



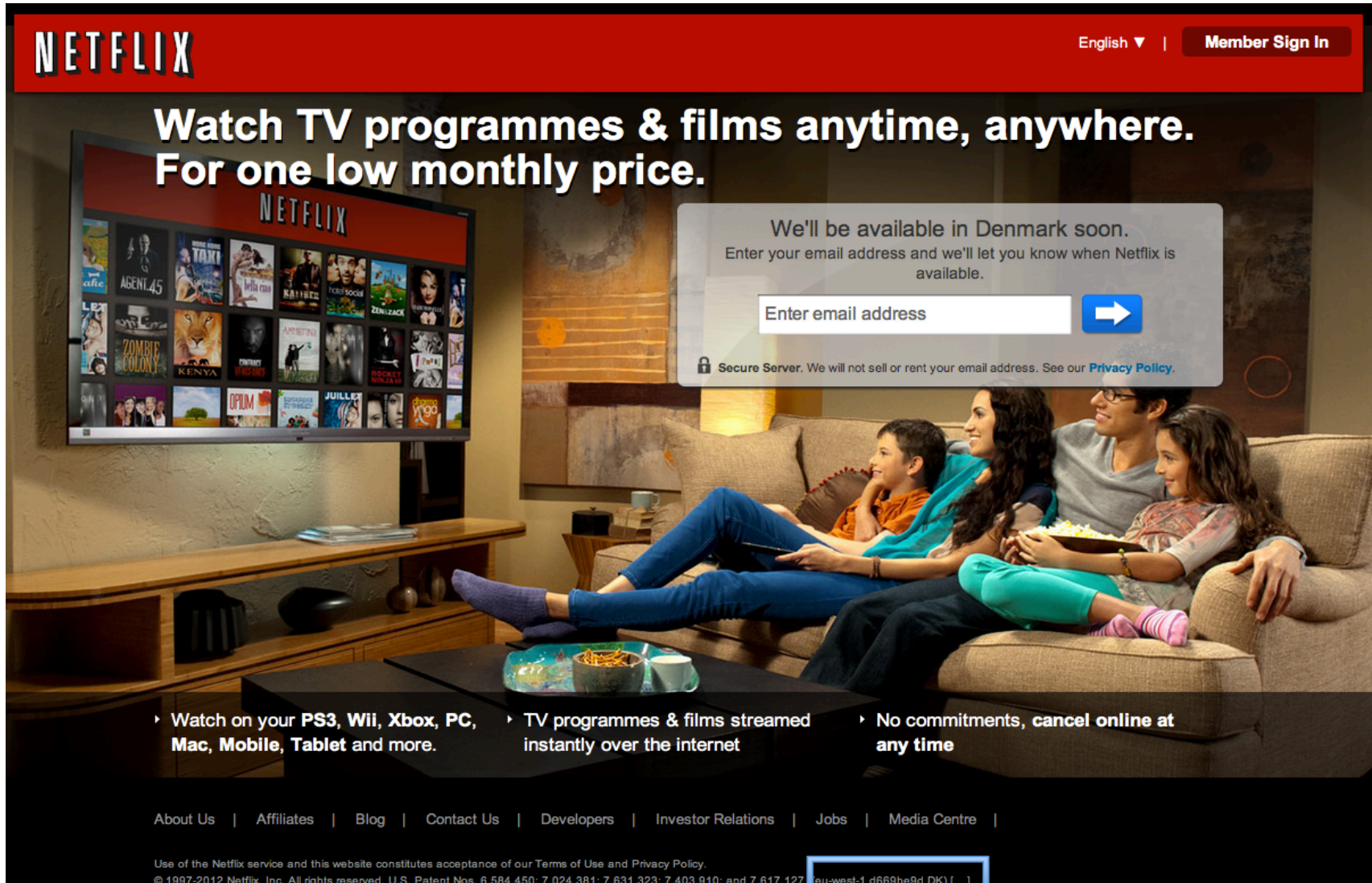
Billedet kan ikke vises. Computeren har muligvis ikke hukommelse nok til at åbne billedet, eller billedet er muligvis blevet beskadiget. Genstart computeren, og åbn derefter filen igen. Hvis det røde x stadig vises, skal du muligvis slette billedet og indsætte det igen.

Running Netflix on Cassandra in the Cloud



Billedet kan ikke vises.
Computeren har muligvis ikke
hukommelse nok til at åbne
billedet, eller billedet er
muligvis blevet beskadiget.
Genstart computeren, og åbn
derefter filen igen. Hvis det

www.netflix.com in DK

A promotional banner for Netflix in Denmark. The background shows a family of four (a man, a woman, and two children) sitting on a couch in a living room, watching a TV. The TV screen displays the Netflix interface with a grid of movie and TV show thumbnails. The banner includes the Netflix logo, a sign-in link, a headline about watching TV programmes and films anytime, anywhere for a low monthly price, a sign-up form for email notifications, and a list of supported devices and streaming capabilities. The footer contains navigation links and legal information.

NETFLIX

English ▾ | **Member Sign In**

**Watch TV programmes & films anytime, anywhere.
For one low monthly price.**

We'll be available in Denmark soon.
Enter your email address and we'll let you know when Netflix is available.

Enter email address

 **Secure Server.** We will not sell or rent your email address. See our [Privacy Policy](#).

▸ Watch on your **PS3, Wii, Xbox, PC, Mac, Mobile, Tablet** and more. ▸ TV programmes & films streamed instantly over the internet ▸ No commitments, **cancel online at any time**

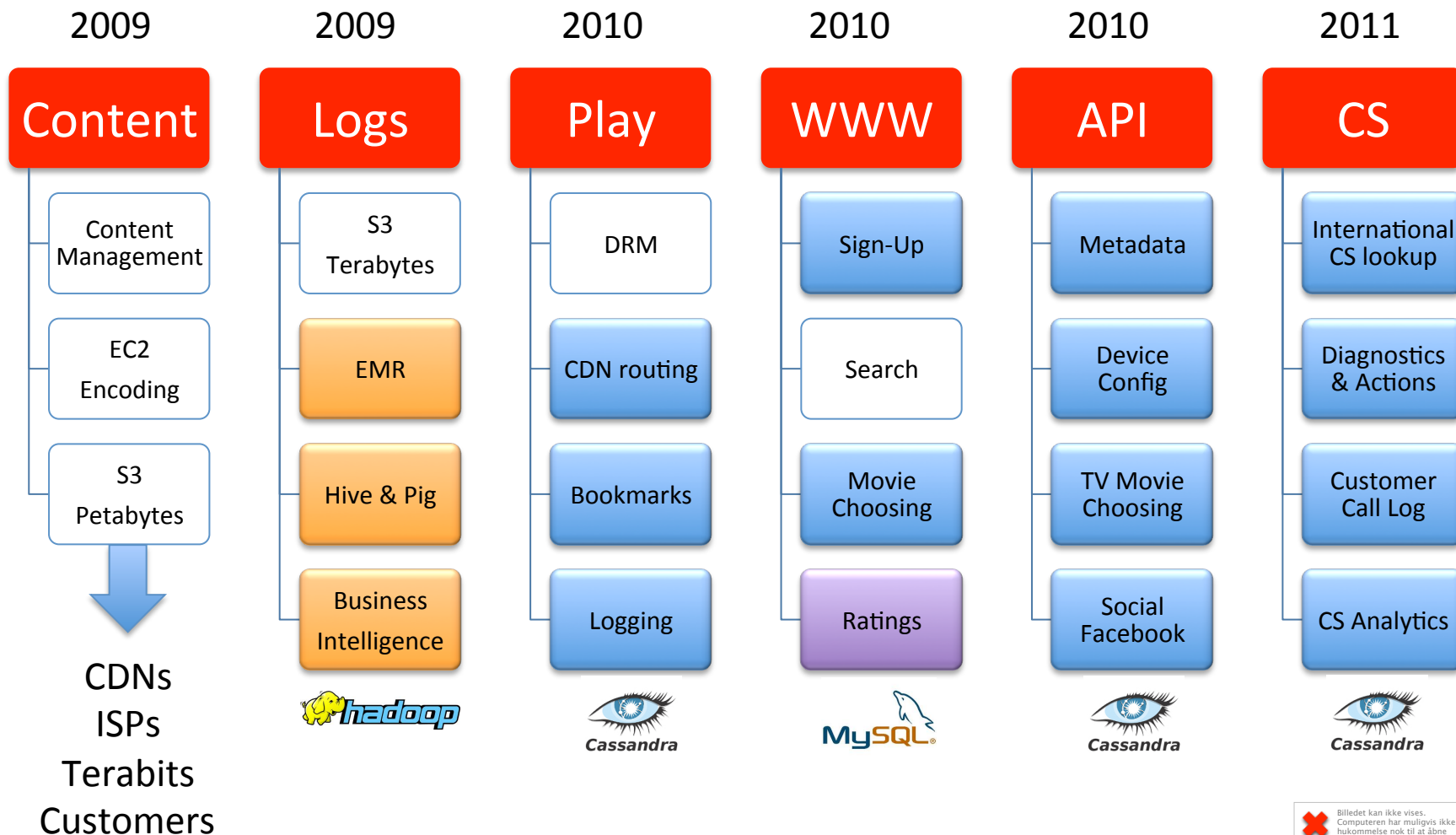
About Us | Affiliates | Blog | Contact Us | Developers | Investor Relations | Jobs | Media Centre |

Use of the Netflix service and this website constitutes acceptance of our [Terms of Use](#) and [Privacy Policy](#).
© 1997-2012 Netflix, Inc. All rights reserved. U.S. Patent Nos. 6,584,450; 7,024,381; 7,631,323; 7,403,910; and 7,617,127 (eu-west-1 d669be9d DK) |

Blah Blah  Blah

(I'm skipping all the cloud intro etc. did that yesterday... Netflix runs in the cloud, if you hadn't figured that out already you aren't paying attention and should go read slideshare.net/netflix)

