Replication

Overview

Hive Replication builds on the metastore event and ExIm features to provide a framework for replicating Hive metadata and data changes between clusters. There is no requirement for the source cluster and replica to run the same Hadoop distribution, Hive version, or metastore RDBMS. The replication system has a fairly 'light touch', exhibiting a low degree of coupling and using the Hive-metastore Thrift service as an integration point. However, the current implementation is not an 'out of the box' solution. In particular it is necessary to provide some kind of orchestration service that is responsible for requesting replication tasks and executing them.

See HiveReplicationDevelopment for information on the design of replication in Hive.

A more advanced replication mechanism is being implemented in Hive to address some of the limitations of this mode. See HiveReplicationv2Development for details.

Potential Uses

- Disaster recovery clusters.
- Copying data into clouds for off-premise processing.

Prerequisites

- You must be running Hive 1.1.0 or later at your replication source (for `DbNotificationListener` support).
- You must be running Hive 0.8.0 or later at your replication destination (for `IMPORT` support).
- You'll require Hive 1.2.0 or later JAR dependencies to instantiate and execute `ReplicationTasks`. This is not a cluster requirement; it is needed only for the service orchestrating the replication.
- You will initially require administration privileges on the source cluster to enable the writing of notifications to the metastore database.

Limitations

- While the metastore events feature allows the sinking of notifications to anything implementing `MetaStoreEventListener`, the implementation of the replication feature can only source events from the metastore database and hence the `DbNotificationListener` must be used.
- Data appended to tables or partitions using the HCatalogWriters will not be automatically replicated as they do not currently generate metastore notifications (HIVE-9577). This is likely only a consideration if data is being written to table by processes outside of Hive.

Configuration

To configure the persistence of metastore notification events it is necessary to set the following `hive-site.xml` properties on the source cluster. A restart of the metastore service will be required for the settings to take effect.

hive-site.xml Configuration for Replication

```
<property>
  <name>hive.metastore.event.listeners</name>
  <value>org.apache.hive.hcatalog.listener.DbNotificationListener</value>
</property>
<property>
  <name>hive.metastore.event.db.listener.timetolive</name>
  <value>86400s</value>
</property>
```

The system uses the `org.apache.hive.hcatalog.api.repl.exim.EximReplicationTaskFactory` by default. This uses `EXPORT` and `IMPORT` commands to capture, move, and ingest the metadata and data that need to be replicated. However, it is possible to provide custom implementations by setting the `hive.repl.task.factory` Hive configuration property.

Typical Mode of Operation

- With the metastore event configuration in place on the source cluster, the `NOTIFICATION_LOG` table in the metastore will be populated with events on the successful execution of metadata operations such as `CREATE`, `ALTER`, and `DROP`.
- These events can be read and converted into `ReplicationTasks` using `org.apache.hive.hcatalog.api.HCatClient.getReplicationTasks(long, int, String, String)`.
- `ReplicationTasks` encapsulate a set of commands to execute on the source Hive instance (typically to export data) and another set to execute on the replica instance (typically to import data). The commands are provided as Hive SQL strings.
- The `ReplicationTask` also serves as a place where database and table name mappings can be declared and `StagingDirectoryProvider` implementations configured for the resolution of paths at both the source and destination:
  - `org.apache.hive.hcatalog.api.repl.ReplicationTask.withDbNameMapping(Function<String, String>)`
  - `org.apache.hive.hcatalog.api.repl.ReplicationTask.withTableNameMapping(Function<String, String>)`
  - `org.apache.hive.hcatalog.api.repl.ReplicationTask.withSrcStagingDirProvider(StagingDirectoryProvider)`
  - `org.apache.hive.hcatalog.api.repl.ReplicationTask.withDstStagingDirProvider(StagingDirectoryProvider)`
- The Hive SQL commands provided by the tasks must then be executed against the source Hive and then the destination (aka the replica). One way of doing this is to open up a JDBC connection to the respective HiveServer and submit the task's Hive SQL queries.
- It is necessary to maintain the position within the notification log so that replication tasks are applied only once. This can be achieved by maintaining a record of the last successfully executed event's id (`task.getEvent().getEventId()`) and providing this as an offset when sourcing the next batch of events.
- To avoid losing or missing events that require replication, it may be wise to poll for replication tasks at a frequency significantly greater than derived from the `hive.metastore.event.db.listener.timetolive` property. If notifications are not consumed in a timely manner they may be purged from the table before they can be actioned by the replication service.

Replication to AWS/EMR/S3

At this time it is not possible to replicate to tables on EMR that have a path location in S3. This is due to a bug in the dependency of the `IMPORT` command in the EMR distribution (checked in AMI-4.2.0). Also, if using the `EximReplicationTaskFactory` you may need to add the relevant S3 protocols to your Hive configurations:

HiveConf Configuration for ExIm on S3

```
<property>
  <name>hive.exim.uri.scheme.whitelist</name>
  <value>hdfs,s3a</value>
</property>
```