Sync data between a source and target

Cloud Sync

NetApp
April 05, 2022

Table of Contents

Sync data between a source and target

Create sync relationships

When you create a sync relationship, the Cloud Sync service copies files from the source to the target. After the initial copy, the service syncs any changed data every 24 hours.

Before you can create some types of sync relationships, you'll first need to create a working environment in Cloud Manager.

Create sync relationships for specific types of working environments

If you want to create sync relationships for any of the following, then you first need to create or discover the working environment:

- Amazon FSx for ONTAP
- Azure NetApp Files
- Cloud Volumes ONTAP
- On-prem ONTAP clusters

Steps

1. Create or discover the working environment.

   ◦ Create an Amazon FSx for ONTAP working environment

   ◦ Setting up and discovering Azure NetApp Files

   ◦ Launching Cloud Volumes ONTAP in AWS

   ◦ Launching Cloud Volumes ONTAP in Azure

   ◦ Launching Cloud Volumes ONTAP in Google Cloud

   ◦ Adding existing Cloud Volumes ONTAP systems

   ◦ Discovering ONTAP clusters

2. Click **Canvas**.

3. Select a working environment that matches any of the types listed above.

4. Select the action menu next to Sync.
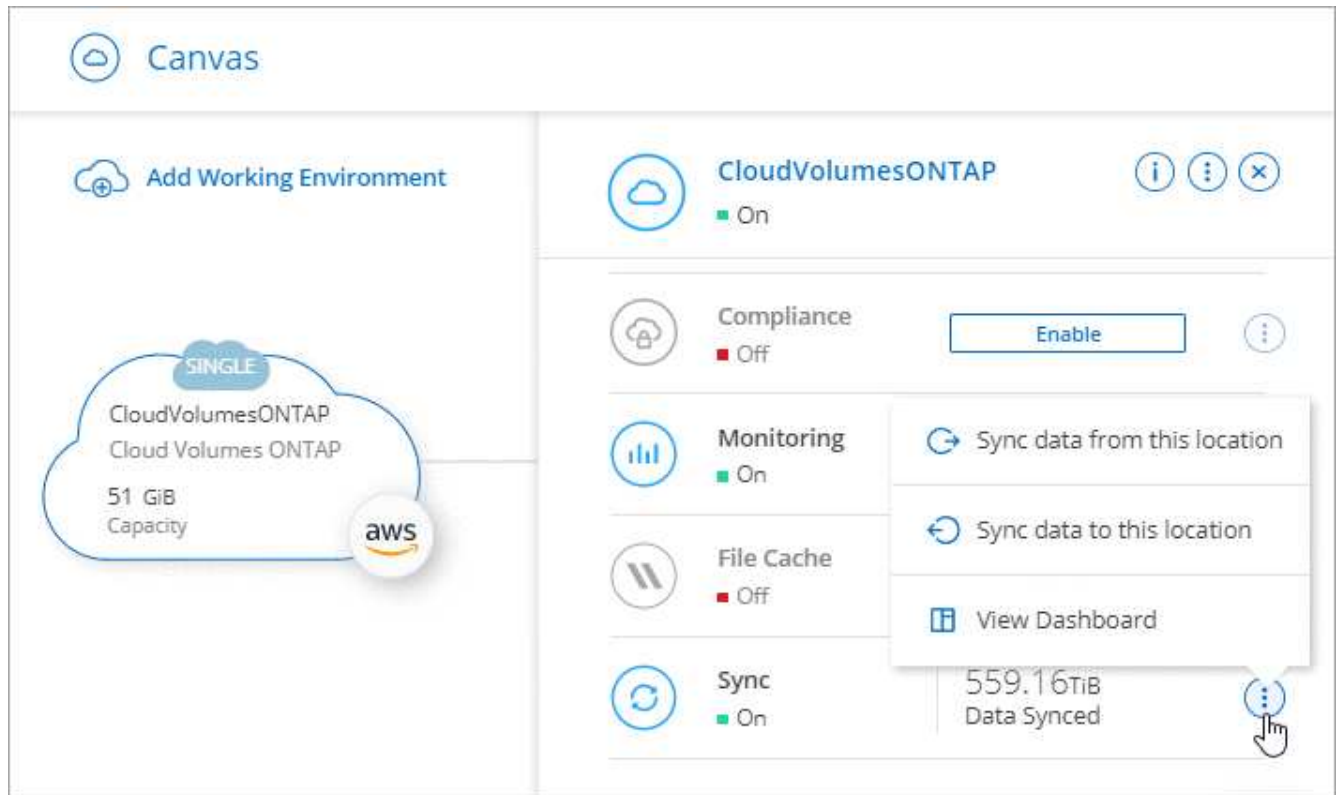
5. Select **Sync data from this location** or **Sync data to this location** and follow the prompts to set up the sync relationship.

