{ topic: "Real-Time Analytics Schema Design and Optimization" }

{ by: "Ryan Nitz" }

Overview of Problem

Service to report real-time server statistics (counters, gauges and derived)

Provide historical view of all data collected

Ability to send alerts based on data collected

Dynamic charts and dashboards

Fairly consistant amount of data being collected each day

Needs to be **F-A-S-T**

Service is **free** so it needs to be **C-H-E-A-P** to run

Initial Solution

Original solution looked good, but had issues:

```
{
    _id: "20110620-bits-mem-1499f25476f6e905ca13a6f6f9e97b32",
    cid: ObjectId("4d1b6314e528c81a1f200e03"),
       d: "20110620",
       g: "mem",
        i: "bits",
     dy: { t: 1444, n: 1440 },
     hy: {
            "0": { t: 60, n: 60 },
            "1": { t: 60, n: 60 },
            ....
           "23": { t: 60, n: 60 }
     },
     mn: {
           "0": { t: 1, n: 1 },
           "1": { t: 1, n: 1 },
           ....
    "1440": { t: 1, n: 1 }
}
```

Initial Solution Sample Code

Java App, but Python is easier to read :-)

```
query = { "_id": "20110620-bits-mem-1499f25476f6e905ca13a6f6f9e97b32" }

toSet = { "g": "mem", "i": "bits", "d": "20110620" }

toInc = { "dy.t": long(1), "dy.n": 1, "hy.20.t": long(1), "hy.20.n": 1, "mn.1200.t" : long(1), "mn.
1200.n": 1 }

col.update(query, { "$set" : toSet, "$inc": toInc }, upsert=True)
```

Initial Issues

Too many database copies/moves

Performance degraded throughout the day

Massive load spike once per day (00:00:00 GMT)

Initial suggestion: Add Hardware and keep blasting out functionality

Reality: This is a free service

Progressively Slower

Overall system performance degraded throughout the day

Solution - break up the minute document by hour:

```
{
    _id: "20110620-bits-mem-1499f25476f6e905ca13a6f6f9e97b32",
     mn: {
          "0": {
                    "0" { t: 1, n: 1}
                    ....
                    "59" { t: 1, n: 1}
           }
          ....
          "23": {
                    "1380" { t: 1, n: 1}
                    ....
                    "1439" { t: 1, n: 1}
           }
}
```

The 00:00:01 Problem

Each group/identity allocates one document per day

Massive amounts of inserts during the first minute of the day

Scaling for one minute a day does not make sense

Solution:

Pre-allocate an empty document for the next day

```
if random.nextInt( 720 ) == 5:
   Create empty next day doc if does not exist
```

Next Problem: Slow Query Time

Inserts/updates are now working great

Querying the same day is great

Historical queries are slow

Accessing multiple documents will likely result in a disk seek

~ 5 - 10 ms per seek (EBS can be challenging)

Historical day charts need 1 - 365 documents

365 * 5 ms == 1.8 seconds (if every doc acces requires a seek)

Slow Query Solution

Move the hourly and daily stats to separate month docs

Pre-allocate in a similar manor as day documents

Hour doc:

```
{
    _id: "20110620-bits-mem-1499f25476f6e905ca13a6f6f9e97b32",
     d: {
          "1": {
                    "0" { t: 1, n: 1}
                    ....
                    "23" { t: 1, n: 1}
           }
          ....
          "30": {
                    "0" { t: 1, n: 1}
                    ....
                    "23" { t: 1, n: 1}
           }
}
```

Slow Query Solution (Day)

Day doc:

```
{
    _id: "20110620-bits-mem-1499f25476f6e905ca13a6f6f9e97b32",
     d: {
          "0": { t: 1, n: 1}
          ....
          "30": { t: 1, n: 1}
      }
}
```

Additional Tweaks

Moved config and data collections to separate servers

Added a bloom filter for next month/day docs

Added another bloom filter for first 10 minutes of the day

Parallelized the queries on a page by using HTTP trickery

Reduced the frequency of metadata $set calls on nodes

Working on a Java extension optimized for reads/queries

Why Not shard?

