MongoDB Monitoring and Performance for The Savvy DBA

Key metrics to focus on for day-to-day MongoDB operations

Bimal Kharel

Senior Technical Services Engineer
Percona Webinar
2017-05-23

PERCONA

What I'll cover

- Key commands to get the metrics
- Key metrics to graph and alert on
- Distinguish between MMAPv1 and WiredTiger storage engine metrics wherever appropriate
- Show examples from our own PMM (free, open-source monitoring tool from Percona)

Starting with key commands

In order of usefulness in day-to-day management

- `db.serverStatus()`
- `rs.status()`
- `db.printReplicationInfo()`
- `sh.status()`
- `db.stats()`

PERCONA

Operating system monitoring

OS level metrics you should set up alerts on and graph for easy trend identification

- disk utilization
- load average and CPU queue
- memory and possibly swapping
- I/O utilization or a combination of load and latency

Data and operations growth - 1

sum up the collection sizes

```
db.getMongo().getDBNames().forEach(function(d) {
  var curr_db = db.getSiblingDB(d);
  var total_size = 0;
  curr_db.getCollectionNames().forEach(function(coll) {
    var coll_size =
curr_db.getCollection(coll).stats().storageSize;
    total_size = total_size + coll_size;
  });
  print(d + ": " + total_size/(1024*1024));
});;
```

- Run the above against the admin database

Data and operations growth - 2

Keep track of operations and alert if they reach N times your normal

- `db.serverStatus()`

```
- opcounters
- metrics.document
- metrics.commands
```

example (some output trimmed)

```
replset:PRIMARY>
db.serverStatus().opcounters
{
        "insert" : 99992,
        "query" : 10,
...
}

replset:PRIMARY>
db.serverStatus().metrics.document
{
        "deleted" : NumberLong(0),
        "inserted" : NumberLong(99992),
        "returned" : NumberLong(362720),
        "updated" : NumberLong(0)
}
```

```
replset:PRIMARY>
db.serverStatus().metrics.commands
{
...
        "insert" : {
                "failed" : NumberLong(0),
                "total" : NumberLong(50046)
...
        "serverStatus" : {
                "failed" : NumberLong(0),

                "total" : NumberLong(5)
        },
  ...
}
```

Journaling

Journaling is on by default and should be left on. It is a write-ahead log that persists writes to disk faster than committing to the database

- For MMAP it will let the node recover data lost within 60s of a crash
- In WiredTiger it occurs every 50ms (100ms prior to 3.2) so it narrows the window of data loss even further as checkpoints are taken every 60s.