Create other types of sync relationships

Use these steps to sync data to or from a supported storage type other than Amazon FSx for ONTAP, Azure NetApp Files, Cloud Volumes ONTAP, or on-prem ONTAP clusters. The steps below provide an example that shows how to set up a sync relationship from an NFS server to an S3 bucket.

1. In Cloud Manager, click **Sync**.

2. On the **Define Sync Relationship** page, choose a source and target.

    The following steps provide an example of how to create a sync relationship from an NFS server to an S3 bucket.

3. On the **NFS Server** page, enter the IP address or fully qualified domain name of the NFS server that you want to sync to AWS.

4. On the **Data Broker Group** page, follow the prompts to create a data broker virtual machine in AWS, Azure, or Google Cloud Platform, or to install the data broker software an existing Linux host.

   For more details, refer to the following pages:

   ◦ Create a data broker in AWS

   ◦ Create a data broker in Azure

   ◦ Create a data broker in Google Cloud

   ◦ Installing the data broker on a Linux host

5. After you install the data broker, click **Continue**.

6. On the **Directories** page, select a top-level directory or subdirectory.

   If Cloud Sync is unable to retrieve the exports, click **Add Export Manually** and enter the name of an NFS export.

   > ℹ️ If you want to sync more than one directory on the NFS server, then you must create additional sync relationships after you are done.

7. On the **AWS S3 Bucket** page, select a bucket:
   ◦ Drill down to select an existing folder within the bucket or to select a new folder that you create inside the bucket.
   ◦ Click **Add to the list** to select an S3 bucket that is not associated with your AWS account. Specific permissions must be applied to the S3 bucket.

8. On the **Bucket Setup** page, set up the bucket:
   ◦ Choose whether to enable S3 bucket encryption and then select an AWS KMS key, enter the ARN of a KMS key, or select AES-256 encryption.
   ◦ Select an S3 storage class. View the supported storage classes.
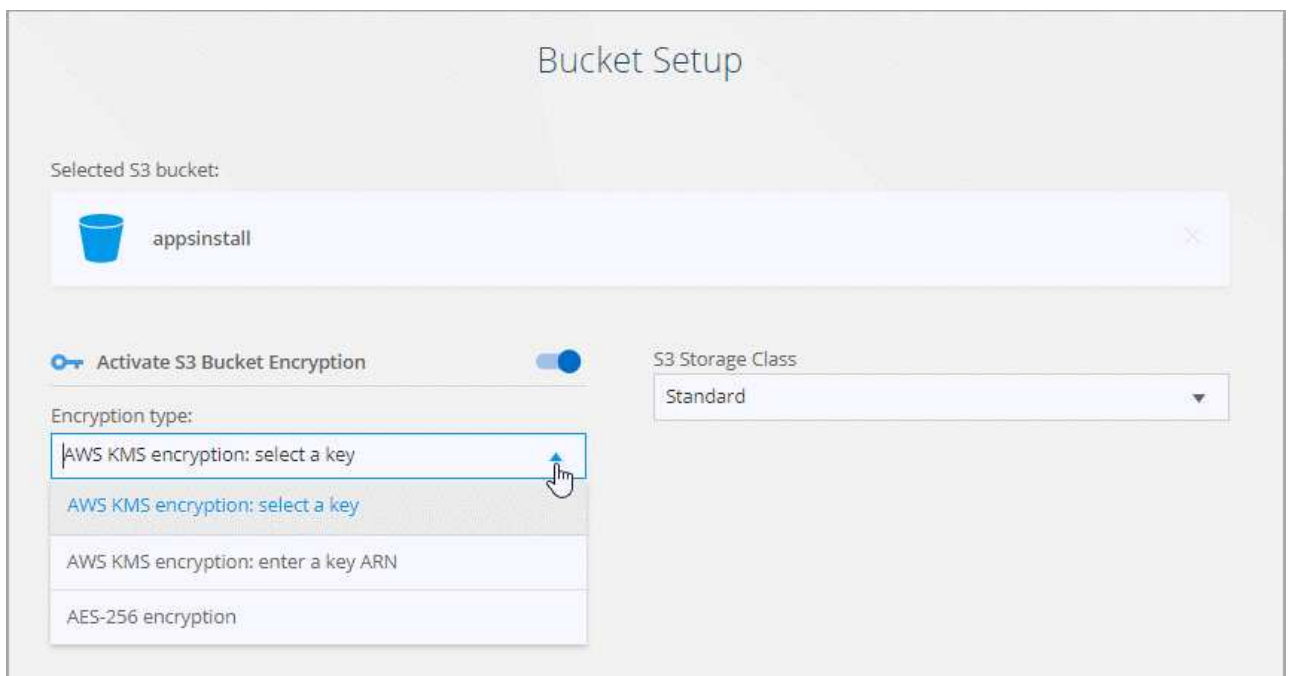
9. On the **Settings** page, define how source files and folders are synced and maintained in the target location:

