Growing Your Inbox:

HBase at Tumblr

Monthly page views

18,000,000,000

600M page views per day

60-80M new posts per day

Peak rate of 50k requests & 2k posts per second

Totals: 22B posts, 53M blogs, 45M users

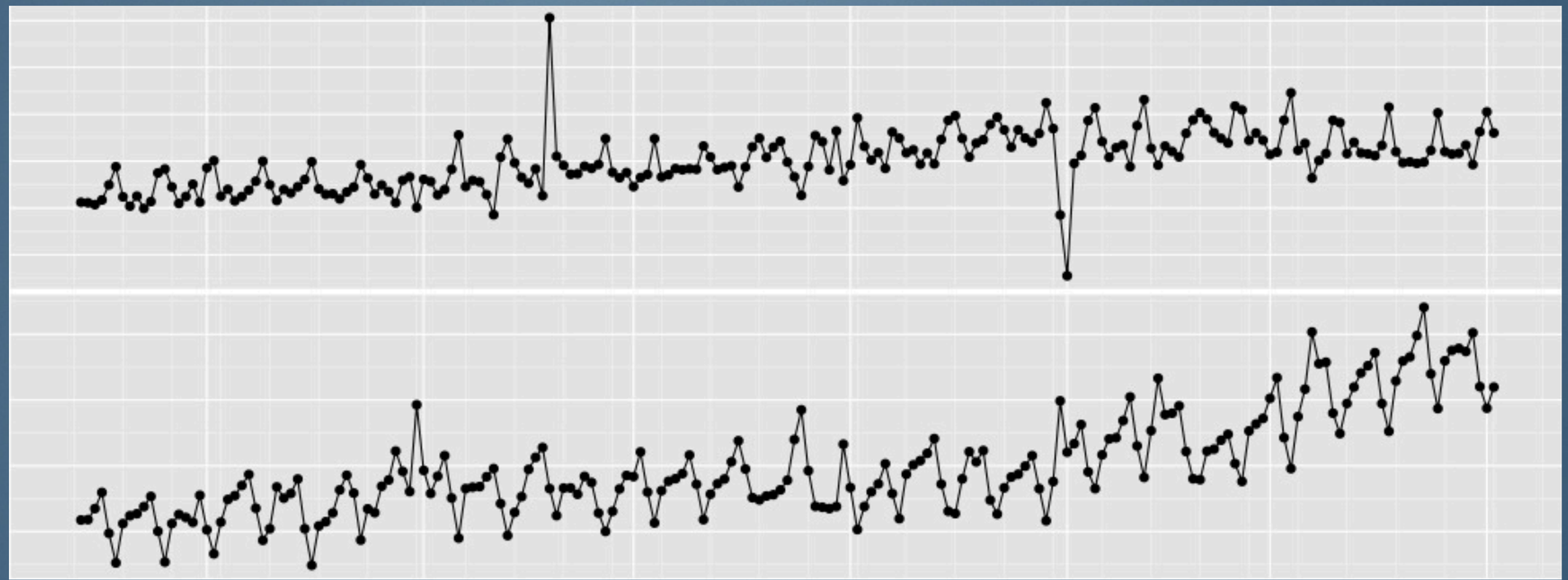
23 in Engineering (7 ops, 4 net/dc, 4 SRE, 8 dev)

