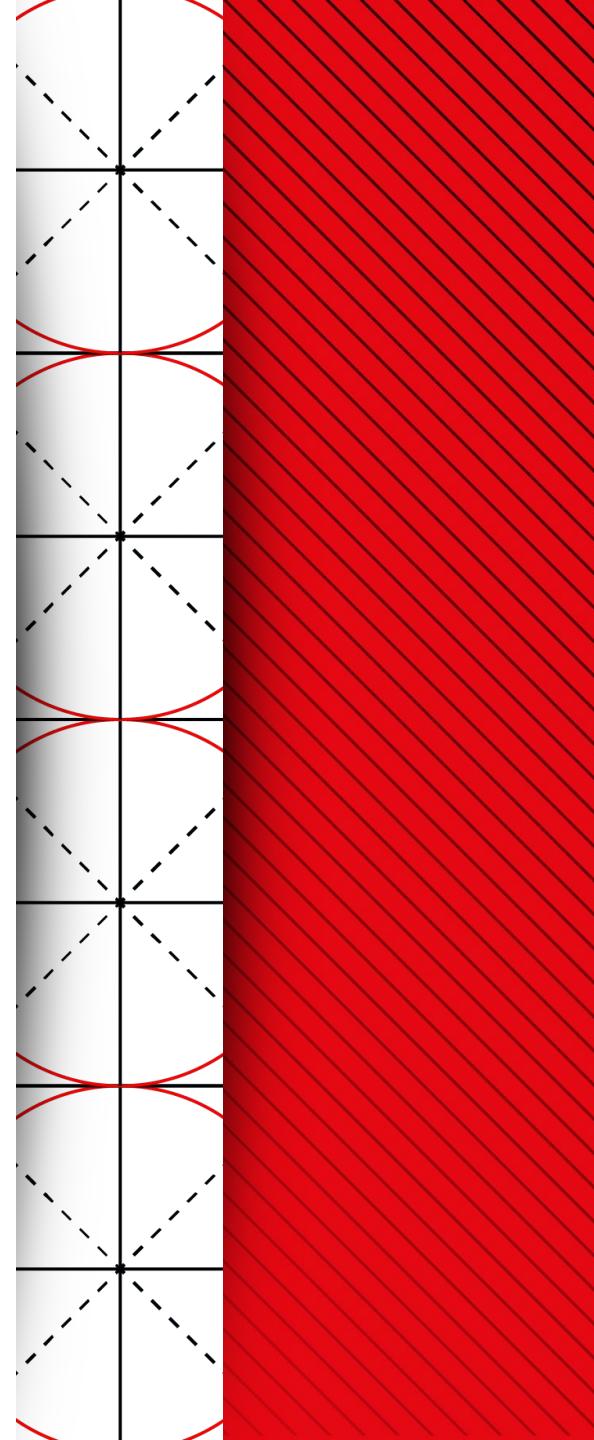


# Building and Running Cloud Native Cassandra

Vinay Chella, Joey Lynch  
Distributed Database Engineers  
Netflix  
NGCC 2019

N



# Speakers



Vinay Chella

Distributed Systems Engineer

Focusing on Apache Cassandra and Data Abstractions

Cloud Data Engineering  
Netflix



Joey Lynch

Distributed Systems Engineer

