Setting up Bigtop to run HBase system tests

Introduction

Bigtop tests assumes that there is a up-and-running HBase cluster. This set of instructions assumes you have this setup already and covers the case where you want to use bigtop to system test development version of hadoop/hbase. This will walk you through getting bigtop, building bigtop test-execution suite, setting up test-execution against particular versions of hbase and hadoop, and then running tests.

Setting up bigtop

First, install bigtop

```
git clone git://git.apache.org/bigtop.git
cd bigtop
```

Bigtop's "input variables" are environment variables. Thus you need to make sure hbase environment settings are set up for it to use.

```
export JAVA_HOME=/usr/java/latest
export HADOOP_HOME=/usr/lib/hadoop
export HADOOP_CONF_DIR=/usr/lib/hadoop/conf
export HBASE_HOME=/usr/lib/hbase
export HBASE_CONF_DIR=/usr/lib/hbase/conf
export ZOOKEEPER_HOME=/usr/lib/zookeeper
```

Also, if you are using Zookeeper in hbase managed mode, you can cheat and to set ZOOKEEPER_HOME=$HBASE_HOME/lib to point to the zk jar.

test-execution prerequisites (aka, make maven work)

Test execution only pulls dependencies from mvn pom files. So, even if you have your cluster setup and environment variables setup, bigtop tests will use only the jars from the test-execution poms. This means you need to make sure that you have the poms with the versions you are using and proper jars installed in you mvn .m2 cache.

Let's start with the bigtop jars. Make maven build and install the bigtop related test execution jars. These command will download the world and then install into your ~/.m2 dir.

```
mvn -f bigtop-tests/test-artifacts/pom.xml  install
mvn -f bigtop-tests/test-execution/conf/pom.xml  install
mvn -f bigtop-tests/test-execution/common/pom.xml  install
```

If you are using a custom build of HBase, build and mvn install it.

```
cd $HBASE_HOME
mvn install -DskipTests
```

Modify the hbase execution pom to point to the particular version of hbase you want. I'm testing an HBase 0.92 release so I've gotten a copy of hbase-0.92 and did a 'mvn install -DskipTests' to install the pom into my local directory. Here's the snippet I added to bigtop-tests/test-execution/smokes/hbase/pom.xml

```
<dependencies>
 <dependency>
    <groupId>org.apache.hbase</groupId>
    <artifactId>hbase</artifactId>
    <version>0.92.0</version>
  </dependency>
</dependencies>
```

This section is specific to cloudera provided tarballs, available here: http://archive.cloudera.com/cdh/3/ . Unfortunately, I don't know how to do this from the Apache Hadoop releases.

If you are using a custom hadoop/hdfs/mr, (some tests write to HDFS directly and thus need to have the proper version of Hadoop jars) build and mvn install these jars to your local the .m2 repo. Hadoop is normally an ant build so to do this (specifically for cdh3 snapshots), run maven using the hadoop.git's cloudera-pom.xml to install on the hadoop repo dir.

```
mvn -f cloudera-pom.xml install -DskipTests
```

Next, add the dependency to the execution pom.xml.

```
<dependencies>
 <dependency>
    <groupId>org.apache.hadoop</groupId>
    <artifactId>hadoop-core</artifactId>
    <version>0.20.2-cdh3u3-SNAPSHOT</version>
  </dependency>
</dependencies>
```

The `hbase classpath` generates a class path and is "smart" enough to pull in cached .m2 artifacts. This may cause a problem if your hbase build depends on a different version of hadoop. You can do a 'mvn clean' afterwards in $HBASE_HOME to remove files the hbase script uses to cache.

test-execution.

To run tests go to the test excution dir and execut mvn verify with the particular include filter. This particular test runs a quick-and-simple create table, put and get.

```
cd bigtop-tests/test-execution/smokes/hbase
# A quick test  (-o makes maven work in "offline mode", and use local .m2 versions of the bigtop jars)
mvn verify -o -Dorg.apache.maven-failsafe-plugin.testInclude=**/TestHBaseSmoke*
```

Victory.

This is a slow but thorough test (took about 50mins on a 10 node cluster, with 20MM entries)

```
mvn verify -o -Dorg.apache.maven-failsafe-plugin.testInclude=**/TestLoadAndVerify*
```

Glorious victory.