70% of traffic is destined for the dashboard

Posts and followers drive growth

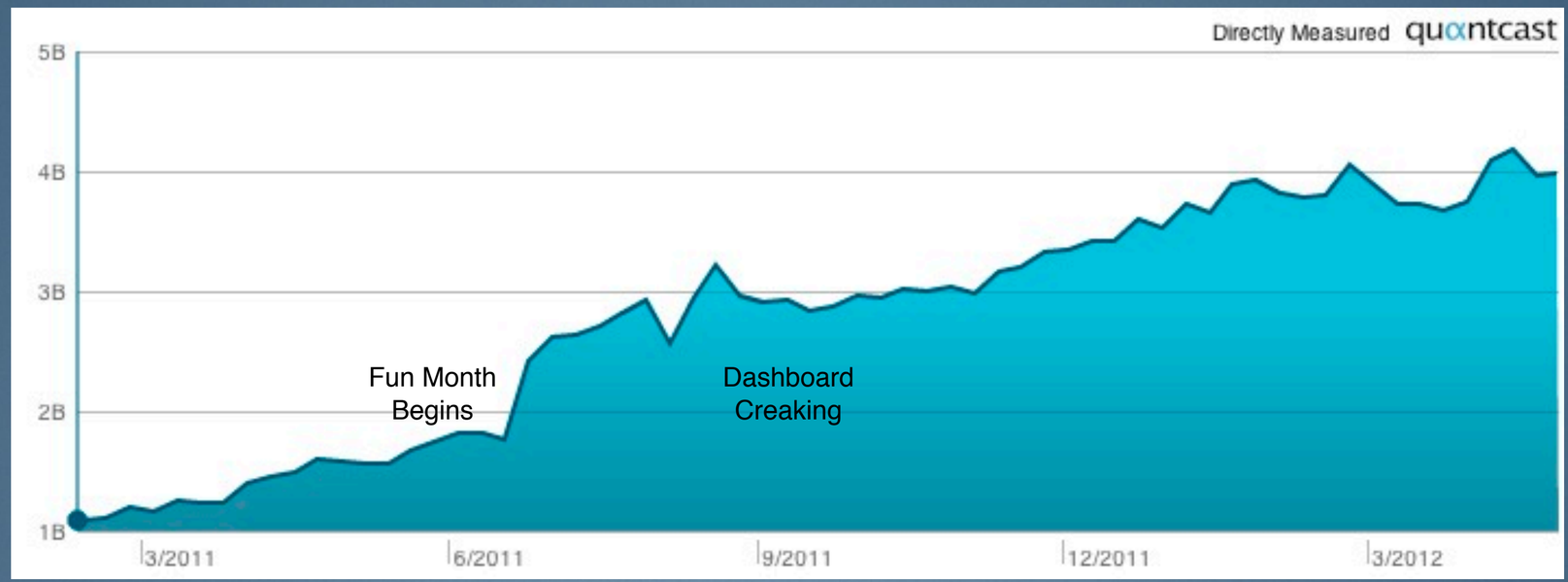
Posts

Average Followers



Traffic growth

Page Views



People

One small problem

The dashboard is assembled using a time ordered essentially unpartitioned MySQL database without caching. Sharding and caching aren't reasonable approaches. Write concurrency will soon introduce slave lag, rendering the dashboard unusable.

Motherboy Motivations and Goals

Motivations

- Awesome Arrested Development reference
- The dashboard is going to die
- Time ordered MySQL sharding == bad
- Inconsistent dashboard experience

Goals

- Durability and availability
- Multi-datacenter tenancy
- Features (read/unread, tags, etc)

Challenges

- Response time < 100ms
- Data volume
- Active legacy PHP code base
- Hardware provisioning, failures, etc.
- Small team, not much in house JVM/Hadoop experience

One small problem

Assumptions

- 60M posts per day
- 500 followers on average
- 40 bytes per row, 24 byte row key
- Compression factor of 0.4
- Replication factor of 3.5 (3 plus scratch space)

Data Set Growth

	Rows	Size
Day	30 Billion	447 GB
Week	210 Billion	3.1 TB
Month	840 Billion	12.2 TB
Year	10 Trillion	146 TB

Data Set Growth (Replicated)

	Size
Day	1.5 TB
Week	10.7 TB
Month	42.8 TB
Year	513 TB

Motherboy Architecture

Goals

- Dashboard posts are persistent for each follower (their inbox)
- Writes are asynchronous and distributed
- Users home to an inbox
- Inboxes are eventually consistent
- Selective materialization of feeds