Schedule

Choose a recurring schedule for future syncs or turn off the sync schedule. You can schedule a relationship to sync data as often as every 1 minute.

Retries

Define the number of times that Cloud Sync should retry to sync a file before skipping it.

Compare By

Choose whether Cloud Sync should compare certain attributes when determining whether a file or directory has changed and should be synced again.

Even if you uncheck these attributes, Cloud Sync still compares the source to the target by checking the paths, file sizes, and file names. If there are any changes, then it syncs those files and directories.

You can choose to enable or disable Cloud Sync from comparing the following attributes:

- **mtime**: The last modified time for a file. This attribute isn't valid for directories.
- **uid**, **gid**, and **mode**: Permission flags for Linux.

Copy for Objects

Enable this option to copy object storage metadata and tags. If a user changes the metadata on the source, Cloud Sync copies this object in the next sync, but if a user changes the tags on the source (and not the data itself), Cloud Sync doesn't copy the object in the next sync.

You can't edit this option after you create the relationship.

Copying tags is supported with sync relationships that include an S3-compatible endpoint (S3, StorageGRID, or IBM Cloud Object Storage).

Copying metadata is supported with "cloud-to-cloud" relationships between any of the following endpoints:

- AWS S3
- Azure Blob
- Google Cloud Storage
- IBM Cloud Object Storage
- StorageGRID

Recently Modified Files

Choose to exclude files that were recently modified prior to the scheduled sync.

Delete Files on Source

Choose to delete files from the source location after Cloud Sync copies the files to the target location. This option includes the risk of data loss because the source files are deleted after they're copied.

If you enable this option, you also need to change a parameter in the local.json file on the data broker. Open the file and update it as follows:

```
{
"workers":{
"transferrer":{
"delete-on-source": true
}
}
}
```

Delete Files on Target

Choose to delete files from the target location, if they were deleted from the source. The default is to never delete files from the target location.

File Types

Define the file types to include in each sync: files, directories, and symbolic links.

Exclude File Extensions

Specify file extensions to exclude from the sync by typing the file extension and pressing **Enter**. For example, type *log* or *.log* to exclude *.log files. A separator isn't required for multiple extensions. The following video provides a short demo:

► https://docs.netapp.com/us-en/cloud-manager-sync//media/video_file_extensions.mp4 *(video)*

File Size

Choose to sync all files regardless of their size or just files that are in a specific size range.

Date Modified

Choose all files regardless of their last modified date, files modified after a specific date, before a specific date, or between a time range.

Date Created

When an SMB server is the source, this setting enables you to sync files that were created after a specific date, before a specific date, or between a specific time range.