Distributed system addict and data wrangler



Cloud Data Engineering  
Netflix

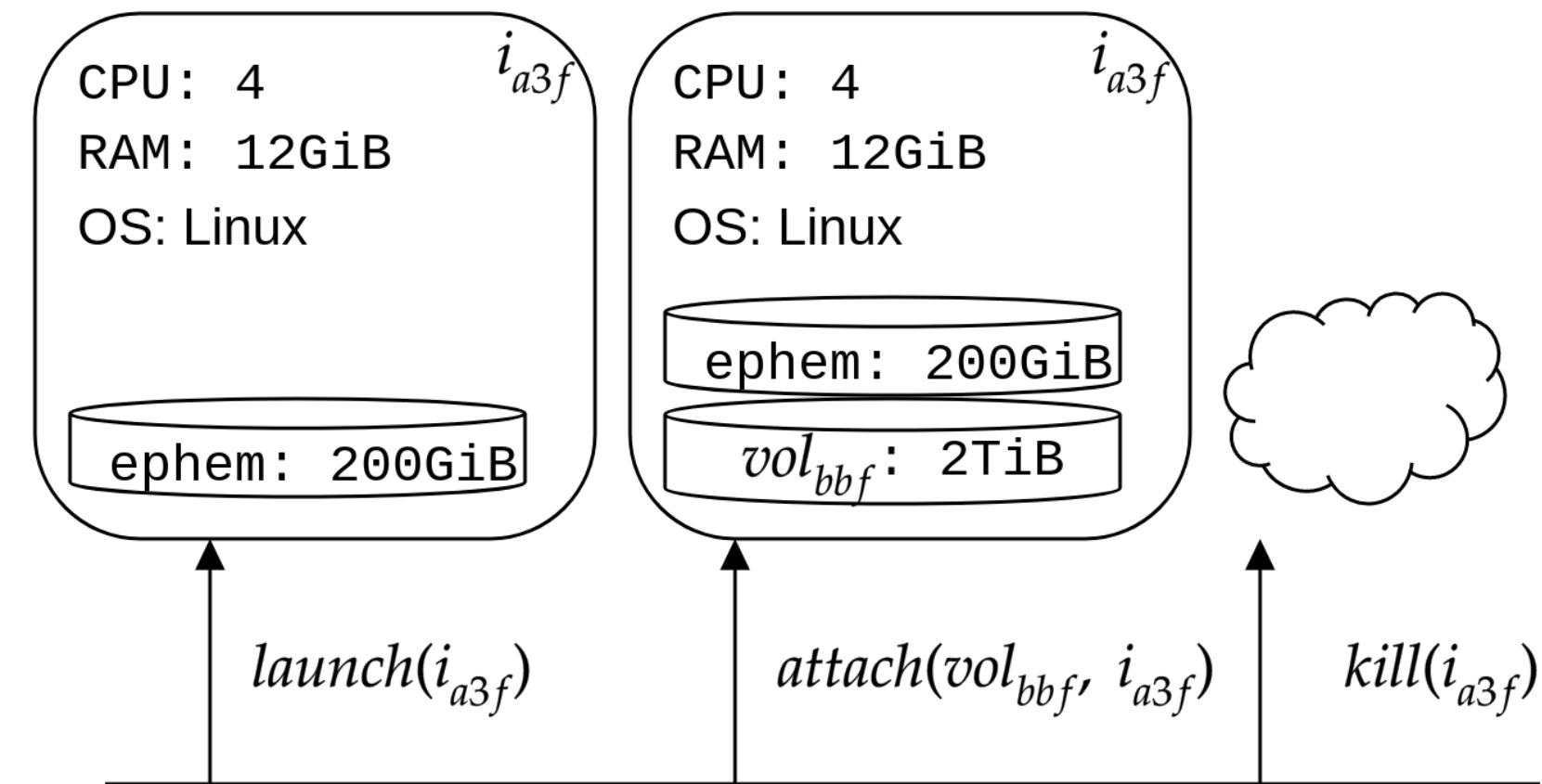
# Outline

Goals to help align the project cloud principles.

- What is Cloud Native?
- Cassandra's Rough Edges
  - ◆ Development
  - ◆ Packaging
  - ◆ Starting a Cluster
  - ◆ Running a Cluster
- Our Proposed Solutions