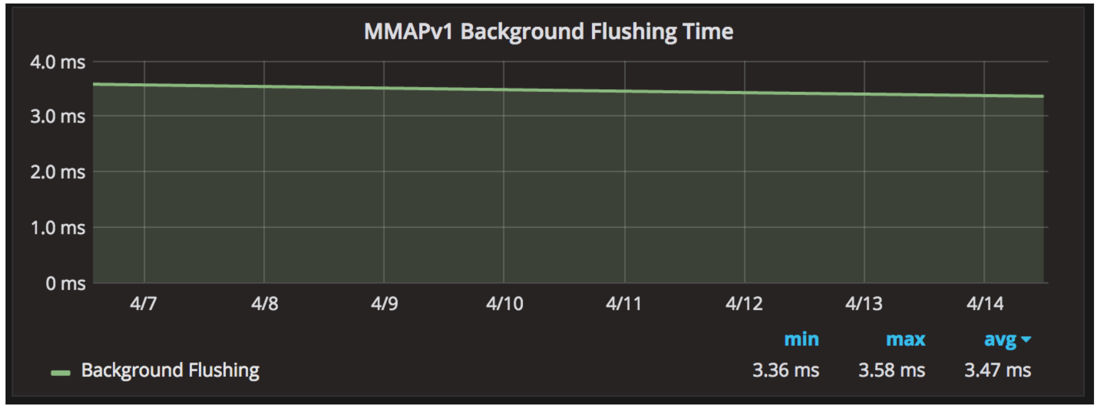
flushing from memory to disk

For MMAP

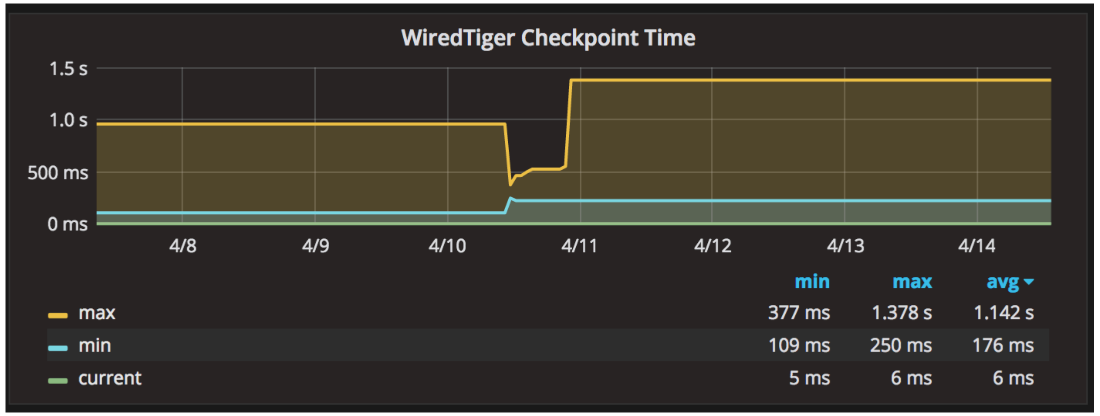
- `db.serverStatus() - backgroundFlushing`

For WiredTiger

- `db.serverStatus() - wiredTiger.transaction`



	min	max	avg
Background Flushing	3.36 ms	3.58 ms	3.47 ms



	min	max	avg
max	377 ms	1.378 s	1.142 s
min	109 ms	250 ms	176 ms
current	5 ms	6 ms	6 ms

