
AWS Transfer Family

User Guide



AWS Transfer Family: User Guide

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

Amazon's trademarks and trade dress may not be used in connection with any product or service that is not Amazon's, in any manner that is likely to cause confusion among customers, or in any manner that disparages or discredits Amazon. All other trademarks not owned by Amazon are the property of their respective owners, who may or may not be affiliated with, connected to, or sponsored by Amazon.

Table of Contents

What is AWS Transfer Family?	1
How AWS Transfer Family works	3
Setting up	5
Regions, endpoints and quotas	5
Sign up for AWS	5
Create an Amazon S3 bucket	5
Amazon S3 access points	6
Create an Amazon EFS file system	6
Amazon EFS file ownership	7
Set up Amazon EFS users for Transfer Family	7
Supported Amazon EFS commands	8
Create an IAM role and policy	8
Example read/write access policy	10
Example session policy	11
Getting started tutorial	15
Prerequisites	15
Sign in to the console	15
Create an SFTP-enabled server	15
Add a service managed user	16
Transfer a file using a client	17
Use Cyberduck	17
Use OpenSSH	18
Creating a server	19
Identity providers	19
Create an SFTP-enabled server	20
Create an FTPS-enabled server	25
Create an FTP-enabled server	30
Create a server in a VPC	35
Create a server endpoint accessible in your VPC only	36
Create an internet-facing endpoint	37
Change the endpoint type	40
Discontinuing the use of VPC_ENDPOINT	42
Updating the server endpoint type to VPC	42
Working with custom hostnames	47
Use Amazon Route 53 as your DNS provider	47
Use other DNS providers	48
Custom hostnames for non-console created servers	48
Working with security policies	49
Cryptographic algorithms	49
TransferSecurityPolicy-2020-06	51
TransferSecurityPolicy-2018-11	52
TransferSecurityPolicy-FIPS-2020-06	53
Managing users	54
Service-managed users	54
Adding Amazon S3 service-managed users	55
Adding Amazon EFS service-managed users	56
Service-managed users	57
Directory services users	60
Using AWS Directory Service for Microsoft Active Directory	60
Using Azure Active Directory	65
Custom identity provider users	69
Lambda as an identity provider	70
Lambda resource-based policy	71
Default Lambda function	71

Using Amazon API Gateway to integrate your identity provider	74
Custom identity provider tutorial	81
Step 1: Create a CloudFormation stack	82
Step 2: Check the API Gateway method configuration for your server and create it.	82
Step 3: Create the Transfer Family server	83
Step 4: Test that your user can connect to the server	83
Step 5: Test the SFTP connection and file transfer	84
Step 6: Limit access to the bucket	84
Update Lambda if using Amazon EFS	85
Using logical directories	87
Rules for using logical directories	87
Implementing logical directories and chroot	87
Chroot	88
Virtual directory structure	88
Transferring files using a client	90
Use OpenSSH	18
Use WinSCP	92
Use Cyberduck	17
Use FileZilla	93
Use a Perl client	94
Post upload processing	94
S3 object metadata	94
S3 event notifications	95
Managing workflows	96
Create a workflow	96
Configure and execute a workflow	97
View workflow details	98
Create a custom workflow step	100
Example Lambda function for a custom workflow step	100
Example custom step	101
Create workflow procedure	102
Exception handling for a workflow	104
Choose pre-defined steps	104
CloudWatch logging for a workflow	105
Construct an execution role for workflows	106
Restrictions and limits	106
Managing servers	108
View a list of servers	108
View server details	108
Edit server details	109
Edit the file transfer protocols	111
Edit the server identity provider	112
Edit the server endpoint	113
Edit Amazon CloudWatch logging	113
Edit the security policy	114
Change the host key for your SFTP-enabled server	114
Put your server online or offline	115
Monitor usage within console	116
Delete a server	117
Managing access controls	118
Creating an S3 bucket access policy	118
Creating a session policy	119
Preventing users from creating a directory in an S3 bucket	121
Monitoring usage	122
Enabling CloudTrail logging	122
Logging S3 API calls to S3 access logs	122
Logging activity with CloudWatch	122

Using CloudWatch metrics	123
Transfer Family dimensions	124
Security	125
Data protection	125
Data encryption	126
Key management	127
Identity and access management	130
Audience	130
Authenticating with identities	130
Managing access using policies	132
How AWS Transfer Family works with IAM	134
Identity-based policy examples	137
Tag-based policy examples	139
Troubleshooting	141
Logging and monitoring	143
AWS Transfer Family information in CloudTrail	143
Understanding AWS Transfer Family log file entries	144
Compliance validation	145
Resilience	145
Infrastructure security	145
Web application firewall	146
AWS managed policies	147
AWSTransferLoggingAccess	147
AWSTransferConsoleFullAccess	148
AWSTransferFullAccess	149
AWSTransferReadOnlyAccess	150
Policy updates	150
Troubleshooting	152
Troubleshoot Amazon EFS Service Managed Users	152
Troubleshoot Amazon API Gateway Issues	152
Too many authentication failures	152
Connection closed	153
Troubleshoot policies for encrypted Amazon S3 buckets	154
Troubleshoot too many authentication failures	154
Troubleshoot workflow-related errors using CloudWatch	154
Troubleshoot workflow copy error	155
Troubleshoot missing POSIX profile	156
Troubleshoot test identity provider	157
Troubleshoot Amazon S3 file upload errors	157
API reference	158
Welcome	158
Actions	159
CreateAccess	161
CreateServer	166
CreateUser	173
CreateWorkflow	179
DeleteAccess	183
DeleteServer	185
DeleteSshPublicKey	187
DeleteUser	190
DeleteWorkflow	192
DescribeAccess	194
DescribeExecution	197
DescribeSecurityPolicy	200
DescribeServer	202
DescribeUser	206
DescribeWorkflow	210

ImportSshPublicKey	213
ListAccesses	217
ListExecutions	220
ListSecurityPolicies	224
ListServers	226
ListTagsForResource	229
ListUsers	232
ListWorkflows	236
SendWorkflowStepState	238
StartServer	241
StopServer	243
TagResource	246
TestIdentityProvider	249
UntagResource	254
UpdateAccess	257
UpdateServer	262
UpdateUser	268
Data Types	272
CopyStepDetails	274
CustomStepDetails	275
DeleteStepDetails	276
DescribedAccess	277
DescribedExecution	280
DescribedSecurityPolicy	282
DescribedServer	284
DescribedUser	288
DescribedWorkflow	291
EfsFileLocation	293
EndpointDetails	294
ExecutionError	296
ExecutionResults	297
ExecutionStepResult	298
FileLocation	299
HomeDirectoryMapEntry	300
IdentityProviderDetails	301
InputFileLocation	303
ListedAccess	304
ListedExecution	306
ListedServer	307
ListedUser	310
ListedWorkflow	312
LoggingConfiguration	313
PosixProfile	314
ProtocolDetails	315
S3FileLocation	317
S3InputFileLocation	319
S3Tag	320
ServiceMetadata	321
SshPublicKey	322
Tag	323
TagStepDetails	324
UserDetails	325
WorkflowDetail	326
WorkflowDetails	327
WorkflowStep	328
Common Parameters	329
Common Errors	330

Document history	333
AWS glossary	337

What is AWS Transfer Family?

AWS Transfer Family is a secure transfer service that enables you to transfer files into and out of AWS storage services.

AWS Transfer Family supports transferring data from or to the following AWS storage services.

- Amazon Simple Storage Service (Amazon S3) storage. For information about Amazon S3, see [Getting started with Amazon Simple Storage Service](#).
- Amazon Elastic File System (Amazon EFS) Network File System (NFS) file system. For information about Amazon EFS, see [What Is Amazon Elastic File System?](#).

AWS Transfer Family supports transferring data over the following protocols:

- Secure Shell (SSH) File Transfer Protocol (SFTP)
- File Transfer Protocol Secure (FTPS)
- File Transfer Protocol (FTP)

Note

For FTP and FTPS data connections, the port range that Transfer Family uses to establish the data channel is 8192–8200.

File transfer protocols are used in data exchange workflows across different industries such as financial services, healthcare, advertising, and retail, among others. Transfer Family simplifies the migration of file transfer workflows to AWS.

Common use cases for Transfer Family with Amazon S3 are the following:

- Data lakes in AWS for uploads from third parties such as vendors and partners.
- Subscription-based data distribution with your customers.
- Internal transfers within your organization.

The following are some common use cases for Transfer Family with Amazon EFS:

- Data distribution
- Supply chain
- Content management
- Web serving applications

With Transfer Family, you get access to a file transfer protocol-enabled server in AWS without the need to run any server infrastructure. You can use this service to migrate your file transfer-based workflows to AWS while maintaining your end users' clients and configurations as is. You first associate your hostname with the server endpoint, then add your users and provision them with the right level of access. After you do this, your users' transfer requests are serviced directly out of your Transfer Family server endpoint.

Transfer Family provides the following benefits:

- A fully managed service that scales in real time to meet your needs.
- You don't need to modify your applications or run any file transfer protocol infrastructure.

- With your data in durable Amazon S3 storage, you can use native AWS services for processing, analytics, reporting, auditing, and archival functions.
- With Amazon EFS as your data store, you get a fully managed elastic file system for use with AWS Cloud services and on-premises resources. Amazon EFS is built to scale on demand to petabytes without disrupting applications, growing and shrinking automatically as you add and remove files. This helps eliminate the need to provision and manage capacity to accommodate growth.
- A fully managed, serverless File Transfer Workflow service that makes it easy to set up, run, automate, and monitor processing of files uploaded using AWS Transfer Family.
- There are no upfront costs, and you pay only for the use of the service.

In the following sections, you can find a description of the different features of Transfer Family, a getting started tutorial, detailed instructions on how to set up the different protocol enabled servers, how to use different types of identity providers, and the service's API reference.

To get started with Transfer Family, see the following:

- [How AWS Transfer Family works \(p. 3\)](#)
- [Setting up \(p. 5\)](#)
- [Tutorial: Getting started with AWS Transfer Family \(p. 15\)](#)

How AWS Transfer Family works

AWS Transfer Family is a fully managed AWS service that you can use to transfer files into and out of Amazon Simple Storage Service (Amazon S3) storage or Amazon Elastic File System (Amazon EFS) file systems over the following protocols:

- Secure Shell (SSH) File Transfer Protocol (SFTP)
- File Transfer Protocol Secure (FTPS)
- File Transfer Protocol (FTP)

AWS Transfer Family supports up to 3 Availability Zones and is backed by an auto scaling, redundant fleet for your connection and transfer requests. For an example on how to build for higher redundancy and minimize network latency by using Latency-based routing, see [Minimize network latency with your AWS Transfer for SFTP servers](#).

Transfer Family Managed File Transfer Workflows (MFTW) is a fully managed, serverless File Transfer Workflow service that makes it easy to set up, run, automate, and monitor processing of files uploaded using AWS Transfer Family. Customers can use MFTW to automate various processing steps such as copying, tagging, scanning, filtering, compressing/decompressing, and encrypting/decrypting the data that is transferred using Transfer Family. This provides end to end visibility for tracking and auditability. For more details, see [Managing workflows for post-upload processing \(p. 96\)](#).

You can get started with AWS Transfer Family by creating a file transfer protocol-enabled server and then assigning users to use the server. To service your AWS Transfer Family users' transfer requests, you create an AWS Identity and Access Management (IAM) role to access your Amazon S3 bucket or Amazon Elastic File System.

To use AWS Transfer Family, you take the following high-level steps:

1. Create an Amazon S3 bucket or Amazon EFS file system.

For information about using Amazon S3, see [Create an Amazon S3 bucket \(p. 5\)](#). For information about using Amazon Elastic File System, see [Create an Amazon EFS file system \(p. 6\)](#).

2. Create an IAM role that contains two IAM policies:

- An IAM policy that includes the permissions to enable AWS Transfer Family to access your Amazon S3 bucket or Amazon EFS file system. This IAM policy determines what level of access you provide your AWS Transfer Family users.
- An IAM policy to establish a trust relationship with AWS Transfer Family.

For more information about creating IAM policies, see [Managing access controls \(p. 118\)](#).

3. (Optional) If you have your own registered domain, associate your registered domain with the server.

You can route file transfer protocol traffic to your server endpoint from a domain, such as `example.com`, or from a subdomain, such as `ftps.accounting.example.com`. For more information, see [Working with custom hostnames \(p. 47\)](#).

4. Create a Transfer Family server and specify the identity provider type used by the service to authenticate your users.

For more information about creating Transfer Family servers, see [Creating a server \(p. 19\)](#). For more information about identity provider types, see [Working with custom identity providers \(p. 69\)](#).

5. If you are working with a server with a service-managed identity provider, as opposed to a custom identity provider, add one or more users.

6. Open a file transfer protocol client and configure the connection to use the endpoint hostname for the server that you want to use. You can get this hostname from the AWS Transfer Family console.

AWS Transfer Family supports any standard file transfer protocol client. Some commonly used clients are the following:

- [OpenSSH](#) – A Macintosh and Linux command line utility.
- [WinSCP](#) – A Windows-only graphical client.
- [Cyberduck](#) – A Linux, Macintosh, and Microsoft Windows graphical client.
- [FileZilla](#) – A Linux, Macintosh, and Windows graphical client.

Setting up

The following sections describe the prerequisites required to use the AWS Transfer Family service. At a minimum, you need to create an Amazon Simple Storage Service (Amazon S3) bucket and provide access to that bucket through an AWS Identity and Access Management (IAM) role. Your role also needs to establish a trust relationship. This trust relationship allows Transfer Family to assume the IAM role to access your bucket so that it can service your users' file transfer requests.

Topics

- [Supported AWS Regions, endpoints and quotas \(p. 5\)](#)
- [Sign up for AWS \(p. 5\)](#)
- [Create an Amazon S3 bucket \(p. 5\)](#)
- [Create an Amazon EFS file system \(p. 6\)](#)
- [Create an IAM role and policy \(p. 8\)](#)

Supported AWS Regions, endpoints and quotas

For information about supported AWS Regions, endpoints, and service quotas, see [AWS Transfer Family endpoints and quotas](#) in the *Amazon Web Services General Reference*.

Sign up for AWS

When you sign up for Amazon Web Services (AWS), your AWS account is automatically signed up for all services in AWS, including AWS Transfer Family. You are charged only for the services that you use.

If you have an AWS account already, skip to the next task. If you don't have an AWS account, use the following procedure to create one.

To create an AWS account

1. Open <https://portal.aws.amazon.com/billing/signup>.
2. Follow the online instructions.

Part of the sign-up procedure involves receiving a phone call and entering a verification code using the phone keypad.

For information about pricing and to use AWS Pricing Calculator to get an estimate of the cost to use Transfer Family, see [AWS Transfer Family pricing](#).

For information about AWS Region availability, see the [AWS Transfer Family endpoints and quotas](#) in the *Amazon Web Services General Reference*.

Create an Amazon S3 bucket

AWS Transfer Family accesses your Amazon S3 bucket to service your users' transfer requests, so you need to provide an Amazon S3 bucket as part of setting up your file transfer protocol-enabled server. You can use an existing bucket, or you can create a new one.

Note

You don't have to use a server and Amazon S3 bucket that are in the same AWS Region, but we recommend this as a best practice.

When you set up your users, you assign them each an IAM role. This role determines the level of access that they have to your Amazon S3 bucket.

For information on creating a new bucket, see [How do I create an S3 bucket?](#) in the *Amazon Simple Storage Service User Guide*.

Note

You can use Amazon S3 Object Lock to prevent objects from being overwritten for a fixed amount of time or indefinitely. This works the same way with Transfer Family as with other services. If an object exists and is protected, writing to that file or deleting it is not allowed.

For more details on Amazon S3 Object Lock, see [Using Amazon S3 Object Lock](#) in the *Amazon Simple Storage Service User Guide*.

Amazon S3 access points

AWS Transfer Family supports [Amazon S3 Access Points](#), a feature of Amazon S3 that allows you to easily manage granular access to shared data sets. You can use S3 Access Point aliases anywhere you use an S3 bucket name. You can create hundreds of access points in Amazon S3 for users who have different permissions to access shared data in an Amazon S3 bucket.

For example, you can use access points to allow three different teams to have access to the same shared dataset where one team can read data from S3, a second team can write data to S3, and the third team can read, write, and delete data from S3. To implement a granular access control as mentioned above, you can create an S3 access point that contains a policy that gives asymmetrical access to different teams. You can use S3 access points with your Transfer Family server to achieve a fine-grained access control, without creating a complex S3 bucket policy that spans hundreds of use cases. To learn more about how to use S3 access points with a Transfer Family server, refer to the [Enhance data access control with AWS Transfer Family and Amazon S3](#) blog post.

Create an Amazon EFS file system

AWS Transfer Family accesses Amazon Elastic File System (Amazon EFS) to service your users' transfer requests. So you must provide an Amazon EFS file system as part of setting up your file transfer protocol-enabled server. You can use an existing file system, or you can create a new one.

The following sections in the *Amazon Elastic File System User Guide* provide more information.

- For information about creating a new Amazon EFS file system, see [Getting started with Amazon Elastic File System](#) in the *Amazon Elastic File System User Guide*.
- For prerequisites to use Amazon EFS with Transfer Family, see [Prerequisites for using AWS Transfer Family with Amazon EFS](#).
- To configure your Amazon EFS to work with AWS Transfer Family, see [Configuring your Amazon EFS to work with AWS Transfer Family](#).
- To set file and directory permissions, see [Setting file and directory permissions for AWS Transfer Family users](#).

Note

When you use a Transfer Family server and an Amazon EFS file system, the server and the file system must be in the same AWS Region.

The server and the file system don't need to be in the same account. If the server and file system are not in the same account, the file system policy must give explicit permission to the user role. For information

about using Amazon EFS, see [Using AWS Transfer Family to access files in your Amazon EFS file system](#) in the *Amazon Elastic File System User Guide*.

For information about how to set up multiple accounts, see [Managing the AWS accounts in your organization](#) in the *AWS Organizations User Guide*.

When you set up your users, you assign them each an IAM role. This role determines the level of access that they have to your Amazon EFS file system.

Amazon EFS file ownership

Amazon EFS uses the Portable Operating System Interface (POSIX) file permission model to represent file ownership.

In POSIX, users in the system are categorized into three distinct permission classes: When you allow a user to access files stored in an Amazon EFS file system using AWS Transfer Family, you must assign them a "POSIX profile." This profile is used to determine their access to files and directories in the Amazon EFS file system.

- User (u): Owner of the file or directory. Usually, the creator of a file or directory is also the owner.
- Group (g): Set of users that need identical access to files and directories that they share.
- Others (o): All other users that have access to the system except for the owner and group members. This permission class is also referred to as "Public."

In the POSIX permission model, every file system object (files, directories, symbolic links, named pipes, and sockets) is associated with the previously mentioned three sets of permissions. Amazon EFS objects have a Unix-style mode associated with them. This mode value defines the permissions for performing actions on that object.

Additionally, on Unix-style systems, users and groups are mapped to numeric identifiers, which Amazon EFS uses to represent file ownership. For Amazon EFS, objects are owned by a single owner and a single group. Amazon EFS uses the mapped numeric IDs to check permissions when a user attempts to access a file system object.

Set up Amazon EFS users for Transfer Family

Before you set your Amazon EFS users, you can do either of the following:

- You can create users and set up their home folders in Amazon EFS. See [Configure Transfer Family users on Amazon EFS \(p. 7\)](#) for details.
- If you are comfortable adding a root user, you can [Create an Amazon EFS root user \(p. 8\)](#).

Configure Transfer Family users on Amazon EFS

Transfer Family maps the users to the UID/GID and directories you specify. If the UID/GID/directories do not already exist in EFS, then you should create them before assigning them in Transfer to a user. The details for creating Amazon EFS users is described in [Working with users, groups, and permissions at the Network File System \(NFS\) Level](#) in the *Amazon Elastic File System User Guide*.

Steps to set up Amazon EFS users in Transfer Family

1. Map the EFS UID and GID for your user in Transfer Family using the `PosixProfile` fields.
2. If you want the user to start in a specific folder upon login, you can specify the EFS directory under the `HomeDirectory` field.

You can automate the process, by using a CloudWatch rule and Lambda function. For an example Lambda function that interacts with EFS, see [Using Amazon EFS for AWS Lambda in your serverless applications](#).

Create an Amazon EFS root user

If your organization is comfortable for you to enable root user access via SFTP/FTPS for the configuration of your users, you can create a user who's UID and GID are 0 (root user), then use that root user to create folders and assign POSIX ID owners for rest of the users. The advantage of this option is that there is no need to mount the Amazon EFS file system.

Perform the steps described in [Adding Amazon EFS service-managed users \(p. 56\)](#), and for both the User ID and Group ID, enter 0 (zero).

Supported Amazon EFS commands

The following commands are supported for Amazon EFS for AWS Transfer Family

- `cd`
- `ls/dir`
- `pwd`
- `put`
- `get`
- `rename`
- `chown`: Only root (that is, users with uid=0) can change ownership and permissions of files and directories.
- `chmod`: Only root can change ownership and permissions of files and directories.
- `chgrp`: Supported either for root or for the file's owner who can only change a file's group to be one of their secondary groups.
- `ln -s/symlink`
- `mkdir`
- `rm/delete`
- `rmdir`
- `chmtime`

Create an IAM role and policy

When you create a user, you make a number of decisions about user access. These decisions include which Amazon S3 buckets or Amazon EFS file systems that the user can access, what portions of each Amazon S3 bucket and which files in the file system are accessible, and what permissions the user has (for example, PUT or GET).

To set access, you create an identity-based AWS Identity and Access Management (IAM) policy and role that provide that access information. As part of this process, you provide access for your user to the Amazon S3 bucket or Amazon EFS file system that is the target or source for file operations. To do this, take the following high-level steps, described in detail later:

1. Create an IAM policy for AWS Transfer Family.
2. Create an IAM role and attach the new IAM policy. See the following example policies:
 - [Example read/write access policy \(p. 10\)](#)
 - [Example session policy \(p. 11\)](#)

For information about session policies, see [Session policies](#) in the *IAM User Guide*.

3. Establish a trust relationship between AWS Transfer Family and the IAM role.

The following procedures describe how to create an IAM policy and role.

To create an IAM policy for AWS Transfer Family

1. Open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Policies**, and then choose **Create policy**.
On the **Create policy** page, select **Choose a service**.
3. Choose **Transfer** from the list of services.
4. On the **Create Policy** page, choose the **JSON** tab.
5. In the editor that appears, replace the contents of the editor with the IAM policy that you want attach to the IAM role.

You can grant read/write access or restrict users to their home directory. For more information, see the following examples:

- [Example read/write access policy \(p. 10\)](#)
 - [Example session policy \(p. 11\)](#)
6. Choose **Review policy** and provide a name and description for your policy, and then choose **Create policy**.

Next, you create an IAM role and attach the new IAM policy to it.

To create an IAM role for AWS Transfer Family

1. In the navigation pane, choose **Roles**, and then choose **Create role**.
On the **Create role** page, make sure that **AWS service** is chosen.
2. Choose **Transfer** from the service list, and then choose **Next: Permissions**. This establishes a trust relationship between AWS Transfer Family and AWS.
3. In the **Attach permissions policies** section, locate and choose the policy that you just created, and choose **Next: Tags**.
4. (Optional) Enter a key and value for a tag, and choose **Next: Review**.
5. On the **Review** page, enter a name and description for your new role, and then choose **Create role**.

Next, you establish a trust relationship between AWS Transfer Family and AWS.

To establish a trust relationship

1. In the IAM console, choose the role that you just created.
2. On the **Summary** page, choose **Trust relationships**, and then choose **Edit trust relationship**.
3. In the **Edit Trust Relationship** editor, make sure **service** is "transfer.amazonaws.com". The access policy is shown following.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Principal": {
```

```
        "Service": "transfer.amazonaws.com"
    },
    "Action": "sts:AssumeRole"
}
}
```

4. Choose **Update Trust Policy** to update the access policy.

You have now created an IAM role that allows AWS Transfer Family to call AWS services on your behalf. You attached to the role the IAM policy that you created to give access to your user. In the [Tutorial: Getting started with AWS Transfer Family \(p. 15\)](#) section, this role and policy are assigned to your user or users.

Optionally, you can create a session policy that limits users' access to their home directories only, as described earlier in this topic. For more information about session policies, see [Example session policy \(p. 11\)](#).

For more general information about IAM roles, see [Creating a role to delegate permissions to an AWS service](#) in the *IAM User Guide*.

To learn more about identity-based policies for Amazon S3 resources, see [Identity and access management in Amazon S3](#) in the *Amazon Simple Storage Service User Guide*.

Example read/write access policy

Grant read/write access to Amazon S3 bucket

The following example policy for AWS Transfer Family grants read/write access to objects in your Amazon S3 bucket.

Note

In the following example, replace `bucket_name` with the name of your S3 bucket.

Also, note that the `GetObjectACL` and `PutObjectACL` statements are only required if you are doing Cross Account Access. That is, your Transfer Family server needs to access a bucket in a different account.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowListingOfUserFolder",
            "Action": [
                "s3>ListBucket"
            ],
            "Effect": "Allow",
            "Resource": [
                "arn:aws:s3:::<bucket_name>"
            ]
        },
        {
            "Sid": "HomeDirObjectAccess",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:GetObject",
                "s3>DeleteObject",
                "s3>DeleteObjectVersion",
                "s3:GetObjectVersion",
                "s3:GetObjectACL",
                "s3:PutObjectACL"
            ]
        }
    ]
}
```

```
        ],
        "Resource": "arn:aws:s3:::bucket_name/*"
    }
}
```

Grant file system access to files in Amazon EFS file system

Note

In addition to the policy, you must also make sure your POSIX file permissions are granting the appropriate access. For more information, see [Working with users, groups, and permissions at the Network File System \(NFS\) Level](#) in the *Amazon Elastic File System User Guide*.

The following example policy grants root file system access to files in your Amazon EFS file system.

Note

In the following examples, replace `region-id` with your region, `account-id` with the account the file is in, and `file-system-id` with the ID of your Amazon Elastic File System (Amazon EFS).

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

The following example policy grants user file system access to files in your Amazon EFS file system.

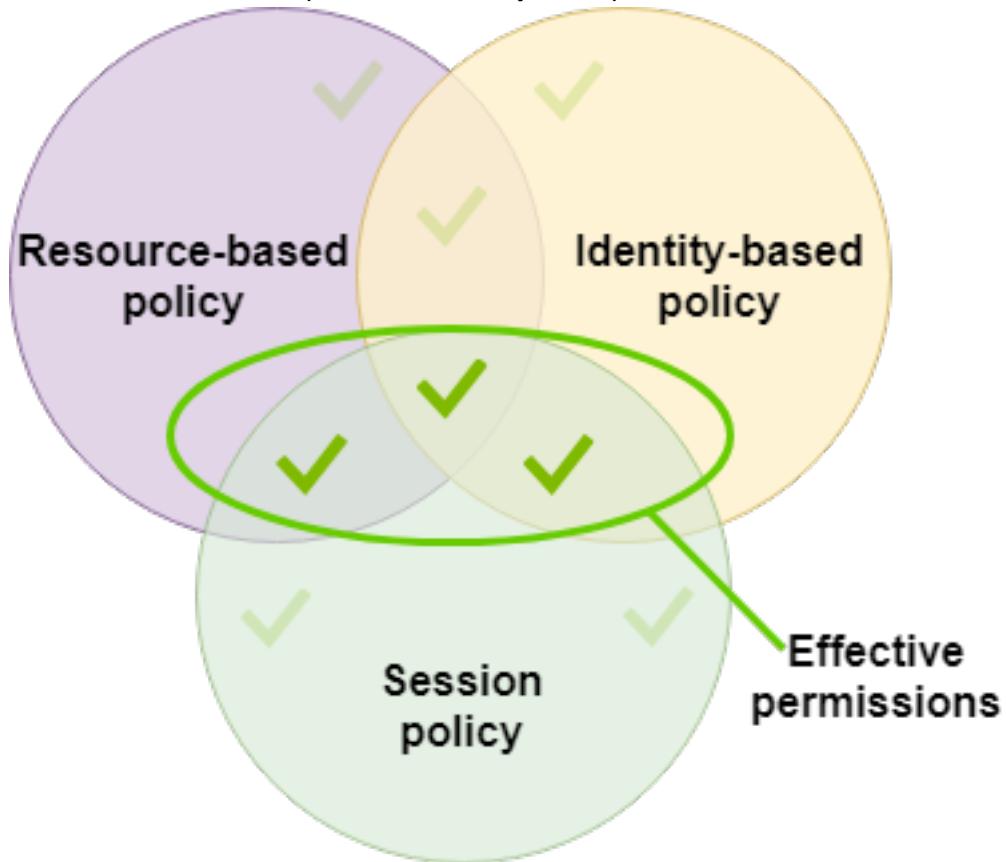
```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "UserFileSystemAccess",
            "Effect": "Allow",
            "Action": [
                "elasticfilesystem:ClientMount",
                "elasticfilesystem:ClientWrite"
            ],
            "Resource": "arn:aws:elasticfilesystem:region-id:account-id:file-system/file-s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

Example session policy

When an administrator creates a role, the role often includes broad permissions to cover multiple use cases or team members. If an administrator configures a [console URL](#), they can reduce permissions

for the resulting session by using a *session policy*. For example, if you create a role with [read/write access \(p. 10\)](#), you can set up a URL that limits users' access to only their home directories.

Session policies are advanced policies that you pass as a parameter when you programmatically create a temporary session for a role or user. Session policies are useful for locking down users so that they have access only to portions of your bucket where object prefixes contain their username. The session policy's permissions are the intersection of the session policies and the resource-based policies plus the intersection of the session policies and identity-based policies.



For more details, see [Session policies](#) in the *IAM User Guide*.

In AWS Transfer Family, a session policy is supported only when you are transferring to or from Amazon S3.

Note

The maximum length of a session policy is 2048 characters. For more details, see the [Policy request parameter](#) for the `CreateUser` action in the *API reference*.

The following example policy is a session policy that limits users' access to their home directories only.

Note

If your Amazon S3 bucket is encrypted using AWS Key Management Service (AWS KMS), you must specify additional permissions in your policy. For details, see [Data encryption \(p. 126\)](#). Additionally, you can see more information about [session policies](#) in the *IAM User Guide*.

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

