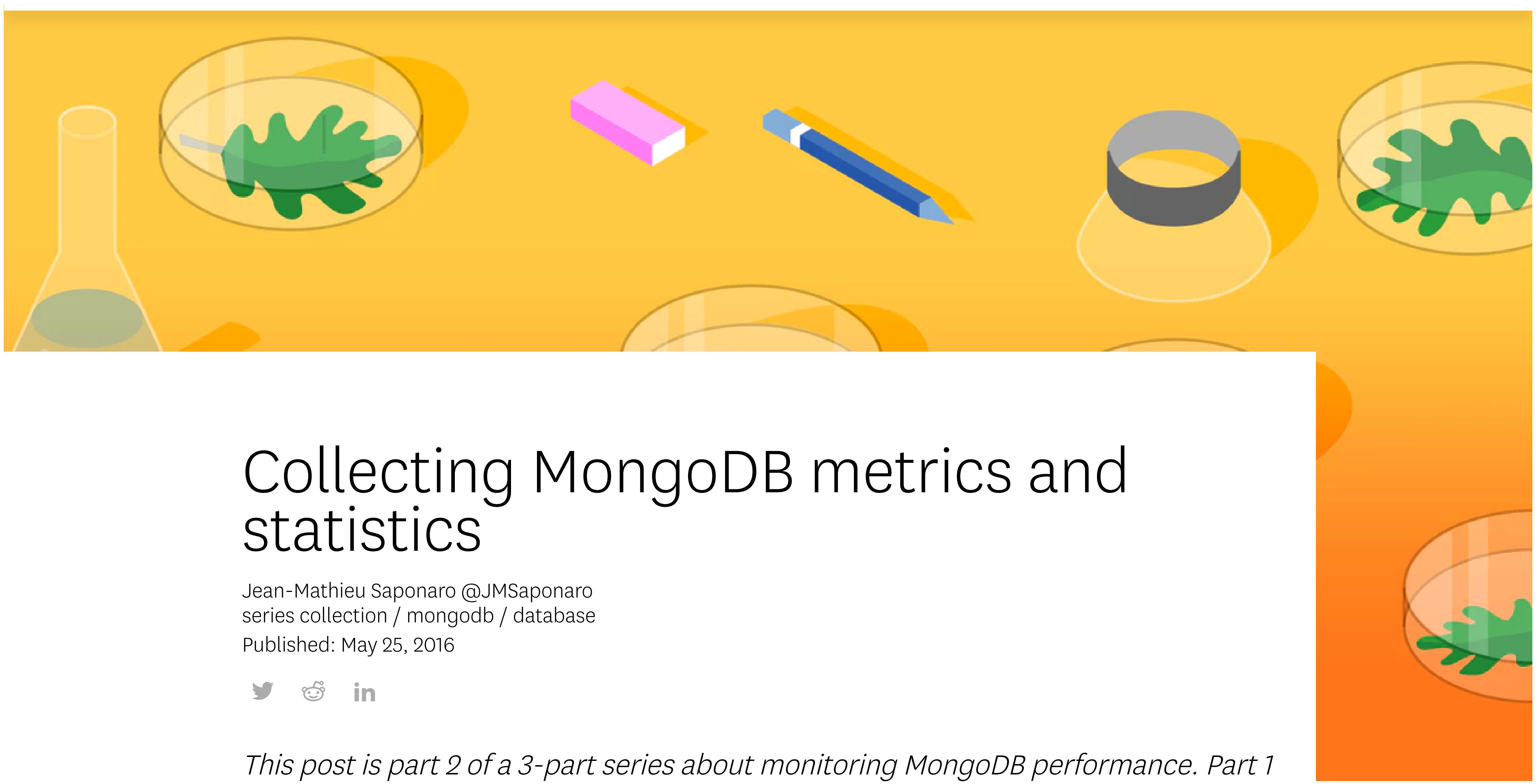
Join us at the **Dash** conference! July 11-12, NYC

DATADOG



Collecting MongoDB metrics and statistics

Jean-Mathieu Saponaro @JMSaponaro
series collection / mongodb / database
Published: May 25, 2016

This post is part 2 of a 3-part series about monitoring MongoDB performance. Part 1 presents the key performance metrics available from MongoDB: there is one post for the WiredTiger storage engine and one for MMAPv1. In Part 3 you will discover how to monitor MongoDB performance with Datadog.

