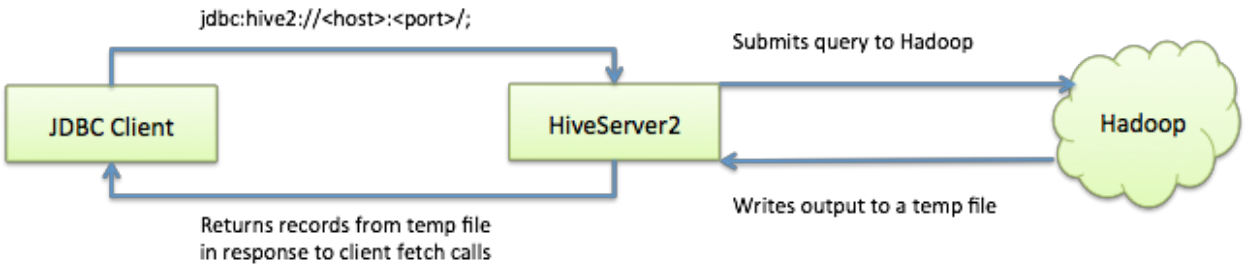
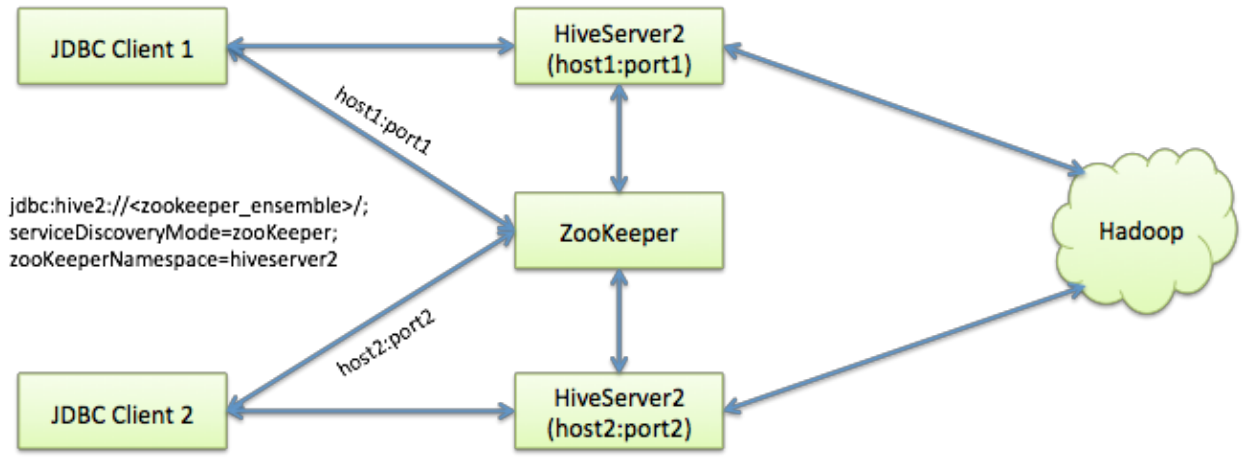
Dynamic Service Discovery in HiveServer2

Currently, the JDBC/ODBC client connection to HiveServer2 works as follows:



Any new client connection needs to statically specify an HiveServer2 URI to connect to. Even if multiple server instances are running, the clients need to be aware of the their URIs. If any server instance is brought down or a new instance is brought up, this needs to be communicated to users trying to connect to that particular instance. As you can see, this is not an ideal setup to support High Availability or Rolling Upgrades.

As a result, HIVE-8376 implements dynamic service discovery for HiveServer2. For implementing dynamic service discovery, we are using Apache ZooKeeper for providing an indirection step between the clients and server instances:



When a HiveServer2 instance comes up, it registers itself with ZooKeeper by adding a znode. The znode name has the format:
`/<hiveserver2_namespace>/serverUri=<host:port>;version=<versionInfo>;sequence=<sequenceNumber>`,
and stores the server host:port as its data. In addition, the server instance sets a watch on the znode which sends a notification to the server when the znode is modified. Specifically, the notification helps the server instance track its removal from the list of servers available for new client connections.

A JDBC/ODBC client trying to connect to HiveServer2, uses the following connection string:
`jdbc:hive2://<zookeeper_ensemble>/;serviceDiscoveryMode=zooKeeper;zooKeeperNamespace=<hiveserver2_zookeeper_namespace>`,
where `<zookeeper_ensemble>` is a comma separated list of ZooKeeper servers forming the ensemble like: `zk_host1:zk_port1,zk_host2:zk_port2,zk_host3:zk_port3`, and `<hiveserver2_zookeeper_namespace>` is namespace on ZooKeeper under which HiveServer2 znodes are added. The JDBC driver connects to ZooKeeper, and selects a HiveServer2 instance at random. This instance is then used by the connecting client for her entire session.

When a HiveServer2 instance is deregistered from ZooKeeper, client sessions on the server are not affected. However, it is removed from the list of servers available for *new* client connections. The server shuts down when the last client session on the server is closed. For deregistering all server instances of a specific version from ZooKeeper, the following new HiveServer2 command can be used:
`hive --service hiveserver2 --deregister <hive_version>`,
where `<hive_version>` is the version number of the hive installation.

Using Dynamic Service Discovery for Rolling Upgrade

Steps to follow to support rolling upgrade for HiveServer2:

A. Extra config settings

1. Set `hive.zookeeper.quorum` to the ZooKeeper ensemble (a comma separated list of ZooKeeper server host:port) running on the cluster.
2. Customize `hive.zookeeper.session.timeout`. This will close the connection between the HiveServer2's client and ZooKeeper if a heartbeat is not received within the timeout.
3. Set `hive.server2.support.dynamic.service.discovery` to `true`.
4. Set `hive.server2.zookeeper.namespace` to the value that you want to use as the root namespace on ZooKeeper. The default value is `hiveserver2`.

In addition, the admin will need to ensure that the ZooKeeper service is running on the cluster and each HiveServer2 instance gets a unique host:port combination to bind to, on its startup.

B. Upgrade steps

1. Bring up HiveServer2 instances of the new version and ensure a successful startup.
2. Now deregister HiveServer2 instances of the old version by issuing:
   `hive --service hiveserver2 --deregister <old_hive_version>`
3. Ensure that you **do not** shut down the HiveServer2 instances of the older version as they could have active client sessions. The server instances shut down on their own when the last client connection is closed.

C. Downgrade steps

1. Bring up HiveServer2 instances of the old version and ensure a successful startup.
2. Now deregister HiveServer2 instances of the new version by issuing:
   `hive --service hiveserver2 --deregister <new_hive_version>`
3. Ensure that you **do not** shut down the HiveServer2 instances of the new version as they could have active client sessions. The server instances shut down on their own when the last client connection is closed.