****

**YCSB Benchmark Summary: HBase 0.98 and Cassandra 2.0.8**

This document describes a test executed by a neutral 3rd party systems integrator on behalf of their client to understand the relative performance of Cassandra 2.0.8 and HBase 0.98 for given workloads using the YCSB benchmark.

The results and other information proprietary to the 3rd party and have been removed or anonymized in order to stay within the bounds of non-disclosure. This document is also considered proprietary and should not be shared without permission. DataStax welcomes the opportunity to reproduce this benchmark in your environment for your use case.

**Test Approach**

Testing priorities were set by the client in the following order:

* Availability
* Performance

While performance is a necessary and vital component, performance is irrelevant if the system is not available.

**Test Environment**

The environments are instanced in AWS on “m1.xlarge” on a 1GB Network with 4 virtual cores, 16 Gigabytes of memory and backed by “gp2”. Gp2 is defined as “General Purpose Storage” in AWS. These particular instances are 4 general disks striped into a single block device. The disk IOPS is capped on these storage devices to 3000 IOPS/sec.

|  |  |  |
| --- | --- | --- |
| Node Specification | Cassandra Cluster | HBase Cluster |
| Amazon AWS  “m1.xlarge”  4 CPU  16 G Memory  1 G Network  3000 IOP Ephemeral Drives | 14 nodes  13 Cassandra data nodes  1 DataStax OpsCenter node | 14 nodes  6 Data and Region nodes  1 HBase Master / Journal node  1 HBase Master node  3 Zookeeper nodes  1 Namenode/Journal Active  1 Namenode/Journal Standby  1 Ambari/Hue/Oozie/Hive/etc node |

**Availability Testing Results Summary**

Raw test result data are not included here in order to stay in bounds of non-disclosure. Relevant information is reported where possible.

\*\* Note that all tests were conducted on spinning disk drives that were each throttled to a maximum of 3000 IOPS.

Test plan details are included following the test results tables.

