Trafodion Blueprint Design Document:
Backup and Restore (Technology Preview)

Title	Backup and Restore (Technology Preview)
Date	January 2015
Author	Trafodion SQL Team
Audience	Open Source community
Abstract	This blueprint describes full offline backup/restore of Trafodion using HBase snapshots. **NOTE:** This is a *Technology Preview (Complete But Not Tested)* feature, meaning that it is functionally complete but has not been tested or debugged. For more information about what *Technology Preview* means, see the Technology Preview Features page on the Trafodion wiki.

Document History

Document Version	Date	Changes
1.0	12/03/2014	• Version to be published on the wiki

Legal Notice

Table of Contents

Full offline backup/restore of Trafodion

Full offline backup/restore of Trafodion can be performed using HBase snapshots. HBase snapshots for all the Trafodion tables, including the Trafodion metadata tables, are taken and exported to an HDFS location. Afterwards, the backup files in the HDFS location can be archived and saved somewhere safe if needed. When we need to restore the backup files from the HDFS location, we import the snapshots to the target system and then restore them. This process is applied to each Trafodion table. The process to do the backup/restore for one table is as follows:

- To do a backup of a table, a snapshot of the underlying HBase table is taken using the HBase shell snapshot command.
- Once the snapshot is taken, it is exported it to an HDFS folder using the ExportSnapshot MapReduce job that is part of HBase install base. The MapReduce job can be run in parallel.
- If needed, the backup files can be moved to a different cluster using hadoop distcp or other methods.
- To restore the table data, the snapshot we saved in HDFS is imported to HBase using the same MapReduce job (ExportSnapshot). The MapReduce job can be run in parallel.
- Once the snapshot is imported, we restore the snapshot using the 'restore_snapshot' HBase shell command.

Trafodion tables

Trafodion tables are all under the TRAFODION catalog. The backup process selects and backs up all tables that have TRAFODION as their catalog. Trafodion tables in HBase have this name pattern: TRAFODION.<schema-name>.<table-name>.

Backup files security

The backup is done using the Trafodion user privileges and the HDFS backup files/folders created and owned by the Trafodion user. On the cluster, the backup files have permissions like (drwxr-xr-x). These permissions are usually controlled by the HDFS umask property.

The restore phase needs to have read access to the backup files in HDFS and also write access to the target HBase folders on the system to which we are restoring.

The restore phase is run using the HBase user (the user under which HBase runs), and the HBase user needs to have at least read access to the backup files.

To grant the HBase user read access to the backup files, we have several options:

- Create the backup files with the default permissions (drwxr-xr-x), which gives read access to everyone. Since the trafodion user is not in the same group as the HBase user, we need to give read access to others, which is everyone. (HBase is a member of everyone.)
- Make HBase and Trafodion part of the same group and give permission only to the group. Others can be excluded
- Use HDFS Access Control Lists (ACL) (available with Hadoop 2.4 and HBase 0.98), which give much better control over who can access what. Here is the link for HDFS ACLs:

  http://hadoop.apache.org/docs/r2.5.0/hadoop-project-dist/hadoop-hdfs/HdfsPermissionsGuide.html#ACLs_Access_Control_Lists

Configurations, issues and workarounds

General issues

The backup and restore phases check whether Trafodion is down or not and give an error if it is not.

Issues with Cloudera distribution

Cloudera distribution configures all the Hadoop user IDs so that you cannot log in to them directly. To run the restore scripts under the HBase user, we can do one of the following:

- Manually change these settings by editing the /etc/passwd or /usr/passwd file:

  Change this line:

  ```
  HBase:x:487:483:HBase:/var/run/HBase:/sbin/nologin
  ```

  To this line (changes highlighted):

  ```
  HBase:x:487:483:HBase:/var/run/HBase:/bin/bash
  ```

- Issue `su -s /bin/bash HBase` to start a shell under the HBase user. (We did not test this approach.)
- Issue `su HBase --command "<insert-HBase-command-here>"`. (We did not test this approach.)