# **What even is “Cloud Native”?**





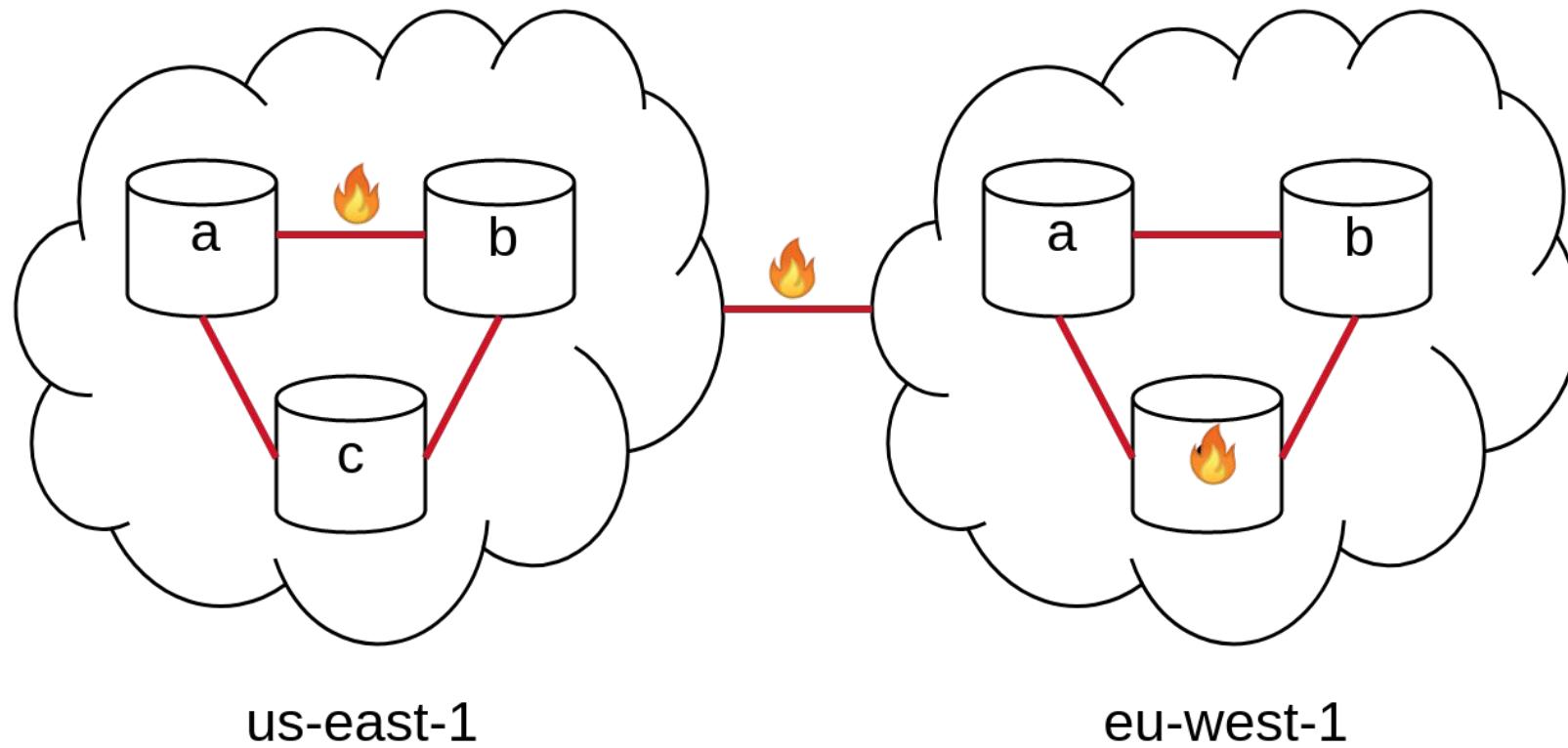
Cloud Control Plane (API)

Any hardware configuration

Any operating system

Cattle not Pets

Will die constantly

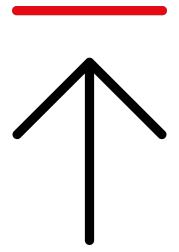


Clouds provide mappings on top of “datacenters”

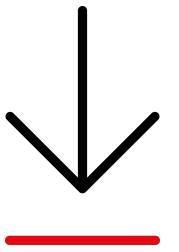
Zones will fail

Regions will fail

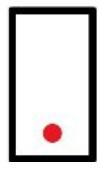
“Durable” storage is not so durable



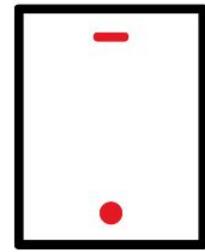
Develop



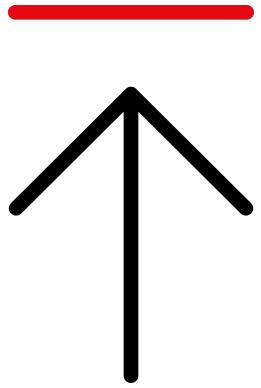
Package



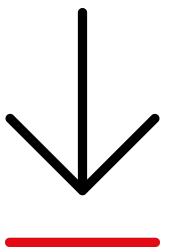
Starting



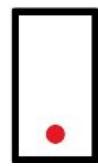
Scaling



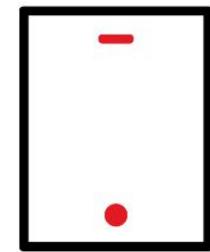
**Develop**



**Package**



**Starting**



**Scaling**

# Friction for New Contributions

docs

Discovery, building mental model

code

Understand and modify

build

Create artifacts (jar, pkg, container)

test

Confidence they haven't broken  
something

## Data Modeling

**Todo**

TODO

## Overview

**Todo**

todo

## First interactions?

- I don't know how the database works
- Where can I run v4.0?

Dynamo  
Gossip

**Todo**

todo

Failure Detection

**Todo**

todo

Token Ring/Ranges

**Todo**

todo

# How Netflix Does This

MkDocs

Home

User Guide ▾

About ▾

Search

◀ Previous

Next ➔

Git

MkDocs

Overview

Installation

Getting Started

Adding pages

Theming our documentation

Changing the Favicon Icon

Building the site

Other Commands and Options

Deploying

Getting help

# MkDocs

Project documentation with Markdown.

## Overview

MkDocs is a **fast**, **simple** and **downright gorgeous** static site generator that's geared towards building project documentation. Documentation source files are written in Markdown, and configured with a single YAML configuration file.

