ファイルシステムのライフサイクルポリシーを設定または変更できます。

- [put-lifecycle-configuration](#) AWS CLI コマンドまたは [PutLifecycleConfiguration](#) API コマンドを実行し、ライフサイクル管理を管理するファイルシステムのファイルシステム ID を指定します。

```
$ aws efs put-lifecycle-configuration \  
--file-system-id File-System-ID \  
--lifecycle-policies "[{\"TransitionToIA\": \"AFTER_60_DAYS\"},  
{\"TransitionToPrimaryStorageClass\": \"AFTER_1_ACCESS\"}, {\"TransitionToArchive\":  
\"AFTER_90_DAYS\"}]" \  
--region us-west-2 \  
--profile adminuser
```

次のレスポンスが返されます。

```
{  
  "LifecyclePolicies": [  
    {  
      "TransitionToIA": "AFTER_60_DAYS"  
    },  
    {  
      "TransitionToPrimaryStorageClass": "AFTER_1_ACCESS"  
    },  
    {  
      "TransitionToArchive": "AFTER_90_DAYS"  
    }  
  ]  
}
```

既存のファイルシステムでのライフサイクル管理の停止 (CLI)

- ライフサイクル管理を停止するファイルシステムのファイルシステム ID を指定して、put-lifecycle-configuration コマンドを実行します。--lifecycle-policies プロパティを空のままにします。

```
$ aws efs put-lifecycle-configuration \  
--file-system-id File-System-ID \  
--lifecycle-policies \  
--region us-west-2 \  
--profile adminuser
```


次のレスポンスが返されます。

```
{
  "LifecyclePolicies": []
}
```

暗号化されたファイルシステムへのアクセスの管理

Amazon EFS を使用して、暗号化されたファイルシステムを作成することができます。Amazon EFS は、ファイルシステムの 2 つの暗号化形式、転送時の暗号化と保管時の暗号化をサポートします。実行する必要のあるキー管理は、保管時の暗号化にのみ関連します。Amazon EFS は、転送時の暗号化のキーを自動的に管理します。

保管時に暗号化を使用してファイルシステムを作成する場合、データとメタデータは保管時に暗号化されます。Amazon EFS はキー管理に AWS Key Management Service (AWS KMS) を使用します。保管時に暗号化を使用してファイルシステムを作成する場合、AWS KMS key を指定します。KMS キーは `aws/elasticfilesystem` (Amazon EFS AWS マネージドキー の) にすることも、管理するカスターマネージドキーにすることもできます。

ファイルの内容などのファイルデータは、ファイルシステムを作成したときに指定した KMS キーを使用して保管時に暗号化されます。メタデータ (ファイル名、ディレクトリ名、ディレクトリの内容) は、Amazon EFS が管理するキーの使用によって暗号化されます。

ファイルシステムの EFS AWS マネージドキー は、ファイル名、ディレクトリ名、ディレクトリコンテンツなど、ファイルシステムのメタデータを暗号化するための KMS キーとして使用されます。ユーザーは、保管時にファイルデータ (ファイルの内容) を暗号化するために使用するカスターマネージドキーを所有します。

KMS キーにアクセスできるユーザーと暗号化されたファイルシステムの内容を管理します。このアクセスは、AWS Identity and Access Management (IAM) ポリシーと の両方によって制御されます AWS KMS。IAM ポリシーは、Amazon EFS API アクションへのユーザーのアクセスを制御します。AWS KMS キーポリシーは、ファイルシステムの作成時に指定した KMS キーへのユーザーのアクセスを制御します。詳細については次を参照してください:

- IAM ユーザーガイド の「[IAM ユーザー](#)」
- 「AWS Key Management Service 開発者ガイド」の「[AWS KMSのキーポリシー](#)」

- 「AWS Key Management Service デベロッパーガイド」の「[AWS KMSでのグラント](#)」

キー管理者として、外部キーをインポートできます。また、キーを有効化、無効化、または削除して変更することもできます。指定した KMS キーの状態 (保管時の暗号化を使用してファイルシステムを作成した場合) は、そのコンテンツへのアクセスに影響します。KMS キーは、そのキーを使って暗号化されたファイルシステムのコンテンツにユーザーがアクセスできるように enabled の状態である必要があります。

EFS ファイルシステム用の KMS キーの管理

カスタマー管理の KMS キーを無効化や削除、KMS キーへの Amazon EFS のアクセス権の取り消しを行うことができます。キーへの Amazon EFS のアクセスを無効化および取り消しは、元に戻せるアクションです。KMS キーを削除するときは注意が必要です。KMS キーの削除は元に戻すことができないアクションです。

マウントされたファイルシステムに使用された KMS キーを無効化または削除する場合は、次の点に注意してください。

- その KMS キーは、保管時に暗号化された新しいファイルシステムのキーとして使用できません。
- その KMS キーを使用する既存の保管時に暗号化されたファイルシステムは、一定期間後に動作を停止します。

既存のマウントされたファイルシステムの許可に対する Amazon EFS アクセスを取り消すと、関連付けられた KMS キーを無効化または削除する場合と同じ動作になります。つまり、保管時に暗号化されるファイルシステムは機能し続けますが、一定期間後に機能しなくなります。

Amazon EFS へのアクセスを無効、削除、取り消した KMS キーを持つ、マウントされ保管時に暗号化されたファイルシステムへのアクセスを防ぐことができます。これを行うには、ファイルシステムをアンマウントし、Amazon EFS マウントターゲットを削除します。

をすぐに削除することはできませんが AWS KMS key、7～30 日以内に削除するようにスケジュールできます。KMS キーの削除が予定されている間は、暗号化オペレーションに KMS キーを使用することはできません。KMS キーの予定された削除をキャンセルすることもできます。

カスタマー マネージド KMS キーを無効にしてから再度有効にする方法については、AWS Key Management Service デベロッパーガイドの「[キーの有効化と無効化](#)」を参照してください。カスタマー マネージド KMS キーの削除をスケジュールする方法については、「AWS Key Management Service デベロッパーガイド」の「[KMS キーの削除](#)」を参照してください。

Amazon EFS のモニタリング

モニタリングは、Amazon EFS と AWS ソリューションの信頼性、可用性、パフォーマンスを維持する上で重要な部分です。マルチポイント障害が発生した場合は、その障害をより簡単にデバッグできるように、AWS ソリューションのすべての部分からモニタリングデータを収集することをお勧めします。ただし、Amazon EFS のモニタリングを開始する前に、以下の質問に対する回答を反映したモニタリング計画を作成します。

- モニタリングの目的は何ですか？
- どのリソースをモニタリングしますか？
- どのくらいの頻度でこれらのリソースをモニタリングしますか？
- どのモニタリングツールを利用しますか？
- 誰がモニタリングタスクを実行しますか？
- 問題が発生したときに誰が通知を受け取りますか？

次のステップでは、さまざまなタイミングと負荷条件でパフォーマンスを測定することにより、お客様の環境で通常の Amazon EFS のパフォーマンスのベースラインを確定します。Amazon EFS をモニタリングするには、モニタリングの履歴データを保存することを検討します。保存データを、最新のパフォーマンスデータと比較するベースラインとして使用し、通常のパフォーマンスのパターンやパフォーマンスの異常を検出して、問題への対応を検討することができます。

たとえば、Amazon EFS を使って、ファイルシステムについてのネットワークのスループット、読み取り、書き込み、およびメタデータオペレーションの I/O、クライアント接続、バーストのクレジットバランスなどをモニタリングできます。もし確立したベースラインからパフォーマンスが外れた場合は、ファイルシステムのサイズ、またはクライアント接続の数を変更して、ワークロードのファイルシステムを最適化する必要がある場合があります。

ベースラインを確立するには、少なくとも次の項目をモニタリングする必要があります。

- ファイルシステムのネットワークスループット。
- ファイルシステムへのクライアント接続の数。
- 読み取りデータ、書き込みデータ、メタデータオペレーションを含む、各ファイルシステムオペレーションのバイト数。

トピック

- [モニタリングツール](#)
- [Amazon EFS がファイルシステムとオブジェクトのサイズをどのように報告するかについて説明します。](#)
- [ストレージクラスサイズの表示](#)
- [Amazon CloudWatch によるメトリクスのモニタリング](#)
- [を使用した Amazon EFS API コールのログ記録 AWS CloudTrail](#)

モニタリングツール

AWS には、Amazon EFS のモニタリングに使用できるさまざまなツールが用意されています。これらのツールの一部はモニタリングを行うように設定できますが、一部のツールは手動による介入が必要です。モニタリングタスクをできるだけ自動化することをお勧めします。

自動モニタリングツール

以下の自動化されたモニタリングツールを使用して、Amazon EFS を監視し、問題が発生したときにレポートできます。

- Amazon CloudWatch アラーム - 指定した期間にわたって単一のメトリクスをモニタリングし、複数の期間にわたる特定のしきい値に対するメトリクスの値に基づいて 1 つ以上のアクションを実行します。アクションは、Amazon Simple Notification Service (Amazon SNS) のトピックまたは Amazon EC2 Auto Scaling のポリシーに送信される通知です。CloudWatch アラームは、特定の状態にあるという理由だけでアクションを呼び出すことはありません。状態が変更され、指定された期間維持される必要があります。詳細については、「[Amazon CloudWatch によるメトリクスのモニタリング](#)」を参照してください。
- Amazon CloudWatch Logs - AWS CloudTrail またはその他の出典からログファイルをモニタリング、保存、およびアクセスします。詳細については、[Amazon CloudWatch ユーザーガイド](#)の「[Amazon CloudWatch Logs とは](#)」を参照してください。Amazon CloudWatch
- Amazon CloudWatch Events - イベントに一致したものを 1 つ以上のターゲットの関数またはストリームに渡して、変更、状態の情報の収集、是正措置を行います。詳細については、「Amazon CloudWatch ユーザーガイド」の「[Amazon CloudWatch Events とは](#)」を参照してください。
- AWS CloudTrail ログモニタリング - アカウント間でログファイルを共有し、CloudWatch Logs に送信CloudWatch CloudTrail ログファイルをリアルタイムでモニタリングし、Java でログ処理アプリケーションを書き込み、CloudTrail による配信後にログファイルが変更されていないことを確認します。詳細については、AWS CloudTrail ユーザーガイドの[CloudTrail ログファイルの操作](#)を参照してください。

手動モニタリングツール

Amazon EFS のモニタリングでもう 1 つ重要な点は、Amazon CloudWatch のアラームの対象外の項目を手動でモニタリングすることです。Amazon EFS、CloudWatch、およびその他の AWS Management Console ダッシュボードには、AWS 環境の状態が at-a-glance ビューが表示されます。ファイルシステムのログファイルも確認することをお勧めします。

- Amazon EFS コンソールから、ファイルシステムについて以下の項目を見つけることができます。
 - 現在の計測サイズ
 - マウントターゲットの数
 - ライフサイクルの状態
- CloudWatch ホームページには、次の内容が表示されます。
 - 現在のアラームとステータス
 - アラームとリソースのグラフ
 - サービスのヘルスステータス

また、CloudWatch を使用して、次のことが可能です。

- [カスタマイズダッシュボード](#)を作成して、使用するサービスをモニタリングします。
- メトリクスデータをグラフ化して、問題をトラブルシューティングして、傾向を確認する。
- すべての AWS リソースメトリクスを検索して参照します。
- 問題があることを通知するアラームを作成および編集する。

Amazon EFS がファイルシステムとオブジェクトのサイズをどのように報告するかについて説明します。

以下のセクションでは、Amazon EFS が、ファイルシステムのサイズ、ファイルシステム内のオブジェクトのサイズ、ファイルシステムのスループットをどのように報告するかについて説明します。

EFS ファイルシステムオブジェクトの計測

EFS ファイルシステムで表示できるオブジェクトには、通常のファイル、ディレクトリ、シンボリックリンク、および特別なファイル (FIFOs とソケット) が含まれます。これらの各オブジェクトは、メタデータ (inode) の 2 キビバイト (KiB) と 4 KiB のデータの 1 つ以上の増分に対して計測され

ます。次の一覧は、さまざまなタイプのファイルシステムオブジェクトの測定データサイズの説明です。

- 通常のファイル - 通常のファイルの計測されたデータサイズは、次の 4 KiB 増分に丸められたファイルの論理サイズですが、スパースファイルの場合はそれより少なくなる場合があります。

スパースファイルは、論理サイズに達する前にファイルのすべての位置にデータが書き込まれないファイルです。スパースファイルについては、使用されている実際のストレージが、次の 4 KiB 増分に丸められた論理サイズよりも小さい場合があります。このような場合、Amazon EFS は計測データサイズとして使用される実際のストレージを報告します。

- ディレクトリ - ディレクトリの計測されたデータサイズは、ディレクトリエントリとそれを保持するデータ構造に使用される実際のストレージで、次の 4 KiB 増分に丸められます。計測されたデータのサイズは、ファイルデータによって使用される実際のストレージは含まれません。
- シンボリックリンクと特別なファイル - これらのオブジェクトの計測されたデータサイズは常に 4 KiB です。

NFSv4.1 `space_used` 属性を通じて Amazon EFS がオブジェクトに使用されたスペースを報告する場合、オブジェクトの現在計測されたデータサイズは含まれますが、メタデータサイズは含まれません。ファイルのディスク使用量を計測するには、`du` および `stat` ユーティリティの 2 つのユーティリティを使用できます。以下は、`-k` オプションを含む空のファイルで `du` ユーティリティを使用して、キロバイト単位で出力を返す方法の例です。

```
$ du -k file
4      file
```

以下は、空のファイルで `stat` ユーティリティを使用して、ファイルのディスク使用状況を返す方法の例です。

```
$ /usr/bin/stat --format="%b*%B" file | bc
4096
```

ディレクトリのサイズを計測するには、`stat` ユーティリティを使用します。Blocks 値を見つけ、その値にブロックサイズを乗算します。以下は、空のディレクトリで `stat` ユーティリティを使用する方法の例です。

```
$ /usr/bin/stat --format="%b*%B" . | bc
4096
```

EFS ファイルシステムの計測サイズ

EFS ファイルシステムの計測サイズには、すべての EFS ストレージクラスの現在のすべてのオブジェクトのサイズの合計が含まれます。各オブジェクトのサイズは、たとえば、午前 8 時から午前 9 時までの計測時間中のオブジェクトのサイズの代表的なサンプリングから計算されます。

空のファイルは、ファイルシステムの計測サイズに 6 KiB (2 KiB メタデータ + 4 KiB データ) を提供します。作成した時点では、ファイルシステムには、空のルートディレクトリが 1 つしかないため、計測されたサイズは、6 KiB になります。

特定のファイルシステムの計測されたサイズは、その時間にそのファイルシステムの所有者アカウントに課金される使用状況を定義します。

Note

計算されて計測されたサイズは、その時間内の特定の時刻におけるファイルシステムの一貫したスナップショットを表すものではありません。むしろ、各時間内のさまざまな時刻に、あるいは 1 時間前にファイルシステムに存在するオブジェクトのサイズを表します。これらのサイズの値は合計され、その時間のファイルシステムの計測されたサイズを決定します。これにより、ファイルシステムの計測されたサイズは、最終的に、ファイルシステムへの書き込みがないときに保存されたオブジェクトの計測されたサイズと一致します。

EFS ファイルシステムの計測サイズは、次の方法で確認できます。

- [describe-file-systems](#) AWS CLI コマンドと [DescribeFileSystem](#) API オペレーションを使用して、レスポンスには以下が含まれます。

```
"SizeInBytes":{
    "Timestamp": 1403301078,
    "Value": 29313744866,
    "ValueInIA": 675432,
    "ValueInStandard": 29312741784
    "ValueInArchive": 327650
}
```

ここで、の計測サイズValueInStandardは、[バースト](#)スループットを使用してファイルシステムの I/O スループットベースラインとバーストレートを決定するためにも使用されます。

- StorageBytes CloudWatch メトリクスを確認すると、各ストレージクラスにあるデータの計測サイズの合計が表示されます。StorageBytes メトリクスの詳細については、「[Amazon EFS の CloudWatch メトリクス](#)」を参照してください。
- EC2 インスタンスのターミナルプロンプトで、Linux の df コマンドを実行します。

レスポンスには、ファイルシステムの計測に使用されたデータがすべて反映されないため、ストレージを計測する目的でファイルシステムのルートで du コマンドを使用しないでください。

Note

ValueInStandard の計測サイズは、I/O スループットのベースラインとバーストレートを決定するためにも使用されます。詳細については、「[スループットのバースト](#)」を参照してください。

低頻度アクセスストレージクラスとアーカイブストレージクラスの計測

EFS 低頻度アクセス (IA) ストレージクラスとアーカイブストレージクラスは 4 KiB 単位で計測され、ファイルあたりの最低請求額は 128 KiB です。IA およびアーカイブファイルのメタデータ (ファイルあたり 2 KiB) は、常に標準ストレージクラスに保存され計測されます。128 KiB 未満のファイルのサポートは、2023 年 11 月 26 日午後 12:00 (太平洋時間) 以降に更新されたライフサイクルポリシーでのみ利用できます。IA ストレージおよびアーカイブストレージのデータアクセスは 128 KiB ごとに計測されます。

StorageBytes CloudWatch メトリクスを使用すると、各ストレージクラスにあるデータの計測サイズを確認できます。このメトリクスには、IA およびアーカイブストレージクラス内で小さいファイルを丸めるために消費された合計バイト数も表示されます。CloudWatch メトリクスの表示方法の詳細については、「[Amazon EFS 用の CloudWatch メトリクスへのアクセス](#)」を参照してください。StorageBytes メトリクスの詳細については、「[Amazon EFS の CloudWatch メトリクス](#)」を参照してください。

計測スループット

Amazon EFS は、他のファイルシステム I/O オペレーションの 3 分の 1 のレートで読み取りリクエストのスループットを計測します。たとえば、読み取りと書き込みスループットの両方を 1 秒あたり 30 メビバイト (MiBps) で駆動している場合、読み取り部分は有効スループットの 10 MiBps、書き込み部分は 30 MiBps、合計計測スループットは 40 MiBps となります。消費レートに対して調整

されたこの組み合わせスループットは、MeteredIOBytes CloudWatch メトリクスに反映されます。

エラスティックスループットの計測

ファイルシステムでエラスティックスループットモードが有効になっている場合は、ファイルシステムから読み書きされたメタデータとデータの量に対してのみ支払いが発生します。Elastic スループットモードを使用する EFS ファイルシステムは、メタデータの読み取りを読み取りオペレーションとして、メタデータの書き込みを書き込みオペレーションとして計測し、請求します。メタデータオペレーションは、最初の 4 KiB の後に 1 KiB 単位で計測されます。データオペレーションは、最初の 32 KiB の後に 1 KiB 単位で計測されます。

Note

Elastic スループットはスループットに合わせて伸縮自在にスケールするように設計されていますが、運用のベストプラクティスの一環として、CloudWatch (MeteredIOBytes) と使用状況アラートを使用してメトリクスをモニタリングし、適切なガバナンスを実装することをお勧めします。これにより、最適なリソース使用率を維持し、計画された運用パラメータを維持できます。詳細については、「[Amazon CloudWatch によるメトリクスのモニタリング](#)」を参照してください。

プロビジョニングされたスループットの計測

プロビジョンドスループットモードを使用するファイルシステムでは、そのスループットが有効化されている時間に対してのみ支払いが発生します。Amazon EFS は、プロビジョンドスループットモードが有効になっているファイルシステムを 1 時間に 1 回計測します。プロビジョンドスループットモードが 1 時間未満の場合の計測では、Amazon EFS はミリ秒の精度で平均時間を計算します。

ストレージクラスサイズの表示

Amazon EFS コンソール、AWS CLI または EFS API を使用して、ファイルシステムの各ストレージクラスに保存されているデータの量を表示できます。

コンソールを使用する

ファイルシステムの詳細ページの[Metered size(従量制サイズ)]のタブには、ファイルシステムの現在の従量制サイズが 2 進数のバイト (キビバイト、メビバイト、ギビバイト、およびテビバイト) で

表示されます。このメトリクスは 15 分ごとに発行され、ファイルシステムの従量制サイズを経時的に表示できます。従量制サイズは、ファイルシステムのストレージサイズに関する次の情報を表示します。

- 合計サイズは、すべてのストレージクラスを含め、ファイルシステムに保存されているデータのサイズ (バイナリバイト単位) です。
- [標準でのサイズ] は、EFS 標準 ストレージクラスに保存されているデータのサイズ (バイナリバイト単位) です。
- [IA のサイズ] は、EFS 低頻度アクセスストレージクラスに保存されているデータのサイズ (バイナリバイト単位) です。128 KiB 未満のファイルは 128 KiB に切り上げられます。
- [アーカイブ内のサイズ] は、EFS アーカイブストレージクラスに保存されているデータのサイズ (バイナリバイト単位) です。128 KiB 未満のファイルは 128 KiB に切り上げられます。

Amazon EFS コンソールのファイルシステム詳細ページの「モニタリング」タブで Storage bytes メトリクスを表示することもできます。詳細については、「[Amazon EFS 用の CloudWatch メトリクスへのアクセス](#)」を参照してください。

の使用 AWS CLI

AWS CLI または EFS API を使用して、ファイルシステムの各ストレージクラスに保存されているデータの量を表示できます。describe-file-systems CLI コマンドを呼び出してデータストレージの詳細を表示します。(対応する API オペレーションは[DescribeFileSystems](#))

```
$ aws efs describe-file-systems \
--region us-west-2 \
--profile adminuser
```

レスポンスでは、ファイルシステムの低頻度アクセスストレージクラスで最後に計測されたサイズ (バイト単位) が ValueInIA によって表示されます。標準ストレージクラスで最後に計測されたサイズ (バイト単位) は ValueInStandard によって表示されます。アーカイブストレージクラスで最後に計測されたサイズ (バイト単位) は ValueInArchive によって表示されます。3 つの値の合計は、Value に表示されるファイルシステム全体のサイズと等しくなります。

```
{
  "FileSystems":[
    {
      "OwnerId":"251839141158",
```

```
{
  "CreationToken": "MyFileSystem1",
  "FileSystemId": "fs-47a2c22e",
  "PerformanceMode": "generalPurpose",
  "CreationTime": 1403301078,
  "LifeCycleState": "created",
  "NumberOfMountTargets": 1,
  "SizeInBytes": {
    "Value": 29313746702,
    "ValueInIA": 675432,
    "ValueInStandard": 29312741784,
    "ValueInArchive": 329486
  },
  "ThroughputMode": "elastic"
}
```

ディスク使用量を表示および測定するその他の方法については、「[EFS ファイルシステムオブジェクトの計測](#)」を参照してください。

Amazon CloudWatch によるメトリクスのモニタリング

Amazon CloudWatch を使用して ファイルシステム をモニタリングすることで、Amazon EFS から raw データを収集し、リアルタイムに近い読み取り可能なメトリクスに加工することができます。これらの統計は 15 か月間記録されるため、ウェブアプリケーションやサービスの動作をよりの確に把握できます。

デフォルトでは、一部のメトリクスについて記載がない限り、Amazon EFS メトリクスデータは 1 分間隔で CloudWatch に自動的に送信されます。Amazon EFS コンソールには、Amazon CloudWatch の raw データに基づいて一連のグラフが表示されます。必要に応じて、コンソールのグラフではなく CloudWatch からファイルシステムのデータを取得することもできます。

Amazon CloudWatch の詳細については、[Amazon CloudWatch ユーザーガイド](#)の「[Amazon CloudWatch とは](#)」を参照してください。Amazon CloudWatch

Amazon EFS CloudWatch メトリクスは raw バイトとしてレポートされます。バイトは、単位の 10 進数または 2 進数の倍数に丸められません。

トピック

- [Amazon EFS の CloudWatch メトリクス](#)

- [Amazon EFS 用の CloudWatch メトリクスへのアクセス](#)
- [Amazon EFS の CloudWatch メトリクスの使用](#)
- [CloudWatch メトリクスでの Metric Math の使用](#)
- [マウント試行の成功と失敗のモニタリング](#)
- [Amazon EFS をモニタリングするための CloudWatch アラームの作成](#)

Amazon EFS の CloudWatch メトリクス

Amazon EFS のメトリクスには、EFS 名前空間が使用されます。AWS/EFS 名前空間には、次のメトリクスが含まれます。TimeSinceLastSync を除くすべてのメトリクスは、FileSystemId を対象とする単一ディメンションのメトリクスです。ファイルシステムの ID は Amazon EFS コンソールにあり、その形式は fs-abcdef0123456789a です。

TimeSinceLastSync

レプリケーション設定内のデスティネーションファイルシステムとの同期が最後に成功してから経過した時間を示します。TimeSinceLastSync 値が正常にレプリケートされる前にソースファイルシステム上のデータに加えられたすべての変更。TimeSinceLastSync の後にソース上で発生した変更は、完全にはレプリケートされない場合があります。

このメトリクスは、次の 2 つのディメンションを使用します。

- FileSystemId ディメンション - レプリケーション設定のソースファイルシステムの ID。
- DestinationFileSystemId ディメンション - レプリケーション設定のデスティネーションファイルシステムの ID。

単位: 秒

有効な統計: Minimum、Maximum、Average

PercentIOLimit

ファイルシステムが汎用パフォーマンス モードの I/O 制限にどれだけ近づいているかを示します。

単位: パーセント

有効な統計: Minimum、Maximum、Average

BurstCreditBalance

ファイルシステムのバーストクレジットの数。バーストクレジットによって、ファイルシステムは一定の期間、ファイルシステムのベースラインレベルを超えたスループットレベルまでバーストすることができます。

Minimum 統計は、期間中の 1 分間の最小のバーストクレジットバランスです。Maximum 統計は、期間中の 1 分間の最大のバーストクレジットバランスです。Average 統計は、期間中の平均のバーストクレジットバランスです。

単位: バイト

有効な統計: Minimum、Maximum、Average

PermittedThroughput

ファイルシステムがドライブできる最大のスループット容量。

- エラスティックスループットモードを使用するファイルシステムでは、この値はファイルシステムの最大書き込みスループットを表します。
- プロビジョンドスループットを使用するファイルシステムでは、EFS 標準ストレージクラスに保存されているデータ量でファイルシステムがプロビジョニング量よりも高いスループットを実現できる場合、このメトリクスは、プロビジョニング量ではなく高い方のスループットを表します。
- バーストスループットのファイルシステムの場合、この値はファイルシステムサイズと の関数ですBurstCreditBalance。

Minimum 統計は、期間中の 1 分間の最小の許可されるスループットです。Maximum 統計は、期間中の 1 分間の最大の許可されるスループットです。Average 統計は、期間中の許可される平均スループットです。

Note

読み取りオペレーションは、他のオペレーションの 3 分の 1 の割合で計測されます。

単位: バイト/秒

有効な統計: Minimum、Maximum、Average

MeteredIOBytes

データ読み取り、データ書き込み、メタデータオペレーションなど、ファイルシステムオペレーションごとに計測されたバイト数。読み取りオペレーションはスループット制限に従って割引されます。

MeteredIOBytes と PermittedThroughput を比較する [CloudWatch メトリックス数式](#) を作成できます。これらの値が等しい場合は、ファイルシステムに割り当てられたスループットの全量が消費されます。この場合、より高いスループットを得るために、ファイルシステムのスループットモードを変更することを検討します。

Sum 統計は、すべてのファイルシステムオペレーションに関連付けられる従量制バイトの総数です。Minimum 統計は、期間中の最小オペレーションのサイズです。Maximum 統計は、期間中の最大オペレーションのサイズです。Average 統計は、期間中のオペレーションの平均サイズです。SampleCount 統計は、すべてのオペレーションの数を提供します。

単位

- Minimum、Maximum、Average、Sum 統計では、バイトです。
- SampleCount では、カウントです。

有効な統計: Minimum、Maximum、Average、Sum、SampleCount

TotalIOBytes

Amazon EFS によって処理された各ファイルシステムオペレーションの実際のバイト数。読み取り割引はありません。この数値には最小値が含まれているため、アプリケーションによってリクエストされた実際の量とは異なる場合があります。この数値は、「」に示されている数値よりも大きい場合もあります PermittedThroughput。

データオペレーションは 32 KiB で計測され、その他のオペレーションは 4 KiB で計測されます。最小の後、すべてのオペレーションは KiB ごとに計測されます。

Sum 統計は、すべてのファイルシステムオペレーションに関連付けられる総バイト数です。Minimum 統計は、期間中の最小オペレーションのサイズです。Maximum 統計は、期間中の最大オペレーションのサイズです。Average 統計は、期間中のオペレーションの平均サイズです。SampleCount 統計は、すべてのオペレーションの数を提供します。

Note

その期間の 1 秒あたりの平均オペレーション数を算出するには、SampleCount 統計をその期間の秒数で割ります。その期間の平均スループット (バイト/秒) を算出するには、Sum 統計をその期間の秒数で割ります。

単位 :

- Minimum、Maximum、Average、Sum 統計では、バイトです。
- SampleCount では、カウントです。

有効な統計: Minimum、Maximum、Average、Sum、SampleCount

DataReadIOBytes

各ファイルシステムの読み取りオペレーションの実際のバイト数。

Sum 統計は読み取りオペレーションと関連付けられる総バイト数です。Minimum 統計は、期間中の最小読み取りオペレーションのサイズです。Maximum 統計は、期間中の最大読み取りオペレーションのサイズです。Average 統計は、期間中の読み取りオペレーションの平均サイズです。SampleCount 統計は読み取りオペレーションの数を提供します。

単位 :

- Minimum、Maximum、Average、および Sum のバイト。
- SampleCount のカウント

有効な統計: Minimum、Maximum、Average、Sum、SampleCount

DataWriteIOBytes

各ファイルシステムの書き込みオペレーションの実際のバイト数。

Sum 統計は書き込みオペレーションと関連付けられる総バイト数です。Minimum 統計は、期間中の最小書き込みオペレーションのサイズです。Maximum 統計は、期間中の最大書き込みオペレーションのサイズです。Average 統計は、期間中の書き込みオペレーションの平均サイズです。SampleCount 統計は書き込みオペレーションの数を提供します。

単位 :

- Minimum、Maximum、Average、Sum 統計での単位はバイトです。

- SampleCount では、カウントです。

有効な統計: Minimum、Maximum、Average、Sum、SampleCount

MetadataIOBytes

各メタデータオペレーションの実際のバイト数。

Sum 統計はメタデータオペレーションと関連付けられる総バイト数です。Minimum 統計は、期間中の最小メタデータオペレーションのサイズです。Maximum 統計は、期間中の最大メタデータオペレーションのサイズです。Average 統計は、期間中のメタデータオペレーションの平均サイズです。SampleCount 統計はメタデータオペレーションの数を提供します。

単位：

- Minimum、Maximum、Average、Sum 統計での単位はバイトです。
- SampleCount では、カウントです。

有効な統計: Minimum、Maximum、Average、Sum、SampleCount

MetadataReadIOBytes

各メタデータ読み取りオペレーションの実際のバイト数。

Sum 統計は、メタデータ読み取りオペレーションに関連付けられた総バイト数です。Minimum 統計は、期間中の最小のメタデータ読み取りオペレーションのサイズです。Maximum 統計は、期間中の最大のメタデータ読み取りオペレーションのサイズです。Average 統計は、期間中のメタデータ読み取りオペレーションの平均サイズです。SampleCount 統計は、メタデータ読み取りオペレーションの数を提供します。

単位:

- Minimum、Maximum、Average、Sum 統計での単位はバイトです。
- SampleCount では、カウントです。

有効な統計: Minimum、Maximum、Average、Sum、SampleCount

MetadataWriteIOBytes

各メタデータ書き込みオペレーションの実際のバイト数。

Sum 統計は、メタデータ書き込みオペレーションに関連付けられた総バイト数です。Minimum 統計は、期間中の最小のメタデータ書き込みオペレーションのサイズです。Maximum 統計は、期間中の最大のメタデータ書き込みオペレーションのサイズです。Average 統計は、期間中のメタ

データ書き込みオペレーションの平均サイズです。SampleCount 統計は、メタデータ書き込みオペレーションの数を提供します。

単位:

- Minimum、Maximum、Average、Sum 統計での単位はバイトです。
- SampleCount では、カウントです。

有効な統計: Minimum、Maximum、Average、Sum、SampleCount

ClientConnections

ファイルシステムへのクライアント接続の数。標準クライアントを使用する場合、マウントされた Amazon EC2 インスタンスごとに 1 つの接続があります。

Note

1 分を超える期間の平均 ClientConnections を算出するには、Sum 統計をその期間の分数で割ります。

単位: クライアント接続数

有効な統計: Sum

StorageBytes

EFS ストレージクラスに保存されているデータの量を含む、ファイルシステムのサイズ (バイト単位)。このメトリクスは 15 分ごとに CloudWatch に発行されます。

StorageBytes メトリクスには、次のディメンションがあります。

- Total は、すべてのストレージクラスにおいて、ファイルシステムに保存されているデータの計測サイズ (バイト単位) です。EFS 低頻度アクセス (IA) および EFS アーカイブストレージクラスについては、128 KiB 未満のファイルは 128 KiB に丸められます。
- Standard は、EFS 標準ストレージクラスに保存されている計測サイズ (バイト単位) です。
- IA は、EFS 低頻度アクセスストレージクラスに保存されているデータの実際の計測サイズ (バイト単位) です。
- IASizeOverhead は、EFS 低頻度アクセスストレージクラス内のデータの実際のサイズ (IA ディメンションで示される値) と、ストレージクラスの計測サイズの差 (バイト単位) です。小さいファイルは 128 KiB に丸められます。

- Archive は、EFS アーカイブストレージクラスに保存されている実際の計測サイズ (バイト単位) です。
- ArchiveSizeOverhead は、EFS アーカイブアクセスストレージクラス内のデータの実際のサイズ (Archive デイメンションで示される値) と、ストレージクラスの計測サイズの差 (バイト単位) です。小さいファイルは 128 KiB に丸められます。

単位: バイト

有効な統計: Minimum、Maximum、Average

 Note

StorageBytes は、Amazon EFS コンソールのファイルシステムメトリクスページに、ベース 1024 単位 (キビバイト、メビバイト、ジビバイト、テビバイト) で表示されます。

Amazon EFS 用の CloudWatch メトリクスへのアクセス

CloudWatch の Amazon EFS メトリクスは、いくつかの方法で表示できます。

- Amazon EFS コンソールで
- CloudWatch コンソールで
- CloudWatch CLI の使用
- CloudWatch API の使用

CloudWatch メトリクスとアラームを表示するには (Amazon EFS コンソール)

1. にサインイン AWS Management Console し、<https://console.aws.amazon.com/efs/> で Amazon EFS コンソールを開きます。
2. [File Systems (ファイルシステム)] を選択します。
3. CloudWatch メトリクスを表示するファイルシステムを選択します。
4. [モニタリング]を選択すると、[ファイルシステムのメトリックス]ページが表示されます。

[ファイルシステムのメトリックス]ページには、ファイルシステムの CloudWatch メトリクスのデフォルトセットが表示されます。設定した CloudWatch アラームも、これらのメトリクスとともに表示されます。最大 I/O パフォーマンスモードを使用するファイルシステムの場合、デフォルト

トのメトリクスセットには、パーセント IO制限の代わりにバーストクレジットバランスが含まれます。デフォルト設定は、設定を開くと表示される [メトリクス設定] ダイアログボックスを使用して上書きできます。

 Note

スループット使用率 (%) メトリクスは CloudWatch メトリックスではなく、CloudWatch メトリクスの計算を使用して導出されます。

5. メトリクスやアラームの表示方法は、[ファイルシステムメトリクス] ページのコントロールを使って以下のように調整できます。
 - 時系列または単一値の間で表示モードを切り替えます。
 - ファイルシステムに設定された CloudWatch アラームを表示または非表示にします。
 - [CloudWatch で詳しく見る] を選択して、CloudWatch でメトリクスを表示します。
 - [ダッシュボードに追加] を選択して CloudWatch ダッシュボードを開き、表示されたメトリクスを追加します。
 - 表示されるメトリクス時間枠を 1 時間から 1 週間に調整します。

CloudWatch メトリクスとアラームを表示するには (CloudWatch コンソール)

1. CloudWatch コンソールの <https://console.aws.amazon.com/cloudwatch/> を開いてください。
2. ナビゲーションペインで [Metrics (メトリクス)] を選択してください。
3. [EFS] 名前空間を選択します。
4. (オプション) メトリクスを表示するには、検索フィールドにその名前を入力します。
5. (オプション) デイメンション別にフィルタリングするには、[FileSystemId] を選択します。

からメトリクスにアクセスするには AWS CLI

- `--namespace "AWS/EFS"` 名前空間で [list-metrics](#) コマンドを使用します。詳細については、「[AWS CLI コマンドリファレンス](#)」を参照してください。

CloudWatch API からメトリクスにアクセスするには

- [GetMetricStatistics](#) を呼び出します。詳細については、「[Amazon CloudWatch API リファレンス](#)」を参照してください。

Amazon EFS の CloudWatch メトリクスの使用

Amazon EFS からレポートされるメトリクスには、さまざまな方法で分析できる情報が含まれています。以下のリストは、メトリクスの一般的な利用方法をいくつか示しています。ここで紹介するのは開始するための提案事項です。すべてを網羅しているわけではありません。

どうすればよいのか？	関連するメトリクス
自分のスループットを決定するにはどうすればよいですか？	毎日の TotalIOBytes メトリクスの Sum 統計をモニタリングして、スループットを確認できます。
転送されたデータの合計量を確認するにはどうすればよいですか？	の日次Sum統計をモニタリングMeteredIOBytes して、転送されたデータの合計を確認できます。
ファイルシステムに接続されている Amazon EC2 インスタンスの数を追跡するにはどうすればよいですか？	ClientConnections メトリクスの Sum 統計をモニタリングできます。1 分を超える期間の平均 ClientConnections を算出するには、合計をその期間の分数で割ります。
私のバーストクレジットバランスを確認するにはどうすればよいですか？	ファイルシステムの BurstCreditBalance メトリクスをモニタリングすることによってバランスを確認できます。バーストとバーストクレジットの詳細については、「 スループットのバースト 」を参照してください。

スループットパフォーマンスのモニタリング

スループットモニタリングのための CloudWatch メトリクス

(TotalIOBytes、ReadIOBytes、WriteIOBytes、MetadataIOBytes) は、ファイルシステムが動作している実際のスループットを表します。メトリクス MeteredIOBytes は、駆動している全体の従量制スループットの計算を表します。Amazon EFS コンソールのモニタリングセクションの「スループット使用率 (%)」グラフを使って、スループット使用率をモニタリングすることができます。カスタムの CloudWatch ダッシュボードや他のモニタリングツールを使用すると、MeteredIOBytes と PermittedThroughput を比較する [CloudWatch メトリクスの数式](#) を作成することができます。

PermittedThroughput はファイルシステムの許容スループットの量を測定します。この値は、以下のいずれかの方法に基づいています。

- エラスティックスループットモードのファイルシステムでは、この値はファイルシステムの最大書き込みスループットを表します。
- プロビジョンドスループットを使用するファイルシステムでは、EFS 標準ストレージクラスに保存されているデータ量でファイルシステムがプロビジョニング量よりも高いスループットを実現できる場合、このメトリクスは、プロビジョニング量ではなく高い方のスループットを表します。
- バーストスループットを使用するファイルシステムでは、この値はファイルシステムサイズと BurstCreditBalance の相関になります。BurstCreditBalance をモニタリングして、ファイルシステムがベースレートではなくバーストレートで動作していることを確認します。バランスが一貫してゼロまたはゼロに近い場合は、追加のスループットを得るために、エラスティックスループットまたはプロビジョンドスループットに切り替えることを検討します。

MeteredIOBytes と PermittedThroughput の値が等しい場合、ファイルシステムが利用可能なすべてのスループットを消費しています。プロビジョンドスループットを使用するファイルシステムでは、追加のスループットをプロビジョニングできます。

CloudWatch メトリクスでの Metric Math の使用

Metric Math を使用すると、複数の Amazon CloudWatch メトリクスをクエリし、数式を使用して、それらのメトリクスに基づく新しい時系列を作成できます。作成された時系列は CloudWatch コンソールで視覚化でき、ダッシュボードに追加できます。たとえば、Amazon EFS メトリクスを使用して、60 で割った DataRead オペレーションのサンプル数を取得できます。結果は、指定された 1 分間の 1 秒あたりのファイルシステムにおける読み取りの平均数です。メトリクス数学の詳細については、「Amazon [CloudWatch ユーザーガイド](#)」の「[CloudWatch メトリクスで数式を使用する](#)」を参照してください。 Amazon CloudWatch

以下は、Amazon EFS で役立ついくつかのメトリクス数式です。

トピック

- [Metric Math: スループット \(MiBps\)](#)
- [Metric Math: パーセントスループット](#)
- [Metric Math: 許容スループット使用割合 \(%\)](#)
- [Metric Math: スループット IOPS](#)
- [Metric Math: IOPS の割合](#)

- [Metric Math: 平均 I/O サイズ \(KiB 単位\)](#)
- [Amazon EFS の AWS CloudFormation テンプレートで Metric Math を使用する](#)

Metric Math: スループット (MiBps)

ある期間の平均スループット (MiBps単位) を算出するには、最初に合計の統計 (DataReadIOBytes、DataWriteIOBytes、MetadataIOBytes、または TotalIOBytes) を選択します。次に、値を MiB に変換して、その期間の秒数で割ります。

たとえば、このようなロジックがあるとします: (TotalIOBytes の合計 ÷ 1048576 (MiB に変換するため)) ÷ その期間の秒数

この場合、CloudWatch メトリクス情報は次のようになります。

ID	使用可能なメトリクス	統計	間隔
m1	• DataReadIOBytes • DataWriteIOBytes • MetadataIOBytes • TotalIOBytes	sum	1 分

メトリクス数学 ID と表現は次のとおりです。

ID	表現
e1	(m1/1048576)/PERIOD(m1)

Metric Math: パーセントスループット

このメトリック数式は、異なる I/O タイプに使用される全体的なスループットの割合 (たとえば、読み取り要求によって駆動される合計スループットの割合) を計算します。I/O タイプ (DataReadIOBytes、DataWriteIOBytes、または MetadataIOBytes) のいずれかが一定期間に

使用した全体的なスループットの割合を計算するには、まず、それぞれの合計の統計に 100 を掛けます。次に、その結果を同じ期間の TotalIOBytes の合計の統計で割ります。

たとえば、このようなロジックがあるとします: (DataReadIOBytes の合計 x 100 (パーセントに変換するため)) ÷ TotalIOBytes の合計

この場合、CloudWatch メトリクス情報は次のようになります。

ID	使用可能なメトリクス	統計	間隔
m1	• TotalIOBytes	sum	1 分
m2	• DataReadIOBytes	sum	1 分

メトリクス数学 ID と表現は次のとおりです。

ID	表現
e1	$(m2 * 100) / m1$

Metric Math: 許容スループット使用割合 (%)

ある期間の許容スループット使用割合 (%) (MeteredIOBytes) を計算するには、スループット (MiBps 単位) を 100 で掛けます。次に、その結果を同じ期間の PermittedThroughput の平均統計 (MiB に換算したもの) で割ります。

例えば、このようなロジックがあるとします: (Metric Math 式、スループット (MiBps 単位) x 100 (百分率に換算するため)) ÷ (PermittedThroughput の合計 ÷ 1,048,576 (バイトを MiB に換算するため))

この場合、CloudWatch メトリクス情報は次のようになります。

ID	使用可能なメトリクス	統計	間隔
m1	MeteredIOBytes	sum	1 分

ID	使用可能なメトリクス	統計	間隔
m2	Permitted Throughput	average	1 分

メトリクス数学 ID と表現は次のとおりです。

ID	表現
e1	$(m1/1048576) / \text{PERIOD } (m1)$
e2	$m2/1048576$
e3	$((e1)*100)/(e2)$

Metric Math: スループット IOPS

ある期間の 1 秒あたりの平均オペレーション数 (IOPS) を算出するには、サンプル数の統計 (DataReadIOBytes、DataWriteIOBytes、MetadataIOBytes または TotalIOBytes) をその期間の秒数で割ります。

たとえば、このようなロジックがあるとします: DataWriteIOBytes のサンプル数 ÷ その期間の秒数

この場合、CloudWatch メトリクス情報は次のようになります。

ID	使用可能なメトリクス	統計	間隔
m1	- DataReadIOBytes - DataWriteIOBytes - MetadataIOBytes	サンプル数	1 分

ID	使用可能なメトリクス	統計	間隔
	TotalIOBytes		

メトリクス数学 ID と表現は次のとおりです。

ID	表現
e1	m1/PERIOD(m1)

Metric Math: IOPS の割合

ある期間のさまざまな I/O タイプ (DataReadIOBytes、DataWriteIOBytes、または MetadataIOBytes) の 1 秒あたりの IOPS の割合を計算するには、まず、それぞれのサンプル数の統計に 100 を掛けます。次に、その値を同じ期間の TotalIOBytes のサンプル数の統計で割ります。

たとえば、このようなロジックがあるとします: (MetadataIOBytes のサンプル数 x 100 (パーセントに変換するため)) ÷ TotalIOBytes のサンプル数

この場合、CloudWatch メトリクス情報は次のようになります。

ID	使用可能なメトリクス	統計	間隔
m1	TotalIOBytes	サンプル数	1 分
m2	- DataReadIOBytes - DataWriteIOBytes - MetadataIOBytes	サンプル数	1 分

メトリクス数学 ID と表現は次のとおりです。

ID	表現
e1	$(m2*100)/m1$

Metric Math: 平均 I/O サイズ (KiB 単位)

ある期間の平均 I/O サイズ (KiB 単位) を計算するに

は、DataReadIOBytes、DataWriteIOBytes、または MetadataIOBytes メトリクスのそれぞれの合計の統計を、そのメトリクスの同じサンプル数の統計で割ります。

たとえば、このようなロジックがあるとします: (DataReadIOBytes の合計 ÷ 1,024 (KiB に変換するため)) ÷ DataReadIOBytes のサンプル数

この場合、CloudWatch メトリクス情報は次のようになります。

ID	使用可能なメトリクス	統計	間隔
m1	- DataReadIOBytes - DataWriteIOBytes - MetadataIOBytes	sum	1 分
m2	- DataReadIOBytes - DataWriteIOBytes - MetadataIOBytes	サンプル数	1 分

メトリクス数学 ID と表現は次のとおりです。

ID	表現
e1	(m1/1024)/m2

Amazon EFS の AWS CloudFormation テンプレートで Metric Math を使用する

AWS CloudFormation テンプレートを使用してメトリクス数式を作成することもできます。そのようなテンプレートの1つは、GitHub にある「[Amazon EFS チュートリアル](#)」からダウンロードしてカスタマイズして使用することができます。AWS CloudFormation テンプレートの使用の詳細については、「ユーザーガイド」の[AWS CloudFormation 「テンプレートの使用」](#)を参照してください。

AWS CloudFormation

マウント試行の成功と失敗のモニタリング

Amazon CloudWatch Logs を使用すると、クライアントにログインすることなく EFS ファイルシステムのマウント試行の成功または失敗のモニタリングとレポートを作成できます。以下の手順を使用して、CloudWatch Logs を使用してファイルシステムのマウント試行の成功または失敗をモニタリングするように EC2 インスタンスを設定します。

CloudWatch ログでマウント試行の成功または失敗の通知を有効にするには

1. ファイルシステムをマウントしている EC2 インスタンスに amazon-efs-utils をインストールします。詳細については、[を使用して Amazon EFS クライアントを自動的にインストールまたは更新する AWS Systems Manager](#)または[Amazon EFS クライアントの手動インストール](#)を参照してください。
2. ファイルシステムをマウントしようとしている EC2 インスタンスに boto-core をインストールします。詳細については、「[boto-core のインストールとアップグレード](#)」を参照してください。
3. amazon-efs-utils で CloudWatch Logs 機能を有効にします。AWS Systems Manager を使用して をインストールおよび設定すると amazon-efs-utils、CloudWatch ログ記録が自動的に行われます。amazon-efs-utils パッケージを手動でインストールする場合、cloudwatch-log セクションの # enabled = true 行のコメントを解除して、/etc/amazon/efs/efs-utils.conf 設定ファイルを手動で更新する必要があります。CloudWatch Logs を手動で有効にするには、以下のコマンドのいずれかを使用します。

Linux インスタンスの場合:

```
sudo sed -i -e '/\[cloudwatch-log\]/{N;s/# enabled = true/enabled = true/}' /etc/  
amazon/efs/efs-utils.conf
```

MacOS インスタンスの場合:

```
EFS_UTILS_VERSION= efs-utils-version  
sudo sed -i -e '/\[cloudwatch-log\]/{N;s/# enabled = true/enabled = true/;}' /usr/  
local/Cellar/amazon-efs-utils/${EFS_UTILS_VERSION}/libexec/etc/amazon/efs/efs-  
utils.conf
```

Mac2 インスタンスの場合:

```
EFS_UTILS_VERSION= efs-utils-version  
sudo sed -i -e '/\[cloudwatch-log\]/{N;s/# enabled = true/enabled = true/;}' /opt/  
homebrew/Cellar/amazon-efs-utils/${EFS_UTILS_VERSION}/libexec/etc/amazon/efs/efs-  
utils.conf
```

- オプションで、CloudWatch Logs のグループ名を設定し、efs-utils.conf ファイルにログの保持日数を設定できます。CloudWatch にマウントされたファイルシステムごとに個別のロググループを作成する場合は、efs-utils.conf ファイル内 log_group_name のフィールドの末尾に /{fs_id} を次のように追加します。

```
[cloudwatch-log]  
log_group_name = /aws/efs/utils/{fs_id}
```

- AmazonElasticFileSystemsUtils AWS マネージドポリシーを EC2 インスタンスにアタッチした IAM ロール、またはインスタンスで設定された AWS 認証情報にアタッチします。Systems Manager を使用してこれを行うことができます。詳細については、「[ステップ 1: 必要なアクセス許可を持つ IAM インスタンスプロファイルを設定する](#)」を参照してください。

マウント試行ステータスのログエントリの例を次に示します。

```
Successfully mounted fs-12345678.efs.us-east-1.amazonaws.com at /home/ec2-user/efs  
Mount failed, Failed to resolve "fs-01234567.efs.us-east-1.amazonaws.com"
```

CloudWatch Logs でマウントステータスを表示するには

- CloudWatch コンソールの <https://console.aws.amazon.com/cloudwatch/> を開いてください。

2. 左側のナビゲーションバーで、[ロググループ] を選択します。
3. [/aws/efs/utils] ロググループを選択します。Amazon EC2 インスタンスと EFS ファイルシステムの組み合わせごとにログストリームが表示されます。
4. ログストリームを選択して、マウントの成功または失敗のステータスを含む特定のログイベントを表示します。

Amazon EFS をモニタリングするための CloudWatch アラームの作成

アラームの状態が変わったら、Amazon SNS メッセージを送信する Amazon CloudWatch のアラームを作成することができます。1つのアラームで、指定した期間中、1つのメトリクスを監視します。その後アラームは、指定された複数の期間にわたるしきい値とメトリクスの値の関係性に基づき、1つ以上のアクションを実行します。アクションは、Amazon SNS のトピックまたは自動スケーリングのポリシーに送信される通知です。

アラームは、持続した状態の変化に対してのみアクションを呼び出します。CloudWatch アラームは、特定の状態にあるという理由だけでアクションを呼び出すことはありません。状態が変更され、指定された期間維持される必要があります。

Amazon EFS での CloudWatch アラームの重要な用途は、ファイルシステムに保管時の暗号化を適用することです。Amazon EFS ファイルシステムの作成時に、保管時のデータの暗号化を有効にすることができます。Amazon EFS ファイルシステムに保管時のデータ暗号化ポリシーを適用するには、Amazon CloudWatch および AWS CloudTrail を使用してファイルシステムの作成を検出し、保管時の暗号化が有効になっていることを確認します。

Note

現時点では、転送時に暗号化を強制することはできません。

次の手順は、Amazon EFS のアラームを作成する方法について説明しています。

コンソールを使用する

CloudWatch コンソールを使用してアラームを設定するには

1. にサインイン AWS Management Console し、<https://console.aws.amazon.com/cloudwatch/> で CloudWatch コンソールを開きます。
2. [Create Alarm] を選択します。これにより、[Create Alarm Wizard] が起動します。

3. [EFS Metrics] を選択し、Amazon EFS メトリクスをスクロールして、アラームを設定するメトリクスを見つけます。このダイアログボックスに Amazon EFS メトリクスのみを表示するには、ファイルシステムのファイルシステム ID で検索します。アラームを作成するメトリクスを選択し、[Next] (次へ) をクリックします。
4. [名前]、[説明]、[次の時] のそれぞれにメトリクスの値を入力します。
5. アラーム状態に達したときに CloudWatch からメールを送信する場合は、[Whenever this alarm: (このアラームが次の時 :)] フィールドで [State is ALARM (状態はアラーム)] を選択します。[通知の送信先:] フィールドで、既存の SNS トピックを選択します。[トピックの作成] を選択すると、新しいメールサブスクリプションリストの名前とメールアドレスを設定できます。このリストは保存され、今後のアラーム用のフィールドに表示されます。

 Note

[トピックの作成] を使用して新しい Amazon SNS トピックを作成する場合、メールアドレスを検証しなければ、そのアドレスで通知を受け取ることができません。メールは、アラームがアラーム状態になったときにのみ送信されます。アラーム状態になった際に、メールアドレスの検証がまだ完了していない場合は、そのアドレスで通知を受け取ることができません。

6. この時点で、[アラームのプレビュー] エリアで、作成するアラームを確認できます。[Create Alarm] (アラームの作成) を選択します。

の使用 AWS CLI

を使用してアラームを設定するには AWS CLI

- [put-metric-alarm](#) を呼び出します。詳細については、「[AWS CLI コマンドリファレンス](#)」を参照してください。

CloudWatch API の使用

CloudWatch API を使用してアラームを設定するには

- [PutMetricAlarm](#) を呼び出します。詳細については、「[Amazon CloudWatch API リファレンス](#)」を参照してください。

を使用した Amazon EFS API コールのログ記録 AWS CloudTrail

Amazon EFS は AWS CloudTrail、Amazon EFS のユーザー、ロール、または のサービスによって実行されたアクションを記録する AWS サービスである EFS と統合されています。CloudTrail は、Amazon EFS コンソールからの呼び出しと Amazon EFS API オペレーションへのコード呼び出しを含む、Amazon EFS のすべての API コールをイベントとしてキャプチャします。

証跡を作成する場合は、Amazon EFS のイベントなど、Amazon S3 バケットへの CloudTrail イベントの継続的な配信を有効にすることができます。証跡を設定しない場合でも、CloudTrail コンソールの [イベント履歴] で最新のイベントを表示できます。CloudTrail で収集された情報を使用して、Amazon EFS に対するリクエスト、リクエスト元の IP アドレス、リクエスト者、リクエスト日時などの詳細を確認できます。

詳細については、「AWS CloudTrail ユーザーガイド」の [「What is AWS CloudTrail ?」](#) を参照してください。

CloudTrail 内の Amazon EFS 情報

CloudTrail は、アカウントの作成 AWS アカウント 時に 有効になります。Amazon EFS でアクティビティが発生すると、そのアクティビティはイベント履歴の他の AWS サービスイベントとともに CloudTrail イベントに記録されます。で最近のイベントを表示、検索、ダウンロードできます AWS アカウント。詳細については、[CloudTrail イベント履歴の使用](#) を参照してください。

Amazon EFS のイベントなど AWS アカウント、 のイベントの継続的な記録については、証跡を作成します。証跡により、CloudTrail はログファイルを 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、証跡はすべての AWS リージョンに適用されます。証跡は、AWS パーティション AWS リージョン 内のすべての 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 ログで収集されたイベントデータをさらに分析して処理するように他の AWS サービスを設定できます。詳細については、『AWS CloudTrail ユーザーガイド:』の以下のトピックを参照してください。

- [AWS アカウントの証跡の作成](#)
- [AWS CloudTrail ログとの サービス統合](#)
- [「CloudTrail の Amazon SNS 通知の設定」](#)
- [CloudTrail ログファイルを複数のリージョンから受け取る、複数のアカウントから CloudTrail ログファイルを受け取る](#)

すべての Amazon EFS [Amazon EFS API](#) は CloudTrail によってログに記録されます。例えば、CreateFileSystem、CreateMountTarget、および CreateTags オペレーションへの呼び出しによって 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。アイデンティティ情報は、以下を判別するのに役立ちます。

- リクエストがルートユーザーまたは AWS Identity and Access Management (IAM) ユーザー認証情報を使用して行われたかどうか。
- リクエストがロールまたはフェデレーションユーザーのテンポラリなセキュリティ認証情報を使用して行われたかどうか。
- リクエストが別の AWS サービスによって行われたかどうか。

詳細については、「AWS CloudTrail ユーザーガイド」の「[CloudTrail userIdentity エlement](#)」を参照してください。

Amazon EFS ログファイルエントリの概要

「トレイル」は、指定した Simple Storage Service (Amazon S3) バケットにイベントをログファイルとして配信するように設定できます。CloudTrail のログファイルは、単一か複数のログエントリを含みます。イベントは、任意の出典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。

次の例は、ファイルシステムのタグがコンソールから作成されたときの CreateTags オペレーションを示す CloudTrail ログエントリを示しています。

```
{
  "eventVersion": "1.06",
  "userIdentity": {
    "type": "Root",
    "principalId": "111122223333",
    "arn": "arn:aws:iam::111122223333:root",
    "accountId": "111122223333",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "attributes": {
        "mfaAuthenticated": "false",
```

```

    "creationDate": "2017-03-01T18:02:37Z"
  }
},
"eventTime": "2017-03-01T19:25:47Z",
"eventSource": "elasticfilesystem.amazonaws.com",
"eventName": "CreateTags",
"awsRegion": "us-west-2",
"sourceIPAddress": "192.0.2.0",
"userAgent": "console.amazonaws.com",
"requestParameters": {
  "fileSystemId": "fs-00112233",
  "tags": [{
    "key": "TagName",
    "value": "AnotherNewTag"
  }]
},
"responseElements": null,
"requestID": "dEXAMPLE-feb4-11e6-85f0-736EXAMPLE75",
"eventID": "eEXAMPLE-2d32-4619-bd00-657EXAMPLEe4",
"eventType": "AwsApiCall",
"apiVersion": "2015-02-01",
"recipientAccountId": "11112223333"
}

```

次の例は、ファイルシステムのタグがコンソールから削除されたときの DeleteTags アクションを示す CloudTrail ログエントリを示しています。

```

{
  "eventVersion": "1.06",
  "userIdentity": {
    "type": "Root",
    "principalId": "11112223333",
    "arn": "arn:aws:iam::11112223333:root",
    "accountId": "11112223333",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2017-03-01T18:02:37Z"
      }
    }
  }
}

```



```

    }
  },
  "eventTime": "2017-03-01T19:25:47Z",
  "eventSource": "elasticfilesystem.amazonaws.com",
  "eventName": "DeleteTags",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "console.amazonaws.com",
  "requestParameters": {
    "fileSystemId": "fs-00112233",
    "tagKeys": []
  },
  "responseElements": null,
  "requestID": "dEXAMPLE-feb4-11e6-85f0-736EXAMPLE75",
  "eventID": "eEXAMPLE-2d32-4619-bd00-657EXAMPLEe4",
  "eventType": "AwsApiCall",
  "apiVersion": "2015-02-01",
  "recipientAccountId": "111122223333"
}

```

EFS サービスにリンクされたロールのログエントリ

Amazon EFS サービスにリンクされたロールは、AWS リソースに対して API コールを行います。EFS サービスにリンクされたロールによって行われたコールには、`username: AWSServiceRoleForAmazonElasticFileSystem` を含む CloudTrail ログエントリが表示されます。EFS およびサービスにリンクされたロールの詳細については、「[Amazon EFS のサービスリンクロールの使用](#)」を参照してください。

次の例は、Amazon EFS が `AWSServiceRoleForAmazonElasticFileSystem` サービスにリンクされたロールを作成したときの `CreateServiceLinkedRole` アクションを示す CloudTrail ログエントリです。

```

{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "111122223333",
    "arn": "arn:aws:iam::111122223333:user/user1",
    "accountId": "111122223333",
    "accessKeyId": "A111122223333",
    "userName": "user1",

```

```

    "sessionContext": {
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2019-10-23T22:45:41Z"
      }
    },
    "invokedBy": "elasticfilesystem.amazonaws.com"
  },
  "eventTime": "2019-10-23T22:45:41Z",
  "eventSource": "iam.amazonaws.com",
  "eventName": "CreateServiceLinkedRole",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "user_agent",
  "requestParameters": {
    "awsserviceName": "elasticfilesystem.amazonaws.com"
  },
  "responseElements": {
    "role": {
      "assumeRolePolicyDocument":
"111122223333-10-111122223333Statement111122223333Action111122223333AssumeRole111122223333Effect%22%3A%20%22Allow%22%2C%20%22Principal%22%3A%20%7B%22Service%22%3A%20%5B%22elasticfilesystem.amazonaws.com%22%5D%7D%7D%5D%7D",
      "arn": "arn:aws:iam::111122223333:role/aws-service-role/elasticfilesystem.amazonaws.com/AWSServiceRoleForAmazonElasticFileSystem",
      "roleId": "111122223333",
      "createDate": "Oct 23, 2019 10:45:41 PM",
      "roleName": "AWSServiceRoleForAmazonElasticFileSystem",
      "path": "/aws-service-role/elasticfilesystem.amazonaws.com/"
    }
  },
  "requestID": "11111111-2222-3333-4444-abcdef123456",
  "eventID": "11111111-2222-3333-4444-abcdef123456",
  "eventType": "AwsApiCall",
  "recipientAccountId": "111122223333"
}

```

次の例は、に記載されている、AWSServiceRoleForAmazonElasticFileSystemサービスにリンクされたロールによって行われたCreateNetworkInterfaceアクションを示す CloudTrail ログエントリを示していますsessionContext。

```

{
  "eventVersion": "1.05",

```

```

    "userIdentity": {
      "type": "AssumedRole",
      "principalId": "AIDACKCEVSQ6C2EXAMPLE",
      "arn": "arn:aws:sts::0123456789ab:assumed-role/
AWSServiceRoleForAmazonElasticFileSystem/0123456789ab",
      "accountId": "0123456789ab",
      "sessionContext": {
        "sessionIssuer": {
          "type": "Role",
          "principalId": "AIDACKCEVSQ6C2EXAMPLE",
          "arn": "arn:aws:iam::0123456789ab:role/aws-service-role/
elasticfilesystem.amazonaws.com/AWSServiceRoleForAmazonElasticFileSystem",
          "accountId": "0123456789ab",
          "userName": "AWSServiceRoleForAmazonElasticFileSystem"
        },
        "webIdFederationData": {},
        "attributes": {
          "mfaAuthenticated": "false",
          "creationDate": "2019-10-23T22:50:05Z"
        }
      },
      "invokedBy": "AWS Internal"
    },
    "eventTime": "2019-10-23T22:50:05Z",
    "eventSource": "ec2.amazonaws.com",
    "eventName": "CreateNetworkInterface",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "elasticfilesystem.amazonaws.com",
    "userAgent": "elasticfilesystem.amazonaws.com",
    "requestParameters": {
      "subnetId": "subnet-71e2f83a",
      "description": "EFS mount target for fs-1234567 (fsmt-1234567)",
      "groupSet": {},
      "privateIpAddressesSet": {}
    },
    "responseElements": {
      "requestId": "0708e4ad-03f6-4802-b4ce-4ba987d94b8d",
      "networkInterface": {
        "networkInterfaceId": "eni-0123456789abcdef0",
        "subnetId": "subnet-12345678",
        "vpcId": "vpc-01234567",
        "availabilityZone": "us-east-1b",
        "description": "EFS mount target for fs-1234567 (fsmt-1234567)",
        "ownerId": "666051418590",

```

```
    "requesterId": "0123456789ab",
    "requesterManaged": true,
    "status": "pending",
    "macAddress": "00:bb:ee:ff:aa:cc",
    "privateIpAddress": "192.0.2.0",
    "privateDnsName": "ip-192-0-2-0.ec2.internal",
    "sourceDestCheck": true,
    "groupSet": {
      "items": [
        {
          "groupId": "sg-c16d65b6",
          "groupName": "default"
        }
      ]
    },
    "privateIpAddressesSet": {
      "item": [
        {
          "privateIpAddress": "192.0.2.0",
          "primary": true
        }
      ]
    },
    "tagSet": {}
  },
  "requestID": "11112222-3333-4444-5555-666666777777",
  "eventID": "aaaabbbb-1111-2222-3333-444444555555",
  "eventType": "AwsApiCall",
  "recipientAccountId": "111122223333"
}
```

EFS 認証のログエントリ

Amazon EFS で NFS クライアントが認可されると、CloudTrail イベントとして NewClientConnection と UpdateClientConnection が発行されます。NewClientConnection イベントは、最初の接続の直後、および再接続の直後に接続が認可されたときに発行されます。UpdateClientConnection は接続が再承認され、許可されるアクションのリストが変更された場合に発行されます。このイベントは、許可された新しいアクションの ClientMount がリストに含まれていない場合にも発生します。EFS 認可の詳細については、「[IAM を使用してファイルシステムへのアクセスを制御する](#)」を参照してください。

次は、NewClientConnection イベントを示す CloudTrail ログエントリの例です。

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn": "arn:aws:sts::0123456789ab:assumed-role/abcdef0123456789",
    "accountId": "0123456789ab",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE ",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::0123456789ab:role/us-east-2",
        "accountId": "0123456789ab",
        "userName": "username"
      },
      "webIdFederationData": {},
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2019-12-23T17:50:16Z"
      },
      "ec2RoleDelivery": "1.0"
    }
  },
  "eventTime": "2019-12-23T18:02:12Z",
  "eventSource": "elasticfilesystem.amazonaws.com",
  "eventName": "NewClientConnection",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "AWS Internal",
  "userAgent": "elasticfilesystem",
  "requestParameters": null,
  "responseElements": null,
  "eventID": "27859ac9-053c-4112-aee3-f3429719d460",
  "readOnly": true,
  "resources": [
    {
      "accountId": "0123456789ab",
      "type": "AWS::EFS::FileSystem",
      "ARN": "arn:aws:elasticfilesystem:us-east-2:0123456789ab:file-system/
fs-01234567"
    },
    {

```

```
        "accountId": "0123456789ab",
        "type": "AWS::EFS::AccessPoint",
        "ARN": "arn:aws:elasticfilesystem:us-east-2:0123456789ab:access-point/
fsap-0123456789abcdef0"
    },
    ],
    "eventType": "AwsServiceEvent",
    "recipientAccountId": "0123456789ab",
    "serviceEventDetails": {
        "permissions": {
            "ClientRootAccess": true,
            "ClientMount": true,
            "ClientWrite": true
        },
        "sourceIpAddress": "10.7.3.72"
    }
}
```

encrypted-at-rest ファイルシステムの Amazon EFS ログファイルエントリ

Amazon EFS では、保管時の暗号化、転送中の暗号化のいずれかまたは両方をファイルシステムに使用できます。詳細については、「[Amazon EFS でのデータの暗号化](#)」を参照してください。

Amazon EFS は、データキーを生成して Amazon EFS データを復号するための AWS KMS API リクエストを行うときに、[暗号化コンテキスト](#)を送信します。ファイルシステム ID は、保管時に暗号化されるすべてのファイルシステムの暗号化コンテキストです。CloudTrail ログエントリの {75} フィールドでは、暗号化コンテキストは次のようになります。

