Cloudera Runtime ..

Sqoop Troubleshooting

Date published: 2023-05-16
Date modified:

Legal Notice

Contents

Unable to read Sqoop metastore created by an older HSQLDB version

After upgrading to certain Cloudera versions, you may encounter issues in reading the Sqoop metastore that was created using an older version of HyperSQL Database (HSQLDB). Learn how to upgrade the Sqoop metastore to resolve this issue.

Cloudera upgraded the HSQLDB dependency in the following versions to the newest HSQLDB 2.7.1 version:

- CDP Private Cloud Base 7.1.7 SP2
- CDP Private Cloud Base 7.1.8 Cumulative hotfix 4
- CDP Public Cloud 7.2.16

The version upgrade causes incompatibility issues in Sqoop jobs that are stored in HSQLDB. Therefore, the Sqoop metastore should be upgraded after you upgrade to the above-mentioned Cloudera versions or higher.

Note: Sqoop stores its metastore files in the current user's home directory. If the Sqoop metastore is on-demand, then its location is ~/.sqoop/metastore.db.If the Sqoop metastore is a service, then its location is ~/.sqoop/shared-metastore.db.

Choose one of the following procedures based on how the Sqoop metastore is created:

For Sqoop metastore created using Sqoop

Perform the following steps if the Sqoop metastore is created using Sqoop:

1. If the Sqoop metastore is started as a service then stop the service by running this command on the host where the service is running:

```
/opt/cloudera/parcels/CDH/lib/sqoop/bin/stop-metastore.sh -p <path to
 Sqoop metastore pid file>
```

2. Download HSQLDB 2.3.6 and SqlTool JAR files to the host by running the following commands:

```
wget https://repo1.maven.org/maven2/org/hsqldb/sqltool/2.3.6/sqltool-2.3
.6.jar
wget https://repo1.maven.org/maven2/org/hsqldb/hsqldb/2.3.6/hsqldb-2.
3.6.jar
```

3. Run the following command to upgrade and convert the Sqoop metastore database files using SqlTool:

   For on-demand Sqoop metastore:

```
$JAVA_HOME/bin/java -cp "sqltool-2.3.6.jar:hsqldb-2.3.6.jar" org.hsqldb
.cmdline.SqlTool --driver org.hsqldb.jdbc.JDBCDriver  --inlineRc="url=jd
bc:hsqldb:file:~/.sqoop/metastore.db;shutdown=true;user=SA,password=" --
sql="SELECT * FROM SQOOP_ROOT;"
```

   For Sqoop metastore as a service:

```
$JAVA_HOME/bin/java -cp "sqltool-2.3.6.jar:hsqldb-2.3.6.jar" org.hsqldb.
cmdline.SqlTool --driver org.hsqldb.jdbc.JDBCDriver  --inlineRc="url=jdb
```

```
c:hsqldb:file:~/.sqoop/shared-metastore.db;shutdown=true;user=SA,passwor
d=" --sql="SELECT * FROM SQOOP_ROOT;"
```

The Sqoop metastore is upgraded and the database files are converted to a format that can easily be read by HSQLDB 2.7.1.

Note: As part of the Cloudera upgrade, Apache Sqoop installation has the dependent HSQLDB 2.7.1 version by default.

4. If the Sqoop metastore service was stopped earlier, restart the service:

```
/opt/cloudera/parcels/CDH/lib/sqoop/bin/start-metastore.sh -p <path to S
qoop metastore pid file> -l <path to Sqoop metastore logs dir>
```

5. You can choose to remove the downloaded HSQLDB JAR files from the host.

For Sqoop metastore created in a separate HSQLDB instance

Perform the following steps if the Sqoop metastore is created in a separate HSQLDB instance:

1. Login to the HSQLDB service host and stop the service. For more information, see the HSQLDB documentation.
2. Work with your IT team to determine the HSQLDB store file that is created to store the Sqoop metastore information.
3. Download HSQLDB 2.3.6 and SqlTool JAR files to the host by running the following commands:

```
wget https://repo1.maven.org/maven2/org/hsqldb/sqltool/2.3.6/sqltool-2.3
.6.jar
wget https://repo1.maven.org/maven2/org/hsqldb/hsqldb/2.3.6/hsqldb-2.
3.6.jar
```

4. Run the following command to upgrade and convert the Sqoop metastore database files using SqlTool:

```
$JAVA_HOME/bin/java -cp "sqltool-2.3.6.jar:hsqldb-2.3.6.jar" org.hsqldb.
cmdline.SqlTool --driver org.hsqldb.jdbc.JDBCDriver  --inlineRc="url=jdb
c:hsqldb:file:<PATH OF SQOOP METASTORE FILE>;shutdown=true;user=SA,pass
word=" --sql="SELECT * FROM SQOOP_ROOT;"
```

5. If the HSQLDB service hosts multiple databases then each of the store files must be upgraded as described in the previous step.