**HBase Availability Test Results**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Record  Count | Load | Thread  Count | Client  Nodes | Throughput TPS Performance | Test Notes |
| 10M | %50 R/W | 16 | 14 | HBase node failure highlight in yellow  Macintosh HD:Users:richreffner:Desktop:Screen Shot 2015-04-10 at 10.49.06 AM.png | *1 Datanode Restart*  Latency: The latency begins to drop as the caching mechanisms engage however **once the active namenode is restarted the latency increases continually and the operations drop dramatically and never recover**. **This is not expected behavior.** The cluster should have seen a spike in latency and a quick drop in operations but recovered quickly.  Ops/Sec: **Operations per second drop significantly with the loss of the active namenode and never recovers. This is not expected.**  **There are two major areas of concern.**  1) Latency is expected to decrease continually with only a minimal spike during the namenode restart. We see the spike, but then latency gradually increases never returning to levels prior to the restart even after the test has completed.  2) Operations per second is expected to drop slightly during the restart but then continue increasing or level out. **This runs contrary to expectations in that after the spike they decrease continually until the test has completed**. |
| 10M | %50 R/W | 16 | 14 | HBase node failure highlight in yellow  Macintosh HD:Users:richreffner:Desktop:Screen Shot 2015-04-10 at 10.50.32 AM.png | *1 Active Namenode Restart*  Latency: The latency begins to drop as the caching mechanisms engage **however once the active namenode is restarted the latency increases continually and the operations drop dramatically and never recover. This is not expected behavior**. The cluster should have seen a spike in latency and a quick drop in operations but recovered quickly.  Ops/Sec: **Operations per second drop significantly with the loss of the active namenode and never recover. This is not expected.**  Two areas of concern:  1: Latency is expected to decrease continually with only a minimal spike during the namenode restart. We see the spike, but then **latency gradually increases never returning to levels prior to the restart even after the test has completed.**  2: Operations per second is expected to drop slightly during the restart but then continue increasing or level out. **This test runs contrary to expectations in that after the spike they decrease continually until the test has completed.** |
| 10M | %50 R/W | 16 | 14 | HBase node failure highlight in yellow  Macintosh HD:Users:richreffner:Desktop:Screen Shot 2015-04-10 at 10.51.23 AM.png | *1 Active Namenode Failure & 1 Datanode Failure*  Latency: The latency remains constant with a dramatic spike when the active namenode and workernode are killed. It recovers quickly as expected.  Ops/Sec: Operations per second drop off when the active namenode and workernode are killed but recover quickly and continue as expected.  \*NOTE:  **Once this test concluded HDFS became unavailable and the cluster would not come back online for testing to proceed. Several hours were spent troubleshooting the cluster and testing was pushed back until the issue could be corrected.** A quick glance showed some corruption in HBase metadata in HDFS. Vendor Ticket # XXXXX was created and includes details. The conclusion was “Operator Error”. Services were shutdown incorrectly leading to the HDFS file system being unavailable until everything was corrected. The system was later brought back online the next day for testing to continue. |
| 10M | %50 R/W | 16 | 14 | HBase node failure highlight in yellow  Macintosh HD:Users:richreffner:Desktop:Screen Shot 2015-04-10 at 10.52.05 AM.png | *1 Active Namenode Failure & 1 Datanode Failure \*Attempt #2*  Latency: The latency remains relatively constant which is what we expect though there is no discernable change that alerts to when the infrastructure nodes were killed.  Ops/Sec: **The operations per second are all over the map with no real correlation to the latency. In some instances we see the exact opposite of what we expect in that latency decreases when operations per second increase it should also be clear when failover is taking place. This is not expected**  \*NOTE:  This test was completed on day three. **The first attempt at this test failed and the system never kicked over to the standby namenode rendering the system unusable**. Thus the testing was delayed an additional day while the investigation into the issue took place. After analysis it was **determined that a critical failover component had stopped and was never restarted**. Once the zkfc (zoo keeper failover controller) was restarted the test ran without error and that is the plot you see below. |
| 10M | %50 R/W | 16 | 14 | HBase  Macintosh HD:Users:richreffner:Desktop:Screen Shot 2015-04-10 at 10.53.47 AM.png | *1 Active Namenode Failure & 1 Zookeeper Failure \*Attempt #2*  Latency: The latency remains relatively constant which is what we expect.  Ops/Sec: **The operations per second are all over the map with no real correlation to the latency. From this it is difficult to identify where the failures took place. This is unexpected as it should be clear when failover is taking place**. |
| 10M | %50 R/W | 16 | 14 | HBase  Macintosh HD:Users:richreffner:Desktop:Screen Shot 2015-04-10 at 10.54.33 AM.png | *Active Namenode Failure & 1 HBase Master Failure*  Latency: The latency remains relatively constant which is what we expect.  Ops/Sec: **The operations per second are all over the map with no real correlation to the latency. From this it is difficult to identify where the failures took place. This is unexpected as it should be clear when failover is taking place** |

**Cassandra Availability Test Results**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Record  Count | Load | Thread  Count | Client  Nodes | Throughput TPS Performance | Test Notes |
| 10M | %50 R/W | 32 | 14 | Cassandra node reset highlight in yellow  Macintosh HD:Users:richreffner:Desktop:Screen Shot 2015-04-10 at 10.42.00 AM.png | *Node Reset | Consistency=1*  Latency: As the cache warms up the latency decreases as expected.  Ops/Sec: Operations per second increase steadily as the cache warms up as expected  **Node Reset had minimal impact to operations as expected** |
| 10M | %50 R/W | 32 | 14 | Cassandra node reset highlight in yellow  Macintosh HD:Users:richreffner:Desktop:Screen Shot 2015-04-10 at 10.44.09 AM.png | *Node Reset | Consistency=quorum*  Latency: The latency distinctly shows the loss of the node, however the cluster recovers as expected and latency returns to levels recorded prior to the loss of the node  Ops/Sec: **Operations per second distinctly show the loss of the node, however the cluster recovers as expected and operations per second return to levels recorded prior to the loss of the node** |
| 10M | %50 R/W | 32 | 14 | Cassandra node reset highlight in yellow  Macintosh HD:Users:richreffner:Desktop:Screen Shot 2015-04-10 at 10.45.08 AM.png | *Node Failure | Consistency=1*  Latency: The latency distinctly shows the loss of the node, however the cluster recovers as expected and latency returns to levels recorded prior to the loss of the node  Ops/Sec: **Operations per second distinctly show the loss of the node, however the cluster recovers as expected and operations per second return to levels recorded prior to the loss of the node** |
| 10M | %50 R/W | 32 | 14 | Cassandra node failure highlight in yellow  Macintosh HD:Users:richreffner:Desktop:Screen Shot 2015-04-10 at 10.46.29 AM.png | *Node Failure | Consistency=quorum*  Latency: The latency distinctly shows the loss of 2 nodes, however the cluster recovers as expected and latency returns to levels recorded prior to the loss of the node. There is a large spike towards the end but it lasts only a few seconds and returns to normal only minimally impacting ops/sec.  Ops/Sec: **Operations per second distinctly show the loss of the 2 nodes, however the cluster recovers as expected and operations per second return to levels recorded prior to the loss of the 2 nodes.** |
| 10M | %50 R/W | 32 | 14 | Cassandra node failure highlight in yellow  Macintosh HD:Users:richreffner:Desktop:Screen Shot 2015-04-10 at 10.47.39 AM.png | *2 Node Failures | Consistency=1*  Latency: The latency distinctly shows the loss of 2 nodes, however the cluster recovers as expected and latency returns to levels recorded prior to the loss of the node  Ops/Sec: **Operations per second distinctly show the loss of the 2 nodes, however the cluster recovers without ever dropping to zero operations. This is expected and operations per second return to levels recorded prior to the loss of the 2 nodes.** |
| 10M | %50 R/W | 32 | 14 | Cassandra node failure highlight in yellow  Macintosh HD:Users:richreffner:Desktop:Screen Shot 2015-04-10 at 10.48.24 AM.png | *2 Node Failures | Consistency=quorum*  Latency: The latency continues to decrease with a slight spike when the node was reset. It recovers quickly as expected.  Ops/Sec: **Operations per second drop slightly when the node was reset but recovers quickly as expected.** |

