Hive Automatic Recovery Solution

Motivation

We are doing log analysis using Hive by submitting queries through Hive Server and we have provided Name Node HA and Job tracker HA to achieve the high availability but Currently Hive Server is a single point of failure. If the machine running Hive Server is down or broken, Hive service cannot be availed till someone notice the Hive Sever failure and bring it up till this time our log analysis is not continuing. To avoid this problem we need an automatic system that can detect the failure and make sure of the high availability of the Server.

Proposal

Deploy two Hive Servers. One of the Hive Server will act as active while the other one will be a Hot Standby. Here we need a system to decide which can be active and which can be standby and a failure detection mechanism it should detect if Active server is down or broken and trigger the switch over (standby to active). This failure detection mechanism will be based on Zookeeper (HA Agent).

The clients of Hive Server should be configured with the address of both servers. While getting the connection it will detect the Active Hive Server & connect to it.

While executing query Hive Server is down after starting Hive Server need to submit the query again but already executed query will run in the background. Continuing this query execution is no use so it is wastage of cluster resource. In this solution once active is down standby will become active to server and it will ensure to stop the already executed query execution (Hive tasks & Map Red jobs).

It is expected to have multiple Hive Servers (currently 2) and one among them should act as active while the other acts as the Standby. If the Active Hive Server down, the Standby Hive Server should take up the role. The switchover should as fast as possible.

Terminology

HA Agent – HA Agent is not any separate process, but is part of HA Enabled Hive Server itself. HA Agent is responsible for electing the Active Hive Server and triggering the failover.

Active Hive Server - The Hive Server which accepts connections from clients & serves their requests

Standby Hive Server - The Hive Server which doesn't provide any service to the clients, but can readily do so when asked to become Active.

Neutral State - In this state Hive Server is neither Active nor Standby. In this state Server won't serve any kind of request. This state is introduced to handle the split brain scenario. If the HA Agent is unable to contact the Zookeeper quorum, it will instruct Hive Server to go to Neutral state.

Intelligent Client – When connection is requested from Hive Client, it can automatically detect the currently active server and obtain the connection. If there is no active server it will throw an exception.

Detailed Use Cases

- Start a single Hive Server it will become Active.

- Start two Hive Servers. One becomes as Active and the other becomes Standby. Normally the server started first will become Active. Active Hive Server accepts connections from Hive client & services the requests. Standby Hive Server does not accept any connections from client, but it will update the Meta data for synch up and start Derby in slave mode

- When Active server, Standby server and zookeeper are healthy, no state changes happen.

- When Active Hive Server goes down (machine down / process down), Standby Hive Server takes up the role of Active & starts serving client requests. It will perform below steps
  - Stop standby server
  - Start Derby in master mode
  - Start the Hive Server

- When Active Hive Server is network partitioned, Standby Hive Server takes up the role of Active & starts serving client requests. The network partitioned server will go to neutral state.

- The Active Hive Server machine crashes (Ex: disk failure). The Standby Hive Server will become Active and serve the clients.

- Start Active Hive Server, it starts serving client requests. Start Standby Hive Server after a long time. Standby synchronizes with Active & takes over when Active goes down.

- When Standby Hive Server fails, Active Hive Server continues normally.

- Start the Hive Server when Zookeeper servers are unavailable. Hive Server remain in a neutral state waiting for the zookeeper server availability. When the Zookeeper service becomes available, one of the Hive server becomes Active and other one becomes Standby.

Use Cases not under the Scope

- The high availability support is to reduce the down time of the Hive server, If Switch happens in middle of query execution the query execution will fail and client needs to submit the query again.

- Active Hive server is running and standby hive server is not available before standby server availed active hive server is down. Now which server is started that become active and it cannot get the previous active hive server data.

Supported Failures

- ✓ When Active Hive Server goes down (machine down / process down), Standby Hive Server takes up the role of Active & starts serving client requests.