   The Sqoop metastore is upgraded and the database files are converted to a format that can easily be read by HSQLDB 2.7.1.

   Note: As part of the Cloudera upgrade, Apache Sqoop installation has the dependent HSQLDB 2.7.1 version by default.

6. Restart the HSQLDB service. For more information, see the HSQLDB documentation.
7. You can choose to remove the downloaded HSQLDB JAR files from the host.

Related Information

HyperSQL Documentation

Merge process stops during Sqoop incremental imports

During Sqoop incremental import operations, if the target directory is located outside of Hadoop Distributed File System (HDFS), such as in Amazon S3 or Azure Blob Storage, the merge phase of the import process does not take effect.

Condition

Sqoop, by default, creates temporary directories within HDFS. However, you must be aware of certain considerations in choosing the target directory location while working with Sqoop's incremental import modes. By default, Sqoop operates seamlessly when the target directory resides within HDFS. However, the merge phase of the import process does not take effect outside the box if the target directory is located outside of HDFS.

Cause

During an import operation, Sqoop generally imports data to a target directory. If this target directory is a non-HDFS location, the merge process tries to acquire the temporary directory required for the merge on the same non-HDFS file system. Since Sqoop creates the temporary directory in HDFS by default, the merge process checks if the temporary directory exists in the target directory's file system and when it does not find it, the merge process simply stops.

Solution

If the target directory is present outside of HDFS, you must modify the default path of the temporary directory by adding the --temporary-rootdir Sqoop option and pointing to a path on the same file where the target directory is located. By aligning the temporary directory path with the file system of the target directory, Sqoop can effectively complete the import process.

Example:

Include the --temporary-rootdir Sqoop option as shown below:

```
sqoop-import --connect jdbc:mysql://.../transaction --username [***USER NAME
***] --table [***TABLE NAME***] --password [***PASSWORD***] --target-dir abf
s://foo@bar/targetdir -m 1 --temporary-rootdir abfs://foo@bar/_sqoop
```

Sqoop Hive import stops when HS2 does not use Kerberos authentication

Learn how to resolve the issue related to Sqoop Hive imports when either LDAP authentication or no authentication mechanism is enabled for the cluster.

Condition

When running Sqoop commands to import data into Hive from either the CLI or Oozie, the import job stops after the Sqoop import is done and while trying to connect to HiveServer (HS2) through JDBC. The following log is displayed and you will notice that the job stops on the last line:

```
23/07/24 18:10:17 INFO hive.HiveImport: Loading uploaded data into Hive
23/07/24 18:10:17 INFO hive.HiveImport: Collecting environment variables whi
ch need to be preserved for beeline invocation
...
23/07/24 18:10:20 INFO hive.HiveImport: SLF4J: Actual binding is of type
[org.apache.logging.slf4j.Log4jLoggerFactory]
23/07/24 18:10:21 INFO hive.HiveImport: Connecting to jdbc:hive2://HOSTNAME/
default;serviceDiscoveryMode=zooKeeper;ssl=true;sslTrustStore=/var/lib/clo
udera-scm-agent/agent-cert/cm-auto-global_truststore.jks;trustStorePassword=
changeit;zooKeeperNamespace=hiveserver2
```

Cause

This issue occurs when Kerberos is not used in the JDBC connection string, which Sqoop uses to connect to HS2. The issue affects unsecure clusters and clusters where LDAP authentication is enabled, and the beeline-site.xml configuration file does not use Kerberos authentication.

The underlying issue is that Beeline prompts for the username and password for a successful connection and since the Sqoop Hive import is a non-interactive session, you are unable to provide the credentials and therefore the import job stops.