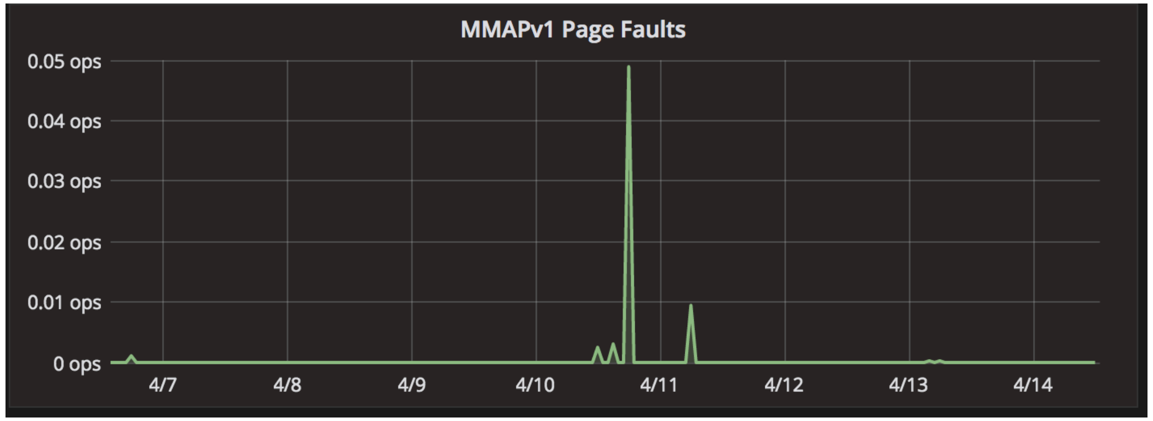
Memory to disk operations

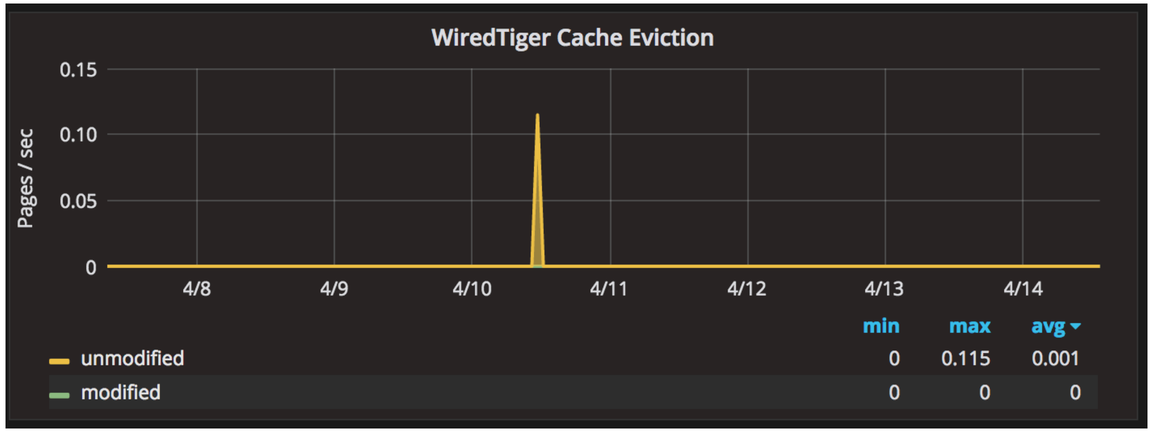
For MMAP

- `db.serverStatus() - extra_info.page_faults`

For WiredTiger

- `db.serverStatus() - wiredtiger.cache`





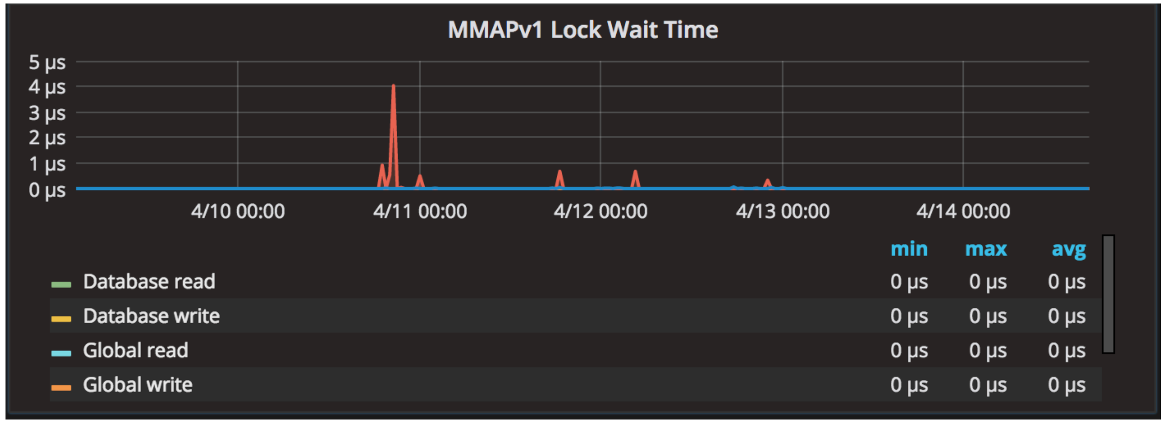
	min	max	avg
unmodified	0	0.115	0.001
modified	0	0	0