```
"EncryptionContextEquals": {}
"aws:elasticfilesystem:filesystem:id" : "fs-4EXAMPLE"
```

AWS 請求レポートと使用状況レポートの使用

Amazon EFS を使用する場合、使用したリソースに対してのみ料金が発生し、最低料金やセットアップ料金は発生しません。他の AWS サービスと同様に、料金は月単位で見積もられますが、使用量は時間単位で按分され、1 か月の平均使用量に対して請求されます。

AWS は、Amazon EFS の次のレポートを提供します。

- 請求レポート – Amazon EFS. AWS は常に EFS ファイルシステムの所有者に Amazon EFS EFS 料金を請求するなど、使用している AWS サービスのすべてのアクティビティの概要を示す複数のレポート。請求レポートの詳細については、「[AWS Billing Amazon EFS の レポート](#)」を参照してください。
- 使用状況レポート – 特定のサービスのアクティビティの概要。時間、日、または月別に集計されます。含める使用タイプ、オペレーションを選択できます。また、データの収集方法も選択できます。詳細については、「[AWS Amazon EFS の 使用状況レポート](#)」を参照してください。

以下のトピックでは、Amazon EFS の請求と使用状況レポートについて説明します。

トピック

- [コスト配分 EFS ファイルシステムタグの使用](#)
- [AWS Billing Amazon EFS の レポート](#)
- [AWS Amazon EFS の 使用状況レポート](#)
- [Amazon EFS の請求と使用状況レポートについて](#)

コスト配分 EFS ファイルシステムタグの使用

個々のプロジェクトまたはプロジェクトのグループのストレージコストやその他の基準を追跡するには、コスト配分タグを使用して EFS ファイルシステムにラベルを付けます。コスト配分タグは、ファイルシステムに関連付けるキーと値のペアです。コスト配分タグを有効にすると、はタグ AWS を使用してコスト配分レポートのリソースコストを整理します。

コスト配分レポートには、製品カテゴリとリンクされたアカウントユーザー別にアカウントの AWS 使用状況が一覧表示されます。レポートには、同じ明細項目が詳細な請求レポートとして含まれ（「[Amazon EFS の請求と使用状況レポートについて](#)」を参照）、タグキーは追加列に含まれます。

AWS には、AWSが生成したタグとユーザー定義のタグの 2 種類のコスト配分タグが用意されています。は、Amazon EFS CreateFileSystem イベントの後に、AWS生成されたaws:createdByタグキー AWS を定義、作成、適用します。ユーザー定義タグを定義し、作成して、ファイルシステムに適用します。

請求レポートに表示するには、請求およびコスト管理コンソールで両方の種類のタグを個別に有効化する必要があります。AWS生成されたタグの詳細については、「AWS Billing ユーザーガイド」の「[コスト配分タグを使用した AWS コストの整理と追跡](#)」を参照してください。

- EFS リソースのユーザー定義タグの作成の詳細については、「」を参照してください[EFS リソースのタグ付け](#)。
- Billing and Cost Management コンソールでユーザー定義タグをアクティブ化する方法の詳細については、「AWS Billing ユーザーガイド」の「[ユーザー定義のコスト配分タグのアクティブ化](#)」を参照してください。
- Billing and Cost Management コンソールで AWS生成されたタグをアクティブ化する方法の詳細については、「AWS Billing ユーザーガイド」の [AWS「で生成されたコスト配分タグのアクティブ化](#)」を参照してください。

ユーザー定義のコスト配分タグ

ユーザー定義コスト配分タグには以下のコンポーネントがあります。

- タグキー。タグキーは、タグ名です。たとえば、project/Trinity タグでは project がキーです。タグキーは大文字小文字を区別する文字列で、1 ~ 128 個の Unicode 文字を含めることができます。
- タグ値。タグ値は必須文字列です。たとえば、project/Trinity タグでは Trinity が値です。タグ値は大文字小文字を区別する文字列で、0 ~ 256 個の Unicode 文字を含めることができます。

ユーザー定義タグの詳細については、「AWS Billing ユーザーガイド」の「[ユーザー定義のコスト配分タグの使用](#)」を参照してください。

AWS Billing Amazon EFS の レポート

の月別請求では、使用状況情報とコストを AWS サービスおよび機能ごとに AWS 分離します。月別 AWS レポート、コスト配分レポート、請求明細レポートなど、複数の請求レポートを利用できます。詳細については、「AWS Billingユーザーガイド」の「[請求書](#)について」を参照してください。

Amazon EFS ストレージの使用状況に関する詳細なレポートについては、AWS Data Exports を使用して AWS Cost and Usage Report (AWS CUR) のエクスポートを作成します。AWS CUR 2.0 では、所有する Amazon Simple Storage Service (Amazon S3) バケットに AWS 請求レポートを発行します。Amazon EFS 使用状況レポートには、使用タイプと別のオペレーションが一覧表示されます AWS リージョン。詳細については、「AWS Data Exports ユーザーガイド」の [「コストと使用状況レポート \(CUR\) 2.0」](#) を参照してください。

EFS ファイルシステムは、次のカテゴリの使用に基づいて請求されます。

- ストレージ (ギガバイトあたり [GB]) – EFS ファイルシステムに保存されるデータの量。ストレージクラス (EFS 標準、EFS 低頻度アクセス (IA)、EFS アーカイブ) によって料金が異なります。
- スループット (GB あたりまたは 1 秒あたりのメビバイト数 [MiBps]) – ファイルシステムのスループットに応じて、転送されるデータの量 (読み取り/書き込みオペレーション) またはプロビジョニングされた量を超える 1 か月あたりの量。
- データアクセスと移行 (MiBps あたり) – EFS 低頻度アクセス (IA) および EFS Archive ストレージクラスに読み取りおよび転送されるデータの量。
- バックアップストレージ (GB あたり) – バックアップによって消費される 1 か月あたりのストレージ容量。バックアップ保持期間を延長したり、ユーザー主導のボリュームバックアップを追加したりすると、ファイルシステムが消費するバックアップストレージの量が増えます。詳細については、[EFS ファイルシステムのバックアップ](#) を参照してください。
- レプリケーション (GB あたり) – ソースおよび宛先 EFS ファイルシステムの一般的なストレージおよび読み取り、書き込み、階層化アクティビティレート、およびそれら間のデータ転送アクティビティに適用される 1 か月あたりの料金。

Amazon EFS 料金の詳細については、[「Amazon EFS 料金表」](#) を参照してください。

Amazon EFS の請求および使用状況レポートで使用されるコードと略語を理解する方法については、「」を参照してください [Amazon EFS の請求と使用状況レポートについて](#)。

AWS Amazon EFS の 使用状況レポート

使用状況レポートをダウンロードする際、使用状況を時間、日、または月単位で集計できます。Amazon EFS 使用状況レポートには、使用状況タイプと別のオペレーションが一覧表示されます AWS リージョン。Amazon EFS ストレージの使用状況に関するより詳細なレポートについては、動的に生成された AWS 使用状況レポートをダウンロードしてください。含める使用タイプ、オペレーション、期間を選択できます。また、データの収集方法も選択できます。使用状況レポートのダ

ダウンロードの詳細については、「[データエクスポートユーザーガイド](#)」の AWS 「[使用状況レポートのダウンロード](#)」を参照してください。AWS

Amazon EFS 使用状況レポートには、次の情報が含まれています。

- サービス – Amazon EFS
- 使用状況 – 以下の値のいずれかです。
 - ストレージのタイプを識別するコード
 - リクエストのタイプを識別するコード
 - データ転送のタイプを識別するコード
 - スループットモードを識別するコード
 - バックアップの使用状況を識別するコード

Amazon EFS の使用タイプの詳細については、「」を参照してください[Amazon EFS の請求と使用状況レポートについて](#)。

- リソース – リストされた使用状況に関連付けられたリソースの名前。
- StartTime – 使用状況が適用される協定世界時 (UTC) による日付の開始時間。
- EndTime – 使用状況が適用がされる協定世界時 (UTC) による日付の終了時間。
- UsageValue – 以下のボリューム値のいずれかです。データの典型的な測定単位はギガバイト (GB) です。ただし、サービスとレポートによっては、テラバイト (TB) が代わりに表示されることがあります。
 - 指定した期間のリクエストの数
 - データ転送量
 - 指定された時間内に格納されたデータの量

Amazon EFS の請求および使用状況レポートで使用されるコードと略語を理解する方法については、「」を参照してください[Amazon EFS の請求と使用状況レポートについて](#)。

Amazon EFS の請求と使用状況レポートについて

Amazon EFS の請求および使用状況レポートは、コードと略語を使用します。次の表の使用タイプについては、をこのリストの略語 *region* に置き換えてください。

- APE1: アジアパシフィック (香港)
- APN1: アジアパシフィック (東京)

- APN2: アジアパシフィック (ソウル)
- APN3: アジアパシフィック (大阪)
- APS1: アジアパシフィック (シンガポール)
- APS2: アジアパシフィック (シドニー)
- APS3: アジアパシフィック (ムンバイ)
- APS4: アジアパシフィック (ジャカルタ)
- APS5: アジアパシフィック (ハイデラバード)
- APS6: アジアパシフィック (メルボルン)
- CAN1: カナダ (中部)
- CAN2: カナダ西部 (カルガリー)
- CNN1: 中国 (北京)
- CNW1: 中国 (寧夏)
- AFS1: アフリカ (ケープタウン)
- EUC2: 欧州 (チューリッヒ)
- EUN1: 欧州 (ストックホルム)
- EUS2: 欧州 (スペイン)
- EUC1: 欧州 (フランクフルト)
- EU: 欧州 (アイルランド)
- EUS1: 欧州 (ミラノ)
- EUW2: 欧州 (ロンドン)
- EUW3: 欧州 (パリ)
- ILC1: イスラエル (テルアビブ)
- MEC1: 中東 (アラブ首長国連邦)
- MES1: 中東 (バーレーン)
- SAE1: 南米 (サンパウロ)
- UGW1: AWS GovCloud (米国西部)
- UGE1: AWS GovCloud (米国東部)
- USE1 (またはプレフィックスなし): 米国東部 (バージニア北部)
- USE2: 米国東部 (オハイオ)
- USW1: 米国西部 (北カリフォルニア)

- USW2: 米国西部 (オレゴン)

料金の詳細については AWS リージョン、[「Amazon EFS 料金表」](#)を参照してください。

次の表の 1 列目は、請求および使用状況レポートに記載される使用タイプです。データの典型的な測定単位はギガバイト (GB) です。ただし、プロビジョニングされたスループット量の場合、代わりに 1 秒あたりのメビバイト数 (MiBps) が使用されます。

Amazon EFS の使用タイプ

使用タイプ	[CloudWatch メトリクス]	単位	詳細度	説明
<i>region</i> -ArchiveDataAccess-Bytes	該当なし	GB	該当なし	EFS Archive ストレージに転送またはアクセスされたデータの量。
<i>region</i> -ArchiveEarlyDelete-ByteHrs	該当なし	GB-Mo	該当なし	90 日間の最小コミットメントが終了する前に EFS Archive ストレージから削除されたファイルの按分されたストレージ使用量。
<i>region</i> -ArchiveEarlyDelete-SmallFiles	該当なし	GB-Mo	該当なし	90 日間の最小コミットメントが終了する前に EFS Archive ストレージから削除された小さなファイル (128 KB 未満) の按分計算されたストレージ使用量。
<i>region</i> -ArchiveTimedStorage-ByteHrs	StorageEstimates.ArchivedStorage	GB-Mo	毎時	EFS Archive ストレージにデータが保存された GB-月の数。
<i>region</i> -ArchiveTimedStorage	StorageEstimates.ArchivedStorage	GB-Mo	毎時	小さなオブジェクト (128 KB 未満) が EFS Archive ストレージに保存された GB-月の数。

使用タイプ	[CloudWatch メトリクス]	単位	詳細度	説明
ge-SmallFiles	veSizeOverhead			
<i>region</i> -ETDataAccess-Bytes	TotalIOBytes	GB	毎時	Elastic スループットを使用してファイルの書き込みまたは読み取りが行われたデータの量。
<i>region</i> -IADataAccess-Bytes	該当なし	GB	毎時	EFS 低頻度アクセス (IA) ストレージでアクセスされたデータの量。
<i>region</i> -IATimedStorage-ByteHrs	StorageElasticIA	GB-Mo	毎時	EFS IA ストレージにデータが保存された GB-月の数。
<i>region</i> -IATimedStorage-ET-ByteHrs	StorageElasticIA	GB-Mo	毎時	Elastic スループットを使用するファイルのデータが EFS IA ストレージに保存された GB-月の数。
<i>region</i> -IATimedStorage-ET-SmallFiles	StorageElasticIASizeOverhead	GB-Mo	毎時	Elastic スループットを使用するファイルについて、小さなファイル (128 KB 未満) が EFS IA ストレージに保存された GB-月の数。
<i>region</i> -IATimedStorage-SmallFiles	StorageElasticIASizeOverhead	GB-Mo	毎時	小さなファイル (128 KB 未満) が EFS IA ストレージに保存された GB-月の数。

使用タイプ	[CloudWatch メトリクス]	単位	詳細度	説明
<i>region</i> -IATimedStorage-Z-ByteHrs	StorageE tes.IA	GB-Mo	毎時	1 ゾーンファイルシステムの EFS IA ストレージに保存されているデータの実際のサイズ。
<i>region</i> -IATimedStorage-Z-SmallFiles	StorageE tes.IASizeOverhe	GB-Mo	毎時	1 ゾーンファイルシステムの EFS IA ストレージに保存された小さなファイル (128 KB 未満) の GB-月の数。
<i>region</i> -IncludedTP-MiBpsHrs	PermittedThrough t	MiBps-Mo	毎時	ファイルシステムがドライブできる最大のスループット容量。
<i>region</i> -ProvisionedTP-MiBpsHrs	PermittedThrough t	MiBps-Mo	毎時	プロビジョンドスループットを使用するファイルシステムの場合、プロビジョンドスループットまたは最大許容スループット。
<i>region</i> -TimedStorage-ByteHrs	StorageE tes.Standard	GB-Mo	毎時	リージョンファイルシステムの EFS Standard ストレージに保存されているデータの実際の合計量。
<i>region</i> -TimedStorage-Z-ByteHrs	StorageE tes.Standard	GB-Mo	毎時	1 ゾーンファイルシステムの EFS 標準ストレージに保存されているデータの実際の合計量。

メモ

- EFS Archive ストレージは、データが 90 日より前に削除された場合でも、最小ストレージ期間である 90 日間課金されます。
- 階層化料金は、EFS 標準ストレージから EFS EFS 低頻度アクセス (IA) ストレージ、および EFS IA ストレージから EFS 標準ストレージに階層化されたデータに適用されます。
- EFS IA および EFS Archive ストレージの最小請求可能ファイルサイズは 128 KiB です。128 KiB 未満のファイルは、これらのストレージクラスに階層化できますが、適切なストレージクラスレートで 128 KiB のストレージに対して課金されます。
- Elastic スループットを使用するファイルシステムの場合、メタデータオペレーションは最初の 4 KiB の後に 1 KiB 単位で計測され、データオペレーションは最初の 32 KiB の後に 1 KiB 単位で計測されます。
- ストレージ使用量はバイナリギガバイト (GB) で計算されます。1 GB は 2^{30} バイトです。Amazon EFS 請求書を計算するためのストレージとスループットの使用量の計測方法[Amazon EFS がファイルシステムとオブジェクトのサイズをどのように報告するかについて説明します。](#)の詳細については、「」を参照してください。
- GB-Month は、GB-hours の合計数を 1 か月で集計し、その月の時間数で割って算出されます。

使用状況レポートでオペレーションを追跡する

オペレーションは、EFS ファイルシステムで実行されたアクションを、指定された使用タイプ別に記述します。オペレーションは、Read や Delete のような、見ればすぐわかるコードによって示されます。ファイルシステムで特定のタイプの使用状況を生成したアクションを確認するには、これらのコードを使用します。使用状況レポートを作成する場合、レポートに [All Operations] (すべてのオペレーション) を含めるか、Write など特定のオペレーションのみを含めるかを選択できます。

Amazon EFS のパフォーマンス仕様

以下のセクションでは、Amazon EFS パフォーマンスの概要と、ファイルシステムの設定が主要なパフォーマンスディメンションにどのように影響するかをご紹介します。また、ファイルシステムのパフォーマンスを最適化するための重要なヒントや推奨事項もいくつか紹介します。

トピック

- [パフォーマンスの概要](#)
- [ストレージクラス](#)
- [パフォーマンスモード](#)
- [スループットモード](#)
- [Amazon EFS パフォーマンスのヒント](#)
- [Amazon EFS のパフォーマンス問題のトラブルシューティング](#)
- [AMI とカーネルの問題のトラブルシューティング](#)

パフォーマンスの概要

ファイルシステムのパフォーマンスは、通常、レイテンシー、スループット、1 秒あたりの入出力オペレーション (IOPS) のディメンションを使用して測定されます。これらの次元にわたる Amazon EFS のパフォーマンスは、ファイルシステムの設定によって異なります。次の設定は、Amazon EFS ファイルシステムのパフォーマンスに影響します。

- ファイルシステムのタイプ — リージョンまたは 1 ザーン
- パフォーマンスモード — 汎用モードまたは最大 I/O

Important

最大 I/O パフォーマンスモードは、汎用パフォーマンスモードよりもオペレーションごとのレイテンシーが高くなります。パフォーマンスを向上させるには、常に汎用パフォーマンスモードを使用することをお勧めします。詳細については、「[パフォーマンスモード](#)」を参照してください。

- スループットモード — エラスティック、プロビジョニングされた、バースティング

次の表は、汎用パフォーマンスモードを使用するファイルシステムのパフォーマンス仕様と、システムタイプとスループットモードから考えられるさまざまな組み合わせの概要を示しています。

汎用パフォーマンスモードを使用するファイルシステムのパフォーマンス仕様

ストレージとスループット設定		レイテンシー		最大 IOPS		最大スループット		
ファイルシステムのタイプ	スループットモード	読み込みオペレーション	書き込みオペレーション	読み込みオペレーション	書き込みオペレーション	ファイルシステム単位の読み取り ¹	ファイルシステム単位の書き込み ¹	クライアントごとの読み取り/書き込み
リージョン別	Elastic	最低 250 マイクロ秒 (µs)	As low as 2.7 milliseconds (ms)	900,000–2,500,000 ²	500,000 ²	20 ~ 60 ギビバイト/秒 (GiBps)	1 ~ 5 GiBps	1,500 メビバイト/秒 (MiBps) ³
リージョン別	Provisioned	最低 250 µs	As low as 2.7 ms	55,000	25,000	3 ~ 10 GiBps	1 ~ 3.33 GiBps	500 MiBps
リージョン別	Bursting	最低 250 µs	As low as 2.7 ms	35,000	7,000	3 ~ 5 GiBps	1 ~ 3 GiBps	500 MiBps
1 ゾーン	Elastic, Provisioned, Bursting	最低 250 µs	最低 1.6 ミリ秒	35,000	7,000	3 GiBps ⁴	1 GiBps ⁴	500 MiBps

- 読み取り/書き込みの最大スループットは、AWS リージョンによって異なります。AWS リージョンの最大スループットを超えるスループットには、スループットクォータの引き上げが必要です。スループット追加のリクエストは、Amazon EFS サービスチームによって個別ケースごと

- に検討されます。承認はワークロードの種類によって異なる場合があります。クォータ増のリクエストについては、「[Amazon EFS のクォータ](#)」を参照してください。
2. デフォルトでは、Elastic スループットを使用するファイルシステムは、アクセス頻度の低いデータに対して最大 90,000 読み取り IOPS、頻繁にアクセスされるデータに対して最大 250,000 読み取り IOPS、および 50,000 書き込み IOPS を駆動します。ワークロードでより多くの IOPS が必要な場合は、これらの数値の最大 10 倍の引き上げをリクエストできます。詳細については、「[引き上げることができる Amazon EFS のクォータ](#)」を参照してください。最大 IOPS を達成するには、さらなる推奨事項が適用されます。詳細については、「[the section called “高いスループットと IOPS を必要とするワークロードの最適化”](#)」を参照してください。
 3. エラスティックスループットを使用し、バージョン 2.0 以降の Amazon EFS クライアント (amazon-efs-utils バージョン) または Amazon EFS CSI ドライバー (aws-efs-csi-driver) を使用してマウントされたファイルシステムでは、読み取りと書き込みを合わせた最大スループットは 1,500 MiBps になります。その他のすべてのファイルシステムでは、スループットの上限は 500 MiBps です。Amazon EFS クライアントの詳細については、「」を参照してください。[Amazon EFS クライアントの手動インストール](#)。
 4. バーストスループットを使用する 1 ゾーンファイルシステムでは、バーストスループットを使用するリージョンファイルシステムと同じファイルシステムあたりの読み取りおよび書き込みスループット量 (読み取りは最大 5 GiBps、書き込みは最大 3 GiBps) を実現できます。

ストレージクラス

Amazon EFS ストレージクラスは、ユースケースに応じて最も効果的なストレージになるように設計されています。

- EFS 標準ストレージクラスは、ソリッドステートドライブ (SSD) ストレージを使用して、頻繁にアクセスされるファイルのレイテンシーを最小限に抑えます。このストレージクラスでは、最初のバイトのレイテンシーは、読み取りで 250 マイクロ秒、書き込みで 2.7 ミリ秒と低くなります。
- EFS 低頻度アクセス (IA) ストレージクラスおよび EFS アーカイブストレージクラスは、頻繁にアクセスされるデータに求められるレイテンシーパフォーマンスを必要としない、アクセス頻度の低いデータを格納します。これらのストレージクラスでは、最初のバイトのレイテンシーが数十ミリ秒になります。

EFS ストレージクラスの詳細については、「[the section called “EFS ストレージクラス”](#)」を参照してください。

パフォーマンスモード

Amazon EFS には、汎用モードと最大 I/O という 2 つのパフォーマンスモードがあります。

- 汎用モードはオペレーションごとのレイテンシーが最も低く、ファイルシステムのデフォルトのパフォーマンスモードです。1 ゾーンファイルシステムでは常に汎用パフォーマンスモードが使用されます。パフォーマンスを向上させるには、常に汎用パフォーマンスモードを使用することをお勧めします。
- 最大 I/O モードは前世代のパフォーマンスタイプで、汎用モードよりも高いレイテンシーに耐えられる高度に並列化されたワークロード向けに設計されています。最大 I/O モードは、1 ゾーンファイルシステムまたはエラスティックスループットを使用するファイルシステムではサポートされません。

Important

最大 I/O ではオペレーションごとのレイテンシーが高くなるため、すべてのファイルシステムに汎用パフォーマンスモードを使用することをお勧めします。

汎用パフォーマンスモードのファイルシステムで利用可能な IOPS 制限内にワークロードが収まるようにするには、PercentIOLimit CloudWatch メトリクスをモニタリングできます。詳細については、「[Amazon EFS の CloudWatch メトリクス](#)」を参照してください。

アプリケーションは、パフォーマンスモードに関連する制限まで IOPS を柔軟にスケールアップできます。IOPS については個別に請求されることはありません。IOPS はファイルシステムのスループット計算に含まれます。すべてのネットワークファイルシステム (NFS) リクエストは、4 KB (スループット)、または実際のリクエストとレスポンスサイズのどちらか大きいほうのスループットとして計上されます。

スループットモード

ファイルシステムのスループットモードによって、ファイルシステムで利用できるスループットが決まります。Amazon EFS には、エラスティック、プロビジョニングされた、バーストの 3 つのスループットモードがあります。読み取りスループットは、書き込みスループットよりも読み取りスループットを高くするために割引されています。各スループットモードで利用できる最大スループットは、AWS リージョンによって異なります。各リージョンのファイルシステムの最大スループットの詳細については、「[Amazon EFS のクォータ](#)」を参照してください。

ファイルシステムは、読み取りと書き込みの合計スループットを 100% 達成できます。たとえば、ファイルシステムが読み取りスループット制限の 33% を使用している場合、ファイルシステムは同時に書き込みスループット制限の最大 67% を達成できます。ファイルシステムのスループット使用率は、コンソールの「ファイルシステム詳細」ページにあるスループット使用率 (%) グラフでモニタリングできます。詳細については、「[スループットパフォーマンスのモニタリング](#)」を参照してください。

ファイルシステムの適正なスループットモードを選択します。

ファイルシステムに適したスループットモードの選択は、ワークロードのパフォーマンス要件によって異なります。

- エラスティックスループット (推奨) - ワークロードが急上昇するなど予測不可能で、パフォーマンス要件を見積もることが難しい場合や、アプリケーションのスループットが平均対ピーク比 5% 以下の場合は、デフォルトのエラスティックスループットを使用します。詳細については、「[エラスティックスループットモード](#)」を参照してください。
- プロビジョンドスループット - ワークロードのパフォーマンス要件がわかっている場合、またはアプリケーションが平均対ピーク比 5% 以上のスループットを実現している場合は、プロビジョンドスループットを使用します。詳細については、「[プロビジョニングされたスループット](#)」を参照してください。
- バーストスループット - ファイルシステムのストレージ容量に合わせてスケールリングするスループットが必要な場合は、バーストスループットを使用します。

バーストスループットを使用した後、アプリケーションのスループットに制約がある場合 (例えば、許容スループットの 80% 以上を使用している場合や、バーストクレジットをすべて使い切った場合) は、エラスティックモードまたはプロビジョンドスループットモードを使用する必要があります。詳細については、「[スループットのバースト](#)」を参照してください。

Amazon EFS メトリクスの詳細については、「[Amazon EFS の CloudWatch メトリクス](#)」を参照してください。

エラスティックスループットモード

エラスティックスループットを使用するファイルシステムでは、Amazon EFS はワークロードアクティビティのニーズに合わせてスループットパフォーマンスを自動的にスケールアップまたはスケールダウンします。エラスティックスループットは、パフォーマンス要件を見積もることが難しい急上昇するワークロードや予測不可能なワークロード、またはスループットの平均がピークスループットの 5% 以下 (平均対ピーク比) になるアプリケーションに最適なスループットモードです。

エラスティックスループットのファイルシステムのスループットパフォーマンスは自動的にスケールされるため、アプリケーションのニーズに合わせてスループットキャパシティを指定したりプロビジョニングしたりする必要はありません。料金は読み書きされたメタデータとデータの量に対してのみ発生し、エラスティックスループットの使用中にバーストクレジットの蓄積や消費が行われることはありません。

Note

Elastic スループットはスループットに合わせて伸縮自在にスケールするように設計されていますが、運用のベストプラクティスの一環として、CloudWatch (MeteredIOBytes) と使用状況アラートを使用してメトリクスをモニタリングし、適切なガバナンスを実装することをお勧めします。これにより、最適なリソース使用率を維持し、計画された運用パラメータを維持できます。詳細については、「[Amazon CloudWatch によるメトリクスのモニタリング](#)」を参照してください。

リージョンごとのエラスティックスループットの制限については、「[引き上げることができる Amazon EFS のクォータ](#)」を参照してください。

プロビジョニングされたスループット

プロビジョンドスループットモードでは、ファイルシステムのサイズやバーストクレジットバランスとは無関係に、ファイルシステムが処理できるスループットのレベルを指定します。プロビジョンドスループットは、ワークロードのパフォーマンス要件がわかっている場合や、アプリケーションが平均対ピーク比 5% 以上のスループットを実現している場合に使用します。

プロビジョンドスループットを使用するファイルシステムでは、そのファイルシステムで有効になっているスループットの量に応じて課金されます。1 か月に請求されるスループット量は、標準ストレージから提供されるファイルシステムのベースラインスループットを超えてプロビジョニングされたスループットを、AWS リージョンの一般的なバーストベースラインスループット制限まで超えてプロビジョニングされたスループットに基づいて決まります。

ファイルシステムのベースラインスループットがプロビジョンドスループット量を超える場合、ファイルシステムに対して許可されるバーストスループット (その中で一般的なバーストベースラインスループット制限まで AWS リージョン) が自動的に使用されます。

リージョンごとのプロビジョンドスループット制限の詳細については、「[引き上げることができる Amazon EFS のクォータ](#)」を参照してください。

スループットのバースト

バーストスループットモードは、ファイルシステムのストレージ容量に合わせてスケーリングするスループットを必要とするワークロードに推奨されます。バーストスループットでは、基本スループットは標準ストレージクラスのファイルシステムのサイズに比例し、ストレージ 1 GiB あたり 50 KiBps の割合で計算されます。バーストクレジットは、ファイルシステムの消費量が基本スループットレートを下回ると発生し、スループットが基本レートを超えると差し引かれます。

バーストクレジットが利用可能な場合、ファイルシステムは、標準ストレージの 1 TiB あたり最大 100 MiBps (1 GiB あたり 50 KiBps) のスループットを、最低 100 MiBps で上限まで AWS リージョン 駆動できます。バーストクレジットが利用できない場合、ファイルシステムはストレージの TiB あたり最大 50 MiBps、最低でも 1 MiBps を駆動できます。

リージョンごとのバーストスループットについては、「[General resource quotas that cannot be changed](#)」を参照してください。

Amazon EFS バーストクレジットについて

バーストスループットでは、各ファイルシステムは、時間の経過に伴ってベースラインレートでバーストクレジットを獲得します。ベースラインレートは、EFS 標準ストレージクラスに格納されているファイルシステムのサイズによって決まります。ベースラインレートは、ストレージの 1 テビバイト (TiB) あたり 50 MiBps (ストレージの 1 GiB あたり 50 KiBps に相当)。Amazon EFS は、読み取りオペレーションを書き込みオペレーションの 3 分の 1 の速度まで計測するため、ファイルシステムは読み取りスループットで GiB あたり 150 KiBps、書き込みスループットで GiB あたり 50 KiBps のベースラインレートを駆動することができます。

ファイルシステムは、ベースラインの従量制レートでスループットを継続的に向上させることができます。ファイルシステムは、非アクティブであるか、スループットをベースラインの従量制レートより低くするたびに、バーストクレジットを蓄積します。蓄積されたバーストクレジットにより、ファイルシステムは、ベースラインレートを上回るスループットを駆動できます。

たとえば、標準ストレージクラスに 100 GiB の従量制データを含むファイルシステムでは、ベースラインスループットは 5 MiBps です。24 時間の非アクティブ期間中、ファイルシステムは 432,000 MiB 分のクレジットを獲得し ($5 \text{ MiB} \times 86,400 \text{ 秒} = 432,000 \text{ MiB}$)、これを使用して 72 分間 100 MiBps でバーストすることができる ($432,000 \text{ MiB} \div 100 \text{ MiBps} = 72 \text{ 分}$)

1 TiB を超えるファイルシステムは、残りの 50 パーセントで非アクティブになっていると、常に最大 50 パーセントの時間バーストすることができます。

次の表に、バーストの動作の例を示します。

ファイルシステムサイズ	バーストスループット	ベースラインスループット
スタンダードストレージの 100 GiB の計測データ	1 日あたり最大 72 分間、読み取り専用で 300 (MiBps) までバーストするか、または 1 日あたり最大 72 分間、書き込み専用で 100 MiBps までバースト	- 最大 15 MiBps まで読み取り専用連続駆動 - 最大 5 MiBps まで書き込み専用連続駆動
スタンダードストレージの 1 TiB の計測データ	1 日 12 時間、読み取り専用で 300 MiBps までバーストするか、または 1 日あたり 12 時間、書き込み専用で 100 MiBps までバースト	150 MiBps まで読み取り専用連続駆動 50 MiBps を書き込み専用連続駆動
スタンダードストレージの 10 TiB の計測データ	1 日 12 時間、読み取り専用で 3 GiBps までバーストするか、または 1 日 12 時間、書き込み専用で 1 GiBps までバーストする	1.5 GiBps を読み取り専用で連続駆動 500 MiBps を書き込み専用連続駆動
一般的には、より大規模なファイルシステム	1 日 12 時間、ストレージの 1 TiB あたり読み取り専用で 300 MiBps までバーストするか、または 1 日 12 時間、ストレージの 1 TiB あたり書き込み専用で 100 MiBps までバースト	- ストレージ 1 TiB あたり読み取り専用の 150 MiBps で連続駆動 - ストレージ 1 TiB あたり書き込み専用の 50 MiBps で連続駆動

Note

Amazon EFS は、ベースラインレートが低くても、すべてのファイルシステムに 1 MiBps のメタリングスループットを提供します。

ベースラインレートとバーストレートを決定するために使用されるファイルシステムサイズは、[DescribeFileSystems](#) API オペレーションで使用可能な ValueInStandard 計測サイズです。

ファイルシステムは、1 TiB より小さいファイルシステムの場合、2.1 TiB、または 1 TiB を超えるファイルシステムの場合は、1 TiB あたり 2.1 TiB の最大クレジットバランスを得ることができます。この動作は、ファイルシステムが連続して最大 12 時間バーストするのに十分なクレジットを蓄積できることを示しています。

スループットの切り替えとプロビジョニング量の変更に関する制限

既存のファイルシステムのスループットモードを切り替えたり、スループット量を変更したりできます。ただし、スループットモードをプロビジョンドスループットに切り替えたり、プロビジョンドスループットの量を変更したりすると、次のアクションが 24 時間制限されます。

- プロビジョンドスループットモードからエラスティックまたはバーストスループットモードに切り替える。
- プロビジョンドスループットの量を引き下げる。

Amazon EFS パフォーマンスのヒント

Amazon EFS を使用する場合は、次のパフォーマンスのヒントに留意してください。

平均 I/O サイズ

Amazon EFS は分散型であるため、可用性、耐久性、およびスケーラビリティが高いレベルで実現されています。分散型のアーキテクチャによって、それぞれのファイル操作のレイテンシーオーバーヘッドも小さくなります。このオペレーションあたりのレイテンシーの影響で、オーバーヘッドがより多くのデータ量に分散されるため、通常は平均 I/O サイズの増加に応じて全体のスループットが向上します。

高いスループットと IOPS を必要とするワークロードの最適化

高いスループットと IOPS を必要とするワークロードには、汎用パフォーマンスモードとエラスティックスループットで構成されたリージョンファイルシステムを使用します。

Note

頻繁にアクセスされるデータの最大読み取り IOPS を実現するには、ファイルシステムが Elastic スループットを使用する必要があります。

最高レベルのパフォーマンスを実現するには、アプリケーションまたはワークロードを以下のように設定して並列化を活用する必要があります。

1. 使用中のクライアントの数と少なくとも同じ数のディレクトリを使用して、すべてのクライアントとディレクトリにワークロードを均等に分散します。
2. 個々のスレッドを個別のデータセットやファイルに配置することで、競合を最小限に抑えます。
3. 10 個以上の NFS クライアントにワークロードを分散し、1 つのマウントターゲットで 1 クライアントあたり少なくとも 64 スレッドにします。

同時接続

Amazon EFS ファイルシステムは、最大数千の Amazon EC2 およびその他の AWS コンピューティングインスタンスに同時にマウントできます。より多くのインスタンス間でアプリケーションの並列化が可能である場合、コンピューティングインスタンス全体でファイルシステムのスループットレベルを向上させることができます。

リクエストモデル

ファイルシステムへの非同期書き込みを有効にすることにより、保留中の書き込みオペレーションは、非同期で Amazon EFS に書き込まれる前に、Amazon EC2 インスタンスでバッファリングされます。非同期書き込みは、通常レイテンシーが低くなります。非同期書き込みを実行するとき、カーネルはキャッシュの追加のメモリを使用します。

同期書き込みが有効になっているファイルシステム、またはキャッシュをバイパスするオプション (たとえば、`O_DIRECT`) を使用してファイルを開く場合、Amazon EFS に対して同期リクエストが発行されます。各オペレーションはクライアントと Amazon EFS の間のラウンドトリップを通過します。

Note

選択したリクエストモデルでは、整合性 (複数の Amazon EC2 インスタンスを使用している場合) と速度にトレードオフがあります。同期書き込みを使用すると、次のリクエストを処理する前に各書き込みリクエストトランザクションを完了できるため、データ整合性が高まります。非同期書き込みを使用すると、保留中の書き込みオペレーションをバッファリングすることでスループットが向上します。

NFS クライアントマウント設定

[EFS ファイルシステムをマウントする](#) および [Linux でのマウントに関する考慮事項](#) で説明されている推奨マウントオプションを使用していることを確認します。

Amazon EC2 インスタンスにファイルシステムをマウントする場合、Amazon EFS はネットワークファイルシステムバージョン 4.0 および 4.1 (NFSv4) プロトコルをサポートします。NFSv4.1 は、NFSv4.0 (1,000 ファイル/秒未満) と比較して、並列小ファイル読み取りオペレーション (10,000 ファイル/秒以上) のパフォーマンスが向上しています。macOS Big Sur を実行している Amazon EC2 macOS インスタンスでは、NFSv4.0 のみがサポートされています。

次のマウントオプションは使用しないでください。

- `noac`、`actimeo=0`、`acregmax=0`、`acdirmax=0` — これらのオプションは属性キャッシュを無効にするため、パフォーマンスに非常に大きな影響を与えます。
- `lookupcache=pos`、`lookupcache=none` — これらのオプションはファイル名検索キャッシュを無効にします。これはパフォーマンスに非常に大きな影響を与えます。
- `fsc` — このオプションはローカルファイルキャッシュを有効にしますが、NFS キャッシュの一貫性は変更されず、レイテンシーも短縮されません。

Note

ファイルシステムをマウントする際には、NFS クライアントの読み取りバッファと書き込みバッファのサイズを 1 MB に増やすことを検討してください。

スモールファイルのパフォーマンスを最適化する

ファイルの再オープンを抑え、並列処理を増やし、可能な限り参照ファイルをバンドルすることで、小容量ファイルのパフォーマンスを向上させることができます。

- サーバーとの往復回数を最小限に抑えます。

後にワークフローで必要になる場合、不必要にファイルを閉じないでください。ファイル記述子を開いたままにしておくと、キャッシュ内のローカルコピーに直接アクセスできます。ファイルのオープン、クローズ、メタデータのオペレーションは、通常、非同期的に行うことも、パイプラインを介して行うこともできません。

小さいファイルの読み取りまたは書き込みでは、2 回のラウンドトリップが余分に必要になります。

1 回のラウンドトリップ (ファイルを開く、ファイルを閉じる) には、メガバイトのバルクデータの読み取りまたは書き込みと同じくらいの時間がかかる場合があります。計算ジョブの開始時に入力ファイルまたは出力ファイルを 1 回開き、ジョブの実行中ずっと開いたままにしておくほうが効率的です。

- 並列処理を使用すると、ラウンドトリップタイムの影響を軽減できます。
- 参照ファイルを .zip のファイルにまとめます。アプリケーションによっては、ほとんど読み取り専用の小さな参照ファイルを大量に使用している場合があります。これらを .zip のファイルにバンドルすると、1 回のオープン/クローズの往復で多数のファイルを読み取ることができます。

.zip 形式では、個々のファイルにランダムにアクセスできます。

ディレクトリパフォーマンスの最適化

同時に変更される非常に大きなディレクトリ (10 万ファイル以上) に対して一覧表示 (ls) を実行すると、Linux NFS クライアントが応答を返さずにハングアップすることがあります。この問題は Amazon Linux 2 カーネル 4.14、5.4、5.10 に移植されたカーネル 5.11 では修正されています。

可能であれば、ファイルシステムのディレクトリ数を 10,000 未満に抑えることをお勧めします。可能な限りネストされたサブディレクトリを使用します。

ディレクトリを一覧表示する際は、ファイル属性はディレクトリ自体に保存されていないため、必要ない場合には取得を避けます。

NFS を最適化する read_ahead_kb サイズ

NFS read_ahead_kb 属性は、シーケンシャル読み取りオペレーション中に Linux カーネルが先読みまたはプリフェッチするキロバイト数を定義します。

5.4.* より前のバージョンの Linux カーネルでは、read_ahead_kb の値は、NFS_MAX_READAHEAD に rsize の値 (マウントオプションで設定されたクライアント設定の読み取りバッファサイズ) を乗じて設定されます。[推奨されるマウントオプション](#)を使用する場合、この式は read_ahead_kb を 15MB に設定します。

 Note

Linux カーネルバージョン 5.4.* 以降、Linux NFS クライアントは、デフォルトの `read_ahead_kb` の値である 128 KB を使用します。この値を 15 MB に増やすことをお勧めします。

`amazon-efs-utils` バージョン 1.33.2 以降で利用可能な Amazon EFS マウントヘルパーは、ファイルシステムをマウントした後、`read_ahead_kb` の値を `15 * rsize`、または 15 MB に自動的に変更します。

Linux カーネル 5.4 以降では、マウントヘルパーを使用してファイルシステムをマウントしない場合は、パフォーマンスを向上させるために `read_ahead_kb` を手動で 15 MB に設定することを検討してください。ファイルシステムをマウントした後、次のコマンドを使用して `read_ahead_kb` 値をリセットできます。コマンドを使用する前に、次の値を置き換えます。

- `read-ahead-value-kb` を希望するサイズ (キロバイト単位) に置き換えます。
- ファイルシステムのマウントポイントで `efs-mount-point` を置き換えます。

```
device_number=$(stat -c '%d' efs-mount-point)
((major = ($device_number & 0xFFF00) >> 8))
((minor = ($device_number & 0xFF) | (($device_number >> 12) & 0xFFF00)))
sudo bash -c "echo read-ahead-value-kb > /sys/class/bdi/$major:$minor/read_ahead_kb"
```

たとえば、次の例では、`read_ahead_kb` サイズを 15 MB に設定します。

```
device_number=$(stat -c '%d' efs)
((major = ($device_number & 0xFFF00) >> 8))
((minor = ($device_number & 0xFF) | (($device_number >> 12) & 0xFFF00)))
sudo bash -c "echo 15000 > /sys/class/bdi/$major:$minor/read_ahead_kb"
```

Amazon EFS のパフォーマンス問題のトラブルシューティング

一般的に、Amazon EFS で解決できない問題が発生した場合は、最新の Linux カーネルを使用していることを確認します。エンタープライズ Linux ディストリビューションを使用している場合は、以下をお勧めします。

- Amazon Linux 2 (カーネル 4.3 以降)

- Amazon Linux 2015.09 以降
- RHEL 7.3 以降
- Ubuntu 16.04 のすべてのバージョン
- カーネル 3.13.0-83 の Ubuntu 14.04 以降
- SLES 12 Sp2 以降

別のディストリビューションまたはカスタムカーネルを使用している場合は、カーネルバージョン 4.3 以降をお勧めします。

 Note

[多くのファイルを並列に開くとパフォーマンスが低下する](#) により、RHEL 6.9 は特定のワークロードには適していない可能性があります。

トピック

- [EFS ファイルシステムを作成できない](#)
- [NFS ファイルシステム上の許可されたファイルへのアクセスが拒否されました](#)
- [Amazon EFS コンソールにアクセスするときにエラーが発生しました](#)
- [Amazon EC2 インスタンスがハングする](#)
- [大量のデータを書き込みしているアプリケーションがハングする](#)
- [多くのファイルを並列に開くとパフォーマンスが低下する](#)
- [カスタム NFS 設定による書き込みの遅延](#)
- [Oracle Recovery Manager を使用したバックアップの作成に時間がかかる](#)

EFS ファイルシステムを作成できない

EFS ファイルシステムの作成要求が失敗し、次のメッセージが表示されます。

```
User: arn:aws:iam::111122223333:user/username is not authorized to
perform: elasticfilesystem:CreateFileSystem on the specified resource.
```

実行するアクション

AWS Identity and Access Management (IAM) ポリシーをチェックして、指定されたリソース条件で EFS ファイルシステムを作成する権限があることを確認します。詳細については、「[Amazon EFS のためのアイデンティティとアクセス管理](#)」を参照してください。

NFS ファイルシステム上の許可されたファイルへのアクセスが拒否されました

16 を超えるアクセスグループ ID (GID) が割り当てられたユーザーが NFS ファイルシステムでオペレーションを実行しようとする、ファイルシステム上の許可されたファイルへのアクセスが拒否される場合があります。この問題は、NFS プロトコルがユーザー 1 人あたり最大 16 の GID をサポートし、[RFC 5531](#) で定義されているように、追加の GID は NFS クライアント要求から切り捨てられるために発生します。

実行するアクション

各ユーザーに割り当てられるアクセスグループ (GID) の数が 16 を超えないように、NFS ユーザーとグループのマッピングを再構築してください。

Amazon EFS コンソールにアクセスするときにエラーが発生しました

このセクションでは、Amazon EFS 管理コンソールにアクセスしたときにユーザーが発生する可能性のあるエラーについて説明します。

ec2:DescribeVPCs の認証情報の認証中にエラーが発生しました

Amazon EFS コンソールにアクセスすると、次のエラーメッセージが表示されます。

```
AuthFailure: An error occurred authenticating your credentials for ec2:DescribeVPCs.
```

このエラーは、ログイン認証情報が Amazon EC2 サービスで正常に認証されなかったことを示します。Amazon EFS コンソールは、選択した VPC に EFS ファイルシステムを作成するときに、ユーザーに代わって Amazon EC2 サービスを呼び出します。

実行するアクション

クライアントが Amazon EFS コンソールにアクセスする時間が正しく設定されていることを確認します。

Amazon EC2 インスタンスがハングする

ファイルシステムをアンマウントせずにファイルシステムのマウントターゲットを削除したため、Amazon EC2 インスタンスがハングすることがあります。

実行するアクション

ファイルシステムのマウントターゲットを削除する前に、ファイルシステムをアンマウントします。Amazon EFS ファイルシステムのアンマウントの詳細については、[ファイルシステムをアンマウントする](#) を参照してください。

大量のデータを書き込みしているアプリケーションがハングする

Amazon EFS に大量のデータを書き込むアプリケーションがハングし、インスタンスが再起動します。

実行するアクション

アプリケーションが Amazon EFS にすべてのデータを書き込む時間が長すぎる場合、プロセスが応答しなくなったように見えるため、Linux が再起動することがあります。この動作は、`kernel.hung_task_panic` と `kernel.hung_task_timeout_secs` という 2 つのカーネルの設定パラメータにより定義されます。

次の例では、停止したプロセスの状態は `ps` コマンドにより D と共にインスタンスの再起動より前に報告されます。これは、プロセスが I/O で待機していることを示します。

```
$ ps aux | grep large_io.py
root 33253 0.5 0.0 126652 5020 pts/3 D+ 18:22 0:00 python large_io.py
/efs/large_file
```

再起動を避けるには、タイムアウト時間を長くするか、またはハングタスクが検出されたときのカーネルパニックを無効にします。次のコマンドは、ほとんどの Linux システムでタスクがハングしたときのカーネルパニックを無効にします。

```
$ sudo sysctl -w kernel.hung_task_panic=0
```

多くのファイルを並列に開くとパフォーマンスが低下する

複数のファイルを並列に開くアプリケーションでは、I/O の並列化のパフォーマンスが期待されるほど向上しません。

実行するアクション

この問題は、Network File System バージョン4 (NFSv4) クライアントおよび NFSv4.1 を使用する RHEL 6 クライアントで発生します。これらの NFS クライアントは NFS OPEN および CLOSE 操作をシリアル化するためです。NFS プロトコルバージョン 4.1 と、この問題が発生しない [Linux ディストリビューション](#) の 1 つを使用してください。

NFSv4.1 を使用できない場合は、Linux NFSv4.0 クライアントによりユーザー ID とグループ ID によるオープンとクローズのリクエストが直列化することに注意してください。この直列化は、複数のプロセスまたは複数のスレッドが同時にリクエストを発行した場合でも発生します。クライアントは、すべての ID が一致したときに、NFS サーバーに一度に 1 つのオープンまたはクローズの操作を送信します。これらの問題を回避するには、次のいずれかの操作を実行します。

- 同じプロセスを、同じ Amazon EC2 インスタンス上の異なるユーザー ID から実行できます。
- すべてのオープンリクエストでユーザー ID を同じにしておき、代わりにグループ ID のセットを変更することができます。
- 別の Amazon EC2 インスタンスから各プロセスを実行できます。

カスタム NFS 設定による書き込みの遅延

カスタム NFS クライアント設定があり、Amazon EC2 インスタンスがファイルシステムで別の Amazon EC2 インスタンスから実行された書き込み操作を参照するのに最大 3 秒かかります。

実行するアクション

この問題が発生した場合は、次のいずれかの方法で解決することができます。

- データを読み取っている Amazon EC2 インスタンスの NFS クライアントが属性のキャッシュを有効にしている場合は、ファイルシステムをアンマウントします。次に、属性のキャッシュを無効にするため、noac オプションを使用して再マウントします。NFSv4.1 では、属性のキャッシュはデフォルトで有効になっています。

Note

クライアント側のキャッシュを無効にすると、アプリケーションのパフォーマンスを低下させる可能性があります。

- NFS プロシージャと互換性のあるプログラミング言語を使用して、必要に応じて属性のキャッシュをクリアすることもできます。これを行うには、読み込みリクエストの直前に ACCESS プロシージャリクエストを送信します。

たとえば、Python プログラミング言語を使用して、以下の呼び出しを作成できます。

```
# Does an NFS ACCESS procedure request to clear the attribute cache, given a path to the file
import os
os.access(path, os.W_OK)
```

Oracle Recovery Manager を使用したバックアップの作成に時間がかかる

バックアップジョブを開始する前に Oracle Recovery Manager が 120 秒間一時停止すると、Oracle Recovery Manager を使用したバックアップの作成が遅くなることがあります。

実行するアクション

この問題が発生した場合は、Oracle ヘルプセンターにある「[NFS の Direct NFS クライアントコントロールの有効化と無効化](#)」の説明に従って、Oracle Direct NFS を無効にします。

Note

Amazon EFS では、Oracle Direct NFS はサポートされていません。

AMI とカーネルの問題のトラブルシューティング

以下に、Amazon EC2 インスタンスから Amazon EFS を使用している時の、特定の Amazon マシンイメージ (AMI) またはカーネルバージョンに関連する問題のトラブルシューティングについての情報を示します。

トピック

- [chown ができない](#)
- [クライアントのバグのためにファイルシステムが繰り返し操作を実行し続ける](#)
- [デッドロッククライアント](#)
- [大きなディレクトリ内のファイルの一覧表示に時間がかかる](#)

chown ができない

Linux の chown コマンドを使用してファイル/ディレクトリの所有権を変更することができません。

このバグのカーネルバージョン

2.6.32

実行するアクション

このエラーは次のように解決できます。

- EFS のルートディレクトリの所有権を変更するために必要な 1 回限りの設定のステップで chown を実行している場合、より新しいカーネルを実行しているインスタンスから chown コマンドを実行できます。たとえば、Amazon Linux の最新バージョンを使用します。
- chown が本稼働ワークフローの一部である場合、chown を使用するには、カーネルバージョンを更新する必要があります。

クライアントのバグのためにファイルシステムが繰り返し操作を実行し続ける

クライアントのバグのために、ファイルシステムが繰り返し操作を実行しスタックします。

実行するアクション

クライアントのソフトウェアを最新バージョンに更新します。

デッドロッククライアント

クライアントがデッドロック状態になります。

このバグのカーネルバージョン

- CentOS-7 でカーネルが Linux 3.10.0-229.20.1.el7.x86_64
- Ubuntu 15.10 でカーネルが Linux 4.2.0-18-generic

実行するアクション

次のいずれかを行います：

- 新しいカーネルバージョンに更新します。CentOS-7 では、カーネルバージョン Linux 3.10.0-327以降に修正が含まれています。
- 古いカーネルバージョンにダウングレードします。

大きなディレクトリ内のファイルの一覧表示に時間がかかる

これは、NFS クライアントがリスト操作を完了するためディレクトリを反復処理している間に、ディレクトリが変更された場合に発生します。この反復処理中にディレクトリの内容が変更されたことを NFS クライアントが認識すると、NFS クライアントは最初から反復処理を再開します。その結果、頻繁に変更されるファイルのある大きなディレクトリで ls コマンドが完了するまでに時間がかかることがあります。

このバグのカーネルバージョン

CentOS および RHEL バージョン 2.6.32-696.1.1.el6 より以前のカーネル

実行するアクション

この問題を解決するには、新しいカーネルバージョンにアップグレードします。

Amazon EFS でのデータのバックアップとレプリケーション

Amazon EFS では、データを保護するために EFS ファイルシステムが自動的にバックアップされます。耐障害性とデータ保護をさらに強化するために、EFS ファイルシステムを AWS リージョンにレプリケートできます。EFS ファイルシステムのバックアップとレプリケーションにより、EFS ファイルシステムのデータに何らかの問題が発生した場合でも、オペレーションやサービスの継続的な提供が可能になります。例えば、データの破損やデータの損失が発生した場合です。

トピック

- [EFS ファイルシステムのバックアップ](#)
- [EFS ファイルシステムのレプリケート](#)

EFS ファイルシステムのバックアップ

Amazon EFS は AWS Backup、Amazon EFS 内のデータを保護するためのバックアップポリシーを作成および管理するために使用できる、フルマネージド型のポリシーベースのサービス EFS とネイティブに統合されています。Amazon EFS コンソールを使用して作成したファイルシステムは、AWS Backup デフォルトで自動的にバックアップされます。CLI または API を使用してファイルシステムを作成すると、自動バックアップはデフォルトで 1 ゾーンファイルシステムに対してのみ有効になります。

AWS Backup for Amazon EFS を使用すると、次のアクションを実行できます。

- バックアッププランを設定して自動バックアップのスケジュールと保持を管理する。バックアップの頻度、バックアップのタイミング、バックアップの保持期間、バックアップのライフサイクルポリシーを指定します。
- Amazon EFS データのバックアップを復元する。ファイルシステムのデータを新規または既存のファイルシステムに復元できます。完全な復元を実行するか、アイテムレベルの復元を実行するかを選択することもできます。

の使用の詳細については AWS Backup、「[AWS Backup デベロッパーガイド](#)」の「[の開始方法 AWS Backup](#)」を参照してください。

トピック

- [が Amazon EFS と AWS Backup 連携する方法](#)
- [必要な IAM 許可](#)
- [バックアップのパフォーマンス](#)
- [EFS ファイルシステムの自動バックアップの管理](#)

が Amazon EFS と AWS Backup 連携する方法

Amazon EFS コンソールを使用して作成したファイルシステムは、AWS Backup デフォルトで自動的にバックアップされます。CLI または API を使用してファイルシステムを作成すると、自動バックアップはデフォルトで 1 ゾーンファイルシステムに対してのみ有効になります。AWS CLI または API を使用して EFS ファイルシステムを作成した後、自動バックアップを有効にできます。デフォルトの EFS バックアッププランでは、自動バックアップに AWS Backup 推奨される設定を使用します。これは、35 日間の保持期間を持つ日次バックアップです。デフォルトの EFS バックアッププランを使用して作成されたバックアップは、デフォルトの EFS バックアップポールドに保存されます。このポールドも、ユーザーの代わりに Amazon EFS によって作成されます。デフォルトのバックアッププランとバックアップポールドは削除できません。

EFS ファイルシステム内のデータはすべて、データのあるストレージクラスに関係なくバックアップされます。ライフサイクル管理が有効になっていて、低頻度アクセス (IA) ストレージクラスまたはアーカイブストレージクラスにデータがある EFS ファイルシステムをバックアップするときは、データアクセス料金は発生しません。復旧ポイントを復元すると、すべてのファイルがスタンダードストレージクラスに復元されます。

増分バックアップ

AWS Backup は EFS ファイルシステムの増分バックアップを実行します。最初のバックアップ時には、ファイルシステム全体のコピーが作成されます。そのファイルシステムのその後のバックアップでは、変更、追加、削除されたファイルとディレクトリのみがコピーされます。増分バックアップごとに、は完全な復元を可能にするために必要な参照データ AWS Backup を保持します。このアプローチにより、完全なバックアップの作成に必要な時間が最小限に抑えられ、データを複製しないことで、ストレージコストが節約できます。

バックアップの整合性

Amazon EFS は高い可用性を持つように設計されています。AWS Backup によるバックアップの実行中も、EFS ファイルシステムへのアクセスや変更を行うことができます。ただし、バックアップの実行中にファイルシステムに変更が加えられると、データの重複、相違、欠落などの不整合が生

じることがあります。これらの変更には、書き込み、名前の変更、移動、削除の操作が含まれます。バックアップの整合性を確保するため、バックアッププロセスの間は、ファイルシステムを変更するアプリケーションやプロセスを一時停止することをお勧めします。または、ファイルシステムが変更中でない期間にバックアップを実行するようにスケジュールします。

バックアップ完了ウィンドウ

オプションとして、バックアップの完了ウィンドウを指定できます。このウィンドウでは、バックアップが完了する必要がある期間を定義します。完了ウィンドウを指定する場合には、予想されるパフォーマンスとファイルシステムのサイズと構成を考慮します。それにより、ウィンドウ内でバックアップを確実に完了させることができます。

指定されたウィンドウ内で完了しなかったバックアップには、未完了のステータスが表示されます。次のスケジュールされたバックアップ中に、は中断した時点で AWS Backup 再開します。すべてのバックアップのステータスは、AWS Backup マネジメントコンソールで確認できます。

オンデマンドバックアップ

を使用すると AWS Backup、1 つのリソースをオンデマンドでバックアップポールドに保存できます。スケジュールされたバックアップとは異なり、オンデマンドバックアップを開始するためには、バックアップ計画を作成する必要はありません。この場合もバックアップにライフサイクルを割り当てることができます。これにより、復旧ポイントは自動的にコールドストレージ層に移動され、削除のタイミングが記録されます。

さらに、は、最新のウォームバックアップに存在しなくなったデータに対してのみ、データをコールドストレージ AWS Backup に自動的に移行します。例えば、バックアップの作成時にファイルシステムに 100 ファイルがあり、バックアップを作成した翌日に 2 ファイルを削除したとします (2 日目の数は 100 ファイル - 2 ファイル = 98 ファイル)。データをコールドストレージに移行すると、削除された 2 ファイルのみがコールドストレージに移動し、残りの 98 ファイルはウォームストレージとして課金されます。

同時バックアップ

AWS Backup は、バックアップをリソースごとに 1 つの同時バックアップに制限します。このため、バックアップジョブが既に進行中の場合には、スケジュールバックアップまたはオンデマンドバックアップが失敗する可能性があります。AWS Backup の制限の詳細については、「AWS Backup デベロッパーガイド」の「[AWS Backup クォータ](#)」を参照してください。

バックアップの削除

デフォルトの EFS バックアップポールドアクセスポリシーは、復旧ポイントの削除を拒否するように設定されています。EFS ファイルシステムの既存のバックアップを削除するには、ポールドアクセスポリシーを変更する必要があります。ポールドアクセスポリシーを変更せずに EFS 復旧ポイントを削除しようとすると、次のエラーメッセージが表示されます。

```
"Access Denied: Insufficient privileges to perform this action. Please consult with the account administrator for necessary permissions."
```

デフォルトのバックアップポールドアクセスポリシーを編集するには、編集するアクセス許可が必要です。詳細については、「IAM ユーザーガイド」の「[Allow all IAM actions \(admin access\)](#)」を参照してください。

必要な IAM 許可

AWS Backup は、ユーザーに代わって アカウントでサービスにリンクされたロールを作成します。このロールには、Amazon EFS バックアップを実行するために必要なアクセス権限が付与されています。

elasticfilesystem:backup および elasticfilesystem:restore のアクションを使用して、IAM エンティティ (ユーザー、グループ、ロールなど) に EFS ファイルシステムのバックアップを作成または復元する能力を許可または拒否することができます。これらのアクションは、ファイルシステムポリシーまたは アイデンティティベースの IAM ポリシーで使用できます。詳細については、「[Amazon EFS のためのアイデンティティとアクセス管理](#)」および「[IAM を使用してファイルシステムへのアクセスを制御する](#)」を参照してください。

バックアップのパフォーマンス

一般的に、では、次のバックアップと復元のレートを期待できます AWS Backup。大きいファイルやディレクトリを含む場合など、ワークロードによってはレートが低くなる場合があります。

- バックアップレートは、1 秒あたり 2,000 ファイル、または 1 秒あたり 400 メガバイト (MBps) のいずれか遅い方です。
- 復元レートは、1 秒あたり 1,500 ファイルまたは 200 MBps のいずれか遅い方です。

でのバックアップオペレーションの最大期間は 30 日 AWS Backup です。

AWS Backup を使用すると、蓄積されたバーストクレジットは消費されず、汎用パフォーマンスモードのファイルオペレーション制限にはカウントされません。詳細については、「[Amazon EFS ファイルシステムのクォータ](#)」を参照してください。

EFS ファイルシステムの自動バックアップの管理

Amazon EFS コンソールを使用してファイルシステムを作成すると、自動バックアップがデフォルトでオンになっています。AWS CLI または API を使用してファイルシステムを作成した後、自動バックアップを有効にできます。

AWS Backup コンソールを使用して、デフォルトのバックアッププラン設定を編集できます。詳細については、「AWS Backup 開発者ガイド」の「[バックアッププランの管理](#)」を参照してください。すべての自動バックアップを確認し、[AWS Backup コンソール](#) を使用してデフォルトの EFS バックアップ計画設定を編集できます。

Amazon EFS は、自動バックアップが有効な場合、値が `enabled` の `aws:elasticfilesystem:default-backup` システムタグキーを EFS ファイルシステムに適用します。

ファイルシステムを作成したら、コンソール、AWS CLI または EFS API を使用して自動バックアップをオンまたはオフにできます。

コンソールを使用する

1. Amazon Elastic File System コンソールの <https://console.aws.amazon.com/efs/> を開いてください。
2. [ファイルシステム] ページで、自動バックアップをオンまたはオフにするファイルシステムを選択し、[ファイルシステムの詳細] ページを表示します。
3. [General(全般)] 設定パネルの [Edit(編集)] を選択します。
4.
 - 自動バックアップをオンにするには、[Enable automatic backups(自動バックアップの有効化)] を選択します。
 - 自動バックアップをオフにするには、[Enable automatic backups(自動バックアップの有効化)] をクリアにします。
5. [Save changes(変更の保存)] を選択します。

の使用 AWS CLI

- `put-backup-policy` CLI コマンド(対応する API 操作は 2)を使用して、既存のファイルシステムの自動バックアップをオンまたはオフにします。
- 次のコマンドを使用して、自動バックアップをオンにします。

```
$ aws efs put-backup-policy --file-system-id fs-01234567 \  
--backup-policy Status="ENABLED"
```

Amazon EFS は新しいバックアップポリシーで応答します。

```
{  
  "BackupPolicy": {  
    "Status": "ENABLING"  
  }  
}
```

- 次のコマンドを使用して、自動バックアップをオフにします。

```
$ aws efs put-backup-policy --file-system-id fs-01234567 \  
--backup-policy Status="DISABLED"
```

Amazon EFS は新しいバックアップポリシーで応答します。

```
{  
  "BackupPolicy": {  
    "Status": "DISABLING"  
  }  
}
```

EFS ファイルシステムのレプリケート

耐障害性とデータ保護を強化するために、AWS リージョン内で EFS ファイルシステムをレプリケートできます。EFS ファイルシステムでレプリケーションを有効にすると、Amazon EFS は、ソースファイルシステムのデータとメタデータをデスティネーションファイルシステムに自動的かつ透過的にレプリケートします。災害が発生した場合や障害対応訓練を実施する場合は、レプリカファイルシステムにフェイルオーバーできます。オペレーションを再開するには、プライマリファイルシステムにフェイルバックできます。

デスティネーションファイルシステムを作成し、ソースファイルシステムとの同期を維持するプロセスを管理するために、Amazon EFS では [\[レプリケーション設定\]](#) を使用します。

レプリケーション設定を作成した後は、Amazon EFS により、ソースファイルシステムとデスティネーションファイルシステムの同期が自動的に維持されます。ソースファイルシステムに加えられた変更は、ポイントインタイムで一貫性がある方法でデスティネーションファイルシステムに転送されるわけではありません。代わりに、レプリケーションの最終同期時刻に基づいて転送されます。[最終同期時刻] は、ソースとデスティネーションの間で最後に正常に同期が完了した時刻を示します。最後に同期した時点でソースファイルシステムに加えられた変更は、デスティネーションファイルシステムにレプリケートされますが、最後の同期以降にソースファイルシステムに加えられた変更はレプリケートされない場合があります。詳細については、「[レプリケーションの詳細の表示](#)」を参照してください。

レプリケーションは、Amazon EFS が利用可能なすべての AWS リージョン で使用できます。デフォルトで無効になっているリージョンで EFS ファイルシステムをレプリケートするには、まずリージョンにオプトインする必要があります。詳細については、「AWS 全般のリファレンスガイド」の「[アカウントで利用できる AWS リージョン の指定](#)」を参照してください。後でリージョンからオプトアウトすると、Amazon EFS はそのリージョンのすべてのレプリケーションアクティビティを一時停止します。リージョンのレプリケーションアクティビティを再開するには、に AWS リージョン 再度オプトインします。

 Note

レプリケーションは、属性ベースのアクセス制御 (ABAC) でのタグの使用をサポートしていません。

トピック

- [コスト](#)
- [レプリケーションのパフォーマンス](#)
- [必要な IAM アクセス許可](#)
- [新しい EFS ファイルシステムへのレプリケーションの設定](#)
- [既存の EFS ファイルシステムへのレプリケーションの設定](#)
- [アカウント間で AWS EFS ファイルシステムをレプリケートする](#)
- [レプリケーションの詳細の表示](#)
- [レプリケーション設定の削除](#)

- [レプリカの使用](#)

コスト

レプリケーションを容易にするために、Amazon EFS はデスティネーションファイルシステムに隠しディレクトリとメタデータを作成します。これらは、請求対象となる約 12 メビバイト (MiB) の計測データに相当します。ファイルシステムの計測ファイルストレージの詳細については、「[Amazon EFS がファイルシステムとオブジェクトのサイズをどのように報告するかについて説明します。](#)」を参照してください。

レプリケーションのパフォーマンス

フェイルバックプロセス中に新しいレプリケーションを作成したり、既存のレプリケーションの方向を逆にしたりすると、Amazon EFS は最初の同期を実行します。これには、レプリケーションをサポートするための一連の 1 回限りのセットアップアクションが含まれます。最初の同期が終了するまでにかかる時間は、ソースファイルシステムのサイズやその中のファイル数などの要素によって異なります。

最初のレプリケーションが終了すると、Amazon EFS はほとんどのファイルシステムに対して 15 分の目標復旧時点 (RPO) を維持します。ただし、ソースファイルシステムに非常に頻繁に変更されるファイルがあり、1 億を超えるファイルや 100 GB を超えるファイルがある場合、レプリケーションに 15 分以上かかる場合があります。前回のレプリケーションが正常に終了したタイミングをモニタリングする方法については、「[レプリケーションの詳細の表示](#)」を参照してください。

コンソール、AWS Command Line Interface (AWS CLI)、API、および Amazon CloudWatch を使用して、前回成功した同期がいつ発生したかをモニタリングできます。CloudWatch では、EFS メトリクスの [TimeSinceLastSync](#) を使用します。詳細については、「[レプリケーションの詳細の表示](#)」を参照してください。

必要な IAM アクセス許可

Amazon EFS は、という名前の EFS サービスにリンクされたロール、`AWSServiceRoleForAmazonElasticFileSystem` または指定した IAM ロールを使用して、ソースファイルシステムと宛先ファイルシステム間のレプリケーションを同期します。IAM ロールを提供するには、レプリケーション設定を作成する IAM ユーザーまたはロールに `iam:PassRole` 許可が必要です。詳細については、「AWS Identity and Access Management ユーザーガイド」の「[AWS サービスにロールを渡すアクセス許可をユーザーに付与する](#)」を参照してください。

- の詳細についてはiam:CreateServiceLinkedRole、「」の例を参照してください[Amazon EFS のサービスリンクロールの使用](#)。
- カスタム IAM ロールの詳細については、「」を参照してください[カスタム信頼ポリシーを使用して IAM ロールを作成する](#)。

Note

クロスアカウントレプリケーションを実行する場合は、レプリケーション設定を作成するときに IAM ロールを指定する必要があります。サービスにリンクされたロールの使用は許可されていません。詳細については、「[アカウント間で AWS EFS ファイルシステムをレプリケートする](#)」を参照してください。

レプリケーション設定の作成時に指定するサービスにリンクされたロールまたは IAM ロールには、レプリケーションに対する次のアクセス許可が必要です。

- elasticfilesystem:DescribeFileSystem
- elasticfilesystem:CreateFileSystem
- elasticfilesystem:CreateReplicationConfiguration
- elasticfilesystem>DeleteReplicationConfiguration
- elasticfilesystem:DescribeReplicationConfigurations

AmazonElasticFileSystemFullAccess 管理ポリシーを使用して、必要なすべての EFS アクセス許可を自動的に取得できます。詳細については、「[AWS マネージドポリシー: AmazonElasticFileSystemFullAccess](#)」を参照してください。

新しい EFS ファイルシステムへのレプリケーションの設定

Amazon EFS は自動的に新しいファイルシステムを作成し、ソースファイルシステム上のデータとメタデータを、選択した の新しい読み取り専用の送信先ファイルシステムにコピー AWS リージョンします。新しいファイルシステムにレプリケートするときは、暗号化に使用するファイルシステムタイプと AWS Key Management Service (AWS KMS) キーを選択します。また、Amazon EFS は、デスティネーションファイルシステムの作成時にマウントターゲットを作成しません。レプリケーション設定を作成したら、[1 つ以上のマウントターゲットを作成してデスティネーションファイルシステムをマウント](#)する必要があります。

 Note

ファイルシステムは、1つのレプリケーション設定にのみ含めることができます。デスティネーションファイルシステムを別のレプリケーション設定のソースファイルシステムとして使用することはできません。

- ファイルシステムのタイプ — ファイルシステムのタイプによって、Amazon EFS ファイルシステムが AWS リージョン内にデータを保存する際の可用性と耐久性が決まります。
- [リージョン] を選択すると、AWS リージョン内のすべてのアベイラビリティーゾーンにわたってデータとメタデータを冗長的に保存するファイルシステムを作成します。
- [1 ゾーン] を選択すると、データとメタデータを単一のアベイラビリティーゾーン内に冗長的に保存するファイルシステムを作成します。

ファイルシステムのタイプの詳細については、「[EFS ファイルシステムのタイプ](#)」を参照してください。

- 暗号化 — すべてのデスティネーションファイルシステムは、保存時の暗号化を有効にして作成されます。送信先ファイルシステムの暗号化に使用される AWS KMS キーを指定できます。KMS キーを指定しない場合、Amazon EFS のサービスマネージド KMS キーが使用されます。

 Important

デスティネーションファイルシステムの作成後は KMS キーを変更できません。

デスティネーションファイルシステムは、ソースファイルシステムに基づいてデフォルト設定で作成されます。追加設定が必要な場合は、作成後に変更できます。

- 自動バックアップ - 1 ゾーンストレージを使用するデスティネーションファイルシステムでは、デフォルトで自動バックアップが有効になります。ファイルシステムの作成後、自動バックアップの設定を変更できます。詳細については、「[EFS ファイルシステムの自動バックアップの管理](#)」を参照してください。
- パフォーマンスモード - デスティネーションファイルシステムのパフォーマンスモードはソースファイルシステムと一致します。ただし、デスティネーションファイルシステムが 1 ゾーンストレージを使用している場合は例外です。その場合は汎用モードが使用されます。パフォーマンスモードは変更できません。

- スループットモード - デスティネーションファイルシステムのスループットモードはソースファイルシステムと一致します。ファイルシステムの作成後、モードを変更できます。

ソースファイルシステムのスループットモードがプロビジョンドの場合、デスティネーションファイルシステムのプロビジョンドスループットの量は、ソースファイルシステムのプロビジョニング量がデスティネーションファイルシステムのリージョン制限を超えない限り、ソースファイルシステムのプロビジョンドスループットの量と一致します。ソースファイルシステムのプロビジョニングされた量がデスティネーションファイルシステムのリージョン制限を超える場合、デスティネーションファイルシステムのプロビジョニングされたスループット量はリージョン制限になります。詳細については、「[引き上げることができる Amazon EFS のクォータ](#)」を参照してください。

- ライフサイクル管理 — ライフサイクル管理は、デスティネーションファイルシステムでは有効になっていません。デスティネーションファイルシステムを作成した後に有効にすることができます。詳細については、「[ストレージライフサイクルの管理](#)」を参照してください。

ステップ 1: レプリケーション設定を作成する

新しいファイルシステムにレプリケートする最初のステップは、レプリケーション設定を作成することです。

コンソールを使用する

1. にサインイン AWS Management Console し、<https://console.aws.amazon.com/efs/> で Amazon EFS コンソールを開きます。
2. レプリケートするファイルシステムを開きます。
 - a. 左のナビゲーションペインで [ファイルシステム] を選択します。
 - b. [ファイルシステム] の一覧から、レプリケートするファイルシステムを選択します。既存のレプリケーション設定では、選択するファイルシステムをソースまたはデスティネーションファイルシステムにすることはできません。
3. [レプリケーション] タブを選択します。
4. [レプリケーション] セクションで、[レプリケーションを作成] を選択します。
5. [レプリケーション設定] セクションで、レプリケーション設定を定義します。
 - a. [レプリケーション設定] で、新しいファイルシステムにレプリケートするか、既存のファイルシステムにレプリケートするかを選択します。

- b. Destination AWS リージョン AWS リージョン で、ファイルシステムをレプリケートするを選択します。
6. [送信先ファイルシステム設定] セクションで、デスティネーションファイルシステムの設定を定義します。
 - a. [ファイルシステムのタイプ] で、ファイルシステムのストレージオプションを選択します。
 - 内の地理的に離れた複数のアベイラビリティーゾーンにデータを冗長的に保存するファイルシステムを作成するには AWS リージョン、リージョンを選択します。
 - の単一のアベイラビリティーゾーン内に冗長的にデータを保存するファイルシステムを作成するには AWS リージョン、1 ゾーンを選択し、アベイラビリティーゾーンを選択します。

詳細については、「[EFS ファイルシステムのタイプ](#)」を参照してください。

 Note

1 ゾーンファイルシステムは、Amazon EFS が利用できる AWS リージョン のすべてのアベイラビリティーゾーンで利用できるわけではありません。

- b. [暗号化] では、保管中のデータの暗号化は、デスティネーションファイルシステム上で自動的に有効になります。デフォルトでは、Amazon EFS は your AWS Key Management Service (AWS KMS) サービスキー () を使用しますaws/elasticfilesystem。別の KMS キーを使用するには、KMS キーを選択するか、キーの Amazon リソースネーム (ARN) を入力します。

 Important

ファイルシステムが作成された後は、KMS キーを変更することはできません。

レプリケーション設定の作成 (AWS CLI)

このセクションでは、AWS CLI で create-replication-configuration コマンドを使用してレプリケーション設定を作成する例を示します。同等の API コマンドは [CreateReplicationConfiguration](#) です。

Example : リージョン別デスティネーションファイルシステムのレプリケーション設定を作成する。

次の例では、ファイルシステム *fs-0123456789abcdef1* のレプリケーション設定を作成します。この例では、Regionパラメータを使用して、で送信先ファイルシステムを作成します *eu-west-2* AWS リージョン。KmsKeyId パラメータは、デスティネーションファイルシステムを暗号化するとき使用する KMS キー ID を指定します。