ACL - Access Control List

Copy ACLs from an SMB server by enabling a setting when you create a relationship or after you create a relationship.

10. On the **Tags/Metadata** page, choose whether to save a key-value pair as a tag on all files transferred to the S3 bucket or to assign a metadata key-value pair on all files.

> This same feature is available when syncing data to StorageGRID and IBM Cloud Object Storage. For Azure and Google Cloud Storage, only the metadata option is available.

11. Review the details of the sync relationship and then click **Create Relationship**.

Result

Cloud Sync starts syncing data between the source and target.

Create sync relationships from Cloud Data Sense

Cloud Sync is integrated with Cloud Data Sense. From within Data Sense, you can select the source files that you'd like to sync to a target location using Cloud Sync.

After you initiate a data sync from Cloud Data Sense, all of the source information is contained in a single step and only requires you to enter a few key details. You then choose the target location for the new sync relationship.

Learn how to start a sync relationship from Cloud Data Sense.

Copying ACLs from SMB shares

Cloud Sync can copy access control lists (ACLs) between a source SMB share and a target SMB share, or from a source SMB share to object storage (except for ONTAP S3). If needed, you also have the option to manually preserve ACLs between SMB shares by using robocopy.

> ⓘ Cloud Sync doesn't support copying ACLs back from object storage to SMB shares.

Choices

- Set up Cloud Sync to automatically copy ACLs
- Manually copy the ACLs between SMB shares

Setting up Cloud Sync to copy ACLs from an SMB server

Copy ACLs from an SMB server by enabling a setting when you create a relationship or after you create a relationship.

What you'll need

This feature works with *any* type of data broker: the AWS, Azure, Google Cloud Platform, or on-prem data broker. The on-prem data broker can run any supported operating system.

Steps for a new relationship

1. From Cloud Sync, click **Create New Sync**.

2. Drag and drop **SMB Server** to the source, choose an SMB server or object storage as the target, and click **Continue**.

3. On the **SMB Server** page:

   a. Enter a new SMB server or select an existing server and click **Continue**.

   b. Enter credentials for the SMB server.

   c. Select **Copy Access Control Lists to the target** and click **Continue**.

4. Follow the remaining prompts to create the sync relationship.

   When you copy ACLs from SMB to object storage, you can choose to copy the ACLs to the object's tags or on the object's metadata, depending on the target. For Azure and Google Cloud Storage, only the metadata option is available.

   The following screenshot shows an example of the step where you can make this choice.

Steps for an existing relationship

1. Hover over the sync relationship and click the action menu.

2. Click **Settings**.

3. Select **Copy Access Control Lists to the target**.

4. Click **Save Settings**.

Result

When syncing data, Cloud Sync preserves the ACLs between the source and target SMB shares, or from a source SMB share to object storage.

Manually copying ACLs between SMB shares

You can manually preserve ACLs between SMB shares by using the Windows robocopy command.

Steps

1. Identify a Windows host that has full access to both SMB shares.

2. If either of the endpoints require authentication, use the **net use** command to connect to the endpoints from the Windows host.

   You must perform this step before you use robocopy.

3. From Cloud Sync, create a new relationship between the source and target SMB shares or sync an existing relationship.

4. After the data sync is complete, run the following command from the Windows host to sync the ACLs and ownership:

   ```
   robocopy /E /COPY:SOU /secfix [source] [target] /w:0 /r:0 /XD ~snapshots
   /UNILOG:"[logfilepath]
   ```

   Both *source* and *target* should be specified using the UNC format. For example: \\<server>\<share>\<path>

Syncing NFS data using data-in-flight encryption

If your business has strict security policies, you can sync NFS data using data-in-flight encryption. This feature is supported from an NFS server to another NFS server and from Azure NetApp Files to Azure NetApp Files.

For example, you might want to sync data between two NFS servers that are in different networks. Or you might need to securely transfer data on Azure NetApp Files across subnets or regions.