Join us at the **Dash** conference! July 11-12, NYC

order to monitor them. There are three ways to collect MongodDB metrics from your hosts:

- Using utilities offered by MongoDB to collect real-time activity statistics
- Using database commands to check the database's current state
- Using a dedicated monitoring tool for more advanced monitoring features and graphing capabilities, which are essential for databases running in production

Utilities

Utilities provide real-time statistics on the current activity of your MongoDB cluster. They can be useful for ad hoc checks, but to get actionable insights and more advanced monitoring features, you should check the last section about dedicated monitoring tools.

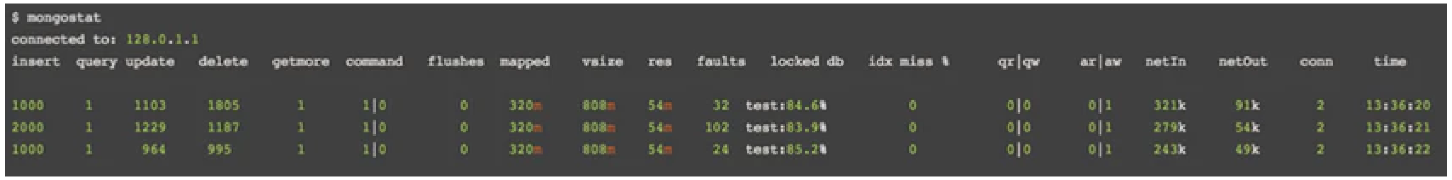
The two main utilities line are **mongostat** and **mongotop**.

mongostat

mongostat is the most powerful utility. It reports real-time statistics about connections, inserts, queries, updates, deletes, queued reads and writes, flushes,

Join us at the **Dash** conference! July 11-12, NYC

However **mongostat** does not provide insights on metrics about Replication and oplog, cursors, storage, resource saturation, asserts, or host-level metrics. **mongostat** returns cache statistics only if you use the WiredTiger storage engine.



```
$ mongostat
connected to: 128.0.1.1
insert  query update   delete   getmore  command   flushes  mapped    vsize   res    faults   locked db   idx miss %     qr|qw     ar|aw   netIn     netOut     conn      time

1000     1     1103     1805        1       1|0          0     320m     808m    54m        32  test:84.6%          0         0|0       0|1    321k       91k        2    13:36:20
2000     1     1229     1187        1       1|0          0     320m     808m    54m       102  test:83.9%          0         0|0       0|1    279k       54k        2    13:36:21
1000     1      964      995        1       1|0          0     320m     808m    54m        24  test:85.2%          0         0|0       0|1    243k       49k        2    13:36:22
```

You can find in the MongoDB documentation the meaning of the different fields returned by mongostat along with the available options.

mongostat relies on the `db.serverStatus()` command (see below).