```
"Sid": "AllowListingOfUserFolder",
"Action": [
    "s3>ListBucket"
],
"Effect": "Allow",
"Resource": [
    "arn:aws:s3:::${transfer:HomeBucket}"
],
"Condition": {
    "Stringlike": {
        "s3:prefix": [
            "${transfer:HomeFolder}/*",
            "${transfer:HomeFolder}"
        ]
    }
}
},
{
    "Sid": "HomeDirObjectAccess",
    "Effect": "Allow",
    "Action": [
        "s3:PutObject",
        "s3:GetObject",
        "s3>DeleteObject",
        "s3>DeleteObjectVersion",
        "s3:GetObjectVersion",
        "s3:GetObjectACL",
        "s3:PutObjectACL"
    ],
    "Resource": "arn:aws:s3:::${transfer:HomeDirectory}*"
}
]
}
```

Note

In the policy above, it is assumed that users have their home directories set to include a trailing slash, to signify that it is a directory. If, on the other hand, you set a user's `HomeDirectory` without the trailing slash, then you should include it as part of your policy.

In the previous example policy, note the use of the `transfer:HomeFolder`, `transfer:HomeBucket`, and `transfer:HomeDirectory` policy parameters. These parameters are set for the `HomeDirectory` that is configured for the user, as described in [HomeDirectory](#) and [Implementing your API Gateway method \(p. 77\)](#). These parameters have the following definitions:

- The `transfer:HomeBucket` parameter is replaced with the first component of `HomeDirectory`.
- The `transfer:HomeFolder` parameter is replaced with the remaining portions of the `HomeDirectory` parameter.
- The `transfer:HomeDirectory` parameter has the leading forward slash (/) removed so that it can be used as part of an S3 Amazon Resource Name (ARN) in a `Resource` statement.

Note

If you are using Logical directories—that is, the user's `homeDirectoryType` is `LOGICAL`—these policy parameters (`HomeBucket`, `HomeDirectory`, and `HomeFolder`) are not supported.

For example, assume that the `HomeDirectory` parameter that is configured for the Transfer Family user is `/home/bob/amazon/stuff/`.

- `transfer:HomeBucket` is set to `/home`.
- `transfer:HomeFolder` is set to `/bob/amazon/stuff/`.
- `transfer:HomeDirectory` becomes `home/bob/amazon/stuff/`.

The first "Sid" allows the user to list all directories starting from /home/bob/amazon/stuff/.

The second "Sid" limits the user's put and get access to that same path, /home/bob/amazon/stuff/.

Tutorial: Getting started with AWS Transfer Family

Use this tutorial to get started with AWS Transfer Family (Transfer Family). You'll learn how to create an SFTP-enabled server with publicly accessible endpoint using Amazon S3 storage, add a user with service-managed authentication, and transfer a file with Cyberduck.

Contents

- [Prerequisites \(p. 15\)](#)
- [Step 1: Sign in to the AWS Transfer Family console \(p. 15\)](#)
- [Step 2: Create an SFTP-enabled server \(p. 15\)](#)
- [Step 3: Add a service managed user \(p. 16\)](#)
- [Step 4: Transfer a file using a client \(p. 17\)](#)

Prerequisites

Before you begin, be sure to complete the requirements in [Setting up \(p. 5\)](#). As part of this setup, you create an Amazon Simple Storage Service (Amazon S3) bucket and an AWS Identity and Access Management (IAM) user role.

Note

The role you create needs permissions defined in **AmazonS3FullAccess**, **AWSTransferConsoleFullAccess**, and **IAMFullAccess** policies.

- **AmazonS3FullAccess** grants permissions to setup and use an Amazon S3 bucket.
- **AWSTransferConsoleFullAccess** grants permissions for your SFTP user to create Transfer Family resources.
- **IAMFullAccess** grants permissions to create the roles and policies you need.

Step 1: Sign in to the AWS Transfer Family console

To sign in to Transfer Family

1. Sign in to the AWS Management Console and open the AWS Transfer Family console at <https://console.aws.amazon.com/transfer/>.
2. For **Account ID or alias**, enter your account ID or alias.
3. For **IAM user name**, enter the name of the user role that you created for Transfer Family.
4. For **Password**, enter your AWS account password.
5. Choose **Sign in**.

Step 2: Create an SFTP-enabled server

Secure Shell (SSH) File Transfer Protocol (SFTP) is a network protocol used for secure transfer of data over the internet. The protocol supports the full security and authentication functionality of SSH. It is

widely used to exchange data, including sensitive information between business partners in a variety of industries such as financial services, healthcare, retail, and advertising.

To create an SFTP-enabled server

1. Select **Servers** from the Navigation pane then choose **Create server**.
2. In **Choose protocols**, select **SFTP**, and then choose **Next**.
3. In **Choose an identity provider**, choose **Service managed** to store user identities and keys in Transfer Family, and then choose **Next**.
4. In **Choose an endpoint**, do the following:
 - a. For **Endpoint type**, choose the **Publicly accessible** endpoint type.
 - b. For **Custom hostname**, choose **None**.
 - c. Choose **Next**.
5. In **Choose a domain**, choose **Amazon S3**.
6. In **Configure additional details**, do the following:
 - a. For **CloudWatch logging**, choose **Create a new role** to allow Transfer Family to automatically create the IAM role, as long as you have the right permissions to create a new role. The IAM role that is created is called `AWSTransferLoggingAccess`.
 - b. For **Cryptographic algorithm options**, choose a security policy that contains the cryptographic algorithms enabled for use by your server. The default security policy is `TransferSecurityPolicy-2020-06`.
 - c. Choose **Next**.
7. In **Review and create**, choose **Create server**. You are taken to the **Servers** page.

It can take a couple of minutes before the status for your new server changes to **Online**. At that point, your server can perform file operations, but you'll need to create a user first.

Step 3: Add a service managed user

To add a user to the SFTP-enabled server

1. On the **Servers** page, select the check box of the server that you want to add a user to.
2. Choose **Add user**.
3. In the **User configuration** section, for **Username**, enter the user name. This user name must be a minimum of 3 and a maximum of 100 characters. You can use the following characters in the user name: a-z, A-Z, 0-9, underscore '_', hyphen '-', period '.', and at sign '@'. The user name can't start with a hyphen, period, or at sign.
4. For **Access**, choose the IAM role that you previously created that provides access to your Amazon S3 bucket.

You created this IAM role using the procedure in [Create an IAM role and policy \(p. 8\)](#). That IAM role includes an IAM policy that provides access to your Amazon S3 bucket. It also includes a trust relationship with the AWS Transfer Family service, defined in another IAM policy.

Note

The IAM role for the service managed user must contain the permissions to access the desired bucket. Permissions to access the desired bucket are covered within **S3FullAccess** which grants administrator level permissions to S3 resources.

5. For **Policy**, choose **None**.

6. For **Home directory**, choose the Amazon S3 bucket to store the data to transfer using AWS Transfer Family. Enter the path to the home directory where your user lands when they log in using their client.

If you leave this parameter blank, the root directory of your Amazon S3 bucket is used. In this case, make sure that your IAM role provides access to this root directory.

Note

We recommend that you choose a directory path that contains the user name of the user, which enables you to effectively use a session policy. The session policy limits user access in the Amazon S3 bucket to that user's home directory.

7. For **Restricted**, select the check box so that your users can't access anything outside of that folder and can't see the Amazon S3 bucket or folder name.

Note

When assigning the user a home directory and restricting the user to that home directory, this should be sufficient enough to lock down the user's access to the designated folder. Use a session policy when you need to apply further controls.

8. For **SSH public key**, enter the public SSH key portion of the SSH key pair.

Your key is validated by the service before you can add your new user.

Important

The format of the SSH public key is ssh-rsa <*string*>. For instructions on how to generate an SSH key pair, see [Generate SSH keys \(p. 127\)](#).

9. (Optional) For **Key** and **Value**, enter one or more tags as key-value pairs, and choose **Add tag**.
10. Choose **Add** to add your new user to the server that you chose.

The new user appears in the **Users** section of the **Server details** page.

Step 4: Transfer a file using a client

You transfer files over the AWS Transfer Family service by specifying the transfer operation in a client. AWS Transfer Family supports several clients. For details, see [Transferring files using a client \(p. 90\)](#).

This section contains procedures for using Cyberduck and OpenSSH.

Topics

- [Use Cyberduck \(p. 17\)](#)
- [Use OpenSSH \(p. 18\)](#)

Use Cyberduck

To transfer files over AWS Transfer Family using Cyberduck

1. Open the [Cyberduck](#) client.
2. Choose **Open Connection**.
3. In the **Open Connection** dialog box, choose **SFTP (SSH File Transfer Protocol)**.
4. For **Server**, enter your server endpoint. The server endpoint is located on the **Server details** page, see [View server details \(p. 108\)](#).
5. For **Port number**, enter **22** for SFTP.
6. For **Username**, enter the name for the user that you created in [Managing users \(p. 54\)](#).
7. For **SSH Private Key**, choose or enter the SSH private key.

8. Choose **Connect**.
9. Perform your file transfer.

Depending on where your files are, do one of the following:

- In your local directory (the source), choose the files that you want to transfer, and drag and drop them into the Amazon S3 directory (the target).
- In the Amazon S3 directory (the source), choose the files that you want to transfer, and drag and drop them into your local directory (the target).

Use OpenSSH

Use the instructions that follow to transfer files from the command line using OpenSSH.

Note

This client works only with an SFTP-enabled server.

To transfer files over AWS Transfer Family using the OpenSSH command line utility

1. On Linux or Macintosh, open a command terminal.
2. At the prompt, enter the following command: % sftp -i transfer-key sftp_user@service_endpoint

In the preceding command, `sftp_user` is the user name and `transfer-key` is the SSH private key. Here, `service_endpoint` is the server's endpoint as shown in the AWS Transfer Family console for the selected server.

An `sftp` prompt should appear.

3. (Optional) To view the user's home directory, enter the following command at the `sftp` prompt:
`sftp> pwd`
4. On the next line, enter the following text: `sftp> cd /mybucket/home/sftp_user`

In this getting-started exercise, this Amazon S3 bucket is the target of the file transfer.

5. On the next line, enter the following command: `sftp> put filename.txt`

The `put` command transfers the file into the Amazon S3 bucket.

A message like the following appears, indicating that the file transfer is in progress, or complete.

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

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

Creating a server

Following, you can find how to create a file transfer protocol enabled server using the AWS Transfer Family service. The following protocols are available:

- Secure Shell (SSH) File Transfer Protocol (SFTP) – file transfer over SSH
- File Transfer Protocol Secure (FTPS) – file transfer with TLS encryption
- File Transfer Protocol (FTP) – unencrypted file transfer

You can create a server with multiple protocols.

Note

If you have multiple protocols enabled for the same server endpoint and want to provide access using the same user name over multiple protocols, you can do so as long as the credentials specific to the protocol have been set up in your identity provider. For FTP, we recommend maintaining separate credentials from SFTP and FTPS. This is because, unlike SFTP and FTPS, FTP transmits credentials in clear text. By isolating FTP credentials from SFTP or FTPS, if FTP credentials are shared or exposed, your workloads using SFTP or FTPS remain secure.

When you create a server, you choose a specific AWS Region to perform the file operation requests of users who are assigned to that server. Along with assigning the server one or more protocols, you also assign one of the following identity provider types:

- Service managed using SSH keys. For details, see [Working with service-managed users \(p. 54\)](#).
- AWS Managed Microsoft AD. This method allows you integrate your Microsoft Active Directory groups to provide access to your Transfer Family servers. For details, see [Using the AWS Directory Service identity provider \(p. 60\)](#).
- A custom method. The custom identity provider method uses Amazon API Gateway and enables you to integrate your directory service to authenticate and authorize your users. The service automatically assigns an identifier that uniquely identifies your server. For details, see [Working with custom identity providers \(p. 69\)](#).

You also assign the server an endpoint type (publicly accessible or VPC hosted) and a hostname using the default server endpoint, or a custom hostname using the Amazon Route 53 service or by using a Domain Name System (DNS) service of your choice. A server hostname must be unique in the AWS Region where it's created.

Additionally, you can assign an Amazon CloudWatch logging role to push events to your CloudWatch Logs, choose a security policy that contains the cryptographic algorithms enabled for use by your server, and add metadata to the server in the form of tags that are key-value pairs.

Important

You incur costs for instantiated servers and for data transfer. For information about pricing and to use AWS Pricing Calculator to get an estimate of the cost to use Transfer Family, see [AWS Transfer Family pricing](#).

Identity provider options

AWS Transfer Family provides several methods for authenticating and managing users. The following table compares the available identity providers you can use with Transfer Family.

Action	AWS Transfer Family service managed	AWS Directory Service for Microsoft Active Directory	Amazon API Gateway	Lambda
Logical home directory	Yes	Yes	Yes	Yes
IAM and POSIX	Yes	Yes	Yes	Yes
Ad hoc access structure	Yes	No	Yes	Yes
Password authentication	No	Yes	Yes	Yes
Key-based authentication	Yes	No	Yes	Yes
AWS Web Application Firewall	No	No	Yes	No

Notes:

- IAM is used to control access for Amazon S3 backing storage, and POSIX is used for Amazon EFS.
- *Ad hoc* refers to the ability to send the user profile at run time. For example, you can land users in their home directories by passing the username as a variable.
- For details on AWS WAF, see [Add a web application firewall \(p. 146\)](#).

In the following procedures, you can create an SFTP-enabled server, FTPS-enabled server, or FTP-enabled server.

Next step

- [Create an SFTP-enabled server \(p. 20\)](#)
- [Create an FTPS-enabled server \(p. 25\)](#)
- [Create an FTP-enabled server \(p. 30\)](#)

Create an SFTP-enabled server

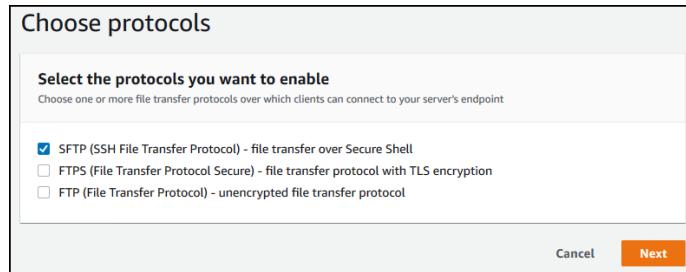
Secure Shell (SSH) File Transfer Protocol (SFTP) is a network protocol used for secure transfer of data over the internet. The protocol supports the full security and authentication functionality of SSH. It is widely used to exchange data, including sensitive information between business partners in a variety of industries such as financial services, healthcare, retail, and advertising.

Note

SFTP servers for Transfer Family operate over port 22.

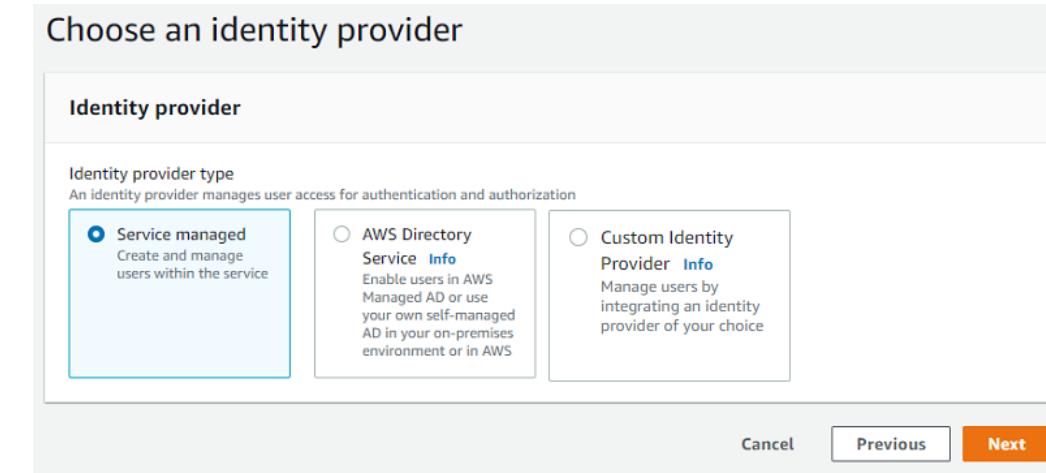
To create an SFTP-enabled server

1. Open the AWS Transfer Family console at <https://console.aws.amazon.com/transfer/> and select **Servers** from the navigation pane, then choose **Create server**.
2. In **Choose protocols**, select **SFTP**, and then choose **Next**.



3. In **Choose an identity provider**, choose the identity provider that you want to use to manage user access. You have the following options:

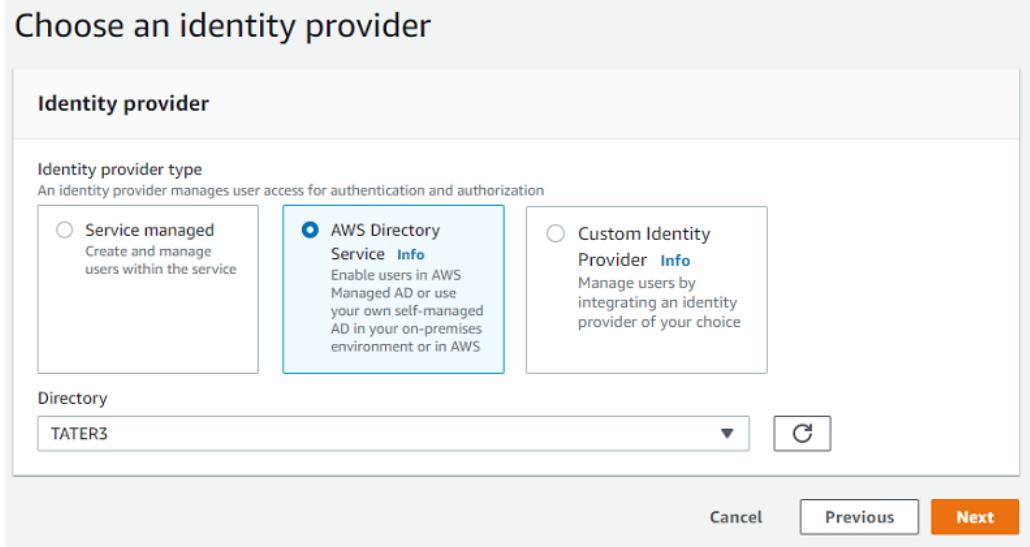
- **Service managed:** you store user identities and keys in AWS Transfer Family.



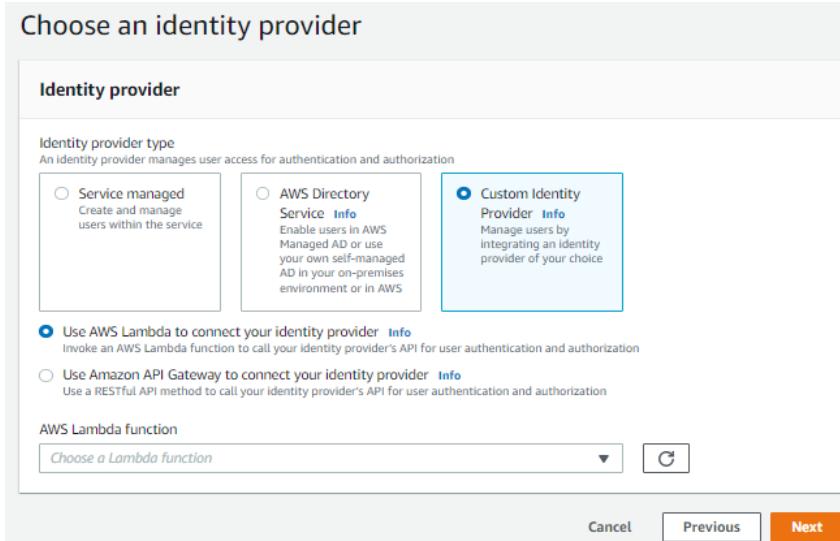
- **AWS Directory Service for Microsoft Active Directory:** you provide an AWS Directory Service directory to access the endpoint. By doing so, you can use credentials stored in your Active Directory to authenticate your users. To learn more about working with AWS Managed Microsoft AD identity providers, see [Using the AWS Directory Service identity provider \(p. 60\)](#).

Note

Cross-Account and Shared directories are not supported for AWS Managed Microsoft AD.



- **Custom:** choose either of the following options:
 - **AWS Lambda** to integrate your identity provider: use your existing identity providers, backed by a Lambda function. You provide the name of the Lambda function. For more details, see [Using AWS Lambda to integrate your identity provider \(p. 70\)](#)
 - **Amazon API Gateway method** backed by a Lambda function: create an API Gateway for use as an identity provider. You provide an Amazon API Gateway URL and an invocation role. For more details, see [Using Amazon API Gateway to integrate your identity provider \(p. 74\)](#).



4. Choose **Next**.
5. In **Choose an endpoint**, do the following:
 - a. For **Endpoint type**, choose the **Publicly accessible** endpoint type. For a **VPC hosted** endpoint, see [Create a server in a virtual private cloud \(p. 35\)](#).
 - b. (Optional) For **Custom hostname**, choose **None**.

You get a server hostname provided by AWS Transfer Family. The server hostname takes the form `serverId.server.transfer.regionId.amazonaws.com`.

For a custom hostname, you specify a custom alias for your server endpoint. To learn more about working with custom hostnames, see [Working with custom hostnames \(p. 47\)](#).

- c. (Optional) For **FIPS Enabled**, select the **FIPS Enabled endpoint** check box to ensure that the endpoint complies with Federal Information Processing Standards (FIPS).

Note

FIPS-enabled endpoints are only available in North American AWS Regions. For available Regions, see [AWS Transfer Family endpoints and quotas](#) in the *AWS General Reference*. For more information about the available FIPS endpoints, see [Federal Information Processing Standard \(FIPS\) 140-2](#).

- d. Choose **Next**.

The screenshot shows the 'Choose an endpoint' configuration page. It has three main sections: 'Endpoint configuration' (with a 'Info' link), 'Endpoint type' (with 'Publicly accessible' selected and 'VPC hosted' as an alternative), and 'Custom hostname' (set to 'None'). At the bottom, there's a 'FIPS Enabled' section with a checkbox labeled 'FIPS Enabled endpoint' which is unchecked. Navigation buttons 'Cancel', 'Previous', and 'Next' are at the bottom right.

6. On the **Choose domain** page, choose the AWS storage service that you want to use to store and access your data over the selected protocol:
 - Choose **Amazon S3** to store and access your files as objects over the selected protocol.
 - Choose **Amazon EFS** to store and access your files in your Amazon EFS file system over the selected protocol.

Choose **Next**.

7. In **Configure additional details**, do the following:

- a. For **CloudWatch logging**, choose one of the following to enable Amazon CloudWatch logging of your user activity:
 - **Create a new role** to allow Transfer Family to create the IAM role automatically, as long as you have the right permissions to create a new role. The IAM role that is created is called `AWSTransferLoggingAccess`.
 - **Choose an existing role** to choose an existing IAM role from your account. Under **Logging role**, choose the role. This IAM role should include a trust policy with **Service** set to `transfer.amazonaws.com`.

For more information about CloudWatch logging, see [Log activity with CloudWatch \(p. 122\)](#).

Note

- You can't view end-user activity in CloudWatch if you don't specify a logging role.
- If you don't want to set up a CloudWatch logging role, choose **Choose an existing role**, but don't select a logging role.

CloudWatch logging [Info](#)

Logging role
IAM role for the server to push events to your CloudWatch logs

Create a new role Choose an existing role

ⓘ AWS Transfer Family needs access to your CloudWatch logs to log your server activity. By continuing, you are allowing us to create a new role to allow this

- b. For **Cryptographic algorithm options**, choose a security policy that contains the cryptographic algorithms enabled for use by your server.

Note

By default:

- If **FIPS Enabled endpoint** is not selected, the `TransferSecurityPolicy-2020-06` security policy is attached to your server.
- If **FIPS Enabled endpoint** is selected, the `TransferSecurityPolicy-FIPS-2020-06` security policy is attached to your server.

For more information about security policies, see [Working with security policies \(p. 49\)](#).

Cryptographic algorithm options [Info](#)

Security Policy
Choose a security policy that contains the cryptographic algorithms enabled for use by your server

TransferSecurityPolicy-2020-06

- c. (Optional) For **Server Host Key**, enter an RSA private key that will be used to identify your server when clients connect to it over SFTP.

Note

This section is only for migrating users from an existing SFTP-enabled server.

Server Host Key [Info](#)

RSA private key - optional
Upload an RSA private key that will be used to identify your server when clients connect to it over SFTP

Enter optional RSA private key

ⓘ You can ignore this section unless you are migrating users from an existing SFTP server.

- d. (Optional) For **Tags**, for **Key** and **Value**, enter one or more tags as key-value pairs, and then choose **Add tag**.
- e. Choose **Next**.

Tags

Key	Value
<input type="text"/>	<input type="text"/>

Remove tag

Add tag

Cancel Previous Next

- f. (Optional) For **Post-upload file-processing WorkflowId**, enter a workflow ID and a corresponding role that Transfer should assume when executing the workflow.

Post-upload processing Info

Workflow
Select the workflow that Transfer should run on all files after they've been uploaded via this server

w-abcdef01234567890 ▼ ⟳ Create a new Workflow

Role
Select the role that Transfer should assume when executing a workflow

transfer-workflows ▼ ⟳

8. In **Review and create**, review your choices.

- If you want to edit any of them, choose **Edit** next to the step.

Note

You will need to review each step after the step you chose to edit.

- If you have no changes, choose **Create server** to create your server. You are taken to the **Servers** page, shown following, where your new server is listed.

It can take a couple of minutes before the status for your new server changes to **Online**. At that point, your server can perform file operations for your users.

Servers (1)									<small>⟳</small>	<small>Actions ▾</small>	<small>Add user</small>	<small>Create server</small>
<input type="checkbox"/>	Hostname	▲	Server ID	▼	State	▼	users	▼	Endpoint type	▼	Domain	▼
<input type="checkbox"/>	-	▲	s- [REDACTED]	▼	Starting	▼	No Users	▼	Public	▼	Amazon S3	▼

Create an FTPS-enabled server

File Transfer Protocol over SSL (FTPS) is an extension to FTP. It uses Transport Layer Security (TLS) and Secure Sockets Layer (SSL) cryptographic protocols to encrypt traffic. FTPS allows encryption of both the control and data channel connections either concurrently or independently.

To create an FTPS-enabled server

1. Open the AWS Transfer Family console at <https://console.aws.amazon.com/transfer/> and select **Servers** from the navigation pane, then choose **Create server**.
2. In **Choose protocols**, select **FTPS**.

For **Server certificate**, choose a certificate stored in AWS Certificate Manager (ACM) which will be used to identify your server when clients connect to it over FTPS and then choose **Next**.

To request a new public certificate, see [Request a public certificate](#) in the *AWS Certificate Manager User Guide*.

To import an existing certificate into ACM, see [Importing certificates into ACM](#) in the *AWS Certificate Manager User Guide*.

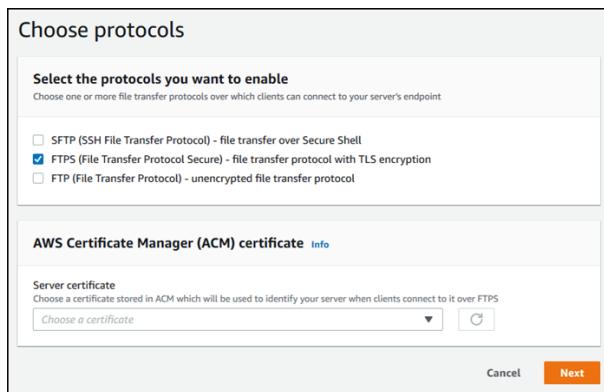
To request a private certificate to use FTPS through private IP addresses, see [Requesting a Private Certificate](#) in the *AWS Certificate Manager User Guide*.

Certificates with the following cryptographic algorithms and key sizes are supported:

- 2048-bit RSA (RSA_2048)
- 4096-bit RSA (RSA_4096)
- Elliptic Prime Curve 256 bit (EC_prime256v1)
- Elliptic Prime Curve 384 bit (EC_secp384r1)
- Elliptic Prime Curve 521 bit (EC_secp521r1)

Note

The certificate must be a valid SSL/TLS X.509 version 3 certificate with FQDN or IP address specified and information about the issuer.

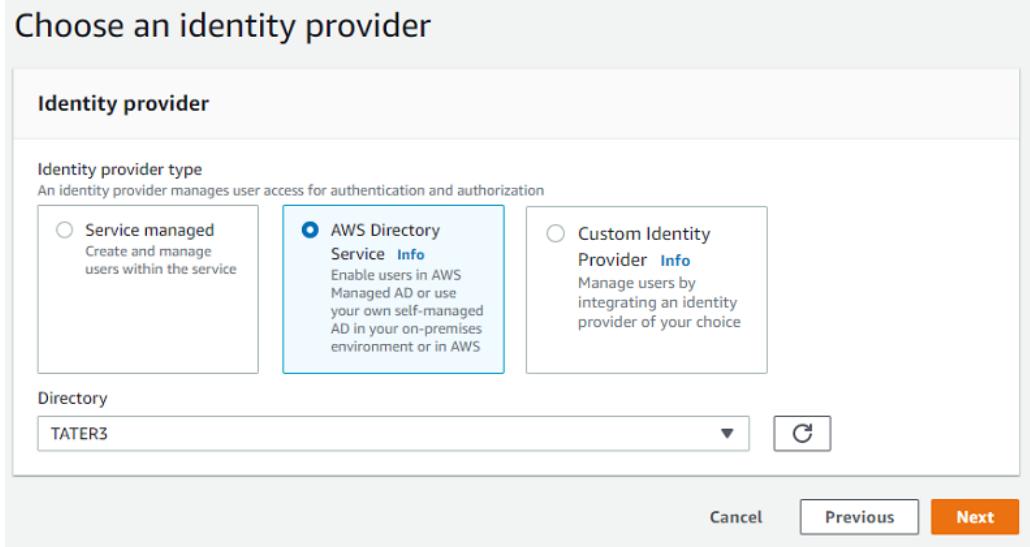


3. In **Choose an identity provider**, choose the identity provider that you want to use to manage user access. You have the following options:

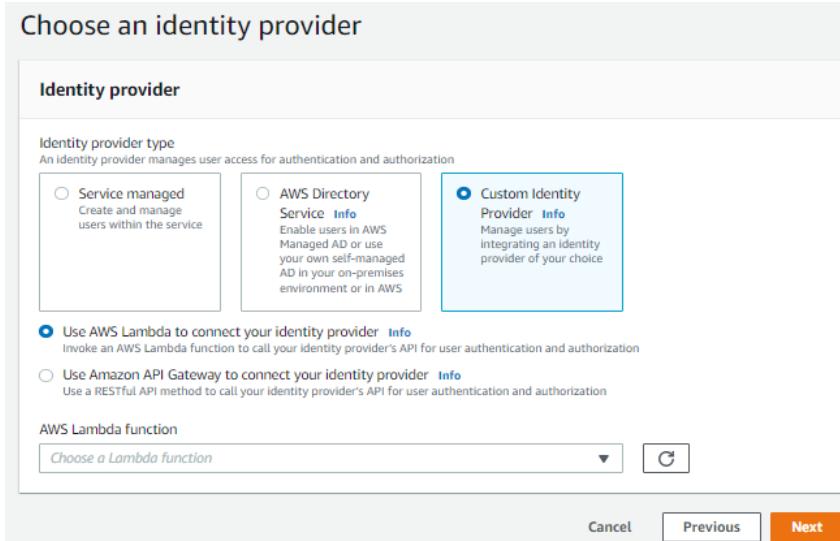
- **AWS Directory Service for Microsoft Active Directory:** you provide an AWS Directory Service directory to access the endpoint. By doing so, you can use credentials stored in your Active Directory to authenticate your users. To learn more about working with AWS Managed Microsoft AD identity providers, see [Using the AWS Directory Service identity provider \(p. 60\)](#).

Note

Cross-Account and Shared directories are not supported for AWS Managed Microsoft AD.



- **Custom:** choose either of the following options:
 - **AWS Lambda** to integrate your identity provider: use your existing identity providers, backed by a Lambda function. You provide the name of the Lambda function. For more details, see [Using AWS Lambda to integrate your identity provider \(p. 70\)](#)
 - **Amazon API Gateway method** backed by a Lambda function: create an API Gateway for use as an identity provider. You provide an Amazon API Gateway URL and an invocation role. For more details, see [Using Amazon API Gateway to integrate your identity provider \(p. 74\)](#).



4. In **Choose an endpoint**, do the following:

Note

FTPS servers for Transfer Family operate over Port 21 (Control Channel) and Port Range 8192-8200 (Data Channel).

- a. For **Endpoint type**, choose the **VPC hosted** endpoint type to host your server's endpoint. For information about setting up your VPC hosted endpoint, see [Create a server in a virtual private cloud \(p. 35\)](#).

Note

Publicly accessible endpoints are not supported.

- b. (Optional) For **FIPS Enabled**, select the **FIPS Enabled endpoint** check box to ensure that the endpoint complies with Federal Information Processing Standards (FIPS).

Note

FIPS-enabled endpoints are only available in North American AWS Regions. For available Regions, see [AWS Transfer Family endpoints and quotas](#) in the *AWS General Reference*. For more information about the available FIPS endpoints, see [Federal Information Processing Standard \(FIPS\) 140-2](#).

- c. Choose **Next**.

The screenshot shows the 'Choose an endpoint' configuration page. It includes fields for selecting the endpoint type (Publicly accessible or VPC hosted), access type (Internal or Internet Facing), choosing a VPC, and enabling FIPS. The 'VPC hosted' option is selected. At the bottom, there are 'Cancel', 'Previous', and 'Next' buttons, with 'Next' being highlighted.

5. On the **Choose domain** page, choose the AWS storage service that you want to use to store and access your data over the selected protocol:
- Choose **Amazon S3** to store and access your files as objects over the selected protocol.
 - Choose **Amazon EFS** to store and access your files in your Amazon EFS file system over the selected protocol.

Choose **Next**.

6. In **Configure additional details**, do the following:

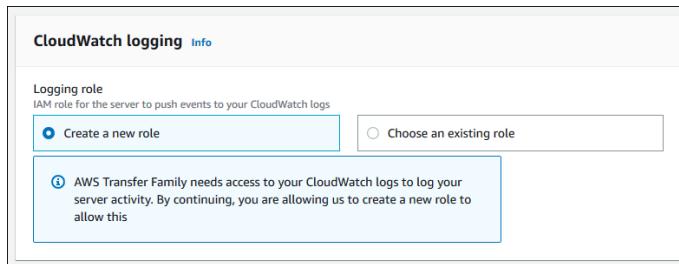
- a. For **CloudWatch logging**, choose one of the following to enable Amazon CloudWatch logging of your user activity:
- Create a new role** to allow Transfer Family to automatically create the IAM role, as long as you have the right permissions to create a new role. The IAM role that is created is called `AWSTransferLoggingAccess`.
 - Choose an existing role** to choose an existing IAM role from your account. Under **Logging role**, choose the role. This IAM role should include a trust policy with **Service** set to `transfer.amazonaws.com`.

For more information about CloudWatch logging, see [Log activity with CloudWatch \(p. 122\)](#).

Note

- You can't view end-user activity in CloudWatch if you don't specify a logging role.

- If you don't want to set up a CloudWatch logging role, choose **Choose an existing role**, but don't select a logging role.



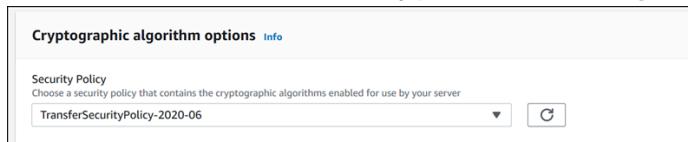
- b. For **Cryptographic algorithm options**, choose a security policy that contains the cryptographic algorithms enabled for use by your server.

Note

By default:

- If **FIPS Enabled endpoint** is not selected, the `TransferSecurityPolicy-2020-06` security policy is attached to your server.
- If **FIPS Enabled endpoint** is selected, the `TransferSecurityPolicy-FIPS-2020-06` security policy is attached to your server.

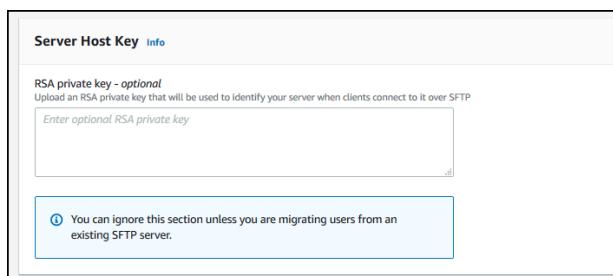
For more information about security policies, see [Working with security policies \(p. 49\)](#).



- c. (Optional) For **Server Host Key**, keep it blank.

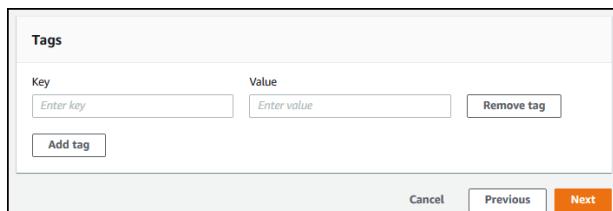
Note

This section is only for migrating users from an existing SFTP-enabled server.



- d. (Optional) For **Tags**, for **Key** and **Value**, enter one or more tags as key-value pairs, and then choose **Add tag**.

- e. Choose **Next**.



- f. (Optional) For **Post-upload file-processing WorkflowId**, enter a workflow ID and a corresponding role that Transfer should assume when executing the workflow.

Post-upload processing [Info](#)

Workflow
Select the workflow that Transfer should run on all files after they've been uploaded via this server

w-abcdef01234567890 [▼](#) [⟳](#) [Create a new Workflow](#)

Role
Select the role that Transfer should assume when executing a workflow

transfer-workflows [▼](#)

7. In **Review and create**, review your choices.

- If you want to edit any of them, choose **Edit** next to the step.

Note

You will need to review each step after the step you chose to edit.

- If you have no changes, choose **Create server** to create your server. You are taken to the **Servers** page, shown following, where your new server is listed.

It can take a couple of minutes before the status for your new server changes to **Online**. At that point, your server can perform file operations for your users.

Servers (1)							⟳	Actions ▾	Add user	Create server
	Hostname	Server ID	State	users	Endpoint type	Domain				
	s-[REDACTED]		Starting	No Users	Public	Amazon S3				

Next steps – For the next step, continue on to [Working with custom identity providers \(p. 69\)](#) to set up users.

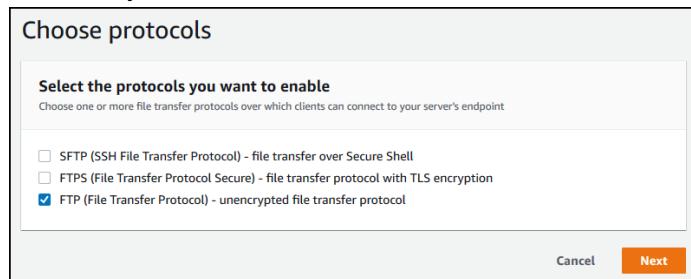
Create an FTP-enabled server

File Transfer Protocol (FTP) is a network protocol used for the transfer of data. FTP uses a separate channel for control and data transfers. The control channel is open until terminated or inactivity timeout. The data channel is active for the duration of the transfer. FTP uses clear text and does not support encryption of traffic.

To create an FTP-enabled server

- Open the AWS Transfer Family console at <https://console.aws.amazon.com/transfer/> and select **Servers** from the navigation pane, then choose **Create server**.

2. In **Choose protocols**, select **FTP**, and then choose **Next**.

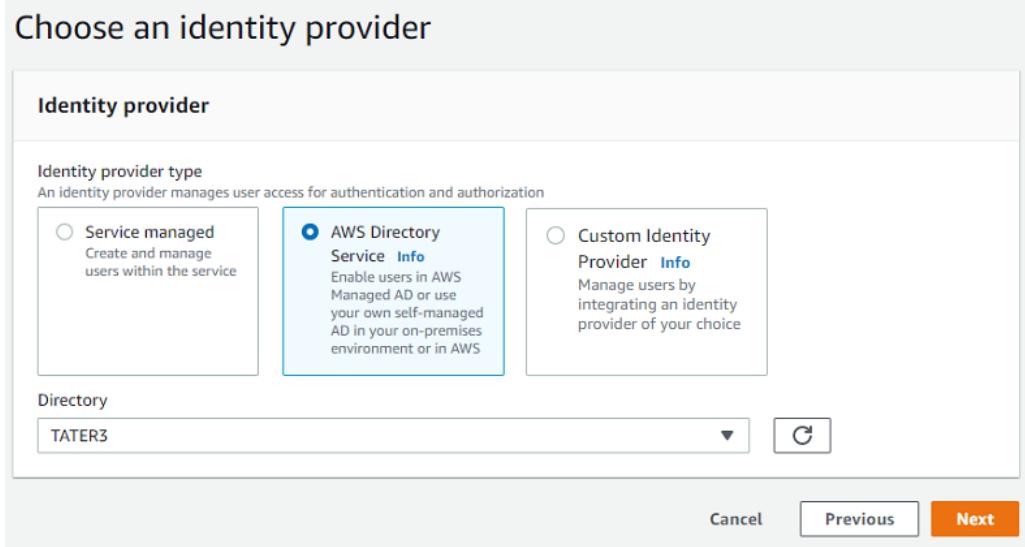


3. In **Choose an identity provider**, choose the identity provider that you want to use to manage user access. You have the following options:

- **AWS Directory Service for Microsoft Active Directory:** you provide an AWS Directory Service directory to access the endpoint. By doing so, you can use credentials stored in your Active Directory to authenticate your users. To learn more about working with AWS Managed Microsoft AD identity providers, see [Using the AWS Directory Service identity provider \(p. 60\)](#).

Note

Cross-Account and Shared directories are not supported for AWS Managed Microsoft AD.



- **Custom:** choose either of the following options:

- **AWS Lambda** to integrate your identity provider: use your existing identity providers, backed by a Lambda function. You provide the name of the Lambda function. For more details, see [Using AWS Lambda to integrate your identity provider \(p. 70\)](#)
- **Amazon API Gateway method** backed by a Lambda function: create an API Gateway for use as an identity provider. You provide an Amazon API Gateway URL and an invocation role. For more details, see [Using Amazon API Gateway to integrate your identity provider \(p. 74\)](#).

Identity provider

Identity provider type
An identity provider manages user access for authentication and authorization

- Service managed
Create and manage users within the service
- AWS Directory Service [Info](#)
Enable users in AWS Managed AD or use your own self-managed AD in your on-premises environment or in AWS
- Custom Identity Provider [Info](#)
Manage users by integrating an identity provider of your choice

AWS Lambda function

[Choose a Lambda function](#) [C](#)

[Cancel](#) [Previous](#) [Next](#)

4. In **Choose an endpoint**, do the following:

Note

FTP servers for Transfer Family operate over Port 21 (Control Channel) and Port Range 8192-8200 (Data Channel).

- a. For **Endpoint type**, choose **VPC hosted** to host your server's endpoint. For information about setting up your VPC hosted endpoint, see [Create a server in a virtual private cloud \(p. 35\)](#).

Note

Publicly accessible endpoints are not supported.

- b. For **FIPS Enabled**, keep the **FIPS Enabled endpoint** check box cleared.

Note

A FIPS-enabled endpoint is not supported.

- c. Choose **Next**.

Endpoint configuration [Info](#)

Endpoint type
Select whether the endpoint will be publicly accessible or hosted inside your VPC

- Publicly accessible
Accessible over the internet
- VPC hosted [Info](#)
Access controlled using Security Groups

Access [Info](#)

- Internal
- Internet Facing

VPC
Select a VPC ID

[Select a VPC ID](#) [C](#) [Create a VPC](#)

FIPS Enabled
Select whether the endpoint should comply with Federal Information Processing Standards (FIPS)

FIPS Enabled endpoint

[Cancel](#) [Previous](#) [Next](#)

5. On the **Choose domain** page, choose the AWS storage service that you want to use to store and access your data over the selected protocol.

- Choose **Amazon S3** to store and access your files as objects over the selected protocol.

- Choose **Amazon EFS** to store and access your files in your Amazon EFS file system over the selected protocol.

Choose Next.

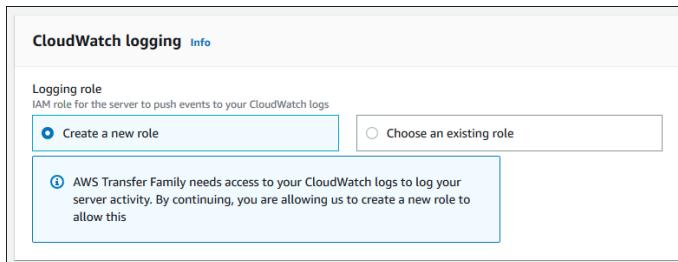
6. In **Configure additional details**, do the following:

- For **CloudWatch logging**, choose one of the following to enable Amazon CloudWatch logging of your user activity:
 - Create a new role** to allow Transfer Family to automatically create the IAM role, as long as you have the right permissions to create a new role. The IAM role that is created is called `AWSTransferLoggingAccess`.
 - Choose an existing role** to choose an existing IAM role from your account. Under **Logging role**, choose the role. This IAM role should include a trust policy with **Service** set to `transfer.amazonaws.com`.

For more information about CloudWatch logging, see [Log activity with CloudWatch \(p. 122\)](#).

Note

- You can't view end-user activity in CloudWatch if you don't specify a logging role.
- If you don't want to set up a CloudWatch logging role, choose **Choose an existing role**, but don't select a logging role.



- For **Cryptographic algorithm options**, choose a security policy that contains the cryptographic algorithms enabled for use by your server.

Note

By default, the `TransferSecurityPolicy-2020-06` security policy is attached to your server.

For more information about security policies, see [Working with security policies \(p. 49\)](#).



- (Optional) For **Server Host Key**, keep it blank.

Note

This section is only for migrating users from an existing SFTP-enabled server.

Server Host Key [Info](#)

RSA private key - *optional*
Upload an RSA private key that will be used to identify your server when clients connect to it over SFTP

Enter optional RSA private key

You can ignore this section unless you are migrating users from an existing SFTP server.

- d. (Optional) For **Tags**, for **Key** and **Value**, enter one or more tags as key-value pairs, and then choose **Add tag**.

- e. Choose **Next**.

Tags

Key	Value
<input type="text"/>	<input type="text"/>

Add tag [Cancel](#) [Previous](#) **Next**

- f. (Optional) For **Post-upload file-processing WorkflowId**, enter a workflow ID and a corresponding role that Transfer should assume when executing the workflow.

Post-upload processing [Info](#)

Workflow
Select the workflow that Transfer should run on all files after they've been uploaded via this server

w-abcdef01234567890 [▼](#) [Create a new Workflow](#)

Role
Select the role that Transfer should assume when executing a workflow

transfer-workflows [▼](#)

7. In **Review and create**, review your choices.

- If you want to edit any of them, choose **Edit** next to the step.

Note

You will need to review each step after the step you chose to edit.

- If you have no changes, choose **Create server** to create your server. You are taken to the **Servers** page, shown following, where your new server is listed.

It can take a couple of minutes before the status for your new server changes to **Online**. At that point, your server can perform file operations for your users.

Servers (1)								Actions ▾	Add user	Create server
	Hostname	Server ID	State	users	Endpoint type	Domain				
	-	s-[REDACTED]	Starting	No Users	Public	Amazon S3				

Next steps – For the next step, continue on to [Working with custom identity providers \(p. 69\)](#) to set up users.

Create a server in a virtual private cloud

You can host your server's endpoint inside a virtual private cloud (VPC) to use for transferring data to and from an Amazon S3 bucket or Amazon EFS file system without going over the public internet.

Note

After May 19, 2021, you won't be able to create a server using `EndpointType=VPC_ENDPOINT` in your AWS account if your account hasn't already done so before May 19, 2021. If you have already created servers with `EndpointType=VPC_ENDPOINT` in your AWS account on or before February 21, 2021, you will not be affected. After this date, use `EndpointType=VPC`. For more information, see the section called "[Discontinuing the use of VPC_ENDPOINT \(p. 42\)](#)".

If you use Amazon Virtual Private Cloud (Amazon VPC) to host your AWS resources, you can establish a private connection between your VPC and a server. You can then use this server to transfer data over your client to and from your Amazon S3 bucket without going over the public internet.

Using Amazon VPC, you can launch AWS resources in a custom virtual network. You can use a VPC to control your network settings, such as the IP address range, subnets, route tables, and network gateways. For more information about VPCs, see [What Is Amazon VPC?](#) in the *Amazon VPC User Guide*.

In the next sections, find instructions on how to create and connect your VPC to a server. As an overview, you do this as follows:

1. Set up a server using a VPC endpoint.
2. Connect to your server using a client that is inside your VPC through the VPC endpoint. Doing this enables you to transfer data that is stored in your Amazon S3 bucket over your client using AWS Transfer Family. You can perform this transfer even though the network is disconnected from the public internet.
3. In addition, if you choose to make your server's endpoint internet-facing, you can associate Elastic IP addresses with your endpoint. Doing this lets clients outside of your VPC connect to your server. You can use VPC security groups to control access to authenticated users whose requests originate only from allowed addresses.

Topics

- [Create a server endpoint that can be accessed only within your VPC \(p. 36\)](#)
- [Create an internet-facing endpoint for your server \(p. 37\)](#)
- [Change the endpoint type for your server \(p. 40\)](#)
- [Discontinuing the use of VPC_ENDPOINT \(p. 42\)](#)
- [Updating the AWS Transfer Family server endpoint type from VPC_ENDPOINT to VPC \(p. 42\)](#)

Create a server endpoint that can be accessed only within your VPC

In the following procedure, you create a server endpoint that is accessible only to resources within your VPC.

To create a server endpoint inside a VPC

1. Open the AWS Transfer Family console at <https://console.aws.amazon.com/transfer/>.
2. From the navigation pane, select **Servers**, then choose **Create server**.
3. In **Choose protocols**, select one or more protocols, and then choose **Next**. For more information about protocols, see [Step 2: Create an SFTP-enabled server \(p. 15\)](#).
4. In **Choose an identity provider**, choose **Service managed** to store user identities and keys in AWS Transfer Family, and then choose **Next**.

Note

This procedure uses the service-managed option. If you choose **Custom**, you provide an Amazon API Gateway endpoint and an AWS Identity and Access Management (IAM) role to access the endpoint. By doing so, you can integrate your directory service to authenticate and authorize your users. To learn more about working with custom identity providers, see [Working with custom identity providers \(p. 69\)](#).

5. In **Choose an endpoint**, do the following:

Note

FTP and FTPS servers for Transfer Family operate over Port 21 (Control Channel) and Port Range 8192-8200 (Data Channel).

- a. For **Endpoint type**, choose the **VPC hosted** endpoint type to host your server's endpoint.
- b. For **Access**, choose **Internal** to make your endpoint only accessible to clients using the endpoint's private IP addresses.

Note

For details on the **Internet Facing** option, see [Create an internet-facing endpoint for your server \(p. 37\)](#). A server that is created in a VPC for internal access only doesn't support custom hostnames.

- c. For **VPC**, choose an existing VPC ID or choose **Create a VPC** to create a new VPC.
- d. In the **Availability Zones** section, choose up to three Availability Zones and associated subnets.
- e. In the **Security Groups** section, choose an existing security group ID or IDs or choose **Create a security group** to create a new security group. For more information about security groups, see [Security groups for your VPC](#) in the *Amazon Virtual Private Cloud User Guide*. To create a security group, see [Creating a security group](#) in the *Amazon Virtual Private Cloud User Guide*.

Note

Your VPC automatically comes with a default security group. If you don't specify a different security group or groups when you launch the server, we associate the default security group with your server.

- f. (Optional) For **FIPS Enabled**, select the **FIPS Enabled endpoint** check box to ensure the endpoint complies with Federal Information Processing Standards (FIPS).

Note

FIPS-enabled endpoints are only available in North American AWS Regions. For available Regions, see [AWS Transfer Family endpoints and quotas](#) in the *AWS General Reference*. For more information about FIPS, see [Federal Information Processing Standard \(FIPS\) 140-2](#).

- g. Choose **Next**.

6. In **Configure additional details**, do the following:

- a. For **CloudWatch logging**, choose one of the following to enable Amazon CloudWatch logging of your user activity:
 - **Create a new role** to allow Transfer Family to automatically create the IAM role, as long as you have the right permissions to create a new role. The IAM role that is created is called `AWSTransferLoggingAccess`.
 - **Choose an existing role** to choose an existing IAM role from your account. Under **Logging role**, choose the role. This IAM role should include a trust policy with **Service** set to `transfer.amazonaws.com`.

For more information about CloudWatch logging, see [Log activity with CloudWatch \(p. 122\)](#).

Note

- You can't view end-user activity in CloudWatch if you don't specify a logging role.
- If you don't want to set up a CloudWatch logging role, choose **Choose an existing role**, but don't select a logging role.

- b. For **Cryptographic algorithm options**, choose a security policy that contains the cryptographic algorithms enabled for use by your server.

Note

By default, the `TransferSecurityPolicy-2020-06` security policy is attached to your server unless you choose a different one.

For more information about security policies, see [Working with security policies \(p. 49\)](#).

- c. (Optional) For **Server Host Key**, enter an RSA private key that will be used to identify your server when clients connect to it over SFTP.

Note

This section is only for migrating users from an existing SFTP-enabled server.

- d. (Optional) For **Tags**, for **Key** and **Value**, enter one or more tags as key-value pairs, and then choose **Add tag**.
- e. Choose **Next**.

7. In **Review and create**, review your choices. If you:

- Want to edit any of them, choose **Edit** next to the step.

Note

You will need to review each step after the step you chose to edit.

- Have no changes, choose **Create server** to create your server. You are taken to the **Servers** page, shown following, where your new server is listed.

It can take a couple of minutes before the status for your new server changes to **Online**. At that point, your server can perform file operations for your users.

Create an internet-facing endpoint for your server

In the following procedure, you create a server endpoint. This endpoint is accessible over the internet only to clients whose source IP addresses are allowed in your VPC's default security group. Additionally, by using Elastic IP addresses to make your endpoint internet-facing, your clients can use the Elastic IP address to allow access to your endpoint in their firewalls.

Note

Only SFTP and FTPS can be used on an internet-facing VPC hosted endpoint.

To create an internet-facing endpoint

1. Open the AWS Transfer Family console at <https://console.aws.amazon.com/transfer/>.
2. From the navigation pane, select **Servers**, then choose **Create server**.
3. In **Choose protocols**, select one or more protocols, and then choose **Next**. For more information about protocols, see [Step 2: Create an SFTP-enabled server \(p. 15\)](#).
4. In **Choose an identity provider**, choose **Service managed** to store user identities and keys in AWS Transfer Family, and then choose **Next**.

Note

This procedure uses the service-managed option. If you choose **Custom**, you provide an Amazon API Gateway endpoint and an AWS Identity and Access Management (IAM) role to access the endpoint. By doing so, you can integrate your directory service to authenticate and authorize your users. To learn more about working with custom identity providers, see [Working with custom identity providers \(p. 69\)](#).

5. In **Choose an endpoint**, do the following:
 - a. For **Endpoint type**, choose the **VPC hosted** endpoint type to host your server's endpoint.
 - b. For **Access**, choose **Internet Facing** to make your endpoint accessible to clients over the internet.

Note

When you choose **Internet Facing**, you can choose an existing Elastic IP address in each subnet or subnets. Or you can go to the VPC console (<https://console.aws.amazon.com/vpc/>) to allocate one or more new Elastic IP addresses. These addresses can be owned either by AWS or by you. You can't associate Elastic IP addresses that are already in use with your endpoint.

- c. (Optional) For **Custom hostname**, choose one of the following:
 - **Amazon Route 53 DNS alias** – if the hostname that you want to use is registered with Route 53. You can then enter the hostname.
 - **Other DNS** – if the hostname that you want to use is registered with another DNS provider. You can then enter the hostname.
 - **None** – to use the server's endpoint and not use a custom hostname. The server hostname takes the form *serverId.server.transfer.regionId.amazonaws.com*.

To learn more about working with custom hostnames, see [Working with custom hostnames \(p. 47\)](#).

- d. For **VPC**, choose an existing VPC ID or choose **Create a VPC** to create a new VPC.
- e. In the **Availability Zones** section, choose up to three Availability Zones and associated subnets. For **IPv4 Addresses**, choose an **Elastic IP address** for each subnet. This is the IP address that your clients can use to allow access to your endpoint in their firewalls.
- f. In the **Security Groups** section, choose an existing security group ID or IDs or choose **Create a security group** to create a new security group. For more information about security groups, see [Security groups for your VPC](#) in the *Amazon Virtual Private Cloud User Guide*. To create a security group, see [Creating a security group](#) in the *Amazon Virtual Private Cloud User Guide*.

Note

Your VPC automatically comes with a default security group. If you don't specify a different security group or groups when you launch the server, we associate the default security group with your server.

- g. (Optional) For **FIPS Enabled**, select the **FIPS Enabled endpoint** check box to ensure the endpoint complies with Federal Information Processing Standards (FIPS).

Note

FIPS-enabled endpoints are only available in North American AWS Regions. For available Regions, see [AWS Transfer Family endpoints and quotas](#) in the *AWS General Reference*. For more information about FIPS, see [Federal Information Processing Standard \(FIPS\) 140-2](#).

- h. Choose **Next**.

6. In **Configure additional details**, do the following:

- a. For **CloudWatch logging**, choose one of the following to enable Amazon CloudWatch logging of your user activity:
 - **Create a new role** to allow Transfer Family to automatically create the IAM role, as long as you have the right permissions to create a new role. The IAM role that is created is called `AWSTransferLoggingAccess`.
 - **Choose an existing role** to choose an existing IAM role from your account. Under **Logging role**, choose the role. This IAM role should include a trust policy with **Service** set to `transfer.amazonaws.com`.

For more information about CloudWatch logging, see [Log activity with CloudWatch \(p. 122\)](#).

Note

- You can't view end-user activity in CloudWatch if you don't specify a logging role.
- If you don't want to set up a CloudWatch logging role, choose **Choose an existing role**, but don't select a logging role.

- b. For **Cryptographic algorithm options**, choose a security policy that contains the cryptographic algorithms enabled for use by your server.

Note

By default, the `TransferSecurityPolicy-2020-06` security policy is attached to your server unless you choose a different one.

For more information about security policies, see [Working with security policies \(p. 49\)](#).

- c. (Optional) For **Server Host Key**, enter an RSA private key that will be used to identify your server when clients connect to it over SFTP.

Note

This section is only for migrating users from an existing SFTP-enabled server.

- d. (Optional) For **Tags**, for **Key** and **Value**, enter one or more tags as key-value pairs, and then choose **Add tag**.
- e. Choose **Next**.
- f. (Optional) For **Post-upload file-processing WorkflowId**, enter a workflow ID and a corresponding role that Transfer should assume when executing the workflow.

The screenshot shows the 'Post-upload processing' configuration screen. At the top, there's a title 'Post-upload processing' with a 'Info' link. Below it, there are two main sections: 'Workflow' and 'Role'. The 'Workflow' section contains a dropdown menu set to 'w-abcdef01234567890', a refresh button, and a 'Create a new Workflow' button. The 'Role' section contains a dropdown menu set to 'transfer-workflows' and a refresh button. The background of the page is light gray.

7. In **Review and create**, review your choices. If you:

- Want to edit any of them, choose **Edit** next to the step.

Note

You will need to review each step after the step you chose to edit.

- Have no changes, choose **Create server** to create your server. You are taken to the **Servers** page, shown following, where your new server is listed.

You can choose the server ID to see the detailed settings of the server that you just created. After the column **Public IPv4 address** has been populated, the Elastic IP addresses that you provided are successfully associated with your server's endpoint.

Note

When your server in a VPC is online, only the subnets can be modified and only through the [UpdateServer](#) API. You must [stop the server \(p. 115\)](#) to add or change the server endpoint's Elastic IP addresses.

Change the endpoint type for your server

If you have an existing server that is accessible over the internet (that is, has a public endpoint type), you can change its endpoint to a VPC endpoint.

Note

If you have an existing server in a VPC displayed as `VPC-ENDPOINT`, we recommend that you modify it to the new VPC endpoint type. With this new endpoint type, you no longer need to use a Network Load Balancer (NLB) to associate Elastic IP addresses with your server's endpoint. Also, you can use VPC security groups to restrict access to your server's endpoint. However, you can continue to use the `VPC-ENDPOINT` endpoint type as needed.

The following procedure assumes that you have a server that uses either the current public endpoint type or the older `VPC-ENDPOINT` type.

To change the endpoint type for your server

- Open the AWS Transfer Family console at <https://console.aws.amazon.com/transfer/>.

2. In the navigation pane, choose **Servers**.
3. Select the check box of the server that you want to change the endpoint type for.

Important

You must stop the server before you can change its endpoint.

4. For **Actions**, choose **Stop**.
5. In the confirmation dialog box that appears, choose **Stop** to confirm that you want to stop the server.

Note

Before proceeding to the next step, in **Endpoint details**, wait for the **Status** of the server to change to **Offline**; this can take a couple of minutes. You might have to choose **Refresh** on the **Servers** page to see the status change.

You won't be able to make any edits until the server is **Offline**.

6. In **Endpoint details**, choose **Edit**.
7. In **Edit endpoint configuration**, do the following:

- a. For **Edit endpoint type**, choose **VPC hosted**.
- b. For **Access**, choose one of the following:

- **Internal** to make your endpoint only accessible to clients using the endpoint's private IP addresses.
- **Internet Facing** to make your endpoint accessible to clients over the public internet.

Note

When you choose **Internet Facing**, you can choose an existing Elastic IP address in each subnet or subnets. Or, you can go to the VPC console (<https://console.aws.amazon.com/vpc/>) to allocate one or more new Elastic IP addresses. These addresses can be owned either by AWS or by you. You can't associate Elastic IP addresses that are already in use with your endpoint.

- c. (Optional for internet facing access only) For **Custom hostname**, choose one of the following:
 - **Amazon Route 53 DNS alias** – if the hostname that you want to use is registered with Route 53. You can then enter the hostname.
 - **Other DNS** – if the hostname that you want to use is registered with another DNS provider. You can then enter the hostname.
 - **None** – to use the server's endpoint and not use a custom hostname. The server hostname takes the form `serverId.server.transfer.regionId.amazonaws.com`.

To learn more about working with custom hostnames, see [Working with custom hostnames \(p. 47\)](#).

- d. For **VPC**, choose an existing VPC ID, or choose **Create a VPC** to create a new VPC.
- e. In the **Availability Zones** section, select up to three Availability Zones and associated subnets. If **Internet Facing** is chosen, also choose an Elastic IP address for each subnet.

Note

If you want the maximum of three Availability Zones, but there are not enough available, create them in the VPC console (<https://console.aws.amazon.com/vpc/>).

If you modify the subnets or Elastic IP addresses, the server takes a few minutes to update. You can't save your changes until the server update is complete.

- f. Choose **Save**.

8. For **Actions**, choose **Start** and wait for the status of the server to change to **Online**; this can take a couple of minutes.

Note

If you changed a public endpoint type to a VPC endpoint type, notice that **Endpoint type** for your server has changed to **VPC**.

The default security group is attached to the endpoint. To change or add additional security groups, see [Creating Security Groups](#).

Discontinuing the use of VPC_ENDPOINT

AWS Transfer Family is discontinuing the ability to create servers with `EndpointType=VPC_ENDPOINT` for new AWS accounts. As of May 19, 2021, AWS accounts that don't own AWS Transfer Family servers with an endpoint type of `VPC_ENDPOINT` will not be able to create new servers with `EndpointType=VPC_ENDPOINT`. If you already own servers that use the `VPC_ENDPOINT` endpoint type, we recommend that you start using `EndpointType=VPC` as soon as possible. For details, see [Update your AWS Transfer Family server endpoint type from VPC_ENDPOINT to VPC](#).

We launched the new VPC endpoint type earlier in 2020. For more information, see [AWS Transfer Family for SFTP supports VPC Security Groups and Elastic IP addresses](#). This new endpoint is more feature rich and cost effective and there are no PrivateLink charges. For more information, see [AWS PrivateLink pricing](#).

This endpoint type is functionally equivalent to the previous endpoint type (`VPC_ENDPOINT`). You can attach Elastic IP addresses directly to the endpoint to make it internet facing and use security groups for source IP filtering. For more information, see [Use IP allow listing to secure your AWS Transfer Family for SFTP servers](#).

You can also host this endpoint in a shared VPC environment. For more information, see [AWS Transfer Family now supports shared services VPC environments](#).

In addition to SFTP, you can use the VPC `EndpointType` to enable FTPS and FTP. We don't plan to add these features and FTPS/FTP support to `EndpointType=VPC_ENDPOINT`. We have also removed this endpoint type as an option from the AWS Transfer Family console.

You can change the endpoint type for your server using the Transfer Family console, AWS CLI, API, SDKs, or AWS CloudFormation. To change your server's endpoint type, see [Updating the AWS Transfer Family server endpoint type from VPC_ENDPOINT to VPC \(p. 42\)](#).

If you have any questions, contact AWS Support or your AWS account team.

Note

We do not plan to add these features and FTPS or FTP support to `EndpointType=VPC_ENDPOINT`. We are no longer offering it as an option on the AWS Transfer Family Console.

If you have additional questions, you can contact us through AWS Support or your account team.

Updating the AWS Transfer Family server endpoint type from VPC_ENDPOINT to VPC

You can use the AWS Management Console, AWS CloudFormation, or the Transfer Family API to update a server's `EndpointType` from `VPC_ENDPOINT` to `VPC`. Detailed procedures and examples for using each of these methods to update a server endpoint type are provided in the following sections. If you have servers in multiple AWS regions and in multiple AWS accounts, you can use the example script provided in the following section, with modifications, to identify servers using the `VPC_ENDPOINT` type that you will need to update.

Topics

- [Identifying servers using the VPC_ENDPOINT endpoint type \(p. 43\)](#)
- [Updating the server endpoint type using the AWS Management Console \(p. 43\)](#)
- [Updating the server endpoint type using AWS CloudFormation \(p. 44\)](#)
- [Updating the server EndpointType using the API \(p. 46\)](#)

Identifying servers using the VPC_ENDPOINT endpoint type

You can identify which servers are using the VPC_ENDPOINT using the AWS Management Console.

To identify servers using the VPC_ENDPOINT endpoint type using the console

1. Open the AWS Transfer Family console at <https://console.aws.amazon.com/transfer/>.
2. Choose **Servers** in the navigation pane to display the list of servers in your account in that region.
3. Sort the list of servers by the **Endpoint type** to see all servers using VPC_ENDPOINT.

To identify servers using VPC_ENDPOINT across multiple AWS Regions and accounts

If you have servers in multiple AWS regions and in multiple AWS accounts, you can use the following example script, with modifications, to identify servers using the VPC_ENDPOINT endpoint type. The example script uses the Amazon EC2 [DescribeRegions](#) and the Transfer Family [ListServers \(p. 226\)](#) API calls to get a list of the server IDs and regions of all your servers using VPC_ENDPOINT. If you have many AWS accounts, you could loop through your accounts using an IAM Role with read only auditor access if you authenticate using session profiles to your identity provider.

1. Following is a simple example.

```
import boto3

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

ec2 = session.client("ec2")

regions = ec2.describe_regions()

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

2. Once you have the list of the servers to update, you can use one of the methods described in the following sections to update the **EndpointType** to **VPC**.

Updating the server endpoint type using the AWS Management Console

1. Open the AWS Transfer Family console at <https://console.aws.amazon.com/transfer/>.
2. In the navigation pane, choose **Servers**.
3. Select the check box of the server that you want to change the endpoint type for.

Important

You must stop the server before you can change its endpoint.

4. For **Actions**, choose **Stop**.
5. In the confirmation dialog box that appears, choose **Stop** to confirm that you want to stop the server.

Note

Before proceeding to the next step, wait for the **Status** of the server to change to **Offline**; this can take a couple of minutes. You might have to choose **Refresh** on the **Servers** page to see the status change.

6. Once the status is **Offline**, Choose the server to display the server details page.
7. In the **Endpoint details** choose **Edit**.
8. Choose **VPC hosted** for the **Endpoint type**.
9. Choose **Save**.
10. For **Actions**, choose **Start** and wait for the status of the server to change to **Online**; this can take a couple of minutes.

Updating the server endpoint type using AWS CloudFormation

This section describes how to use AWS CloudFormation to update a server's `EndpointType` to `VPC`. Use this procedure for Transfer Family servers that you have deployed using AWS CloudFormation. In this example, the original AWS CloudFormation template used to deploy the Transfer Family server is shown as follows:

```
AWSTemplateFormatVersion: '2010-09-09'
Description: 'Create AWS Transfer Server with VPC_ENDPOINT endpoint type'
Parameters:
  SecurityGroupId:
    Type: AWS::EC2::SecurityGroup::Id
  SubnetIds:
    Type: List<AWS::EC2::Subnet::Id>
  VpcId:
    Type: AWS::EC2::VPC::Id
Resources:
  TransferServer:
    Type: AWS::Transfer::Server
    Properties:
      Domain: S3
      EndpointDetails:
        VpcEndpointId: !Ref VPCEndpoint
      EndpointType: VPC_ENDPOINT
      IdentityProviderType: SERVICE_MANAGED
      Protocols:
        - SFTP
  VPCEndpoint:
    Type: AWS::EC2::VPCEndpoint
    Properties:
      ServiceName: com.amazonaws.us-east-1.transfer.server
      SecurityGroupIds:
        - !Ref SecurityGroupId
      SubnetIds:
        - !Select [0, !Ref SubnetIds]
        - !Select [1, !Ref SubnetIds]
        - !Select [2, !Ref SubnetIds]
      VpcEndpointType: Interface
      VpcId: !Ref VpcId
```

The template is updated with the following changes:

- The `EndpointType` was changed to `VPC`.
- The `AWS::EC2::VPCEndpoint` resource is removed.
- The `SecurityGroupId`, `SubnetIds`, and `VpcId` were moved to the `EndpointDetails` section of the `AWS::Transfer::Server` resource,

- The `VpcEndpointId` property of `EndpointDetails` was removed.

The updated template looks as follows:

```
AWSTemplateFormatVersion: '2010-09-09'
Description: 'Create AWS Transfer Server with VPC endpoint type'
Parameters:
  SecurityGroupId:
    Type: AWS::EC2::SecurityGroup::Id
  SubnetIds:
    Type: List<AWS::EC2::Subnet::Id>
  VpcId:
    Type: AWS::EC2::VPC::Id
Resources:
  TransferServer:
    Type: AWS::Transfer::Server
    Properties:
      Domain: S3
      EndpointDetails:
        SecurityGroupIds:
          - !Ref SecurityGroupId
        SubnetIds:
          - !Select [0, !Ref SubnetIds]
          - !Select [1, !Ref SubnetIds]
          - !Select [2, !Ref SubnetIds]
        VpcId: !Ref VpcId
      EndpointType: VPC
      IdentityProviderType: SERVICE_MANAGED
      Protocols:
        - SFTP
```

To update the endpoint type of Transfer Family servers deployed using AWS CloudFormation

1. Stop the server that you want to update using the following steps.
 - a. Open the AWS Transfer Family console at <https://console.aws.amazon.com/transfer/>.
 - b. In the navigation pane, choose **Servers**.
 - c. Select the check box of the server that you want to change the endpoint for.

Important
You must stop the server before you can change its endpoint.

 - d. For **Actions**, choose **Stop**.
 - e. In the confirmation dialog box that appears, choose **Stop** to confirm that you want to stop the server.

Note
Before proceeding to the next step, wait for the **Status** of the server to change to **Offline**; this can take a couple of minutes. You might have to choose **Refresh** on the **Servers** page to see the status change.
2. Update the CloudFormation stack
 - a. Open the AWS CloudFormation console at <https://console.aws.amazon.com/cloudformation>.
 - b. Choose the stack used to create the Transfer Family server.
 - c. Choose **Update**.
 - d. Choose **Replace current template**
 - e. Upload the new template. CloudFormation Change Sets help you understand how template changes will affect running resources before you implement them. In this example, the Transfer server resource will be modified, and the VPCEndpoint resource will be removed.

The VPC endpoint type server creates a VPC Endpoint on your behalf, replacing the original VPCEndpoint resource.

After uploading the new template, the change set will look similar to the following:

Change set preview				
Changes (2)				
Action	Logical ID	Physical ID	Resource type	Replacement
Modify	TransferServer	arn:aws:transfer:us-east-1:364810874344:server/s-6a7d04e12d49ec98	AWS::Transfer::Server	Conditional
Remove	VPCEndpoint	vpce-04e685f8702849573	AWS::EC2::VPCEndpoint	-

- f. Update the stack.
3. Once the stack update is complete, navigate to the Transfer Family management console at <https://console.aws.amazon.com/transfer/>.
4. Restart the server. Choose the server you updated in AWS CloudFormation, and then choose **Start** from the **Actions** menu.

Updating the server EndpointType using the API

You can use the [describe-server](#) AWS CLI command, or the [UpdateServer \(p. 262\)](#) API command. The following example script stops the Transfer Family server, updates the EndpointType, removes the VPC_ENDPOINT, and starts the server.

```
import boto3
import time

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

session = boto3.Session(profile_name=profile)

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

group_ids=[]

transfer_description = transfer.describe_server(ServerId=server_id)
if transfer_description['Server']['EndpointType']=='VPC_ENDPOINT':
    transfer_vpc_endpoint = transfer_description['Server']['EndpointDetails'][
        'VpcEndpointId']
    transfer_vpc_endpoint_descriptions =
    ec2.describe_vpc_endpoints(VpcEndpointIds=[transfer_vpc_endpoint])
    for transfer_vpc_endpoint_description in
    transfer_vpc_endpoint_descriptions['VpcEndpoints']:
        subnet_ids=transfer_vpc_endpoint_description['SubnetIds']
```

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

Working with custom hostnames

Your *server host name* is the hostname that your users enter in their clients when they connect to your server. You can use a custom domain that you have registered for your server hostname when you work with AWS Transfer Family. For example, you might use a custom hostname like `mysftpserver.mysubdomain.domain.com`.

To redirect traffic from your registered custom domain to your server endpoint, you can use Amazon Route 53 or any Domain Name System (DNS) provider. Route 53 is the DNS service that AWS Transfer Family natively supports.

Topics

- [Use Amazon Route 53 as your DNS provider \(p. 47\)](#)
- [Use other DNS providers \(p. 48\)](#)
- [Custom hostnames for non-console created servers \(p. 48\)](#)

On the console, you can choose one of these options for setting up a custom hostname:

- **Amazon Route 53 DNS alias** – if the hostname that you want to use is registered with Route 53. You can then enter the hostname.
- **Other DNS** – if the hostname that you want to use is registered with another DNS provider. You can then enter the hostname.
- **None** – to use the server's endpoint and not use a custom hostname.

You set this option when you create a new server or edit the configuration of an existing server. For more information about creating a new server, see [Step 2: Create an SFTP-enabled server \(p. 15\)](#). For more information about editing the configuration of an existing server, see [Edit server details \(p. 109\)](#).

For more details about using your own domain for the server hostname and how AWS Transfer Family uses Route 53, see the following sections.

Use Amazon Route 53 as your DNS provider

When you create a server, you can use Amazon Route 53 as your DNS provider. Before you use a domain with Route 53, you register the domain. For more information, see [How Domain Registration Works](#) in the *Amazon Route 53 Developer Guide*.

When you use Route 53 to provide DNS routing to your server, AWS Transfer Family uses the custom hostname that you entered to extract its hosted zone. When AWS Transfer Family extracts a hosted zone, three things can happen:

1. If you're new to Route 53 and don't have a hosted zone, AWS Transfer Family adds a new hosted zone and a CNAME record. The value of this CNAME record is the endpoint hostname for your server. A CNAME is an alternate domain name.
2. If you have a hosted zone in Route 53 without any CNAME records, AWS Transfer Family adds a CNAME record to the hosted zone.
3. If the service detects that a CNAME record already exists in the hosted zone, you see an error indicating that a CNAME record already exists. In this case, change the value of the CNAME record to the hostname of your server.

Note

If this step is part of a server creation workflow, your server is successfully created and your custom hostname is set to **None**.

For more information about hosted zones in Route 53, see [Hosted Zone](#) in the *Amazon Route 53 Developer Guide*.

Use other DNS providers

When you create a server, you can also use DNS providers other than Amazon Route 53. If you use an alternate DNS provider, make sure that traffic from your domain is directed to your server endpoint.

To do so, set your domain to the endpoint hostname for the server. An endpoint hostname looks like this in the console:

`serverid.server.transfer.region.amazonaws.com`

Custom hostnames for non-console created servers

When you create a server using AWS Cloud Development Kit (CDK) or through the CLI, you must add tags if you want that server to have a custom hostname. When you create a Transfer Family server by using the console, the tagging is done automatically.

Note

You also need to create a DNS record to redirect traffic from your domain to your server endpoint.

To add the necessary tags, you need to add 2 keys, one for `aws:transfer:customHostname`, and one for `aws:transfer:rout53HostedZoneId`.

```
"Tags": [  
  {  
    "Key": "aws:transfer:rout53HostedZoneId",  
    "Value": "/hostedzone/HOSTED-ZONE-ID"  
  },  
  {  
    "Key": "aws:transfer:customHostname",  
    "Value": "abc.example.com"  
  }]
```

where `HOSTED-ZONE-ID` is your hosted zone ID, and `abc.example.com` is the custom hostname that you want to use.

Note

Your public, hosted zones and their IDs are available on Amazon Route 53.

Sign in to the AWS Management Console and open the Route 53 console at <https://console.aws.amazon.com/route53/>.

To add these tags, you can use the console or issue a CLI command.

Console

To add Key-Value pair tags for a custom hostname

1. Open the AWS Transfer Family console at <https://console.aws.amazon.com/transfer/>.
2. In the **Tags** section, select **Manage tags**.
3. On the **Manage tags** screen, select **Add tag**.
4. Enter your key and value for the hosted zone ID and click **Save**. Repeat this step to add the key and value for the custom hostname.

API

To add the key-value pairs using the API, issue the following commands:

```
aws transfer tag-resource --arn "server-ARN:server/server-ID" --tags  
Key="aws:transfer:route53HostedZoneId",Value=/hostedzone/HOSTED-ZONE-ID"  
aws transfer tag-resource --arn "server-ARN:server/server-ID" --tags  
Key="aws:transfer:customHostname",Value="abc.example.com"
```

Working with security policies

Server security policies in AWS Transfer Family allow you to limit the set of cryptographic algorithms (message authentication codes (MACs), key exchanges (KEXs), and cipher suites) associated with your server. For a list of supported cryptographic algorithms, see [Cryptographic algorithms \(p. 49\)](#).

Note

We support TLS 1.2.

The available security policies for the server are:

- [TransferSecurityPolicy-2020-06 \(p. 51\)](#)
- [TransferSecurityPolicy-2018-11 \(p. 52\)](#)
- [TransferSecurityPolicy-FIPS-2020-06 \(p. 53\)](#)

Note

- [TransferSecurityPolicy-2020-06](#) is the default security policy attached to your server when creating a server using the console.
- [TransferSecurityPolicy-2018-11](#) is the default security policy attached to your server when creating a server using the API or CLI.

Cryptographic algorithms

The following is a list of supported cryptographic algorithms for each security policy.

Security policy	2020-06	FIPS-2020-06	2018-11
SSH ciphers			
aes128-ctr	◆	◆	◆
aes128-gcm@openssh.com	◆	◆	◆
aes192-ctr	◆	◆	◆
aes256-ctr	◆	◆	◆
aes256-gcm@openssh.com	◆	◆	◆
chacha20-poly1305@openssh.com	◆		◆
KEXs			
diffie-hellman-group14-sha256	◆	◆	◆
diffie-hellman-group16-sha512	◆	◆	◆
diffie-hellman-group18-sha512	◆	◆	◆
ecdh-sha2-nistp384	◆	◆	◆
ecdh-sha2-nistp521	◆	◆	◆
diffie-hellman-group-exchange-sha256	◆	◆	◆
diffie-hellman-group14-sha1			◆
ecdh-sha2-nistp256	◆	◆	◆
curve25519-sha256@libssh.org			◆
curve25519-sha256			◆
MACs			
hmac-sha2-256-etm@openssh.com	◆	◆	◆
hmac-sha2-256	◆	◆	◆
hmac-sha2-512-etm@openssh.com	◆	◆	◆
hmac-sha2-512	◆	◆	◆
hmac-sha1-etm@openssh.com			◆
hmac-sha1			◆

Security policy	2020-06	FIPS-2020-06	2018-11
umac-128- etm@openssh.com	◆		◆
umac-128@openssh.com	◆		◆
umac-64- etm@openssh.com			◆
umac-64@openssh.com			◆
TLS ciphers			
TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256	◆		◆
TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256	◆		◆
TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384	◆		◆
TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384	◆		◆
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256	◆		◆
TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256	◆		◆
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384	◆		◆
TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384	◆		◆
TLS_RSA_WITH_AES_128_CBC_SHA256			◆
TLS_RSA_WITH_AES_256_CBC_SHA256			◆

TransferSecurityPolicy-2020-06

The following shows the TransferSecurityPolicy-2020-06 security policy. This security policy contains all supported, but non-deprecated cryptographic algorithms. This is the default security policy for non-FIPS enabled server endpoints.

```
{
  "SecurityPolicy": {
    "Fips": false,
    "SecurityPolicyName": "TransferSecurityPolicy-2020-06",
    "SshCiphers": [
      "chacha20-poly1305@openssh.com",
      "aes128-ctr",
      "aes192-ctr",
      "aes256-ctr",
      "aes128-gcm@openssh.com",
      "aes256-gcm@openssh.com"
    ],
    "SshKexs": [
      "ecdh-sha2-nistp256",
      "ecdh-sha2-nistp384",
      "ecdh-sha2-nistp521",
      "diffie-hellman-group-exchange-sha256",
      "diffie-hellman-group16-sha512",
      "diffie-hellman-group18-sha512",
      "diffie-hellman-group14-sha256"
    ],
    "SshMacs": [
      ...
    ]
  }
}
```

```

        "umac-128-etm@openssh.com",
        "hmac-sha2-256-etm@openssh.com",
        "hmac-sha2-512-etm@openssh.com",
        "umac-128@openssh.com",
        "hmac-sha2-256",
        "hmac-sha2-512"
    ],
    "TlsCiphers": [
        "TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256",
        "TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256",
        "TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256",
        "TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256",
        "TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384",
        "TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384",
        "TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384",
        "TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384"
    ]
}
}

```

TransferSecurityPolicy-2018-11

The following shows the TransferSecurityPolicy-2018-11 security policy. This security policy is the current set of all supported cryptographic algorithms.

```

{
    "SecurityPolicy": {
        "Fips": false,
        "SecurityPolicyName": "TransferSecurityPolicy-2018-11",
        "SshCiphers": [
            "chacha20-poly1305@openssh.com",
            "aes128-ctr",
            "aes192-ctr",
            "aes256-ctr",
            "aes128-gcm@openssh.com",
            "aes256-gcm@openssh.com"
        ],
        "SshKexs": [
            "curve25519-sha256",
            "curve25519-sha256@libssh.org",
            "ecdh-sha2-nistp256",
            "ecdh-sha2-nistp384",
            "ecdh-sha2-nistp521",
            "diffie-hellman-group-exchange-sha256",
            "diffie-hellman-group16-sha512",
            "diffie-hellman-group18-sha512",
            "diffie-hellman-group14-sha256",
            "diffie-hellman-group14-sha1"
        ],
        "SshMacs": [
            "umac-64-etm@openssh.com",
            "umac-128-etm@openssh.com",
            "hmac-sha2-256-etm@openssh.com",
            "hmac-sha2-512-etm@openssh.com",
            "hmac-sha1-etm@openssh.com",
            "umac-64@openssh.com",
            "umac-128@openssh.com",
            "hmac-sha2-256",
            "hmac-sha2-512",
            "hmac-sha1"
        ],
        "TlsCiphers": [
            "TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256",

```

```

        "TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256",
        "TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256",
        "TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256",
        "TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384",
        "TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384",
        "TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384",
        "TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384",
        "TLS_RSA_WITH_AES_128_CBC_SHA256",
        "TLS_RSA_WITH_AES_256_CBC_SHA256"
    ],
}
}

```

TransferSecurityPolicy-FIPS-2020-06

The following shows the TransferSecurityPolicy-FIPS-2020-06 security policy. This security policy contains all supported FIPS compliant cryptographic algorithms. This is the default security policy for FIPS enabled server endpoints.

Note

The FIPS service endpoint and TransferSecurityPolicy-FIPS-2020-06 security policy is only available in some AWS Regions. For more information, see [AWS Transfer Family endpoints and quotas](#) in the *AWS General Reference*.

```
{
  "SecurityPolicy": {
    "Fips": true,
    "SecurityPolicyName": "TransferSecurityPolicy-FIPS-2020-06",
    "SshCiphers": [
      "aes128-ctr",
      "aes192-ctr",
      "aes256-ctr",
      "aes128-gcm@openssh.com",
      "aes256-gcm@openssh.com"
    ],
    "SshKexs": [
      "ecdh-sha2-nistp256",
      "ecdh-sha2-nistp384",
      "ecdh-sha2-nistp521",
      "diffie-hellman-group-exchange-sha256",
      "diffie-hellman-group16-sha512",
      "diffie-hellman-group18-sha512",
      "diffie-hellman-group14-sha256"
    ],
    "SshMacss": [
      "hmac-sha2-256-etm@openssh.com",
      "hmac-sha2-512-etm@openssh.com",
      "hmac-sha2-256",
      "hmac-sha2-512"
    ],
    "TlsCiphers": [
      "TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256",
      "TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256",
      "TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256",
      "TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256",
      "TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384",
      "TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384",
      "TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384",
      "TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384"
    ]
  }
}
```

Managing users

In the following sections, you can find information about how to add users using AWS Transfer Family, AWS Directory Service for Microsoft Active Directory or a custom identity provider.

If you use a service-managed identity type, you add users to your file transfer protocol enabled server. When you do so, each user name must be unique on your server.

As part of each user's properties, you also store that user's Secure Shell (SSH) public key. Doing so is required for key based authentication, which this procedure uses. The private key is stored locally on your user's computer. When your user sends an authentication request to your server by using a client, your server first confirms that the user has access to the associated SSH private key. The server then successfully authenticates the user.

In addition, you specify a user's home directory, or landing directory, and assign an AWS Identity and Access Management (IAM) role to the user. Optionally, you can provide a session policy to limit user access only to the home directory of your Amazon S3 bucket.

Amazon EFS vs. Amazon S3

Characteristics of each storage option:

- To limit access: Amazon S3 supports session policies; Amazon EFS supports POSIX user, group, and secondary group IDs
- Both support public/private keys
- Both support home directories
- Both support logical directories

Note

For Amazon S3, most of the support for logical directories is via API/CLI. You can use the **Restricted** check box in the console to lock down a user to their home directory, but you cannot specify a virtual directory structure.

Topics

- [Working with service-managed users \(p. 54\)](#)
- [Using the AWS Directory Service identity provider \(p. 60\)](#)
- [Working with custom identity providers \(p. 69\)](#)
- [Tutorial: Setting up an Amazon API Gateway method as a custom identity provider \(p. 81\)](#)

Working with service-managed users

You can add either Amazon S3 or Amazon EFS service-managed users to your server, depending on the server's **Domain** setting. For more information, see [Creating a server \(p. 19\)](#).

Topics

- [Adding Amazon S3 service-managed users \(p. 55\)](#)
- [Adding Amazon EFS service-managed users \(p. 56\)](#)
- [Managing service-managed users \(p. 57\)](#)

Adding Amazon S3 service-managed users

To add an Amazon S3 service-managed user to your server

1. Open the AWS Transfer Family console at <https://console.aws.amazon.com/transfer/>, then select **Servers** from the navigation pane.
2. On the **Servers** page, select the check box of the server that you want to add a user to.
3. Choose **Add user**.
4. In the **User configuration** section, for **Username**, enter the user name. This user name must be a minimum of 3 and a maximum of 100 characters. You can use the following characters in the user name: a-z, A-Z, 0-9, underscore '_', hyphen '-', period '.', and at sign "@". The user name can't start with a hyphen '-', period '.', or at sign "@".
5. For **Access**, choose the IAM role that you previously created that provides access to your Amazon S3 bucket.

You created this IAM role using the procedure in [Create an IAM role and policy \(p. 8\)](#). That IAM role includes an IAM policy that provides access to your Amazon S3 bucket. It also includes a trust relationship with the AWS Transfer Family service, defined in another IAM policy.

6. (Optional) For **Policy**, choose one of the following:
 - **None**
 - **Existing policy**
 - **Select a policy from IAM** to choose an existing policy. Choose **View** to see a JSON object containing the details of the policy.

To learn more about session policies, see [Create an IAM role and policy \(p. 8\)](#). To learn more about creating a session policy, see [Creating a session policy for an Amazon S3 bucket \(p. 119\)](#).

7. For **Home directory**, choose the Amazon S3 bucket to store the data to transfer using AWS Transfer Family. Enter the path to the home directory where your user lands when they log in using their client.

If you keep this parameter blank, the root directory of your Amazon S3 bucket is used. In this case, make sure that your IAM role provides access to this root directory.

Note

We recommend that you choose a directory path that contains the user name of the user, which enables you to effectively use a session policy. The session policy limits user access in the Amazon S3 bucket to that user's home directory.

8. (Optional) For **Restricted**, select the check box so that your users can't access anything outside of that folder and can't see the Amazon S3 bucket or folder name.

Note

Assigning the user a home directory and restricting the user to that home directory should be sufficient to lock down the user's access to the designated folder. If you need to apply further controls, use a session policy.

9. For **SSH public key**, enter the public SSH key portion of the SSH key pair.

Your key is validated by the service before you can add your new user.

Important

The format of the SSH public key is ssh-rsa <string>. For instructions on how to generate an SSH key pair, see [Generate SSH keys \(p. 127\)](#).

10. (Optional) For **Key** and **Value**, enter one or more tags as key-value pairs, and choose **Add tag**.
11. Choose **Add** to add your new user to the server that you chose.

The new user appears in the **Users** section of the **Server details** page.

Next steps – For the next step, continue on to [Transferring files using a client \(p. 90\)](#).

Adding Amazon EFS service-managed users

Amazon EFS uses the Portable Operating System Interface (POSIX) file permission model to represent file ownership.

- For more details on Amazon EFS file ownership, see [Amazon EFS file ownership \(p. 7\)](#).
- For more details on setting up directories for your EFS users, see [Set up Amazon EFS users for Transfer Family \(p. 7\)](#).

To add an Amazon EFS service-managed user to your server

1. Open the AWS Transfer Family console at <https://console.aws.amazon.com/transfer/>, then select **Servers** from the navigation pane.
2. On the **Servers** page, select the Amazon EFS server that you want to add a user to.
3. Choose **Add user** to display the **Add user** page.
4. In the **User configuration** section, use the following settings.
 - a. For **Username**, enter the user name. This user name must be a minimum of 3 and a maximum of 100 characters. You can use the following characters in the user name: a-z, A-Z, 0-9, underscore '_', hyphen '-', period '.', and at sign '@'. The user name can't start with a hyphen '-', period '.', or at sign '@'.
 - b. For **User ID** and **Group ID**, note the following:
 - For the first user that you create, we recommend that you enter a value of 0 for both **Group ID** and **User ID**. This grants the user administrator privileges for Amazon EFS.
 - For additional users, enter the user's POSIX user ID and group ID. These IDs are used for all Amazon Elastic File System operations performed by the user.
 - For **User ID** and **Group ID**, do not use any leading zeroes. For example, **12345** is acceptable, **012345** is not.
 - c. (Optional) For **Secondary Group IDs**, enter one or more additional POSIX group IDs for each user, separated by commas.
 - d. For **Access**, choose the IAM role that:
 - Gives the user access to only the Amazon EFS resources (file systems) that you want them to access.
 - Defines which file system operations that the user can and cannot perform.

We recommend that you use the IAM role for Amazon EFS file system selection with mount access and read/write permissions. For example, the combination of the following two AWS managed policies, while quite permissive, grants the necessary permissions for your user:

- **AmazonElasticFileSystemClientFullAccess**
- **AWSTransferConsoleFullAccess**

For more information, see the blog post [New – AWS Transfer Family support for Amazon Elastic File System](#).

- e. For **Home directory**, do the following:

- Choose the Amazon EFS file system that you want to use for storing the data to transfer using AWS Transfer Family.
- Decide whether to set the home directory to **Restricted**. Setting the home directory to **Restricted** has the following effects:
 - Amazon EFS users can't access any files or directories outside of that folder.
 - Amazon EFS users can't see the Amazon EFS file system name (**fs-xxxxxx**).

Note

When you select the **Restricted** option, symlinks don't resolve for Amazon EFS users.

- (Optional) Enter the path to the home directory that you want users to be in when they log in using their client.

If you don't specify a home directory, the root directory of your Amazon EFS file system is used. In this case, make sure that your IAM role provides access to this root directory.

5. For **SSH public key**, enter the public SSH key portion of the SSH key pair.

Your key is validated by the service before you can add your new user.

Important

The format of the SSH public key is `ssh-rsa <string>`. For instructions on how to generate an SSH key pair, see [Generate SSH keys \(p. 127\)](#).

6. (Optional) Enter any tags for the user. For **Key** and **Value**, enter one or more tags as key-value pairs, and choose **Add tag**.
7. Choose **Add** to add your new user to the server that you chose.

The new user appears in the **Users** section of the **Server details** page.

Issues that you might encounter when you first SFTP to your Transfer Family server:

- If you run the `sftp` command and the prompt doesn't appear, you might encounter the following message:

```
Couldn't canonicalize: Permission denied
```

```
Need cwd
```

In this case, you must increase the policy permissions for your user's role. You can add an AWS managed policy, such as `AmazonElasticFileSystemClientFullAccess`.

