NodeTool

More and more instrumentation is being added to Cassandra via standard JMX apis.

The `nodetool` utility (`nodeprobe` in versions prior to 0.6) provides a simple command line interface to these exposed operations and attributes.

See Operations for a more high-level view of when you would want to use the actions described here.

Note: This utility currently requires the same environment as cassandra itself, namely the same classpath (including log4j.properties), and a valid storage-conf property.

Running `bin/nodetool` with no arguments produces some usage output.

```
Starting NodeTool
Missing required option: h
usage: java org.apache.cassandra.tools.NodeCmd --host <arg> <command>

 -h,--host <arg>        node hostname or ip address
 -p,--port <arg>        remote jmx agent port number
 -pw,--password <arg>   remote jmx agent password
 -u,--username <arg>    remote jmx agent username

Available commands:
  ring                   - Print informations on the token ring
  join                   - Join the ring
  info                   - Print node informations (uptime, load, ...)
  cfstats                - Print statistics on column families
  clearsnapshot          - Remove all existing snapshots
  version                - Print cassandra version
  tpstats                - Print usage statistics of thread pools
  drain                  - Drain the node (stop accepting writes and flush all column families)
  decommission           - Decommission the node
  loadbalance            - Loadbalance the node
  compactionstats        - Print statistics on compactions
  disablegossip          - Disable gossip (effectively marking the node dead)
  enablegossip           - Reenable gossip
  disablethrift          - Disable thrift server
  enablethrift           - Reenable thrift server
  snapshot [snapshotname] - Take a snapshot using optional name snapshotname
  netstats [host]        - Print network information on provided host (connecting node by default)
  move <new token>       - Move node on the token ring to a new token
  removetoken status|force|<token> - Show status of current token removal, force completion of pending removal
or remove provided token
  flush [keyspace] [cfnames] - Flush one or more column family
  repair [keyspace] [cfnames] - Repair one or more column family
  cleanup [keyspace] [cfnames] - Run cleanup on one or more column family
  compact [keyspace] [cfnames] - Force a (major) compaction on one or more column family
  scrub [keyspace] [cfnames] - Scrub (rebuild sstables for) one or more column family
  invalidatekeycache [keyspace] [cfnames] - Invalidate the key cache of one or more column family
  invalidaterowcache [keyspace] [cfnames] - Invalidate the key cache of one or more column family
  getcompactionthreshold <keyspace> <cfname> - Print min and max compaction thresholds for a given column family
  cfhistograms <keyspace> <cfname> - Print statistic histograms for a given column family
  setcachecapacity <keyspace> <cfname> <keycachecapacity> <rowcachecapacity> - Set the key and row cache
capacities of a given column family
  setcompactionthreshold <keyspace> <cfname> <minthreshold> <maxthreshold> - Set the min and max compaction
thresholds for a given column family
```

The -host argument is required, the -port argument is optional and will default to 8080 if not supplied.

Ring

The ring command will present node status and an ascii art rendition of the ring, *as determined by the node being queried.*

```
bin/nodetool -host 10.176.0.146 ring
Address       Status     Load          Range                                      Ring
10.176.0.146  Up         459.27 MB     75603446264197340449435394672681112420     |<--|
10.176.1.161  Up         382.53 MB     137462771597874153173150284137310597304    |   |
10.176.1.162  Up         511.34 MB     63538518574533451921556363897953848387     |-->|
```

The format is a little different for later versions - this is from v0.7.6:

```
Address         Status State   Load            Owns    Token
                                                       113427455640312821154458202477256070484
10.176.0.146    Up     Normal  459.27 MB       33.33%  0
10.176.1.161    Up     Normal  382.53 MB       33.33%  56713727820156410577229101238628035242
10.176.1.162    Up     Normal  511.34 MB       33.33%  113427455640312821154458202477256070484
```

The `Owns` column indicates the percentage of the ring (keyspace) handled by that node

The largest token is repeated at the top of the list to indicate that we have a ring. i.e, the first and last printed token are the same, suggesting some kind of continuity

Join

Triggers the specified node to join the ring. This assumes that the node was started with `-Dcassandra.join_ring=false` so that it did not join the ring upon startup.

Info

Outputs node information including the token, load info (on disk storage), generation number (times started), uptime in seconds, and heap memory usage.

```
bin/nodetool -host 10.176.0.146 info
Token(137462771597874153173150284137310597304)
Load Info        : 0 bytes.
Generation No    : 1
Uptime (seconds) : 697595
Heap Memory (MB) : 28.18 / 759.81
```

Cleanup

Triggers the immediate cleanup of keys no longer belonging to this node.

```
bin/nodetool -host 10.176.0.146 cleanup
```

Compact

Initiates an immediate table compaction.