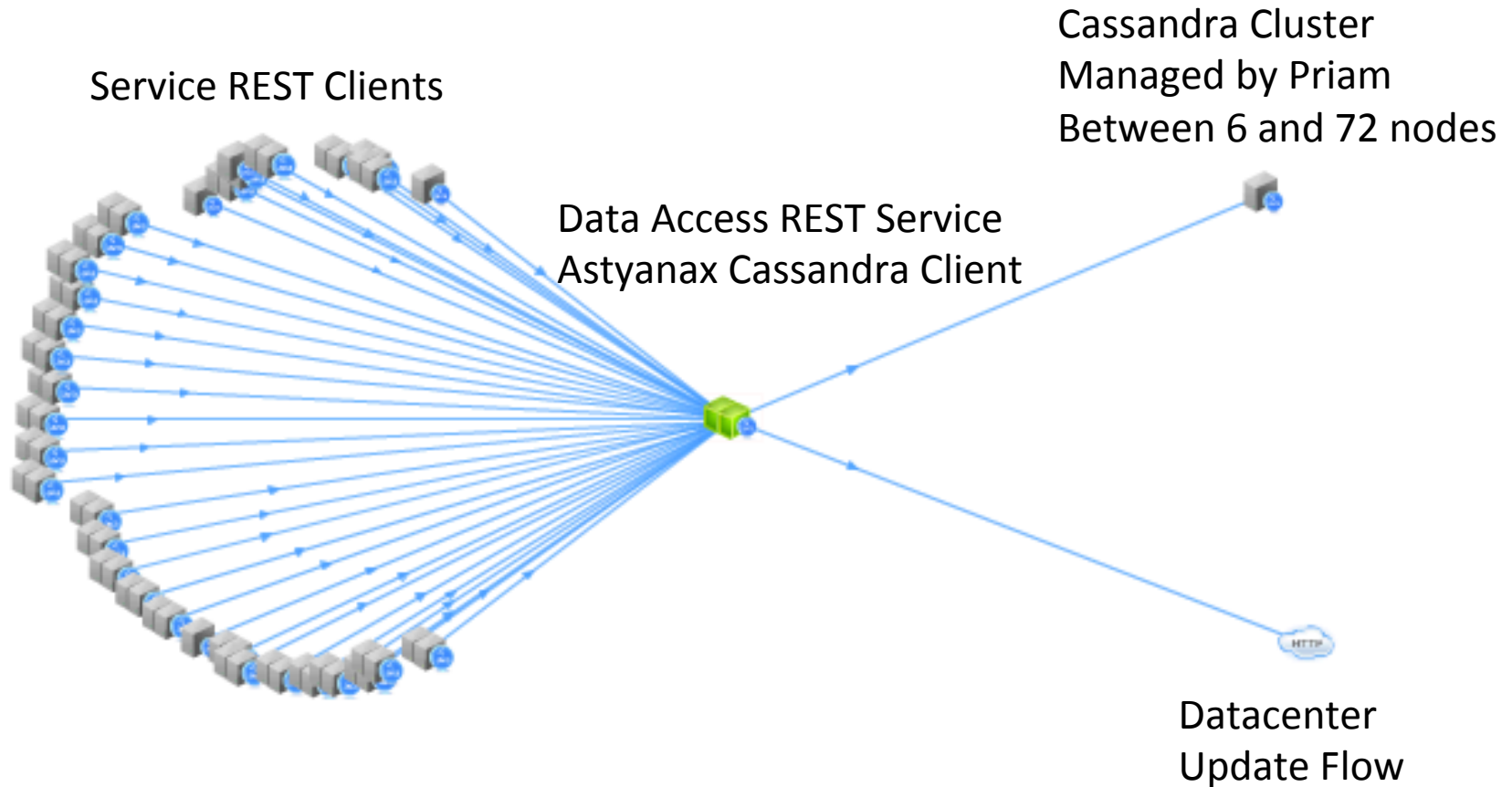
Netflix Deployed on AWS



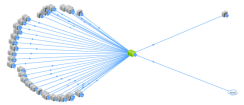
Cassandra on AWS

A highly available and durable
deployment pattern

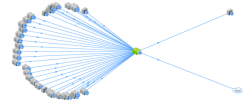
Cassandra Service Pattern



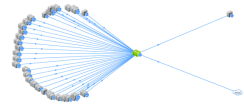
Production Deployment



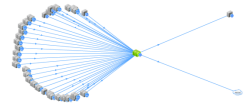
Over 50 Cassandra Clusters



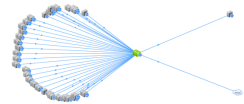
Over 500 nodes



Over 30TB of daily backups



Biggest cluster 72 nodes



1 cluster over 250Kwrites/s

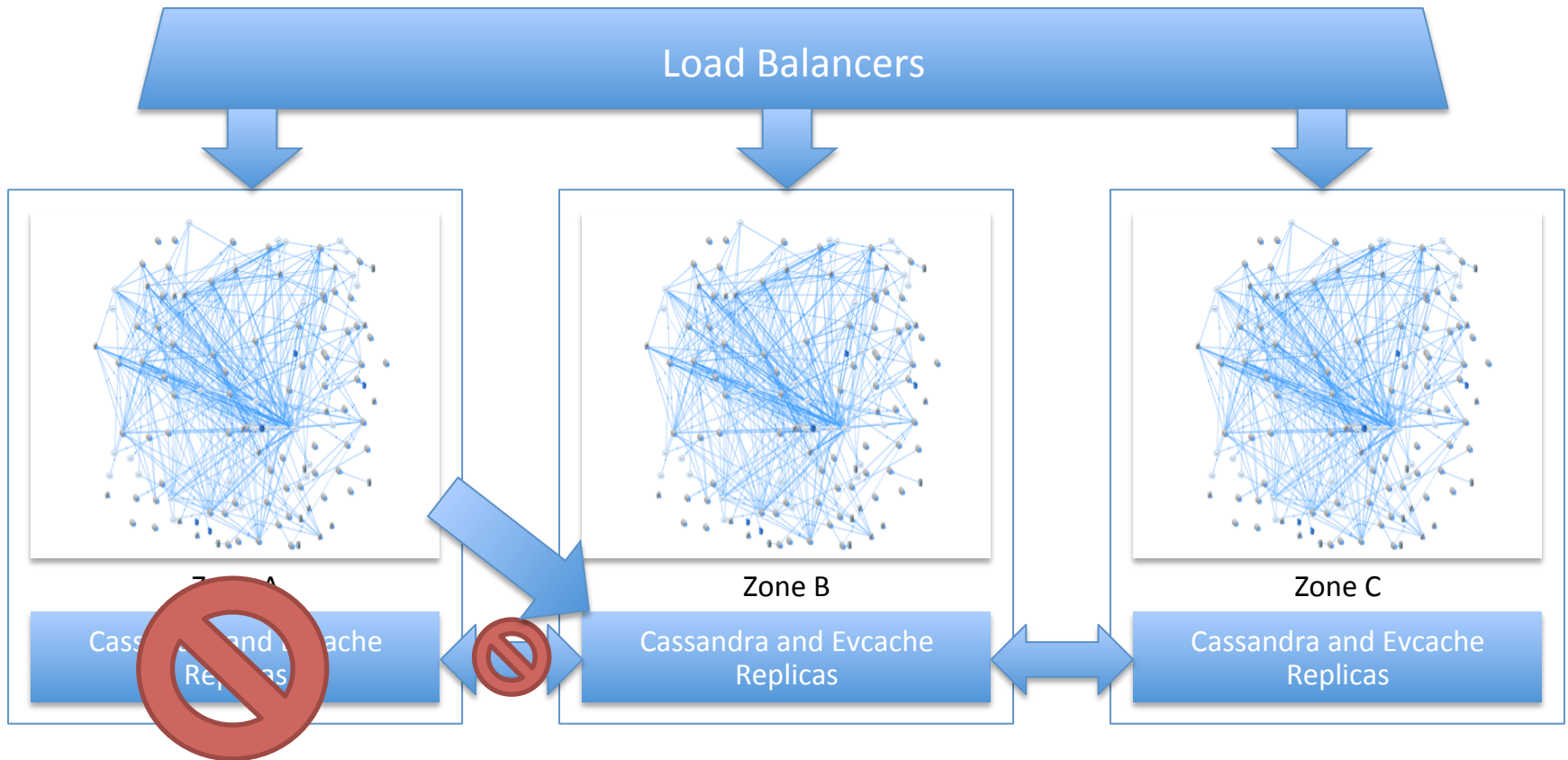
High Availability

- Cassandra stores 3 local copies, 1 per zone
 - Synchronous access, durable, highly available
 - Read/Write One fastest, use for fire and forget
 - Read/Write Quorum 2 of 3, use for read-after-write
- AWS Availability Zones
 - Separate buildings
 - Separate power etc.
 - Fairly close together



Triple Replicated Persistence

Cassandra maintenance drops individual replicas



Cassandra Instance Architecture

Linux Base AMI (CentOS)

Priam

Cassandra
Manager

Token
Management,
Backups,
Autoscaling
Tomcat/Java7

Java7

AppDynamics
appagent
monitoring

GC and thread
dump logging

Cassandra 1.09

Monitoring
Log rotation
AppDynamics
machineagent
Etc.

Priam – Cassandra Automation

Available at <http://github.com/netflix>

- Netflix Platform Tomcat Code
- Zero touch auto-configuration
- State management for Cassandra JVM
- Token allocation and assignment
- Broken node auto-replacement
- Full and incremental backup to S3
- Restore sequencing from S3
- Grow/Shrink Cassandra “ring”

Astyanax

Available at <http://github.com/netflix>

- Features
 - Complete abstraction of connection pool from RPC protocol
 - Fluent Style API
 - Operation retry with backoff
 - Token aware
- Recipes
 - Distributed row lock (without zookeeper)
 - Multi-DC row lock
 - Uniqueness constraint
 - Multi-row uniqueness constraint
 - Chunked and multi-threaded large file storage

Astyanax Query Example

Paginate through all columns in a row

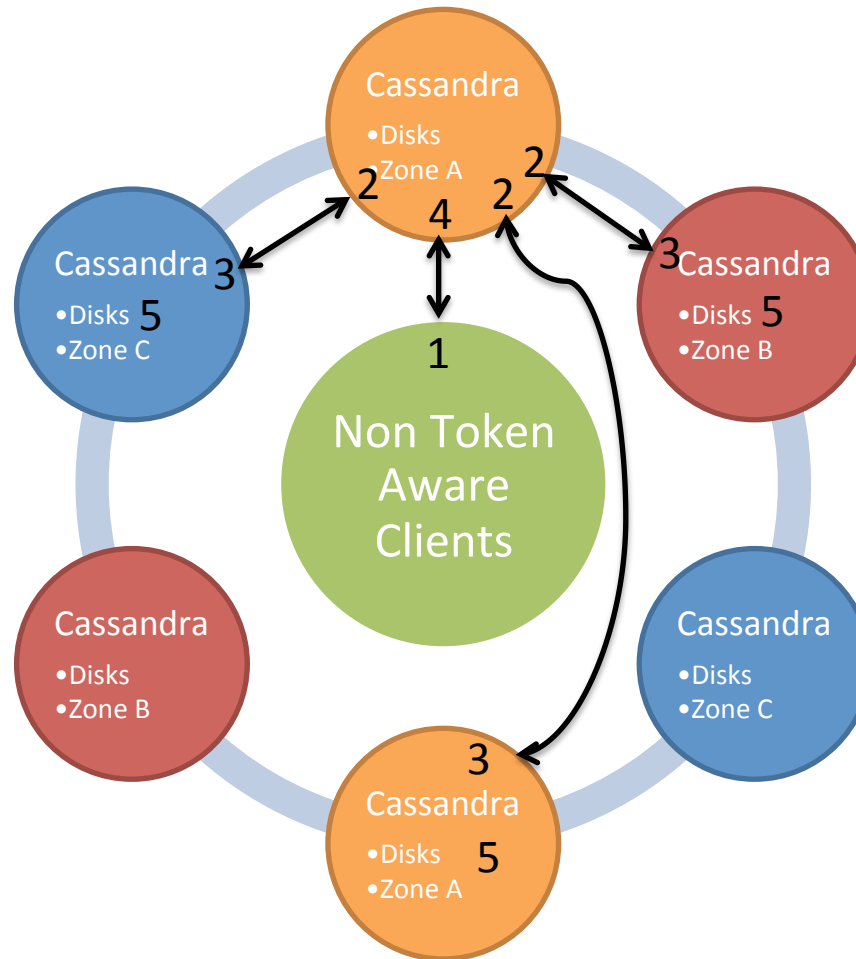
```
ColumnList<String> columns;
int pageize = 10;
try {
    RowQuery<String, String> query = keyspace
        .prepareQuery(CF_STANDARD1)
        .getKey("A")
        .setIsPaginating()
        .withColumnRange(new RangeBuilder().setMaxSize(pageize).build());

    while (!(columns = query.execute().getResult()).isEmpty()) {
        for (Column<String> c : columns) {
        }
    }
} catch (ConnectionException e) {
}
```

“Traditional” Cassandra Write Data Flows

Single Region, Multiple Availability Zone, Not Token Aware

1. Client Writes to any Cassandra Node
2. Coordinator Node replicates to nodes and Zones
3. Nodes return ack to coordinator
4. Coordinator returns ack to client
5. Data written to internal commit log disk (no more than 10 seconds later)



If a node goes offline, hinted handoff completes the write when the node comes back up.

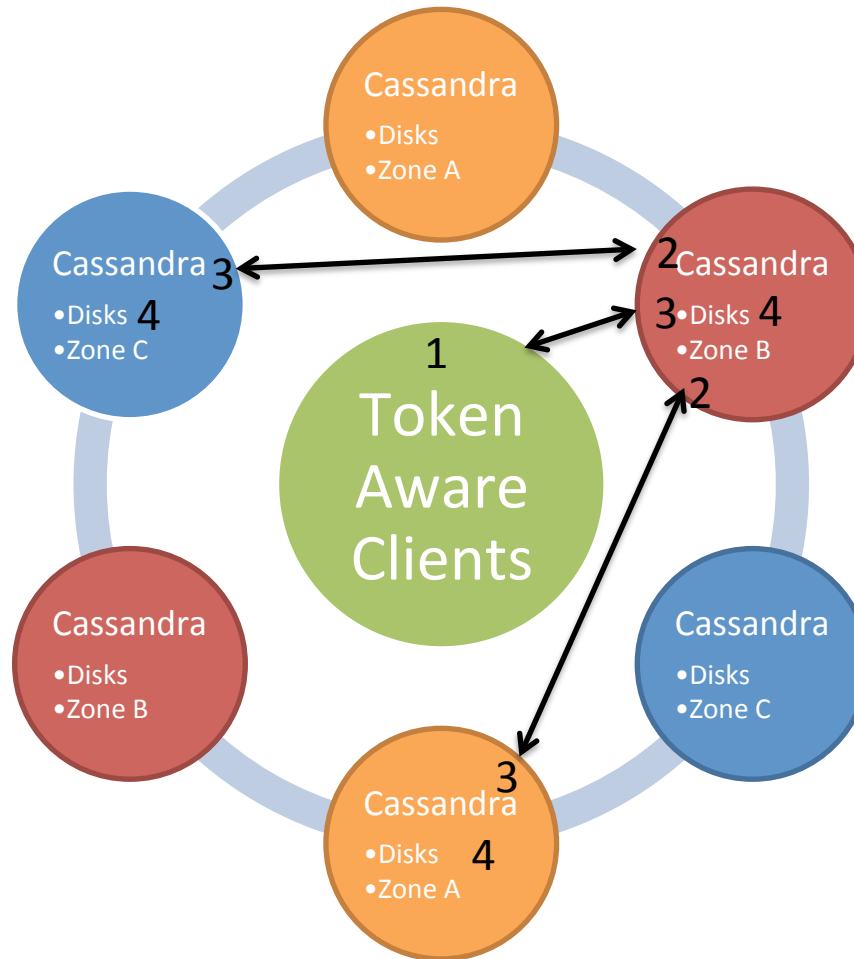
Requests can choose to wait for one node, a quorum, or all nodes to ack the write

SSTable disk writes and compactions occur asynchronously

Astyanax - Cassandra Write Data Flows

Single Region, Multiple Availability Zone, Token Aware

1. Client Writes to local coordinator
2. Coordinator writes to other zones
3. Nodes return ack
4. Data written to internal commit log disks (no more than 10 seconds later)



If a node goes offline, hinted handoff completes the write when the node comes back up.

Requests can choose to wait for one node, a quorum, or all nodes to ack the write

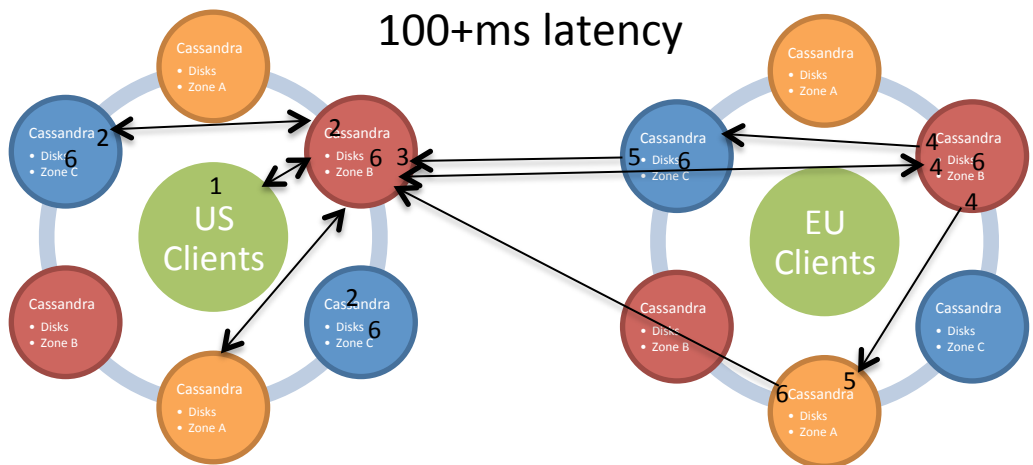
SSTable disk writes and compactions occur asynchronously

Data Flows for Multi-Region Writes

Token Aware, Consistency Level = Local Quorum

1. Client writes to local replicas
2. Local write acks returned to Client which continues when 2 of 3 local nodes are committed
3. Local coordinator writes to remote coordinator.
4. When data arrives, remote coordinator node acks and copies to other remote zones
5. Remote nodes ack to local coordinator
6. Data flushed to internal commit log disks (no more than 10 seconds later)

If a node or region goes offline, hinted handoff completes the write when the node comes back up. Nightly global compare and repair jobs ensure everything stays consistent.