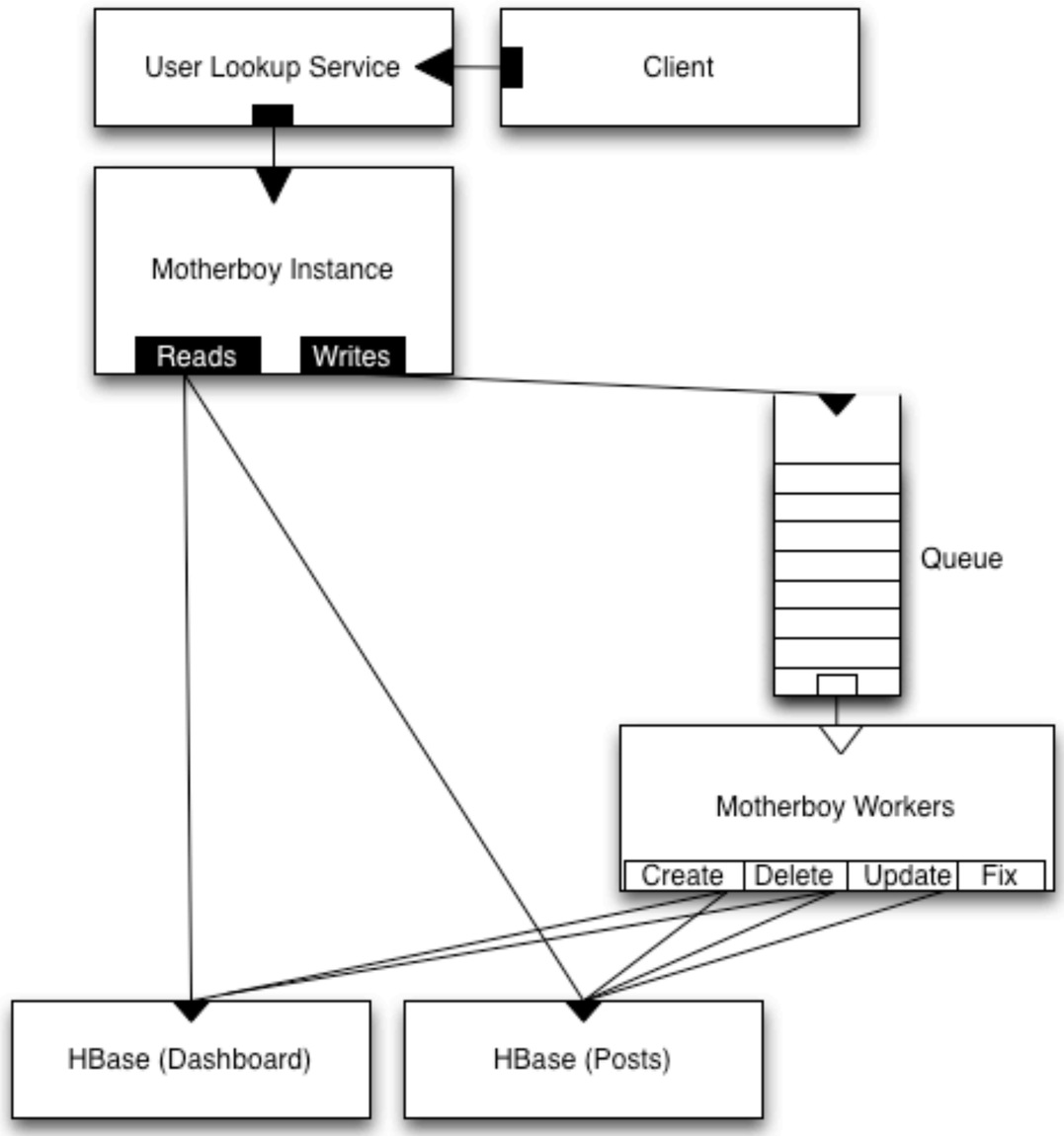
Failure Handling

- Reads can be fulfilled from any store
- Writes can catch up when a store is back online

Motherboy V0

Implementation

- **Goal:** Get our feet wet!
- Single 6 node HBase cluster
- Writes to Motherboy via Finagle/Thrift RPC service
- User homed to Motherboy cluster


User Lookup Service
Client
Motherboy Instance
Reads
Writes
Queue
Motherboy Workers
Create
Delete
Update
Fix
HBase (Dashboard)
HBase (Posts)