How data-in-flight encryption works

Data-in-flight encryption encrypts NFS data when it's sent over the network between two data brokers. The following image shows a relationship between two NFS servers and two data brokers:
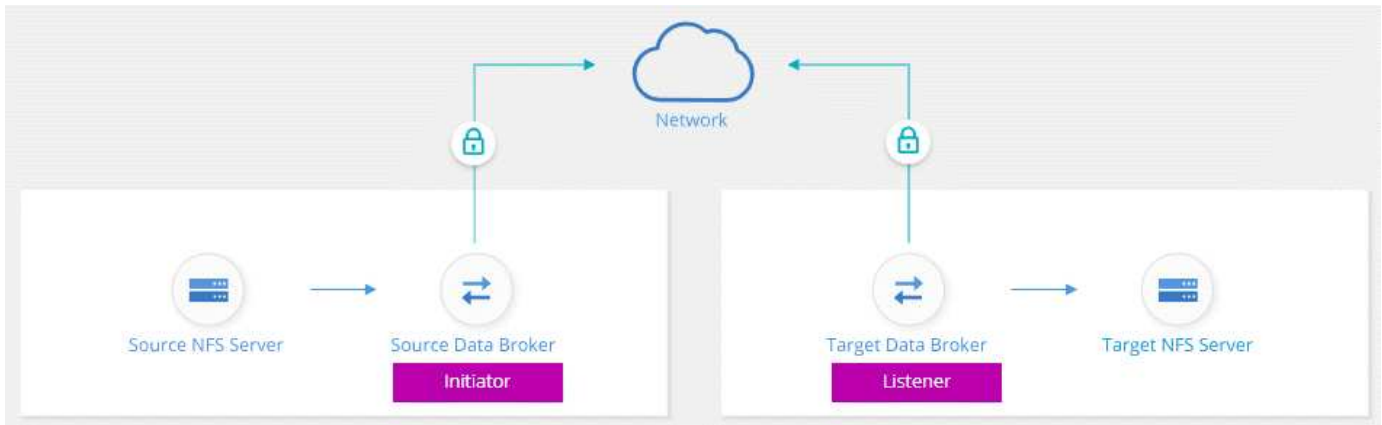
One data broker functions as the *initiator*. When it's time to sync data, it sends a connection request to the other data broker, which is the *listener*. That data broker listens for requests on port 443. You can use a different port, if needed, but be sure to check that the port is not in use by another service.

For example, if you sync data from an on-premises NFS server to a cloud-based NFS server, you can choose which data broker listens for the connection requests and which sends them.

Here's how in-flight encryption works:

1. After you create the sync relationship, the initiator starts an encrypted connection with the other data broker.
2. The source data broker encrypts data from the source using TLS 1.3.
3. It then sends the data over the network to the target data broker.
4. The target data broker decrypts the data before sending it to the target.
5. After the initial copy, the service syncs any changed data every 24 hours. If there is data to sync, the process starts with the initiator opening an encrypted connection with the other data broker.

   If you prefer to sync data more frequently, you can change the schedule after you create the relationship.

Supported NFS versions

- For NFS servers, data-in-flight encryption is supported with NFS versions 3, 4.0, 4.1, and 4.2.
- For Azure NetApp Files, data-in-flight encryption is supported with NFS versions 3 and 4.1.

Proxy server limitation

If you create an encrypted sync relationship, the encrypted data is sent over HTTPS and isn't routable through a proxy server.

What you'll need to get started

Be sure to have the following:

- Two NFS servers that meet source and target requirements or Azure NetApp Files in two subnets or regions.
- The IP addresses or fully qualified domain names of the servers.
- Network locations for two data brokers.

You can select an existing data broker but it must function as the initiator. The listener data broker must be a *new* data broker.

If you want to use an existing data broker group, the group must have only one data broker. Multiple data brokers in a group aren't supported with encrypted sync relationships.

If you have not yet deployed a data broker, review the data broker requirements. Because you have strict security policies, be sure to review the networking requirements, which includes outbound traffic from port 443 and the internet endpoints that the data broker contacts.

- ◦ Review AWS installation
- ◦ Review Azure installation
- ◦ Review Google Cloud installation
- ◦ Review Linux host installation

Syncing NFS data using data-in-flight encryption

Create a new sync relationship between two NFS servers or between Azure NetApp Files, enable the in-flight encryption option, and follow the prompts.