```
aws efs create-replication-configuration \  
--source-file-system-id fs-0123456789abcdef1 \  
--destinations "[{\"Region\":\"eu-west-2\", \"KmsKeyId\":\"arn:aws:kms:us-east-2:111122223333:key/abcd1234-ef56-ab78-cd90-1111abcd2222\"}]"
```

は次のように AWS CLI 応答します。

```
{  
  "SourceFileSystemArn": "arn:aws:elasticfilesystem:us-east-1:111122223333:file-system/fs-0123456789abcdef1",  
  "SourceFileSystemRegion": "us-east-1",  
  "Destinations": [  
    {  
      "Status": "ENABLING",  
      "FileSystemId": "fs-0123456789abcde22",  
      "Region": "eu-west-2"  
    }  
  ],  
  "SourceFileSystemId": "fs-0123456789abcdef1",  
  "CreationTime": 1641491892.0,  
  "OriginalSourceFileSystemArn": "arn:aws:elasticfilesystem:us-east-1:111122223333:file-system/fs-0123456789abcdef1"  
}
```

Example : 1 ゾーンデスティネーションファイルシステムのレプリケーション設定を作成します。

次の例では、ファイルシステム *fs-0123456789abcdef1* のレプリケーション設定を作成します。この例では、AvailabilityZoneName パラメータを使用して、*us-west-2a* アベイラビリティゾーンの 1 ゾーンデスティネーションファイルシステムを作成します。KMS キーが指定されていないため、デスティネーションファイルシステムは、アカウントのデフォルトの AWS KMS サービスキー (aws/elasticfilesystem) を使用して暗号化されます。


```
aws efs create-replication-configuration \  
--source-file-system-id fs-0123456789abcdef1 \  
--destinations AvailabilityZoneName=us-west-2a
```

ステップ 2: デスティネーションファイルシステムをマウントする

Amazon EFS では、デスティネーションファイルシステムを作成するときに、マウントターゲットを作成しません。デスティネーションファイルシステムをマウントするには、1 つ以上のマウントターゲットを作成する必要があります。詳細については、「[EFS ファイルシステムをマウントする](#)」を参照してください。

既存の EFS ファイルシステムへのレプリケーションの設定

Amazon EFS は、ソースファイルシステムのデータとメタデータを、選択した宛先ファイルシステムにレプリケート AWS リージョン します。レプリケーション中、Amazon EFS はファイルシステム間のデータの差異を識別し、その差異をデスティネーションファイルシステムに適用します。

既存のファイルシステムにレプリケートするには、次の手順を実行します。

トピック

- [ステップ 1: ファイルシステムのレプリケーションの上書き保護を無効にする](#)
- [ステップ 2: レプリケーション設定を作成する](#)

Note

ファイルシステムは、1 つのレプリケーション設定にのみ含めることができます。デスティネーションファイルシステムを別のレプリケーション設定のソースファイルシステムとして使用することはできません。

ステップ 1: ファイルシステムのレプリケーションの上書き保護を無効にする

Amazon EFS ファイルシステムを作成すると、そのレプリケーション上書き保護はデフォルトで有効になります。レプリケーション上書き保護は、ファイルシステムがレプリケーション設定のデスティネーションとして使用されるのを防ぎます。ファイルシステムをレプリケーション設定のデスティネーションとして使用する前に、保護を無効にする必要があります。レプリケーション設定を削

除すると、ファイルシステムのレプリケーション上書き保護が再び有効になり、ファイルシステムが書き込み可能になります。

Amazon EFS ファイルシステムのレプリケーション上書き保護のステータスは、次の表で説明されている値のいずれかになります。

ファイルシステムの状態	説明
有効	ファイルシステムをレプリケーション設定のデスティネーションファイルシステムにすることはできません。ファイルシステムは書き込み可能です。レプリケーション上書き保護はデフォルトで ENABLED です。
無効	ファイルシステムをレプリケーション設定のデスティネーションファイルシステムにすることはできません。
レプリケーション	ファイルシステムをレプリケーション設定のデスティネーションファイルシステムにすることはできません。ファイルシステムは読み取り専用で、レプリケーション中に Amazon EFS によってのみ変更されます。

必要なアクセス許可

レプリケーション上書き保護を無効にするに

は、elasticfilesystem:UpdateFileSystemProtection アクションへのアクセス許可が必要です。詳細については、「[AWS マネージドポリシー: AmazonElasticFileSystemFullAccess](#)」を参照してください。

コンソールを使用する

1. にサインイン AWS Management Console し、<https://console.aws.amazon.com/efs/> で Amazon EFS コンソールを開きます。
2. 左のナビゲーションペインで [ファイルシステム] を選択します。
3. [ファイルシステム] のリストで、レプリケーション設定でデスティネーションファイルシステムとして使用したい Amazon EFS ファイルシステムを選択します。
4. [ファイルシステムの保護] セクションで、[レプリケーションの上書き保護] をオフにします。

レプリケーション上書き保護を無効にするには (AWS CLI)

次の例では、update-file-system-protection CLI コマンドは、指定されたファイルシステムのレプリケーション上書き保護を無効にします。同等の API コマンドは [UpdateFileSystemProtection](#) です。

```
aws efs update-file-system-protection
--file-system-id fs-0a8b2be428114d97c
--replication-overwrite-protection DISABLED
```

は次のように AWS CLI 応答します。

```
{
  "ReplicationOverwriteProtection": "DISABLED"
}
```

ステップ 2: レプリケーション設定を作成する

レプリケート先ファイルシステムでレプリケーション上書き保護を無効にすると、レプリケーション設定を作成できます。既存のファイルシステムにレプリケートする場合、レプリケート先ファイルシステムは、ソースファイルシステムと同じアカウントにあるか、ソースファイルシステムとは異なるアカウントにある可能性があります。

ソースファイルシステムが暗号化されている場合は、デスティネーションファイルシステムも暗号化する必要があります。さらに、ソースファイルが暗号化されておらず、デスティネーションファイルシステムが暗号化されている場合、フェイルオーバーを実行した後にソースデスティネーションにフェイルバックすることはできません。暗号化の詳細については、「[Amazon EFS でのデータの暗号化](#)」を参照してください。

Note

既存のファイルシステムへのレプリケーションを最初に設定すると、Amazon EFS はソースファイルシステム内のデータと一致するように、送信先ファイルシステムにデータを書き込みまたは削除します。送信先ファイルシステムのデータを変更しない場合は、代わりに新しいファイルシステムにレプリケートする必要があります。詳細については、「[新しい EFS ファイルシステムへのレプリケーションの設定](#)」を参照してください。

前提条件

使用する送信先ファイルシステム ID (同一アカウントレプリケーションの場合) または送信先ファイルシステム ARN (クロスアカウントレプリケーションの場合) のコピーを用意します。

送信先ファイルシステムがソースファイルシステム AWS アカウント とは異なる にある場合は、Amazon EFS がレプリケーションを実行し、ファイルシステムにリソースポリシーを割り当てることを許可する IAM ロールを作成します。詳細については、「[アカウント間で AWS EFS ファイルシステムをレプリケートする](#)」を参照してください。

コンソールを使用する

1. にサインイン AWS Management Console し、<https://console.aws.amazon.com/efs/> で Amazon EFS コンソールを開きます。
2. レプリケートするファイルシステムを開きます。
 - a. 左のナビゲーションペインで [ファイルシステム] を選択します。
 - b. ファイルシステムのリストから、レプリケートする Amazon EFS ファイルシステムを選択します。既存のレプリケーション設定では、選択するファイルシステムをソースまたはデスティネーションファイルシステムにすることはできません。
3. [レプリケーション] タブを選択します。
4. [レプリケーション] セクションで、[レプリケーションを作成] を選択します。
5. [レプリケーション設定] で、既存のファイルシステムを選択します。
6. 送信先ファイルシステムを選択します。
 - ソースファイルシステム AWS アカウント と同じ にあるファイルシステムにレプリケートするには：
 1. このアカウントでファイルシステムを選択 を選択し、送信先 AWS リージョン でファイルシステムをレプリケートする AWS リージョン を選択します。
 2. [EFS を参照] を選択し、ファイルシステムを選択します。デスティネーションファイルシステムへのパスが [送信先] ボックスに表示されます。
 - ソースファイルシステム AWS アカウント とは異なる にあるファイルシステムにレプリケートするには：
 1. 別のアカウントでファイルシステムを指定する を選択します。
 2. 送信先ファイルシステムの ARN には、送信先ファイルシステムの Amazon リソースネーム (ARN) を入力します。

Note

ファイルシステムでレプリケーションの上書き保護が有効になっている場合は、警告が表示されます。[保護を無効にする] を選択して新しいタブでファイルシステムを開き、[レプリケーションの上書き保護] をオフにします。保護を無効にしたら [レプリケーションを作成] タブに戻り、[更新] ボタンをクリックしてメッセージをクリアします。

7. IAM ロールには、Amazon EFS が宛先ファイルシステムにレプリケートできるようにする IAM ロールの ARN を入力します。これは、同じアカウントレプリケーションではオプションですが、クロスアカウントレプリケーションでは必須です。詳細については、「[アカウント間で AWS EFS ファイルシステムをレプリケートする](#)」を参照してください。
8. [レプリケーションを作成] を選択し、確認メッセージの入力ボックスに「confirm」と入力して、[レプリケーションを作成] を選択します。[レプリケーション] セクションにレプリケーションの詳細が表示されます。

レプリケーション設定を作成するには (AWS CLI)

このセクションでは、AWS CLI で `create-replication-configuration` コマンドを使用してレプリケーション設定を作成する例を示します。同等の API コマンドは [CreateReplicationConfiguration](#) です。

Example : 別のリージョンの既存の送信先ファイルシステムへのレプリケーション設定を作成する

次の例では、ファイルシステム ID が のファイルシステム ID `fs-0a8b2be428114d97c` に `fs-0123456789abcdef1` プリケートされるレプリケーション設定を作成します `eu-west-2` AWS リージョン。

```
aws efs create-replication-configuration \
--source-file-system-id fs-0123456789abcdef1 \
--destinations "[{\"Region\":\"eu-west-2\",\"FileSystemId\":\"fs-0a8b2be428114d97c\"}]"
```

は次のように AWS CLI 応答します。

```
{
  "SourceFileSystemId": "fs-0123456789abcdef1",
  "SourceFileSystemRegion": "us-east-1",
  "SourceFileSystemArn": "arn:aws:elasticfilesystem:us-east-1:111122223333:file-
system/fs-0123456789abcdef1",
```

```

    "OriginalSourceFileSystemArn": "arn:aws:elasticfilesystem:us-
east-1:111122223333:file-system/fs-0123456789abcdef1",
    "CreationTime": "2024-10-20T20:40:13+00:00",
    "Destinations": [
      {
        "Status": "ENABLING",
        "FileSystemId": "fs-0a8b2be428114d97c",
        "Region": "eu-west-2",
        "OwnerId": "123456789012",
      }
    ],
    "SourceFileSystemOwnerId": "123456789012"
  }
}

```

Example : クロスアカウントレプリケーション設定を作成する

次の例では、レプリケート元ファイルシステムとレプリケート先ファイルシステムが異なるレプリケーション設定を作成します AWS アカウント。アカウント **555666777888** **fs-0123456789abcdef1** のソースファイルシステム ID は、アカウント **123456789012** **fs-0a8b2be428114d97c** のファイルシステム ID にレプリケートされます。この例では、送信先ファイルシステムの Amazon リソースネーム (ARN) と、Amazon EFS が代わりにレプリケーションを実行できるようにするソースアカウントの IAM ロールの ARN を指定します。KMS キーが指定されていないため、送信先ファイルシステムはアカウントのデフォルトの AWS KMS サービスキー () を使用して暗号化されます `aws/elasticfilesystem`。

```

aws efs
--region $REGION
--endpoint $ENDPOINT create-replication-configuration
--source-file-system-id fs-0123456789abcdef1
--destinations Region=eu-west-2,FileSystemId=arn:aws:elasticfilesystem:eu-
west-2:123456789012:file-system/
fs-0a8b2be428114d97c,RoleArn=arn:aws:iam::555666777888:role/cross-account-replication

```

は次のように AWS CLI 応答します。

```

{
  "SourceFileSystemId": "fs-0123456789abcdef1",
  "SourceFileSystemRegion": "us-east-1",
  "SourceFileSystemArn": "arn:aws:elasticfilesystem:us-east-1:555666777888:file-
system/fs-0123456789abcdef1",
  "OriginalSourceFileSystemArn": "arn:aws:elasticfilesystem:us-
east-1:555666777888:file-system/fs-0123456789abcdef1",
}

```

```
{
  "CreationTime": "2024-10-20T20:40:13+00:00",
  "Destinations": [
    {
      "Status": "ENABLING",
      "FileSystemId": "fs-0a8b2be428114d97c",
      "Region": "eu-west-2",
      "OwnerId": "123456789012",
      "RoleArn": "arn:aws:iam::555666777888:role/cross-account-replication"
    }
  ],
  "SourceFileSystemOwnerId": "555666777888"
}
```

アカウント間で AWS EFS ファイルシステムをレプリケートする

EFS ファイルシステムをレプリケートできます AWS アカウント。アカウント間でレプリケートすることで、ディザスタリカバリ (DR) 戦略の全体的な耐障害性と信頼性が向上し、企業のコンプライアンス義務を満たすのに役立ちます。

例えば、コンプライアンスポリシーでは、環境 (本番稼働、ステージング、ディザスタリカバリ (DR) など) ごとに異なるアカウントを使用する必要がある場合があります。または、異なる にまたがるレプリケーション AWS アカウント により、より強力な分離、アクセス許可とアクセスポリシーのよりきめ細かな制御、リソースのより簡単な監査が可能になります。本番稼働用アカウントが侵害された場合 (セキュリティ違反、設定ミス、インサイダーの脅威など)、DR サーバーを別のアカウントに配置すると、攻撃者がそれらにアクセスするのを防ぎ、セキュリティインシデントの爆発的な範囲を減らし、不正な変更のリスクを最小限に抑えることができます。

間でレプリケートするには、追加のセキュリティとポリシーの設定 AWS アカウント が必要です。サービスにリンクされたロールを使用してクロスアカウントレプリケーションを実行する代わりに、レプリケート先アカウントでレプリケーションを実行するアクセス許可を Amazon EFS に付与する IAM ロールを作成する必要があります。また、アカウント間で共有するファイルシステムにポリシーを作成する必要があります。IAM ロールとファイルシステムポリシーを作成したら、レプリケーション設定を作成します。

トピック

- [カスタム信頼ポリシーを使用して IAM ロールを作成する](#)
- [送信元ファイルシステムと送信先ファイルシステムにポリシーを作成する](#)
- [レプリケーション設定を作成する](#)

カスタム信頼ポリシーを使用して IAM ロールを作成する

Amazon EFS がソースアカウントに代わってクロスアカウントレプリケーションを実行するには、ソースアカウントに IAM ロールを作成する必要があります。ロールには、Amazon EFS がロールを引き受け、サービスプリンシパルとして機能することを許可する `elasticfilesystem.amazonaws.com` 信頼ポリシーが必要です。ロールには、レプリケーションの実行に必要なすべての IAM アクセス許可 (「」を参照 [必要な IAM アクセス許可](#)) が含まれ、レプリケート先アカウントのファイルシステムにレプリケートするための明示的なアクセス許可を付与する必要があります。

前提条件

ソースアカウントの IAM ロールを作成する前に、レプリケーション設定でソースファイルシステムとレプリケート先ファイルシステムの両方を作成する必要があります。Amazon EFS は、レプリケーション中に送信先ファイルシステムを作成することはできません。さらに、各ファイルシステムの Amazon リソースネーム (ARN) を把握して提供する必要があります。

クロスアカウントレプリケーション用の IAM ロールを作成するには

以下は、Amazon EFS とのクロスアカウントレプリケーション用のカスタム信頼ポリシーを使用して IAM ロールを作成する一般的な手順です。IAM ロールを作成する step-by-step については、「AWS Identity and Access Management ユーザーガイド」の [「カスタム信頼ポリシーを使用してロールを作成する」](#) を参照してください。

1. ソースアカウントのコンソール AWS Identity and Access Management で、次の信頼ポリシーを使用する IAM ロールを作成します。手順については、AWS 「Identity and Access Management ユーザーガイド」の [「カスタム信頼ポリシーを使用してロールを作成する」](#) を参照してください。

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



```
}
```

2. ロールを作成したら、ロールに次のアクセス許可を割り当てます。宛先ファイルシステムの ARN `DESTINATION_FILE_SYSTEM_ARN` に置き換え、ソースファイルシステムの ARN `SOURCE_FILE_SYSTEM_ARN` に置き換えます。ロールにアクセス許可を割り当てる手順については、[「JSON エディタを使用したポリシーの作成」](#)を参照してください。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "elasticfilesystem:DescribeFileSystems",
        "elasticfilesystem:CreateReplicationConfiguration",
        "elasticfilesystem:DescribeReplicationConfigurations",
        "elasticfilesystem>DeleteReplicationConfiguration",
        "elasticfilesystem:ReplicationWrite"
      ],
      "Resource": "DESTINATION_FILE_SYSTEM_ARN"
    },
    {
      "Effect": "Allow",
      "Action": [
        "elasticfilesystem:ReplicationRead",
        "elasticfilesystem:DescribeFileSystems"
      ],
      "Resource": "SOURCE_FILE_SYSTEM_ARN"
    }
  ]
}
```

3. IAM ロールの ARN をコピーまたは書き留めます。レプリケーション設定を作成するときには、ARN を指定する必要があります。

送信元ファイルシステムと送信先ファイルシステムにポリシーを作成する

Amazon EFS でファイルシステムクロスアカウントを共有するには、送信先ファイルシステムとソースファイルシステムの両方にポリシーを割り当てる必要があります。ポリシーは、アカウント間で、それらが適用されるファイルシステムへのアクセスを許可または制限します。ファイルシステム

を編集する権限を持つアカウント所有者のみが、アカウントのファイルシステムにポリシーを割り当てることができます。

⚠ Important

ポリシーは、アカウント間のアクセスを許可または制限するだけでなく、クライアントがなどのファイルシステムと連携するために必要な他のアクセス許可を付与する必要があります `elasticfilesystem:ClientMount`。そうしないと、クライアントがファイルシステムにアクセスできなくなる可能性があります。ポリシーの例については、「[Amazon EFS のリソースベースのポリシーの例](#)」を参照してください。

送信先ファイルシステムのポリシー

レプリケート元アカウントにレプリケート先ファイルシステムへのレプリケートとレプリケート先アカウントからのレプリケーション設定の削除を許可するには、レプリケート先ファイルシステムに次のポリシーを作成する必要があります。を、ソースファイルシステムを所有するアカウントの ID `SOURCE_ACCOUNT_ROOT` に置き換えます。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Permissions for source account calls",
      "Effect": "Allow",
      "Principal": {
        "AWS": "SOURCE_ACCOUNT_ROOT"
      },
      "Action": [
        "elasticfilesystem:DescribeFileSystems",
        "elasticfilesystem:CreateReplicationConfiguration",
        "elasticfilesystem:DescribeReplicationConfigurations",
        "elasticfilesystem>DeleteReplicationConfiguration",
        "elasticfilesystem:ReplicationWrite"
      ],
      "Resource": "DESTINATION_FILE_SYSTEM_ARN"
    }
  ]
}
```

ソースファイルシステムのポリシー

レプリケート先アカウントがレプリケート元アカウントからレプリケーション設定を削除できるようにするには、次のポリシーをソースファイルシステムに割り当てる必要があります。を、送信先ファイルシステムを所有するアカウントの ID DESTINATION_ACCOUNT_ROOTに置き換えます。

```
{
  "Version": "2012-10-17",
  "Id": "efs-policy",
  "Statement": [
    {
      "Sid": "Permission to delete the replication by the destination account",
      "Effect": "Allow",
      "Principal": {
        "AWS": "DESTINATION_ACCOUNT_ROOT"
      },
      "Action": "elasticfilesystem:DeleteReplicationConfiguration",
      "Resource": "SOURCE_FILE_SYSTEM_ARN"
    }
  ]
}
```

ファイルシステムポリシーを作成するには

前のセクションのポリシーを使用して、送信先ファイルシステムとソースファイルシステムの両方で次の手順を実行します。

1. ファイルシステムを所有するアカウント AWS Management Console で にサインインし、Amazon EFS コンソールで [Amazon EFS コンソール](#)を開きます。
2. ファイルシステムを開きます。
 - a. 左のナビゲーションペインで [ファイルシステム] を選択します。
 - b. ファイルシステムリストで、ファイルシステムを選択します。
3. ファイルシステムポリシータブで、編集を選択します。
4. ポリシーエディタ {Json} にポリシーを貼り付け、保存を選択します。

レプリケーション設定を作成する

IAM ロールを作成し、ソースファイルシステムと宛先ファイルシステムにファイルシステムポリシーを追加したら、「」の手順に従ってレプリケーション設定 [既存の EFS ファイルシステムへのレプリケーションの設定](#) を作成します。

レプリケーションの詳細の表示

レプリケーション設定で前回正常に同期が完了した時間をモニタリングできます。この時間より前にソースファイルシステム上のデータに加えられた変更は、デスティネーションファイルシステムに正常にレプリケートされました。この時間以降に発生した変更は、完全にはレプリケートされない可能性があります。最後のレプリケーションが正常に終了したタイミングをモニタリングするには、Amazon EFS コンソール、API AWS CLI、または Amazon CloudWatch を使用できます。

- EFS コンソール - [ファイルシステムの詳細] > [レプリケーション] セクションの [最終同期] プロパティに、ソースとデスティネーションの間で最後に同期が成功した時刻が表示されます。
- AWS CLI または API で `-Destination` オブジェクトの `LastReplicatedTimestamp` プロパティは、最後に成功した同期が完了した時刻を示します。このプロパティにアクセスするには、`describe-replication-configurations` CLI コマンドを使用します。[DescribeReplicationConfigurations](#) は同等の API オペレーションです。
- CloudWatch — Amazon EFS の `TimeSinceLastSync` CloudWatch メトリクスには、前回正常に同期が完了してから経過した時間が表示されます。詳細については、「[Amazon EFS の CloudWatch メトリクス](#)」を参照してください。

レプリケーション設定には、次の表で説明されているステータス値のいずれかを持つことができます。

レプリケーションステータス	説明
ENABLED	レプリケーション設定は正常な状態にあり、使用可能です。
ENABLING	Amazon EFS はレプリケーション設定を作成中です。
DELETING	Amazon EFS は、ユーザーが開始する削除リクエストに応答して、レプリケーション設定を削除しています。
PAUSING	Amazon EFS はレプリケーションを一時停止中です。

レプリケーション インスタタス	説明
PAUSED	設定に問題があるため、レプリケーションは一時停止されます。問題に関する追加情報が提供されます。 レプリケーションが一時停止する原因となる問題には、次のようなものがあります。 • 認可エラー。アクセス許可の問題により、Amazon EFS はレプリケーションを実行できません。レプリケーション設定の作成に使用される IAM ロールに、レプリケーションを実行するアクセス許可があることを確認します。また、ファイルシステムポリシーが正しいことを確認してください。 • AWS アカウントは利用できません。送信元アカウントと送信先アカウントが適切に オプトイン AWS リージョン されており、停止されていないことを確認します。詳細については、「AWS 全般のリファレンスガイド」のAWS リージョン「アカウントで利用できるを指定する」を参照してください。 • 送信元または送信先のファイルシステムの KMS キーにアクセスできません。各ファイルシステムに割り当てられた KMS キーにアクセスできることを確認します。詳細については、「EFS ファイルシステム用の KMS キーの管理」を参照してください。
ERROR	レプリケーション設定が失敗状態であり、回復できません。レプリケーション設定を削除し、新しい設定を作成する必要があります。 問題に関する追加情報が提供されます。クロスアカウントまたはクロスリージョンレプリケーションの場合、レプリケーション設定が他の AWS アカウント または から削除されたためにエラーが発生する可能性があります AWS リージョン。

コンソールを使用する

1. Amazon Elastic File System コンソール の<https://console.aws.amazon.com/efs/>を開いてください。
2. 左のナビゲーションペインで [ファイルシステム] を選択します。
3. リストからファイルシステムを選択します。

4. [レプリケーション] タブを選択し、[レプリケーション] セクションを表示します。

[レプリケーション] セクションには、レプリケーション設定に関する以下の情報が表示されます。

- レプリケーションステータスは、[有効中]、[有効済み]、[削除中]、[一時停止中]、[一時停止済み] または [エラー] の場合があります。Amazon EFS は、一時停止状態またはエラー状態の原因に関する詳細を表示します。
- レプリケーション方向は、データがレプリケートされている方向を示します。最初にリストされているファイルシステムがソースで、そのデータは 2 番目にリストされているファイルシステム (デスティネーション) に対してレプリケートされています。
- [最終同期] には、デスティネーションファイルシステムで前回正常に同期が行われた日時が表示されます。この時刻より前にソースファイルシステム上のデータに加えられた変更は、すべてデスティネーションファイルシステムに正常にレプリケートされました。この時間以降に発生した変更は、完全にはレプリケートされない可能性があります。
- レプリケーションファイルシステムは、レプリケーション設定内の各ファイルシステムを、そのファイルシステム ID、レプリケーション設定内のロール (送信元または送信先)、AWS リージョン、それが配置されている、およびそのアクセス許可によって一覧表示します。ソースファイルシステムのアクセス許可は書き込み可能で、デスティネーションファイルシステムのアクセス許可は読み取り専用です。

の使用 AWS CLI

レプリケーション設定を表示するには、`describe-replication-configurations` コマンドを使用します。特定のファイルシステムのレプリケーション設定、または AWS アカウント の特定の のすべてのレプリケーション設定を表示できます AWS リージョン。同等の API コマンドは [DescribeReplicationConfigurations](#) です。

レプリケーション設定のステータスが PAUSED または の場合 ERROR、問題の原因と修正方法に関する情報が `StatusMessage` パラメータで返されます。

Example : 特定のファイルシステムのレプリケーション設定を表示する

次の例では、ファイルシステム のレプリケーション設定について説明します `fs-0123456789abcdef1`。

```
aws efs describe-replication-configurations --file-system-id fs-0123456789abcdef1
```

は次のように AWS CLI 応答します。

```
{
  "Replications": [
    {
      "SourceFileSystemArn": "arn:aws:elasticfilesystem:eu-west-1:111122223333:file-system/fs-abcdef0123456789a",
      "CreationTime": 1641491892.0,
      "SourceFileSystemRegion": "eu-west-1",
      "OriginalSourceFileSystemArn": "arn:aws:elasticfilesystem:eu-west-1:111122223333:file-system/fs-abcdef0123456789a",
      "SourceFileSystemId": "fs-abcdef0123456789a",
      "Destinations": [
        {
          "Status": "ENABLED",
          "FileSystemId": "fs-0123456789abcdef1",
          "Region": "us-east-1"
        }
      ]
    }
  ]
}
```

Example : アカウントのすべてのレプリケーション設定のレプリケーション設定を表示する

次の例では、ファイルシステム内のアカウントのすべてのレプリケーション設定 AWS リージョンのレプリケーション設定について説明します。

```
aws efs describe-replication-configurations
```

は次のように AWS CLI 応答します。

```
{
  "Replications": [
    {
      "SourceFileSystemArn": "arn:aws:elasticfilesystem:eu-west-1:555555555555:file-system/fs-0123456789abcdef1",
      "CreationTime": 1641491892.0,
      "SourceFileSystemRegion": "eu-west-1",
```

```

      "OriginalSourceFileSystemArn": "arn:aws:elasticfilesystem:eu-
west-1:555555555555:file-system/fs-0123456789abcdef1",
      "SourceFileSystemId": "fs-0123456789abcdef1",
      "Destinations": [
        {
          "Status": "ENABLED",
          "FileSystemId": "fs-abcdef0123456789a",
          "Region": "us-east-1",
          "LastReplicatedTimestamp": 1641491802.375
        }
      ]
    },
    {
      "SourceFileSystemArn": "arn:aws:elasticfilesystem:eu-
west-1:555555555555:file-system/fs-021345abcdef6789a",
      "CreationTime": 1641491822.0,
      "SourceFileSystemRegion": "eu-west-1",
      "OriginalSourceFileSystemArn": "arn:aws:elasticfilesystem:eu-
west-1:555555555555:file-system/fs-021345abcdef6789a",
      "SourceFileSystemId": "fs-021345abcdef6789a",
      "Destinations": [
        {
          "Status": "ENABLED",
          "FileSystemId": "fs-012abc3456789def1",
          "Region": "us-east-1",
          "LastReplicatedTimestamp": 1641491823.575
        }
      ]
    }
  ]
}

```

レプリケーション設定の削除

デスティネーションファイルシステムにフェイルオーバーする必要がある場合は、そのファイルシステムがメンバーとなっているレプリケーション設定を削除します。レプリケーション設定を削除すると、デスティネーションファイルシステムが書き込み可能になり、レプリケーション上書き保護が再び有効になります。詳細については、「[レプリカの使用](#)」を参照してください。

レプリケーション設定を削除し、デスティネーションファイルシステムを書き込み可能に変更すると、完了するまでに数分かかることがあります。設定が削除されると、Amazon EFS はデスティ

ネーションファイルシステムのルートディレクトリの lost+found ディレクトリに、以下の命名規則を使ってデータを書き込む場合があります。

```
efs-replication-lost+found-source-file-system-id-TIMESTAMP
```

 Note

レプリケーション設定の一部であるファイルシステムを削除することはできません。ファイルシステムを削除する前に、レプリケーション設定を削除する必要があります。

Amazon EFS コンソール、または API を使用して、ソースファイルシステム AWS CLI または宛先ファイルシステムから既存のレプリケーション設定を削除できます。

クロスアカウントレプリケーションまたはクロスリージョンレプリケーションの場合、Amazon EFS はレプリケート元アカウントとレプリケート先アカウントまたはリージョンの両方からレプリケーション設定を削除します。Amazon EFS がレプリケーション設定を両側から削除できない設定またはアクセス許可の問題がある場合は、ローカル側 (削除が実行されるアカウントまたはリージョン) からのみ設定を削除できます。ローカル設定を削除すると、他のアカウントまたはリージョンの設定は回復できなくなります。

コンソールを使用する

1. Amazon Elastic File System コンソールの <https://console.aws.amazon.com/efs/> を開いてください。
2. 左のナビゲーションペインで [ファイルシステム] を選択します。
3. 削除するレプリケーション設定に含まれるソースファイルシステムまたはデスティネーションファイルシステムを選択します。
4. [レプリケーション] タブを選択し、[レプリケーション] セクションを表示します。
5. [レプリケーションを削除] を選択して、レプリケーション設定を削除します。プロンプトが表示されたら、選択内容を確認します。

クロスアカウントレプリケーション設定を削除していて、送信元側と送信先側の両方から設定を削除できない問題がある場合は、このファイルシステムの設定のみを削除するオプションを選択できます。

 Note

Amazon EFS がレプリケート元アカウントとレプリケート先アカウントまたはリージョンの両方でレプリケーション設定を削除できない場合にのみ、ファイルシステムの設定を削除します。ローカル設定を削除すると、他のアカウントまたはリージョンの設定は回復できなくなります。

の使用 AWS CLI

レプリケーション設定を削除するには、CLI `delete-replication-configuration` を使用します。同等の API コマンドは [DeleteReplicationConfiguration](#) です。

次の の例では、ソースファイルシステム のレプリケーション設定を削除します `fs-0123456789abcdef1`。

```
aws efs --region us-west-2 delete-replication-configuration \  
--source-file-system-id fs-0123456789abcdef1
```

設定またはアクセス許可の問題により Amazon EFS がレプリケーション設定を両側から削除できない場合は、ローカル側 (削除が実行されるアカウントまたはリージョン) からのみ設定を削除できます。ローカル設定を削除すると、他のアカウントまたはリージョンの設定は回復できなくなります。同等の API パラメータは `DeletionMode` で、値は `LOCAL_CONFIGURATION_ONLY` です。

次の の例では、ソースファイルシステム `fs-0123456789abcdef1` のレプリケーション設定をローカル側からのみ削除します。

```
aws efs --region us-west-2 delete-replication-configuration \  
--source-file-system-id fs-0123456789abcdef1 \  
--deletion-mode LOCAL_CONFIGURATION_ONLY
```

レプリカの使用

災害が発生した場合や、障害対応訓練を実施する場合、レプリカファイルシステムのレプリケーション設定を削除することでレプリカファイルシステムにフェイルオーバーできます。レプリケーション設定が削除されると、レプリカは書き込み可能になり、アプリケーションワークフローで使用できるようになります。災害が落ち着いた状況になるか、障害対応訓練が終了したら、レプリカをプライマ

リファイルシステムとして使用し続けることも、フェイルバックを実行して元のプライマリファイルシステムでの操作を再開することもできます。

フェイルバックプロセス中、レプリカファイルシステムに加えられた変更を破棄するか、プライマリにコピーして保存するかを選択できます。

- フェイルオーバー中にレプリカに加えられた変更を破棄するには、レプリカファイルシステムをレプリケーションデスティネーションとするプライマリファイルシステムに元のレプリケーション設定を再作成します。レプリケーション中、Amazon EFS はレプリカファイルシステムのデータをプライマリのデータと一致するように更新することで、ファイルシステムを同期します。
- フェイルオーバー中にレプリカに加えられた変更をレプリケートするには、プライマリファイルシステムをレプリケーションデスティネーションとするレプリカファイルシステムにレプリケーション設定を作成します。レプリケーション中、Amazon EFS は差異を識別し、レプリカファイルシステムからプライマリファイルシステムに転送します。レプリケーションが完了したら、元のレプリケーション設定を再作成するか、新しい設定を作成することで、プライマリファイルシステムのレプリケーションを再開できます。

Amazon EFS がレプリケーションプロセスを完了するまでにかかる時間は、ファイルシステムのサイズやその中のファイル数などの要素によって異なります。詳細については、「[レプリケーションのパフォーマンス](#)」を参照してください。

Amazon EFS でのデータの保護

AWS [責任共有モデル](#)、Amazon Elastic File System でのデータ保護に適用されます。このモデルで説明されているように、AWS はすべての を実行するグローバルインフラストラクチャを保護する責任があります AWS クラウド。ユーザーは、このインフラストラクチャでホストされるコンテンツに対する管理を維持する責任があります。また、使用する「AWS のサービス」のセキュリティ設定と管理タスクもユーザーの責任となります。データプライバシーの詳細については、[データプライバシーに関するよくある質問](#)を参照してください。欧州でのデータ保護の詳細については、AWS セキュリティブログに投稿された [AWS 責任共有モデルおよび GDPR](#) のブログ記事を参照してください。

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

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

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

トピック

- [Amazon EFS でのデータの暗号化](#)
- [Amazon EFS のためのアイデンティティとアクセス管理](#)
- [IAM を使用してファイルシステムへのアクセスを制御する](#)
- [NFS クライアントの EFS ファイルシステムへのネットワークアクセスの制御](#)
- [ネットワークファイルシステム \(NFS\) レベルのユーザー、グループ、およびアクセス許可](#)
- [アクセスポイントの使用](#)
- [EFS ファイルシステムへのパブリックアクセスのブロック](#)
- [Amazon EFS のコンプライアンス検証](#)
- [Amazon EFS の耐障害性](#)
- [Amazon EFS のネットワーク分離](#)

Amazon EFS でのデータの暗号化

Amazon EFS は、ファイルシステムの 2 つの暗号化形式、転送時の暗号化と保管時のデータの暗号化をサポートします。Amazon EFS ファイルシステムを作成する場合、保管時のデータの暗号化を有効にすることができます。後でファイルシステムをマウントする時、伝送中のデータの暗号化を有効にすることができます。

コマンドラインインターフェイスまたは API AWS を介して にアクセスするときに FIPS 140-2 検証済みの暗号化モジュールが必要な場合は、FIPS エンドポイントを使用します。利用可能な FIPS エンドポイントの詳細については、「[連邦情報処理規格 \(FIPS\) 140-2](#)」を参照してください。

保管時のデータとメタデータの暗号化が必要な企業または規制ポリシーの対象となる組織の場合は、転送中のデータの暗号化を使用してファイルシステムをマウントする保管時に暗号化されたファイルシステムを作成することをおすすめします。

トピック

- [AWS KMS](#)
- [保管中のデータの暗号化](#)
- [Encrypting data in transit](#)
- [暗号化のトラブルシューティング](#)

AWS KMS

Amazon EFS は、キー管理のために AWS Key Management Service (AWS KMS) と統合されています。Amazon EFS は、次のようにカスタマーマネージドキーを使用してファイルシステムを暗号化します。

- 保存時のメタデータの暗号化 – Amazon EFS は Amazon EFS に AWS マネージドキー、`aws/elasticfilesystem` を使用して、ファイルシステムメタデータ (つまり、ファイル名、ディレクトリ名、ディレクトリの内容) を暗号化および復号します。
- 保管中のデータの暗号化 – ファイルデータ (ファイルの内容) の暗号化と復号に使用するカスタマーマネージドキーを選択します。このカスタマーマネージドキーの許可を有効化、無効化、または削除することができます。このカスタマーマネージドキーは、以下の 2 つのタイプのいずれかになります。
- AWS マネージドキー Amazon EFS の - これはデフォルトのカスタマーマネージドキーです `aws/elasticfilesystem`。カスタマーマネージドキーの作成と保存には料金はかかりませんが、利用料金はかかります。詳細については、「[AWS Key Management Service 料金表](#)」を参照してください。
- カスタマー管理キー - これは、キーポリシーと許可を複数のユーザーまたはサービスに設定できる、最も柔軟性のある KMS キーです。カスタマーマネージドキーの作成の詳細については、「デベロッパーガイド」の「[キーの作成](#)」を参照してください。AWS Key Management Service

ファイルデータ暗号化と復号化のためにカスタマーマネージドキーを使用する場合は、キーローテーションを有効にできます。キーローテーションを有効にすると、はキーを 1 年に 1 回 AWS KMS 自動的にローテーションします。また、カスタマー管理キーでは、カスタマー管理キーへのアクセスを無効にしたり、再び有効化したり、削除したり、取り消すタイミングを随時選択することができます。詳細については、「[暗号化されたファイルシステムへのアクセスの管理](#)」を参照してください。

Important

Amazon EFS では、対称カスタマーマネージドキーのみを使用できます。Amazon EFS では、非対称カスタマーマネージドキーを使用することはできません。

保管時のデータの暗号化と復号は透過的に処理されます。ただし、Amazon EFS に固有の AWS アカウント IDs、AWS KMS アクションに関連する AWS CloudTrail ログに表示されます。詳細については、「[encrypted-at-rest ファイルシステムの Amazon EFS ログファイルエントリ](#)」を参照してください。

の Amazon EFS キーポリシー AWS KMS

キーポリシーはカスターマネージドキーへのアクセスを制御するための主要な方法です。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「[AWS KMS のキーポリシー](#)」を参照してください。次のリストでは、暗号化された保管時のファイルシステムに対して Amazon EFS が必要とする、またはその他の方法でサポートする、AWS KMSに関連するすべてのアクセス許可について説明します。

- kms:Encrypt - (オプション) プレーンテキストを暗号化テキストに暗号化します。この許可は、デフォルトのキーポリシーに含まれています。
- kms:Decrypt - (必須) 暗号化テキストを復号します。暗号文は、以前に暗号化された平文です。このアクセス許可は、デフォルトのキーポリシーに含まれています。
- kms:ReEncrypt - (オプション) クライアント側にデータのプレーンテキストを公開することなく、サーバー側で新しいカスターマネージドキーを使用してデータを暗号化します。データは最初に復号化され、次に再暗号化されます。このアクセス許可は、デフォルトのキーポリシーに含まれています。
- kms:GenerateDataKeyWithoutPlaintext - (必須) カスターマネージドキーで暗号化されたデータ暗号化キーを返します。この許可は、kms:GenerateDataKey* のデフォルトのキーポリシーに含まれています。
- kms:CreateGrant - (必須) キーを使用できるユーザーとその条件を指定する許可をキーに付与します。付与は、主要なポリシーに対する代替の許可メカニズムです。グラントの詳細については、「デベロッパーガイド」の「[のグラン AWS KMS](#)」を参照してください。AWS Key Management Service このアクセス許可は、デフォルトのキーポリシーに含まれています。
- kms:DescribeKey - (必須) 指定されたカスターマネージドキーに関する詳細情報を提供します。このアクセス許可は、デフォルトのキーポリシーに含まれています。
- kms:ListAliases - (オプション) アカウント内のキーエイリアスをすべて一覧表示します。コンソールを使用して暗号化されたファイルシステムを作成すると、このアクセス許可により KMS マネージドキーの選択リストが設定されます。最高のユーザーエクスペリエンスを提供するには、この許可を使用することをお勧めします。このアクセス許可は、デフォルトのキーポリシーに含まれています。

AWS マネージドキー for Amazon EFS KMS ポリシー

Amazon EFS の AWS マネージドキー の KMS ポリシー JSON `aws/elasticfilesystem` は次のとおりです。

```
{
  "Version": "2012-10-17",
  "Id": "auto-elasticfilesystem-1",
  "Statement": [
    {
      "Sid": "Allow access to EFS for all principals in the account that are
authorized to use EFS",
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
      },
      "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:CreateGrant",
        "kms:DescribeKey"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "kms:ViaService": "elasticfilesystem.us-east-2.amazonaws.com",
          "kms:CallerAccount": "111122223333"
        }
      }
    },
    {
      "Sid": "Allow direct access to key metadata to the account",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:root"
      },
      "Action": [
        "kms:Describe*",
        "kms:Get*",
        "kms:List*",
        "kms:RevokeGrant"
      ],
    }
  ]
}
```



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

保管中のデータの暗号化

暗号化されたファイルシステムは AWS Management Console、AWS CLI、または Amazon EFS API またはいずれかの AWS SDKs を使用してプログラムで作成できます。組織では、特定の分類に合致する、または特定のアプリケーション、ワークロード、環境に関連するすべてのデータを暗号化する必要が生じる場合があります。

EFS ファイルシステムを作成した後、暗号化設定を変更することはできません。つまり、暗号化されていないファイルシステムを暗号化するように変更することはできません。代わりに、暗号化されたファイルシステムを新たに作成する必要があります。

Note

AWS キー管理インフラストラクチャは、連邦情報処理標準 (FIPS) 140-2 で承認された暗号化アルゴリズムを使用します。このインフラストラクチャは、米国標準技術局 (NIST) 800-57 レコメンデーションに一致しています。

保管時に暗号化される EFS ファイルシステムの作成

AWS Identity and Access Management (IAM) アイデンティティベースのポリシーにある `elasticfilesystem:Encrypted` IAM 条件キーを使用して、ユーザーが保管時に暗号化される Amazon EFS ファイルシステムを作成できるかどうか制御します。条件キーの使用に関する詳細については、「[例: 暗号化されたファイルシステムの作成を強制する](#)」を参照してください。

内部で AWS Organizations サービスコントロールポリシー (SCPs) を定義して、組織内のすべての EFS AWS アカウント暗号化を適用することもできます。のサービスコントロールポリシーの詳細については AWS Organizations、「AWS Organizations ユーザーガイド」の「[サービスコントロールポリシー](#)」を参照してください。

コンソールを使用して保管時のファイルシステムを暗号化する

Amazon EFS コンソールを使用して新しいファイルシステムを作成すると、保管時の暗号化がデフォルトで有効になります。

Note

AWS CLI、API、および SDK を使用して新しいファイルシステムを作成すると、保管時の暗号化がデフォルトで有効になりません。詳細については、「[を使用してを作成する AWS CLI](#)」を参照してください。

保存時の暗号化の方法

暗号化されたファイルシステムの場合、データとメタデータはファイルシステムに書き込まれる前に自動的に暗号化されます。同様に、データとメタデータが読み取られると、アプリケーションに提示される前に自動的に復号化されます。このプロセスは Amazon EFS で透過的に処理されるため、アプリケーションを変更する必要はありません。

Amazon EFS は保管時の EFS データおよびメタデータを暗号化するために、業界標準の AES-256 暗号化アルゴリズムを使用します。詳細については、「AWS Key Management Service デベロッパーガイド」の[AWS KMS 「Cryptography essentials」](#)を参照してください。

Encrypting data in transit

EFS ファイルシステムの転送中のデータの暗号化を有効にするには、Amazon EFS マウントヘルパーを使用してファイルシステムをマウントするときに Transport Layer Security (TLS) EFS を有効にします。詳細については、「[EFS マウントヘルパーを使用した EFS ファイルシステムのマウント](#)」を参照してください。

転送時のデータ暗号化が Amazon EFS ファイルシステムのマウントオプションとして宣言されている場合、マウントヘルパーはクライアント stunnel プロセスを初期化します。stunnel はオープンソースの多目的ネットワークリレーです。クライアント stunnel プロセスはインバウンドトラフィックのローカルポートをリッスンし、マウントヘルパーはネットワークファイルシステム (NFS) クライアントトラフィックをこのローカルポートにリダイレクトします。マウントヘルパーはファイルシステムとの通信に TLS バージョン 1.2 を使用します。

転送中の暗号化の動作

転送時のデータの暗号化を有効にするには、TLS を使用して Amazon EFS に接続します。EFS マウントヘルパーを使用してファイルシステムをマウントすることをお勧めします。これは、NFS mount でマウントする場合と比べてマウントプロセスを簡略化できるためです。EFS マウントヘルパーは TLS に stunnel を使用してプロセスを管理します。マウントヘルパーを使用していない場

合でも、転送中のデータの暗号化を有効にすることができます。そうする場合の手順の概略は以下のとおりです。

EFS マウントヘルパーを使用せずに転送中のデータの暗号化を有効にするには

1. `stunnel` をダウンロードしてインストールし、アプリケーションがリッスンするポートを書き留めます。その手順については、「[stunnel のアップグレード](#)」を参照してください。
2. `stunnel` を実行して、TLS を使用してポート 2049 の EFS ファイルシステムに接続します。
3. NFS クライアントを使用して、最初のステップで書き留めたポート `localhost:port` に、`port` をマウントします。

接続ごとに転送中のデータ暗号化が設定されているため、設定された各マウントにはインスタンスで実行される専用の `stunnel` プロセスがあります。デフォルトでは、EFS マウントヘルパーが使用する `stunnel` プロセスは 20449 から 21049 までのローカルポートでリッスンし、ポート 2049 で Amazon EFS に接続します。

Note

デフォルトでは、Amazon EFS マウントヘルパーを TLS で使用するとき、マウントヘルパーによって証明書ホスト名のチェックが適用されます。Amazon EFS マウントヘルパーは、TLS 機能の `stunnel` プログラムを使用します。Linux のバージョンによっては、これらの TLS 機能をサポートする `stunnel` のバージョンがデフォルトで含まれていない場合があります。このような Linux バージョンのいずれかを使用している場合は、TLS を使用した EFS ファイルシステムのマウントに失敗します。

`amazon-efs-utils` パッケージのインストール後に、システムの `stunnel` のバージョンをアップグレードするには、「[stunnel のアップグレード](#)」を参照してください。

暗号化の問題については、「[暗号化のトラブルシューティング](#)」を参照してください。

転送中のデータの暗号化を使用する場合、NFS クライアント設定が変更されます。アクティブにマウントされたファイルシステムを検査する場合、次の例に示すように、`127.0.0.1` または `localhost` にマウントされたことが表示されます。

```
$ mount | column -t
127.0.0.1:/ on /home/ec2-user/efs          type nfs4
(rw,relatime,vers=4.1,rsize=1048576,wsiz=1048576,namlen=255,hard,proto=tcp,port=20127,timeo=600,symlinks)
```

TLS および Amazon EFS マウントヘルパーを使用してマウントする場合、ローカルポートにマウントするために NFS クライアントを再設定します。EFS マウントヘルパーは、このローカルポートをリッスンするクライアント `stunnel` プロセスを開始し、`stunnel` は TLS を使用して、EFS ファイルシステムへの暗号化された接続を開きます。EFS マウントヘルパーは、この暗号化された接続と関連する設定をセットアップして維持します。

どの EFS ファイルシステム ID がどのローカルマウントポイントに対応するかを判断するには、次のコマンドを使用できます。`efs-mount-point` を、ファイルシステムをマウントしたローカルパスに置き換えます。

```
grep -E "Successfully mounted.*efs-mount-point" /var/log/amazon/efs/mount.log | tail -1
```

転送中のデータの暗号化にマウントヘルパーを使用する場合は、`amazon-efs-mount-watchdog` というプロセスも作成されます。このプロセスにより、各マウントの `stunnel` プロセスが実行され、EFS ファイルシステムがアンマウントされると `stunnel` が停止します。何らかの理由で、`stunnel` プロセスが予期せず終了した場合、ウォッチドッグプロセスにより再開されます。

暗号化のトラブルシューティング

以下では、Amazon EFS での暗号化に関する問題のトラブルシューティングについての情報を見つけることができます。

- [転送中のデータの暗号化を行うマウントが失敗する](#)
- [転送中のデータの暗号化を行うマウントが中断する](#)
- [Encrypted-at-rest ファイルシステムを作成できない](#)
- [使用できない暗号化されたファイルシステム](#)

転送中のデータの暗号化を行うマウントが失敗する

デフォルトでは、Amazon EFS マウントヘルパーを Transport Layer Security (TLS) で使用するとき、ホスト名のチェックが強制されます。一部のシステム (Red Hat Enterprise Linux や CentOS など) では、この機能はサポートされません。このような場合は、TLS を使用した EFS ファイルシステムのマウントは失敗します。

実行するアクション

クライアントで `stunnel` のバージョンをアップグレードして、ホスト名のチェックに対応することをお勧めします。詳細については、「[stunnel のアップグレード](#)」を参照してください。

転送中のデータの暗号化を行うマウントが中断する

ごくまれに、Amazon EFS ファイルシステムへの暗号化された接続がハングしたり、クライアント側のイベントによって中断されることがあります。

実行するアクション

転送時のデータ暗号化を使用した Amazon EFS ファイルシステムへの接続が中断される場合、次の手順を実行します。

1. クライアントで stunnel サービスが実行されていることを確認します。
2. ウォッチドッグアプリケーション amazon-efs-mount-watchdog が、クライアントで実行されていることを確認します。このアプリケーションが実行されているかどうかを確認するには、次のコマンドを使用します。

```
ps aux | grep [a]mazon-efs-mount-watchdog
```

3. サポートログを確認します。詳細については、「[サポートログの取得](#)」を参照してください。
4. 必要に応じて、stunnel ログを有効にしてそれらの情報を確認することもできます。/etc/amazon/efs/efs-utils.conf でログの設定を変更して、stunnel ログを有効にできます。ただし、変更を有効にするためには、マウントヘルパーでファイルシステムをアンマウントしてから再マウントする必要があります。

Important

stunnel ログを有効にすると、ファイルシステムのいくらかの容量が使用されます。

中断が続く場合は、AWS サポートにお問い合わせください。

Encrypted-at-rest ファイルシステムを作成できない

保管時に暗号化されるファイルシステムを新しく作成しようとしたが、ただし、AWS KMS が使用できないというエラーメッセージが表示されます。

実行するアクション

このエラーは、で一時的に使用 AWS KMS できなくなったまれなケースで発生する可能性があります AWS リージョン。この場合、AWS KMS が完全な可用性に戻るまで待ってから、ファイルシステムの作成を再試行してください。

使用できない暗号化されたファイルシステム

暗号化されたファイルシステムが一貫して NFS サーバーエラーを返します。これらのエラーは、次のいずれか AWS KMS の理由で EFS が からマスターキーを取得できない場合に発生する可能性があります。

- キーが無効になっています。
- キーが削除されました。
- Amazon EFS がキーを使用するためのアクセス許可が失効しました。
- AWS KMS は一時的に利用できません。

実行するアクション

まず、AWS KMS キーが有効になっていることを確認します。コンソールでキーを表示することで確認できます。詳細については、AWS Key Management Service デベロッパーガイドの「[キーの表示](#)」を参照してください。

キーが有効になっていない場合は、有効化します。詳細については、AWS Key Management Service デベロッパーガイドの「[キーの有効化と無効化](#)」を参照してください。

キーの削除が保留中の場合、このステータスによってキーが無効になります。削除をキャンセルして、キーを再度有効にできます。詳細については、AWS Key Management Service デベロッパーガイドの「[キーの削除をスケジュールし、キャンセルする](#)」を参照してください。

キーが有効で、まだ問題が発生している場合、またはキーの再有効化に問題がある場合は、AWS サポートにお問い合わせください。

Amazon EFS のためのアイデンティティとアクセス管理

AWS Identity and Access Management (IAM) は、管理者が AWS リソースへのアクセスを安全に制御 AWS のサービス するのに役立つです。IAM 管理者は、誰を認証 (サインインを許可) し、誰に Amazon EFS リソースの使用を承認する (アクセス許可を付与する) かを制御します。IAM は、追加料金なしで使用できる AWS のサービス 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](#)
- [Amazon Elastic File System で IAM が機能する仕組み](#)
- [Amazon Elastic File System のアイデンティティベースのポリシーの例](#)
- [Amazon EFS のリソースベースのポリシーの例](#)
- [AWS Amazon EFS の マネージドポリシー](#)
- [Amazon EFS でのタグの使用](#)
- [Amazon EFS のサービスリンクロールの使用](#)
- [Amazon Elastic File System アイデンティティとアクセスのトラブルシューティング](#)

対象者

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

サービスユーザー – ジョブを実行するために Amazon EFS サービスを使用する場合は、管理者から必要なアクセス許可と認証情報が与えられます。さらに多くの Amazon EFS 機能を使用して作業を行うには、追加のアクセス許可が必要になることがあります。アクセスの管理方法を理解すると、管理者に適切なアクセス許可をリクエストするのに役に立ちます。Amazon EFS の機能にアクセスできない場合は、「[Amazon Elastic File System アイデンティティとアクセスのトラブルシューティング](#)」を参照してください。

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

IAM 管理者 - 管理者は、Amazon EFS へのアクセス権を管理するポリシーの作成方法の詳細について確認する場合があります。IAM で使用可能な、Amazon EFS アイデンティティベースのポリシーの例を確認するには、「[Amazon Elastic File System のアイデンティティベースのポリシーの例](#)」を参照してください。

アイデンティティを使用した認証

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

ID ソース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場合、IAM ロールを使用して、前もって管理者により ID フェデレーションが設定されています。フェデレーションを使用して にアクセスすると、間接的 AWS にロールを引き受けれます。

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 Software Development Kit (SDK) とコマンドラインインターフェイス (CLI) AWS を提供し、認証情報を使用してリクエストを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。リクエストに自分で署名する推奨方法の使用については、「IAM ユーザーガイド」の「[API リクエストに対するAWS Signature Version 4](#)」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。たとえば、では、多要素認証 (MFA) を使用してアカウントのセキュリティを向上させる 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](#)」および「IAM ユーザーガイド」の「[IAM のAWS 多要素認証](#)」を参照してください。

AWS アカウント ルートユーザー

を作成するときは AWS アカウント、アカウント内のすべての 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 つのサインインアイデンティティから始めます。この ID は AWS アカウント ルートユーザーと呼ばれ、アカウントの作成に使用した E メールアドレスとパスワードでサインインすることでアクセスできます。日常的なタスクには、ルートユーザーを使用しないことを強くお勧めします。ルートユーザーの認証情報は保護し、ルートユーザーでしか実行できないタスクを実行するときに使用します。ルートユーザーとしてサインインする必要があるタスクの完全なリストについては、「IAM ユーザーガイド」の「[ルートユーザー認証情報が必要なタスク](#)」を参照してください。

フェデレーティッドアイデンティティ

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

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 ID プロバイダー、AWS Directory Service アイデンティティセンターディレクトリ、または ID ソースを通じて提供された認証情報 AWS のサービス を使用して にアクセスする任意のユーザーです。フェデレーティッドアイデンティティがアクセスすると AWS アカウント、ロールを引き受け、ロールは一時的な認証情報を提供します。

アクセスを一元管理する場合は、AWS IAM Identity Centerを使用することをお勧めします。IAM Identity Center でユーザーとグループを作成するか、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、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「[What is IAM Identity Center?](#)」(IAM Identity Center とは) を参照してください。

IAM ユーザーとグループ

[IAM ユーザー](#)は、単一のユーザーまたはアプリケーションに対して特定のアクセス許可 AWS アカウント を持つ 内のアイデンティティです。可能であれば、パスワードやアクセスキーなどの長期的な認証情報を保有する IAM ユーザーを作成する代わりに、一時的な認証情報を使用することをお勧めします。ただし、IAM ユーザーでの長期的な認証情報が必要な特定のユースケースがある場合は、アクセスキーをローテーションすることをお勧めします。詳細については、「IAM ユーザーガイド」の「[長期的な認証情報を必要とするユースケースのためにアクセスキーを定期的にローテーションする](#)」を参照してください。

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

ユーザーは、ロールとは異なります。ユーザーは 1 人の人または 1 つのアプリケーションに一意に関連付けられますが、ロールはそれを必要とする任意の人が引き受けるようになっています。ユーザーには永続的な長期の認証情報がありますが、ロールでは一時認証情報が提供されます。詳細については、「IAM ユーザーガイド」の「[IAM ユーザーに関するユースケース](#)」を参照してください。

IAM ロール

[IAM ロール](#)は、特定のアクセス許可 AWS アカウント を持つ 内のアイデンティティです。これは IAM ユーザーに似ていますが、特定のユーザーには関連付けられていません。で IAM ロールを一時的に引き受けるには AWS Management Console、[ユーザーから IAM ロール \(コンソール\) に切り替える](#)ことができます。ロールを引き受けるには、または AWS API オペレーションを AWS CLI 呼び出すか、カスタム URL を使用します。ロールを使用する方法の詳細については、「IAM ユーザーガイド」の「[ロールを引き受けるための各種方法](#)」を参照してください。

IAM ロールと一時的な認証情報は、次の状況で役立ちます:

- フェデレーションユーザーアクセス – フェデレーティッド 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 \(フェデレーション\) 用のロールを作成する](#)」を参照してください。IAM Identity Center を使用する場合は、許可セットを設定します。アイデンティティが認証後にアクセスできるものを制御するため、IAM Identity Center は、権限セットを IAM のロールに関連付けます。アクセス許可セットの詳細については、「AWS IAM Identity Center User Guide」の「[Permission sets](#)」を参照してください。
- 一時的な IAM ユーザー権限 - IAM ユーザーまたはロールは、特定のタスクに対して複数の異なる権限を一時的に IAM ロールで引き受けることができます。
- クロスアカウントアクセス - IAM ロールを使用して、自分のアカウントのリソースにアクセスすることを、別のアカウントの人物 (信頼済みプリンシパル) に許可できます。クロスアカウントアクセス権を付与する主な方法は、ロールを使用することです。ただし、一部の では AWS のサービス、(ロールをプロキシとして使用する代わりに) リソースに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におけるロールとリソースベースのポリシーの違いについては、「IAM ユーザーガイド」の「[IAM でのクロスアカウントのリソースへのアクセス](#)」を参照してください。
- クロスサービスアクセス – 一部の は他の の機能 AWS のサービス を使用します AWS のサービス。例えば、あるサービスで呼び出しを行うと、通常そのサービスによって Amazon EC2 でアプリケーションが実行されたり、Amazon S3 にオブジェクトが保存されたりします。サービスでは、呼び出し元プリンシパルの許可、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を AWS のサービス、ダウンストリームサービス AWS の

サービス へのリクエストのリクエストと組み合わせて使用します。FAS リクエストは、サービスが他の AWS のサービス またはリソースとのやり取りを完了する必要があるリクエストを受け取った場合にのみ行われます。この場合、両方のアクションを実行するためのアクセス許可が必要です。FAS リクエストを行う際のポリシーの詳細については、「[転送アクセスセッション](#)」を参照してください。

- サービスロール - サービスがユーザーに代わってアクションを実行するために引き受ける [IAM ロール](#)です。IAM 管理者は、IAM 内からサービスロールを作成、変更、削除することができます。詳細については、「IAM ユーザーガイド」の「[AWS のサービスに許可を委任するロールを作成する](#)」を参照してください。
- サービスにリンクされたロール - サービスにリンクされたロールは、にリンクされたサービスロールの一種です AWS のサービス。サービスは、ユーザーに代わってアクションを実行するロールを引き受けることができます。サービスにリンクされたロールは に表示され AWS アカウント、 サービスによって所有されます。IAM 管理者は、サービスリンクロールのアクセス許可を表示できますが、編集することはできません。
- Amazon EC2 で実行されているアプリケーション - IAM ロールを使用して、EC2 インスタンスで実行され、AWS CLI または AWS API リクエストを行うアプリケーションの一時的な認証情報を管理できます。これは、EC2 インスタンス内でのアクセスキーの保存に推奨されます。AWS ロールを EC2 インスタンスに割り当て、そのすべてのアプリケーションで使用できるようにするには、インスタンスにアタッチされたインスタンスプロファイルを作成します。インスタンスプロファイルにはロールが含まれ、EC2 インスタンスで実行されるプログラムは一時的な認証情報を取得できます。詳細については、「IAM ユーザーガイド」の「[Amazon EC2 インスタンスで実行されるアプリケーションに IAM ロールを使用して許可を付与する](#)」を参照してください。

ポリシーを使用したアクセスの管理

でアクセスを制御する AWS には、ポリシーを作成し、ID AWS またはリソースにアタッチします。ポリシーは AWS、アイデンティティまたはリソースに関連付けられているときにアクセス許可を定義する のオブジェクトです。 は、プリンシパル (ユーザー、ルートユーザー、またはロールセッション) がリクエストを行うときに、これらのポリシー AWS を評価します。ポリシーでの権限により、リクエストが許可されるか拒否されるかが決まります。ほとんどのポリシーは JSON ドキュメント AWS として に保存されます。JSON ポリシードキュメントの構造と内容の詳細については、IAM ユーザーガイドの [JSON ポリシー概要](#)を参照してください。

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

デフォルトでは、ユーザーやロールに権限はありません。IAM 管理者は、リソースで必要なアクションを実行するための権限をユーザーに付与する IAM ポリシーを作成できます。その後、管理者はロールに IAM ポリシーを追加し、ユーザーはロールを引き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方法を問わず、アクションの許可を定義します。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 API からロール情報を取得できます。

アイデンティティベースのポリシー

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

アイデンティティベースのポリシーは、さらにインラインポリシーまたはマネージドポリシーに分類できます。インラインポリシーは、単一のユーザー、グループ、またはロールに直接埋め込まれています。管理ポリシーは、内の複数のユーザー、グループ、ロールにアタッチできるスタンドアロンポリシーです AWS アカウント。管理ポリシーには、AWS 管理ポリシーとカスタマー管理ポリシーが含まれます。マネージドポリシーまたはインラインポリシーのいずれかを選択する方法については、「IAM ユーザーガイド」の「[管理ポリシーとインラインポリシーのいずれかを選択する](#)」を参照してください。

リソースベースのポリシー

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

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

アクセスコントロールリスト (ACL)

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

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

その他のポリシータイプ

AWS は、追加のあまり一般的ではないポリシータイプをサポートしています。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ポリシーによって IAM エンティティ (IAM ユーザーまたはロール) に付与できる権限の上限を設定する高度な機能です。エンティティにアクセス許可の境界を設定できます。結果として得られる権限は、エンティティのアイデンティティベースポリシーとそのアクセス許可の境界の共通部分になります。Principal フィールドでユーザーまたはロールを指定するリソースベースのポリシーでは、アクセス許可の境界は制限されません。これらのポリシーのいずれかを明示的に拒否した場合、権限は無効になります。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 (SCP) を一部またはすべてのアカウントに適用できます。SCP は、各 を含むメンバーアカウントのエンティティのアクセス許可を制限します AWS アカウントのルートユーザー。Organizations と SCP の詳細については、「AWS Organizations ユーザーガイド」の「[サービスコントロールポリシー \(SCP\)](#)」を参照してください。
- **リソースコントロールポリシー (RCP)** - RCP は、所有する各リソースにアタッチされた IAM ポリシーを更新することなく、アカウント内のリソースに利用可能な最大数のアクセス許可を設定するために使用できる JSON ポリシーです。RCP は、メンバーアカウントのリソースに対するアクセス許可を制限し、組織に属しているかどうかにかかわらず AWS アカウントのルートユーザー、を含む ID に対する有効なアクセス許可に影響を与える可能性があります。RCP AWS のサービスをサポートする のリストなど、Organizations と RCPs「[リソースコントロールポリシー \(RCPs\)](#)」を参照してください。AWS Organizations

- セッションポリシー - セッションポリシーは、ロールまたはフェデレーションユーザーの一時的なセッションをプログラムで作成する際にパラメータとして渡す高度なポリシーです。結果としてセッションの権限は、ユーザーまたはロールのアイデンティティベースポリシーとセッションポリシーの共通部分になります。また、リソースベースのポリシーから権限が派生する場合もあります。これらのポリシーのいずれかを明示的に拒否した場合、権限は無効になります。詳細については、「IAM ユーザーガイド」の「[セッションポリシー](#)」を参照してください。

複数のポリシータイプ

1 つのリクエストに複数のタイプのポリシーが適用されると、結果として作成される権限を理解するのがさらに難しくなります。複数のポリシータイプが関係してい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。

Amazon Elastic File System で IAM が機能する仕組み

IAM を使用して Amazon EFS へのアクセスを管理する前に、Amazon EFS で使用できる IAM 機能について理解しておく必要があります。

Amazon Elastic File System で使用できる IAM の機能

IAM 機能	Amazon EFS のサポート
アイデンティティベースポリシー	はい
リソースベースのポリシー	はい
ポリシーアクション	はい
ポリシーリソース	はい
ポリシー条件キー (サービス固有)	はい
ACL	いいえ
ABAC (ポリシー内のタグ)	部分的
一時的な認証情報	はい

IAM 機能	Amazon EFS のサポート
プリンシパル権限	はい
サービスロール	はい
サービスリンクロール	はい

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

Amazon EFS のアイデンティティベースの ポリシー

アイデンティティベースのポリシーのサポート: あり

アイデンティティベースポリシーは、IAM ユーザーグループ、ユーザーのグループ、ロールなど、アイデンティティにアタッチできる JSON 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 ベースのポリシーの作成方法については、「IAM ユーザーガイド」の「[カスタマー管理ポリシーでカスタム IAM アクセス許可を定義する](#)」を参照してください。

IAM アイデンティティベースのポリシーでは、許可または拒否するアクションとリソース、およびアクションを許可または拒否する条件を指定できます。プリンシパルは、それが添付されているユーザーまたはロールに適用されるため、アイデンティティベースのポリシーでは指定できません。JSON ポリシーで使用するすべての要素について学ぶには、「IAM ユーザーガイド」の「[IAM JSON ポリシーの要素のリファレンス](#)」を参照してください。

Amazon EFS のアイデンティティベースのポリシー例

Amazon EFS のアイデンティティベースポリシーの例を確認するには、「[Amazon Elastic File System のアイデンティティベースのポリシーの例](#)」を参照してください。

Amazon EFS 内のリソースベースのポリシー

リソースベースのポリシーのサポート: あり

リソースベースのポリシーは、リソースに添付する JSON ポリシードキュメントです。リソースベースのポリシーには例として、IAM ロールの信頼ポリシーや Amazon S3 バケットポリシーがあげられます。リソースベースのポリシーをサポートするサービスでは、サービス管理者はポリシーを

使用して特定のリソースへのアクセスを制御できます。ポリシーがアタッチされているリソースの場合、指定されたプリンシパルがそのリソースに対して実行できるアクションと条件は、ポリシーによって定義されます。リソースベースのポリシーでは、[プリンシパルを指定する](#)必要があります。プリンシパルには、アカウント、ユーザー、ロール、フェデレーティッドユーザー、またはを含めることができます AWS のサービス。

クロスアカウントアクセスを有効にするには、アカウント全体、または別のアカウントの IAM エンティティをリソースベースのポリシーのプリンシパルとして指定します。リソースベースのポリシーにクロスアカウントのプリンシパルを追加しても、信頼関係は半分しか確立されない点に注意してください。プリンシパルとリソースが異なる場合 AWS アカウント、信頼されたアカウントの IAM 管理者は、プリンシパルエンティティ (ユーザーまたはロール) 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。IAM 管理者は、アイデンティティベースのポリシーをエンティティにアタッチすることで権限を付与します。ただし、リソースベースのポリシーで、同じアカウントのプリンシパルへのアクセス権が付与されている場合は、アイデンティティベースのポリシーをさらに付与する必要はありません。詳細については、「IAM ユーザーガイド」の「[IAM でのクロスアカウントリソースアクセス](#)」を参照してください。

リソースポリシーを使用してファイルシステムのデータアクセスを制御する方法については、「[IAM を使用してファイルシステムへのアクセスを制御する](#)」を参照してください。リソースベースのポリシーをファイルシステムにアタッチする方法については、「[ファイルシステムポリシーの作成](#)」を参照してください。

Amazon EFS 内のリソースベースのポリシーの例

Amazon EFS リソースベースのポリシーの例を表示するには、「[Amazon EFS のリソースベースのポリシーの例](#)」を参照してください。

Amazon EFS のポリシーアクション

ポリシーアクションのサポート:あり

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

JSON ポリシーの Action 要素にはポリシー内のアクセスを許可または拒否するために使用できるアクションが記述されます。ポリシーアクションの名前は通常、関連付けられた AWS API オペレーションと同じです。一致する API オペレーションのない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アクセス許可を付与するポリシーで使用されます。

Amazon EFS のアクションのリストを確認するには、「サービス認可リファレンス」の「[Amazon Elastic File System で定義されるアクション](#)」を参照してください。

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