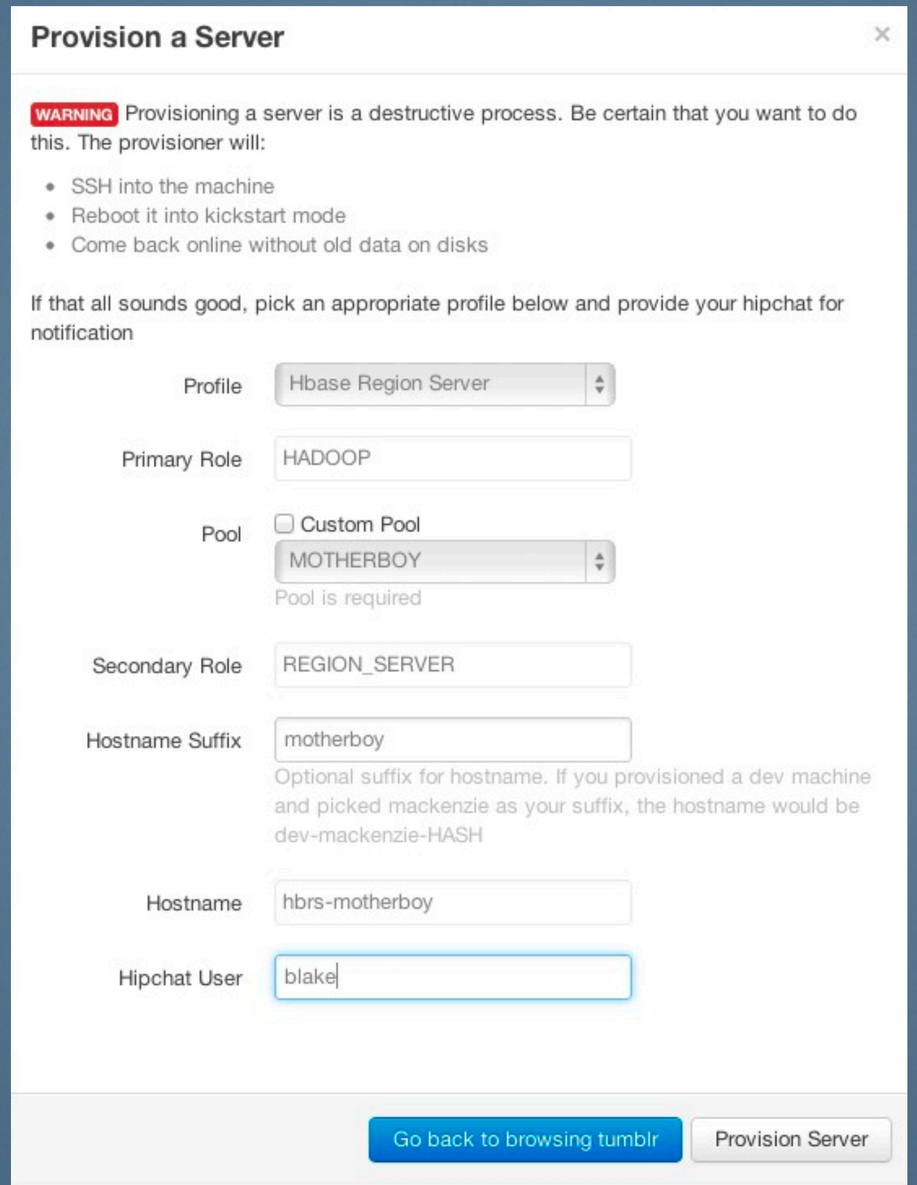
Motherboy V0 learnings

- **Conclusion:** We needed more experience
- We had no clue what we were doing
- Poor automation ➜ Hard to recover in event of failure
- No test infrastructure ➜ Hard to evaluate effects of cluster performance tuning
- Very little insight into environment

Sidebar: OpenTSDB

- **Goal:** Get some hands on experience running a production cluster
- Build out fully automated cluster provisioning
- Build out a cluster monitoring infrastructure
- Build out cluster trending/visualization

Automated provisioning



Automated monitoring

Current Network Status
Last Updated: Sun Apr 29 02:30:24 EDT 2012
Updated every 90 seconds
Nagios® Core™ 3.3.1 - www.nagios.org
Logged in as *tumblr*

View Status Overview For All Host Groups
View Service Status Detail For This Host Group
View Host Status Detail For This Host Group
View Status Summary For This Host Group
View Status Grid For This Host Group