Note: the compacted tables will not immediately be cleared from the hard disk and will remain on the system until the JVM performs a GC. For more information, read about MemtableSSTables.

```
bin/nodetool -host 10.176.0.146 compact
```

Decommission

Tells the node to move its data elsewhere; opposite of bootstrap. Since 0.5. See https://issues.apache.org/jira/browse/CASSANDRA-435

Removenode

Removing a node that does not physically exist anymore is done in two steps:

```
bin/nodetool removenode <UUID>
bin/nodetool removenode force
```

The first command will block forever if the computer attached to that UUID was physically removed (or does not run Cassandra anymore). Just click Ctrl-C after a second or two before running the second command. Obviously, it is better to first decommission a node if possible or you may lose some of your data.

The "bin/nodetool status" command shows the UUID of your nodes.

Drain

Flushes memtables on the node and stop accepting writes. Reads will still be processed. Useful for rolling upgrades.

Flush

Flushes memtables (in memory) to SSTables (on disk), which also enables CommitLog segments to be deleted.

Removetoken

Removes a dead node from the ring - this command is issued to any other live node (since clearly the dead node cannot respond!).

Scrub

Cassandra v0.7.1 and v0.7.2 shipped with a bug that caused incorrect row-level bloom filters to be generated when compacting sstables generated with earlier versions. This would manifest in IOExceptions during column name-based queries. v0.7.3 provides "nodetool scrub" to rebuild sstables with correct bloom filters, with no data lost. (If your cluster was never on 0.7.0 or earlier, you don't have to worry about this.) Note that nodetool scrub will snapshot your data files before rebuilding, just in case.

Upgradesstables

While "scrub" does rebuild your sstables, it will also discard data it deems broken and create a snapshot, which you have to remove manually. If you just wish to rebuild your sstables without all that jazz, then use "nodetool upgradesstables". This is useful e.g. when you are upgrading your server, or changing compression options.

upgradesstables is available from Cassandra 1.0.4 onwards.

Setcompactionthroughput

As of Cassandra 1.0, the amount of resources that compactions can use can be easily controlled using a single value: the compaction throughput, which is expressed in Megabytes/second. You can (and probably should) specify this in your cassandra.yaml file, but in some cases it can be very beneficial to change it live using the nodetool.

For example, in this presentation Edward Capriolo explains how their company throttles compaction during the day so that I/O is mostly reserved for serving requests, whereas during the night they allocate more capability for running compactions. This can be e.g. accomplished through a simple cron script:

```
# Script increases compaction throughput to 999 MB/s (i.e. nearly unlimited) for 00-06.
#
# turn into Mr.batch at night
0 0 * * * root nodetool -h `hostname` setcompactionthroughput 999
# turn back into Dr.Realtime for day
0 6 * * * root nodetool -h `hostname` setcompactionthroughput 16
```

Setting the compaction thresholds to zero disables compaction. This may be useful in some cases if you e.g. wish to avoid the compaction I/O during extremely busy periods. It is not a good idea to leave it on for a long period, since you will end up with a large number of very small sstables, which will start to slow down your reads.

Setting the compaction throughput to zero, however, disables throttling.

Cfhistograms

Excellent description from: http://narendrasharma.blogspot.com/2011/04/cassandra-07x-understanding-output-of.html

The output of the command has following 6 columns:

- Offset
- SSTables
- Write Latency
- Read Latency
- Row Size
- Column Count

Interpreting the output

- Offset: This represents the series of values to which the counts for below 5 columns correspond. This corresponds to the X axis values in histograms. The unit is determined based on the other columns.
- SSTables: This represents the number of SSTables accessed per read. For eg if a read operation involved accessing 3 SSTables then you will find a +ve value against Offset 3. The values are recent i.e. for duration lapsed between two calls.
- Write Latency: This shows the distribution of number of operations across the range of Offset values representing latency in microseconds. For eg. If 100 operations took say 5 ms then you will find a +ve value against offset 5.
- Read Latency: This is similar to write latency. The values are recent i.e. for duration lapsed between two calls.
- Row Size: This shows the distribution of rows across the range of Offset values representing size in bytes. For eg. If you have 100 rows of size 2000bytes then you will find a +ve value against offset 2000.
- Column Count: This is similar to row size. The offset values represent column count.

Some additional details

Typically in a histogram the values are plotted over discrete intervals. Similarly Cassandra defines buckets. The number of buckets is 1 more than the bucket offsets. The last element is values greater than the last offset. The values you see in the Offset column in the output is bucket offsets. The bucket offset starts at 1 and grows by 1.2 each time (rounding and removing duplicates). It goes from 1 to around 36M by default (creating 90+1 buckets), which will give us timing resolution from microseconds to 36 seconds, with less precision as the numbers get larger. (see EstimatedHistogram class)