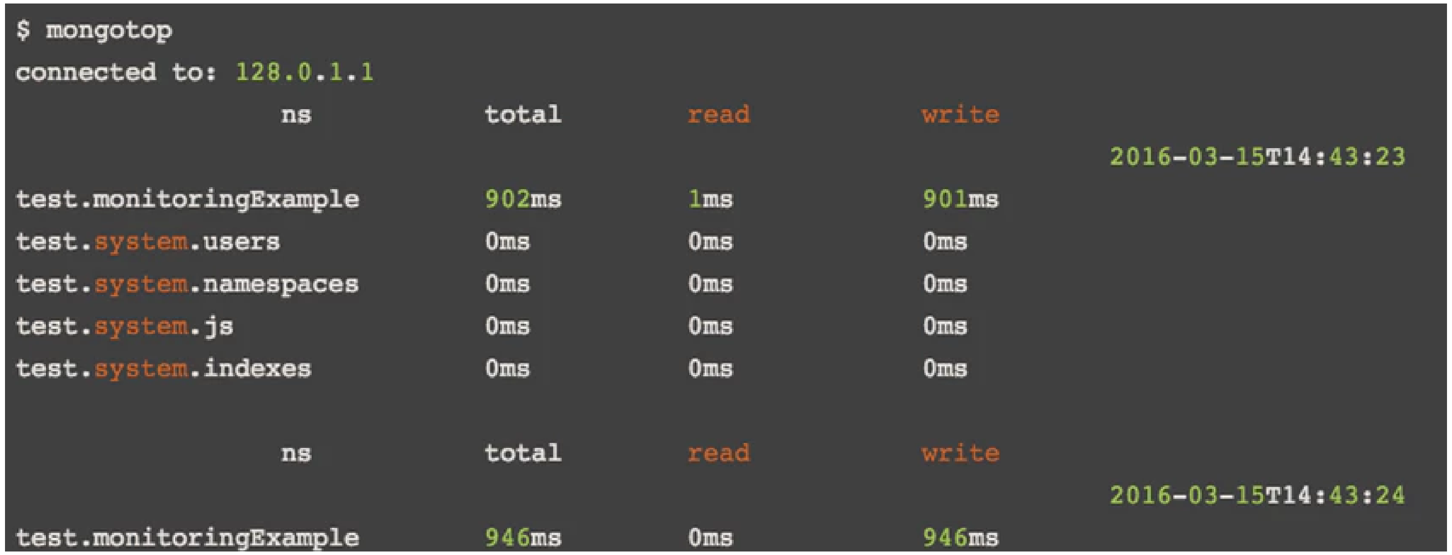
NOTE: Prior version 3.2, MongoDB offered an HTTP console displaying monitoring statistics on a web page, but this has been deprecated since v3.2.

mongotop

mongotop returns the amount of time a MongoDB instance spends performing read and write operations. It is broken down by collection (namespace). This allows you to

Join us at the **Dash** conference! July 11-12, NYC

DATADOG

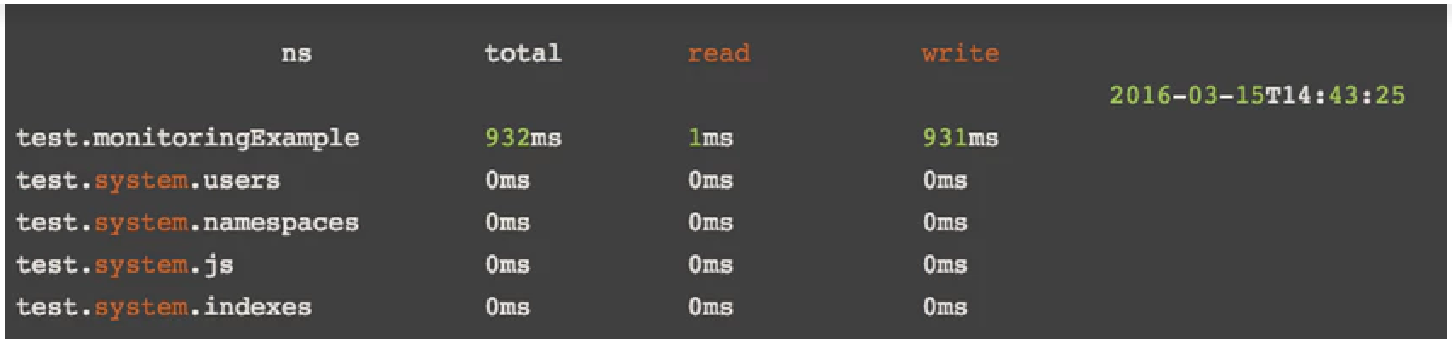


```
$ mongotop
connected to: 128.0.1.1
                    ns            total          read           write
                                                                              2016-03-15T14:43:23
test.monitoringExample            902ms          1ms            901ms
test.system.users                 0ms            0ms            0ms
test.system.namespaces            0ms            0ms            0ms
test.system.js                    0ms            0ms            0ms
test.system.indexes               0ms            0ms            0ms

                    ns            total          read           write
                                                                              2016-03-15T14:43:24
test.monitoringExample            946ms          0ms            946ms
```