We also need to run the following commands to create an HBase folder under /user:

```
sudo -u hdfs hadoop fs -mkdir /user/HBase
sudo -u hdfs hadoop fs -chown HBase:HBase /user/HBase
```

Testing

The test cases that we performed so far include:

- Backup/restore within a single system with Trafodion 0.9 and HBase 0.98 on a single system to an HDFS location, removing all the Trafodion tables and then restoring them from the backup location to the same HBase 0.98. This type of tests was done on:
  - Workstation with Trafodion 0.9 and HBase 0.98 with a Cloudera distribution (development environment). Test succeeded.
  - Cluster with four nodes that has Trafodion 0.9 and HBase 0.98 (Hortonworks distribution). Test succeeded.
  - Cluster with two nodes that has Trafodion 0.9 and HBase 0.98 (Cloudera distribution). Test succeeded.
  - The issues faced with these tests are mainly related to file permissions and privileges.
- Backup/Restore using two different systems, both having Trafodion 0.9 and HBase 0.98:
  - The steps involved in these tests are:
    - Backup of the source Trafodion to an HDFS location
    - Moving the HDFS folder to another system with HBase 0.98. This step involved copying the backup files from HDFS to the local disk, then using the scp linux command to move the files to the target system, and then copying the backup files to an HDFS location on the target system. Another way to move the backup

files is to use hadoop distcp, but currently it does not work because the IP ports associated with distcp are not open.

        - Restoring the backup on the target system from the backup files

    - These types of tests were done:

        - From a cluster with four nodes and the Hortonworks distribution to a cluster with two nodes and Cloudera distribution. Tests succeeded.

        - From a workstation to another workstation (development environment), both having HBase from a Cloudera distribution. Tests succeeded.

    - The issues faced with these tests are mainly related to file permissions and privileges.

- Backup/Restore using two different systems, both having Trafodion 0.9, where the source system has HBase 0.94.6 and the target system has HBase 0.98.

    - Backup of the HBase 0.94.6: while working on this test case, we faced a few issues that can be summarized as follows:

        - File permissions on the HBase and HDFS system do not seem to be correct and the Export of Snapshots to HDFS did not work in the beginning. The solution we used to resolve the permission issues was to disable permission checking. The issues could be caused by a configuration issue with the cluster (rhel-ah1.hpl.hp.com) that we are using.

        - Exporting Snapshot that is empty (table is empty) from HBase 0.94.6 fails and generates an exception. This is a known issue, which is described in https://issues.apache.org/jira/browse/HBASE-8199. This export issue was fixed in HBase 0.94.7 but was not in HBase0.94.6.

        - There are compatibility issues between HBase in general (including snapshots) and between HBase 0.94.6 and later versions, including 0.98. Because of this issue, the import of the HBase 0.94.6 snapshot to HBase 0.98 does not work.

        - One way to solve this compatibility issue may be to upgrade from HBase 0.94.6 to 0.98.x before doing the backup if we want to restore to 0.98 HBase. The upgrade itself from HBase 0.94.6 to HBase 0.98 needs to happen in two steps: (1) upgrade from HBase 0.94.6 to HBase 0.96, and then (2) upgrade from HBase 0.96 to HBase 0.98. The upgrade steps are described here: http://HBase.apache.org/book/ch03s04.html

        - This test case is still work in progress because of the issues listed above and may require a lot of work if we want to continue working on it.

    - Restore to an HBase 0.98 system: This was not done because of the issues mentioned above.

Scripts

There are two main scripts to perform the offline backup/restore operations. To obtain those scripts in zipped format, click here:

Full_offline_backup_restore_scripts.zip

Backup script: run_full_trafodion_backup.sh

The help from this script is shown below:

```
This script may need to run under trafodion user
  ./run_full_trafodion_backup.sh -b backup_folder -r hbase_root_dir -m mappers
  -b  backup_folder  : folder where the ALL the  Trafodion object are exported for
backup
  -m  mappers        : number of mappers.
 Example: ./run_full_trafodion_backup.sh  -b hdfs://localhost:31400/bulkload/backup  -
m 4
 some systems require the hdfs://<host>:<port> to be part of the URI otherwhise export
fails
```

Restore: run_full_trafodion_backup.sh.

The help from this script is shown below:

```
This script may need to run under hbase user
  ./run_full_trafodion_restore.sh -b backup_folder -r hbase_root_dir -m mappers
  -b  backup_folder  : folder where the ALL the  Trafodion object were exported
  -r  hbase_root_dir : hbase root directory. usually this is in hbase-site.xml with
                       hbase.root.dir property. in Cloudera 5.1 this value is /hbase
                       by default. And for HortonWorks it is set to /apps/hbase/data
                       by default.
  -m  mappers        : number of mappers.
 some systems require the hdfs://<host>:<port> to be part of the URI otherwhise export
fails
 Example: ./run_full_trafodion_restore.sh  -b hdfs://localhost:31400/bulkload/backup -
r hdfs://localhost:31400/hbase -m 4
```