```
elasticfilesystem
```

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

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

Amazon EFS のアイデンティティベースポリシーの例を確認するには、「[Amazon Elastic File System のアイデンティティベースのポリシーの例](#)」を参照してください。

Amazon EFS のポリシーリソース

ポリシーリソースのサポート: あり

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

Resource JSON ポリシー要素はアクションが適用されるオブジェクトを指定します。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、[アマゾン 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": "*"
```

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

Amazon EFS のアイデンティティベースポリシーの例を確認するには、「[Amazon Elastic File System のアイデンティティベースのポリシーの例](#)」を参照してください。

Amazon EFS のポリシー条件キー

サービス固有のポリシー条件キーのサポート: あり

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

Condition 要素 (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して、ポリシーの条件とリクエスト内の値を一致させることができます。

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 ユーザーに、IAM ユーザー名がタグ付けされている場合のみリソースにアクセスできる権限を付与することができます。詳細については、「IAM ユーザーガイド」の「[IAM ポリシーの要素: 変数およびタグ](#)」を参照してください。

AWS は、グローバル条件キーとサービス固有の条件キーをサポートしています。すべての AWS グローバル条件キーを確認するには、IAM ユーザーガイドの [AWS 「グローバル条件コンテキストキー」](#) を参照してください。

Amazon EFS の条件キーのリストを確認するには、「サービス認可リファレンス」の「[Amazon Elastic File System の条件キー](#)」を参照してください。条件キーを使用できるアクションとリソースについては、「[Amazon Elastic File System で定義されるアクション](#)」を参照してください。

Amazon EFS のアイデンティティベースポリシーの例を確認するには、「[Amazon Elastic File System のアイデンティティベースのポリシーの例](#)」を参照してください。

Amazon EFS での ACL

ACL のサポート: なし

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

Amazon EFS での ABAC

ABAC (ポリシー内のタグ) のサポート: 一部

属性ベースのアクセス制御 (ABAC) は、属性に基づいてアクセス許可を定義する認可戦略です。では AWS、これらの属性はタグと呼ばれます。タグは、IAM エンティティ (ユーザーまたはロール) および多くの AWS リソースにアタッチできます。エンティティとリソースのタグ付けは、ABAC の最初の手順です。その後、プリンシパルのタグがアクセスしようとしているリソースのタグと一致した場合にオペレーションを許可するように ABAC ポリシーをします。

ABAC は、急成長する環境やポリシー管理が煩雑になる状況で役立ちます。

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

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

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

Amazon EFS での一時的な認証情報の使用

一時的な認証情報のサポート: あり

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

ユーザー名とパスワード以外の AWS Management Console 方法でサインインする場合は、一時的な認証情報を使用します。たとえば、会社のシングルサインオン (SSO) リンク AWS を使用してアクセスすると、そのプロセスによって一時的な認証情報が自動的に作成されます。また、ユーザーとしてコンソールにサインインしてからロールを切り替える場合も、一時的な認証情報が自動的に作成されます。ロールの切り替えに関する詳細については、「IAM ユーザーガイド」の「[ユーザーから IAM ロールに切り替える \(コンソール\)](#)」を参照してください。

一時的な認証情報は、AWS CLI または AWS API を使用して手動で作成できます。その後、これらの一時的な認証情報を使用してアクセスすることができます AWS。長期的なアクセスキーを使用する代わりに、一時的な認証情報を動的に生成 AWS することをお勧めします。詳細については、「[IAM の一時的セキュリティ認証情報](#)」を参照してください。

Amazon EFS のクロスサービスプリンシパル許可

転送アクセスセッション (FAS) のサポート: あり

IAM ユーザーまたはロールを使用して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、呼び出すプリンシパルのアクセス許可を AWS のサービス、ダウンストリームサービス AWS のサービス へのリクエストのリクエストと組み合わせて使用します。FAS リクエストは、サービスが他の AWS のサービス またはリソースとのやり取りを完了する必要があるリクエストを受け取った場合にのみ行われます。この場合、両方のアクションを実行するためのアクセス許可が必要です。FAS リクエストを行う際のポリシーの詳細については、「[転送アクセスセッション](#)」を参照してください。

Amazon EFS のサービスロール

サービスロールのサポート: あり

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

Warning

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

Amazon EFS のサービスにリンクされたロール

サービスリンクロールのサポート: あり

サービスにリンクされたロールは、にリンクされたサービスロールの一種です AWS のサービス。サービスは、ユーザーに代わってアクションを実行するロールを引き受けることができます。サービスにリンクされたロールは に表示され AWS アカウント、 サービスによって所有されます。IAM 管理者は、サービスリンクロールのアクセス許可を表示できますが、編集することはできません。

Amazon EFS でのサービスにリンクされたロールの作成または管理の詳細については、「[Amazon EFS のサービスリンクロールの使用](#)」を参照してください。

Amazon Elastic File System のアイデンティティベースのポリシーの例

デフォルトでは、ユーザーとロールには Amazon EFS 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 Command Line Interface (AWS CLI) AWS Management Console、または AWS API を使用してタスクを実行することはできません。IAM 管理者は、リソースに必要なアクションを実行するための権限をユーザーに付与する IAM ポリシーを作成できます。その後、管理者はロールに IAM ポリシーを追加し、ユーザーはロールを引き継ぐことができます。

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

Amazon EFS で定義されるアクションとリソースタイプ (リソースタイプごとの ARN の形式を含む)の詳細については、「サービス認可リファレンス」の「[Amazon Elastic File System のアクション、リソース、条件キー](#)」を参照してください。

トピック

- [ポリシーに関するベストプラクティス](#)
- [Amazon EFS コンソールの使用](#)
- [例: ユーザーにそれぞれのアクセス権限の表示を許可する](#)
- [例: 暗号化されたファイルシステムの作成を強制する](#)
- [例: 暗号化されていないファイルシステムの作成を強制する](#)

ポリシーに関するベストプラクティス

ID ベースのポリシーは、ユーザーのアカウント内で誰かが Amazon EFS 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 管理ポリシーを開始し、最小特権のアクセス許可に移行する – ユーザーとワークロードにアクセス許可を付与するには、多くの一般的なユースケースにアクセス許可を付与するAWS 管理ポリシーを使用します。これらは 使用できます AWS アカウント。ユースケースに固有の AWS カスタマー管理ポリシーを定義することで、アクセス許可をさらに減らすことをお勧めします。詳細については、「IAM ユーザーガイド」の「[AWS マネージドポリシー](#)」または「[ジョブ機能のAWS マネージドポリシー](#)」を参照してください。
- 最小特権を適用する – IAM ポリシーで許可を設定する場合は、タスクの実行に必要な許可のみを付与します。これを行うには、特定の条件下で特定のリソースに対して実行できるアクションを定義します。これは、最小特権アクセス許可とも呼ばれています。IAM を使用して許可を適用する方法の詳細については、「IAM ユーザーガイド」の「[IAM でのポリシーとアクセス許可](#)」を参照してください。
- IAM ポリシーで条件を使用してアクセスをさらに制限する – ポリシーに条件を追加して、アクションやリソースへのアクセスを制限できます。例えば、ポリシー条件を記述して、すべてのリクエストを SSL を使用して送信するように指定できます。条件を使用して、サービスアクションが などの特定の を通じて使用されている場合に AWS のサービス、サービスアクションへのアクセスを許可することもできます AWS CloudFormation。詳細については、「IAM ユーザーガイド」の「[IAM JSON ポリシー要素:条件](#)」を参照してください。
- IAM Access Analyzer を使用して IAM ポリシーを検証し、安全で機能的な権限を確保する – IAM Access Analyzer は、新規および既存のポリシーを検証して、ポリシーが IAM ポリシー言語 (JSON) および IAM のベストプラクティスに準拠するようにします。IAM アクセスアナライザーは 100 を超えるポリシーチェックと実用的な推奨事項を提供し、安全で機能的なポリシーの作成をサポートします。詳細については、「IAM ユーザーガイド」の「[IAM Access Analyzer でポリシーを検証する](#)」を参照してください。
- 多要素認証 (MFA) を要求する – IAM ユーザーまたはルートユーザーを必要とするシナリオがある場合は AWS アカウント、セキュリティを強化するために MFA を有効にします。API オペレーションが呼び出されるときに MFA を必須にするには、ポリシーに MFA 条件を追加します。詳細については、「IAM ユーザーガイド」の「[MFA を使用した安全な API アクセス](#)」を参照してください。

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

Amazon EFS コンソールの使用

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

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

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

`AmazonElasticFileSystemReadOnlyAccess` およびその他の [AWS Amazon EFS の マネージドポリシー](#) の Amazon EFS マネージドサービスポリシーが表示されます。

例: ユーザーにそれぞれのアクセス権限の表示を許可する

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

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ViewOwnUserInfo",
      "Effect": "Allow",
      "Action": [
        "iam:GetUserPolicy",
        "iam:ListGroupsForUser",
        "iam:ListAttachedUserPolicies",
        "iam:ListUserPolicies",
        "iam:GetUser"
      ],
      "Resource": ["arn:aws:iam::*:user/${aws:username}"]
    }
  ]
}
```

```

    },
    {
      "Sid": "NavigateInConsole",
      "Effect": "Allow",
      "Action": [
        "iam:GetGroupPolicy",
        "iam:GetPolicyVersion",
        "iam:GetPolicy",
        "iam:ListAttachedGroupPolicies",
        "iam:ListGroupPolicies",
        "iam:ListPolicyVersions",
        "iam:ListPolicies",
        "iam:ListUsers"
      ],
      "Resource": "*"
    }
  ]
}

```

例: 暗号化されたファイルシステムの作成を強制する

次の例は、プリンシパルに暗号化されたファイルシステムのみを作成することを許可するアイデンティティベースのポリシーを示しています。

```

{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "elasticfilesystem:CreateFileSystem",
      "Condition": {
        "Bool": {
          "elasticfilesystem:Encrypted": "true"
        }
      },
      "Resource": "*"
    }
  ]
}

```

このポリシーが暗号化されていないファイルシステムを作成しようとするユーザーに割り当てられている場合、リクエストは失敗します。ユーザーは、、、AWS API または SDK AWS Management Consoleを使用しているかどうかにかかわらず AWS CLI、次のようなメッセージを表示します。


```
User: arn:aws:iam::111122223333:user/username is not authorized to
perform: elasticfilesystem:CreateFileSystem on the specified resource.
```

例: 暗号化されていないファイルシステムの作成を強制する

次の例は、プリンシパルが暗号化されていないファイルシステムのみを作成することを許可するアイデンティティベースのポリシーを示しています。

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "elasticfilesystem:CreateFileSystem",
      "Condition": {
        "Bool": {
          "elasticfilesystem:Encrypted": "false"
        }
      },
      "Resource": "*"
    }
  ]
}
```

このポリシーが暗号化されたファイルシステムを作成しようとするユーザーに割り当てられている場合、リクエストは失敗します。ユーザーは、、、AWS API または SDK AWS Management Console を使用しているかどうかにかかわらず AWS CLI、次のようなメッセージを表示します。

```
User: arn:aws:iam::111122223333:user/username is not authorized to
perform: elasticfilesystem:CreateFileSystem on the specified resource.
```

また、AWS Organizations サービスコントロールポリシー (SCP) を作成することで、暗号化または暗号化されていない Amazon EFS ファイルシステムの作成を強制することもできます。のサービスコントロールポリシーの詳細については AWS Organizations、「AWS Organizations ユーザーガイド」の「[サービスコントロールポリシー](#)」を参照してください。

Amazon EFS のリソースベースのポリシーの例

このセクションでは、さまざまな Amazon EFS アクションに対してアクセス許可を付与または拒否するファイルシステムポリシーの例を示します。Amazon EFS ファイルシステムポリシーには

20,000 文字の制限があります。リソースベースのポリシーの要素については、「[Amazon EFS 内のリソースベースのポリシー](#)」を参照してください。

⚠ Important

ファイルシステムポリシーで個々の IAM ユーザーまたはロールにアクセス許可を付与する場合、ポリシーがファイルシステムで有効な間は、そのユーザーまたはロールを削除または再作成しないでください。そのような操作を行うと、そのユーザーまたはロールはファイルシステムから事実上ロックアウトされ、アクセスできなくなります。詳細については、IAM ユーザーガイドの「[プリンシパルの指定](#)」を参照してください。

ファイルシステムポリシーの作成方法の詳細については、「[ファイルシステムポリシーの作成](#)」を参照してください。

トピック

- [例: 特定の AWS ロールへの読み取りおよび書き込みアクセスを許可する](#)
- [例: 読み取り専用アクセスの付与](#)
- [例: クロスアカウントレプリケーションを設定した後、接続されたクライアントがアクセスを保持していることを確認する](#)
- [例: EFS アクセスポイントにアクセスを許可する](#)

例: 特定の AWS ロールへの読み取りおよび書き込みアクセスを許可する

この例では、EFS ファイルシステムポリシーには、以下のような特徴があります。

- 効果は Allow です。
- プリンシパルは、AWS アカウントの Testing_Role に設定されています。
- アクションは ClientMount (読み取り)、および ClientWrite に設定されます。
- アクセス許可を付与する条件は AccessedViaMountTarget に設定されています。

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
```

```

        "AWS": "arn:aws:iam::111122223333:role/Testing_Role"
    },
    "Action": [
        "elasticfilesystem:ClientWrite",
        "elasticfilesystem:ClientMount"
    ],
    "Resource": "arn:aws:elasticfilesystem:us-east-2:111122223333:file-system/
fs-1234abcd",
    "Condition": {
        "Bool": {
            "elasticfilesystem:AccessedViaMountTarget": "true"
        }
    }
}
]
}

```

例: 読み取り専用アクセスの付与

次のファイルシステムポリシーはClientMount、IAM ロールに または読み取り専用EfsReadOnlyのアクセス許可のみを付与します。

```

{
    "Id": "read-only-example-policy02",
    "Statement": [
        {
            "Sid": "efs-statement-example02",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:role/EfsReadOnly"
            },
            "Action": [
                "elasticfilesystem:ClientMount"
            ],
            "Resource": "arn:aws:elasticfilesystem:us-east-2:111122223333:file-system/
fs-12345678"
        }
    ]
}

```

特定の管理ワークステーションを除くすべての IAM プリンシパルに対してルートアクセスを拒否するなど、追加のファイルシステムポリシーを設定する方法については、「[NFS クライアントの IAM 認可を使用したルートスカッシュの有効化](#)」を参照してください。

例: クロスアカウントレプリケーションを設定した後、接続されたクライアントがアクセスを保持していることを確認する

次のリソースベースのポリシーを使用して、ファイルシステムに接続されているすべてのクライアントが、ファイルシステムのクロスアカウントレプリケーションを設定した後もアクセスを保持できます。クロスアカウントレプリケーションの詳細については、「」を参照してください。[アカウント間で AWS EFS ファイルシステムをレプリケートする](#)

ポリシーを作成するときは、次の要件が適用されます。

- EFS マウントヘルパーを使用してファイルシステムをマウントします。ファイルシステムが NFS クライアントを使用してマウントされている場合、接続されたクライアントはサーバーエラーによってアクセスが拒否されます。
- mount コマンドで `-o iam` または `-o EFS tls` オプションを使用して、EFS マウントターゲットに認証情報を渡します。

```
{
  "Version": "2012-10-17",
  "Id": "efs-policy-wizard-15ad9567-2546-4bbb-8168-5541b6fc0e55",
  "Statement": [
    {
      "Sid": "efs-statement-14a7191c-9401-40e7-a388-6af6cfb7dd9c",
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
      },
      "Action": [
        "elasticfilesystem:ClientMount",
        "elasticfilesystem:ClientWrite",
        "elasticfilesystem:ClientRootAccess"
      ],
      "Resource": "arn:aws:elasticfilesystem:us-east-2:111122223333:file-system/fs-1234abcd",
      "Condition": {
        "Bool": {
          "elasticfilesystem:AccessedViaMountTarget": "true"
        }
      }
    }
  ]
}
```

```

    }
  }
}
]
}

```

例: EFS アクセスポイントにアクセスを許可する

EFS アクセスポリシーを使用して、EFS ファイルシステムの共有ファイルベースのデータセットに関するアプリケーション固有のビューを、NFS クライアントに提供します。ファイルシステムポリシーを使用して、ファイルシステムへのアクセス許可をアクセスポイントに付与します。

このファイルポリシーの例では、条件要素を使用して、ARN によって識別された特定のアクセスポイントに対してファイルシステムへのフルアクセスを許可します。

EFS アクセスポイントの使用方法的詳細については、「[アクセスポイントの使用](#)」を参照してください。

```

{
  "Id": "access-point-example03",
  "Statement": [
    {
      "Sid": "access-point-statement-example03",
      "Effect": "Allow",
      "Principal": {"AWS": "arn:aws:iam::555555555555:role/EfsAccessPointFullAccess"},
      "Action": "elasticfilesystem:Client*",
      "Resource": "arn:aws:elasticfilesystem:us-east-2:111122223333:file-system/fs-12345678",
      "Condition": {
        "StringEquals": {
          "elasticfilesystem:AccessPointArn": "arn:aws:elasticfilesystem:us-east-2:555555555555:access-point/fsap-12345678" }
        }
      }
    ]
  }
}

```

AWS Amazon EFS の マネージドポリシー

AWS 管理ポリシーは、によって作成および管理されるスタンドアロンポリシーです AWS。AWS 管理ポリシーは、ユーザー、グループ、ロールにアクセス許可の割り当てを開始できるように、多くの一般的なユースケースにアクセス許可を提供するように設計されています。

AWS 管理ポリシーは、すべての AWS お客様が使用できるため、特定のユースケースに対して最小特権のアクセス許可を付与しない場合があります。ユースケースに固有の[カスタマー管理ポリシー](#)を定義して、アクセス許可を絞り込むことをお勧めします。

AWS 管理ポリシーで定義されているアクセス許可は変更できません。が AWS マネージドポリシーで定義されたアクセス許可 AWS を更新すると、ポリシーがアタッチされているすべてのプリンシパル ID (ユーザー、グループ、ロール) に影響します。AWS は、新しい が起動されるか、新しい API オペレーション AWS のサービス が既存のサービスで使えるようになったときに、AWS マネージドポリシーを更新する可能性が高くなります。

詳細については「IAM ユーザーガイド」の「[AWS マネージドポリシー](#)」を参照してください。

AWS マネージドポリシー: AWSServiceRoleForAmazonElasticFileSystem

Amazon EFS は、という名前のサービスにリンクされたロール AWSServiceRoleForAmazonElasticFileSystemを使用して、Amazon EFS がユーザーに代わって AWS リソースを管理できるようにします。このロールは elasticfilesystem.amazonaws.com サービスを信頼してそのロールを引き受けます。詳細については、「[Amazon EFS のサービスリンクロールの使用](#)」を参照してください。

AWS マネージドポリシー: AmazonElasticFileSystemFullAccess

AmazonElasticFileSystemFullAccess ポリシーは IAM ID にアタッチできます。

このポリシーは、Amazon EFS へのフルアクセスと、を介した関連 AWS サービスへのアクセスを許可する管理アクセス許可を付与します AWS Management Console。

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。

- elasticfilesystem – プリンシパルが Amazon EFS コンソールですべてのアクションを実行することを許可します。また、プリンシパルが を使用してバックアップを作成 (elasticfilesystem:Backup) および復元 (elasticfilesystem:Restore) することもできます AWS Backup。

- `cloudwatch` – プリンシパルが Amazon EFS コンソールでメトリックスの Amazon CloudWatch ファイルシステムのメトリックスとアラームを記述できるようにします。
- `ec2` – Amazon EFS コンソールで、プリンシパルがネットワークインターフェイスの作成、削除、記述、ネットワークインターフェイス属性の記述と変更、アベイラビリティーゾーン、セキュリティグループ、サブネット、仮想プライベートクラウド (VPCs)、および EFS ファイルシステムに関連付けられた VPC 属性を記述できるようにします。
- `kms` – プリンシパルが AWS Key Management Service (AWS KMS) キーのエイリアスを一覧表示し、Amazon EFS コンソールで KMS キーを記述できるようにします。
- `iam` – Amazon EFS がユーザーに代わって AWS リソースを管理できるようにするサービスにリンクされたロールを作成するアクセス許可を付与します。
- `iam:PassRole` – Amazon EFS に IAM ロールを渡すアクセス許可を付与します。

このポリシーのアクセス許可を確認するには、AWS 「マネージドポリシーリファレンスガイド [AmazonElasticFileSystemFullAccess](#)」の「」を参照してください。

AWS マネージドポリシー: AmazonElasticFileSystemReadOnlyAccess

AmazonElasticFileSystemReadOnlyAccess ポリシーは IAM ID にアタッチできます。

このポリシーは、を介して Amazon EFS への読み取り専用アクセスを許可します AWS Management Console。

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。

- `elasticfilesystem` – プリンシパルは、Amazon EFS コンソールのアカウント設定、バックアップおよびファイルシステムポリシー、ライフサイクル設定、マウントターゲットとそのセキュリティグループ、タグ、アクセスポイントなど、Amazon EFS ファイルシステムの属性を記述できるようにします。
- `cloudwatch` – プリンシパルが CloudWatch メトリクスを取得し、Amazon EFS コンソールでメトリックスのアラームを記述できるようにします。
- `ec2` – Amazon EFS コンソールで、プリンシパルがアベイラビリティーゾーン、ネットワークインターフェイスとその属性、セキュリティグループ、サブネット、VPC、およびそれらの属性を表示できるようにします。

- kms – プリンシパルが Amazon EFS コンソールで AWS KMS キーのエイリアスを一覧表示できるようにします。

このポリシーのアクセス許可を確認するには、AWS 「マネージドポリシーリファレンスガイド」[AmazonElasticFileSystemReadOnlyAccess](#) の「」を参照してください。

AWS マネージドポリシー: AmazonElasticFileSystemClientFullAccess

IAM エンティティに、AmazonElasticFileSystemClientFullAccess ポリシーをアタッチできます。

このポリシーはEFS ファイルシステムへの読み取りおよび書き込みクライアントアクセスを許可します。このポリシーにより、NFS クライアントは EFS ファイルシステムをマウント、読み取り、書き込みできます。

このポリシーのアクセス許可を確認するには、AWS 「マネージドポリシーリファレンスガイド」[AmazonElasticFileSystemFullAccess](#) の「」を参照してください。

AWS マネージドポリシー: AmazonElasticFileSystemClientReadWriteAccess

IAM エンティティに、AmazonElasticFileSystemClientReadWriteAccess ポリシーをアタッチできます。

このポリシーはEFS ファイルシステムへの読み取りおよび書き込みクライアントアクセスを許可します。このポリシーにより、NFS クライアントは EFS ファイルシステムをマウント、読み取り、書き込みできます。

このポリシーのアクセス許可を確認するには、AWS 「マネージドポリシーリファレンスガイド」[AmazonElasticFileSystemClientReadWriteAccess](#) の「」を参照してください。

AWS マネージドポリシーに対する Amazon EFS の更新

このサービスがこれらの変更の追跡を開始してからの Amazon EFS 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。このページへの変更に関する自動アラートについては、Amazon EFSページのRSSフィードを購読してください。

変更	説明	日付
既存のポリシーの更新	ポリシー: AmazonElasticFileSystemFullAccess	2024 年 11 月 7 日

変更	説明	日付
	Amazon EFS は以下を追加しました。 • ReplicationRead および ReplicationWrite は、レプリケーション用のファイルシステムデータを読み書きするアクセス許可を付与します。 • iam:PassRole レプリケーション設定を作成するアクセス許可を Amazon EFS に付与します。	
既存のポリシーの更新	ポリシー: AmazonElasticFileSystemServiceRolePolicy Amazon EFS に ReplicationRead と が追加されReplicationWrite 、レプリケーション用のファイルシステムデータの読み取りと書き込みのアクセス許可が付与されました。	2024 年 11 月 7 日
既存のポリシーの更新	ポリシー: AmazonElasticFileSystemReadOnlyAccess Amazon EFS は、レプリケーション用のファイルシステムデータを読み取るアクセス許可を付与する ReplicationRead アクションを追加しました。	2024 年 11 月 7 日
既存のポリシーの更新	ポリシー: AmazonElasticFileSystemReadOnlyAccess Amazon EFS は、ソースアカウントと宛先アカウントにクロスアカウントレプリケーション用のファイルシステムへのアクセスを許可する新しいアクセス許可を追加しました。	2024 年 8 月 7 日

変更	説明	日付
既存のポリシーの更新	ポリシー: AmazonElasticFileSystemFullAccess Amazon EFS には、プリンシパルがファイルシステムの保護を無効または有効にすることを可能にする新しいアクセス許可が追加されました。このアクセス許可は、Amazon EFS が既存のファイルシステムにレプリケートできるようにするために必要です。	2023 年 11 月 27 日
既存のポリシーの更新	ポリシー: AmazonElasticFileSystemServiceRolePolicy Amazon EFS では、プリンシパルが Amazon EFS レプリケーションの作成、説明、および削除と Amazon EFS ファイルシステムを作成できるようにするための新しいアクセス許可が追加されました。Amazon EFS がユーザーに代わってファイルシステムのレプリケーション設定を管理できるようにするには、アクセス許可が必要です。	2022 年 1 月 25 日
既存のポリシーの更新	ポリシー: AmazonElasticFileSystemReadOnlyAccess Amazon EFS では、プリンシパルが Amazon EFS レプリケーションを記述できるようにするための新しいアクセス許可が追加されました。このアクセス許可は、ユーザーがファイルシステムのレプリケーション設定を表示できるようにするために必要です。	2022 年 1 月 25 日

変更	説明	日付
既存のポリシーの更新	ポリシー: AmazonElasticFileSystemFullAccess Amazon EFS では、プリンシパルが Amazon EFS レプリケーションを作成、記述、および削除することを可能にする新しいアクセス許可が追加されました。このアクセス許可は、ユーザーがファイルシステムのレプリケーション設定を管理できるようにするために必要です。	2022 年 1 月 25 日
追跡ポリシーを開始	ポリシー: AmazonElasticFileSystemClientReadWriteAccess Amazon EFS ファイルシステムの読み取りおよび書き込み権限を NFS クライアントに付与します。	2022 年 1 月 3 日
追跡ポリシーを開始	ポリシー: AmazonElasticFileSystemServiceRolePolicy Amazon EFS のサービスリンクロール許可	2021 年 10 月 8 日
既存のポリシーの更新	ポリシー: AmazonElasticFileSystemFullAccess Amazon EFS は、プリンシパルが Amazon EFS アカウントの設定を変更および説明できるようにするための新しいアクセス許可を追加しました。アクセス許可は、ユーザーが Amazon EFS コンソールでアカウント設定を表示および設定できるようにするために必要です。	2021 年 5 月 7 日
既存のポリシーの更新	ポリシー: AmazonElasticFileSystemReadOnlyAccess Amazon EFS は、プリンシパルが Amazon EFS アカウントの設定を記述できるようにするための新しいアクセス許可を追加しました。アクセス許可は、ユーザーが Amazon EFS コンソールでアカウント設定を表示できるようにするために必要です。	2021 年 5 月 7 日

変更	説明	日付
Amazon EFS が変更の追跡を開始しました。	Amazon EFS は、AWS 管理ポリシーの変更の追跡を開始しました。	2021 年 5 月 7 日

Amazon EFS でのタグの使用

タグを使用すると、Amazon EFS リソースへのアクセスをコントロールしたり、属性ベースのアクセスコントロール (ABAC) を実装したりできます。詳細については、以下を参照してください。

- [EFS リソースのタグ付け](#)
- [リソースのタグに基づいてアクセスをコントロールする](#)
- IAM ユーザーガイド [AWS の ABAC とは](#)

Note

Amazon EFS レプリケーションは、属性ベースのアクセス制御 (ABAC) でのタグの使用をサポートしていません。

作成中に Amazon EFS リソースにタグを適用するには、ユーザーは特定の AWS Identity and Access Management (IAM) 許可を持っている必要があります。

リソース作成時にタグ付けするアクセス許可の付与

次の作成時のタグ付けの Amazon EFS API アクションを使用すると、リソースの作成時にタグを指定できます。

- CreateAccessPoint
- CreateFileSystem

ユーザーが作成時にリソースにタグを付けることができるようにするに

は、elasticfilesystem:CreateAccessPoint や elasticfilesystem:CreateFileSystem などのリソースを作成するアクションを使用するためのアクセス許可が必要です。リソース作成アクションでタグが指定されている場合、はelasticfilesystem:TagResourceアクションに対して追加の認可 AWS を実行して、ユーザーがタグを作成するアクセス許可を持っているかどうかを確認

します。そのため、ユーザーにはelasticfilesystem:TagResource アクションを使用する明示的なアクセス権限が必要です。

elasticfilesystem:TagResource アクションの IAM ポリシー定義で、Condition 要素と elasticfilesystem:CreateAction 条件キーを使用して、リソースを作成するアクションにタグ付けのアクセス許可を付与します。

Example ポリシー: 作成時にのみファイルシステムにタグを追加できるようにする

次のポリシー例では、ユーザーがファイルシステムを作成し、作成時にのみタグを適用するようにします。ユーザーには既存のリソースへのタグ付けが許可されません (elasticfilesystem:TagResource アクションを直接呼び出すことはできません)。

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "elasticfilesystem:CreateFileSystem"
      ],
      "Resource": "arn:aws:elasticfilesystem:region:account-id:file-system/*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "elasticfilesystem:TagResource"
      ],
      "Resource": "arn:aws:elasticfilesystem:region:account-id:file-system/*",
      "Condition": {
        "StringEquals": {
          "elasticfilesystem:CreateAction": "CreateFileSystem"
        }
      }
    }
  ]
}
```

タグを使用した Amazon EFS リソースへのアクセスのコントロール

Amazon EFS とアクションへのアクセスをコントロールするには、タグに基づいて IAM ポリシーを使用できます。コントロールは 2 つの方法で可能です。

- それらのリソースのタグに基づいて、Amazon EFS へのアクセスをコントロールできます。
- IAM リクエストの条件でどのタグを渡すかをコントロールできます。

タグを使用して AWS リソースへのアクセスを制御する方法については、IAM ユーザーガイドの[タグを使用したアクセスの制御](#)を参照してください。

リソースのタグに基づいてアクセスをコントロールする

ユーザーまたはロールが Amazon EFS リソースで実行できるアクションをコントロールするには、リソースでタグを使用できます。例えば、リソースのタグのキーバリューのペアに基づいて、ファイルシステムリソースに対する特定の API オペレーションを許可または拒否することが必要な場合があります。

Example ポリシー: 特定のタグが使用されている場合にのみファイルシステムを作成する

このポリシー例により、ユーザーは、特定のタグとキーと値のペア (この例では、key=Department、value=Finance) でタグ付けした場合にのみ、ファイルシステムを作成できます。

```
{
  "Effect": "Allow",
  "Action": [
    "elasticfilesystem:CreateFileSystem",
    "elasticfilesystem:TagResource"
  ],
  "Resource": "arn:aws:elasticfilesystem:region:account-id:file-system/*",
  "Condition": {
    "StringEquals": {
      "aws:RequestTag/Department": "Finance"
    }
  }
}
```

Example ポリシー: 特定のタグを持つファイルシステムを削除する

このポリシー例により、ユーザーは Department=Finance でタグ付けされたファイルシステムのみを削除できます。

```
{
  "Version": "2012-10-17",
```

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "elasticfilesystem:DeleteFileSystem"
    ],
    "Resource": "arn:aws:elasticfilesystem:region:account-id:file-system/*",
    "Condition": {
      "StringEquals": {
        "aws:ResourceTag/Department": "Finance"
      }
    }
  }
]
```

Amazon EFS のサービスリンクロールの使用

Amazon Elastic File System は、AWS Identity and Access Management (IAM) [サービスにリンクされたロール](#)を使用します。Amazon EFS サービスリンクロールは、Amazon EFS に直接リンクされた特殊なタイプの IAM ロールです。事前定義された Amazon EFS サービスにリンクされたロールには、サービスが AWS のサービス ユーザーに代わって他の を呼び出すために必要なアクセス許可が含まれています。

サービスにリンクされたロールを使用することで、必要なアクセス許可を手動で追加する必要がなくなるため、Amazon EFS の設定が簡単になります。Amazon EFS は、サービスにリンクされたロールのアクセス権限を定義します。Amazon EFS のみがそのロールを引き受けることができます。定義された許可には信頼ポリシーと許可ポリシーが含まれ、その許可ポリシーを他の IAM エンティティにアタッチすることはできません。

Amazon EFS サービスにリンクされたロールを削除するには、最初に Amazon EFS ファイルシステムを削除する必要があります。これは、リソースにアクセスするための許可を誤って削除できないため、Amazon EFS リソースを保護します。

サービスにリンクされたロールを使用すると、すべての API コールを表示できます AWS CloudTrail。これがモニタリングと監査の要件を満たすのに役立つのは、Amazon EFS によってユーザーに代わって実行されるすべてのアクションを追跡できるためです。詳細については、「[EFS サービスにリンクされたロールのログエントリ](#)」を参照してください。

詳細についてはIAM ユーザーガイドの「[サービスにリンクされた役割のアクセス許可](#)」を参照してください。

Amazon EFS のサービスリンクロール許可

Amazon EFS は、`AWSServiceRoleForAmazonElasticFileSystem` という名前のサービスにリンクされたロールを使用して、Amazon EFS が EFS ファイルシステムに代わって AWS リソースを呼び出して管理できるようにします。

`AWSServiceRoleForAmazonElasticFileSystem` サービスにリンクされたロールは、信頼 `elasticfilesystem.amazonaws.com` してロールを引き受けます。

ロールのアクセス許可ポリシーは、ポリシー定義 JSON に含まれるアクションを実行することを Amazon EFS に許可します。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "backup-storage:MountCapsule",
        "ec2:CreateNetworkInterface",
        "ec2:DeleteNetworkInterface",
        "ec2:DescribeSecurityGroups",
        "ec2:DescribeSubnets",
        "ec2:DescribeNetworkInterfaceAttribute",
        "ec2:ModifyNetworkInterfaceAttribute",
        "tag:GetResources"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "kms:DescribeKey"
      ],
      "Resource": "arn:aws:kms:*:*:key/*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "backup:CreateBackupVault",
        "backup:PutBackupVaultAccessPolicy"
      ],
      "Resource": [
```



```

        "arn:aws:backup:*:*:backup-vault:aws/efs/automatic-backup-vault"
    ],
},
{
    "Effect": "Allow",
    "Action": [
        "backup:CreateBackupPlan",
        "backup:CreateBackupSelection"
    ],
    "Resource": [
        "arn:aws:backup:*:*:backup-plan:*"
    ]
},
{
    "Effect": "Allow",
    "Action": [
        "iam:CreateServiceLinkedRole"
    ],
    "Resource": "*",
    "Condition": {
        "StringEquals": {
            "iam:AWSServiceName": [
                "backup.amazonaws.com"
            ]
        }
    }
},
{
    "Effect": "Allow",
    "Action": [
        "iam:PassRole"
    ],
    "Resource": [
        "arn:aws:iam:*:*:role/aws-service-role/backup.amazonaws.com/
AWSServiceRoleForBackup"
    ],
    "Condition": {
        "StringLike": {
            "iam:PassedToService": "backup.amazonaws.com"
        }
    }
},
{
    "Effect": "Allow",

```

```
    "Action": [  
        "elasticfilesystem:DescribeFileSystems",  
        "elasticfilesystem:CreateReplicationConfiguration",  
        "elasticfilesystem:DescribeReplicationConfigurations",  
        "elasticfilesystem>DeleteReplicationConfiguration",  
        "elasticfilesystem:ReplicationRead",  
        "elasticfilesystem:ReplicationWrite"  
    ],  
    "Resource": "*" ]  
}
```

Note

保管時に暗号化された新しい EFS ファイルシステムを作成する AWS KMS ときは、の IAM アクセス許可を手動で設定する必要があります。詳細については[保管中のデータの暗号化](#)を参照してください。

サービスリンク役割の作成、編集、削除を IAM エンティティ (ユーザー、グループ、役割など) に許可するにはアクセス許可を設定する必要があります。詳細については、「IAM ユーザーガイド」の「[サービスリンクされたロールのアクセス許可](#)」を参照してください。

Amazon EFS のサービスリンクロールの作成

ほとんどの場合、サービスにリンクされたロールを手動で作成する必要はありません。、または AWS API で EFS ファイルシステムのマウントターゲット AWS Management Console AWS CLIまたはレプリケーション設定を作成すると、Amazon EFS によってサービスにリンクされたロールが作成されます。

さらに、このservice-linked-roleを手動で削除し、再度作成する必要がある場合は、同じプロセスを使用してアカウントでロールを再作成できます。EFS ファイルシステムのマウントターゲットまたはレプリケーション設定を作成すると、Amazon EFS によってサービスにリンクされたロールが作成されます。

ただし、Amazon EFS がservice-linked-roleを作成しない場合、またはサービスにリンクされたロールをサポートする前に Amazon EFS の使用を開始した場合は、サービスにリンクされたロールを手動で作成できます。手順については、IAM ユーザーガイドの「[サービスにリンクされたロールの作成](#)」を参照してください。

Amazon EFS のサービスリンクロールの編集

Amazon EFS では、`AWSServiceRoleForAmazonElasticFileSystem` のサービスにリンクされたロールを編集することはできません。サービスリンクロールの作成後は、さまざまなエンティティがロールを参照する可能性があるため、ロール名を変更することはできません。ただし、IAM を使用してロールの説明を編集することはできます。詳細については、「IAM ユーザーガイド」の「[サービスリンクロールの更新](#)」を参照してください。

Amazon EFS のサービスリンクロールの削除

サービスリンクロールを必要とする機能やサービスが不要になった場合は、ロールを削除することをお勧めします。そうすることで、使用していないエンティティがアクティブにモニタリングされたり、メンテナンスされたりすることがなくなります。ただし、手動で削除する前に、サービスリンクロールのリソースをクリーンアップする必要があります。詳細については、「[リソースのクリーンアップと AWS アカウントの保護](#)」を参照してください。

Note

リソースを削除しようとしているときに Amazon EFS サービスがロールを使用している場合は、削除が失敗する可能性があります。失敗した場合は数分待ってから操作を再試行してください。

IAM コンソール、AWS CLI、または AWS API を使用し

て、`AWSServiceRoleForAmazonElasticFileSystem` サービスにリンクされたロールを削除します。詳細については、「IAM ユーザーガイド」の「[サービスにリンクされたロールの削除](#)」を参照してください。

Amazon EFS サービスにリンクされたロールでサポートされているリージョン

Amazon EFS は、サービスが利用可能なすべての AWS リージョン サービスにリンクされたロールの使用をサポートしています。詳細については、「AWS 全般のリファレンス ユーザーガイド」の[AWS 「サービスエンドポイント」](#)を参照してください。

Amazon Elastic File System アイデンティティとアクセスのトラブルシューティング

Amazon EFS と IAM の使用に伴って発生する可能性がある一般的な問題の診断や修復には、次の情報を利用してください。

トピック

- [Amazon EFS でアクションを実行する認可がありません](#)
- [iam:PassRole を実行する権限がありません](#)
- [自分の 以外のユーザーに Amazon EFS リソース AWS アカウント へのアクセスを許可したい](#)

Amazon EFS でアクションを実行する認可がありません

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

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

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

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

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

iam:PassRole を実行する権限がありません

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

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

次の例のエラーは、marymajor という IAM ユーザーがコンソールを使用して Amazon EFS でアクションを実行しようとする場合に発生します。ただし、このアクションをサービスが実行するには、サービスロールから付与された権限が必要です。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。サインイン資格情報を提供した担当者が管理者です。

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

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

詳細については、以下を参照してください:

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

IAM を使用してファイルシステムへのアクセスを制御する

IAM のアイデンティティ ポリシーとリソースポリシーの両方を使用して、クラウド環境向けのスケラブルで最適化された方法により、Amazon EFS リソースへのクライアントアクセスを制御で

きます。IAM を使用すると、読み取り専用アクセス、書き込みアクセス、ルートアクセスなどの特定のアクションをファイルシステムに実行することをクライアントに許可できます。IAM アイデンティティポリシーまたはファイルシステムリソースポリシーのいずれかで、アクションに「許可」権限を付与すると、そのアクションへのアクセスが許可されます。ID ポリシーとリソースポリシーの両方で権限を付与する必要はありません。

NFS クライアントは、EFS ファイルシステムに接続するときに、IAM ロールを使用してそれ自体を識別できます。クライアントがファイルシステムに接続すると、Amazon EFS はファイルシステムの IAM リソースポリシー (ファイルシステムポリシー) とアイデンティティベースの IAM ポリシーを評価し、付与する適切なファイルシステムアクセス許可を決定します。

NFS クライアントの IAM 認可を使用すると、クライアント接続と IAM 認可の決定が AWS CloudTrail に記録されます。CloudTrail を使用して Amazon EFS API コールを記録する方法の詳細については、「[を使用した Amazon EFS API コールのログ記録 AWS CloudTrail](#)」を参照してください。

Important

IAM 認可を使用してクライアントアクセスを制御するには、EFS マウントヘルパーを使用して EFS ファイルシステムをマウントする必要があります。詳細については、「[IAM 認可を使用してマウントする](#)」を参照してください。

デフォルトの EFS ファイルシステムポリシー

デフォルトの EFS ファイルシステムポリシーは、認証に IAM を使用せず、マウントターゲットを使用して、ファイルシステムに接続できる任意の匿名クライアントにフルアクセスを許可します。デフォルトポリシーは、ファイルシステムの作成時を含め、ユーザーが設定したファイルシステムポリシーが有効でない場合は常に有効になります。デフォルトのファイルシステムポリシーが有効な場合、[DescribeFileSystemPolicy](#) API オペレーションは PolicyNotFound レスポンスを返します。

クライアントの EFS アクション

ファイルシステムポリシーを使用してファイルシステムにアクセスするクライアントに対して、以下のアクションを指定できます。

アクション	説明
<code>elasticfilesystem:ClientMount</code>	ファイルシステムへの読み取り専用アクセス許可を付与します。
<code>elasticfilesystem:ClientWrite</code>	ファイルシステムへの書き込みアクセス許可を付与します。
<code>elasticfilesystem:ClientRootAccess</code>	ファイルシステムへのアクセス時にルートユーザーの使用を許可します。

クライアントの EFS 条件キー

条件を表すには、あらかじめ定義された条件キーを使用します。Amazon EFS には、NFS クライアント用に次の事前定義された条件キーがあります。IAM コントロールを使用して EFS ファイルシステムへのアクセスを保護する場合、その他の条件キーは適用されません。

EFS 条件キー	説明	演算子
<code>aws:SecureTransport</code>	このキーを使用して、EFS ファイルシステムに接続するときに TLS の使用をクライアントに要求します。	ブール値
<code>aws:SourceIp</code>	このキーを使用して、リクエストの IP アドレスをポリシーで指定した IP アドレスと比較します。 <code>aws:SourceIp</code> 条件キーは、パブリック IP アドレス範囲にのみ使用できません。	String
<code>elasticfilesystem:AccessPointArn</code>	クライアントが接続している EFS アクセスポイントの ARN。	String

EFS 条件キー	説明	演算子
<code>elasticfilesystem:AccessedViaMountTarget</code>	このキーを使用して、ファイルシステムマウントターゲットを使用していないクライアントによる EFS ファイルシステムへのアクセスを防止します。	ブール値

ファイルシステムポリシーの例

Amazon EFS ファイルシステムポリシーの例を表示するには、「[Amazon EFS のリソースベースのポリシーの例](#)」を参照してください。

NFS クライアントの EFS ファイルシステムへのネットワークアクセスの制御

ネットワーク層セキュリティおよび EFS ファイルシステムポリシーを使用して、Amazon EFS ファイルシステムへの NFS クライアントによるアクセスを制御できます。VPC セキュリティグループルールやネットワーク ACL など、Amazon EC2 で使用できるネットワーク層セキュリティメカニズムを使用できます。AWS IAM を使用して、EFS ファイルシステムポリシーとアイデンティティベースのポリシーで NFS アクセスを制御することもできます。

トピック

- [VPC セキュリティグループの使用](#)
- [Amazon EFS でのインターフェイス VPC エンドポイントの使用](#)

VPC セキュリティグループの使用

Amazon EFS を使用する場合は、EC2 インスタンスの VPC セキュリティグループと、ファイルシステムに関連付けられた EFS マウントターゲットのセキュリティグループを指定します。セキュリティグループはファイアウォールとして機能し、追加するルールはトラフィックフローを定義します。[開始方法の演習](#)では、EFS インスタンスを起動したときに 1 つのセキュリティグループを作成しました。次に、別のセキュリティグループを EFS マウントターゲット (デフォルト VPC のデフォルトのセキュリティグループ) に関連付けました。このアプローチは、「開始方法」の演習で使用で

きます。ただし、本稼働システムでは、Amazon EFS で使用するための最小限のアクセス許可でセキュリティグループを設定する必要があります。

EFS ファイルシステムへのインバウンドおよびアウトバウンドのアクセスを許可できます。そのためには、EFS インスタンスがネットワークファイルシステム (NFS) ポートを使用してマウントターゲットを介して EFS ファイルシステムに接続できるようにするルールを追加します。

- ファイルシステムをマウントする各 EC2 インスタンスには、NFS ポートのマウントターゲットへのアウトバウンドアクセスを許可するルールを持つセキュリティグループが必要です。
- EFS マウントターゲットには、ファイルシステムをマウントする各 EC2 インスタンスからのインバウンドアクセスを許可するルールを持つセキュリティグループが必要です。

Amazon EFS を操作するためのソースポート

一連の NFS クライアントをサポートするために、Amazon EFS はソースポートからのすべての接続を許可します。権限のあるユーザーのみが Amazon EFS にアクセスできるようにするには、以下のクライアントのファイアウォールルールを使用することをお勧めします。SSH を使用してファイルシステムに接続し、次のコマンドを実行します。

```
iptables -I OUTPUT 1 -m owner --uid-owner 1-4294967294 -m tcp -p tcp --dport 2049 -j DROP
```

このコマンドは、OUTPUT チェーン (-I OUTPUT 1) の開始時に新しいルールを挿入します。このルールにより、権限のない、非カーネルプロセス (-m owner --uid-owner 1-4294967294) が、NFS ポート (-m tcp -p tcp -dport 2049) への接続を開くことを防ぎます。

ネットワークアクセスに関するセキュリティ上の考慮事項

ファイルシステムのマウントターゲットの NFS ポート (TCP ポート 2049) の 1 つに接続できる場合にのみ、NFS バージョン 4.1 (NFSv4.1) クライアントはファイルシステムをマウントできます。同様に、このネットワーク接続ができる場合にのみ、NFSv 4.1 クライアントはファイルシステムにアクセスするときにユーザーおよびグループ ID をアサートできます。

このネットワーク接続を行うことができるかどうかは、以下の組み合わせによって管理されます。

- マウントターゲットの VPC によって提供されるネットワーク分離 – ファイルシステムのマウントターゲットにはパブリック IP アドレスを関連付けることはできません。ファイルシステムをマウントできる唯一のターゲットは次のとおりです。
 - ローカル Amazon VPC 内の Amazon EC2 インスタンス

- 接続された VPC の EC2 インスタンス
- AWS Direct Connect と AWS Virtual Private Network (VPN) を使用して Amazon VPC に接続されたオンプレミスサーバー
- マウントターゲットのサブネット外からアクセスするための、クライアントおよびマウントターゲットの VPC 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 (ACL) – ファイルシステムをマウントするには、クライアントはマウントターゲットの NFS ポートへの TCP 接続を確立する必要があります。また、リターントラフィックを受信する必要があります。
- すべてのアクセス用のクライアントおよびマウントターゲットの VPC セキュリティグループのルール ファイルシステムをマウントする EC2 インスタンスの場合、次のセキュリティグループルールが有効である必要があります。
 - ファイルシステムには、インスタンスからの NFS ポートでインバウンド接続を有効にするルールを持つセキュリティグループがネットワークインターフェイスにあるマウントターゲットが必要です。IP アドレス (CIDR 範囲) またはセキュリティグループのいずれかを使用してインバウンド接続を有効化できます。マウントターゲットネットワークインターフェイス上のインバウンド NFS ポートセキュリティグループルールのソースは、ファイルシステムのアクセスコントロールの重要な要素です。NFS ポート以外のインバウンドルール、およびどのようなアウトバウンドルールも、ファイルシステムのマウントターゲットネットワークインターフェイスには使用されません。
 - インスタンスのマウントには、ファイルシステムのマウントターゲットのいずれかの NFS ポートへのアウトバウンド接続を有効にするセキュリティグループルールがあるネットワークインターフェイスが必要です。IP アドレス (CIDR 範囲) またはセキュリティグループのいずれかを使用してアウトバウンド接続を有効化できます。

詳細については、「[マウントターゲットの管理](#)」を参照してください。

セキュリティグループの作成

EC2 インスタンスと EFS マウントターゲットのセキュリティグループを作成するには

以下は、Amazon EFS のセキュリティグループを作成するときに実行する一般的な手順です。セキュリティグループの作成手順については、「Amazon VPC [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。

1. EC2 インスタンスの場合は、次のルールを使用してセキュリティグループを作成します。
 - 任意のホストからの Secure Shell (SSH) の使用を許可するインバウンドアクセスを許可するインバウンドルール。オプションで、[ソース] アドレスを制限します。

- すべてのトラフィックの退出を許可するアウトバウンドルール。セキュリティグループを作成すると、デフォルトでアウトバウンドルールで作成されるため、追加する必要はありません。
2. EFS マウントターゲットの場合は、次のルールを使用してセキュリティグループを作成します。
- EC2 セキュリティグループからのアクセスを許可するインバウンドルール。EC2 セキュリティグループをソースとして識別します。
 - すべての NFS ポートで TCP 接続を開くアウトバウンドルール。EC2 セキュリティグループのデスティネーションを指定します。

Amazon EFS でのインターフェイス VPC エンドポイントの使用

仮想プライベートクラウド (VPC) と Amazon EFS API の間にプライベート接続を確立するには、インターフェイス VPC エンドポイントを作成できます。このエンドポイントは、インターネットゲートウェイ、NAT インスタンス、または仮想プライベートネットワーク (VPN) 接続を必要とせずに、Amazon EFS API への安全な接続を提供します。詳細については、「Amazon [VPC ユーザーガイド](#)」の「[インターフェイス VPC エンドポイントを使用して AWS サービスにアクセスする](#)」を参照してください。

インターフェイス VPC エンドポイントは AWS PrivateLink、プライベート IP アドレスを使用して AWS サービス間のプライベート通信を可能にする機能である [AWS PrivateLink](#) を利用しています。AWS PrivateLink を使用するには、Amazon VPC コンソール、API、または CLI を使用して、VPC に Amazon EFS のインターフェイス VPC エンドポイントを作成します。これによって Elastic Network Interface がサブネットに作成され、そのプライベート IP アドレスが Amazon EFS API リクエストを処理します。または VPC ピアリングを使用して AWS VPN AWS Direct Connect、オンプレミス環境または他の VPCs から VPC エンドポイントにアクセスすることもできます。詳細については、「Amazon [VPC ユーザーガイド](#)」の「[を使用して VPC を サービスに接続する AWS PrivateLink](#)」を参照してください。

Amazon EFS 用のインターフェイスエンドポイントの作成

Amazon EFS のインターフェイス VPC エンドポイントを作成するには、次のいずれかを使用します。

- **com.amazonaws.*region*.elasticfilesystem** – Amazon EFS API オペレーションのエンドポイントを作成します。

- **com.amazonaws.*region*.elasticfilesystem-fips** — [連邦情報処理規格 \(FIPS\) 140-2](#) に準拠した Amazon EFS API のエンドポイントを作成します。

Amazon EFS エンドポイントの完全なリストについては、の[Amazon Elastic File System 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。

インターフェイスエンドポイントの作成方法の詳細については、「Amazon [VPC ユーザーガイド](#)」の「[インターフェイス VPC エンドポイントを使用して AWS サービスにアクセスする](#)」を参照してください。

Amazon EFS 用の VPC エンドポイントポリシーの作成

Amazon EFS API へのアクセスを制御するには、VPC エンドポイントに AWS Identity and Access Management (IAM) ポリシーをアタッチします。本ポリシーでは、以下を規定します。

- アクションを実行できるプリンシパル。
- 実行可能なアクション。
- アクションを実行できるリソース。

詳細については、「Amazon VPC ユーザーガイド」の「[エンドポイントポリシーを使用して VPC エンドポイントへのアクセスを制御する](#)」を参照してください。

次の例は、エンドポイントを介して EFS ファイルシステムを作成するアクセス許可を全員に対して拒否する VPC エンドポイントポリシーを示しています。このポリシー例では、他のすべてのアクションを実行するアクセス許可も全員に付与しています。

```
{
  "Statement": [
    {
      "Action": "*",
      "Effect": "Allow",
      "Resource": "*",
      "Principal": "*"
    },
    {
      "Action": "elasticfilesystem:CreateFileSystem",
      "Effect": "Deny",
      "Resource": "*",
      "Principal": "*"
    }
  ]
}
```

```
    }  
  ]  
}
```

ネットワークファイルシステム (NFS) レベルのユーザー、グループ、およびアクセス許可

ファイルシステムを作成した後、デフォルトでは、ルートユーザー (UID 0) のみが読み取り/書き込み/実行のアクセス許可を持ちます。他のユーザーがファイルシステムを変更できるようにするには、ルートユーザーは、明示的にアクセス許可を付与する必要があります。アクセスポイントを使用すると、root 以外のユーザーによる書き込み可能なディレクトリの作成を自動化できます。詳細については、「[アクセスポイントの使用](#)」を参照してください。

EFS ファイルシステムオブジェクトには、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。Unix スタイルのシステムに精通しているユーザーは、これらのアクセス許可に関して Amazon EFS がどのように動作するかを簡単に理解できます。

さらに、Unix 形式のシステムでは、ユーザーとグループは Amazon EFS がファイルの所有権を表すために使用する数値 ID にマッピングされます。Amazon EFS では、ファイルシステムオブジェクト (ファイル、ディレクトリなど) は、単一の所有者と単一のグループによって所有されます。Amazon EFS は、ユーザーがファイルシステムオブジェクトにアクセスしようとする、マッピングされた数値 ID を使用してアクセス許可をチェックします。

Note

NFS プロトコルは 1 ユーザーあたり最大 16 のグループ ID (GID) をサポートし、追加の GID は NFS クライアント要求から切り捨てられます。詳細については、「[NFS ファイルシステム上の許可されたファイルへのアクセスが拒否されました](#)」を参照してください。

以下では、アクセス許可の例、および Amazon EFS での NFS のアクセス許可に関する考慮事項について説明します。

トピック

- [ファイルおよびディレクトリのアクセス許可](#)
- [EFS ファイルシステムのユースケースとアクセス許可の例](#)

- [ファイルシステム内のファイルとディレクトリに対するユーザー ID およびグループ ID のアクセス許可](#)
- [ルートスカッシュなし](#)
- [アクセス許可のキャッシュ](#)
- [ファイルシステムオブジェクトの所有権の変更](#)
- [EFS アクセスポイント](#)

ファイルおよびディレクトリのアクセス許可

EFS ファイルシステムのファイルとディレクトリは、EFS アクセスポイントによって上書きされない限り、NFSv4.1 クライアントのマウントを使用してアサートされたユーザー ID とグループ ID に基づいて、標準 UNIX 形式の読み取り、書き込み、および実行のアクセス許可をサポートします。詳細については、「[ネットワークファイルシステム \(NFS\) レベルのユーザー、グループ、およびアクセス許可](#)」を参照してください。

Note

デフォルトでは、このアクセスコントロールのレイヤーは、NFSv4.1 クライアントをユーザー ID とグループ ID のアサーションで信頼するかどうかに依存します。AWS Identity and Access Management (IAM) リソースベースのポリシーと ID ポリシーを使用して、NFS クライアントを承認し、読み取り専用、書き込み、およびルートアクセス許可を提供できます。EFS アクセスポイントを使用して、NFS クライアントから提供されるオペレーティングシステムのユーザー ID およびグループの ID 情報を上書きできます。詳細については、「[IAM を使用してファイルシステムへのアクセスを制御する](#)」および「[アクセスポイントの作成](#)」を参照してください。

ファイルとディレクトリの読み取り、書き込み、実行のアクセス許可の例として、Alice は、ファイルシステムの個人ディレクトリ /alice にある任意のファイルを読み書きするアクセス許可を持っています。ただし、この例では、Alice は同じファイルシステムの Mark の個人ディレクトリ /mark にあるファイルを読み書きすることはできません。Alice と Mark の両方が共有ディレクトリ /share 内のファイルを読み取ることはできますが、書き込むことはできません。

EFS ファイルシステムのユースケースとアクセス許可の例

EFS ファイルシステムを作成し、VPC 内のファイルシステムのターゲットをマウントしたら、リモートファイルシステムを Amazon EC2 インスタンスにローカルにマウントできます。mount コマ

ンドは、ファイルシステムの任意のディレクトリをマウントできます。ただし、最初にファイルシステムを作成するときは、/ にルートディレクトリは 1 つしかありません。ルートユーザーとルートグループは、マウントされたディレクトリを所有します。

次の mount コマンドは、ファイルシステムの DNS 名で識別される Amazon EFS ファイルシステムのルートディレクトリを ローカルディレクトリにマウントします。

```
sudo mount -t nfs -o
nfsvers=4.1,rsiz=1048576,wsiz=1048576,hard,timeo=600,retrans=2,noresvport file-
system-id.efs.aws-region.amazonaws.com:/ efs-mount-point
```

最初のアクセス許可のモードでは、次のことを許可します。

- 所有者の read-write-execute ルートに対する アクセス許可
- グループの read-execute ルートに対する アクセス許可
- 他のユーザーに対する read-execute アクセス許可

ルートユーザーのみがこのディレクトリを変更できます。たとえば、ルートユーザーは、他のユーザーにこのディレクトリに書き込むアクセス許可を付与することもできます。

- 書き込み可能なユーザーごとのサブディレクトリを作成します。手順については、「[チュートリアル: 書き込み可能なユーザーごとのサブディレクトリを作成する](#)」を参照してください。
- ユーザーに EFS ファイルシステムのルートへの書き込みを許可します。ルート権限を持つユーザーは、他のユーザーにファイルシステムへのアクセスを許可することができます。
- EFS ファイルシステムの所有権を非ルートユーザーおよびグループに変更するには、以下を使用します。

```
$ sudo chown user:group /EFSroot
```

- ファイルシステムのアクセス許可をより制限の低いものに変更するには、以下を使用します。

```
$ sudo chmod 777 /EFSroot
```

このコマンドは、ファイルシステムがマウントされているすべての EC2 インスタンス上のすべてのユーザーに読み取り/書き込み/実行の権限を許可します。

ファイルシステム内のファイルとディレクトリに対するユーザー ID およびグループ ID のアクセス許可

EFS ファイルシステムのファイルとディレクトリは、ユーザー ID とグループ IDs に基づく標準の Unix スタイルの読み取り、書き込み、実行のアクセス許可をサポートします。NFS クライアントがアクセスポイントを使用せずに EFS ファイルシステムをマウントした場合は、クライアントから提供されたユーザー ID とグループ ID が信頼されます。EFS アクセスポイントにより、NFS クライアントによって使用されるユーザー ID とグループ ID を上書きできます。ユーザーがファイルとディレクトリにアクセスしようとする、Amazon EFS はユーザー ID とグループ ID を確認して、ユーザーにオブジェクトへのアクセス権限があることを確認します。Amazon EFS は、これらの ID をユーザーが作成する新しいファイルやディレクトリの所有者およびグループ所有者を示すために使用します。Amazon EFS は、ユーザー名またはグループ名を調べません。数値 ID のみを使用します。

Note

EC2 インスタンスでユーザーを作成すると、任意の数値のユーザー ID (UID) とグループ ID (GID) をユーザーに割り当てることができます。数値のユーザー ID は、Linux システムの `/etc/passwd` ファイルに設定されます。数値グループ ID は、`/etc/group` ファイルにあります。これらのファイルは、名前と ID の間のマッピングを定義します。EC2 インスタンスの外部では、Amazon EFS は、0 のルート ID を含むこれらの ID の認証を実行しません。

ユーザーが 2 つの異なる EC2 インスタンスから EFS ファイルシステムにアクセスする場合、ユーザーの UID がそれらのインスタンスで同じか異なるかに応じて、次のように異なる動作が表示されます。

- ユーザー ID が両方の EC2 インスタンスで同じである場合、Amazon EFS は使用された EC2 インスタンスに関係なく、それらが同じユーザーを示すとみなします。ファイルシステムにアクセスする際のユーザーエクスペリエンスは、両方の EC2 インスタンスで同じです。
- ユーザー ID が両方の EC2 インスタンスで同じでない場合、Amazon EFS はユーザーを異なるユーザーと見なします。2 つの異なる EC2 インスタンスから EFS ファイルシステムにアクセスする場合、ユーザーエクスペリエンスは同じではありません。
- 異なる EC2 インスタンスの 2 人の異なるユーザーが ID を共有する場合、Amazon EFS はそれらを同じユーザーとみなします。

EC2 インスタンス間でユーザー ID マッピングを一貫して管理することを検討することもできます。ユーザーは、`id` コマンドを使用して数値 ID を確認することができます。

```
$ id

uid=502(joe) gid=502(joe) groups=502(joe)
```

ID マッパーを無効にする

オペレーティングシステムの NFS ユーティリティには、ユーザー名と ID の間のマッピングを管理する ID マッパーと呼ばれるデーモンが含まれています。Amazon Linux では、デーモンは `rpc.idmapd` と呼ばれ、Ubuntu では、`idmapd` と呼ばれます。これは、ユーザーおよびグループ ID を名前に変換し、逆も行います。ただし、Amazon EFS は、数値 ID のみを扱います。EC2 インスタンスでこのプロセスを無効にすることをお勧めします。Amazon Linux では、マッパーは通常無効です。この場合、ID マッパーを有効にしないでください。ID マッパーを無効にするには、以下に示すコマンドを使用します。

```
$ service rpcidmapd status
$ sudo service rpcidmapd stop
```