- If you enter `pwd` at the `sftp` prompt to view the user's home directory, you might see the following message, where `user-home-directory` is the home directory for the SFTP user.:

```
remote readdir("/user-home-directory"): No such file or directory
```

In this case, you should be able to navigate to the parent directory (`cd ..`), and create the user's home directory (`mkdir username`).

Next steps – For the next step, continue on to [Transferring files using a client \(p. 90\)](#).

Managing service-managed users

In this section, you can find information about how to view a list of users, how to edit user details, and how to add an SSH public key.

- [View a list of users \(p. 58\)](#)
- [View or edit user details \(p. 58\)](#)
- [Delete a user \(p. 59\)](#)
- [Edit an SSH public key \(p. 59\)](#)

To find a list of your users

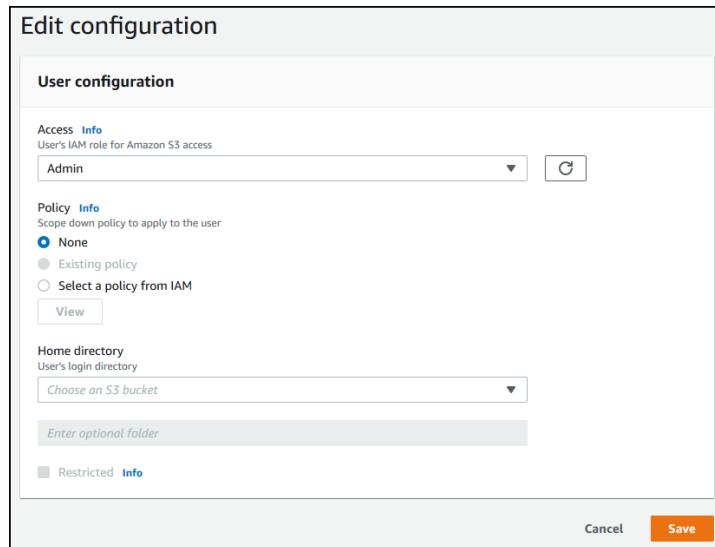
1. Open the AWS Transfer Family console at <https://console.aws.amazon.com/transfer/>.
2. Select **Servers** from the navigation pane to display the **Servers** page.
3. Choose the identifier in the **Server ID** column to see the **Server details** page.
4. Under **Users**, view a list of users.

To view or edit user details

1. Open the AWS Transfer Family console at <https://console.aws.amazon.com/transfer/>.
2. Select **Servers** from the navigation pane to display the **Servers** page.
3. Choose the identifier in the **Server ID** column to see the **Server details** page.
4. Under **Users**, choose a user name to see the **User details** page.

You can change the user's properties on this page by choosing **Edit**.

5. On the **Users details** page, choose **Edit** next to **User configuration**.



6. On the **Edit configuration** page, for **Access**, choose the IAM role that you previously created that provides access to your Amazon S3 bucket.

You created this IAM role using the procedure in [Create an IAM role and policy \(p. 8\)](#). That IAM role includes an IAM policy that provides access to your Amazon S3 bucket. It also includes a trust relationship with the AWS Transfer Family service, defined in another IAM policy.

7. (Optional) For **Policy**, choose one of the following:

- **None**
- **Existing policy**
- **Select a policy from IAM** to choose an existing policy. Choose **View** to see a JSON object containing the details of the policy.

To learn more about session policies, see [Create an IAM role and policy \(p. 8\)](#). To learn more about creating a session policy, see [Creating a session policy for an Amazon S3 bucket \(p. 119\)](#).

8. For **Home directory**, choose the Amazon S3 bucket to store the data to transfer using AWS Transfer Family. Enter the path to the home directory where your user lands when they log in using their client.

If you leave this parameter blank, the root directory of your Amazon S3 bucket is used. In this case, make sure that your IAM role provides access to this root directory.

Note

We recommend that you choose a directory path that contains the user name of the user, which enables you to effectively use a session policy. The session policy limits user access in the Amazon S3 bucket to that user's home directory.

9. (Optional) For **Restricted**, select the check box so that your users can't access anything outside of that folder and can't see the Amazon S3 bucket or folder name.

Note

When assigning the user a home directory and restricting the user to that home directory, this should be sufficient enough to lock down the user's access to the designated folder. Use a session policy when you need to apply further controls.

10. Choose **Save** to save your changes.

To delete a user

1. Open the AWS Transfer Family console at <https://console.aws.amazon.com/transfer/>.
2. Select **Servers** from the navigation pane to display the **Servers** page.
3. Choose the identifier in the **Server ID** column to see the **Server details** page.
4. Under **Users**, choose a user name to see the **User details** page.
5. On the **Users details** page, choose **Delete** to the right of the user name.
6. In the confirmation dialog box that appears, enter the word **delete**, and then choose **Delete** to confirm that you want to delete the user.

The user is deleted from the **users** list.

To edit an SSH public key for a user

1. Open the AWS Transfer Family console at <https://console.aws.amazon.com/transfer/>.
2. In the navigation pane, choose **Servers**.
3. Choose the identifier in the **Server ID** column to see the **Server details** page.
4. Under **Users**, choose a user name to see the **User details** page.
5. Under **SSH public keys**, you can add or delete an SSH (Secure Shell) public key.
 - To add a public key
 1. Choose **Add SSH public key** to add a new SSH public key to a user.

Note

SSH keys are used only servers that are enabled for Secure Shell (SSH) File Transfer Protocol (SFTP). For information about how to generate an SSH key pair, see [Generate SSH keys \(p. 127\)](#).

2. For **SSH public key**, enter the SSH public key portion of the SSH key pair.

Your key is validated by the service before you can add your new user. The format of the SSH key is `ssh-rsa string`. To generate an SSH key pair, see [Generate SSH keys \(p. 127\)](#).

3. Choose **Add key**.

- To delete a public key, select the SSH key check box and choose **Delete**.

Using the AWS Directory Service identity provider

This topic describes how to use the AWS Directory Service identity provider for AWS Transfer Family.

Topics

- [Using AWS Directory Service for Microsoft Active Directory \(p. 60\)](#)
- [Using AWS Directory Service for Azure Active Directory Domain Services \(p. 65\)](#)

Using AWS Directory Service for Microsoft Active Directory

You can use AWS Transfer Family to authenticate your file transfer end users using AWS Directory Service for Microsoft Active Directory. It enables seamless migration of file transfer workflows that rely on Active Directory authentication without changing end users' credentials or needing a custom authorizer.

With AWS Managed Microsoft AD, you can securely provide AWS Directory Service users and groups access over SFTP, FTPS, and FTP for data stored in Amazon Simple Storage Service (Amazon S3) or Amazon Elastic File System (Amazon EFS). If you use Active Directory to store your users' credentials, you now have an easier way to enable file transfers for these users.

You can provide access to Active Directory groups in AWS Managed Microsoft AD in your on-premises environment or in the AWS Cloud using Active Directory connectors. You can give users that are already configured in your Microsoft Windows environment, either in the AWS Cloud or in their on-premises network, access to an AWS Transfer Family server that uses AWS Managed Microsoft AD for identity.

Note

Transfer Family does not support Simple AD.

To use AWS Managed Microsoft AD, you must perform the following steps:

1. Create one or more AWS Managed Microsoft AD directories using the AWS Directory Service console.
2. Use the Transfer Family console to create a server that uses AWS Managed Microsoft AD as its identity provider.
3. Create access from one or more of your AWS Directory Service groups.
4. Although not required, we recommend that you test and verify user access.

Topics

- [Before you start \(p. 61\)](#)
- [Choosing AWS Managed Microsoft AD as your identity provider \(p. 61\)](#)
- [Granting access to groups \(p. 62\)](#)
- [Testing users \(p. 64\)](#)
- [Deleting server access for a group \(p. 64\)](#)
- [Connecting to the server using SSH \(Secure Shell\) \(p. 65\)](#)

Before you start

Before you can use AWS Managed Microsoft AD, you must provide a unique identifier for each group in your Microsoft AD directory. You can use the security identifier (SID) for each group to do this. The users of the group that you associate have access to your Amazon S3 or Amazon EFS resources over the enabled protocols using AWS Transfer Family.

Use the following Windows PowerShell command to retrieve the SID for a group, replacing *YourGroupName* with the name of the group.

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

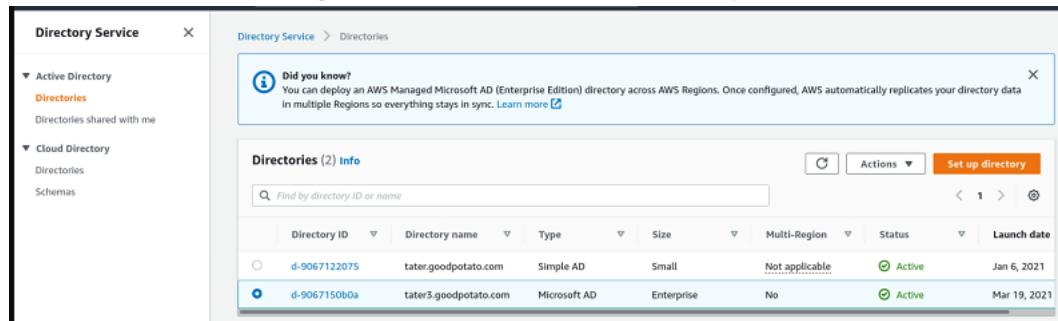
Choosing AWS Managed Microsoft AD as your identity provider

This section describes how to use AWS Directory Service for Microsoft Active Directory with a server.

To use AWS Managed Microsoft AD with Transfer Family

1. Sign in to the AWS Management Console and open the AWS Directory Service console at <https://console.aws.amazon.com/directoryservicev2/>.

Use the AWS Directory Service console to configure one or more managed directories. For more information, see [AWS Managed Microsoft AD](#) in the [AWS Directory Service Admin Guide](#).

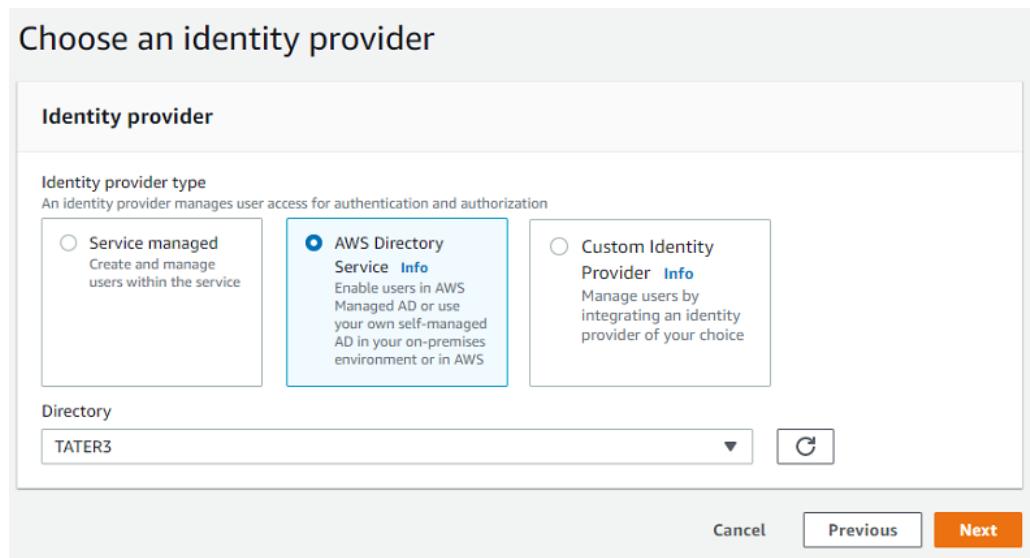


2. Open the AWS Transfer Family console at <https://console.aws.amazon.com/transfer/>, and choose **Create server**.
3. On the **Choose protocols** page, choose one or more protocols from the list.

Note

If you select **FTPS**, you must provide the AWS Certificate Manager certificate.

4. For **Choose an identity provider**, choose **AWS Directory Service**.



5. The **Directory** list contains all the managed directories that you have configured. Choose a directory from the list, and choose **Next**.

Note

Cross-Account and Shared directories are not supported for AWS Managed Microsoft AD.

6. To finish creating the server, use one of the following procedures:

- [Create an SFTP-enabled server \(p. 20\)](#)
- [Create an FTPS-enabled server \(p. 25\)](#)
- [Create an FTP-enabled server \(p. 30\)](#)

In those procedures, continue with the step that follows choosing an identity provider.

Important

You can't delete a Microsoft AD directory in AWS Directory Service if you used it in a Transfer Family server. You must delete the server first, and then you can delete the directory.

Granting access to groups

After you create the server, you must choose which groups in the directory should have access to upload and download files over the enabled protocols using AWS Transfer Family. You do this by creating an *access*.

Note

Users must belong *directly* to the group to which you are granting access. For example, assume that Bob is a user and he belongs to groupA, and groupA itself is included in groupB.

- If you grant access to groupA, Bob is granted access.
- If you grant access to groupB (and not to groupA), Bob does not have access.

To grant access to a group

1. Open the AWS Transfer Family console at <https://console.aws.amazon.com/transfer/>.
2. Navigate to your server details page.
3. In the **Accesses** section, choose **Create access**.

4. Enter the SID for the AWS Managed Microsoft AD directory that you want to have access to this server.

Note

For information about how to find the SID for your group, see [the section called "Before you start" \(p. 61\)](#).

5. For **Access**, choose an AWS Identity and Access Management (IAM) role for the group.
6. In the **Policy** section, choose a policy. The default setting is **None**.
7. For **Home directory**, choose an S3 bucket that corresponds to the group's home directory.

Note

You can limit the portions of the bucket that users see by creating a session policy. For example, to limit users to their own folder under the /filetest directory, enter the following text in the box.

```
/filetest/${transfer:UserName}
```

To learn more about creating a session policy, see [Creating a session policy for an Amazon S3 bucket \(p. 119\)](#).

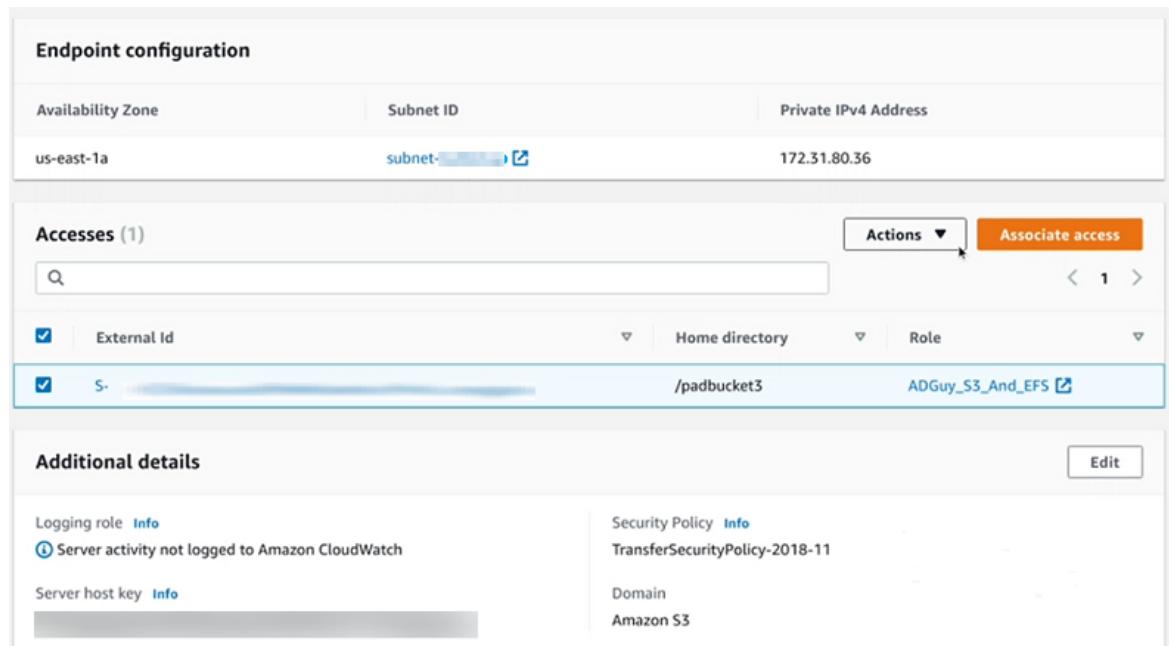
8. Choose **Add** to create the association.
9. Choose your server.
10. Choose **Create access**.
 - Enter the SID for the group.

Note

For information about how to find the SID, see [the section called "Before you start" \(p. 61\)](#).

11. Choose **Create access**.

In the **Accesses** section, the accesses for the server are listed.



Endpoint configuration			
Availability Zone	Subnet ID	Private IPv4 Address	
us-east-1a	subnet-_____	172.31.80.36	

Accesses (1)			
		Actions ▾	Associate access
<input type="checkbox"/> External Id		Home directory	Role
<input checked="" type="checkbox"/> S-		/padbucket3	ADGuy_S3_And_EFS

Additional details	
Logging role Info ⓘ Server activity not logged to Amazon CloudWatch	Security Policy Info TransferSecurityPolicy-2018-11
Server host key Info 	Domain Amazon S3

Testing users

You can test whether a user has access to the AWS Managed Microsoft AD directory for your server.

Note

A user must be in exactly one group (an external ID) that is listed in the **Access** section of the **Endpoint configuration** page. If the user is in no groups, or is in more than a single group, that user is not granted access.

To test whether a specific user has access

1. On the server details page, choose **Actions**, and then choose **Test**.
2. For **Identity provider testing**, enter the user name and password for a user that is in one of the groups that has access.
3. Choose **Test**.

You see a successful identity provider test, showing that the selected user has been granted access to the server.

Identity provider testing

User configuration Info

Username	Password
transferuser1

Response

```
{
  "Response": "{'homeDirectory': '/tmp/abucket31', 'homeDirectoryDetails': 'null', 'homeDirectoryType': '1', 'PATH': '/tmp/posixProfile', 'null', 'publicKeys': 'null', 'role': 'arn:aws:iam::19588E157075:role/MDGuy_SS_And_EFS', 'policy': 'null', 'username': '/tmp/transferuser1', 'IdentityProviderType': 'null', 'userConfigMessage': 'null'}",
  "StatusCode": 200,
  "Message": ""
}
```

Cancel Test

If the user belongs to more than one group that has access, you receive the following response.

```
"Response": "",  
"StatusCode": 200,  
"Message": "More than one associated access found for user's groups."
```

Deleting server access for a group

To delete server access for a group

1. On the server details page, choose **Actions**, and then choose **Delete Access**.

2. In the dialog box, confirm that you want to remove access for this group.

When you return to the server details page, you see that the access for this group is no longer listed.

Connecting to the server using SSH (Secure Shell)

After you configure your server and users, you can connect to the server using SSH and use the fully qualified user name for a user that has access.

```
sftp user@active-directory-domain@vpc-endpoint
```

For example: transferuserexample@mycompany.com@vpce-0123456abcdef-789xyz.vpc-svc-987654zyxabc.us-east-1.vpce.amazonaws.com

This format targets the search of the federation, limiting the search of a potentially large Active Directory.

Note

You can specify the simple user name. However, in this case, the Active Directory code has to search all the directories in the federation. This might limit the search, and authentication might fail even if the user should have access.

After authenticating, the user is located in the home directory that you specified when you configured the user.

Using AWS Directory Service for Azure Active Directory Domain Services

- To take advantage of your existing Active Directory forest for your SFTP Transfer needs, you can use [Active Directory Connector](#).
- If you want the benefits of Active Directory and high availability in a fully managed service, you can use AWS Directory Service for Microsoft Active Directory. For details, see [Using the AWS Directory Service identity provider \(p. 60\)](#).

This topic describes how to use an Active Directory Connector and [Azure Active Directory Domain Services \(Azure ADDS\)](#) to authenticate SFTP Transfer users with [Azure Active Directory](#).

Topics

- [Before you start \(p. 65\)](#)
- [Step 1: Adding Azure Active Directory Domain Services \(p. 66\)](#)
- [Step 2: Creating a service account \(p. 67\)](#)
- [Step 3: Setting up AWS Directory using AD Connector \(p. 67\)](#)
- [Step 4: Setting up AWS Transfer Family server \(p. 68\)](#)
- [Step 5: Granting access to groups \(p. 68\)](#)
- [Step 6: Testing users \(p. 69\)](#)

Before you start

For AWS, you need the following:

- A virtual private cloud (VPC) in an AWS region where you are using your Transfer Family servers
- At least two private subnets in your VPC
- The VPC must have internet connectivity
- A customer gateway and Virtual private gateway for site-to-site VPN connection with Microsoft Azure

For Microsoft Azure, you need the following:

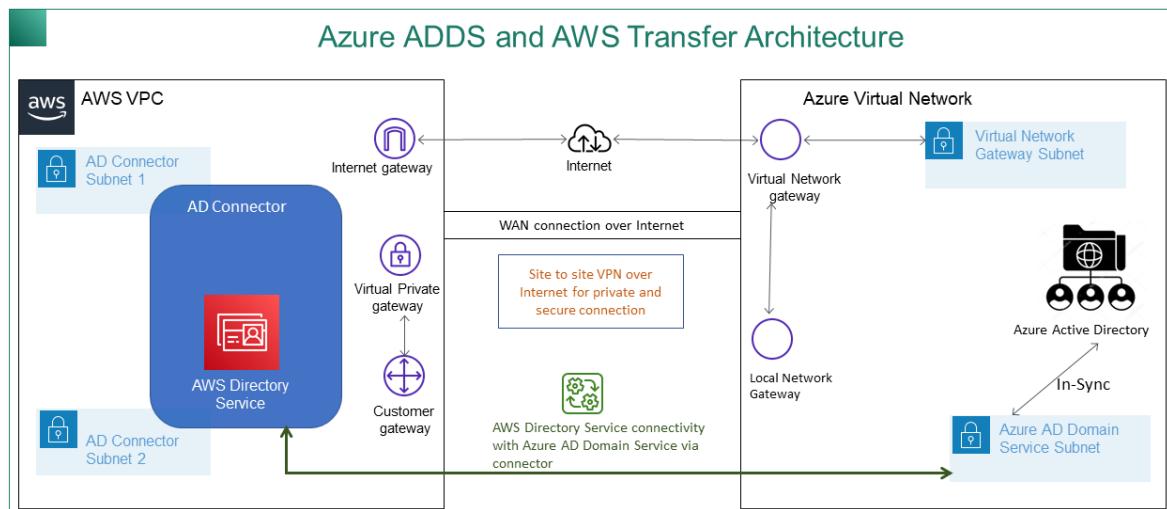
- An Azure Active Directory and Active directory domain service (Azure ADDS)
- An Azure resource group
- An Azure virtual network
- VPN connectivity between your Amazon VPC and your Azure resource group

Note

This can be through native IPSEC tunnels or using VPN appliances. In this topic, we use IPSEC tunnels between an Azure Virtual network gateway and local network gateway. The tunnels must be configured to allow traffic between your Azure ADDS endpoints and the subnets that house your AWS VPC.

- A customer gateway and Virtual private gateway for site-to-site VPN connection with Microsoft Azure

The following diagram shows the configuration needed before you begin.



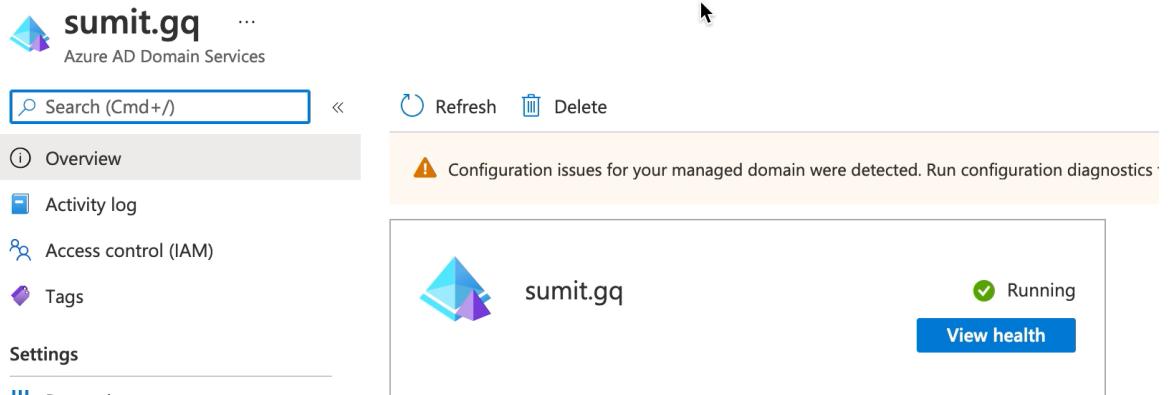
Step 1: Adding Azure Active Directory Domain Services

Azure AD does not support Domain joining instances by default. To perform actions like Domain Join, and to use tools such as Group Policy, administrators must enable Azure Active Directory Domain Services. If you have not already added Azure AD DS, or your existing implementation is not associated with the domain that you want your SFTP Transfer server to use, you must add a new instance.

For information about enabling Azure Active Directory Domain Services (Azure ADDS), see [Tutorial: Create and configure an Azure Active Directory Domain Services managed domain](#).

Note

When you enable Azure ADDS, make sure it is configured for the resource group and the Azure AD domain to which you are connecting your SFTP Transfer server.



Step 2: Creating a service account

Azure AD must have one service account that is part of an Admin group in Azure ADDS. This account is used with the AWS Active Directory connector. Make sure this account is in sync with Azure ADDS.

[Home](#) > [Default Directory](#) > [Users](#) > [sumithardwaj](#)

Name	First name	Last name
sumithardwaj	Sumit	Bhardwaj

Step 3: Setting up AWS Directory using AD Connector

After you have configured Azure ADDS, and created a service account with IPSEC VPN tunnels between your AWS VPC and Azure Virtual network, you can test the connectivity by pinging the Azure ADDS DNS IP address from any AWS EC2 instance.

After you verify the connection is active, you can continue below.

To set up your AWS Directory using AD Connector

1. Open the [Directory Service](#) console and select **Directories**.
2. Select **Set up directory**.
3. For directory type, choose **AD Connector**.

4. Select a directory size, select **Next**, then select your VPC and Subnets.
5. Select **Next**, then fill in the fields as follows:
 - **Directory DNS name:** enter the domain name you are using for your Azure ADDS.
 - **DNS IP addresses:** enter your Azure ADDS IP addresses.
 - **Server account username** and **password:** enter the details for the service account you created in *Step 2: Create a service account*.
6. Complete the screens to create the directory service.

Now the directory status should be **Active**, and it is ready to be used with an SFTP Transfer server.

Directory ID	Directory name	Type	Size	Multi-Region	Status	Launch date
d-906752c0d7	sumit.gq	AD Connector	Small	Not applicable	Active	Nov 3, 2021

Step 4: Setting up AWS Transfer Family server

Create an Transfer Family server with the SFTP protocol, and the identity provider type of **AWS Directory Service**. From **Directory** drop down list, select the directory you added in *Step 3: Setup AWS Directory using AD Connector*.

Note

You can't delete a Microsoft AD directory in AWS Directory Service if you used it in a Transfer Family server. You must delete the server first, and then you can delete the directory.

Step 5: Granting access to groups

After you create the server, you must choose which groups in the directory should have access to upload and download files over the enabled protocols using AWS Transfer Family. You do this by creating an access.

Note

Users must belong *directly* to the group to which you are granting access. For example, assume that Bob is a user and he belongs to groupA, and groupA itself is included in groupB.

- If you grant access to groupA, Bob is granted access.
- If you grant access to groupB (and not to groupA), Bob does not have access.

In order to grant access you need to retrieve the SID for the group.

Use the following Windows PowerShell command to retrieve the SID for a group, replacing **YourGroupName** with the name of the group.

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

```
Windows PowerShell
Copyright (C) 2016 Microsoft Corporation. All rights reserved.

PS C:\Users\sumitbhardwaj> Get-ADGroup -Filter {samAccountName -like "AAD DC Administrators*"} -Properties * | Select SamAccountName, ObjectSid
SamAccountName          ObjectSid
-----          -----
AAD DC Administrators  S-1-5-21-375932292-1747164136-3628472596-1104
```

Grant access to groups

1. Open <https://console.aws.amazon.com/transfer/>.
2. Navigate to your server details page and in the **Accesses** section, choose **Create access**.
3. Enter the SID you received from the output of the previous procedure.
4. For **Access**, choose an AWS Identity and Access Management role for the group.
5. In the **Policy** section, choose a policy. The default value is **None**.
6. For **Home directory**, choose an S3 bucket that corresponds to the group's home directory.
7. Choose **Add** to create the association.

The details from your Transfer server should look similar to the following:

Protocols	Identity provider
Protocols over which clients can connect to your server's endpoint <ul style="list-style-type: none">SFTP	Identity provider type AWS Directory Service Directory ID d-906752c0d7

Accesses (1)		Actions	Add access
<input type="checkbox"/> External Id	S-1-5-21-375932292-1747164136-3628472596-1104	Home directory	Role
		s3://sumitbhardwaj	sftp-user-role

Step 6: Testing users

You can test ([Testing users \(p. 64\)](#)) whether a user has access to the AWS Managed Microsoft AD directory for your server. A user must be in exactly one group (an external ID) that is listed in the **Access** section of the **Endpoint configuration** page. If the user is in no groups, or is in more than a single group, that user is not granted access.

Working with custom identity providers

To authenticate your users, you can use your existing identity provider with AWS Transfer Family. You integrate your identity provider using an AWS Lambda function, which authenticates and authorizes your users for access to Amazon S3 or Amazon Elastic File System (Amazon EFS). For details, see [Using AWS Lambda to integrate your identity provider \(p. 70\)](#).

Alternatively, you can provide a RESTful interface with a single Amazon API Gateway method. Transfer Family calls this method to connect to your identity provider, which authenticates and authorizes your users for access to Amazon S3 or Amazon EFS. Use this option if you need a RESTful API to

integrate your identity provider or if you want to use AWS WAF to leverage its capabilities for geo-blocking or rate-limiting requests. For details, see [Using Amazon API Gateway to integrate your identity provider \(p. 74\)](#).

In either case, you can create a new server using the [AWS Transfer Family console](#) or the [CreateServer \(p. 166\)](#) API operation.

Topics

- [Using AWS Lambda to integrate your identity provider \(p. 70\)](#)
- [Lambda resource-based policy \(p. 71\)](#)
- [Lambda functions for authentication \(p. 71\)](#)
- [Using Amazon API Gateway to integrate your identity provider \(p. 74\)](#)

Using AWS Lambda to integrate your identity provider

Create an AWS Lambda function that connects to your custom identity provider. You can use any custom identity provider, such as Okta, Secrets Manager, OneLogin, or a custom data store that includes authorization and authentication logic.

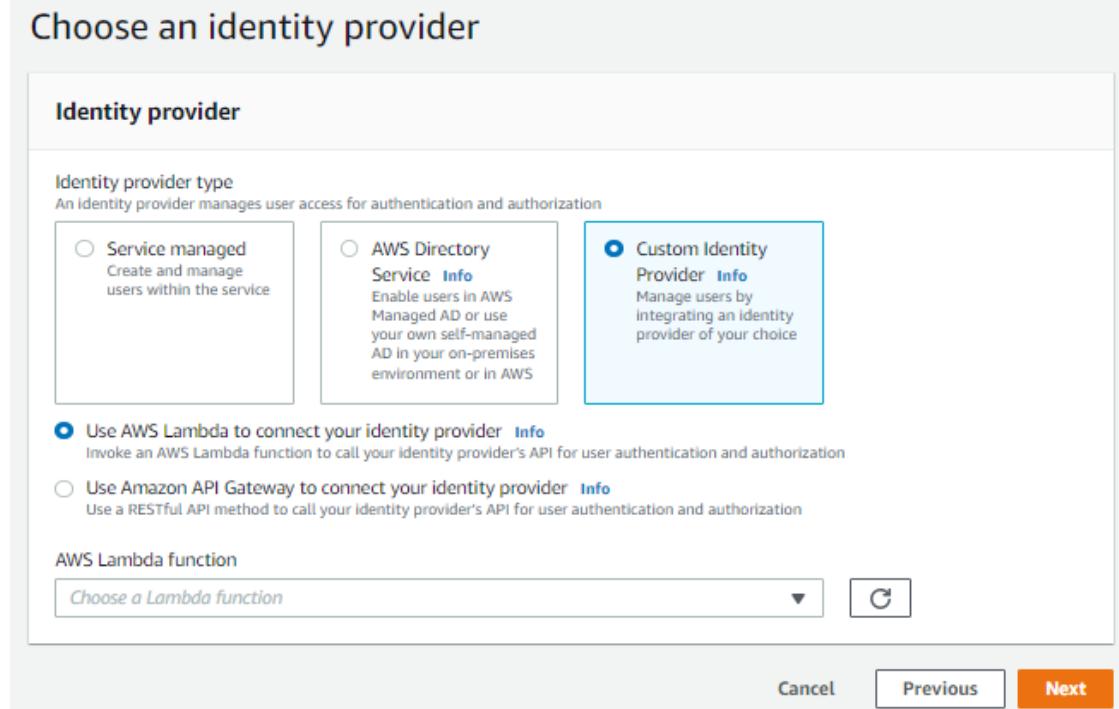
Note

Before you create a Transfer Family server that uses Lambda as the identity provider, you must create the function. For an example Lambda function, see [Default Lambda function \(p. 79\)](#).

Or, you can deploy a CloudFormation stack that uses a [Lambda function template \(p. 71\)](#).

Also, make sure your Lambda function uses a resource-based policy that trusts Transfer Family. For an example policy, see [Lambda resource-based policy \(p. 71\)](#).

1. Open the AWS Transfer Family console at <https://console.aws.amazon.com/transfer/>.
2. Choose **Create server** to open the **Create server** page. For **Choose an identity provider**, choose **Custom Identity Provider**, as shown in the following screenshot.



3. Make sure the default value, **Use AWS Lambda to connect your identity provider**, is selected.
4. For **AWS Lambda function**, choose the name of your Lambda function.
5. Fill in the remaining boxes, and then choose **Create server**. For details on the remaining steps for creating a server, see [Creating a server \(p. 19\)](#).

Lambda resource-based policy

You must have a policy that references the Transfer Family server and Lambda ARNs. For example, you could use the following policy with your Lambda function that connects to your identity provider.

```
{  
    "Version": "2012-10-17",  
    "Id": "default",  
    "Statement": [  
        {  
            "Sid": "AllowTransferInvocation",  
            "Effect": "Allow",  
            "Principal": {  
                "Service": "transfer.amazonaws.com"  
            },  
            "Action": "lambda:InvokeFunction",  
            "Resource": "$lambda_arn",  
            "Condition": {  
                "ArnLike": {  
                    "AWS:SourceArn": "$server_arn"  
                }  
            }  
        }  
    ]  
}
```

Lambda functions for authentication

To implement different authentication strategies, edit the Lambda function. To help you meet your application's needs, you can deploy a CloudFormation stack. For more information about Lambda, see the [AWS Lambda Developer Guide](#) or [Building Lambda functions with Node.js](#).

Lambda function template

You can deploy an AWS CloudFormation stack that uses a Lambda function for authentication. The function in the template uses Amazon Cognito to authenticate and authorize your users using username and password. You can modify this template or AWS Lambda code to further customize user access.

To create an AWS CloudFormation stack to use for authentication

1. Open the AWS CloudFormation console at <https://console.aws.amazon.com/cloudformation>.
2. Follow the instructions for deploying an AWS CloudFormation stack from an existing template in [Selecting a stack template](#) in the *AWS CloudFormation User Guide*.
3. Use the [Amazon Cognito stack template](#) to create a Lambda function to use for authentication in Transfer Family.

Important

We recommend that you edit the default user and password credentials.

After the stack has been deployed, you can view details about it on the **Outputs** tab in the CloudFormation console.

Valid Lambda values

The following table describes details for the values that Transfer Family accepts for Lambda functions that are used for custom identity providers.

Value	Description	Required
<code>Role</code>	Specifies the Amazon Resource Name (ARN) of the IAM role that controls your users' access to your Amazon S3 bucket or Amazon EFS file system. The policies attached to this role determine the level of access that you want to provide your users when transferring files into and out of your Amazon S3 or Amazon EFS file system. The IAM role should also contain a trust relationship that allows the server to access your resources when servicing your users' transfer requests.	Required
<code>PosixProfile</code>	The full POSIX identity, including user ID (<code>Uid</code>), group ID (<code>Gid</code>), and any secondary group IDs (<code>SecondaryGids</code>), that controls your users' access to your Amazon EFS file systems. The POSIX permissions that are set on files and directories in your file system determine the level of access your users get when transferring files into and out of your Amazon EFS file systems.	Required for Amazon EFS backing storage
<code>PublicKeys</code>	A list of SSH public key values that are valid for this user. An empty list implies that this is not a valid login. Must not be returned during password authentication.	Optional
<code>Policy</code>	A session policy for your user so that you can use the same IAM role across multiple users. This policy scopes down user access to portions of their Amazon S3 bucket.	Optional
<code>HomeDirectoryType</code>	The type of landing directory (folder) that you want your users' home directory to be when they log in to the server. If you set it to <code>PATH</code> , the user	Optional

Value	Description	Required
	sees the absolute Amazon S3 bucket or Amazon EFS paths as is in their file transfer protocol clients. If you set it to LOGICAL, you must provide mappings in the HomeDirectoryMappings parameter for how you want to make Amazon S3 or Amazon EFS paths visible to your users.	
<code>HomeDirectoryDetails</code>	Logical directory mappings that specify which Amazon S3 or Amazon EFS paths and keys should be visible to your user and how you want to make them visible. You must specify the <code>Entry</code> and <code>Target</code> pair, where <code>Entry</code> shows how the path is made visible and <code>Target</code> is the actual Amazon S3 or Amazon EFS path.	Required if <code>HomeDirectoryType</code> has a value of <code>LOGICAL</code>
<code>HomeDirectory</code>	The landing directory for a user when they log in to the server using the client.	Optional

Testing your configuration

After you create your custom identity provider, you should test your configuration.

Console

To test your configuration by using the AWS Transfer Family console

1. Open the AWS Transfer Family console at <https://console.aws.amazon.com/transfer/>.
2. On the **Servers** page, choose your new server, choose **Actions**, and then choose **Test**.
3. Enter the text for **Username** and **Password** that you set when you deployed the AWS CloudFormation stack. If you kept the default options, the user name is `myuser` and the password is `MySuperSecretPassword`.
4. Choose the **Server protocol** and enter the IP address for **Source IP**, if you set them when you deployed the AWS CloudFormation stack.

CLI

To test your configuration by using the AWS CLI

1. Run the `test-identity-provider` command. Replace each `user input placeholder` with your own information, as described in the subsequent steps.

```
aws transfer test-identity-provider --server-id s-1234abcd5678efgh --user-name myuser --user-password MySuperSecretPassword --server-protocol FTP --source-ip 127.0.0.1
```

2. Enter the server ID.

3. Enter the user name and password that you set when you deployed the AWS CloudFormation stack. If you kept the default options, the user name is `myuser` and the password is `MySuperSecretPassword`.
4. Enter the server protocol and source IP address, if you set them when you deployed the AWS CloudFormation stack.

If user authentication succeeds, the test returns a `Status Code`: 200 HTTP response and a JSON object containing the details of the user's roles and permissions, as shown in the following example.

```
{  
    "Response": "{\"Policy\": \"{\n        \"Version\": \"2012-10-17\",\n        \"Statement\": [\n            {\n                \"Sid\": \"ReadAndListAllBuckets\",\n                \"Effect\": \"Allow\",\n                \"Action\": [\n                    \"s3>ListAllMybuckets\",\n                    \"s3>GetBucketLocation\",\n                    \"s3>ListBucket\",\n                    \"s3>GetObjectVersion\",\n                    \"s3>GetObjectVersion\"\n                ],\n                \"Resource\": \"*\n            ]\n        }\n    }\",  
    \"Role\": \"arn:aws:iam:000000000000:role/MyUserS3AccessRole\",  
    \"HomeDirectory\": \"/\"},  
    \"StatusCode\": 200,  
    \"Message\": "",  
    \"Url\": \"https://abcde1234.execute-api.us-east-2.amazonaws.com/prod/servers/  
s-123a4567bcd891e23/users/myuser/config\"\n}
```

Note

The `Url` line is returned only if you are using an API Gateway method as your custom identity provider.

Using Amazon API Gateway to integrate your identity provider

This section describes how to use an AWS Lambda function to back an API Gateway method.

Authenticating using an API Gateway method

You can create an API Gateway method for use as an identity provider for Transfer Family. This approach provides a highly secure way for you to create and provide APIs. With API Gateway, you can create an HTTPS endpoint so that all incoming API calls are transmitted with greater security. For more details about the API Gateway service, see the [API Gateway Developer Guide](#).

API Gateway offers an authentication method named `AWS_IAM`, which gives you the same authentication based on AWS Identity and Access Management (IAM) that AWS uses internally. If you enable authentication with `AWS_IAM`, only callers with explicit permissions to call an API can reach that API's API Gateway method.

To use your API Gateway method as a custom identity provider for Transfer Family, enable IAM for your API Gateway method. As part of this process, you provide an IAM role with permissions for Transfer Family to use your gateway.

Note

To improve security, you can configure a web application firewall. AWS WAF is a web application firewall that lets you monitor the HTTP and HTTPS requests that are forwarded to an Amazon API Gateway. For details, see [Add a web application firewall \(p. 146\)](#).

To use your API Gateway method for custom authentication with Transfer Family

1. Create an AWS CloudFormation stack. To do this:

- a. Open the AWS CloudFormation console at <https://console.aws.amazon.com/cloudformation>.
- b. Follow the instructions for deploying an AWS CloudFormation stack from an existing template in [Selecting a stack template](#) in the *AWS CloudFormation User Guide*.
- c. Use one of the following basic templates to create an AWS Lambda-backed API Gateway method for use as a custom identity provider in Transfer Family.
 - [Basic stack template](#)

By default, your API Gateway method is used as a custom identity provider to authenticate a single user in a single server using a hardcoded SSH (Secure Shell) key or password. After deployment, you can modify the Lambda function code to do something different.

- [AWS Secrets Manager stack template](#)

By default, your API Gateway method authenticates against an entry in Secrets Manager of the format `SFTP/username`. Additionally, the secret must hold the key-value pairs for all user properties returned to Transfer Family. After deployment, you can modify the Lambda function code to do something different. For more information, see [Enable password authentication for AWS Transfer Family using AWS Secrets Manager](#).

- [Okta stack template](#)

Your API Gateway method integrates with Okta as a custom identity provider in Transfer Family. For more information, see [Using Okta as an identity provider with AWS Transfer Family](#).

Deploying one of these stacks is the easiest way to integrate a custom identity provider into the Transfer Family workflow. Each stack uses the Lambda function to support your API method based on API Gateway. You can then use your API method as a custom identity provider in Transfer Family. By default, the Lambda function authenticates a single user called `myuser` with a password of `MySuperSecretPassword`. After deployment, you can edit these credentials or update the Lambda function code to do something different.

Important

We recommend that you edit the default user and password credentials.

After the stack has been deployed, you can view details about it on the **Outputs** tab in the CloudFormation console. These details include the stack's Amazon Resource Name (ARN), the ARN of the IAM role that the stack created, and the URL for your new gateway.

Note

If you are using the custom identity provider option to enable password-based authentication for your users, and you enable the request and response logging provided by API Gateway, API Gateway logs your users' passwords to your Amazon CloudWatch Logs. We don't recommend using this log in your production environment. For more information, see [Set up CloudWatch API logging in API Gateway](#) in the *API Gateway Developer Guide*.

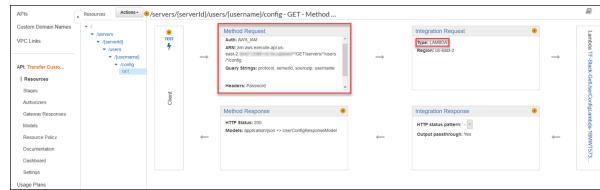
2. Check the API Gateway method configuration for your server. To do this:

- a. Open the API Gateway console at <https://console.aws.amazon.com/apigateway/>.
- b. Choose the **Transfer Custom Identity Provider basic template API** that the AWS CloudFormation template generated.

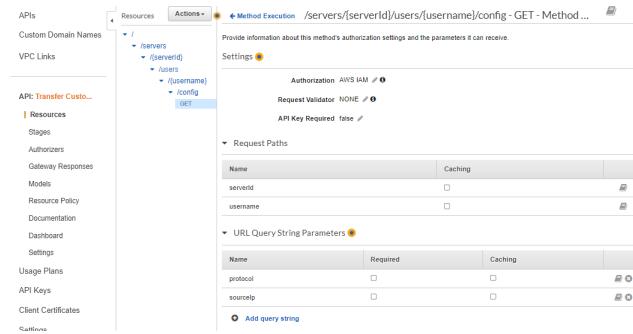
The following screenshot shows the complete API configuration. In this example, the method is backed by a Lambda function, but many other integration types are also possible.

AWS Transfer Family User Guide

Using Amazon API Gateway to integrate your identity provider



- c. In the **Resources** pane, choose **GET**, and then choose **Method Request**. The following screenshot shows the correct method configuration.



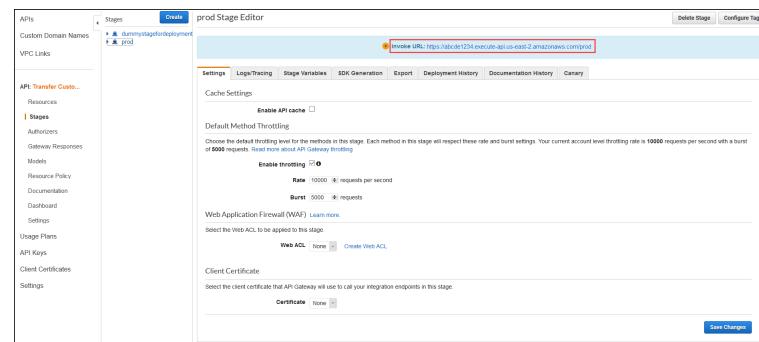
At this point, your API gateway is ready to be deployed.

3. For **Actions**, choose **Deploy API**. For **Deployment stage**, choose **prod**, and then choose **Deploy**.

After the API Gateway method is successfully deployed, view its performance in the **Stage Editor** section, as shown in the following screenshot.

Note

Copy the **Invoke URL** address that appears at the top of the screen. You will need it for the next step.



4. Open the AWS Transfer Family console at <https://console.aws.amazon.com/transfer/>.
5. Choose **Create server** to open the **Create server** page. For **Choose an identity provider**, choose **Custom**, then select **Use Amazon API Gateway to connect to your identity provider**, as shown in the following screenshot.

Choose an identity provider

Identity provider

Identity provider type
An Identity provider manages user access for authentication and authorization

Service managed
Create and manage users within the service

AWS Directory Service [Info](#)
Enable users in AWS Managed AD or use your own self-managed AD in your on-premises environment or in AWS

Custom Identity Provider [Info](#)
Manage users by integrating an identity provider of your choice

Use AWS Lambda to connect your identity provider [Info](#)
Invoke an AWS Lambda function to call your identity provider's API for user authentication and authorization

Use Amazon API Gateway to connect your identity provider [Info](#)
Use a RESTful API method to call your identity provider's API for user authentication and authorization

Provide an Amazon API Gateway URL

Role
IAM role for the service to invoke your Amazon API Gateway URL

[▼](#) [C](#)

[Cancel](#) [Previous](#) [Next](#)

6. In the **Provide an Amazon API Gateway URL** text box, paste the **Invoke URL** address of the API Gateway endpoint that you created in step 3 of this procedure.
7. For **Role**, choose the IAM role that was created by the AWS CloudFormation template. This role allows Transfer Family to invoke your API gateway method.

The invocation role contains the AWS CloudFormation stack name that you selected for the stack that you created in step 1. It has the following format: ***CloudFormation-stack-name-TransferIdentityProviderRole-ABC123DEF456GHI***.

8. Fill in the remaining boxes, and then choose **Create server**. For details on the remaining steps for creating a server, see [Creating a server \(p. 19\)](#).

Implementing your API Gateway method

To create a custom identity provider for Transfer Family, your API Gateway method must implement a single method that has a resource path of `/servers/serverId/users/username/config`. The *serverId* and *username* values come from the RESTful resource path. Optionally, you can add the *sourceIp* and *protocol* values to the RESTful resource path.

Note

The user name must be a minimum of 3 and a maximum of 100 characters. You can use the following characters in the user name: a–z, A–Z, 0–9, underscore (_), hyphen (-), period (.), and at sign (@). However, the user name can't start with a hyphen (-), period (.), or at sign (@).

If Transfer Family attempts password authentication for your user, the service supplies a `Password:` header field. In the absence of a `Password:` header, Transfer Family attempts public key authentication to authenticate your user.

When you are using an identity provider to authenticate and authorize end users, in addition to validating their credentials, you can allow or deny access requests based on the IP addresses of the clients used by your end users. You can use this feature to ensure that data stored in your S3 buckets or your Amazon EFS file system can be accessed over the supported protocols only from IP addresses that you have specified as trusted. To enable this feature, you must include the *sourceIp* value in the RESTful resource path.

If you have multiple protocols enabled for your server and want to provide access using the same user name over multiple protocols, you can do so as long as the credentials specific to each protocol have been set up in your identity provider. To enable this feature, you must include the *protocol* value in the RESTful resource path.

Your API Gateway method should always return HTTP status code 200. Any other HTTP status code means that there was an error accessing the API.

Amazon S3 example response

The example response body is a JSON document of the following form for Amazon S3.

```
{  
    "Role": "IAM role with configured S3 permissions",  
    "PublicKeys": [  
        "ssh-rsa public-key1",  
        "ssh-rsa public-key2"  
    ],  
    "Policy": "STS Assume role session policy",  
    "HomeDirectory": "/bucketName/path/to/home/directory"  
}
```

Note

The policy is escaped JSON as a string. For example:

```
"Policy":  
"{  
    \"Version\": \"2012-10-17\",  
    \"Statement\":  
    [  
        {\"Condition\":  
            {\"StringLike\":  
                {\"s3:prefix\":  
                    [\"user/*\", \"user/\"]}},  
            \"Resource\": \"arn:aws:s3:::bucket\",  
            \"Action\": \"s3>ListBucket\",  
            \"Effect\": \"Allow\",  
            \"Sid\": \"ListHomeDir\",  
            \"Resource\": \"arn:aws:s3:::*\",  
            \"Action\": [\"s3:PutObject\",  
                      \"s3:GetObject\",  
                      \"s3>DeleteObjectVersion\",  
                      \"s3:DeleteObject\",  
                      \"s3:GetObjectVersion\",  
                      \"s3:GetObjectACL\",  
                      \"s3:PutObjectACL\"],  
            \"Effect\": \"Allow\",  
            \"Sid\": \"HomeDirObjectAccess\"}  
    ]  
}"
```

The following example response shows that a user has a logical home directory type.

```
{
```

```
\\"Role\\": \\"arn:aws:iam::123456789012:role/role-api-gateway-s3\\",
\\"HomeDirectoryType\\": \"LOGICAL\",
\\"HomeDirectoryDetails\\": \"[{\\"Entry\\\": \\"/\\"/\\"}, \\"Target\\\": \\"/\\"/\\"/my-home-bucket\\"}]\\",
\\"PublicKeys\\": [\"\""]
}
```

Amazon EFS example response

The example response body is a JSON document of the following form for Amazon EFS.

```
{
  \"Role\": \"IAM role with configured EFS permissions\",
  \"PublicKeys\": [
    \"ssh-rsa public-key1\",
    \"ssh-rsa public-key2\"
  ],
  \"PosixProfile\": {
    \"Uid\": \"POSIX user ID\",
    \"Gid\": \"POSIX group ID\",
    \"SecondaryGids\": [Optional list of secondary Group IDs],
  },
  \"HomeDirectory\": \"/fs-id/path/to/home/directory\"
}
```

The Role field shows that successful authentication occurred. When doing password authentication (when you supply a Password: header), you don't need to provide SSH public keys. If a user can't be authenticated, for example, if the password is incorrect, your method should return a response without Role set. An example of such a response is an empty JSON object.

The following example response shows a user that has a logical home directory type.

```
{
  \"Role\": \\"arn:aws:iam::123456789012:role/role-api-gateway-efs\\",
  \"HomeDirectoryType\": \"LOGICAL\",
  \"HomeDirectoryDetails\\": \"[{\\"Entry\\\": \\"/\\"/\\"}, \\"Target\\\": \\"/\\"/\\"/faa1a123\\"}]\\",
  \"PublicKeys\\": [\"\""],
  \"PosixProfile\\\": {\\"Uid\\\": 65534, \\"Gid\\\": 65534}
}
```

You can include user policies in the Lambda function in JSON format. For more information about configuring user policies in Transfer Family, see [Managing access controls \(p. 118\)](#).

Default Lambda function

To implement different authentication strategies, edit the Lambda function that your gateway uses. To help you meet your application's needs, you can use the following example Lambda functions in Node.js. For more information about Lambda, see the [AWS Lambda Developer Guide](#) or [Building Lambda functions with Node.js](#).

The following example Lambda function takes your user name, password (if you're performing password authentication), server ID, protocol, and client IP address. You can use a combination of these inputs to look up your identity provider and determine if the login should be accepted.

Note

If you have multiple protocols enabled for your server and want to provide access using the same user name over multiple protocols, you can do so as long as the credentials specific to the protocol have been set up in your identity provider.

For File Transfer Protocol (FTP), we recommend maintaining separate credentials from Secure Shell (SSH) File Transfer Protocol (SFTP) and File Transfer Protocol over SSL (FTPS). We recommend maintaining separate credentials for FTP because, unlike SFTP and FTPS, FTP transmits credentials in clear text. By isolating FTP credentials from SFTP or FTPS, if FTP credentials are shared or exposed, your workloads using SFTP or FTPS remain secure.

This example function works only with Amazon S3. This example function returns the role and logical home directory details, along with the public keys (if it performs public key authentication).

The [HomeDirectoryType](#) parameter specifies the type of landing directory (folder) that you want your user's home directory to be when they log in to the server. If you set this parameter to `PATH`, the user sees the absolute Amazon S3 bucket or Amazon EFS paths as is in their file transfer protocol clients. If you set this parameter to `LOGICAL`, you must provide mappings in the [HomeDirectoryMappings](#) parameter for how you want to make Amazon S3 or Amazon EFS paths visible to your users.

PATH home directory

This function is for users that are using `PATH` for their [HomeDirectoryType](#).

```
// GetUserConfig Lambda

exports.handler = (event, context, callback) => {
    console.log("Username:", event.username, "ServerId: ", event.serverId);

    var response;
    // Check if the user name presented for authentication is correct. This doesn't check
    // the value of the serverId, only that it is provided.
    // There is also event.protocol (one of "FTP", "FTPS", "SFTP") and event.sourceIp
    // (e.g., "127.0.0.1") to further restrict logins.
    if (event.serverId !== "" && event.username == '${UserName}') {
        response = {
            Role: '${UserRoleArn}', // The user will be authenticated if and only if the Role
            // field is not blank
            Policy: '', // Optional JSON blob to further restrict this user's permissions
            HomeDirectory: '${UserHomeDirectory}' // Not required, defaults to '/'
        };

        // Check if password is provided
        if (event.password == "") {
            // If no password provided, return the user's SSH public key
            response['PublicKeys'] = [ "${UserPublicKey1}" ];
            // Check if password is correct
            } else if (event.password !== '${UserPassword}') {
                // Return HTTP status 200 but with no role in the response to indicate
                // authentication failure
                response = {};
            }
        } else {
            // Return HTTP status 200 but with no role in the response to indicate
            // authentication failure
            response = {};
        }
        callback(null, response);
    };
}
```

LOGICAL home directory

The following Lambda function provides the details for a user that has a [logical home directory \(p. 87\)](#). The user, role, POSIX profile, password, and home directory details in this function are all examples, and must be replaced with your actual values.

```
// GetUserConfig Lambda
```

```
exports.handler = (event, context, callback) => {
    console.log("Username:", event.username, "ServerId: ", event.serverId);

    var response;
    // Check if the username presented for authentication is correct. This doesn't check
    the value of the serverId, only that it is provided.
    if (event.serverId !== "" && event.username == 'example-user') {
        response = {
            Role: 'arn:aws:iam::123456789012:role/role-api-gateway', // The user will be
authenticated if and only if the Role field is not blank
            PosixProfile: {Gid": 65534, "Uid": 65534},
            HomeDirectoryDetails: "[{\\"Entry\\": \"/\", \\"Target\\": \"/fs-faa1a123\"}]",
            HomeDirectoryType: "LOGICAL",
            //HomeDirectory: '/fs-faa1a123' // Not required, defaults to '/'
        };

        // Check if password is provided
        if (event.password == "") {
            // If no password provided, return the user's SSH public key
            response['PublicKeys'] = [ " " ];
            // Check if password is correct
        } else if (event.password == 'Password1234') {
            // Return HTTP status 200 but with no role in the response to indicate
authentication failure
            response = {};
        }
        } else {
            // Return HTTP status 200 but with no role in the response to indicate
authentication failure
            response = {};
        }
        callback(null, response);
    };
};
```

Lambda function for use with AWS Secrets Manager

To use AWS Secrets Manager as your identity provider, you can work with the Lambda function in the sample AWS CloudFormation template. The Lambda function queries the Secrets Manager service with your credentials and, if successful, returns a designated secret. For more information about Secrets Manager, see the [AWS Secrets Manager User Guide](#).

To download a sample AWS CloudFormation template that uses this Lambda function, go to the [Amazon S3 bucket provided by AWS Transfer Family](#).

Tutorial: Setting up an Amazon API Gateway method as a custom identity provider

This tutorial illustrates how to set up an Amazon API Gateway method and use it as a custom identity provider to upload files to an AWS Transfer Family server. This tutorial uses the [Basic stack template](#), and other basic functionality as an example only.

Contents

- [Step 1: Create a CloudFormation stack \(p. 82\)](#)
- [Step 2: Check the API Gateway method configuration for your server and create it. \(p. 82\)](#)
- [Step 3: Create the Transfer Family server \(p. 83\)](#)

- Step 4: Test that your user can connect to the server (p. 83)
- Step 5: Test the SFTP connection and file transfer (p. 84)
- Step 6: Limit access to the bucket (p. 84)

Step 1: Create a CloudFormation stack

1. Open the AWS CloudFormation console at <https://console.aws.amazon.com/cloudformation>.
2. Select **Create stack**, and choose **With new resources (standard)**.
3. In the **Prerequisite - Prepare template** pane, choose **Template is ready**.
4. Copy this link, **Basic stack template**, and paste it into the **Amazon S3 URL** field.
5. Click **Next**.
6. Specify parameters, including a name for your stack. Be sure to do the following:
 - Replace the default values for **UserName** and **UserPassword**.
 - Replace the default **UserRoleArn** with an actual AWS Identity and Access Management (IAM) role that has the appropriate permissions. For an example IAM role and bucket policy, see [Step 6: Limit access to the bucket \(p. 84\)](#).
 - If you want to authenticate using a public key instead of a password, enter your public key in the **UserPublicKey1** field. The first time that you connect to the server using SFTP, you then provide the private key instead of a password.
7. Choose **Next**, and then choose **Next** again on the **Configure stack options** page.
8. Review the details for the stack that you are creating, and then choose **Create stack**.

Note

At the bottom of the page, under **Capabilities**, you must acknowledge that AWS CloudFormation might create IAM resources.

Step 2: Check the API Gateway method configuration for your server and create it.

Note

To improve security, you can configure a web application firewall. AWS WAF is a web application firewall that lets you monitor the HTTP and HTTPS requests that are forwarded to an Amazon API Gateway. For details, see [Add a web application firewall \(p. 146\)](#).

To check the API Gateway method configuration for your server and create it.

1. Open the API Gateway console at <https://console.aws.amazon.com/apigateway/>.
2. Choose the **Transfer Custom Identity Provider basic template API** that the AWS CloudFormation template generated.
3. In the **Resources** pane, choose **GET**, and then choose **Method Request**.
4. For **Actions**, choose **Deploy API**. For **Deployment stage**, choose **prod**, and then choose **Deploy**.

After the API Gateway method is successfully deployed, view its performance in the **Stage Editor** section.

Note

Copy the **Invoke URL** address that appears at the top of the page. You will need it for the next step.

Step 3: Create the Transfer Family server

To create a Transfer Family server

1. Open the AWS Transfer Family console at <https://console.aws.amazon.com/transfer/>.
2. Choose **Create server** to open the **Create server** page.
3. For **Choose protocols**, choose **SFTP (SSH File Transfer Protocol)**, and then choose **Next**.
4. For **Choose an identity provider**, choose **Custom**.
5. For **Custom provider**, paste the **Invoke URL** address of the API Gateway endpoint that you created in [Step 2 \(p. 82\)](#).
6. For **Invocation role**, choose the IAM role that was created by the AWS CloudFormation template. This role allows Transfer Family to invoke your API Gateway method.

The invocation role contains the AWS CloudFormation stack name that you selected for the stack that you created earlier. It has the following format: *CloudFormation-stack-name-TransferIdentityProviderRole-ABC123DEF456GHI*.

7. Ensure that you add the correct CloudWatch logging role. It has the following format:

CloudFormation-stack-name-CloudWatchLoggingRole-ABC123DEF45

Note

Make sure the logging role for the server has a trust relationship with Transfer Family. For details, see [To establish a trust relationship \(p. 9\)](#).

8. Fill in the remaining options, and then choose **Create server**.

Note

If you choose Amazon EFS for the storage option, you need to add a Posix Profile setting to the Lambda function. For details, see [Update Lambda if using Amazon EFS \(p. 85\)](#).

Step 4: Test that your user can connect to the server

To test that your user can connect to the server, using the Transfer Family console

1. Open the AWS Transfer Family console at <https://console.aws.amazon.com/transfer/>.
2. On the **Servers** page, choose your new server, choose **Actions**, and then choose **Test**.
3. Enter the text for **Username** and **Password** that you set when you deployed the AWS CloudFormation stack.
4. For **Server Protocol**, select **SFTP**, and for **Source IP**, enter **127.0.0.1**.
5. Choose **Test**.

