Configure XCP

XCP

NetApp
March 14, 2024

Table of Contents

Configure XCP

Configure the INI file

Steps to configure the INI file for XCP.

Configure the INI file for a root user

You can use the following procedure to configure the INI file for an XCP NFS root user.

Steps

1. Add the catalog location for the XCP server in the host configuration file by using the `vi` editor:

> Catalog location should be exported before modifying the details in the `xcp.ini` XCP configuration file. Catalog location (NFSv3) should be mountable by the XCP Linux host but not necessarily be mounted.

```
[root@localhost /]# vi /opt/NetApp/xFiles/xcp/xcp.ini
```

2. Verify that the XCP Linux client host configuration file entries for the catalog were modified:

```
[root@localhost /]# cat /opt/NetApp/xFiles/xcp/xcp.ini
# Sample xcp config
[xcp]
catalog = 10.61.82.210:/vol/xcpvol/
```

Configure the INI file for a non-root user

As a non-root user, you do not have permission to mount the NFS file system. A root user is required to first mount the catalog volume and then, as a non-root user running XCP, if you have read/write permission to the catalog volume, you can access the mounted catalog volume by using a POSIX connector. After the volume is mounted, you can add catalog the path:

```
(t/10.237.170.53_catalog_vol - This is the path where catalog volume is
mounted)as follows.

[user1@scspr2474004001 xcp]$ ls -ltr
total 8
drwxrwxr-x 2 user1 user1  21 Sep 20 02:04 xcplogs
-rw-rw-r-- 1 user1 user1  71 Sep 20 02:04 xcp.ini
-rwxr-xr-x 1 user1 user1 352 Sep 20 02:10 license
[user1@scspr2474004001 xcp]$ cat /home/user1/NetApp/xFiles/xcp/xcp.ini

Sample xcp config [xcp]
catalog = file:///t/10.237.170.53_catalog_vol
```

Performance tuning

For XCP NFS, after planning the migration by using the `show` and `scan` commands, you can migrate data.

> When you are performing data migration as a non-root user, a root user can perform the following step.

For the optimal performance and reliability, NetApp recommends setting the following Linux kernel TCP performance parameters in `/etc/sysctl.conf` on the XCP Linux client host. Run `sysctl -p` or the `reboot` command to commit the changes:

```
net.core.rmem_default = 1342177
net.core.rmem_max = 16777216
net.core.rmem_max = 16777216
net.core.wmem_default = 1342177
net.core.wmem_max = 16777216
net.ipv4.tcp_rmem = 4096 1342177 16777216
net.ipv4.tcp_wmem = 4096 1342177 16777216
net.core.netdev_max_backlog = 300000
net.ipv4.tcp_fin_timeout = 10
```

> For a non-root user, the setting must be performed by a root user.

Environment variable

Optional environment variable configuration for XCP NFS systems.

> A non-root user can also use the following variables.

The environment variable `XCP_CONFIG_DIR` overrides the default location, `/opt/NetApp/xFiles/xcp`. If set, the value should be an OS filesystem path, possibly to a mounted NFS directory. When the `XCP_CONFIG_DIR` variable is set, a new directory with the same name as the host name is created inside the custom configuration directory path, new logs are stored at this location.

```
[root@localhost /]# export XCP_CONFIG_DIR ='/tmp/xcp_config_dir_path'
```

The environment variable `XCP_LOG_DIR` overrides the default location that stores the XCP log in the configuration directory. If set, the value should be an OS filesystem path, possibly to a mounted NFS directory. When the `XCP_LOG_DIR` variable is set, a new directory with the same name as the host name is created inside the custom log directory path, new logs are stored at this location.

```
[root@localhost /]# export XCP_LOG_DIR='/tmp/xcp_log_dir_path'
```

The environment variable `XCP_CATALOG_PATH` overrides the setting in xcp.ini. If set, the value should be in the xcp path format, `server:export[:subdirectory]`.

```
[root@localhost /]# export XCP_CATALOG_PATH='10.61.82.210:/vol/xcpvol/'
```

> For a non-root user, you must replace `XCP_CATALOG_PATH` from the exported path with the POSIX path.

Configure the POSIX connector

XCP NFS now supports the use of POSIX connectors to provide source and destination paths for data migration.