locking and tickets - 1

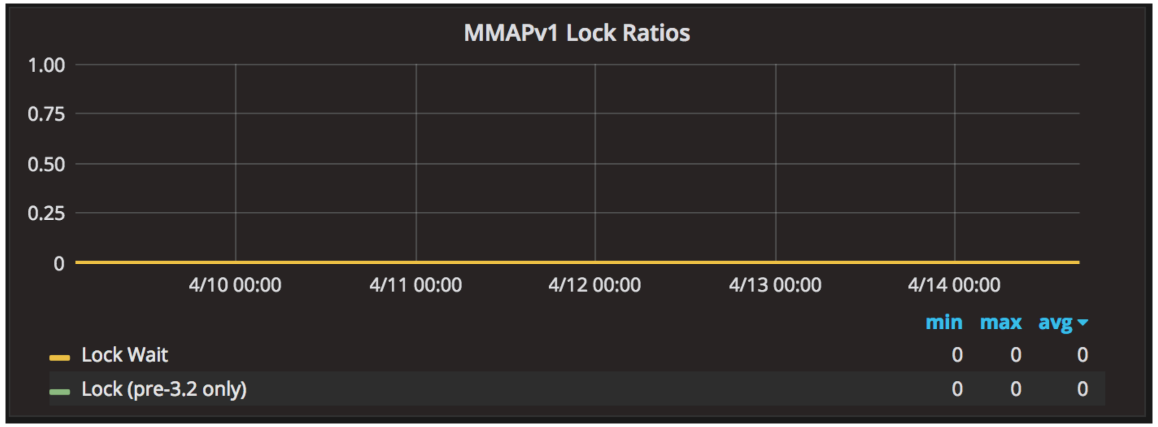
For MMAP

- `db.serverStatus()`
  - `globalLock`
  - `locks`

- `locks timeAcquiringMicros` and `acquireWaitCount` can help you spot trends in average lock times



	min	max	avg
Database read	0 µs	0 µs	0 µs
Database write	0 µs	0 µs	0 µs
Global read	0 µs	0 µs	0 µs
Global write	0 µs	0 µs	0 µs



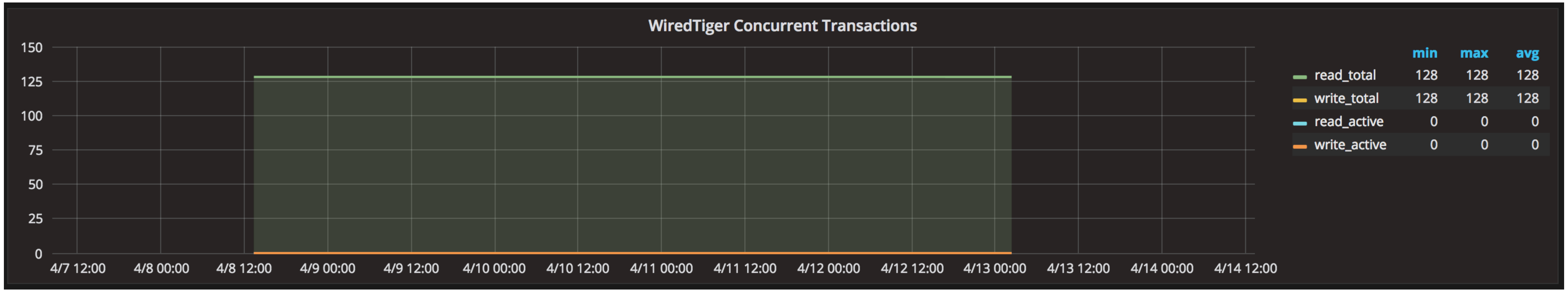
	min	max	avg
Lock Wait	0	0	0
Lock (pre-3.2 only)	0	0	0

PERCONA

locking and tickets - 2

For WiredTiger

- `db.serverStatus()`
  `- wiredTiger.concurrentTransactions`



	min	max	avg
read_total	128	128	128
write_total	128	128	128
read_active	0	0	0
write_active	0	0	0