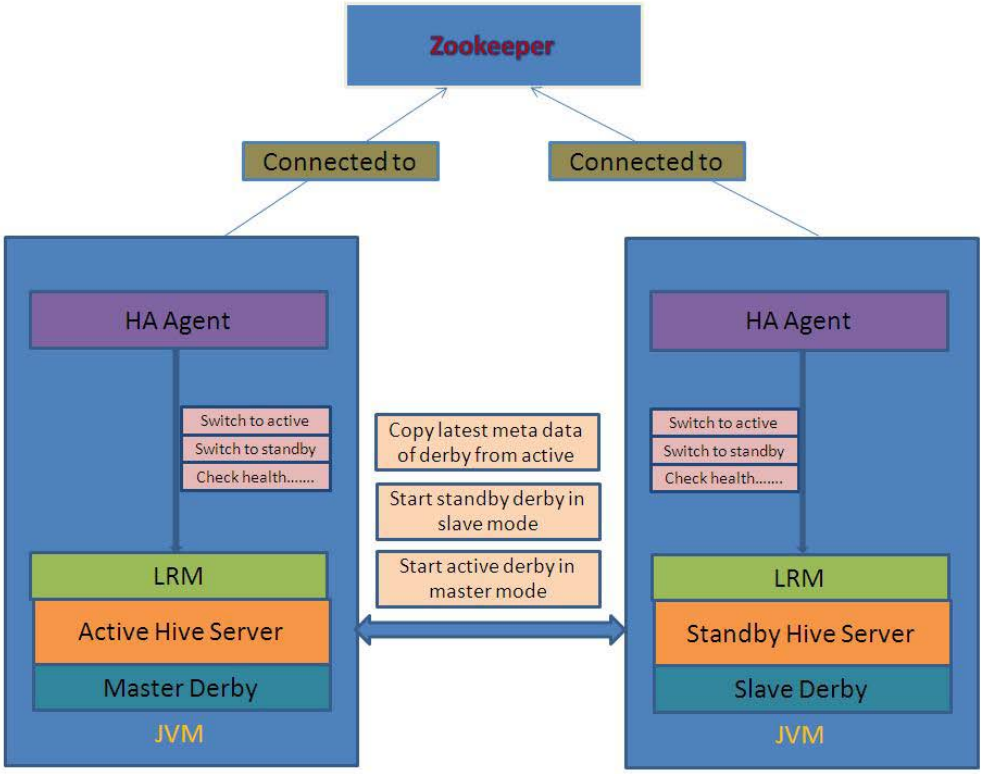
- ✓ When Active Hive Server is network partitioned, Standby Hive Server takes up the role of Active & starts serving client requests, if Standby can connect successfully to Zookeeper service.

- ✓ When Zookeeper service is unavailable to both Active & Standby Hive Servers, both of them transition to neutral state.

Design / Solution of Hive Server HA

The sub system designs identified for the implementation are

1. Hive servers.
2. Embedded derby internal to the Hive servers.
3. Hive client.



Active & Standby Hive Servers

HA Agent – The HA enabler

We have a generic failure detection system, the HA Agent, which can be used across all the systems which need to provide High Availability.

The solution is based on Zookeeper & complies with the recipe for Leader Election as mentioned in http://zookeeper.apache.org/doc/r3.3.3/recipes.html. A more detailed description of this can be found in the design document attached for https://issues.apache.org/jira/browse/ZOOKEEPER-1080.