Solution

Procedure

Perform one of the following steps to resolve this issue:

If...	Then...
No authentication is enabled for the cluster	Include the --hs2-url option in the Sqoop import command and provide the JDBC connection string. `--hs2-url <HS2 JDBC string>` This allows for a successful connection without prompting for the credentials.
LDAP authentication is enabled for the cluster	Include the --hs2-user and --hs2-password options in the Sqoop import command and provide the credentials. `--hs2-user <username>` `--hs2-password <password>`

Sqoop jobs fail with unnamed module exception after JDK 17 upgrade

After upgrading to JDK 17, Sqoop jobs may fail due to strict enforcement of the Java Platform Module System (JPMS).

Condition

After upgrading to Cloudera Runtime 7.1.9 SP1 (or later) cluster to JDK 17, Sqoop jobs that previously worked on JDK 8 or JDK 11 begin to fail.

The failures can impact both Sqoop import and export jobs and might disrupt critical workflows.

Depending on the job type, the YARN application logs may show one or more of the following errors:

- IllegalAccessError
- InaccessibleObjectException
- Module access errors involving an unnamed module
- Failures referencing restricted Java packages such as:
  - java.lang
  - java.net
  - sun.net.dns
  - sun.net.util
  - com.sun.jmx.mbeanserver

In some cases:

- Sqoop import jobs fail immediately after the JDK 17 upgrade.
- Sqoop export jobs continue to fail even after the YARN Application Master JVM options are configured.

Cause

JDK 17 strictly enforces the Java Platform Module System (JPMS), which restricts access to internal JDK APIs that Sqoop and the underlying MapReduce framework rely on.

Sqoop was originally built for earlier Java versions and uses reflection to access these internal APIs. Under JDK 17, this access is blocked unless explicitly allowed using JVM options such as --add-opens and --add-exports.

Additionally:

- YARN Application Master JVM options (yarn.app.mapreduce.am.command-opts) do not automatically propagate to MapReduce task JVMs.
- For Sqoop export jobs, the map task JVM requires additional module access (for example, java.net) to interact with external databases.
- The YARN configuration Add add-opens flags to MR containers is disabled by default, regardless of the JDK or Cloudera version. When using JDK 17, required module flags must be explicitly provided to the relevant JVMs.

As a result, the import jobs fail due to restricted module access in the Application Master, and export jobs fail because the map task JVM does not receive all required module permissions.

Solution

Procedure

1. Configure YARN Application Master JVM options (required for all Sqoop jobs).

- Open Cloudera Manager. Cloudera Manager YARN Configuration
- Search for the property yarn.app.mapreduce.am.command-opts.
- Add the following JVM arguments:

```
--add-opens=java.base/java.lang=ALL-UNNAMED
--add-opens=java.management/com.sun.jmx.mbeanserver=ALL-UNNAMED
--add-exports=java.management/com.sun.jmx.mbeanserver=ALL-UNNAMED
--add-exports=java.base/sun.net.dns=ALL-UNNAMED
--add-exports=java.base/sun.net.util=ALL-UNNAMED
```

- Click Save Changes.
- Restart the YARN services.

2. Provide additional JVM options for Sqoop export jobs.

Sqoop export jobs require additional module access in the map task JVM, which is not covered by the Application Master configuration alone.

When running Sqoop export jobs, explicitly pass the required JVM options using the mapreduce.map.java.opts property, for example:

```
sqoop export \
  -D mapreduce.map.java.opts="--add-opens=java.base/java.lang=ALL-UNNAMED
 \
  --add-opens=java.management/com.sun.jmx.mbeanserver=ALL-UNNAMED \
  --add-exports=java.management/com.sun.jmx.mbeanserver=ALL-UNNAMED \
  --add-exports=java.base/sun.net.dns=ALL-UNNAMED \
  --add-exports=java.base/sun.net.util=ALL-UNNAMED \
  --add-opens=java.base/java.net=ALL-UNNAMED" \
  --connect ... \
  --table ... \
  --export-dir ...
```

This ensures that the map task JVM can access the required internal Java packages needed for database connectivity during export operations.

3. Verify the solution.

   Re-run the Sqoop import or export job and confirm that the job completes successfully without module-related exceptions.

   Verify in YARN logs that no IllegalAccessError or InaccessibleObjectException messages are present.