Host Status Totals

Up	Down	Unreachable	Pending
15	0	0	0

All Problems	All Types
0	15

Service Status Totals

Ok	Warning	Unknown	Critical	Pending
156	0	5	6	1

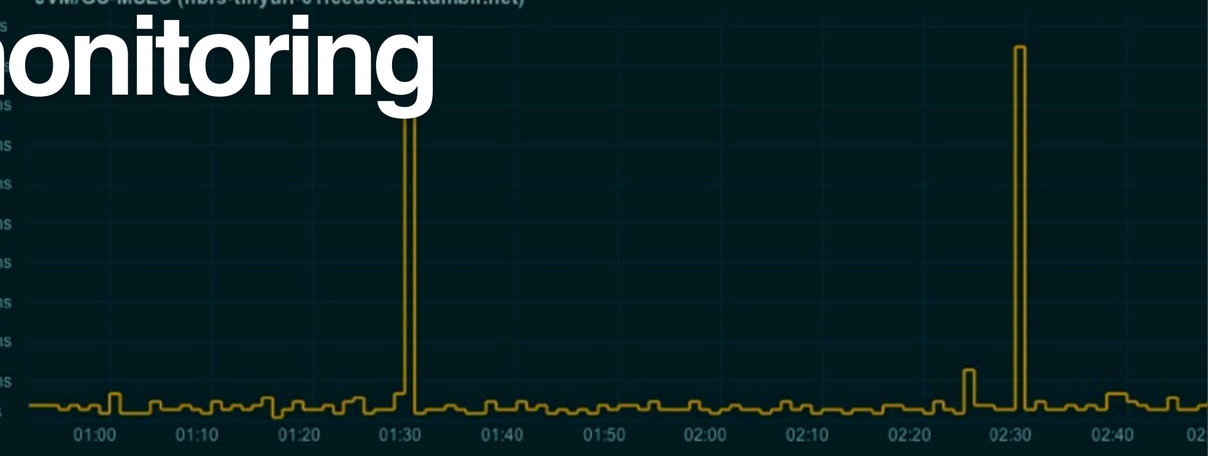
All Problems	All Types
11	168

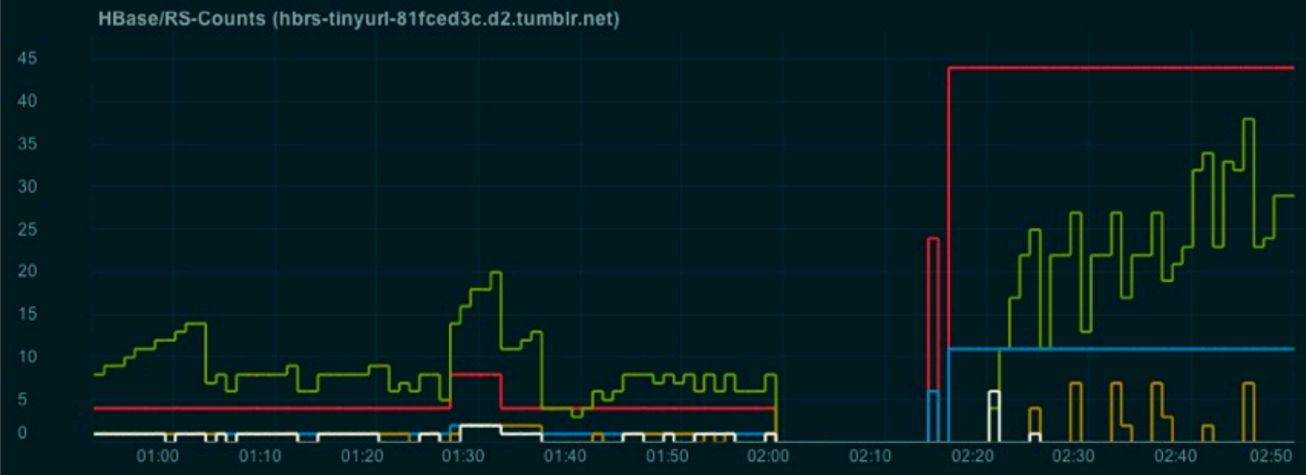
Service Overview For Host Group 'POOL:URL_SHORTENER'

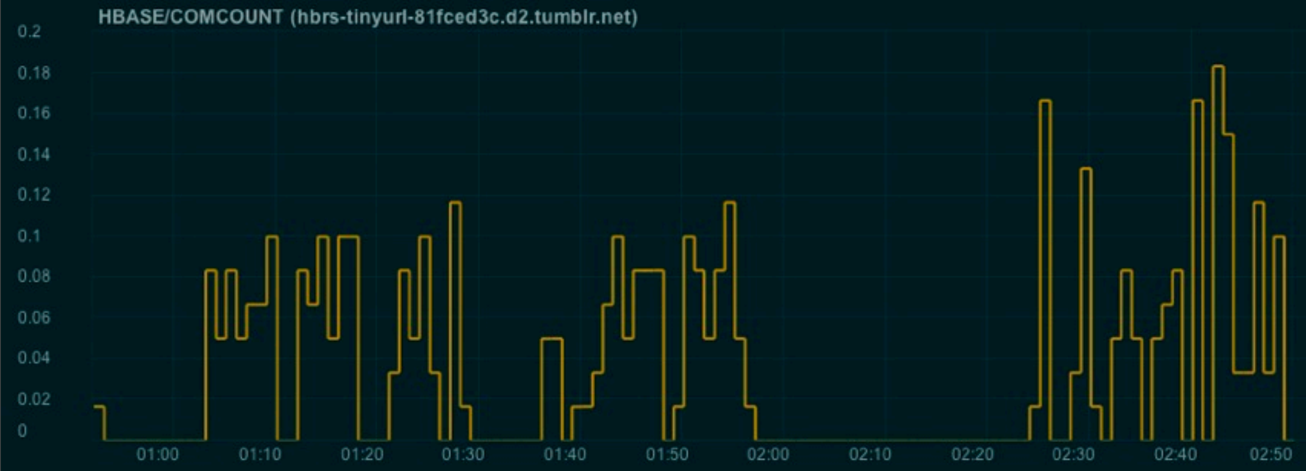
POOL:URL_SHORTENER (POOL:URL_SHORTENER)

