TokuMX VS MongoDB Bake Off Based on a Primary AOL Use case

Monday, November 18, 2013

Agenda

- Scaling opportunities with MongoDB
- What is TokuMX and Why it exists
- What TokuMX does different from MongoDB
- Intent of the test
- The Test Plan
- Summary Results
- Detailed Results
- Conclusions

Why Are we here?
Scaling opportunities with MongoDB

- MongoDB databases tend to take up a lot of space
  - Replicated field names
  - No built in compression
  - Fragmentation
- MongoDB does not support clustered indexes
  - No clustered indexes means scanning through a single users data can mean loading all pages in the DB
- MongoDB doesn't really know how to do I/O
  - Depends on the Virtual memory manager to do DB I/O for it. The Virtual memory manager may or may not do this effectively.
- MongoDB has limited concurrency
  - All database writes (insert, update and delete) require that the entire database be locked. As long as the entire DB is in memory or the update rate is low this is not a problem.

NO! Honestly Why are we here

- We have a MongoDB database with a couple billion smallish very related documents that get scanned in sets of a few thousand at a time. A lot!
- Relational Thinking. With relational databases, good! With MongoDB WRONG!
- Experienced developers use the tools they know. Normalization is good, maybe? With MongoDB WRONG!

What is TokuMX
Why does it exist

- A few years ago TokuTek built a KV storage engine for MySQL
- As the poplarity of MongoDB has grown people have been finding performance and scaling problems with MongoDB .
- The Toku folks decided to mate their KV engine to MongoDB, TokuMX is the result.
- TokuMX includes
  - Native built in compression
  - Clustered indexes to minimize IO when scanning significant parts of the Database..
  - Does its own I/O optimized to stream data, minimizing head movement
  - Talk on TokuTek Fractal key indexes http://cdn.oreillystatic.com/en/assets/1/event/36/How%20TokuDB%20Fractal%20Tree%20Databases%20Work%20Presentation.pdf

What TokuMX does Differently

- Outside of the points on the previous slide
- Concurrency is performed at the document level instead of locking the DB for each update.
- Indexes are all based on Fractal index technology. Not B-Trees. More like Hbase or LevelDB sorted string tables. Will not fragment like MongoDB does.
- Multi-document operations are transactional
- Collection Counts are not stored in the Name Space
- Name spaces are not fixed in size
- Each Collection and Index is stored in its own host file

What Toku Claims

- Database 66% to 80% smaller
- Update concurrency on larger databases up to 25X
- Limited Transactional support for multi document operations
- More consistent operation times, especially when the database significantly exceeds the size of available host memory.
- Dramatically better I/O device utilization.

Intent of the test

- Determine if TokuMX could be used to replace MongoDB to:
  - Reduce cost
  - Improve Reliability
  - For Specific Use cases

High Level test plan

- Steal the data from a single database shard.
- 2025 unique Owners. To limit owner collection sizes the 2025 Owners were altered slightly twice to make 6075 Owners total.
- 16.8 million sample test documents.
- Part I
  - Load the selected data in to both TokuMX 1.0 and MongoDB 2.2.3 using the standard mongo import tool
  - Compare sizes

High Level test plan

- Part 2
  - Extract the owners
  - Extract the documents removing all _id, owner and lid references
  - Using the extracted owners and documents load a database as fast as possible to at least 1X memory size. _id is set to owner + lid for each document written to the SummaryModel collection.
  - Run a simulated full sync program against the db for both MongoDB and TokuMX to determine size, operational performance and resource consumption. No attempt is made to make direct use of the _id field. Ordering and selection is done on the basis of the owner_lid index.
  - Replication is not included in this test.

Hardware/Software used for the tests

- Driver hosts – 6 Mid tier Virtual hosts 4X16
- Target hosts
  - 1 Penguin 4X36 / 1 1TB Sata drive
  - 1 High Tier VM 8X64 / 1 1.28 TB Virtual drive
  - 1 IOPS3 HP DL380E/G8 12X144 / 1.2 TB Fusion IO MLC PCI card
- Software
  - MongoDB 2.2.3
  - TokuMX 1.0 RC
  - Centos 6.2
  - Python 2.6 and Pymongo 2.5.2 for drivers and data extractors
  - Python, Gawk, Toga/argus and Excel for data analysis.