If user authentication succeeds, the test returns a **Status Code**: 200 HTML response and a JSON object containing the details of the user's roles and permissions. For example:

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

If the test fails, add one of the API Gateway AWS managed policies to the role that you are using for your API.

Step 5: Test the SFTP connection and file transfer

To test the SFTP connection

1. On a Linux or macOS device, open a command terminal.
2. Enter one of the following commands, depending on whether you are using a password or a key pair for authentication.
 - If you are using a password, enter this command:

```
sftp myuser@server-ID.server.transfer.region-code.amazonaws.com
```

When prompted, enter your password.

- If you are using a key pair, enter this command:

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

Note

For these sftp commands, insert the code for the AWS Region where your Transfer Family server is located. For example, if your server is in US East (Ohio), enter **us-east-2**.

3. At the sftp> prompt, make sure that you can upload (put), download (get), and view directories and files (pwd and ls).

Step 6: Limit access to the bucket

You can limit who can access a specific Amazon S3 bucket. The following example shows the settings to use in your CloudFormation stack and in the policy that you select for your user.

In this example, we set the following parameters for the AWS CloudFormation stack:

- **CreateServer**: true
- **UserHomeDirectory**: /myuser-bucket
- **UserName**: myuser
- **UserPassword**: MySuperSecretPassword

Important

This is an example password. When you configure your API Gateway method, make sure that you enter a strong password.

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

The **UserPublicKey1** is a public key that you have generated as part of a public/private key pair.

The *role-id* is unique to the user role that you create. The policy attached to the myuser-api-gateway-role is the following:

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "VisualEditor0",  
            "Effect": "Allow",  
            "Action": "s3:GetObject",  
            "Resource": "arn:aws:s3:::myuser-bucket/*"  
        }  
    ]  
}
```

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

To connect to the server using SFTP, enter one of the following commands at the prompt.

- If you are using a password to authenticate, run the following command:

```
sftp myuser@transfer-server-ID.server.transfer.region-id.amazonaws.com
```

When prompted, enter your password.

- If you are using a key pair to authenticate, run the following command:

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

Note

For these sftp commands, use the ID for the AWS Region where your Transfer Family server is located. For example, if your server is in US East (Ohio), use us-east-2.

At the sftp prompt, you are directed to your home directory, which you can view by running the `pwd` command. For example:

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

The user cannot view any directories above the home directory. For example:

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

Update Lambda if using Amazon EFS

If you selected Amazon EFS as the storage option for your Transfer Family server, you need to edit the lambda function for your stack.

To add a Posix Profile to your Lambda function

1. Open the Lambda console at <https://console.aws.amazon.com/lambda/>.

2. Select the Lambda function that you created earlier. The Lambda function has the format of **stack-name-GetUserConfigLambda-lambda-identifier**, where **stack-name** is the CloudFormation stack name and **lambda-identifier** is the identifier for the function.
3. In the **Code** tab, select **index.js** to display the code for the function.
4. In the **response**, add the following line between **Policy** and **HomeDirectory**:

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

Where the **uid-value** and **gid-value** are integers, 0 or greater, that represent the User ID and Group ID respectively.

For example, after you add the Posix profile, the response field might look like the following:

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

Using logical directories to simplify your Transfer Family directory structures

To simplify your AWS Transfer Family server directory structure, you can use logical directories. With logical directories, you can construct a virtual directory structure that uses user-friendly names that your users navigate when they connect to your Amazon S3 bucket or Amazon EFS file system. When you use logical directories, you can avoid disclosing absolute directory paths, Amazon S3 bucket names, and EFS file system names to your end users.

You can use logical directories to set the user's root directory to a desired location within your storage hierarchy, by performing what is known as a **chroot** operation. In this mode, users are not able to navigate to a directory outside of the home or root directory that you've configured for them.

For example, although an Amazon S3 user has been scoped down to access only `/mybucket/home/${transfer:UserName}`, some clients allow users to traverse up a folder to `/mybucket/home`. In this situation, the user lands back on their intended home directory only after logging out of and back in to the Transfer Family server again. Performing a **chroot** operation can prevent this situation from occurring.

You can create your own directory structure across buckets and prefixes. This feature is useful if you have a workflow that is expecting a specific directory structure that you can't replicate through bucket prefixes. You can also link to multiple non-contiguous locations within Amazon S3, similar to creating a symbolic link in a Linux file system where your directory path references a different location in the file system.

Rules for using logical directories

Before you build your logical directory mappings, you should understand the following rules:

- When `Entry` is `"/"`, you can have only one mapping because overlapping paths are not allowed.
- Targets can use the `${Transfer:UserName}` variable if the bucket or file system path has been parameterized based on the user name.
- Targets can be paths in different buckets or file systems, but you must make sure that the mapped IAM role (the `Role` parameter in the response) provides access to those buckets or file systems.
- Do not specify the `HomeDirectory` parameter because this value is implied by the `Entry Target` pairs when using the `LOGICAL` value for the `HomeDirectoryType` parameter.

Implementing logical directories and chroot

To use logical directories and **chroot** features, you must do the following:

Turn on logical directories for each user. Do this by setting the `HomeDirectoryType` parameter to `LOGICAL` when you create or update your user.

```
"HomeDirectoryType": "LOGICAL"
```

Chroot

For **chroot**, create a directory structure that consists of a single **Entry** and **Target** pairing for each user. The root folder is the **Entry** point, and the **Target** is the location in your bucket or file system to map to.

Example for Amazon S3

```
[{"Entry": "/", "Target": "/mybucket/jane"}]
```

Example for Amazon EFS

```
[{"Entry": "/", "Target": "/fs-faa1a123/jane"}]
```

You can use an absolute path as in the previous example, or you can use a dynamic substitution for the user name with `#{Transfer:UserName}`, as in the following example.

```
[{"Entry": "/", "Target": "/mybucket/${Transfer:UserName}"}]
```

In the preceding example, the user is locked to their root directory and cannot traverse up higher in the hierarchy.

Virtual directory structure

For a virtual directory structure, you can create multiple **Entry Target** pairings, with targets anywhere in your S3 buckets or EFS file systems, including across multiple buckets or file systems, as long as the user's IAM role mapping has permissions to access them.

In the following virtual structure example, when the user logs into AWS SFTP, they are in the root directory with subdirectories of /pics, /doc, /reporting, and /anotherpath/subpath/financials.

```
[{"Entry": "/pics", "Target": "/bucket1/pics"}, {"Entry": "/doc", "Target": "/bucket1/anotherpath/docs"}, {"Entry": "/reporting", "Target": "/reportingbucket/Q1"}, {"Entry": "/anotherpath/subpath/financials", "Target": "/reportingbucket/financials"}]
```

Note

You can only upload files to the specific folders that you map. This means that in the previous example, you cannot upload to /anotherpath or anotherpath/subpath directories; only anotherpath/subpath/financials. You also cannot map to those paths directly, as overlapping paths are not allowed.

For example, assume that you create the following mappings:

```
{ "Entry": "/pics", "Target": "/mybucket/pics"}, {
```

```
        "Entry": "/doc",
        "Target": "/mybucket/mydocs"
    },
{
    "Entry": "/temp",
    "Target": "/mybucket"
}
```

You can only upload files to those buckets. When you first connect through sftp, you are dropped into the root directory, /. If you attempt to upload a file to that directory, the upload fails. The following commands show an example sequence:

```
sftp> pwd
Remote working directory: /
sftp> put file
Uploading file to /file
remote open("/file"): No such file or directory
```

To upload to any directory/sub-directory, you must explicitly map the path to the sub-directory.

For more information about configuring logical directories and **chroot** for your users, including an AWS CloudFormation template that you can download and use, see [Simplify your AWS SFTP Structure with chroot and logical directories](#) in the AWS Storage Blog.

Transferring files using a client

You transfer files over the AWS Transfer Family service by specifying the transfer operation in a client. AWS Transfer Family supports the following clients:

- OpenSSH (macOS and Linux)

Note

This client works only with servers that are enabled for Secure Shell (SSH) File Transfer Protocol (SFTP).

- WinSCP (Microsoft Windows only)
- Cyberduck (Windows, macOS, and Linux)
- FileZilla (Windows, macOS, and Linux)

The following limitations apply to every client:

- For File Transfer Protocol over SSL (FTPS), only Explicit mode is supported. Implicit mode is not supported.
- For File Transfer Protocol (FTP) and FTPS, only Passive mode is supported.
- For FTP and FTPS, only STREAM mode is supported.
- For FTP and FTPS, only Image/Binary mode is supported.
- For FTP and FTPS, TLS - PROT C (unprotected) TLS for the data connection is the default but PROT C is not supported in the AWS Transfer Family FTPS protocol. So for FTPS, you will need to issue PROT P for your data operation to be accepted.
- If you are using Amazon S3 for your server's storage, and if your client contains an option to use multiple connections for a single transfer, make sure to disable the option. Otherwise, large file uploads can fail in unpredictable ways. Note that if you are using Amazon EFS as your storage backend, EFS *does* support multiple connections for a single transfer.

The following is a list of available commands for FTP and FTPS:

Available commands					
ABOR	FEAT	MLST	PASS	RETR	STOR
AUTH	LANG	MKD	PASV	RMD	STOU
CDUP	LIST	MODE	PBSZ	RNFR	STRU
CWD	MDTM	NLST	PROT	RNTO	SYST
DELE	MFMT	NOOP	PWD	SIZE	TYPE
EPSV	MLSD	OPTS	QUIT	STAT	USER

Note

APPE is not supported.

For SFTP, the following operations are currently not supported for users that are using the logical home directory on servers that are using Amazon Elastic File System (Amazon EFS).

Unsupported SFTP commands			
SSH_FXP_READLINK	SSH_FXP_SYMLINK	SSH_FXP_STAT when the requested file is a symlink	SSH_FXP_REALPATH when the requested path contains any symlink components

Generate public-private key pair

Before you can transfer a file, you must have a public-private key pair available. If you have not previously generated a key pair, see [Generate SSH keys \(p. 127\)](#).

Following, you can find how to transfer files with each client.

Topics

- [Use OpenSSH \(p. 18\)](#)
- [Use WinSCP \(p. 92\)](#)
- [Use Cyberduck \(p. 17\)](#)
- [Use FileZilla \(p. 93\)](#)
- [Use a Perl client \(p. 94\)](#)
- [Post upload processing \(p. 94\)](#)

Use OpenSSH

Use the instructions that follow to transfer files from the command line using OpenSSH.

Note

This client works only with an SFTP-enabled server.

To transfer files over AWS Transfer Family using the OpenSSH command line utility

1. On Linux or macOS, open a command terminal.
2. At the prompt, enter the following command:

```
sftp -i transfer-key sftp_user@service_endpoint
```

In the preceding command, *sftp_user* is the user name and *transfer-key* is the SSH private key. Here, *service_endpoint* is the server's endpoint as shown in the AWS Transfer Family console for the selected server.

An *sftp* prompt should appear.

3. (Optional) To view the user's home directory, enter the following command at the *sftp* prompt:

```
pwd
```

4. To upload a file from your file system to the Transfer Family server, use the *put* command. For example, to upload *hello.txt* (assuming that file is in your current directory on your file system), run the following command at the *sftp* prompt:

```
put hello.txt
```

A message similar to the following appears, indicating that the file transfer is in progress, or complete.

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

```
hello.txt 100% 127 0.1KB/s 00:00
```

Note

After your server is created, it can take a few minutes for the server endpoint hostname to be resolvable by the DNS service in your environment.

Use WinSCP

Use the instructions that follow to transfer files from the command line using WinSCP.

Note

If you are using WinSCP 5.19, you can directly connect to Amazon S3 using your AWS credentials and upload/download files. For more details, see [Connecting to Amazon S3 service](#).

To transfer files over AWS Transfer Family using WinSCP

1. Open the WinSCP client.
2. In the **Login** dialog box, for **File protocol**, choose a protocol: **SFTP** or **FTP**.

If you chose FTP, for **Encryption**, choose one of the following:

- **No encryption** for FTP
 - **TLS/SSL Explicit encryption** for FTPS
3. For **Host name**, enter your server endpoint. The server endpoint is located on the **Server details** page, see [View server details \(p. 108\)](#).
 4. For **Port number**, enter the following:
 - **22** for SFTP
 - **21** for FTP/FTPS
 5. For **User name**, enter the name for the user that you created in [Managing users \(p. 54\)](#).
 6. Choose **Advanced** to open the **Advanced Site Settings** dialog box. In the **SSH** section, choose **Authentication**.
 7. For **Private key file**, browse for and choose the SSH private key file from your file system.

Note

If WinSCP offers to convert your SSH private key to the PPK format, choose **OK**.

8. Choose **OK** to return to the **Login** dialog box, and then choose **Save**.
9. In the **Save session as site** dialog box, choose **OK** to complete your connection setup.
10. In the **Login** dialog box, choose **Tools**, and then choose **Preferences**.
11. In the **Preferences** dialog box, for **Transfer**, choose **Endurance**.

For the **Enable transfer resume/transfer to temporary filename for** option, choose **Disable**.

Note

If you leave this option enabled, it increases upload costs, substantially decreasing upload performance. It also can lead to failures of large file uploads.

12. For **Transfer**, choose **Background**, and clear the **Use multiple connections for single transfer** check box.

Note

If you leave this option selected, large file uploads can fail in unpredictable ways. For example, orphaned multipart uploads that incur Amazon S3 charges can be created. Silent data corruption can also occur.

13. Perform your file transfer.

You can use drag-and-drop methods to copy files between the target and source windows. You can use toolbar icons to upload, download, delete, edit, or modify the properties of files in WinSCP.

Note

Because Amazon S3 manages object timestamps, be sure to disable WinSCP timestamp settings before you perform file transfers. To do so, in the **WinSCP Transfer settings** dialog box, disable the **Set permissions** upload option and the **Preserve timestamp** common option.

Use Cyberduck

Use the instructions that follow to transfer files from the command line using Cyberduck.

To transfer files over AWS Transfer Family using Cyberduck

1. Open the [Cyberduck](#) client.
2. Choose **Open Connection**.
3. In the **Open Connection** dialog box, choose a protocol: **SFTP (SSH File Transfer Protocol)**, **FTP-SSL (Explicit AUTH TLS)**, or **FTP (File Transfer Protocol)**.
4. For **Server**, enter your server endpoint. The server endpoint is located on the **Server details** page. For more information, see [View server details \(p. 108\)](#).
5. For **Port number**, enter the following:
 - **22** for SFTP
 - **21** for FTP/FTPS
6. For **Username**, enter the name for the user that you created in [Managing users \(p. 54\)](#).
7. If SFTP is selected, for **SSH Private Key**, choose or enter the SSH private key.
8. Choose **Connect**.
9. Perform your file transfer.

Depending on where your files are, do one of the following:

- In your local directory (the source), choose the files that you want to transfer, and drag and drop them into the Amazon S3 directory (the target).
- In the Amazon S3 directory (the source), choose the files that you want to transfer, and drag and drop them into your local directory (the target).

Use FileZilla

Use the instructions that follow to transfer files using FileZilla.

To set up FileZilla for a file transfer

1. Open the FileZilla client.
2. Choose **File**, and then choose **Site Manager**.
3. In the **Site Manager** dialog box, choose **New site**.
4. On the **General** tab, for **Protocol**, choose a protocol: **SFTP** or **FTP**.

If you chose FTP, for **Encryption**, choose one of the following:

- **Only use plain FTP (insecure)** – for FTP
 - **Use explicit FTP over TLS if available** – for FTPS
5. For **Host name**, enter the protocol that you are using, followed by your server endpoint. The server endpoint is located on the **Server details** page. For more information, see [View server details \(p. 108\)](#).
- If you are using SFTP, enter: `sftp://hostname`
 - If you are using FTPS, enter: `ftps://hostname`
- Make sure to replace `hostname` with your actual server endpoint.
6. For **Port number**, enter the following:
- **22** for SFTP
 - **21** for FTP/FTPS
7. If SFTP is selected, for **Logon Type**, choose **Key file**.
For **Key file**, choose or enter the SSH private key.
8. For **User**, enter the name for the user that you created in [Managing users \(p. 54\)](#).
9. Choose **Connect**.
10. Perform your file transfer.

Note

If you interrupt a file transfer in progress, AWS Transfer Family might write a partial object in your Amazon S3 bucket. If you interrupt an upload, check that the file size in the Amazon S3 bucket matches the file size of the source object before continuing.

Use a Perl client

If you use the `NET::SFTP::Foreign` perl client, you must set the `queue_size` to 1. For example:

```
my $sftp = Net::SFTP::Foreign->new('user@s-12345.server.transfer.us-east-2.amazonaws.com', queue_size => 1);
```

Note

This workaround is needed for revisions of `Net::SFTP::Foreign` prior to [1.92.02](#).

Post upload processing

You can view post upload processing information including Amazon S3 object metadata and event notifications.

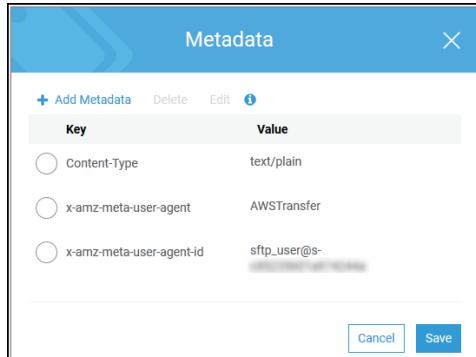
Topics

- [Amazon S3 object metadata \(p. 94\)](#)
- [Amazon S3 event notifications \(p. 95\)](#)

Amazon S3 object metadata

As a part of your object's metadata you will see a key called `x-amz-meta-user-agent` whose value is `AWSTransfer` and `x-amz-meta-user-agent-id` whose value is `username@server-id`. The

`username` is the Transfer Family user who uploaded the file and `server-id` is the server used for the upload. This information can be accessed using the [HeadObject](#) operation on the S3 object inside your Lambda function.



Amazon S3 event notifications

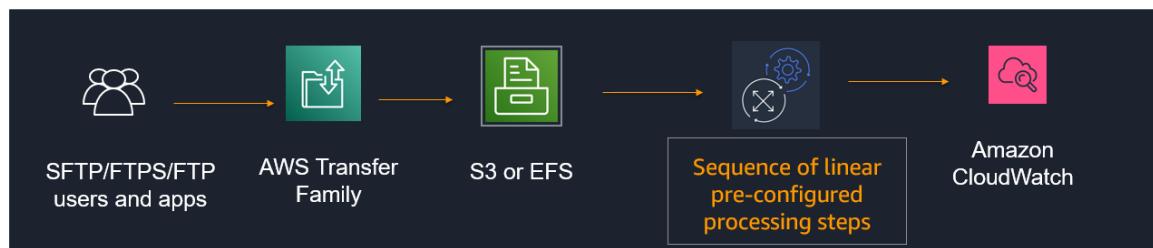
When an object is uploaded to your S3 bucket using Transfer Family, `RoleSessionName` is contained in the `Requester` field in the [S3 event notification structure](#) as `[AWS:Role Unique Identifier]/username.sessionid@server-id`. For example, the following are the contents for a sample Requester field from an S3 access log for a file that was copied to the S3 bucket.

```
arn:aws:sts::AWS-Account-ID:assumed-role/IamRoleName/username.sessionid@server-id
```

In the Requester field above, it shows the IAM Role called `IamRoleName`. For more information about configuring S3 event notifications, see [Configuring Amazon S3 event notifications](#) in the *Amazon Simple Storage Service Developer Guide*. For more information about AWS Identity and Access Management (IAM) role unique identifiers, see [Unique identifiers](#) in the *AWS Identity and Access Management User Guide*.

Managing workflows for post-upload processing

AWS Transfer Family supports managed workflows for file processing, enabling you to create, automate, and monitor file transfers over SFTP, FTPS, and FTP. Using this feature, you can securely and cost effectively meet your compliance requirements for Business to Business (B2B) file exchanges by coordinating all the necessary steps required for file processing, while benefiting from end-to-end auditing and visibility.



Managing your post-processing workflows using Transfer Family:

- is easy to set up and customize
- is auditable
- supports a library of commonly used operations that you can use individually or chain together
- supports searching and viewing the real-time status of in-progress and completed workflows

First, set up your workflow by defining a sequence of action steps such as copying, tagging, or your own custom step based on your requirements. Next, map the workflow to one of your AWS Transfer Family's managed file servers, so upon file arrival, actions specified in this workflow are evaluated and triggered in real-time.

Using the AWS Transfer Family console, you can search for and view the real-time status of in-progress workflows. You can view completed workflows in CloudWatch logs.

Topics

- [Create a workflow \(p. 96\)](#)
- [Configure and execute a workflow \(p. 97\)](#)
- [View workflow details \(p. 98\)](#)
- [Create a custom workflow step \(p. 100\)](#)
- [Exception handling for a workflow \(p. 104\)](#)
- [Choose pre-defined steps \(p. 104\)](#)
- [CloudWatch logging for a workflow \(p. 105\)](#)
- [Managed workflows restrictions and limitations \(p. 106\)](#)

Create a workflow

When you're creating workflows, keep in mind that the maximum number of workflows per account is 10, and the maximum number of steps per workflow is 8.

Create a workflow

1. Open the AWS Transfer Family console at <https://console.aws.amazon.com/transfer/>.
2. In the left navigation pane, choose **Workflows**.
3. On the **Workflows** page, choose **Create workflow**.
4. On the **Provide workflow description** page, enter a description. This description appears on the **Workflows** page.
5. On the **Configure nominal steps** page, choose **Add step**. Add one or more steps.
 - a. Choose a step type. Choose from the available options.
 - b. Choose **Next**, then configure parameters for the step.
 - c. Choose **Next**, then review the details for the step.
 - d. Choose **OK** to add the step and continue.
 - e. Continue adding steps as needed. The maximum number of steps in a workflow is 8.
 - f. After you have added all the necessary steps, choose **Next** to continue to **Configure Exception Handlers**.

Note

We recommend that you set up Exception Handlers and steps to execute when your workflow fails, so you are informed of failures in real-time

6. You configure exception handlers by adding steps, in the same manner as described previously. If a file causes any step to throw an exception, your exception handlers are invoked one by one.
7. Review the configuration, and choose **Create workflow**.

Configure and execute a workflow

Configure Transfer Family to run a workflow on uploaded files

1. Open the AWS Transfer Family console at <https://console.aws.amazon.com/transfer/>.
2. In the left navigation pane, choose **Servers**. Choose the server that you want to use for your workflow.
3. On the details page for the server, scroll down to the **Additional details** section, and then choose **Edit**.

Note

By default, servers do not have any associated workflows. You use the **Additional details** section to associate a workflow with the selected server.

4. On the **Edit additional details** page, in the **Post-upload file-processing** section, select a workflow to be run on all uploads.

Note

If you do not already have a workflow, use the **Create a Workflow** link to create one.

- a. Select the workflow ID to use, and then choose **Save**.
- b. Choose an execution role. This is the role that Managed File Transfer Workflows assumes when executing the workflow's steps. For details, see [Construct an execution role for workflows \(p. 106\)](#).

Post-upload processing Info

Workflow
Select the workflow that Transfer should run on all files after they've been uploaded via this server

w-abcdef01234567890 ▼ C Create a new Workflow ↗

Role
Select the role that Transfer should assume when executing a workflow

transfer-workflows ▼ C

Cancel Save

Execute a workflow

To execute a workflow, you upload a file to a Transfer Family server that you configured with an associated workflow.

Example

```
# Execute a workflow
> sftp bob@s-1234567890abcdef0.server.transfer.us-east-1.amazonaws.com

Connected to s-1234567890abcdef0.server.transfer.us-east-1.amazonaws.com.
sftp> put doc1.pdf
Uploading doc1.pdf to /my-cool-bucket/home/users/bob/doc1.pdf
doc1.pdf                                         100% 5013KB
  601.0KB/s   00:08
sftp> exit
>
```

After your file has been uploaded, the action defined is performed on your file. For example, if your workflow contains a copy step, the file is copied to the location that you defined in that step. You can use Amazon CloudWatch Logs to track steps that executed and their status after your execution completes.

View workflow details

View workflow details

You can view details pertaining to previously created workflows or to workflow executions. To view these details, you can use the console or the AWS Command Line Interface (AWS CLI).

Console

View workflow details

1. Open the AWS Transfer Family console at <https://console.aws.amazon.com/transfer/>.
2. In the left navigation pane, choose **Workflows**.
3. On the **Workflows** page, choose a workflow.

The workflow details page opens.

The screenshot shows the AWS Transfer Family Workflows details page. The workflow is named "Test workflow A". It contains one nominal step, "tag_step", which is a TAG type step. It also contains one exception handler, "delete_if_exception", which is a DELETE type handler. There are no in-flight executions listed.

CLI

To view the workflow details, use the `describe-workflow` CLI command, as shown in the following example.

```
# View Workflow details
> aws transfer describe-workflow --workflow-id w-1234567890abcdef0
{
    "Workflow": {
        "Arn": "arn:aws:transfer:us-east-1:111122223333:workflow/w-1234567890abcdef0",
        "WorkflowId": "w-1234567890abcdef0",
        "Name": "Copy file to shared_files",
        "Steps": [
            {
                "Type": "COPY",
                "CopyStepDetails": {
                    "Name": "Copy to shared",
                    "FileLocation": {
                        "S3FileLocation": {
                            "Bucket": "my-cool-bucket",
                            "Key": "home/shared_files/"
                        }
                    }
                }
            }
        ],
        "OnException": {}
    }
}
```

Create a custom workflow step

With a custom workflow step, you must configure the Lambda function to call the [SendWorkflowStepState](#) API in order to notify the execution that the step completed with either a success or a failure. The response returned from the Lambda function cannot update the state of the step. If the Lambda function fails or times out, the step fails, and you see `StepErrored` in your CloudWatch logs. If the Lambda function is part of the nominal step and you send a failure on the API, or if it fails or times out, the flow continues with the Exception handler steps: the workflow does not continue to execute the remaining (if any) nominal steps.

When you call the [SendWorkflowStepState](#) API you must send the following parameters:

```
{  
    "ExecutionId": "string",  
    "Status": "string",  
    "Token": "string",  
    "WorkflowId": "string"  
}
```

You can extract the `ExecutionId`, `Token`, and `WorkflowId` from the input event that is passed when the Lambda function executes (examples are shown below). The status can be either `SUCCESS` or `FAILURE`.

To be able to call the [SendWorkflowStepState](#) from your Lambda function, you need to use the latest version of the AWS SDK. To use the latest version of the AWS SDK, you can deploy a Lambda function with the dependencies by using a ZIP archive (see [custom package](#) in the *AWS Lambda Developer Guide*) or you can [create a Lambda layer](#) to integrate the latest version of the AWS SDK to the Lambda function.

Furthermore, the Lambda execution role must have permissions to call the [SendWorkflowStepState](#) API, otherwise it fails. You can add the following policy to your Lambda execution role:

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "VisualEditor0",  
            "Effect": "Allow",  
            "Action": "transfer:SendWorkflowStepState",  
            "Resource": "*"  
        }  
    ]}  
}
```

Example Lambda function for a custom workflow step

The following Lambda function extracts the information regarding the execution and calls the [SendWorkflowStepState](#) API to return the status to the workflow for the step—`SUCCESS` or `FAILURE`. Before your function calls the [SendWorkflowStepState](#) API, you can configure Lambda to take an action based on your workflow logic.

```
import json  
import boto3  
  
transfer = boto3.client('transfer')  
  
def lambda_handler(event, context):  
    print(json.dumps(event))
```

```

# call the SendWorkflowStepState API to notify workflows for an update on the step with
a SUCCESS or a FAILURE
response = transfer.send_workflow_step_state(
    WorkflowId=event['serviceMetadata']['executionDetails']['workflowId'],
    ExecutionId=event['serviceMetadata']['executionDetails']['executionId'],
    Token=event['token'],
    Status='SUCCESS|FAILURE'
)
print(json.dumps(response))

return {
    'statusCode': 200,
    'body': json.dumps(response)
}

```

Example custom step

The following examples show the code for a custom step. One example uses a Transfer Family server where the domain is configured with Amazon S3 and one where the domain uses Amazon EFS.

Custom step Amazon S3 domain

```
{
    "token": "MzI0Nzc4ZDktMGRmMi00MjFhLTgxMjUtYWZmZmRmODNkYjc0",
    "serviceMetadata": {
        "executionDetails": {
            "workflowId": "w-1234567890example",
            "executionId": "abcd1234-aa11-bb22-cc33-abcdef123456"
        },
        "transferDetails": {
            "sessionId": "36688ff5d2deda8c",
            "userName": "myuser",
            "serverId": "s-example1234567890"
        }
    },
    "fileLocation": {
        "domain": "S3",
        "bucket": "mybucket",
        "key": "path/to/mykey",
        "eTag": "d8e8fcfa2dc0f896fd7cb4cb0031ba249",
        "versionId": null
    }
}
```

Custom step Amazon EFS domain

```
{
    "token": "MTg0N2Y3N2UtNWI5Ny00ZmZlTk5YTgtZTU3YzViYjllNmZm",
    "serviceMetadata": {
        "executionDetails": {
            "workflowId": "w-1234567890example",
            "executionId": "abcd1234-aa11-bb22-cc33-abcdef123456"
        },
        "transferDetails": {
            "sessionId": "36688ff5d2deda8c",
            "userName": "myuser",
            "serverId": "s-example1234567890"
        }
    },
    "fileLocation": {
```

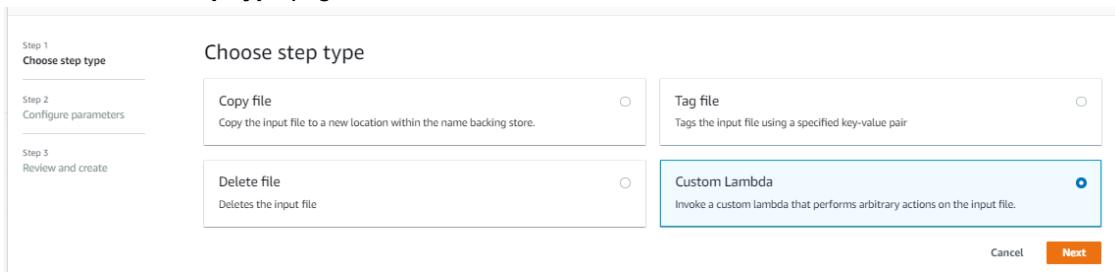
```
        "domain": "EFS",
        "fileSystemId": "fs-1234567",
        "path": "/path/to/myfile"
    }
```

Create workflow procedure

To create a custom workflow step, you specify the Amazon Resource Name (ARN) for the Lambda function. With a custom workflow step, you can write your own code in AWS Lambda to process your incoming files. After incoming files are uploaded to Amazon S3 or Amazon EFS, AWS Transfer Family invokes a Lambda function that you provide using a custom step.

Create a custom workflow step

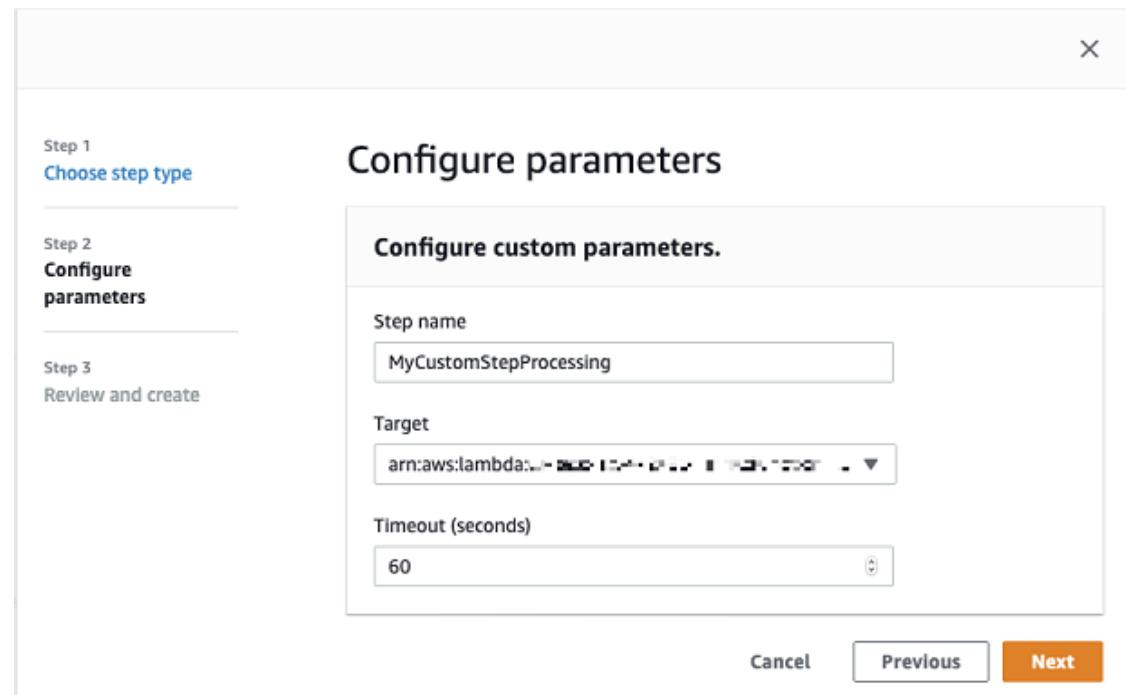
1. Open the AWS Transfer Family console at <https://console.aws.amazon.com/transfer/>.
2. In the left navigation pane, choose **Workflows**.
3. On the **Workflows** page, choose **Create workflow**.
4. Enter a description for the workflow, and then choose **Add step** in the **Nominal steps** section.
5. On the **Choose step type** page, choose **Custom Lambda**, and then choose **Next**.



6. On the **Configure parameters** page, enter a name for the step, the Amazon Resource Name (ARN) for the Lambda function to execute, and a timeout value, in minutes. The maximum value for the timeout is 30.

Note

In order to complete this step, you must have a functional Lambda code ready.



7. Choose **Next**, and then choose **Create step**.

The following sample AWS Lambda code takes the workflow event data and sends it to Amazon EventBridge, so that you can integrate the workflow event with your other applications.

```
import json, boto3
def lambda_handler(event, context):
    # TODO implement
    print (event)
    client = boto3.client('events')
    event_serviceMetadata_transferDetails_serverid = event['serviceMetadata']
    ['transferDetails']['serverId']
    response = client.put_events(
        Entries=[
            {
                'Source': 'dev.transfer',
                'Resources': [
                    event_serviceMetadata_transferDetails_serverid,
                ],
                'DetailType': 'Event message relayed from an AWS Transfer Workflow customstep',
                'Detail': json.dumps(event)
            },
        ]
    )

    return {
        'statusCode': 200,
        'body': json.dumps(response)
    }
```

Exception handling for a workflow

You specify the error-handling steps for a workflow in the same manner as you specify the workflow steps themselves. If any errors occur during the workflow execution, the error-handling steps that you specified are executed. For information about troubleshooting workflows, see [Troubleshoot workflow-related errors using CloudWatch \(p. 154\)](#).

Choose pre-defined steps

Copy, tag, and delete steps are pre-defined actions for post-upload file-processing. For example, you can create a workflow step to copy your uploaded file to another location. You can create this step for servers that use Amazon S3.

Note

Currently, copying and tagging are supported only on Amazon S3.

If your server uses Amazon S3, you must provide the bucket name and a key for the destination of the copy. The key can be either a path name or a file name. Whether the key is treated as a path name or a file name is determined by whether you end the key with the forward slash (/) character.

If the final character is /, your file is copied to the folder, and its name does not change. If the final character is alphanumeric, your uploaded file is renamed to the key value. In this case, if a file with that name already exists, the existing file is overwritten with the new one. For example, if your key value is `shared-files/`, your uploaded files are copied to the `shared-files` folder. If your key value is `shared-files/today`, every file you upload is copied to a file named `today` in the `shared-files` folder, and each succeeding file overwrites the previous one.

Example copy step

For example, consider the following workflow steps file:

```
[  
  {  
    "Type": "COPY",  
    "CopyStepDetails": {  
      "Name": "copyToShared",  
      "DestinationFileLocation": {  
        "S3FileLocation": {  
          "Bucket": "bob-bucket",  
          "Key": "shared_files/"  
        }  
      }  
    }  
  }  
]
```

After you create a workflow from this file, and then attach it to a Transfer Family server, all uploaded files are copied to the `/bob-bucket/shared-files` folder.

Parametrize the destination folder

You can also parametrize the destination by username. To do this, you set the Key to `#{transfer:UserName}`. In this case, the `DestinationFileLocation` code looks similar to this:

```
"DestinationFileLocation": {  
  "S3FileLocation": {
```

```
    "Bucket": "main-bucket",
    "Key": "${transfer:UserName}/processed/"
}
```

In this example, the destination folder for each authorized user is `main-bucket/user-id/processed/`. So, for a user whose ID is **bob-usa**, their files get copied into `main-bucket/bob-usa/processed/`.

CloudWatch logging for a workflow

Amazon CloudWatch monitors your AWS resources and the applications that you run in the AWS Cloud in real time. You can use CloudWatch to collect and track metrics, which are variables that you can measure for your workflows. CloudWatch provides consolidated auditing and logging for workflow progress and results.

View Amazon CloudWatch logs for workflows

1. Open the Amazon CloudWatch console at <https://console.aws.amazon.com/cloudwatch/>.
2. In the left navigation pane, choose **Logs** then choose **Log groups**.
3. On the **Log groups** page, on the navigation bar, choose the correct region for your Transfer Family server.
4. Choose the log group that corresponds to your server.

For example, if your server ID is `s-1234567890abcdef0`, your log group is `/aws/transfer/s-1234567890abcdef0`.

5. On the log page for your server, the most recent log streams are displayed. There are log streams for the user that you are exploring: one for each sftp session, as well as a log stream for the workflow that is being executed for your server. The format for the log stream for the workflow is `username.workflowID.uniqueStreamSuffix`.

For example, if your user is `bob-usa`, you have the following log streams:

```
bob-usa-east.1234567890abcdef0
bob.w-abcdef01234567890.021345abcdef6789
```

Note

The 16-digit alphanumeric identifiers listed in this example are fictitious : the values you see in Amazon CloudWatch are different.

The **Log events** page for `bob-usa-east.1234567890abcdef0` displays the details for user session, and the `bob.w-abcdef01234567890.021345abcdef6789` log stream contains the details for the workflow.

The following is a sample log stream for `bob.w-abcdef01234567890.021345abcdef6789`, based on a workflow (`w-abcdef01234567890`) that contains a copy step.

```
{"type":"ExecutionStarted","details":
  {"input":
    {"initialFileLocation":{"bucket":"my-bucket","key":"bob/
workflowSteps2.json","versionId":"version-id","etag":"etag-id"}
  }
},
"workflowId":"w-abcdef01234567890","executionId":"execution-id",
"transferDetails":{"serverId":"s-server-id","username":"bob","sessionId":"session-id"}
}
{"type":"StepStarted","details":
```

```
{
    "input": {
        "fileLocation": {
            "backingStore": "S3", "bucket": "my-bucket", "key": "bob/workflowSteps2.json", "versionId": "version-id", "etag": "etag-id"}
        },
        "stepType": "COPY", "stepName": "copyToShared"}, "workflowId": "w-abcdef01234567890", "executionId": "execution-id", "transferDetails": {"serverId": "s-server-id", "username": "bob", "sessionId": "session-id"}}
    }
    {"type": "StepCompleted", "details": {
        "output": {}, "stepType": "COPY", "stepName": "copyToShared"}, "workflowId": "w-abcdef01234567890", "executionId": "execution-id", "transferDetails": {"serverId": "s-server-id", "username": "bob", "sessionId": "session-id"}}
    }
    {"type": "ExecutionCompleted", "details": {
        {}, "workflowId": "w-abcdef01234567890", "executionId": "execution-id", "transferDetails": {"serverId": "s-server-id", "username": "bob", "sessionId": "session-id"}}
    }
}
```

Construct an execution role for workflows

When you add a workflow to a server, you need to select an execution role. The server uses this role when it executes the workflow. If the role does not have the proper permissions, Transfer Family cannot run the workflow. This section describes one possible set of permissions that can be used to execute a workflow.

- Create a new role, and add the AWS managed policy, **AWSTransferFullAccess**, to the role.
- Create another policy with the following permissions, and attach it to your role.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:GetObject",
                "s3>DeleteObjectVersion",
                "s3>ListBucket",
                "s3>DeleteObject",
                "s3:GetBucketLocation",
                "s3:GetObjectVersion",
                "s3:PutObjectTagging"
            ],
            "Resource": "*"
        }
    ]
}
```

Save this role and specify it as the execution role when you add a workflow to a server.

Managed workflows restrictions and limitations

Restrictions

The following restrictions currently apply to post-upload processing workflows for AWS Transfer Family.

- Cross-account and cross-Region Lambda functions are not supported.
- For a copy step, the source and destination Amazon S3 buckets must be in the same Region.
- Only asynchronous custom steps are supported.
- Custom step timeouts are approximate. That is, it may take slightly longer to time out than specified. Additionally, the workflow is dependent upon the Lambda function. Therefore, if the function is delayed during execution, the workflow is not aware of the delay.
- The only supported outputs from a custom step are `Success` or `Failure`.
- Copying across storage services and Regions is not supported. You can, however, copy across accounts, provided that your AWS Identity and Access Management (IAM) policies are correctly configured.
- If you exceed your throttling limit, Transfer Family doesn't add workflow operations to the queue.
- Workflows are triggered upon successful file upload—end of STOR data stream for FTP, or CLOSE after write for SFTP—and not on partial upload (for example an abrupt disconnect without closing).
- Workflows are not triggered for files that have a size of 0. Files with a size greater than 0 do trigger the associated workflow.

Limitations

Additionally, the following functional limits apply to workflows for Transfer Family:

- The number of workflows per account is limited to 10.
- The maximum timeout for custom steps is 30 minutes.
- The maximum number of steps in a workflow is 8.
- The maximum number of tags per workflow is 50.

Managing servers

In this section, you can find information on how to view a list of your servers, how to view your server details, how to edit your server details, and how to change the host key for your SFTP-enabled server.

Topics

- [View a list of servers \(p. 108\)](#)
- [View server details \(p. 108\)](#)
- [Edit server details \(p. 109\)](#)
- [Monitoring server usage in the console \(p. 116\)](#)
- [Delete a server \(p. 117\)](#)

View a list of servers

On the AWS Transfer Family console, you can find a list of all your servers that are located in the AWS Region that you chose.

To find a list of your servers that exist in an AWS Region

- Open the AWS Transfer Family console at <https://console.aws.amazon.com/transfer/>.

If you have one or more servers in the current AWS Region, the console opens to show a list of your servers. If you don't see a list of servers, make sure that you are in the correct Region. You can also choose **Servers** from the navigation pane.

For more information about viewing your server details, see [View server details \(p. 108\)](#).

View server details

You can find a list of details and properties for an individual AWS Transfer Family server. Server properties include protocols, identity provider, status, endpoint type, custom hostname, endpoint, users, logging role, server host key, and tags.

To view server details

1. Open the AWS Transfer Family console at <https://console.aws.amazon.com/transfer/>.
2. In the navigation pane, choose **Servers**.
3. Choose the identifier in the **Server ID** column to see the **Server details** page, shown following.

You can change the server's properties on this page by choosing **Edit**. For more information about editing server details, see [Edit server details \(p. 109\)](#).

The screenshot shows the 'Edit server details' page for an AWS Transfer Family server. The interface is divided into several sections:

- Protocols:** Shows SFTP as the selected protocol. An 'Edit' button is available.
- Identity provider:** Shows 'Service managed' as the provider type.
- Endpoint details:** Displays the status as 'Online', endpoint type as 'Public', and FIPS Enabled status as 'No'. It also shows a custom hostname and the full endpoint URL (s-26bee3cf4a145fb.server.transfer.us-east-2.amazonaws.com).
- Users (1):** Lists a single user named 'scooter' with a home directory of '/homebucket-scooter' and a role of 'scooter-transfer-s3'.
- Additional details:** Includes logging role (AWSTransferLoggingAccess), server host key (SHA256: [redacted]), post-upload processing, security policy (TransferSecurityPolicy-2020-06), domain (Amazon S3), and post-upload processing execution role.
- Monitoring:** Provides a line chart showing BytesIn, BytesOut, FilesIn, and FilesOut over time (20:15 to 21:00). The chart shows minimal activity with values around 0.50 to 1.00.
- Tags (0):** A section for adding tags, currently empty.

Edit server details

After you create an AWS Transfer Family server, you can edit the server configuration.

Topics

- [Edit the file transfer protocols \(p. 111\)](#)
- [Edit the server identity provider \(p. 112\)](#)
- [Edit the server endpoint \(p. 113\)](#)
- [Edit Amazon CloudWatch logging \(p. 113\)](#)
- [Edit the security policy \(p. 114\)](#)
- [Change the host key for your SFTP-enabled server \(p. 114\)](#)
- [Put your server online or offline \(p. 115\)](#)

To edit a server's configuration

1. Open the AWS Transfer Family console at <https://console.aws.amazon.com/transfer/>.

2. In the navigation pane, choose **Servers**.
3. Choose the identifier in the **Server ID** column to see the **Server details** page, shown following.

You can change the server's properties on this page by choosing **Edit**:

- To change the protocols, see [Edit the file transfer protocols \(p. 111\)](#).
- To change the identity provider, see [Edit the server identity provider \(p. 112\)](#).
- To change the endpoint type or custom hostname, see [Edit the server endpoint \(p. 113\)](#).
- To change the logging role, see [Edit Amazon CloudWatch logging \(p. 113\)](#).
- To change the security policy, see [Edit the security policy \(p. 114\)](#).
- To change the server host key, see [Change the host key for your SFTP-enabled server \(p. 114\)](#).
- To start or stop your server, see [Put your server online or offline \(p. 115\)](#).
- To delete a server, see [Delete a server \(p. 117\)](#).
- To edit a user's properties, see [Managing access controls \(p. 118\)](#).

The screenshot shows the AWS Transfer Family Server details page with the following sections:

- Protocols**: Shows SFTP as the protocol. An **Edit** button is available.
- Identity provider**: Shows Service managed.
- Endpoint details**: Status is Online, Endpoint type is Public, FIPS Enabled is No. It lists a custom hostname and an endpoint URL.
- Users (1)**: Shows one user named scooter with a home directory of /homebucket-scooter and a role of scooter-transfer-s3.
- Additional details**: Shows Logging role (AWSTransferLoggingAccess), Server host key (SHA256: [REDACTED]), Post-upload processing, Security Policy (TransferSecurityPolicy-2020-06), Domain (Amazon S3), and Post-upload processing execution role.
- Monitoring**: Shows BytesIn, BytesOut, FilesIn, and FilesOut metrics over time from 20:15 to 21:00.
- Tags (0)**: Shows an empty tags section.

Edit the file transfer protocols

On the AWS Transfer Family console, you can edit the file transfer protocol. The file transfer protocol connects the client to your server's endpoint.

To edit the protocols

1. On the **Server details** page, choose **Edit** next to **Protocols**.
2. On the **Edit protocols** page, select or clear the protocol check box or check boxes to add or remove the following file transfer protocols:

- Secure Shell (SSH) File Transfer Protocol (SFTP) – file transfer over SSH

For more information about SFTP, see [Create an SFTP-enabled server \(p. 20\)](#).

- File Transfer Protocol Secure (FTPS) – file transfer with TLS encryption

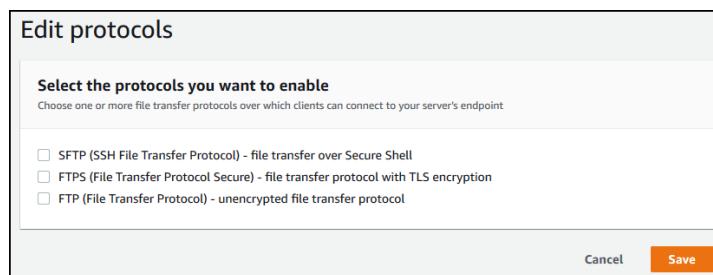
For more information about FTP, see [Create an FTPS-enabled server \(p. 25\)](#).

- File Transfer Protocol (FTP) – unencrypted file transfer

For more information about FTPS, see [Create an FTP-enabled server \(p. 30\)](#).

Note

If you have an existing server enabled only for SFTP, and you want to add FTPS and FTP, you must ensure that you have the right identity provider and endpoint type settings that are compatible with FTPS and FTP.



If you select **FTPS**, you must choose a certificate stored in AWS Certificate Manager (ACM) which will be used to identify your server when clients connect to it over FTPS.

To request a new public certificate, see [Request a public certificate](#) in the *AWS Certificate Manager User Guide*.

To import an existing certificate into ACM, see [Importing certificates into ACM](#) in the *AWS Certificate Manager User Guide*.

To request a private certificate to use FTPS through private IP addresses, see [Requesting a Private Certificate](#) in the *AWS Certificate Manager User Guide*.

Certificates with the following cryptographic algorithms and key sizes are supported:

- 2048-bit RSA (RSA_2048)
- 4096-bit RSA (RSA_4096)
- Elliptic Prime Curve 256 bit (EC_prime256v1)
- Elliptic Prime Curve 384 bit (EC_secp384r1)
- Elliptic Prime Curve 521 bit (EC_secp521r1)

Note

The certificate must be a valid SSL/TLS X.509 version 3 certificate with FQDN or IP address specified and information about the issuer.

Choose protocols

Select the protocols you want to enable
Choose one or more file transfer protocols over which clients can connect to your server's endpoint

SFTP (SSH File Transfer Protocol) - file transfer over Secure Shell
 FTPS (File Transfer Protocol Secure) - file transfer protocol with TLS encryption
 FTP (File Transfer Protocol) - unencrypted file transfer protocol

AWS Certificate Manager (ACM) certificate [Info](#)

Server certificate
Choose a certificate stored in ACM which will be used to identify your server when clients connect to it over FTPS

Choose a certificate [Select](#)

Cancel **Next**

3. Choose **Save**. You are returned to the **Server details** page.

Edit the server identity provider

On the AWS Transfer Family console, you can edit the details of your identity provider such as the API Gateway URL and invocation role. The identity provider manages user access for authentication and authorization.

Note

You can't change a server's identity provider type after you create the server. To change the identity provider, delete the server and create a new one with the identity provider that you want.

To edit the server identity provider

1. On the **Server details** page, choose **Edit** next to **Identity provider**.
2. On the **Edit identity provider** page, choose one of the following identity provider types:
 - **Service managed** – creates, manages, and stores user identities and keys in AWS Transfer Family.
 - **Custom** – you must provide an Amazon API Gateway URL and an AWS Identity and Access Management (IAM) role for the service to invoke your Amazon API Gateway URL endpoint. To learn more about working with custom identity providers, see [Working with custom identity providers \(p. 69\)](#).

Edit identity provider

Identity provider

Identity provider type
An identity provider manages user access for authentication and authorization

Service managed
Create and manage users within the service

Custom [Info](#)
Provide an Amazon API Gateway URL

Cancel **Save**

3. Choose **Save**. You are returned to the **Server details** page.

Edit the server endpoint

On the AWS Transfer Family console, you can modify the server endpoint type and custom hostname.

To edit the server endpoint details

1. On the **Server details** page, choose **Edit** next to **Endpoint details**.
2. On the **Edit endpoint configuration** page, for **Endpoint type**, choose one of the following:
 - **Public** – makes your server accessible over the internet.
 - **VPC** – makes your server accessible in your virtual private cloud (VPC). For information about VPC, see [Create a server in a virtual private cloud \(p. 35\)](#).
3. For **Custom hostname**, choose one of the following:
 - **None** – if you don't want to use a custom domain.

You get a server hostname provided by AWS Transfer Family. The server hostname takes the form `serverId.server.transfer.regionId.amazonaws.com`.

- **Amazon Route 53 DNS alias** – to use a DNS alias automatically created for you in Route 53.
- **Other DNS** – to use a hostname that you already own in an external DNS service.

Choosing **Amazon Route 53 DNS alias** or **Other DNS** specifies the name resolution method to associate with your server's endpoint.

For example, your custom domain might be `sftp.inbox.example.com`. A custom hostname uses a DNS name that you provide and that a DNS service can resolve. You can use Route 53 as your DNS resolver, or use your own DNS service provider. To learn how AWS Transfer Family uses Route 53 to route traffic from your custom domain to the server endpoint, see [Working with custom hostnames \(p. 47\)](#).

The screenshot shows the 'Edit endpoint configuration' dialog. It has a title bar 'Edit endpoint configuration'. Under 'Endpoint configuration', there is a section for 'Endpoint type' with two radio buttons: 'Public' (selected) and 'VPC'. Below that is a section for 'Custom hostname' with a dropdown menu showing 'None'. At the bottom right are 'Cancel' and 'Save' buttons.

4. Choose **Save**. You are returned to the **Server details** page.

Edit Amazon CloudWatch logging

On the AWS Transfer Family console, you can enable Amazon S3 event logging using Amazon CloudWatch.

Note

If Transfer Family created a CloudWatch logging IAM role for you when you created a server, the IAM role is called `AWSTransferLoggingAccess`. You can use it for all your servers.

To edit the CloudWatch logging IAM role

1. On the **Server details** page, choose **Edit** next to **Additional details**.

2. On the **CloudWatch logging** page, do one of the following:
 - If Transfer Family created a CloudWatch logging IAM role for you when you created a server, the IAM role is called `AWSTransferLoggingAccess`. Choose it from the **Logging role** list.
 - If you chose an existing CloudWatch logging IAM role or you didn't choose a CloudWatch logging IAM role at all when you created this server, choose or modify the CloudWatch logging IAM role from the **Logging role** list.

For more information about CloudWatch logging, see [Log activity with CloudWatch \(p. 122\)](#).

Note

You can't view end-user activity in CloudWatch if you don't specify a logging role.

Edit additional details

CloudWatch logging [Info](#)

Logging role - optional
IAM role for the server to push events to your CloudWatch logs

AWSTransferLoggingAccess

C

3. Choose **Save**. You are returned to the **Server details** page.

Edit the security policy

On the AWS Transfer Family console, you can modify the security policy attached to your server.

To edit the security policy

1. On the **Server details** page, choose **Edit** next to **Additional details**.
2. On the **Cryptographic algorithm options** page, choose a security policy that contains the cryptographic algorithms enabled for use by your server.

Note

If your endpoint is FIPS-enabled, you can't change the FIPS security policy.

For more information about security policies, see [Working with security policies \(p. 49\)](#).

Cryptographic algorithm options [Info](#)

Security Policy
Choose a security policy that contains the cryptographic algorithms enabled for use by your server

TransferSecurityPolicy-2020-06

C

3. Choose **Save**. You are returned to the **Server details** page.

Change the host key for your SFTP-enabled server

Important

If you aren't planning to migrate existing users from an existing SFTP-enabled server to a new SFTP-enabled server, ignore this section. Accidentally changing a server's host key can be disruptive.

By default, AWS Transfer Family provides a host key for your SFTP-enabled server. You can replace the default host key with a host key from another server. Do so only if you plan to move existing users from an existing SFTP-enabled server to your new SFTP-enabled server.

To prevent your users from getting notified to verify the authenticity of your SFTP-enabled server again, import the host key for your on-premises server to the SFTP-enabled server. Doing this also prevents your users from getting a warning about a potential man-in-the-middle attack.

On the AWS Transfer Family console, you can change the server host key.

To change the server host key

1. On the **Server details** page, choose **Edit** next to **Additional details**.
2. On the **Server Host Key** page, enter an RSA private key that will be used to identify your server when clients connect to it over the SFTP-enabled server.
3. Choose **Save**. You are returned to the **Server details** page.

To change the host key using the AWS CLI, use the [the section called “UpdateServer” \(p. 262\)](#) API operation and provide the new host key. If you create a new SFTP-enabled server, you provide your host key as a parameter in the [the section called “CreateServer” \(p. 166\)](#) API operation. You can also use the AWS CLI to update the host key.

The following example updates the host key for the specified SFTP-enabled server.

```
aws transfer update-server --server-id "your-server-id" --host-key file://my-host-key
{
    "ServerId": "server-id"
}
```

Put your server online or offline

On the AWS Transfer Family console, you can bring your server online or take it offline.

To bring your server online

1. Open the AWS Transfer Family console at <https://console.aws.amazon.com/transfer/>.
2. In the navigation pane, choose **Servers**.
3. Select the check box of the server that is offline.
4. For **Actions**, choose **Start**.

It can take a couple of minutes for a server to switch from offline to online.

Note

When you stop a server to take it offline, currently you are still accruing service charges for that server. To eliminate additional server-based charges, delete that server.

To take your server offline

1. Open the AWS Transfer Family console at <https://console.aws.amazon.com/transfer/>.
2. In the navigation pane, choose **Servers**.
3. Select the check box of the server that is online.
4. For **Actions**, choose **Stop**.

While a server is starting up or shutting down, servers aren't available for file operations. The console doesn't show the starting and stopping states.

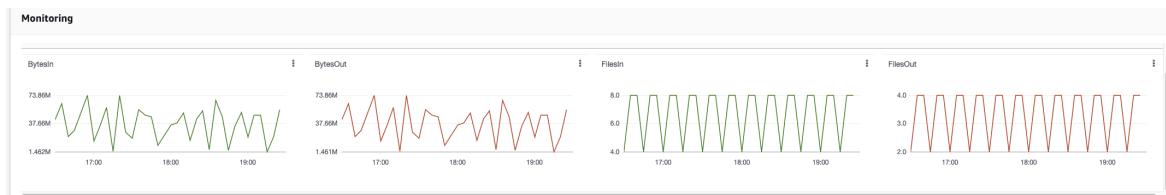
If you find the error condition `START_FAILED` or `STOP_FAILED`, contact AWS Support to help resolve your issues.

Monitoring server usage in the console

You can get information about your server's metrics on its **Server details** page. This provides you with a single place to monitor your file-transfers workloads. You can track how many files you have exchanged with your partners and closely track their usage using a centralized dashboard. For details, see [View server details \(p. 108\)](#). The following table describes the metrics available for Transfer Family.

Namespace	Metric	Description
AWS/Transfer	BytesIn	The total number of bytes transferred into the server. Units: Count Period: 5 minutes
	BytesOut	The total number of bytes transferred out of the server. Unit: Count Period: 5 minutes
	FilesIn	The total number of files transferred into the server. Units: Count Period: 5 minutes
	FilesOut	The total number of files transferred out of the server. Units: Count Period: 5 minutes
	OnUploadExecution	The total number of workflow executions started on the server. Units: Count Period: 1 minute
	OnUploadExecutionSuccess	The total number of successful workflow executions on the server. Units: Count Period: 1 minute
	OnUploadExecutionFailure	The total number of unsuccessful workflow executions on the server. Units: Count Period: 1 minute

The **Monitoring** section contains four, individual graphs.



You can select the expand icon () to open the selected graph in its own window. You can also click a graph's vertical ellipsis () to open a dropdown menu with the following items:

- **Enlarge:** opens the selected graph in its own window.
- **Refresh:** reloads the graph using most recent data.
- **View in metrics:** opens the corresponding metrics details in CloudWatch.
- **View logs:** opens the corresponding log group in CloudWatch.

Delete a server

On the AWS Transfer Family console, you can delete your server.

Important

You are billed, for each of the protocols enabled to access your endpoint, until you delete the server.

Warning

Deleting a server will result in all its users being deleted. Data in the bucket that was accessed using the server will not be deleted and remains accessible to AWS users that have privileges to those S3 buckets.

To delete a server

1. Open the AWS Transfer Family console at <https://console.aws.amazon.com/transfer/>.
2. In the navigation pane, choose **Servers**.
3. Select the check box of the server that you want to delete.
4. For **Actions**, choose **Delete**.
5. In the confirmation dialog box that appears, enter the word **delete**, and then choose **Delete** to confirm that you want to delete the server.

The server is deleted from the **Servers** page and you are no longer billed for it.

Managing access controls

Note

This section applies only to the service-managed identity provider. If you are using a custom identity provider, see [Working with custom identity providers \(p. 69\)](#).

Topics

- Allowing read and write access to an Amazon S3 bucket (p. 118)
 - Creating a session policy for an Amazon S3 bucket (p. 119)
 - Preventing users from creating a directory in an S3 bucket (p. 121)

You can control a user's access to AWS Transfer Family resources by using an AWS Identity and Access Management (IAM) policy. An IAM policy is a statement, typically in JSON format, that allows a certain level of access to a resource. You use an IAM policy to define what file operations that you want to allow your users to perform and not perform. You can also use an IAM policy to define what Amazon S3 bucket or buckets that you want to give your users access to. To specify these policies for users, you create an IAM role for AWS Transfer Family that has the IAM policy and trust relationship associated with it.

Each user is assigned an IAM role. When a user logs in to your server, AWS Transfer Family assumes the IAM role mapped to the user. To learn about creating an IAM role that provides a user access to an Amazon S3 bucket, see following. For information about how to create a role and delegate permissions, see [Creating a role to delegate permissions to an AWS service](#) in the *IAM User Guide*.

The type of IAM role that AWS Transfer Family uses is called a service role.

Note

If your Amazon S3 bucket is encrypted using AWS Key Management Service (AWS KMS), you must specify additional permissions in your policy. For details, see [Data encryption \(p. 126\)](#). Additionally, you can see more information about [session policies](#) in the *IAM User Guide*.

Allowing read and write access to an Amazon S3 bucket

Following, you can see how to create an IAM policy that allows read and write access to a specific Amazon S3 bucket. Assigning an IAM role that has this IAM policy to your user gives that user read/write access to the specified Amazon S3 bucket.

The following policy provides programmatic read and write access to an Amazon S3 bucket.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "ReadWriteS3",  
            "Action": [  
                "s3>ListBucket"  
            ],  
            "Effect": "Allow",  
            "Resource": ["arn:aws:s3:::bucketname"]  
        },  
        {  
            "Effect": "Allow",  
            "Action": [  
                "s3:GetObject",  
                "s3:PutObject",  
                "s3:DeleteObject",  
                "s3:ListBucket"  
            ]  
        }  
    ]  
}
```

```
    "s3:PutObject",
    "s3:GetObject",
    "s3:DeleteObject",
    "s3:DeleteObjectVersion",
    "s3:GetObjectVersion",
    "s3:GetObjectACL",
    "s3:PutObjectACL"
],
"Resource": ["arn:aws:s3:::bucketname/*"]
}
]
```

The `ListBucket` action requires permission to the bucket itself. The `PUT`, `GET`, and `DELETE` actions require object permissions. Because these are different entities, they are specified using different Amazon Resource Names (ARNs).