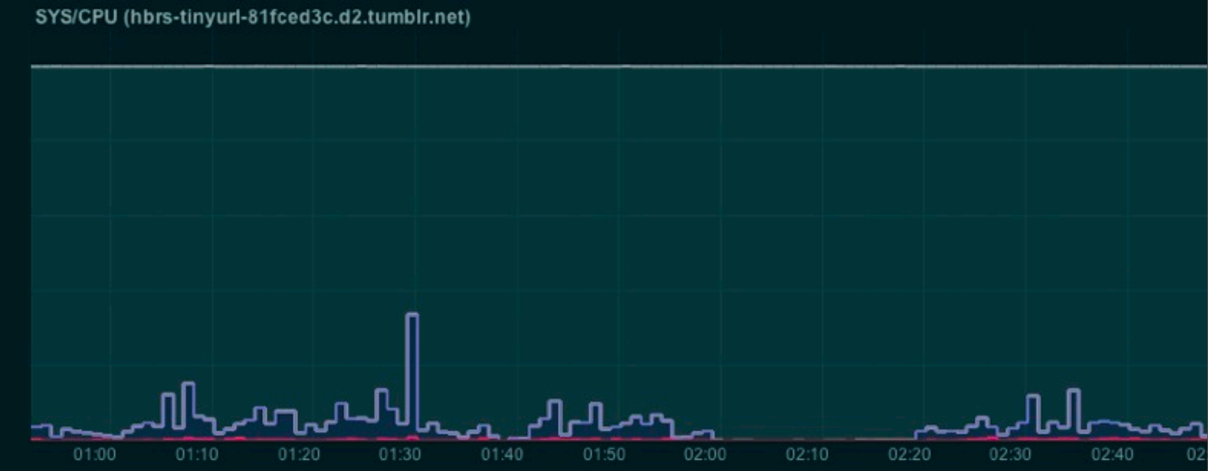
Host	Status	Services	Actions
hbm-tinyurl-1fa64c4c.d2.tumblr.net	UP	10 OK	
hbrs-tinyurl-32d119dc.d2.tumblr.net	UP	10 OK	
hbrs-tinyurl-3e0a1105.d2.tumblr.net	UP	10 OK	
hbrs-tinyurl-51a00c8d.d2.tumblr.net	UP	10 OK	
hbrs-tinyurl-7c82f19e.d2.tumblr.net	UP	10 OK	
hbrs-tinyurl-81fced3c.d2.tumblr.net	UP	10 OK	
hbrs-tinyurl-fd3bf333.d2.tumblr.net	UP	10 OK	
nn-tinyurl-b29d0515.d2.tumblr.net	UP	10 OK 1 CRITICAL	

