Replacing Datacenter Oracle with Global Apache Cassandra on AWS

July 11, 2011

Adrian Cockcroft

@adrianco #netflixcloud

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

Netflix Inc.

With more than 23 million subscribers in the United States and Canada, Netflix, Inc. is the world's leading Internet subscription service for enjoying movies and TV shows.

<u>International Expansion</u>

We plan to expand into an additional market in the second half of 2011… If the second market meets our expectations… we will continue to invest and expand aggressively in 2012.

Source: http://ir.netflix.com

NETFLIX

Building a Global Netflix Service

Netflix Cloud Migration
Data Migration to Cassandra
Highly Available and Globally Distributed Data
Backups and Archives in the Cloud
Monitoring Cassandra
Contributions and Organization

Why Use Public Cloud?

Frictionless Deployment (JFDI)

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

NETFLIX

Better Business Agility

Netflix could not build new datacenters fast enough

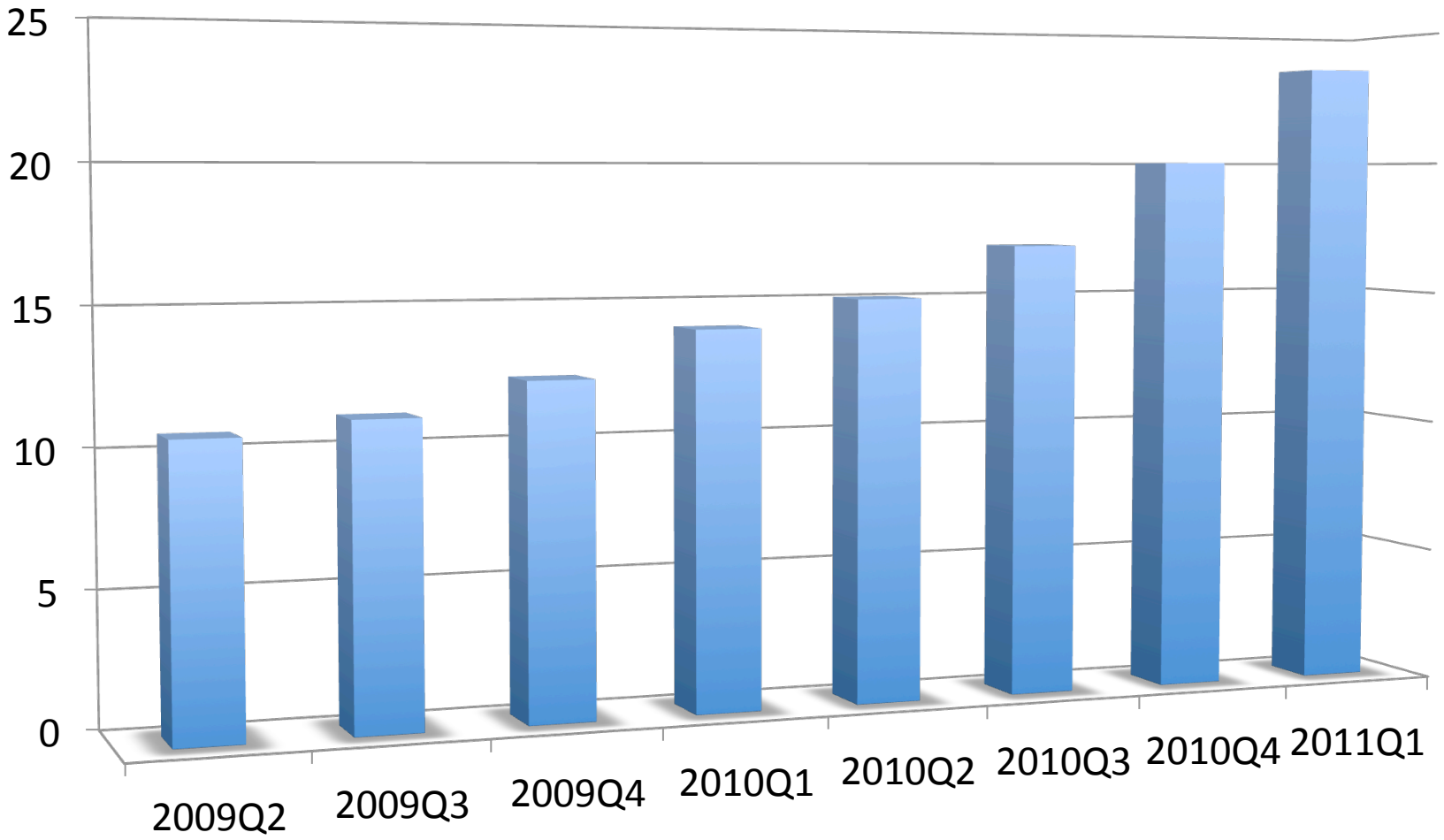
Capacity growth is accelerating, unpredictable