If your bucket is enabled for AWS Key Management Service (AWS KMS) encryption, you need to enable additional actions in the policy. For more information about AWS KMS, see [What is AWS Key Management Service?](#)

To further restrict your users' access to only the home directory of the specified Amazon S3 bucket, see [Creating a session policy for an Amazon S3 bucket \(p. 119\)](#).

Creating a session policy for an Amazon S3 bucket

A *session policy* is an AWS Identity and Access Management (IAM) policy that restricts users to certain portions of an Amazon S3 bucket. It does so by evaluating access in real time.

Note

Session policies are only used with Amazon S3. For Amazon EFS, you use Posix file permissions to limit access.

You can use a session policy when you need to give the same access to a group of users to a particular portion of your Amazon S3 bucket. For example, a group of users might need access to only the home directory. That group of users share the same IAM role.

Note

The maximum length of a session policy is 2048 characters. For more details, see the [Policy request parameter](#) for the `CreateUser` action in the *API reference*.

To create a session policy, use the following policy variables in your IAM policy:

- `#{transfer:HomeBucket}`
- `#{transfer:HomeDirectory}`
- `#{transfer:HomeFolder}`
- `#{transfer:UserName}`

Note

You can't use the variables listed preceding as policy variables in an IAM role definition. You create these variables in an IAM policy and supply them directly when setting up your user. Also, you can't use the `#{aws:Username}` variable in this session policy. This variable refers to an IAM user name and not the user name required by AWS Transfer Family.

An example of a session policy is shown in the code example following.

```
{
```

```

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

```

Note

In the policy above, it is assumed that users have their home directories set to include a trailing slash, to signify that it is a directory. If, on the other hand, you set a user's `HomeDirectory` without the trailing slash, then you should include it as part of your policy.

In the previous example policy, note the use of the `transfer:HomeFolder`, `transfer:HomeBucket`, and `transfer:HomeDirectory` policy parameters. These parameters are set for the `HomeDirectory` that is configured for the user, as described in [HomeDirectory](#) and [Implementing your API Gateway method \(p. 77\)](#). These parameters have the following definitions:

- The `transfer:HomeBucket` parameter is replaced with the first component of `HomeDirectory`.
- The `transfer:HomeFolder` parameter is replaced with the remaining portions of the `HomeDirectory` parameter.
- The `transfer:HomeDirectory` parameter has the leading forward slash (/) removed so that it can be used as part of an S3 Amazon Resource Name (ARN) in a `Resource` statement.

Note

If you are using Logical directories—that is, the user's `homeDirectoryType` is `LOGICAL`—these policy parameters (`HomeBucket`, `HomeDirectory`, and `HomeFolder`) are not supported.

For example, assume that the `HomeDirectory` parameter that is configured for the Transfer Family user is `/home/bob/amazon/stuff/`.

- `transfer:HomeBucket` is set to `/home`.

- `transfer:HomeFolder` is set to `/bob/amazon/stuff/`.
- `transfer:HomeDirectory` becomes `home/bob/amazon/stuff/`.

The first "Sid" allows the user to list all directories starting from `/home/bob/amazon/stuff/`.

The second "Sid" limits the user's put and get access to that same path, `/home/bob/amazon/stuff/`.

With the preceding policy in place, when a user logs in, they can access only objects in their home directory. At connection time, AWS Transfer Family replaces these variables with the appropriate values for the user. Doing this makes it easier to apply the same policy documents to multiple users. This approach reduces the overhead of IAM role and policy management for managing your users' access to your Amazon S3 bucket.

You can also use a session policy to customize access for each of your users based on your business requirements. For more information, see [Permissions for AssumeRole, AssumeRoleWithSAML, and AssumeRoleWithWebIdentity](#) in the *IAM User Guide*.

Note

AWS Transfer Family stores the policy JSON, instead of the Amazon Resource Name (ARN) of the policy. So when you change the policy in the IAM console, you need to return to AWS Transfer Family console and update your users with the latest policy contents. You can update the user under **Policy Info** tab in the **User configuration** section. For more information, see [Managing access controls \(p. 118\)](#).

If you are using the AWS CLI, you can use the following command to update the policy.

```
aws transfer update-user --server-id server --user-name user --policy \  
    "$(aws iam get-policy-version --policy-arn policy --version-id version --output  
    json)"
```

Preventing users from creating a directory in an S3 bucket

You can prevent users from creating a directory in an Amazon S3 bucket. To do so, you create an IAM policy that allows the `s3:PutObject` action but also denies it when the key ends with a "/" (forward slash).

The following example policy allows users to upload files to an Amazon S3 bucket but doesn't allow them to create a directory in the bucket. That is, it denies the `mkdir` command in the Amazon S3 bucket.

```
{  
    "Sid": "DenyMkdir",  
    "Action": [  
        "s3:PutObject"  
    ],  
    "Effect": "Deny",  
    "Resource": "arn:aws:s3:::my-sftp-bucket/*"
```

Monitoring server usage

You can monitor activity in your server using Amazon CloudWatch and AWS CloudTrail. For further analysis, you can also record server activity as readable, near real-time metrics.

Topics

- [Enable AWS CloudTrail logging \(p. 122\)](#)
- [Logging Amazon S3 API calls to S3 access logs \(p. 122\)](#)
- [Log activity with CloudWatch \(p. 122\)](#)
- [Using CloudWatch metrics for Transfer Family \(p. 123\)](#)

Enable AWS CloudTrail logging

You can monitor AWS Transfer Family API calls using AWS CloudTrail. By monitoring API calls, you can get useful security and operational information. For more information about how to work with CloudTrail and AWS Transfer Family, see [Logging and monitoring in AWS Transfer Family \(p. 143\)](#).

If you have [Amazon S3 object level logging enabled](#), RoleSessionName is contained in the Requester field as [AWS:Role Unique Identifier]/username.sessionid@server-id. For more information about AWS Identity and Access Management (IAM) role unique identifiers, see [Unique identifiers](#) in the *AWS Identity and Access Management User Guide*.

Important

The maximum length of the RoleSessionName is 64 characters. If the RoleSessionName is longer, the server-id gets truncated.

Logging Amazon S3 API calls to S3 access logs

If you are [using Amazon S3 access logs to identify S3 requests](#) made on behalf of your file transfer users, RoleSessionName is used to display which IAM role was assumed to service the file transfers. It also displays additional information such as the user name, session id, and server-id used for the transfers. The format is [AWS:Role Unique Identifier]/username.sessionid@server-id and is contained in the Requester field. For example, the following are the contents for a sample Requester field from an S3 access log for a file that was copied to the S3 bucket.

```
arn:aws:sts::AWS-Account-ID:assumed-role/IamRoleName/username.sessionid@server-id
```

In the Requester field above, it shows the IAM Role called IamRoleName. For more information about IAM role unique identifiers, see [Unique identifiers](#) in the *AWS Identity and Access Management User Guide*.

Log activity with CloudWatch

To set access, you create a resource-based IAM policy and an IAM role that provides that access information.

To enable Amazon CloudWatch logging, you start by creating an IAM policy that enables CloudWatch logging. You then create an IAM role and attach the policy to it. You can do this when you are [creating a server \(p. 15\)](#) or by [editing an existing server \(p. 109\)](#). For more information about CloudWatch, see [What Is Amazon CloudWatch?](#) and [What is Amazon CloudWatch Logs?](#) in the *Amazon CloudWatch User Guide*.

To create an IAM policy

- Use the following example policy to create your own IAM policy that allows CloudWatch logging. For information about how to create a policy for AWS Transfer Family, see [Create an IAM role and policy \(p. 8\)](#).

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "VisualEditor0",  
            "Effect": "Allow",  
            "Action": [  
                "logs:CreateLogStream",  
                "logs:DescribeLogStreams",  
                "logs:CreateLogGroup",  
                "logs:PutLogEvents"  
            ],  
            "Resource": "arn:aws:logs:*:log-group:/aws/transfer/*"  
        }  
    ]  
}
```

You then create a role and attach the CloudWatch Logs policy that you created.

To create an IAM role and attach a policy

1. In the navigation pane, choose **Roles**, and then choose **Create role**.
On the **Create role** page, make sure that **AWS service** is chosen.
2. Choose **Transfer** from the service list, and then choose **Next: Permissions**. This establishes a trust relationship between AWS Transfer Family and the IAM role.
3. In the **Attach permissions policies** section, locate and choose the CloudWatch Logs policy that you just created, and choose **Next: Tags**.
4. (Optional) Enter a key and value for a tag, and choose **Next: Review**.
5. On the **Review** page, enter a name and description for your new role, and then choose **Create role**.
6. To view the logs, choose the **Server ID** to open the server configuration page, and choose **View logs**. You are redirected to the CloudWatch console where you can see your log streams.

On the CloudWatch page for your server, you can see records of user authentication (success and failure), data uploads (**PUT** operations), and data downloads (**GET** operations).

Using CloudWatch metrics for Transfer Family

Note

You can also get metrics for Transfer Family from within the Transfer Family console itself. For details, see [Monitoring server usage in the console \(p. 116\)](#).

You can get information about your server using CloudWatch metrics. A *metric* represents a time-ordered set of data points that are published to CloudWatch. When using metrics, you must specify the Transfer Family namespace, metric name, and [dimension \(p. 124\)](#). For more information about metrics, see [Metrics](#) in the *Amazon CloudWatch User Guide*.

The following table describes the CloudWatch metrics for Transfer Family.

Namespace	Metric	Description
AWS/Transfer	BytesIn	The total number of bytes transferred into the server. Units: Count Period: 5 minutes
	BytesOut	The total number of bytes transferred out of the server. Unit: Count Period: 5 minutes
	FilesIn	The total number of files transferred into the server. Units: Count Period: 5 minutes
	FilesOut	The total number of files transferred out of the server. Units: Count Period: 5 minutes
	OnUploadExecution	The total number of workflow executions started on the server. Units: Count Period: 1 minute
	OnUploadExecutionSuccess	The total number of successful workflow executions on the server. Units: Count Period: 1 minute
	OnUploadExecutionFailure	The total number of unsuccessful workflow executions on the server. Units: Count Period: 1 minute

Transfer Family dimensions

A *dimension* is a name/value pair that is part of the identity of a metric. For more information about dimensions, see [Dimensions](#) in the *Amazon CloudWatch User Guide*.

The following table describes the CloudWatch dimension for Transfer Family.

Dimension	Description
ServerId	The unique ID of the server.

Security in AWS Transfer Family

Cloud security at AWS is the highest priority. As an AWS customer, you benefit from a data center and network architecture that is built to meet the requirements of the most security-sensitive organizations.

Security is a shared responsibility between AWS and you. The [shared responsibility model](#) describes this as security of the cloud and security *in* the cloud:

- **Security of the cloud** – AWS is responsible for protecting the infrastructure that runs AWS services in the AWS Cloud. AWS also provides you with services that you can use securely. Third-party auditors regularly test and verify the effectiveness of our security as part of the [AWS compliance programs](#). To learn about the compliance programs that apply to AWS Transfer Family, see [AWS Services in Scope by Compliance Program](#).
- **Security in the cloud** – Your responsibility is determined by the AWS service that you use. You are also responsible for other factors including the sensitivity of your data, your company's requirements, and applicable laws and regulations.

This documentation helps you understand how to apply the shared responsibility model when using AWS Transfer Family. The following topics show you how to configure AWS Transfer Family to meet your security and compliance objectives. You also learn how to use other AWS services that help you to monitor and secure your AWS Transfer Family resources.

Topics

- [Data protection in AWS Transfer Family \(p. 125\)](#)
- [Identity and access management for AWS Transfer Family \(p. 130\)](#)
- [Logging and monitoring in AWS Transfer Family \(p. 143\)](#)
- [Compliance validation for AWS Transfer Family \(p. 145\)](#)
- [Resilience in AWS Transfer Family \(p. 145\)](#)
- [Infrastructure security in AWS Transfer Family \(p. 145\)](#)
- [Add a web application firewall \(p. 146\)](#)
- [AWS managed policies for AWS Transfer Family \(p. 147\)](#)

Data protection in AWS Transfer Family

The AWS [shared responsibility model](#) applies to data protection in AWS Transfer Family (Transfer Family). As described in this model, AWS is responsible for protecting the global infrastructure that runs all of the AWS Cloud. You are responsible for maintaining control over your content that is hosted on this infrastructure. This content includes the security configuration and management tasks for the AWS services that you use. For more information about data privacy, see the [Data Privacy FAQ](#). For information about data protection in Europe, see the [AWS Shared Responsibility Model and GDPR](#) blog post on the [AWS Security Blog](#).

For data protection purposes, we recommend that you protect AWS account credentials and set up individual user accounts with AWS Identity and Access Management (IAM). That way each user is given only the permissions necessary to fulfill their job duties. We also recommend that you secure your data in the following ways:

- Use multi-factor authentication (MFA) with each account.
- Use SSL/TLS to communicate with AWS resources. We support TLS 1.2.

- Set up API and user activity logging with AWS CloudTrail.
- Use AWS encryption solutions, along with all default security controls within AWS services.
- Use advanced managed security services such as Amazon Macie, which assists in discovering and securing personal data that is stored in Amazon S3.
- If you require FIPS 140-2 validated cryptographic modules when accessing AWS through a command line interface or an API, use a FIPS endpoint. For more information about the available FIPS endpoints, see [Federal Information Processing Standard \(FIPS\) 140-2](#).

We strongly recommend that you never put sensitive identifying information, such as your customers' account numbers, into free-form fields such as a **Name** field. This includes when you work with Transfer Family or other AWS services using the console, API, AWS CLI, or AWS SDKs. Any data that you enter into Transfer Family or other services might get picked up for inclusion in diagnostic logs. When you provide a URL to an external server, don't include credentials information in the URL to validate your request to that server.

Data encryption

AWS Transfer Family uses the default encryption options you set for your Amazon S3 bucket to encrypt your data. When you enable encryption on a bucket, all objects are encrypted when they are stored in the bucket. The objects are encrypted using server-side encryption with either Amazon S3-managed keys (SSE-S3) or AWS KMS-managed keys (SSE-KMS). For information about server-side encryption, see [Protecting data using server-side encryption](#) in the *Amazon Simple Storage Service User Guide*.

The following steps show you how to encrypt data in AWS Transfer Family.

To allow encryption in AWS Transfer Family

1. Enable default encryption for your Amazon S3 bucket. For instructions, see [Amazon S3 default encryption for S3 buckets](#) in the *Amazon Simple Storage Service User Guide*.
2. Update the AWS Identity and Access Management (IAM) role policy that is attached to the user to grant the required AWS Key Management Service (AWS KMS) permissions.
3. If you are using session policy for the user, the session policy must grant the required AWS KMS permissions.

The following example shows an IAM policy that grants the minimum permissions required when using AWS Transfer Family with an Amazon S3 bucket that is enabled for AWS KMS encryption. Include this example policy in both the user IAM role policy and session policy, if you are using one.

```
{  
  "Sid": "Stmt1544140969635",  
  "Action": [  
    "kms:Decrypt",  
    "kms:Encrypt",  
    "kms:GenerateDataKey"  
  ],  
  "Effect": "Allow",  
  "Resource": "arn:aws:kms:region:account-id:key/kms-key-id"  
}
```

Note

The KMS key ID that you specify in this policy must be the same as the one specified for the default encryption in step 1.

Root, or the IAM role that is used for the user, must be allowed in the AWS KMS key policy. For information about the AWS KMS key policy, see [Using key policies in AWS KMS](#) in the *AWS Key Management Service Developer Guide*.

Key management

In this section, you can find information about SSH keys, how to generate them, and how to rotate them.

Note

Currently, Transfer Family does not accept elliptical curve keys (keys beginning with ecdsa).

Topics

- [Generate SSH keys \(p. 127\)](#)
- [Rotate SSH keys \(p. 128\)](#)

Generate SSH keys

You can set up your server to authenticate users using the service managed authentication method, where user names and SSH keys are stored within the service. The user's public SSH key is uploaded to the server as a user's property. This key is used by the server as part of a standard key-based authentication process. Each user can have multiple public SSH keys on file with an individual server. For limits on number of keys that can be stored per user, see the [AWS service quotas](#) in the *AWS General Reference*.

As an alternative to the service managed authentication method, you can authenticate users using a custom identity provider. This allows you to plug in an existing identity provider using an Amazon API Gateway endpoint. For more information, see [Authenticating using an API Gateway method \(p. 74\)](#).

A server can only authenticate users using one method (service managed or custom identity provider), and that method cannot be changed after the server is created.

Topics

- [Creating SSH Keys on macOS, Linux, or UNIX \(p. 127\)](#)
- [Creating SSH Keys on Microsoft Windows \(p. 128\)](#)

Creating SSH Keys on macOS, Linux, or UNIX

On the macOS, Linux, or UNIX operating systems, you use the `ssh-keygen` command to create an SSH public key and SSH private key also known as a key pair.

To create SSH keys on a macOS, Linux, or UNIX operating system

1. On macOS, Linux, or UNIX operating systems, open a command terminal.
2. At the prompt, enter the following command: `ssh-keygen -P "" -m PEM -f key_name`.

Note

key_name is the SSH key pair file name.

The following shows an example of the `ssh-keygen` output.

```
% ssh-keygen -P "" -f my_key_pair
Generating public/private rsa key pair.
Your identification has been saved in my_key_pair.
Your public key has been saved in my_key_pair.pub.
The key fingerprint is:
SHA256:8tDDwPmanTFcEzjTwPGETVWOGW1nVz+gtCCE8hL7PrQ
jeff@amazon.com
The key's randomart image is:
+---[RSA 2048]---+
|       o+oX+oo+. o|
|   o .Bo=.o+=|=|
| = + otoo..+o| |
| o * . : . . . |
| o o S | |
| o B = | |
| o + + | |
| E | |
| . | |
+---[SHA256]---+
```

3. Navigate to the `key_name`.pub file and open it.
4. Copy the text and paste it in **SSH public key**.

Note

When you run the `ssh-keygen` command as shown preceding, it creates the public and private keys as files in the current directory.

Creating SSH Keys on Microsoft Windows

Windows uses a slightly different SSH key pair format. The public key must be in the `PUB` format, and the private key must be in the `PPK` format. On Windows, you can use PuTTYgen to create an SSH key pair in the appropriate formats. You can also use PuTTYgen to convert a private key generated using `ssh-keygen` to a `.ppk` file.

Note

If you present WinSCP with a private key file not in `.ppk` format, that client offers to convert the key into `.ppk` format for you.

For a tutorial on creating SSH keys using PuTTYgen on Windows, see the [SSH.com website](#).

Rotate SSH keys

For security, we recommend the best practice of rotating your SSH keys. Usually, this rotation is specified as a part of a security policy and is implemented in some automated fashion. Depending upon the level of security, for a highly sensitive communication, an SSH key pair might be used only once. Doing this eliminates any risk due to stored keys. However, it is much more common to store SSH credentials for a period of time and set an interval that doesn't place undue burden on users. A time interval of three months is common.

There are two methods used to perform SSH key rotation:

- On the console, you can upload a new SSH public key and delete an existing SSH public key.
- Using the API, you can update existing users by using the [DeleteSshPublicKey](#) API to delete a user's Secure Shell (SSH) public key and the [ImportSshPublicKey](#) API to add a new Secure Shell (SSH) public key to the user's account.

Console

To perform a key rotation in the console

1. Open the AWS Transfer Family console at <https://console.aws.amazon.com/transfer/>.
2. Navigate to the **Servers** page.
3. Choose the identifier in the **Server ID** column to see the **Server details** page.
4. Under **Users**, select the check box of the user whose SSH public key that you want to rotate, then choose **Actions**, and then choose **Add key** to see the **Add key** page.

or

Choose the user name to see the **User details** page, and then choose **Add SSH public key** to see the **Add key** page.

5. Enter the new SSH public key and choose **Add key**.

Important

The format of the SSH public key is `ssh-rsa <string>`.

You are returned to the **User details** page, and the new SSH public key that you just entered appears in the **SSH public keys** section.

6. Select the check box of the old key that you want to delete and then choose **Delete**.
7. Confirm the deletion operation by entering the word `delete`, and then choose **Delete**.

API

To perform a key rotation using the API

1. On macOS, Linux, or UNIX operating systems, open a command terminal.
2. Retrieve the SSH key that you want to delete by entering the following command:

```
aws transfer describe-user --server-id='serverID' --user-name='username'
```

where `serverID` is the Server ID for your Transfer Family server, and `username` is your user name.

The command returns details about the user. Copy the contents of the "SshPublicKeyId": field: you need to enter this value later in this procedure.

```
"SshPublicKeys": [ { "SshPublicKeyBody": "public-key", "SshPublicKeyId": "keyID", "DateImported": 1621969331.072 } ],
```

3. Next, import a new SSH key for your user. At the prompt, enter the following command:

```
aws transfer import-ssh-public-key --server-id='serverID' --user-name='username' --ssh-public-key-body='publickey'
```

Where the `publickey` is the fingerprint of your new public key.

4. Finally, delete the old key by running the following command:

```
aws transfer delete-ssh-public-key --server-id='serverID' --user-name='username' --ssh-public-key-body='publickey-from-step-2'
```

where `publickey-from-step-2` is the key value you copied in step 2 of this procedure.

Identity and access management for AWS Transfer Family

AWS Identity and Access Management (IAM) is an AWS service that helps an administrator securely control access to AWS resources. IAM administrators control who can be *authenticated* (signed in) and *authorized* (have permissions) to use AWS Transfer Family resources. IAM is an AWS service that you can use with no additional charge.

Topics

- [Audience \(p. 130\)](#)
- [Authenticating with identities \(p. 130\)](#)
- [Managing access using policies \(p. 132\)](#)
- [How AWS Transfer Family works with IAM \(p. 134\)](#)
- [AWS Transfer Family identity-based policy examples \(p. 137\)](#)
- [AWS Transfer Family tag-based policy examples \(p. 139\)](#)
- [Troubleshooting AWS Transfer Family identity and access \(p. 141\)](#)

Audience

How you use AWS Identity and Access Management (IAM) differs, depending on the work that you do in AWS Transfer Family.

Service user – If you use the AWS Transfer Family service to do your job, then your administrator provides you with the credentials and permissions that you need. As you use more AWS Transfer Family features to do your work, you might need additional permissions. Understanding how access is managed can help you request the right permissions from your administrator. If you cannot access a feature in AWS Transfer Family, see [Troubleshooting AWS Transfer Family identity and access \(p. 141\)](#).

Service administrator – If you're in charge of AWS Transfer Family resources at your company, you probably have full access to AWS Transfer Family. It's your job to determine which AWS Transfer Family features and resources your employees should access. You must then submit requests to your IAM administrator to change the permissions of your service users. Review the information on this page to understand the basic concepts of IAM. To learn more about how your company can use IAM with AWS Transfer Family, see [How AWS Transfer Family works with IAM \(p. 134\)](#).

IAM administrator – If you're an IAM administrator, you might want to learn details about how you can write policies to manage access to AWS Transfer Family. To view example AWS Transfer Family identity-based policies that you can use in IAM, see [AWS Transfer Family identity-based policy examples \(p. 137\)](#).

Authenticating with identities

Authentication is how you sign in to AWS using your identity credentials. For more information about signing in using the AWS Management Console, see [Signing in to the AWS Management Console as an IAM user or root user](#) in the *IAM User Guide*.

You must be *authenticated* (signed in to AWS) as the AWS account root user, an IAM user, or by assuming an IAM role. You can also use your company's single sign-on authentication or even sign in using Google or Facebook. In these cases, your administrator previously set up identity federation using IAM roles. When you access AWS using credentials from another company, you are assuming a role indirectly.

To sign in directly to the [AWS Management Console](#), use your password with your root user email address or your IAM user name. You can access AWS programmatically using your root user or IAM

users access keys. AWS provides SDK and command line tools to cryptographically sign your request using your credentials. If you don't use AWS tools, you must sign the request yourself. Do this using *Signature Version 4*, a protocol for authenticating inbound API requests. For more information about authenticating requests, see [Signature Version 4 signing process](#) in the *AWS General Reference*.

Regardless of the authentication method that you use, you might also be required to provide additional security information. For example, AWS recommends that you use multi-factor authentication (MFA) to increase the security of your account. To learn more, see [Using multi-factor authentication \(MFA\) in AWS](#) in the *IAM User Guide*.

AWS account root user

When you first create an AWS account, you begin with a single sign-in identity that has complete access to all AWS services and resources in the account. This identity is called the *AWS account root user* and is accessed by signing in with the email address and password that you used to create the account. We strongly recommend that you do not use the root user for your everyday tasks, even the administrative ones. Instead, adhere to the [best practice of using the root user only to create your first IAM user](#). Then securely lock away the root user credentials and use them to perform only a few account and service management tasks.

IAM users and groups

An *IAM user* is an identity within your AWS account that has specific permissions for a single person or application. An IAM user can have long-term credentials such as a user name and password or a set of access keys. To learn how to generate access keys, see [Managing access keys for IAM users](#) in the *IAM User Guide*. When you generate access keys for an IAM user, make sure you view and securely save the key pair. You cannot recover the secret access key in the future. Instead, you must generate a new access key pair.

An *IAM group* is an identity that specifies a collection of IAM users. You can't sign in as a group. You can use groups to specify permissions for multiple users at a time. Groups make permissions easier to manage for large sets of users. For example, you could have a group named *IAMAdmins* and give that group permissions to administer IAM resources.

Users are different from roles. A user is uniquely associated with one person or application, but a role is intended to be assumable by anyone who needs it. Users have permanent long-term credentials, but roles provide temporary credentials. To learn more, see [When to create an IAM user \(instead of a role\)](#) in the *IAM User Guide*.

IAM roles

An *IAM role* is an identity within your AWS account that has specific permissions. It is similar to an IAM user, but is not associated with a specific person. You can temporarily assume an IAM role in the AWS Management Console by [switching roles](#). You can assume a role by calling an AWS CLI or AWS API operation or by using a custom URL. For more information about methods for using roles, see [Using IAM roles](#) in the *IAM User Guide*.

IAM roles with temporary credentials are useful in the following situations:

- **Temporary IAM user permissions** – An IAM user can assume an IAM role to temporarily take on different permissions for a specific task.
- **Federated user access** – Instead of creating an IAM user, you can use existing identities from AWS Directory Service, your enterprise user directory, or a web identity provider. These are known as *federated users*. AWS assigns a role to a federated user when access is requested through an [identity provider](#). For more information about federated users, see [Federated users and roles](#) in the *IAM User Guide*.

- **Cross-account access** – You can use an IAM role to allow someone (a trusted principal) in a different account to access resources in your account. Roles are the primary way to grant cross-account access. However, with some AWS services, you can attach a policy directly to a resource (instead of using a role as a proxy). To learn the difference between roles and resource-based policies for cross-account access, see [How IAM roles differ from resource-based policies](#) in the *IAM User Guide*.
- **Cross-service access** – Some AWS services use features in other AWS services. For example, when you make a call in a service, it's common for that service to run applications in Amazon EC2 or store objects in Amazon S3. A service might do this using the calling principal's permissions, using a service role, or using a service-linked role.
 - **Principal permissions** – When you use an IAM user or role to perform actions in AWS, you are considered a principal. Policies grant permissions to a principal. When you use some services, you might perform an action that then triggers another action in a different service. In this case, you must have permissions to perform both actions. To see whether an action requires additional dependent actions in a policy, see [Actions, resources, and condition keys for AWS Transfer Family](#) in the *Service Authorization Reference*.
 - **Service role** – A service role is an [IAM role](#) that a service assumes to perform actions on your behalf. An IAM administrator can create, modify, and delete a service role from within IAM. For more information, see [Creating a role to delegate permissions to an AWS service](#) in the *IAM User Guide*.
 - **Service-linked role** – A service-linked role is a type of service role that is linked to an AWS service. The service can assume the role to perform an action on your behalf. Service-linked roles appear in your IAM account and are owned by the service. An IAM administrator can view, but not edit the permissions for service-linked roles.
- **Applications running on Amazon EC2** – You can use an IAM role to manage temporary credentials for applications that are running on an EC2 instance and making AWS CLI or AWS API requests. This is preferable to storing access keys within the EC2 instance. To assign an AWS role to an EC2 instance and make it available to all of its applications, you create an instance profile that is attached to the instance. An instance profile contains the role and enables programs that are running on the EC2 instance to get temporary credentials. For more information, see [Using an IAM role to grant permissions to applications running on Amazon EC2 instances](#) in the *IAM User Guide*.

To learn whether to use IAM roles or IAM users, see [When to create an IAM role \(instead of a user\)](#) in the *IAM User Guide*.

Managing access using policies

You control access in AWS by creating policies and attaching them to IAM identities or AWS resources. A policy is an object in AWS that, when associated with an identity or resource, defines their permissions. You can sign in as the root user or an IAM user, or you can assume an IAM role. When you then make a request, AWS evaluates the related identity-based or resource-based policies. Permissions in the policies determine whether the request is allowed or denied. Most policies are stored in AWS as JSON documents. For more information about the structure and contents of JSON policy documents, see [Overview of JSON policies](#) in the *IAM User Guide*.

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

Every IAM entity (user or role) starts with no permissions. In other words, by default, users can do nothing, not even change their own password. To give a user permission to do something, an administrator must attach a permissions policy to a user. Or the administrator can add the user to a group that has the intended permissions. When an administrator gives permissions to a group, all users in that group are granted those permissions.

IAM policies define permissions for an action regardless of the method that you use to perform the operation. For example, suppose that you have a policy that allows the `iam:GetRole` action. A user with that policy can get role information from the AWS Management Console, the AWS CLI, or the AWS API.

Identity-based policies

Identity-based policies are JSON permissions policy documents that you can attach to an identity, such as an IAM user, group of users, or role. These policies control what actions users and roles can perform, on which resources, and under what conditions. To learn how to create an identity-based policy, see [Creating IAM policies](#) in the *IAM User Guide*.

Identity-based policies can be further categorized as *inline policies* or *managed policies*. Inline policies are embedded directly into a single user, group, or role. Managed policies are standalone policies that you can attach to multiple users, groups, and roles in your AWS account. Managed policies include AWS managed policies and customer managed policies. To learn how to choose between a managed policy or an inline policy, see [Choosing between managed policies and inline policies](#) in the *IAM User Guide*.

Resource-based policies

Resource-based policies are JSON policy documents that you attach to a resource. Examples of resource-based policies are IAM *role trust policies* and Amazon S3 *bucket policies*. In services that support resource-based policies, service administrators can use them to control access to a specific resource. For the resource where the policy is attached, the policy defines what actions a specified principal can perform on that resource and under what conditions. You must [specify a principal](#) in a resource-based policy. Principals can include accounts, users, roles, federated users, or AWS services.

Resource-based policies are inline policies that are located in that service. You can't use AWS managed policies from IAM in a resource-based policy.

Access control lists (ACLs)

Access control lists (ACLs) control which principals (account members, users, or roles) have permissions to access a resource. ACLs are similar to resource-based policies, although they do not use the JSON policy document format.

Amazon S3, AWS WAF, and Amazon VPC are examples of services that support ACLs. To learn more about ACLs, see [Access control list \(ACL\) overview](#) in the *Amazon Simple Storage Service Developer Guide*.

Other policy types

AWS supports additional, less-common policy types. These policy types can set the maximum permissions granted to you by the more common policy types.

- **Permissions boundaries** – A permissions boundary is an advanced feature in which you set the maximum permissions that an identity-based policy can grant to an IAM entity (IAM user or role). You can set a permissions boundary for an entity. The resulting permissions are the intersection of entity's identity-based policies and its permissions boundaries. Resource-based policies that specify the user or role in the `Principal` field are not limited by the permissions boundary. An explicit deny in any of these policies overrides the allow. For more information about permissions boundaries, see [Permissions boundaries for IAM entities](#) in the *IAM User Guide*.
- **Service control policies (SCPs)** – SCPs are JSON policies that specify the maximum permissions for an organization or organizational unit (OU) in AWS Organizations. AWS Organizations is a service for grouping and centrally managing multiple AWS accounts that your business owns. If you enable all features in an organization, then you can apply service control policies (SCPs) to any or all of your accounts. The SCP limits permissions for entities in member accounts, including each AWS account root user. For more information about Organizations and SCPs, see [How SCPs work](#) in the *AWS Organizations User Guide*.
- **Session policies** – Session policies are advanced policies that you pass as a parameter when you programmatically create a temporary session for a role or federated user. The resulting session's permissions are the intersection of the user or role's identity-based policies and the session policies.

Permissions can also come from a resource-based policy. An explicit deny in any of these policies overrides the allow. For more information, see [Session policies](#) in the *IAM User Guide*.

Multiple policy types

When multiple types of policies apply to a request, the resulting permissions are more complicated to understand. To learn how AWS determines whether to allow a request when multiple policy types are involved, see [Policy evaluation logic](#) in the *IAM User Guide*.

How AWS Transfer Family works with IAM

Before you use AWS Identity and Access Management (IAM) to manage access to AWS Transfer Family, you should understand what IAM features are available to use with AWS Transfer Family. To get a high-level view of how AWS Transfer Family and other AWS services work with IAM, see [AWS services that work with IAM](#) in the *IAM User Guide*.

Topics

- [AWS Transfer Family identity-based policies \(p. 134\)](#)
- [AWS Transfer Family resource-based policies \(p. 136\)](#)
- [Authorization based on AWS Transfer Family tags \(p. 137\)](#)
- [AWS Transfer Family IAM roles \(p. 137\)](#)

AWS Transfer Family identity-based policies

With IAM identity-based policies, you can specify allowed or denied actions and resources as well as the conditions under which actions are allowed or denied. AWS Transfer Family supports specific actions, resources, and condition keys. To learn about all of the elements that you use in a JSON policy, see [IAM JSON policy elements reference](#) in the *AWS Identity and Access Management User Guide*.

Actions

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The **Action** element of a JSON policy describes the actions that you can use to allow or deny access in a policy. Policy actions usually have the same name as the associated AWS API operation. There are some exceptions, such as *permission-only actions* that don't have a matching API operation. There are also some operations that require multiple actions in a policy. These additional actions are called *dependent actions*.

Include actions in a policy to grant permissions to perform the associated operation.

Policy actions in AWS Transfer Family use the following prefix before the action: `transfer:`. For example, to grant someone permission to create a server, with the Transfer Family `CreateServer` API operation, you include the `transfer:CreateServer` action in their policy. Policy statements must include either an **Action** or **NotAction** element. AWS Transfer Family defines its own set of actions that describe tasks that you can perform with this service.

To specify multiple actions in a single statement, separate them with commas as follows.

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

You can specify multiple actions using wildcards (*). For example, to specify all actions that begin with the word `Describe`, include the following action.

```
"Action": "transfer:Describe*"
```

To see a list of AWS Transfer Family actions, see [Actions defined by AWS Transfer Family](#) in the *IAM User Guide*.

Resources

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The `Resource` JSON policy element specifies the object or objects to which the action applies. Statements must include either a `Resource` or a `NotResource` element. As a best practice, specify a resource using its [Amazon Resource Name \(ARN\)](#). You can do this for actions that support a specific resource type, known as *resource-level permissions*.

For actions that don't support resource-level permissions, such as listing operations, use a wildcard (*) to indicate that the statement applies to all resources.

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

The Transfer Family server resource has the following ARN.

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

For example, to specify the s-01234567890abcdef Transfer Family server in your statement, use the following ARN.

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

For more information about the format of ARNs, see [Amazon Resource Names \(ARNs\) and AWS service namespaces](#).

To specify all instances that belong to a specific account, use the wildcard (*).

```
"Resource": "arn:aws:transfer:us-east-1:123456789012:server/*"
```

Some AWS Transfer Family actions are performed on multiple resources, such as those used in IAM policies. In those cases, you must use the wildcard (*).

```
"Resource": "arn:aws:transfer:*:123456789012:server/*"
```

In some cases you need to specify more than one type of resource, for example, if you create a policy that allows access to Transfer Family servers and users. To specify multiple resources in a single statement, separate the ARNs with commas.

```
"Resource": [  
    "resource1",  
    "resource2"]
```

]

To see a list of AWS Transfer Family resource types and their ARNs, see [Resources defined by AWS Transfer Family](#) in the *AWS Identity and Access Management User Guide*. To learn with which actions you can specify the ARN of each resource, see [Actions defined by AWS Transfer Family](#).

Condition keys

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The **Condition** element (or **Condition block**) lets you specify conditions in which a statement is in effect. The **Condition** element is optional. You can create conditional expressions that use [condition operators](#), such as equals or less than, to match the condition in the policy with values in the request.

If you specify multiple **Condition** elements in a statement, or multiple keys in a single **Condition** element, AWS evaluates them using a logical AND operation. If you specify multiple values for a single condition key, AWS evaluates the condition using a logical OR operation. All of the conditions must be met before the statement's permissions are granted.

You can also use placeholder variables when you specify conditions. For example, you can grant an IAM user permission to access a resource only if it is tagged with their IAM user name. For more information, see [IAM policy elements: variables and tags](#) in the *IAM User Guide*.

AWS supports global condition keys and service-specific condition keys. To see all AWS global condition keys, see [AWS global condition context keys](#) in the *IAM User Guide*.

AWS Transfer Family defines its own set of condition keys and also supports using some global condition keys. To see all AWS global condition keys, see [AWS Global condition context keys](#) in the *AWS Identity and Access Management User Guide*.

To see a list of AWS Transfer Family condition keys, see [Condition keys for AWS Transfer Family](#) in the *AWS Identity and Access Management User Guide*. To learn with which actions and resources you can use a condition key, see [Actions defined by AWS Transfer Family](#).

Examples

To view examples of AWS Transfer Family identity-based policies, see [AWS Transfer Family identity-based policy examples \(p. 137\)](#).

AWS Transfer Family resource-based policies

Resource-based policies are JSON policy documents that specify what actions a specified principal can perform on the AWS Transfer Family resource and under what conditions. Amazon S3 supports resource-based permissions policies for Amazon S3 [buckets](#). Resource-based policies let you grant usage permission to other accounts on a per-resource basis. You can also use a resource-based policy to allow an AWS service to access your Amazon S3 [buckets](#).

To enable cross-account access, you can specify an entire account or IAM entities in another account as the [principal in a resource-based policy](#). Adding a cross-account principal to a resource-based policy is only half of establishing the trust relationship. When the principal and the resource are in different AWS accounts, you must also grant the principal entity permission to access the resource. Grant permission by attaching an identity-based policy to the entity. However, if a resource-based policy grants access to a principal in the same account, no additional identity-based policy is required. For more information, see [How IAM roles differ from resource-based policies](#) in the *AWS Identity and Access Management User Guide*.

The Amazon S3 service supports only one type of resource-based policy called a **bucket policy**, which is attached to a **bucket**. This policy defines which principal entities (accounts, users, roles, and federated users) can perform actions on the object.

Examples

To view examples of AWS Transfer Family resource-based policies, see [AWS Transfer Family tag-based policy examples \(p. 139\)](#).

Authorization based on AWS Transfer Family tags

You can attach tags to AWS Transfer Family resources or pass tags in a request to AWS Transfer Family. To control access based on tags, you provide tag information in the `condition element` of a policy using the `transfer:ResourceTag/key-name`, `aws:RequestTag/key-name`, or `aws:TagKeys` condition keys. For information about how to use tags to control access to AWS Transfer Family resources, see [AWS Transfer Family tag-based policy examples \(p. 139\)](#).

AWS Transfer Family IAM roles

An **IAM role** is an entity within your AWS account that has specific permissions.

Using temporary credentials with AWS Transfer Family

You can use temporary credentials to sign in with federation, assume an IAM role, or to assume a cross-account role. You obtain temporary security credentials by calling AWS STS API operations such as `AssumeRole` or `GetFederationToken`.

AWS Transfer Family supports using temporary credentials.

AWS Transfer Family identity-based policy examples

By default, IAM users and roles don't have permission to create or modify AWS Transfer Family resources. They also can't perform tasks using the AWS Management Console, AWS CLI, or AWS API. An IAM administrator must create IAM policies that grant users and roles permission to perform specific API operations on the specified resources they need. The administrator must then attach those policies to the IAM users or groups that require those permissions.

To learn how to create an IAM identity-based policy using these example JSON policy documents, see [Creating policies on the JSON tab](#) in the *AWS Identity and Access Management User Guide*.

Topics

- [Policy best practices \(p. 137\)](#)
- [Using the AWS Transfer Family console \(p. 138\)](#)
- [Allow users to view their own permissions \(p. 138\)](#)

Policy best practices

Identity-based policies are very powerful. They determine whether someone can create, access, or delete AWS Transfer Family resources in your account. These actions can incur costs for your AWS account. When you create or edit identity-based policies, follow these guidelines and recommendations:

- **Get started using AWS managed policies** – To start using AWS Transfer Family quickly, use AWS managed policies to give your employees the permissions they need. These policies are already available in your account and are maintained and updated by AWS. For more information, see [Get started using permissions with AWS managed policies](#) in the *IAM User Guide*.

- **Grant least privilege** – When you create custom policies, grant only the permissions required to perform a task. Start with a minimum set of permissions and grant additional permissions as necessary. Doing so is more secure than starting with permissions that are too lenient and then trying to tighten them later. For more information, see [Grant least privilege](#) in the *IAM User Guide*.
- **Enable MFA for sensitive operations** – For extra security, require IAM users to use multi-factor authentication (MFA) to access sensitive resources or API operations. For more information, see [Using multi-factor authentication \(MFA\) in AWS](#) in the *IAM User Guide*.
- **Use policy conditions for extra security** – To the extent that it's practical, define the conditions under which your identity-based policies allow access to a resource. For example, you can write conditions to specify a range of allowable IP addresses that a request must come from. You can also write conditions to allow requests only within a specified date or time range, or to require the use of SSL or MFA. For more information, see [IAM JSON policy elements: Condition](#) in the *IAM User Guide*.

Using the AWS Transfer Family console

To access the AWS Transfer Family console, you must have a minimum set of permissions. These permissions must allow you to list and view details about the AWS Transfer Family resources in your AWS account. If you create an identity-based policy that is more restrictive than the minimum required permissions, the console won't function as intended for entities (IAM users or roles) with that policy. For more information, see [Adding permissions to a user](#) in the *AWS Identity and Access Management User Guide*.

You don't need to allow minimum console permissions for users that are making calls only to the AWS CLI or the AWS API. Instead, allow access to only the actions that match the API operation that you're trying to perform.

Allow users to view their own permissions

This example shows how you might create a policy that allows IAM users to view the inline and managed policies that are attached to their user identity. This policy includes permissions to complete this action on the console or programmatically using the AWS CLI or AWS API.

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

```
        ],
        "Resource": "*"
    }
}
```

AWS Transfer Family tag-based policy examples

The following are examples of how to control access to AWS Transfer Family resources based on tags.

Using tags to control access to AWS Transfer Family resources

Conditions in IAM policies are part of the syntax that you use to specify permissions to AWS Transfer Family resources. You can control access to AWS Transfer Family resources (such as users, servers, roles, and other entities) based on tags on those resources. Tags are key-value pairs. For more information about tagging resources, see [Tagging AWS resources](#) in the *AWS General Reference*.

In AWS Transfer Family, resources can have tags, and some actions can include tags. When you create an IAM policy, you can use tag condition keys to control the following:

- Which users can perform actions on an AWS Transfer Family resource, based on tags that the resource has.
- What tags can be passed in an action's request.
- Whether specific tag keys can be used in a request.

By using tag-based access control, you can apply finer control than at the API level. You also can apply more dynamic control than by using resource-based access control. You can create IAM policies that allow or deny an operation based on tags provided in the request (request tags). You can also create IAM policies based on tags on the resource that is being operated on (resource tags). In general, resource tags are for tags that are already on resources, request tags are for when you're adding tags to or removing tags from a resource.

For the complete syntax and semantics of tag condition keys, see [Controlling access to AWS resources using resource tags](#) in the *IAM User Guide*. For details about specifying IAM policies with API Gateway, see [Control access to an API with IAM permissions](#) in the *API Gateway Developer Guide*.

Example 1: Deny actions based on resource tags

You can deny an action to be performed on a resource based on tags. The following example policy denies `TagResource`, `UntagResource`, `StartServer`, `StopServer`, `DescribeServer`, and `DescribeUser` operations if the user or server resource is tagged with the key `stage` and the value `prod`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": [
                "transfer:TagResource",
                "transfer:UntagResource",
                "transfer:StartServer",
                "transfer:StopServer",
                "transfer:DescribeServer",
                "transfer:DescribeUser"
            ],
            "Resource": "*",
            "Condition": {
                "StringLike": {
                    "aws:RequestTag/stage": "prod"
                }
            }
        }
    ]
}
```

```

        "Condition": {
            "StringEquals": {
                "aws:ResourceTag/stage": "prod"
            }
        }
    ]
}

```

Example 2: Allow actions based on resource tags

You can allow an action to be performed on a resource based on tags. The following example policy allows TagResource, UntagResource, StartServer, StopServer, DescribeServer, and DescribeUser operations if the user or server resource is tagged with the key stage and the value prod.

```

{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "transfer:TagResource",
                "transfer:UntagResource",
                "transfer:StartServer",
                "transfer:StopServer",
                "transfer:DescribeServer",
                "transfer:DescribeUser"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/stage": "prod"
                }
            }
        }
    ]
}

```

Example 3: Deny creation of a user or server based on request tags

The following example policy contains two statements. The first statement denies the CreateServer operation on all resources if the cost center key for the tag doesn't have a value.

The second statement denies the CreateServer operation if the cost center key for the tag contains any other value besides 1, 2 or 3.

Note

This policy does allow creating or deleting a resource that contains a key called costcenter and a value of 1, 2, or 3.

```

{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": [
                "transfer>CreateServer"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}

```

```
        ],
        "Condition": {
            "Null": {
                "aws:RequestTag/costcenter": "true"
            }
        }
    },
    {
        "Effect": "Deny",
        "Action": "transfer>CreateServer",
        "Resource": [
            "*"
        ],
        "Condition": {
            "ForAnyValue:StringNotEquals": {
                "aws:RequestTag/costcenter": [
                    "1",
                    "2",
                    "3"
                ]
            }
        }
    }
]
```

Troubleshooting AWS Transfer Family identity and access

Use the following information to help you diagnose and fix common issues that you might encounter when working with AWS Transfer Family and IAM.

Topics

- [I am not authorized to perform an action in AWS Transfer Family \(p. 141\)](#)
- [I am not authorized to perform iam:PassRole \(p. 142\)](#)
- [I want to view my access keys \(p. 142\)](#)
- [I'm an administrator and want to allow others to access AWS Transfer Family \(p. 142\)](#)
- [I want to allow people outside of my AWS account to access my AWS Transfer Family resources \(p. 142\)](#)

I am not authorized to perform an action in AWS Transfer Family

If the AWS Management Console tells you that you're not authorized to perform an action, then you must contact your administrator for assistance. Your administrator is the person that provided you with your user name and password.

The following example error occurs when the `mateojackson` IAM user tries to use the console to view details about a `widget` but does not have `transfer:GetWidget` permissions.

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

In this case, Mateo asks his administrator to update his policies to allow him to access the `my-example-widget` resource using the `transfer;:GetWidget` action.

I am not authorized to perform iam:PassRole

If you receive an error that you're not authorized to perform the `iam:PassRole` action, then you must contact your administrator for assistance. Your administrator is the person that provided you with your user name and password. Ask that person to update your policies to allow you to pass a role to AWS Transfer Family.

Some AWS services allow you to pass an existing role to that service, instead of creating a new service role or service-linked role. To do this, you must have permissions to pass the role to the service.

The following example error occurs when an IAM user named `marymajor` tries to use the console to perform an action in AWS Transfer Family. However, the action requires the service to have permissions granted by a service role. Mary does not have permissions to pass the role to the service.

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

In this case, Mary asks her administrator to update her policies to allow her to perform the `iam:PassRole` action.

I want to view my access keys

After you create your IAM user access keys, you can view your access key ID at any time. However, you can't view your secret access key again. If you lose your secret key, you must create a new access key pair.

Access keys consist of two parts: an access key ID (for example, `AKIAIOSFODNN7EXAMPLE`) and a secret access key (for example, `wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY`). Like a user name and password, you must use both the access key ID and secret access key together to authenticate your requests. Manage your access keys as securely as you do your user name and password.

Important

Do not provide your access keys to a third party, even to help [find your canonical user ID](#). By doing this, you might give someone permanent access to your account.

When you create an access key pair, you are prompted to save the access key ID and secret access key in a secure location. The secret access key is available only at the time you create it. If you lose your secret access key, you must add new access keys to your IAM user. You can have a maximum of two access keys. If you already have two, you must delete one key pair before creating a new one. To view instructions, see [Managing access keys](#) in the *IAM User Guide*.

I'm an administrator and want to allow others to access AWS Transfer Family

To allow others to access AWS Transfer Family, you must create an IAM entity (user or role) for the person or application that needs access. They will use the credentials for that entity to access AWS. You must then attach a policy to the entity that grants them the correct permissions in AWS Transfer Family.

To get started right away, see [Creating your first IAM delegated user and group](#) in the *IAM User Guide*.

I want to allow people outside of my AWS account to access my AWS Transfer Family resources

You can create a role that users in other accounts or people outside of your organization can use to access your resources. You can specify who is trusted to assume the role. For services that support resource-based policies or access control lists (ACLs), you can use those policies to grant people access to your resources.

To learn more, consult the following:

- To learn whether AWS Transfer Family supports these features, see [How AWS Transfer Family works with IAM \(p. 134\)](#).
- To learn how to provide access to your resources across AWS accounts that you own, see [Providing access to an IAM user in another AWS account that you own](#) in the *IAM User Guide*.
- To learn how to provide access to your resources to third-party AWS accounts, see [Providing access to AWS accounts owned by third parties](#) in the *IAM User Guide*.
- To learn how to provide access through identity federation, see [Providing access to externally authenticated users \(identity federation\)](#) in the *IAM User Guide*.
- To learn the difference between using roles and resource-based policies for cross-account access, see [How IAM roles differ from resource-based policies](#) in the *IAM User Guide*.

Logging and monitoring in AWS Transfer Family

AWS Transfer Family is integrated with AWS CloudTrail, a service that provides a record of actions taken by a user, role, or an AWS service in AWS Transfer Family. CloudTrail captures all API calls for AWS Transfer Family as events. The calls captured include calls from the AWS Transfer Family console and code calls to the AWS Transfer Family API operations.

If you create a trail, you can enable continuous delivery of CloudTrail events to an Amazon S3 bucket, including events for AWS Transfer Family. If you don't configure a trail, you can still view the most recent events in the CloudTrail console in [Event history](#).

Using the information collected by CloudTrail, you can determine the request that was made to AWS Transfer Family, the IP address from which the request was made, who made the request, when it was made, and additional details.

To learn more about CloudTrail, see the [AWS CloudTrail User Guide](#).

AWS Transfer Family information in CloudTrail

CloudTrail is enabled on your AWS account when you create the account. When activity occurs in AWS Transfer Family, that activity is recorded in a CloudTrail event along with other AWS service events in [Event history](#). You can view, search, and download recent events in your AWS account. For more information, see [Viewing events with CloudTrail event history](#).

For an ongoing record of events in your AWS account, including events for AWS Transfer Family, create a trail. A *trail* enables CloudTrail to deliver log files to an Amazon S3 bucket. By default, when you create a trail in the console, the trail applies to all AWS Regions. The trail logs events from all Regions in the AWS partition and delivers the log files to the Amazon S3 bucket that you specify. Additionally, you can configure other AWS services to further analyze and act upon the event data collected in CloudTrail logs. For more information, see the following:

- [Overview for creating a trail](#)
- [CloudTrail supported services and integrations](#)
- [Configuring Amazon SNS notifications for CloudTrail](#)
- [Receiving CloudTrail log files from multiple regions](#) and [Receiving CloudTrail log files from multiple accounts](#)

All AWS Transfer Family actions are logged by CloudTrail and are documented in the [Actions API Reference](#). For example, calls to the `CreateServer`, `ListUsers` and `StopServer` actions generate entries in the CloudTrail log files.

Every event or log entry contains information about who generated the request. The identity information helps you determine the following:

- Whether the request was made with root or AWS Identity and Access Management (IAM) user credentials.
- Whether the request was made with temporary security credentials for a role or federated user.
- Whether the request was made by another AWS service.

For more information, see the [CloudTrail userIdentity element](#).

Understanding AWS Transfer Family log file entries

A trail is a configuration that enables delivery of events as log files to an Amazon S3 bucket that you specify. CloudTrail log files contain one or more log entries. An event represents a single request from any source and includes information about the requested action, the date and time of the action, request parameters, and so on. CloudTrail log files aren't an ordered stack trace of the public API calls, so they don't appear in any specific order.

The following example shows a CloudTrail log entry that demonstrates the `CreateServer` action.

```
{  
    "eventVersion": "1.05",  
    "userIdentity": {  
        "type": "AssumedRole",  
        "principalId": "AAAAA4FFF5HHHHH6NNWWWW:user1",  
        "arn": "arn:aws:sts::123456789102:assumed-role/Admin/user1",  
        "accountId": "123456789102",  
        "accessKeyId": "AAAAA52C2WWWWWW3BB4Z",  
        "sessionContext": {  
            "attributes": {  
                "mfaAuthenticated": "false",  
                "creationDate": "2018-12-18T20:03:57Z"  
            },  
            "sessionIssuer": {  
                "type": "Role",  
                "principalId": "AAAAA4FFF5HHHHH6NNWWWW",  
                "arn": "arn:aws:iam::123456789102:role/Admin",  
                "accountId": "123456789102",  
                "userName": "Admin"  
            }  
        }  
    },  
    "eventTime": "2018-12-18T20:30:05Z",  
    "eventSource": "transfer.amazonaws.com",  
    "eventName": "CreateServer",  
    "awsRegion": "us-east-2",  
    "sourceIPAddress": "11.22.1.2",  
    "userAgent": "aws-internal/3 aws-sdk-java/1.11.462  
Linux/4.9.124-0.1.ac.198.73.329.metal1.x86_64 OpenJDK_64-Bit_Server_VM/25.192-b12  
java/1.8.0_192",  
    "requestParameters": {  
        "loggingRole": "arn:aws:iam::123456789102:role/sftp-role"  
    },  
    "responseElements": {  
        "serverId": "s-b6118b5f29a04a9e8"  
    },  
    "requestID": "5f348905-7b83-4527-920f-ec8e6088ffd5",  
    "eventID": "82360721-d3db-4acc-b5dc-14c58c1e9899",  
    "eventType": "AwsApiCall",  
    "recipientAccountId": "123456789102"
```

},

Compliance validation for AWS Transfer Family

Third-party auditors assess the security and compliance of AWS Transfer Family as part of multiple AWS compliance programs. These include SOC, PCI, HIPAA, and others.

For a list of AWS services in scope of specific compliance programs, see [AWS Services in Scope by Compliance Program](#). For general information, see [AWS Compliance Programs](#).

You can download third-party audit reports using AWS Artifact. For more information, see [Downloading reports in AWS Artifact](#).

Your compliance responsibility when using AWS Transfer Family is determined by the sensitivity of your data, your company's compliance objectives, and applicable laws and regulations. AWS provides the following resources to help with compliance:

- [Security and Compliance Quick Start Guides](#) – These deployment guides discuss architectural considerations and provide steps for deploying security- and compliance-focused baseline environments on AWS.
- [Architecting for HIPAA Security and Compliance Whitepaper](#) – This whitepaper describes how companies can use AWS to create HIPAA-compliant applications.
- [AWS Compliance Resources](#) – This collection of workbooks and guides might apply to your industry and location.
- [AWS Config](#) – This AWS service assesses how well your resource configurations comply with internal practices, industry guidelines, and regulations.
- [AWS Security Hub](#) – This AWS service provides a comprehensive view of your security state within AWS that helps you check your compliance with security industry standards and best practices.

Resilience in AWS Transfer Family

The AWS global infrastructure is built around AWS Regions and Availability Zones. AWS Regions provide multiple physically separated and isolated Availability Zones, which are connected with low-latency, high-throughput, and highly redundant networking. With Availability Zones, you can design and operate applications and databases that automatically fail over between Availability Zones without interruption. Availability Zones are more highly available, fault tolerant, and scalable than traditional single or multiple data center infrastructures.

AWS Transfer Family supports up to 3 Availability Zones and is backed by an auto scaling, redundant fleet for your connection and transfer requests. For an example on how to build for higher redundancy and minimize network latency by using Latency-based routing, see [Minimize network latency with your AWS Transfer for SFTP servers](#).

For more information about AWS Regions and Availability Zones, see [AWS Global Infrastructure](#).

Infrastructure security in AWS Transfer Family

As a managed service, AWS Transfer Family is protected by the AWS global network security procedures that are described in the [Amazon Web Services: Overview of Security Processes](#) whitepaper.

You use AWS published API calls to access AWS Transfer Family through the network. Clients must support Transport Layer Security (TLS) 1.0 or later. We recommend TLS 1.2 or later. Clients must also support cipher suites with perfect forward secrecy (PFS) such as Ephemeral Diffie-Hellman (DHE) or Elliptic Curve Ephemeral Diffie-Hellman (ECDHE). Most modern systems such as Java 7 and later support these modes.

Additionally, requests must be signed by using an access key ID and a secret access key that is associated with an IAM principal. Or you can use the [AWS Security Token Service](#) (AWS STS) to generate temporary security credentials to sign requests.

Add a web application firewall

AWS WAF is a web application firewall that helps protect web applications and APIs from attacks. You can use it to configure a set of rules known as a *web access control list* (web ACL) that allow, block, or count web requests based on customizable web security rules and conditions that you define. For more information, see [Using AWS WAF to protect your APIs](#).

To add AWS WAF

1. Open the API Gateway console at <https://console.aws.amazon.com/apigateway/>.
2. In the **APIs** navigation pane, and then choose your custom identity provider template.
3. Choose **Stages**.
4. In the **Stages** pane, choose the name of the stage.
5. In the **Stage Editor** pane, choose the **Settings** tab.
6. Do one of the following:
 - Under **Web Application Firewall (WAF)**, for **Web ACL**, choose the web ACL that you want to associate with this stage.
 - If the web ACL you need doesn't exist, you will need to create one by doing the following:
 1. Choose **Create Web ACL**.
 2. On the AWS WAF service homepage, choose **Create web ACL**.
 3. In **Web ACL details**, for **Name**, type the name of the web ACL.
 4. In **Rules**, choose **Add rules**, then choose **Add my own rules and rule groups**.
 5. For **Rule type**, choose IP set to identify a specific list of IP addresses.
 6. For **Rule**, enter the name of the rule.
 7. For **IP set**, choose an existing IP set. To create an IP set, see [Creating an IP set](#).
 8. For **IP address to use as the originating address**, choose **IP address in header**.
 9. For **Header field name**, enter **SourceIP**.
 10. For **Position inside header**, choose **First IP address**.
 11. For **Fallback for missing IP address**, choose **Match** or **No Match** depending on how you want to handle an invalid (or missing) IP address in the header.
 12. For **Action**, choose the action of the IP set.
 13. For **Default web ACL action for requests that don't match any rules**, choose **Allow** or **Block** and then click **Next**.
 14. For steps 4 and 5, choose **Next**.
 15. In **Review and create**, review your choices, and then choose **Create web ACL**.
7. Choose **Save Changes**.
8. Choose **Resources**.
9. For **Actions**, choose **Deploy API**.

For information on how secure AWS Transfer Family with AWS web application firewall, see [Securing AWS Transfer Family with AWS Application Firewall and Amazon API Gateway](#)

AWS managed policies for AWS Transfer Family

To add permissions to users, groups, and roles, it is easier to use AWS managed policies than to write policies yourself. It takes time and expertise to [create AWS Identity and Access Management \(IAM\) customer managed policies](#) that provide your team with only the permissions that they need. To get started quickly, you can use our AWS managed policies. These policies cover common use cases and are available in your AWS account. For more information about AWS managed policies, see [AWS managed policies](#) in the *IAM User Guide*.

AWS services maintain and update AWS managed policies. You can't change the permissions in AWS managed policies. Services occasionally add additional permissions to an AWS managed policy to support new features. This type of update affects all identities (users, groups, and roles) where the policy is attached. Services are most likely to update an AWS managed policy when a new feature is launched or when new operations become available. Services do not remove permissions from an AWS managed policy, so policy updates won't break your existing permissions.

Additionally, AWS supports managed policies for job functions that span multiple services. For example, the `ReadOnlyAccess` AWS managed policy provides read-only access to all AWS services and resources. When a service launches a new feature, AWS adds read-only permissions for new operations and resources. For a list and descriptions of job function policies, see [AWS managed policies for job functions](#) in the *IAM User Guide*.

AWS managed policy: AWSTransferLoggingAccess

The `AWSTransferLoggingAccess` policy grants AWS Transfer Family full access to create log streams and groups and put log events to your account.

Permissions details

This policy includes the following permissions for Amazon CloudWatch Logs.