Additional machines cost more $$$

Scaling system vertically first

Eventually, the system will be sharded as dataset grows

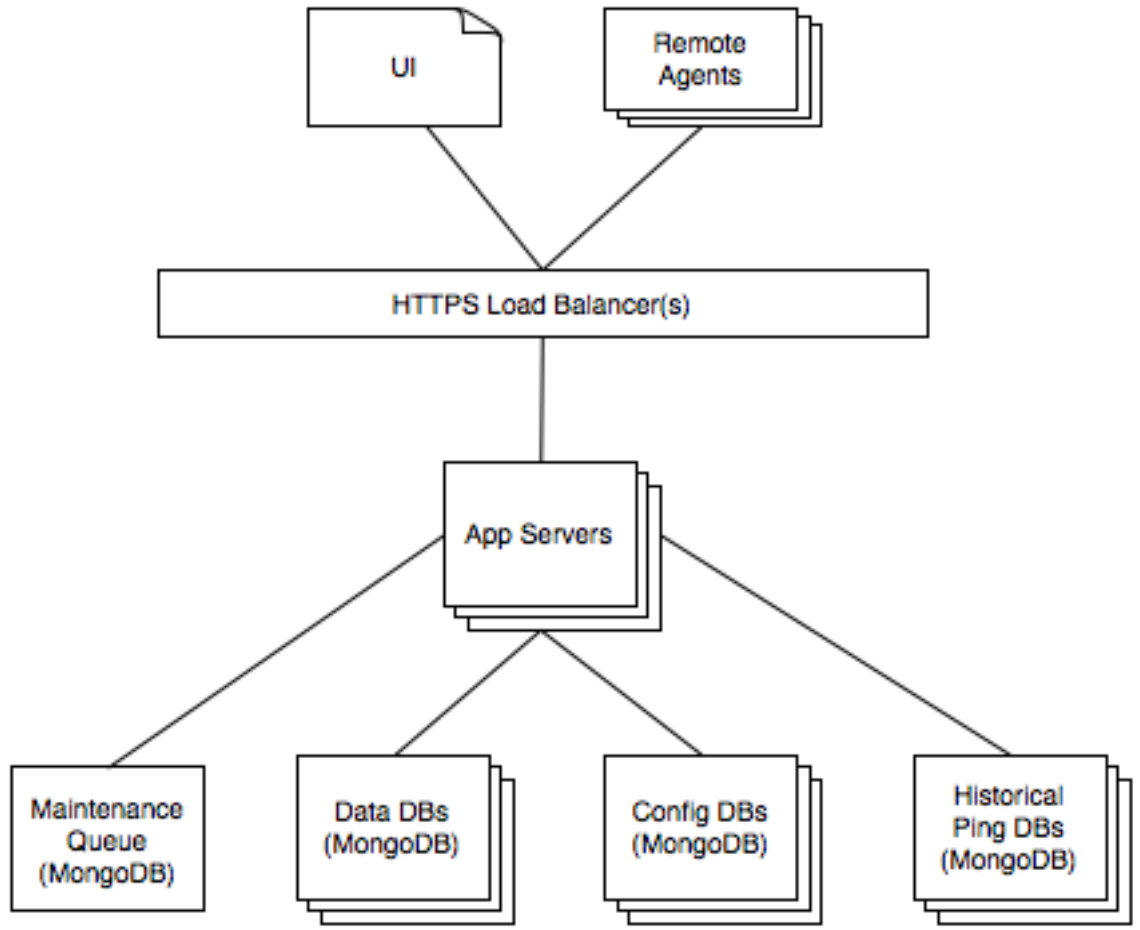
MongoDB Monitoring Service (MMS)

Overview

Currently in private beta for commercial support customers

Open to the everyone soon...

System Overview



Questions?

A simple example of a $set and $inc operation:

https://github.com/deftlabs/mongodb-meetup-june-21-2011

GitHub: https://github.com/rgnitz