End2EndTests

As discussed in HIVE-2985 (http://issues.apache.org/jira/browse/HIVE-2985), the goal of this exercise is to create a end to end test framework.

Currently, Facebook is using a deployment of hive which is pretty close to trunk. We would like to continue doing so, but would like to invest more in a test framework in order to do so.

The high level idea is to replicate the deployment framework from Facebook.

This will us get the changes tested thoroughly in our environment before they are committed. Also, it make easier for contributors outside Facebook to test/debug their changes in this environment and make sure they are not breaking anything.

It would be perfect to get all changes to the apache hive trunk run a subset of Facebook workload on a test cluster (inside Facebook) using the deployment model in Facebook (including all the custom hooks, and the configurations) before being committed. This would give us the confidence that none of these changes are breaking any internal Facebook deployment. However, if any test breaks, and the patch is being contributed by an outside Facebook contributor,
it is impossible for the outside committer to debug these issues, since there is no access to this test environment. The work-around is to mimic this workload,
and make it available in open source, so that all committers can run/debug these tests. So, we are trying to create a set of tests, which will run (via jenkins) on the internal Facebook test cluster for every patch, and the patch should not be committed if these tests break. The assumption is that these tests can be run outside also, assuming you have a mysql and hadoop installation. Eventually, these tests can be run outside Facebook, but for that, someone needs to host
a hadoop cluster for running these. I dont want to sign up for that right now, since that will increase the scope of this exercise.

The following assumptions are being made here:

- The issues caused by the hive patches are independent of hadoop. So, if the jenkins test fails (while running a version of Facebook Hadoop), it will also fail
  in any version of Hadoop supported by Hive.

This document lists the steps in detail of how to get there:

- Create a set of tests which are run in parallel, on a real hadoop cluster.
- Make the hooks available used by Facebook be used in these tests.
- Use the same configuration options as are used internally by Facebook. (For eg. local metastore, statistics collection etc.)

Having said that. there are some hooks in facebook which are very specific to Facebook. For eg., there is a hook which transparently redirects traffic to a production cluster. Those hooks do not make any sense for anyone outside. So, instead of getting all the hooks, the plan is to only get those hooks in open source,
which are generic. We can slowly convert all the hooks to become more generic.

I haven't looked in HIVE-2670 (http://issues.apache.org/jira/browse/HIVE-2670) in detail, but based on my initial understanding have the following thoughts.

- Both of them (HIVE-2670 and HIVE-2985) are trying to achieve the same goals
- I can definitely use the framework proposed by HIVE-2670
- Some differences in the above jiras:
  - HIVE-2985 is trying to use the Facebook configuration. This is the major change.
  - HIVE-2670 is running the tests in mysql, followed by hive. As long as the tests are deterministic, they need not be run in MYSQL every time. The results from these
    tests can be stored in MYSQL.
  - HIVE-2670 tries to run the hive test sequentially, which may not the best approach.

The first one is a blocker (different confs). We should be able to reuse most of the work from HIVE-2670 to get the test-framework.

So, instead of writing new scripts, we can reuse the scripts from HIVE-2670 and run them in jenkins for every commit.