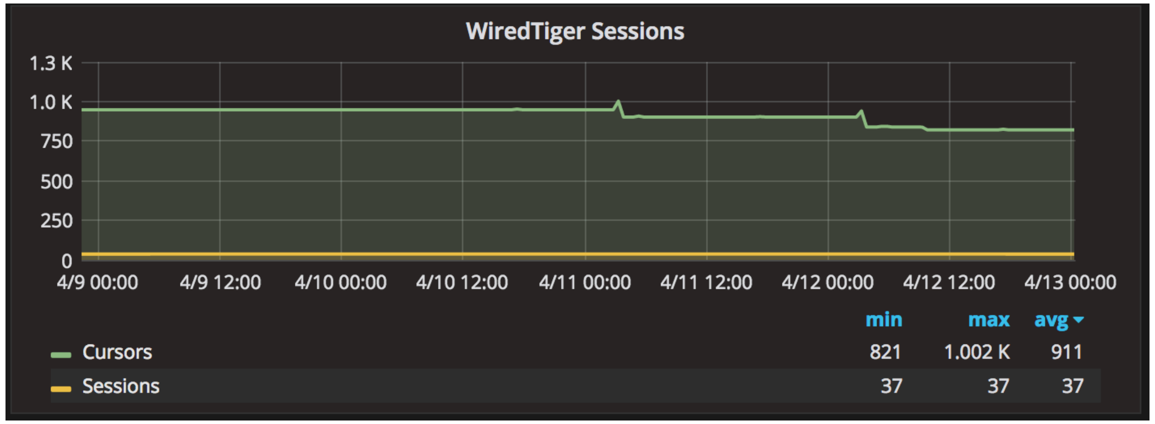
connections, cursors and sessions - 1

- Badly designed apps will create a new connection for every query
- Each connection has a 1MB overhead so this can add up quickly
- All major drivers provide connection pooling
- `db.serverStatus()`
  - `globalLock.activeClients`
  - `connections`
  - `metrics.cursor`

connections, cursors and sessions - 2

For WiredTiger in PMM we monitor sessions

```
db.serverStatus()
- wiredTiger.session
```

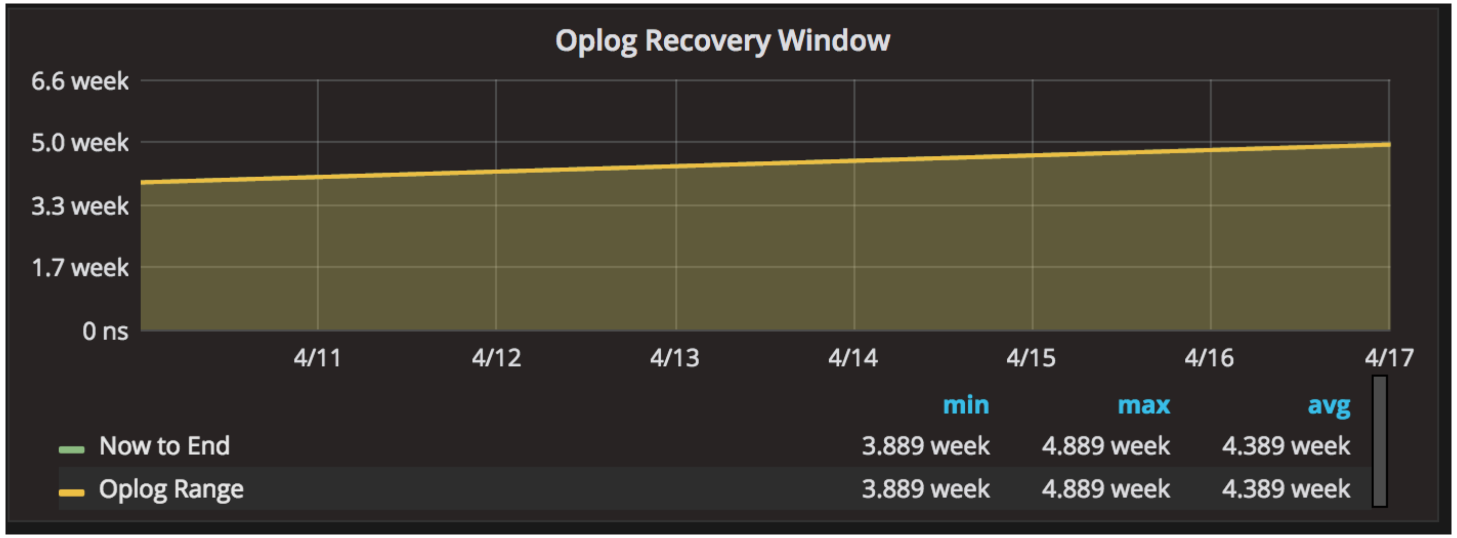


	min	max	avg
🟩 Cursors	821	1.002 K	911
🟨 Sessions	37	37	37

Replication metrics

Get information about the operations log (oplog)