## Host anywhere

MkDocs builds completely static HTML sites that you can host on GitHub pages, Amazon S3, or [anywhere](#) else you choose.

## Great themes available

There's a stack of good looking themes available for MkDocs. Choose between the built in themes: [mkdocs](#) and [readthedocs](#), select one of the 3rd party themes in the [MkDocs wiki](#), or [build your own](#).

## Preview your site as you work

The built-in dev-server allows you to preview your documentation as you're writing it. It will even auto-reload and refresh your browser whenever you save your changes.

## Easy to customize

Get your project documentation looking just the way you want it by customizing the theme.



# How Netflix Does This

Branch: **gh-pages** ▾

New pull request



**Unknown** Deployed b2f5295 with MkDocs version: 1.0.4



**assets**

Deployed b2f5295 with MkDocs version: 1.0.4



**images**

Deployed b2f5295 with MkDocs version: 1.0.4



**latest**

Deployed b2f5295 with MkDocs version: 1.0.4



**search**

Deployed b2f5295 with MkDocs version: 1.0.4



**.nojekyll**

Deployed b2f5295 with MkDocs version: 1.0.4



**404.html**

Deployed b2f5295 with MkDocs version: 1.0.4



**index.html**

Deployed b2f5295 with MkDocs version: 1.0.4



**sitemap.xml**

Deployed b2f5295 with MkDocs version: 1.0.4



**sitemap.xml.gz**

Deployed b2f5295 with MkDocs version: 1.0.4



# How Netflix Does This

Source

A screenshot of a GitHub repository page. The repository path is 'cdemkdocs / docs / cassandra /'. The pull request is titled 'Markdown' and is from 'Joseph Lynch' to 'feature/josephl\_tradeoffs' pointing to 'master'. The commit message is 'Adding Material design and site layout'. The pull request has 11 commits. The code review interface shows several files like 'api.md', 'data\_explorer.md', 'datamodel.md', 'dual\_writes.md', and 'faq.md' with their respective descriptions.



Joseph Lynch

feature/josephl\_tradeoffs → master

## Write the tradeoff docs

Overview Diff Commits

Pull Request

Data Model

Search

CDE

Home

Getting Started ▾

Cassandra ^

Overview

Spring Data Cassandra ▾

Aeneas client ▾

Astyanax client ▾

## CQL Data Modeling

## Immediate Deploy

The most important choice you make when modeling data in Cassandra is what your **partition** key will be. Cassandra takes the data in the partition key, hashes it, and that determines which Cassandra host(s) will store your data.

If this partitioning is uneven, meaning that you end up with lots of data mapping to the certain partitions but not others, then you have created a **hotspot**, where query load (queries, and data size) is concentrated on a small number of nodes. Hotspots can knock over Cassandra clusters, harm your application availability and result in inconsistent performance. Avoid hotspots at all costs in your data models.

# Proposal

docs

Move svn website to git branch in main repo

code

Replace sphinx with markdown (pandoc)?

build

Automatically build and publish docs (jenkins + docker?)

|

N

# Proposal

Apache Beam <https://beam.apache.org>

python java big-data beam

docs

code

build

test

This screenshot shows a GitHub repository page for the Apache Beam website. The page includes a sidebar with category labels: 'docs' (black text), 'code' (light gray text), 'build' (light gray text), and 'test' (light gray text). The main content area displays the repository statistics and a list of recent commits.

**Statistics:**

- 253 commits
- 40 branches
- 118 releases
- 482 contributors
- Apache-2.0 license

**Branches:**

- Branch: asf-site ▾
- New pull request
- Create new file
- Upload files
- Find File
- Clone or download ▾