Steps

1. Click **Create New Sync**.

2. Drag and drop **NFS Server** to the source and target locations or **Azure NetApp Files** to the source and target locations and select **Yes** to enable data-in-flight encryption.

3. Follow the prompts to create the relationship:

   a. **NFS Server/Azure NetApp Files**: Choose the NFS version and then specify a new NFS source or select an existing server.

   b. **Define Data Broker Functionality**: Define which data broker *listens* for connection requests on a port and which one *initiates* the connection. Make your choice based on your networking requirements.

   c. **Data Broker**: Follow the prompts to add a new source data broker or select an existing data broker.

      Note the following:

      ▪ If you want to use an existing data broker group, the group must have only one data broker. Multiple data brokers in a group aren't supported with encrypted sync relationships.

      ▪ If the source data broker acts as the listener, then it must be a new data broker.

      ▪ If you need a new data broker, Cloud Sync prompts you with the installation instructions. You can deploy the data broker in the cloud or download an installation script for your own Linux host.

   d. **Directories**: Choose the directories that you want to sync by selecting all directories, or by drilling down and selecting a subdirectory.

      Click **Filter Source Objects** to modify settings that define how source files and folders are synced and maintained in the target location.
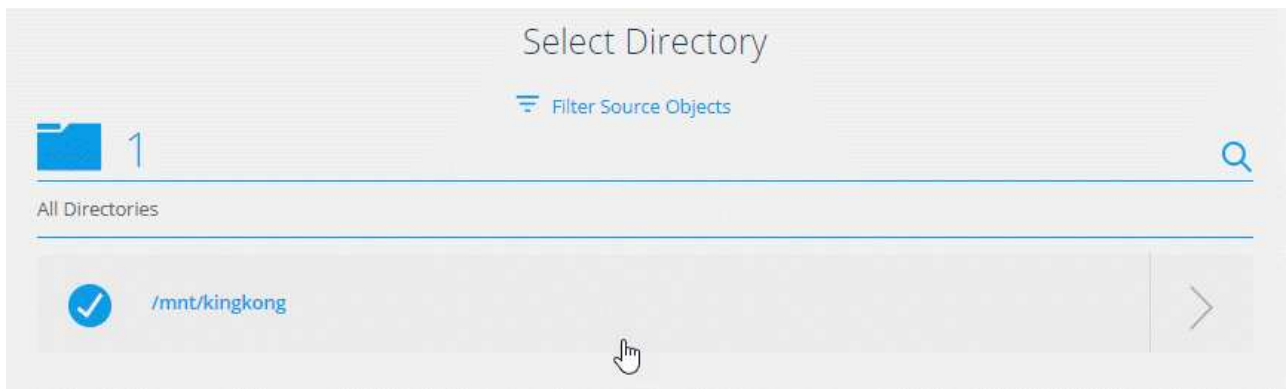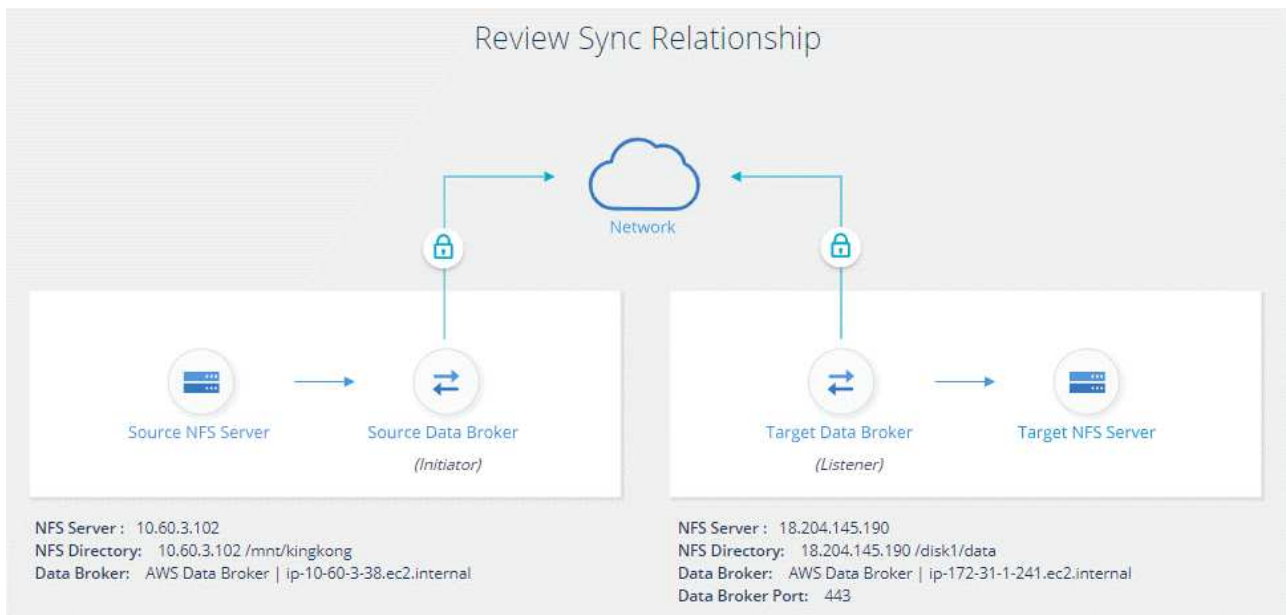
e. **Target NFS Server/Target Azure NetApp Files**: Choose the NFS version and then enter a new NFS target or select an existing server.