- `db.getReplicationInfo()`
  - `logSizeMB`
  - `usedMB`
  - `timeDiffHours`



	min	max	avg
Now to End	3.889 week	4.889 week	4.389 week
Oplog Range	3.889 week	4.889 week	4.389 week

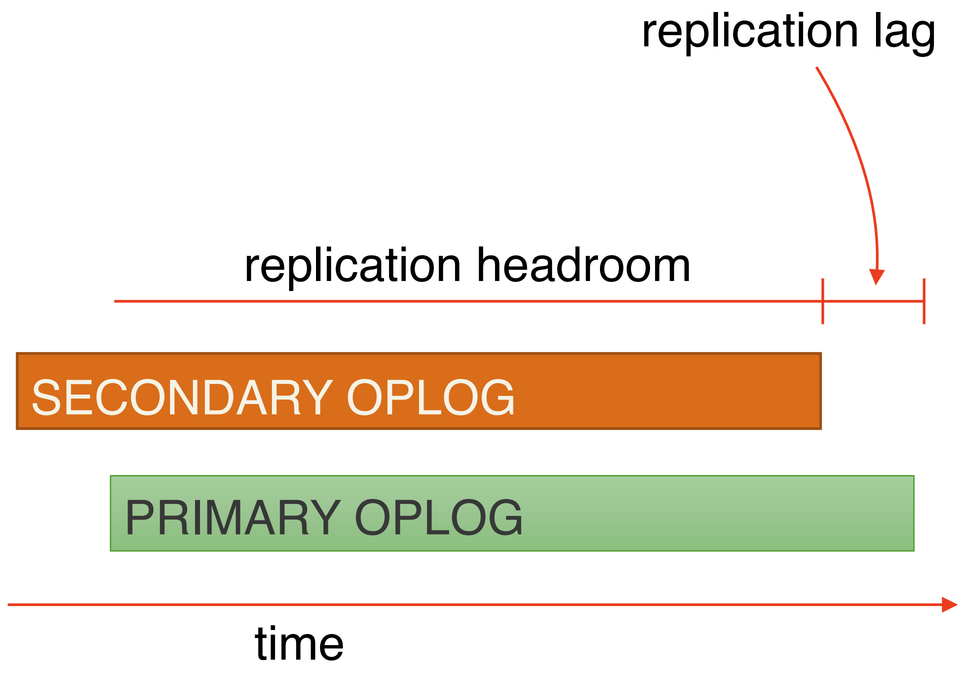
replication lag and headroom -1

Lag is a derived value

- `rs.status()`
  - `members[].optimeDate`
- it is the difference of between the Primary and the Secondary nodes