**Recent Commits:**

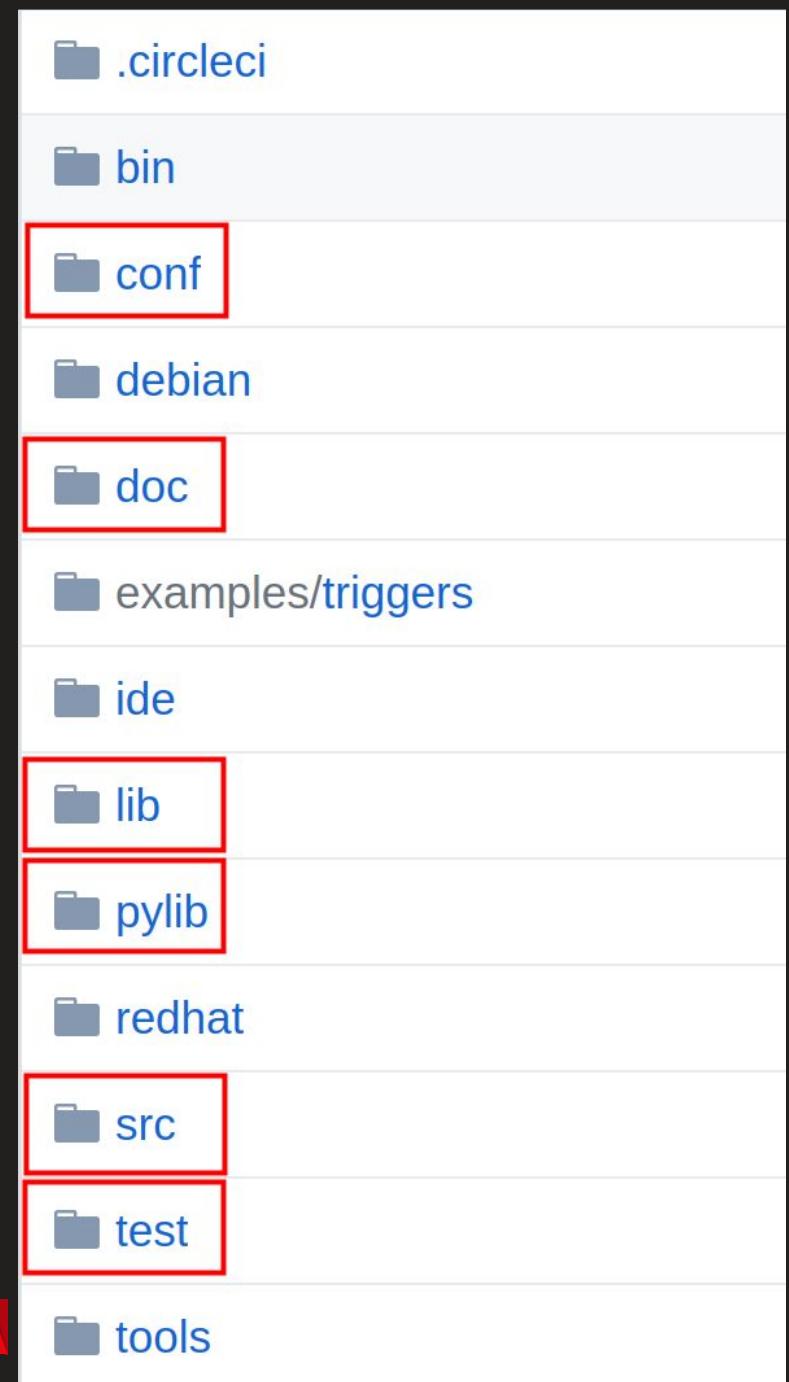
File	Commit Message	Date
jenkins	Publishing website 2019/09/06 15:52:26 at commit 8b7a3e3	Latest commit 766d92d yesterday
website/generated-content	Publishing website 2019/09/06 15:52:26 at commit 8b7a3e3	yesterday
.gitignore	[BEAM-5669] Stop empty commits being pushed by PostCommit_Website_Pub...	11 months ago
README.md	Add a README for the asf-site branch	last year
content	Publishing website 2018/09/26 01:51:44 at commit e9b7e5c	last year

**README.md:**

This branch contains the generated code for <http://beam.apache.org>.

To contribute to the website, please modify the [website sources on master](#). See the [contribution guide](#) for details.





# Non standard layout

# 2260 line XML build.xml

# Checked in library .jars

# Checked in python library



681



3.6K



34.3K



355



69.2K



9.5K



2.8K



3h



1.8K



1.4K



1h

## What is Gradle?

Gradle is a build tool with a focus on build automation and support for multi-language development. If you are building, testing, publishing, and deploying software on any platform, Gradle offers a flexible model that can support the entire development lifecycle from compiling and packaging code to publishing web sites.

## What is Apache Maven?

Maven allows a project to build using its project object model (POM) and a set of plugins that are shared by all projects using Maven, providing a uniform build system. Once you familiarize yourself with how one Maven project builds you automatically know how all Maven projects build saving you immense amounts of time when trying to navigate many projects.

```
rootProject.name='ndbench'

include 'ndbench-api'
include 'ndbench-cass-plugins'
include 'ndbench-cli'
include 'ndbench-core'
include 'ndbench-cockroachdb-plugins'
include 'ndbench-dynamodb-plugins'
include 'ndbench-dax-plugins'
include 'ndbench-es-plugins'
include 'ndbench-geode-plugins'
include 'ndbench-janusgraph-plugins'
include 'ndbench-sample-plugins'
include 'ndbench-web'
include 'ndbench-aws'
```

