Bulk Load Replication in HBASE

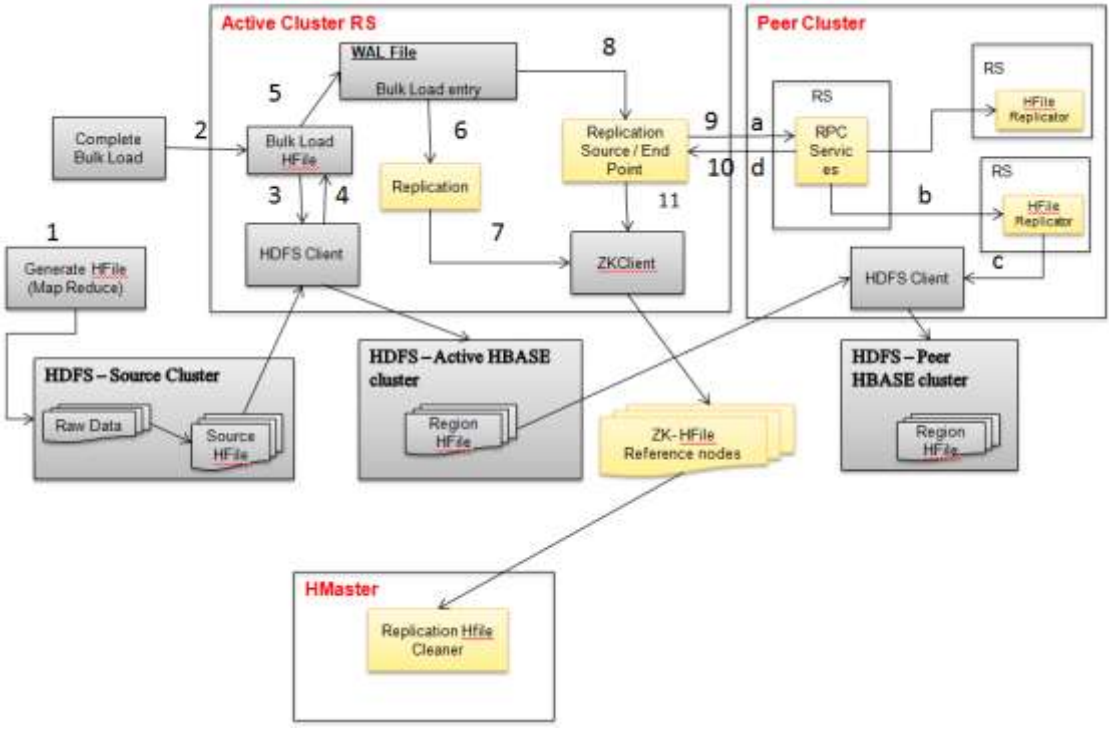
Ashish Singhi & Bhupendra Kumar Jain
Huawei Technologies
August 2015
JIRA: HBASE-13153

1 Requirement Scenario

HBase Replication currently doesn't replicate the bulk loaded hfile data.

Currently replication happens thru replaying the WAL entries from Active cluster to peer cluster. Since Bulk Load process by-passes the WAL edits, so bulk load data will not be replicated.

2 High Level Solution



- HDFS- Source cluster: Contains RAW data and generated hfiles
- HDFS- Active HBASE cluster: HDFS cluster used by active HBASE cluster
- HDFS- Peer HBASE cluster: HDFS cluster used by Peer HBASE cluster

Replication module will be enhanced to support bulk loaded hfiles also. The bulk load events are already captured as wal entry in the wal file (refer HBASE-11567). The wal entry for bulk load event contains the loaded hfile info. If this feature is enabled, Replication scope will be added for these wal entries.