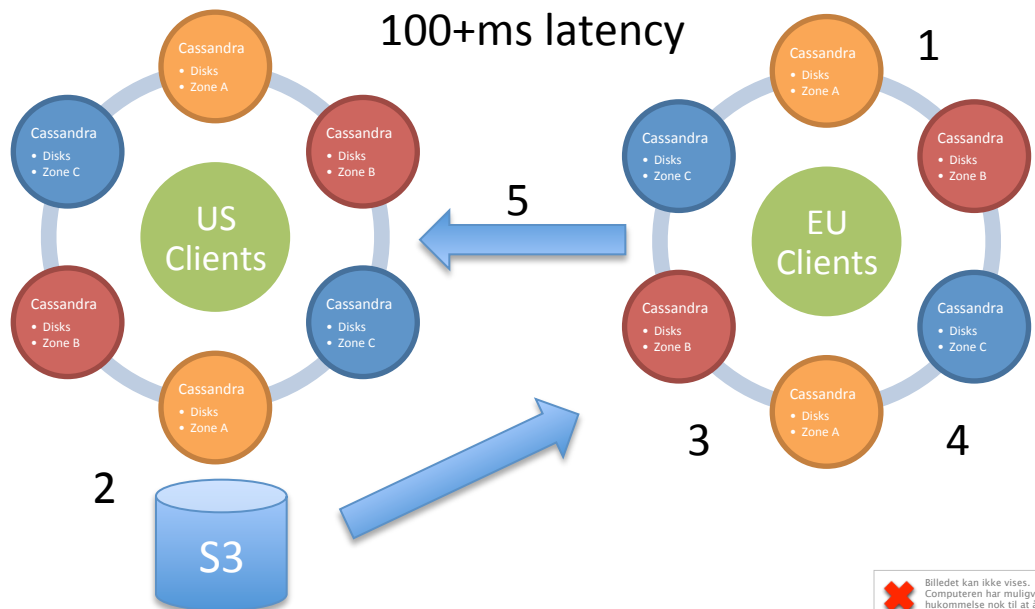
Extending to Multi-Region

Added production UK/Ireland support with no downtime

Minimize impact on original cluster using bulk backup move

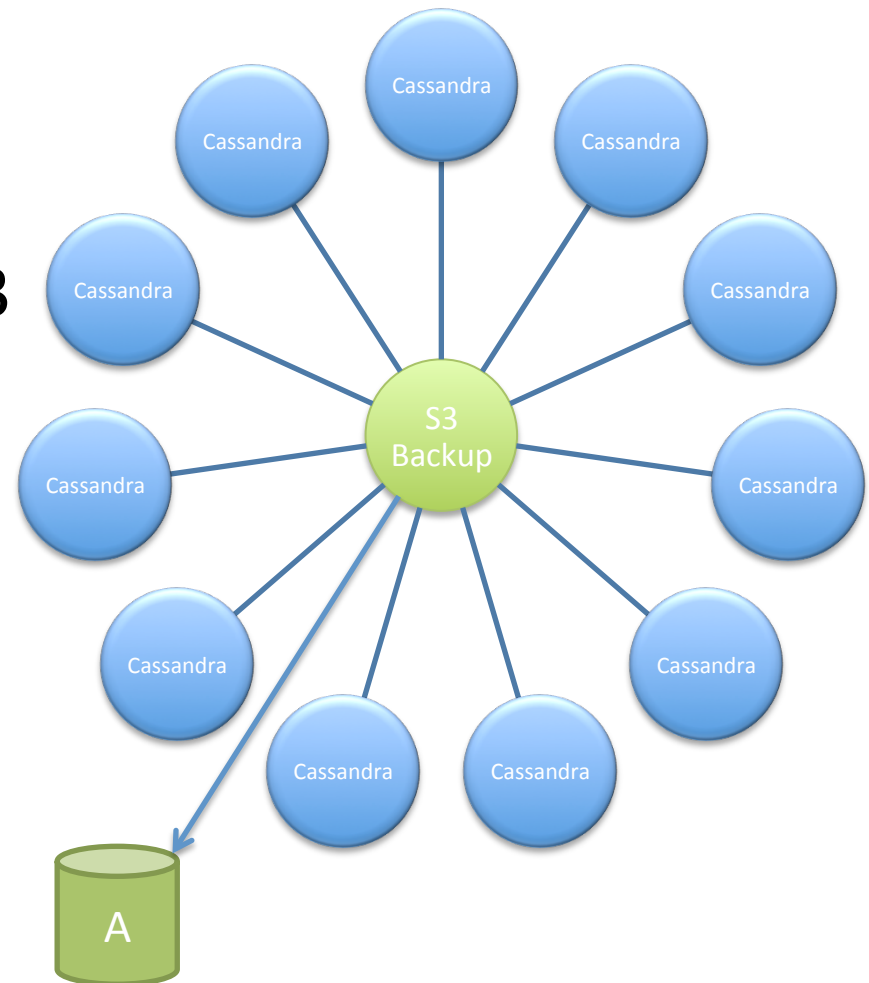
1. Create cluster in EU
2. Backup US cluster to S3
3. Restore backup in EU
4. Local repair EU cluster
5. Global repair/join

Take a Boeing 737 on a domestic flight, upgrade it to a 747 by adding more engines, fuel and bigger wings and fly it to Europe without landing it on the way...



Cassandra Backup

- Full Backup
 - Time based snapshot
 - SSTable compress -> S3
- Incremental
 - SSTable write triggers compressed copy to S3
- Archive
 - Copy cross region



Cassandra Explorer

Open source on github soon

CASSANDRA EXPLORER: -

Region: test.eu-west-1

[Explorer](#) | [Admin](#) | [Dashboard](#)

Filter: 0 Refresh

[Home](#)

Cluster: CASS_SANDBOX

Keyspace: CacheContent

StrategyClass org.apache.cassandra.locator.NetworkTopologyStrategy
us-east 3
Column Families: [RegionalManifestIndex](#) [ReportedManifest](#) [RegionalManifest](#)

Keyspace: vault2

StrategyClass org.apache.cassandra.locator.NetworkTopologyStrategy
us-east 3
Column Families: [mopstore](#)

Keyspace: AstyanaxUnitTests

StrategyClass org.apache.cassandra.locator.SimpleStrategy
replication_factor 3
Column Families: [CompositeKev](#) [LongColumn1](#) [users](#) [CompositeColumn](#) [ClickStream](#) [Standard2](#) [Standard1](#)
[CompositeCsv](#) [TimeUUID1](#) [Counter1](#)

Keyspace: trackid_Keyspace

StrategyClass org.apache.cassandra.locator.NetworkTopologyStrategy
us-east 3
Column Families: [trackIds](#)

Keyspace: vault

- ABCCASSANDRA
- CASS_ABMR_2
- CASS_API_MULTIREGION
- CASS_API_TEST
- CASS_BIG_SCALE
- CASS_BOOTSTRAP
- CASS_BRISK_TEST
- CASS_BULKPREDICTION
- CASS_CCS
- CASS_CDE
- CASS_DMS
- CASS_ECOMM
- CASS_GENPOP
- CASS_GPSAPPLICATION_US
- CASS_MERCH
- CASS_MRGENPOP
- CASS_MRSUB
- CASS_MRTEST
- CASS_NCCP_US
- CASS_ORESTES
- CASS_PBS_US
- CASS_PERF_SUB
- CASS_QAREG_SUBSCRIBER
- CASS_RTAL_GPS
- ▲ CASS_SANDBOX
 - CacheContent
 - vault2
 - AstyanaxUnitTests
 - trackid_Keyspace
 - vault
 - video_presentations_Keyspace
 - IntTest
- CASS_SOCIAL
- CASS_STREAM_LATAM
- CASS_STREAM_MR
- CASS_STREAM_REFRESH
- CASS_SUBSCRIBER
- CASS_TEST_JYOTI
- CASS_TRACERS
- CASS_TURTLE
- CASS_VMS



Billedet kan ikke vises.
Computeren har muligvis ikke
hukommelse nok til at åbne
billedet, eller billedet er
muligvis blevet beskadiget.
Genstart computeren, og åbn
derefter filen igen. Hvis det

ETL for Cassandra

- Data is de-normalized over many clusters!
- Too many to restore from backups for ETL
- Solution – read backup files using Hadoop
- Aegisthus
 - <http://techblog.netflix.com/2012/02/aegisthus-bulk-data-pipeline-out-of.html>
 - High throughput raw SSTable processing
 - Re-normalizes many clusters to a consistent view
 - Extract, Transform, then Load into Teradata

Asgard

<http://techblog.netflix.com/2012/06/asgard-web-based-cloud-management-and.html>

- Replacement for AWS Console at Scale
 - Groovy/Grails/JVM based
 - Supports all AWS regions on a global basis
 - Specific to AWS feature set
- Hides the AWS credentials
 - Use AWS IAM to issue restricted keys for Asgard
 - Each Asgard instance manages one account
 - One install each for test, prod, audit accounts

Open Source Projects

Legend

Github / Techblog

Apache Contributions

Techblog Post

Coming Soon

Priam

Cassandra as a Service

Astyanax

Cassandra client for Java

CassJMeter

Cassandra test suite

Cassandra Multi-region EC2
datastore support

Aegisthus

Hadoop ETL for Cassandra

Explorers

Governator - Library lifecycle
and dependency injection

Odin

Workflow orchestration

Async logging

Exhibitor

Zookeeper as a Service

Curator

Zookeeper Patterns

EVCache

Memcached as a Service

Eureka / Discovery
Service Directory

Archaius

Dynamics Properties Service

EntryPoints

Server-side latency/error
injection

REST Client + mid-tier LB

Configuration REST endpoints

Servo and Autoscaling Scripts

Honu

Log4j streaming to Hadoop

Circuit Breaker

Robust service pattern

Asgard - AutoScaleGroup
based AWS console

Chaos Monkey

Robustness verification

Latency Monkey

Janitor Monkey

Bakeries and AMI

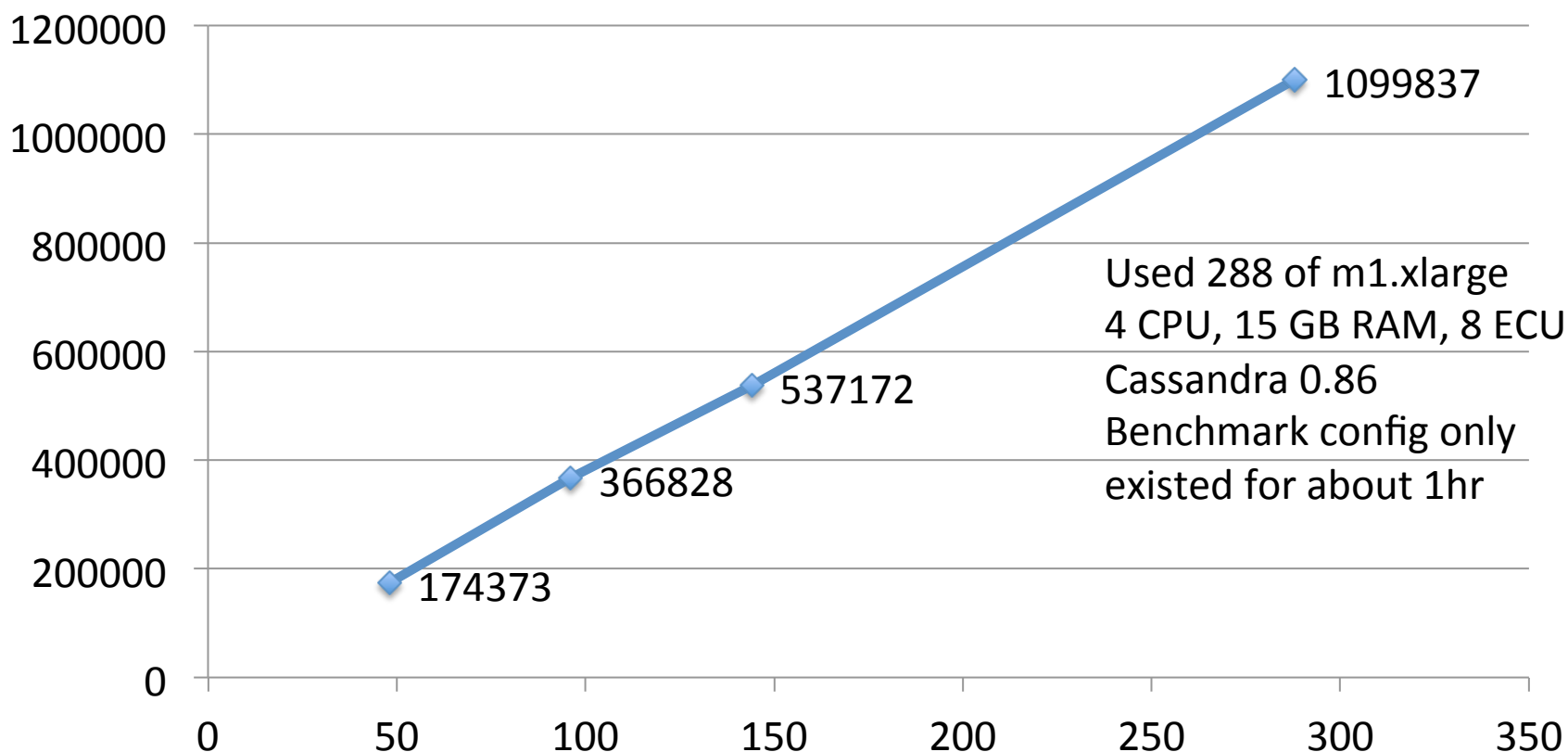
Build dynaslaves

Benchmarks and Scalability

Scalability from 48 to 288 nodes on AWS

<http://techblog.netflix.com/2011/11/benchmarking-cassandra-scalability-on.html>

Client Writes/s by node count – Replication Factor = 3



ONE MILLION WRITES/SEC in the CLOUD? That's Fast.
APACHE CASSANDRA™ Fast.