- `CreateLogStream` – Grants permissions for principals to create a log stream.
- `DescribeLogStreams` – Grants permissions for principals to list the log streams for the log group.
- `CreateLogGroup` – Grants permissions for principals to create log groups.
- `PutLogEvents` – Grants permissions for principals to upload a batch of log events to a log stream.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": [  
                "logs:CreateLogStream",  
                "logs:DescribeLogStreams",  
                "logs>CreateLogGroup",  
                "logs:PutLogEvents"  
            ],  
            "Resource": "*"  
        }  
    ]  
}
```

AWS managed policy: AWSTransferConsoleFullAccess

The `AWSTransferConsoleFullAccess` policy provides full access to Transfer Family through the AWS Management Console.

Permissions details

This policy includes the following permissions.

- `acm>ListCertificates` – Grants permission to retrieve a list of the certificate Amazon Resource Names (ARNs) and the domain name for each ARN.
- `ec2DescribeAddresses` – Grants permission to describe one or more Elastic IP addresses.
- `ec2DescribeAvailabilityZones` – Grants permission to describe one or more of the Availability Zones that are available to you.
- `ec2DescribeNetworkInterfaces` – Grants permission to describe one or more elastic network interfaces.
- `ec2DescribeSecurityGroups` – Grants permission to describe one or more security groups.
- `ec2DescribeSubnets` – Grants permission to describe one or more subnets.
- `ec2DescribeVpcs` – Grants permission to describe one or more virtual private clouds (VPCs).
- `ec2DescribeVpcEndpoints` – Grants permission to describe one or more VPC endpoints.
- `healthDescribeEventAggregates` – Returns the number of events of each event type (issue, scheduled change, and account notification).
- `iamGetPolicyVersion` – Grants permission to retrieve information about a version of the specified managed policy, including the policy document.
- `iamListPolicies` – Grants permission to list all managed policies.
- `iamListRoles` – Grants permission to list the IAM roles that have the specified path prefix.
- `iamPassRole` – Grants permission to pass an IAM role to Transfer Family. For more details, see [Granting a user permissions to pass a role to an AWS service](#).
- `route53ListHostedZones` – Grants permission to get a list of the public and private hosted zones that are associated with the current AWS account.
- `s3ListAllMyBuckets` – Grants permission to list all buckets owned by the authenticated sender of the request.
- `transfer*` – Grants access to Transfer Family resources. The asterisk (*) grants access to all resources.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": "iam:PassRole",  
            "Resource": "*",  
            "Condition": {  
                "StringEquals": {  
                    "iam:PassedToService": "transfer.amazonaws.com"  
                }  
            }  
        },  
        {  
            "Effect": "Allow",  
            "Action": [  
                "acm>ListCertificates",  
                "ec2DescribeAddresses",  
                "ec2DescribeAvailabilityZones",  
                "route53ListHostedZones",  
                "s3ListAllMyBuckets",  
                "transfer*"  
            ]  
        }  
    ]  
}
```

```

        "ec2:DescribeNetworkInterfaces",
        "ec2:DescribeSecurityGroups",
        "ec2:DescribeSubnets",
        "ec2:DescribeVpcs",
        "ec2:DescribeVpcEndpoints",
        "health:DescribeEventAggregates",
        "iam:GetPolicyVersion",
        "iam>ListPolicies",
        "iam>ListRoles",
        "route53>ListHostedZones",
        "s3>ListAllMyBuckets",
        "transfer:*"
    ],
    "Resource": "*"
}
]
}

```

AWS managed policy: AWSTransferFullAccess

The AWSTransferFullAccess policy provides full access to Transfer Family services.

Permissions details

This policy includes the following permissions.

- **transfer:*** – Grants permission to access Transfer Family resources. The asterisk (*) grants access to all resources.
- **iam:PassRole** – Grants permission to pass an IAM role to Transfer Family. For more details, see [Granting a user permissions to pass a role to an AWS service](#).
- **ec2:DescribeAddresses** – Grants permission to describe one or more Elastic IP addresses.
- **ec2:DescribeNetworkInterfaces** – Grants permission to describe one or more network interfaces.
- **ec2:DescribeVpcEndpoints** – Grants permission to describe one or more VPC endpoints.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "transfer:*",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "transfer.amazonaws.com"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeVpcEndpoints",
                "ec2:DescribeNetworkInterfaces",
                "ec2:DescribeAddresses"
            ],
            "Resource": "*"
        }
    ]
}
```

```
        "Resource": "*"
    ]
}
```

AWS managed policy: AWSTransferReadOnlyAccess

The `AWSTransferReadOnlyAccess` policy provides read-only access to Transfer Family services.

Permissions details

This policy includes the following permissions for Transfer Family.

- `DescribeUser` – Grants permissions for principals to view the descriptions for users.
- `DescribeServer` – Grants permissions for principals to view the descriptions for servers.
- `ListUsers` – Grants permissions for principals to list users for a server.
- `ListServers` – Grants permissions for principals to list the servers for the account.
- `TestIdentityProvider` – Grants permissions for principals to test whether the configured identity provider is set up correctly.
- `ListTagsForResource` – Grants permissions for principals to list the tags for a resource.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "VisualEditor0",
      "Effect": "Allow",
      "Action": [
        "transfer:DescribeUser",
        "transfer:DescribeServer",
        "transfer>ListUsers",
        "transfer>ListServers",
        "transfer:TestIdentityProvider",
        "transfer>ListTagsForResource"
      ],
      "Resource": "*"
    }
  ]
}
```

AWS Transfer Family updates to AWS managed policies

View details about updates to AWS managed policies for AWS Transfer Family since this service began tracking these changes. For automatic alerts about changes to this page, subscribe to the RSS feed on the [Document history for AWS Transfer Family \(p. 333\)](#) page.

Change	Description	Date
Documentation update	Added sections for each of the Transfer Family managed policies.	January 27, 2022

Change	Description	Date
AWS Transfer Family added new permissions to allow the policy to read AWS Managed Microsoft AD. – Update to an existing policy	AWS Transfer Family added new permissions to allow the policy to read AWS Managed Microsoft AD.	September 30, 2021
AWS Transfer Family started tracking changes	AWS Transfer Family started tracking changes for its AWS managed policies.	June 15, 2021

Troubleshooting

This section describes some common issues that arise when using AWS Transfer Family and how to fix them.

Topics

- [Troubleshoot Amazon EFS Service Managed Users \(p. 152\)](#)
- [Troubleshoot Amazon API Gateway Issues \(p. 152\)](#)
- [Troubleshoot policies for encrypted Amazon S3 buckets \(p. 154\)](#)
- [Troubleshoot too many authentication failures \(p. 154\)](#)
- [Troubleshoot workflow-related errors using CloudWatch \(p. 154\)](#)
- [Troubleshoot workflow copy error \(p. 155\)](#)
- [Troubleshoot missing POSIX profile \(p. 156\)](#)
- [Troubleshoot test identity provider \(p. 157\)](#)
- [Troubleshoot Amazon S3 file upload errors \(p. 157\)](#)

Troubleshoot Amazon EFS Service Managed Users

Description

You run the `sftp` command and the prompt doesn't appear, and instead you see the following message:

```
Couldn't canonicalize: Permission denied  
Need cwd
```

Cause

Your user's role does not have permission to access the Amazon Elastic File System (Amazon EFS).

Solution

Increase the policy permissions for your user's role. You can add an AWS managed policy, such as `AmazonElasticFileSystemClientFullAccess`.

Troubleshoot Amazon API Gateway Issues

This section describes possible solutions for the following issues:

- [Too many authentication failures \(p. 152\)](#)
- [Connection closed \(p. 153\)](#)

Too many authentication failures

Description

When you try to connect to your server using Secure Shell (SSH) File Transfer Protocol (SFTP), you get the following error:

```
Received disconnect from 3.15.127.197 port 22:2: Too many authentication failures
Authentication failed.
Couldn't read packet: Connection reset by peer
```

Cause

You might have entered an incorrect password for your user. Try again to enter the correct password.

If the password is correct, the issue might be caused by an invalid role Amazon Resource Name (ARN). To confirm that this is the issue, test the identity provider for your server. If you see a response similar to the following, the role ARN is a placeholder only, as indicated by the role ID value of all zeros:

```
{
    "Response": "{\"Role\": \"arn:aws:iam::000000000000:role/MyUserS3AccessRole\",
\"HomeDirectory\": \"/\",
\"StatusCode\": 200,
\"Message\": '',
\"Url\": \"https://api-gateway-ID.execute-api.us-east-1.amazonaws.com/prod/
servers/transfer-server-ID/users/myuser/config\""
}
```

Solution

Replace the placeholder role ARN with an actual role that has permission to access the server.

To update the role

1. Open the AWS CloudFormation console at <https://console.aws.amazon.com/cloudformation>.
2. Select your stack, and choose the **Parameters** tab.
3. Choose **Update**, choose **Use current template**, and then choose **Next**.
4. Replace **UserRoleArn** with a role ARN that contains sufficient permissions for accessing your Transfer Family server.

Note

To grant the necessary permissions, you can add the `AmazonAPIGatewayAdministrator` and the `AmazonS3FullAccess` managed policies to your role.

5. Choose **Next**, select **I acknowledge that AWS CloudFormation might create IAM resources**, and then choose **Update stack**.

Connection closed

Description

When you try to connect to your server using Secure Shell (SSH) File Transfer Protocol (SFTP), you get the following error:

```
Connection closed
```

Cause

One possible cause for this issue is that your CloudWatch logging role does not have a trust relationship with Transfer Family.

Solution

Make sure the logging role for the server has a trust relationship with Transfer Family. For details, see [To establish a trust relationship \(p. 9\)](#).

Troubleshoot policies for encrypted Amazon S3 buckets

Description

You have an encrypted Amazon S3 bucket that you are using as storage for your Transfer Family server. If you try to upload a file to the server, you receive the error `Couldn't close file: Permission denied`.

And if you view the server logs, you see the following errors:

```
ERROR Message="Access denied" Operation=CLOSE Path=/bucket/user/test.txt BytesIn=13
ERROR Message="Access denied"
```

Cause

Your policy does not have permission to access the encrypted bucket.

Solution

You must specify additional permissions in your policy. For details, see [Data encryption \(p. 126\)](#).

Troubleshoot too many authentication failures

Description

When you try to connect to your server using Secure Shell (SSH) File Transfer Protocol (SFTP), you receive a message similar to the following:

```
Received disconnect from 3.130.115.105 port 22:2: Too many authentication failures
Authentication failed.
```

Cause

You have not added an RSA key pair for your user, and you need to authenticate using a password.

Solution

You can specify the `-o PubkeyAuthentication=no` option when you run the `sftp` command. This forces the system to request your password. For example:

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

Troubleshoot workflow-related errors using CloudWatch

Description

If you are having issues with your workflows, you can use Amazon CloudWatch to investigate the cause.

Cause

There can be several causes: use CloudWatch Logs to investigate.

Solution

Transfer Family emits workflow execution status into CloudWatch Logs. The following types of workflow errors can appear in CloudWatch Logs:

- "type": "StepErrored"
- "type": "ExecutionErrored"
- "type": "ExecutionThrottled"

You can filter your workflows execution logs using different filter and pattern syntax. For example, you can create a log filter in your CloudWatch logs to capture workflow execution logs that contain the **ExecutionErrored** message. For details, see [Real-time processing of log data with subscriptions](#) and [Filter and pattern syntax](#) in the *Amazon CloudWatch User Guide*.

StepErrored

```
2021-10-29T12:57:26.272-05:00
  {"type":"StepErrored","details": {
    "errorType":"BAD_REQUEST","errorMessage":"Cannot
      tag Efs file","stepType":"TAG","stepName":"successful_tag_step"}, 
    "workflowId":"w-
abcdef01234567890","executionId":"1234abcd-56ef-78gh-90ij-1234klmno567",
    "transferDetails":
    {"serverId":"s-1234567890abcdef0","username":"lhr","sessionId":"1234567890abcdef0"}}
```

Here, **StepErrored** indicates that a step within the workflow has generated an error. In a single workflow, you can have multiple steps configured. This error tells you in which step the error occurred and provides an error message. In this particular example, the step was configured to tag a file: however, tagging a file in an EFS file system is not supported, so the step errored out.

ExecutionErrored

```
2021-10-29T12:57:26.618-05:00
  {"type":"ExecutionErrored","details":{}, "workflowId":"w-w-abcdef01234567890",
    "executionId":"1234abcd-56ef-78gh-90ij-1234klmno567","transferDetails":
    {"serverId":"s-1234567890abcdef0",
      "username":"lhr","sessionId":"1234567890abcdef0"}}
```

When a workflow is not able to execute any step, it generates an **ExecutionErrored** message. For example, if you have configured a single step in a given workflow, and if the step is not able to execute, the overall workflow fails.

Executionthrottled

Execution is throttled if a workflow is getting triggered at rate that is faster than the system can support. This log message indicates that you need to slow down your execution rate for workflows. If you are not able to scale down your workflow execution rate, contact technical support at [Contact AWS](#).

Troubleshoot workflow copy error

Description

If you are executing a workflow that contains a step to copy the uploaded file, you could encounter the following error:

```
{
  "type": "StepErrored", "details": {
    "errorType": "BAD_REQUEST", "errorMessage": "Bad Request (Service: Amazon S3; Status Code: 400; Error Code: 400 Bad Request;
    Request ID: ESYKP3R7W4VT1RB0; S3 Extended Request ID: request-ID Proxy: null)",
    "stepType": "COPY", "stepName": "copy-step-name" },
    "workflowId": "workflow-ID",
    "executionId": "execution-ID",
    "transferDetails": {
      "serverId": "server-ID",
      "username": "user-name",
      "sessionId": "session-ID"
    }
}
```

Cause

The source file is in an Amazon S3 bucket that is in a different region than the destination bucket.

Solution

If you are executing a workflow that includes a copy step, ensure that the source and destination buckets are in the same region.

Troubleshoot missing POSIX profile

Description

If you are using Amazon EFS storage for your server, and using a custom identity provider, you need to provide your Lambda function with a POSIX profile.

Cause

One possible cause is that the templates we provide for creating a AWS Lambda-backed API Gateway method do not currently contain POSIX information. Alternatively, the format that you use for providing the POSIX information might not be getting parsed correctly by Transfer Family.

Solution

You need to make sure you are providing a JSON element to Transfer Family for the `PosixProfile`.

For example, if you are using Python, you could add the following line where you parse the `PosixProfile` parameter:

```
if PosixProfile:
    response_data["PosixProfile"] = json.loads(PosixProfile)
```

Or, in javascript, you could add the following:

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

Where the `uid-value` and `gid-value` are integers, 0 or greater, that represent the User ID and Group ID respectively.

These code examples send the PosixProfile to Transfer Family as a JSON object, rather than as a string.

Also, within Secrets Manager, you need to store the PosixProfile as the following:

```
{"Uid": your-uid, "Gid": your-gid, "SecondaryGids": []}
```

Replace *your-uid* and *your-gid* with your actual values for Gid and Uid.

Troubleshoot test identity provider

Description

If you test your identity provider using the console or the `TestIdentityProvider` API call, the returned message is blank.

Cause

The most likely cause is that the authentication failed due to an incorrect username or password.

Solution

Make sure that you are using the correct credentials for your user, and make updates to the username or password, if necessary.

Troubleshoot Amazon S3 file upload errors

Description

You receive the following error message when you are attempting to upload a file to S3 storage using Transfer: AWS Transfer does not support random access writes to S3 objects.

Cause

Transfer Family does not support multiple connections for a single transfer, when you are using Amazon S3 for your server's storage.

Solution

If your Transfer Family server is using Amazon S3 for its storage, make sure to disable any options for your client software that mention using multiple connections for a single transfer.

API reference

The following sections document the AWS Transfer Family API service calls, data types, parameters, and errors. Please refer to the [Welcome to the AWS Transfer Family API \(p. 158\)](#) section for information about API entities and conventions that are in use with the AWS Transfer Family API.

Topics

- [Welcome to the AWS Transfer Family API \(p. 158\)](#)
- [Actions \(p. 159\)](#)
- [Data Types \(p. 272\)](#)
- [Common Parameters \(p. 329\)](#)
- [Common Errors \(p. 330\)](#)

Welcome to the AWS Transfer Family API

AWS Transfer Family is a secure transfer service that you can use to transfer files into and out of Amazon Simple Storage Service (Amazon S3) storage over the following protocols:

- Secure Shell (SSH) File Transfer Protocol (SFTP)
- File Transfer Protocol Secure (FTPS)
- File Transfer Protocol (FTP)

File transfer protocols are used in data exchange workflows across different industries such as financial services, healthcare, advertising, and retail, among others. AWS Transfer Family simplifies the migration of file transfer workflows to AWS.

To use the AWS Transfer Family service, you instantiate a server in the AWS Region of your choice. You can create the server, list available servers, and update and delete servers. The server is the entity that requests file operations from AWS Transfer Family. Servers have a number of important properties. The server is a named instance as identified by a system assigned `ServerId` identifier. You can optionally assign a hostname, or even a custom hostname to a server. The service bills for any instantiated servers (even ones `OFFLINE`), and for the amount of data transferred.

Users must be known to the server that requests file operations. A user as identified by their user name is assigned to a server. User names are used to authenticate requests. A server can have only one authentication method: `AWS_DIRECTORY_SERVICE`, `SERVICE_MANAGED`, `AWS_LAMBDA`, or `API_GATEWAY`.

You can use any of the following identity provider types to authenticate users:

- For `SERVICE_MANAGED`, an SSH public key is stored with the user's properties on a server. A user can have one or more SSH public keys on file for the `SERVICE_MANAGED` authentication method. When a client requests a file operation for `SERVICE_MANAGED` method, the client provides the user name and SSH private key, which is authenticated, and access is provided.
- You can manage user authentication and access with your Microsoft Active Directory groups by selecting the `AWS_DIRECTORY_SERVICE` authentication method.
- You can connect to a custom identity provider by using AWS Lambda. Choose the `AWS_LAMBDA` authentication method.
- You can also authenticate user requests using a custom authentication method that provides both user authentication and access. This method relies on the Amazon API Gateway to use your API call from your identity provider to validate user requests. This method is referred to as `API_GATEWAY` in API

calls, and as **Custom** in the console. You might use this custom method to authenticate users against a directory service, a database name/password pair, or some other mechanism.

Users are assigned a policy with a trust relationship between themselves and an Amazon S3 bucket. They might be able to access all or part of a bucket. For a server to act on a user's behalf, the server must inherit the trust relationship from the user. An AWS Identity and Access Management (IAM) role is created that contains the trust relationship, and that role is assigned an `AssumeRole` action. The server then can perform file operations as if it were the user.

Users who have a `home` directory property set will have that directory (or folder) act as the target and source of file operations. When no `home` directory is set, the bucket's `root` directory becomes the landing directory.

Servers, users, and roles are all identified by their Amazon Resource Name (ARN). You can assign tags, which are key-value pairs, to entities with an ARN. Tags are metadata that can be used to group or search for these entities. One example where tags are useful is for accounting purposes.

The following conventions are observed in AWS Transfer Family ID formats:

- `ServerId` values take the form `s-01234567890abcdef`.
- `SshPublicKeyId` values take the form `key-01234567890abcdef`.

Amazon Resource Name (ARN) formats take the following form:

- For servers, ARNs take the form `arn:aws:transfer:region:account-id:server/server-id`.

An example of a server ARN is: `arn:aws:transfer:us-east-1:123456789012:server/s-01234567890abcdef`.

- For users, ARNs take the form `arn:aws:transfer:region:account-id:user/server-id/username`.

An example is `arn:aws:transfer:us-east-1:123456789012:user/s-01234567890abcdef/user1`.

DNS entries (endpoints) in use are as follows:

- API endpoints take the form `transfer.region.amazonaws.com`.
- Server endpoints take the form `server.transfer.region.amazonaws.com`.

For a list of Transfer Family endpoints by AWS Region, see the [AWS Transfer Family endpoints and quotas](#) in the [AWS General Reference](#).

This API interface reference for AWS Transfer Family contains documentation for a programming interface that you can use to manage AWS Transfer Family. The reference structure is as follows:

- For the alphabetical list of API actions, see [Actions \(p. 159\)](#).
- For the alphabetical list of data types, see [Data Types \(p. 272\)](#).
- For a list of common query parameters, see [Common Parameters \(p. 329\)](#).
- For descriptions of the error codes, see [Common Errors \(p. 330\)](#).

Actions

The following actions are supported:

- [CreateAccess \(p. 161\)](#)
- [CreateServer \(p. 166\)](#)
- [CreateUser \(p. 173\)](#)
- [CreateWorkflow \(p. 179\)](#)
- [DeleteAccess \(p. 183\)](#)
- [DeleteServer \(p. 185\)](#)
- [DeleteSshPublicKey \(p. 187\)](#)
- [DeleteUser \(p. 190\)](#)
- [DeleteWorkflow \(p. 192\)](#)
- [DescribeAccess \(p. 194\)](#)
- [DescribeExecution \(p. 197\)](#)
- [DescribeSecurityPolicy \(p. 200\)](#)
- [DescribeServer \(p. 202\)](#)
- [DescribeUser \(p. 206\)](#)
- [DescribeWorkflow \(p. 210\)](#)
- [ImportSshPublicKey \(p. 213\)](#)
- [ListAccesses \(p. 217\)](#)
- [ListExecutions \(p. 220\)](#)
- [ListSecurityPolicies \(p. 224\)](#)
- [ListServers \(p. 226\)](#)
- [ListTagsForResource \(p. 229\)](#)
- [ListUsers \(p. 232\)](#)
- [ListWorkflows \(p. 236\)](#)
- [SendWorkflowStepState \(p. 238\)](#)
- [StartServer \(p. 241\)](#)
- [StopServer \(p. 243\)](#)
- [TagResource \(p. 246\)](#)
- [TestIdentityProvider \(p. 249\)](#)
- [UntagResource \(p. 254\)](#)
- [UpdateAccess \(p. 257\)](#)
- [UpdateServer \(p. 262\)](#)
- [UpdateUser \(p. 268\)](#)

CreateAccess

Used by administrators to choose which groups in the directory should have access to upload and download files over the enabled protocols using AWS Transfer Family. For example, a Microsoft Active Directory might contain 50,000 users, but only a small fraction might need the ability to transfer files to the server. An administrator can use `CreateAccess` to limit the access to the correct set of users who need this ability.

Request Syntax

```
{  
    "ExternalId": "string",  
    "HomeDirectory": "string",  
    "HomeDirectoryMappings": [  
        {  
            "Entry": "string",  
            "Target": "string"  
        }  
    ],  
    "HomeDirectoryType": "string",  
    "Policy": "string",  
    "PosixProfile": {  
        "Gid": number,  
        "SecondaryGids": [ number ],  
        "Uid": number  
    },  
    "Role": "string",  
    "ServerId": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 329\)](#).

The request accepts the following data in JSON format.

ExternalId (p. 161)

A unique identifier that is required to identify specific groups within your directory. The users of the group that you associate have access to your Amazon S3 or Amazon EFS resources over the enabled protocols using AWS Transfer Family. If you know the group name, you can view the SID values by running the following command using Windows PowerShell.

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

In that command, replace *YourGroupName* with the name of your Active Directory group.

The regex used to validate this parameter is a string of characters consisting of uppercase and lowercase alphanumeric characters with no spaces. You can also include underscores or any of the following characters: =,:@:/-

Type: String

Length Constraints: Minimum length of 1. Maximum length of 256.

Pattern: ^S-1-[\d-]+\$

Required: Yes

[HomeDirectory \(p. 161\)](#)

The landing directory (folder) for a user when they log in to the server using the client.

A HomeDirectory example is `/bucket_name/home/mydirectory`.

Type: String

Length Constraints: Maximum length of 1024.

Pattern: `^$ | /.*`

Required: No

[HomeDirectoryMappings \(p. 161\)](#)

Logical directory mappings that specify what Amazon S3 or Amazon EFS paths and keys should be visible to your user and how you want to make them visible. You must specify the `Entry` and `Target` pair, where `Entry` shows how the path is made visible and `Target` is the actual Amazon S3 or Amazon EFS path. If you only specify a target, it is displayed as is. You also must ensure that your AWS Identity and Access Management (IAM) role provides access to paths in `Target`. This value can only be set when `HomeDirectoryType` is set to `LOGICAL`.

The following is an `Entry` and `Target` pair example.

```
[ { "Entry": "/directory1", "Target": "/bucket_name/home/mydirectory" } ]
```

In most cases, you can use this value instead of the session policy to lock down your user to the designated home directory ("chroot"). To do this, you can set `Entry` to `/` and set `Target` to the `HomeDirectory` parameter value.

The following is an `Entry` and `Target` pair example for chroot.

```
[ { "Entry": "/", "Target": "/bucket_name/home/mydirectory" } ]
```

Type: Array of [HomeDirectoryMapEntry \(p. 300\)](#) objects

Array Members: Minimum number of 1 item. Maximum number of 50 items.

Required: No

[HomeDirectoryType \(p. 161\)](#)

The type of landing directory (folder) you want your users' home directory to be when they log into the server. If you set it to `PATH`, the user will see the absolute Amazon S3 bucket or EFS paths as is in their file transfer protocol clients. If you set it `LOGICAL`, you need to provide mappings in the `HomeDirectoryMappings` for how you want to make Amazon S3 or EFS paths visible to your users.

Type: String

Valid Values: `PATH` | `LOGICAL`

Required: No

[Policy \(p. 161\)](#)

A session policy for your user so that you can use the same IAM role across multiple users. This policy scopes down user access to portions of their Amazon S3 bucket. Variables that you can use inside this policy include `${Transfer:UserName}`, `${Transfer:HomeDirectory}`, and `${Transfer:HomeBucket}`.

Note

This only applies when the domain of `ServerId` is S3. EFS does not use session policies.

For session policies, AWS Transfer Family stores the policy as a JSON blob, instead of the Amazon Resource Name (ARN) of the policy. You save the policy as a JSON blob and pass it in the `Policy` argument.

For an example of a session policy, see [Example session policy](#).

For more information, see [AssumeRole](#) in the *AWS Security Token Service API Reference*.

Type: String

Length Constraints: Maximum length of 2048.

Required: No

[PosixProfile \(p. 161\)](#)

The full POSIX identity, including user ID (`Uid`), group ID (`Gid`), and any secondary groups IDs (`SecondaryGids`), that controls your users' access to your Amazon EFS file systems. The POSIX permissions that are set on files and directories in your file system determine the level of access your users get when transferring files into and out of your Amazon EFS file systems.

Type: [PosixProfile \(p. 314\)](#) object

Required: No

[Role \(p. 161\)](#)

Specifies the Amazon Resource Name (ARN) of the IAM role that controls your users' access to your Amazon S3 bucket or EFS file system. The policies attached to this role determine the level of access that you want to provide your users when transferring files into and out of your Amazon S3 bucket or EFS file system. The IAM role should also contain a trust relationship that allows the server to access your resources when servicing your users' transfer requests.

Type: String

Length Constraints: Minimum length of 20. Maximum length of 2048.

Pattern: `arn:.*role/.*`

Required: Yes

[ServerId \(p. 161\)](#)

A system-assigned unique identifier for a server instance. This is the specific server that you added your user to.

Type: String

Length Constraints: Fixed length of 19.

Pattern: `^s-([0-9a-f]{17})$`

Required: Yes

Response Syntax

```
{  
    "ExternalId": "string",  
    "ServerId": "string"  
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

[ExternalId \(p. 163\)](#)

The external ID of the group whose users have access to your Amazon S3 or Amazon EFS resources over the enabled protocols using AWS Transfer Family.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 256.

Pattern: ^S-1-[\d-]+\$

[ServerId \(p. 163\)](#)

The ID of the server that the user is attached to.

Type: String

Length Constraints: Fixed length of 19.

Pattern: ^s-([0-9a-f]{17})\$

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 330\)](#).

InternalServiceError

This exception is thrown when an error occurs in the AWSTransfer Family service.

HTTP Status Code: 500

InvalidRequestException

This exception is thrown when the client submits a malformed request.

HTTP Status Code: 400

ResourceExistsException

The requested resource does not exist.

HTTP Status Code: 400

ResourceNotFoundException

This exception is thrown when a resource is not found by the AWSTransfer Family service.

HTTP Status Code: 400

ServiceUnavailableException

The request has failed because the AWSTransfer Family service is not available.

HTTP Status Code: 500

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)

- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

CreateServer

Instantiates an auto-scaling virtual server based on the selected file transfer protocol in AWS. When you make updates to your file transfer protocol-enabled server or when you work with users, use the service-generated `ServerId` property that is assigned to the newly created server.

Request Syntax

```
{  
    "Certificate": "string",  
    "Domain": "string",  
    "EndpointDetails": {  
        "AddressAllocationIds": [ "string" ],  
        "SecurityGroupIds": [ "string" ],  
        "SubnetIds": [ "string" ],  
        "VpcEndpointId": "string",  
        "VpcId": "string"  
    },  
    "EndpointType": "string",  
    "HostKey": "string",  
    "IdentityProviderDetails": {  
        "DirectoryId": "string",  
        "Function": "string",  
        "InvocationRole": "string",  
        "Url": "string"  
    },  
    "IdentityProviderType": "string",  
    "LoggingRole": "string",  
    "ProtocolDetails": {  
        "PassiveIp": "string",  
        "TlsSessionResumptionMode": "string"  
    },  
    "Protocols": [ "string" ],  
    "SecurityPolicyName": "string",  
    "Tags": [  
        {  
            "Key": "string",  
            "Value": "string"  
        }  
    ],  
    "WorkflowDetails": {  
        "OnUpload": [  
            {  
                "ExecutionRole": "string",  
                "WorkflowId": "string"  
            }  
        ]  
    }  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 329\)](#).

The request accepts the following data in JSON format.

Certificate (p. 166)

The Amazon Resource Name (ARN) of the AWS Certificate Manager (ACM) certificate. Required when `Protocols` is set to `FTPS`.

To request a new public certificate, see [Request a public certificate](#) in the *AWS Certificate Manager User Guide*.

To import an existing certificate into ACM, see [Importing certificates into ACM](#) in the *AWS Certificate Manager User Guide*.

To request a private certificate to use FTPS through private IP addresses, see [Request a private certificate](#) in the *AWS Certificate Manager User Guide*.

Certificates with the following cryptographic algorithms and key sizes are supported:

- 2048-bit RSA (RSA_2048)
- 4096-bit RSA (RSA_4096)
- Elliptic Prime Curve 256 bit (EC_prime256v1)
- Elliptic Prime Curve 384 bit (EC_secp384r1)
- Elliptic Prime Curve 521 bit (EC_secp521r1)

Note

The certificate must be a valid SSL/TLS X.509 version 3 certificate with FQDN or IP address specified and information about the issuer.

Type: String

Length Constraints: Maximum length of 1600.

Required: No

Domain (p. 166)

The domain of the storage system that is used for file transfers. There are two domains available: Amazon Simple Storage Service (Amazon S3) and Amazon Elastic File System (Amazon EFS). The default value is S3.

Note

After the server is created, the domain cannot be changed.

Type: String

Valid Values: S3 | EFS

Required: No

EndpointDetails (p. 166)

The virtual private cloud (VPC) endpoint settings that are configured for your server. When you host your endpoint within your VPC, you can make it accessible only to resources within your VPC, or you can attach Elastic IP addresses and make it accessible to clients over the internet. Your VPC's default security groups are automatically assigned to your endpoint.

Type: [EndpointDetails \(p. 294\)](#) object

Required: No

EndpointType (p. 166)

The type of endpoint that you want your server to use. You can choose to make your server's endpoint publicly accessible (PUBLIC) or host it inside your VPC. With an endpoint that is hosted in a VPC, you can restrict access to your server and resources only within your VPC or choose to make it internet facing by attaching Elastic IP addresses directly to it.

Note

After May 19, 2021, you won't be able to create a server using `EndpointType=VPC_ENDPOINT` in your AWS account if your account hasn't

already done so before May 19, 2021. If you have already created servers with `EndpointType=VPC_ENDPOINT` in your AWS account on or before May 19, 2021, you will not be affected. After this date, use `EndpointType=VPC`.

For more information, see [Discontinuing the use of VPC_ENDPOINT \(p. 42\)](#).

It is recommended that you use VPC as the `EndpointType`. With this endpoint type, you have the option to directly associate up to three Elastic IPv4 addresses (BYO IP included) with your server's endpoint and use VPC security groups to restrict traffic by the client's public IP address. This is not possible with `EndpointType` set to `VPC_ENDPOINT`.

Type: String

Valid Values: `PUBLIC` | `VPC` | `VPC_ENDPOINT`

Required: No

[HostKey \(p. 166\)](#)

The RSA private key as generated by the `ssh-keygen -N "" -m PEM -f my-new-server-key` command.

Important

If you aren't planning to migrate existing users from an existing SFTP-enabled server to a new server, don't update the host key. Accidentally changing a server's host key can be disruptive.

For more information, see [Change the host key for your SFTP-enabled server](#) in the *AWS Transfer Family User Guide*.

Type: String

Length Constraints: Maximum length of 4096.

Required: No

[IdentityProviderDetails \(p. 166\)](#)

Required when `IdentityProviderType` is set to `AWS_DIRECTORY_SERVICE` or `API_GATEWAY`. Accepts an array containing all of the information required to use a directory in `AWS_DIRECTORY_SERVICE` or invoke a customer-supplied authentication API, including the API Gateway URL. Not required when `IdentityProviderType` is set to `SERVICE_MANAGED`.

Type: [IdentityProviderDetails \(p. 301\)](#) object

Required: No

[IdentityProviderType \(p. 166\)](#)

Specifies the mode of authentication for a server. The default value is `SERVICE_MANAGED`, which allows you to store and access user credentials within the AWS Transfer Family service.

Use `AWS_DIRECTORY_SERVICE` to provide access to Active Directory groups in AWS Managed Active Directory or Microsoft Active Directory in your on-premises environment or in AWS using AD Connectors. This option also requires you to provide a Directory ID using the `IdentityProviderDetails` parameter.

Use the `API_GATEWAY` value to integrate with an identity provider of your choosing. The `API_GATEWAY` setting requires you to provide an API Gateway endpoint URL to call for authentication using the `IdentityProviderDetails` parameter.

Use the `AWS_LAMBDA` value to directly use a Lambda function as your identity provider. If you choose this value, you must specify the ARN for the lambda function in the `Function` parameter for the `IdentityProviderDetails` data type.

Type: String

Valid Values: SERVICE_MANAGED | API_GATEWAY | AWS_DIRECTORY_SERVICE | AWS_LAMBDA

Required: No

[LoggingRole \(p. 166\)](#)

Specifies the Amazon Resource Name (ARN) of the AWS Identity and Access Management (IAM) role that allows a server to turn on Amazon CloudWatch logging for Amazon S3 or Amazon EFS events. When set, user activity can be viewed in your CloudWatch logs.

Type: String

Length Constraints: Minimum length of 20. Maximum length of 2048.

Pattern: arn:.*role/.*

Required: No

[ProtocolDetails \(p. 166\)](#)

The protocol settings that are configured for your server.

Use the `PassiveIp` parameter to indicate passive mode (for FTP and FTPS protocols). Enter a single dotted-quad IPv4 address, such as the external IP address of a firewall, router, or load balancer.

Use the `TlsSessionResumptionMode` parameter to determine whether or not your Transfer server resumes recent, negotiated sessions through a unique session ID.

Type: [ProtocolDetails \(p. 315\)](#) object

Required: No

[Protocols \(p. 166\)](#)

Specifies the file transfer protocol or protocols over which your file transfer protocol client can connect to your server's endpoint. The available protocols are:

- **SFTP** (Secure Shell (SSH) File Transfer Protocol): File transfer over SSH
- **FTPS** (File Transfer Protocol Secure): File transfer with TLS encryption
- **FTP** (File Transfer Protocol): Unencrypted file transfer

Note

If you select `FTPS`, you must choose a certificate stored in AWS Certificate Manager (ACM) which is used to identify your server when clients connect to it over `FTPS`.

If `Protocol` includes either `FTP` or `FTPS`, then the `EndpointType` must be `VPC` and the `IdentityProviderType` must be `AWS_DIRECTORY_SERVICE` or `API_GATEWAY`.

If `Protocol` includes `FTP`, then `AddressAllocationIds` cannot be associated.

If `Protocol` is set only to `SFTP`, the `EndpointType` can be set to `PUBLIC` and the `IdentityProviderType` can be set to `SERVICE_MANAGED`.

Type: Array of strings

Array Members: Minimum number of 1 item. Maximum number of 3 items.

Valid Values: `SFTP` | `FTP` | `FTPS`

Required: No

[SecurityPolicyName \(p. 166\)](#)

Specifies the name of the security policy that is attached to the server.

Type: String

Length Constraints: Maximum length of 100.

Pattern: TransferSecurityPolicy- . +

Required: No

[Tags \(p. 166\)](#)

Key-value pairs that can be used to group and search for servers.

Type: Array of [Tag \(p. 323\)](#) objects

Array Members: Minimum number of 1 item. Maximum number of 50 items.

Required: No

[WorkflowDetails \(p. 166\)](#)

Specifies the workflow ID for the workflow to assign and the execution role used for executing the workflow.

Type: [WorkflowDetails \(p. 327\)](#) object

Required: No

Response Syntax

```
{  
    "ServerId": "string"  
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

[ServerId \(p. 170\)](#)

The service-assigned ID of the server that is created.

Type: String

Length Constraints: Fixed length of 19.

Pattern: ^s-([0-9a-f]{17})\$

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 330\)](#).

AccessDeniedException

You do not have sufficient access to perform this action.

HTTP Status Code: 400

InternalServiceError

This exception is thrown when an error occurs in the AWSTransfer Family service.

HTTP Status Code: 500

InvalidRequestException

This exception is thrown when the client submits a malformed request.

HTTP Status Code: 400

ResourceExistsException

The requested resource does not exist.

HTTP Status Code: 400

ResourceNotFoundException

This exception is thrown when a resource is not found by the AWSTransfer Family service.

HTTP Status Code: 400

ServiceUnavailableException

The request has failed because the AWSTransfer Family service is not available.

HTTP Status Code: 500

ThrottlingException

The request was denied due to request throttling.

HTTP Status Code: 400

HTTP Status Code: 400

Examples

Example

The following example creates a new server using a VPC_ENDPOINT.

Sample Request

```
{  
    "EndpointType": "VPC",  
    "EndpointDetails": ...,  
    "HostKey": "Your RSA private key",  
    "IdentityProviderDetails": "IdentityProvider",  
    "IdentityProviderType": "SERVICE_MANAGED",  
    "LoggingRole": "CloudWatchLoggingRole",  
    "Tags": [  
        {  
            "Key": "Name",  
            "Value": "MyServer"  
        }  
    ]  
}
```

Example

This example illustrates one usage of CreateServer.

Sample Response

```
{  
    "ServerId": "s-01234567890abcdef"  
}
```

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

CreateUser

Creates a user and associates them with an existing file transfer protocol-enabled server. You can only create and associate users with servers that have the `IdentityProviderType` set to `SERVICE_MANAGED`. Using parameters for `CreateUser`, you can specify the user name, set the home directory, store the user's public key, and assign the user's AWS Identity and Access Management (IAM) role. You can also optionally add a session policy, and assign metadata with tags that can be used to group and search for users.

Request Syntax

```
{  
    "HomeDirectory": "string",  
    "HomeDirectoryMappings": [  
        {  
            "Entry": "string",  
            "Target": "string"  
        }  
    ],  
    "HomeDirectoryType": "string",  
    "Policy": "string",  
    "PosixProfile": {  
        "Gid": number,  
        "SecondaryGids": [ number ],  
        "Uid": number  
    },  
    "Role": "string",  
    "ServerId": "string",  
    "SshPublicKeyBody": "string",  
    "Tags": [  
        {  
            "Key": "string",  
            "Value": "string"  
        }  
    ],  
    "UserName": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 329\)](#).

The request accepts the following data in JSON format.

HomeDirectory (p. 173)

The landing directory (folder) for a user when they log in to the server using the client.

A `HomeDirectory` example is `/bucket_name/home/mydirectory`.

Type: String

Length Constraints: Maximum length of 1024.

Pattern: `^$ | /.*`

Required: No

[HomeDirectoryMappings \(p. 173\)](#)

Logical directory mappings that specify what Amazon S3 or Amazon EFS paths and keys should be visible to your user and how you want to make them visible. You must specify the `Entry` and `Target` pair, where `Entry` shows how the path is made visible and `Target` is the actual Amazon S3 or Amazon EFS path. If you only specify a target, it is displayed as is. You also must ensure that your AWS Identity and Access Management (IAM) role provides access to paths in `Target`. This value can only be set when `HomeDirectoryType` is set to `LOGICAL`.

The following is an `Entry` and `Target` pair example.

```
[ { "Entry": "/directory1", "Target": "/bucket_name/home/mydirectory" } ]
```

In most cases, you can use this value instead of the session policy to lock your user down to the designated home directory ("chroot"). To do this, you can set `Entry` to `/` and set `Target` to the `HomeDirectory` parameter value.

The following is an `Entry` and `Target` pair example for chroot.

```
[ { "Entry": "/", "Target": "/bucket_name/home/mydirectory" } ]
```

Type: Array of [HomeDirectoryMapEntry \(p. 300\)](#) objects

Array Members: Minimum number of 1 item. Maximum number of 50 items.

Required: No

[HomeDirectoryType \(p. 173\)](#)

The type of landing directory (folder) you want your users' home directory to be when they log into the server. If you set it to `PATH`, the user will see the absolute Amazon S3 bucket or EFS paths as is in their file transfer protocol clients. If you set it `LOGICAL`, you need to provide mappings in the `HomeDirectoryMappings` for how you want to make Amazon S3 or EFS paths visible to your users.

Type: String

Valid Values: `PATH` | `LOGICAL`

Required: No

[Policy \(p. 173\)](#)

A session policy for your user so that you can use the same IAM role across multiple users. This policy scopes down user access to portions of their Amazon S3 bucket. Variables that you can use inside this policy include `Transfer:UserName`, `Transfer:HomeDirectory`, and `Transfer:HomeBucket`.

Note

This only applies when the domain of `ServerId` is `S3`. EFS does not use session policies. For session policies, AWS Transfer Family stores the policy as a JSON blob, instead of the Amazon Resource Name (ARN) of the policy. You save the policy as a JSON blob and pass it in the `Policy` argument.

For an example of a session policy, see [Example session policy](#).

For more information, see [AssumeRole](#) in the [AWS Security Token Service API Reference](#).

Type: String

Length Constraints: Maximum length of 2048.

Required: No

[PosixProfile \(p. 173\)](#)

Specifies the full POSIX identity, including user ID (`Uid`), group ID (`Gid`), and any secondary groups IDs (`SecondaryGids`), that controls your users' access to your Amazon EFS file systems. The POSIX

permissions that are set on files and directories in Amazon EFS determine the level of access your users get when transferring files into and out of your Amazon EFS file systems.

Type: [PosixProfile \(p. 314\)](#) object

Required: No

[Role \(p. 173\)](#)

Specifies the Amazon Resource Name (ARN) of the IAM role that controls your users' access to your Amazon S3 bucket or EFS file system. The policies attached to this role determine the level of access that you want to provide your users when transferring files into and out of your Amazon S3 bucket or EFS file system. The IAM role should also contain a trust relationship that allows the server to access your resources when servicing your users' transfer requests.

Type: String

Length Constraints: Minimum length of 20. Maximum length of 2048.

Pattern: `arn:.*role/.*`

Required: Yes

[ServerId \(p. 173\)](#)

A system-assigned unique identifier for a server instance. This is the specific server that you added your user to.

Type: String

Length Constraints: Fixed length of 19.

Pattern: `^s-([0-9a-f]{17})$`

Required: Yes

[SshPublicKeyBody \(p. 173\)](#)

The public portion of the Secure Shell (SSH) key used to authenticate the user to the server.

Note

Currently, Transfer Family does not accept elliptical curve keys (keys beginning with ecdsa).

Type: String

Length Constraints: Maximum length of 2048.

Pattern: `^ssh-rsa\s+[A-Za-z0-9+/]+[=]{0,3}(\s+.+)?\s*$`

Required: No

[Tags \(p. 173\)](#)

Key-value pairs that can be used to group and search for users. Tags are metadata attached to users for any purpose.

Type: Array of [Tag \(p. 323\)](#) objects

Array Members: Minimum number of 1 item. Maximum number of 50 items.

Required: No

[UserName \(p. 173\)](#)

A unique string that identifies a user and is associated with a `ServerId`. This user name must be a minimum of 3 and a maximum of 100 characters long. The following are valid characters: a-z, A-

Z, 0-9, underscore '_', hyphen '-', period '.', and at sign '@'. The user name can't start with a hyphen, period, or at sign.

Type: String

Length Constraints: Minimum length of 3. Maximum length of 100.

Pattern: `^[\w][\w@.-]{2,99}$`

Required: Yes

Response Syntax

```
{  
    "ServerId": "string",  
    "UserName": "string"  
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

[ServerId \(p. 176\)](#)

The ID of the server that the user is attached to.

Type: String

Length Constraints: Fixed length of 19.

Pattern: `^s-([0-9a-f]{17})$`

[UserName \(p. 176\)](#)

A unique string that identifies a user account associated with a server.

Type: String

Length Constraints: Minimum length of 3. Maximum length of 100.

Pattern: `^[\w][\w@.-]{2,99}$`

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 330\)](#).

InternalServiceError

This exception is thrown when an error occurs in the AWSTransfer Family service.

HTTP Status Code: 500

InvalidRequestException

This exception is thrown when the client submits a malformed request.

HTTP Status Code: 400

ResourceExistsException

The requested resource does not exist.

HTTP Status Code: 400

ResourceNotFoundException

This exception is thrown when a resource is not found by the AWSTransfer Family service.

HTTP Status Code: 400

ServiceUnavailableException

The request has failed because the AWSTransfer Family service is not available.

HTTP Status Code: 500

Examples

Example

The following example associates a user with a server.

Sample Request

```
{  
    "HomeDirectory": "/bucket_name/home/mydirectory",  
    "HomeDirectoryMappings": [  
        {  
            "Entry": "/directory1",  
            "Target": "/bucket_name/home/mydirectory"  
        }  
    ],  
    "HomeDirectoryType": "PATH",  
    "Policy": {  
        "Version": "2012-10-17",  
        "Statement": [  
            {  
                "Sid": "AllowFullAccessToBucket",  
                "Action": [  
                    "s3:*"  
                ],  
                "Effect": "Allow",  
                "Resource": [  
                    "arn:aws:s3:::bucket_name",  
                    "arn:aws:s3:::bucket_name/*"  
                ]  
            }  
        ]  
    },  
    "SshPublicKeyBody":  
        "AAAAAB3NzaC1yc2EAAAQABAAQCOtfCAis3aHfM6yc8KWAlMQxVDBHyccCde9MdLf4DQNXd8HjAHf  
        +Bc1vGGCAREFUL1NO2PEEKING3ALLOWEDfif  
        +JBecywfO35Cm6IKIV0JF2YOPXvOuQRs80hQaBUvQL9xw6VEb4xzbit2QB6",  
    "Role": "arn:aws:iam::176354371281:role/my_role",  
    "ServerId": "s-01234567890abcdef",  
    "Tags": [  
        {  
            "Key": "Group",  
            "Value": "UserGroup1"  
        }  
    ],  
}
```

```
    "UserName": "my_user"  
}
```

Example

This example illustrates one usage of CreateUser.

Sample Response

```
{  
    "ServerId": "s-01234567890abcdef",  
    "UserName": "my_user"  
}
```

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

CreateWorkflow

Allows you to create a workflow with specified steps and step details the workflow invokes after file transfer completes. After creating a workflow, you can associate the workflow created with any transfer servers by specifying the `workflow-details` field in `CreateServer` and `UpdateServer` operations.

Request Syntax

```
{
    "Description": "string",
    "OnExceptionSteps": [
        {
            "CopyStepDetails": {
                "DestinationFileLocation": {
                    "EfsFileLocation": {
                        "FileSystemId": "string",
                        "Path": "string"
                    },
                    "S3FileLocation": {
                        "Bucket": "string",
                        "Key": "string"
                    }
                },
                "Name": "string",
                "OverwriteExisting": "string"
            },
            "CustomStepDetails": {
                "Name": "string",
                "Target": "string",
                "TimeoutSeconds": number
            },
            "DeleteStepDetails": {
                "Name": "string"
            },
            "TagStepDetails": {
                "Name": "string",
                "Tags": [
                    {
                        "Key": "string",
                        "Value": "string"
                    }
                ]
            },
            "Type": "string"
        }
    ],
    "Steps": [
        {
            "CopyStepDetails": {
                "DestinationFileLocation": {
                    "EfsFileLocation": {
                        "FileSystemId": "string",
                        "Path": "string"
                    },
                    "S3FileLocation": {
                        "Bucket": "string",
                        "Key": "string"
                    }
                },
                "Name": "string",
                "OverwriteExisting": "string"
            },
            "CustomStepDetails": {
```

```
        "Name": "string",
        "Target": "string",
        "TimeoutSeconds": number
    },
    "DeleteStepDetails": {
        "Name": "string"
    },
    "TagStepDetails": {
        "Name": "string",
        "Tags": [
            {
                "Key": "string",
                "Value": "string"
            }
        ],
        "Type": "string"
    }
],
"Tags": [
    {
        "Key": "string",
        "Value": "string"
    }
]
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 329\)](#).

The request accepts the following data in JSON format.

Description (p. 179)

A textual description for the workflow.

Type: String

Length Constraints: Maximum length of 256.

Pattern: ^[\w-]*\$

Required: No

OnExceptionSteps (p. 179)

Specifies the steps (actions) to take if errors are encountered during execution of the workflow.

Note

For custom steps, the lambda function needs to send FAILURE to the call back API to kick off the exception steps. Additionally, if the lambda does not send SUCCESS before it times out, the exception steps are executed.

Type: Array of [WorkflowStep \(p. 328\)](#) objects

Array Members: Maximum number of 8 items.

Required: No

Steps (p. 179)

Specifies the details for the steps that are in the specified workflow.

The `TYPE` specifies which of the following actions is being taken for this step.

- *Copy*: copy the file to another location
- *Custom*: custom step with a lambda target
- *Delete*: delete the file
- *Tag*: add a tag to the file

Note

Currently, copying and tagging are supported only on S3.

For file location, you specify either the S3 bucket and key, or the EFS filesystem ID and path.

Type: Array of [WorkflowStep \(p. 328\)](#) objects

Array Members: Maximum number of 8 items.

Required: Yes

Tags (p. 179)

Key-value pairs that can be used to group and search for workflows. Tags are metadata attached to workflows for any purpose.

Type: Array of [Tag \(p. 323\)](#) objects

Array Members: Minimum number of 1 item. Maximum number of 50 items.

Required: No

Response Syntax

```
{  
  "WorkflowId": "string"  
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

WorkflowId (p. 181)

A unique identifier for the workflow.

Type: String

Length Constraints: Fixed length of 19.

Pattern: ^w-([a-zA-Z0-9]{17})\$

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 330\)](#).

AccessDeniedException

You do not have sufficient access to perform this action.

HTTP Status Code: 400

InternalServiceError

This exception is thrown when an error occurs in the AWSTransfer Family service.

HTTP Status Code: 500

InvalidRequestException

This exception is thrown when the client submits a malformed request.

HTTP Status Code: 400

ResourceExistsException

The requested resource does not exist.

HTTP Status Code: 400

ServiceUnavailableException

The request has failed because the AWSTransfer Family service is not available.

HTTP Status Code: 500

ThrottlingException

The request was denied due to request throttling.

HTTP Status Code: 400

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

DeleteAccess

Allows you to delete the access specified in the `ServerId` and `ExternalId` parameters.

Request Syntax

```
{  
    "ExternalId": "string",  
    "ServerId": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 329\)](#).

The request accepts the following data in JSON format.

[ExternalId \(p. 183\)](#)

A unique identifier that is required to identify specific groups within your directory. The users of the group that you associate have access to your Amazon S3 or Amazon EFS resources over the enabled protocols using AWS Transfer Family. If you know the group name, you can view the SID values by running the following command using Windows PowerShell.

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

In that command, replace `YourGroupName` with the name of your Active Directory group.

The regex used to validate this parameter is a string of characters consisting of uppercase and lowercase alphanumeric characters with no spaces. You can also include underscores or any of the following characters: =,.,@:/-

Type: String

Length Constraints: Minimum length of 1. Maximum length of 256.

Pattern: ^S-1-[\d-]+\$

Required: Yes

[ServerId \(p. 183\)](#)

A system-assigned unique identifier for a server that has this user assigned.

Type: String

Length Constraints: Fixed length of 19.

Pattern: ^s-([0-9a-f]{17})\$

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 330\)](#).

InternalServiceError

This exception is thrown when an error occurs in the AWSTransfer Family service.

HTTP Status Code: 500

InvalidRequestException

This exception is thrown when the client submits a malformed request.

HTTP Status Code: 400

ResourceNotFoundException

This exception is thrown when a resource is not found by the AWSTransfer Family service.

HTTP Status Code: 400

ServiceUnavailableException

The request has failed because the AWSTransfer Family service is not available.

HTTP Status Code: 500

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

DeleteServer

Deletes the file transfer protocol-enabled server that you specify.

No response returns from this operation.

Request Syntax

```
{  
    "ServerId": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 329\)](#).

The request accepts the following data in JSON format.

ServerId (p. 185)

A unique system-assigned identifier for a server instance.

Type: String

Length Constraints: Fixed length of 19.

Pattern: ^s-([0-9a-f]{17})\$

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 330\)](#).

AccessDeniedException

You do not have sufficient access to perform this action.

HTTP Status Code: 400

InternalServiceError

This exception is thrown when an error occurs in the AWSTransfer Family service.

HTTP Status Code: 500

InvalidRequestException

This exception is thrown when the client submits a malformed request.

HTTP Status Code: 400

ResourceNotFoundException

This exception is thrown when a resource is not found by the AWSTransfer Family service.

HTTP Status Code: 400

ServiceUnavailableException

The request has failed because the AWSTransfer Family service is not available.

HTTP Status Code: 500

Examples

Example

The following example deletes a server.

Sample Request

```
{  
    "ServerId": "s-01234567890abcdef"  
}
```

Example

This example illustrates one usage of DeleteServer.

Sample Response

```
{  
}
```

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

DeleteSshPublicKey

Deletes a user's Secure Shell (SSH) public key.

Request Syntax

```
{  
    "ServerId": "string",  
    "SshPublicKeyId": "string",  
    "UserName": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 329\)](#).

The request accepts the following data in JSON format.

[ServerId \(p. 187\)](#)

A system-assigned unique identifier for a file transfer protocol-enabled server instance that has the user assigned to it.

Type: String

Length Constraints: Fixed length of 19.

Pattern: ^s-([0-9a-f]{17})\$

Required: Yes

[SshPublicKeyId \(p. 187\)](#)

A unique identifier used to reference your user's specific SSH key.

Type: String

Length Constraints: Fixed length of 21.

Pattern: ^key-[0-9a-f]{17}\$

Required: Yes

[UserName \(p. 187\)](#)

A unique string that identifies a user whose public key is being deleted.

Type: String

Length Constraints: Minimum length of 3. Maximum length of 100.

Pattern: ^[\w][\w@.-]{2,99}\$

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 330\)](#).

InternalServiceError

This exception is thrown when an error occurs in the AWSTransfer Family service.

HTTP Status Code: 500

InvalidRequestException

This exception is thrown when the client submits a malformed request.

HTTP Status Code: 400

ResourceNotFoundException

This exception is thrown when a resource is not found by the AWSTransfer Family service.

HTTP Status Code: 400

ServiceUnavailableException

The request has failed because the AWSTransfer Family service is not available.

HTTP Status Code: 500

ThrottlingException

The request was denied due to request throttling.

HTTP Status Code: 400

HTTP Status Code: 400

Examples

Example

The following example deletes a user's SSH public key.

Sample Request

```
{  
    "ServerId": "s-01234567890abcdef",  
    "SshPublicKeyId": "MyPublicKey",  
    "UserName": "my_user"  
}
```

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)

- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

DeleteUser

Deletes the user belonging to a file transfer protocol-enabled server you specify.

No response returns from this operation.

Note

When you delete a user from a server, the user's information is lost.

Request Syntax

```
{  
    "ServerId": "string",  
    "UserName": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 329\)](#).

The request accepts the following data in JSON format.

[ServerId \(p. 190\)](#)

A system-assigned unique identifier for a server instance that has the user assigned to it.

Type: String

Length Constraints: Fixed length of 19.

Pattern: ^s-([0-9a-f]{17})\$

Required: Yes

[UserName \(p. 190\)](#)

A unique string that identifies a user that is being deleted from a server.

Type: String

Length Constraints: Minimum length of 3. Maximum length of 100.

Pattern: ^[\w][\w@.-]{2,99}\$

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 330\)](#).

InternalServiceError

This exception is thrown when an error occurs in the AWSTransfer Family service.

HTTP Status Code: 500

InvalidRequestException

This exception is thrown when the client submits a malformed request.

HTTP Status Code: 400

ResourceNotFoundException

This exception is thrown when a resource is not found by the AWSTransfer Family service.

HTTP Status Code: 400

ServiceUnavailableException

The request has failed because the AWSTransfer Family service is not available.

HTTP Status Code: 500

Examples

Example

The following example deletes a user account assigned to a server.

Sample Request

```
{  
    "ServerId": "s-01234567890abcdef",  
    "UserNames": "my_user"  
}
```

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

DeleteWorkflow

Deletes the specified workflow.

Request Syntax

```
{  
    "WorkflowId": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 329\)](#).

The request accepts the following data in JSON format.

WorkflowId (p. 192)

A unique identifier for the workflow.

Type: String

Length Constraints: Fixed length of 19.

Pattern: ^w-([a-zA-Z0-9]{17})\$

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 330\)](#).

AccessDeniedException

You do not have sufficient access to perform this action.

HTTP Status Code: 400

InternalServiceError

This exception is thrown when an error occurs in the AWSTransfer Family service.

HTTP Status Code: 500

InvalidRequestException

This exception is thrown when the client submits a malformed request.

HTTP Status Code: 400

ResourceNotFoundException

This exception is thrown when a resource is not found by the AWSTransfer Family service.

HTTP Status Code: 400

ServiceUnavailableException

The request has failed because the AWSTransfer Family service is not available.

HTTP Status Code: 500

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

DescribeAccess

Describes the access that is assigned to the specific file transfer protocol-enabled server, as identified by its `ServerId` property and its `ExternalId`.

The response from this call returns the properties of the access that is associated with the `ServerId` value that was specified.

Request Syntax

```
{  
    "ExternalId": "string",  
    "ServerId": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 329\)](#).

The request accepts the following data in JSON format.

[ExternalId \(p. 194\)](#)

A unique identifier that is required to identify specific groups within your directory. The users of the group that you associate have access to your Amazon S3 or Amazon EFS resources over the enabled protocols using AWS Transfer Family. If you know the group name, you can view the SID values by running the following command using Windows PowerShell.

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

In that command, replace `YourGroupName` with the name of your Active Directory group.

The regex used to validate this parameter is a string of characters consisting of uppercase and lowercase alphanumeric characters with no spaces. You can also include underscores or any of the following characters: =, @, :, /

Type: String

Length Constraints: Minimum length of 1. Maximum length of 256.

Pattern: ^S-1-[\d-]+\$

Required: Yes

[ServerId \(p. 194\)](#)

A system-assigned unique identifier for a server that has this access assigned.

Type: String

Length Constraints: Fixed length of 19.

Pattern: ^s-([0-9a-f]{17})\$

Required: Yes

Response Syntax

```
{  
    "Access": {  
        "ExternalId": "string",  
        "HomeDirectory": "string",  
        "HomeDirectoryMappings": [  
            {  
                "Entry": "string",  
                "Target": "string"  
            }  
        ],  
        "HomeDirectoryType": "string",  
        "Policy": "string",  
        "PosixProfile": {  
            "Gid": number,  
            "SecondaryGids": [ number ],  
            "Uid": number  
        },  
        "Role": "string"  
    },  
    "ServerId": "string"  
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

[Access \(p. 195\)](#)

The external ID of the server that the access is attached to.

Type: [DescribedAccess \(p. 277\)](#) object

[ServerId \(p. 195\)](#)

A system-assigned unique identifier for a server that has this access assigned.

Type: String

Length Constraints: Fixed length of 19.

Pattern: ^s-([0-9a-f]{17})\$

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 330\)](#).

InternalServiceError

This exception is thrown when an error occurs in the AWSTransfer Family service.

HTTP Status Code: 500

InvalidRequestException

This exception is thrown when the client submits a malformed request.

HTTP Status Code: 400

ResourceNotFoundException

This exception is thrown when a resource is not found by the AWSTransfer Family service.

HTTP Status Code: 400

ServiceUnavailableException

The request has failed because the AWSTransfer Family service is not available.

HTTP Status Code: 500

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

DescribeExecution

You can use `DescribeExecution` to check the details of the execution of the specified workflow.

Request Syntax

```
{  
    "ExecutionId": "string",  
    "WorkflowId": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 329\)](#).

The request accepts the following data in JSON format.

ExecutionId (p. 197)

A unique identifier for the execution of a workflow.

Type: String

Length Constraints: Fixed length of 36.

Pattern: ^[0-9a-fA-F]{8}\-[0-9a-fA-F]{4}\-[0-9a-fA-F]{4}\-[0-9a-fA-F]{4}\-[0-9a-fA-F]{12}\$

Required: Yes

WorkflowId (p. 197)

A unique identifier for the workflow.

Type: String

Length Constraints: Fixed length of 19.

Pattern: ^w-([a-zA-Z0-9]{17})\$

Required: Yes

Response Syntax

```
{  
    "Execution": {  
        "ExecutionId": "string",  
        "ExecutionRole": "string",  
        "InitialFileLocation": {  
            "EfsFileLocation": {  
                "FileSystemId": "string",  
                "Path": "string"  
            },  
            "S3FileLocation": {  
                "Bucket": "string",  
                "Etag": "string",  
                "Key": "string",  
                "VersionId": "string"  
            }  
        }  
    }  
}
```

```

        }
    },
    "LoggingConfiguration": {
        "LoggingRole": "string",
        "LogGroupName": "string"
    },
    "PosixProfile": {
        "Gid": number,
        "SecondaryGids": [ number ],
        "Uid": number
    },
    "Results": {
        "OnExceptionSteps": [
            {
                "Error": {
                    "Message": "string",
                    "Type": "string"
                },
                "Outputs": "string",
                "StepType": "string"
            }
        ],
        "Steps": [
            {
                "Error": {
                    "Message": "string",
                    "Type": "string"
                },
                "Outputs": "string",
                "StepType": "string"
            }
        ]
    },
    "ServiceMetadata": {
        "UserDetails": {
            "ServerId": "string",
            "SessionId": "string",
            "UserName": "string"
        }
    },
    "Status": "string"
},
"WorkflowId": "string"
}

```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

[Execution \(p. 197\)](#)

The structure that contains the details of the workflow's execution.