Product launch spikes - iPhone, Wii, PS3, XBox

NETFLIX

23 Million Customers

2011-Q1 year/year customers +69%

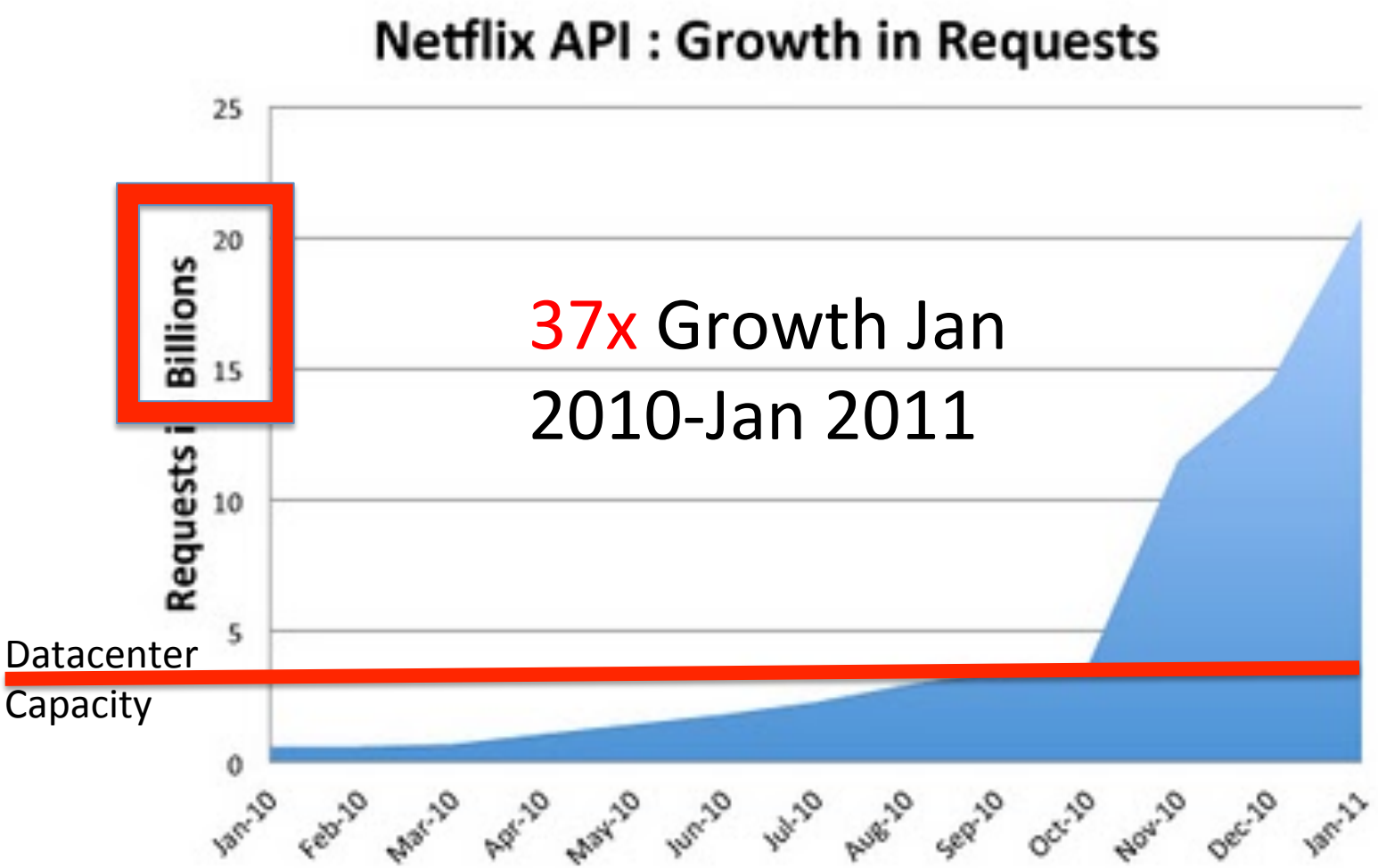


Source: http://ir.netflix.com

NETFLIX

Out-Growing Data Center

http://techblog.netflix.com/2011/02/redesigning-netflix-api.html



NETFLIX

Netflix.com is now ~100% Cloud

Account sign-up is currently being moved to cloud

All international product is cloud based

USA specific logistics remains in the Datacenter

Netflix Choice was AWS with our own platform and tools

Unique platform requirements and extreme agility and flexibility

Leverage AWS Scale
“the biggest public cloud”

AWS investment in features and automation

Use AWS zones and regions for high availability, scalability and global deployment

amazon web services™

NETFLIX

We want to use clouds, we don't have time to build them

Public cloud for agility and scale

AWS because they are big enough to allocate thousands of instances per hour when we need to

Netflix Deployed on AWS

Content	Logs	Play	WWW	API
Video Masters	S3	DRM	Sign-Up	Metadata
EC2	EMR Hadoop	CDN routing	Search	Device Config
S3	Hive	Bookmarks	Movie Choosing	TV Movie Choosing
CDN	Business Intelligence	Logging	Ratings	Mobile iPhone

Port to Cloud Architecture

Short term investment, long term payback!

Pay down technical debt

Robust patterns

Transition

- The Goals
  - Faster, Scalable, Available and Productive
- Anti-patterns and Cloud Architecture
  - The things we wanted to change and why
- Data Migration
  - Minimizing datacenter dependencies

Datacenter Anti-Patterns

What do we currently do in the datacenter that prevents us from meeting our goals?

Old Datacenter vs. New Cloud Arch

Central SQL Database	Distributed Key/Value NoSQL
Sticky In-Memory Session	Shared Memcached Session
Chatty Protocols	Latency Tolerant Protocols
Tangled Service Interfaces	Layered Service Interfaces
Instrumented Code	Instrumented Service Patterns
Fat Complex Objects	Lightweight Serializable Objects
Components as Jar Files	Components as Services

NETFLIX

The Central SQL Database

- Datacenter has central Oracle databases
  - Everything in one place is convenient until it fails
  - Customers, movies, history, configuration
- Schema changes require downtime

Anti-pattern impacts scalability, availability

NETFLIX

The Distributed Key-Value Store

- Cloud has many key-value data stores
  - More complex to keep track of, do backups etc.
  - Each store is much simpler to administer
  - Joins take place in java code

- No schema to change, no scheduled downtime
- Latency for typical queries
  - Memcached is dominated by network latency <1ms
  - Cassandra replication takes a few milliseconds
  - Oracle for simple queries is a few milliseconds
  - SimpleDB replication and REST auth overheads >10ms

NETFLIX

Data Migration to Cassandra

Transitional Steps

- Bidirectional Replication
  - Oracle to SimpleDB
  - Queued reverse path using SQS
  - Backups remain in Datacenter via Oracle
- New Cloud-Only Data Sources
  - Cassandra based
  - No replication to Datacenter
  - Backups performed in the cloud

NETFLIX

API

AWS EC2

Front End Load Balancer

Discovery Service

API Proxy

API etc.