ルースカッシュなし

デフォルトでは、ルースカッシュは EFS ファイルシステムで無効になっています。Amazon EFS は、`no_root_squash` を使用する Linux NFS サーバーのように動作します。ユーザーまたはグループ ID が 0 の場合、Amazon EFS はそのユーザーを root ユーザーとして処理し、アクセス許可チェックをバイパスします (すべてのファイルシステムオブジェクトへのアクセスと変更を許可します)。AWS Identity and Access Management (AWS IAM) ID またはリソースポリシーが `ClientRootAccess` アクションへのアクセスを許可していない場合は、クライアント接続でルースカッシュを有効にできます。ルースカッシュが有効になっている場合、ルートユーザーは NFS サーバー上で制限されたアクセス許可を持つユーザーに変換されます。

詳細については、「[IAM を使用してファイルシステムへのアクセスを制御する](#)」を参照してください。

NFS クライアントの IAM 認可を使用したルースカッシュの有効化

単一の管理ワークステーションを除くすべての AWS プリンシパルの EFS ファイルシステムへのルートアクセスを防ぐように Amazon EFS を設定できます。そのためには、ネットワークファイルシステム (NFS) クライアントの AWS Identity and Access Management (IAM) 認可を設定します。

これを行うには、次に示すように、2 つの IAM アクセス許可ポリシーを設定する必要があります。

- ファイルシステムへの読み取りおよび書き込みアクセスを明示的に許可し、ルートアクセスを暗黙的に拒否する EFS ファイルシステムポリシーを作成します。
- ECAmazon EC2 インスタンスプロファイルを使用して、ファイルシステムへのルートアクセスを必要とする Amazon EC2 管理ワークステーションに IAM ID を割り当てます。Amazon EC2 インスタンスプロファイルの詳細については、「[ユーザーガイド](#)」の「[インスタンスプロファイルを使用する](#)」を参照してください。AWS Identity and Access Management
- AmazonElasticFileSystemClientFullAccess AWS 管理ポリシーを管理ワークステーションの IAM ロールに割り当てます。Amazon EFS の AWS マネージドポリシーの詳細については、「[Amazon EFS のためのアイデンティティとアクセス管理](#)」を参照してください。

NFS クライアントの IAM 認可を使用してルートスカッシュを有効にするには、以下の手順を使用します。

ファイルシステムへのルートアクセスを無効にするには

1. Amazon Elastic File System コンソールの <https://console.aws.amazon.com/efs/> を開いてください。
2. [File Systems (ファイルシステム)] を選択します。
3. ルートスカッシュを有効にするファイルシステムを選択します。
4. [File System details] ページで、[ファイルシステムポリシー]、続いて編集を選択します。[File system policy (ファイルシステムポリシー)] ページが表示されます。
5. ポリシーオプションの選択デフォルトで root アクセスを禁止する*を選択します。ポリシー JSON オブジェクトがポリシーエディターに表示されます。
6. [Save (保存)] を選択して、ファイルシステムポリシーを保存します。

非匿名クライアントは、アイデンティティベースのポリシーを通じてファイルシステムへのルートアクセスを取得できます。AmazonElasticFileSystemClientFullAccess 管理ポリシーをワークステーションのロールにアタッチすると、IAM はその ID ポリシーに基づいてワークステーションへのルートアクセスを許可します。

管理ワークステーションからルートアクセスを有効にするには

1. IAM コンソール (<https://console.aws.amazon.com/iam/>) を開きます。

2. Amazon EC2 の「EFS-client-root-access」というロールを作成します。IAM は、作成した EC2 ロールと同じ名前を使用してインスタンスプロファイルを作成します。
3. 作成した EC2 ロール AmazonElasticFileSystemClientFullAccess に AWS 管理ポリシーを割り当てます。このポリシーの内容は次のとおりです。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "elasticfilesystem:ClientMount",
        "elasticfilesystem:ClientRootAccess",
        "elasticfilesystem:ClientWrite",
        "elasticfilesystem:DescribeMountTargets"
      ],
      "Resource": "*"
    }
  ]
}
```

4. 以下に説明するように、管理ワークステーションとして使用している EC2 インスタンスにインスタンスプロファイルをアタッチします。詳細については、Linux インスタンス用 Amazon EC2 ユーザーガイドの「[IAM ロールをインスタンスにアタッチする](#)」を参照してください。
 - a. Amazon EC2 コンソールの <https://console.aws.amazon.com/ec2/> を開いてください。
 - b. ナビゲーションペインで、[インスタンス] を選択してください。
 - c. インスタンスを選択します。[Actions (アクション)] で [インスタンスの設定] を選択し、[IAM ロールの割り当て/置換] を選択します。
 - d. 最初のステップで作成した IAM ロール (EFS-client-root-access) を選択してから、[Apply (適用)] を選択します。
5. 管理ワークステーションに EFS マウントヘルパーをインストールします。EFS マウントヘルパー マウントヘルパーと amazon-efs-utils パッケージの詳細については、「」を参照してください [Amazon EFS クライアントの手動インストール](#)。
6. 以下のコマンドと iam マウントオプションを使用して、管理ワークステーションに EFS ファイルシステムをマウントします。

```
$ sudo mount -t efs -o tls,iam file-system-id:/ efs-mount-point
```

IAM 認可を使用してファイルシステムを自動的にマウントするように Amazon EC2 インスタンスを設定できます。IAM 認可を使用した EFS ファイルシステムのマウントの詳細については、「[IAM 認可を使用してマウントする](#)」を参照してください。

アクセス許可のキャッシュ

Amazon EFS は、短期間、ファイルのアクセス許可をキャッシュします。その結果、最近アクセスを取り消されたユーザーが、まだそのオブジェクトに短期間アクセスできることがあります。

ファイルシステムオブジェクトの所有権の変更

Amazon EFS は、POSIX `chown_restricted` 属性を適用します。これは、ルートユーザーだけがファイルシステムオブジェクトの所有者を変更できることを意味します。ルートユーザーまたは所有者ユーザーは、ファイルシステムオブジェクトの所有者グループを変更できます。ただし、ユーザーがルートでない限り、そのグループは所有者ユーザーがメンバーになっているもののみ変更できます。

EFS アクセスポイント

アクセスポイントは、アクセスポイントを介したすべてのシステム要求に対して、オペレーティングシステムのユーザー、グループ、およびファイルシステムのパスを適用します。アクセスポイントのオペレーティングシステムのユーザーおよびグループは、NFS クライアントから提供されるすべての ID 情報を上書きします。ファイルシステムのパスは、アクセスポイントのルートディレクトリとしてクライアントに公開されます。このアプローチにより、各アプリケーションは共有ファイルベースのデータセットにアクセスするときに、常に正しいオペレーティングシステム ID と正しいディレクトリを使用できます。アクセスポイントを使用するアプリケーションは、それ自体のディレクトリ以下のデータにのみアクセスできます。アクセスポイントの詳細については、「[アクセスポイントの使用](#)」を参照してください。

アクセスポイントの使用

EFS アクセスポイントは、EFS ファイルシステムへのアプリケーション固有のエントリポイントであり、共有データセットへのアプリケーションアクセスを簡単に管理できます。アクセスポイントを使用すると、アクセスポイントを介したすべてのファイルシステム要求に対してユーザーアイデンティティ (ユーザーの POSIX グループなど) を適用できます。また、ファイルシステムに対して別のルートディレクトリを適用し、このディレクトリまたはそのサブディレクトリ内のデータに対してのみ、クライアントにアクセスを許可することもできます。

AWS Identity and Access Management (IAM) ポリシーを使用して、特定のアプリケーションが特定のアクセスポイントを使用するように強制できます。IAM ポリシーとアクセスポイントを組み合わせると、アプリケーションから特定のデータセットへのアクセスを簡単に保護できます。

Note

アクセスポイントを使用するには、EFS ファイルシステムに少なくとも 1 つのマウントターゲットを作成する必要があります。

、AWS Command Line Interface (AWS CLI) AWS Management Console、および Amazon EFS API を使用して、既存の EFS ファイルシステムのアクセスポイントを作成できます。EFS ファイルシステムは、[最大 10,000 個のアクセスポイント](#)を持つことができます。既存のアクセスポイントは、作成後に変更することはできません。

アクセスポイントを作成する手順については、「[アクセスポイントの作成](#)」を参照してください。

アクセスポイントを使用してファイルシステムをマウントする場合は、EFS マウントヘルパーを使用します。マウントコマンドには、以下の例に示すように、ファイルシステム ID、アクセスポイント ID、tls マウントオプションを含めます。

```
$ mount -t efs -o tls,iam,accesspoint=fsap-abcdef0123456789a fs-  
abc0123def456789a: /localmountpoint
```

アクセスポイントを使用したファイルシステムのマウントの詳細については、「[Amazon EFS アクセスポイントを使用してマウントする](#)」を参照してください。

トピック

- [アクセスポイントを使用したユーザー ID の適用](#)
- [アクセスポイントを使用したルートディレクトリの適用](#)
- [IAM ポリシーでのアクセスポイントの使用](#)

アクセスポイントを使用したユーザー ID の適用

アクセスポイントを使用すると、アクセスポイントを介したすべてのファイルシステム要求に対して、ユーザーおよびグループ情報を適用できます。この機能を有効にするには、アクセスポイントの作成時に、適用するオペレーティングシステム ID を指定する必要があります。

その一環として、以下の情報を提供します。

- ユーザー ID - ユーザーの数値 POSIX ユーザー ID。
- グループ ID - ユーザーの数値 POSIX グループ ID。
- セカンダリグループ ID - オプションのセカンダリグループ ID のリスト。

ユーザー適用を有効にすると、Amazon EFS は NFS クライアントのユーザー ID およびグループ ID を、アクセスポイントですべてのファイルシステムオペレーションに対して設定されている アイデンティティ に置き換えます。ユーザー適用では、以下のことも行います。

- 新しいファイルとディレクトリの所有者とグループは、アクセスポイントのユーザー ID とグループ ID に設定されます。
- EFS は、ファイルシステムのアクセス許可を評価するときに、アクセスポイントのユーザー ID、グループ ID、およびセカンダリグループ ID を考慮します。EFS は、NFS クライアントの ID を無視します。

Important

ユーザー ID の適用は、ClientRootAccess IAM アクセス許可を条件とします。
たとえば、場合によっては、アクセスポイントのユーザー ID、グループ ID、またはその両方をルートに設定する (つまり、UID、GID、またはその両方を 0 に設定する) ことがあります。そのような場合、NFS クライアントに ClientRootAccess IAM アクセス許可を付与する必要があります。

アクセスポイントを使用したルートディレクトリの適用

アクセスポイントを使用して、ファイルシステムのルートディレクトリを上書きできます。ルートディレクトリを適用すると、アクセスポイントを使用する NFS クライアントは、ファイルシステムのルートディレクトリではなく、アクセスポイントに設定されているルートディレクトリを使用します。

この機能を有効にするには、アクセスポイントの作成時にアクセスポイントの Path 属性を設定します。Path 属性は、このアクセスポイントを介したすべてのファイルシステム要求に対するファイルシステムのルートディレクトリのフルパスです。フルパスの長さは 100 文字を超えることはできません。最大 4 つのサブディレクトリを含めることができます。

アクセスポイントにルートディレクトリを指定すると、これがアクセスポイントをマウントする NFS クライアントのファイルシステムのルートディレクトリになります。たとえば、アクセスポイントのルートディレクトリが /data であるとします。この場合、アクセスポイントを使用して fs-12345678:/ をマウントすると、アクセスポイントを使用せずに fs-12345678:/data をマウントするのと同じ効果があります。

アクセスポイントでルートディレクトリを指定する場合は、アクセスポイントのユーザーがファイルシステムを正常にマウントできるようにディレクトリ権限が設定されていることを確認します。具体的には、アクセスポイントのユーザーまたはグループ、またはすべてのユーザーに対して実行ビットが設定されていることを確認します。たとえば、ディレクトリ権限の値が 755 の場合、ディレクトリユーザーの所有者はファイルのリスト表示、ファイルの作成およびマウントを実行できます。他のすべてのユーザーはファイルのリスト表示とマウントを実行できます。

アクセスポイントのルートディレクトリの作成

アクセスポイントのルートディレクトリパスがファイルシステム上に存在しない場合、Amazon EFS は指定された所有権とアクセス許可を持つルートディレクトリを自動的に作成します。作成時にディレクトリの所有権とアクセス許可を指定しない場合、Amazon EFS はルートディレクトリを作成しません。このアプローチにより、Linux ホストからファイルシステムをマウントすることなく、特定のユーザーまたはアプリケーションに対してファイルシステムアクセスをプロビジョニングできます。ルートディレクトリを作成するには、アクセスポイントの作成時に以下の属性を使用して、ルートディレクトリの所有権とアクセス許可を設定する必要があります。

- OwnerUid - ルートディレクトリの所有者として使用する数値 POSIX ユーザー ID。
- OwnerGiD - ルートディレクトリの所有者グループとして使用する数値 POSIX グループ ID。
- アクセス許可 - ディレクトリの Unix モード。一般的な設定は 755 です。アクセスポイントユーザーがマウントできるように、実行ビットが設定されていることを確認します。この設定では、ディレクトリ所有者に、ディレクトリ内の新しいファイルの入力、一覧表示、書き込みのアクセス許可が付与されます。他のすべてのユーザーに、ファイルを入力および一覧表示するアクセス許可が付与されます。Unix のファイルモードとディレクトリモードの使用の詳細については、[「ネットワークファイルシステム \(NFS\) レベルのユーザー、グループ、およびアクセス許可」](#)を参照してください。

Amazon EFS では、OwnUid、OwnGID、およびパーミッションがディレクトリに指定されている場合にのみ、アクセスポイントルートディレクトリが作成されます。この情報を指定しない場合、Amazon EFS はルートディレクトリを作成しません。ルートディレクトリが存在しない場合に、アクセスポイントを使用してマウントしようとすると、失敗します。

アクセスポイントを使用してファイルシステムをマウントすると、アクセスポイントのルートディレクトリがまだ存在しない場合は、そのアクセスポイントのルートディレクトリが作成されます。ただし、アクセスポイントの作成時にルートディレクトリの OwnerUid とアクセス許可が指定されていることが条件です。アクセスポイントのルートディレクトリがマウント時間前にすでに存在する場合、既存のアクセス許可はアクセスポイントによって上書きされません。ルートディレクトリを削除すると、次にアクセスポイントを使用してファイルシステムをマウントするときに、EFS によってルートディレクトリが再作成されます。

Note

アクセスポイントのルートディレクトリの所有権とアクセス許可を指定しない場合、Amazon EFS はルートディレクトリを作成しません。アクセスポイントをマウントしようとすると、失敗します。

アクセスポイントのルートディレクトリのセキュリティモデル

ルートディレクトリの上書きが有効になっている場合、Amazon EFS は `no_subtree_check` オプションを有効にした Linux NFS サーバーと同じように動作します。

NFS プロトコルでは、ファイルのアクセス時に一意の参照としてクライアントで使用されるファイルハンドルがサーバーで作成されます。EFS は、EFS ファイルシステム固有の予測可能なファイルハンドルを安全に生成します。ルートディレクトリオーバーライドが設定されている場合、Amazon EFS は指定されたルートディレクトリ外のファイルのファイルハンドルを公開しません。ただし、場合によっては、ユーザーは帯域外メカニズムを使用して、アクセスポイント外のファイルのファイルハンドルを取得することがあります。たとえば、2 番目のアクセスポイントにアクセスできる場合は、これを行う可能性があります。これを行うと、ファイルに対して読み取りおよび書き込みオペレーションを実行できます。

ユーザーのアクセスポイントのルートディレクトリ内外のファイルへのアクセスに対しては、ファイルの所有権とアクセス許可が常に適用されます。

IAM ポリシーでのアクセスポイントの使用

IAM ポリシーでは、IAM ロールで特定した NFS クライアントに対して、特定のアクセスポイントへのアクセスのみを許可できます。そのためには、`elasticfilesystem:AccessPointArn` IAM 条件キーを使用します。AccessPointArn は、ファイルシステムがマウントされているアクセスポイントの Amazon リソースネーム (ARN) です。

以下に示しているのは、IAM ロール app1 にアクセスポイント fsap-01234567 を使用したファイルシステムへのアクセスを許可するファイルシステムポリシーの例です。また、このポリシーでは、app2 にアクセスポイント fsap-89abcdef を使用したファイルシステムへのアクセスを許可しています。

```
{
  "Version": "2012-10-17",
  "Id": "MyFileSystemPolicy",
  "Statement": [
    {
      "Sid": "App1Access",
      "Effect": "Allow",
      "Principal": { "AWS": "arn:aws:iam::111122223333:role/app1" },
      "Action": [
        "elasticfilesystem:ClientMount",
        "elasticfilesystem:ClientWrite"
      ],
      "Condition": {
        "StringEquals": {
          "elasticfilesystem:AccessPointArn" : "arn:aws:elasticfilesystem:us-east-1:222233334444:access-point/fsap-01234567"
        }
      }
    },
    {
      "Sid": "App2Access",
      "Effect": "Allow",
      "Principal": { "AWS": "arn:aws:iam::111122223333:role/app2" },
      "Action": [
        "elasticfilesystem:ClientMount",
        "elasticfilesystem:ClientWrite"
      ],
      "Condition": {
        "StringEquals": {
          "elasticfilesystem:AccessPointArn" : "arn:aws:elasticfilesystem:us-east-1:222233334444:access-point/fsap-89abcdef"
        }
      }
    }
  ]
}
```

EFS ファイルシステムへのパブリックアクセスのブロック

Amazon EFS のパブリックアクセスのブロック機能は、EFS ファイルシステムへのパブリックアクセスを管理するために役立つ設定を提供します。デフォルトでは、新しい EFS ファイルシステムではパブリックアクセスが許可されません。ただし、パブリックアクセスを許可するようにファイルシステムポリシーを変更することはできます。

Important

[パブリックアクセスをブロック] を有効にすると、ファイルシステムに直接アタッチされているリソースポリシーを通じてパブリックアクセスが付与されるのを防ぎ、リソースを保護することができます。[パブリックアクセスをブロック] を有効にすること以外にも、次のポリシーを慎重に検査して、パブリックアクセスを付与していないことを確認します。

- 関連付けられた AWS プリンシパルにアタッチされたアイデンティティベースのポリシー (IAM ロールなど)
- 関連付けられたリソースにアタッチされた AWS リソースベースのポリシー (例:AWS Key Management Service (KMS) キー)

トピック

- [AWS Transfer Familyを使用したパブリックアクセスのブロック](#)
- [「パブリック」の意味](#)

AWS Transfer Familyを使用したパブリックアクセスのブロック

Amazon EFS を で使用すると AWS Transfer Family、ファイルシステムがパブリックアクセスを許可すると、ファイルシステムとは異なるアカウントが所有する Transfer Family サーバーから受信したファイルシステムアクセスリクエストがブロックされます。Amazon EFS はファイルシステムの IAM ポリシーを評価し、ポリシーがパブリックである場合は、リクエストをブロックします。ファイルシステム AWS Transfer Family へのアクセスを許可するには、ファイルシステムポリシーを更新して、パブリックと見なされないようにします。

Note

Amazon EFS での Transfer Family の使用は、2021 AWS アカウント年 1 月 6 日より前に作成されたパブリックアクセスを許可するポリシーを持つ EFS ファイルシステムを持つ で

は、デフォルトで無効になっています。Transfer Family を使用してファイルシステムにアクセスできるようにするには、AWS サポートにお問い合わせください。

「パブリック」の意味

ファイルシステムがパブリックアクセスを許可するかどうかを評価する場合、Amazon EFS はファイルシステムポリシーがパブリックであると見なします。その後、ファイルシステムのポリシーを評価して、非パブリックとしての資格があるかどうかを判断します。非パブリックと見なすには、ファイルシステムポリシーは、次のうち 1 つ以上の固定値 (ワイルドカードを含まない値) にのみアクセスを許可する必要があります。

- `aws:SourceIp` を使用した一連のクラスレスドメイン間ルーティング (CIDR)。CIDR の詳細については、RFC Editor ウェブサイト [RFC 4632](https://www.rfc-editor.org/rfc/rfc4632) の「」を参照してください。
- AWS プリンシパル、ユーザー、ロール、またはサービスプリンシパル (例: `aws:PrincipalOrgID`)
- `aws:SourceArn`
- `aws:SourceVpc`
- `aws:SourceVpce`
- `aws:SourceOwner`
- `aws:SourceAccount`
- `elasticfilesystem:AccessedViaMountTarget`
- `aws:userid`, outside the pattern `"AROLEID:*"`

これらのルールでは、次のポリシー例はパブリックと見なされます。

```
{
  "Version": "2012-10-17",
  "Id": "efs-policy-wizard-15ad9567-2546-4bbb-8168-5541b6fc0e55",
  "Statement": [
    {
      "Sid": "efs-statement-14a7191c-9401-40e7-a388-6af6cfb7dd9c",
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
      },
      "Action": [
```

```

        "elasticfilesystem:ClientMount",
        "elasticfilesystem:ClientWrite",
        "elasticfilesystem:ClientRootAccess"
    ]
}
]
}

```

このファイルシステムポリシーを非公開にするには、EFS の条件キー `elasticfilesystem:AccessedViaMountTarget` を `[true]` に設定します。 `elasticfilesystem:AccessedViaMountTarget` を使用すると、ファイルシステムのマウントターゲットを使用して EFS ファイルシステムにアクセスするクライアントに対して、指定された EFS アクションを許可することができます。以下の非公開ポリシーでは、`elasticfilesystem:AccessedViaMountTarget` 条件のキーを `[true]` に設定して使用します。

```

{
  "Version": "2012-10-17",
  "Id": "efs-policy-wizard-15ad9567-2546-4bbb-8168-5541b6fc0e55",
  "Statement": [
    {
      "Sid": "efs-statement-14a7191c-9401-40e7-a388-6af6cfb7dd9c",
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
      },
      "Action": [
        "elasticfilesystem:ClientMount",
        "elasticfilesystem:ClientWrite",
        "elasticfilesystem:ClientRootAccess"
      ],
      "Condition": {
        "Bool": {
          "elasticfilesystem:AccessedViaMountTarget": "true"
        }
      }
    }
  ]
}

```

Amazon EFS における条件キーの詳細については、「[クライアントの EFS 条件キー](#)」を参照してください。ファイルシステムポリシーの作成の詳細については、「[ファイルシステムポリシーの作成](#)」を参照してください。

Amazon EFS のコンプライアンス検証

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範囲内」](#)を参照して、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、「[Downloading Reports in 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、お客様のデータの機密性、貴社のコンプライアンス目的、適用される法律および規制によって決まります。 は、コンプライアンスに役立つ以下のリソース AWS を提供します。

- [セキュリティのコンプライアンスとガバナンス](#) – これらのソリューション実装ガイドでは、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の機能をデプロイする手順を示します。
- [HIPAA 対応サービスのリファレンス](#) – HIPAA 対応サービスの一覧が提供されています。すべての AWS のサービス が HIPAA の対象となるわけではありません。
- [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、お客様の業界と地域に適用される場合があります。
- [AWS カスタマーコンプライアンスガイド](#) – コンプライアンスの観点から責任共有モデルを理解します。このガイドでは、複数のフレームワーク (米国国立標準技術研究所 (NIST)、Payment Card Industry Security Standards Council (PCI)、国際標準化機構 (ISO) など) にわたるセキュリティコントロールを保護し、そのガイダンスに AWS のサービス マッピングするためのベストプラクティスをまとめています。
- 「[デベロッパーガイド](#)」の「[ルールによるリソースの評価](#)」 – この AWS Config サービスは、リソース設定が内部プラクティス、業界ガイドライン、規制にどの程度準拠しているかを評価します。AWS Config
- [AWS Security Hub](#) – これにより AWS のサービス、内のセキュリティ状態を包括的に把握できます 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一覧については、[Security Hub のコントロールリファレンス](#)を参照してください。

- [Amazon GuardDuty](#) – 環境をモニタリングして AWS アカウント、疑わしいアクティビティや悪意のあるアクティビティがないか調べることで、ワークロード、コンテナ、データに対する潜在的な脅威 AWS のサービスを検出します。GuardDuty を使用すると、特定のコンプライアンスフレームワークで義務付けられている侵入検知要件を満たすことで、PCI DSS などのさまざまなコンプライアンス要件に対応できます。
- [AWS Audit Manager](#) – これにより AWS のサービス、AWS 使用状況を継続的に監査し、リスクの管理方法と規制や業界標準への準拠を簡素化できます。

Amazon EFS の耐障害性

AWS グローバルインフラストラクチャは、AWS リージョン およびアベイラビリティゾーン (AZs) を中心に構築されています。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 AZs AWS リージョン を提供します。AZ では、ゾーン間で中断なく自動的にフェイルオーバーするアプリケーションとデータベースを設計および運用することができます。AZ 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、拡張性に優れています。

Amazon EFS ファイルシステムは、AWS リージョン内の1つ以上のアベイラビリティゾーンの障害に対して回復力があります。マウントターゲットは高い可用性を実現できるように設計されています。高可用性と他の AZ へのフェイルオーバーを設計する場合、各 AZ のマウントターゲットの IP アドレスと DNS は静的ですが、これらは複数のリソースによってバックアップされた冗長コンポーネントであることに注意してください。詳細については、「[Amazon EFS と Amazon EC2 の連携方法](#)」を参照してください。

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。

Amazon EFS のネットワーク分離

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

AWS 公開された API コールを使用して、ネットワーク経由で Amazon EFS にアクセスします。クライアントは以下をサポートする必要があります。

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

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

これらの API は任意のネットワークの場所から呼び出すことができますが、Amazon EFS ではリソースベースのアクセスポリシーがサポートされているため、送信元 IP アドレスに基づく制限を含めることができます。また、Amazon EFS ポリシーを使用して、特定の Amazon Virtual Private Cloud (Amazon VPC) エンドポイントまたは特定の VPC からのアクセスを管理することもできます。これにより、実質的に、ネットワーク内の特定の VPC からのみ、特定の Amazon EFS リソースへの AWS ネットワークアクセスが分離されます。

Amazon EFS のクォータ

Amazon EFS を使用する際のクォータについて以下に説明します。

トピック

- [引き上げることができる Amazon EFS のクォータ](#)
- [変更できない Amazon EFS リソースクォータ](#)
- [NFS クライアントのクォータ](#)
- [Amazon EFS ファイルシステムのクォータ](#)
- [サポートされていない NFSv4.0 および 4.1 機能](#)
- [追加の考慮事項](#)
- [クォータ関連のファイル操作エラーのトラブルシューティング](#)

引き上げることができる Amazon EFS のクォータ

Service Quotas は、1 つの場所からクォータまたは制限を管理するのに役立つ AWS サービスです。[Service Quotas コンソール](#)では、Amazon EFS 制限値を表示し、内の EFS ファイルシステムの数 AWS リージョン と、頻繁にアクセスされるデータの読み取り IOPS のクォータ引き上げをリクエストできます。

AWS サポートに連絡して、次の Amazon EFS クォータの引き上げをリクエストすることもできます。詳細については[クォータ引き上げのリクエスト](#)を参照してください。Amazon EFS サービスチームは各リクエストを個別に確認します。

- カスタマーアカウントごとのファイルシステムの数。
- 各ファイルシステムのアクセスポイントの数。
- Elastic スループットを使用したファイルシステムあたりの最大読み取り IOPS。頻繁にアクセスされるファイルシステムの読み取り IOPS が増加すると、アクセス頻度の低いファイルシステムの読み取り IOPS と書き込み IOPS の両方も増加します。
- で接続されているすべてのクライアントのリージョンファイルシステムあたりのエラスティックスループット AWS リージョン。
- で接続されているすべてのクライアントのリージョンファイルシステムあたりのプロビジョンドスループット AWS リージョン。

次の表に、変更をリクエストできる一般的なデフォルトリソースクォータを示します。

リソース	デフォルトのクォータ
内の各顧客アカウントのファイルシステムの数 AWS リージョン	1,000
各ファイルシステムのアクセスポイント数	10,000
Elastic スループットを使用したファイルシステムあたりの最大 IOPS	アクセス頻度の低いデータの読み取り: 90,000 頻繁にアクセスされるデータの読み取り: 250,000 書き込み: 50,000
頻繁にアクセスされるデータの読み取り IOPS のデフォルトクォータの最大 10 倍の引き上げをリクエストできます。頻繁にアクセスされるデータの読み取り IOPS が増加すると、アクセス頻度の低いデータの読み取り IOPS と書き込み IOPS も増加します。	

次の表に、各 で接続されているすべてのクライアントのファイルシステムあたりの Elastic スループットクォータを示します AWS リージョン。

リージョンファイルシステム – 各 で接続されたすべてのクライアントについて、ファイルシステムあたりのデフォルトの Elastic スループットの合計 AWS リージョン

AWS リージョン	最大読み込みスループット	最大書き込みスループット (計測スループット)
米国東部 (オハイオ) リージョン	60 ギビバイト/秒 (GiBps)	5 GiBps

AWS リージョン	最大読み込みスループット	最大書き込みスループット (計測スループット)
米国東部(バージニア州北部) リージョン		
米国西部 (オレゴン) リージョン		
Asia Pacific (Mumbai) Region		
Asia Pacific (Seoul) Region		
アジアパシフィック (シンガポール) リージョン		
アジアパシフィック (シドニー) リージョン		
Asia Pacific (Tokyo) Region		
欧州 (フランクフルト) リージョン		
欧州 (アイルランド) リージョン		
欧州 (ロンドン) リージョン		
その他すべて AWS リージョン	20 GiBps	1 GiBps

次の表に、各 で接続されているすべてのクライアントのファイルシステムあたりのプロビジョンドスループットクォータを示します AWS リージョン。

リージョンファイルシステム – 各 で接続されているすべてのクライアントについて、ファイルシステムあたりのデフォルトのプロビジョンドスループットの合計 AWS リージョン

AWS リージョン	最大読み込みスループット	最大書き込みスループット (計測スループット)
米国東部 (オハイオ) リージョン	10 GiBps	3.33 GiBps
米国東部(バージニア州北部) リージョン		
米国西部 (オレゴン) リージョン		
欧州 (アイルランド) リージョン		
その他すべて AWS リージョン	3 GiBps	1 GiBps

クォータ引き上げのリクエスト

を通じてこれらのクォータの引き上げをリクエストするには AWS サポート、次の手順を実行します。Amazon EFS チームが各クォータの引き上げリクエストを検討します。

を使用してクォータの引き上げをリクエストするには AWS サポート

1. [AWS サポート センター](#)のページを開き、必要に応じてサインインします。続いて、[Create Case (ケースの作成)] を選択します。
2. [Create case (ケースの作成)] で [Service Limit Increase (サービスの上限緩和)] を選択します。
3. [Limit Type (制限のタイプ)] で、引き上げる制限のタイプを選択します。フォームの必要なフィールドに入力し、希望する連絡方法を選択します。

変更できない Amazon EFS リソースクォータ

以下を含むいくつかの Amazon EFS リソースのクォータは変更できません。

- 各ファイルシステムの接続数など、一般的なリソースのクォータ。
- 1 つの AWS リージョン内のすべての接続されたクライアントに対する、1 ゾーンファイルシステムあたりのエラスティックおよびプロビジョンドスループットクォータ。
- 1 つの AWS リージョン内のすべての接続されたクライアントに対する、リージョンまたは 1 ゾーンファイルシステムあたりのバーストスループットクォータ。

次の表に、変更できない一般的なリソースクォータを示します。

変更できない一般的なリソースクォータ

リソース	クォータ
各ファイルシステムの接続数	25,000
アベイラビリティゾーンごとのファイルシステムあたりのマウントターゲットの数	1
仮想プライベートクラウド (VPC) 毎のマウントターゲット数	1,400
マウントターゲットあたりのセキュリティグループの数	5
ファイルシステムあたりのタグの数	50
ファイルシステムあたりの VPC の数	1

Note

クライアントは、ファイルシステムとは異なるアカウントまたは VPC にあるマウントターゲットに接続することもできます。詳細については、「[別の AWS アカウント または VPC からの EFS ファイルシステムのマウント](#)」を参照してください。

次の表に、各 で接続されているすべてのクライアントについて、ファイルシステムあたりのデフォルトの Elastic スループット制限とプロビジョンドスループット制限の合計を示します AWS リージョン。

1 ゾーンファイルシステム - 各 AWS リージョン内のすべての接続されたクライアントに対する、ファイルシステムあたりのデフォルトのエラスティックおよびプロビジョンドスループットの合計

AWS リージョン	最大読み込みスループット	最大書き込みスループット (計測スループット)
すべて AWS リージョン	3 GiBps	1 GiBps

次の表に、各 で接続されているすべてのクライアントについて、ファイルシステムあたりのバーストスループット制限の合計を示します AWS リージョン。

リージョンファイルシステムと 1 ゾーンファイルシステム — 各 で接続されているすべてのクライアントに対するファイルシステムあたりの総バーストスループット AWS リージョン

AWS リージョン	最大読み込みスループット	最大書き込みスループット
米国東部 (オハイオ) リージョン	5 GiBps	3 GiBps
米国東部(バージニア州北部) リージョン		
米国西部 (オレゴン) リージョン		
アジアパシフィック (シドニー) リージョン		
欧州 (アイルランド) リージョン		
その他すべて AWS リージョン	3 GiBps	1 GiBps

NFS クライアントのクォータ

Linux NFSv4.1 クライアントを想定して、NFS クライアントに次のクォータが適用されます。

- エラスティックスループットを使用し、バージョン 2.0 以降の Amazon EFS クライアント (amazon-efs-utils バージョン) または Amazon EFS CSI ドライバー (aws-efs-csi-driver) を使用してマウントされたファイルシステムでは、読み取りと書き込みを合わせた最大スループットは 1,500 メビバイト/秒 (MiBps) になります。他のすべてのファイルシステムでは、最大スループットは 500 MiBps です。パフォーマンスの詳細については、「[パフォーマンスの概要](#)」を参照してください。NFS クライアントのスループットは、送受信された合計バイト数として計算されます。最小の NFS 要求サイズは 4 KB です (読み取り要求に 1/3 の計測率を適用した後)。
- 同時にファイルを開くことができるアクティブユーザー数は、クライアントあたり最大 65,536 人です。
- インスタンス上で同時に開くことができるファイル数は最大 65,536 個です。ディレクトリの内容を一覧表示することは、ファイルを開くこととしてカウントされません。
- クライアント上のそれぞれ固有のマウントは、接続ごとに合計 65,536 ロックまで取得できます。
- Amazon EFS に接続する場合、オンプレミスまたは別の AWS リージョンにある NFS クライアントは、同じ AWS リージョンから EFS に接続する場合よりもスループットは低くなります。この影響は、ネットワークレイテンシーが増加するためです。クライアントごとの最大スループットを達成するには、ネットワークレイテンシーを 1 ミリ秒以下にする必要があります。オンプレミス NFS サーバーから EFS に大量のデータセットを移行する場合は、DataSync データ移行サービスを使用してください。
- NFS プロトコルは 1 ユーザーあたり最大 16 のグループ ID (GID) をサポートし、追加の GID は NFS クライアント要求から切り捨てられます。詳細については、「[NFS ファイルシステム上の許可されたファイルへのアクセスが拒否されました](#)」を参照してください。
- Microsoft Windows での Amazon EFS の使用はサポートされていません。

Amazon EFS ファイルシステムのクォータ

以下のクォータは、Amazon EFS ファイルシステムに固有のものです。

リソース	クォータ
ファイル名の長さ (バイト単位)	255
シンボリックリンク (symlink) の長さ (バイト単位)	4,080
ファイルへのハードリンクの数	177

リソース	クォータ
ファイル 1 つのサイズ	52,673,613,135,872 バイト (47.9 TiB)
ディレクトリ深度のレベル数	1,000
すべてのインスタンスとユーザーを対象とするファイル 1 つのロック数	512
各ファイルシステムポリシーの文字数制限	20,000
* 汎用モードの 1 秒あたりのファイルオペレーション数	250,000

* 汎用モードの 1 秒あたりのファイルオペレーション数の詳細については、「[パフォーマンスの概要](#)」を参照してください。

サポートされていない NFSv4.0 および 4.1 機能

Amazon EFS では、NFSv2 と NFSv3 はサポートされていませんが、NFSv4.1 と NFSv4.0 は、どちらも次の機能を除いてサポートされています。

- pNFS
- 拡張属性
- 任意のタイプのクライアント委任またはコールバック
 - オペレーション OPEN は、委任タイプとして常に OPEN_DELEGATE_NONE を返します。
 - オペレーション OPEN は、CLAIM_DELEGATE_CUR と CLAIM_DELEGATE_PREV クレームタイプに対して、NFSERR_NOTSUPP を返します。
- 強制ロック

Amazon EFS のすべてのロックはアドバイザーです。つまり、オペレーションが実行される前に、読み取りと書き込みオペレーションで競合するロックがチェックされないことを意味します。

- 共有の拒否

NFS は、共有拒否の概念をサポートしています。共有拒否は、主に Windows クライアントが、開いている特定のファイルへの他のユーザーのアクセスを拒否するために使用します。Amazon EFS はこれをサポートしておらず、OPEN4_SHARE_DENY_NONE 以外の共有拒否値を指定する

クォータ関連のファイル操作エラーのトラブルシューティング

EFS ファイルシステムにアクセスすると、ファイルシステム内のファイルに特定の制限が適用されます。これらの制限を超えると、ファイル操作エラーが発生します。Amazon EFS でのファイルベースの制限の詳細については、「[Amazon EFS のクォータ](#)」を参照してください。

以下に、いくつかの一般的なファイル操作エラーと各エラーに関連する制限を示します。

トピック

- [「Disk quota exceeded」エラーでコマンドが失敗する](#)
- [「I/O error」でコマンドが失敗する](#)
- [「File name is too long」エラーでコマンドが失敗する](#)
- [「File not found」エラーでコマンドが失敗する](#)
- [「Too many links」エラーでコマンドが失敗する](#)
- [「File too large」エラーでコマンドが失敗する](#)

「Disk quota exceeded」エラーでコマンドが失敗する

Amazon EFS は現在、ユーザーディスククォータをサポートしていません。このエラーは、次のいずれかの制限を超えた場合に発生することがあります。

- 同時にファイルを開くことができるアクティブユーザー数は最大 65,536 人です。複数回ログインしたユーザーアカウントは 1 人のアクティブユーザーとしてカウントされます。
- 1 つのインスタンスで同時に開くことができるファイル数は最大 65,536 個です。ディレクトリの内容を一覧表示することは、ファイルを開くこととしてカウントされません。
- クライアント上のそれぞれ固有のマウントは、接続ごとに合計 65,536 ロックまで取得できます。

実行するアクション

この問題が発生した場合は、上記のいずれの制限を超過しているかを特定し、その制限を満たすように変更することで解決できます。詳細については、「[NFS クライアントのクォータ](#)」を参照してください。

「I/O error」でコマンドが失敗する

このエラーは、以下の問題のいずれかが発生したときに発生します。

- インスタンスあたり 65,536 を超えるアクティブユーザーアカウントが同時にファイルを開いています。

実行するアクション

この問題が発生した場合は、インスタンスで開いているファイルのサポートされている制限を満たすことで解決できます。これを行うには、インスタンス上で Amazon EFS ファイルシステムからのファイルを同時に開いているアクティブユーザーの数を減らします。

- ファイルシステムの AWS KMS 暗号化キーが削除されました。

実行するアクション

この問題が発生した場合、そのキーで暗号化されたデータを復号化できなくなります。これは、そのデータが回復不能になることを意味します。

「File name is too long」エラーでコマンドが失敗する

このエラーは、ファイル名またはシンボリックリンク (symlink) のサイズが長すぎる場合に発生します。ファイル名には以下の制限があります。

- 名前の長さは、最大 255 バイトまでです。
- シンボリックリンクのサイズは最大 4080 バイトまでです。

実行するアクション

この問題が発生した場合は、サポートされている制限を満たすようにファイル名またはシンボリックリンクの長さを縮小することで解決できます。

「File not found」エラーでコマンドが失敗する

このエラーは、Oracle E-Business スイートの一部の古い 32 ビットバージョンが 32 ビットのファイル I/O インターフェイスを使用し、EFS が 64 ビットの inode 番号を使用するために発生します。失敗する可能性のあるシステムコールには、stat () と readdir () があります。

実行するアクション

このエラーが発生した場合は、nfs.enable_ino64=0 kernel ブートオプションを使用して解決できます。このオプションでは、64 ビットの EFS inode 番号が 32 ビットに圧縮されます。カーネルブートオプションは、Linux ディストリビューションごとに異なる方法で処理されます。Amazon Linux

では、`/etc/default/grub` で `GRUB_CMDLINE_LINUX_DEFAULT` 変数に `nfs.enable_ino64=0` `kernel` を追加することで、このオプションをオンにします。カーネルブートオプションを有効にする方法については、お使いのディストリビューションのドキュメントを参照してください。

「Too many links」エラーでコマンドが失敗する

このエラーは、ファイルへのハードリンクが多すぎる場合に発生します。1 つのファイルに持つことができるハードリンクは最大 177 個までです。

実行するアクション

この問題が発生した場合は、ファイルへのハードリンクの数をサポートされている制限を満たすように減らすことで解決できます。

「File too large」エラーでコマンドが失敗する

このエラーは、ファイルが大きすぎる場合に発生します。1 つのファイルのサイズは、最大 52,673,613,135,872 バイト (47.9 TiB) までです。

実行するアクション

この問題が発生した場合は、サポートされている制限を満たすようにファイルのサイズを小さくすることで解決できます。

Amazon EFS API

Amazon EFS API は、[HTTP \(RFC 2616\)](https://tools.ietf.org/html/rfc2616) に基づくネットワークプロトコルです。API コールごとに、ファイルシステムを管理する AWS リージョン のリージョン固有の Amazon EFS API エンドポイントに HTTP リクエストを行います。API は、HTTP リクエスト/応答本文に JSON (RFC 4627) ドキュメントを使用します。

Amazon EFS API は RPC モデルです。このモデルでは、オペレーションの固定の設定があり、各オペレーションの構文は、事前に操作しなくてもクライアントに知られています。次のセクションでは、理論上の RPC 表記を使用する各 API オペレーションについて説明します。それぞれには、オンラインでは表示されないオペレーション名があります。各オペレーションでは、トピックが HTTP リクエスト要素のマッピングを指定します。

リクエストがマッピングされる特定の Amazon EFS オペレーションは、リクエストのメソッド (GET、PUT、POST、または DELETE) と、Request-URI がマッチする様々なパターンの組み合わせによって決定されます。オペレーションが PUT または POST の場合、Amazon EFS は Request-URI パスセグメント、クエリ パラメータ、およびリクエストボディ内の JSON オブジェクトからコール引数を抽出します。

Note

などのオペレーション名 `CreateFileSystem` はワイヤに表示されませんが、これらの名前は AWS Identity and Access Management (IAM) ポリシーで意味があります。詳細については、「[Amazon EFS のためのアイデンティティとアクセス管理](#)」を参照してください。オペレーション名は、コマンドラインツールのコマンド名や AWS SDK API の要素にも使用されます。たとえば、`CreateFileSystem` オペレーションにマッピング `create-file-system` される という名前の AWS CLI コマンドがあります。オペレーション名は、Amazon EFS API コールの AWS CloudTrail ログにも表示されます。

API エンドポイント

エンドポイントは、AWS ウェブサービスのエン트리ポイントとして機能する URL です。Amazon EFS は、次のエンドポイントタイプをサポートしています。

- IPv4 エンドポイント
- デュアルスタック (IPv4 と IPv6 の両方) エンドポイント

• [FIPS エンドポイント](#)

リクエストを行うと、使用するエンドポイントを指定できます。エンドポイントを指定しない場合、デフォルトで IPv4 エンドポイントが使用されます。別のエンドポイントタイプを使用するには、リクエストで指定する必要があります。これを行う方法の例については、「[エンドポイントの指定](#)」を参照してください。使用可能なエンドポイントの表については、「[Amazon EFS エンドポイントとクォータ](#)」を参照してください。

IPv4 エンドポイント

IPv4 エンドポイントは IPv4 トラフィックのみをサポートします。IPv4 エンドポイントは、全リージョンで利用できます。IPv4 エンドポイントの形式は、リージョンによって異なります。

- AWS 商用リージョンの IPv4 エンドポイントの場合、形式は `elasticfilesystem.region.amazonaws.com`。

たとえば、エンドポイント `elasticfilesystem.us-west-2.amazonaws.com` としてを指定した場合、リクエストは米国西部 (オレゴン) リージョン (us-west-2) エンドポイントに転送されます。

- 中国リージョンの IPv4 エンドポイントの場合、形式は `elasticfilesystem.region.amazonaws.com.cn`。

たとえば、中国 (北京) リージョンの Amazon EFS API エンドポイントは `elasticfilesystem.cn-north-1.amazonaws.com.cn`。

デュアルスタック (IPv4 および IPv6) エンドポイント

デュアルスタックエンドポイントは、IPv4 と IPv6 トラフィックの両方をサポートします。デュアルスタックエンドポイントにリクエストを行うと、エンドポイント URL は、ネットワークとクライアントが使用するプロトコルに応じて IPv6 または IPv4 アドレスに解決されます。デュアルスタックエンドポイントの形式は、リージョンによって異なります。

- AWS 商用リージョンのデュアルスタックエンドポイントの場合、形式は `efs.aws-region.api.aws`。

たとえば、米国西部 (オレゴン) リージョン (us-west-2) のデュアルスタックエンドポイントは `efs.eu-west-2.api.aws`。

- 中国リージョンのデュアルスタックエンドポイントの場合、形式は `efs.region.api.amazonwebservices.com.cn`。

たとえば、中国 (北京) リージョンのデュアルスタックエンドポイントは `efs.cn-north-1.api.amazonwebservices.com.cn`。

エンドポイントの指定

次の例は、Amazon EFS の を使用して米国西部 (北カリフォルニア) リージョン (us-west-1) AWS CLI のエンドポイントを指定する方法を示しています。

IPv4

```
aws efs get-rest-apis --region us-west-1 --endpoint-url https://elasticfilesystem.us-west-1.amazonaws.com
```

デュアルスタック

```
aws efs get-rest-apis --region us-west-1 --endpoint-url https://efs.us-west-1.api.aws
```

API バージョン

コールに使用される API のバージョンは、リクエスト URI の最初のパスセグメントにより特定されます。この形式は ISO 8601 の日付になります。例については、「[CreateFileSystem](#)」を参照してください。

ドキュメントでは、API バージョン 2015-02-01 について説明されています。

関連トピック

以下のセクションでは、API オペレーション、認証リクエスト用の署名を作成する方法、IAM ポリシーを使用して、これらの API オペレーションのためのアクセス許可を付与する方法を説明します。

- [Amazon EFS のためのアイデンティティとアクセス管理](#)
- [アクション](#)
- [データ型](#)

Amazon EFS のクエリ API リクエスト率の使用

Amazon EFS API リクエストは AWS アカウント、サービスのパフォーマンスを向上させるために、リージョンごとに スロットリングされます。すべての Amazon EFS API コールは、アプリケーション、AWS CLI Amazon EFS コンソールのいずれからのものであっても、許可される API リクエストの最大レートを超えることはできません。API リクエストの最大レートはさまざまです AWS リージョン。行われた API リクエストは、基盤となる に属性付けられます AWS アカウント。

API リクエストがそのカテゴリの API リクエスト率を超過する場合、`ThrottlingException` エラーコードが返されます。このエラーを回避するには、アプリケーションが API リクエストを再試行する率を低くします。これは、ポーリングの際に注意深くし、エクスポネンシャルバックオフの再試行を使用することにより行えます。

ポーリング

アプリケーションにより API オペレーションを繰り返しコールして、ステータスの更新をチェックする必要がある場合があります。ポーリングを開始する前に、リクエストの予想完了時間を指定します。ポーリングを開始するとき、連続するリクエストの間に適切なスリープ間隔を使用します。最良の結果を得るには、漸増スリープ間隔を使用します。

再試行またはバッチ処理

アプリケーションは、API リクエストが失敗した後に再試行するか、複数のリソースを処理する必要がある場合があります (たとえば、Amazon EFS ファイルシステムすべて)。API リクエストの率を下げるには、連続するリクエストの間に適切なスリープ間隔を使用します。最良の結果を得るには、漸増または可変スリープ間隔を使用します。

スリープ間隔の計算

API リクエストをポーリングまたは再試行する必要がある場合は、エクスポネンシャルバックオフアルゴリズムを使用して API コール間のスリープ間隔を計算することをお勧めします。エクスポネンシャルバックオフの背後にある考え方は、連続したエラー応答の再試行間の待機時間を徐々に長く使用することです。このアルゴリズムの詳細と実装例については、の [「再試行動作」](#) を参照してください Amazon Web Services 全般のリファレンス。

アクション

以下のアクションがサポートされています:

- [CreateAccessPoint](#)
- [CreateFileSystem](#)
- [CreateMountTarget](#)
- [CreateReplicationConfiguration](#)
- [CreateTags](#)
- [DeleteAccessPoint](#)
- [DeleteFileSystem](#)
- [DeleteFileSystemPolicy](#)
- [DeleteMountTarget](#)
- [DeleteReplicationConfiguration](#)
- [DeleteTags](#)
- [DescribeAccessPoints](#)
- [DescribeAccountPreferences](#)
- [DescribeBackupPolicy](#)
- [DescribeFileSystemPolicy](#)
- [DescribeFileSystems](#)
- [DescribeLifecycleConfiguration](#)
- [DescribeMountTargets](#)
- [DescribeMountTargetSecurityGroups](#)
- [DescribeReplicationConfigurations](#)
- [DescribeTags](#)
- [ListTagsForResource](#)
- [ModifyMountTargetSecurityGroups](#)
- [PutAccountPreferences](#)
- [PutBackupPolicy](#)
- [PutFileSystemPolicy](#)
- [PutLifecycleConfiguration](#)
- [TagResource](#)
- [UntagResource](#)
- [UpdateFileSystem](#)

- [UpdateFileSystemProtection](#)

CreateAccessPoint

EFS アクセスポイントの作成 アクセスポイントは、EFS ファイルシステムに対するアプリケーション固有のビューで、オペレーティングシステムのユーザーとグループ、およびファイルシステムパスを、アクセスポイントを介して行われたすべてのファイルシステムリクエストに適用されます。オペレーティングシステムのユーザーおよびグループは、NFS クライアントから提供されるすべてのアイデンティティ情報を上書きします。ファイルシステムのパスは、アクセスポイントのルートディレクトリとしてクライアントに公開されます。アクセスポイントを使用するアプリケーションは、アプリケーション自体のディレクトリとサブディレクトリ内のデータにのみアクセスできます。ファイルシステムは、引き上げをリクエストしない限り、最大 10,000 個のアクセスポイントを持つことができます。詳細はこちら、[「EFS アクセスポイントを使用してファイルシステムをマウントする」](#)を参照してください。

Note

同じファイルシステムにアクセスポイントを作成する複数のリクエストが迅速に連続して送信され、ファイルシステムがアクセスポイントの制限に近い場合、これらのリクエストに対してスロットリングレスポンスが発生する可能性があります。これは、ファイルシステムが指定されたアクセスポイントの制限を超えないようにするためです。

このオペレーションには、`elasticfilesystem:CreateAccessPoint` アクションに対するアクセス許可が必要です。

アクセスポイントは作成時にタグ付けをすることができます。作成アクションでタグが指定されている場合、IAM は `elasticfilesystem:TagResource` アクションに対して追加の認可を実行して、ユーザーがタグを作成する認可を持っているかどうかを確認します。したがって、`elasticfilesystem:TagResource` アクションを使用するための明示的なアクセス許可を付与する必要があります。詳細については、[「リソース作成時にタグ付けするアクセス許可の付与」](#)を参照してください。

リクエストの構文

```
POST /2015-02-01/access-points HTTP/1.1
Content-type: application/json

{
  "ClientToken": "string",
  "FileSystemId": "string",
```

```
"PosixUser": {
  "Gid": number,
  "SecondaryGids": [ number ],
  "Uid": number
},
"RootDirectory": {
  "CreationInfo": {
    "OwnerGid": number,
    "OwnerUid": number,
    "Permissions": "string"
  },
  "Path": "string"
},
"Tags": [
  {
    "Key": "string",
    "Value": "string"
  }
]
}
```

URI リクエストパラメータ

リクエストでは URI パラメータを使用しません。

リクエストボディ

リクエストは以下の JSON 形式のデータを受け入れます。

ClientToken

べき等の作成を保証するために Amazon EFS が使用する最大 64 の ASCII 文字の文字列。

タイプ: 文字列

長さの制約: 最小長は 1 です。最大長 64

パターン: .+

必須: はい

FileSystemId

アクセスポイントがアクセスを提供する EFS ファイルシステムの ID。

タイプ: 文字列

長さの制約: 最大長は 128 です。

パターン: `^(arn:aws[-a-z]*:elasticfilesystem:[0-9a-z-:]+:file-system/fs-[0-9a-f]{8,40}|fs-[0-9a-f]{8,40})$`

必須: はい

[PosixUser](#)

アクセスポイントを使用して行われたすべてのファイル システムリクエストに適用されたオペレーティングシステムのユーザーおよびグループ。

型: [PosixUser](#) オブジェクト

必須: いいえ

[RootDirectory](#)

アクセスポイントを使用する NFS クライアントに対して、アクセスポイントがファイルシステムのルートディレクトリとして公開する EFS ファイルシステム上のディレクトリを指定します。アクセスポイントを使用するクライアントは、ルートディレクトリ以下にのみアクセスできます。指定した `RootDirectory > Path` が存在しない場合、Amazon EFS はそれを作成し、クライアントがアクセスポイントに接続する際に `CreationInfo` 設定を適用します。RootDirectory を指定する際には、Path、CreationInfo を指定する必要があります。

Amazon EFS では、`CreationInfo` (`OwnUid`、`OwnGID`、ディレクトリのアクセス許可) を指定した場合にのみ、ルートディレクトリが作成されます。この情報を指定しない場合、Amazon EFS はルートディレクトリを作成しません。ルートディレクトリが存在しない場合に、アクセスポイントを使用してマウントしようとすると、失敗します。

型: [RootDirectory](#) オブジェクト

必須: いいえ

[Tags](#)

アクセスポイントに関連付けられたタグを作成します。各タグはキーバリューのペアです。各キーは一意である必要があります。詳細については、「[AWS 全般のリファレンスガイド](#)」の [AWS 「リソースのタグ付け」](#) を参照してください。

タイプ: [Tag](#) オブジェクトの配列

必須: いいえ

レスポンスの構文

```
HTTP/1.1 200
Content-type: application/json

{
  "AccessPointArn": "string",
  "AccessPointId": "string",
  "ClientToken": "string",
  "FileSystemId": "string",
  "LifeCycleState": "string",
  "Name": "string",
  "OwnerId": "string",
  "PosixUser": {
    "Gid": number,
    "SecondaryGids": [ number ],
    "Uid": number
  },
  "RootDirectory": {
    "CreationInfo": {
      "OwnerGid": number,
      "OwnerUid": number,
      "Permissions": "string"
    },
    "Path": "string"
  },
  "Tags": [
    {
      "Key": "string",
      "Value": "string"
    }
  ]
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

[AccessPointArn](#)

アクセスポイントに関連付けられた一意の Amazon リソースネーム (ARN)。

タイプ: 文字列

長さの制約: 最大長は 128 です。

パターン: `^arn:aws[-a-z]*:elasticfilesystem:[0-9a-z-:]+:access-point/fsap-[0-9a-f]{8,40}$`

[AccessPointId](#)

Amazon EFS によって割り当てられたアクセスポイントの ID。

タイプ: 文字列

長さの制約: 最大長は 128 です。

パターン: `^(arn:aws[-a-z]*:elasticfilesystem:[0-9a-z-:]+:access-point/fsap-[0-9a-f]{8,40}|fsap-[0-9a-f]{8,40})$`

[ClientToken](#)

冪等性の作成を保証するためにリクエストで指定された不透明な文字列。

タイプ: 文字列

長さの制約: 最小長は 1 です。最大長 64

パターン: `.+`

[FileSystemId](#)

アクセスポイントが適用される EFS ファイルシステムの ID。

タイプ: 文字列

長さの制約: 最大長は 128 です。

パターン: `^(arn:aws[-a-z]*:elasticfilesystem:[0-9a-z-:]+:file-system/fs-[0-9a-f]{8,40}|fs-[0-9a-f]{8,40})$`

[LifeCycleState](#)

アクセスポイントのライフサイクルフェーズを識別します。

型: 文字列

有効な値: `creating | available | updating | deleting | deleted | error`

[Name](#)

このアクセスポイントの名前。これは、Name タグの値です。

タイプ: 文字列

[OwnerId](#)

アクセスポイントリソースを所有 AWS アカウント する を識別します。

タイプ: 文字列

長さの制限: 最大長は 14 です。

パターン: `^(\d{12})|(\d{4}-\d{4}-\d{4})$`

[PosixUser](#)

アクセスポイント上の、ユーザー ID、グループ ID、およびセカンダリグループ ID を含む完全な POSIX アイデンティティ。アクセスポイントを使用する NFS クライアントによるすべてのファイル オペレーションに使用されます。

型: [PosixUser](#) オブジェクト

[RootDirectory](#)

アクセスポイントを使用して、アクセスポイントが NFS クライアントにルートディレクトリとして公開する EFS ファイルシステム上のディレクトリ。

型: [RootDirectory](#) オブジェクト

[Tags](#)

タグオブジェクトの配列として表示される、アクセスポイントに関連付けられたタグ。

型: [Tag](#) オブジェクトの配列

エラー

AccessPointAlreadyExists

作成しようとしているアクセスポイントがすでに存在し、リクエストで指定した作成 トークンとともに返されます。

HTTP ステータスコード: 409

AccessPointLimitExceeded

AWS アカウント がファイルシステムごとに許可されるアクセスポイントの最大数を既に作成している場合に返されます。詳細については、「<https://docs.aws.amazon.com/efs/latest/ug/limits.html#limits-efs-resources-per-account-per-region>」を参照してください。

HTTP ステータスコード: 403

BadRequest

リクエストの形式が正しくない場合や、無効なパラメータ値や必須パラメータがないなどのエラーが含まれている場合に返されます。

HTTP ステータスコード: 400

FileSystemNotFound

指定されたFileSystemId値がリクエストの に存在しない場合に返されます AWS アカウント。

HTTP ステータスコード: 404

IncorrectFileSystemLifecycleState

ファイルシステムのライフサイクル状態が「使用可能」でない場合に返されます。

HTTP ステータスコード: 409

InternalServerError

サーバー側でエラーが発生した場合に返されます。

HTTP ステータスコード: 500

ThrottlingException

CreateAccessPoint API アクションの呼び出しが速すぎて、ファイルシステム上のアクセスポイントの数が [120 の制限](#) に近づいた場合に返されます。

HTTP ステータスコード: 429

以下も参照してください。

言語固有の AWS SDKs のいずれかでこの API を使用方法の詳細については、以下を参照してください。

- [AWS コマンドラインインターフェイス](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go v2](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript V3](#)
- [AWS SDK for Kotlin](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

CreateFileSystem

空のファイルシステムを新しく作成します。このオペレーションでは、Amazon EFS がべき等の作成を保証するために使用するリクエストに作成トークンが必要です (同じ作成トークンを使用してオペレーションを呼び出しても効果はありません)。指定された作成トークン AWS アカウント を持つ発信者の が所有するファイルシステムが現在存在しない場合、このオペレーションは以下を実行します。

- 空のファイルシステムを新しく作成します。ファイルシステムには、Amazon EFS に割り当てられた ID と、初期ライフサイクル状態 `creating` があります。
- 作成したファイルシステムの説明を返します。

それ以外の場合、このオペレーションは既存のファイルシステムの ID とともに `FileSystemAlreadyExists` エラーを返します。

Note

基本的なユースケースでは、作成トークンにランダムに生成された UUID を使用できます。

べき等のオペレーションを使用すると、余分なファイルシステムを作成するリスクなしに `CreateFileSystem` コールを再試行できます。これは、ファイルシステムが実際に作成されたかどうか不明確なまま、最初のコールが失敗したときに発生する可能性があります。例としては、トランスポートレベルのタイムアウトが発生したか、接続がリセットされたことが挙げられます。同じ作成トークンを使用している限り、最初のコールでファイルシステムの作成に成功した場合、クライアントは `FileSystemAlreadyExists` エラーからその存在を知ることができます。

詳細については、[Amazon EFS ユーザーガイド](#)の「ファイルシステムの作成」を参照してください。

Note

`CreateFileSystem` コールは、ファイルシステムのライフサイクル状態がまだ `creating` である間に戻ります。[DescribeFileSystems](#) オペレーションをコールすることでファイルシステムの作成ステータスをチェックできます。これにより、ファイルシステムの状態が返されます。

このオペレーションでは、ファイルシステム用に選択したオプションの PerformanceMode パラメータを受け入れます。すべてのファイルシステムに generalPurpose PerformanceMode をお勧めします。maxIO モードは前世代のパフォーマンスタイプであり、generalPurpose モードよりも高いレイテンシーを許容できる高度に並列化されたワークロード向けに設計されています。MaxIO モードは、1 ゾーンファイルシステムやエラスティックスループットを使用するファイルシステムではサポートされません。

PerformanceMode は、ファイルシステムの作成後は変更できません。詳細については、「[Amazon EFS パフォーマンスモード](#)」を参照してください。

ファイルシステムのスループットモードは、ThroughputMode パラメータで設定できます。

ファイルシステムが完全に作成されると、Amazon EFS はそのライフサイクル状態を available に設定します。この時点で、VPC でファイルシステムの 1 つ以上のマウントターゲットを作成できます。詳細については、「[CreateMountTarget](#)」を参照してください。マウントターゲットを使用して、Amazon EFS ファイルシステムを VPC 内の EC2 インスタンスにマウントします。詳細については、「[Amazon EFS の仕組み](#)」を参照してください。

このオペレーションには、elasticfilesystem:CreateFileSystem アクションに対するアクセス許可が必要です。

ファイルシステムは作成時にタグ付けできます。作成アクションでタグが指定されている場合、IAM は elasticfilesystem:TagResource アクションに対して追加の認可を実行して、ユーザーがタグを作成する認可を持っているかどうかを確認します。したがって、elasticfilesystem:TagResource アクションを使用するための明示的なアクセス許可を付与する必要があります。詳細については、「[リソース作成時にタグ付けするアクセス許可の付与](#)」を参照してください。

リクエストの構文

```
POST /2015-02-01/file-systems HTTP/1.1
Content-type: application/json

{
  "AvailabilityZoneName": "string",
  "Backup": boolean,
  "CreationToken": "string",
  "Encrypted": boolean,
  "KmsKeyId": "string",
  "PerformanceMode": "string",
  "ProvisionedThroughputInMibps": number,
```

```
"Tags": [  
  {  
    "Key": "string",  
    "Value": "string"  
  }  
],  
"ThroughputMode": "string"  
}
```

URI リクエストパラメータ

リクエストでは URI パラメータを使用しません。

リクエストボディ

リクエストは以下の JSON 形式のデータを受け入れます。

AvailabilityZoneName

1 ゾーンファイルシステムの場合は、ファイルシステムを作成する AWS アベイラビリティーゾーンを指定します。形式 us-east-1a を使用して、アベイラビリティーゾーンを指定します。1 ゾーンファイルシステムの詳細については、「Amazon EFS ユーザーガイド」の「[EFS ファイルシステムのタイプ](#)」を参照してください。

Note

1 ゾーンファイルシステムは、Amazon EFS AWS リージョン が利用可能な のすべてのアベイラビリティーゾーンで利用できるわけではありません。

タイプ: 文字列

長さの制約: 最小長は 1 です。最大長 64

パターン: .+

必須: いいえ

Backup

作成するファイルシステム上で自動バックアップを有効にするかどうかを指定します。自動バックアップを有効にするには、値を true に設定します。1 ゾーンファイルシステムを作成してい

る場合、デフォルトで自動バックアップが有効になります。詳細については、Amazon EFS ユーザーガイドの「[自動バックアップ](#)」を参照してください。

デフォルトは false です。ただし、AvailabilityZoneName を指定した場合、デフォルトは true です。

 Note

AWS Backup は、Amazon EFS が利用可能なすべての AWS リージョン で利用できるわけではありません。

型: ブール値

必須: いいえ

CreationToken

最大 64 バイトの ASCII 文字の文字列。Amazon EFS では、べき等の作成を保証するためにこれを使用します。

タイプ: 文字列

長さの制約: 最小長は 1 です。最大長 64

パターン: .+

必須: はい

Encrypted

このブール値が true の場合、暗号化されたファイルシステムが作成されます。暗号化されたファイルシステムを作成するときは、既存の AWS Key Management Service キー (KMS キー) を指定するオプションがあります。KMS キーを指定しない場合、Amazon EFS のデフォルトの KMS キー (/aws/elasticfilesystem) が暗号化されたファイルシステムの保護に使用されます。

型: ブール値

必須: いいえ

KmsKeyId

暗号化されたファイルシステムの保護に使用される KMS キーの ID。このパラメータは、デフォルト以外の KMS キーを使用する場合にのみ必要です。このパラメータが指定されない場

合、Amazon EFS のデフォルトの KMS キーが使用されます。以下の形式を使用して、KMS キー ID を指定できます。

- キー ID - キーの一意の識別子 (例: 1234abcd-12ab-34cd-56ef-1234567890ab)。
- ARN - キーの Amazon リソースネーム (ARN) (例: arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab)。
- キーのエイリアス - 以前に作成したキーの表示名 (例: alias/projectKey1)。
- キー エイリアス ARN - キー エイリアスの ARN (例: arn:aws:kms:us-west-2:444455556666:alias/projectKey1)。

KmsKeyId を使用した場合、[CreateFileSystem:Encrypted](#) パラメータを true に設定する必要があります。

Important

EFS は、対称 KMS キーのみを受け入れます。Amazon EFS ファイルシステムでは非対称 KMS キーを使用することはできません。

タイプ: 文字列

長さの制約: 最大長は 2,048 です。

パターン: `^([0-9a-f]{8}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{12}|mrk-[0-9a-f]{32}|alias/[a-zA-Z0-9/_-]+|(arn:aws[-a-z]*:kms:[a-z0-9-]+:\d{12}:((key/[0-9a-f]{8}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{12})|(key/mrk-[0-9a-f]{32})|(alias/[a-zA-Z0-9/_-]+))))$`

必須: いいえ

[PerformanceMode](#)

ファイル システムのパフォーマンス モード。すべてのファイルシステムに generalPurpose パフォーマンスモードをお勧めします。maxIO パフォーマンスモードを使用しているファイル システムでは、ほとんどのファイルオペレーションのレイテンシーがわずかに大きくなる代わりに、より高いレベルの集計スループットと 1 秒あたりのオペレーションにスケールできます。パフォーマンスモードは、ファイルシステムの作成後は変更できません。maxIO モードは、1 ゾーンファイルシステムではサポートされません。

⚠ Important

最大 I/O ではオペレーションごとのレイテンシーが高くなるため、すべてのファイルシステムに汎用パフォーマンスモードを使用することをお勧めします。

デフォルトは `generalPurpose` です。

型: 文字列

有効な値: `generalPurpose` | `maxIO`

必須: いいえ

ProvisionedThroughputInMibps

作成しているファイルシステムにプロビジョニングするスループットを、毎秒メビバイト (MiBps) で計測します。ThroughputMode が `provisioned` に設定されている場合は必須です。有効な値は 1~3414 MiBps で、上限はリージョンによって異なります。この制限を引き上げるには、[にお問い合わせください サポート](#)。詳細については、Amazon EFS ユーザーガイドの「[引き上げ可能な Amazon EFS クォータ](#)」を参照してください。

型: 倍精度浮動小数点数

値の範囲: 最小値 は 1.0 です。

必須: いいえ

Tags

ファイルシステムに関連付けられている 1 つ以上のタグを作成するために使用します。各タグはユーザー定義のキーバリューのペアです。"Key": "Name", "Value": "{value}" キーバリューのペアを含めることで、作成時にファイルシステムに名前を付けます。各キーは一意である必要があります。詳細については、「[AWS 全般のリファレンスガイド](#)」の [AWS 「リソースのタグ付け」](#) を参照してください。

タイプ: [Tag](#) オブジェクトの配列

必須: いいえ

ThroughputMode

ファイルシステムのスループットモードを指定します。モードには、`bursting`、`provisioned`、`elastic` などがあります。ThroughputMode を

provisioned に設定した場合、ProvisionedThroughputInMibps の値も設定する必要があります。ファイルシステムを作成した後は、一定の時間制限付きで、ファイルシステムのプロビジョニングされたスループットを下げたり、スループットモードを変更したりすることができます。詳細については、Amazon EFS ユーザーガイドの「[プロビジョニングモードでのスループットの指定](#)」を参照してください。