# Can use Gradle for Cassandra too

```
sourceSets.remove(sourceSets.main)
sourceSets.remove(sourceSets.test)

ant.importBuild('build.xml') { antTargetName ->
    'ant' + antTargetName
}
```

Import  
build.xml

```
dependencies {
    compile "com.netflix.priam:priam-cass-extensions:${priam_extensions_version}"
    compile 'com.jeffjirsa.cassandra.db.compaction:TimeWindowCompactionStrategy:2.1'
}
```

Add any jars we want at build time

# Can use Gradle for Cassandra too

```
ext {  
    pname = "nfcassandra-30x"  
    cass_version = ant.properties['version']  
    cass_home_dir =  
    cass_filename = "nf-cassandra-${project.ext.cass_version}-bin.tar.gz"  
    priam_extensions_version = "3.11.52"  
    priam_extensions_jar = "priam-cass-extensions-${priam_extensions_ver.  
}  
}
```

```
buildDeb {  
    dependsOn customizeFiles  
}  
  
ospackage {  
    packageName project.ext.pname  
    version = project.ext.cass_version  
  
    println "Building deb for nf-cassandra ${version}"  
    into("${project.ext.cass_home_dir}/lib") {  
        from configurations.runtime.filter { it.name.startsWith('priam') }  
        ...  
    }
```

Build debs  
directly

# Proposal

docs

Start by importing build.xml

code

Gradually modernize build system  
via **gradle** multi-project builds

build

Use out of the box **dependency locking**

test

# Proposal

docs

code

build

test

The screenshot shows GitHub search results for Apache projects. On the left, there are four large, semi-transparent text labels: "docs", "code", "build", and "test". To the right of these labels are two GitHub repository cards.

**Top Card (Apache Kafka):** Shows the repository "apache / kafka". It has a "Code" tab (selected), 663 pull requests, and an "Actions" button. A dropdown menu shows "Branch: trunk". Below it, a pull request by "rajinisivaram" is listed for "KAFKA-8760; New Java Aut".

**Bottom Card (Apache Beam):** Shows the repository "apache / beam". It has a "Code" tab (selected), 127 pull requests, and an "Actions" button. A dropdown menu shows "Branch: master". Below it, a pull request by "Hannah-Jiang" is listed for "BEAM-7909 Python3 docker containers".

**Contributor Count:** The Kafka card shows 78 contributors with small profile icons. The Beam card shows 55 contributors with small profile icons.

[Back to Project](#)[Status](#)[Changes](#)[Console Output](#)[View as plain text](#)[View Build Information](#)[Polling Log](#)[Timings](#)[Environment Variables](#)[Git Build Data](#)[Parameters](#)[Test Result](#)[Open Blue Ocean](#)[Embeddable Build Status](#)[Previous Build](#)

## Build #841 (Sep 6, 2019 5:44:10 AM)

REF = origin/trunk

COMMIT = fc4381ca89ab39a82c9018e5171975285cc3bfe7



### Build Artifacts

[test\\_stdout.txt](#) 301.34 KB [view](#)

### Changes

1. Use `rm -f` in rpm spec to prevent failure on missing file ([detail](#))
2. Define upstream\_version in rpm spec to deal with s/~/-/ ([detail](#))
3. Add auditlogviewer and fqldtool to rpm spec ([detail](#))

[Started by an SCM change \(3 times\)](#)

This run spent:

- 9 hr 41 min waiting;
- 9 hr 59 min build duration;
- 19 hr total from scheduled to completion.



Revision: fc4381ca89ab39a82c9018e5171975285cc3bfe7

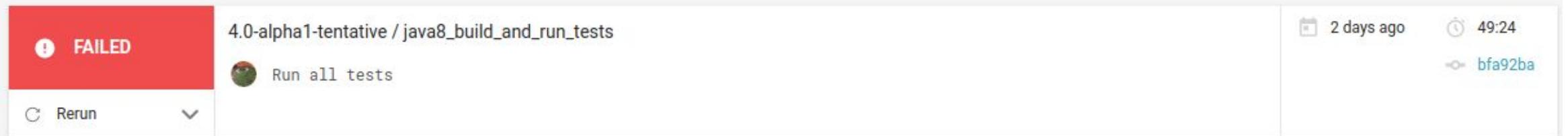
- origin/trunk

[Test Result \(3 failures / -1\)](#)[Show all failed tests >>>](#)