Load Balancer

Component Services

API

SQS

Cassandra

EC2 Internal Disks

memcached

memcached

Replication

Oracle

S3

SimpleDB

Netflix Data Center

NETFLIX

Cutting the Umbilical

- Transition Oracle Data Sources to Cassandra
  - Offload Datacenter Oracle hardware
  - Free up capacity for growth of remaining services
- Transition SimpleDB+Memcached to Cassandra
  - Primary data sources that need backup
  - Keep simple use cases like configuration service
- New challenges
  - Backup, restore, archive, business continuity
  - Business Intelligence integration

API

AWS EC2

Front End Load Balancer

Discovery Service

API Proxy

Load Balancer

Component Services

API

memcached

Cassandra

EC2 Internal Disks

Backup

S3

SimpleDB

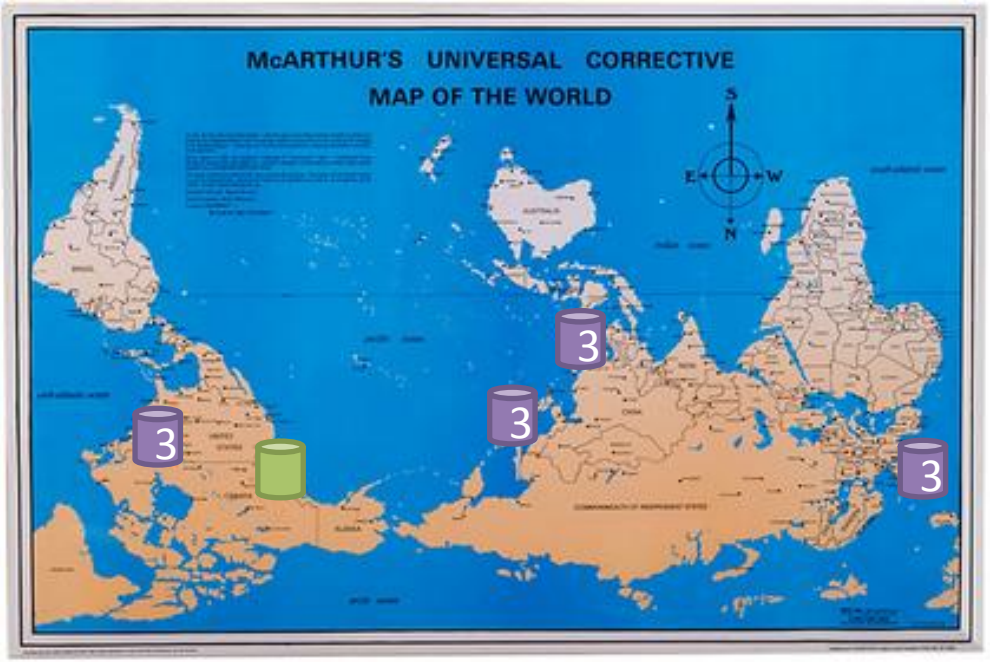
NETFLIX

High Availability

- Cassandra stores 3 local copies, 1 per zone
  - Synchronous access, durable, highly available
  - Read/Write One fastest, least consistent - ~1ms
  - Read/Write Quorum 2 of 3, consistent - ~3ms
- AWS Availability Zones
  - Separate buildings
  - Separate power etc.
  - Close together

Remote Copies

- Cassandra duplicates across AWS regions
  - Asynchronous write, replicates at destination
  - Doesn't directly affect local read/write latency
- Global Coverage
  - Business agility
  - Follow AWS…
- Local Access
  - Better latency
  - Fault Isolation



NETFLIX

Cassandra Backup

- Full Backup
  - Cron on each node
  - Snapshot -> tar.gz -> S3
- Incremental
  - SSTable write triggers copy to S3
- Continuous
  - Scrape commit log
  - Write to EBS every 30s

NETFLIX

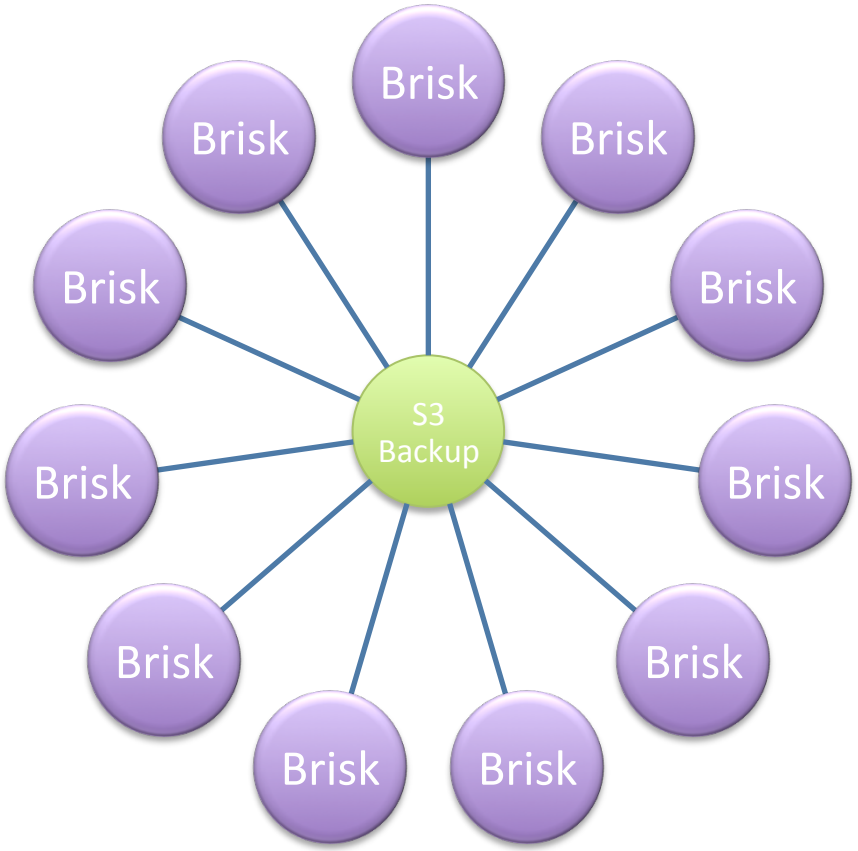
Cassandra Restore

- Full Restore
  - Replace previous data
- New Ring from Backup
  - New name old data
  - One line command!