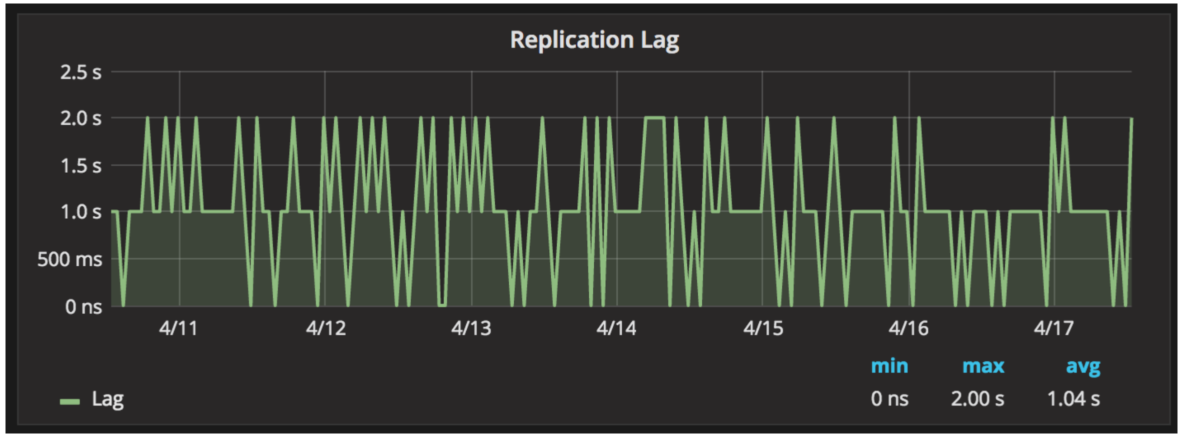
Headroom is also a derived value

- `db.getReplicationInfo()`
  - `(timeDiffHours - lag (converted to hours))`

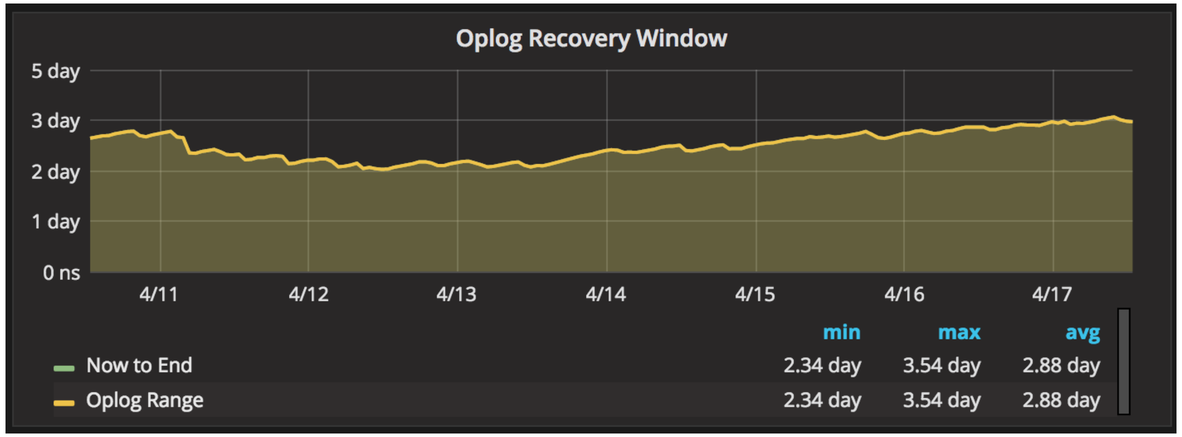


replication lag and headroom -2

Replication lag and headroom graphs taken from PMM



	min	max	avg
🟩 Lag	0 ns	2.00 s	1.04 s



	min	max	avg
🟩 Now to End	2.34 day	3.54 day	2.88 day
🟨 Oplog Range	2.34 day	3.54 day	2.88 day

sharding metrics - 1

Run against a mongos instance

- `sh.status()`

  This returns a report rather than JSON so you may have to do additional parsing or opt for

  ```
  >use config
  ```

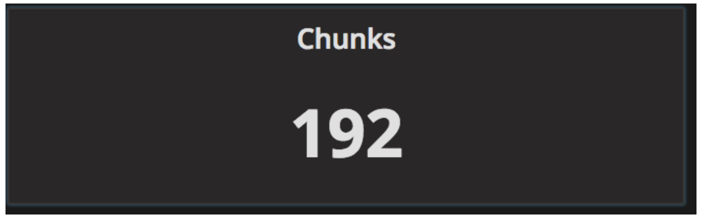
  and run queries against the `chunks`, `collections` and `shards` collections to access the metrics you want

Balancer commands

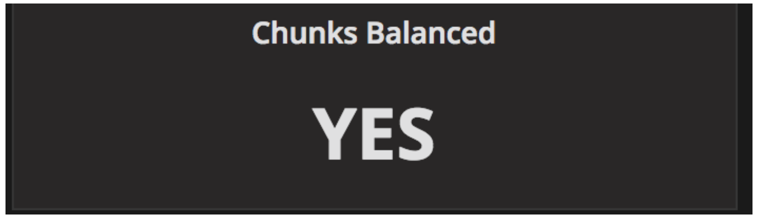
-
  ```
  - sh.getBalancerState()
  - sh.isBalancerRunning()
  ```