**Performance Test Results Summary**

Charts of test result data are not included here in order to stay in bounds of non-disclosure. The relevant information is reported where possible.

**HBase Performance Test Results**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Records | Load | Threads | Client  Workers | Throughput TPS Performance | Test Notes |
| 1M | 100% W | 32 | 6 clients | H=7747/client  Macintosh HD:Users:richreffner:Desktop:Screen Shot 2015-04-10 at 10.25.57 AM.png | HBase  Insert latency is not in proportion to cleanup latency. Infers **records are stored to memory and flushed to disc at the end of the test**.  **Not a valid production configuration.**  1) Latency sits near zero for the entire duration of the test. When the operation count hits the 1 Million record target you then see a massive spike in latency when the data is written to disk. This means that while the initial 1 million rows were **inserting they were hitting only memory on HBase and not being durably written to disk**.  2) Once the insert requests reach their target of 1 million records HBase flushes everything to disc causing a gigantic jump in latency. Whereas this will not cause any issues on small data sets we will see **as the testing continues that this creates profound problems**. |
| 10M | 100% W | 1 | 6 clients | H=5881/client  Macintosh HD:Users:richreffner:Desktop:Screen Shot 2015-04-10 at 10.28.20 AM.png | HBase  Latency: Insert latency is not in proportion to cleanup latency. Infers **many records are stored to memory and flushed to disc at the end of the test**.  **Not a valid production configuration.**  Operations: Operation count stops increasing inferring that the system is momentarily unavailable. This is **unexpected behavior and must be addressed by policy**. May be related to a fault in the HBase YCSB Driver; this behavior could be explained if the records are being stored in memory and then sent to HBase to be flushed to disc. This **would not be a realistic production solution and it skews the results**; therefore, **additional configuration to the system must be made in order to show HBase is a viable solution**.  Ops/Sec: **Unexpected behavior** as expressed above is also evident by the radical behavior of ops/sec. Again, investigation is needed to ensure future tests present realistic behavior.  Areas of concern  1) We expect operations per second to be steady with marginal spikes when nodes leave or arrive into the cluster. **With operations spiking constantly with no deviation in latency, is an area for concern**.  2) **Inserts actually halt completely for upwards of 40+ seconds in places inferring that the cluster is unavailable during this time period.** **When operations spike high when the cluster becomes responsive there is no change in latency which is another concern**. This could be due to the inserts hitting memory and trying to dump to disk on the backend as memory fills up but without more metrics this determination cannot be made definitively which is also concerning.  3) As the prior plot pointed out, **once the test concluded there was a gigantic leap in latency as HBase flushed everything to disk. The expectation is that inserts are constantly being flushed to disk as they come in so this behavior is unexpected and creates a massive latency spike once the test concludes**. |
| 10M | 100% W | 32 | 6 clients | H=4963/client  Macintosh HD:Users:richreffner:Desktop:Screen Shot 2015-04-10 at 10.30.56 AM.png | HBase  Latency: Insert latency is not in proportion to cleanup latency. Infers **many records are stored to memory and flushed to disc at the end of the test**.  **Not a valid production configuration.**  Operations: See above chart for a more detailed explanation of the problem being described. However, in this particular plot, **the system was temporarily unavailable for over 60 seconds, indicating system failure**.  Ops/Sec: I**nvestigation is needed to ensure future tests present realistic behavior**.  Highlighted below are four major areas of concern.  1, 2, & 3: **Complete and total unresponsiveness of the system that progress’s to get worse from 1 to 2 to 3 yielding over 2 full minutes of when the system is un-addressable**. The **effects from the previous test are amplified as the thread count was increased**. **If this system needs to meet SLA’s, being unresponsive for such a duration would constitute a failure of the system**.  4: As the prior plot pointed out, once the test concluded there  was a gigantic leap in latency as HBase flushed everything to disk. **The expectation is that inserts are constantly being flushed to disk as they come in so this behavior is unexpected and creates a massive latency spike once the test concludes**. |
| 1M | 100% R | 32 | 6 clients | H=520/client  Macintosh HD:Users:richreffner:Desktop:Screen Shot 2015-04-10 at 10.39.33 AM.png | HBase  Latency: As the cache warms up the latency decreases as expected.  Ops/Sec: Operations per second increase steadily as the cache warms up as expected |
| 10M | 100% R | 10 | 6 clients | H=249/client  Macintosh HD:Users:richreffner:Desktop:Screen Shot 2015-04-10 at 10.40.17 AM.png | HBase results consistent with above note. |
| 10M | 100% R | 32 | 6 clients | H=429/client  Macintosh HD:Users:richreffner:Desktop:Screen Shot 2015-04-10 at 10.41.01 AM.png | HBase results consistent with above note. |