NETFLIX

Cassandra Data Extraction

- Business Intelligence
  - Re-normalize data using Hadoop job
- Daily Extraction
  - Create Brisk ring
  - Extract backup
  - Run Hadoop job
  - Remove Brisk ring
  - Under 1hr…



NETFLIX

Cassandra Online BI

- Intra-Day Extraction
  - Use split Brisk ring
  - Size each separately
  - Hourly Hadoop job

NETFLIX

Cassandra Archive

Appropriate level of paranoia needed…

- Archive could be un-readable
  - Base on restored S3 backup and BI extracted data
- Archive could be stolen
  - Encrypt archive
- AWS East Region could have a problem
  - Copy data to AWS West
- Production AWS Account could have an issue
  - Separate Archive account with no-delete S3 ACL
- AWS S3 could have a global problem
  - Create an extra copy on a different cloud vendor

Tools and Automation

- Developer and Build Tools
  - Jira, Perforce, Eclipse, Jenkins, Ivy, Artifactory
  - Builds, creates .war file, .rpm, bakes AMI and launches
- Custom Netflix Application Console
  - AWS Features at Enterprise Scale (hide the AWS security keys!)
  - Auto Scaler Group is unit of deployment to production
- Open Source + Support
  - Apache, Tomcat, Cassandra, Hadoop, OpenJDK, CentOS
  - Datastax support for Cassandra, AWS support for Hadoop via EMR
- Monitoring Tools
  - Datastax Opscenter for monitoring Cassandra
  - AppDynamics – Developer focus for cloud http://appdynamics.com

NETFLIX

Developer Migration

- Detailed SQL to NoSQL Transition Advice
  - Sid Anand - QConSF Nov 5th – Netflix' Transition to High Availability Storage Systems
  - Blog - http://practicalcloudcomputing.com/
  - Download Paper PDF - http://bit.ly/bhOTLu
- Mark Atwood, "Guide to NoSQL, redux"
  - YouTube http://youtu.be/zAbFRiyT3LU

NETFLIX

Cloud Operations

Cassandra Use Cases
Model Driven Architecture
Capacity Planning & Monitoring
Chaos Monkey

Cassandra Use Cases

- Key by Customer
  - Several separate Cassandra rings, read-intensive
  - Sized to fit in memory using m2.4xl Instances
- Key by Customer:Movie – e.g. Viewing History
  - Growing fast, write intensive – m1.xl instances
  - Sized to hold hot data in memory only
- Large scale data logging – lots of writes
  - Column data expires after time period
  - Working on using distributed counters…

Model Driven Architecture

- Datacenter Practices
  - Lots of unique hand-tweaked systems
  - Hard to enforce patterns

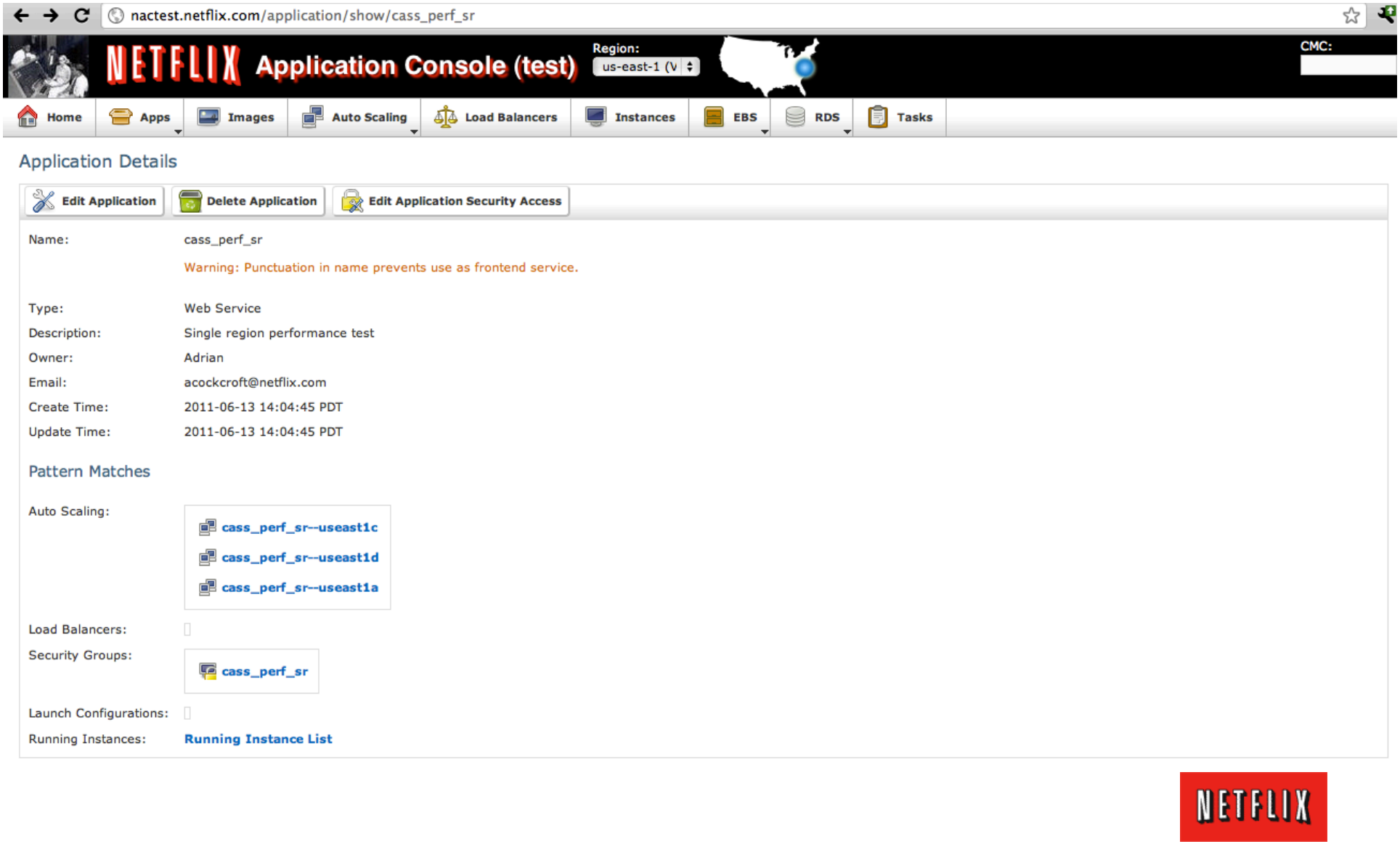
- Model Driven Cloud Architecture
  - Perforce/Ivy/Jenkins based builds for *everything*
  - Every production instance is a pre-baked AMI
  - Every application is managed by an Autoscaler