✗ Billedet kan ikke vises.
Computeren har muligvis ikke
hukommelse nok til at åbne
billedet, eller billedet er
muligvis blevet beskadiget.
Genstart computeren, og åbn
derefter filen igen. Hvis det

“Some people skate to the puck,
I skate to where the puck is going to be”
Wayne Gretzky



Cassandra on AWS

The Past

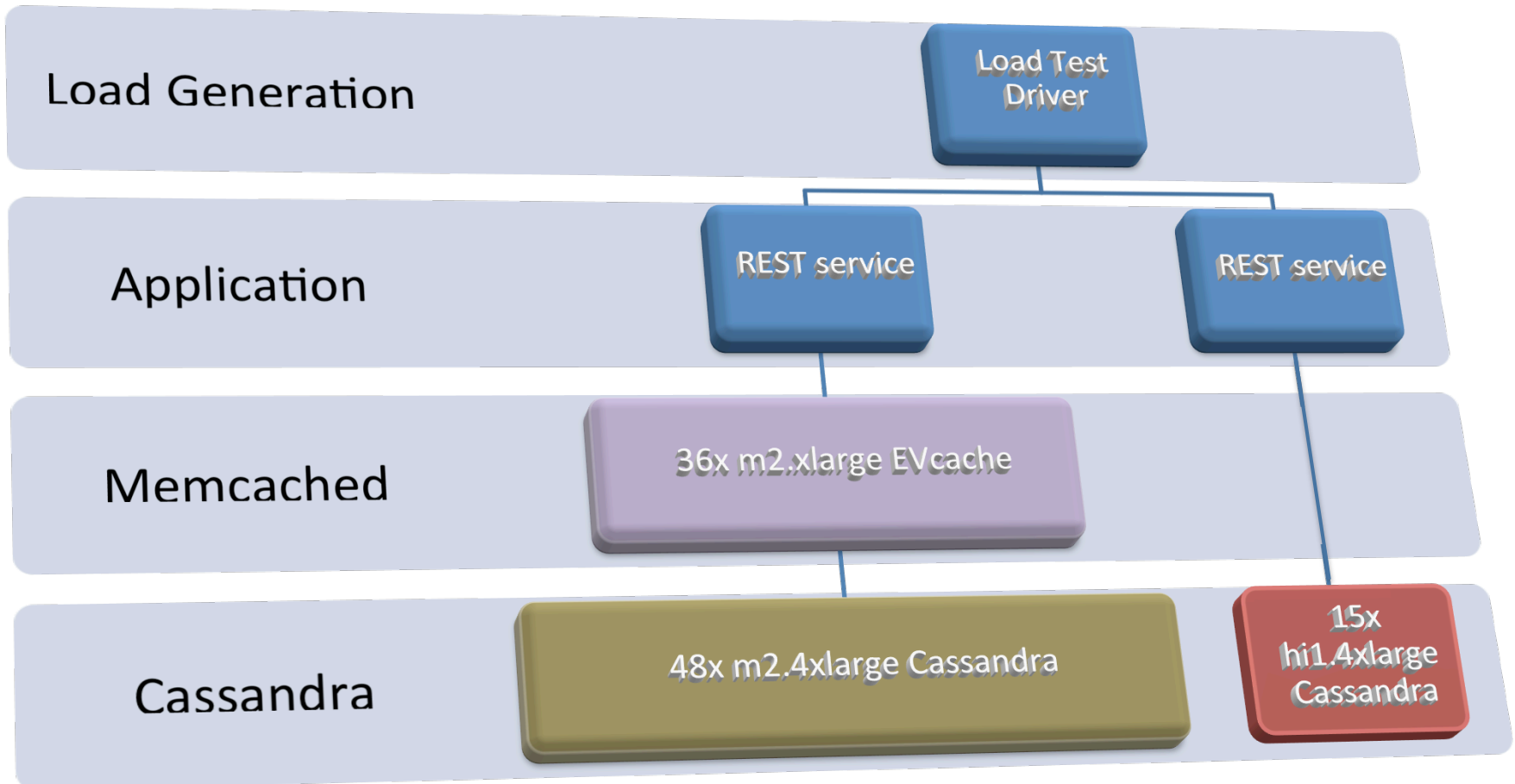
- Instance: m2.4xlarge
- Storage: 2 drives, 1.7TB
- CPU: 8 Cores, 26 ECU
- RAM: 68GB
- Network: 1Gbit
- IOPS: ~500
- Throughput: ~100Mbyte/s
- Cost: \$1.80/hr

The Future

- Instance: hi1.4xlarge
- Storage: 2 SSD volumes, 2TB
- CPU: 8 HT cores, 35 ECU
- RAM: 64GB
- Network: 10Gbit
- IOPS: ~100,000
- Throughput: ~1Gbyte/s
- Cost: \$3.10/hr

Cassandra Disk vs. SSD Benchmark

Same Throughput, Lower Latency, Half Cost



Get stuck with wrong config

Wait Wait File tickets

Ask permission Wait Wait

Wait Things we don't do Wait

Run out of space/power

Plan capacity in advance

Have meetings with IT Wait

Things we do do. Run benchmarks.
Live on stage.
(but this time it's a recording).

A Mars lander is shown in the process of landing on the surface of Mars. The lander is a complex structure with a descent stage and a rover. The descent stage is positioned above the rover, and both are illuminated by bright yellow lights. The word "YOLO" is superimposed in large, white, bold letters with a black outline across the center of the image. The background is a hazy, orange-brown sky with wispy clouds.

YOLO

Live Demo Workload

- Jenkins automation
 - Jmeter load driver
 - Asgard provisioning
 - Priam instance management
- Traffic
 - Reading/writing whole 100 column rows
 - Randomly selected from 25M row keys
 - Run for 10minutes, then double ring size

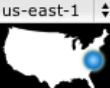
Screenshots from Live Demo

Cassandra Summit 2012 last July

Live on stage demo

Asgard

cass_perf apps, with no instances running

 **ASGARD** test us-east-1 

[Home](#) [App](#) [AMI](#) [Cluster](#) [ELB](#) [EC2](#) [SDB](#) [SNS](#) [SQS](#) [RDS](#) [Task](#)

Registered Applications

+ Create New Application		Filter: cass_perf 3						
Name	ASGs	Instances	Type	Description	Email	Owner	Create Time	Update Time
 cass_perf_m2_test	 0	 0	Web Service	Test Cassandra perf		Denis	2012-08-07 15:37:37 PDT	2012-08-07 15:37:37 PDT
 cass_perf_sub	 0	 0	Web Service	Test Subscriber with live data		Denis Sheahan	2011-08-15 17:02:26 PDT	2011-08-15 17:02:26 PDT
 cass_perf_test	 0	 0	Standalone Application	Cassandra performance test		Denis	2012-08-03 06:50:13 PDT	2012-08-03 06:50:13 PDT

Jenkins

Jenkins perf_test jobs



[Back to Dashboard](#)

[Status](#)

[Changes](#)

[Workspace](#)

[Build Now](#)

[Delete Project](#)

[Configure](#)

[Job Config History](#)

Build History [\(trend\)](#)

#5	Aug 8, 2012 10:35:54 AM	35MB
#4	Aug 8, 2012 9:07:48 AM	48MB
#3	Aug 8, 2012 8:18:29 AM	141KB
#2	Aug 8, 2012 7:27:44 AM	5MB
#1	Aug 8, 2012 6:53:30 AM	115KB

[RSS for all](#) [RSS for failures](#)

Project ZZ-cassandra-m2-redux-test

test launching Cassandra



[Workspace](#)



[Last Successful Artifacts](#)



[Recent Changes](#)

Permalinks

- [Last build \(#5\), 1 day 2 hr ago](#)
- [Last stable build \(#5\), 1 day 2 hr ago](#)
- [Last successful build \(#5\), 1 day 2 hr ago](#)
- [Last unsuccessful build \(#5\), 1 day 2 hr ago](#)



[Back to Dashboard](#)

[Status](#)

[Changes](#)

[Workspace](#)

[Build Now](#)

[Delete Project](#)

[Configure](#)

[Job Config History](#)

Build History [\(trend\)](#)

#4	Aug 8, 2012 10:36:01 AM	53MB
#3	Aug 8, 2012 8:42:54 AM	52MB
#2	Aug 8, 2012 7:27:34 AM	298KB
#1	Aug 7, 2012 10:29:02 PM	348KB

[RSS for all](#) [RSS for failures](#)

Project ZZ-cassandra-redux-test

test launching Cassandra



[Workspace](#)



[Last Successful Artifacts](#)



[Recent Changes](#)

Permalinks

- [Last build \(#4\), 1 day 2 hr ago](#)
- [Last stable build \(#4\), 1 day 2 hr ago](#)
- [Last successful build \(#4\), 1 day 2 hr ago](#)
- [Last failed build \(#1\), 1 day 14 hr ago](#)
- [Last unsuccessful build \(#2\), 1 day 5 hr ago](#)

Jmeter Setup

Build parameters

ns

search

aco

-cassandra-redux-test #4 Parameters

ENABLE A

Project

Output [raw]

Information

Build

ters

Environment Variables

Build

Build #4

Parameters

WORKLOAD_COMMENT

Add a comment to describe what this run is doing

PROFILE

parallel

Will the run be parallel, step,sinusoid or eof (run until single file exhausted)

DURATION

1800

For Timed experiment how long to run

INSTANCES

20

How many instances should we launch

JMETER_EXPERIMENT

cass_write_read_rows.jmx

The JMeter .jmx experiment to run. By default copied from depot/cloud/performance/jmeter-test-harness/performance/tests

JMETER_PROPERTY

values to pass to the Jmeter call. Format can be name,value. Will be passed to JMeter as -Jname=value. Can leave blank

REMOVE_JMETER_INSTANCES

1

Should we cleanup the JMeter instances at the end

CSV_LIST

active1aa.csv,active1ab.csv,active2aa.csv,active2ab.csv

CSV files to copy over from data directory. Can be "none" By default copied from depot/cloud/performance/jmeter-test-harness/performance/data

SPLIT_CSV

1

Jmeter Setup

Build parameters

!-cassandra-redux-test

#4

Parameters

ENABLE #

SPLIT_CSV

CSV files to copy over from data directory. Can be "none" By default copied from depot/cloud/performance/jmeter-test-harness/performance/data

1

THREAD_LIST

Split the CSV files evenly across the clients

20

LOOP_COUNT

List of thread counts for each experiment eg 10 20 30. Experiments will be run in order

timed

REGION

Type of experiment - "timed", "forever" or number_of_iterations

us-east-1

TARGET_ASG

Region to run the test in

cass_perf_test,0,11

JMETER_SAMPLE_NAMES

The service ASG that we are targeting performance. Will create a servers.csv file with a list of instances in this service. Parameter is ,,

Cassandra_Batch_Put,Cassandra_Get_Range_Slice

MERGE_DATA

List of Samplers in the JMeter experiment, this list will be post processed and plotted

0

EPIC_CONFIG

Should we merge the JMeter results from all the client machines

cass_scale_list,cluster:cass_perf_test@--useast1e.awstest+cass_scale_list,cluster:cass_perf_test@--useast1c.awstest+cass

NODESTATS

List of epics to dump at the end of the run parameter format list1,cluster:list2:cluster.... list format: output_file,title,counter1...counterN

cass_perf_test,0,3

COLLECT_PROFILE

Collect stats on Target ASG. Parameter is , or none

none

Collect Studio profile on node, Parameter is , or none



Billedet kan ikke vises.
Computeren har muligvis ikke
hukommelse nok til at åbne
billedet, eller billedet er
muligvis blevet beskadiget.
Genstart computeren, og åbn
derefter filen igen. Hvis det

Jmeter Setup

Build parameters

!-cassandra-redux-test

#4

Parameters

[ENABLE #](#)

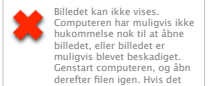
	Collect Studio profile on node, Parameter is , or none
COLLECT_HEAP_ALLOCATIONS	none
	Collect Heap allocations on node with hprof. Parameter is , or none
CLEANUP_SCRIPT	none
	script,parameter to reset the environment after the run. Example is an AB delete script
TARGET_DEPLOY	ami-69f85000 hi1.4xlarge 1 1 create_keyspace
	How to deploy the target AMI instance-type
CASSANDRA	1
JMETER_INSTANCE_TYPE	m2.2xlarge
	The size to use for JMeter
TARGET_TEARDOWN	1
	Delete the target after test
PRESERVE_RUN	0
	Should this run be preserved for a replay
PARALLEL_SCRIPT	800:cass_perf_test.create_double_ring
	Kick this script of at the start of the load to run in parallel
DELTA	240
	Offset subsequent thread groups
REPLAY_BUILD	jmeter_client-ZZ_cassandra_test-54



[ocalize this page](#)

Page generated: Aug 9, 2012 1:32:52 PM

[Jenki](#)



Asgard

initial set of cass instances up and running

 **ASGARD** test us-east-1 

[Home](#) [App](#) [AMI](#) [Cluster](#) [ELB](#) [EC2](#) [SDB](#) [SNS](#) [SQS](#) [RDS](#) [Task](#)