Type: [DescribedExecution \(p. 280\)](#) object

[WorkflowId \(p. 197\)](#)

A unique identifier for the workflow.

Type: String

Length Constraints: Fixed length of 19.

Pattern: ^w-([a-zA-Z0-9]{17})\$

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 330\)](#).

InternalServiceError

This exception is thrown when an error occurs in the AWSTransfer Family service.

HTTP Status Code: 500

InvalidRequestException

This exception is thrown when the client submits a malformed request.

HTTP Status Code: 400

ResourceNotFoundException

This exception is thrown when a resource is not found by the AWSTransfer Family service.

HTTP Status Code: 400

ServiceUnavailableException

The request has failed because the AWSTransfer Family service is not available.

HTTP Status Code: 500

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

DescribeSecurityPolicy

Describes the security policy that is attached to your file transfer protocol-enabled server. The response contains a description of the security policy's properties. For more information about security policies, see [Working with security policies](#).

Request Syntax

```
{  
    "SecurityPolicyName": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 329\)](#).

The request accepts the following data in JSON format.

SecurityPolicyName (p. 200)

Specifies the name of the security policy that is attached to the server.

Type: String

Length Constraints: Maximum length of 100.

Pattern: TransferSecurityPolicy- . +

Required: Yes

Response Syntax

```
{  
    "SecurityPolicy": {  
        "Fips": boolean,  
        "SecurityPolicyName": "string",  
        "SshCiphers": [ "string" ],  
        "SshKexs": [ "string" ],  
        "SshMacs": [ "string" ],  
        "TlsCiphers": [ "string" ]  
    }  
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

SecurityPolicy (p. 200)

An array containing the properties of the security policy.

Type: [DescribedSecurityPolicy \(p. 282\)](#) object

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 330\)](#).

InternalServiceError

This exception is thrown when an error occurs in the AWSTransfer Family service.

HTTP Status Code: 500

InvalidRequestException

This exception is thrown when the client submits a malformed request.

HTTP Status Code: 400

ResourceNotFoundException

This exception is thrown when a resource is not found by the AWSTransfer Family service.

HTTP Status Code: 400

ServiceUnavailableException

The request has failed because the AWSTransfer Family service is not available.

HTTP Status Code: 500

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

DescribeServer

Describes a file transfer protocol-enabled server that you specify by passing the `ServerId` parameter.

The response contains a description of a server's properties. When you set `EndpointType` to VPC, the response will contain the `EndpointDetails`.

Request Syntax

```
{  
    "ServerId": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 329\)](#).

The request accepts the following data in JSON format.

`ServerId` (p. 202)

A system-assigned unique identifier for a server.

Type: String

Length Constraints: Fixed length of 19.

Pattern: ^s-([0-9a-f]{17})\$

Required: Yes

Response Syntax

```
{  
    "Server": {  
        "Arn": "string",  
        "Certificate": "string",  
        "Domain": "string",  
        "EndpointDetails": {  
            "AddressAllocationIds": [ "string" ],  
            "SecurityGroupIds": [ "string" ],  
            "SubnetIds": [ "string" ],  
            "VpcEndpointId": "string",  
            "VpcId": "string"  
        },  
        "EndpointType": "string",  
        "HostKeyFingerprint": "string",  
        "IdentityProviderDetails": {  
            "DirectoryId": "string",  
            "Function": "string",  
            "InvocationRole": "string",  
            "Url": "string"  
        },  
        "IdentityProviderType": "string",  
        "LoggingRole": "string",  
        "ProtocolDetails": {  
            "PassiveIp": "string",  
            "Port": "string",  
            "Protocol": "string",  
            "SshPort": "string",  
            "Transport": "string"  
        }  
    }  
}
```

```
        "TlsSessionResumptionMode": "string"
    },
    "Protocols": [ "string" ],
    "SecurityPolicyName": "string",
    "ServerId": "string",
    "State": "string",
    "Tags": [
        {
            "Key": "string",
            "Value": "string"
        }
    ],
    "UserCount": number,
    "WorkflowDetails": {
        "OnUpload": [
            {
                "ExecutionRole": "string",
                "WorkflowId": "string"
            }
        ]
    }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

[Server \(p. 202\)](#)

An array containing the properties of a server with the `ServerID` you specified.

Type: [DescribedServer \(p. 284\)](#) object

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 330\)](#).

InternalServiceError

This exception is thrown when an error occurs in the AWSTransfer Family service.

HTTP Status Code: 500

InvalidRequestException

This exception is thrown when the client submits a malformed request.

HTTP Status Code: 400

ResourceNotFoundException

This exception is thrown when a resource is not found by the AWSTransfer Family service.

HTTP Status Code: 400

ServiceUnavailableException

The request has failed because the AWSTransfer Family service is not available.

HTTP Status Code: 500

Examples

Example

The following example returns the properties assigned to a server.

Sample Request

```
{  
    "ServerId": "s-01234567890abcdef"  
}
```

Example

This example illustrates one usage of `DescribeServer`.

Sample Response

```
{  
    "Server": {  
        "Arn": "arn:aws:transfer:us-east-1:176354371281:server/s-01234567890abcdef",  
        "EndpointDetails": {  
            "AddressAllocationIds": [  
                "eipalloc-01a2eabe3c04d5678",  
                "eipalloc-102345be"  
            ],  
            "SubnetIds": [  
                "subnet-047eaa7f0187a7cde",  
                "subnet-0a2d0f474daffde18"  
            ],  
            "VpcEndpointId": "vpce-03fe0080e7cb008b8",  
            "VpcId": "vpc-09047a51f1c8e1634"  
        },  
        "EndpointType": "VPC",  
        "HostKeyFingerprint": "your host key",  
        "IdentityProviderType": "SERVICE_MANAGED",  
        "ServerId": "s-01234567890abcdef",  
        "State": "ONLINE",  
        "Tags": [],  
        "UserCount": 0  
    }  
}
```

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)

- [AWS SDK for Ruby V3](#)

DescribeUser

Describes the user assigned to the specific file transfer protocol-enabled server, as identified by its `ServerId` property.

The response from this call returns the properties of the user associated with the `ServerId` value that was specified.

Request Syntax

```
{  
    "ServerId": "string",  
    "UserName": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 329\)](#).

The request accepts the following data in JSON format.

[ServerId \(p. 206\)](#)

A system-assigned unique identifier for a server that has this user assigned.

Type: String

Length Constraints: Fixed length of 19.

Pattern: ^s-([0-9a-f]{17})\$

Required: Yes

[UserName \(p. 206\)](#)

The name of the user assigned to one or more servers. User names are part of the sign-in credentials to use the AWS Transfer Family service and perform file transfer tasks.

Type: String

Length Constraints: Minimum length of 3. Maximum length of 100.

Pattern: ^[\w][\w@.-]{2,99}\$

Required: Yes

Response Syntax

```
{  
    "ServerId": "string",  
    "User": {  
        "Arn": "string",  
        "HomeDirectory": "string",  
        "HomeDirectoryMappings": [  
            {  
                "Entry": "string",  
                "Target": "string"  
            }  
        ]  
    }  
}
```

```
        }
    ],
    "HomeDirectoryType": "string",
    "Policy": "string",
    "PosixProfile": {
        "Gid": number,
        "SecondaryGids": [ number ],
        "Uid": number
    },
    "Role": "string",
    "SshPublicKeys": [
        {
            "DateImported": number,
            "SshPublicKeyBody": "string",
            "SshPublicKeyId": "string"
        }
    ],
    "Tags": [
        {
            "Key": "string",
            "Value": "string"
        }
    ],
    "UserName": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

[ServerId \(p. 206\)](#)

A system-assigned unique identifier for a server that has this user assigned.

Type: String

Length Constraints: Fixed length of 19.

Pattern: ^s-([0-9a-f]{17})\$

[User \(p. 206\)](#)

An array containing the properties of the user account for the `ServerID` value that you specified.

Type: [DescribedUser \(p. 288\)](#) object

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 330\)](#).

InternalServiceError

This exception is thrown when an error occurs in the AWSTransfer Family service.

HTTP Status Code: 500

InvalidRequestException

This exception is thrown when the client submits a malformed request.

HTTP Status Code: 400

ResourceNotFoundException

This exception is thrown when a resource is not found by the AWSTransfer Family service.

HTTP Status Code: 400

ServiceUnavailableException

The request has failed because the AWSTransfer Family service is not available.

HTTP Status Code: 500

Examples

Example

The following example returns the properties assigned to a server.

Sample Request

```
{  
    "ServerId": "s-01234567890abcdef",  
    "UserName": "my_user"  
}
```

Example

This example illustrates one usage of `DescribeUser`.

Sample Response

```
{  
    "User": {  
        "Arn": "arn:aws:transfer:us-east-1:176354371281:server/s-01234567890abcdef",  
        "HomeDirectory": "/home/mydirectory",  
        "HomeDirectoryType": "PATH",  
        "Role": "arn:aws:iam::176354371281:role/my_role",  
        "SshPublicKeys":  
            "AAAAB3NzaC1yc2EAAAQABAAQOCotfCAis3aHfM6yc8KWAlMQxVDBHyccCde9MdLf4DQNXd8HjAHf  
            +Bc1vGGCAREFUL1NO2PEEKING3ALLOWEDfif  
            +JBecywfO35Cm6IKIV0JF2YOPXvOuQRs80hQaBUvQL9xw6VEb4xzbit2QB6",  
        "Tags": [  
            {  
                "Key": "Name",  
                "Value": "MyServer"  
            }  
            "UserName": "my_user",  
        ]  
    }  
}
```

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)

- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

DescribeWorkflow

Describes the specified workflow.

Request Syntax

```
{  
    "WorkflowId": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 329\)](#).

The request accepts the following data in JSON format.

[WorkflowId \(p. 210\)](#)

A unique identifier for the workflow.

Type: String

Length Constraints: Fixed length of 19.

Pattern: ^w-([a-zA-Z0-9]{17})\$

Required: Yes

Response Syntax

```
{  
    "Workflow": {  
        "Arn": "string",  
        "Description": "string",  
        "OnExceptionSteps": [  
            {  
                "CopyStepDetails": {  
                    "DestinationFileLocation": {  
                        "EfsFileLocation": {  
                            "FileSystemId": "string",  
                            "Path": "string"  
                        },  
                        "S3FileLocation": {  
                            "Bucket": "string",  
                            "Key": "string"  
                        }  
                    },  
                    "Name": "string",  
                    "OverwriteExisting": "string"  
                },  
                "CustomStepDetails": {  
                    "Name": "string",  
                    "Target": "string",  
                    "TimeoutSeconds": number  
                },  
                "DeleteStepDetails": {  
                    "Name": "string"  
                }  
            }  
        ]  
    }  
}
```

```

        },
        "TagStepDetails": {
            "Name": "string",
            "Tags": [
                {
                    "Key": "string",
                    "Value": "string"
                }
            ],
            "Type": "string"
        }
    ],
    "Steps": [
        {
            "CopyStepDetails": {
                "DestinationFileLocation": {
                    "EfsFileLocation": {
                        "FileSystemId": "string",
                        "Path": "string"
                    },
                    "S3FileLocation": {
                        "Bucket": "string",
                        "Key": "string"
                    }
                },
                "Name": "string",
                "OverwriteExisting": "string"
            },
            "CustomStepDetails": {
                "Name": "string",
                "Target": "string",
                "TimeoutSeconds": number
            },
            "DeleteStepDetails": {
                "Name": "string"
            },
            "TagStepDetails": {
                "Name": "string",
                "Tags": [
                    {
                        "Key": "string",
                        "Value": "string"
                    }
                ],
                "Type": "string"
            }
        ],
        "Tags": [
            {
                "Key": "string",
                "Value": "string"
            }
        ],
        "WorkflowId": "string"
    }
}

```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

[Workflow \(p. 210\)](#)

The structure that contains the details of the workflow.

Type: [DescribedWorkflow \(p. 291\)](#) object

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 330\)](#).

InternalServiceError

This exception is thrown when an error occurs in the AWSTransfer Family service.

HTTP Status Code: 500

InvalidRequestException

This exception is thrown when the client submits a malformed request.

HTTP Status Code: 400

ResourceNotFoundException

This exception is thrown when a resource is not found by the AWSTransfer Family service.

HTTP Status Code: 400

ServiceUnavailableException

The request has failed because the AWSTransfer Family service is not available.

HTTP Status Code: 500

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

ImportSshPublicKey

Adds a Secure Shell (SSH) public key to a user account identified by a `UserName` value assigned to the specific file transfer protocol-enabled server, identified by `ServerId`.

The response returns the `UserName` value, the `ServerId` value, and the name of the `SshPublicKeyId`.

Request Syntax

```
{  
    "ServerId": "string",  
    "SshPublicKeyBody": "string",  
    "UserName": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 329\)](#).

The request accepts the following data in JSON format.

[ServerId \(p. 213\)](#)

A system-assigned unique identifier for a server.

Type: String

Length Constraints: Fixed length of 19.

Pattern: ^s-([0-9a-f]{17})\$

Required: Yes

[SshPublicKeyBody \(p. 213\)](#)

The public key portion of an SSH key pair.

Type: String

Length Constraints: Maximum length of 2048.

Pattern: ^ssh-rsa\s+[A-Za-z0-9+/]+[=]{0,3}(\s+.+)?\s*\$

Required: Yes

[UserName \(p. 213\)](#)

The name of the user account that is assigned to one or more servers.

Type: String

Length Constraints: Minimum length of 3. Maximum length of 100.

Pattern: ^[\w][\w@.-]{2,99}\$

Required: Yes

Response Syntax

```
{
```

```
    "ServerId": "string",
    "SshPublicKeyId": "string",
    "UserName": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

[ServerId \(p. 213\)](#)

A system-assigned unique identifier for a server.

Type: String

Length Constraints: Fixed length of 19.

Pattern: ^s-([0-9a-f]{17})\$

[SshPublicKeyId \(p. 213\)](#)

The name given to a public key by the system that was imported.

Type: String

Length Constraints: Fixed length of 21.

Pattern: ^key-[0-9a-f]{17}\$

[UserName \(p. 213\)](#)

A user name assigned to the `ServerID` value that you specified.

Type: String

Length Constraints: Minimum length of 3. Maximum length of 100.

Pattern: ^[\w][\w@.-]{2,99}\$

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 330\)](#).

InternalServiceError

This exception is thrown when an error occurs in the AWSTransfer Family service.

HTTP Status Code: 500

InvalidRequestException

This exception is thrown when the client submits a malformed request.

HTTP Status Code: 400

ResourceExistsException

The requested resource does not exist.

HTTP Status Code: 400

ResourceNotFoundException

This exception is thrown when a resource is not found by the AWSTransfer Family service.

HTTP Status Code: 400

ServiceUnavailableException

The request has failed because the AWSTransfer Family service is not available.

HTTP Status Code: 500

ThrottlingException

The request was denied due to request throttling.

HTTP Status Code: 400

HTTP Status Code: 400

Examples

Example

The following example returns the properties assigned to a server.

Sample Request

```
{  
    "ServerId": "s-01234567890abcdef",  
    "SshPublicKeyBody":  
        "AAAAAB3NzaC1yc2EAAAQABAAQOCotfCAis3aHfM6yc8KWAlMQxVDBHyccCde9MdLf4DQNxn8HjAHf  
        +Bc1vGGCAREFUL1NO2PEEKING3ALLOWEDfif  
        +JBecywfo35Cm6IKIV0JF2YOPXvOuQRs80hQaBUvQL9xw6VEb4xzbit2QB6",  
    "UserName": "my_user"  
}
```

Example

This example illustrates one usage of ImportSshPublicKey.

Sample Response

```
{  
    "User": {  
        "ServerId": "s-01234567890abcdef",  
        "SshPublicKeyId": "MySSHPublicKey",  
        "UserName": "my_user"  
    }  
}
```

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

ListAccesses

Lists the details for all the accesses you have on your server.

Request Syntax

```
{  
    "MaxResults": number,  
    "NextToken": "string",  
    "ServerId": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 329\)](#).

The request accepts the following data in JSON format.

[MaxResults \(p. 217\)](#)

Specifies the maximum number of access SIDs to return.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 1000.

Required: No

[NextToken \(p. 217\)](#)

When you can get additional results from the `ListAccesses` call, a `NextToken` parameter is returned in the output. You can then pass in a subsequent command to the `NextToken` parameter to continue listing additional accesses.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 6144.

Required: No

[ServerId \(p. 217\)](#)

A system-assigned unique identifier for a server that has users assigned to it.

Type: String

Length Constraints: Fixed length of 19.

Pattern: ^s-([0-9a-f]{17})\$

Required: Yes

Response Syntax

```
{  
    "Accesses": [  
        {
```

```
        "ExternalId": "string",
        "HomeDirectory": "string",
        "HomeDirectoryType": "string",
        "Role": "string"
    }
],
"NextToken": "string",
"ServerId": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

[Accesses \(p. 217\)](#)

Returns the accesses and their properties for the `ServerId` value that you specify.

Type: Array of [ListedAccess \(p. 304\)](#) objects

[NextToken \(p. 217\)](#)

When you can get additional results from the `ListAccesses` call, a `NextToken` parameter is returned in the output. You can then pass in a subsequent command to the `NextToken` parameter to continue listing additional accesses.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 6144.

[ServerId \(p. 217\)](#)

A system-assigned unique identifier for a server that has users assigned to it.

Type: String

Length Constraints: Fixed length of 19.

Pattern: ^s-([0-9a-f]{17})\$

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 330\)](#).

InternalServiceError

This exception is thrown when an error occurs in the AWSTransfer Family service.

HTTP Status Code: 500

InvalidNextTokenException

The `NextToken` parameter that was passed is invalid.

HTTP Status Code: 400

InvalidRequestException

This exception is thrown when the client submits a malformed request.

HTTP Status Code: 400

ResourceNotFoundException

This exception is thrown when a resource is not found by the AWSTransfer Family service.

HTTP Status Code: 400

ServiceUnavailableException

The request has failed because the AWSTransfer Family service is not available.

HTTP Status Code: 500

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

ListExecutions

Lists all executions for the specified workflow.

Request Syntax

```
{  
    "MaxResults": number,  
    "NextToken": "string",  
    "WorkflowId": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 329\)](#).

The request accepts the following data in JSON format.

[MaxResults \(p. 220\)](#)

Specifies the maximum number of executions to return.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 1000.

Required: No

[NextToken \(p. 220\)](#)

`ListExecutions` returns the `NextToken` parameter in the output. You can then pass the `NextToken` parameter in a subsequent command to continue listing additional executions.

This is useful for pagination, for instance. If you have 100 executions for a workflow, you might only want to list first 10. If so, call the API by specifying the `max-results`:

```
aws transfer list-executions --max-results 10
```

This returns details for the first 10 executions, as well as the pointer (`NextToken`) to the eleventh execution. You can now call the API again, supplying the `NextToken` value you received:

```
aws transfer list-executions --max-results 10 --next-token  
$somePointerReturnedFromPreviousListResult
```

This call returns the next 10 executions, the 11th through the 20th. You can then repeat the call until the details for all 100 executions have been returned.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 6144.

Required: No

[WorkflowId \(p. 220\)](#)

A unique identifier for the workflow.

Type: String

Length Constraints: Fixed length of 19.

Pattern: ^w-([a-z0-9]{17})\$

Required: Yes

Response Syntax

```
{  
    "Executions": [  
        {  
            "ExecutionId": "string",  
            "InitialFileLocation": {  
                "EfsFileLocation": {  
                    "FileSystemId": "string",  
                    "Path": "string"  
                },  
                "S3FileLocation": {  
                    "Bucket": "string",  
                    "Etag": "string",  
                    "Key": "string",  
                    "VersionId": "string"  
                }  
            },  
            "ServiceMetadata": {  
                "UserDetails": {  
                    "ServerId": "string",  
                    "SessionId": "string",  
                    "UserName": "string"  
                }  
            },  
            "Status": "string"  
        }  
    ],  
    "NextToken": "string",  
    "WorkflowId": "string"  
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

Executions (p. 221)

Returns the details for each execution.

- **NextToken:** returned from a call to several APIs, you can use pass it to a subsequent command to continue listing additional executions.
- **StartTime:** timestamp indicating when the execution began.
- **Executions:** details of the execution, including the execution ID, initial file location, and Service metadata.
- **Status:** one of the following values: IN_PROGRESS, COMPLETED, EXCEPTION, HANDLING_EXCEPTION.

Type: Array of [ListedExecution \(p. 306\)](#) objects

NextToken (p. 221)

`ListExecutions` returns the `NextToken` parameter in the output. You can then pass the `NextToken` parameter in a subsequent command to continue listing additional executions.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 6144.

[WorkflowId \(p. 221\)](#)

A unique identifier for the workflow.

Type: String

Length Constraints: Fixed length of 19.

Pattern: ^w-([a-zA-Z0-9]{17})\$

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 330\)](#).

InternalServiceError

This exception is thrown when an error occurs in the AWSTransfer Family service.

HTTP Status Code: 500

InvalidNextTokenException

The `NextToken` parameter that was passed is invalid.

HTTP Status Code: 400

InvalidRequestException

This exception is thrown when the client submits a malformed request.

HTTP Status Code: 400

ResourceNotFoundException

This exception is thrown when a resource is not found by the AWSTransfer Family service.

HTTP Status Code: 400

ServiceUnavailableException

The request has failed because the AWSTransfer Family service is not available.

HTTP Status Code: 500

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)

- [AWS SDK for Ruby V3](#)

ListSecurityPolicies

Lists the security policies that are attached to your file transfer protocol-enabled servers.

Request Syntax

```
{  
    "MaxResults": number,  
    "NextToken": string  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 329\)](#).

The request accepts the following data in JSON format.

[MaxResults \(p. 224\)](#)

Specifies the number of security policies to return as a response to the `ListSecurityPolicies` query.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 1000.

Required: No

[NextToken \(p. 224\)](#)

When additional results are obtained from the `ListSecurityPolicies` command, a `NextToken` parameter is returned in the output. You can then pass the `NextToken` parameter in a subsequent command to continue listing additional security policies.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 6144.

Required: No

Response Syntax

```
{  
    "NextToken": string,  
    "SecurityPolicyNames": [ string ]  
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

[NextToken \(p. 224\)](#)

When you can get additional results from the `ListSecurityPolicies` operation, a `NextToken` parameter is returned in the output. In a following command, you can pass in the `NextToken` parameter to continue listing security policies.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 6144.

[SecurityPolicyNames \(p. 224\)](#)

An array of security policies that were listed.

Type: Array of strings

Length Constraints: Maximum length of 100.

Pattern: TransferSecurityPolicy- .+

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 330\)](#).

InternalServiceError

This exception is thrown when an error occurs in the AWSTransfer Family service.

HTTP Status Code: 500

InvalidNextTokenException

The `NextToken` parameter that was passed is invalid.

HTTP Status Code: 400

InvalidRequestException

This exception is thrown when the client submits a malformed request.

HTTP Status Code: 400

ServiceUnavailableException

The request has failed because the AWSTransfer Family service is not available.

HTTP Status Code: 500

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

ListServers

Lists the file transfer protocol-enabled servers that are associated with your AWS account.

Request Syntax

```
{  
    "MaxResults": number,  
    "NextToken": string  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 329\)](#).

The request accepts the following data in JSON format.

MaxResults (p. 226)

Specifies the number of servers to return as a response to the `ListServers` query.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 1000.

Required: No

NextToken (p. 226)

When additional results are obtained from the `ListServers` command, a `NextToken` parameter is returned in the output. You can then pass the `NextToken` parameter in a subsequent command to continue listing additional servers.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 6144.

Required: No

Response Syntax

```
{  
    "NextToken": string,  
    "Servers": [  
        {  
            "Arn": string,  
            "Domain": string,  
            "EndpointType": string,  
            "IdentityProviderType": string,  
            "LoggingRole": string,  
            "ServerId": string,  
            "State": string,  
            "UserCount": number  
        }  
    ]  
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

NextToken (p. 226)

When you can get additional results from the `ListServers` operation, a `NextToken` parameter is returned in the output. In a following command, you can pass in the `NextToken` parameter to continue listing additional servers.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 6144.

Servers (p. 226)

An array of servers that were listed.

Type: Array of [ListedServer \(p. 307\)](#) objects

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 330\)](#).

InternalServiceError

This exception is thrown when an error occurs in the AWSTransfer Family service.

HTTP Status Code: 500

InvalidNextTokenException

The `NextToken` parameter that was passed is invalid.

HTTP Status Code: 400

InvalidRequestException

This exception is thrown when the client submits a malformed request.

HTTP Status Code: 400

ServiceUnavailableException

The request has failed because the AWSTransfer Family service is not available.

HTTP Status Code: 500

Examples

Example

The following example lists the servers that exist in your AWS account.

Sample Request

```
{
```

```
        "MaxResults": 100,
        "NextToken": "eyJNYXJrZXIiOiBudWxsLCAiYm90b1X0cnVuU2F0ZV9hbW91bnQiOiAyfQ=="
    }
```

Example

This example illustrates one usage of ListServers.

Sample Response

```
{
    "NextToken": "eyJNYXJrZXIiOiBudWxsLCAiYm90b1X0cnVuU2F0ZV9hbW91bnQiOiAyfQ==",
    "Servers": [
        {
            "Arn": "arn:aws:transfer:us-east-1:176354371281:server/s-01234567890abcdef",
            "LoggingRole": "CloudWatchLoggingRole",
            "ServerId": "s-01234567890abcdef",
            "State": "ONLINE",
            "Tags": [
                {
                    "Key": "Name",
                    "Value": "MyServer"
                }
            ]
        }
    ]
}
```

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

ListTagsForResource

Lists all of the tags associated with the Amazon Resource Name (ARN) that you specify. The resource can be a user, server, or role.

Request Syntax

```
{  
    "Arn": "string",  
    "MaxResults": number,  
    "NextToken": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 329\)](#).

The request accepts the following data in JSON format.

[Arn \(p. 229\)](#)

Requests the tags associated with a particular Amazon Resource Name (ARN). An ARN is an identifier for a specific AWS resource, such as a server, user, or role.

Type: String

Length Constraints: Minimum length of 20. Maximum length of 1600.

Pattern: arn:.*

Required: Yes

[MaxResults \(p. 229\)](#)

Specifies the number of tags to return as a response to the `ListTagsForResource` request.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 1000.

Required: No

[NextToken \(p. 229\)](#)

When you request additional results from the `ListTagsForResource` operation, a `NextToken` parameter is returned in the input. You can then pass in a subsequent command to the `NextToken` parameter to continue listing additional tags.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 6144.

Required: No

Response Syntax

```
{  
    "Arn": "string",
```

```
"NextToken": "string",
"Tags": [
  {
    "Key": "string",
    "Value": "string"
  }
]
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

[Arn \(p. 229\)](#)

The ARN you specified to list the tags of.

Type: String

Length Constraints: Minimum length of 20. Maximum length of 1600.

Pattern: arn:.*

[NextToken \(p. 229\)](#)

When you can get additional results from the `ListTagsForResource` call, a `NextToken` parameter is returned in the output. You can then pass in a subsequent command to the `NextToken` parameter to continue listing additional tags.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 6144.

[Tags \(p. 229\)](#)

Key-value pairs that are assigned to a resource, usually for the purpose of grouping and searching for items. Tags are metadata that you define.

Type: Array of [Tag \(p. 323\)](#) objects

Array Members: Minimum number of 1 item. Maximum number of 50 items.

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 330\)](#).

InternalServiceError

This exception is thrown when an error occurs in the AWSTransfer Family service.

HTTP Status Code: 500

InvalidNextTokenException

The `NextToken` parameter that was passed is invalid.

HTTP Status Code: 400

InvalidRequestException

This exception is thrown when the client submits a malformed request.

HTTP Status Code: 400

ServiceUnavailableException

The request has failed because the AWSTransfer Family service is not available.

HTTP Status Code: 500

Examples

Example

The following example lists the tags for the resource with the ARN you specified.

Sample Request

```
{  
    "Arn": "arn:aws:transfer:us-east-1:176354371281:server/s-01234567890abcdef"  
}
```

Example

This example illustrates one usage of ListTagsForResource.

Sample Response

```
{  
    "Tags": [  
        {  
            "Key": "Name",  
            "Value": "MyServer"  
        }  
    ]  
}
```

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

ListUsers

Lists the users for a file transfer protocol-enabled server that you specify by passing the `ServerId` parameter.

Request Syntax

```
{  
    "MaxResults": number,  
    "NextToken": string,  
    "ServerId": string  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 329\)](#).

The request accepts the following data in JSON format.

[MaxResults \(p. 232\)](#)

Specifies the number of users to return as a response to the `ListUsers` request.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 1000.

Required: No

[NextToken \(p. 232\)](#)

When you can get additional results from the `ListUsers` call, a `NextToken` parameter is returned in the output. You can then pass in a subsequent command to the `NextToken` parameter to continue listing additional users.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 6144.

Required: No

[ServerId \(p. 232\)](#)

A system-assigned unique identifier for a server that has users assigned to it.

Type: String

Length Constraints: Fixed length of 19.

Pattern: ^s-([0-9a-f]{17})\$

Required: Yes

Response Syntax

```
{  
    "NextToken": string,  
    "ServerId": string,
```

```
"Users": [  
  {  
    "Arn": "string",  
    "HomeDirectory": "string",  
    "HomeDirectoryType": "string",  
    "Role": "string",  
    "SshPublicKeyCount": number,  
    "UserName": "string"  
  }  
]  
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

[NextToken \(p. 232\)](#)

When you can get additional results from the `ListUsers` call, a `NextToken` parameter is returned in the output. You can then pass in a subsequent command to the `NextToken` parameter to continue listing additional users.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 6144.

[ServerId \(p. 232\)](#)

A system-assigned unique identifier for a server that the users are assigned to.

Type: String

Length Constraints: Fixed length of 19.

Pattern: ^s-([0-9a-f]{17})\$

[Users \(p. 232\)](#)

Returns the user accounts and their properties for the `ServerId` value that you specify.

Type: Array of [ListedUser \(p. 310\)](#) objects

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 330\)](#).

InternalServiceError

This exception is thrown when an error occurs in the AWSTransfer Family service.

HTTP Status Code: 500

InvalidNextTokenException

The `NextToken` parameter that was passed is invalid.

HTTP Status Code: 400

InvalidRequestException

This exception is thrown when the client submits a malformed request.

HTTP Status Code: 400

ResourceNotFoundException

This exception is thrown when a resource is not found by the AWSTransfer Family service.

HTTP Status Code: 400

ServiceUnavailableException

The request has failed because the AWSTransfer Family service is not available.

HTTP Status Code: 500

Examples

Example

The `ListUsers` API call returns a list of users associated with a server you specify.

Sample Request

```
{  
    "MaxResults": 100,  
    "NextToken": "eyJNYXJrZXIiOiBudWxsLCAiYm90b1X0cnVuU2F0ZV9hbW91bnQiOiAyfQ==",  
    "ServerId": "s-01234567890abcdef"  
}
```

Example

This example illustrates one usage of `ListUsers`.

Sample Response

```
{  
    "NextToken": "eyJNYXJrZXIiOiBudWxsLCAiYm90b1X0cnVuU2F0ZV9hbW91bnQiOiAyfQ==",  
    "ServerId": "s-01234567890abcdef",  
    "Users": [  
        {  
            "Arn": "arn:aws:transfer:us-east-1:176354371281:user/s-01234567890abcdef/charlie",  
            "HomeDirectory": "/tests/home/charlie",  
            "SshPublicKeyCount": 1,  
            "Role": "arn:aws:iam::176354371281:role/transfer-role1",  
            "Tags": [  
                {  
                    "Key": "Name",  
                    "Value": "user1"  
                }  
            ],  
            "UserName": "my_user"  
        }  
    ]  
}
```

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

ListWorkflows

Lists all of your workflows.

Request Syntax

```
{  
    "MaxResults": number,  
    "NextToken": string  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 329\)](#).

The request accepts the following data in JSON format.

[MaxResults \(p. 236\)](#)

Specifies the maximum number of workflows to return.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 1000.

Required: No

[NextToken \(p. 236\)](#)

`ListWorkflows` returns the `NextToken` parameter in the output. You can then pass the `NextToken` parameter in a subsequent command to continue listing additional workflows.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 6144.

Required: No

Response Syntax

```
{  
    "NextToken": string,  
    "Workflows": [  
        {  
            "Arn": string,  
            "Description": string,  
            "WorkflowId": string  
        }  
    ]  
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

[NextToken \(p. 236\)](#)

`ListWorkflows` returns the `NextToken` parameter in the output. You can then pass the `NextToken` parameter in a subsequent command to continue listing additional workflows.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 6144.

[Workflows \(p. 236\)](#)

Returns the `Arn`, `WorkflowId`, and `Description` for each workflow.

Type: Array of [ListedWorkflow \(p. 312\)](#) objects

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 330\)](#).

InternalServiceError

This exception is thrown when an error occurs in the AWSTransfer Family service.

HTTP Status Code: 500

InvalidNextTokenException

The `NextToken` parameter that was passed is invalid.

HTTP Status Code: 400

InvalidRequestException

This exception is thrown when the client submits a malformed request.

HTTP Status Code: 400

ServiceUnavailableException

The request has failed because the AWSTransfer Family service is not available.

HTTP Status Code: 500

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

SendWorkflowStepState

Sends a callback for asynchronous custom steps.

The `ExecutionId`, `WorkflowId`, and `Token` are passed to the target resource during execution of a custom step of a workflow. You must include those with their callback as well as providing a status.

Request Syntax

```
{  
    "ExecutionId": "string",  
    "Status": "string",  
    "Token": "string",  
    "WorkflowId": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 329\)](#).

The request accepts the following data in JSON format.

ExecutionId (p. 238)

A unique identifier for the execution of a workflow.

Type: String

Length Constraints: Fixed length of 36.

Pattern: ^[0-9a-fA-F]{8}\-[0-9a-fA-F]{4}\-[0-9a-fA-F]{4}\-[0-9a-fA-F]{4}\-[0-9a-fA-F]{12}\$

Required: Yes

Status (p. 238)

Indicates whether the specified step succeeded or failed.

Type: String

Valid Values: `SUCCESS` | `FAILURE`

Required: Yes

Token (p. 238)

Used to distinguish between multiple callbacks for multiple Lambda steps within the same execution.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 64.

Pattern: \w+

Required: Yes

WorkflowId (p. 238)

A unique identifier for the workflow.

Type: String

Length Constraints: Fixed length of 19.

Pattern: ^w-([a-zA-Z0-9]{17})\$

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 330\)](#).

AccessDeniedException

You do not have sufficient access to perform this action.

HTTP Status Code: 400

InternalServiceError

This exception is thrown when an error occurs in the AWSTransfer Family service.

HTTP Status Code: 500

InvalidRequestException

This exception is thrown when the client submits a malformed request.

HTTP Status Code: 400

ResourceNotFoundException

This exception is thrown when a resource is not found by the AWSTransfer Family service.

HTTP Status Code: 400

ServiceUnavailableException

The request has failed because the AWSTransfer Family service is not available.

HTTP Status Code: 500

ThrottlingException

The request was denied due to request throttling.

HTTP Status Code: 400

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)

- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

StartServer

Changes the state of a file transfer protocol-enabled server from `OFFLINE` to `ONLINE`. It has no impact on a server that is already `ONLINE`. An `ONLINE` server can accept and process file transfer jobs.

The state of `STARTING` indicates that the server is in an intermediate state, either not fully able to respond, or not fully online. The values of `START_FAILED` can indicate an error condition.

No response is returned from this call.

Request Syntax

```
{  
    "ServerId": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 329\)](#).

The request accepts the following data in JSON format.

ServerId (p. 241)

A system-assigned unique identifier for a server that you start.

Type: String

Length Constraints: Fixed length of 19.

Pattern: `^s-[0-9a-f]{17}$`

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 330\)](#).

InternalServiceError

This exception is thrown when an error occurs in the AWSTransfer Family service.

HTTP Status Code: 500

InvalidRequestException

This exception is thrown when the client submits a malformed request.

HTTP Status Code: 400

ResourceNotFoundException

This exception is thrown when a resource is not found by the AWSTransfer Family service.

HTTP Status Code: 400

ServiceUnavailableException

The request has failed because the AWSTransfer Family service is not available.

HTTP Status Code: 500

ThrottlingException

The request was denied due to request throttling.

HTTP Status Code: 400

HTTP Status Code: 400

Examples

Example

The following example starts a server.

Sample Request

```
{  
    "ServerId": "s-01234567890abcdef"  
}
```

Example

This example illustrates one usage of StartServer.

Sample Response

```
{  
    "ServerId": "s-01234567890abcdef"  
}
```

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

StopServer

Changes the state of a file transfer protocol-enabled server from `ONLINE` to `OFFLINE`. An `OFFLINE` server cannot accept and process file transfer jobs. Information tied to your server, such as server and user properties, are not affected by stopping your server.

Note

Stopping the server will not reduce or impact your file transfer protocol endpoint billing; you must delete the server to stop being billed.

The state of `STOPPING` indicates that the server is in an intermediate state, either not fully able to respond, or not fully offline. The values of `STOP_FAILED` can indicate an error condition.

No response is returned from this call.

Request Syntax

```
{  
    "ServerId": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 329\)](#).

The request accepts the following data in JSON format.

ServerId (p. 243)

A system-assigned unique identifier for a server that you stopped.

Type: String

Length Constraints: Fixed length of 19.

Pattern: `^s-([0-9a-f]{17})$`

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 330\)](#).

InternalServiceError

This exception is thrown when an error occurs in the AWSTransfer Family service.

HTTP Status Code: 500

InvalidRequestException

This exception is thrown when the client submits a malformed request.

HTTP Status Code: 400

ResourceNotFoundException

This exception is thrown when a resource is not found by the AWSTransfer Family service.

HTTP Status Code: 400

ServiceUnavailableException

The request has failed because the AWSTransfer Family service is not available.

HTTP Status Code: 500

ThrottlingException

The request was denied due to request throttling.

HTTP Status Code: 400

HTTP Status Code: 400

Examples

Example

The following example stops a server.

Sample Request

```
{  
    "ServerId": "s-01234567890abcdef"  
}
```

Example

This example illustrates one usage of StopServer.

Sample Response

```
{  
    "ServerId": "s-01234567890abcdef"  
}
```

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)

- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

TagResource

Attaches a key-value pair to a resource, as identified by its Amazon Resource Name (ARN). Resources are users, servers, roles, and other entities.

There is no response returned from this call.

Request Syntax

```
{  
    "Arn": "string",  
    "Tags": [  
        {  
            "Key": "string",  
            "Value": "string"  
        }  
    ]  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 329\)](#).

The request accepts the following data in JSON format.

[Arn \(p. 246\)](#)

An Amazon Resource Name (ARN) for a specific AWS resource, such as a server, user, or role.

Type: String

Length Constraints: Minimum length of 20. Maximum length of 1600.

Pattern: arn: .*

Required: Yes

[Tags \(p. 246\)](#)

Key-value pairs assigned to ARNs that you can use to group and search for resources by type. You can attach this metadata to user accounts for any purpose.

Type: Array of [Tag \(p. 323\)](#) objects

Array Members: Minimum number of 1 item. Maximum number of 50 items.

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 330\)](#).

InternalServiceError

This exception is thrown when an error occurs in the AWSTransfer Family service.

HTTP Status Code: 500

InvalidRequestException

This exception is thrown when the client submits a malformed request.

HTTP Status Code: 400

ResourceNotFoundException

This exception is thrown when a resource is not found by the AWSTransfer Family service.

HTTP Status Code: 400

ServiceUnavailableException

The request has failed because the AWSTransfer Family service is not available.

HTTP Status Code: 500

Examples

Example

The following example adds a tag to a file transfer protocol-enabled server.

Sample Request

```
{  
    "Arn": "arn:aws:transfer:us-east-1:176354371281:server/s-01234567890abcdef",  
    "Tags": [  
        {  
            "Key": "Group",  
            "Value": "Europe"  
        }  
    ]  
}
```

Example

This example illustrates one usage of TagResource.

Sample Response

```
HTTP 200 response with an empty HTTP body.
```

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

TestIdentityProvider

If the `IdentityProviderType` of a file transfer protocol-enabled server is `AWS_DIRECTORY_SERVICE` or `API_Gateway`, tests whether your identity provider is set up successfully. We highly recommend that you call this operation to test your authentication method as soon as you create your server. By doing so, you can troubleshoot issues with the identity provider integration to ensure that your users can successfully use the service.

The `ServerId` and `UserName` parameters are required. The `ServerProtocol`, `SourceIp`, and `UserPassword` are all optional.

Note

You cannot use `TestIdentityProvider` if the `IdentityProviderType` of your server is `SERVICE_MANAGED`.

- If you provide any incorrect values for any parameters, the `Response` field is empty.
- If you provide a server ID for a server that uses service-managed users, you get an error:

An error occurred (`InvalidRequestException`) when calling the `TestIdentityProvider` operation: `s-server-ID` not configured for external auth

- If you enter a Server ID for the `--server-id` parameter that does not identify an actual Transfer server, you receive the following error:

An error occurred (`ResourceNotFoundException`) when calling the `TestIdentityProvider` operation: Unknown server

Request Syntax

```
{  
    "ServerId": "string",  
    "ServerProtocol": "string",  
    "SourceIp": "string",  
    "UserName": "string",  
    "UserPassword": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 329\)](#).

The request accepts the following data in JSON format.

[ServerId \(p. 249\)](#)

A system-assigned identifier for a specific server. That server's user authentication method is tested with a user name and password.

Type: String

Length Constraints: Fixed length of 19.

Pattern: `^s-([0-9a-f]{17})$`

Required: Yes

[ServerProtocol \(p. 249\)](#)

The type of file transfer protocol to be tested.

The available protocols are:

- Secure Shell (SSH) File Transfer Protocol (SFTP)
- File Transfer Protocol Secure (FTPS)
- File Transfer Protocol (FTP)

Type: String

Valid Values: SFTP | FTP | FTPS

Required: No

[SourceIp \(p. 249\)](#)

The source IP address of the user account to be tested.

Type: String

Length Constraints: Maximum length of 32.

Pattern: ^\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}\$

Required: No

[UserName \(p. 249\)](#)

The name of the user account to be tested.

Type: String

Length Constraints: Minimum length of 3. Maximum length of 100.

Pattern: ^[\w][\w@.-]{2,99}\$

Required: Yes

[UserPassword \(p. 249\)](#)

The password of the user account to be tested.

Type: String

Length Constraints: Maximum length of 1024.

Required: No

Response Syntax

```
{  
    "Message": "string",  
    "Response": "string",  
    "StatusCode": number,  
    "Url": "string"  
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

[Message \(p. 250\)](#)

A message that indicates whether the test was successful or not.

Note

If an empty string is returned, the most likely cause is that the authentication failed due to an incorrect username or password.

Type: String

[Response \(p. 250\)](#)

The response that is returned from your API Gateway.

Type: String

[StatusCode \(p. 250\)](#)

The HTTP status code that is the response from your API Gateway.

Type: Integer

[Url \(p. 250\)](#)

The endpoint of the service used to authenticate a user.

Type: String

Length Constraints: Maximum length of 255.

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 330\)](#).

InternalServiceError

This exception is thrown when an error occurs in the AWSTransfer Family service.

HTTP Status Code: 500

InvalidRequestException

This exception is thrown when the client submits a malformed request.

HTTP Status Code: 400

ResourceNotFoundException

This exception is thrown when a resource is not found by the AWSTransfer Family service.

HTTP Status Code: 400

ServiceUnavailableException

The request has failed because the AWSTransfer Family service is not available.

HTTP Status Code: 500

Examples

[Example](#)

The following request returns a message from an identity provider that a user name and password combination is a valid identity to use with AWS Transfer Family.

Sample Request

```
{  
    "ServerID": "s-01234567890abcdef",  
    "UserName": "my_user",  
    "UserPassword": "MyPassword-1"  
}
```

Example

The following response shows a sample response for a successful test.

Sample Response

```
"Response":  
    {"homeDirectory": "/mybucket001", "homeDirectoryDetails": null, "homeDirectoryType":  
    "PATH", "posixProfile": null,  
    "publicKeys": "[ssh-rsa-key]", "role": "arn:aws:iam::123456789012:role/my_role",  
    "policy": null, "username": "transferuser002",  
    "identityProviderType": null, "userConfigMessage": null}  
    "StatusCode": "200",  
    "Message": ""
```

Example

The following response indicates that the specified user belongs to more than one group that has access.

```
"Response": "",  
"StatusCode": 200,  
"Message": "More than one associated access found for user's groups."
```

Example

If you have created and configured a custom identity provider by using an API Gateway, you can enter the following command to test your user:

```
aws transfer test-identity-provider --server-id s-0123456789abcdefg --user-name myuser
```

where *s-0123456789abcdefg* is your transfer server, and *myuser* is the username for your custom user.

If the command succeeds, your response is similar to the following, where:

- AWS account ID is *012345678901*
- User role is *user-role-api-gateway*
- Home directory is *myuser-bucket*
- Public key is *public-key*
- Invocation URL is *invocation-URL*

```
{
```

```
    "Response": "{\"Role\": \"arn:aws:iam::012345678901:role/user-role-api-gateway\",  
\"HomeDirectory\": \"/myuser-bucket\", \"PublicKeys\": \"[public-key]\"}",  
    "StatusCode": 200,  
    "Message": "",  
    "Url": "https://invocation-URL/servers/s-0123456789abcdefg/users/myuser/config"  
}
```

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

UntagResource

Detaches a key-value pair from a resource, as identified by its Amazon Resource Name (ARN). Resources are users, servers, roles, and other entities.

No response is returned from this call.

Request Syntax

```
{  
    "Arn": "string",  
    "TagKeys": [ "string" ]  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 329\)](#).

The request accepts the following data in JSON format.

[Arn \(p. 254\)](#)

The value of the resource that will have the tag removed. An Amazon Resource Name (ARN) is an identifier for a specific AWS resource, such as a server, user, or role.

Type: String

Length Constraints: Minimum length of 20. Maximum length of 1600.

Pattern: `arn: .*`

Required: Yes

[TagKeys \(p. 254\)](#)

TagKeys are key-value pairs assigned to ARNs that can be used to group and search for resources by type. This metadata can be attached to resources for any purpose.

Type: Array of strings

Array Members: Minimum number of 1 item. Maximum number of 50 items.

Length Constraints: Maximum length of 128.

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 330\)](#).

InternalServiceError

This exception is thrown when an error occurs in the AWSTransfer Family service.

HTTP Status Code: 500

InvalidRequestException

This exception is thrown when the client submits a malformed request.

HTTP Status Code: 400

ResourceNotFoundException

This exception is thrown when a resource is not found by the AWSTransfer Family service.

HTTP Status Code: 400

ServiceUnavailableException

The request has failed because the AWSTransfer Family service is not available.

HTTP Status Code: 500

Examples

Example

The following example removes a tag of a file transfer protocol-enabled server.

Sample Request

```
{  
    "Arn": "arn:aws:transfer:us-east-1:176354371281:server/s-01234567890abcdef",  
    "TagKeys": "Europe" }  
}
```

Example

This example illustrates one usage of UntagResource.

Sample Response

```
HTTP 200 response with an empty HTTP body.
```

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

UpdateAccess

Allows you to update parameters for the access specified in the `ServerId` and `ExternalId` parameters.

Request Syntax

```
{  
    "ExternalId": "string",  
    "HomeDirectory": "string",  
    "HomeDirectoryMappings": [  
        {  
            "Entry": "string",  
            "Target": "string"  
        }  
    ],  
    "HomeDirectoryType": "string",  
    "Policy": "string",  
    "PosixProfile": {  
        "Gid": number,  
        "SecondaryGids": [ number ],  
        "Uid": number  
    },  
    "Role": "string",  
    "ServerId": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 329\)](#).

The request accepts the following data in JSON format.

ExternalId (p. 257)

A unique identifier that is required to identify specific groups within your directory. The users of the group that you associate have access to your Amazon S3 or Amazon EFS resources over the enabled protocols using AWS Transfer Family. If you know the group name, you can view the SID values by running the following command using Windows PowerShell.

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

In that command, replace `YourGroupName` with the name of your Active Directory group.

The regex used to validate this parameter is a string of characters consisting of uppercase and lowercase alphanumeric characters with no spaces. You can also include underscores or any of the following characters: =,.@:/-

Type: String

Length Constraints: Minimum length of 1. Maximum length of 256.

Pattern: ^S-1-[\d-]+\$

Required: Yes

HomeDirectory (p. 257)

The landing directory (folder) for a user when they log in to the server using the client.

A `HomeDirectory` example is `/bucket_name/home/mydirectory`.

Type: String

Length Constraints: Maximum length of 1024.

Pattern: `^$ | /.*`

Required: No

[HomeDirectoryMappings \(p. 257\)](#)

Logical directory mappings that specify what Amazon S3 or Amazon EFS paths and keys should be visible to your user and how you want to make them visible. You must specify the `Entry` and `Target` pair, where `Entry` shows how the path is made visible and `Target` is the actual Amazon S3 or Amazon EFS path. If you only specify a target, it is displayed as is. You also must ensure that your AWS Identity and Access Management (IAM) role provides access to paths in `Target`. This value can only be set when `HomeDirectoryType` is set to `LOGICAL`.

The following is an `Entry` and `Target` pair example.

```
[ { "Entry": "/directory1", "Target": "/bucket_name/home/mydirectory" } ]
```

In most cases, you can use this value instead of the session policy to lock down your user to the designated home directory ("chroot"). To do this, you can set `Entry` to `/` and set `Target` to the `HomeDirectory` parameter value.

The following is an `Entry` and `Target` pair example for chroot.

```
[ { "Entry": "/", "Target": "/bucket_name/home/mydirectory" } ]
```

Type: Array of [HomeDirectoryMapEntry \(p. 300\)](#) objects

Array Members: Minimum number of 1 item. Maximum number of 50 items.

Required: No

[HomeDirectoryType \(p. 257\)](#)

The type of landing directory (folder) you want your users' home directory to be when they log into the server. If you set it to `PATH`, the user will see the absolute Amazon S3 bucket or EFS paths as is in their file transfer protocol clients. If you set it `LOGICAL`, you need to provide mappings in the `HomeDirectoryMappings` for how you want to make Amazon S3 or EFS paths visible to your users.

Type: String

Valid Values: `PATH` | `LOGICAL`

Required: No

[Policy \(p. 257\)](#)

A session policy for your user so that you can use the same IAM role across multiple users. This policy scopes down user access to portions of their Amazon S3 bucket. Variables that you can use inside this policy include `${Transfer:UserName}`, `${Transfer:HomeDirectory}`, and `${Transfer:HomeBucket}`.

Note

This only applies when the domain of `ServerId` is `S3`. EFS does not use session policies. For session policies, AWS Transfer Family stores the policy as a JSON blob, instead of the Amazon Resource Name (ARN) of the policy. You save the policy as a JSON blob and pass it in the `Policy` argument.

For an example of a session policy, see [Example session policy](#).

For more information, see [AssumeRole](#) in the *AWS Security Token Service API Reference*.

Type: String

Length Constraints: Maximum length of 2048.

Required: No

[PosixProfile \(p. 257\)](#)

The full POSIX identity, including user ID (`Uid`), group ID (`Gid`), and any secondary groups IDs (`SecondaryGids`), that controls your users' access to your Amazon EFS file systems. The POSIX permissions that are set on files and directories in your file system determine the level of access your users get when transferring files into and out of your Amazon EFS file systems.

Type: [PosixProfile \(p. 314\)](#) object

Required: No

[Role \(p. 257\)](#)

Specifies the Amazon Resource Name (ARN) of the IAM role that controls your users' access to your Amazon S3 bucket or EFS file system. The policies attached to this role determine the level of access that you want to provide your users when transferring files into and out of your Amazon S3 bucket or EFS file system. The IAM role should also contain a trust relationship that allows the server to access your resources when servicing your users' transfer requests.

Type: String

Length Constraints: Minimum length of 20. Maximum length of 2048.

Pattern: `arn:.*role/.*`

Required: No

[ServerId \(p. 257\)](#)

A system-assigned unique identifier for a server instance. This is the specific server that you added your user to.

Type: String

Length Constraints: Fixed length of 19.

Pattern: `^s-([0-9a-f]{17})$`

Required: Yes

Response Syntax

```
{  
    "ExternalId": "string",  
    "ServerId": "string"  
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

[ExternalId \(p. 259\)](#)

The external ID of the group whose users have access to your Amazon S3 or Amazon EFS resources over the enabled protocols using AWSTransfer Family.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 256.

Pattern: ^S-1-[\d-]+\$

[ServerId \(p. 259\)](#)

The ID of the server that the user is attached to.

Type: String

Length Constraints: Fixed length of 19.

Pattern: ^s-([0-9a-f]{17})\$

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 330\)](#).

InternalServiceError

This exception is thrown when an error occurs in the AWSTransfer Family service.

HTTP Status Code: 500

InvalidRequestException

This exception is thrown when the client submits a malformed request.

HTTP Status Code: 400

ResourceExistsException

The requested resource does not exist.

HTTP Status Code: 400

ResourceNotFoundException

This exception is thrown when a resource is not found by the AWSTransfer Family service.

HTTP Status Code: 400

ServiceUnavailableException

The request has failed because the AWSTransfer Family service is not available.

HTTP Status Code: 500

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)

- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

UpdateServer

Updates the file transfer protocol-enabled server's properties after that server has been created.

The `UpdateServer` call returns the `ServerId` of the server you updated.

Request Syntax

```
{  
    "Certificate": "string",  
    "EndpointDetails": {  
        "AddressAllocationIds": [ "string" ],  
        "SecurityGroupIds": [ "string" ],  
        "SubnetIds": [ "string" ],  
        "VpcEndpointId": "string",  
        "VpcId": "string"  
    },  
    "EndpointType": "string",  
    "HostKey": "string",  
    "IdentityProviderDetails": {  
        "DirectoryId": "string",  
        "Function": "string",  
        "InvocationRole": "string",  
        "Url": "string"  
    },  
    "LoggingRole": "string",  
    "ProtocolDetails": {  
        "PassiveIp": "string",  
        "TlsSessionResumptionMode": "string"  
    },  
    "Protocols": [ "string" ],  
    "SecurityPolicyName": "string",  
    "ServerId": "string",  
    "WorkflowDetails": {  
        "OnUpload": [  
            {  
                "ExecutionRole": "string",  
                "WorkflowId": "string"  
            }  
        ]  
    }  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 329\)](#).

The request accepts the following data in JSON format.

[Certificate \(p. 262\)](#)

The Amazon Resource Name (ARN) of the AWS Certificate Manager (ACM) certificate. Required when `Protocols` is set to `FTPS`.

To request a new public certificate, see [Request a public certificate](#) in the *AWS Certificate Manager User Guide*.

To import an existing certificate into ACM, see [Importing certificates into ACM](#) in the *AWS Certificate Manager User Guide*.

To request a private certificate to use FTPS through private IP addresses, see [Request a private certificate](#) in the *AWS Certificate Manager User Guide*.

Certificates with the following cryptographic algorithms and key sizes are supported:

- 2048-bit RSA (RSA_2048)
- 4096-bit RSA (RSA_4096)
- Elliptic Prime Curve 256 bit (EC_prime256v1)
- Elliptic Prime Curve 384 bit (EC_secp384r1)
- Elliptic Prime Curve 521 bit (EC_secp521r1)

Note

The certificate must be a valid SSL/TLS X.509 version 3 certificate with FQDN or IP address specified and information about the issuer.

Type: String

Length Constraints: Maximum length of 1600.

Required: No

EndpointDetails (p. 262)

The virtual private cloud (VPC) endpoint settings that are configured for your server. When you host your endpoint within your VPC, you can make it accessible only to resources within your VPC, or you can attach Elastic IP addresses and make it accessible to clients over the internet. Your VPC's default security groups are automatically assigned to your endpoint.

Type: [EndpointDetails \(p. 294\)](#) object

Required: No

EndpointType (p. 262)

The type of endpoint that you want your server to use. You can choose to make your server's endpoint publicly accessible (PUBLIC) or host it inside your VPC. With an endpoint that is hosted in a VPC, you can restrict access to your server and resources only within your VPC or choose to make it internet facing by attaching Elastic IP addresses directly to it.

Note

After May 19, 2021, you won't be able to create a server using `EndpointType=VPC_ENDPOINT` in your AWSaccount if your account hasn't already done so before May 19, 2021. If you have already created servers with `EndpointType=VPC_ENDPOINT` in your AWSaccount on or before May 19, 2021, you will not be affected. After this date, use `EndpointType=VPC`.

For more information, see [Discontinuing the use of VPC_ENDPOINT \(p. 42\)](#).

It is recommended that you use VPC as the `EndpointType`. With this endpoint type, you have the option to directly associate up to three Elastic IPv4 addresses (BYO IP included) with your server's endpoint and use VPC security groups to restrict traffic by the client's public IP address. This is not possible with `EndpointType` set to `VPC_ENDPOINT`.

Type: String

Valid Values: PUBLIC | VPC | VPC_ENDPOINT

Required: No

HostKey (p. 262)

The RSA private key as generated by `ssh-keygen -N "" -m PEM -f my-new-server-key`.

Important

If you aren't planning to migrate existing users from an existing server to a new server, don't update the host key. Accidentally changing a server's host key can be disruptive.

For more information, see [Change the host key for your SFTP-enabled server](#) in the *AWS Transfer Family User Guide*.

Type: String

Length Constraints: Maximum length of 4096.

Required: No

[IdentityProviderDetails \(p. 262\)](#)

An array containing all of the information required to call a customer's authentication API method.

Type: [IdentityProviderDetails \(p. 301\)](#) object

Required: No

[LoggingRole \(p. 262\)](#)

Specifies the Amazon Resource Name (ARN) of the AWS Identity and Access Management (IAM) role that allows a server to turn on Amazon CloudWatch logging for Amazon S3 or Amazon EFS events. When set, user activity can be viewed in your CloudWatch logs.

Type: String

Length Constraints: Maximum length of 2048.

Pattern: ^\$ | arn:.*role/.*

Required: No

[ProtocolDetails \(p. 262\)](#)

The protocol settings that are configured for your server.

Use the `PassiveIp` parameter to indicate passive mode (for FTP and FTPS protocols). Enter a single dotted-quad IPv4 address, such as the external IP address of a firewall, router, or load balancer.

Use the `TlsSessionResumptionMode` parameter to determine whether or not your Transfer server resumes recent, negotiated sessions through a unique session ID.

Type: [ProtocolDetails \(p. 315\)](#) object

Required: No

[Protocols \(p. 262\)](#)

Specifies the file transfer protocol or protocols over which your file transfer protocol client can connect to your server's endpoint. The available protocols are:

- Secure Shell (SSH) File Transfer Protocol (SFTP): File transfer over SSH
- File Transfer Protocol Secure (FTPS): File transfer with TLS encryption
- File Transfer Protocol (FTP): Unencrypted file transfer

Note

If you select `FTPS`, you must choose a certificate stored in AWS Certificate Manager (ACM) which will be used to identify your server when clients connect to it over `FTPS`.

If `Protocol` includes either `FTP` or `FTPS`, then the `EndpointType` must be `VPC` and the `IdentityProviderType` must be `AWS_DIRECTORY_SERVICE` or `API_GATEWAY`.

If `Protocol` includes `FTP`, then `AddressAllocationIds` cannot be associated.

If `Protocol` is set only to `SFTP`, the `EndpointType` can be set to `PUBLIC` and the `IdentityProviderType` can be set to `SERVICE_MANAGED`.

Type: Array of strings

Array Members: Minimum number of 1 item. Maximum number of 3 items.

Valid Values: SFTP | FTP | FTPS

Required: No

[SecurityPolicyName \(p. 262\)](#)

Specifies the name of the security policy that is attached to the server.

Type: String

Length Constraints: Maximum length of 100.

Pattern: TransferSecurityPolicy-.+

Required: No

[ServerId \(p. 262\)](#)

A system-assigned unique identifier for a server instance that the user account is assigned to.

Type: String

Length Constraints: Fixed length of 19.

Pattern: ^s-([0-9a-f]{17})\$

Required: Yes

[WorkflowDetails \(p. 262\)](#)

Specifies the workflow ID for the workflow to assign and the execution role used for executing the workflow.

Type: [WorkflowDetails \(p. 327\)](#) object

Required: No

Response Syntax

```
{  
    "ServerId": "string"  
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

[ServerId \(p. 265\)](#)

A system-assigned unique identifier for a server that the user account is assigned to.

Type: String

Length Constraints: Fixed length of 19.

Pattern: ^s-([0-9a-f]{17})\$

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 330\)](#).

AccessDeniedException

You do not have sufficient access to perform this action.

HTTP Status Code: 400

ConflictException

This exception is thrown when the `UpdateServer` is called for a file transfer protocol-enabled server that has VPC as the endpoint type and the server's `VpcEndpointID` is not in the available state.

HTTP Status Code: 400

InternalServiceError

This exception is thrown when an error occurs in the AWSTransfer Family service.

HTTP Status Code: 500

InvalidRequestException

This exception is thrown when the client submits a malformed request.

HTTP Status Code: 400

ResourceExistsException

The requested resource does not exist.

HTTP Status Code: 400

ResourceNotFoundException

This exception is thrown when a resource is not found by the AWSTransfer Family service.

HTTP Status Code: 400

ServiceUnavailableException

The request has failed because the AWSTransfer Family service is not available.

HTTP Status Code: 500

ThrottlingException

The request was denied due to request throttling.

HTTP Status Code: 400

HTTP Status Code: 400

Examples

Example

The following example updates the role of a server.

Sample Request



```
{  
    "EndpointDetails": {  
        "VpcEndpointId": "vpce-01234f056f3g13",  
        "LoggingRole": "CloudWatchS3Events",  
        "ServerId": "s-01234567890abcdef"  
    }  
}
```

Example

This example illustrates one usage of UpdateServer.

Sample Response

```
{  
    "ServerId": "s-01234567890abcdef"  
}
```

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

UpdateUser

Assigns new properties to a user. Parameters you pass modify any or all of the following: the home directory, role, and policy for the `UserName` and `ServerId` you specify.