Every change is a new AMI

Netflix Platform Cassandra AMI

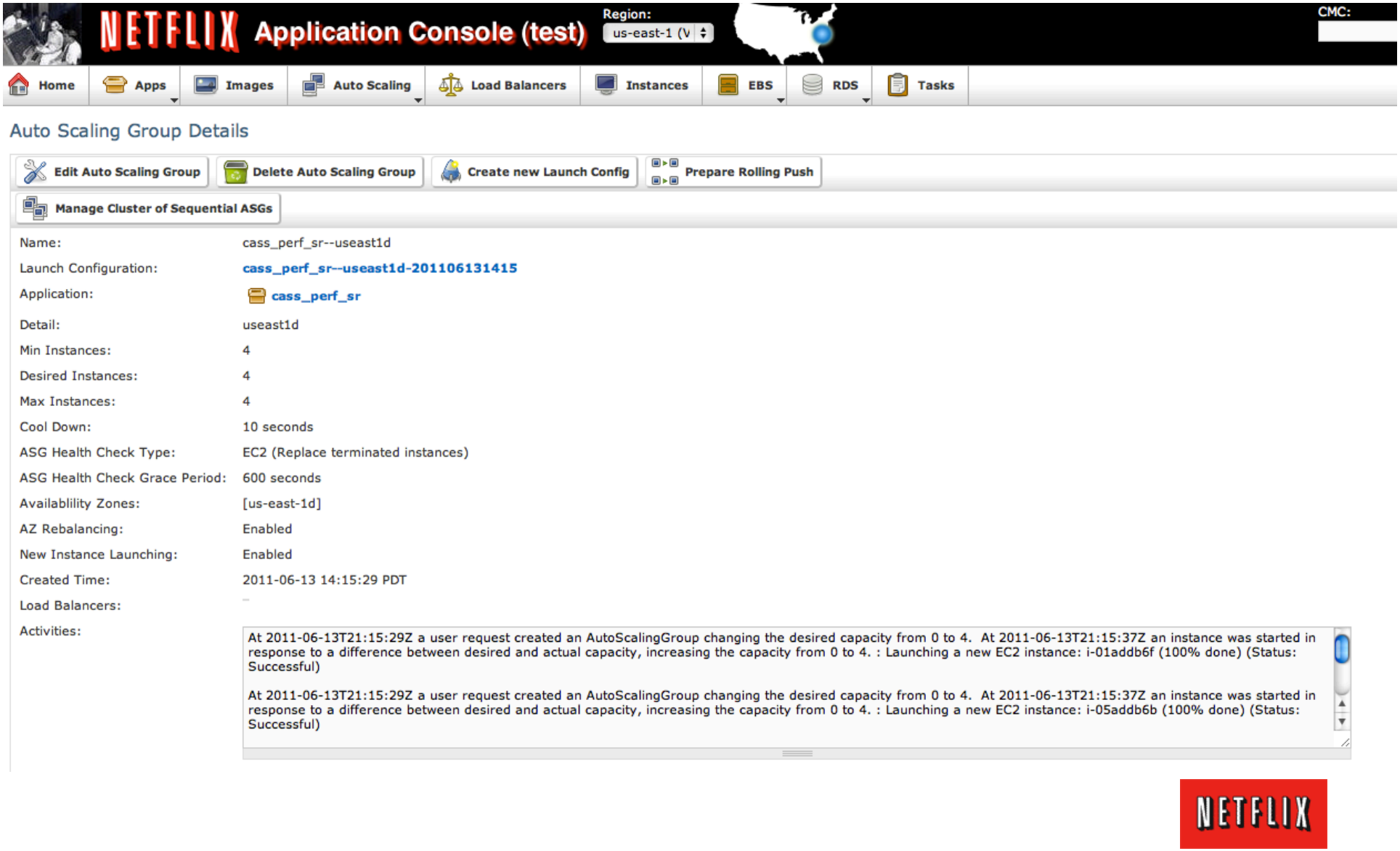
- Tomcat server
  - Always running, registers with platform
  - Manages Cassandra state, tokens, backups
- SimpleDB configuration
  - Stores token slots and options
  - Avoids circular “bootstrap problems”
- Removed Root Disk Dependency on EBS
  - Use S3 backed AMI for stateful services
  - Normally use EBS backed AMI for fast provisioning

Netflix App Console



NETFLIX

Auto Scale Group Configuration



NETFLIX

Chaos Monkey

- Make sure systems are resilient
  - Allow any instance to fail without customer impact
- Chaos Monkey hours
  - Monday-Thursday 9am-3pm random instance kill
- Application configuration option
  - Apps now have to opt-out from Chaos Monkey
- Computers (Datacenter or AWS) randomly die
  - Fact of life, but too infrequent to test resiliency

NETFLIX

Capacity Planning & Monitoring

Capacity Planning in Clouds

(a few things have changed…)

- ~~Capacity is expensive~~
- ~~Capacity takes time to buy and provision~~
- ~~Capacity only increases, can't be shrunk easily~~
- ~~Capacity comes in big chunks, paid up front~~
- ~~Planning errors can cause big problems~~
- ~~Systems are clearly defined assets~~
- ~~Systems can be instrumented in detail~~
- ~~Depreciate assets over 3 years~~ (reservations!)