Results

- Size comparisons
- Load Rates
- Sync Test Results

Final Counts and Sizes (Part One revisited)

Host	TokuMX Documents	Mongo Documents	TokuMX Disk Size	Mongo Disk Size	TokuMX Effective Stored Doc size	Mongo Effective Stored Doc size	Compression Ratio
Penguin (C6-10)	103,333,547	36,671,606	50,600,148,992	57,155,511,296	490	1,559	3.18
Virtual 8 CPU	648,327,768	90,138,631	239,365,981,766	126,590,390,278	369	1,404	3.80
Fusion IO 12 CPU	1,034,420,568	206,498,714	367,999,544,380	274,694,406,167	356	1,330	3.74

All size data was taken at the end of the test

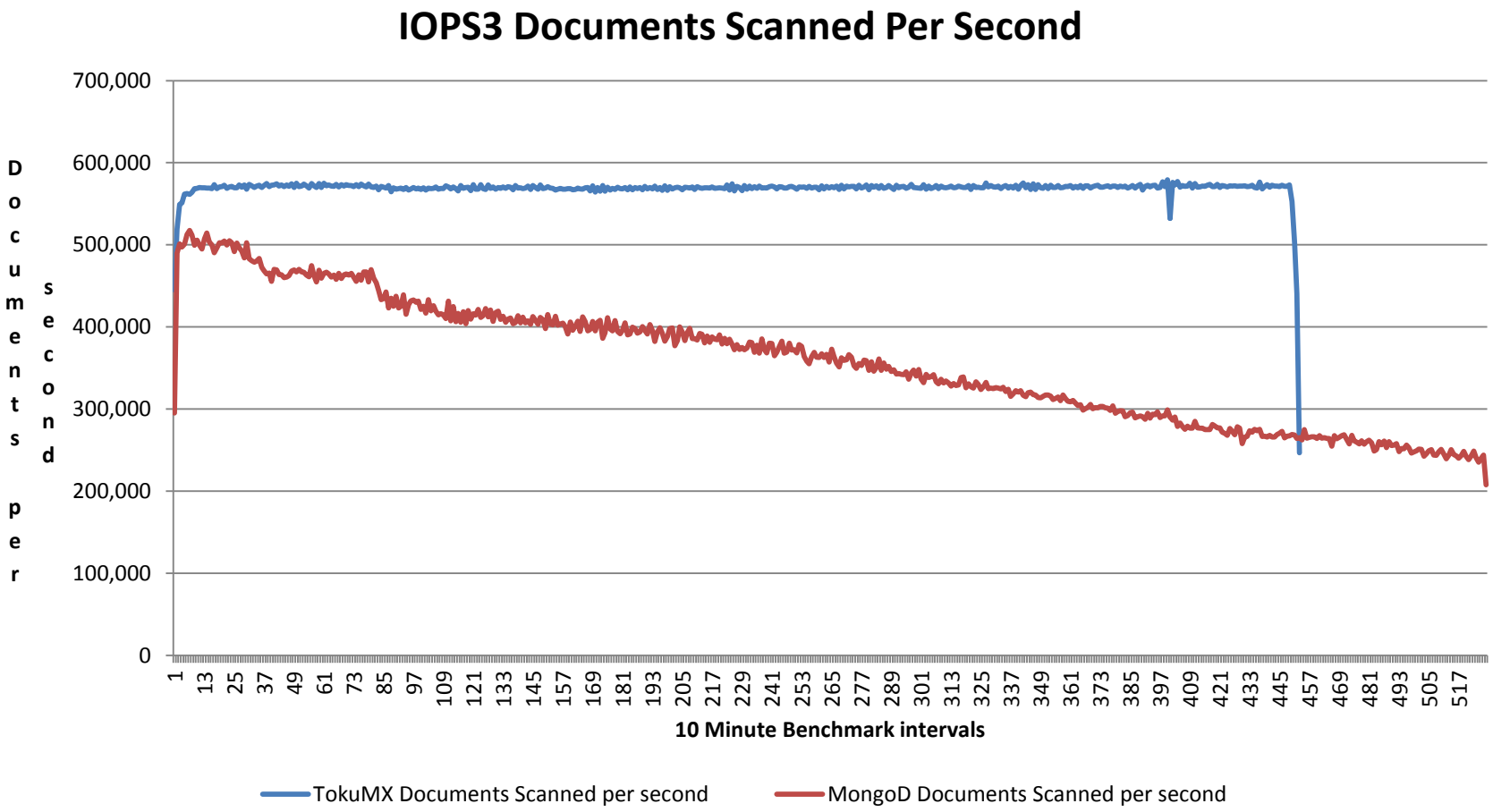
Load Period Comparison

Data loads	TokuMX Inserts per second	Mongo Inserts per second	TokuMX Speedup
Penguin	3,211	N/A	NA
High Tier 8 CPU	8,159	1,189	6.86
IOPS3	12,837	3,228	3.98