Registered Applications

[+ Create New Application](#) Filter: 3

Name	ASGs	Instances	Type	Description	Email	Owner	Create Time	Update Time
 cass_perf_m2_test	 3	 12	Web Service	Test Cassandra perf		Denis	2012-08-07 15:37:37 PDT	2012-08-07 15:37:37 PDT
 cass_perf_sub	 0	 0	Web Service	Test Subscriber with live data		Denis Sheahan	2011-08-15 17:02:26 PDT	2011-08-15 17:02:26 PDT
 cass_perf_test	 3	 12	Standalone Application	Cassandra performance test		Denis	2012-08-03 06:50:13 PDT	2012-08-03 06:50:13 PDT

Kiklos (open source soon)

Clusters growing from 12 to 24

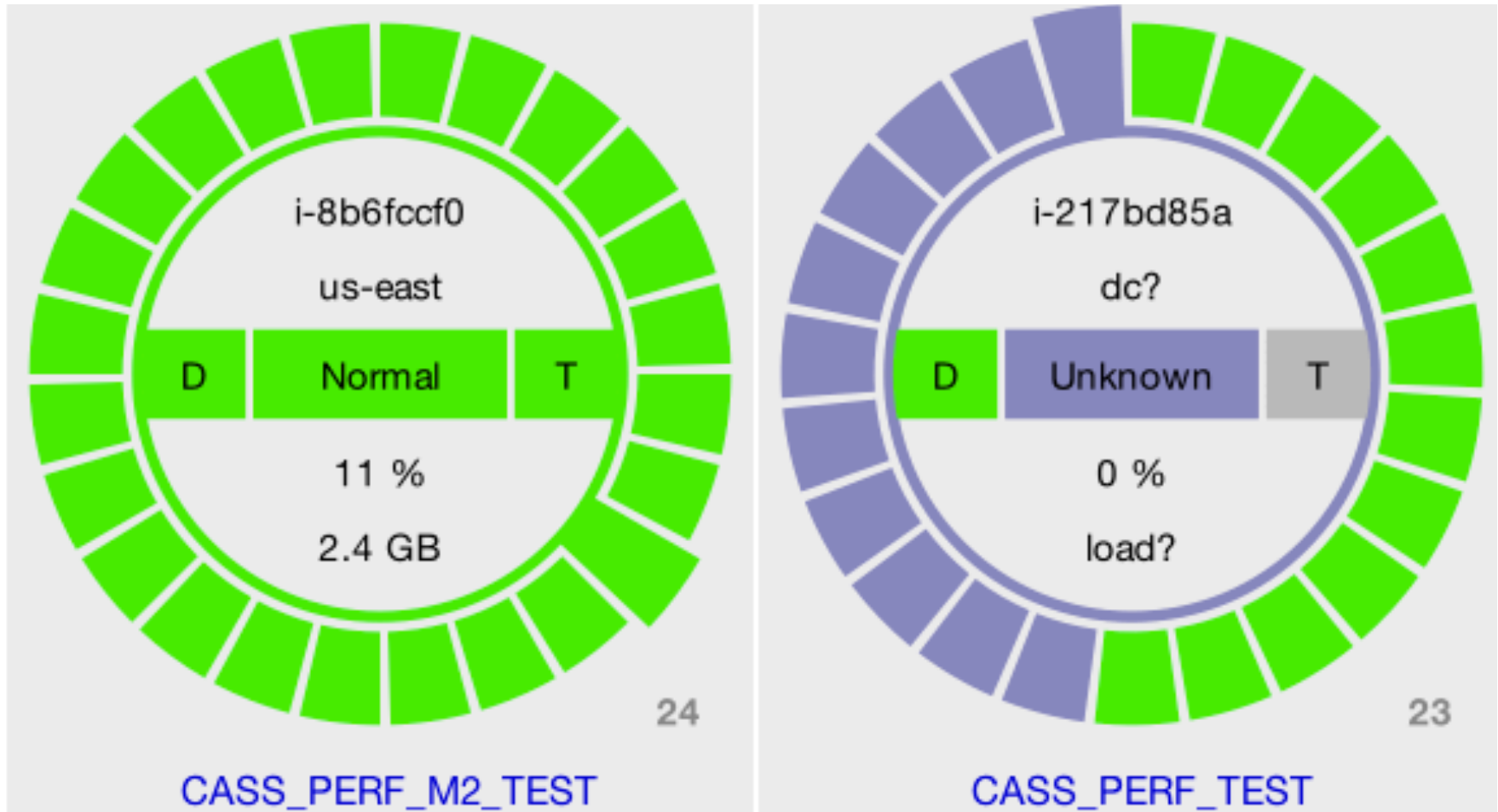
in-service, bootstrapping, garbage-collecting, cass-down



Kiklos

Clusters growing from 12 to 24

in-service, bootstrapping, garbage-collecting, cass-down

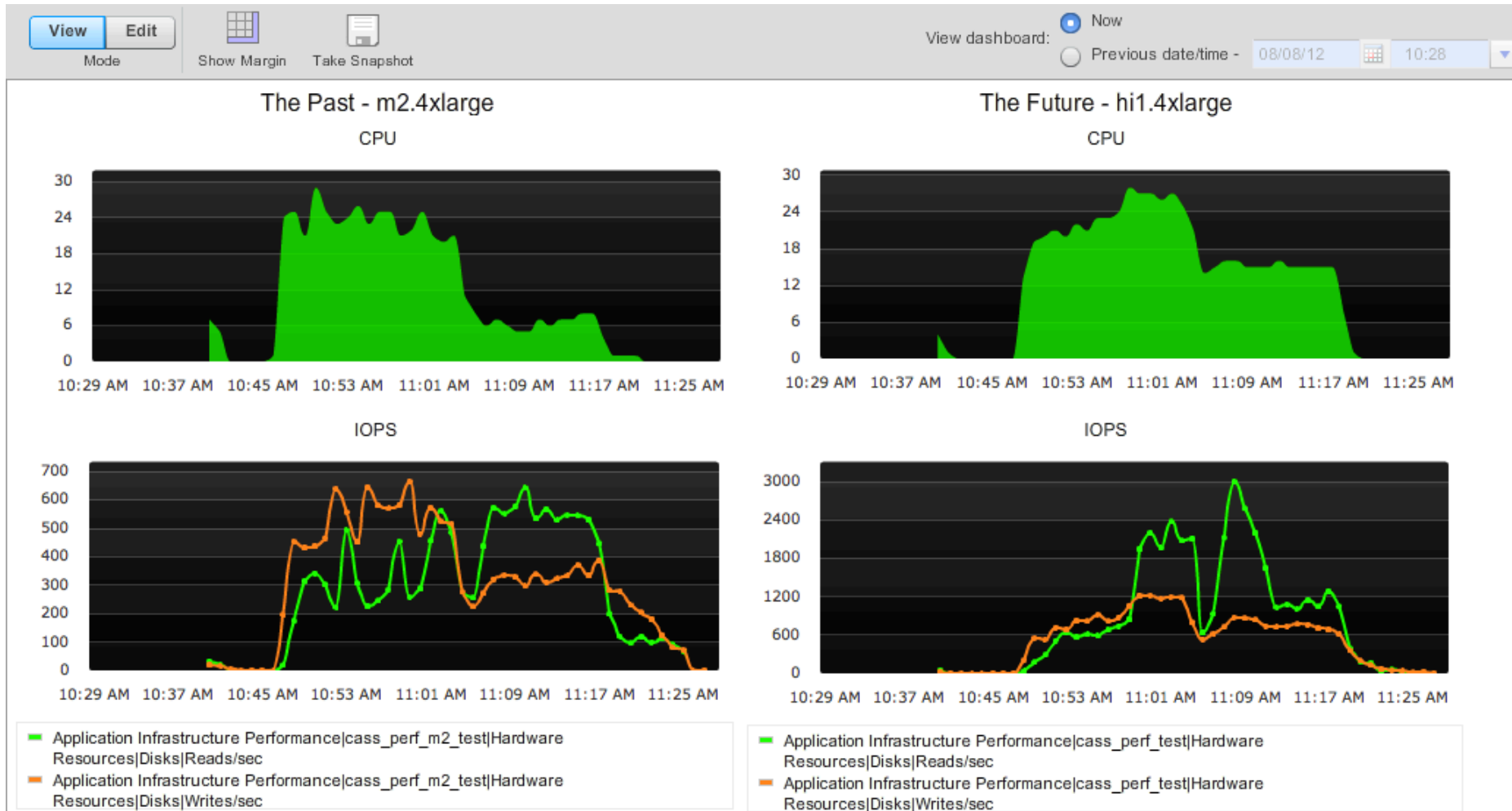


Disclaimers

- We didn't have time to tune the demo
- These are the plots from the live demo run
- Run's need to be longer to get to steady state
- Data size only reached around 5GB per node
- Plenty of "I wonder why it did that" remains
- It's a fair comparison, but not the best absolute performance possible for this workload and configuration
- When you remove the IO bottleneck, the next few bottlenecks appear...

Activity during the talk 10:30-11:30

Custom AppDynamics dashboard showing CPU and IOPS per node



Jmeter Plots

- Plots are the output of the Jenkins build
- Each instance has its own set of plots
- Each availability zone has its own summary plots
- One of the three zone summary plots is compared for each metric
- Plot collection is currently duplicated as we are transitioning from “Epic” to “Atlas”

Jenkins

Collected results and graphs after job has completed

Jenkins / 22-cassandra-redux-test / #4

 [Back to Project](#)

 [Status](#)

 [Changes](#)

 [Console Output](#) [\[raw\]](#)

 [Edit Build Information](#)

 [Delete Build](#)

 [Parameters](#)

 [Injected Environment Variables](#)

 [Rebuild](#)

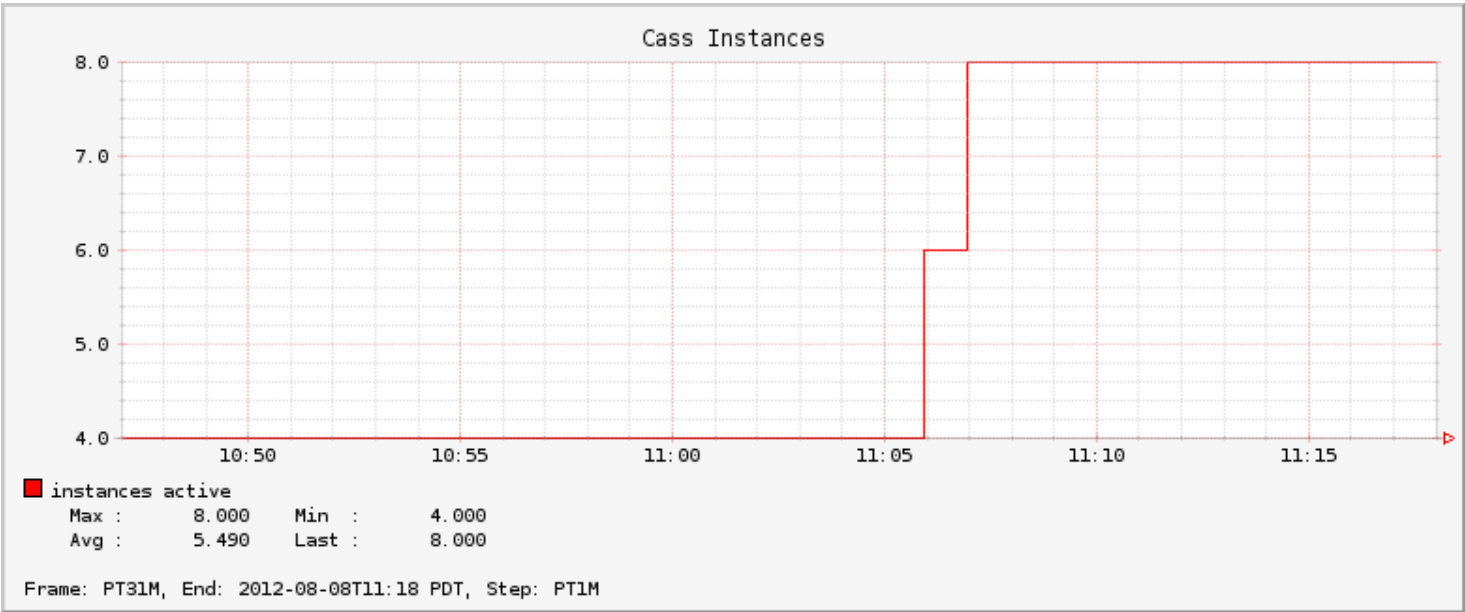
 [Previous Build](#)

 [cloud](#) / [performance](#) / [jmeter-test-harness](#) / [performance](#) / [results](#) / [combined_results](#) / *.png