sharding metrics - 2

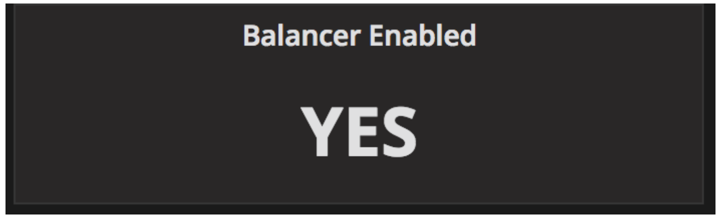
Some sharding metrics from PMM


Chunks

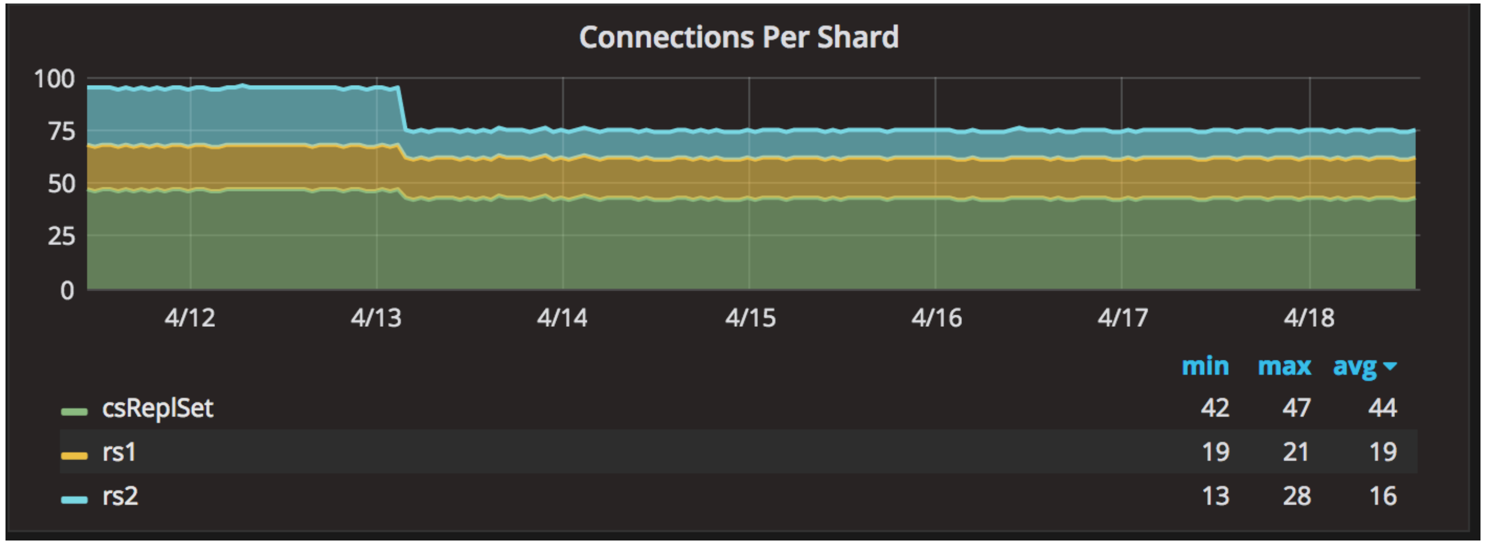
192


Chunks Balanced

YES


Balancer Enabled

YES



	min	max	avg ▾
▬ csReplSet	42	47	44
▬ rs1	19	21	19
▬ rs2	13	28	16

Time for questions and links

PMM - Percona Monitoring and Management

- https://www.percona.com/software/database-tools/percona-monitoring-and-management

About me:

- Bimal Kharel
- bimal.kharel@percona.com
- 1-737-346-2418

About Percona

- Solutions for your success with MySQL and MongoDB
- Support, Managed Services, Software
- Our Software is 100% Open Source
- Support Broad Ecosystem – MySQL, MariaDB, Amazon RDS
- In Business for 10 years
- More than 3000 customers, including top Internet companies and enterprises

PERCONA

Database Performance Matters