NETFLIX

Data Sources

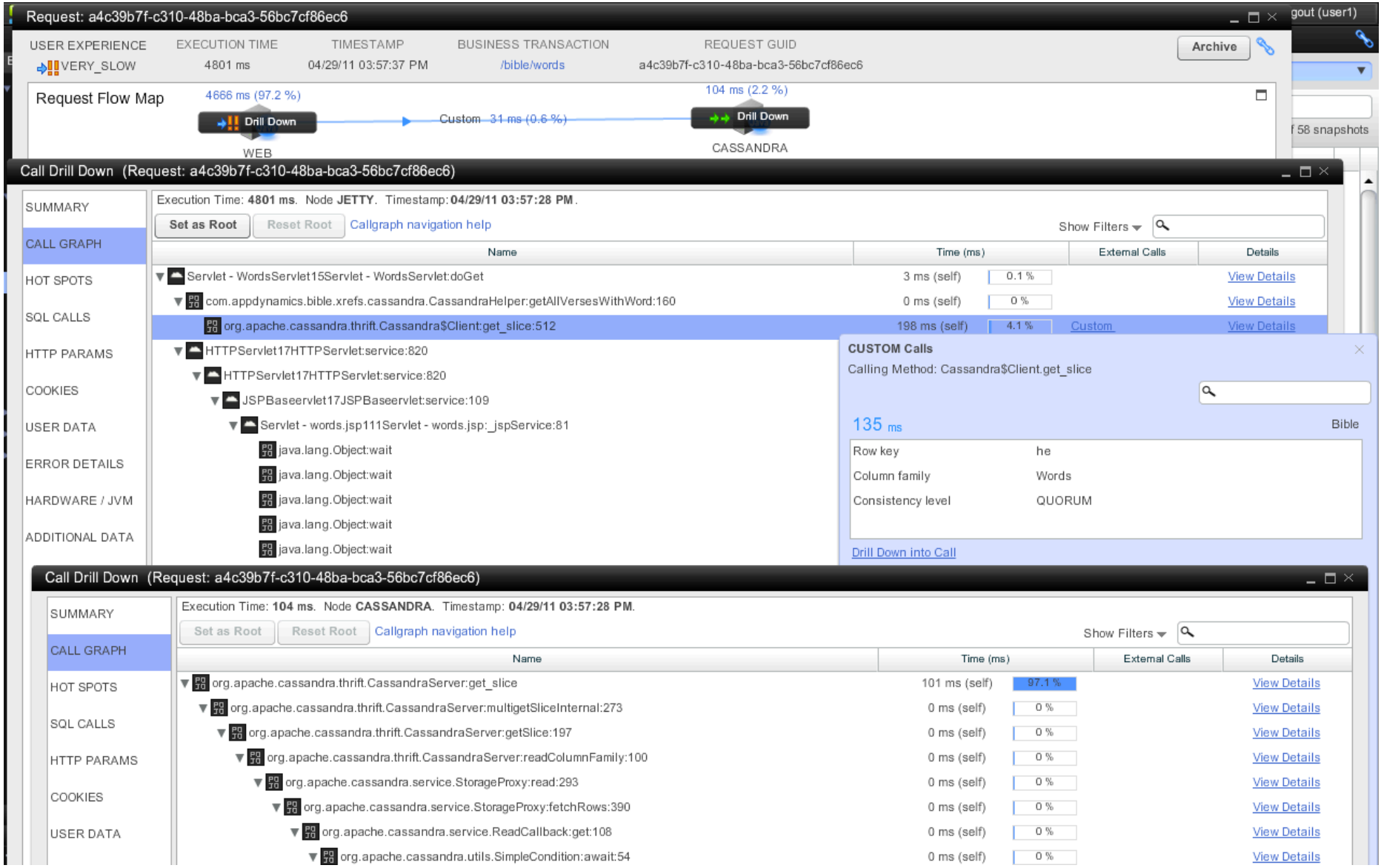
External Testing	•External URL availability and latency alerts and reports – Keynote •Stress testing - SOASTA
Request Trace Logging	•Netflix REST calls – Chukwa to DataOven with GUID transaction identifier •Generic HTTP – AppDynamics service tier aggregation, end to end tracking
Application logging	•Tracers and counters – log4j, tracer central, Chukwa to DataOven •Trackid and Audit/Debug logging – DataOven, Appdynamics GUID cross reference
JMX Metrics	•Application specific real time – Datastax Opscenter, Appdynamics •Service and SLA percentiles – Appdynamics, Epic logged to DataOven
Tomcat and Apache logs	•Stdout logs – S3 – DataOven •Standard format Access and Error logs – S3 – DataOven
JVM	•Garbage Collection – Appdynamics •Memory usage, call stacks, resource/call - AppDynamics
Linux	•system CPU/Net/RAM/Disk metrics – AppDynamics •SNMP metrics – Epic, Network flows – boundary.com
AWS	•Load balancer traffic – Amazon Cloudwatch, SimpleDB usage stats •System configuration - CPU count/speed and RAM size, overall usage - AWS