Join us at the **Dash** conference! July 11-12, NYC

DATADOG



```
                     ns              total          read            write
                                                                                2016-03-15T14:43:25
test.monitoringExample               932ms          1ms             931ms
test.system.users                    0ms            0ms             0ms
test.system.namespaces               0ms            0ms             0ms
test.system.js                       0ms            0ms             0ms
test.system.indexes                  0ms            0ms             0ms
```

By default, values are printed every second but you can specify the frequency. For example if you want it to return every 20 seconds, you can run mongotop 20. Many other options are available as well.

Utilities are great for quick checks and ad hoc investigations, but for more detailed insights into the health and performance of your database, explore MongoDB commands discussed in the next section.

Commands

MongoDB provides several commands that can be used to collect the different metrics from your database presented in Part 1. Here are the most useful ones.

serverStatus

Join us at the **Dash** conference! July 11-12, NYC

DATADOG

connections, operations, journaling, background flushing, locking, cursors, memory, asserts, etc. You can find the full list of metrics it can return here.

This command is used by most third party monitoring tools to collect MongoDB metrics along with the dbStats and replSetGetStatus commands that are still necessary to collect storage metrics and statistics about your replica sets (see next paragraphs).

dbStats

dbStats (`db.stats()` in the mongo shell) provides metrics about storage usage of the database: number of objects, or memory taken by documents and padding in the database (see memory metrics in Part 1 of this series). Here is the full list of metrics it returns.

collStats

collStats (`db.collection.stats()` in the shell) returns metrics similar to the dbStats output but for a specified collection: size of a collection, number of objects inside it, average size of objects, number of indexes in the collection, etc. See the full list here.

Join us at the **Dash** conference! July 11-12, NYC

db.runCommand({ collStats : “restaurant”, scale: 1024 })

getReplicationInfo

getReplicationInfo (`db.printReplicationInfo()` in the shell) returns metrics about oplogs of the different members of a replica set like the oplog size or the oplog window. See the list of output fields here.

replSetGetStatus

replSetGetStatus (`rs.status()` from the shell) reports metrics about members of your replica set: state, metrics required to calculate replication lag. See Part 1 for more info about these metrics. This command is used to check the health of a replica set’s members and make sure replication is correctly configured. You can find the full list of metrics of the output here.

sh.status

Sh.status (`sh.status()` from the shell) provides metrics about sharding configuration and existing chunks (contiguous range of shard key values in a specific shard) for a

Join us at the **Dash** conference! July 11-12, NYC

getProfilingStatus (`db.getProfilingStatus()` in the shell) returns the current profile level and the defined threshold above which the profiler considers a query slow (slowOpThresholdMs).

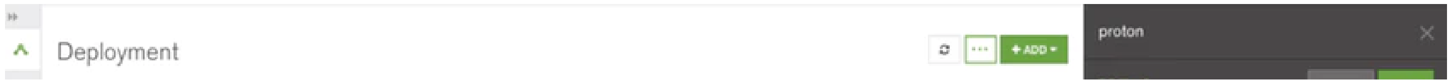
Production monitoring

The first two sections of this post cover built-in ways to manually access MongoDB metrics using simple lightweight tools. For databases running in production, you will likely want a more comprehensive monitoring system that ingests MongoDB metrics as well as metrics from other technologies in your stack.