デフォルトは bursting です。

型: 文字列

有効な値: bursting | provisioned | elastic

必須: いいえ

レスポンスの構文

```
HTTP/1.1 201
Content-type: application/json

{
  "AvailabilityZoneId": "string",
  "AvailabilityZoneName": "string",
  "CreationTime": number,
  "CreationToken": "string",
  "Encrypted": boolean,
  "FileSystemArn": "string",
  "FileSystemId": "string",
  "FileSystemProtection": {
    "ReplicationOverwriteProtection": "string"
  },
  "KmsKeyId": "string",
  "LifeCycleState": "string",
  "Name": "string",
  "NumberOfMountTargets": number,
  "OwnerId": "string",
  "PerformanceMode": "string",
  "ProvisionedThroughputInMibps": number,
  "SizeInBytes": {
    "Timestamp": number,
    "Value": number,
    "ValueInArchive": number,
    "ValueInIA": number,
```



```
    "ValueInStandard": number
  },
  "Tags": [
    {
      "Key": "string",
      "Value": "string"
    }
  ],
  "ThroughputMode": "string"
}
```

レスポンス要素

アクションが成功すると、HTTP 201 レスポンスが返されます。

サービスから以下のデータが JSON 形式で返されます。

[AvailabilityZoneId](#)

ファイルシステムが配置されているアベイラビリティゾーンの一意で一貫性のある識別子。これは、1 ゾーンファイルシステムでのみ有効です。たとえば、use1-az1は us-east-1 のアベイラビリティゾーン ID であり AWS リージョン、すべての に同じ場所があります AWS アカウント。

タイプ: 文字列

[AvailabilityZoneName](#)

ファイルシステムが配置されている AWS アベイラビリティゾーンについて説明し、1 ゾーンファイルシステムでのみ有効です。詳細については、「Amazon EFS ユーザーガイド」の「[EFS ストレージクラスの使用](#)」を参照してください。

タイプ: 文字列

長さの制約: 最小長は 1 です。最大長 64

パターン: .+

[CreationTime](#)

ファイルシステムが作成された時間 (秒単位) (1970-01-01T00:00:00Z から)。

型: タイムスタンプ

[CreationToken](#)

リクエストで指定された不透明な文字列。

タイプ: 文字列

長さの制約: 最小長は 1 です。最大長 64

パターン: .+

[Encrypted](#)

true の場合はファイルシステムの暗号化を示すブール値。

型: ブール値

[FileSystemArn](#)

EFS ファイルシステムの Amazon リソースネーム (ARN) で、形式は `arn:aws:elasticfilesystem:region:account-id:file-system/file-system-id` です。サンプルデータの例: `arn:aws:elasticfilesystem:us-west-2:1111333322228888:file-system/fs-01234567`

タイプ: 文字列

[FileSystemId](#)

Amazon EFS によって割り当てられるファイルシステムの ID。

タイプ: 文字列

長さの制約: 最大長は 128 です。

パターン: `^(arn:aws[-a-z]*:elasticfilesystem:[0-9a-z-:]+:file-system/fs-[0-9a-f]{8,40}|fs-[0-9a-f]{8,40})$`

[FileSystemProtection](#)

ファイルシステムの保護について説明します。

型: [FileSystemProtectionDescription](#) オブジェクト

[KmsKeyId](#)

暗号化されたファイルシステムの保護 AWS KMS key に使用される の ID。

タイプ: 文字列

長さの制約: 最大長は 2,048 です。

パターン: `^([0-9a-f]{8}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{12}|mrk-[0-9a-f]{32}|alias/[a-zA-Z0-9/_-]+|(arn:aws[-a-z]*:kms:[a-z0-9-]+:\d{12}:((key/[0-9a-f]{8}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{12})|(key/mrk-[0-9a-f]{32})|(alias/[a-zA-Z0-9/_-]+))))$`

LifeCycleState

ファイルシステムのライフサイクルフェーズ。

型: 文字列

有効な値: `creating | available | updating | deleting | deleted | error`

Name

ファイルシステムには、Name タグをはじめとするタグを追加することができます。詳細については、「[CreateFileSystem](#)」を参照してください。ファイルシステムに Name タグがある場合、Amazon EFS はこのフィールドの値を返します。

タイプ: 文字列

長さの制約: 最大長は 256 です。

パターン: `^([\p{L}\p{Z}\p{N}_.: /+=\-\@]*)$`

NumberOfMountTargets

ファイルシステムが持つ現在のマウントターゲットの数。詳細については、「[CreateMountTarget](#)」を参照してください。

型: 整数

有効範囲: 最小値 は 0 です。

OwnerId

ファイルシステムを AWS アカウント 作成した。

タイプ: 文字列

長さの制限: 最大長は 14 です。

パターン: `^(\d{12})|(\d{4}-\d{4}-\d{4})$`

[PerformanceMode](#)

ファイル システムのパフォーマンス モード。

型: 文字列

有効な値: `generalPurpose` | `maxIO`

[ProvisionedThroughputInMibps](#)

ファイルシステムのプロビジョニングされたスループットの量を MiBps で表したものの。ThroughputMode を provisioned に設定したファイルシステムに有効です。

型: 倍精度浮動小数点数

有効な範囲: 最小値 は 1.0 です。

[SizeInBytes](#)

ファイルシステムに保存されているデータの最新の測定サイズ (バイト) を Value のフィールドに、そのサイズが決定された時間を Timestamp のフィールドに入力しています。Timestamp 値は、1970-01-01T00:00:00Z 以降の整数秒数です。SizeInBytes値は、ファイルシステムの一貫したスナップショットのサイズを表すものではありませんが、ファイルシステムへの書き込みがない場合に結果整合性があります。つまり、SizeInBytesは、ファイルシステムが 2 時間以上変更されていない場合のみ、実際のサイズを表します。それ以外の場合、値はファイルシステムの特定の時点での正確なサイズではありません。

型: [FileSystemSize](#) オブジェクト

[Tags](#)

ファイルシステムに関連するタグで、Tag のオブジェクトの配列として表示されます。

型: [Tag](#) オブジェクトの配列

[ThroughputMode](#)

ファイルシステムのスループットモードを表示。詳細については、「Amazon EFS ユーザーガイド」の「[スループットモード](#)」を参照してください。

型: 文字列

有効な値: `bursting` | `provisioned` | `elastic`

エラー

BadRequest

リクエストの形式が正しくない場合や、無効なパラメータ値や必須パラメータの欠落などのエラーが含まれている場合に返されます。

HTTP ステータスコード: 400

FileSystemAlreadyExists

作成しようとしているファイルシステムがすでに存在し、指定した作成トークンを使用して返されます。

HTTP ステータスコード: 409

FileSystemLimitExceeded

AWS アカウント がアカウントごとに許可されるファイルシステムの最大数を既に作成している場合に返されます。

HTTP ステータスコード: 403

InsufficientThroughputCapacity

追加のスループットをプロビジョニングするのに十分な容量がない場合に返されます。この値は、プロビジョニングされたスループットモードでファイルシステムを作成しようとしたとき、既存のファイルシステムのプロビジョニングされたスループットを上げようとしたとき、または既存のファイルシステムをバーストからプロビジョニングされたスループットモードに変更しようとしたときに返されることがあります。後でもう一度お試しください。

HTTP ステータスコード: 503

InternalServerError

サーバー側でエラーが発生した場合に返されます。

HTTP ステータスコード: 500

ThroughputLimitExceeded

スループット制限の 1024 MiB/s に達したため、スループットモードまたはプロビジョニングされたスループットの量を変更できない場合に返されます。

HTTP ステータスコード: 400

UnsupportedAvailabilityZone

リクエストされた Amazon EFS 機能が指定されたアベイラビリティーゾーンで利用できない場合に返されます。

HTTP ステータスコード: 400

例

暗号化された EFS ファイルシステムの作成

次の例では、POST リクエストを送信して、自動バックアップが有効になっている us-west-2 リージョンにファイルシステムを作成します。このリクエストでは、べき等の作成トークンとして myFileSystem1 を指定します。

リクエスト例

```
POST /2015-02-01/file-systems HTTP/1.1
Host: elasticfilesystem.us-west-2.amazonaws.com
x-amz-date: 20140620T215117Z
Authorization: <...>
Content-Type: application/json
Content-Length: 42

{
  "CreationToken" : "myFileSystem1",
  "PerformanceMode" : "generalPurpose",
  "Backup": true,
  "Encrypted": true,
  "Tags":[
    {
      "Key": "Name",
      "Value": "Test Group1"
    }
  ]
}
```

レスポンス例

```
HTTP/1.1 201 Created
x-amzn-RequestId: 01234567-89ab-cdef-0123-456789abcdef
Content-Type: application/json
```

Content-Length: 319

```
{
  "ownerId": "251839141158",
  "CreationToken": "myFileSystem1",
  "Encrypted": true,
  "PerformanceMode": "generalPurpose",
  "fileSystemId": "fs-01234567",
  "CreationTime": "1403301078",
  "LifeCycleState": "creating",
  "numberOfMountTargets": 0,
  "SizeInBytes": {
    "Timestamp": 1403301078,
    "Value": 29313618372,
    "ValueInArchive": 201156,
    "ValueInIA": 675432,
    "ValueInStandard": 29312741784
  },
  "Tags": [
    {
      "Key": "Name",
      "Value": "Test Group1"
    }
  ],
  "ThroughputMode": "elastic"
}
```

1 ゾーンの可用性を備えた暗号化 EFS ファイルシステムを作成する

次の例では、POST リクエストを送信して、自動バックアップが有効になっている us-west-2 リージョンにファイルシステムを作成します。ファイルシステムには、us-west-2b アベイラビリティーゾーンに 1 ゾーンストレージがあります。

リクエスト例

```
POST /2015-02-01/file-systems HTTP/1.1
Host: elasticfilesystem.us-west-2.amazonaws.com
x-amz-date: 20140620T215117Z
Authorization: <...>
Content-Type: application/json
Content-Length: 42

{
```

```
"CreationToken" : "myFileSystem2",
"PerformanceMode" : "generalPurpose",
"Backup": true,
"AvailabilityZoneName": "us-west-2b",
"Encrypted": true,
"ThroughputMode": "elastic",
"Tags":[
  {
    "Key": "Name",
    "Value": "Test Group1"
  }
]
```

レスポンス例

```
HTTP/1.1 201 Created
x-amzn-RequestId: 01234567-89ab-cdef-0123-456789abcdef
Content-Type: application/json
Content-Length: 319
```

```
{
  "ownerId":"251839141158",
  "CreationToken":"myFileSystem1",
  "Encrypted": true,
  "AvailabilityZoneId": "usew2-az2",
  "AvailabilityZoneName": "us-west-2b",
  "PerformanceMode" : "generalPurpose",
  "fileSystemId":"fs-01234567",
  "CreationTime":"1403301078",
  "LifeCycleState":"creating",
  "numberOfMountTargets":0,
  "SizeInBytes":{"
    "Timestamp": 1403301078,
    "Value": 29313618372,
    "ValueInArchive": 201156,
    "ValueInIA": 675432,
    "ValueInStandard": 29312741784
  },
  "Tags":[
    {
      "Key": "Name",
      "Value": "Test Group1"
    }
  ]
}
```



```
    }  
  ],  
  "ThroughputMode": "elastic"  
}
```

以下の資料も参照してください。

言語固有の AWS SDKs のいずれかでこの API を使用方法の詳細については、以下を参照してください。

- [AWS コマンドラインインターフェイス](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go v2](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript V3](#)
- [AWS SDK for Kotlin](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

CreateMountTarget

ファイルシステムのマウントターゲットを作成します。その後、マウントターゲットを使用してファイルシステムを EC2 インスタンスにマウントできます。マウントターゲットの詳細については、[「マウントターゲットの管理」](#)を参照してください。

EFS ファイルシステムのマウントターゲットを作成するには、ファイルシステムのライフサイクル状態が `available` である必要があります。詳細については、[「DescribeFileSystems」](#)を参照してください。

マウントターゲットを作成すると、Amazon EFS は `MountTargetId` と IP アドレス (`IpAddress` または `Ipv6Address`) を含むレスポンスを返します。この IP アドレスは、ファイルシステムを EC2 インスタンスにマウントするときに使用します。ファイルシステムをマウントするときにマウントターゲットの DNS 名を使用することもできます。マウントターゲットを使用してファイルシステムをマウントする EC2 インスタンスは、マウントターゲットの DNS 名をその IP アドレスに解決できます。詳細については、[「Amazon EFS の仕組み」](#)を参照してください。

ファイルシステムのマウントターゲットは 1 つの VPC でのみ作成でき、アベイラビリティゾーンごとにマウントターゲットは 1 つだけ存在できます。詳細については、[「マウントターゲットの作成」](#)を参照してください。

リクエストが要件を満たす場合、Amazon EFS は次のことを行います。

- 指定したサブネットに新しいマウントターゲットを作成します。
- サブネットに次の `NetworkInterfaceId` を含む新しいネットワークインターフェイスを作成します。
 - の説明。Mount target `fsmt-id` for file system `fs-id`。 `fsmt-id` はマウントターゲット ID、 `fs-id` は `FileSystemId` です。
 - ネットワークインターフェイスの `requesterManaged` プロパティを `true` に設定し、`requesterId` 値を `requesterManaged` に設定します EFS。

各マウントターゲットには、対応するリクエストマネージド EC2 ネットワークインターフェイスが 1 つあります。ネットワークインターフェイスが作成されると、Amazon EFS はマウントターゲットの説明の `NetworkInterfaceId` フィールドをネットワークインターフェイス ID に設定し、IP アドレスを `IpAddress` に設定します。ネットワークインターフェイスの作成に失敗すると、`CreateMountTarget` オペレーション全体が失敗します。

 Note

CreateMountTarget コールは、ネットワークインターフェイスを作成した後で、マウントターゲットの状態が `creating` のままの場合にのみ返されます。[DescribeMountTargets](#) オペレーションをコールすることでマウントターゲットの作成ステータスをチェックできます。これにより、マウントターゲットの状態が返されます。

このオペレーションには、ファイルシステムに対する次のアクションに対するアクセス許可が必要です。

- `elasticfilesystem:CreateMountTarget`

このオペレーションには、次の Amazon EC2 アクションに対するアクセス許可も必要です。

- `ec2:DescribeSubnets`
- `ec2:DescribeNetworkInterfaces`
- `ec2:CreateNetworkInterface`

リクエストの構文

```
POST /2015-02-01/mount-targets HTTP/1.1
Content-type: application/json
```

```
{
  "FileSystemId": "string",
  "IpAddress": "string",
  "IpAddressType": "string",
  "Ipv6Address": "string",
  "SecurityGroups": [ "string" ],
  "SubnetId": "string"
}
```

URI リクエストパラメータ

リクエストでは URI パラメータを使用しません。

リクエストボディ

リクエストは以下の JSON 形式のデータを受け入れます。

[FileSystemId](#)

マウントターゲットを作成するファイルシステムの ID。

タイプ: 文字列

長さの制約: 最大長は 128 です。

パターン: `^(arn:aws[-a-z]*:elasticfilesystem:[0-9a-z-:]+:file-system/fs-[0-9a-f]{8,40}|fs-[0-9a-f]{8,40})$`

必須: はい

[IpAddress](#)

マウントターゲット `IpAddressType` の が `IPv4` (`IPV4_ONLY` または `DUAL_STACK`) の場合は、使用する `IPv4` アドレスを指定します。を指定しない場合 `IpAddress`、Amazon EFS は に指定されたサブネットから未使用の IP アドレスを選択します `SubnetId`。

タイプ: 文字列

長さの制限: 最小長は 7 です。最大長は 15 です。

パターン: `^[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}$`

必須: いいえ

[IpAddressType](#)

マウントターゲットの IP アドレスタイプ。指定できる値は、`IPV4_ONLY` (`IPv4` アドレスのみ)、`IPV6_ONLY` (`IPv6` アドレスのみ)、`DUAL_STACK` (デュアルスタック、`IPv4` アドレスと `IPv6` アドレスの両方) です。を指定しない場合 `IpAddressType`、`IPV4_ONLY` が使用されます。

Note

は、サブネットの IP タイプと一致する `IpAddressType` 必要があります。さらに、`IpAddressType` パラメータは、VPC 内のサブネットのデフォルト IP アドレスとして設定された値を上書きします。たとえば、`IpAddressType` が `IPV4_ONLY` で、`AssignIpv6AddressOnCreation` が の場合 `true`、`IPv4` がマウントターゲットに使用

されます。詳細については「[サブネットの IP アドレス指定属性を変更する](#)」を参照してください。

型: 文字列

有効な値: IPV4_ONLY | IPV6_ONLY | DUAL_STACK

必須: いいえ

Ipv6Address

マウントターゲット `IpAddressType` のが IPv6 (IPV6_ONLY または DUAL_STACK) の場合は、使用する IPv6 アドレスを指定します。を指定しない場合 `Ipv6Address`、Amazon EFS はに指定されたサブネットから未使用の IP アドレスを選択します `SubnetId`。

タイプ: 文字列

長さの制約: 最小長は 3 です。最大長は 39 です。

必須: いいえ

SecurityGroups

形式の VPC セキュリティグループ `Idsg-xxxxxxxx`。これらは、指定されたサブネットと同じ VPC 用である必要があります。セキュリティグループの最大数は、アカウントクォータによって異なります。詳細については、「[Amazon VPC ユーザーガイド](#)」の「[Amazon VPC クォータ](#)」を参照してください (セキュリティグループの表を参照)。セキュリティグループを指定しない場合、Amazon EFS はサブネットの VPC にデフォルトのセキュリティグループを使用します。

型: 文字列の配列

配列メンバー: 最大数は 100 項目です。

長さの制限: 最小長は 11 です。最大長は 43 です。

パターン: `^sg-[0-9a-f]{8,40}`

必須: いいえ

SubnetId

マウントターゲットを追加するサブネットの ID。1 ゾーンファイルシステムの場合は、ファイルシステムのアベイラビリティゾーンに関連付けられたサブネットを使用します。サブネットタイプは、と同じタイプである必要があります `IpAddressType`。

タイプ: 文字列

長さの制限: 最小長は 15 です。最大長は 47 です。

パターン: ^subnet-[0-9a-f]{8,40}\$

必須: はい

レスポンスの構文

```
HTTP/1.1 200
Content-type: application/json

{
  "AvailabilityZoneId": "string",
  "AvailabilityZoneName": "string",
  "FileSystemId": "string",
  "IpAddress": "string",
  "Ipv6Address": "string",
  "LifeCycleState": "string",
  "MountTargetId": "string",
  "NetworkInterfaceId": "string",
  "OwnerId": "string",
  "SubnetId": "string",
  "VpcId": "string"
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

[AvailabilityZoneId](#)

マウントターゲットが存在するアベイラビリティーゾーンの一意で一貫性のある識別子。たとえば、use1-az1は us-east-1 リージョンの AZ ID であり、すべての 同じ場所があります AWS アカウント。

タイプ: 文字列

AvailabilityZoneName

マウントターゲットが配置されているアベイラビリティーゾーンの名前。アベイラビリティーゾーンは、それぞれの名前に個別にマッピングされます AWS アカウント。たとえば、us-east-1a のアベイラビリティーゾーンが別の のアベイラビリティ us-east-1a ゾーンと同じ場所ではない AWS アカウント 場合があります AWS アカウント。

タイプ: 文字列

長さの制約: 最小長は 1 です。最大長 64

パターン: .+

FileSystemId

マウントターゲットが意図されているファイルシステムの ID。

タイプ: 文字列

長さの制約: 最大長は 128 です。

パターン: `^(arn:aws[-a-z]*:elasticfilesystem:[0-9a-z-:]+:file-system/fs-[0-9a-f]{8,40}|fs-[0-9a-f]{8,40})$`

IpAddress

マウントターゲットの IPv4 アドレス。

タイプ: 文字列

長さの制限: 最小長は 7 です。最大長は 15 です。

パターン: `^[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}$`

Ipv6Address

マウントターゲットの IPv6 アドレス。

タイプ: 文字列

長さの制約: 最小長は 3 です。最大長は 39 です。

LifeCycleState

マウントターゲットのライフサイクル状態。

型: 文字列

有効な値: creating | available | updating | deleting | deleted | error

MountTargetId

システム割り当てマウントターゲット ID。

タイプ: 文字列

長さの制限: 最小長は 13 です。最大長は 45 です。

パターン: ^fsmt-[0-9a-f]{8,40}\$

NetworkInterfaceId

マウントターゲットを作成したときに Amazon EFS が作成したネットワークインターフェイスの ID。

タイプ: 文字列

OwnerId

AWS アカウント リソースを所有する ID。

タイプ: 文字列

長さの制限: 最大長は 14 です。

パターン: ^(\d{12})|(\d{4}-\d{4}-\d{4})\$

SubnetId

マウントターゲットのサブネットの ID。

タイプ: 文字列

長さの制限: 最小長は 15 です。最大長は 47 です。

パターン: ^subnet-[0-9a-f]{8,40}\$

VpcId

マウントターゲットが設定されている 仮想プライベートクラウド (VPC) ID。

タイプ: 文字列

エラー

AvailabilityZonesMismatch

マウントターゲットに指定されたアベイラビリティゾーンがファイルシステムのアベイラビリティゾーンと異なる場合に返されます。詳細については、「[リージョナルと 1 ゾーンのスケーラビリティの冗長性](#)」を参照してください。

HTTP ステータスコード: 400

BadRequest

リクエストの形式が正しくない場合や、無効なパラメータ値や必須パラメータがないなどのエラーが含まれている場合に返されます。

HTTP ステータスコード: 400

FileSystemNotFound

指定されたFileSystemId値がリクエストの に存在しない場合に返されます AWS アカウント。

HTTP ステータスコード: 404

IncorrectFileSystemLifeCycleState

ファイルシステムのライフサイクル状態が「使用可能」でない場合に返されます。

HTTP ステータスコード: 409

InternalServerError

サーバー側でエラーが発生した場合に返されます。

HTTP ステータスコード: 500

IpAddressInUse

リクエストでサブネットで既に使用されている IP アドレス (IpAddress または Ipv6Address) が指定されている場合に返されます。

HTTP ステータスコード: 409

MountTargetConflict

ファイルシステムの既存のマウントターゲットに基づいて、マウントターゲットが指定された制限のいずれかに違反した場合に返されます。

HTTP ステータスコード: 409

NetworkInterfaceLimitExceeded

通話アカウントが、特定の AWS リージョンの Elastic Network Interface の制限に達しました。一部のネットワークインターフェースを削除するか、アカウントクォータの引き上げをリクエストしてください。詳細については、「[Amazon VPC クォータ](#)」の「Amazon VPC ユーザーガイド」を参照してください (ネットワークインターフェースの表にある「VPC ごとのネットワークインターフェース」エントリを参照)。

HTTP ステータスコード: 409

NoFreeAddressesInSubnet

リクエストに IpAddress または Ipv6Address が指定されておらず、指定されたサブネットに空き IP アドレスがありません。

HTTP ステータスコード: 409

SecurityGroupLimitExceeded

リクエストで SecurityGroups 指定された の数が、アカウントクォータに基づく制限より大きい場合に返されます。一部のセキュリティグループを削除するか、アカウントクォータの引き上げをリクエストします。詳細については、「[Amazon VPC ユーザーガイド](#)」の「[Amazon VPC クォータ](#)」を参照してください (セキュリティグループの表を参照)。

HTTP ステータスコード: 400

SecurityGroupNotFound

指定したセキュリティグループの 1 つがサブネットの仮想プライベートクラウド (VPC) に存在しない場合に返されます。

HTTP ステータスコード: 400

SubnetNotFound

リクエストで提供された ID SubnetId のサブネットが存在しない場合に返されます。

HTTP ステータスコード: 400

UnsupportedAvailabilityZone

リクエストされた Amazon EFS 機能が指定されたアベイラビリティゾーンで利用できない場合に返されます。

HTTP ステータスコード: 400

例

サブネットで使用可能な IPv4 アドレスにマウントターゲットを作成する

次のリクエストは、マウントターゲットのファイルシステムとサブネットのみを指定します。ターゲットは、指定されたサブネットで使用可能な IPv4 アドレスに作成され、デフォルトのセキュリティグループが VPC に関連付けられます。

リクエスト例

```
POST /2015-02-01/mount-targets HTTP/1.1
Host: elasticfilesystem.us-west-2.amazonaws.com
x-amz-date: 20140620T221118Z
Authorization: <...>
Content-Type: application/json
Content-Length: 160

{"SubnetId": "subnet-748c5d03", "FileSystemId": "fs-01234567"}
```

レスポンス例

```
HTTP/1.1 200 OK
x-amzn-RequestId: 01234567-89ab-cdef-0123-456789abcdef
Content-Type: application/json
Content-Length: 252

{
  "OwnerId": "231243201240",
  "MountTargetId": "fsmt-55a4413c",
  "FileSystemId": "fs-01234567",
  "SubnetId": "subnet-01234567",
  "LifecycleState": "available",
  "IpAddress": "172.31.22.183"
  "NetworkInterfaceId": "eni-1bcb7772"
```

```
"AvailabilityZoneId": "eus1-az2",  
"AvailabilityZoneName": "eu-south-1b",  
"VpcId": "vpc-08d45b31fa009a15e"  
}
```

特定の IPv4 アドレスにマウントターゲットを作成する

次のリクエストは、マウントターゲットに使用するファイルシステム、サブネット、セキュリティグループ、および IPv4 アドレスを指定します。

リクエスト例

```
POST /2015-02-01/mount-targets HTTP/1.1  
Host: elasticfilesystem.us-west-2.amazonaws.com  
x-amz-date: 20140620T221118Z  
Authorization: <...>  
Content-Type: application/json  
Content-Length: 160  
  
{  
  "FileSystemId": "fs-01234567",  
  "SubnetId": "subnet-01234567",  
  "IpAddress": "10.0.2.42",  
  "SecurityGroups": [  
    "sg-01234567"  
  ]  
}
```

レスポンス例

```
HTTP/1.1 200 OK  
x-amzn-RequestId: 01234567-89ab-cdef-0123-456789abcdef  
Content-Type: application/json  
Content-Length: 252  
  
{  
  "OwnerId": "251839141158",  
  "MountTargetId": "fsmt-9a13661e",  
  "FileSystemId": "fs-01234567",  
  "SubnetId": "subnet-fd04ff94",  
  "LifecycleState": "available",  
  "IpAddress": "10.0.2.42",  
  "NetworkInterfaceId": "eni-1bcb7772"
```

```
"AvailabilityZoneId": "eus1-az2",  
"AvailabilityZoneName": "eu-south-1b",  
"VpcId": "vpc-08d45b31fa009a15e"  
}
```

特定の IPv6 アドレスにマウントターゲットを作成する

次のリクエストは、マウントターゲットに使用するファイルシステム、サブネット、セキュリティグループ、および IPv6 アドレスを指定します。

リクエスト例

```
POST /2015-02-01/mount-targets HTTP/1.1  
Host: elasticfilesystem.us-west-2.amazonaws.com  
x-amz-date: 20140620T221118Z  
Authorization: <...>  
Content-Type: application/json  
Content-Length: 160  
  
{  
  "FileSystemId": "fs-01234567",  
  "SubnetId": "subnet-01234567",  
  "Ipv6Address": "2001:0db8:85a3:0000:0000:8a2e:0370:7334",  
  "IpAddressType": "IPV6_ONLY",  
  "SecurityGroups": [  
    "sg-01234567"  
  ]  
}
```

レスポンス例

```
HTTP/1.1 200 OK  
x-amzn-RequestId: 01234567-89ab-cdef-0123-456789abcdef  
Content-Type: application/json  
Content-Length: 252  
  
{  
  "OwnerId": "251839141158",  
  "MountTargetId": "fsmt-9a13661e",  
  "FileSystemId": "fs-01234567",  
  "SubnetId": "subnet-fd04ff94",  
  "LifeCycleState": "available",  
  "Ipv6Address": "2001:0db8:85a3:0000:0000:8a2e:0370:7334",  
}
```

```
"NetworkInterfaceId": "eni-1bcb7772"  
"AvailabilityZoneId": "eus1-az2",  
"AvailabilityZoneName": "eu-south-1b",  
"VpcId": "vpc-08d45b31fa009a15e"  
}
```

デュアルスタックサブネットで使用可能な IPv4 アドレスにマウントターゲットを作成する

次のリクエストは、マウントターゲットのファイルシステム、サブネット、およびアドレスタイプを指定します。ターゲットは、指定されたデュアルスタックサブネットで使用可能な IPv4 アドレスに作成され、デフォルトのセキュリティグループが VPC に関連付けられます。

リクエスト例

```
POST /2015-02-01/mount-targets HTTP/1.1  
Host: elasticfilesystem.us-west-2.amazonaws.com  
x-amz-date: 20140620T221118Z  
Authorization: <...>  
Content-Type: application/json  
Content-Length: 160  
  
{"SubnetId": "subnet-748c5d03", "FileSystemId": "fs-01234567"}
```

レスポンス例

```
HTTP/1.1 200 OK  
x-amzn-RequestId: 01234567-89ab-cdef-0123-456789abcdef  
Content-Type: application/json  
Content-Length: 252  
  
{  
  "OwnerId": "251839141158",  
  "MountTargetId": "fsmt-55a4413c",  
  "FileSystemId": "fs-01234567",  
  "SubnetId": "subnet-fd04ff94",  
  "LifecycleState": "available",  
  "IpAddress": "172.31.22.183",  
  "Ipv6Address": "2a05:d01a:419:8611:3e57:75ab:5719:b238",  
  "NetworkInterfaceId": "eni-01234567",  
  "AvailabilityZoneId": "eus1-az2",  
  "AvailabilityZoneName": "eu-south-1b",  
  "VpcId": "vpc-08d45b31fa009a15e"
```

```
}
```

デュアルスタックサブネット上の特定の Ipv4 アドレスと IPv6 アドレスにマウントターゲットを作成する

次のリクエストは、マウントターゲットのファイルシステム、サブネット、セキュリティグループ、IPv4 アドレス、IPv6 アドレスを指定します。ターゲットは、デュアルスタックサブネットの指定された IPv4 アドレスと IPv6 アドレスに作成されます。

リクエスト例

```
POST /2015-02-01/mount-targets HTTP/1.1
Host: elasticfilesystem.us-west-2.amazonaws.com
x-amz-date: 20140620T221118Z
Authorization: <...>
Content-Type: application/json
Content-Length: 160

{
  "FileSystemId": "fs-01234567",
  "SubnetId": "subnet-01234567",
  "IpAddressType": "DUAL_STACK",
  "IpAddress": "10.0.1.25",
  "Ipv6Address": "2001:0db8:85a3:0000:0000:8a2e:0370:7334",
  "SecurityGroups": [
    "sg-01234567"
  ]
}
```

レスポンス例

```
HTTP/1.1 200 OK
x-amzn-RequestId: 01234567-89ab-cdef-0123-456789abcdef
Content-Type: application/json
Content-Length: 252

{
  "OwnerId": "231243201240",
  "MountTargetId": "fsmt-55a4413c",
  "FileSystemId": "fs-01234567",
  "SubnetId": "subnet-01234567",
  "LifecycleState": "available",
```

```
"IpAddress": "10.0.1.25",  
"Ipv6Address": "2001:0db8:85a3:0000:0000:8a2e:0370:7334",  
"NetworkInterfaceId": "eni-01234567",  
"AvailabilityZoneId": "eus1-az2",  
"AvailabilityZoneName": "eu-south-1b",  
"VpcId": "vpc-08d45b31fa009a15e"  
}
```

以下の資料も参照してください。

言語固有の AWS SDKs のいずれかでこの API を使用方法の詳細については、以下を参照してください。

- [AWS コマンドラインインターフェイス](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go v2](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript V3](#)
- [AWS SDK for Kotlin](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

CreateReplicationConfiguration

新規または既存の EFS ファイルシステムにレプリケーション設定を作成します。詳細については、「Amazon EFS ユーザーガイド」の「[Amazon EFS レプリケーション](#)」を参照してください。レプリケーション設定では、以下を指定します。

- ・ ソースファイルシステム – レプリケートする EFS ファイルシステム。
- ・ 送信先ファイルシステム – 送信元ファイルシステムがレプリケートされる送信先ファイルシステム。1 つのレプリケーション設定にはデスティネーションファイルシステムは 1 つしか存在できません。

Note

ファイルシステムは、1 つのレプリケーション設定にのみ含めることができます。

レプリケーション設定の送信先パラメータは、新しいファイルシステムにレプリケートするか、既存のファイルシステムにレプリケートするか、レプリケートするかによって異なります AWS アカウント。詳細については「[DestinationToCreate](#)」を参照してください。

このオペレーションには、elasticfilesystem:CreateReplicationConfiguration アクションに対するアクセス許可が必要です。さらに、ファイルシステムのレプリケーション方法に応じて、他のアクセス許可が必要です。詳細については、「Amazon EFS ユーザーガイド」の「[レプリケーションに必要なアクセス許可](#)」を参照してください。

リクエストの構文

```
POST /2015-02-01/file-systems/SourceFileSystemId/replication-configuration HTTP/1.1
Content-type: application/json
```

```
{
  "Destinations": [
    {
      "AvailabilityZoneName": "string",
      "FileSystemId": "string",
      "KmsKeyId": "string",
      "Region": "string",
      "RoleArn": "string"
    }
  ]
}
```

```
]
}
```

URI リクエストパラメータ

リクエストでは、次の URI パラメータを使用します。

[SourceFileSystemId](#)

レプリケートする Amazon EFS ファイルシステムを指定します。このファイルシステムは、別のレプリケーション設定で既にソースまたはデスティネーションファイルシステムになっていることはできません。

長さの制約: 最大長は 128 です。

パターン: `^(arn:aws[-a-z]*:elasticfilesystem:[0-9a-z-:]+:file-system/fs-[0-9a-f]{8,40}|fs-[0-9a-f]{8,40})$`

必須: はい

リクエストボディ

リクエストは以下の JSON 形式のデータを受け入れます。

[Destinations](#)

送信先設定オブジェクトの配列です。サポートされるデスティネーション設定オブジェクトは 1 つだけです。

型: [DestinationToCreate](#) オブジェクトの配列

必須: はい

レスポンスの構文

```
HTTP/1.1 200
Content-type: application/json

{
  "CreationTime": number,
  "Destinations": [
```

```
{
  "FileSystemId": "string",
  "LastReplicatedTimestamp": number,
  "OwnerId": "string",
  "Region": "string",
  "RoleArn": "string",
  "Status": "string",
  "StatusMessage": "string"
},
"OriginalSourceFileSystemArn": "string",
"SourceFileSystemArn": "string",
"SourceFileSystemId": "string",
"SourceFileSystemOwnerId": "string",
"SourceFileSystemRegion": "string"
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

CreationTime

レプリケーション設定の作成日を記述します。

タイプ: タイムスタンプ

Destinations

送信先オブジェクトの配列。サポートされているデスティネーションオブジェクトは 1 つだけです。

型: [Destination](#) オブジェクトの配列

OriginalSourceFileSystemArn

レプリケーション設定の元のソース EFS ファイルシステムの Amazon リソースネーム (ARN)。

タイプ: 文字列

SourceFileSystemArn

レプリケーション設定内の現在のソースファイルシステムの Amazon リソースネーム (ARN)。

タイプ: 文字列

SourceFileSystemId

レプリケートされたソースの Amazon EFS ファイルシステムの ID。

タイプ: 文字列

長さの制約: 最大長は 128 です。

パターン: `^(arn:aws[-a-z]*:elasticfilesystem:[0-9a-z-:]+:file-system/fs-[0-9a-f]{8,40}|fs-[0-9a-f]{8,40})$`

SourceFileSystemOwnerId

AWS アカウント ソースファイルシステムが存在する の ID。

タイプ: 文字列

長さの制限: 最大長は 14 です。

パターン: `^(\\d{12})|(\\d{4}-\\d{4}-\\d{4})$`

SourceFileSystemRegion

ソース EFS ファイルシステム AWS リージョン がある 。

タイプ: 文字列

長さの制約: 最小長は 1 です。最大長 64

パターン: `^[a-z]{2}-((iso[a-z]{0,1}-)|(gov-)){0,1}[a-z]+-{0,1}[0-9]{0,1}$`

エラー

BadRequest

リクエストの形式が正しくない場合や、無効なパラメータ値や必須パラメータの欠落などのエラーが含まれている場合に返されます。

HTTP ステータスコード: 400

ConflictException

レプリケーションのソースファイルシステムが暗号化されている一方で、デスティネーションファイルシステムが暗号化されていない場合に返されます。

HTTP ステータスコード: 409

FileSystemLimitExceeded

AWS アカウント がアカウントごとに許可されるファイルシステムの最大数を既に作成している場合に返されます。

HTTP ステータスコード: 403

FileSystemNotFound

指定されたFileSystemId値がリクエストの に存在しない場合に返されます AWS アカウント。

HTTP ステータスコード: 404

IncorrectFileSystemLifeCycleState

ファイルシステムのライフサイクル状態が「使用可能」でない場合に返されます。

HTTP ステータスコード: 409

InsufficientThroughputCapacity

追加のスループットをプロビジョニングするのに十分な容量がない場合に返されます。この値は、プロビジョニングされたスループットモードでファイルシステムを作成しようとしたとき、既存のファイルシステムのプロビジョニングされたスループットを上げようとしたとき、または既存のファイルシステムをバーストからプロビジョニングされたスループットモードに変更しようとしたときに返されることがあります。後でもう一度お試しください。

HTTP ステータスコード: 503

InternalServerError

サーバー側でエラーが発生した場合に返されます。

HTTP ステータスコード: 500

ReplicationNotFound

指定したファイルシステムにレプリケーション設定がない場合に返されます。

HTTP ステータスコード: 404

ThroughputLimitExceeded

スループット制限の 1024 MiB/s に達したため、スループットモードまたはプロビジョニングされたスループットの量を変更できない場合に返されます。

HTTP ステータスコード: 400

UnsupportedAvailabilityZone

リクエストされた Amazon EFS 機能が指定されたアベイラビリティーゾーンで利用できない場合に返されます。

HTTP ステータスコード: 400

ValidationException

リクエストが行われた AWS リージョン で AWS Backup サービスが利用できない場合に返されます。

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 AWS SDKs のいずれかでこの API を使用方法の詳細については、以下を参照してください。

- [AWS コマンドラインインターフェイス](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go v2](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript V3](#)
- [AWS SDK for Kotlin](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

CreateTags

Note

非推奨 - CreateTags は非推奨であり、維持されません。EFS リソースのタグを作成するには、[TagResource](#) の API アクションを使用します。

ファイルシステムに関連付けられたタグを作成または上書きします。各タグはキーバリューのペアです。リクエストで指定されたタグキーがファイルシステム上にすでに存在する場合、このオペレーションは、リクエストで指定された値でその値を上書きします。Name のタグをファイルシステムに追加すると、Amazon EFS は [DescribeFileSystems](#) のオペレーションに対する応答でタグを返します。

このオペレーションには elasticfilesystem:CreateTags アクションに対するアクセス許可が必要です。

リクエストの構文

```
POST /2015-02-01/create-tags/FileSystemId HTTP/1.1
```

```
Content-type: application/json
```

```
{
  "Tags": [
    {
      "Key": "string",
      "Value": "string"
    }
  ]
}
```

URI リクエストパラメータ

リクエストでは、次の URI パラメータを使用します。

[FileSystemId](#)

変更するタグが付けられたファイルシステムの ID (文字列)。このオペレーションでは、ファイルシステムは変更されず、タグのみが変更されます。

長さの制約: 最大長は 128 です。

パターン: `^(arn:aws[-a-z]*:elasticfilesystem:[0-9a-z-:]+:file-system/fs-[0-9a-f]{8,40}|fs-[0-9a-f]{8,40})$`

必須: はい

リクエストボディ

リクエストは以下の JSON 形式のデータを受け入れます。

Tags

追加する Tag オブジェクトの配列。各 Tag オブジェクトタグはキーバリューのペアです。

型: [Tag](#) オブジェクトの配列

必須: はい

レスポンスの構文

```
HTTP/1.1 204
```

レスポンス要素

アクションが成功した場合、サービスは空の HTTP 本文を持つ HTTP 204 レスポンスを返します。

エラー

BadRequest

リクエストの形式が正しくない場合や、無効なパラメータ値や必須パラメータの欠落などのエラーが含まれている場合に返されます。

HTTP ステータスコード: 400

FileSystemNotFound

指定された `FileSystemId` 値がリクエストの に存在しない場合に返されます AWS アカウント。

HTTP ステータスコード: 404

InternalServerError

サーバー側でエラーが発生した場合に返されます。

HTTP ステータスコード: 500

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用方法の詳細については、以下を参照してください。

- [AWS コマンドラインインターフェイス](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go v2](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript V3](#)
- [AWS SDK for Kotlin](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

DeleteAccessPoint

指定されたアクセスポイントを削除します。削除が完了すると、新しいクライアントはアクセスポイントに接続できなくなります。削除時にアクセスポイントに接続されたクライアントは、接続を終了するまで引き続き機能します。

このオペレーションには、`elasticfilesystem:DeleteAccessPoint` アクションに対する許可が必要です。

リクエストの構文

```
DELETE /2015-02-01/access-points/AccessPointId HTTP/1.1
```

URI リクエストパラメータ

リクエストでは、次の URI パラメータを使用します。

AccessPointId

削除するアクセスポイントの ID。

長さの制約: 最大長は 128 です。

パターン: `^(arn:aws[-a-z]*:elasticfilesystem:[0-9a-z-:]+:access-point/fsap-[0-9a-f]{8,40}|fsap-[0-9a-f]{8,40})$`

必須: はい

リクエストボディ

リクエストにリクエスト本文がありません。

レスポンスの構文

```
HTTP/1.1 204
```

レスポンス要素

アクションが成功した場合、サービスは空の HTTP 本文を持つ HTTP 204 レスポンスを返します。

エラー

AccessPointNotFound

指定されたAccessPointId値がリクエストの に存在しない場合に返されます AWS アカウント。

HTTP ステータスコード: 404

BadRequest

リクエストの形式が正しくない場合や、無効なパラメータ値や必須パラメータがないなどのエラーが含まれている場合に返されます。

HTTP ステータスコード: 400

InternalServerError

サーバー側でエラーが発生した場合に返されます。

HTTP ステータスコード: 500

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用方法の詳細については、以下を参照してください。

- [AWS コマンドラインインターフェイス](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go v2](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript V3](#)
- [AWS SDK for Kotlin](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

DeleteFileSystem

ファイルシステムを削除し、その内容へのアクセスを永久に切断します。戻ると、ファイルシステムは存在しなくなり、削除されたファイルシステムのコンテンツにアクセスできなくなります。

EFS ファイルシステムを削除する前に、ファイルシステムに接続されているマウントターゲットを手動で削除する必要があります。このステップは、AWS コンソールを使用してファイルシステムを削除するときに自動的に実行されます。

Note

EFS レプリケーション設定の一部であるファイルシステムは削除できません。レプリケーション設定を最初に削除する必要があります。

使用中のファイルシステムを削除することはできません。つまり、ファイルシステムにマウントターゲットがある場合は、まずマウントターゲットを削除する必要があります。詳細については、「[DescribeMountTargets](#)」および「[DeleteMountTarget](#)」を参照してください。

Note

DeleteFileSystem コールは、ファイルシステムの状態がまだ deleting である間に戻ります。ファイルシステムの削除状況をチェックするには、アカウント内のファイルシステムのリストを返す [DescribeFileSystems](#) オペレーションをコールします。削除されたファイルシステムのファイルシステム ID または作成 トークンを渡した場合、[DescribeFileSystems](#) は 404 FileSystemNotFound のエラーを返します。

このオペレーションには、elasticfilesystem:DeleteFileSystem アクションに対する許可が必要です。

リクエストの構文

```
DELETE /2015-02-01/file-systems/FileSystemId HTTP/1.1
```

URI リクエストパラメータ

リクエストでは、次の URI パラメータを使用します。

FileSystemId

削除するファイルシステムの ID。

長さの制約: 最大長は 128 です。

パターン: `^(arn:aws[-a-z]*:elasticfilesystem:[0-9a-z-:]+:file-system/fs-[0-9a-f]{8,40}|fs-[0-9a-f]{8,40})$`

必須: はい

リクエストボディ

リクエストにリクエスト本文がありません。

レスポンスの構文

```
HTTP/1.1 204
```

レスポンス要素

アクションが成功した場合、サービスは空の HTTP 本文を持つ HTTP 204 レスポンスを返します。

エラー

BadRequest

リクエストの形式が正しくない場合や、無効なパラメータ値や必須パラメータの欠落などのエラーが含まれている場合に返されます。

HTTP ステータスコード: 400

FileSystemInUse

ファイルシステムに、マウントターゲットがある場合に返されます。

HTTP ステータスコード: 409

FileSystemNotFound

指定されたFileSystemId値がリクエストの に存在しない場合に返されます AWS アカウント。

HTTP ステータスコード: 404

InternalServerError

サーバー側でエラーが発生した場合に返されます。

HTTP ステータスコード: 500

例

ファイルシステムの削除

次の例では、file-systems エンドポイント (elasticfilesystem.us-west-2.amazonaws.com/2015-02-01/file-systems/fs-01234567) に DELETE リクエストを送信して、ID が fs-01234567 のファイルシステムを削除します。

リクエスト例

```
DELETE /2015-02-01/file-systems/fs-01234567 HTTP/1.1
Host: elasticfilesystem.us-west-2.amazonaws.com
x-amz-date: 20140622T233021Z
Authorization: <...>
```

レスポンス例

```
HTTP/1.1 204 No Content
x-amzn-RequestId: a2d125b3-7ebd-4d6a-ab3d-5548630bff33
Content-Length: 0
```

以下の資料も参照してください。

言語固有の AWS SDKs のいずれかでこの API を使用方法の詳細については、以下を参照してください。

- [AWS コマンドラインインターフェイス](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go v2](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript V3](#)

- [AWS SDK for Kotlin](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

DeleteFileSystemPolicy

指定されたファイルシステムの FileSystemPolicy を削除します。既存のポリシーが削除されると、デフォルトの FileSystemPolicy が有効になります。デフォルトのファイル システムポリシーの詳細については、「[EFS でリソースベースのポリシーを使用する](#)」を参照してください。

このオペレーションには、elasticfilesystem:DeleteFileSystemPolicy アクションに対する許可が必要です。

リクエストの構文

```
DELETE /2015-02-01/file-systems/FileSystemId/policy HTTP/1.1
```

URI リクエストパラメータ

リクエストでは、次の URI パラメータを使用します。

FileSystemId

FileSystemPolicy を削除する EFS ファイルシステムを指定します。

長さの制約: 最大長は 128 です。

パターン: `^(arn:aws[-a-z]*:elasticfilesystem:[0-9a-z-:]+:file-system/fs-[0-9a-f]{8,40}|fs-[0-9a-f]{8,40})$`

必須: はい

リクエストボディ

リクエストにリクエスト本文がありません。

レスポンスの構文

```
HTTP/1.1 200
```

レスポンス要素

アクションが成功した場合、サービスは空の HTTP 本文を持つ HTTP 200 レスポンスを返します。

エラー

BadRequest

リクエストの形式が正しくない場合や、無効なパラメータ値や必須パラメータの欠落などのエラーが含まれている場合に返されます。

HTTP ステータスコード: 400

FileSystemNotFound

指定されたFileSystemId値がリクエストの に存在しない場合に返されます AWS アカウント。

HTTP ステータスコード: 404

IncorrectFileSystemLifecycleState

ファイルシステムのライフサイクル状態が「使用可能」でない場合に返されます。

HTTP ステータスコード: 409

InternalServerError

サーバー側でエラーが発生した場合に返されます。

HTTP ステータスコード: 500

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用方法の詳細については、以下を参照してください。

- [AWS コマンドラインインターフェイス](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go v2](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript V3](#)
- [AWS SDK for Kotlin](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)

- [AWS SDK for Ruby V3](#)

DeleteMountTarget

指定されたマウントターゲットを削除します。

このオペレーションにより、削除されるマウントターゲットを使用しているファイルシステムのマウントが強制的に解除されるため、そのマウントを使用しているインスタンスやアプリケーションが停止する可能性があります。アプリケーションが突然切断されないように、可能であれば、マウントターゲットのマウントをすべてアンマウントすることを確認してください。このオペレーションにより、関連付けられたネットワークインターフェイスも削除されます。コミットされていない書き込みは失われる可能性があります。このオペレーションを使用してマウント・ターゲットを破っても、ファイル・システム自体が破損することはありません。作成したファイルシステムは残ります。別のマウントターゲットを使用して VPC に EC2 インスタンスをマウントできます。

このオペレーションには、ファイルシステムに対する次のアクションに対するアクセス許可が必要です。

- `elasticfilesystem:DeleteMountTarget`

Note

マウントターゲットの状態がまだ `deleting` にある間に `DeleteMountTarget` コールが戻ります。マウントターゲットの削除を確認するには、指定されたファイルシステムのマウントターゲットの説明のリストを返す [DescribeMountTargets](#) オペレーションをコールします。

オペレーションには、マウントターゲットのネットワーク インターフェイスで次の Amazon EC2 アクションに対するアクセス許可も必要です。

- `ec2:DeleteNetworkInterface`

リクエストの構文

```
DELETE /2015-02-01/mount-targets/MountTargetId HTTP/1.1
```

URI リクエストパラメータ

リクエストでは、次の URI パラメータを使用します。

MountTargetId

削除するマウントターゲットの ID (文字列)。

長さの制限: 最小長は 13 です。最大長は 45 です。

パターン: `^fsmt-[0-9a-f]{8,40}$`

必須: はい

リクエストボディ

リクエストにリクエスト本文がありません。

レスポンスの構文

```
HTTP/1.1 204
```

レスポンス要素

アクションが成功した場合、サービスは空の HTTP 本文を持つ HTTP 204 レスポンスを返します。

エラー

BadRequest

リクエストの形式が正しくない場合や、無効なパラメータ値や必須パラメータの欠落などのエラーが含まれている場合に返されます。

HTTP ステータスコード: 400

DependencyTimeout

リクエストを履行しようとしてサービスがタイムアウトしたため、クライアントはコールを再試行する必要があります。

HTTP ステータスコード: 504

InternalServerError

サーバー側でエラーが発生した場合に返されます。

HTTP ステータスコード: 500

MountTargetNotFound

指定された ID のマウントターゲットが発信者の AWS アカウントで見つからない場合に返されます。

HTTP ステータスコード: 404

例

ファイルシステムのマウントターゲットの削除

次の例では、DELETE リクエストを送信して、特定のマウントターゲットを削除します。

リクエスト例

```
DELETE /2015-02-01/mount-targets/fsmt-9a13661e HTTP/1.1
Host: elasticfilesystem.us-west-2.amazonaws.com
x-amz-date: 20140622T232908Z
Authorization: <...>
```

レスポンス例

```
HTTP/1.1 204 No Content
x-amzn-RequestId: 01234567-89ab-cdef-0123-456789abcdef
```

以下の資料も参照してください。

言語固有の AWS SDKs のいずれかでこの API を使用方法の詳細については、以下を参照してください。

- [AWS コマンドラインインターフェイス](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go v2](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript V3](#)
- [AWS SDK for Kotlin](#)

- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

DeleteReplicationConfiguration

レプリケーション設定を削除する。レプリケーション設定を削除すると、レプリケーションプロセスは終了します。レプリケーション設定が削除されると、デスティネーションファイルシステムが Writeable となり、そのレプリケーション上書き保護が再び有効になります。詳細については、「[レプリケーション設定の削除](#)」を参照してください。

このオペレーションには、elasticfilesystem:DeleteReplicationConfiguration アクションに対する許可が必要です。

リクエストの構文

```
DELETE /2015-02-01/file-systems/SourceFileSystemId/replication-configuration?
deletionMode=DeletionMode HTTP/1.1
```

URI リクエストパラメータ

リクエストでは、次の URI パラメータを使用します。

[DeletionMode](#)

デフォルトでは AWS リージョン、Amazon EFS AWS アカウント はレプリケート元アカウントとレプリケート先アカウントまたはリージョン (ALL_CONFIGURATIONS) の両方からレプリケーション設定を削除します。Amazon EFS がレプリケーション設定を両側から削除できない設定またはアクセス許可の問題がある場合は、LOCAL_CONFIGURATION_ONLY モードを使用して、ローカル側 (削除が実行されるアカウントまたはリージョン) からのみレプリケーション設定を削除できます。

Note

Amazon EFS がレプリケート元アカウントとレプリケート先アカウントまたはリージョンの両方でレプリケーション設定を削除できない場合のみ、LOCAL_CONFIGURATION_ONLY モードを使用します。ローカル設定を削除すると、他のアカウントまたはリージョンの設定は回復できなくなります。さらに、同じアカウント、同じリージョンのレプリケーションにこのモードを使用しないでください。そうすると、BadRequest 例外エラーが発生します。

有効な値 : ALL_CONFIGURATIONS | LOCAL_CONFIGURATION_ONLY

SourceFileSystemId

レプリケーション設定内のソースファイルシステムの ID。

長さの制約: 最大長は 128 です。

パターン: `^(arn:aws[-a-z]*:elasticfilesystem:[0-9a-z-:]+:file-system/fs-[0-9a-f]{8,40}|fs-[0-9a-f]{8,40})$`

必須: はい

リクエストボディ

リクエストにリクエスト本文がありません。

レスポンスの構文

```
HTTP/1.1 204
```

レスポンス要素

アクションが成功した場合、サービスは空の HTTP 本文を持つ HTTP 204 レスポンスを返します。

エラー

BadRequest

リクエストの形式が正しくない場合や、無効なパラメータ値や必須パラメータの欠落などのエラーが含まれている場合に返されます。

HTTP ステータスコード: 400

FileSystemNotFound

指定された `FileSystemId` 値がリクエストの に存在しない場合に返されます AWS アカウント。

HTTP ステータスコード: 404

InternalServerError

サーバー側でエラーが発生した場合に返されます。

HTTP ステータスコード: 500

ReplicationNotFound

指定したファイルシステムにレプリケーション設定がない場合に返されます。

HTTP ステータスコード: 404

以下の資料も参照してください。

言語固有の AWS SDKs のいずれかでこの API を使用方法の詳細については、以下を参照してください。

- [AWS コマンドラインインターフェイス](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go v2](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript V3](#)
- [AWS SDK for Kotlin](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

DeleteTags

Note

非推奨 - DeleteTags は非推奨であり、維持されません。EFS リソースからタグを削除するには、[UntagResource](#) API アクションを使用します。

ファイルシステムから指定されたタグを削除します。DeleteTags のリクエストに存在しないタグキーが含まれていても、Amazon EFS はそれを無視し、エラーを起こしません。タグおよび関連する制限の詳細については、「AWS Billing and Cost Management ユーザーガイド」の「[タグの制限](#)」を参照してください。

このオペレーションには、elasticfilesystem:DeleteTags アクションに対する許可が必要です。

リクエストの構文

```
POST /2015-02-01/delete-tags/FileSystemId HTTP/1.1
Content-type: application/json

{
  "TagKeys": [ "string" ]
}
```

URI リクエストパラメータ

リクエストでは、次の URI パラメータを使用します。

[FileSystemId](#)

削除するタグが付けられたファイルシステムの ID (文字列)。

長さの制約: 最大長は 128 です。

パターン: `^(arn:aws[-a-z]*:elasticfilesystem:[0-9a-z-:]+:file-system/fs-[0-9a-f]{8,40}|fs-[0-9a-f]{8,40})$`

必須: はい

リクエストボディ

リクエストは以下の JSON 形式のデータを受け入れます。

[TagKeys](#)

削除するタグキーのリスト。

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 1、最大長は 128 です。

パターン: `^(?![aA]{1}[wW]{1}[sS]{1}:)([\\p{L}\\p{Z}\\p{N}_.:/=+\\-@]+)$`

必須: はい

レスポンスの構文

```
HTTP/1.1 204
```

レスポンス要素

アクションが成功した場合、サービスは空の HTTP 本文を持つ HTTP 204 レスポンスを返します。

エラー

BadRequest

リクエストの形式が正しくない場合や、無効なパラメータ値や必須パラメータの欠落などのエラーが含まれている場合に返されます。

HTTP ステータスコード: 400

FileSystemNotFound

指定されたFileSystemId値がリクエストの に存在しない場合に返されます AWS アカウント。

HTTP ステータスコード: 404

InternalServerError

サーバー側でエラーが発生した場合に返されます。

HTTP ステータスコード: 500

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用方法の詳細については、以下を参照してください。

- [AWS コマンドラインインターフェイス](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go v2](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript V3](#)
- [AWS SDK for Kotlin](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

DescribeAccessPoints

AccessPointId が提供されている場合、特定の Amazon EFS アクセスポイントの説明を返します。EFS FileSystemId を指定すると、そのファイルシステムのすべてのアクセスポイントの説明が返されます。リクエストには、AccessPointId または FileSystemId のどちらかを指定できますが、両方を指定することはできません。

このオペレーションには、elasticfilesystem:DescribeAccessPoints アクションに対する許可が必要です。

リクエストの構文

```
GET /2015-02-01/access-points?  
AccessPointId=AccessPointId&FileSystemId=FileSystemId&MaxResults=MaxResults&NextToken=NextToken  
HTTP/1.1
```

URI リクエストパラメータ

リクエストでは、次の URI パラメータを使用します。

AccessPointId

(オプション) 応答で記述するEFSアクセスポイントを指定します。FileSystemId とは相互に排他的です。

長さの制約: 最大長は 128 です。

パターン: `^(arn:aws[-a-z]*:elasticfilesystem:[0-9a-z-:]+:access-point/fsap-[0-9a-f]{8,40}|fsap-[0-9a-f]{8,40})$`

FileSystemId

(オプション)FileSystemIdの場合、EFS はそのファイルシステムのすべてのアクセスポイントを返します。AccessPointId とは相互に排他的です。

長さの制約: 最大長は 128 です。

パターン: `^(arn:aws[-a-z]*:elasticfilesystem:[0-9a-z-:]+:file-system/fs-[0-9a-f]{8,40}|fs-[0-9a-f]{8,40})$`

MaxResults

(オプション) ファイルシステムのすべてのアクセスポイントを取得する際に、オプションで MaxItems パラメータを指定して、応答で返されるオブジェクトの数を制限することができます。デフォルト値は 100 です。

有効な範囲: 最小値 は 1 です。

NextToken

NextToken は応答がページ割りされている場合に存在します。後続のリクエストで NextMarker を使うと、次のページのアクセスポイントの説明を取得することができます。

長さの制限: 最小長 1、最大長は 128 です。

Pattern: .+

リクエストボディ

リクエストにリクエスト本文がありません。

レスポンスの構文

```
HTTP/1.1 200
Content-type: application/json

{
  "AccessPoints": [
    {
      "AccessPointArn": "string",
      "AccessPointId": "string",
      "ClientToken": "string",
      "FileSystemId": "string",
      "LifecycleState": "string",
      "Name": "string",
      "OwnerId": "string",
      "PosixUser": {
        "Gid": number,
        "SecondaryGids": [ number ],
        "Uid": number
      },
      "RootDirectory": {
```

```
    "CreationInfo": {
      "OwnerGid": number,
      "OwnerUid": number,
      "Permissions": "string"
    },
    "Path": "string"
  },
  "Tags": [
    {
      "Key": "string",
      "Value": "string"
    }
  ]
},
"NextToken": "string"
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

[AccessPoints](#)

アクセスポイントの説明の配列。

型: [AccessPointDescription](#) オブジェクトの配列

[NextToken](#)

応答で返されるよりも多くのアクセスポイントがある場合に表示されます。後続のリクエストで NextMarker を使用して、追加の説明をフェッチできます。

タイプ: 文字列

長さの制限: 最小長は 1 です。最大長は 128 です。

パターン: .+

エラー

AccessPointNotFound

指定されたAccessPointId値がリクエストの に存在しない場合に返されます AWS アカウント。

HTTP ステータスコード: 404

BadRequest

リクエストの形式が正しくない場合や、無効なパラメータ値や必須パラメータがないなどのエラーが含まれている場合に返されます。

HTTP ステータスコード: 400

FileSystemNotFound

指定されたFileSystemId値がリクエストの に存在しない場合に返されます AWS アカウント。

HTTP ステータスコード: 404

InternalServerError

サーバー側でエラーが発生した場合に返されます。

HTTP ステータスコード: 500

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用方法の詳細については、以下を参照してください。

- [AWS コマンドラインインターフェイス](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go v2](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript V3](#)
- [AWS SDK for Kotlin](#)
- [AWS SDK for PHP V3](#)

- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

DescribeAccountPreferences

現在の、リクエストを行うユーザー AWS アカウントに関連付けられた のアカウント設定を返します AWS リージョン。

リクエストの構文

```
GET /2015-02-01/account-preferences HTTP/1.1
Content-type: application/json
```

```
{
  "MaxResults": number,
  "NextToken": "string"
}
```

URI リクエストパラメータ

リクエストでは URI パラメータを使用しません。

リクエストボディ

リクエストは以下の JSON 形式のデータを受け入れます。

MaxResults

(オプション) アカウントの設定を取得する際に、オプションで MaxItems パラメータを指定して、レスポンスで返されるオブジェクトの数を制限することができます。デフォルト値は 100 です。

型: 整数

有効な範囲: 最小値 は 1 です。

必須: いいえ

NextToken

(オプション) レスポンスのペイロードがページ割りされている場合、次のリクエストで NextToken を使用すると、AWS アカウント のプリファレンスの次のページをフェッチできます。

タイプ: 文字列

長さの制限: 最小長は 1 です。最大長は 128 です。

パターン: .+

必須: いいえ

レスポンスの構文

```
HTTP/1.1 200
Content-type: application/json

{
  "NextToken": "string",
  "ResourceIdPreference": {
    "ResourceIdType": "string",
    "Resources": [ "string" ]
  }
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

[NextToken](#)

応答で返されるよりも多くのレコードがある場合に表示されます。後続のリクエストで NextToken を使うと、追加の説明をフェッチすることができます。

型: 文字列

長さの制限: 最小長は 1 です。最大長は 128 です。

パターン: .+

[ResourceIdPreference](#)

現在の、リクエストを行うユーザー AWS アカウントに関連付けられたのリソース ID 設定について説明します AWS リージョン。

型: [ResourceIdPreference](#) オブジェクト

エラー

InternalServerError

サーバー側でエラーが発生した場合に返されます。

HTTP ステータスコード: 500

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用方法の詳細については、以下を参照してください。

- [AWS コマンドラインインターフェイス](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go v2](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript V3](#)
- [AWS SDK for Kotlin](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

DescribeBackupPolicy

指定された EFS ファイルシステムのバックアップポリシーを返します。

リクエストの構文

```
GET /2015-02-01/file-systems/FileSystemId/backup-policy HTTP/1.1
```

URI リクエストパラメータ

リクエストでは、次の URI パラメータを使用します。

[FileSystemId](#)

BackupPolicy を取得する EFS ファイルシステムを指定します。

長さの制約: 最大長は 128 です。

パターン: `^(arn:aws[-a-z]*:elasticfilesystem:[0-9a-z-:]+:file-system/fs-[0-9a-f]{8,40}|fs-[0-9a-f]{8,40})$`

必須: はい

リクエストボディ

リクエストにリクエスト本文がありません。

レスポンスの構文

```
HTTP/1.1 200
Content-type: application/json
```

```
{
  "BackupPolicy": {
    "Status": "string"
  }
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

[BackupPolicy](#)

ファイルシステムのバックアップポリシーについて説明し、自動バックアップをオンにするかオフにするかを示します。

型: [BackupPolicy](#) オブジェクト

エラー

BadRequest

リクエストの形式が正しくない場合や、無効なパラメータ値や必須パラメータの欠落などのエラーが含まれている場合に返されます。

HTTP ステータスコード: 400

FileSystemNotFound

指定されたFileSystemId値がリクエストの に存在しない場合に返されます AWS アカウント。

HTTP ステータスコード: 404

InternalServerError

サーバー側でエラーが発生した場合に返されます。

HTTP ステータスコード: 500

PolicyNotFound

no backup が 1 ゾーン EFS ファイルシステムに指定されている場合に返されます。

HTTP ステータスコード: 404

ValidationException

リクエストが行われた AWS リージョン で AWS Backup サービスが利用できない場合に返されます。

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 AWS SDKs のいずれかでこの API を使用方法の詳細については、以下を参照してください。

- [AWS コマンドラインインターフェイス](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go v2](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript V3](#)
- [AWS SDK for Kotlin](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

DescribeFileSystemPolicy

指定されたEFSファイルシステムの FileSystemPolicy を返します。

このオペレーションには、elasticfilesystem:DescribeFileSystemPolicy アクションに対する許可が必要です。

リクエストの構文

```
GET /2015-02-01/file-systems/FileSystemId/policy HTTP/1.1
```

URI リクエストパラメータ

リクエストでは、次の URI パラメータを使用します。

[FileSystemId](#)

FileSystemPolicy を取得するEFSファイルシステムを指定します。

長さの制約: 最大長は 128 です。

パターン: `^(arn:aws[-a-z]*:elasticfilesystem:[0-9a-z-:]+:file-system/fs-[0-9a-f]{8,40}|fs-[0-9a-f]{8,40})$`

必須: はい

リクエストボディ

リクエストにリクエスト本文がありません。

レスポンスの構文

```
HTTP/1.1 200
Content-type: application/json

{
  "FileSystemId": "string",
  "Policy": "string"
}
```


レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

[FileSystemId](#)

FileSystemPolicy が適用される EFS ファイルシステムを指定します。

タイプ: 文字列

長さの制約: 最大長は 128 です。

パターン: `^(arn:aws[-a-z]*:elasticfilesystem:[0-9a-z-:]+:file-system/fs-[0-9a-f]{8,40}|fs-[0-9a-f]{8,40})$`

[Policy](#)

EFS ファイルシステムの JSON 形式の FileSystemPolicy です。

タイプ: 文字列

長さの制約: 最小長は 1 です。最大長は 20000 です。

パターン: `[\s\S]+`

エラー

BadRequest

リクエストの形式が正しくない場合や、無効なパラメータ値や必須パラメータの欠落などのエラーが含まれている場合に返されます。

HTTP ステータスコード: 400

FileSystemNotFound

指定されたFileSystemId値がリクエストの に存在しない場合に返されます AWS アカウント。

HTTP ステータスコード: 404

InternalServerError

サーバー側でエラーが発生した場合に返されます。

HTTP ステータスコード: 500

PolicyNotFound

no backup が 1 ゾーン EFS ファイルシステムに指定されている場合に返されます。

HTTP ステータスコード: 404

例

例

この例では、DescribeFileSystemPolicy の 1 つの使用方法について説明します。

リクエスト例

```
GET /2015-02-01/file-systems/fs-01234567/policy HTTP/1.1
```

レスポンス例

```
{
  "FileSystemId": "fs-01234567",
  "Policy": "{
    \"Version\": \"2012-10-17\",
    \"Id\": \"efs-policy-wizard-cdef0123-aaaa-6666-5555-444455556666\",
    \"Statement\": [
      {
        \"Sid\": \"efs-statement-abcdef01-1111-bbbb-2222-111122224444\",
        \"Effect\" : \"Deny\",
        \"Principal\": {
          \"AWS\": \"*\"
        },
        \"Action\": \"*\",
        \"Resource\": \"arn:aws:elasticfilesystem:us-east-2:111122223333:file-
system/fs-01234567\",
        \"Condition\": {
          \"Bool\": {
            \"aws:SecureTransport\": \"false\"
          }
        }
      },
      {
        \"Sid\": \"efs-statement-01234567-aaaa-3333-4444-111122223333\",
```

