Cassandra & Puppet:
Scaling data at $15/month

Constant Contact

March 2011

Dave Connors – VP Operations
Jim Ancona – Systems Architect
Mark Schena – Manager Systems Automation

Constant Contact

2000 – 2010

Market leader for Small Businesses

- Email, Event & Survey
- Over 400k paying customers
- No. 134 on the Deloitte Technology Fast 500 listing

Business model

- Many customers pay as little as $15 a month
- ~2 million database transactions per minute

The business problem

Small Businesses are looking to us for help with Social Media marketing

- Social Media → 10-100 times more data
- Challenge with our business model

The Key Challenge

Integrate social media data

- Solution = NoSQL
- Cost = Low
- Time to market = ?

Implementing NoSQL

Ops and Dev both face issues

- Data model
- Monitoring
- Authentication
- Logging
- Risk profile
- Roles & Responsibilities

DevOps

Apache Cassandra

- Developed at Facebook
- Open sourced in 2008
- Incubated at Apache
- Became an Apache top-level project in 2010
  - http://cassandra.apache.org
- In use at Digg, Facebook, Twitter, Reddit, Rackspace, Cloudkick, Cisco, …
- Largest production cluster has over 100 TB of data in over 150 machines

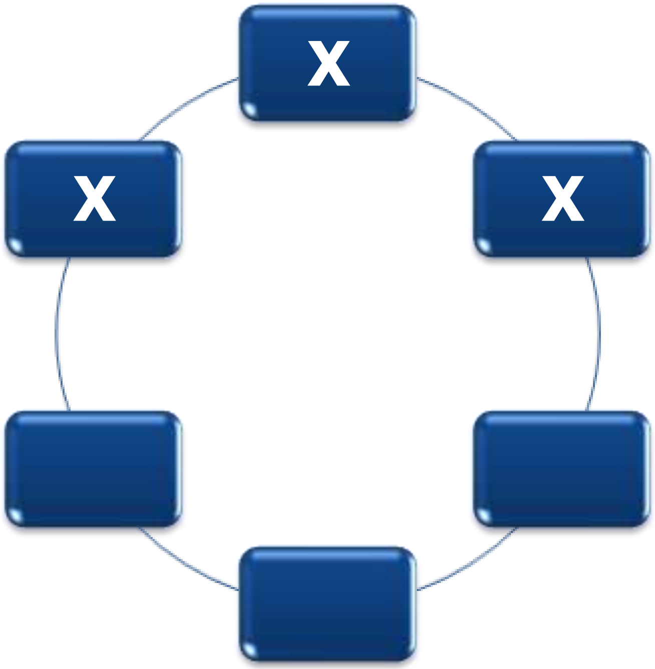
What is Cassandra

- Implemented in Java
- Fault Tolerant
- Elastic
- Durable

- Rich data model
- Replicated data
- Consistency options

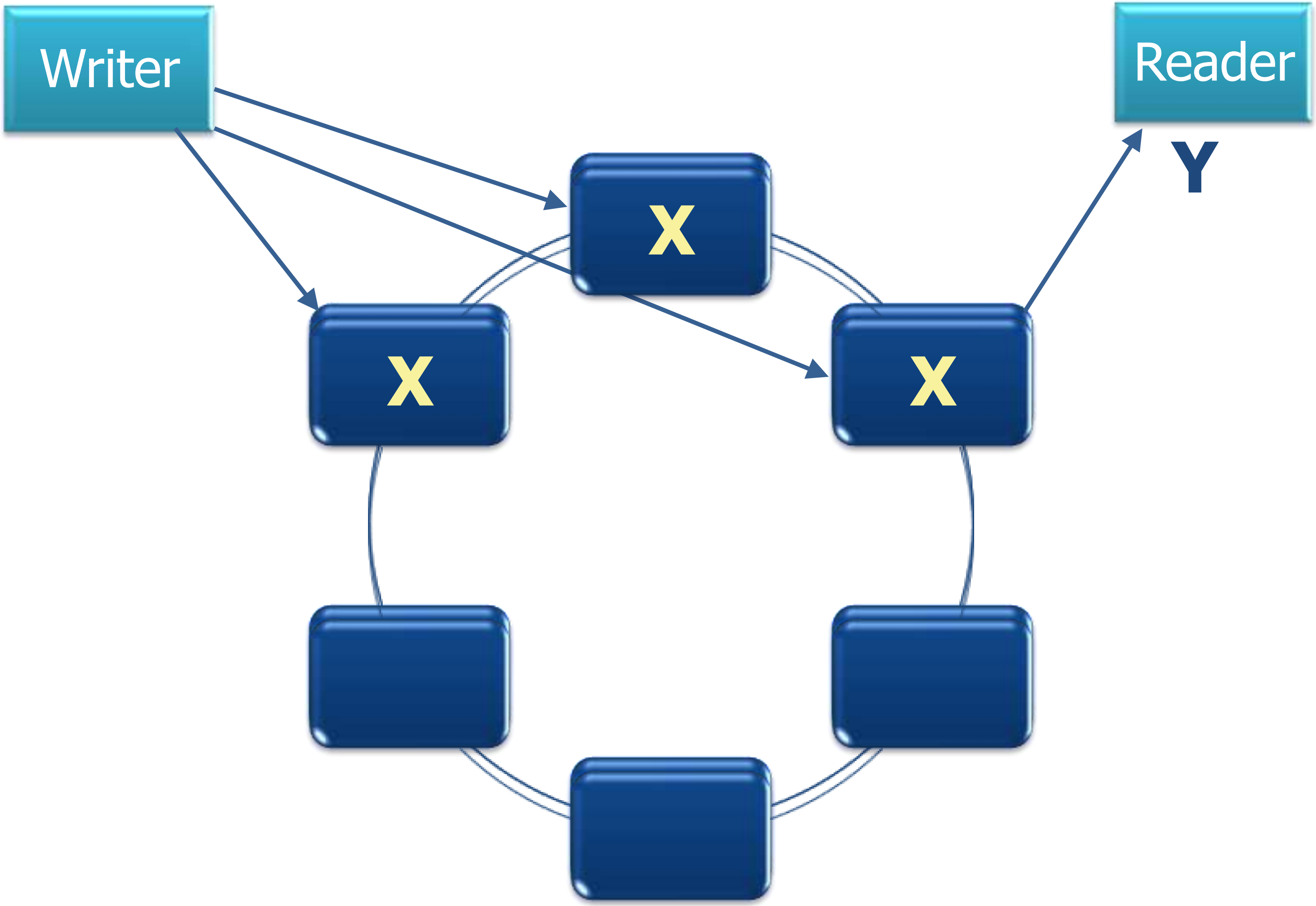
Replication

How many copies of each piece of data do we want?