The response returns the `ServerId` and the `UserName` for the updated user.

Request Syntax

```
{  
    "HomeDirectory": "string",  
    "HomeDirectoryMappings": [  
        {  
            "Entry": "string",  
            "Target": "string"  
        }  
    ],  
    "HomeDirectoryType": "string",  
    "Policy": "string",  
    "PosixProfile": {  
        "Gid": number,  
        "SecondaryGids": [ number ],  
        "Uid": number  
    },  
    "Role": "string",  
    "ServerId": "string",  
    "UserName": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 329\)](#).

The request accepts the following data in JSON format.

HomeDirectory (p. 268)

The landing directory (folder) for a user when they log in to the server using the client.

A `HomeDirectory` example is `/bucket_name/home/mydirectory`.

Type: String

Length Constraints: Maximum length of 1024.

Pattern: `^$ | /.*`

Required: No

HomeDirectoryMappings (p. 268)

Logical directory mappings that specify what Amazon S3 or Amazon EFS paths and keys should be visible to your user and how you want to make them visible. You must specify the `Entry` and `Target` pair, where `Entry` shows how the path is made visible and `Target` is the actual Amazon S3 or Amazon EFS path. If you only specify a target, it is displayed as is. You also must ensure that your AWS Identity and Access Management (IAM) role provides access to paths in `Target`. This value can only be set when `HomeDirectoryType` is set to `LOGICAL`.

The following is an `Entry` and `Target` pair example.

```
[ { "Entry": "/directory1", "Target": "/bucket_name/home/mydirectory" } ]
```

In most cases, you can use this value instead of the session policy to lock down your user to the designated home directory ("chroot"). To do this, you can set `Entry` to '/' and set `Target` to the `HomeDirectory` parameter value.

The following is an `Entry` and `Target` pair example for chroot.

```
[ { "Entry": "/", "Target": "/bucket_name/home/mydirectory" } ]
```

Type: Array of [HomeDirectoryMapEntry \(p. 300\)](#) objects

Array Members: Minimum number of 1 item. Maximum number of 50 items.

Required: No

[HomeDirectoryType \(p. 268\)](#)

The type of landing directory (folder) you want your users' home directory to be when they log into the server. If you set it to `PATH`, the user will see the absolute Amazon S3 bucket or EFS paths as is in their file transfer protocol clients. If you set it `LOGICAL`, you need to provide mappings in the `HomeDirectoryMappings` for how you want to make Amazon S3 or EFS paths visible to your users.

Type: String

Valid Values: `PATH` | `LOGICAL`

Required: No

[Policy \(p. 268\)](#)

A session policy for your user so that you can use the same IAM role across multiple users. This policy scopes down user access to portions of their Amazon S3 bucket. Variables that you can use inside this policy include `Transfer:UserName`, `Transfer:HomeDirectory`, and `Transfer:HomeBucket`.

Note

This only applies when the domain of `ServerId` is `S3`. EFS does not use session policies. For session policies, AWS Transfer Family stores the policy as a JSON blob, instead of the Amazon Resource Name (ARN) of the policy. You save the policy as a JSON blob and pass it in the `Policy` argument.

For an example of a session policy, see [Example session policy](#).

For more information, see [AssumeRole](#) in the [AWS Security Token Service API Reference](#).

Type: String

Length Constraints: Maximum length of 2048.

Required: No

[PosixProfile \(p. 268\)](#)

Specifies the full POSIX identity, including user ID (`Uid`), group ID (`Gid`), and any secondary groups IDs (`SecondaryGids`), that controls your users' access to your Amazon Elastic File Systems (Amazon EFS). The POSIX permissions that are set on files and directories in your file system determines the level of access your users get when transferring files into and out of your Amazon EFS file systems.

Type: [PosixProfile \(p. 314\)](#) object

Required: No

[Role \(p. 268\)](#)

Specifies the Amazon Resource Name (ARN) of the IAM role that controls your users' access to your Amazon S3 bucket or EFS file system. The policies attached to this role determine the level of access

that you want to provide your users when transferring files into and out of your Amazon S3 bucket or EFS file system. The IAM role should also contain a trust relationship that allows the server to access your resources when servicing your users' transfer requests.

Type: String

Length Constraints: Minimum length of 20. Maximum length of 2048.

Pattern: `arn:.*role/.*`

Required: No

[ServerId \(p. 268\)](#)

A system-assigned unique identifier for a server instance that the user account is assigned to.

Type: String

Length Constraints: Fixed length of 19.

Pattern: `^s-([0-9a-f]{17})$`

Required: Yes

[UserName \(p. 268\)](#)

A unique string that identifies a user and is associated with a server as specified by the `ServerId`. This user name must be a minimum of 3 and a maximum of 100 characters long. The following are valid characters: a-z, A-Z, 0-9, underscore '_', hyphen '-', period '.', and at sign '@'. The user name can't start with a hyphen, period, or at sign.

Type: String

Length Constraints: Minimum length of 3. Maximum length of 100.

Pattern: `^[\w][\w@.-]{2,99}$`

Required: Yes

Response Syntax

```
{  
    "ServerId": "string",  
    "UserName": "string"  
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

[ServerId \(p. 270\)](#)

A system-assigned unique identifier for a server instance that the user account is assigned to.

Type: String

Length Constraints: Fixed length of 19.

Pattern: `^s-([0-9a-f]{17})$`

[UserName \(p. 270\)](#)

The unique identifier for a user that is assigned to a server instance that was specified in the request.

Type: String

Length Constraints: Minimum length of 3. Maximum length of 100.

Pattern: ^[\w][\w@.-]{2,99}\$

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 330\)](#).

InternalServiceError

This exception is thrown when an error occurs in the AWSTransfer Family service.

HTTP Status Code: 500

InvalidRequestException

This exception is thrown when the client submits a malformed request.

HTTP Status Code: 400

ResourceNotFoundException

This exception is thrown when a resource is not found by the AWSTransfer Family service.

HTTP Status Code: 400

ServiceUnavailableException

The request has failed because the AWSTransfer Family service is not available.

HTTP Status Code: 500

ThrottlingException

The request was denied due to request throttling.

HTTP Status Code: 400

HTTP Status Code: 400

Examples

Example

The following example updates a user account.

Sample Request

```
{  
    "HomeDirectory": "/bucket2/documentation",  
    "HomeDirectoryMappings": [  
        {  
            "Entry": "/directory1",  
            "Target": "/bucket_name/home/mydirectory"  
        }  
    ]  
}
```

```
        ],
    "HomeDirectoryType": "PATH",
    "Role": "AssumeRole",
    "ServerId": "s-01234567890abcdef",
    "UserName": "my_user"
}
```

Example

This example illustrates one usage of UpdateUser.

Sample Response

```
{
    "ServerId": "s-01234567890abcdef",
    "UserName": "my_user"
}
```

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

Data Types

The following data types are supported:

- [CopyStepDetails \(p. 274\)](#)
- [CustomStepDetails \(p. 275\)](#)
- [DeleteStepDetails \(p. 276\)](#)
- [DescribedAccess \(p. 277\)](#)
- [DescribedExecution \(p. 280\)](#)
- [DescribedSecurityPolicy \(p. 282\)](#)
- [DescribedServer \(p. 284\)](#)
- [DescribedUser \(p. 288\)](#)
- [DescribedWorkflow \(p. 291\)](#)
- [EfsFileLocation \(p. 293\)](#)
- [EndpointDetails \(p. 294\)](#)
- [ExecutionError \(p. 296\)](#)
- [ExecutionResults \(p. 297\)](#)

- [ExecutionStepResult \(p. 298\)](#)
- [FileLocation \(p. 299\)](#)
- [HomeDirectoryMapEntry \(p. 300\)](#)
- [IdentityProviderDetails \(p. 301\)](#)
- [InputFileLocation \(p. 303\)](#)
- [ListedAccess \(p. 304\)](#)
- [ListedExecution \(p. 306\)](#)
- [ListedServer \(p. 307\)](#)
- [ListedUser \(p. 310\)](#)
- [ListedWorkflow \(p. 312\)](#)
- [LoggingConfiguration \(p. 313\)](#)
- [PosixProfile \(p. 314\)](#)
- [ProtocolDetails \(p. 315\)](#)
- [S3FileLocation \(p. 317\)](#)
- [S3InputFileLocation \(p. 319\)](#)
- [S3Tag \(p. 320\)](#)
- [ServiceMetadata \(p. 321\)](#)
- [SshPublicKey \(p. 322\)](#)
- [Tag \(p. 323\)](#)
- [TagStepDetails \(p. 324\)](#)
- [UserDetails \(p. 325\)](#)
- [WorkflowDetail \(p. 326\)](#)
- [WorkflowDetails \(p. 327\)](#)
- [WorkflowStep \(p. 328\)](#)

CopyStepDetails

Each step type has its own `StepDetails` structure.

Contents

DestinationFileLocation

Specifies the location for the file being copied. Only applicable for Copy type workflow steps. Use `#{Transfer:username}` in this field to parametrize the destination prefix by username.

Type: [InputFileLocation \(p. 303\)](#) object

Required: No

Name

The name of the step, used as an identifier.

Type: String

Length Constraints: Maximum length of 30.

Pattern: `^[\w-]*$`

Required: No

OverwriteExisting

A flag that indicates whether or not to overwrite an existing file of the same name. The default is `FALSE`.

Type: String

Valid Values: `TRUE` | `FALSE`

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

CustomStepDetails

Each step type has its own `StepDetails` structure.

Contents

Name

The name of the step, used as an identifier.

Type: String

Length Constraints: Maximum length of 30.

Pattern: `^[\w-]*$`

Required: No

Target

The ARN for the lambda function that is being called.

Type: String

Length Constraints: Maximum length of 170.

Pattern: `arn:[a-z-]+:lambda:.*$`

Required: No

TimeoutSeconds

Timeout, in seconds, for the step.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 1800.

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

DeleteStepDetails

The name of the step, used to identify the delete step.

Contents

Name

The name of the step, used as an identifier.

Type: String

Length Constraints: Maximum length of 30.

Pattern: ^[\w-]*\$

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

DescribedAccess

Describes the properties of the access that was specified.

Contents

ExternalId

A unique identifier that is required to identify specific groups within your directory. The users of the group that you associate have access to your Amazon S3 or Amazon EFS resources over the enabled protocols using AWS Transfer Family. If you know the group name, you can view the SID values by running the following command using Windows PowerShell.

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

In that command, replace *YourGroupName* with the name of your Active Directory group.

The regex used to validate this parameter is a string of characters consisting of uppercase and lowercase alphanumeric characters with no spaces. You can also include underscores or any of the following characters: =,.@:/-

Type: String

Length Constraints: Minimum length of 1. Maximum length of 256.

Pattern: ^S-1-[\d-]+\$

Required: No

HomeDirectory

The landing directory (folder) for a user when they log in to the server using the client.

A *HomeDirectory* example is /bucket_name/home/mydirectory.

Type: String

Length Constraints: Maximum length of 1024.

Pattern: ^\$ | / . *

Required: No

HomeDirectoryMappings

Logical directory mappings that specify what Amazon S3 or Amazon EFS paths and keys should be visible to your user and how you want to make them visible. You must specify the *Entry* and *Target* pair, where *Entry* shows how the path is made visible and *Target* is the actual Amazon S3 or Amazon EFS path. If you only specify a target, it is displayed as is. You also must ensure that your AWS Identity and Access Management (IAM) role provides access to paths in *Target*. This value can only be set when *HomeDirectoryType* is set to *LOGICAL*.

In most cases, you can use this value instead of the session policy to lock down the associated access to the designated home directory ("chroot"). To do this, you can set *Entry* to '/' and set *Target* to the *HomeDirectory* parameter value.

Type: Array of [HomeDirectoryMapEntry \(p. 300\)](#) objects

Array Members: Minimum number of 1 item. Maximum number of 50 items.

Required: No

HomeDirectoryType

The type of landing directory (folder) you want your users' home directory to be when they log into the server. If you set it to `PATH`, the user will see the absolute Amazon S3 bucket or EFS paths as is in their file transfer protocol clients. If you set it `LOGICAL`, you need to provide mappings in the `HomeDirectoryMappings` for how you want to make Amazon S3 or EFS paths visible to your users.

Type: String

Valid Values: `PATH` | `LOGICAL`

Required: No

Policy

A session policy for your user so that you can use the same IAM role across multiple users. This policy scopes down user access to portions of their Amazon S3 bucket. Variables that you can use inside this policy include `Transfer:UserName`, `Transfer:HomeDirectory`, and `Transfer:HomeBucket`.

Type: String

Length Constraints: Maximum length of 2048.

Required: No

PosixProfile

The full POSIX identity, including user ID (`Uid`), group ID (`Gid`), and any secondary groups IDs (`SecondaryGids`), that controls your users' access to your Amazon EFS file systems. The POSIX permissions that are set on files and directories in your file system determine the level of access your users get when transferring files into and out of your Amazon EFS file systems.

Type: [PosixProfile \(p. 314\)](#) object

Required: No

Role

Specifies the Amazon Resource Name (ARN) of the IAM role that controls your users' access to your Amazon S3 bucket or EFS file system. The policies attached to this role determine the level of access that you want to provide your users when transferring files into and out of your Amazon S3 bucket or EFS file system. The IAM role should also contain a trust relationship that allows the server to access your resources when servicing your users' transfer requests.

Type: String

Length Constraints: Minimum length of 20. Maximum length of 2048.

Pattern: `arn:.*role/.*`

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)

- [AWS SDK for Ruby V3](#)

DescribedExecution

The details for an execution object.

Contents

ExecutionId

A unique identifier for the execution of a workflow.

Type: String

Length Constraints: Fixed length of 36.

Pattern: ^[0-9a-fA-F]{8}\-[0-9a-fA-F]{4}\-[0-9a-fA-F]{4}\-[0-9a-fA-F]{4}\-[0-9a-fA-F]{12}\$

Required: No

ExecutionRole

The IAM role associated with the execution.

Type: String

Length Constraints: Minimum length of 20. Maximum length of 2048.

Pattern: arn:.*role/.*

Required: No

InitialFileLocation

A structure that describes the Amazon S3 or EFS file location. This is the file location when the execution begins: if the file is being copied, this is the initial (as opposed to destination) file location.

Type: [FileLocation \(p. 299\)](#) object

Required: No

LoggingConfiguration

The IAM logging role associated with the execution.

Type: [LoggingConfiguration \(p. 313\)](#) object

Required: No

PosixProfile

The full POSIX identity, including user ID (`Uid`), group ID (`Gid`), and any secondary groups IDs (`SecondaryGids`), that controls your users' access to your Amazon EFS file systems. The POSIX permissions that are set on files and directories in your file system determine the level of access your users get when transferring files into and out of your Amazon EFS file systems.

Type: [PosixProfile \(p. 314\)](#) object

Required: No

Results

A structure that describes the execution results. This includes a list of the steps along with the details of each step, error type and message (if any), and the `OnExceptionSteps` structure.

Type: [ExecutionResults \(p. 297\)](#) object

Required: No

ServiceMetadata

A container object for the session details associated with a workflow.

Type: [ServiceMetadata \(p. 321\)](#) object

Required: No

Status

The status is one of the execution. Can be in progress, completed, exception encountered, or handling the exception.

Type: String

Valid Values: IN_PROGRESS | COMPLETED | EXCEPTION | HANDLING_EXCEPTION

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

DescribedSecurityPolicy

Describes the properties of a security policy that was specified. For more information about security policies, see [Working with security policies](#).

Contents

Fips

Specifies whether this policy enables Federal Information Processing Standards (FIPS).

Type: Boolean

Required: No

SecurityPolicyName

Specifies the name of the security policy that is attached to the server.

Type: String

Length Constraints: Maximum length of 100.

Pattern: TransferSecurityPolicy-.+

Required: Yes

SshCiphers

Specifies the enabled Secure Shell (SSH) cipher encryption algorithms in the security policy that is attached to the server.

Type: Array of strings

Length Constraints: Maximum length of 50.

Required: No

SshKexs

Specifies the enabled SSH key exchange (KEX) encryption algorithms in the security policy that is attached to the server.

Type: Array of strings

Length Constraints: Maximum length of 50.

Required: No

SshMacs

Specifies the enabled SSH message authentication code (MAC) encryption algorithms in the security policy that is attached to the server.

Type: Array of strings

Length Constraints: Maximum length of 50.

Required: No

TlsCiphers

Specifies the enabled Transport Layer Security (TLS) cipher encryption algorithms in the security policy that is attached to the server.

Type: Array of strings

Length Constraints: Maximum length of 50.

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

DescribedServer

Describes the properties of a file transfer protocol-enabled server that was specified.

Contents

Arn

Specifies the unique Amazon Resource Name (ARN) of the server.

Type: String

Length Constraints: Minimum length of 20. Maximum length of 1600.

Pattern: `arn:.*`

Required: Yes

Certificate

Specifies the ARN of the AWS Certificate Manager (ACM) certificate. Required when `Protocols` is set to `FTPS`.

Type: String

Length Constraints: Maximum length of 1600.

Required: No

Domain

Specifies the domain of the storage system that is used for file transfers.

Type: String

Valid Values: `S3` | `EFS`

Required: No

EndpointDetails

The virtual private cloud (VPC) endpoint settings that are configured for your server. When you host your endpoint within your VPC, you can make it accessible only to resources within your VPC, or you can attach Elastic IP addresses and make it accessible to clients over the internet. Your VPC's default security groups are automatically assigned to your endpoint.

Type: [EndpointDetails \(p. 294\)](#) object

Required: No

EndpointType

Defines the type of endpoint that your server is connected to. If your server is connected to a VPC endpoint, your server isn't accessible over the public internet.

Type: String

Valid Values: `PUBLIC` | `VPC` | `VPC_ENDPOINT`

Required: No

HostKeyFingerprint

Specifies the Base64-encoded SHA256 fingerprint of the server's host key. This value is equivalent to the output of the `ssh-keygen -l -f my-new-server-key` command.

Type: String

Required: No

IdentityProviderDetails

Specifies information to call a customer-supplied authentication API. This field is not populated when the `IdentityProviderType` of a server is `AWS_DIRECTORY_SERVICE` or `SERVICE_MANAGED`.

Type: [IdentityProviderDetails \(p. 301\)](#) object

Required: No

IdentityProviderType

Specifies the mode of authentication for a server. The default value is `SERVICE_MANAGED`, which allows you to store and access user credentials within the AWS Transfer Family service.

Use `AWS_DIRECTORY_SERVICE` to provide access to Active Directory groups in AWS Managed Active Directory or Microsoft Active Directory in your on-premises environment or in AWS using AD Connectors. This option also requires you to provide a Directory ID using the `IdentityProviderDetails` parameter.

Use the `API_GATEWAY` value to integrate with an identity provider of your choosing. The `API_GATEWAY` setting requires you to provide an API Gateway endpoint URL to call for authentication using the `IdentityProviderDetails` parameter.

Use the `AWS_LAMBDA` value to directly use a Lambda function as your identity provider. If you choose this value, you must specify the ARN for the lambda function in the `Function` parameter for the `IdentityProviderDetails` data type.

Type: String

Valid Values: `SERVICE_MANAGED` | `API_GATEWAY` | `AWS_DIRECTORY_SERVICE` | `AWS_LAMBDA`

Required: No

LoggingRole

Specifies the Amazon Resource Name (ARN) of the AWS Identity and Access Management (IAM) role that allows a server to turn on Amazon CloudWatch logging for Amazon S3 or Amazon EFS events. When set, user activity can be viewed in your CloudWatch logs.

Type: String

Length Constraints: Minimum length of 20. Maximum length of 2048.

Pattern: `arn:.*role/.*`

Required: No

ProtocolDetails

The protocol settings that are configured for your server.

Use the `PassiveIp` parameter to indicate passive mode. Enter a single dotted-quad IPv4 address, such as the external IP address of a firewall, router, or load balancer.

Type: [ProtocolDetails \(p. 315\)](#) object

Required: No

Protocols

Specifies the file transfer protocol or protocols over which your file transfer protocol client can connect to your server's endpoint. The available protocols are:

- **SFTP** (Secure Shell (SSH) File Transfer Protocol): File transfer over SSH
- **FTPS** (File Transfer Protocol Secure): File transfer with TLS encryption
- **FTP** (File Transfer Protocol): Unencrypted file transfer

Type: Array of strings

Array Members: Minimum number of 1 item. Maximum number of 3 items.

Valid Values: SFTP | FTP | FTPS

Required: No

SecurityPolicyName

Specifies the name of the security policy that is attached to the server.

Type: String

Length Constraints: Maximum length of 100.

Pattern: TransferSecurityPolicy- .+

Required: No

ServerId

Specifies the unique system-assigned identifier for a server that you instantiate.

Type: String

Length Constraints: Fixed length of 19.

Pattern: ^s-([0-9a-f]{17})\$

Required: No

State

Specifies the condition of a server for the server that was described. A value of **ONLINE** indicates that the server can accept jobs and transfer files. A **State** value of **OFFLINE** means that the server cannot perform file transfer operations.

The states of **STARTING** and **STOPPING** indicate that the server is in an intermediate state, either not fully able to respond, or not fully offline. The values of **START_FAILED** or **STOP_FAILED** can indicate an error condition.

Type: String

Valid Values: OFFLINE | ONLINE | STARTING | STOPPING | START_FAILED | STOP_FAILED

Required: No

Tags

Specifies the key-value pairs that you can use to search for and group servers that were assigned to the server that was described.

Type: Array of [Tag \(p. 323\)](#) objects

Array Members: Minimum number of 1 item. Maximum number of 50 items.

Required: No

UserCount

Specifies the number of users that are assigned to a server you specified with the `ServerId`.

Type: Integer

Required: No

WorkflowDetails

Specifies the workflow ID for the workflow to assign and the execution role used for executing the workflow.

Type: [WorkflowDetails \(p. 327\)](#) object

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

DescribedUser

Describes the properties of a user that was specified.

Contents

Arn

Specifies the unique Amazon Resource Name (ARN) for the user that was requested to be described.

Type: String

Length Constraints: Minimum length of 20. Maximum length of 1600.

Pattern: arn:.*

Required: Yes

HomeDirectory

The landing directory (folder) for a user when they log in to the server using the client.

A HomeDirectory example is /bucket_name/home/mydirectory.

Type: String

Length Constraints: Maximum length of 1024.

Pattern: ^\$ | / .*

Required: No

HomeDirectoryMappings

Logical directory mappings that specify what Amazon S3 or Amazon EFS paths and keys should be visible to your user and how you want to make them visible. You must specify the `Entry` and `Target` pair, where `Entry` shows how the path is made visible and `Target` is the actual Amazon S3 or Amazon EFS path. If you only specify a target, it is displayed as is. You also must ensure that your AWS Identity and Access Management (IAM) role provides access to paths in `Target`. This value can only be set when `HomeDirectoryType` is set to `LOGICAL`.

In most cases, you can use this value instead of the session policy to lock your user down to the designated home directory ("chroot"). To do this, you can set `Entry` to '/' and set `Target` to the `HomeDirectory` parameter value.

Type: Array of [HomeDirectoryMapEntry \(p. 300\)](#) objects

Array Members: Minimum number of 1 item. Maximum number of 50 items.

Required: No

HomeDirectoryType

The type of landing directory (folder) you want your users' home directory to be when they log into the server. If you set it to `PATH`, the user will see the absolute Amazon S3 bucket or EFS paths as is in their file transfer protocol clients. If you set it `LOGICAL`, you need to provide mappings in the `HomeDirectoryMappings` for how you want to make Amazon S3 or EFS paths visible to your users.

Type: String

Valid Values: `PATH` | `LOGICAL`

Required: No

Policy

A session policy for your user so that you can use the same IAM role across multiple users. This policy scopes down user access to portions of their Amazon S3 bucket. Variables that you can use inside this policy include \${Transfer:UserName}, \${Transfer:HomeDirectory}, and \${Transfer:HomeBucket}.

Type: String

Length Constraints: Maximum length of 2048.

Required: No

PosixProfile

Specifies the full POSIX identity, including user ID (`Uid`), group ID (`Gid`), and any secondary groups IDs (`SecondaryGids`), that controls your users' access to your Amazon Elastic File System (Amazon EFS) file systems. The POSIX permissions that are set on files and directories in your file system determine the level of access your users get when transferring files into and out of your Amazon EFS file systems.

Type: [PosixProfile \(p. 314\)](#) object

Required: No

Role

Specifies the Amazon Resource Name (ARN) of the IAM role that controls your users' access to your Amazon S3 bucket or EFS file system. The policies attached to this role determine the level of access that you want to provide your users when transferring files into and out of your Amazon S3 bucket or EFS file system. The IAM role should also contain a trust relationship that allows the server to access your resources when servicing your users' transfer requests.

Type: String

Length Constraints: Minimum length of 20. Maximum length of 2048.

Pattern: `arn:.*role/.*`

Required: No

SshPublicKeys

Specifies the public key portion of the Secure Shell (SSH) keys stored for the described user.

Type: Array of [SshPublicKey \(p. 322\)](#) objects

Array Members: Maximum number of 5 items.

Required: No

Tags

Specifies the key-value pairs for the user requested. Tag can be used to search for and group users for a variety of purposes.

Type: Array of [Tag \(p. 323\)](#) objects

Array Members: Minimum number of 1 item. Maximum number of 50 items.

Required: No

UserName

Specifies the name of the user that was requested to be described. User names are used for authentication purposes. This is the string that will be used by your user when they log in to your server.

Type: String

Length Constraints: Minimum length of 3. Maximum length of 100.

Pattern: `^[\w][\w@.-]{2,99}$`

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

DescribedWorkflow

Describes the properties of the specified workflow

Contents

Arn

Specifies the unique Amazon Resource Name (ARN) for the workflow.

Type: String

Length Constraints: Minimum length of 20. Maximum length of 1600.

Pattern: `arn: .*`

Required: Yes

Description

Specifies the text description for the workflow.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[\w-]*$`

Required: No

OnExceptionSteps

Specifies the steps (actions) to take if errors are encountered during execution of the workflow.

Type: Array of [WorkflowStep \(p. 328\)](#) objects

Array Members: Maximum number of 8 items.

Required: No

Steps

Specifies the details for the steps that are in the specified workflow.

Type: Array of [WorkflowStep \(p. 328\)](#) objects

Array Members: Maximum number of 8 items.

Required: No

Tags

Key-value pairs that can be used to group and search for workflows. Tags are metadata attached to workflows for any purpose.

Type: Array of [Tag \(p. 323\)](#) objects

Array Members: Minimum number of 1 item. Maximum number of 50 items.

Required: No

WorkflowId

A unique identifier for the workflow.

Type: String

Length Constraints: Fixed length of 19.

Pattern: ^w-([a-zA-Z0-9]{17})\$

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

EfsFileLocation

Reserved for future use.

Contents

FileSystemId

The ID of the file system, assigned by Amazon EFS.

Type: String

Length Constraints: Maximum length of 128.

Pattern: `^arn:aws[-a-z]*:elasticfilesystem:[0-9a-z-]+:(access-point/fsap|file-system/fs)-[0-9a-f]{8,40}|fs(ap)?-[0-9a-f]{8,40})$`

Required: No

Path

The pathname for the folder being used by a workflow.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 100.

Pattern: `^(\/|(\//(?!\.)+[^\$#<>|^&?{}^*/\n]+){1,4})$`

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

EndpointDetails

The virtual private cloud (VPC) endpoint settings that are configured for your file transfer protocol-enabled server. With a VPC endpoint, you can restrict access to your server and resources only within your VPC. To control incoming internet traffic, invoke the [UpdateServer API](#) and attach an Elastic IP address to your server's endpoint.

Note

After May 19, 2021, you won't be able to create a server using `EndpointType=VPC_ENDPOINT` in your AWS account if your account hasn't already done so before May 19, 2021. If you have already created servers with `EndpointType=VPC_ENDPOINT` in your AWS account on or before May 19, 2021, you will not be affected. After this date, use `EndpointType=VPC`.

For more information, see [Discontinuing the use of VPC_ENDPOINT \(p. 42\)](#).

Contents

AddressAllocationIds

A list of address allocation IDs that are required to attach an Elastic IP address to your server's endpoint.

Note

This property can only be set when `EndpointType` is set to `VPC` and it is only valid in the [UpdateServer API](#).

Type: Array of strings

Required: No

SecurityGroupIds

A list of security groups IDs that are available to attach to your server's endpoint.

Note

This property can only be set when `EndpointType` is set to `VPC`.

You can edit the `SecurityGroupIds` property in the [UpdateServer API](#) only if you are changing the `EndpointType` from `PUBLIC` or `VPC_ENDPOINT` to `VPC`. To change security groups associated with your server's VPC endpoint after creation, use the Amazon EC2 [ModifyVpcEndpoint API](#).

Type: Array of strings

Length Constraints: Minimum length of 11. Maximum length of 20.

Pattern: `^sg-[0-9a-f]{8,17}$`

Required: No

SubnetIds

A list of subnet IDs that are required to host your server endpoint in your VPC.

Note

This property can only be set when `EndpointType` is set to `VPC`.

Type: Array of strings

Required: No

VpcEndpointId

The ID of the VPC endpoint.

Note

This property can only be set when `EndpointType` is set to `VPC_ENDPOINT`.
For more information, see [Discontinuing the use of VPC_ENDPOINT \(p. 42\)](#).

Type: String

Length Constraints: Fixed length of 22.

Pattern: ^vpce-[0-9a-f]{17}\$

Required: No

VpcId

The VPC ID of the VPC in which a server's endpoint will be hosted.

Note

This property can only be set when `EndpointType` is set to `VPC`.

Type: String

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

ExecutionError

Specifies the error message and type, for an error that occurs during the execution of the workflow.

Contents

Message

Specifies the descriptive message that corresponds to the `ErrorType`.

Type: String

Required: Yes

Type

Specifies the error type: currently, the only valid value is `PERMISSION_DENIED`, which occurs if your policy does not contain the correct permissions to complete one or more of the steps in the workflow.

Type: String

Valid Values: `PERMISSION_DENIED`

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

ExecutionResults

Specifies the steps in the workflow, as well as the steps to execute in case of any errors during workflow execution.

Contents

OnExceptionSteps

Specifies the steps (actions) to take if errors are encountered during execution of the workflow.

Type: Array of [ExecutionStepResult \(p. 298\)](#) objects

Array Members: Minimum number of 1 item. Maximum number of 50 items.

Required: No

Steps

Specifies the details for the steps that are in the specified workflow.

Type: Array of [ExecutionStepResult \(p. 298\)](#) objects

Array Members: Minimum number of 1 item. Maximum number of 50 items.

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

ExecutionStepResult

Specifies the following details for the step: error (if any), outputs (if any), and the step type.

Contents

Error

Specifies the details for an error, if it occurred during execution of the specified workflow step.

Type: [ExecutionError \(p. 296\)](#) object

Required: No

Outputs

The values for the key/value pair applied as a tag to the file. Only applicable if the step type is TAG.

Type: String

Length Constraints: Maximum length of 65536.

Required: No

StepType

One of the available step types.

- *Copy*: copy the file to another location
- *Custom*: custom step with a lambda target
- *Delete*: delete the file
- *Tag*: add a tag to the file

Type: String

Valid Values: COPY | CUSTOM | TAG | DELETE

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

FileLocation

Specifies the Amazon S3 or EFS file details to be used in the step.

Contents

EfsFileLocation

Specifies the Amazon EFS ID and the path for the file being used.

Type: [EfsFileLocation \(p. 293\)](#) object

Required: No

S3FileLocation

Specifies the S3 details for the file being used, such as bucket, Etag, and so forth.

Type: [S3FileLocation \(p. 317\)](#) object

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

HomeDirectoryMapEntry

Represents an object that contains entries and targets for HomeDirectoryMappings.

The following is an `Entry` and `Target` pair example for chroot.

```
[ { "Entry": "/", "Target": "/bucket_name/home/mydirectory" } ]
```

Contents

Entry

Represents an entry for HomeDirectoryMappings.

Type: String

Length Constraints: Maximum length of 1024.

Pattern: ^/.*

Required: Yes

Target

Represents the map target that is used in a HomeDirectorymapEntry.

Type: String

Length Constraints: Maximum length of 1024.

Pattern: ^/.*

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

IdentityProviderDetails

Returns information related to the type of user authentication that is in use for a file transfer protocol-enabled server's users. A server can have only one method of authentication.

Contents

DirectoryId

The identifier of the AWS Directory Service directory that you want to stop sharing.

Type: String

Length Constraints: Fixed length of 12.

Pattern: ^d-[0-9a-f]{10}\$

Required: No

Function

The ARN for a lambda function to use for the Identity provider.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 170.

Pattern: ^arn:[a-z-]+:lambda:.*\$

Required: No

InvocationRole

Provides the type of `InvocationRole` used to authenticate the user account.

Type: String

Length Constraints: Minimum length of 20. Maximum length of 2048.

Pattern: arn: .*role/.*

Required: No

Url

Provides the location of the service endpoint used to authenticate users.

Type: String

Length Constraints: Maximum length of 255.

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)

- [AWS SDK for Ruby V3](#)

InputFileLocation

Specifies the location for the file being copied. Only applicable for the Copy type of workflow steps.

Contents

EfsFileLocation

Reserved for future use.

Type: [EfsFileLocation \(p. 293\)](#) object

Required: No

S3FileLocation

Specifies the details for the S3 file being copied.

Type: [S3InputFileLocation \(p. 319\)](#) object

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

ListedAccess

Lists the properties for one or more specified associated accesses.

Contents

ExternalId

A unique identifier that is required to identify specific groups within your directory. The users of the group that you associate have access to your Amazon S3 or Amazon EFS resources over the enabled protocols using AWS Transfer Family. If you know the group name, you can view the SID values by running the following command using Windows PowerShell.

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

In that command, replace *YourGroupName* with the name of your Active Directory group.

The regex used to validate this parameter is a string of characters consisting of uppercase and lowercase alphanumeric characters with no spaces. You can also include underscores or any of the following characters: =,.@:/-

Type: String

Length Constraints: Minimum length of 1. Maximum length of 256.

Pattern: ^S-1-[\d-]+\$

Required: No

HomeDirectory

The landing directory (folder) for a user when they log in to the server using the client.

A *HomeDirectory* example is /bucket_name/home/mydirectory.

Type: String

Length Constraints: Maximum length of 1024.

Pattern: ^\$ | /.*

Required: No

HomeDirectoryType

The type of landing directory (folder) you want your users' home directory to be when they log into the server. If you set it to PATH, the user will see the absolute Amazon S3 bucket or EFS paths as is in their file transfer protocol clients. If you set it LOGICAL, you need to provide mappings in the *HomeDirectoryMappings* for how you want to make Amazon S3 or EFS paths visible to your users.

Type: String

Valid Values: PATH | LOGICAL

Required: No

Role

Specifies the Amazon Resource Name (ARN) of the IAM role that controls your users' access to your Amazon S3 bucket or EFS file system. The policies attached to this role determine the level of access that you want to provide your users when transferring files into and out of your Amazon S3 bucket

or EFS file system. The IAM role should also contain a trust relationship that allows the server to access your resources when servicing your users' transfer requests.

Type: String

Length Constraints: Minimum length of 20. Maximum length of 2048.

Pattern: `arn:.*role/.*`

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

ListedExecution

Returns properties of the execution that is specified.

Contents

ExecutionId

A unique identifier for the execution of a workflow.

Type: String

Length Constraints: Fixed length of 36.

Pattern: ^[0-9a-fA-F]{8}\-[0-9a-fA-F]{4}\-[0-9a-fA-F]{4}\-[0-9a-fA-F]{4}\-[0-9a-fA-F]{12}\$

Required: No

InitialFileLocation

A structure that describes the Amazon S3 or EFS file location. This is the file location when the execution begins: if the file is being copied, this is the initial (as opposed to destination) file location.

Type: [FileLocation \(p. 299\)](#) object

Required: No

ServiceMetadata

A container object for the session details associated with a workflow.

Type: [ServiceMetadata \(p. 321\)](#) object

Required: No

Status

The status is one of the execution. Can be in progress, completed, exception encountered, or handling the exception.

Type: String

Valid Values: IN_PROGRESS | COMPLETED | EXCEPTION | HANDLING_EXCEPTION

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

ListedServer

Returns properties of a file transfer protocol-enabled server that was specified.

Contents

Arn

Specifies the unique Amazon Resource Name (ARN) for a server to be listed.

Type: String

Length Constraints: Minimum length of 20. Maximum length of 1600.

Pattern: arn:.*

Required: Yes

Domain

Specifies the domain of the storage system that is used for file transfers.

Type: String

Valid Values: S3 | EFS

Required: No

EndpointType

Specifies the type of VPC endpoint that your server is connected to. If your server is connected to a VPC endpoint, your server isn't accessible over the public internet.

Type: String

Valid Values: PUBLIC | VPC | VPC_ENDPOINT

Required: No

IdentityProviderType

Specifies the mode of authentication for a server. The default value is SERVICE_MANAGED, which allows you to store and access user credentials within the AWS Transfer Family service.

Use AWS_DIRECTORY_SERVICE to provide access to Active Directory groups in AWS Managed Active Directory or Microsoft Active Directory in your on-premises environment or in AWS using AD Connectors. This option also requires you to provide a Directory ID using the IdentityProviderDetails parameter.

Use the API_GATEWAY value to integrate with an identity provider of your choosing. The API_GATEWAY setting requires you to provide an API Gateway endpoint URL to call for authentication using the IdentityProviderDetails parameter.

Use the AWS_LAMBDA value to directly use a Lambda function as your identity provider. If you choose this value, you must specify the ARN for the lambda function in the Function parameter for the IdentityProviderDetails data type.

Type: String

Valid Values: SERVICE_MANAGED | API_GATEWAY | AWS_DIRECTORY_SERVICE | AWS_LAMBDA

Required: No

LoggingRole

Specifies the Amazon Resource Name (ARN) of the AWS Identity and Access Management (IAM) role that allows a server to turn on Amazon CloudWatch logging for Amazon S3 or Amazon EFS events. When set, user activity can be viewed in your CloudWatch logs.

Type: String

Length Constraints: Minimum length of 20. Maximum length of 2048.

Pattern: `arn:.*role/.*`

Required: No

ServerId

Specifies the unique system assigned identifier for the servers that were listed.

Type: String

Length Constraints: Fixed length of 19.

Pattern: `^s-([0-9a-f]{17})$`

Required: No

State

Specifies the condition of a server for the server that was described. A value of `ONLINE` indicates that the server can accept jobs and transfer files. A `State` value of `OFFLINE` means that the server cannot perform file transfer operations.

The states of `STARTING` and `STOPPING` indicate that the server is in an intermediate state, either not fully able to respond, or not fully offline. The values of `START_FAILED` or `STOP_FAILED` can indicate an error condition.

Type: String

Valid Values: `OFFLINE` | `ONLINE` | `STARTING` | `STOPPING` | `START_FAILED` | `STOP_FAILED`

Required: No

UserCount

Specifies the number of users that are assigned to a server you specified with the `ServerId`.

Type: Integer

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

ListedUser

Returns properties of the user that you specify.

Contents

Arn

Provides the unique Amazon Resource Name (ARN) for the user that you want to learn about.

Type: String

Length Constraints: Minimum length of 20. Maximum length of 1600.

Pattern: arn: .*

Required: Yes

HomeDirectory

The landing directory (folder) for a user when they log in to the server using the client.

A HomeDirectory example is /bucket_name/home/mydirectory.

Type: String

Length Constraints: Maximum length of 1024.

Pattern: ^\$ | / .*

Required: No

HomeDirectoryType

The type of landing directory (folder) you want your users' home directory to be when they log into the server. If you set it to PATH, the user will see the absolute Amazon S3 bucket or EFS paths as is in their file transfer protocol clients. If you set it LOGICAL, you need to provide mappings in the HomeDirectoryMappings for how you want to make Amazon S3 or EFS paths visible to your users.

Type: String

Valid Values: PATH | LOGICAL

Required: No

Role

Specifies the Amazon Resource Name (ARN) of the IAM role that controls your users' access to your Amazon S3 bucket or EFS file system. The policies attached to this role determine the level of access that you want to provide your users when transferring files into and out of your Amazon S3 bucket or EFS file system. The IAM role should also contain a trust relationship that allows the server to access your resources when servicing your users' transfer requests.

Note

The IAM role that controls your users' access to your Amazon S3 bucket for servers with Domain=S3, or your EFS file system for servers with Domain=EFS.

The policies attached to this role determine the level of access you want to provide your users when transferring files into and out of your S3 buckets or EFS file systems.

Type: String

Length Constraints: Minimum length of 20. Maximum length of 2048.

Pattern: arn:.*role/.*

Required: No

SshPublicKeyCount

Specifies the number of SSH public keys stored for the user you specified.

Type: Integer

Required: No

UserName

Specifies the name of the user whose ARN was specified. User names are used for authentication purposes.

Type: String

Length Constraints: Minimum length of 3. Maximum length of 100.

Pattern: ^[\w][\w@.-]{2,99}\$

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

ListedWorkflow

Contains the ID, text description, and Amazon Resource Name (ARN) for the workflow.

Contents

Arn

Specifies the unique Amazon Resource Name (ARN) for the workflow.

Type: String

Length Constraints: Minimum length of 20. Maximum length of 1600.

Pattern: `arn: .*`

Required: No

Description

Specifies the text description for the workflow.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[\w-]*$`

Required: No

WorkflowId

A unique identifier for the workflow.

Type: String

Length Constraints: Fixed length of 19.

Pattern: `^w-([a-zA-Z0-9]{17})$`

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

LoggingConfiguration

Consists of the logging role and the log group name.

Contents

LoggingRole

Specifies the Amazon Resource Name (ARN) of the AWS Identity and Access Management (IAM) role that allows a server to turn on Amazon CloudWatch logging for Amazon S3 or Amazon EFS events. When set, user activity can be viewed in your CloudWatch logs.

Type: String

Length Constraints: Minimum length of 20. Maximum length of 2048.

Pattern: `arn:.*role/.*`

Required: No

LogGroupName

The name of the CloudWatch logging group for the AWS Transfer server to which this workflow belongs.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 512.

Pattern: `[\. \-/_#A-Za-z0-9]*`

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

PosixProfile

The full POSIX identity, including user ID (`Uid`), group ID (`Gid`), and any secondary groups IDs (`SecondaryGids`), that controls your users' access to your Amazon EFS file systems. The POSIX permissions that are set on files and directories in your file system determine the level of access your users get when transferring files into and out of your Amazon EFS file systems.

Contents

Gid

The POSIX group ID used for all EFS operations by this user.

Type: Long

Valid Range: Minimum value of 0. Maximum value of 4294967295.

Required: Yes

SecondaryGids

The secondary POSIX group IDs used for all EFS operations by this user.

Type: Array of longs

Array Members: Minimum number of 0 items. Maximum number of 16 items.

Valid Range: Minimum value of 0. Maximum value of 4294967295.

Required: No

Uid

The POSIX user ID used for all EFS operations by this user.

Type: Long

Valid Range: Minimum value of 0. Maximum value of 4294967295.

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

ProtocolDetails

The protocol settings that are configured for your server.

Contents

PassiveIp

Indicates passive mode, for FTP and FTPS protocols. Enter a single dotted-quad IPv4 address, such as the external IP address of a firewall, router, or load balancer. For example:

```
aws transfer update-server --protocol-details PassiveIp=0.0.0.0
```

Replace `0.0.0.0` in the example above with the actual IP address you want to use.

Note

If you change the `PassiveIp` value, you must stop and then restart your Transfer server for the change to take effect. For details on using Passive IP (PASV) in a NAT environment, see [Configuring your FTPS server behind a firewall or NAT with AWS Transfer Family](#).

Type: String

Length Constraints: Maximum length of 15.

Required: No

TlsSessionResumptionMode

A property used with Transfer servers that use the FTPS protocol. TLS Session Resumption provides a mechanism to resume or share a negotiated secret key between the control and data connection for an FTPS session. `TlsSessionResumptionMode` determines whether or not the server resumes recent, negotiated sessions through a unique session ID. This property is available during `CreateServer` and `UpdateServer` calls. If a `TlsSessionResumptionMode` value is not specified during `CreateServer`, it is set to `ENFORCED` by default.

- `DISABLED`: the server does not process TLS session resumption client requests and creates a new TLS session for each request.
- `ENABLED`: the server processes and accepts clients that are performing TLS session resumption. The server doesn't reject client data connections that do not perform the TLS session resumption client processing.
- `ENFORCED`: the server processes and accepts clients that are performing TLS session resumption. The server rejects client data connections that do not perform the TLS session resumption client processing. Before you set the value to `ENFORCED`, test your clients.

Note

Not all FTPS clients perform TLS session resumption. So, if you choose to enforce TLS session resumption, you prevent any connections from FTPS clients that don't perform the protocol negotiation. To determine whether or not you can use the `ENFORCED` value, you need to test your clients.

Type: String

Valid Values: `DISABLED` | `ENABLED` | `ENFORCED`

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

S3FileLocation

Specifies the details for the file location for the file being used in the workflow. Only applicable if you are using S3 storage.

Contents

Bucket

Specifies the S3 bucket that contains the file being used.

Type: String

Length Constraints: Minimum length of 3. Maximum length of 63.

Pattern: ^[a-zA-Z0-9][\.\-\a-zA-Z0-9]{1,61}[a-zA-Z0-9]\$

Required: No

Etag

The entity tag is a hash of the object. The ETag reflects changes only to the contents of an object, not its metadata.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 65536.

Pattern: ^ . +\$

Required: No

Key

The name assigned to the file when it was created in S3. You use the object key to retrieve the object.

Type: String

Length Constraints: Maximum length of 1024.

Pattern: [\P{M}\p{M}]*

Required: No

VersionId

Specifies the file version.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 1024.

Pattern: ^ . +\$

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

S3InputFileLocation

Specifies the customer input S3 file location. If it is used inside `copyStepDetails.DestinationFileLocation`, it should be the S3 copy destination.

You need to provide the bucket and key. The key can represent either a path or a file. This is determined by whether or not you end the key value with the forward slash (/) character. If the final character is "/", then your file is copied to the folder, and its name does not change. If, rather, the final character is alphanumeric, your uploaded file is renamed to the path value. In this case, if a file with that name already exists, it is overwritten.

For example, if your path is `shared-files/bob/`, your uploaded files are copied to the `shared-files/bob/` folder. If your path is `shared-files/today`, each uploaded file is copied to the `shared-files` folder and named `today`: each upload overwrites the previous version of the `bob` file.

Contents

Bucket

Specifies the S3 bucket for the customer input file.

Type: String

Length Constraints: Minimum length of 3. Maximum length of 63.

Pattern: `^[a-zA-Z0-9][\.\-\a-zA-Z0-9]{1,61}[a-zA-Z0-9]$`

Required: No

Key

The name assigned to the file when it was created in S3. You use the object key to retrieve the object.

Type: String

Length Constraints: Maximum length of 1024.

Pattern: `[\P{M}\p{M}]^*`

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

S3Tag

Specifies the key-value pair that are assigned to a file during the execution of a Tagging step.

Contents

Key

The name assigned to the tag that you create.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 128.

Pattern: `^([\p{L}\p{Z}]\p{N}_.:/+\\-@]*$`

Required: Yes

Value

The value that corresponds to the key.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^([\p{L}\p{Z}]\p{N}_.:/+\\-@]*$`

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

ServiceMetadata

A container object for the session details associated with a workflow.

Contents

UserDetails

The Server ID (`ServerId`), Session ID (`SessionId`) and user (`UserName`) make up the `UserDetails`.

Type: [UserDetails \(p. 325\)](#) object

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

SshPublicKey

Provides information about the public Secure Shell (SSH) key that is associated with a user account for the specific file transfer protocol-enabled server (as identified by `ServerId`). The information returned includes the date the key was imported, the public key contents, and the public key ID. A user can store more than one SSH public key associated with their user name on a specific server.

Contents

DateImported

Specifies the date that the public key was added to the user account.

Type: Timestamp

Required: Yes

SshPublicKeyBody

Specifies the content of the SSH public key as specified by the `PublicKeyId`.

Type: String

Length Constraints: Maximum length of 2048.

Pattern: ^ssh-rsa\s+[A-Za-z0-9+/\]+[=]{0,3}(\s+.+)?\s*\$

Required: Yes

SshPublicKeyId

Specifies the `SshPublicKeyId` parameter contains the identifier of the public key.

Type: String

Length Constraints: Fixed length of 21.

Pattern: ^key-[0-9a-f]{17}\$

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

Tag

Creates a key-value pair for a specific resource. Tags are metadata that you can use to search for and group a resource for various purposes. You can apply tags to servers, users, and roles. A tag key can take more than one value. For example, to group servers for accounting purposes, you might create a tag called `Group` and assign the values `Research` and `Accounting` to that group.

Contents

Key

The name assigned to the tag that you create.

Type: String

Length Constraints: Maximum length of 128.

Required: Yes

Value

Contains one or more values that you assigned to the key name you create.

Type: String

Length Constraints: Maximum length of 256.

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

TagStepDetails

Each step type has its own `StepDetails` structure.

The key/value pairs used to tag a file during the execution of a workflow step.

Contents

Name

The name of the step, used as an identifier.

Type: String

Length Constraints: Maximum length of 30.

Pattern: `^[\w-]*$`

Required: No

Tags

Array that contains from 1 to 10 key/value pairs.

Type: Array of [S3Tag \(p. 320\)](#) objects

Array Members: Minimum number of 1 item. Maximum number of 10 items.

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

UserDetails

Specifies the user name, server ID, and session ID for a workflow.

Contents

ServerId

The system-assigned unique identifier for a Transfer server instance.

Type: String

Length Constraints: Fixed length of 19.

Pattern: ^s-([0-9a-f]{17})\$

Required: Yes

SessionId

The system-assigned unique identifier for a session that corresponds to the workflow.

Type: String

Length Constraints: Minimum length of 3. Maximum length of 32.

Pattern: ^[\w-]*\$

Required: No

UserName

A unique string that identifies a user account associated with a server.

Type: String

Length Constraints: Minimum length of 3. Maximum length of 100.

Pattern: ^[\w][\w@.-]{2,99}\$

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

WorkflowDetail

Specifies the workflow ID for the workflow to assign and the execution role used for executing the workflow.

Contents

ExecutionRole

Includes the necessary permissions for S3, EFS, and Lambda operations that Transfer can assume, so that all workflow steps can operate on the required resources

Type: String

Length Constraints: Minimum length of 20. Maximum length of 2048.

Pattern: `arn:.*role/.*`

Required: Yes

WorkflowId

A unique identifier for the workflow.

Type: String

Length Constraints: Fixed length of 19.

Pattern: `^w-([a-zA-Z0-9]{17})$`

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

WorkflowDetails

Container for the `WorkflowDetail` data type. It is used by actions that trigger a workflow to begin execution.

Contents

OnUpload

A trigger that starts a workflow: the workflow begins to execute after a file is uploaded.

Type: Array of [WorkflowDetail \(p. 326\)](#) objects

Array Members: Maximum number of 1 item.

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

WorkflowStep

The basic building block of a workflow.

Contents

CopyStepDetails

Details for a step that performs a file copy.

Consists of the following values:

- A description
- An S3 location for the destination of the file copy.
- A flag that indicates whether or not to overwrite an existing file of the same name. The default is `FALSE`.

Type: [CopyStepDetails \(p. 274\)](#) object

Required: No

CustomStepDetails

Details for a step that invokes a lambda function.

Consists of the lambda function name, target, and timeout (in seconds).

Type: [CustomStepDetails \(p. 275\)](#) object

Required: No

DeleteStepDetails

Details for a step that deletes the file.

Type: [DeleteStepDetails \(p. 276\)](#) object

Required: No

TagStepDetails

Details for a step that creates one or more tags.

You specify one or more tags: each tag contains a key/value pair.

Type: [TagStepDetails \(p. 324\)](#) object

Required: No

Type

Currently, the following step types are supported.

- *Copy*: copy the file to another location
- *Custom*: custom step with a lambda target
- *Delete*: delete the file
- *Tag*: add a tag to the file

Type: String

Valid Values: `COPY` | `CUSTOM` | `TAG` | `DELETE`

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

Common Parameters

The following list contains the parameters that all actions use for signing Signature Version 4 requests with a query string. Any action-specific parameters are listed in the topic for that action. For more information about Signature Version 4, see [Signature Version 4 Signing Process](#) in the *Amazon Web Services General Reference*.

Action

The action to be performed.

Type: string

Required: Yes

Version

The API version that the request is written for, expressed in the format YYYY-MM-DD.

Type: string

Required: Yes

X-Amz-Algorithm

The hash algorithm that you used to create the request signature.

Condition: Specify this parameter when you include authentication information in a query string instead of in the HTTP authorization header.

Type: string

Valid Values: AWS4-HMAC-SHA256

Required: Conditional

X-Amz-Credential

The credential scope value, which is a string that includes your access key, the date, the region you are targeting, the service you are requesting, and a termination string ("aws4_request"). The value is expressed in the following format: *access_key/YYYYMMDD/region/service/aws4_request*.

For more information, see [Task 2: Create a String to Sign for Signature Version 4](#) in the *Amazon Web Services General Reference*.

Condition: Specify this parameter when you include authentication information in a query string instead of in the HTTP authorization header.

Type: string

Required: Conditional

X-Amz-Date

The date that is used to create the signature. The format must be ISO 8601 basic format (YYYYMMDD'T'HHMMSS'Z'). For example, the following date time is a valid X-Amz-Date value: 20120325T120000Z.

Condition: X-Amz-Date is optional for all requests; it can be used to override the date used for signing requests. If the Date header is specified in the ISO 8601 basic format, X-Amz-Date is not required. When X-Amz-Date is used, it always overrides the value of the Date header. For more information, see [Handling Dates in Signature Version 4](#) in the *Amazon Web Services General Reference*.

Type: string

Required: Conditional

X-Amz-Security-Token

The temporary security token that was obtained through a call to AWS Security Token Service (AWS STS). For a list of services that support temporary security credentials from AWS Security Token Service, go to [AWS Services That Work with IAM](#) in the *IAM User Guide*.

Condition: If you're using temporary security credentials from the AWS Security Token Service, you must include the security token.

Type: string

Required: Conditional

X-Amz-Signature

Specifies the hex-encoded signature that was calculated from the string to sign and the derived signing key.

Condition: Specify this parameter when you include authentication information in a query string instead of in the HTTP authorization header.

Type: string

Required: Conditional

X-Amz-SignedHeaders

Specifies all the HTTP headers that were included as part of the canonical request. For more information about specifying signed headers, see [Task 1: Create a Canonical Request For Signature Version 4](#) in the *Amazon Web Services General Reference*.

Condition: Specify this parameter when you include authentication information in a query string instead of in the HTTP authorization header.

Type: string

Required: Conditional

Common Errors

This section lists the errors common to the API actions of all AWS services. For errors specific to an API action for this service, see the topic for that API action.

AccessDeniedException

You do not have sufficient access to perform this action.

HTTP Status Code: 400

IncompleteSignature

The request signature does not conform to AWS standards.

HTTP Status Code: 400

InternalFailure

The request processing has failed because of an unknown error, exception or failure.

HTTP Status Code: 500

InvalidAction

The action or operation requested is invalid. Verify that the action is typed correctly.

HTTP Status Code: 400

InvalidClientTokenId

The X.509 certificate or AWS access key ID provided does not exist in our records.

HTTP Status Code: 403

InvalidParameterCombination

Parameters that must not be used together were used together.

HTTP Status Code: 400

InvalidParameterValue

An invalid or out-of-range value was supplied for the input parameter.

HTTP Status Code: 400

InvalidQueryParameter

The AWS query string is malformed or does not adhere to AWS standards.

HTTP Status Code: 400

MalformedQueryString

The query string contains a syntax error.

HTTP Status Code: 404

MissingAction

The request is missing an action or a required parameter.

HTTP Status Code: 400

MissingAuthenticationToken

The request must contain either a valid (registered) AWS access key ID or X.509 certificate.

HTTP Status Code: 403

MissingParameter

A required parameter for the specified action is not supplied.

HTTP Status Code: 400

NotAuthorized

You do not have permission to perform this action.

HTTP Status Code: 400

OptInRequired

The AWS access key ID needs a subscription for the service.

HTTP Status Code: 403

RequestExpired

The request reached the service more than 15 minutes after the date stamp on the request or more than 15 minutes after the request expiration date (such as for pre-signed URLs), or the date stamp on the request is more than 15 minutes in the future.

HTTP Status Code: 400

ServiceUnavailable

The request has failed due to a temporary failure of the server.

HTTP Status Code: 503

ThrottlingException

The request was denied due to request throttling.

HTTP Status Code: 400

ValidationException

The input fails to satisfy the constraints specified by an AWS service.

HTTP Status Code: 400

Document history for AWS Transfer Family

The following table describes the documentation for this release of AWS Transfer Family.

- **API version:** transfer-2018-11-05
- **Latest documentation update:** November 17, 2021

Change	Description	Date
Support for AWS Lambda as an identity provider	You can now connect to a custom identity provider using AWS Lambda with their Transfer Family servers. Previously, you had to supply an Amazon API Gateway URL to integrate a custom identity provider. For more information, see Using AWS Lambda to integrate your identity provider (p. 70) .	November 16, 2021
Support for Managed File Transfer Workflows	Managed File Transfer Workflows provide you with post-upload processing abstractions for the common tasks that you currently perform manually. For more information, see Managing workflows for post-upload processing (p. 96) .	September 2, 2021
Support for AWS Directory Service for Microsoft Active Directory	In addition to service managed and custom identity providers, you can now use AWS Directory Service for Microsoft Active Directory to manage user access for authentication and authorization. For more information, see Using the AWS Directory Service identity provider (p. 60) .	May 24, 2021
New AWS Regions	AWS Transfer Family is now available in the Africa (Cape Town) Region. For more information about Transfer Family endpoints, see AWS Transfer Family endpoints and quotas in the AWS General Reference .	February 24, 2021
New AWS Regions	AWS Transfer Family is now available in the Asia Pacific	February 17, 2021

Change	Description	Date
	(Hong Kong) and Middle East (Bahrain) Regions. For more information about Transfer Family endpoints, see AWS Transfer Family endpoints and quotas in the AWS General Reference .	
Support for Amazon EFS as a data store	Transfer Family now supports file transfers into and out of Amazon Elastic File System (Amazon EFS). Amazon EFS is a simple, scalable, fully managed elastic NFS file system. For more information, see Create an Amazon EFS file system (p. 6) .	January 06, 2021
Support for WAF	Transfer Family now supports Web Application Firewall (WAF), which uses AWS WAF, a web application firewall that helps protect web applications and APIs from attacks. For more information, see Add a web application firewall (p. 146) .	November 24, 2020
Support for multiple security groups in a virtual private cloud (VPC)	You can now attach multiple security groups to a server in a VPC. For more information, see Create a server in a virtual private cloud (p. 35) .	October 15, 2020
New AWS Regions	Transfer Family is now available in the AWS GovCloud (US) Regions. For more information about Transfer Family endpoints for AWS GovCloud (US) Regions, see AWS Transfer Family endpoints and quotas in the AWS General Reference . For information about using Transfer Family in the AWS GovCloud (US) Regions, see AWS Transfer Family in the AWS GovCloud (US) User Guide .	September 30, 2020
A security policy with supported cryptographic algorithms can now be attached to your server	You can now attach a security policy that contains a set of supported cryptographic algorithms to your server. For more information, see Working with security policies (p. 49) .	August 12, 2020

Change	Description	Date
Support for Federal Information Processing Standard (FIPS) endpoints	FIPS-enabled endpoints are now available in North American AWS Regions. For available regions, see AWS Transfer Family endpoints and quotas in the <i>AWS General Reference</i> . To enable FIPS for an SFTP-enabled server endpoint, see Create an SFTP-enabled server (p. 20) . To enable FIPS for an FTPS-enabled server endpoint, see Create an FTPS-enabled server (p. 25) . To enable FIPS for an FTP-enabled server endpoint, see Create an FTP-enabled server (p. 30) .	August 12, 2020
User name character length increase and additional allowed characters	User names can now have the at sign '@' and period '.' and be a maximum length of 100 characters. To add a user, see Managing users (p. 54) .	August 12, 2020
Support for automatic CloudWatch logging AWS Identity and Access Management (IAM) role creation	Transfer Family now supports automatic creation of a CloudWatch logging IAM role to view end-user activity. For more information, see Create an SFTP-enabled server (p. 20) , Create an FTPS-enabled server (p. 25) , or Create an FTP-enabled server (p. 30) .	July 30, 2020
AWS Transfer Family now supports Source IP as a factor for authorization.	Transfer Family adds support for using end users' source IP addresses as a factor for authorization, enabling you to apply an additional layer of security when authorizing access over Secure File Transfer Protocol (SFTP), File Transfer Protocol over SSL (FTPS), or File Transfer Protocol (FTP). For more information, see Working with custom identity providers (p. 69) .	June 9, 2020

Change	Description	Date
AWS Transfer for SFTP is now AWS Transfer Family and adds support for FTP and FTPS.	You can now use two additional protocols for your users' file transfers: File Transfer Protocol Secure (FTPS) and File Transfer Protocol (FTP). Users can move, run, secure, and integrate FTP over SSL (FTPS) and cleartext FTP based workflows in AWS, in addition to existing Secure File Transfer Protocol (SFTP) support.	April 23, 2020
Support for virtual private cloud (VPC) security groups and Elastic IP addresses	You can now create an allow list for incoming IP addresses using security groups, providing an additional layer of security for servers. You can also associate Elastic IP addresses with your server's endpoint. By doing this, you can enable users behind firewalls to allow access to that endpoint. For more information, see Create a server in a virtual private cloud (p. 35) .	January 10, 2020
Support for working in a VPC	You can now create a server in a VPC. You can use your server to transfer data over your client to and from an Amazon S3 bucket without going over the public internet. For more information, see Create a server in a virtual private cloud (p. 35) .	March 27, 2019
First version of AWS Transfer Family released.	This initial release includes setting up directions, describes how to get started, and provides information on client configuration, user configuration, and monitoring activity.	November 25, 2018

AWS glossary

For the latest AWS terminology, see the [AWS glossary](#) in the *AWS General Reference*.