Backup and restore data procedures

On this page

- Cold Backup/Restore
- Hot Backups
  - Backups
  - Restore
- Backup and restore scripts
  - backup.sh
  - restore-single_node_311.sh
  - On Linux

Scripts

The following are the script files used in this installation:

- backup.sh
- backup_all.sh
- restore-single_node_311.sh

This page provides the instructions for cold and hot backups, and the scripts required to perform hot backup and restore of the Cassandra database. These backup and restore scripts should be executed on a Linux server where Cassandra 4 is running.

Cold Backup/Restore

A cold backup is a backup taken via regular system backups, while the database is shut down in a consistent state. Once the database is shut down in a consistent state, we back up the directories pointed to by the **data_file_directories** and **commitlog_directory** values in your **cassandra.yaml** file.

Backup procedure consists of:

- Stop the WebApp service
- Stop the Teamwork Cloud service
- Stop the Authserver service
- Commit all the data to disk by issuing the "nodetool drain" command
- Stop the Cassandra service
- Perform the backup (this can be a VM snapshot, a backup of the specified directories using any backup software, or copying the directories to another location)
- Start the Cassandra service
- Start the Teamwork Cloud service
- Start the Authserver service
- Start the WebApp service

Restore procedure consists of:

- Stop the WebApp service
- Stop the Teamwork Cloud service
- Stop the Authserver service
- Stop the Cassandra service
- Delete the **data_file_directories** and **commitlog_directory**
- Restore the data from the backup to the original location
- Start the Cassandra service
- Start the Teamwork Cloud service
- Start the Authserver service
- Start WebApp service

Hot Backups

Backups

Hot backups are backups which are taken while the database is running, therefore eliminating the need for downtime. This is accomplished by using Cassandra's internal ability to perform a snapshot of the database. The backup process consists of flushing data to disk, stopping any compactions which may be taking place, creating a snapshot (the snapshot creates a set of hard links to the current immutable files), archiving the state of the immutable files to a backup location, and clearing the snapshots directories created in the database. For disaster recovery, this directory should be located on a different physical drive than where the data resides. Each snapshot will consume approximately the same amount of space as the data directory. The backup script does not perform any directory maintenance or purging. This can be done either by using a wrapper script which will delete the current backup before creating the new snapshot, or by a scheduled task which is to be executed before the backup. If point in time recovery is desired, the backup directory can be backed up as part of the regular backup procedures for the server.

When backing up a cluster, all nodes must be backed up individually, preferably at the same time. **Also, on a cluster, it is imperative that the clocks on all machines be synchronized.**

Hot backups should be executed at a time when there is no client activity, in order to ensure model consistency. If you cannot be sure that there is no client traffic at the time of the snapshot, please stop the Teamwork Cloud service momentarily. Once the "nodetool snapshot" command has completed, restart the Teamwork Cloud service.

Restore

The restore procedure will stop the Cassandra database, delete the commit logs and the data files in the repository, restore the snapshot files and relocate them back to their source directories. Once this is done, Cassandra will be started and the nodetool utility will be invoked to repair the keyspaces.

When restoring a cluster, all nodes must be restored from their individual backups (backups taken at the same time). You must first stop the Teamwork Cloud, authserver, and Cassandra service on each of the nodes. Once the Cassandra nodes are off, proceed to restore the seed node. Once you have completed the restoration of the seed node, proceed to restore the second node. Upon completion of the second node, restore the third node. During the restoration process of the first 2 nodes, you may receive an error from the keyspace repair process - this is normal since some of the nodes have not joined the cluster. The last node to be restored will take care of the keyspace repairing process.

Backup and restore scripts

To backup and restore Cassandra database, you need the following files. Click the file to download it.

backup.sh

restore-single_node_311.sh

backup.sh

This script creates snapshots of Cassandra database. The script must run on the host running Cassandra and Cassandra must be running. The Cassandra bin directory or nodetool must be added to **$PATH** (on Linux, nodetool is available when the Cassandra package is installed without further modifications). The backup script preserves the database file permissions and owner. On Linux, you should be running on a shell with root permissions.

The command format for backing up is:

```
./backup.sh -dir CASSANDRA_DB_PATH -rf BACKUP_DIR
```

where **CASSANDRA_DB_PATH** is the directory below where the Cassandra data is located, and **BACKUP_DIR** is the location in which the snapshot archives are to be saved.

The archive is named using a pattern **cassandra_backup_yyyy.mm.dd-hh.mm.ss.tar**

If any of the parameters are omitted then the script will prompt you for the locations, and perform sanity checks to make sure the values are correct.

Path names

When calling the backup/restore scripts passing parameters, if your path names contain spaces, they must be quoted.

For example:

```
./backup.sh -dir "/path to data containing spaces/data" -rf "/path to backups containing spaces/"
```

restore-single_node_311.sh

This script should be run as root on Linux and Cygwin console must be launched with the administrator privileges. This script restores data from the backup file. While restoring Cassandra, Teamwork Cloud server must be turned off. The script stops and starts Cassandra if needed. The backup script preserves the database file permissions and owner, and then restores backup on another machine with another user. Therefore, the file permissions must be changed manually otherwise Cassandra may not start. The following variables may be passed onto the script:

```
Usage: ./restore-single_node_311.sh [--dir <arg>] [--commitlog <arg>] [--rf <arg>] [--cassandra <arg>] [-h|--
help]
--dir: cassandra database path (default:cassandra_database)
--commitlog: commitlog path (default: commitlog_path)
--rf: full path to snapshot archive (default: rf)
--cassandra: cassandra installation path when cassandra is not executed as a service (default: cassandra)
-h,--help: Prints help
```

Even though parameters can be passed to the restore script, restores are typically carried out in an interactive manner.

Variable	Description
commitlog_path	The path of commitlog files. They are removed when restoring database to prevent recovery of commits made earlier. DataStax recommends to store the commitlog files in a separate hard-disk for performance sake.
cassandra_database	The Cassandra database path. It is used to clean existing database and restoring database from the recovery file.
cassandra	The Cassandra installation path. It is the required parameter when Cassandra is launched not as service.
service	The Boolean parameter indicating that Cassandra is started as a service. The default value is true so you need to set this variable to *false* when Cassandra is launched by the other script or manually.

rf	The recovery file with the full path to it.

On Linux

Type:

```
$ su ./restore-single_node_311.sh
```

or

```
$ sudo su
$ ./restore-single_node_311.sh
```

If you are logged in as root, the command is:

```
$ ./restore-single_node_311.sh
```

Related pages

- Data manager