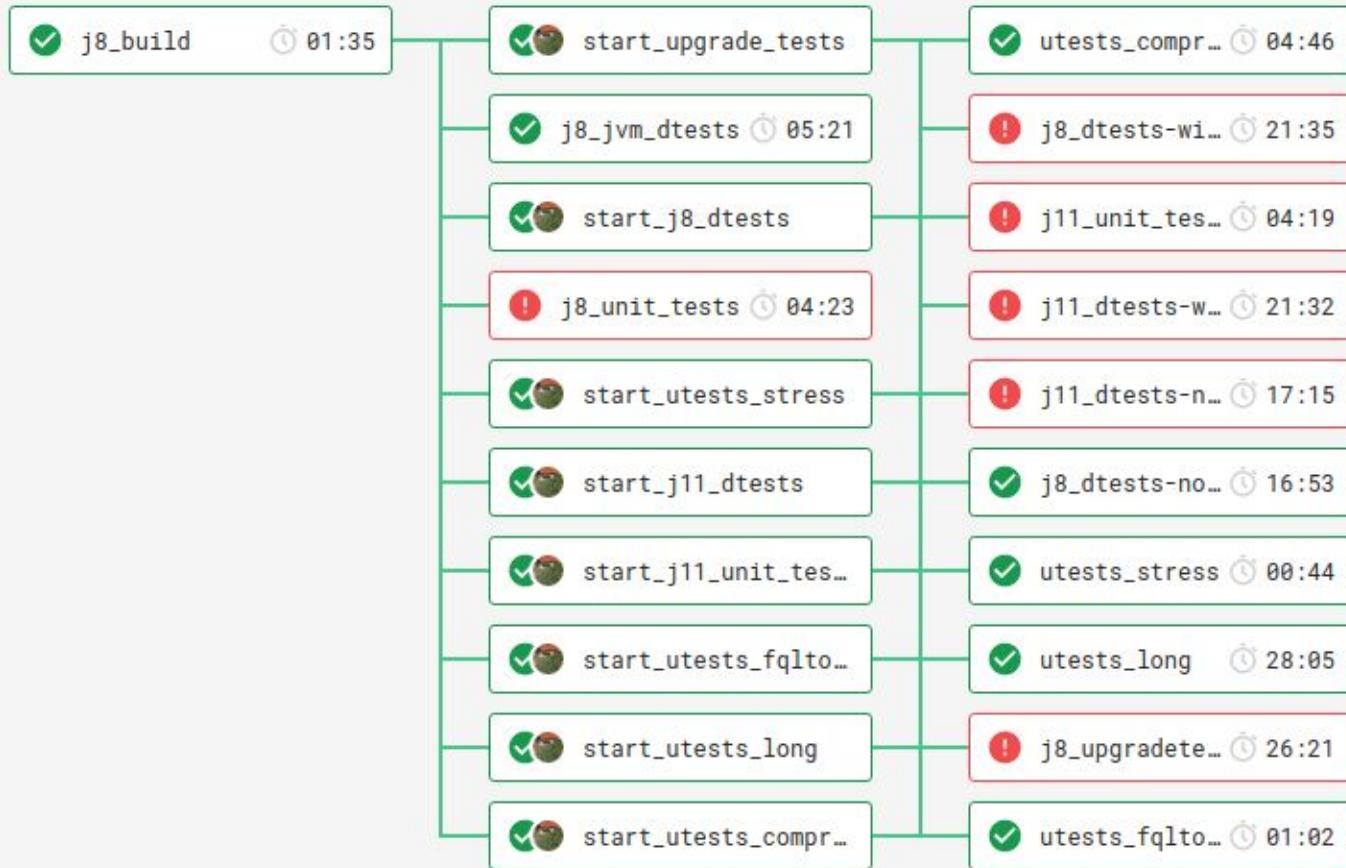
# Jenkins

# 10 hour builds

# Who can run?



21 jobs in this workflow



# CircleCI

## 20 minute builds

## Anyone can run them



# Proposal

docs

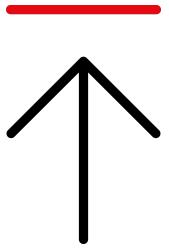
Keep moving with the CircleCI integration for testing (artifacts stay on jenkins)

code

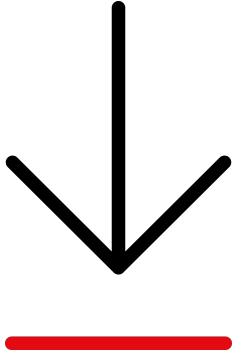
build

Run unit tests on every pull request (5 minutes, doable with free tier)

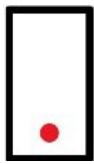
test



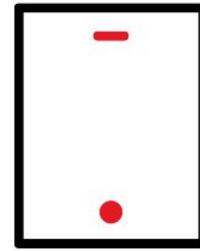
Develop



Distribute



Launch



Scale

# Cloud Native Distribution

source

Building from source must be easy

packages

Integration with package managers  
apt, yum, brew . . .