AppDynamics

How to look deep inside your cloud applications

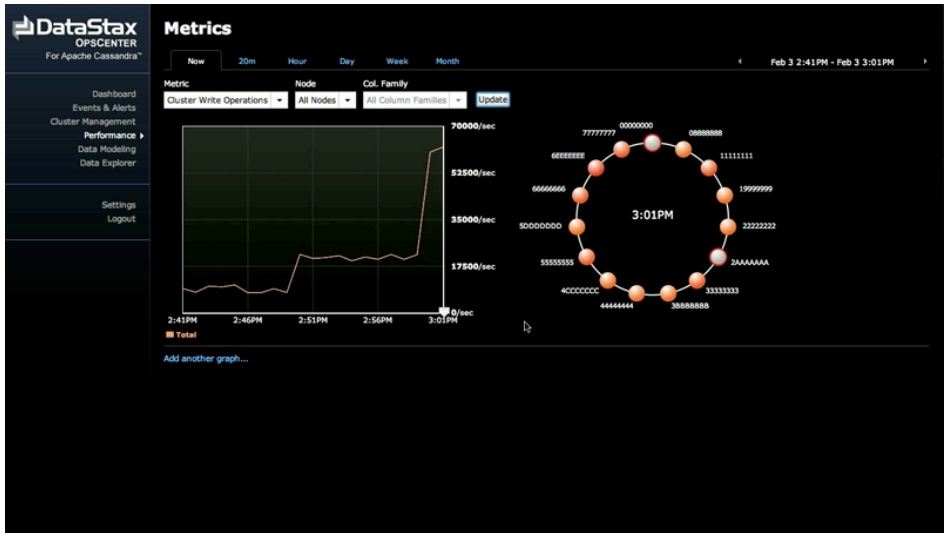
- Automatic Monitoring
  - Base AMI bakes in all monitoring tools
  - Outbound calls only – no discovery/polling issues
  - Inactive instances removed after a few days
- Incident Alarms (deviation from baseline)
  - Business Transaction latency and error rate
  - Alarm thresholds discover their own baseline
  - Email contains URL to Incident Workbench UI

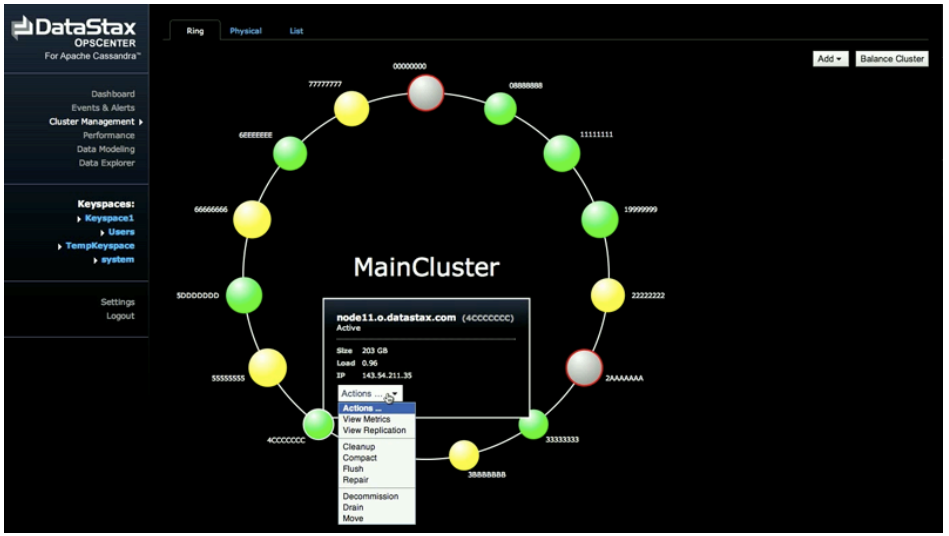
AppDynamics Monitoring of Cassandra – Automatic Discovery