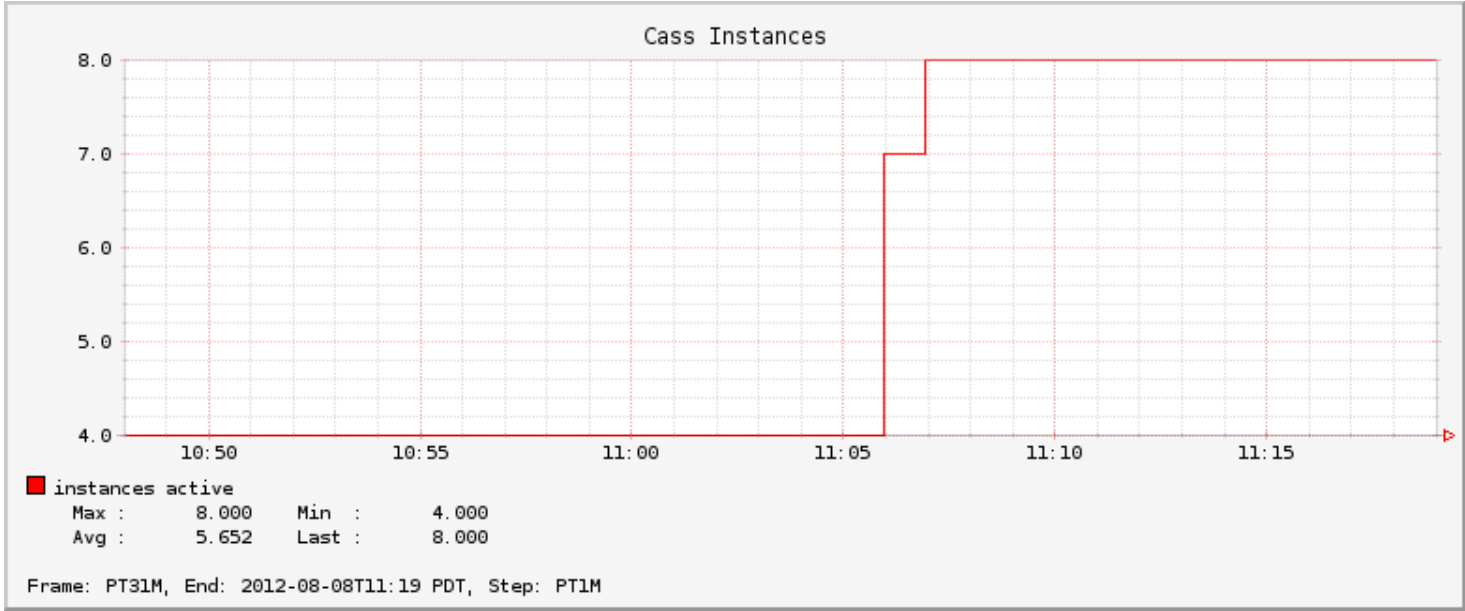
 Cass-Big-Compact.cass_perf_test-4--useast1c.awstest.20.atlas.png	12762	 view
 Cass-Big-Compact.cass_perf_test-4--useast1c.awstest.20.png	24937	 view
 Cass-Big-Compact.cass_perf_test-4--useast1d.awstest.20.atlas.png	12574	 view
 Cass-Big-Compact.cass_perf_test-4--useast1d.awstest.20.png	24881	 view
 Cass-Big-Compact.cass_perf_test-4--useast1e.awstest.20.atlas.png	12670	 view
 Cass-Big-Compact.cass_perf_test-4--useast1e.awstest.20.png	25322	 view
 Cass-CPU.cass_perf_test-4--useast1c.awstest.20.atlas.png	11298	 view
 Cass-CPU.cass_perf_test-4--useast1c.awstest.20.png	22025	 view
 Cass-CPU.cass_perf_test-4--useast1d.awstest.20.atlas.png	11263	 view
 Cass-CPU.cass_perf_test-4--useast1d.awstest.20.png	21766	 view
 Cass-CPU.cass_perf_test-4--useast1e.awstest.20.atlas.png	11160	 view
 Cass-CPU.cass_perf_test-4--useast1e.awstest.20.png	22270	 view
 Cass-Compact-Complete.cass_perf_test-4--useast1c.awstest.20.atlas.png	12696	 view
 Cass-Compact-Complete.cass_perf_test-4--useast1c.awstest.20.png	27405	 view
 Cass-Compact-Complete.cass_perf_test-4--useast1d.awstest.20.atlas.png	13187	 view
 Cass-Compact-Complete.cass_perf_test-4--useast1d.awstest.20.png	24989	 view
 Cass-Compact-Complete.cass_perf_test-4--useast1e.awstest.20.atlas.png	13244	 view
 Cass-Compact-Complete.cass_perf_test-4--useast1e.awstest.20.png	25210	 view
 Cass-Compact-Pending.cass_perf_test-4--useast1c.awstest.20.atlas.png	12579	 view
 Cass-Compact-Pending.cass_perf_test-4--useast1c.awstest.20.png	24243	 view
 Cass-Compact-Pending.cass_perf_test-4--useast1d.awstest.20.atlas.png	12674	 view
 Cass-Compact-Pending.cass_perf_test-4--useast1d.awstest.20.png	24164	 view
 Cass-Compact-Pending.cass_perf_test-4--useast1e.awstest.20.atlas.png	13001	 view
 Cass-Compact-Pending.cass_perf_test-4--useast1e.awstest.20.png	24322	 view

The past
m2.4xlarge

Instances
per zone

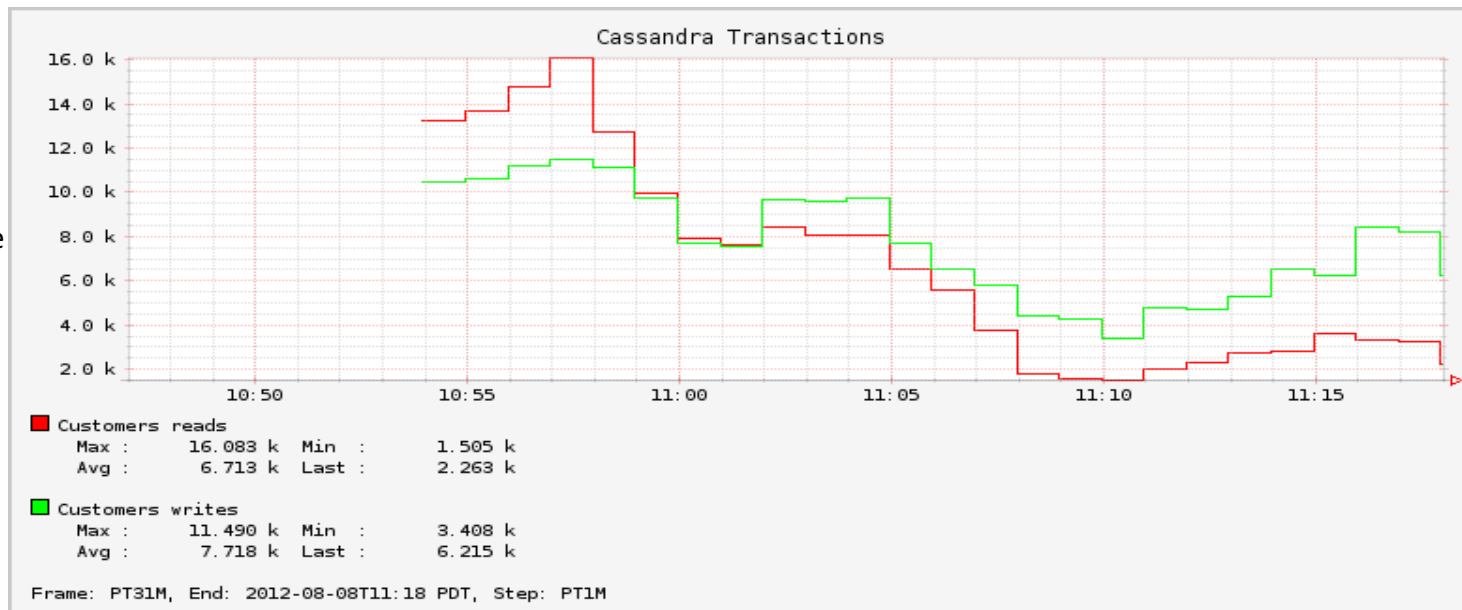


The future
hi1.4xlarge

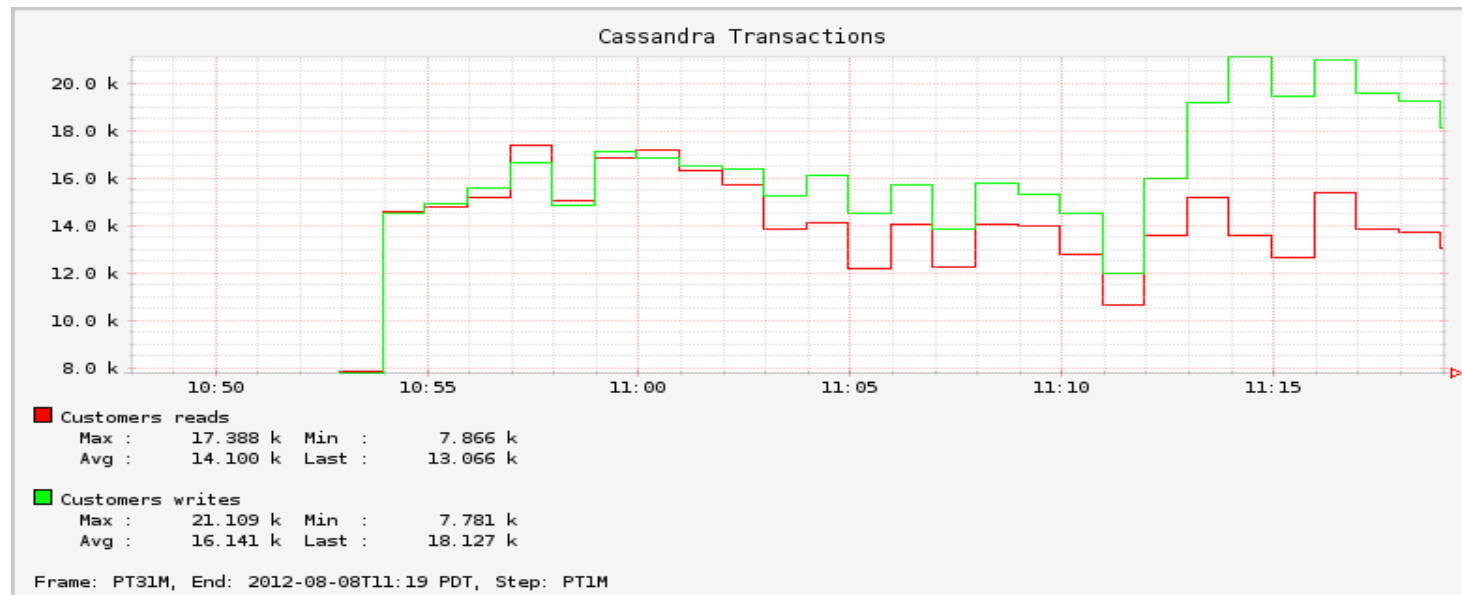


The past
m2.4xlarge

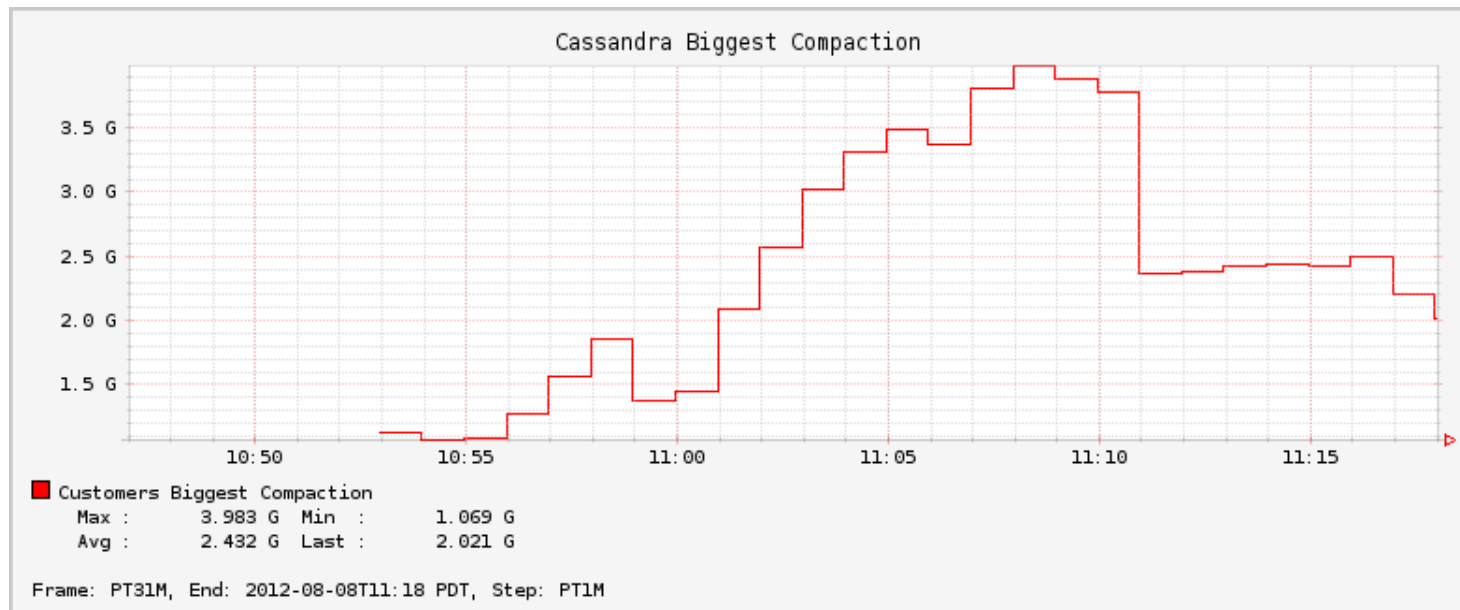
Transactions
per zone, same
as total client
transactions