**Cassandra Performance Test Results**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Records | Load | Threads | Client  Workers | Throughput TPS Performance | Test Notes |
| 1M | 100% W | 10 | 4 clients | C=1942/client  Macintosh HD:Users:richreffner:Desktop:Screen Shot 2015-04-10 at 10.19.17 AM.png | Cassandra throughput consistent with insert latency & cleanup, as expected. |
| 1M | 100% W | 32 | 4 clients | C=2791/client  Macintosh HD:Users:richreffner:Desktop:Screen Shot 2015-04-10 at 10.21.07 AM.png | Cassandra results consistent with above note.  Ops trail off as client workers complete their workloads as expected. |
| 10M | 100% W | 10 | 4 clients | C=1941/client  Macintosh HD:Users:richreffner:Desktop:Screen Shot 2015-04-10 at 10.22.49 AM.png | Cassandra results consistent with above note. |
| 10M | 100% W | 32 | 4 clients | C=3197/client  Macintosh HD:Users:richreffner:Desktop:Screen Shot 2015-04-10 at 10.25.07 AM.png | Cassandra results consistent with above note. |
| 1M | 100% R | 10 | 4 clients | C=1710/client  Macintosh HD:Users:richreffner:Desktop:Screen Shot 2015-04-10 at 10.34.40 AM.png | Cassandra  Latency: As the cache warms up the latency decreases as expected.  Operations: A steady increase of ops as expected  Ops/Sec: Operations per second increase steadily as the cache warms up as expected |
| 1M | 100% R | 32 | 4 clients | C=5828/client  Macintosh HD:Users:richreffner:Desktop:Screen Shot 2015-04-10 at 10.36.24 AM.png | Cassandra results consistent with above note. |
| 10M | 100% R | 10 | 4 clients | C=1251/client  Macintosh HD:Users:richreffner:Desktop:Screen Shot 2015-04-10 at 10.37.29 AM.png | Cassandra results consistent with above note. |
| 10M | 100% R | 32 | 4 clients | C=4983/client  Macintosh HD:Users:richreffner:Desktop:Screen Shot 2015-04-10 at 10.38.24 AM.png | Cassandra results consistent with above note. |

**Test Plans**

HBase test plan executed was:

1. Load 10MM records into the database
2. 100% Insert Tests
   1. 1MM Records, 10 Threads, 6 Worker Nodes (14 total nodes)
   2. 10MM Records, 10 Threads, 6 Worker Nodes (14 total nodes)
   3. 1MM Records, 32 Threads, 6 Worker Nodes (14 total nodes)
   4. 10MM Records, 32 Threads, 6 Worker Nodes (14 total nodes)
3. 100% Read Tests
   1. 1MM Records, 10 Threads, 6 Worker Nodes (14 total nodes)
   2. 10MM Records, 10 Threads, 6 Worker Nodes (14 total nodes)
   3. 1MM Records, 32 Threads, 6 Worker Nodes (14 total nodes)
   4. 10MM Records, 32 Threads, 6 Worker Nodes (14 total nodes)
4. Availability Testing
   1. Kill Name Node and Data Node (executed twice due to Hadoop Crash)
      1. Start 10MM Records 32 Threads, 6 Worker Nodes (14 total nodes). Reach Equilibrium
      2. Kill Name Node and Data Node simultaneou. Reach Equilibrium
      3. Restart Node. Reach Equilibrium
   2. Kill Name Node and Zoo Keeper (executed twice due to Hadoop Crash)
      1. Start 10MM Records 32 Threads, 6 Worker Nodes (14 total nodes). Reach Equilibrium
      2. Kill Name Node and Zoo Keeper simultaneously. Reach Equilibrium
      3. Restart Nodes, one after the other. Reach Equilibrium
   3. Kill HBase Master and Zoo Keeper (executed twice due to Hadoop Crash)
      1. Start 10MM Records 32 Threads, 6 Worker Nodes (14 total nodes). Reach Equilibrium
      2. Kill Name Node and Zoo Keeper simultaneously. Reach Equilibrium
      3. Restart Nodes, one after the other. Reach Equilibrium
   4. Restart Name Node
      1. Start 10MM Records 32 Threads, 6 Worker Nodes (14 total nodes). Reach Equilibrium
      2. Restart Name Node via Ambari. Reach Equilibrium

Cassandra test plan executed was:

1. Load 10MM records into the database
2. 100% Insert Tests
   1. 1MM Records, 10 Threads, 14 Nodes
   2. 10MM Records, 10 Threads, 14 Nodes
   3. 1MM Records, 32 Threads, 14 Nodes
   4. 10MM Records, 32 Threads, 14 Nodes
3. 100% Read Tests
   1. 1MM Records, 10 Threads, 14 Nodes
   2. 10MM Records, 10 Threads, 14 Nodes
   3. 1MM Records, 32 Threads, 14 Nodes
   4. 10MM Records, 32 Threads, 14 Nodes
4. Availability Testing

All tests were run in two circumstances:

* Consistency Level = ONE and Replication Factor = 3
* Consistency Level = QUORUM and Replication Factor = 5
  1. 1 Node Kill
     1. Start 10MM Records 32 Threads, 14 Nodes. Reach Equilibrium
     2. Kill 1 Node. Reach Equilibrium
     3. Start Node. Reach Equilibrium
  2. 2 Node Kill
     1. Start 10MM Records 32 Threads, 14 Nodes. Reach Equilibrium
     2. Kill 2 Nodes Simultaneously. Reach Equilibrium
     3. Start Nodes, one after the other. Reach Equilibrium
  3. Restart Node
     1. Start 10MM Records 32 Threads, 14 Nodes. Reach Equilibrium
     2. Restart Node using Ops Center. Reach Equilibrium

**Availability Test Plan Detail**

Random “Data Node(s)” Crash & Recovery

1. “Soft” – reset via cluster GUI interface or via OS level restart ie; “sudo init 6”
2. “Hard” – force halt the cluster process/powerdown ie; “kill -9”
3. “Multiple” – 2 nodes simultaneously

Crash & Recovery of “Key” infrastructure

* “Soft” – reset via cluster GUI interface or via OS level restart ie; “sudo init 6”
* HDP Targets Test 1: Active Namenode & Random Datanode
* HDP Targets Test 2: Zookeeper Node & Active Namenode
* HDP Targets Test 3: Zookeeper Node & HBase Master
* C\* Targets Test 1: Any random 2 nodes {consistency=one, replication = 3}
* C\* Targets Test 2: Any random 2 nodes {consistency=quorum, replication=5}