Last Transaction Safety on MongoDB with-out compromising on Performance

In this short white paper, two pioneers collaborate to bring NoSQL to a new level:

- **MongoDB** is a leader in operational database management systems, and specifically in document-based NoSQL. MongoDB has recently released several key performance optimizations within major version 3 and its newly acquired WiredTiger storage engine.
- **Plexistor** is a pioneer in software for non-volatile memory, designed from the ground up to marry the performance of memory with the capacity and cost of FLASH. This architecture was coined as Software Defined Memory (SDM). Plexistor's SDM is available for download.

Optimized MongoDB Performance at Maximal Durability

Traditionally, MongoDB users have had to sacrifice performance for assuring that the data has been written. Plexistor's Software Defined Memory (SDM) accelerates performance for MongoDB by liberating it from the overhead of the ordinary Linux operating system's I/O stack and the constraints of decades-old conventional storage architectures which are overdue to be replaced. Through the revolutionary approach of Plexistor's SDM and NVDIMM-N memory cards, MongoDB performance no longer must be sacrificed to ensure data persistency or durability.

As demonstrated in the benchmarks, Plexistor's SDM software accelerates MongoDB performance by 450% operations per second while cutting down their latency by an order of magnitude without putting any data at risk (i.e., with maximum durability).

Background

Traditionally, there has always been a tradeoff between two desired properties: performance and the safety of the data being written. Data safety is accomplished by means of synchronous Writes associated with all updates. For example, a fast update to the in-memory image of the working data set will still have to wait for confirmation from one or more other destinations where the update is also recorded to persistent storage. These wait times detract from performance, and the penalty is increased when the storage is inefficient or fundamentally slow. Therefore, the tradeoff between performance and data persistence is a common characteristic affecting both NoSQL in general and MongoDB in particular.

To accommodate this tradeoff, MongoDB supports a range of options:

- MongoDB clients can, per I/O, decide to wait for an acknowledgement from the server, and if that acknowledgement should be after the inserted/updated data is persistent or not;
- MongoDB servers can be configured for different levels of data durability, where the increased levels of risk correspond to hypothetically better performance.

United Software Associates™ wrote a benchmarking paper in March 2015 which quantified this tradeoff across several NoSQL databases ("High Performance Benchmarking: MongoDB and NoSQL Systems")[1]. The main optimization factors that defined traditional performance and durability tradeoff were:

(**T**) "Throughput Optimized" (all data written since last checkpoint is at risk);
(**B**) "Balanced" (some data loss [<100 MB] is risked);
(**D**) "Durability Optimized" (no risk of any data loss whatsoever).

United Software Associates denote that configuration (**T**) is considered to be a level of risk much higher than a typical commercial enterprise customer would be likely to commonly implement in production environments. So they focused on configurations (**B**) and (**D**). For the purposes of making an apples-to-apples comparison, we adopted the same classification and focus. We re-ran the experiment in a similar environment and reached similar results and conclusions when we used traditional storage software.

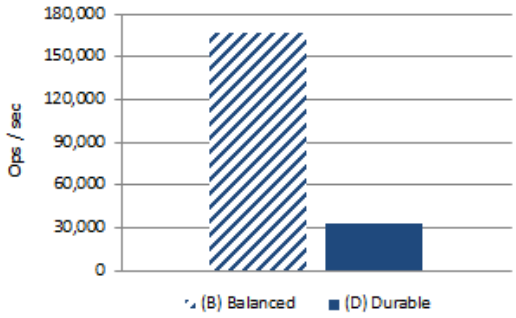

Figure 1 The traditional tradeoff between Performance and Durability

Figure 1 illustrates that there is a clear tradeoff between performance and durability when using traditional storage software. These results demonstrate why many database administrators and application developers strategize recovery methods for "tolerable" amounts of data loss, and longer recovery times because doing so enables urgently needed performance improvements.

Benchmarking Methodology

We leveraged the previous[1] benchmarking methodology using SDM and have shown that the traditional trade-off between performance and durability is no longer significant. As a result, database administrators can eliminate the risk of losing non-persistent data, and application developers no longer have to accept "tolerable data loss" and "long recovery times" in exchange for higher performance and can focus on the business logic.

[1] The United Software Associates paper has no reference number, but is available for download on: https://www.mongodb.com/collateral/comparative-benchmarks-mongodb-vs-couchbase-vs-cassandra

Figure 2 The two physical machines used for MongoDB client and server

Below are example command lines for running the benchmarks:

MongoDB server:
```
numactl --interleave=all
mongod --dbpath=/mnt/dev0 --journal --storageEngine=wiredTiger
--wiredTigerJournalCompressor=none
```
(This flag has negligible effect)

YCSB client:
```
/ycsb-mongodb/bin/ycsb run mongodb -P A -s -threads 40
-p mongodb.database=ycsb -p mongodb.writeConcern=journaled
```

	Baseline	Plexistor SDM
Compute	RHEL 7.1 Linux running on a dual socket XEON E5-2650 v3	
Storage	XFS using: 1. 64GB DDR4 DIMM (Linux Page cache) 2. SanDisk CloudSpeed SATA SSD	Plexistor's SDM using 1. 64GB* DDR4 NVDIMM-N 2. SanDisk CloudSpeed SATA SSD

Table 1 Hardware and platform configurations of the MongoDB server

Results

Performance of maximal durability configurations were evaluated for a balanced 50% read, 50% update workload (YCSB A). MongoDB operations are fully durable (D).

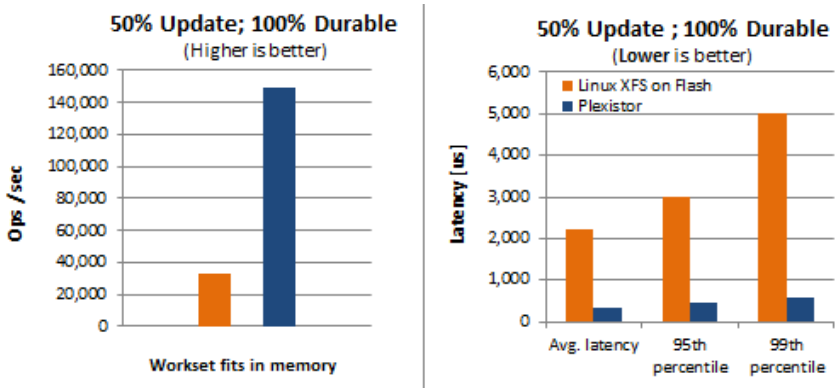

Figure 3 (A) Throughput(operations per second) (B) Latency

Figure 3A reveals that Throughput with Plexistor's SDM was **over 4.5 times better** than Linux XFS using enterprise-grade SATA SSD. The huge throughput improvements achieved with Plexistor SDM are matched with even greater latency reduction. Figure 3B shows that despite serving more operations per second per server, Plexistor SDM **improved operation latency by 6.4 – 8.8 times** compared to the baseline.

Comparing figures 1A and 3 shows that Plexistor's SDM fully durable (D) results are within 10% of the Balanced (B) configuration results. Thus, **SDM provides last-transaction safety without compromising on Performance.**