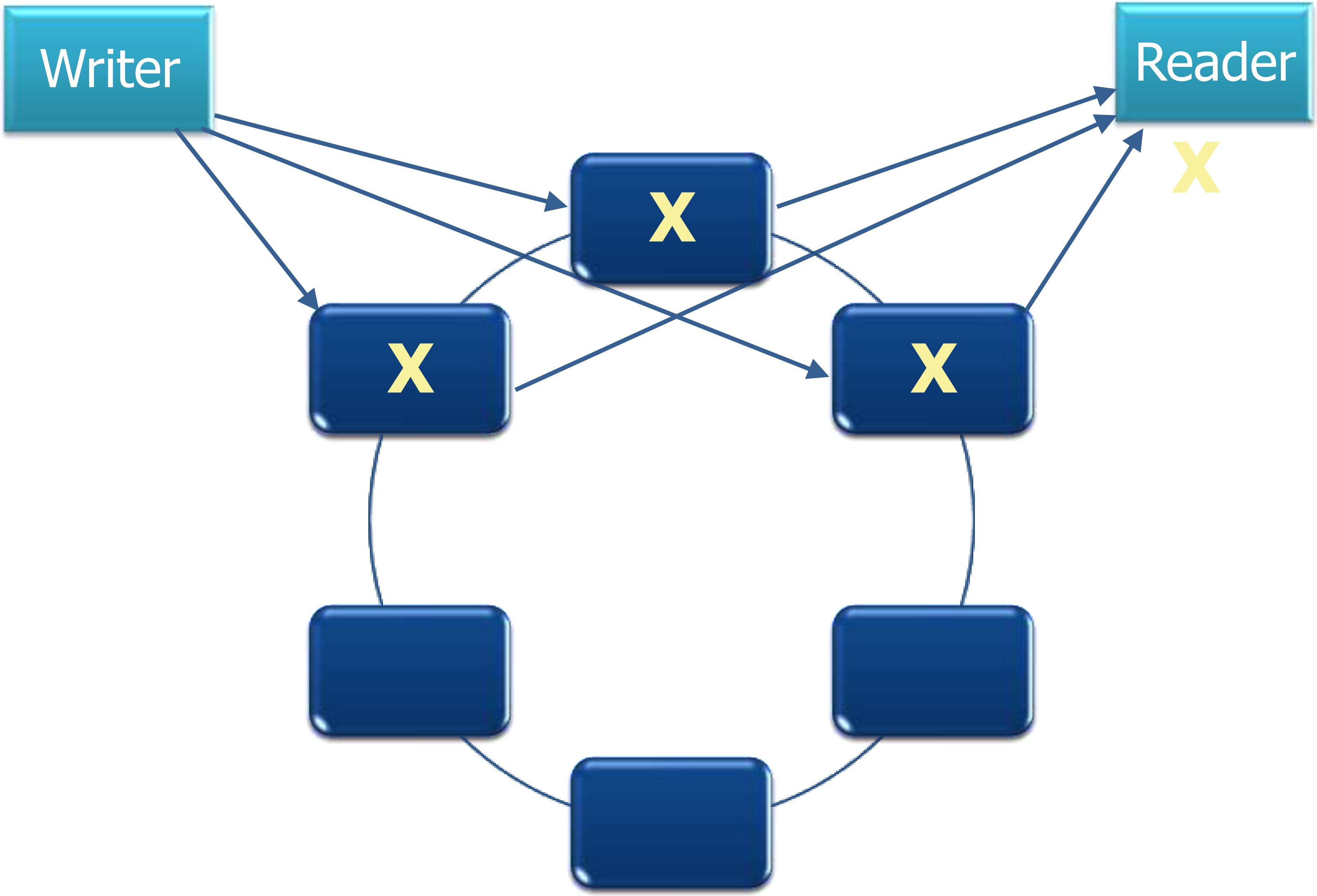


N=3

Consistency Level One



Consistency Level Quorum



Risks and Mitigation

- Moving target
- Developer unfamiliarity
- Operational procedures
- Reliability concerns

- Deployment automation
- Community involvement
- Training/Consulting
- Application selection
- Lots of monitoring
- Phased rollout

Development Challenges

Understanding the data model

Choosing a client

- Clients available for Java, Python, .NET, Ruby, PHP
- Don’t use Thrift

Moving target

Open Source

- Not “one neck to wring”
- Paid support and training is available:
  http://datastax.com
- Community
  - Mailing lists
  - IRC #cassandra at freenode
- Contribute