Automated monitoring









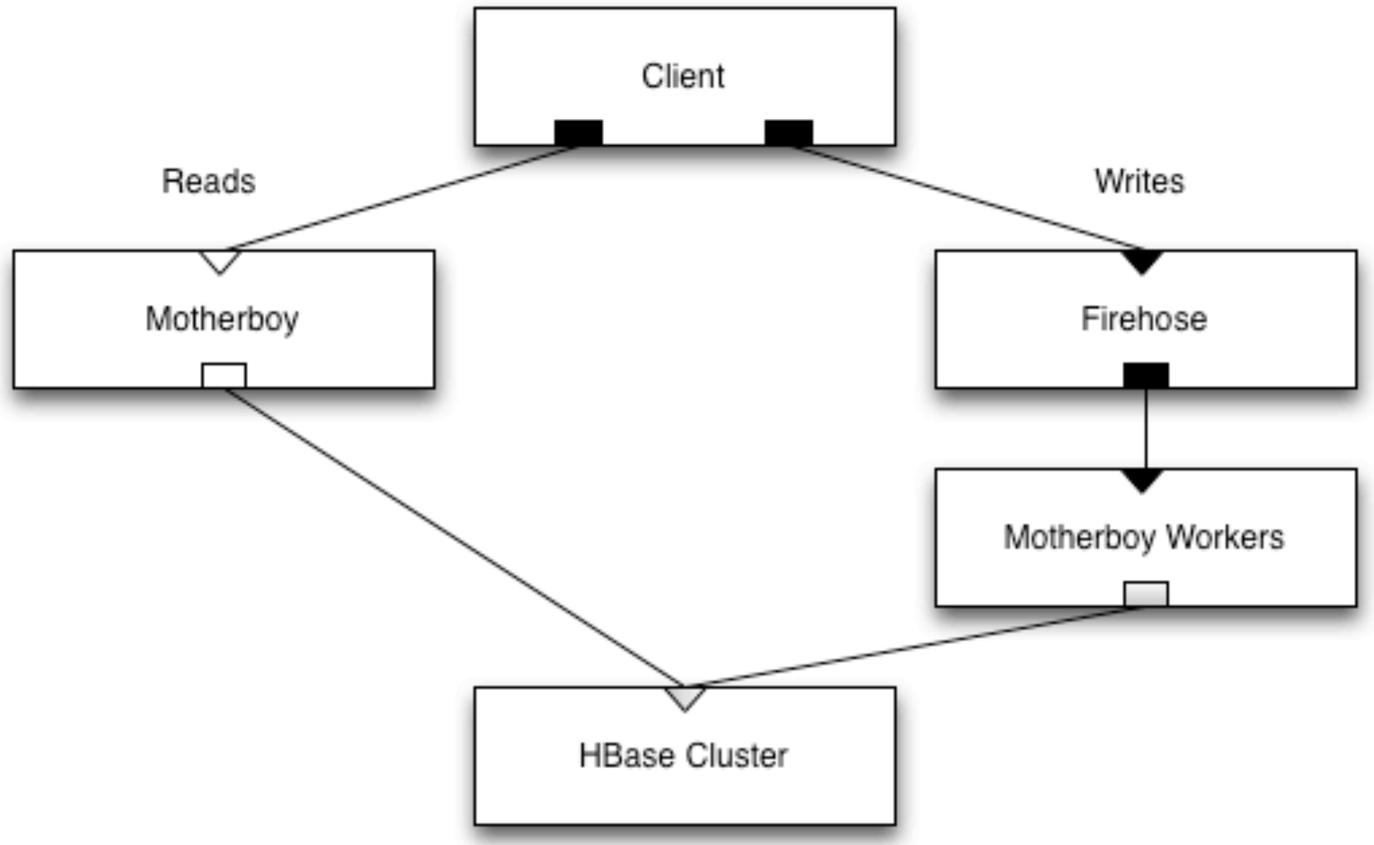
OpenTSDB: Cluster 0

- 10k data points per second, 500k at peak
- 1200 servers reporting via collectd
- Application stats via collectd
- 864M data points per day
- 10 node cluster
- **Making progress**

Motherboy V1: Try again

Implementation

- **Goal:** Into testing!
- Single 10 node HBase cluster
- No writes directly to Motherboy, consumed via Firehose
- Enable reads via feature flag for engineers
- Ignore discovery mechanism for now


Client
Reads
Writes
Motherboy
Firehose
Motherboy Workers
HBase Cluster

Motherboy V1 learnings

The black-art of tuning

- Disable major compactions
- Disable auto-splits, self manage
- `regionserver.handler.count`
- `hregion.max.filesize`
- `hregion.memstore.mslab.enabled`
- `block.cache.size`
- Table design is super important, a few bytes matter

Challenges

- Region sizing (# regions & size) is crucial and workload dependent
- YCSB was valuable for load testing but required extensive modification
- Too hard to test changes, versions, etc.
- Failure recovery is manual and time consuming

Sidebar: Cluster testing

- **Goal:** Make it easy!
- Fixture data
- Load test tools
- Historical data, look for regressions
- Create a baseline!
- Test different versions, patches, schema, compression, split methods, configurations

Testing setup

Overview

- Standard test cluster: 6 RS/DN, 1HM, 1NN, 3ZK
- Drive load via separate 23 node Hadoop cluster
- Test dataset created for each application
- Results recorded and annotated in OpenTSDB

Motherboy Testing

- 60M posts, 24GB LZO compressed
- 51 mappers
- Map posts into tuple of Long's - 24 byte row key and 16 bytes of data in single CF
- Write to HBase cluster as fast as possible

Sample test results

Scenario	Test Time	Posts per Second	Avg. Request Time	Avg. CPU Load
Baseline	2:49:46	5801.4	3.7ms	15%
Handler Count = 100	2:39:57	6157.5	3.4ms	12%
Disabled Auto Flush	1:58:53	8284.5	3.4ms	7%
Pre-Created Regions	30:08	32684.3	2.2ms	3%