Supported features

The following features are supported for POSIX connectors:

- For POSIX file systems that support nanosecond `atime`, `mtime`, and `ctime`, the `scan` command gets the full values (seconds and nanoseconds) and the `copy` command sets them
- POSIX connectors are more secure than XCP with NFSv3 TCP sockets.

Path Syntax

The path syntax for a POSIX connector is `file:///<mounted path on linux>`.

Set up a POSIX connector

To set up a POSIX connector, you must perform the following tasks:

- Mount a source and a destination volume
- Verify that the destination path has the necessary permission to write the data

A destination and a catalog are mounted in the following example:

```
root@scspr2395903001 ~]# findmnt -t nfs4
TARGET SOURCE FSTYPE OPTIONS
/t/10.237.170.39_src_vol 10.237.170.39:/source_vol nfs4
rw,relatime,vers=4.0,rsize=65536,wsize=65536,namlen=255,hard,proto=t
cp,timeo=600,retrans=2,sec=sys,clien
/t/10.237.170.53_dest_vol 10.237.170.53:/dest_vol nfs4
rw,relatime,vers=4.0,rsize=65536,wsize=65536,namlen=255,hard,proto=t
cp,timeo=600,retrans=2,sec=sys,clien
/t/10.237.170.53_catalog_vol 10.237.170.53:/xcp_catalog nfs4
rw,relatime,vers=4.0,rsize=65536,wsize=65536,namlen=255,hard,proto=t
cp,timeo=600,retrans=2,sec=sys,clien
[root@scspr2395903001 ~]#
```

POSIX connectors access a source and destination volume by using the POSIX syntax `file://`. In the above example, the source path is `file:///t/10.237.170.39_src_vol` and the destination path is `file:// /t/10.237.170.53_dest_vol`.

You can manage the sample configuration of an XCP catalog shared by non-root users by creating a Linux group for XCP users. For non-root users, the following the permissions are required for Linux group users to perform migrations.

In the following sample output, `demo` is the non-root user and `/mnt/xcp-catalog` is the path where catalog volume is mounted:

```
sudo groupadd -g 7711 xcp_users
sudo usermod -G xcp_users -a demo
sudo chown -R :xcp_users /mnt/xcp-catalog
sudo chmod -R g+w /mnt/xcp-catalog
```

The XCP catalog does not store data but it does store scan and copy file names, directory names, and other metadata. Therefore, it is recommended that you configure the catalog file system permissions for allowed users to give them the capability to secure the stored metadata.

Ownership (UID and GID)

When you are set up as a regular user, by default, a `copy` command to a POSIX or NFS3 destination does not attempt to set the ownership (user ID (UID) and group ID (GID)). Setting the ownership is typically performed by an administrator. When user A copies files from user B, user A expects to own the destination. However, this is not the case when a root user copies the files. When a root user copies the files, the `-chown` option changes the behavior so that a non-root `copy` command with `-chown` attempts to set the UID and GID.

Increase the maximum number of open file descriptors

For optimal performance and reliability, you can increase the maximum number of open file descriptors for the XCP user on all nodes.

Steps

1. Open the file by using the following command:
   `vi /etc/security/limits.conf`
2. Add the following line to the file:
   `<username> - nofile 999999`

Example

```
root - nofile 999999
```

See Red Hat solutions for more information.

Configure the HDFS connector

For XCP NFS, the Hadoop Distributed File System (HDFS) connector (hdfs://) gives XCP the capability to access any HDFS file system that is available with different vendors.

Supported features

The `copy` command operation from HDFS to NFS is supported for HDFS connectors.

Path Syntax

The path syntax for a HDFS connector is `hdfs://[user@host:port]/full-path`.

> If you do not specify a user, host, and port, XCP calls `hdfsConnect` with the host set to `default` and the port set to `0`.

Set up a HDFS connector

To run the HDFS `copy` command, you must set the HDFS client on the Linux system, and based on the Hadoop vendor, follow the setup configuration available on the internet. For example, you can set the client for a MapR cluster by using `https://docs.datafabric.hpe.com/60/AdvancedInstallation/SettingUptheClient-redhat.html`.