Active HBase RS will also send these wal entries to peer cluster. Peer cluster will

i) Read these wal entries

ii) Copy these hfiles in the peer cluster hbase staging directory

iii) Load these hfiles [Load mechanism will be similar as complete bulk load]

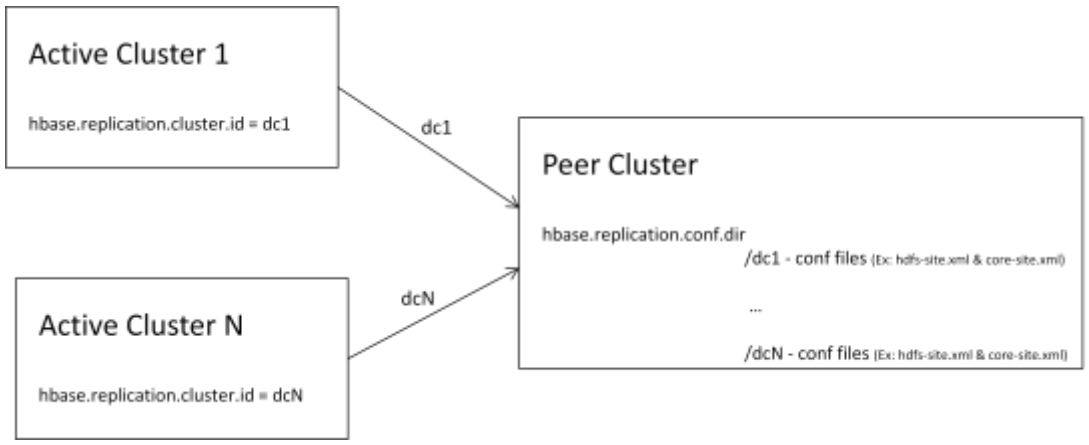
Copy Mechanism: Peer cluster will copy bulk loaded hfiles from active cluster FS to its own FS. To do this peer cluster will need active cluster FS client configurations.

In case of multiple active clusters, peer cluster should be able to identify their respective client configurations while copying these hfiles, so to handle this;

- Each active cluster must **configure a unique replication cluster id** if this feature is enabled. This id will be sent to the peer cluster as part of replication request.
- Provide an **interface called SourceFSConfigurationProvider**

  User can implement this interface and provide his/her own implementation to get source cluster file system client configurations and configure this implementation as value for "hbase.replication.source.fs.conf.provider" property in hbase-site.xml file.

  The default implementation of SourceFSConfigurationProvider is DefaultSourceFSConfigurationProvider, which will work as follows,



Each peer cluster will have a replication configuration directory (configurable). This directory will have sub-directory for each of the active cluster. Sub-directory name must be same as active cluster unique replication id. These will contain the respective FS client configurations.

Note:

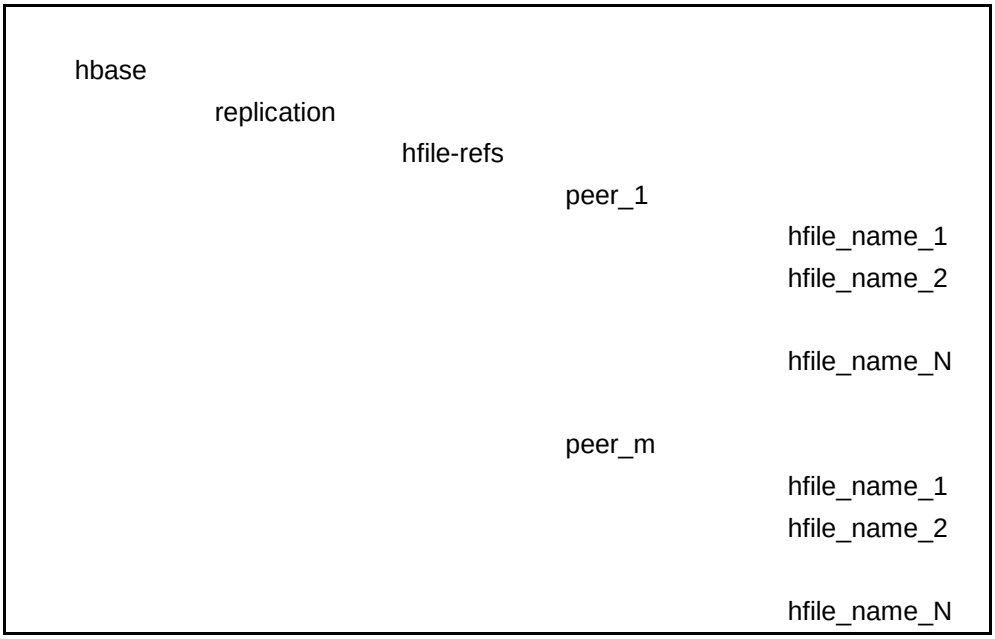
i) If there is any change in FS client configurations in replication configuration directory then all the RS in peer cluster must be restarted.

ii) Only files having 'xml' as extension will be read from the replication configuration directory.