Conclusions

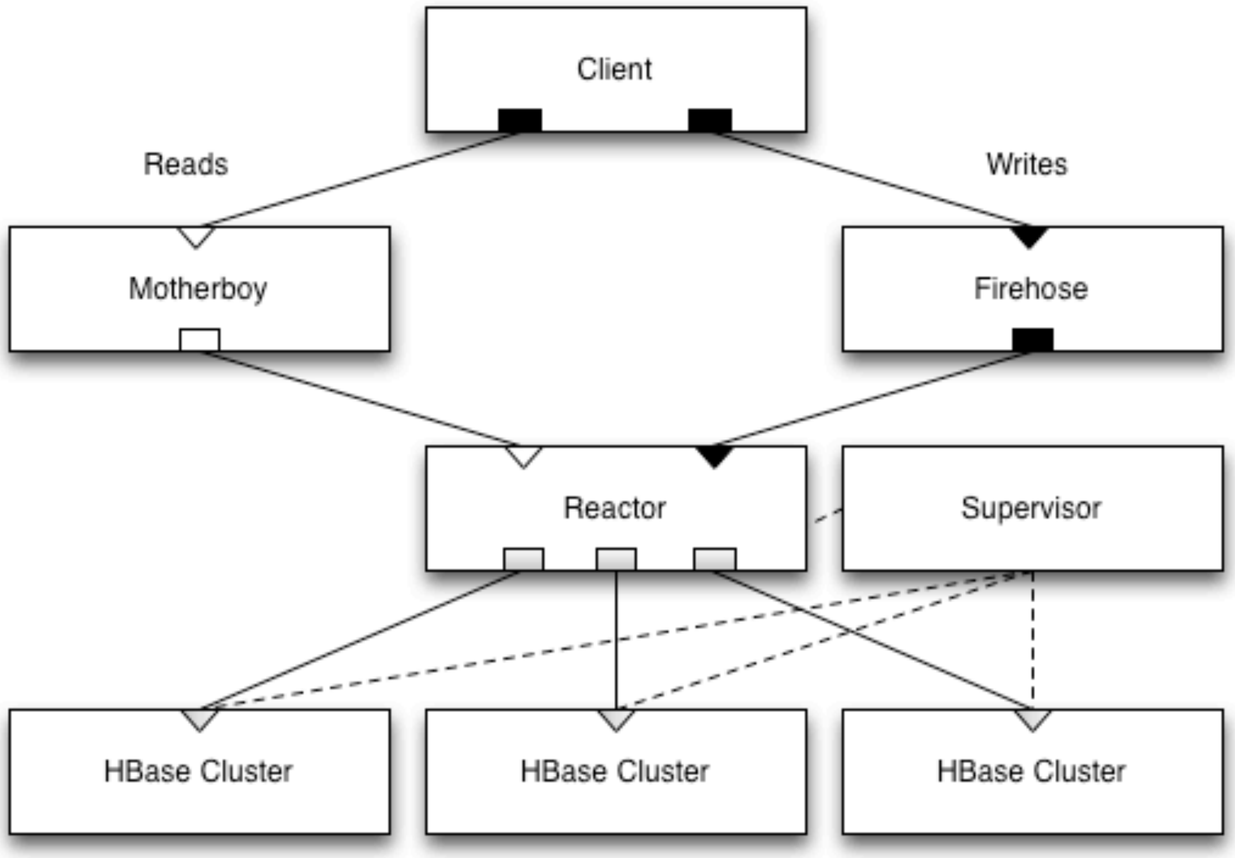
- More region servers, better throughput
- Less round-trips, better throughput
- Not earth shattering
- If experimenting is easy, people will do it

Tests must be designed for intended workload

Motherboy V2: In progress

Implementation

- **Goal:** Into production!
- Multiple 10 node HBase clusters
- Writes/reads mediated by reactor layer
- Supervisor managed migrations and region splits


Client
Reads
Writes
Motherboy
Firehose
Reactor
Supervisor
HBase Cluster
HBase Cluster
HBase Cluster

Lessons learned

1. The value of automation can't be understated
2. Rolling upgrades and deploys are easy, but refer to #1
3. Technology choice does play a role in how easily you can scale
4. At scale, nothing works as advertised
5. Operational tooling is the single most important thing you can do up front

Questions?

Blake Matheny: bmatheny@tumblr.com