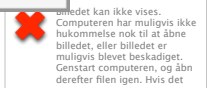
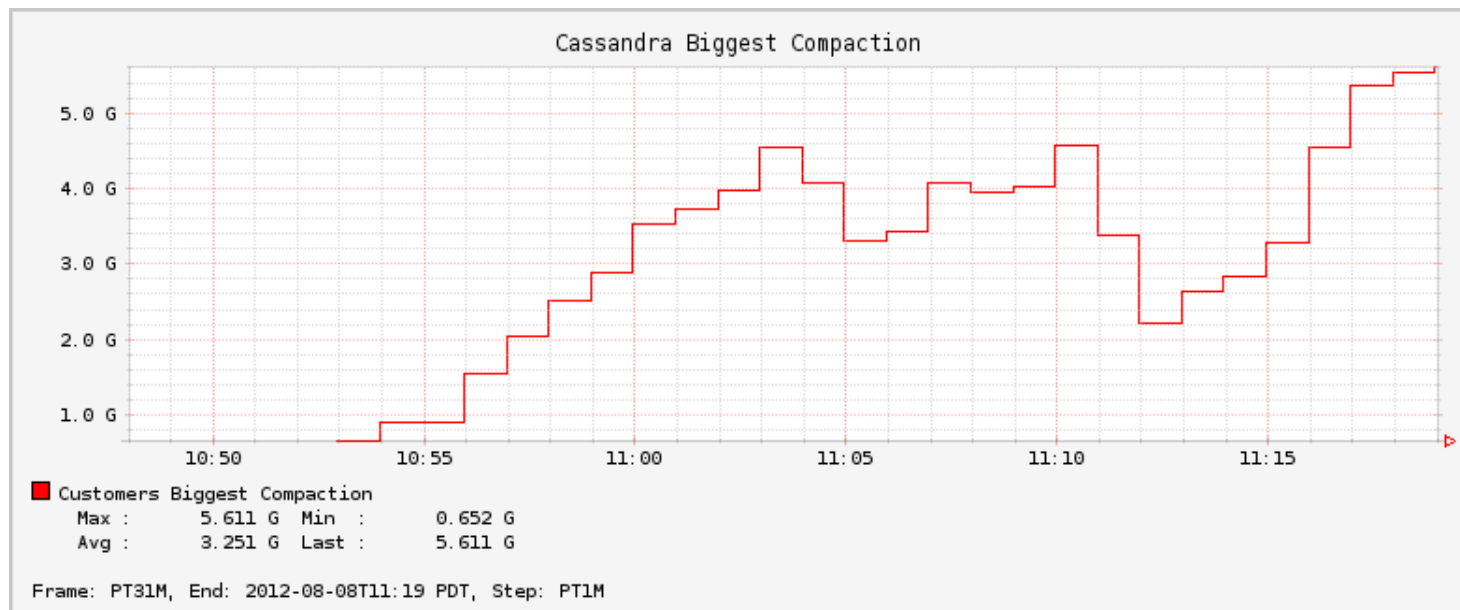
The future
hi1.4xlarge