Cyclic Replication detection: Active cluster ids will also be persisted as part of bulk load wal entry. These active cluster ids will be used for cycle detection. [Same as existing design]

Replication HFile Cleaner: If compaction/merge /split happen, then the hfile will be moved to archive folder. This will be cleaned up by Hfile cleaner periodically. To avoid this before replication completes, replication module will maintain these bulk loaded hfile paths into ZK for each peer. Cleaner will not clean the hfiles till its entry is found in ZK. Once the hfile is replicated successfully, the ZK entry will be deleted by replication module for that particular peer.

ZK Replication hfile references



- HFile path node will be created only under those peer nodes for which the replication is enabled for this particular table/cf.

Other approaches to avoid hfile cleanup:

a) Maintain the hfile reference in a new system table. [Can be thought along with HBASE-10295]

b) Master cleaner thread, sends the call to each RS to fetch the bulk loaded hfile paths from its wals. [Reading wal on each cleanup call might be heavy, So ignored this option]

Failover Handling

All failover scenarios are already handled by existing replication module. Same will hold good for hfile replication also.

3 Constraints and Limitations

- If data in Visibility Labels table is different in active and peer cluster, then Visibility expressions will fail during scan in peer cluster.

  Example:

  Active cluster visibility table has entry such as: SECRET 1

  Peer cluster visibility table has entry such as: SECRET 2

  Bulk loaded Hfile of Active cluster will contain the value as '1', this hfile will be directly copied to Peer cluster during replication, So peer cluster hfile will still have entry as '1'. During Scan there will not be any matching entry corresponding to '1' in Peer cluster Visibility Tables. So scan will fail to parse this visibility expression.
- Peer cluster will require the Read permission for active HDFS cluster.
- Peer cluster must have Compression codec library used in active cluster for hfile compression.

4 Performance Scenarios

- Hfile Data Transfer: To reduce the amount of data sent over the network, active hfile should be compressed.
- Table Split: In case, Active and Peer cluster have different split points for table regions, then hfile will be split before loading into peer cluster. It will slow down the replication process. To avoid this, Active and Peer cluster table should have same split points.
- Bandwidth control: The RPC call from hbase contains only hfile Paths. So there is no need to control the data limit in hbase. The main network data traffic is generated by copy of hfiles from one HDFS cluster to another.

5 Backward Compatibility

- No changes in existing public APIs.

6 Security

- Peer cluster will require Read permission for active HDFS cluster.
- Other security configuration remains same as existing.

7 Metrics

All below metrics are at active RS level / global.

- Number of bulk load files pending
- Number of bulk load entries shipped

All below metrics are at peer RS level.

- Number of bulk load entries applied

{

"name" : Hadoop:service=HBase,name=RegionServer,sub=Replication",

"modelerType" : "RegionServer,sub=Replication",

source.sizeOfHFileRefsQueue : 1

source.<peer_id>.sizeOfHFileRefsQueue : 1

source.shippedHFiles: 1

source.<peer_id>.shippedHFiles: 1

sink.appliedHFiles: 1

}

8 Configuration

Name	Description	Type	Default
hbase.replication.bulkload.enabled	To enable replication for bulk loaded data	boolean	False
hbase.replication.cluster.id	Mandatory if replication for bulk loaded data is enabled. This is unique ID to identify the source hbase cluster. It needs to be defined in source cluster.	String	
hbase.replication.conf.dir	This represents the directory where all the active FS cluster's client configurations are defined in their respective subfolders. This configuration needs to be defined in peer cluster.	String	HBase configuration directory (HBASE_CONF_DIR)
hbase.replication.source.fs.conf.provider	This represents the class which provides the source cluster file system configuration to peer cluster. This configuration needs to be defined in peer cluster.	String	org.apache.hadoop.hbase.replication.regionserver.DefaultSourceFSConfigurationProvider

9 Interfaces (API)

No external interfaces