Phased Rollout

- Switchable modes
- Mirroring
- Dial-able traffic

Collaboration

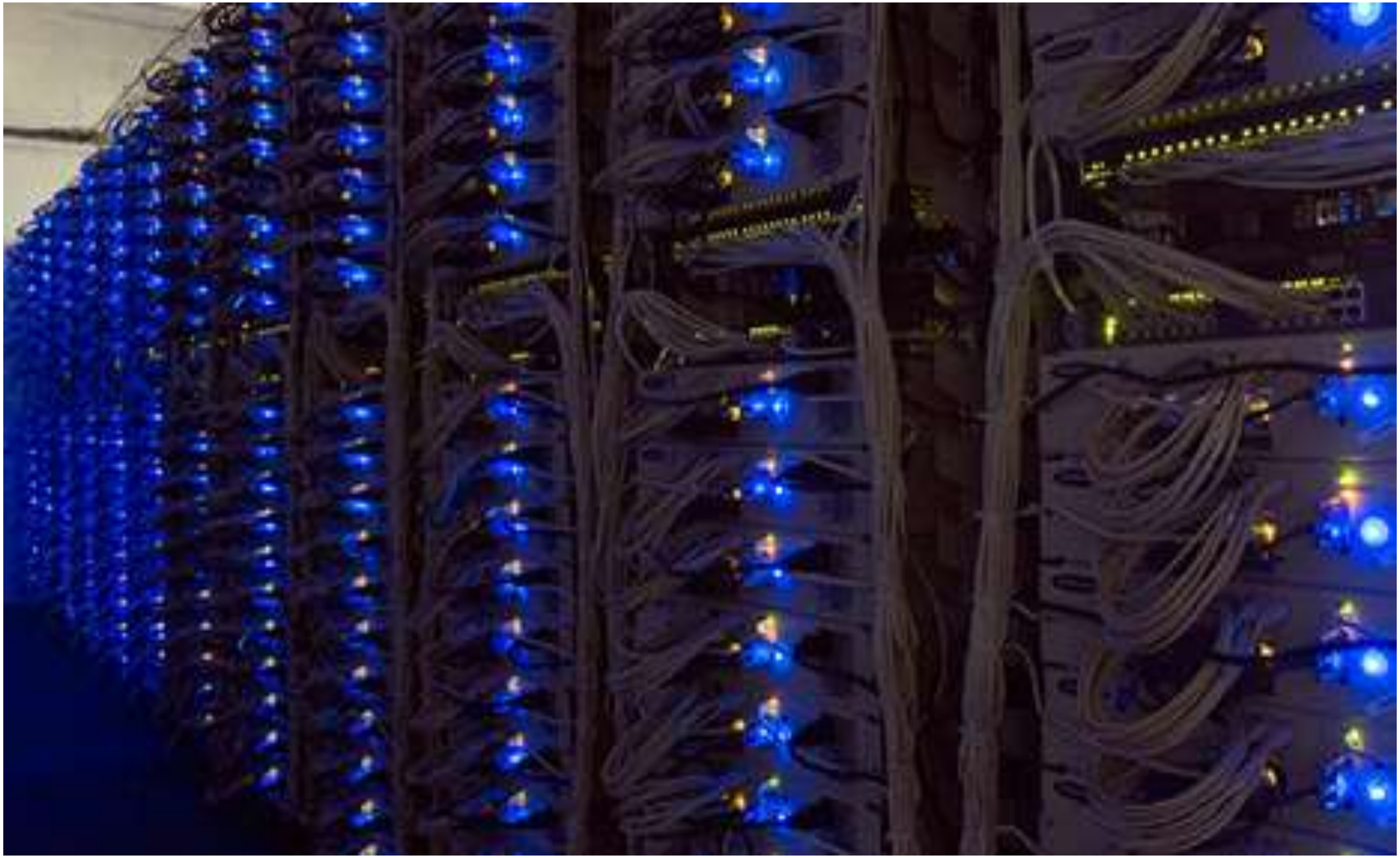
- Big, complex project
- Close collaboration
- Flexible roles
- Ability to iterate

DevOps

“Are you sure you really want that?”

- 3 500G disks
- 1 250G disk
- No SWAP
- RAID Zero Root Partition and Data Storage
- 32G Memory

We will need how many servers?