MongoDB's own tools

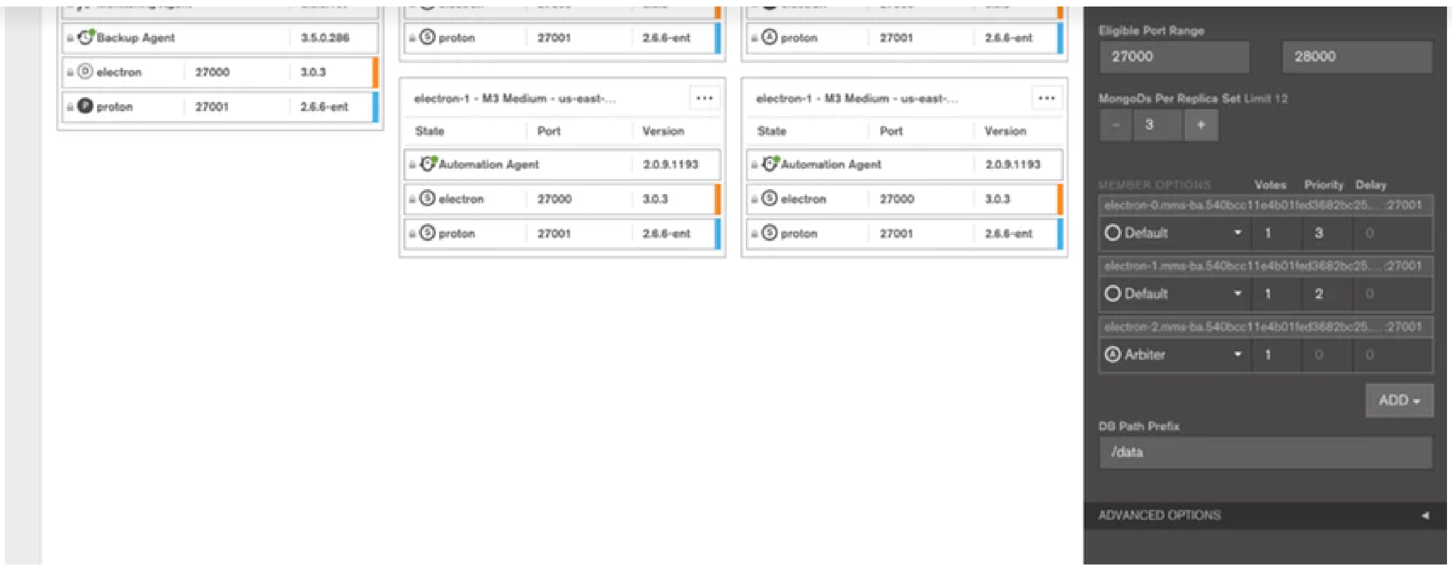
With MongoDB Enterprise Advanced, you will be able to collect performance metrics, automate, and backup your deployment through MongoDB's management tools:

- Ops Manager is the easiest way to manage MongoDB from your own data center
- Cloud Manager allows you to manage your MongoDB deployment through MongoDB's cloud service



Join us at the **Dash** conference! July 11-12, NYC

DATADOG



Backup Agent		3.5.0.286
electron	27000	3.0.3
proton	27001	2.6.6-ent

proton	27001	2.6.6-ent

proton	27001	2.6.6-ent

electron-1 - M3 Medium - us-east-...

State	Port	Version
Automation Agent		2.0.9.1193
electron	27000	3.0.3
proton	27001	2.6.6-ent

electron-1 - M3 Medium - us-east-...

State	Port	Version
Automation Agent		2.0.9.1193
electron	27000	3.0.3
proton	27001	2.6.6-ent

Eligible Port Range: 27000 – 28000

MongoDs Per Replica Set Limit 12: 3

MEMBER OPTIONS	Votes	Priority	Delay
electron-0.mms-ba.540bcc11e4b01fed3682bc25... :27001			
Default	1	3	0
electron-1.mms-ba.540bcc11e4b01fed3682bc25... :27001			
Default	1	2	0
electron-2.mms-ba.540bcc11e4b01fed3682bc25... :27001			
Arbiter	1	0	0

DB Path Prefix: /data

If you have it, MongoDB Ops Manager will likely be your go-to place to take actions to monitor, prevent or resolve MongoDB performance issues.