After you complete the HFDS client setup, you must complete the configuration on the client. To use the HDFS paths with XCP commands, you must have the following environment variables:

- NHDFS_LIBHDFS_PATH
- NHDFS_LIBJVM_PATH

In the following examples, the settings work with MapR and java-1.8.0-openjdk-devel on CentOS:

```
export JAVA_HOME=$(dirname $(dirname $(readlink $(readlink $
(which javac)))))
export NHDFS_LIBJVM_PATH=`find $JAVA_HOME -name "libjvm.so"` export
NHDFS_LIBHDFS_PATH=/opt/mapr/lib/libMapRClient.so
```

```
[demo@mapr0 ~]$ hadoop fs -ls Found 3 items
drwxr-xr-x - demo mapr 0 2021-01-14 00:02 d1
drwxr-xr-x - demo mapr 0 2021-01-14 00:02 d2
drwxr-xr-x - demo mapr 0 2021-01-14 00:02 d3
```

Configure multinode scale-out

For XCP NFS, you can overcome the performance limits of a single node by using a single `copy` (or `scan -md5`) command to run workers on multiple Linux systems or cluster nodes.

Supported features

Multinode scale-out is helpful in any environment where the performance of a single system is not sufficient, for example, in the following scenarios:

- When it takes months for a single node to copy petabytes of data
- When high latency connections to cloud object stores slows down an individual node
- In large HDFS cluster farms where you run a very large number of I/O operations

Path syntax

The path syntax for multinode scale-out is `--nodes worker1,worker2,worker3`.

Set up multinode scale-out

Consider a setup with four Linux hosts with similar CPU and RAM configurations. You can use all four hosts for migration because XCP can coordinate the copy operations across all the host nodes. To make use of these nodes in a scale-out environment, you must identify one of the four nodes as the master node and other nodes as worker nodes. For example, for a Linux four-node setup, name the nodes as "master", "worker1", "worker2", and "worker3" and then set up the configuration on the master node:

1. Copy XCP in the home directory.
2. Install and activate the XCP license.
3. Modify the `xcp.ini` file and add the catalog path.
4. Set passwordless Secure Shell (SSH) from the master node to the worker nodes:
   a. Generate the key on the master node:

      ```
      ssh-keygen -b 2048 -t rsa -f /root/.ssh/id_rsa -q -N ''
      ```

   b. Copy the key to all the worker nodes:

      ```
      ssh-copy-id -i /root/.ssh/id_rsa.pub root@worker1
      ```

The XCP master node uses SSH to run workers on other nodes. You must configure the worker nodes to enable passwordless SSH access for the user running XCP on the master node. For example, to enable a user demonstration on a master node to use node "worker1" as an XCP worker node, you must copy XCP binary from the master node to all the worker nodes in the home directory.

MaxStartups

When you start up multiple XCP workers simultaneously, to avoid errors, you should increase the `sshd MaxStartups` parameter on each worker node as shown in the following example:

```
echo "MaxStartups 100" | sudo tee -a /etc/ssh/sshd_config
sudo systemctl restart sshd
```

The "nodes.ini" file

When XCP runs a worker on a cluster node, the worker process inherits the environment variables from the main XCP process on the master node. To customize a particular node environment, you must set the variables in the `nodes.ini` file in the configuration directory only on the master node (worker nodes do not have a configuration directory or catalog). For example, for an ubuntu server mars that has its `libjvm.so` in a different location to the master node, such as wave (which is CentOS), it requires a configuration directory to allow a worker on mars to use the HDFS connector. This setup is shown in the following example:

```
[schay@wave ~]$ cat /opt/NetApp/xFiles/xcp/nodes.ini [mars]
NHDFS_LIBJVM_PATH=/usr/lib/jvm/java-8-openjdk-amd64/jre/lib/
amd64/server/libjvm.so
```

If you are using a multisession with POSIX and HDFS file paths, you must mount the file system and the source and destination exported file system on the master node and all worker nodes.

When XCP runs on a worker node, the worker node has no local configuration (no license, log files, or catalog). XCP binary only is required on the system in your home directory. For example, to run the `copy` command, the master node and all worker nodes need access to the source and destination. For `xcp copy --nodes linux1,linux2 hdfs:///user/demo/test file:///mnt/ontap`, the `linux1` and `linux2` hosts must have the HDFS client software configured and the NFS export mounted on /mnt/ontap, and, as mentioned previously, a copy of the XCP binary in the home directory.