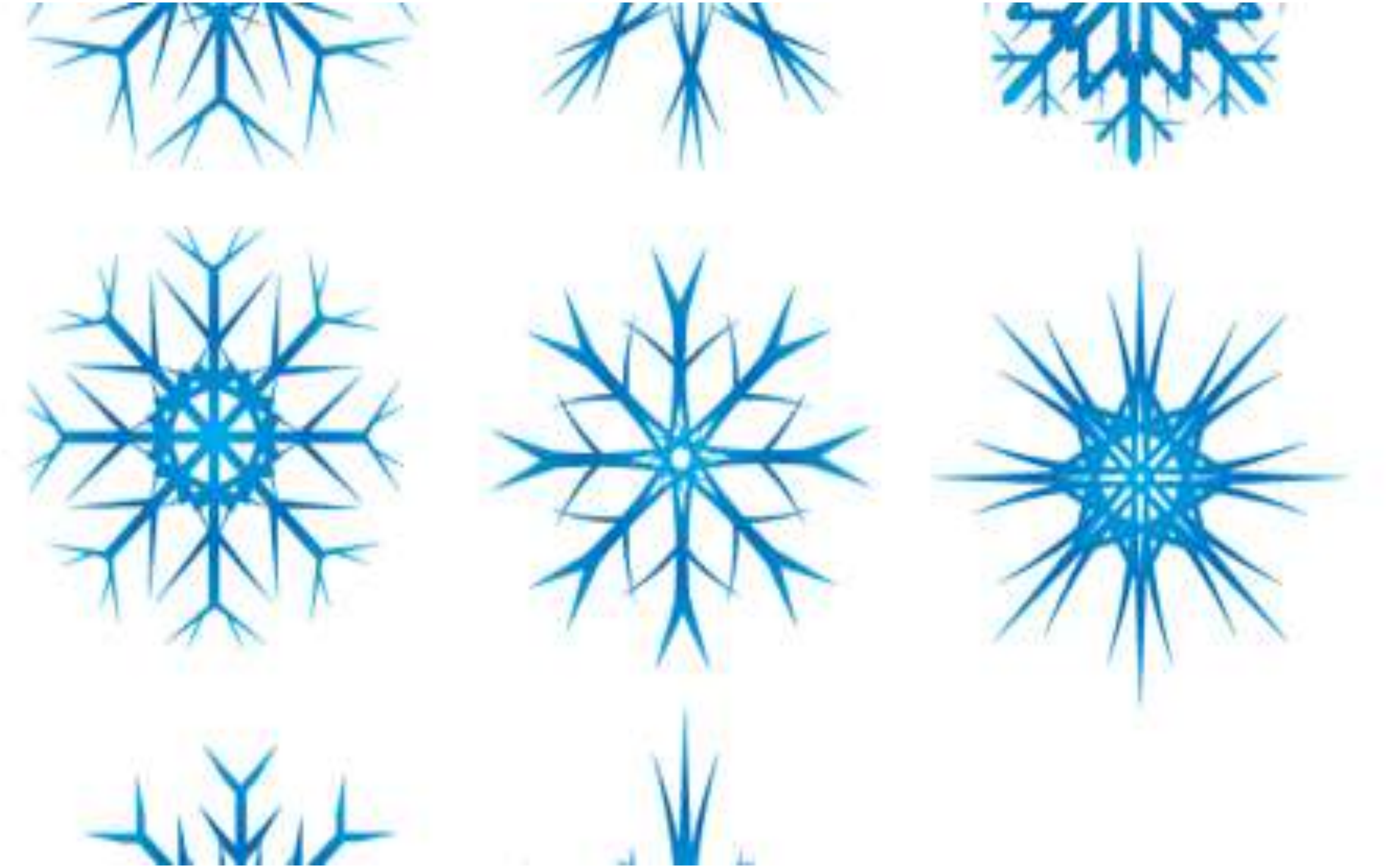


How many nodes?

- Quorum = 3
- Multiple Datacenters = 2
- Use only half the available disk = 2
- 12 Servers = ~1 TB Of Data Storage
- ~6 TB of Data Storage

3 x 2 = 6 x 2 = 12 x 6 = 72

Random Partitioner



Tool Chain

fedora

Anaconda/Kickstart

Func

puppet labs

Nagios

MUNIN

DevOps with Puppet

- Puppet is the shared framework between Operations and Development
- Versioning of puppet code allows for adoption of development best practices
- Leverage Domain specific knowledge and skill