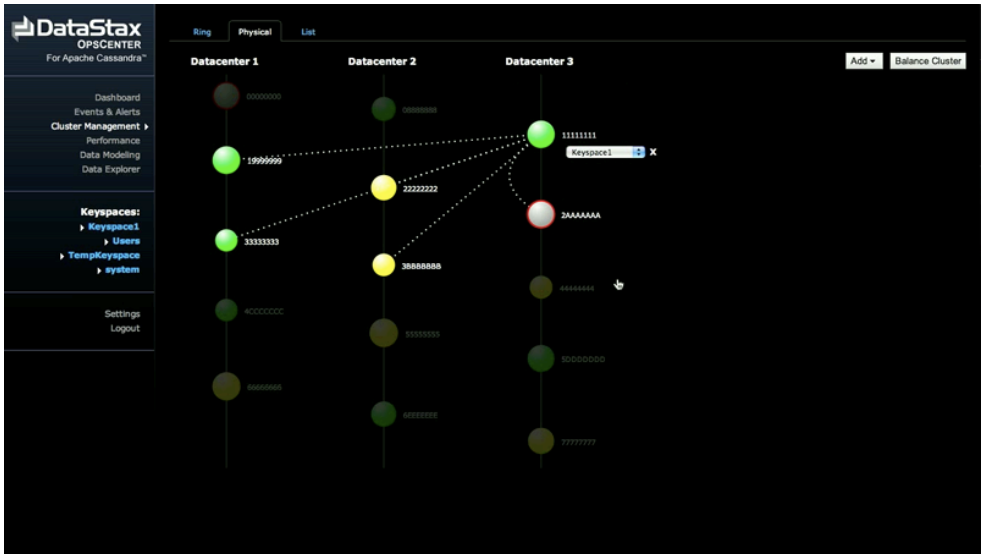


NETFLIX

DataStax OpsCenter





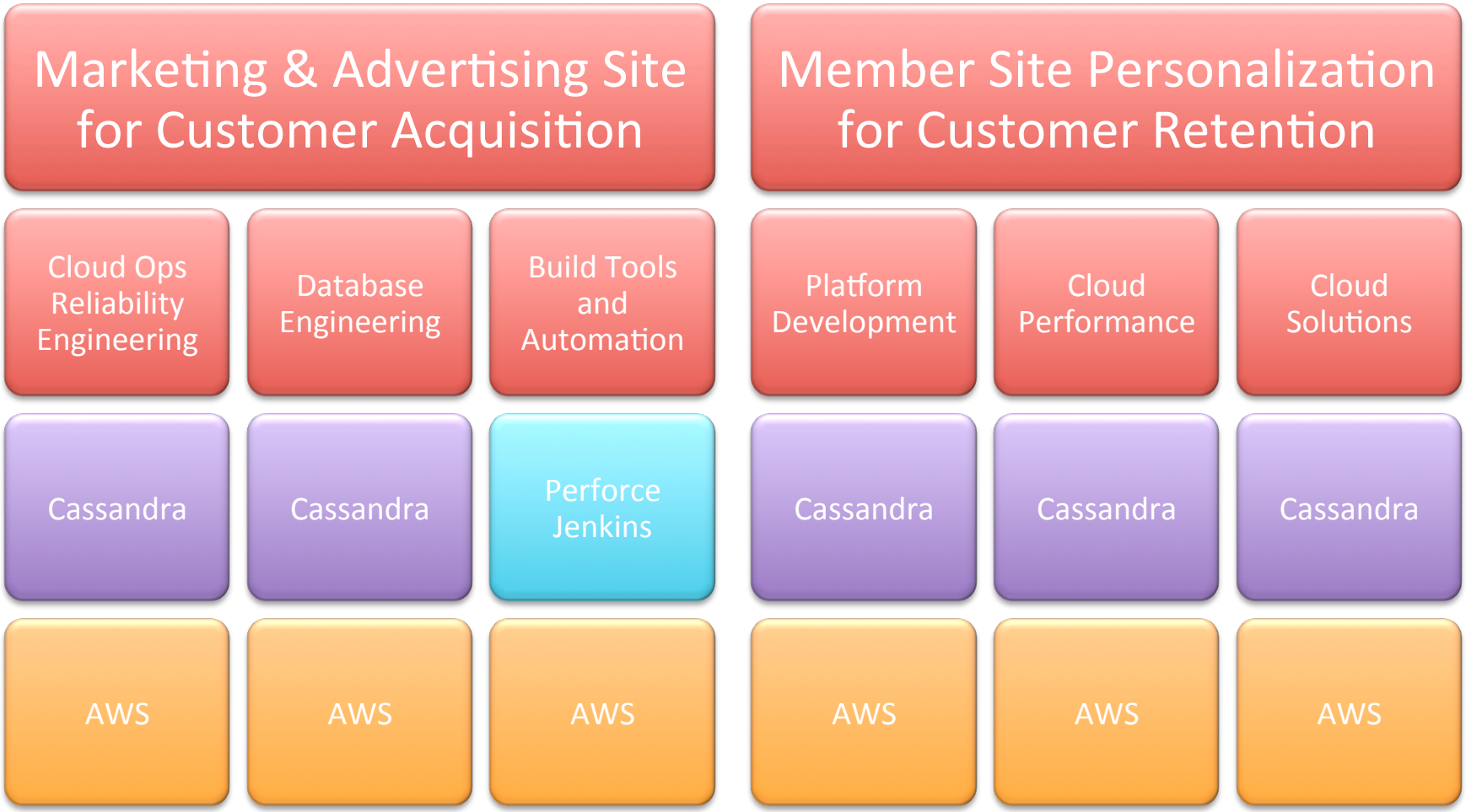


NETFLIX

Netflix Contributions to Cassandra

- Cassandra as a mutable toolkit
  - Cassandra is in Java, pluggable, well structured
  - Netflix has a building full of Java engineers….
- Actual Contributions delivered in 0.8
  - First prototype of off-heap row cache (Vijay)
  - Incremental backup SSTable write callback
- Work In Progress
  - AWS integration and backup using Tomcat helper
  - Total re-write of Hector Java client library (Eran)

Netflix “NoOps” Organization



NETFLIX

Takeaway

Netflix is using Cassandra on AWS as a key infrastructure component of its globally distributed streaming product.

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

@adrianco #netflixcloud

Amazon Cloud Terminology Reference

See http://aws.amazon.com/ This is not a full list of Amazon Web Service features

- AWS – Amazon Web Services (common name for Amazon cloud)
- AMI – Amazon Machine Image (archived boot disk, Linux, Windows etc. plus application code)
- EC2 – Elastic Compute Cloud
  - Range of virtual machine types m1, m2, c1, cc, cg. Varying memory, CPU and disk configurations.
  - Instance – a running computer system. Ephemeral, when it is de-allocated nothing is kept.
  - Reserved Instances – pre-paid to reduce cost for long term usage
  - Availability Zone – datacenter with own power and cooling hosting cloud instances
  - Region – group of Availability Zones – US-East, US-West, EU-Eire, Asia-Singapore, Asia-Japan
- ASG – Auto Scaling Group (instances booting from the same AMI)
- S3 – Simple Storage Service (http access)
- EBS – Elastic Block Storage (network disk filesystem can be mounted on an instance)
- RDS – Relational Database Service (managed MySQL master and slaves)
- SDB – Simple Data Base (hosted http based NoSQL data store)
- SQS – Simple Queue Service (http based message queue)
- SNS – Simple Notification Service (http and email based topics and messages)
- EMR – Elastic Map Reduce (automatically managed Hadoop cluster)
- ELB – Elastic Load Balancer
- EIP – Elastic IP (stable IP address mapping assigned to instance or ELB)
- VPC – Virtual Private Cloud (extension of enterprise datacenter network into cloud)
- IAM – Identity and Access Management (fine grain role based security keys)

amazon web services™

NETFLIX