Combine POSIX and HDFS connectors, multinode scale-out, and security features

You can use the POSIX and HDFS connectors, multinode scale-out, and security features in combination. For example, the following `copy` and `verify` commands combine POSIX and HDFS connectors with the security and scale-out features:

- `copy` command example:

  ```
  ./xcp copy hdfs:///user/demo/d1 file:///mnt/nfs-server0/d3
  ./xcp copy -match "'USER1 in name'" file:///mnt/nfs-server0/d3
  hdfs:///user/demo/d1
  ./xcp copy —node worker1,worker2,worker3 hdfs:///user/demo/d1
  file:///mnt/nfs-server0/d3
  ```

- `verify` command example:

  ```
  ./xcp verify hdfs:///user/demo/d2 file:///mnt/nfs-server0/d3
  ```

Configure the S3 connector

Beginning with XCP 1.9.2, the Simple Storage Service (S3) connector enhances the scope of XCP data migration by enabling data migration from Hadoop Distributed File System (HDFS) file systems to S3 object storage.

Supported migration use cases

The following migration use cases are supported for the S3 connectors:

- Migration from HDFS to NetApp StorageGRID
- Migration from HDFS to Amazon S3
- Migration from HDFS to NetApp ONTAP S3

> Currently MapR is only qualified and supported for HDFS.

Supported features

Support for the `scan`, `copy`, `verify`, `resume` and `delete` commands is available for the S3 connectors.

Unsupported Features

Support for the `sync` command is not available for the S3 connectors.

Path Syntax

The path syntax for the S3 connector is `s3://<bucket in S3>`.

- You can provide a specific S3 profile for the XCP commands using the `-s3.profile` option.
- You can use the `s3.endpoint` option to modify the endpoint value to communicate with S3

> Endpoint usage is mandatory for StorageGRID and ONTAP S3.

Set up an S3 connector

Steps

1. To run the XCP command with the S3 connector, create a bucket in S3 by following the online documentation for the respective platforms:
   - ONTAP S3 object storage management
   - StorageGRID: Use a tenant account overview

   > Before continuing, you must have the `access key`, `secret key`, certificate authority (CA) certificate bundle, and `endpoint url` information. XCP identifies and connects to the S3 bucket using these parameters before initiating an operation.

2. Install the Amazon Web Services (AWS) CLI packages and run the AWS CLI commands to configure the keys and Secure Sockets Layer (SSL) certificates for S3 accounts:
   - See Installing or updating the latest version of the AWS CLI to install the AWS packages.
   - See the AWS CLI Command Reference for more information.
3. Use the `aws configure` command to configure your credentials file. By default, the file's location is

`/root/.aws/credentials`. The credentials file should specify the access key and secret access key.

4. Use the `aws configure set` command to specify a CA certificate bundle, which is a file with the `.pem` extension that is used when verifying SSL certificates. By default, the file's location is `/root/.aws/config`.

   Example:

```
[root@client1 ~]# aws configure
AWS Access Key ID [None]: <access_key>
AWS Secret Access Key [None]: <secret_key>
Default region name [None]:
Default output format [None]:
[root@client1 ~]# cat /root/.aws/credentials
[default]
aws_access_key_id = <access_key>
aws_secret_access_key = <secret_key>
[root@client1 ~]#
[root@client1 ~]# aws configure set default.ca_bundle
/u/xxxx/s3/ca/aws_cacert.pem
[root@client1 ~]# cat /root/.aws/config
[default]
ca_bundle = /u/xxxx/s3/ca/aws_cacert.pem
```

5. After the required setup configuration is completed, confirm that the AWS CLI commands can access the S3 buckets from the Linux client before running the XCP commands:
   `aws s3 ls --endpoint-url <endpoint_url> s3://bucket-name/`

   `aws s3 ls --profile <profile> --endpoint-url <endpoint_url> s3://bucket-name`

   Example:

```
[root@client1 linux]# aws s3 ls --profile <profile> --endpoint
<endpoint_url>  s3://<bucket-name>
                           PRE 1G/
                           PRE aws_files/
                           PRE copied_folders/
                           PRE d1/
                           PRE d2/
                           PRE giant_size_dirs/
                           PRE medium_size_dirs/
                           PRE small_size_dirs/
[root@client1 l
```