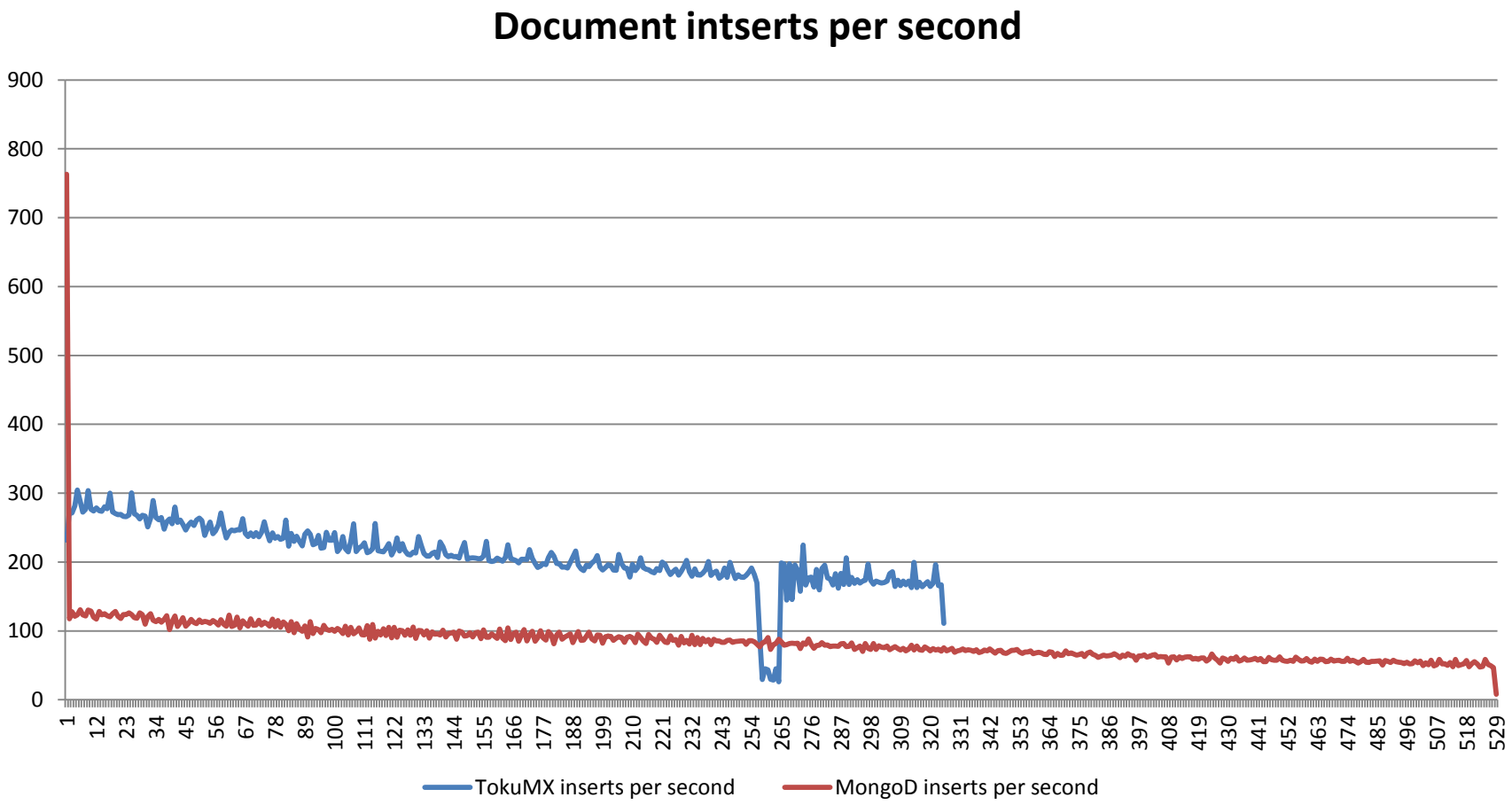
Average Sync Scan Rates

Sync Scan Rates	TokuMX Docs per second	Mongo Docs per second	TokuMX Speedup
Penguin	261,685	137,911	1.90
High Tier 8 CPU	266,122	160,953	1.65
IOPS3	559,764	358,263	1.56