```
        "Effect": "Allow",
        "Principal": {
            "AWS": "*"
        },
        "Action": [
            "elasticfilesystem:ClientMount",
            "elasticfilesystem:ClientWrite"
        ],
        "Resource" : "arn:aws:elasticfilesystem:us-east-2:111122223333:file-
system/fs-01234567"
    }
}
```

以下の資料も参照してください。

言語固有の AWS SDKs のいずれかでこの API を使用方法の詳細については、以下を参照してください。

- [AWS コマンドラインインターフェイス](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go v2](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript V3](#)
- [AWS SDK for Kotlin](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

DescribeFileSystems

ファイルシステム CreationToken またはファイルシステム FileSystemId のいずれかが提供された場合、特定の Amazon EFS ファイルシステムの説明を返します。それ以外の場合、呼び出しているエンドポイント AWS リージョン の AWS アカウント で、呼び出し元の が所有するすべてのファイルシステムの説明を返します。

すべてのファイルシステムの説明を取得する際に、オプションで MaxItems パラメータを指定して、応答に含まれる説明の数を制限することができます。この数値は自動的に 100 に設定されます。より多くのファイルシステムの説明が残っている場合、Amazon EFS は応答で不透明なトークンである NextMarker を返します。この場合、Marker リクエストのパラメーターを NextMarker の値に設定して、次のリクエストを送信する必要があります。

ファイルシステムの説明の一覧を取得するために、このオペレーションは反復プロセスで使用されます。ここでは、DescribeFileSystems が Marker なしで最初にコールされ、Marker パラメータが前の応答の NextMarker の値に設定された状態で、応答に NextMarker がなくなるまで、オペレーションはコールし続けます。

1回の DescribeFileSystems コールの応答で返されるファイルシステムの順序と、複数回コールの反復の応答全体で返されるファイルシステムの順序は、指定されていません。

このオペレーションには、elasticfilesystem:DescribeFileSystems アクションに対する許可が必要です。

リクエストの構文

```
GET /2015-02-01/file-systems?  
CreationToken=CreationToken&FileSystemId=FileSystemId&Marker=Marker&MaxItems=MaxItems  
HTTP/1.1
```

URI リクエストパラメータ

リクエストでは、次の URI パラメータを使用します。

CreationToken

(オプション) リストを、この作成トークン(文字列)を持つファイルシステムに制限します。作成トークンは、Amazon EFS ファイルシステムを作成するときに指定します。

長さの制限：最小長 1、最大長 64

パターン: .+

FileSystemId

(オプション) 説明を取得するファイルシステムの ID (文字列)。

長さの制約: 最大長は 128 です。

パターン: `^(arn:aws[-a-z]*:elasticfilesystem:[0-9a-z-:]+:file-system/fs-[0-9a-f]{8,40}|fs-[0-9a-f]{8,40})$`

Marker

(オプション) 前の DescribeFileSystems オペレーション (文字列) から返された不透明なページ割りトークン。存在する場合は、返すコールが中断された場所からリストを続行するように指定します。

長さの制限: 最小長 1、最大長は 128 です。

パターン: .+

MaxItems

(オプション) レスポンスで返すファイルシステムの最大数を指定します (整数)。この数値は自動的に 100 に設定されます。100 を超えるファイルシステムがある場合、応答はページあたり 100 でページ割りされます。

有効な範囲: 最小値 は 1 です。

リクエスト本文

リクエストにリクエスト本文がありません。

レスポンスの構文

```
HTTP/1.1 200
Content-type: application/json

{
  "FileSystems": [
    {
      "AvailabilityZoneId": "string",
      "AvailabilityZoneName": "string",
```

```

    "CreationTime": number,
    "CreationToken": "string",
    "Encrypted": boolean,
    "FileSystemArn": "string",
    "FileSystemId": "string",
    "FileSystemProtection": {
        "ReplicationOverwriteProtection": "string"
    },
    "KmsKeyId": "string",
    "LifeCycleState": "string",
    "Name": "string",
    "NumberOfMountTargets": number,
    "OwnerId": "string",
    "PerformanceMode": "string",
    "ProvisionedThroughputInMibps": number,
    "SizeInBytes": {
        "Timestamp": number,
        "Value": number,
        "ValueInArchive": number,
        "ValueInIA": number,
        "ValueInStandard": number
    },
    "Tags": [
        {
            "Key": "string",
            "Value": "string"
        }
    ],
    "ThroughputMode": "string"
}
],
"Marker": "string",
"NextMarker": "string"
}

```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

FileSystems

ファイルシステムの説明の配列。

型: [FileSystemDescription](#) オブジェクトの配列

[Marker](#)

リクエストで発信者から提供された場合に存在します (文字列)。

タイプ: 文字列

長さの制限: 最小長は 1 です。最大長は 128 です。

パターン: .+

[NextMarker](#)

応答で返されたファイルシステムよりも多くのファイルシステムがある場合に表示されます (文字列)。後続のリクエストで NextMarker を使うと、説明をフェッチすることができます。

タイプ: 文字列

長さの制限: 最小長は 1 です。最大長は 128 です。

パターン: .+

エラー

BadRequest

リクエストの形式が正しくない場合や、無効なパラメータ値や必須パラメータの欠落などのエラーが含まれている場合に返されます。

HTTP ステータスコード: 400

FileSystemNotFound

指定されたFileSystemId値がリクエストの に存在しない場合に返されます AWS アカウント。

HTTP ステータスコード: 404

InternalServerError

サーバー側でエラーが発生した場合に返されます。

HTTP ステータスコード: 500

例

10 個のファイルシステムのリストの取得

次の例では、GET リクエストを file-systems エンドポイント (elasticfilesystem.us-west-2.amazonaws.com/2015-02-01/file-systems) に送信します。このリクエストでは、MaxItems クエリ パラメータを指定して、ファイルシステムの説明の数を 10 に制限します。

リクエスト例

```
GET /2015-02-01/file-systems?MaxItems=10 HTTP/1.1
Host: elasticfilesystem.us-west-2.amazonaws.com
x-amz-date: 20140622T191208Z
Authorization: <...>
```

レスポンス例

```
HTTP/1.1 200 OK
x-amzn-RequestId: 01234567-89ab-cdef-0123-456789abcdef
Content-Type: application/json
Content-Length: 499
{
  "FileSystems": [
    {
      "OwnerId": "251839141158",
      "CreationToken": "MyFileSystem1",
      "FileSystemId": "fs-01234567",
      "PerformanceMode": "generalPurpose",
      "CreationTime": "1403301078",
      "LifecycleState": "created",
      "Name": "my first file system",
      "NumberOfMountTargets": 1,
      "SizeInBytes": {
        "Timestamp": 1403301078,
        "Value": 29313618372,
        "ValueInArchive": 201156,
        "ValueInIA": 675432,
        "ValueInStandard": 29312741784
      }
    }
  ]
}
```


以下の資料も参照してください。

言語固有の AWS SDKs のいずれかでこの API を使用方法の詳細については、以下を参照してください。

- [AWS コマンドラインインターフェイス](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go v2](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript V3](#)
- [AWS SDK for Kotlin](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

DescribeLifecycleConfiguration

指定された EFS ファイルシステムの現在の LifecycleConfiguration オブジェクトを返します。ライフサイクル管理では、LifecycleConfiguration オブジェクトを使用して、ストレージクラス間でファイルを移動するタイミングを特定します。LifecycleConfiguration オブジェクトがないファイルシステムの場合、コールの応答には空の配列が返されます。

このオペレーションには、elasticfilesystem:DescribeLifecycleConfiguration オペレーションに対する許可が必要です。

リクエストの構文

```
GET /2015-02-01/file-systems/FileSystemId/lifecycle-configuration HTTP/1.1
```

URI リクエストパラメータ

リクエストでは、次の URI パラメータを使用します。

FileSystemId

取得する LifecycleConfiguration オブジェクトのファイルシステムの ID (文字列)。

長さの制約: 最大長は 128 です。

パターン: `^(arn:aws[-a-z]*:elasticfilesystem:[0-9a-z-:]+:file-system/fs-[0-9a-f]{8,40}|fs-[0-9a-f]{8,40})$`

必須: はい

リクエストボディ

リクエストにリクエスト本文がありません。

レスポンスの構文

```
HTTP/1.1 200
Content-type: application/json
```

```
{
  "LifecyclePolicies": [
```

```
{
  "TransitionToArchive": "string",
  "TransitionToIA": "string",
  "TransitionToPrimaryStorageClass": "string"
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

[LifecyclePolicies](#)

ライフサイクル管理ポリシーの配列。EFS では、ファイルシステムごとに最大1つのポリシーがサポートされています。

型: [LifecyclePolicy](#) オブジェクトの配列

配列メンバー: 最大数は 3 項目です。

エラー

BadRequest

リクエストの形式が正しくない場合や、無効なパラメータ値や必須パラメータの欠落などのエラーが含まれている場合に返されます。

HTTP ステータスコード: 400

FileSystemNotFound

指定されたFileSystemId値がリクエストの に存在しない場合に返されます AWS アカウント。

HTTP ステータスコード: 404

InternalServerError

サーバー側でエラーが発生した場合に返されます。

HTTP ステータスコード: 500

例

ファイルシステムのライフサイクル設定を取得する

次のリクエストは、指定されたファイルシステムの LifecycleConfiguration オブジェクトを取得します。

リクエスト例

```
GET /2015-02-01/file-systems/fs-01234567/lifecycle-configuration HTTP/1.1
Host: elasticfilesystem.us-west-2.amazonaws.com
x-amz-date: 20181120T221118Z
Authorization: <...>
```

レスポンス例

```
HTTP/1.1 200 OK
    x-amzn-RequestId: 01234567-89ab-cdef-0123-456789abcdef
    Content-Type: application/json
    Content-Length: 86
{
  "LifecyclePolicies": [
    {
      "TransitionToArchive": "AFTER_270_DAYS"
    },
    {
      "TransitionToIA": "AFTER_14_DAYS"
    },
    {
      "TransitionToPrimaryStorageClass": "AFTER_1_ACCESS"
    }
  ]
}
```

以下の資料も参照してください。

言語固有の AWS SDKs のいずれかでこの API を使用方法の詳細については、以下を参照してください。

- [AWS コマンドラインインターフェイス](#)
- [AWS SDK for .NET](#)

- [AWS SDK for C++](#)
- [AWS SDK for Go v2](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript V3](#)
- [AWS SDK for Kotlin](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

DescribeMountTargets

ファイルシステムのすべての現在のマウントターゲット、または特定のマウントターゲットの説明を返します。現在のマウント・ターゲットをすべてリクエストする場合、応答で返されるマウント・ターゲットの順序は指定されていません。

このオペレーションには、FileSystemId で指定したファイルシステム ID、または MountTargetId で指定したマウントターゲットのファイルシステムのいずれかで、elasticfilesystem:DescribeMountTargets のアクションに対する許可が必要です。

対応する CLI コマンドは describe-mount-targets です。

リクエストの構文

```
GET /2015-02-01/mount-targets?  
AccessPointId=AccessPointId&FileSystemId=FileSystemId&Marker=Marker&MaxItems=MaxItems&MountTarg  
HTTP/1.1
```

URI リクエストパラメータ

リクエストでは、次の URI パラメータを使用します。

[AccessPointId](#)

(オプション) リスト表示するマウントターゲットがあるアクセスポイントの ID。FileSystemId または MountTargetId がリクエストに含まれていない場合は、必ずリクエストに含める必要があります。アクセスポイント ID または ARN を入力として受け入れます。

長さの制約: 最大長は 128 です。

パターン: `^(arn:aws[-a-z]*:elasticfilesystem:[0-9a-z-:]+:access-point/fsap-[0-9a-f]{8,40}|fsap-[0-9a-f]{8,40})$`

[FileSystemId](#)

(オプション) マウントターゲットをリストするファイルシステムの ID (文字列)。AccessPointId または MountTargetId がリクエストに含まれていない場合は、必ずリクエストに含める必要があります。ファイルシステム ID または ARN を入力として受け入れます。

長さの制約: 最大長は 128 です。

パターン: `^(arn:aws[-a-z]*:elasticfilesystem:[0-9a-z-:]+:file-system/fs-[0-9a-f]{8,40}|fs-[0-9a-f]{8,40})$`

Marker

(オプション) 前の DescribeMountTargets オペレーション (文字列) から返された不透明なページ割りトークン。存在する場合は、前回のコールが中断された場所からリストを続行するように指定します。

長さの制限: 最小長 1、最大長は 128 です。

パターン: `.+`

MaxItems

(オプション) 応答で返すマウントターゲットの最大数。現在、この数値は自動的に 10 に設定され、その他の値は無視されます。100 を超えるマウントターゲットがある場合、レスポンスはページあたり 100 でページ割りされます。

有効な範囲: 最小値 は 1 です。

MountTargetId

(オプション) 説明するマウントターゲットの ID (文字列)。FileSystemId が含まれていない場合は、リクエストに含める必要があります。マウントターゲット ID または ARN を入力として受け入れます。

長さの制限: 最小長は 13 です。最大長は 45 です。

Pattern: `^fsmt-[0-9a-f]{8,40}$`

リクエストボディ

リクエストにリクエスト本文がありません。

レスポンスの構文

```
HTTP/1.1 200
Content-type: application/json

{
  "Marker": "string",
  "MountTargets": [
```

```
{
  "AvailabilityZoneId": "string",
  "AvailabilityZoneName": "string",
  "FileSystemId": "string",
  "IpAddress": "string",
  "Ipv6Address": "string",
  "LifeCycleState": "string",
  "MountTargetId": "string",
  "NetworkInterfaceId": "string",
  "OwnerId": "string",
  "SubnetId": "string",
  "VpcId": "string"
},
"NextMarker": "string"
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

[Marker](#)

リクエストに Marker が含まれていた場合、応答はこのフィールドにその値を返します。

タイプ: 文字列

長さの制限: 最小長は 1 です。最大長は 128 です。

パターン: .+

[MountTargets](#)

ファイルシステムのマウントターゲットを MountTargetDescription オブジェクトの配列として返します。

型: [MountTargetDescription](#) オブジェクトの配列

[NextMarker](#)

値が存在する場合、返されるマウントターゲットがさらに多くなります。後続のリクエストでは、この値を使ってリクエストに Marker を指定することで、次のマウントターゲットの設定を取得することができます。

タイプ: 文字列

長さの制限: 最小長は 1 です。最大長は 128 です。

パターン: .+

エラー

AccessPointNotFound

指定されたAccessPointId値がリクエストの に存在しない場合に返されます AWS アカウント。

HTTP ステータスコード: 404

BadRequest

リクエストの形式が正しくない場合や、無効なパラメータ値や必須パラメータがないなどのエラーが含まれている場合に返されます。

HTTP ステータスコード: 400

FileSystemNotFound

指定されたFileSystemId値がリクエストの に存在しない場合に返されます AWS アカウント。

HTTP ステータスコード: 404

InternalServerError

サーバー側でエラーが発生した場合に返されます。

HTTP ステータスコード: 500

MountTargetNotFound

指定された ID のマウントターゲットが発信者の AWS アカウントで見つからない場合に返されます。

HTTP ステータスコード: 404

例

ファイル・システム用に作成されたマウント・ターゲットの説明を取得する

次のリクエストは、指定されたファイルシステムに対して作成されたマウントターゲットの説明を取得します。

リクエスト例

```
GET /2015-02-01/mount-targets?FileSystemId=fs-01234567 HTTP/1.1
Host: elasticfilesystem.us-west-2.amazonaws.com
x-amz-date: 20140622T191252Z
Authorization: <...>
```

レスポンス例

```
HTTP/1.1 200 OK
x-amzn-RequestId: 01234567-89ab-cdef-0123-456789abcdef
Content-Type: application/json
Content-Length: 357

{
  "MountTargets": [
    {
      "OwnerId": "251839141158",
      "MountTargetId": "fsmt-01234567",
      "FileSystemId": "fs-01234567",
      "SubnetId": "subnet-01234567",
      "LifeCycleState": "added",
      "IpAddress": "10.0.2.42",
      "NetworkInterfaceId": "eni-1bcb7772"
    }
  ]
}
```

以下の資料も参照してください。

言語固有の AWS SDKs のいずれかでこの API を使用方法の詳細については、以下を参照してください。

- [AWS コマンドラインインターフェイス](#)
- [AWS SDK for .NET](#)

- [AWS SDK for C++](#)
- [AWS SDK for Go v2](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript V3](#)
- [AWS SDK for Kotlin](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

DescribeMountTargetSecurityGroups

マウントターゲットに対して現在有効なセキュリティグループを返します。このオペレーションには、マウントターゲットのネットワークインターフェースが作成されており、マウントターゲットのライフサイクル状態が `deleted` でないことが必要です。

このオペレーションには、次のアクションに対する許可が必要です。

- マウントターゲットのファイルシステム上での `elasticfilesystem:DescribeMountTargetSecurityGroups` アクション。
- マウントターゲットのネットワークインターフェース上での `ec2:DescribeNetworkInterfaceAttribute` アクション。

リクエストの構文

```
GET /2015-02-01/mount-targets/MountTargetId/security-groups HTTP/1.1
```

URI リクエストパラメータ

リクエストでは、次の URI パラメータを使用します。

MountTargetId

取得するセキュリティグループがあるマウントターゲットの ID。

長さの制限: 最小長は 13 です。最大長は 45 です。

パターン: `^fsmt-[0-9a-f]{8,40}$`

必須: はい

リクエストボディ

リクエストにリクエスト本文がありません。

レスポンスの構文

```
HTTP/1.1 200
Content-type: application/json
```

```
{  
  "SecurityGroups": [ "string" ]  
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

[SecurityGroups](#)

セキュリティグループの配列。

型: 文字列の配列

配列メンバー: 最大数は 100 項目です。

長さの制限: 最小長は 11 です。最大長は 43 です。

パターン: `^sg-[0-9a-f]{8,40}`

エラー

BadRequest

リクエストの形式が正しくない場合や、無効なパラメータ値や必須パラメータの欠落などのエラーが含まれている場合に返されます。

HTTP ステータスコード: 400

IncorrectMountTargetState

マウントターゲットがオペレーションで正しい状態でない場合に返されます。

HTTP ステータスコード: 409

InternalServerError

サーバー側でエラーが発生した場合に返されます。

HTTP ステータスコード: 500

MountTargetNotFound

指定された ID のマウントターゲットが発信者の AWS アカウントで見つからない場合に返されます。

HTTP ステータスコード: 404

例

ファイルシステムに有効なセキュリティグループを取得

次の例では、マウントターゲットに関連付けられたネットワーク インターフェイスに有効なセキュリティグループを取得します。

リクエスト例

```
GET /2015-02-01/mount-targets/fsmt-9a13661e/security-groups HTTP/1.1
Host: elasticfilesystem.us-west-2.amazonaws.com
x-amz-date: 20140620T223513Z
Authorization: <...>
```

レスポンス例

```
HTTP/1.1 200 OK
x-amzn-RequestId: 01234567-89ab-cdef-0123-456789abcdef
Content-Length: 57

{
  "SecurityGroups" : [
    "sg-188d9f74"
  ]
}
```

以下の資料も参照してください。

言語固有の AWS SDKs のいずれかでこの API を使用方法の詳細については、以下を参照してください。

- [AWS コマンドラインインターフェイス](#)
- [AWS SDK for .NET](#)

- [AWS SDK for C++](#)
- [AWS SDK for Go v2](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript V3](#)
- [AWS SDK for Kotlin](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

DescribeReplicationConfigurations

特定のファイルシステムに関するレプリケーション設定を取得します。ファイルシステムが指定されていない場合、AWS アカウント のすべてのレプリケーション設定 AWS リージョン が取得されます。

リクエストの構文

```
GET /2015-02-01/file-systems/replication-configurations?  
FileSystemId=FileSystemId&MaxResults=MaxResults&NextToken=NextToken HTTP/1.1
```

URI リクエストパラメータ

リクエストでは、次の URI パラメータを使用します。

[FileSystemId](#)

特定のファイルシステムのレプリケーション設定は、ファイルシステム ID を指定することで取得できます。クロスアカウント、クロスリージョンレプリケーションの場合、アカウントは独自のリージョン内のファイルシステムのレプリケーション設定のみを記述できます。

長さの制約: 最大長は 128 です。

パターン: `^(arn:aws[-a-z]*:elasticfilesystem:[0-9a-z-:]+:file-system/fs-[0-9a-f]{8,40}|fs-[0-9a-f]{8,40})$`

[MaxResults](#)

(オプション) レスポンスで返されるオブジェクトの数を制限するには、MaxItems パラメータを指定します。デフォルト値は 100 です。

有効な範囲: 最小値 は 1 です。

[NextToken](#)

NextToken は応答がページ割りされている場合に存在します。後続のリクエストで NextToken を使うと、次のページの出力をフェッチすることができます。

長さの制限: 最小長 1、最大長は 128 です。

Pattern: `.+`

リクエストボディ

リクエストにリクエスト本文がありません。

レスポンスの構文

```
HTTP/1.1 200
Content-type: application/json

{
  "NextToken": "string",
  "Replications": [
    {
      "CreationTime": number,
      "Destinations": [
        {
          "FileSystemId": "string",
          "LastReplicatedTimestamp": number,
          "OwnerId": "string",
          "Region": "string",
          "RoleArn": "string",
          "Status": "string",
          "StatusMessage": "string"
        }
      ],
      "OriginalSourceFileSystemArn": "string",
      "SourceFileSystemArn": "string",
      "SourceFileSystemId": "string",
      "SourceFileSystemOwnerId": "string",
      "SourceFileSystemRegion": "string"
    }
  ]
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

[NextToken](#)

後続のリクエストで前のレスポンスから NextToken を使うと、追加の説明をフェッチすることができます。

タイプ: 文字列

長さの制限: 最小長は 1 です。最大長は 128 です。

パターン: .+

[Replications](#)

返されるレプリケーション設定のコレクション。

型: [ReplicationConfigurationDescription](#) オブジェクトの配列

エラー

BadRequest

リクエストの形式が正しくない場合や、無効なパラメータ値や必須パラメータの欠落などのエラーが含まれている場合に返されます。

HTTP ステータスコード: 400

FileSystemNotFound

指定されたFileSystemId値がリクエストの に存在しない場合に返されます AWS アカウント。

HTTP ステータスコード: 404

InternalServerError

サーバー側でエラーが発生した場合に返されます。

HTTP ステータスコード: 500

ReplicationNotFound

指定したファイルシステムにレプリケーション設定がない場合に返されます。

HTTP ステータスコード: 404

ValidationException

リクエストが行われた AWS リージョン で AWS Backup サービスが利用できない場合に返されます。

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 AWS SDKs のいずれかでこの API を使用方法の詳細については、以下を参照してください。

- [AWS コマンドラインインターフェイス](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go v2](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript V3](#)
- [AWS SDK for Kotlin](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

DescribeTags

Note

非推奨 - DescribeTags アクションは非推奨であり、維持されません。EFS リソースに関連付けられているタグを表示するには、ListTagsForResource API アクションを使用します。

ファイルシステムに関連付けられたタグを返します。1回の DescribeTags コールの応答で返されるタグの順序と、複数回のコールの反復 (ページ割りを使用する場合) の応答全体で返されるタグの順序は、指定されていません。

このオペレーションには、elasticfilesystem:DescribeTags アクションに対する許可が必要です。

リクエストの構文

```
GET /2015-02-01/tags/FileSystemId?Marker=Marker&MaxItems=MaxItems HTTP/1.1
```

URI リクエストパラメータ

リクエストでは、次の URI パラメータを使用します。

FileSystemId

取得するタグセットが付けられたファイルシステムの ID。

長さの制約: 最大長は 128 です。

パターン: `^(arn:aws[-a-z]*:elasticfilesystem:[0-9a-z-:]+:file-system/fs-[0-9a-f]{8,40}|fs-[0-9a-f]{8,40})$`

必須: はい

Marker

(オプション) 前の DescribeTags オペレーション (文字列) から返された不透明ページ割りトークン。存在する場合は、前のコールが中断された場所からリストを続行するように指定します。

長さの制限：最小長 1、最大長は 128 です。

パターン: .+

[MaxItems](#)

(オプション) レスポンスで返すファイル システムタグの最大数。現在、この数値は自動的に 100 に設定され、その他の値は無視されます。100 を超えるタグがある場合、レスポンスはページあたり 100 でページ割りされます。

有効範囲: 最小値 1

リクエスト本文

リクエストにリクエスト本文がありません。

レスポンスの構文

```
HTTP/1.1 200
Content-type: application/json
```

```
{
  "Marker": "string",
  "NextMarker": "string",
  "Tags": [
    {
      "Key": "string",
      "Value": "string"
    }
  ]
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

[Marker](#)

リクエストにMarkerが含まれている場合、応答はこのフィールドにその値を返します。

タイプ: 文字列

長さの制限: 最小長は 1 です。最大長は 128 です。

パターン: .+

[NextMarker](#)

値が存在する場合、返されるタグがさらに多くなります。次のリクエストでは、Marker パラメータの値として NextMarker の値を指定することで、次のタグ設定を取得することができます。

タイプ: 文字列

長さの制限: 最小長は 1 です。最大長は 128 です。

パターン: .+

[Tags](#)

ファイルシステムに関連付けられたタグを Tag オブジェクトの配列として返します。

型: [Tag](#) オブジェクトの配列

エラー

BadRequest

リクエストの形式が正しくない場合や、無効なパラメータ値や必須パラメータの欠落などのエラーが含まれている場合に返されます。

HTTP ステータスコード: 400

FileSystemNotFound

指定されたFileSystemId値がリクエストの に存在しない場合に返されます AWS アカウント。

HTTP ステータスコード: 404

InternalServerError

サーバー側でエラーが発生した場合に返されます。

HTTP ステータスコード: 500

例

ファイルシステムに関連付けられているタグの取得

次のリクエストは、指定されたファイルシステムに関連付けられたタグ (キーバリューのペア) を取得します。

リクエスト例

```
GET /2015-02-01/tags/fs-01234567/ HTTP/1.1
Host: elasticfilesystem.us-west-2.amazonaws.com
x-amz-date: 20140620T215404Z
Authorization: <...>
```

レスポンス例

```
HTTP/1.1 200 OK
x-amzn-RequestId: 01234567-89ab-cdef-0123-456789abcdef
Content-Type: application/json
Content-Length: 288

{
  "Tags": [
    {
      "Key": "Name",
      "Value": "my first file system"
    },
    {
      "Key": "Fleet",
      "Value": "Development"
    },
    {
      "Key": "Developer",
      "Value": "Alice"
    }
  ]
}
```

以下の資料も参照してください。

言語固有の AWS SDKs のいずれかでこの API を使用方法の詳細については、以下を参照してください。

- [AWS コマンドラインインターフェイス](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go v2](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript V3](#)
- [AWS SDK for Kotlin](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

ListTagsForResource

最上位の EFS リソースのすべてのタグをリスト表示します。タグを取得するリソースの ID を入力する必要があります。

このオペレーションには、elasticfilesystem:DescribeAccessPoints アクションに対する許可が必要です。

リクエストの構文

```
GET /2015-02-01/resource-tags/ResourceId?MaxResults=MaxResults&NextToken=NextToken
HTTP/1.1
```

URI リクエストパラメータ

リクエストでは、次の URI パラメータを使用します。

MaxResults

(オプション) レスポンスで返すタグオブジェクトの最大数を指定します。デフォルト値は 100 です。

有効範囲: 最小値 1

NextToken

(オプション) 応答のペイロードがページ割りされている場合、次のリクエストで NextToken を使用して、アクセスポイントの説明の次のページをフェッチできます。

長さの制限: 最小長 1、最大長は 128 です。

パターン: .+

ResourceId

タグを取得する EFS リソースを指定します。この API エンドポイントを使用して、EFS ファイルシステムおよびアクセスポイントのタグを取得できます。

長さの制約: 最大長は 128 です。

パターン: `^(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: はい

リクエストボディ

リクエストにリクエスト本文がありません。

レスポンスの構文

```
HTTP/1.1 200
Content-type: application/json
```

```
{
  "NextToken": "string",
  "Tags": [
    {
      "Key": "string",
      "Value": "string"
    }
  ]
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

[NextToken](#)

NextToken はレスポンスペイロードがページ割りされている場合に存在します。後続のリクエストで NextToken を使うと、次のページのアクセスポイントの説明をフェッチすることができます。

タイプ: 文字列

長さの制限: 最小長は 1 です。最大長は 128 です。

パターン: .+

[Tags](#)

指定された EFS リソースのタグの配列。

型: [Tag](#) オブジェクトの配列

エラー

AccessPointNotFound

指定されたAccessPointId値がリクエストの に存在しない場合に返されます AWS アカウント。

HTTP ステータスコード: 404

BadRequest

リクエストの形式が正しくない場合や、無効なパラメータ値や必須パラメータがないなどのエラーが含まれている場合に返されます。

HTTP ステータスコード: 400

FileSystemNotFound

指定されたFileSystemId値がリクエストの に存在しない場合に返されます AWS アカウント。

HTTP ステータスコード: 404

InternalServerError

サーバー側でエラーが発生した場合に返されます。

HTTP ステータスコード: 500

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用方法の詳細については、以下を参照してください。

- [AWS コマンドラインインターフェイス](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go v2](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript V3](#)

- [AWS SDK for Kotlin](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

ModifyMountTargetSecurityGroups

マウントターゲットに対して有効なセキュリティグループの設定を変更します。

マウントターゲットを作成すると、Amazon EFS によって新しいネットワーク インターフェイスも作成されます。詳細については、「[CreateMountTarget](#)」を参照してください。このオペレーションでは、マウントターゲットに関連するネットワークインターフェイスに有効なセキュリティグループを、リクエストで指定された SecurityGroups に置き換えます。このオペレーションには、マウントターゲットのネットワークインターフェイスが作成されており、マウントターゲットのライフサイクル状態が `deleted` でないことが必要です。

このオペレーションには、次のアクションに対する許可が必要です。

- マウントターゲットのファイルシステム上での `elasticfilesystem:ModifyMountTargetSecurityGroups` アクション。
- マウントターゲットのネットワークインターフェイス上での `ec2:ModifyNetworkInterfaceAttribute` アクション。

リクエストの構文

```
PUT /2015-02-01/mount-targets/MountTargetId/security-groups HTTP/1.1
Content-type: application/json

{
  "SecurityGroups": [ "string" ]
}
```

URI リクエストパラメータ

リクエストでは、次の URI パラメータを使用します。

[MountTargetId](#)

変更するセキュリティグループがあるマウントターゲットの ID。

長さの制限: 最小長は 13 です。最大長は 45 です。

パターン: `^fsmt-[0-9a-f]{8,40}$`

必須: はい

リクエストボディ

リクエストは以下の JSON 形式のデータを受け入れます。

[SecurityGroups](#)

VPC セキュリティグループ IDs の配列。

型: 文字列の配列

配列メンバー: 最大数は 100 項目です。

長さの制限: 最小長は 11 です。最大長は 43 です。

パターン: `^sg-[0-9a-f]{8,40}`

必須: いいえ

レスポンスの構文

```
HTTP/1.1 204
```

レスポンス要素

アクションが成功した場合、サービスは空の HTTP 本文を持つ HTTP 204 レスポンスを返します。

エラー

BadRequest

リクエストの形式が正しくない場合や、無効なパラメータ値や必須パラメータの欠落などのエラーが含まれている場合に返されます。

HTTP ステータスコード: 400

IncorrectMountTargetState

マウントターゲットがオペレーションで正しい状態でない場合に返されます。

HTTP ステータスコード: 409

InternalServerError

サーバー側でエラーが発生した場合に返されます。

HTTP ステータスコード: 500

MountTargetNotFound

指定された ID のマウントターゲットが発信者の AWS アカウントで見つからない場合に返されます。

HTTP ステータスコード: 404

SecurityGroupLimitExceeded

リクエストでSecurityGroups指定された の数が、アカウントクォータに基づく制限より大きい場合に返されます。一部のセキュリティグループを削除するか、アカウントクォータの引き上げをリクエストします。詳細については、[「Amazon VPC ユーザーガイド」の「Amazon VPC クォータ」](#)を参照してください (セキュリティグループの表を参照)。

HTTP ステータスコード: 400

SecurityGroupNotFound

指定したセキュリティグループの 1 つがサブネットの仮想プライベートクラウド (VPC) に存在しない場合に返されます。

HTTP ステータスコード: 400

例

マウントターゲットのセキュリティグループを置き換える

次の例では、マウントターゲットに関連付けられたネットワーク インターフェイスに有効なセキュリティグループを置き換えます。

リクエスト例

```
PUT /2015-02-01/mount-targets/fsmt-9a13661e/security-groups HTTP/1.1
Host: elasticfilesystem.us-west-2.amazonaws.com
x-amz-date: 20140620T223446Z
Authorization: <...>
Content-Type: application/json
Content-Length: 57

{
  "SecurityGroups" : [
```

```
"sg-188d9f74"  
]  
}
```

レスポンス例

```
HTTP/1.1 204 No Content  
x-amzn-RequestId: 01234567-89ab-cdef-0123-456789abcdef
```

以下の資料も参照してください。

言語固有の AWS SDKs のいずれかでこの API を使用方法の詳細については、以下を参照してください。

- [AWS コマンドラインインターフェイス](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go v2](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript V3](#)
- [AWS SDK for Kotlin](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

PutAccountPreferences

このオペレーションを使用して、新しい EFS ファイルシステムとマウントターゲットリソースに、長い 17 文字 (63 ビット) または短い 8 文字 (32 ビット) のリソース ID を使用するように現在の AWS リージョン のアカウント環境設定を設定します。既存のリソース ID はすべて、変更の影響を受けません。EFS が長いリソース ID に移行するときに、オプトイン期間中に ID プリファレンスを設定できます。詳細については、「[Amazon EFS リソース ID の管理](#)」を参照してください。

Note

2021 年 10 月以降、短い 8 文字形式のリソース ID を使用するようにアカウント設定をしようとすると、エラーが表示されます。エラーが表示され、ファイルシステムとマウントターゲットリソースに短い IDs を使用する必要がある場合は、AWS サポートにお問い合わせください。

リクエストの構文

```
PUT /2015-02-01/account-preferences HTTP/1.1
Content-type: application/json

{
  "ResourceIdType": "string"
}
```

URI リクエストパラメータ

リクエストでは URI パラメータを使用しません。

リクエストボディ

リクエストは以下の JSON 形式のデータを受け入れます。

[ResourceIdType](#)

ユーザーの に設定する EFS リソース ID 設定を AWS アカウント、現在の LONG_ID (17 文字) AWS リージョン、または SHORT_ID (8 文字) で指定します。

Note

2021 年 10 月以降、アカウント設定を SHORT_ID にするとエラーが表示されます。エラーが表示され、ファイルシステムとマウントターゲットリソースに短い IDs を使用する必要がある場合は、AWS サポートにお問い合わせください。

型: 文字列

有効な値: LONG_ID | SHORT_ID

必須: はい

レスポンスの構文

```
HTTP/1.1 200
Content-type: application/json

{
  "ResourceIdPreference": {
    "ResourceIdType": "string",
    "Resources": [ "string" ]
  }
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

[ResourceIdPreference](#)

現在の AWS アカウントのリソースタイプとそのユーザーの ID 設定について説明します AWS リージョン。

型: [ResourceIdPreference](#) オブジェクト

エラー

BadRequest

リクエストの形式が正しくない場合や、無効なパラメータ値や必須パラメータの欠落などのエラーが含まれている場合に返されます。

HTTP ステータスコード: 400

InternalServerError

サーバー側でエラーが発生した場合に返されます。

HTTP ステータスコード: 500

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用方法の詳細については、以下を参照してください。

- [AWS コマンドラインインターフェイス](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go v2](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript V3](#)
- [AWS SDK for Kotlin](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

PutBackupPolicy

ファイルシステムのバックアップポリシーを更新します。このアクションは、ファイルシステムの自動バックアップをスタートまたは停止するために使用します。

リクエストの構文

```
PUT /2015-02-01/file-systems/FileSystemId/backup-policy HTTP/1.1
Content-type: application/json
```

```
{
  "BackupPolicy": {
    "Status": "string"
  }
}
```

URI リクエストパラメータ

リクエストでは、次の URI パラメータを使用します。

[FileSystemId](#)

バックアップポリシーを更新する EFS ファイルシステムを指定します。

長さの制約: 最大長は 128 です。

パターン: `^(arn:aws[-a-z]*:elasticfilesystem:[0-9a-z-:]+:file-system/fs-[0-9a-f]{8,40}|fs-[0-9a-f]{8,40})$`

必須: はい

リクエストボディ

リクエストは以下の JSON 形式のデータを受け入れます。

[BackupPolicy](#)

PutBackupPolicy リクエストに含まれるバックアップポリシー。

型: [BackupPolicy](#) オブジェクト

必須: はい

レスポンスの構文

```
HTTP/1.1 200
Content-type: application/json

{
  "BackupPolicy": {
    "Status": "string"
  }
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

[BackupPolicy](#)

ファイルシステムのバックアップポリシーについて説明し、自動バックアップをオンにするかオフにするかを示します。

型: [BackupPolicy](#) オブジェクト

エラー

BadRequest

リクエストの形式が正しくない場合や、無効なパラメータ値や必須パラメータの欠落などのエラーが含まれている場合に返されます。

HTTP ステータスコード: 400

FileSystemNotFound

指定されたFileSystemId値がリクエストの に存在しない場合に返されます AWS アカウント。

HTTP ステータスコード: 404

IncorrectFileSystemLifecycleState

ファイルシステムのライフサイクル状態が「使用可能」でない場合に返されます。

HTTP ステータスコード: 409

InternalServerError

サーバー側でエラーが発生した場合に返されます。

HTTP ステータスコード: 500

ValidationException

リクエストが行われた AWS リージョン で AWS Backup サービスが利用できない場合に返されます。

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 AWS SDKs のいずれかでこの API を使用方法の詳細については、以下を参照してください。

- [AWS コマンドラインインターフェイス](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go v2](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript V3](#)
- [AWS SDK for Kotlin](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

PutFileSystemPolicy

Amazon EFS ファイルシステムに Amazon EFS FileSystemPolicy を適用します。ファイル システムポリシーは IAM リソーススペースのポリシーで、複数のポリシーステートメントを含めることができます。ファイルシステムは、常に1つのファイルシステムポリシーを持っています。これは、デフォルトポリシー、またはこの API オペレーションを使用して設定または更新された明示的なポリシーにすることができます。EFS ファイル システムポリシーには 20,000 文字の制限があります。明示的なポリシーが設定されると、デフォルトポリシーが上書きされます。デフォルトのファイルシステムポリシーの詳細については、「[デフォルトの EFS ファイル システムポリシー](#)」を参照してください。

Note

EFS ファイル システムポリシーには 20,000 文字の制限があります。

このオペレーションには、elasticfilesystem:PutFileSystemPolicy アクションに対する許可が必要です。

リクエストの構文

```
PUT /2015-02-01/file-systems/FileSystemId/policy HTTP/1.1
Content-type: application/json

{
  "BypassPolicyLockoutSafetyCheck": boolean,
  "Policy": "string"
}
```

URI リクエストパラメータ

リクエストでは、次の URI パラメータを使用します。

[FileSystemId](#)

FileSystemPolicy の作成または更新を行う EFS ファイルシステムの ID。

長さの制約: 最大長は 128 です。

パターン: `^(arn:aws[-a-z]*:elasticfilesystem:[0-9a-z-:]+:file-system/fs-[0-9a-f]{8,40}|fs-[0-9a-f]{8,40})$`

必須: はい

リクエストボディ

リクエストは以下の JSON 形式のデータを受け入れます。

[BypassPolicyLockoutSafetyCheck](#)

(オプション) FileSystemPolicy ロックアウトのセーフティチェックをバイパスするかどうかを指定するブール値。ロックアウトセーフティチェックでは、リクエストに含まれるポリシーによって、ファイルシステムでの将来の PutFileSystemPolicy リクエストを行うことからリクエストを行う IAM プリンシパル をロックアウトするか、回避するかを決定します。BypassPolicyLockoutSafetyCheck から True に設定するのは、リクエストを行っている IAM プリンシパルがこのファイルシステムで後続の PutFileSystemPolicy リクエストを行わないようにする場合のみです。デフォルト値は False です。

型: ブール値

必須: いいえ

[Policy](#)

作成している FileSystemPolicy。JSON 形式のポリシー定義を受け入れます。EFS ファイルシステム ポリシーには 20,000 文字の制限があります。ファイルシステムポリシーを構成する要素の詳細については、「[Amazon EFS 内のリソースベースのポリシー](#)」を参照してください。

タイプ: 文字列

長さの制約: 最小長は 1 です。最大長は 20000 です。

パターン: `[\s\S]+`

必須: はい

レスポンスの構文

```
HTTP/1.1 200
Content-type: application/json

{
  "FileSystemId": "string",
```



```
"Policy": "string"  
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

[FileSystemId](#)

FileSystemPolicy が適用される EFS ファイルシステムを指定します。

タイプ: 文字列

長さの制約: 最大長は 128 です。

パターン: `^(arn:aws[-a-z]*:elasticfilesystem:[0-9a-z-:]+:file-system/fs-[0-9a-f]{8,40}|fs-[0-9a-f]{8,40})$`

[Policy](#)

EFS ファイルシステムの JSON 形式の FileSystemPolicy です。

タイプ: 文字列

長さの制約: 最小長は 1 です。最大長は 20000 です。

パターン: `[\s\S]+`

エラー

BadRequest

リクエストの形式が正しくない場合や、無効なパラメータ値や必須パラメータの欠落などのエラーが含まれている場合に返されます。

HTTP ステータスコード: 400

FileSystemNotFound

指定されたFileSystemId値がリクエストの に存在しない場合に返されます AWS アカウント。

HTTP ステータスコード: 404

IncorrectFileSystemLifeCycleState

ファイルシステムのライフサイクル状態が「使用可能」でない場合に返されます。

HTTP ステータスコード: 409

InternalServerError

サーバー側でエラーが発生した場合に返されます。

HTTP ステータスコード: 500

InvalidPolicyException

FileSystemPolicy が不正な形式であったり、有効ではないパラメータ値や必須パラメータの欠落などのエラーを含む場合に返されます。ポリシー ロックアウトセーフティ チェックエラーの場合に返却されます。

HTTP ステータスコード: 400

例

EFS ファイルシステムポリシーの作成

次のリクエストは、すべての AWS プリンシパル FileSystemPolicy が指定された EFS ファイルシステムを読み取りおよび書き込みアクセス許可でマウントできるようにする を作成します。

リクエスト例

```
PUT /2015-02-01/file-systems/fs-01234567/file-system-policy HTTP/1.1
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "elasticfilesystem:ClientMount",
        "elasticfilesystem:ClientWrite"
      ],
      "Principal": {
        "AWS": ["*"]
      }
    }
  ]
}
```

```
]
}
```

レスポンス例

```
{
  "Version": "2012-10-17",
  "Id": "1",
  "Statement": [
    {
      "Sid": "efs-statement-abcdef01-1111-bbbb-2222-111122224444",
      "Effect": "Allow",
      "Action": [
        "elasticfilesystem:ClientMount",
        "elasticfilesystem:ClientWrite"
      ],
      "Principal": {
        "AWS": ["*"]
      },
      "Resource": "arn:aws:elasticfilesystem:us-east-1:1111222233334444:file-
system/fs-01234567"
    }
  ]
}
```

以下の資料も参照してください。

言語固有の AWS SDKs のいずれかでこの API を使用方法の詳細については、以下を参照してください。

- [AWS コマンドラインインターフェイス](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go v2](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript V3](#)
- [AWS SDK for Kotlin](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)

- [AWS SDK for Ruby V3](#)

PutLifecycleConfiguration

このアクションを使用して、ファイルシステムのストレージを管理します。LifecycleConfiguration は、以下を定義する 1 つ以上の LifecyclePolicy オブジェクトで構成されます。

- **TransitionToIA** — ファイルシステム内のファイルをプライマリストレージ (標準ストレージクラス) から低頻度アクセス (IA) ストレージに移動するタイミング。
- **TransitionToArchive** — ファイルシステム内のファイルを現在のストレージクラス (IA または標準ストレージ) からアーカイブストレージに移動するタイミング。

ファイルシステムは、IA ストレージに移行する前にアーカイブストレージに移行することはできません。したがって、TransitionToArchive を設定しないか、TransitionToIA よりも後に設定する必要があります。

Note

アーカイブストレージクラスは、エラスティックスループットモードと汎用パフォーマンスモードを使用するファイルシステムでのみ利用できます。

- **TransitionToPrimaryStorageClass** — IA ストレージまたはアーカイブストレージでアクセスされたファイルシステム内のファイルを、プライマリストレージ (標準ストレージクラス) に戻すかどうか。

詳細については、「[ファイルシステムのストレージの管理](#)」を参照してください。

各 Amazon EFS ファイルシステムは、ファイルシステム内のすべてのファイルに適用される 1 つのライフサイクル設定をサポートしています。指定されたファイルシステムに LifecycleConfiguration のオブジェクトがすでに存在する場合、PutLifecycleConfiguration のコールは既存の設定を変更します。リクエスト本文に空の LifecyclePolicies 配列を含む PutLifecycleConfiguration 呼び出しは、既存の LifecycleConfiguration をすべて削除します。リクエストでは次のようにを指定します。

- ライフサイクル管理を有効化、無効化、または変更するファイルシステムの ID。
- ファイルをいつ IA ストレージやアーカイブストレージに移動し、いつプライマリストレージに戻すかを定義する、LifecyclePolicy オブジェクトの LifecyclePolicies 配列。

Note

Amazon EFS では、各 LifecyclePolicy オブジェクトが1つのトランジションしかないため、LifecyclePolicies 配列は別々の LifecyclePolicy オブジェクトで構成する必要があります。詳細については、次のセクションのリクエスト例を参照してください。

このオペレーションには、elasticfilesystem:PutLifecycleConfiguration オペレーションに対する許可が必要です。

暗号化されたファイルシステムにLifecycleConfigurationオブジェクトを適用するには、暗号化されたファイルシステムを作成したときと同じ AWS Key Management Service アクセス許可が必要です。

リクエストの構文

```
PUT /2015-02-01/file-systems/FileSystemId/lifecycle-configuration HTTP/1.1
Content-type: application/json
```

```
{
  "LifecyclePolicies": [
    {
      "TransitionToArchive": "string",
      "TransitionToIA": "string",
      "TransitionToPrimaryStorageClass": "string"
    }
  ]
}
```

URI リクエストパラメータ

リクエストでは、次の URI パラメータを使用します。

FileSystemId

LifecycleConfiguration オブジェクトを作成するファイルシステムの ID (文字列)。

長さの制約: 最大長は 128 です。

パターン: `^(arn:aws[-a-z]*:elasticfilesystem:[0-9a-z-:]+:file-system/fs-[0-9a-f]{8,40}|fs-[0-9a-f]{8,40})$`

必須: はい

リクエストボディ

リクエストは以下の JSON 形式のデータを受け入れます。

[LifecyclePolicies](#)

ファイルシステムの LifecycleConfiguration オブジェクトを定義する LifecyclePolicy オブジェクトの配列。LifecycleConfiguration オブジェクトは、ライフサイクル管理に次の通知を行います。

- **TransitionToIA** — ファイルシステム内のファイルをプライマリストレージ (標準ストレージクラス) から低頻度アクセス (IA) ストレージに移動するタイミング。
- **TransitionToArchive** — ファイルシステム内のファイルを現在のストレージクラス (IA または標準ストレージ) からアーカイブストレージに移動するタイミング。

ファイルシステムは、IA ストレージに移行する前にアーカイブストレージに移行することはできません。したがって、TransitionToArchive を設定しないか、TransitionToIA よりも後に設定する必要があります。

Note

アーカイブストレージクラスは、エラスティックスループットモードと汎用パフォーマンスモードを使用するファイルシステムでのみ利用できます。

- **TransitionToPrimaryStorageClass** — IA ストレージまたはアーカイブストレージでアクセスされたファイルシステム内のファイルを、プライマリストレージ (標準ストレージクラス) に戻すかどうか。

Note

put-lifecycle-configuration CLI コマンドまたは PutLifecycleConfiguration API アクションを使用する場合、Amazon EFS では、各 LifecyclePolicy オブジェクトに移行が 1 つだけあることが求められます。つまり、リクエスト本文では、LifecyclePolicies が LifecyclePolicy オブジェクトの配列として構成される必要があります (ストレージの移行ごとに 1 オブジェクト)。詳細については、次のセクションのリクエスト例を参照してください。

型: [LifecyclePolicy](#) オブジェクトの配列

配列メンバー: 最大数は 3 項目です。

必須: はい

レスポンスの構文

```
HTTP/1.1 200
Content-type: application/json

{
  "LifecyclePolicies": [
    {
      "TransitionToArchive": "string",
      "TransitionToIA": "string",
      "TransitionToPrimaryStorageClass": "string"
    }
  ]
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

[LifecyclePolicies](#)

ライフサイクル管理ポリシーの配列。EFS では、ファイルシステムごとに最大1つのポリシーがサポートされています。

型: [LifecyclePolicy](#) オブジェクトの配列

配列メンバー: 最大数は 3 項目です。

エラー

BadRequest

リクエストの形式が正しくない場合や、無効なパラメータ値や必須パラメータの欠落などのエラーが含まれている場合に返されます。

HTTP ステータスコード: 400

FileSystemNotFound

指定されたFileSystemId値がリクエストの に存在しない場合に返されます AWS アカウント。

HTTP ステータスコード: 404

IncorrectFileSystemLifecycleState

ファイルシステムのライフサイクル状態が「使用可能」でない場合に返されます。

HTTP ステータスコード: 409

InternalServerError

サーバー側でエラーが発生した場合に返されます。

HTTP ステータスコード: 500

例

ライフサイクル設定の作成

次の例では、PutLifecycleConfiguration アクションを使用して LifecyclePolicy オブジェクトを作成します。この例では、EFS に次のことを指示するライフサイクルポリシーを作成します。

- 過去 30 日間に標準ストレージでアクセスされていないファイルシステム内のファイルをすべて IA ストレージに移動する。
- 過去 90 日間に標準ストレージでアクセスされていないファイルシステム内のファイルをすべて アーカイブストレージに移動する。
- IA ストレージまたはアーカイブストレージでアクセスされたファイルを標準ストレージに戻す。
アーカイブストレージクラスは、エラスティックスループットモードと汎用パフォーマンスモードを使用するファイルシステムでのみ利用できます。

詳細については、「[EFS ストレージクラス](#)」と「[ファイルシステムのストレージの管理](#)」を参照してください。

リクエスト例

```
PUT /2015-02-01/file-systems/fs-0123456789abcdefb/lifecycle-configuration HTTP/1.1
Host: elasticfilesystem.us-west-2.amazonaws.com
x-amz-date: 20181122T232908Z
Authorization: <...>
Content-type: application/json
Content-Length: 86

{
  "LifecyclePolicies": [
    {
      "TransitionToArchive": "AFTER_90_DAYS"
    },
    {
      "TransitionToIA": "AFTER_30_DAYS"
    },
    {
      "TransitionToPrimaryStorage": "AFTER_1_ACCESS"
    }
  ]
}
```

レスポンス例

```
HTTP/1.1 200 OK
x-amzn-RequestId: 01234567-89ab-cdef-0123-456789abcdef
Content-type: application/json
Content-Length: 86

{
  "LifecyclePolicies": [
    {
      "TransitionToArchive": "AFTER_90_DAYS"
    },
    {
      "TransitionToIA": "AFTER_30_DAYS"
    },
    {
      "TransitionToPrimaryStorage": "AFTER_1_ACCESS"
    }
  ]
}
```

ライフサイクル設定 CLI リクエストの例

この例では、PutLifecycleConfiguration の使い方の一例を示します。

リクエスト例

```
aws efs put-lifecycle-configuration \  
  --file-system-id fs-0123456789abcdefb \  
  --lifecycle-policies "[{"TransitionToArchive":"AFTER_90_DAYS"},  
    {"TransitionToIA":"AFTER_30_DAYS"},  
    {"TransitionToPrimaryStorageClass":"AFTER_1_ACCESS"}]  
  --region us-west-2 \  
  --profile adminuser
```

レスポンス例

```
{  
  "LifecyclePolicies": [  
    {  
      "TransitionToArchive": "AFTER_90_DAYS"  
    },  
    {  
      "TransitionToIA": "AFTER_30_DAYS"  
    },  
    {  
      "TransitionToPrimaryStorageClass": "AFTER_1_ACCESS"  
    }  
  ]  
}
```

ライフサイクル管理の無効化

次の例では、指定したファイルシステムのライフサイクル管理を無効にします。

リクエスト例

```
PUT /2015-02-01/file-systems/fs-01234567/lifecycle-configuration HTTP/1.1  
Host: elasticfilesystem.us-west-2.amazonaws.com  
x-amz-date: 20181122T232908Z  
Authorization: <...>
```

```
Content-type: application/json
Content-Length: 86

{
  "LifecyclePolicies": [ ]
}
```

レスポンス例

```
HTTP/1.1 200 OK
x-amzn-RequestId: 01234567-89ab-cdef-0123-456789abcdef
Content-type: application/json
Content-Length: 86

{
  "LifecyclePolicies": [ ]
}
```

以下の資料も参照してください。

言語固有の AWS SDKs のいずれかでこの API を使用方法の詳細については、以下を参照してください。

- [AWS コマンドラインインターフェイス](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go v2](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript V3](#)
- [AWS SDK for Kotlin](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

TagResource

EFS リソースのタグの作成。この API オペレーションを使用して、EFS ファイルシステムおよびアクセスポイントのタグを作成できます。

このオペレーションには、elasticfilesystem:TagResource アクションに対する許可が必要です。

リクエストの構文

```
POST /2015-02-01/resource-tags/ResourceId HTTP/1.1
Content-type: application/json
```

```
{
  "Tags": [
    {
      "Key": "string",
      "Value": "string"
    }
  ]
}
```

URI リクエストパラメータ

リクエストでは、次の URI パラメータを使用します。

ResourceId

タグを作成する EFS リソースを指定する ID。

長さの制約: 最大長は 128 です。

パターン: `^(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: はい

リクエストボディ

リクエストは以下の JSON 形式のデータを受け入れます。

Tags

追加する Tag オブジェクトの配列。各 Tag オブジェクトタグはキーバリューのペアです。

型: [Tag](#) オブジェクトの配列

必須: はい

レスポンスの構文

```
HTTP/1.1 200
```

レスポンス要素

アクションが成功した場合、サービスは空の HTTP 本文を持つ HTTP 200 レスポンスを返します。

エラー

AccessPointNotFound

指定されたAccessPointId値がリクエストの に存在しない場合に返されます AWS アカウント。

HTTP ステータスコード: 404

BadRequest

リクエストの形式が正しくない場合や、無効なパラメータ値や必須パラメータがないなどのエラーが含まれている場合に返されます。

HTTP ステータスコード: 400

FileSystemNotFound

指定されたFileSystemId値がリクエストの に存在しない場合に返されます AWS アカウント。

HTTP ステータスコード: 404

InternalServerError

サーバー側でエラーが発生した場合に返されます。

HTTP ステータスコード: 500

例

ファイルシステム上でのタグの作成

次のリクエストは、指定されたファイルシステムに 3 つのタグ ("key1","key2",および "key3") を作成します。

リクエスト例

```
POST /2015-02-01/tag-resource/fs-01234567 HTTP/1.1
Host: elasticfilesystem.us-west-2.amazonaws.com
x-amz-date: 20140620T221118Z
Authorization: <...>
Content-Type: application/json
Content-Length: 160
```

```
{
  "Tags": [
    {
      "Key": "key1",
      "Value": "value1"
    },
    {
      "Key": "key2",
      "Value": "value2"
    },
    {
      "Key": "key3",
      "Value": "value3"
    }
  ]
}
```

レスポンス例

```
HTTP/1.1 204 no content
x-amzn-RequestId: 01234567-89ab-cdef-0123-456789abcdef
```

以下の資料も参照してください。

言語固有の AWS SDKs のいずれかでこの API を使用方法の詳細については、以下を参照してください。

- [AWS コマンドラインインターフェイス](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go v2](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript V3](#)
- [AWS SDK for Kotlin](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

UntagResource

EFS リソースからタグを削除します この API オペレーションを使用して、EFS ファイルシステムおよびアクセスポイントからタグを削除できます。

このオペレーションには、elasticfilesystem:UntagResource アクションに対する許可が必要です。

リクエストの構文

```
DELETE /2015-02-01/resource-tags/ResourceId?tagKeys=TagKeys HTTP/1.1
```

URI リクエストパラメータ

リクエストでは、次の URI パラメータを使用します。

ResourceId

タグを削除する EFS リソースを指定します。

長さの制約: 最大長は 128 です。

パターン: `^(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: はい

TagKeys

指定された EFS リソースから削除するキーバリュー タグペアのキー。

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 1、最大長は 128 です。

パターン: `^(?![aA]{1}[wW]{1}[sS]{1}:)([\\p{L}\\p{Z}\\p{N}_.:/=+\\-@]+)$`

必須: はい

リクエストボディ

リクエストにリクエスト本文がありません。

レスポンスの構文

```
HTTP/1.1 200
```

レスポンス要素

アクションが成功した場合、サービスは空の HTTP 本文を持つ HTTP 200 レスポンスを返します。

エラー

AccessPointNotFound

指定されたAccessPointId値がリクエストの に存在しない場合に返されます AWS アカウント。

HTTP ステータスコード: 404

BadRequest

リクエストの形式が正しくない場合や、無効なパラメータ値や必須パラメータがないなどのエラーが含まれている場合に返されます。

HTTP ステータスコード: 400

FileSystemNotFound

指定されたFileSystemId値がリクエストの に存在しない場合に返されます AWS アカウント。

HTTP ステータスコード: 404

InternalServerError

サーバー側でエラーが発生した場合に返されます。

HTTP ステータスコード: 500

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用方法の詳細については、以下を参照してください。

- [AWS コマンドラインインターフェイス](#)
- [AWS SDK for .NET](#)

- [AWS SDK for C++](#)
- [AWS SDK for Go v2](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript V3](#)
- [AWS SDK for Kotlin](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

UpdateFileSystem

スループットモード、または既存のファイルシステムのプロビジョニングされたスループットの量を更新します。

リクエストの構文

```
PUT /2015-02-01/file-systems/FileSystemId HTTP/1.1
Content-type: application/json

{
  "ProvisionedThroughputInMibps": number,
  "ThroughputMode": "string"
}
```

URI リクエストパラメータ

リクエストでは、次の URI パラメータを使用します。

[FileSystemId](#)

更新するファイル システムの ID。

長さの制約: 最大長は 128 です。

パターン: `^(arn:aws[-a-z]*:elasticfilesystem:[0-9a-z-:]+:file-system/fs-[0-9a-f]{8,40}|fs-[0-9a-f]{8,40})$`

必須: はい

リクエストボディ

リクエストは以下の JSON 形式のデータを受け入れます。

[ProvisionedThroughputInMibps](#)

(オプション) 作成するファイルシステムにプロビジョニングするスループットを、毎秒メビバイト (MiBps) で計測します。ThroughputMode が provisioned に設定されている場合は必須です。有効な値は 1~3414 MiBps で、上限はリージョンによって異なります。この制限を引き上げるには、[お問い合わせ](#)ください サポート。詳細については、Amazon EFS ユーザーガイドの「[引き上げ可能な Amazon EFS クォータ](#)」を参照してください。

型: 倍精度浮動小数点数

値の範囲: 最小値 は 1.0 です。

必須: いいえ

ThroughputMode

(オプション) ファイル システムのスループットモードを更新します。スループットモードを更新しない場合は、リクエストでこの値を指定する必要はありません。ThroughputMode を provisioned に変更する場合、ProvisionedThroughputInMibps の値も設定する必要があります。

型: 文字列

有効な値 : bursting | provisioned | elastic

必須 : いいえ

レスポンスの構文

HTTP/1.1 202

Content-type: application/json

```
{
  "AvailabilityZoneId": "string",
  "AvailabilityZoneName": "string",
  "CreationTime": number,
  "CreationToken": "string",
  "Encrypted": boolean,
  "FileSystemArn": "string",
  "FileSystemId": "string",
  "FileSystemProtection": {
    "ReplicationOverwriteProtection": "string"
  },
  "KmsKeyId": "string",
  "LifeCycleState": "string",
  "Name": "string",
  "NumberOfMountTargets": number,
  "OwnerId": "string",
  "PerformanceMode": "string",
  "ProvisionedThroughputInMibps": number,
```

```
{
  "SizeInBytes": {
    "Timestamp": number,
    "Value": number,
    "ValueInArchive": number,
    "ValueInIA": number,
    "ValueInStandard": number
  },
  "Tags": [
    {
      "Key": "string",
      "Value": "string"
    }
  ],
  "ThroughputMode": "string"
}
```

レスポンス要素

アクションが成功すると、HTTP 202 レスポンスが返されます。

サービスから以下のデータが JSON 形式で返されます。

[AvailabilityZoneId](#)

ファイルシステムが配置されているアベイラビリティゾーンの一意で一貫性のある識別子。これは、1 ゾーンファイルシステムでのみ有効です。たとえば、use1-az1は us-east-1 のアベイラビリティゾーン ID であり AWS リージョン、すべてのに同じ場所があります AWS アカウント。

タイプ: 文字列

[AvailabilityZoneName](#)

ファイルシステムが配置されている AWS アベイラビリティゾーンについて説明し、1 ゾーンファイルシステムでのみ有効です。詳細については、「Amazon EFS ユーザーガイド」の「[EFS ストレージクラスの使用](#)」を参照してください。

タイプ: 文字列

長さの制約: 最小長は 1 です。最大長 64

パターン: .+

[CreationTime](#)

ファイルシステムが作成された時間 (秒単位) (1970-01-01T00:00:00Z から)。

型: タイムスタンプ

[CreationToken](#)

リクエストで指定された不透明な文字列。

タイプ: 文字列

長さの制約: 最小長は 1 です。最大長 64

パターン: .+

[Encrypted](#)

true の場合はファイルシステムの暗号化を示すブール値。

型: ブール値

[FileSystemArn](#)

EFS ファイルシステムの Amazon リソースネーム (ARN) で、形式は `arn:aws:elasticfilesystem:region:account-id:file-system/file-system-id` です。サンプルデータの例: `arn:aws:elasticfilesystem:us-west-2:1111333322228888:file-system/fs-01234567`

タイプ: 文字列

[FileSystemId](#)

Amazon EFS によって割り当てられるファイルシステムの ID。

タイプ: 文字列

長さの制約: 最大長は 128 です。

パターン: `^(arn:aws[-a-z]*:elasticfilesystem:[0-9a-z-:]+:file-system/fs-[0-9a-f]{8,40}|fs-[0-9a-f]{8,40})$`

[FileSystemProtection](#)

ファイルシステムの保護について説明します。

型: [FileSystemProtectionDescription](#) オブジェクト

KmsKeyId

暗号化されたファイルシステムの保護 AWS KMS key に使用される の ID。

タイプ: 文字列

長さの制約: 最大長は 2,048 です。

パターン: `^([0-9a-f]{8}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{12}|mrk-[0-9a-f]{32}|alias/[a-zA-Z0-9/_-]+|(arn:aws[-a-z]*:kms:[a-z0-9-]+:\d{12}:((key/[0-9a-f]{8}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{12})|(key/mrk-[0-9a-f]{32})|(alias/[a-zA-Z0-9/_-]+))))$`

LifeCycleState

ファイルシステムのライフサイクルフェーズ。

型: 文字列

有効な値: `creating | available | updating | deleting | deleted | error`

Name

ファイルシステムには、Name タグをはじめとするタグを追加することができます。詳細については、「[CreateFileSystem](#)」を参照してください。ファイルシステムに Name タグがある場合、Amazon EFS はこのフィールドの値を返します。

タイプ: 文字列

長さの制約: 最大長は 256 です。

パターン: `^([\p{L}\p{Z}\p{N}_.: /+=\-\@]*)$`

NumberOfMountTargets

ファイルシステムが持つ現在のマウントターゲットの数。詳細については、「[CreateMountTarget](#)」を参照してください。

型: 整数

有効範囲: 最小値 は 0 です。

OwnerId

ファイルシステムを AWS アカウント 作成した。

タイプ: 文字列

長さの制限: 最大長は 14 です。

パターン: `^(\d{12})|(\d{4}-\d{4}-\d{4})$`

[PerformanceMode](#)

ファイル システムのパフォーマンス モード。

型: 文字列

有効な値 : `generalPurpose` | `maxIO`

[ProvisionedThroughputInMibps](#)

ファイルシステムのプロビジョニングされたスループットの量を MiBps で表したものの。ThroughputMode を `provisioned` に設定したファイルシステムに有効です。

型: 倍精度浮動小数点数

有効な範囲: 最小値 は 1.0 です。

[SizeInBytes](#)

ファイルシステムに保存されているデータの最新の測定サイズ (バイト) を Value のフィールドに、そのサイズが決定された時間を Timestamp のフィールドに入力しています。Timestamp 値は、1970-01-01T00:00:00Z 以降の整数秒数です。SizeInBytes値は、ファイルシステムの一貫したスナップショットのサイズを表すものではありませんが、ファイルシステムへの書き込みがない場合に結果整合性があります。つまり、SizeInBytesは、ファイルシステムが 2 時間以上変更されていない場合のみ、実際のサイズを表します。それ以外の場合、値はファイルシステムの特定の時点での正確なサイズではありません。

型: [FileSystemSize](#) オブジェクト

[Tags](#)

ファイルシステムに関連するタグで、Tag のオブジェクトの配列として表示されます。

型: [Tag](#) オブジェクトの配列

[ThroughputMode](#)

ファイルシステムのスループットモードを表示。詳細については、「Amazon EFS ユーザーガイド」の「[スループットモード](#)」を参照してください。

型: 文字列

有効な値: `bursting` | `provisioned` | `elastic`

エラー

BadRequest

リクエストの形式が正しくない場合や、無効なパラメータ値や必須パラメータの欠落などのエラーが含まれている場合に返されます。

HTTP ステータスコード: 400

FileSystemNotFound

指定されたFileSystemId値がリクエストの に存在しない場合に返されます AWS アカウント。

HTTP ステータスコード: 404

IncorrectFileSystemLifecycleState

ファイルシステムのライフサイクル状態が「使用可能」でない場合に返されます。

HTTP ステータスコード: 409

InsufficientThroughputCapacity

追加のスループットをプロビジョニングするのに十分な容量がない場合に返されます。この値は、プロビジョニングされたスループットモードでファイルシステムを作成しようとしたとき、既存のファイルシステムのプロビジョニングされたスループットを上げようとしたとき、または既存のファイルシステムをバーストからプロビジョニングされたスループットモードに変更しようとしたときに返されることがあります。後でもう一度お試しください。

HTTP ステータスコード: 503

InternalServerError

サーバー側でエラーが発生した場合に返されます。

HTTP ステータスコード: 500

ThroughputLimitExceeded

スループット制限の 1024 MiB/s に達したため、スループットモードまたはプロビジョニングされたスループットの量を変更できない場合に返されます。

HTTP ステータスコード: 400

TooManyRequests

スループットモードを変更するか、プロビジョニングされたスループット値を減らす前に少なくとも 24 時間以上待たない場合に返されます。

HTTP ステータスコード: 429

以下も参照してください。

言語固有の AWS SDKs のいずれかでこの API を使用方法の詳細については、以下を参照してください。

- [AWS コマンドラインインターフェイス](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go v2](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript V3](#)
- [AWS SDK for Kotlin](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

UpdateFileSystemProtection

ファイルシステムの保護を更新します。

このオペレーションには、elasticfilesystem:UpdateFileSystemProtection アクションに対する許可が必要です。

リクエストの構文

```
PUT /2015-02-01/file-systems/FileSystemId/protection HTTP/1.1
Content-type: application/json

{
  "ReplicationOverwriteProtection": "string"
}
```

URI リクエストパラメータ

リクエストでは、次の URI パラメータを使用します。

[FileSystemId](#)

更新するファイルシステムの ID。

長さの制約: 最大長は 128 です。

パターン: `^(arn:aws[-a-z]*:elasticfilesystem:[0-9a-z-:]+:file-system/fs-[0-9a-f]{8,40}|fs-[0-9a-f]{8,40})$`

必須: はい

リクエストボディ

リクエストは以下の JSON 形式のデータを受け入れます。

[ReplicationOverwriteProtection](#)

ファイルシステムのレプリケーション上書き保護の状態。

- **ENABLED** - ファイルシステムをレプリケーション設定のデステイネーションファイルシステムとして使用することはできません。ファイルシステムは書き込み可能です。レプリケーション上書き保護はデフォルトで **ENABLED** です。

- **DISABLED** - ファイルシステムをレプリケーション設定のデスティネーションファイルシステムとして使用することはできません。ファイルシステムは読み取り専用で、EFS レプリケーションでのみ変更できます。
- **REPLICATING**— ファイルシステムがレプリケーション設定のデスティネーションファイルシステムとして使用されています。ファイルシステムは読み取り専用で、EFS レプリケーションでのみ変更されます。

レプリケーション設定を削除すると、ファイルシステムのレプリケーション上書き保護が再び有効になり、ファイルシステムが書き込み可能になります。

型: 文字列

有効な値: ENABLED | DISABLED | REPLICATING

必須: いいえ

レスポンスの構文