Visibility into all your infrastructure with Datadog

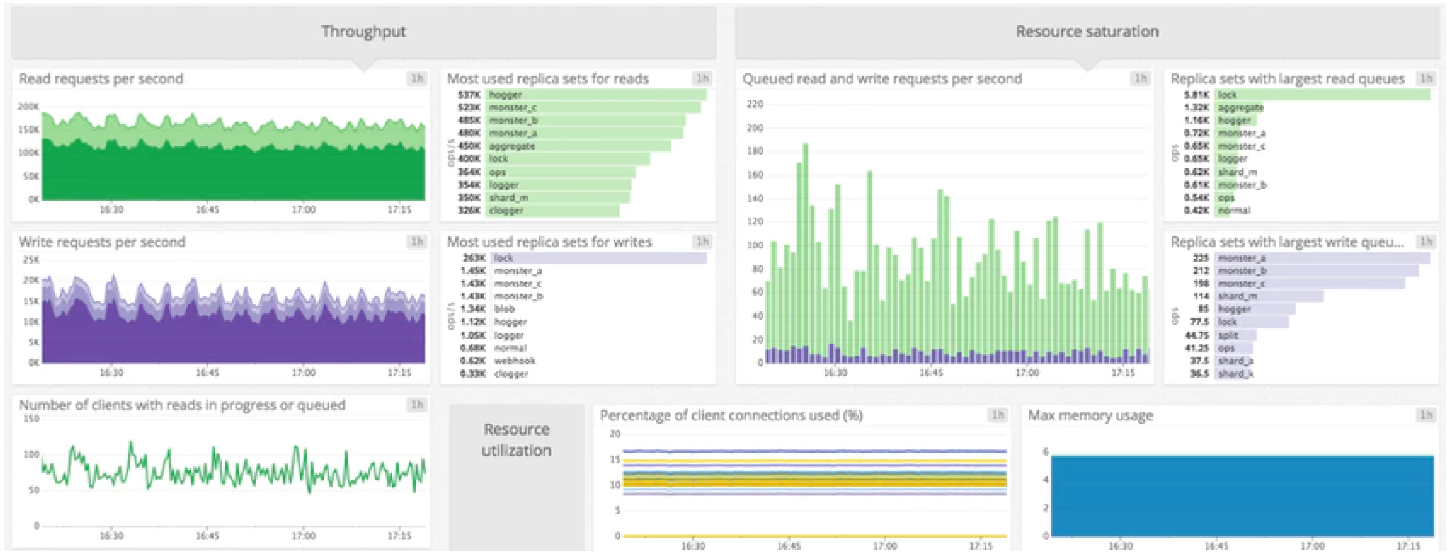
At Datadog, we worked with MongoDB's team to develop a strong integration. Using Datadog you can start collecting, graphing, and monitoring all MongoDB metrics from your instances with a minimum of overhead, and immediately correlate what's happening in MongoDB with the rest of your stack

Datadog offers extended monitoring functionality, such as:

Join us at the **Dash** conference! July 11-12, NYC

DATADOG

Advanced alerting mechanisms


Throughput
Resource saturation
Read requests per second
Most used replica sets for reads
Queued read and write requests per second
Replica sets with largest read queues
Write requests per second
Most used replica sets for writes
Replica sets with largest write queu...
Number of clients with reads in progress or queued
Resource utilization
Percentage of client connections used (%)
Max memory usage

Join us at the **Dash** conference! July 11-12, NYC

DATADOG

For more details, check out our guide to monitoring MongoDB metrics with Datadog in the third and last part of this series.

Source Markdown for this post is available on GitHub. Questions, corrections, additions, etc.? Please let us know.

Want to write articles like this one? Our team is hiring!

Join us at the **Dash** conference! July 11-12, NYC

DATADOG

Explore key steps for implementing a successful cloud-scale monitoring strategy.

DOWNLOAD TO LEARN MORE

Related Posts

How Connectifier unfroze MongoDB with Datadog

Monitoring MongoDB performance metrics (WiredTiger)

Join us at the **Dash** conference! July 11-12, NYC

DATADOG

Start monitoring your metrics in minutes

FIND OUT HOW

FREE TRIAL

Product

Features
APM
Log Management
Integrations
Alerts
API
Pricing

Documentation
Support
Resources
Security

About

Contact Us
Partners

Social

Blog
Español

Join us at the **Dash** conference! July 11-12, NYC

DATADOG

© Datadog 2018

Terms | Privacy | Cookies