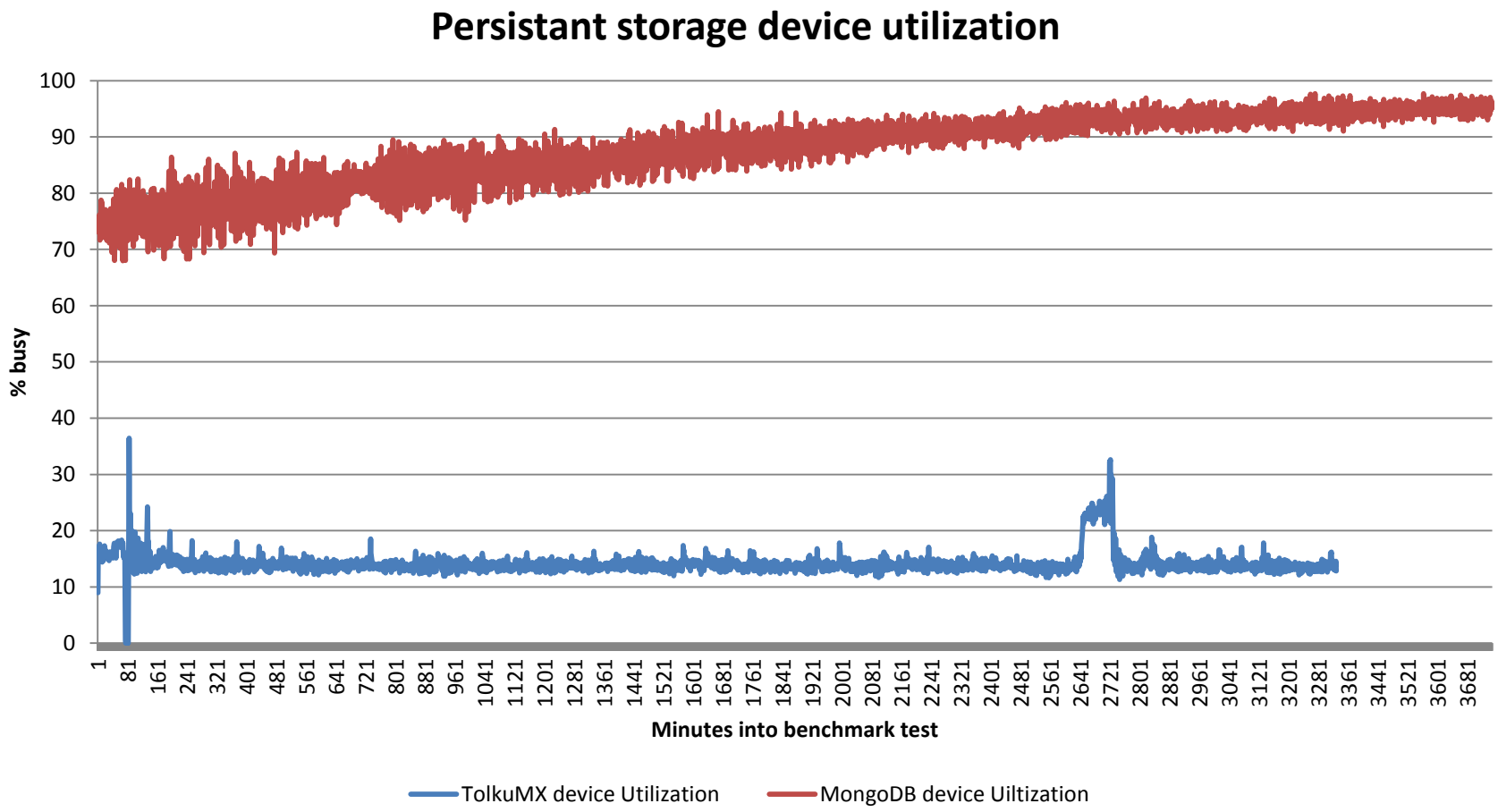
Scanned documents per second (IOPS3)