The past
m2.4xlarge

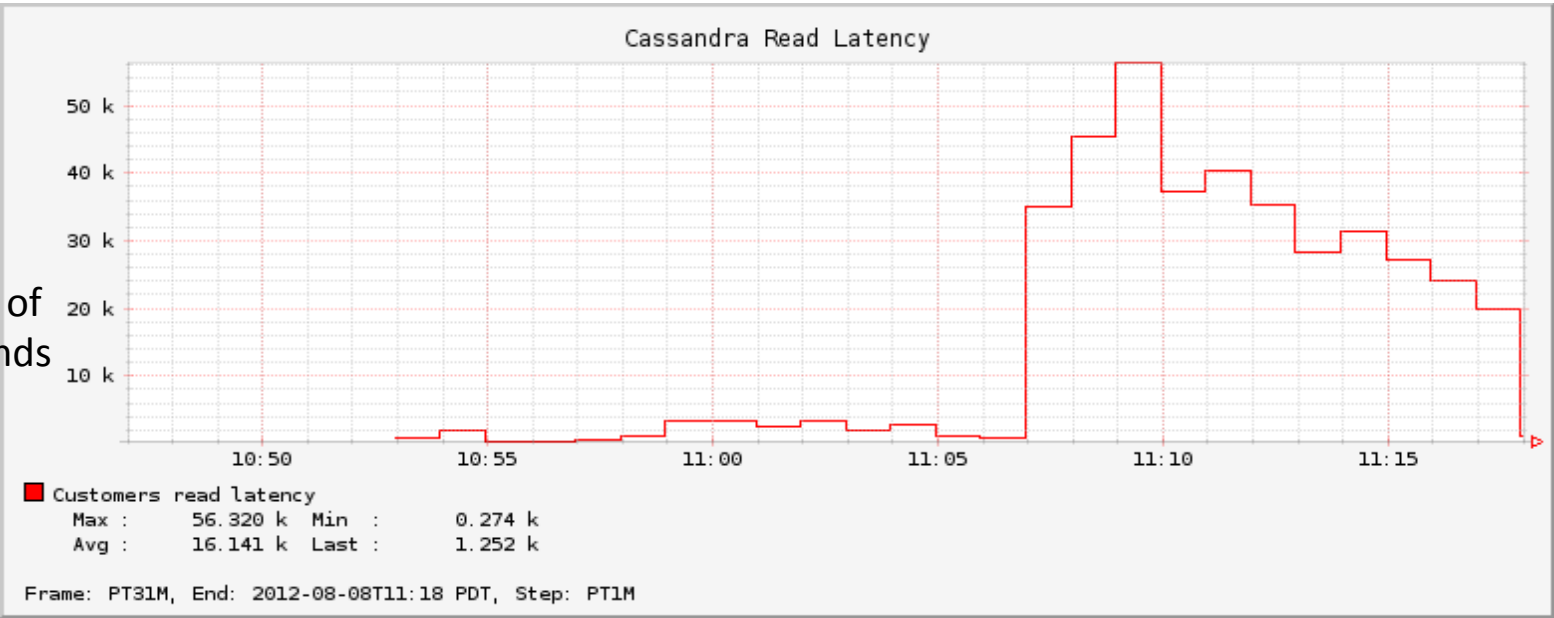


The future
hi1.4xlarge

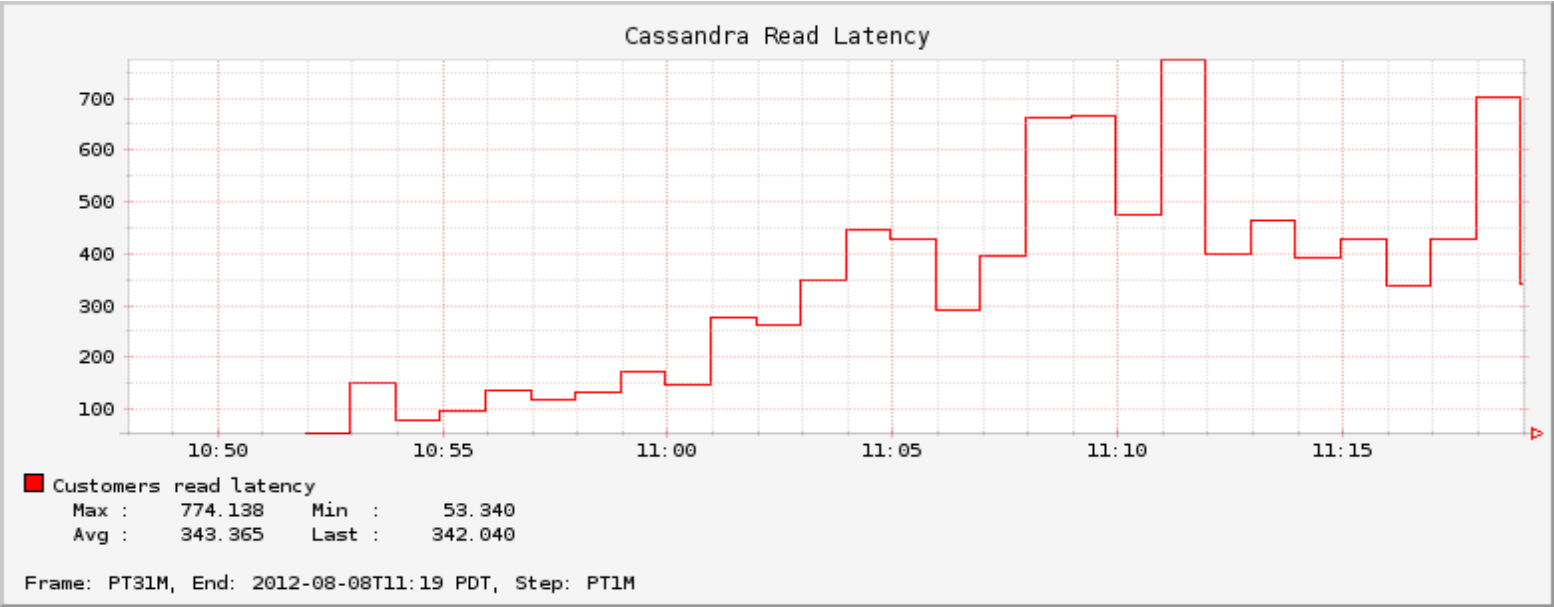


The past
m2.4xlarge

Thousands of
Microseconds

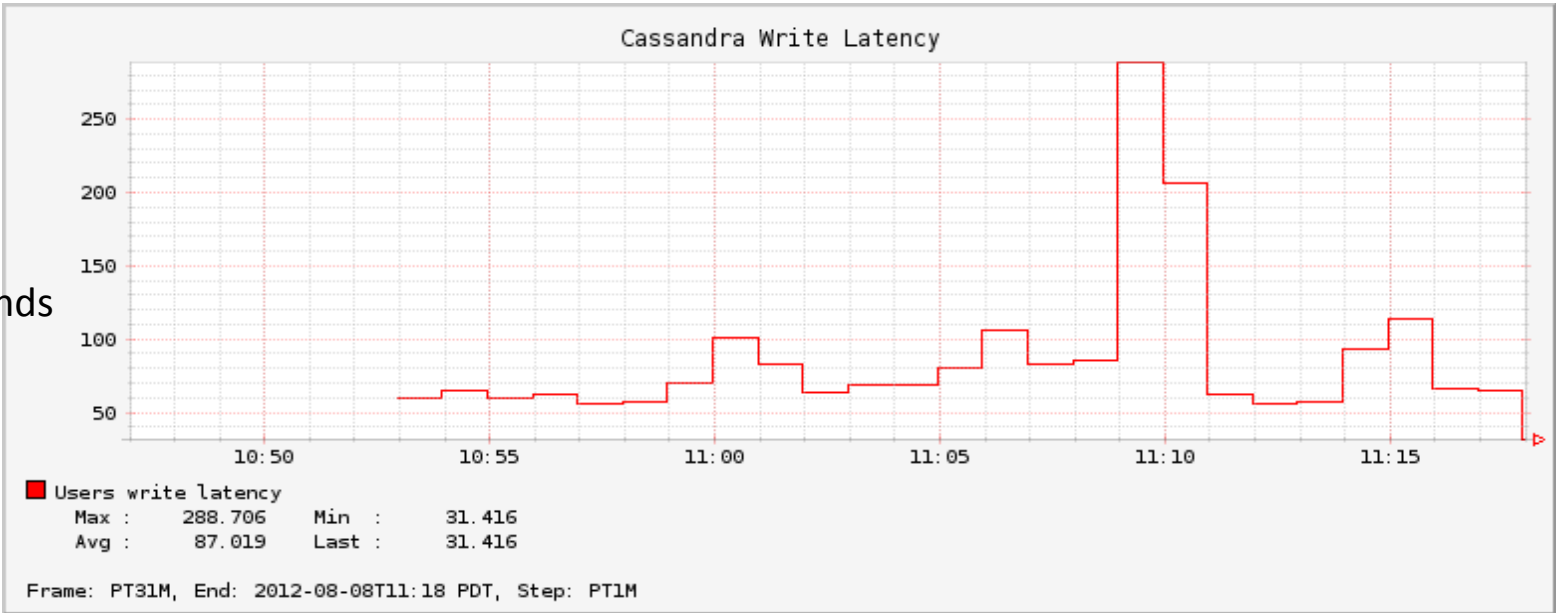


The future
hi1.4xlarge

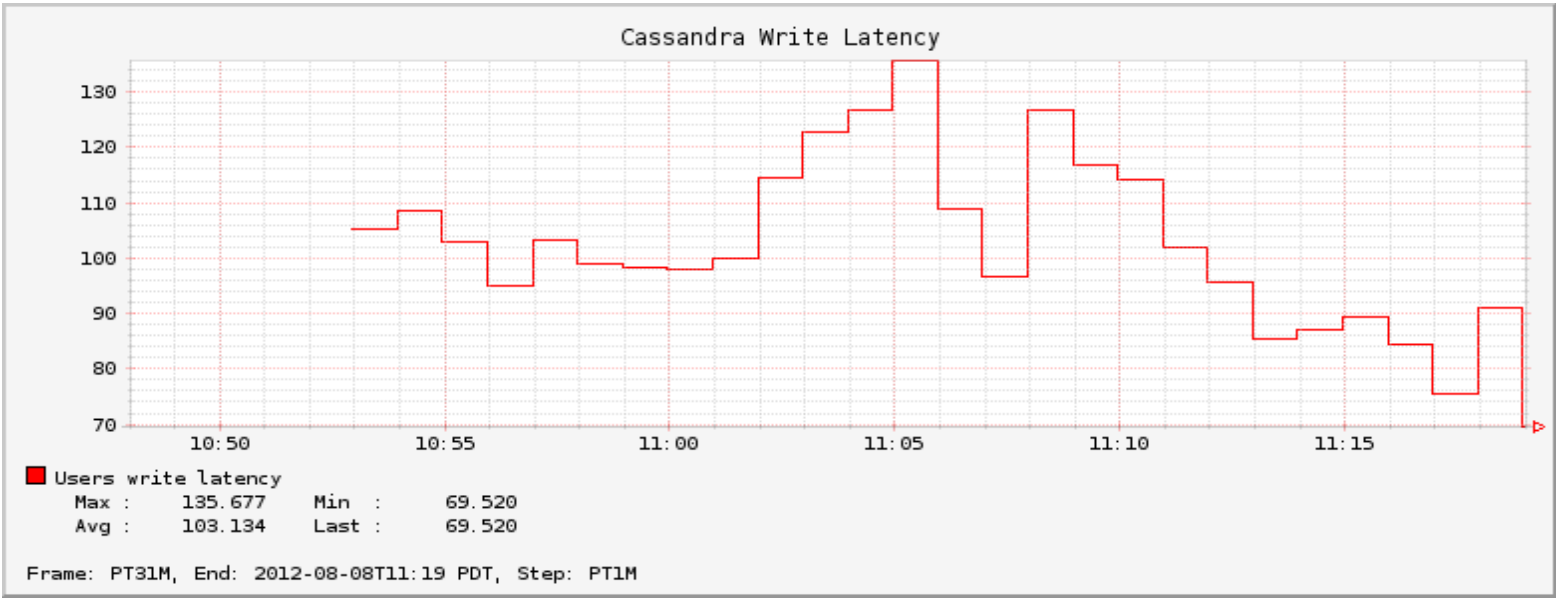


The past
m2.4xlarge

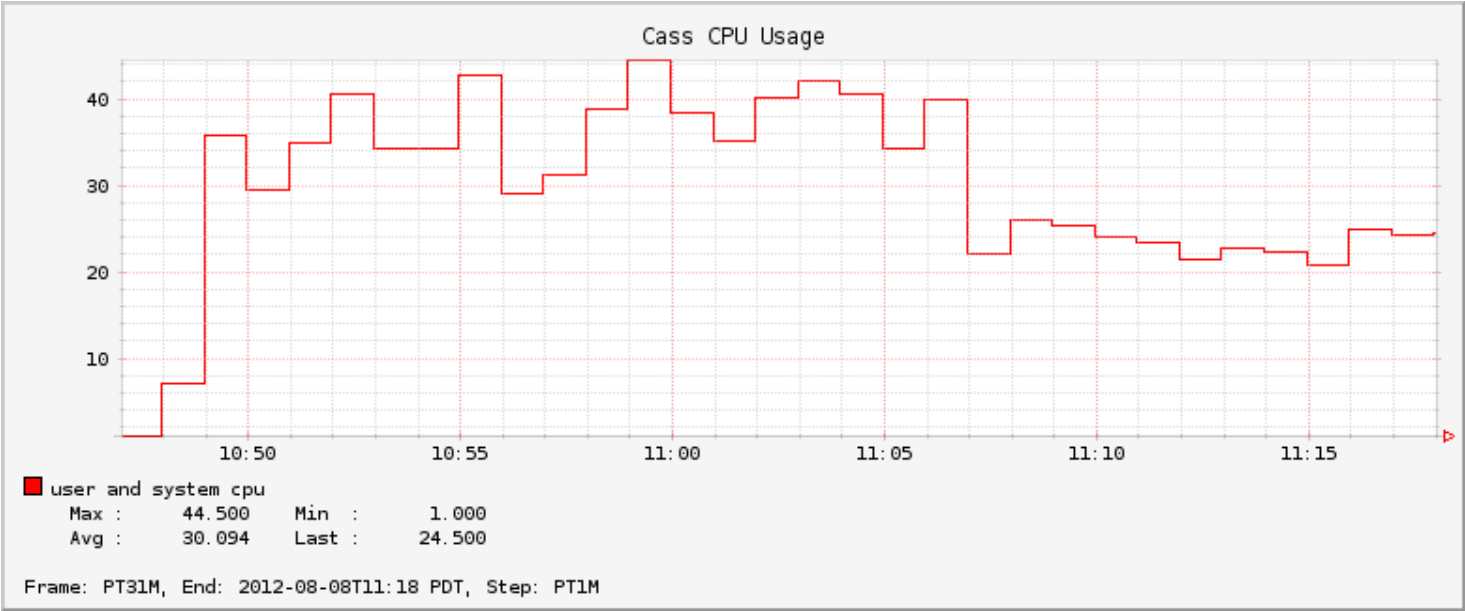
Microseconds



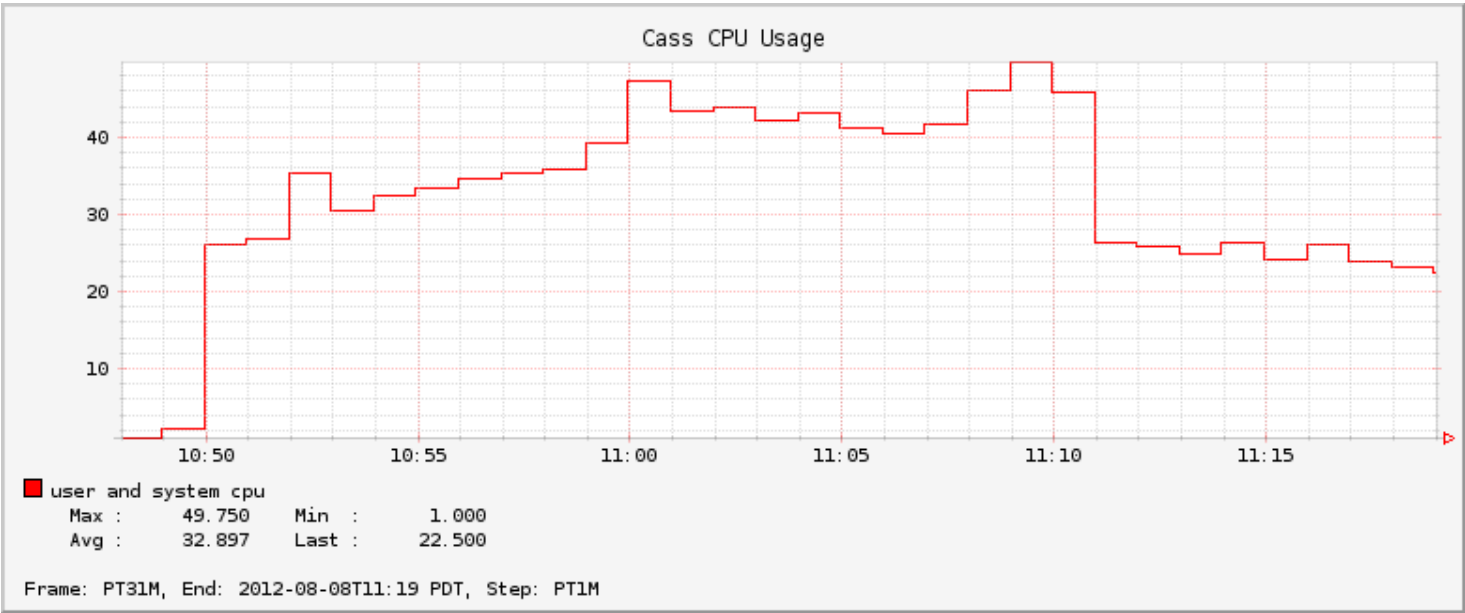
The future
hi1.4xlarge



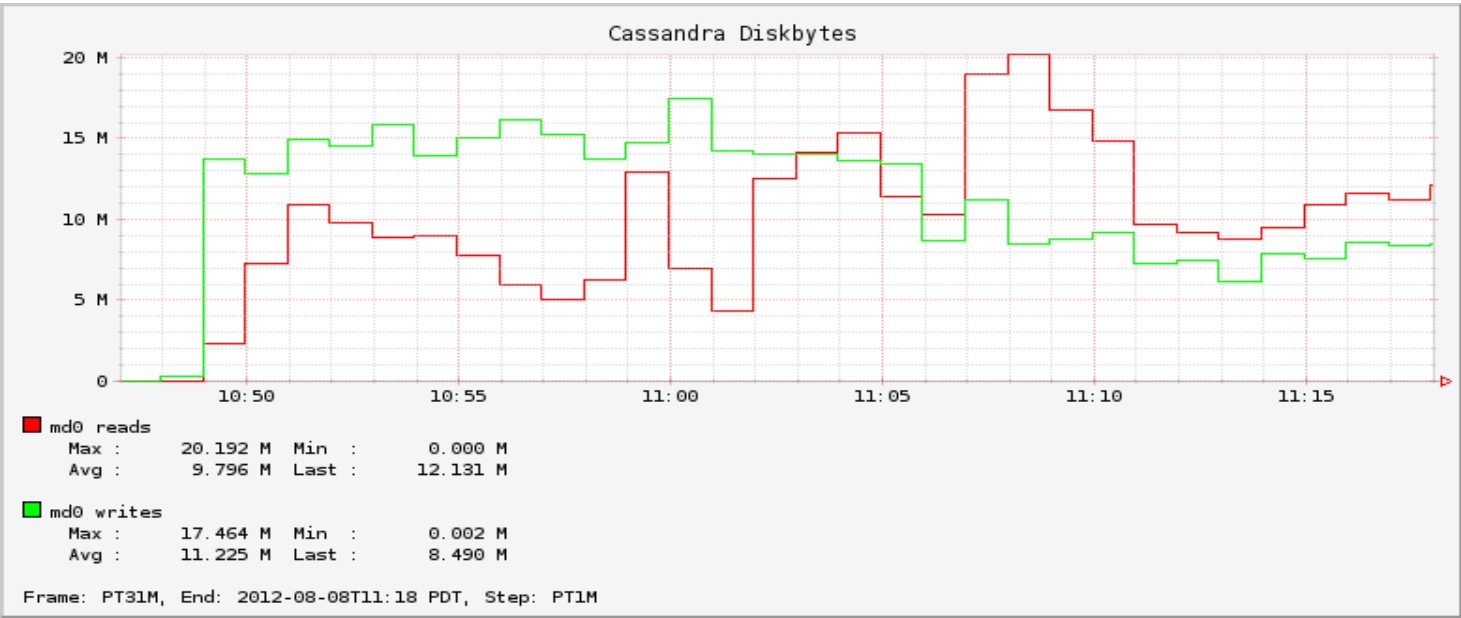
The past
m2.4xlarge



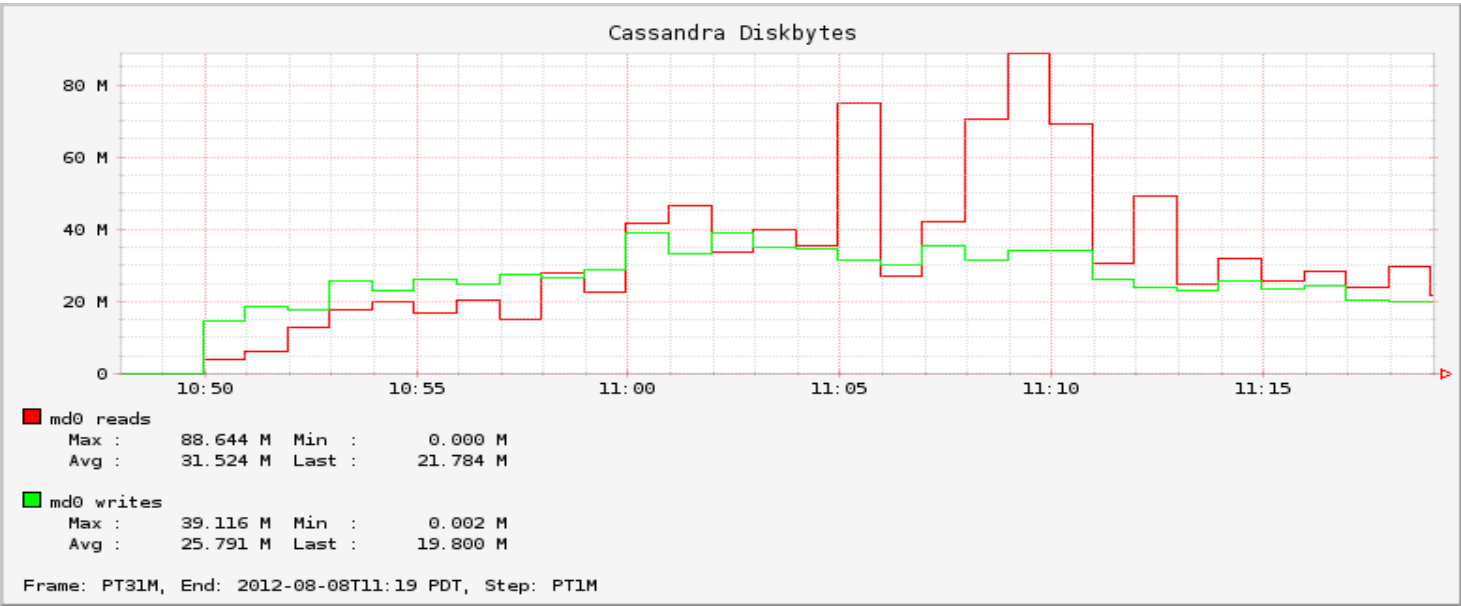
The future
hi1.4xlarge



The past
m2.4xlarge



The future
hi1.4xlarge



Next Steps

- Migrate Production Cassandra to SSD
 - Several clusters done
 - ~100 SSD nodes running
- Autoscale Cassandra using Priam
 - Cassandra 1.2 Vnodes make this easier
 - Shrink Cassandra cluster every night

Skynet

A Netflix Hackday project that might just terminate the world...

(hack currently only implemented in Powerpoint – luckily)

The Plot (kinda)

- Skynet is a sentient computer that defends itself when people try to turn it off
- Connor is the guy who eventually turns it off
- Terminator is the robot sent to kill Connor

The Hacktors

- Cass_skynet is a Cassandra cluster that detects that it is being attacked and responds
- Connor_monkey kills cass_skynet nodes
- Terminator_monkey kills connor_monkey nodes



The Hacktion

- Cass_skynet stores a history of its world and action scripts that trigger from what it sees
- Action response to losing a node
 - Auto-replace node and grow cluster size by one
- Action response to losing more nodes
 - Replicate cluster into a new zone or region
- Action response to seeing a Connor_monkey
 - Startup a Terminator_monkey in that zone

Implementation (plan)

- Priam
 - Autoreplace missing nodes
 - Grow cass_skynet cluster
 - Increase replication to new zone (new code)
 - Increase replication to new region (new code)
- Cassandra Keyspaces
 - Actions – scripts to be run
 - Memory – record event log of everything seen
- Cron job once a minute
 - Extract actions from Cassandra and execute
- Chaos Monkey configuration
 - Terminator_monkey: pick a zone, kill any connor_monkey
 - Connor_monkey: kill any cass_skynet or terminator_monkey

Simulation



Takeaway

Netflix has built and deployed a scalable global platform based on Cassandra and AWS.

Key components of the Netflix PaaS are being released as Open Source projects so you can build your own custom PaaS.

SSD's in the cloud are awesome....

<http://github.com/Netflix>

<http://techblog.netflix.com>

<http://slideshare.net/Netflix>

<http://www.linkedin.com/in/adriancockcroft>

@adrianco #netflixcloud #cassandra12