f. **Target Data Broker**: Follow the prompts to add a new source data broker or select an existing data broker.

If the target data broker acts as the listener, then it must be a new data broker.

Here's an example of the prompt when the target data broker functions as the listener. Notice the option to specify the port.

g. **Target Directories**: Select a top-level directory, or drill down to select an existing subdirectory or to create a new folder inside an export.

h. **Settings**: Define how source files and folders are synced and maintained in the target location.

i. **Review**: Review the details of the sync relationship and then click **Create Relationship**.

Review Sync Relationship

NFS Server : 10.60.3.102
NFS Directory: 10.60.3.102 /mnt/kingkong
Data Broker: AWS Data Broker | ip-10-60-3-38.ec2.internal

NFS Server : 18.204.145.190
NFS Directory: 18.204.145.190 /disk1/data
Data Broker: AWS Data Broker | ip-172-31-1-241.ec2.internal
Data Broker Port: 443

Result

Cloud Sync starts creating the new sync relationship. When it's done, click **View in Dashboard** to view details about the new relationship.

Setting up a data broker group to use an external HashiCorp Vault

When you create a sync relationship that requires Amazon S3, Azure, or Google Cloud credentials, you need to specify those credentials through the Cloud Sync user interface or API. An alternative is to set up the data broker group to access the credentials (or *secrets*) directly from an external HashiCorp Vault.

This feature is supported through the Cloud Sync API with sync relationships that require Amazon S3, Azure, or Google Cloud credentials.

1 **Prepare the vault**

Prepare the vault to supply credentials to the data broker group by setting up the URLs. The URLs to the secrets in the vault must end with *Creds*.

2 **Prepare the data broker group**

Prepare the data broker group to fetch credentials from the external vault by modifying the local config file for each data broker in the group.

3 **Create a sync relationship using the API**

Now that everything is set up, you can send an API call to create a sync relationship that uses your vault to get the secrets.

Preparing the vault

You'll need to provide Cloud Sync with the URL to the secrets in your vault. Prepare the vault by setting up those URLs. You need to set up URLs to the credentials for each source and target in the sync relationships that you plan to create.

The URL must be set up as follows:

```
/<path>/<requestid>/<endpoint-protocol>Creds
```

Path

> The prefix path to the secret. This can be any value that's unique to you.

Request ID

> A request ID that you need to generate. You'll need to provide the ID in one of the headers in the API POST request when you create the sync relationship.

Endpoint protocol

> One of the following protocols, as defined in the post relationship v2 documentation: S3, AZURE, or GCP (each must be in uppercase).

Creds

> The URL must end with *Creds*.

Examples

The following examples show URLs to secrets.