Always Move Forward

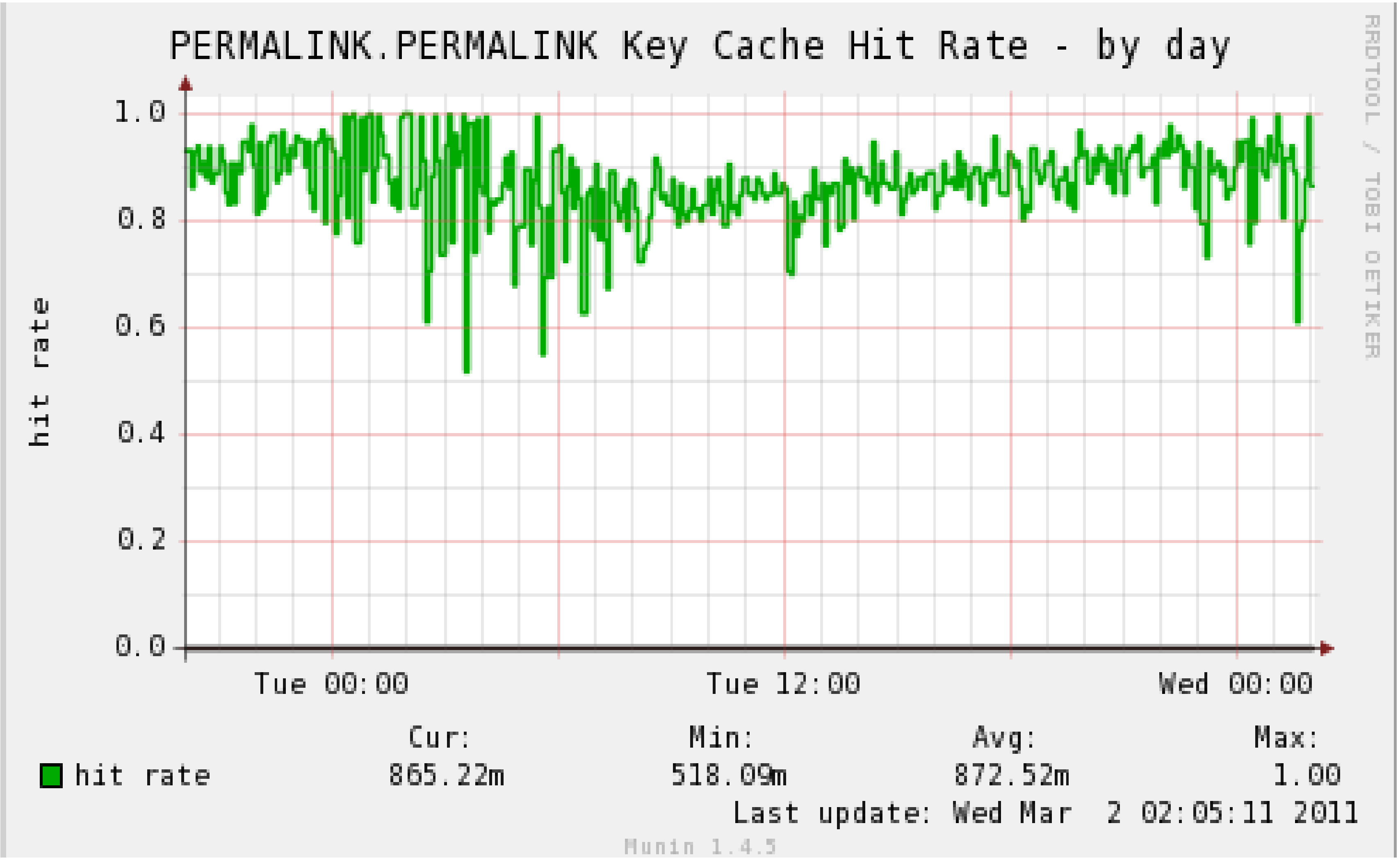
Operational Efficiencies

- Remote logging is a requirement
- Cassandra uses log4j natively
- Resources not available for remote log4j development
- Scribed with Puppet provides the solution

Development takes the Operational Lead

- Munin
- JMX trending
- Identify critical data points
- Rapid development of graphs
- Puppet Definitions are used for rapid deployment

Sample Munin Graph



Example: Munin Puppet Code

```
define munin::cassandracolumnfamily ( ) {
 include cassandravirtual
 File <| title == "jmxbin" |>

 $confdir="/opt/cassandra-munin-plugins”
$plugindir="/etc/munin/plugins"
 $target="/opt/cassandra-munin-plugins/jmx_"

 # Match 3 strings separated by periods
 $pattern = '^([^.]*)[.]([^.]*)[.]([^.]*)$'

 $keyspace = regsubst($name, $pattern, '\1')
 $columnfamily = regsubst($name, $pattern, '\2')
 $file = regsubst($name, $pattern, '\3')

 file {"${keyspace}_${columnfamily}_${file}.conf":
   owner => 'root', ensure => 'file', group => 'root', type => 'file',
   path => "${confdir}/${keyspace}_${columnfamily}_${file}.conf",
   mode => '644',
   content => template("munin/attribute_${file}.conf.erb"),
   require => [ Package['munin-node'], File['/opt/cassandra-munin-plugins'], File['jmxquery'], ],
 }

 file {"$plugindir/${keyspace}_${columnfamily}_${file}":
   ensure => 'link', owner => 'root', group => 'root', mode => '511', type => 'link',
   target => "$target",
   require => [ File['/opt/cassandra-munin-
plugins'], File["${keyspace}_${columnfamily}_${file}.conf"], File['jmxquery'], Package['munin-node'], ],
```

Conclusion

- Cassandra as an appliance
- Development Best Practices with Life Cycle Management
- Traditional vs. Today
  - Infrastructure

    4 weeks → 4 hours to build 72 nodes

  - Development to Deployment

    9 months → 3 months

  - Cost

    Millions → 150k

Constant Contact®
Connect. Inform. Grow.

Thank You!