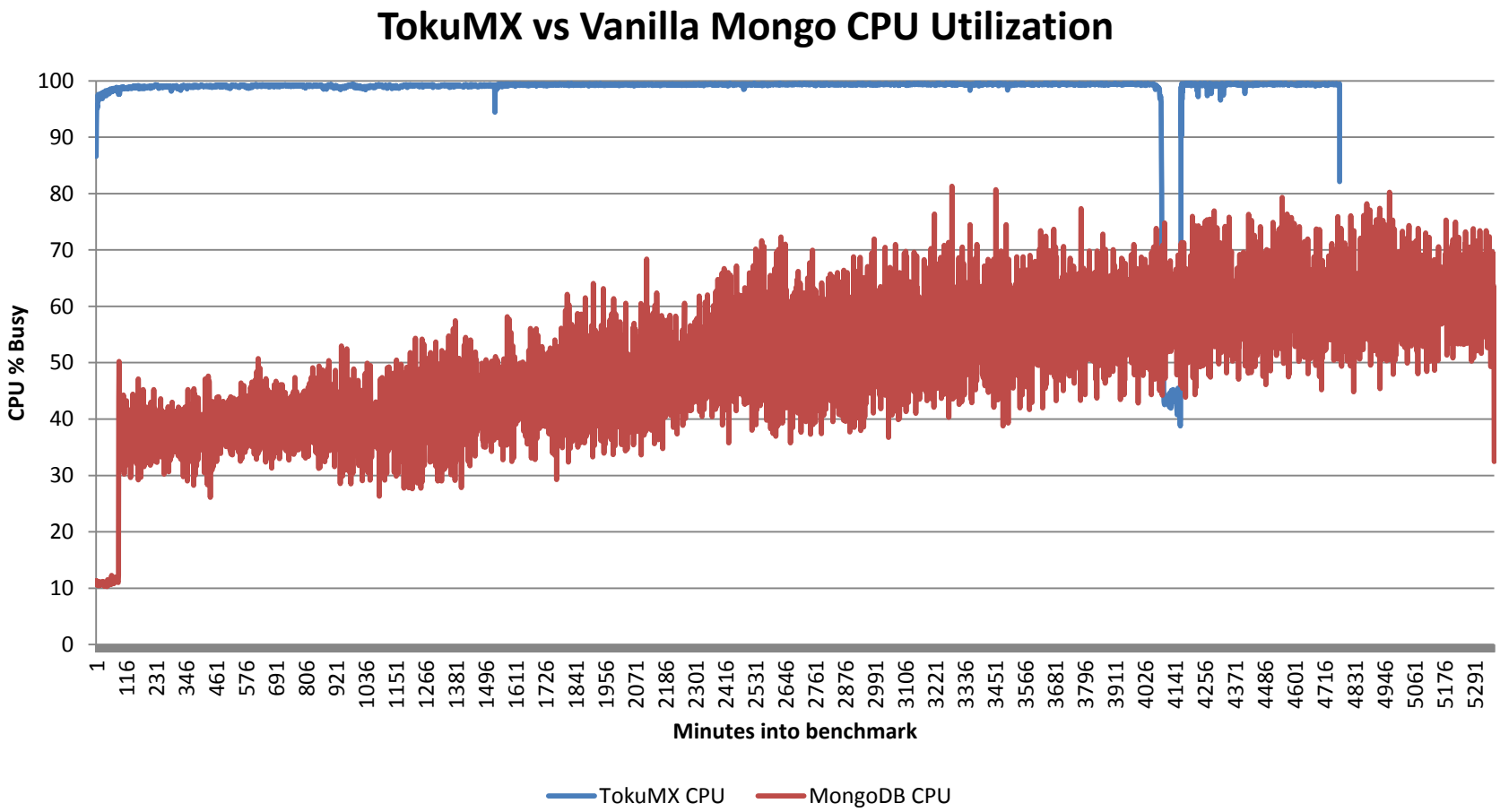
Document Inserts Per second (IOPS3)



Persistent Storage Device Utilization IOPS3



CPU utilization IOPS3



TokuMX Challenges

- Db.collection.count() is very slow compared to MongoDB. This is only for count with no selection argument. --Minor
- Db.collection.stats() for large, close to one billion document collections stats() can be wildly misleading. --Minor
- TokuMX replication while functionally similar to MongoDB is not compatible. E.G. TokuMX servers cannot be replicas for MongoDB

Conclusions

- Space per document for MongoDB databases will be reduced by at least 66%. Likely as much as 75%
- Host memory while important is no longer a serious resource constraint. Now CPUs and to a lesser extent disk I/O bandwidth are the principle constrained resources.
- We should be able to make full use of the available persistent storage on each host.
- It is reasonable to assume that we can put 3X to 4X the amount of data and associated workload on a host compared to MongoDB.
- TokuMX provides more consistent operation times than MongDB does, improving the customer experience.
- TokuMX has the potential to save significant hardware cost