As tried to depict in the above picture, both HA Agent and Hive Server runs in the same JVM. HA Agent being a generic framework, it talks to Hive Server through the interface LocalResourceManager (LRM). This implies that the HA implementation is not tightly coupled to any particular HA Agent implementation. It will be able to work perfectly with other HA Agents, one example is having an HA Agent based on Linux Heartbeat.
The Zookeeper based HA Agent works as follows.

a) On startup, HA Agent will connect to Zookeeper and check if the master path is available or not. (The master path is an ephemeral node in the Zookeeper).
b) If master path is not available, then it will create the master path and ask the Hive Server to start in Active mode. (This communication happens over the LocalResourceManager interface).
c) If master path is available, then HA Agent will add a watcher to the master path and ask the Hive Server to start in Standby mode.
d) Zookeeper periodically checks the session status of the ephemeral node. If the Active server goes down, zookeeper detects that connection of the ephemeral node is broken. It sends notification to all the watchers. The HA agent in the Standby Hive Server gets this notification and ask the Hive Server to switch to Active mode.

Interface from Hive to interact with HA Agent

This is the implementation of the general interface provided by HA Agent. It mainly has following interaction points.

- StartAsActive
- StartAsStandby
- StandbyToActive

Embedded derby internal to the Hive servers

For the high availability need to synch up the metadata between Active and Standby Hive Servers, the metadata synchronization is achieved by using DB replication facility provided by derby. Derby supports starting DB in Master and Slave modes, and master derby updates will be updated with slave derby also. The ACTIVE Hive server will use the Derby in master mode and the STANDBY Hive Server will use the Derby in slave mode. So switching the Hive servers with up-to-date data is possible.

StartAsActive

Starting the Hive server in ACTIVE mode is just start the server as in normal start. As there is no STANDBY available it need not do any specific operation to manage synchronization with STANDBY.

StartAsStandby

Starting the Hive server in STANDBY will do the below sequential operations.

- Copy the latest metadata of derby from ACTIVE Hive server to STANDBY Hive server derby.
- Start the STANDBY derby in slave mode.
- Start the ACTIVE derby in master mode and start the DB replication.

The stand by Hive server is not serving any requests as ACTIVE server does. It just starts a daemon to keep the STANDBY alive and keeps a slave derby for replication of metadata from master derby of the ACTIVE Hive server to synch up the metadata.

StandbyToActive

The change of STANDBY to ACTIVE switch will happen once the ACTIVE Hive went down and the switching request came from the HA Agent. The switching to ACTIVE is nothing but stopping the STNADBY server and starts in ACTIVE mode. The operations done are,

- Stop the STANDBY daemon running.
- Stop the slave derby running.
- Start the Hive Server in normal mode as starting in startAsActive. (As both the master metadata derby and slave metadata derby are in synch the new active hive also will have the same state as previous)

Neutral state:

In this state Hive Server is neither Active nor Standby. In this state Server won't serve any kind of request. Whenever Hive server is not able to connect to zookeeper it will go to Neutral state.

Intelligent Clients

The Hive client will get the connection to the Hive server and executes queries.
When Active Hive Server goes down & Standby takes over the role, the clients should understand it & communicate to the newly Active Hive Server, from that point of time.

The high level approach is,
In the new high availability environment, the Hive client will be supplied with multiple Hive server URLs and the client will communicate to the ACTIVE one. Getting connection internally from these URLs will be like client will try to connect to all the provided URLs one by one until getting an ACTIVE connection for the first connection and next time onwards it will get the connection directly from the known active if known active is down again it will check the status from the configured URL's and get the connection from the active. [Only ACTIVE Hive server allows client to connect to it and STANDBY not allows, and only one ACTIVE at a time in cluster].

Test Results

- ✓ Tested with 2 servers one will act as Active server and other will act as Standby server.
- ✓ Ran 1 week with continuous switching and executed all kinds of queries with the Meta data size of 100 MB.
- ✓ After running 1 week there is no memory leak.
- ✓ Time taken for an online Standby Hive Server to switch & fully function as the Active Hive Server (with 100MB metadata) is less than 5 minutes.

Future Work

- In middle of query execution if switch happens after query execution it should able to get the result without resubmitting the query and client should be able to connect to the active server automatically when switch happens
- Dynamically client should be able to known about the new hive servers if any started newly other than the configured URL's