```
HTTP/1.1 200
Content-type: application/json

{
  "ReplicationOverwriteProtection": "string"
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

[ReplicationOverwriteProtection](#)

ファイルシステムのレプリケーション上書き保護の状態。

- **ENABLED** - ファイルシステムをレプリケーション設定のデスティネーションファイルシステムとして使用することはできません。ファイルシステムは書き込み可能です。レプリケーション上書き保護はデフォルトで **ENABLED** です。
- **DISABLED** - ファイルシステムをレプリケーション設定のデスティネーションファイルシステムとして使用することはできません。ファイルシステムは読み取り専用で、EFS レプリケーションでのみ変更できます。

- REPLICATING— ファイルシステムがレプリケーション設定のデステイネーションファイルシステムとして使用されています。ファイルシステムは読み取り専用であり、EFS レプリケーションによってのみ変更されます。

レプリケーション設定を削除すると、ファイルシステムのレプリケーション上書き保護が再び有効になり、ファイルシステムが書き込み可能になります。

型: 文字列

有効な値 : ENABLED | DISABLED | REPLICATING

エラー

BadRequest

リクエストの形式が正しくない場合や、無効なパラメータ値や必須パラメータの欠落などのエラーが含まれている場合に返されます。

HTTP ステータスコード: 400

FileSystemNotFound

指定されたFileSystemId値がリクエストの に存在しない場合に返されます AWS アカウント。

HTTP ステータスコード: 404

IncorrectFileSystemLifecycleState

ファイルシステムのライフサイクル状態が「使用可能」でない場合に返されます。

HTTP ステータスコード: 409

InsufficientThroughputCapacity

追加のスループットをプロビジョニングするのに十分な容量がない場合に返されます。この値は、プロビジョニングされたスループットモードでファイルシステムを作成しようとしたとき、既存のファイルシステムのプロビジョニングされたスループットを上げようとしたとき、または既存のファイルシステムをバーストからプロビジョニングされたスループットモードに変更しようとしたときに返されることがあります。後でもう一度お試しください。

HTTP ステータスコード: 503

InternalServerError

サーバー側でエラーが発生した場合に返されます。

HTTP ステータスコード: 500

ReplicationAlreadyExists

ファイルシステムが既にレプリケーション設定に含まれている場合に返されます。>

HTTP ステータスコード: 409

ThroughputLimitExceeded

スループット制限の 1024 MiB/s に達したため、スループットモードまたはプロビジョニングされたスループットの量を変更できない場合に返されます。

HTTP ステータスコード: 400

TooManyRequests

スループットモードを変更するか、プロビジョニングされたスループット値を減らす前に少なくとも 24 時間以上待たない場合に返されます。

HTTP ステータスコード: 429

以下も参照してください。

言語固有の AWS SDKs のいずれかでこの API を使用方法の詳細については、以下を参照してください。

- [AWS コマンドラインインターフェイス](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go v2](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript V3](#)
- [AWS SDK for Kotlin](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)

- [AWS SDK for Ruby V3](#)

データ型

以下のデータ型 (タイプ) がサポートされています。

- [AccessPointDescription](#)
- [BackupPolicy](#)
- [CreationInfo](#)
- [Destination](#)
- [DestinationToCreate](#)
- [FileSystemDescription](#)
- [FileSystemProtectionDescription](#)
- [FileSystemSize](#)
- [LifecyclePolicy](#)
- [MountTargetDescription](#)
- [PosixUser](#)
- [ReplicationConfigurationDescription](#)
- [ResourceIdPreference](#)
- [RootDirectory](#)
- [Tag](#)

AccessPointDescription

EFS ファイル システムアクセスポイントについて説明します。

コンテンツ

AccessPointArn

アクセスポイントに関連付けられている一意の Amazon リソースネーム (ARN)。

タイプ: 文字列

長さの制約: 最大長は 128 です。

Pattern: `^arn:aws[-a-z]*:elasticfilesystem:[0-9a-z-:]+:access-point/fsap-[0-9a-f]{8,40}$`

必須: いいえ

AccessPointId

Amazon EFS によって割り当てられたアクセスポイントの ID。

タイプ: 文字列

長さの制約: 最大長は 128 です。

Pattern: `^(arn:aws[-a-z]*:elasticfilesystem:[0-9a-z-:]+:access-point/fsap-[0-9a-f]{8,40}|fsap-[0-9a-f]{8,40})$`

必須: いいえ

ClientToken

冪等性の作成を保証するためにリクエストで指定された不透明な文字列。

タイプ: 文字列

長さの制約: 最小長は 1 です。最大長 64

パターン: `.+`

必須: いいえ

FileSystemId

アクセスポイントが適用される EFS ファイルシステムの ID。

タイプ: 文字列

長さの制約: 最大長は 128 です。

Pattern: `^(arn:aws[-a-z]*:elasticfilesystem:[0-9a-z-:]+:file-system/fs-[0-9a-f]{8,40}|fs-[0-9a-f]{8,40})$`

必須: いいえ

LifeCycleState

アクセスポイントのライフサイクルフェーズを識別します。

タイプ: 文字列

有効な値: `creating | available | updating | deleting | deleted | error`

必須: いいえ

Name

このアクセスポイントの名前。これは、Name タグの値です。

タイプ: 文字列

必須: いいえ

OwnerId

アクセスポイントリソースを所有 AWS アカウント する を識別します。

タイプ: 文字列

長さの制限: 最大長は 14 です。

パターン: `^(\\d{12})|(\\d{4}-\\d{4}-\\d{4})$`

必須: いいえ

PosixUser

アクセスポイント上の、ユーザー ID、グループ ID、およびセカンダリグループ ID を含む完全な POSIX アイデンティティ。アクセスポイントを使用する NFS クライアントによるすべてのファイルオペレーションに使用されます。

型: [PosixUser](#) オブジェクト

必須: いいえ

RootDirectory

アクセスポイントを使用して、アクセスポイントが NFS クライアントにルートディレクトリとして公開する EFS ファイルシステム上のディレクトリ。

型: [RootDirectory](#) オブジェクト

必須: いいえ

Tags

アクセスポイントに関連付けられたタグ。タグオブジェクトの配列として表示されます。

型: [Tag](#) オブジェクトの配列

必須: いいえ

以下の資料も参照してください。

言語固有の AWS SDKs のいずれかでこの API を使用方法の詳細については、以下を参照してください。

- [AWS SDK for C++](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

BackupPolicy

毎日の自動バックアップを作成するために使用されるファイルシステムのバックアップポリシー。ステータスの値が **ENABLED** の場合、ファイルシステムは自動的にバックアップされます。詳細については、「[自動バックアップ](#)」を参照してください。

内容

Status

ファイルシステムの バックアップ ポリシーの状態を説明します。

- **ENABLED** - EFS はファイルシステムを自動的にバックアップします。
- **ENABLING** - EFS はファイルシステムの自動バックアップをオンにします。
- **DISABLED** - ファイルシステムの自動バックアップがオフになっています。
- **DISABLING** - EFS はファイルシステムの自動バックアップをオフにします。

タイプ: 文字列

有効な値: **ENABLED** | **ENABLING** | **DISABLED** | **DISABLING**

必須: はい

以下の資料も参照してください。

言語固有の AWS SDKs のいずれかでこの API を使用方法の詳細については、以下を参照してください。

- [AWS SDK for C++](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

CreationInfo

指定した `RootDirectory > Path` が存在しない場合は必須です。アクセスポイントの `RootDirectory > Path` に適用する POSIX ID とアクセス許可を指定します。アクセスポイントのルートディレクトリが存在しない場合、クライアントがアクセスポイントに接続すると EFS はこれらの設定を使用して作成します。CreationInfo を指定する場合は、すべてのプロパティの値を含める必要があります。

Amazon EFS では、CreationInfo (OwnUid、OwnGID、ディレクトリのアクセス許可) を指定した場合にのみ、ルートディレクトリが作成されます。この情報を指定しない場合、Amazon EFS はルートディレクトリを作成しません。ルートディレクトリが存在しない場合に、アクセスポイントを使用してマウントしようとする、失敗します。

Important

CreationInfo を指定せず、指定した `RootDirectory` が存在しない場合、アクセスポイントを使用してファイルシステムをマウントしようとする、失敗します。

内容

OwnerGid

`RootDirectory` に適用する POSIX グループ ID を指定します。0 から 2^{32} (4294967295) までの値を受け入れます。

型: Long

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

OwnerUid

`RootDirectory` に適用する POSIX ユーザー ID を指定します。0 から 2^{32} (4294967295) までの値を受け入れます。

型: Long

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

Permissions

RootDirectory に適用する POSIX アクセス許可を、ファイルのモードビットを表す 8 進数の形式で指定します。

タイプ: 文字列

長さの制約: 最小長は 3 です。最大長は 4 です。

Pattern: `^[0-7]{3,4}$`

必須: はい

以下の資料も参照してください。

言語固有の AWS SDKs のいずれかでこの API を使用方法の詳細については、以下を参照してください。

- [AWS SDK for C++](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

Destination

レプリケーション設定内のデスティネーションファイルシステムを記述します。

内容

FileSystemId

デスティネーション Amazon EFS ファイルシステムの ID。

タイプ: 文字列

長さの制約: 最大長は 128 です。

Pattern: `^(arn:aws[-a-z]*:elasticfilesystem:[0-9a-z-:]+:file-system/fs-[0-9a-f]{8,40}|fs-[0-9a-f]{8,40})$`

必須: はい

Region

AWS リージョン 送信先ファイルシステムが配置されている。

タイプ: 文字列

長さの制約: 最小長は 1 です。最大長 64

Pattern: `^[a-z]{2}-((iso[a-z]{0,1}-)|(gov-)){0,1}[a-z]+-{0,1}[0-9]{0,1}$`

必須: はい

Status

レプリケーション設定のステータスについて説明します。レプリケーションステータスの詳細については、「[Amazon EFS ユーザーガイド](#)」の「[レプリケーションの詳細の表示](#)」を参照してください。

タイプ: 文字列

有効な値: ENABLED | ENABLING | DELETING | ERROR | PAUSED | PAUSING

必須: はい

LastReplicatedTimestamp

デスティネーションファイルシステムで最新の同期が正常に完了した時刻。この時間より前にソースファイルシステム上のデータに加えられた変更は、デスティネーションファイルシステムに正常にレプリケートされました。この時間以降に発生した変更は、完全にはレプリケートされない可能性があります。

型: タイムスタンプ

必須: いいえ

OwnerId

AWS アカウント 送信先ファイルシステムが存在する の ID。

タイプ: 文字列

長さの制限: 最大長は 14 です。

パターン: `^\d{12})|(\d{4}-\d{4}-\d{4})$`

必須: いいえ

RoleArn

Amazon EFS が代わりにレプリケーションを実行できるようにするソースアカウントの IAM ロールの Amazon リソースネーム (ARN)。これは、同じアカウントのレプリケーションではオプションであり、クロスアカウントのレプリケーションでは必須です。

タイプ: 文字列

長さの制約: 最大長は 2,048 です。

パターン: `arn:(aws[a-zA-Z-]*)?:iam::\d{12}:role/?[a-zA-Z_0-9+=,.\@-_/]+`

必須: いいえ

StatusMessage

レプリケーション先設定の PAUSEDまたはERRORR状態に関する詳細を提供するメッセージ。レプリケーションステータスメッセージの詳細については、「[Amazon EFS ユーザーガイド](#)」の「[レプリケーションの詳細の表示](#)」を参照してください。

タイプ: 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 AWS SDKs のいずれかでこの API を使用方法の詳細については、以下を参照してください。

- [AWS SDK for C++](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

DestinationToCreate

レプリケーション設定の新規または既存のデスティネーションファイルシステムについて説明します。

- 新しいファイルシステムにレプリケートする場合は、送信先ファイルシステムのファイルシステム ID を指定しないでください。Amazon EFS は、新しい空のファイルシステムを作成します。1 ゾーンストレージの場合は、ファイルシステムを作成するアベイラビリティーゾーンを指定します。デフォルトの KMS AWS Key Management Service キー以外の キーを使用するには、それを指定します。詳細については、[「Amazon EFS ユーザーガイド」の「新しい Amazon EFS ファイルシステムへのレプリケーションの設定」](#)を参照してください。 EFS

Note

ファイルシステムの作成後は、KMS キーまたはパフォーマンスモードを変更することはできません。

- ソースファイルシステムと同じアカウントにある既存のファイルシステムにレプリケートする場合は、レプリケート先のファイルシステムの ID または Amazon リソースネーム (ARN) を指定する必要があります。ファイルシステムのレプリケーション上書き保護を無効にする必要があります。詳細については、[「Amazon EFS ユーザーガイド」の「既存のファイルシステムへのレプリケーション」](#)を参照してください。 EFS
- ファイルシステムをソースファイルシステムとは異なるアカウントにあるファイルシステムにレプリケートする場合 (クロスアカウントレプリケーション)、ファイルシステムの ARN と、Amazon EFS がレプリケート先アカウントでレプリケーションを実行できるようにする IAM ロールを指定する必要があります。ファイルシステムのレプリケーション上書き保護を無効にする必要があります。詳細については、[「Amazon EFS ユーザーガイド」の「レプリケートする AWS アカウント」](#)を参照してください。 EFS

内容

AvailabilityZoneName

1 ゾーンストレージを使用するファイルシステムを作成するには、デスティネーションファイルシステムを作成するアベイラビリティーゾーンの名前を指定します。

タイプ: 文字列

長さの制約: 最小長は 1 です。最大長 64

パターン: .+

必須: いいえ

FileSystemId

送信先に使用するファイルシステムの ID または ARN。クロスアカウントレプリケーションの場合、これは ARN である必要があります。ファイルシステムのレプリケーション上書きレプリケーションを無効にする必要があります。ID または ARN が指定されていない場合、新しいファイルシステムが作成されます。

Note

既存のファイルシステムへのレプリケーションを最初に設定すると、Amazon EFS はソースファイルシステム内のデータと一致するように、送信先ファイルシステムにデータを書き込むか、送信先ファイルシステムから既存のデータを削除します。送信先ファイルシステムのデータを変更しない場合は、代わりに新しいファイルシステムにレプリケートする必要があります。詳細については、「<https://docs.aws.amazon.com/efs/latest/ug/create-replication.html>」を参照してください。

タイプ: 文字列

長さの制約: 最大長は 128 です。

パターン: `^(arn:aws[-a-z]*:elasticfilesystem:[0-9a-z-:]+:file-system/fs-[0-9a-f]{8,40}|fs-[0-9a-f]{8,40})$`

必須: いいえ

KmsKeyId

送信先ファイルシステムの暗号化に使用する AWS Key Management Service (AWS KMS) キーを指定します。KMS キーを指定しない場合、Amazon EFS は、Amazon EFS 用のデフォルト KMS キー、`/aws/elasticfilesystem` を使用します。この ID は以下のいずれかの形式になります。

- キー ID - キーの一意の識別子 (例: 1234abcd-12ab-34cd-56ef-1234567890ab)。
- ARN - キーの ARN。例: `arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab`。

- キーのエイリアス - 以前に作成したキーの表示名 (例: alias/projectKey1)。
- キー エイリアス ARN - キー エイリアスの ARN (例: arn:aws:kms:us-west-2:444455556666:alias/projectKey1)。

タイプ: 文字列

長さの制約: 最大長は 2,048 です。

パターン: `^([0-9a-f]{8}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{12}|mrk-[0-9a-f]{32}|alias/[a-zA-Z0-9/_-]+|(arn:aws[-a-z]*:kms:[a-z0-9-]+:\d{12}:((key/[0-9a-f]{8}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{12})|(key/mrk-[0-9a-f]{32})|(alias/[a-zA-Z0-9/_-]+))))$`

必須: いいえ

Region

リージョンストレージを使用するファイルシステムを作成するには、送信先ファイルシステム AWS リージョン を作成する を指定します。リージョンは、ソースファイルシステムを所有 AWS アカウント する に対して有効にする必要があります。詳細については、AWS 「全般リファレンスガイド」の [「の管理 AWS リージョン」](#) を参照してください。

タイプ: 文字列

長さの制約: 最小長は 1 です。最大長 64

パターン: `^[a-z]{2}-((iso[a-z]{0,1}-)|(gov-)){0,1}[a-z]+-{0,1}[0-9]{0,1}$`

必須: いいえ

RoleArn

Amazon EFS が代わりにレプリケーションを実行できるようにするソースアカウントの IAM ロールの Amazon リソースネーム (ARN)。これは、同じアカウントのレプリケーションではオプションであり、クロスアカウントのレプリケーションで必要です。

タイプ: 文字列

長さの制約: 最大長は 2,048 です。

パターン: `arn:(aws[a-zA-Z-]*)?:iam::\d{12}:role/?[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 AWS SDKs のいずれかでこの API を使用方法の詳細については、以下を参照してください。

- [AWS SDK for C++](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

FileSystemDescription

ファイルシステムの説明。

内容

CreationTime

ファイルシステムが作成された時間 (秒単位) (1970-01-01T00:00:00Z 以降)。

型: タイムスタンプ

必須: はい

CreationToken

リクエストで指定された不透明な文字列。

タイプ: 文字列

長さの制約: 最小長は 1 です。最大長 64

Pattern: .+

必須: はい

FileSystemId

Amazon EFS によって割り当てられたファイルシステムの ID。

タイプ: 文字列

長さの制約: 最大長は 128 です。

Pattern: `^(arn:aws[-a-z]*:elasticfilesystem:[0-9a-z-:]+:file-system/fs-[0-9a-f]{8,40}|fs-[0-9a-f]{8,40})$`

必須: はい

LifeCycleState

ファイルシステムのライフサイクルフェーズ。

タイプ: 文字列

有効な値: creating | available | updating | deleting | deleted | error

必須: はい

NumberOfMountTargets

ファイル・システムが持つ現在のマウント・ターゲットの数。詳細については、「[CreateMountTarget](#)」を参照してください。

型: 整数

値の範囲: 最小値 は 0 です。

必須: はい

OwnerId

ファイルシステムを AWS アカウント 作成した。

タイプ: 文字列

長さの制限: 最大長は 14 です。

Pattern: ^(\d{12})|(\d{4}-\d{4}-\d{4})\$

必須: はい

PerformanceMode

ファイル システムのパフォーマンス モード。

タイプ: 文字列

有効な値: generalPurpose | maxIO

必須: はい

SizeInBytes

ファイルシステムに保存されているデータの最新の既知の計測サイズ (バイト単位) を Value のフィールドに、そのサイズが決定された時刻を Timestamp のフィールドに入力しています。Timestamp 値は、1970-01-01T00:00:00Z 以降の整数秒数です。SizeInBytes値は、ファイルシステムの一貫したスナップショットのサイズを表すものではありませんが、ファイルシステムへの書き込みがない場合に結果整合性があります。つまり、SizeInBytesは、ファイルシ

システムが 2 時間以上変更されていない場合のみ、実際のサイズを表します。それ以外の場合、値はファイルシステムの特定の時点での正確なサイズではありません。

型: [FileSystemSize](#) オブジェクト

必須: はい

Tags

ファイルシステムに関連するタグで、Tag オブジェクトの配列として表示されます。

型: [Tag](#) オブジェクトの配列

必須: はい

AvailabilityZoneId

ファイルシステムが配置されているアベイラビリティゾーンの一意で一貫性のある識別子。これは、1 ゾーンファイルシステムでのみ有効です。たとえば、use1-az1は us-east-1 のアベイラビリティゾーン ID であり AWS リージョン、すべての に同じ場所があります AWS アカウント。

タイプ: 文字列

必須: いいえ

AvailabilityZoneName

ファイルシステムが配置されている AWS アベイラビリティゾーンについて説明し、1 ゾーンファイルシステムでのみ有効です。詳細については、「Amazon EFS ユーザーガイド」の「[EFS ストレージクラスの使用](#)」を参照してください。

タイプ: 文字列

長さの制約: 最小長は 1 です。最大長 64

パターン: .+

必須: いいえ

Encrypted

true の場合はファイルシステムの暗号化を示すブール値。

型: ブール

必須: いいえ

FileSystemArn

EFS ファイルシステムの Amazon リソースネーム (ARN)、フォーマット `arn:aws:elasticfilesystem:region:account-id:file-system/file-system-id`。サンプルデータの例: `arn:aws:elasticfilesystem:us-west-2:1111333322228888:file-system/fs-01234567`

タイプ: 文字列

必須: いいえ

FileSystemProtection

ファイルシステムの保護について説明します。

型: [FileSystemProtectionDescription](#) オブジェクト

必須: いいえ

KmsKeyId

暗号化されたファイルシステムの保護 AWS KMS key に使用される の ID。

タイプ: 文字列

長さの制約: 最大長は 2,048 です。

パターン: `^([0-9a-f]{8}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{12}|mrk-[0-9a-f]{32}|alias/[a-zA-Z0-9/_-]+|(arn:aws[-a-z]*:kms:[a-z0-9-]+:\d{12}:((key/[0-9a-f]{8}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{12})|(key/mrk-[0-9a-f]{32})|(alias/[a-zA-Z0-9/_-]+))))$`

必須: いいえ

Name

ファイルシステムには、Name タグをはじめとするタグを追加することができます。詳細については、「[CreateFileSystem](#)」を参照してください。ファイルシステムに Name タグがある場合、Amazon EFS はこのフィールドの値を返します。

タイプ: 文字列

長さの制約: 最大長は 256 です。

パターン: `^([\p{L}\p{Z}\p{N}_.: /+=\ -@]*)$`

必須: いいえ

ProvisionedThroughputInMibps

ファイルシステムのプロビジョニングされたスループットの量を MiBps で表したものの。ThroughputMode を provisioned に設定したファイルシステムに有効です。

型: 倍精度浮動小数点数

値の範囲: 最小値 は 1.0 です。

必須: いいえ

ThroughputMode

ファイルシステムのスループットモードを表示します。詳細については、「Amazon EFS ユーザーガイド」の「[スループットモード](#)」を参照してください。

タイプ: 文字列

有効な値: bursting | provisioned | elastic

必須: いいえ

以下の資料も参照してください。

言語固有の AWS SDKs のいずれかでこの API を使用方法の詳細については、以下を参照してください。

- [AWS SDK for C++](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

FileSystemProtectionDescription

ファイルシステムの保護について説明します。

内容

ReplicationOverwriteProtection

ファイルシステムのレプリケーション上書き保護の状態。

- **ENABLED** - ファイルシステムをレプリケーション設定のデスティネーションファイルシステムとして使用することはできません。ファイルシステムは書き込み可能です。レプリケーション上書き保護はデフォルトで **ENABLED** です。
- **DISABLED** - ファイルシステムをレプリケーション設定のデスティネーションファイルシステムとして使用することはできません。ファイルシステムは読み取り専用で、EFS レプリケーションでのみ変更できます。
- **REPLICATING**— ファイルシステムがレプリケーション設定のデスティネーションファイルシステムとして使用されています。ファイルシステムは読み取り専用で、EFS レプリケーションによってのみ変更されます。

レプリケーション設定を削除すると、ファイルシステムのレプリケーション上書き保護が再び有効になり、ファイルシステムが書き込み可能になります。

タイプ: 文字列

有効な値: **ENABLED** | **DISABLED** | **REPLICATING**

必須: いいえ

以下の資料も参照してください。

言語固有の AWS SDKs のいずれかでこの API を使用方法の詳細については、以下を参照してください。

- [AWS SDK for C++](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

FileSystemSize

ファイル・システムに格納されているデータの最新の既知のメーター・サイズ (バイト単位)。Valueフィールド、およびそのサイズがその中で決定された時刻Timestampフィールド。この値は、ファイルシステムの一貫したスナップショットのサイズを表すものではありませんが、ファイルシステムへの書き込みがない際に結果整合性があります。つまり、この値は実際のサイズを表し、ファイルシステムが数時間以上変更されない場合のみです。それ以外の場合、値は必ずしもファイルシステムの特定の時点での正確なサイズではありません。

内容

Value

ファイルシステムに保存されているデータの最新の既知の従量制サイズ (バイト単位)。

型: Long

有効な範囲: 最小値 は 0 です。

必須: はい

Timestamp

Value フィールドで返されたデータのサイズが決定された時間です。値は、1970-01-01T00:00:00Z 以降の整数秒数です。

型: タイムスタンプ

必須: いいえ

ValueInArchive

アーカイブストレージクラスに保存されているデータの最新の既知の計測サイズ (バイト単位)。

タイプ : 長整数

有効な範囲: 最小値 は 0 です。

必須: いいえ

ValueInIA

低頻度アクセス ストレージクラスに保存されているデータの最新の既知の従量制サイズ (バイト単位)。

型: Long

有効な範囲: 最小値 は 0 です。

必須: いいえ

ValueInStandard

スタンダード ストレージクラスに保存されているデータの最新の既知の計測されたサイズ (バイト単位)。

型: Long

有効な範囲: 最小値 は 0 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 AWS SDKs のいずれかでこの API を使用方法の詳細については、以下を参照してください。

- [AWS SDK for C++](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

LifecyclePolicy

ライフサイクル管理で使用する、ストレージクラス間でファイルを移行するタイミングを指定するポリシーについて説明します。詳細については、「[ファイルシステムのストレージの管理](#)」を参照してください。

Note

put-lifecycle-configuration CLIコマンドまたは PutLifecycleConfiguration APIアクションを使用する場合、Amazon EFS では、各 LifecyclePolicy オブジェクトに移行が 1 つだけあることが求められます。つまり、リクエスト本文では、LifecyclePolicies が LifecyclePolicy オブジェクトの配列として構成される必要があります (移行ごとに 1 オブジェクト)。詳細については、[PutLifecycleConfiguration](#) の「リクエスト例」を参照してください。

内容

TransitionToArchive

プライマリストレージ (標準ストレージクラス) でファイルが最後にアクセスされてから、アーカイブストレージに移動するまでの日数。ディレクトリの内容を一覧表示するなどのメタデータオペレーションは、ファイルアクセスイベントとしてカウントされません。

タイプ: 文字列

有効な値: AFTER_1_DAY | AFTER_7_DAYS | AFTER_14_DAYS | AFTER_30_DAYS
| AFTER_60_DAYS | AFTER_90_DAYS | AFTER_180_DAYS | AFTER_270_DAYS |
AFTER_365_DAYS

必須: いいえ

TransitionToIA

プライマリストレージ (標準ストレージクラス) でファイルが最後にアクセスされてから、低頻度アクセス (IA) ストレージに移動するまでの日数。ディレクトリの内容を一覧表示するなどのメタデータオペレーションは、ファイルアクセスイベントとしてカウントされません。

タイプ: 文字列

有効な値: AFTER_7_DAYS | AFTER_14_DAYS | AFTER_30_DAYS | AFTER_60_DAYS
| AFTER_90_DAYS | AFTER_1_DAY | AFTER_180_DAYS | AFTER_270_DAYS |
AFTER_365_DAYS

必須: いいえ

TransitionToPrimaryStorageClass

IA ストレージまたはアーカイブストレージでアクセスしたファイルをプライマリ (標準) ストレージに戻すかどうか。ディレクトリの内容を一覧表示するなどのメタデータオペレーションは、ファイルアクセスイベントとしてカウントされません。

タイプ: 文字列

有効な値: AFTER_1_ACCESS

必須: いいえ

以下の資料も参照してください。

言語固有の AWS SDKs のいずれかでこの API を使用方法の詳細については、以下を参照してください。

- [AWS SDK for C++](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

MountTargetDescription

マウントターゲットの説明です。

内容

FileSystemId

マウントターゲットの対象となるファイルシステムの ID。

タイプ: 文字列

長さの制約: 最大長は 128 です。

パターン: `^(arn:aws[-a-z]*:elasticfilesystem:[0-9a-z-:]+:file-system/fs-[0-9a-f]{8,40}|fs-[0-9a-f]{8,40})$`

必須: はい

LifeCycleState

マウントターゲットのライフサイクル状態。

型: 文字列

有効な値: `creating | available | updating | deleting | deleted | error`

必須: はい

MountTargetId

システム割り当てマウントターゲット ID。

タイプ: 文字列

長さの制限: 最小長は 13 です。最大長は 45 です。

パターン: `^fsmt-[0-9a-f]{8,40}$`

必須: はい

SubnetId

マウントターゲットのサブネットの ID。

タイプ: 文字列

長さの制限: 最小長は 15 です。最大長は 47 です。

パターン: `^subnet-[0-9a-f]{8,40}$`

必須: はい

AvailabilityZoneId

マウントターゲットが存在するアベイラビリティーゾーンの一意で一貫性のある識別子。たとえば、`use1-az1`は `us-east-1` リージョンの AZ ID であり、すべての 同じ場所があります AWS アカウント。

タイプ: 文字列

必須: いいえ

AvailabilityZoneName

マウントターゲットが配置されているアベイラビリティーゾーンの名前。アベイラビリティーゾーンは、それぞれの名前に個別にマッピングされます AWS アカウント。たとえば、`us-east-1a`のアベイラビリティーゾーンが別の のアベイラビリティ `us-east-1a`ーゾーンと同じ場所ではない AWS アカウント 場合があります AWS アカウント。

タイプ: 文字列

長さの制約: 最小長は 1 です。最大長 64

パターン: `.+`

必須: いいえ

IpAddress

マウントターゲットの IPv4 アドレス。

タイプ: 文字列

長さの制限: 最小長は 7 です。最大長は 15 です。

パターン: `^[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}$`

必須: いいえ

Ipv6Address

マウントターゲットの IPv6 アドレス。

タイプ: 文字列

長さの制約: 最小長は 3 です。最大長は 39 です。

必須: いいえ

NetworkInterfaceId

Amazon EFS がマウントターゲットを作成したときに作成したネットワーク インターフェイスの ID。

タイプ: 文字列

必須: いいえ

OwnerId

AWS アカウント リソースを所有する ID。

タイプ: 文字列

長さの制限: 最大長は 14 です。

パターン: `^\d{12}|\d{4}-\d{4}-\d{4}$`

必須: いいえ

VpcId

マウントターゲットが設定されている Virtual Private Cloud (VPC) ID。

タイプ: 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 AWS SDKs のいずれかでこの API を使用方法の詳細については、以下を参照してください。

- [AWS SDK for C++](#)

- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

PosixUser

アクセスポイント上の、ユーザー ID、グループ ID、およびセカンダリグループ ID を含む完全な POSIX アイデンティティ。アクセスポイントを使用する NFS クライアントによって実行されるすべてのファイル システムオペレーションに使用されます。

内容

Gid

このアクセスポイントを使用するすべての ファイル システムオペレーションに使用される POSIX グループ ID。

タイプ: 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

このアクセスポイントを使用するすべての ファイル システムオペレーションに使用される POSIX ユーザーID。

型: Long

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

このアクセスポイントを使用するすべての ファイル システムオペレーションに使用されるセカンダリ POSIX グループ ID。

型: longの配列

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 AWS SDKs のいずれかでこの API を使用方法の詳細については、以下を参照してください。

- [AWS SDK for C++](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

ReplicationConfigurationDescription

特定のファイルシステムのレプリケーション設定を記述します。

内容

CreationTime

レプリケーション設定の作成日を記述します。

タイプ: タイムスタンプ

必須: はい

Destinations

デスティネーションオブジェクトの配列。サポートされているデスティネーションオブジェクトは 1 つだけです。

型: [Destination](#) オブジェクトの配列

必須: はい

OriginalSourceFileSystemArn

レプリケーション設定の元のソース EFS ファイルシステムの Amazon リソースネーム (ARN)。

タイプ: 文字列

必須: はい

SourceFileSystemArn

レプリケーション設定内の現在のソースファイルシステムの Amazon リソースネーム (ARN)。

タイプ: 文字列

必須: はい

SourceFileSystemId

レプリケートされたソースの Amazon EFS ファイルシステムの ID。

タイプ: 文字列

長さの制約: 最大長は 128 です。

Pattern: `^(arn:aws[-a-z]*:elasticfilesystem:[0-9a-z-:]+:file-system/fs-[0-9a-f]{8,40}|fs-[0-9a-f]{8,40})$`

必須: はい

SourceFileSystemRegion

AWS リージョン ソース EFS ファイルシステムが配置されている。

タイプ: 文字列

長さの制約: 最小長は 1 です。最大長 64

Pattern: `^[a-z]{2}-((iso[a-z]{0,1}-)|(gov-)){0,1}[a-z]+-{0,1}[0-9]{0,1}$`

必須: はい

SourceFileSystemOwnerId

AWS アカウント ソースファイルシステムが存在する の ID。

タイプ: 文字列

長さの制限: 最大長は 14 です。

パターン: `^(\d{12})|(\d{4}-\d{4}-\d{4})$`

必須: いいえ

以下の資料も参照してください。

言語固有の AWS SDKs のいずれかでこの API を使用方法の詳細については、以下を参照してください。

- [AWS SDK for C++](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

ResourceIdPreference

現在の AWS アカウントのリソースタイプとそのユーザーの ID 設定について説明します AWS リージョン。

内容

ResourceIdType

EFS リソース ID 設定を、LONG_ID(17 文字) またはSHORT_ID(8 文字)のいずれかで識別します。

タイプ: 文字列

有効な値: LONG_ID | SHORT_ID

必須: いいえ

Resources

ID プリファレンス設定が適用される Amazon EFS リソースをFILE_SYSTEMおよびMOUNT_TARGETで識別します。

タイプ: 文字列の配列

有効な値: FILE_SYSTEM | MOUNT_TARGET

必須: いいえ

以下の資料も参照してください。

言語固有の AWS SDKs のいずれかでこの API を使用方法の詳細については、以下を参照してください。

- [AWS SDK for C++](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

RootDirectory

アクセスポイントがアクセスを提供する Amazon EFS ファイルシステム上のディレクトリを指定します。アクセスポイントは、指定されたファイルシステムのパスを、アクセスポイントを使用するアプリケーションにファイルシステムのルートディレクトリとして公開します。アクセスポイントを使用する NFS クライアントは、アクセスポイントの RootDirectory とそのサブディレクトリ内のデータのみアクセスできます。

内容

CreationInfo

(オプション) アクセスポイントの RootDirectory に適用する POSIX ID とアクセス許可を指定します。指定した RootDirectory > Path が存在しない場合、クライアントがアクセスポイントに接続すると EFS は CreationInfo 設定を使用してルートディレクトリを作成します。CreationInfo を指定する場合は、すべてのプロパティに値を指定する必要があります。

Important

CreationInfo を指定せず、指定した RootDirectory > Path が存在しない場合、アクセスポイントを使用してファイルシステムをマウントしようとすると失敗します。

型: [CreationInfo](#) オブジェクト

必須: いいえ

Path

アクセスポイントを使用して、EFS ファイルシステムにアクセスする NFS クライアントにルートディレクトリとして公開する EFS ファイルシステムのパスを指定します。パスには、最大 4 つのサブディレクトリを含めることができます。指定したパスが存在しない場合は、CreationInfo を指定する必要があります。

タイプ: 文字列

長さの制約: 最小長は 1 です。最大長は 100 です。

パターン: `^(\\|\\(?!\\.)+[\\$#<>;`|&?{}^*/\\n]+){1,4})$`

必須: いいえ

以下の資料も参照してください。

言語固有の AWS SDKs のいずれかでこの API を使用方法の詳細については、以下を参照してください。

- [AWS SDK for C++](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

Tag

タグはキーバリューのペアで構成されます。使用許可の文字は、UTF-8で表現できる文字、空白、数字、および次の文字です： + - = . _ : /。

内容

Key

タグキー (文字列)。キーのスタートを aws: にすることはできません。

タイプ: 文字列

長さの制限: 最小長は 1 です。最大長は 128 です。

Pattern: `^(?![aA]{1}[wW]{1}[sS]{1}:)([\p{L}\p{Z}\p{N}_.: /+=\ -@]+)$`

必須: はい

Value

タグキーの値。

タイプ: 文字列

長さの制約: 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 AWS SDKs のいずれかでこの API を使用方法の詳細については、以下を参照してください。

- [AWS SDK for C++](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

ドキュメント履歴

- API バージョン: 2015-02-01
- ドキュメントの最終更新日: 2025 年 6 月 9 日

以下の表に、2018 年 7 月以降の『Amazon Elastic File System ユーザーガイド』の重要な変更点を示します。ドキュメントの更新に関する通知については、RSS フィードにサブスクライブできます。

変更	説明	日付
IPv6 のサポート	インターネットプロトコルバージョン 6 (IPv6) は、Amazon EFS Service APIs とマウントターゲットでサポートされています。Amazon EFS API の詳細については、 「Amazon EFS API」 を参照してください。マウントターゲットの詳細については、 「マウントターゲットの管理」 を参照してください。	2025 年 6 月 9 日
特定のでのエラスティックスループットの増加 AWS リージョン	Elastic スループットを使用するファイルシステムの最大読み取りスループットは、アジアパシフィック (ムンバイ)、アジアパシフィック (ソウル)、アジアパシフィック (シドニー)、欧州 (ロンドン) の 20 GiBps から 60 GiBps に増加しました。詳細については、 「増やすことができる Amazon EFS クォータ」 を参照してください。	2025 年 5 月 2 日

[特定のリージョンでのElastic スループットの増加 AWS リージョン](#)

特定のリージョンで Elastic スループットを使用するファイルシステムの最大読み取りスループットは 60 GiBps です。他のすべてのリージョンでは、Elastic スループットを使用するファイルシステムの読み取りスループット制限が 10 GiBps から 20 GiBps に引き上げられました。詳細については、[「増やすことができる Amazon EFS クォータ」](#)を参照してください。

2025 年 4 月 10 日

[アクセスポイントクォータの増加](#)

単一のファイルシステムが持つことができるアクセスポイントの最大数が 1,000 から 10,000 に増加しました。この制限の引き上げをリクエストすることもできます。詳細については、[「引き上げることができるリソースクォータ」](#)を参照してください。

2025 年 2 月 10 日

[バックアップ率と復元率の向上](#)

バックアップと復元の実行速度が向上しました。詳細については、[「バックアップパフォーマンス」](#)を参照してください。

2025 年 1 月 8 日

[リクエストによる IOPS クォータの増加](#)

Elastic スループットモードを使用してEFS ファイルシステムの IOPS の増加をリクエストできるようになりました。詳細については、[「増やすことができる Amazon EFS クォータ」](#)を参照してください。

2024 年 11 月 26 日

[クロスアカウントレプリケーションのサポート](#)

Amazon EFS はクロスアカウントレプリケーションをサポートしています。詳細については、[「間で EFS ファイルシステムをレプリケートする AWS アカウント」](#)を参照してください。

2024 年 11 月 19 日

[既存の AWS 管理ポリシーを更新しました](#)

ReplicationRead 、ReplicationWrite 、iam:PassRole がAmazonElasticFileSystemFullAccess ポリシーに追加されます。詳細については、[「AmazonElasticFileSystemFullAccess」](#)を参照してください。

2024 年 11 月 7 日

[既存の AWS 管理ポリシーを更新しました](#)

ReplicationRead と ReplicationWrite に追加されましたAmazonElasticFileSystemServiceRolePolicy 。詳細については、[AmazonElasticFileSystemServiceRolePolicy](#)」を参照してください。

2024 年 11 月 7 日

既存の AWS 管理ポリシーを更新しました

既存のAmazonElasticFileSystemReadOnlyAccess ポリシーにReplicationRead 追加されたアクセス許可。詳細については、「[AmazonElasticFileSystemReadOnlyAccess](#)」を参照してください。

2024 年 11 月 7 日

エラスティックスループットの上限の引き上げ

エラスティックスループットの上限が、特定の AWS リージョンで 60 ギビバイト/秒 (GiBps) に、その他のすべてのリージョンで 10 GiBps に引き上げられました。詳細については、「[各 AWS リージョンにおけるすべての接続されたクライアントのデフォルトでのエラスティックスループット合計](#)」を参照してください。

2024 年 10 月 14 日

既存の AWS マネージドポリシーを更新しました

AmazonElasticFileSystemReadOnlyAccess ポリシーステートメントに、オプションの Sid (ステートメント ID) 要素が含まれるようになりました。Sid 値は ElasticFileSystemReadOnlyAccess です。Sid ポリシー要素の詳細については、「[IAM JSON ポリシー要素: Sid](#)」を参照してください。

2024 年 8 月 7 日

[エラスティックスループット の上限の引き上げ](#)

特定の AWS リージョンでエラスティックスループットの上限が引き上げられました。詳細については、「[各 AWS リージョンにおけるすべての接続されたクライアントのデフォルトでのエラスティックスループット合計](#)」を参照してください。

2024 年 7 月 31 日

[マウントターゲットのクォータの引き上げ](#)

仮想プライベートクラウド (VPC) ごとのマウントターゲットの最大数が 400 から 1,400 に引き上げられました。詳細については、「[変更できない Amazon EFS リソースクォータ](#)」を参照してください。

2024 年 5 月 15 日

[Elastic File System での結合されたスループットの上限の引き上げ](#)

エラスティックスループットを使用し、バージョン 2.0 以降の Amazon EFS クライアント (amazon-efs-utils バージョン) または Amazon EFS CSI ドライバー (aws-efs-csi-driver) を使用してマウントされたファイルシステムでは、読み取りと書き込みを合わせた最大スループットは 1,500 MiBps になります。詳細については、「[Amazon EFS パフォーマンス](#)」のパフォーマンスの概要の表を参照してください。

2024 年 4 月 30 日

エラスティックスループット の上限の引き上げ

特定の AWS リージョンでエラスティックスループットの上限が引き上げられました。詳細については、「[各 AWS リージョンにおけるすべての接続されたクライアントのデフォルトでのエラスティックスループット合計](#)」を参照してください。

2024 年 3 月 13 日

IOPS の向上

エラスティックスループットを使用するファイルシステムでは、アクセス頻度の低いデータに対して最大 90,000 の読み取りを実現できます。詳細については、「[パフォーマンスの概要](#)」を参照してください。

2024 年 1 月 22 日

既存の AWS マネージドポリシーを更新しました

アクセス許可 elasticfilesystem:UpdateFilesystemProtection が既存の AmazonElasticFilesystemFullAccess ポリシーに追加され、プリンシパルがファイルシステムの保護を更新できるようになりました。詳細については、「[Amazon EFS updates to AWS managed policies](#)」を参照してください。

2023 年 11 月 27 日

既存のファイルシステムへの レプリケート

ファイルシステムを既存のファイルシステムにレプリケートできるようになったため、フェイルバックを目的としたファイルシステム間の変更の同期が容易になりました。詳細については、「[デスティネーションファイルシステム](#)」を参照してください。

2023 年 11 月 27 日

ファイルシステムの保護が追加されました。

レプリケーションの上書き保護がファイルシステムに追加され、デフォルトで有効になりました。この保護により、ファイルシステムがレプリケーション設定のデスティネーションとして使用されるのを防ぐことができます。詳細については、「[ファイルシステムの保護](#)」を参照してください。

2023 年 11 月 27 日

新しいストレージクラス、 ファイルシステムのタイプ、 ライフサイクルポリシー

Amazon EFS は、EFS アーカイブストレージクラス、ファイルシステムのタイプ、およびアーカイブへの移行ライフサイクルポリシーを提供するようになりました。詳細については、「[ファイルシステムのタイプとストレージクラス](#)」を参照してください。

2023 年 11 月 26 日

IOPS の向上

エラスティックスループットファイルシステムでは、アクセス頻度の低いデータに対して最大 65,000 IOPS の読み取りオペレーションと 50,000 IOPS の書き込みオペレーションがサポートされるようになりました。また、頻繁にアクセスされるデータに対しては 250,000 IOPS の読み込みがサポートされるようになりました。詳細については、「[パフォーマンスの概要](#)」を参照してください。

2023 年 11 月 26 日

ソースファイルシステムからレプリケーション設定を削除する

レプリケーション設定をソースファイルシステムから削除できるようになりました。詳細については、「[Deleting a replication configuration](#)」を参照してください。

2023 年 9 月 19 日

追加の AWS リージョン サポートが追加されました

Amazon EFS がイスラエル (テルアビブ) リージョンの全ユーザー向けに使用可能になりました。

2023 年 8 月 7 日

汎用モードファイルシステムのパフォーマンスが向上しました

Amazon EFS の汎用モードファイルシステムで、1 秒あたり最大 55,000 回の読み取りオペレーションと 25,000 回の書き込みオペレーションがサポートされるようになりました。詳細については、「[Amazon EFS ファイルシステムのクォータ](#)」を参照してください。

2023 年 8 月 3 日

[プロビジョンドスループット の上限の引き上げ](#)

プロビジョニングされたスループット制限が特定の AWS リージョンに対して増加しました。詳細については、「[各接続クライアントの合計デフォルトプロビジョンドスループット AWS リージョン](#)」を参照してください。

2023 年 6 月 21 日

[EFS レプリケーションのリージョンサポートの拡大](#)

EFS レプリケーションが、EFS が利用可能なすべての AWS リージョンで利用可能になりました。詳細については、「[Amazon EFS レプリケーション](#)」を参照してください。

2023 年 4 月 28 日

[エラスティックスループット の上限の引き上げ](#)

特定の AWS リージョンでエラスティックスループットの上限が引き上げられました。詳細については、表「[各接続クライアントの合計デフォルト Elastic スループット AWS リージョン](#)」を参照してください。

2023 年 4 月 17 日

[デフォルトのスループット モードがバーストからエラス ティックに変更](#)

ファイルシステムのデフォルト (かつ推奨) のスループットモードは、バーストではなくエラスティックになりました。詳細は、「[スループットモード](#)」を参照してください。

2023 年 4 月 13 日

[追加の AWS リージョン サポートが追加されました](#)

Amazon EFS が、アジアパシフィック (メルボルン) リージョンですべてのユーザーに対して使用できるようになりました。

2023 年 4 月 12 日

[macOS Ventura のサポートを追加](#)

macOS Ventura で実行されている EC2 Mac インスタンスに Amazon EFS をインストールできるようになりました。詳細については、「[サポートされているディストリビューション](#)」を参照してください。

2023 年 4 月 10 日

[追加の AWS リージョン サポートが追加されました](#)

Amazon EFS が、アジアパシフィック (ハイデラバード) リージョンですべてのユーザーに対して使用できるようになりました。

2023 年 2 月 16 日

[追加の AWS リージョン サポートが追加されました](#)

Amazon EFS が欧州 (スペイン) AWS リージョンですべてのユーザーに対して利用可能になりました。

2023 年 1 月 19 日

[ファイルシステムのアクセスポイントの制限が引き上げられる](#)

1 つのファイルシステムに割り当てることができるアクセスポイントの最大数が 120 から 1,000 に増えました。詳細については、「[リソースクォータ](#)」を参照してください。

2023 年 1 月 17 日

[追加の AWS リージョン サポートが追加されました](#)

Amazon EFS が欧州 (チューリッヒ) のすべてのユーザーに利用可能になりました AWS リージョン。

2022 年 12 月 15 日

[1 日間のライフサイクルポリシーのサポートが追加](#)

IA ライフサイクルへの移行ポリシーに 1 日を選択できるようになりました。詳細については、「[Using Lifecycle policies](#)」を参照してください。

2022 年 11 月 27 日

[読み取りと書き込みのレイテンシーが短縮](#)

1 ゾーンストレージと標準ストレージファイルシステムの両方で、ファイルデータの読み取りと書き込みのレイテンシーが短縮しました。パフォーマンスの詳細については、「[Performance summary](#)」を参照してください。

2022 年 11 月 27 日

[さらにスループットモードが追加](#)

Amazon EFS ファイルシステムのスループットオプションとして、エラスティックスループットモードが追加されました。詳細については、「[エラスティックスループット](#)」を参照してください。

2022 年 11 月 27 日

[追加の AWS リージョン サポートが追加されました](#)

Amazon EFS が、中東 (アラブ首長国連邦) リージョンですべてのユーザーに対して利用可能になりました。

2022 年 10 月 17 日

[EFS レプリケーションに関するサポートが追加されました](#)

Amazon EFS は、EFS レプリケーションがソケットと名前付きパイプ、または FIFOs。

2022 年 9 月 15 日

[接続ごとのファイルロック数の上限が増加](#)

接続ごとのファイルロックの数が 8192 から 65,536 に増えました。詳細については、「[NFS クライアント用のクォータ](#)」を参照してください。

2022 年 5 月 4 日

[ファイルロックを使用するプロセスの制限を解除](#)

Amazon EFS は、1 つのインスタンスで最大 256 のプロセスが同時にファイルロックを使用できるという以前の制限を削除しました。詳細については、「[NFS クライアント用のクォータ](#)」を参照してください。

2022 年 5 月 4 日

[追加の AWS リージョン サポートが追加されました](#)

Amazon EFS が、アジアパシフィック (ジャカルタ) AWS リージョンで使用できるようになりました。

2022 年 1 月 27 日

[EFS レプリケーションに関するサポートが追加されました](#)

EFS レプリケーションを使用して、EFS ファイルシステムのデータとメタデータを、選択した別の EFS ファイルシステムにレプリケート AWS リージョンします。詳細については、「[Amazon EFS レプリケーション](#)」を参照してください。

2022 年 1 月 25 日

[ファイルシステムおよびマウントターゲットリソースは 17 文字のリソース ID 形式を使用します。](#)

新しい Amazon EFS ファイルシステムおよびマウントターゲットリソースに 17 文字の ID が割り当てられるようになりました。詳細については、「[Amazon EFS リソースでの作業](#)」を参照してください。

2021 年 10 月 22 日

[EFS Intelligent-Tiering のサポートが追加されました](#)

EFS Intelligent-Tiering は、EFS ライフサイクル管理を使用してファイルアクセスパターンを監視し、対応する低頻度アクセス (IA) ストレージクラスとの間でファイルを自動的に移行するように設計されています。詳細については、「[EFS Intelligent-Tiering とライフサイクル管理](#)」を参照してください。

2021 年 9 月 2 日

[17 文字のリソース ID 形式のテストのためのサポートが追加されました](#)

Amazon EFS は、2021 年 10 月 1 日にファイルシステムおよびマウントターゲットに対して使用する ID を 8 文字の ID から 17 文字の ID に移行しています。この移行中、オプトインして、1 AWS リージョンベースで 17 文字のリソース IDs の使用を開始できます。詳細については、「[リソース ID](#)」を参照してください。

2021 年 5 月 5 日

[1 ゾーンファイルシステムを別のアベイラビリティゾーンから Amazon EFS マウントヘルパーにマウントするためのサポートが追加されました](#)

EFS マウントヘルパーを使用して、1 ゾーンストレージクラスを使用する Amazon EFS ファイルシステムを別のアベイラビリティゾーンにある EC2 インスタンスにマウントできるようになりました。新しい az オプションを使用して、Amazon EFS ファイルシステムのアベイラビリティゾーンを指定します。詳細については、「[1 ゾーン ストレージクラスを使用したファイルシステムをマウントする](#)」を参照してください。

2021 年 4 月 6 日

[EFS 1 ゾーンストレージクラスのSupport が追加されました](#)

Amazon EFS 1 ゾーンストレージクラスは、AWS リージョンの単一のアベイラビリティゾーン内にデータを冗長的に保存します。EFS 1 ゾーンおよび 1 ゾーン低頻度アクセス (1 ゾーン — IA) ストレージクラスは、EFS 標準および標準 IA ストレージクラスのマルチ AZ 復元を必要としないデータを保存するためのコスト効率の高いオプションです。詳細については、「[EFS ストレージクラスの使用](#)」を参照してください。

2021 年 3 月 9 日

[追加の AWS リージョン サポートが追加されました](#)

Amazon EFS がアジアパシフィック (大阪) AWS リージョンで、すべてのユーザーに対して利用可能になりました。

2021 年 3 月 3 日

[macOS Big Sur を実行する Amazon EC2 mac インスタンスに関するサポートが追加されました](#)

EFS マウントヘルパーまたは NFS マウントコマンドを使用して、macOS Big Sur を実行する EC2 mac インスタンスから Amazon EFS ファイルシステムをマウントできるようになりました。詳細については、「[EFS マウントヘルパーを使用してマウントする](#)または [EFS マウントヘルパーを使用せずにファイルシステムをマウントする](#)」を参照してください。

2021 年 2 月 23 日

[新しい Amazon EFS コンソールが AWS GovCloud \(US\) リージョンで利用可能に](#)

新しい Amazon EFS コンソールが で利用可能になりました AWS GovCloud (US) AWS リージョン。

2021 年 2 月 10 日

[新しい Amazon EFS
CloudWatch メトリクス
MeteredIOBytes の
Support が追加されました](#)

読み取りデータ、書き込みデータ、メタデータオペレーションを含む、各ファイルのシステムオペレーションのバイト数を計測する MeteredIOBytes の使用が可能です。読み取りオペレーションは、他のオペレーションの 3 分の 1 の速度で計測されます。詳細については、「[Amazon EFS の Amazon CloudWatch メトリクス](#)」を参照してください。

2021 年 1 月 28 日

[Amazon EFS はファイルシステムの読み取りスループットを 300% 向上](#)

Amazon EFS ファイルシステムは、読み取りリクエストを他のリクエストの 3 分の 1 の速度で計測するようになりました。

2021 年 1 月 28 日

[新しい Amazon EFS
CloudWatch メトリクス
StorageBytes のSupport
が追加されました](#)

StorageBytes を使用すると、標準および低頻度アクセスストレージクラスに格納されているデータ量など、ファイルシステムのサイズをバイト単位で測定およびモニタリングできます。詳細については、「[Amazon EFS の Amazon CloudWatch メトリクス](#)」を参照してください。

2021 年 1 月 11 日

[AWS Transfer Family を使用して Amazon EFS ファイルシステムにアクセスする](#)

を使用して AWS Transfer Family、Amazon EFS ファイルシステムとの間でファイルを転送できます。詳細については、「[AWS Transfer Family を使用して EFS ファイルシステムのファイルにアクセスする](#)」を参照してください。

2021 年 1 月 6 日

[AWS Systems Manager を使用して Amazon EFS クライアントを管理する \(amazon-efs-utils \)](#)

を使用して AWS Systems Manager、EC2 インスタンスに Amazon EFS クライアント (amazon-efs-utils) を自動的にインストールまたは更新できます。詳細については、[AWS 「Systems Manager を使用して Amazon EFS クライアントを自動的にインストールまたは更新する](#)」を参照してください。

2020 年 9 月 29 日

[暗号化された EFS ファイルシステムの作成を強制する](#)

elasticfilesystem: Encrypted AWS Identity and Access Management (IAM) 条件キーを使用して、ユーザーが保管時に暗号化された Amazon EFS ファイルシステムを作成するように強制できます。詳細については、「[保管時に暗号化された Amazon EFS ファイルシステムの作成を強制する](#)」を参照してください。

2020 年 9 月 16 日

[Amazon EFS クライアントあたりのスループットが 100% 増加しました](#)

EFS では、クライアントあたり最大 500 MB/s のスループットがサポートされるようになりました。これは、以前の制限の 250 MB/秒から 100% 増加しました。詳細については、「[Amazon EFS ファイルシステムのクォータ](#)」を参照してください。

2020 年 7 月 23 日

[Amazon EFS ファイルシステムの毎日の自動バックアップ Support が追加されました](#)

EFS コンソールを使用してファイルシステムを作成するときに、自動日次バックアップがデフォルトで有効になるようになりました。詳細については、「[Amazon EFS AWS Backup での の使用](#)」を参照してください。

2020 年 7 月 16 日

[新しいクイック作成ワークフローにより、Amazon EFS ファイルシステムの作成が簡素化されます](#)

EFS コンソールの [クイック作成] オプションを使用すると、サービス推奨設定を使用して EFS ファイルシステムを 1 つのボタンで作成できます。詳細については、[EFS ファイルシステムを作成する](#)」を参照してください。

2020 年 7 月 16 日

[新しい Amazon EFS コンソールが利用可能になりました](#)

新しい EFS コンソールにより、Amazon EFS の使用が簡単になり、EFS ファイルシステムの管理が簡素化されます。

2020 年 7 月 16 日

[Amazon EFS はファイルシステムの最小スループットを向上](#)

バーストスループットを使用する Amazon EFS ファイルシステムの最小スループットが 1 MiB/秒になりました。詳細は、「[スループットモード](#)」を参照してください。

2020 年 6 月 30 日

[汎用モードファイルシステムのパフォーマンスが向上しました](#)

Amazon EFS の汎用モードファイルシステムで、1 秒あたり最大 35,000 回の読み取り操作がサポートされるようになりました。以前の制限である 7,000 回から 400% 増加したことになります。詳細については、「[Amazon EFS ファイルシステムのクォータ](#)」を参照してください。

2020 年 4 月 1 日

[追加の AWS リージョン サポートが追加されました](#)

Amazon EFS が北京と寧夏ですべてのユーザーが利用できるようになりました AWS リージョン。

2020 年 1 月 22 日

[NFS クライアントの IAM 認可に関するサポートが追加されました](#)

AWS Identity and Access Management (IAM) を使用して、Amazon EFS ファイルシステムへの NFS 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、「[IAM を使用して Amazon EFS AWS への NFS アクセスを制御する](#)」を参照してください。

2020 年 1 月 13 日

EFS アクセスポイントに関するサポートが追加されました

Amazon EFS アクセスポイントは、EFS ファイルシステムへのアプリケーション固有のエントリポイントです。これにより、共有データセットへのアプリケーションアクセスが管理しやすくなります。詳細については、「[Amazon EFS アクセスポイントの使用](#)」を参照してください。

2020 年 1 月 13 日

AWS Backup 部分復元のサポートが追加されました。

復旧ポイントの完全復元に加えて、部分復元を使用して特定のファイルやディレクトリを復元できるようになりました。詳細については、「[Amazon EFS AWS Backup での使用](#)」を参照してください。

2020 年 1 月 13 日

IAM サービスにリンクされたロールに関するサポートが追加されました

Amazon EFS では、IAM に基づいてサービスにリンクされたロールが使用されるようになりました。これにより、必要なアクセス許可が自動的に追加されるため、EFS が設定しやすくなります。詳細については、「[Amazon EFS のサービスリンクロールの使用](#)」を参照してください。

2019 年 12 月 10 日

追加の AWS リージョン サポートが追加されました

Amazon EFS が欧州 (ストックホルム) のすべてのユーザーに利用可能になりました AWS リージョン。

2019 年 11 月 20 日

[追加の AWS リージョン サポートが追加されました](#)

Amazon EFS は、アジアパシフィック (香港) のすべてのユーザーが利用できるようになりました AWS リージョン。

2019 年 11 月 20 日

[追加の AWS リージョン サポートが追加されました](#)

Amazon EFS が南米 (サンパウロ) のすべてのユーザーが利用できるようになりました AWS リージョン。

2019 年 11 月 20 日

[追加の AWS リージョン サポートが追加されました](#)

Amazon EFS が中東 (バーレーン) のすべてのユーザーが利用できるようになりました AWS リージョン。

2019 年 11 月 20 日

[新しい 7 日間のライフサイクル管理ポリシーが追加されました](#)

7 日経ったらデータをコスト効率の良い低頻度アクセスストレージクラスに移動するポリシーがライフサイクル管理にさらに追加されました。詳細については、「[EFS のライフサイクル管理](#)」を参照してください。

2019 年 11 月 6 日

[インターフェイス VPC エンドポイントのサポートが追加されました](#)

仮想プライベートクラウドと Amazon EFS の間にプライベート接続を確立して、EFS API を呼び出すことができます。詳細については、「[VPC エンドポイントの使用](#)」を参照してください。

2019 年 10 月 22 日

[新しい EC2 インスタンスの起動時に EFS ファイルシステムをマウントします。](#)

EC2 インスタンス起動ウィザードで、起動時に EFS ファイルシステムにマウントするよう、新しい Amazon EC2 インスタンスを設定できるようになりました。詳細については、「[ステップ 2](#)」を参照してください。[EC2 リソースを作成し、EC2 インスタンスを起動する](#)

2019 年 10 月 17 日

[Support for Service Quotas を追加](#)

Service Quotas コンソールですべての Amazon EFS 制限を表示できるようになりました。詳細については、「[Amazon EFS の制限](#)」を参照してください。

2019 年 9 月 10 日

[新しいライフサイクル管理ポリシーが追加されました](#)

ライフサイクル管理を使用するときに、4 つのライフサイクルポリシーのうち 1 つを選択して、コスト効率の良い低頻度アクセスストレージクラスにファイルを移行するタイミングを定義できるようになりました。詳細については、「[EFS のライフサイクル管理](#)」を参照してください。

2019 年 7 月 9 日

[EFS ライフサイクル管理は、すべての EFS ファイルシステムで利用できるようになりました。](#)

EFS ライフサイクル管理機能が、すべての EFS ファイルシステムで利用できるようになりました。ファイルシステムの作成時期に基づく以前の制限は削除されました。詳細については、「[EFS のライフサイクル管理](#)」を参照してください。

2019 年 7 月 9 日

[追加の AWS リージョン サポートが追加されました](#)

Amazon EFS が欧州 (パリ) のすべてのユーザーが利用できるようになりました AWS リージョン。

2019 年 6 月 12 日

[追加の AWS リージョン サポートが追加されました](#)

Amazon EFS は、アジアパシフィック (ムンバイ) のすべてのユーザーが利用できるようになりました AWS リージョン。

2019 年 6 月 5 日

[追加の AWS リージョン サポートが追加されました](#)

Amazon EFS は、カナダ (中部) のすべてのユーザーが利用できるようになりました AWS リージョン。

2019 年 5 月 1 日

[API の更新: タグが CreateFileSystem オペレーションペイロードの一部になりました](#)

AWS API および CLI CreateFileSystem オペレーションを使用して Amazon EFS ファイルシステムを作成するときにタグを含めることができるようになりました。詳細については、[CreateFileSystem](#)「」および「[CLI AWS を使用したファイルシステムの作成](#)」を参照してください。

2019 年 2 月 19 日

[新機能: EFS 低頻度アクセス ストレージクラスと EFS ライ フサイクル管理](#)

Amazon EFS 低頻度アクセスは、アクセスが頻繁ではないファイル用のコスト最適化ストレージクラスです。EFS ライフサイクル管理は、ファイルを標準ストレージから低頻度アクセスストレージに自動的に移行します。詳細については、「[EFS ストレージクラス](#)」を参照してください。

2019 年 2 月 13 日

[追加の AWS リージョン サ ポートが追加されました](#)

Amazon EFS が欧州 (ロンドン) のすべてのユーザーに利用可能になりました AWS リージョン。

2019 年 1 月 23 日

[AWS Backup Amazon EFS と のサービス統合](#)

Amazon EFS ファイルシステムは AWS Backup、クラウドおよびオンプレミス AWS のサービス間でデータをバックアップするためのフルマネージド型の一元化された自動バックアップサービスであるを使用してバックアップできます。詳細については、「[AWS Backup および Amazon EFS](#)」を参照してください。

2019 年 1 月 16 日

[オンプレミスストレージシステムへのトランジットゲートウェイ接続のサポートが追加されました。](#)

Amazon EFS ファイルシステムは、オンプレミスストレージシステムへの Transit Gateway 接続を使用してアクセスできるようになりました。詳細については、「[別のアカウントまたは VPC からマウントする](#)」および「[ウォークスルー: 別の VPC からファイルシステムをマウントする](#)」を参照してください。

2018 年 12 月 6 日

[EFS ファイル同期が新しい AWS DataSync サービスの一部になりました。](#)

AWS DataSync は、オンプレミスストレージシステムとストレージサービス間の大量のデータの同期を簡素化するマネージドデータ転送 AWS サービスです。詳細については、「[を使用してオンプレミスファイルシステムから Amazon EFS にファイルを転送する AWS DataSync](#)」を参照してください。

2018 年 11 月 26 日

[VPN とリージョン間 VPC ピアリング接続のサポートが追加されました](#)

Amazon EFS が、VPN 接続とリージョン間 VPC ピアリング接続経由でアクセス可能になりました。詳細については、「[を使用してオンプレミスファイルシステムから Amazon EFS にファイルを転送する AWS DataSync](#)」を参照してください。

2018 年 10 月 23 日

[VPN とリージョン間 VPC ピアリング接続のサポートが追加されました](#)

Amazon EFS ファイルシステムは、VPN 接続やリージョン間 VPC ピアリング接続でアクセスできるようになりました。詳細については、「[別のアカウントまたは VPC からマウントする](#)」および「[Amazon EFS が Direct Connect および VPN と動作する仕組み](#)」を参照してください。

2018 年 10 月 23 日

[追加の AWS リージョン サポートが追加されました](#)

Amazon EFS がアジアパシフィック (シンガポール) AWS リージョンですべてのユーザーに対して利用可能になりました。

2018 年 7 月 13 日

[プロビジョニングされたスループットモードの紹介](#)

新しいプロビジョニングされたスループットモードを使用して、新規または既存のファイルシステムのスループットをプロビジョニングできるようになりました。詳細は、「[スループットモード](#)」を参照してください。

2018 年 7 月 12 日

[追加の AWS リージョン サポートが追加されました](#)

Amazon EFS がアジアパシフィック (東京) AWS リージョンですべてのユーザーに対して利用可能になりました。

2018 年 7 月 11 日

次の表に、2018 年 6 月以前の Amazon Elastic File System ユーザーガイドの重要な変更点を示します。

変更	説明	変更日
追加の AWS リージョン サポートが追加されました	Amazon EFS が、アジアパシフィック (ソウル) AWS リージョンですべてのユーザーに対して使用できるようになりました。	2018 年 5 月 30 日
CloudWatch Metric Math のサポートが追加されました	Metric Math により複数の CloudWatch メトリクスをクエリし、数式を使用して、これらのメトリクスに基づく新しい時系列を作成できます。詳細については、「 CloudWatch メトリクスでの Metric Math の使用 」を参照してください。	2018 年 4 月 4 日
amazon-efs-utils の一連のオープンソースツールが追加、および伝送中の暗号化が追加	amazon-efs-utils ツールは、一連のオープンソースの実行可能ファイルで、マウントなどの Amazon EFS の利用を簡略化します。amazon-efs-utils は追加コストなしで使用でき、GitHub からこれらのツールをダウンロードできます。詳細については、「 Amazon EFS クライアントの手動インストール 」を参照してください。 また、このリリースでは、Amazon EFS により Transport Layer Security (TLS) トンネリングを使用して、転送時の暗号化がサポートされるようになりました。詳細については、「 Amazon EFS でのデータの暗号化 」を参照してください。	2018 年 4 月 4 日
あたりの更新されたファイルシステム制限 AWS リージョン	Amazon EFS では、すべての AWS リージョンで、すべてのアカウントに対して、ファイルシステムの数の制限を引き上げました。詳細については、「 変更できない Amazon EFS リソースクォータ 」を参照してください。	2018 年 3 月 15 日
追加の AWS リージョン サポートが追加されました	Amazon EFS は、米国西部 (北カリフォルニア) のすべてのユーザーが利用できるようになりました AWS リージョン。	2018 年 3 月 14 日

変更	説明	変更日
保管時のデータ暗号化	Amazon EFS は、保管中のデータの暗号化をサポートするようになりました。詳細については、「 Amazon EFS でのデータの暗号化 」を参照してください。	2017 年 8 月 14 日
その他のリージョンのサポートが追加されました	Amazon EFS が欧州 (フランクフルト) リージョンですべてのユーザーに対して利用可能になりました。	2017 年 7 月 20 日
ドメインネームシステム (DNS) を使用したファイルシステム名	Amazon EFS ではファイルシステムの DNS 名をサポートするようになりました。ファイルシステムの DNS 名は、接続する Amazon EC2 インスタンスの Availability Zone 内のマウントターゲットの IP アドレスを自動的に解決します。詳細については、「 DNS 名を使用して Amazon EC2 にマウントする 」を参照してください。	2016 年 12 月 20 日
ファイルシステムのタグサポートの強化	Amazon EFS では、ファイルシステムあたり 50 のタグをサポートするようになりました。Amazon EFS のタグの詳細については、「 EFS リソースのタグ付け 」を参照してください。	2016 年 8 月 29 日
一般提供	Amazon EFS は米国東部 (バージニア北部)、米国西部 (オレゴン、欧州 (アイルランド) の各リージョンで一般のすべてのユーザーが利用できるようになりました。	2016 年 6 月 28 日
ファイルシステムの制限の引き上げ	AWS リージョン のアカウントごとに作成できる Amazon EFS ファイルシステムの数 は 5 から 10 に増加しました。	2015 年 8 月 21 日
「使用開始」の演習が更新されました	「使用開始」の演習は、使用開始のプロセスを簡略化するために更新されました。	2015 年 8 月 17 日
新規ガイド	これは Amazon Elastic File System ユーザーガイドの最初のリリースです。	2015 年 5 月 26 日

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