Example for the full URL and path for source credentials

> http://example.vault.com:8200/my-path/all-secrets/hb312vdasr2/S3Creds
>
> As you can see in the example, the prefix path is */my-path/all-secrets/*, the request ID is *hb312vdasr2* and the source endpoint is S3.

Example for the full URL and path for target credentials

> http://example.vault.com:8200/my-path/all-secrets/n32hcbnejk2/AZURECreds
>
> The prefix path is */my-path/all-secrets/*, the request ID is *n32hcbnejk2*, and the target endpoint is Azure.

Preparing the data broker group

Prepare the data broker group to fetch credentials from the external vault by modifying the local config file for each data broker in the group.

Steps

1. SSH to a data broker in the group.

2. Edit the local.json file that resides in /opt/netapp/databroker/config.

3. Set enable to **true** and set the config parameter fields under *external-integrations.hashicorp* as follows:

   enabled
   - Valid values: true/false

- Type: Boolean

- Default value: false

- True: The data broker gets secrets from your own external HashiCorp Vault

- False: The data broker stores credentials in its local vault

url

- Type: string

- Value: The URL to your external vault

path

- Type: string

- Value: Prefix path to the secret with your credentials

Reject-unauthorized

- Determines if you want the data broker to reject unauthorized external vault

- Type: Boolean

- Default: false

auth-method

- The authentication method that the data broker should use to access credentials from the external vault

- Type: string

- Valid values: "aws-iam" / "role-app" / "gcp-iam"

role-name

- Type: string

- Your role name (in case you use aws-iam or gcp-iam)

Secretid & rootid

- Type: string (in case you use app-role)

Namespace

- Type: string

- Your namespace (X-Vault-Namespace header if needed)

4. Repeat these steps for any other data brokers in the group.

Example for aws-role authentication

```
{
        "external-integrations": {
                "hashicorp": {
                        "enabled": true,
                        "url": "https://example.vault.com:8200",
                        "path": ""my-path/all-secrets",
                        "reject-unauthorized": false,
                        "auth-method": "aws-role",
                        "aws-role": {
                                "role-name": "my-role"
                        }
                }
        }
}
```

Example for gcp-iam authentication

```
{
"external-integrations": {
    "hashicorp": {
      "enabled": true,
      "url": http://ip-10-20-30-55.ec2.internal:8200,
      "path": "v1/secret",
      "namespace": "",
      "reject-unauthorized": true,
      "auth-method": "gcp-iam",
      "aws-iam": {
        "role-name": ""
      },
      "app-role": {
        "root_id": "",
        "secret_id": ""
      },
"gcp-iam": {
        "role-name": "my-iam-role"
      }
    }
  }
}
```

Setting up permissions when using gcp-iam authentication

If you're using the *gcp-iam* authentication method, then the data broker must have the following GCP permission:

```
    - iam.serviceAccounts.signJwt
```

Learn more about GCP permission requirements for the data broker.

Creating a new sync relationship using secrets from the vault

Now that everything is set up, you can send an API call to create a sync relationship that uses your vault to get the secrets.

Post the relationship using the Cloud Sync REST API.

```
Headers:
Authorization: Bearer <user-token>
Content-Type: application/json
x-account-id: <accountid>
x-netapp-external-request-id-src: request ID as part of path for source
credentials
x-netapp-external-request-id-trg: request ID as part of path for target
credentials
Body: post relationship v2 body
```

- To obtain a user token and your Cloud Central account ID, refer to this page in the documentation.
- To build a body for your post relationship, refer to the relationships-v2 API call.

Example

Example for the POST request:

```
url: https://api.cloudsync.netapp.com/api/relationships-v2
headers:
"x-account-id": "CS-SasdW"
"x-netapp-external-request-id-src": "hb312vdasr2"
"Content-Type": "application/json"
"Authorization": "Bearer eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCIsImtpZCI6Ik…"
Body:
{
"dataBrokerId": "5e6e111d578dtyuu1555sa60",
"source": {
        "protocol": "s3",
        "s3": {
                "provider": "sgws",
                "host": "1.1.1.1",
                "port": "443",
                "bucket": "my-source"
    },
"target": {
        "protocol": "s3",
        "s3": {
                "bucket": "my-target-bucket"
        }
    }
}
```