containers

Having Docker containers for  
testing, as well as production

# Cloud Native Distribution

source

Already talked about this, we can improve in this area with previous proposals

packages

containers

# Cloud Native Distribution

source

packages

Already have pretty great deb and rpm integration.

containers

Custom packages slightly difficult



## make the debian package never start by default

### Details

Type:	Improvement	Status:	OPEN
Priority:	<input type="radio"/> Low	Resolution:	Unresolved
Component/s:	Packaging	Fix Version/s:	4.x
Labels:			

### Description

Currently the debian package that installs cassandra starts cassandra by default. It sounds like that is a standard debian packaging convention. However, if you want to bootstrap a new node and want to configure it before it creates any sort of state information, it's a pain. I would think that the common use case would be to have it install all of the init scripts and such but **not** have it start up by default. That way an admin can configure cassandra with seed, token, host, etc. information and then start it. That makes it easier to programmatically do this as well - have chef/puppet install cassandra, do some configuration, then do the service start.

With the current setup, it sounds like cassandra creates state on startup that has to be cleaned before a new configuration can take effect. So the process of installing turns into:

- install debian package
- shutdown cassandra
- clean out state (data/log dirs)
- configure cassandra
- start cassandra

That seems suboptimal for the default case, especially when trying to automate new nodes being bootstrapped.

Another case might be when a downed node comes back up and starts by default and tries to claim a token that has already been claimed by another newly bootstrapped node. Rob is more familiar with that case so I'll let him explain it in the comments.

# Autostart is not Cloud Native

# Cloud Native Distribution

source

packages

containers

Not doing well at containers.

<https://github.com/docker-library/cassandra>

---

## Maintained by: the Docker Community

---

This is the Git repo of the Docker "Official Image" for `cassandra` (not to be confused with any official `cassandra` image provided by `cassandra` upstream). See [the Docker Hub page](#) for the full readme on how to use this Docker image and for information regarding contributing and issues.

The full image description on Docker Hub is generated/maintained over in [the docker-library/docs repository](#), specifically in the `cassandra` directory.

# How Netflix Does Testing Images

```
# Either tearing out nflx customizations we can't use in the container
# or setting things so we startup faster
cass_config.update({
    "cluster_name": "docker",
    "listen_address": os.environ.get("LISTEN_ADDRESS", "127.0.0.1"),
    "rpc_address": os.environ.get("RPC_ADDRESS", "0.0.0.0"),
    "broadcast_rpc_address": os.environ.get("BROADCAST_RPC_ADDRESS",
    "partitioner": "org.apache.cassandra.dht.Murmur3Partitioner",
    "auto_bootstrap": False,
    "server_encryption_options": {"internode_encryption": "none"},
    "client_encryption_options": {"enabled": False},
    "endpoint_snitch": "SimpleSnitch",
    "seed_provider": [
        {
            "class_name": "org.apache.cassandra.locator.SimpleSeedProvider",
            "parameters": [{"seeds": "127.0.0.1"}]
        },
        "default_keyspace_rf": 1,
        "minimum_keyspace_rf": 1,
        "initial_token": "0",
        "file_cache_size_in_mb": 10,
    ])
}

# Allow the user to override any configuration options they want
extra_config = os.environ.get("CONFIG_OVERRIDE_JSON", "{}")
print("Found overrides: {}".format(extra_config))
cass_config.update(json.loads(extra_config))
```

~5s startup\*

Easy to customize

Minimal memory footprint

\*skip\_wait\_for\_gossip\_to\_settle=0,  
 disable vnodes, etc ..



# Proposal

source

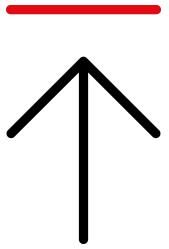
Do earlier suggestions

packages

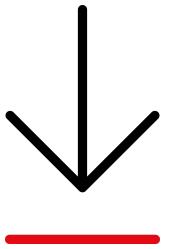
Stop autostarting Cassandra  
packages

containers

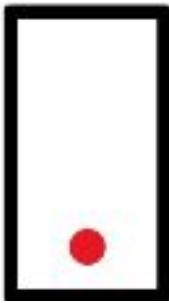
Publish official testing container  
Publish official kubernetes/swarm  
containers



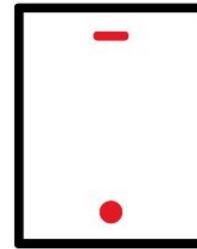
Develop



Distribute



Launch



Scale

# Launching Nodes

seeds

Which nodes will initiate the cluster

tokens

How will we assign tokens

process

How to manage Cassandra process

# How does Netflix launch?

seeds

Ask AWS cloud control plane

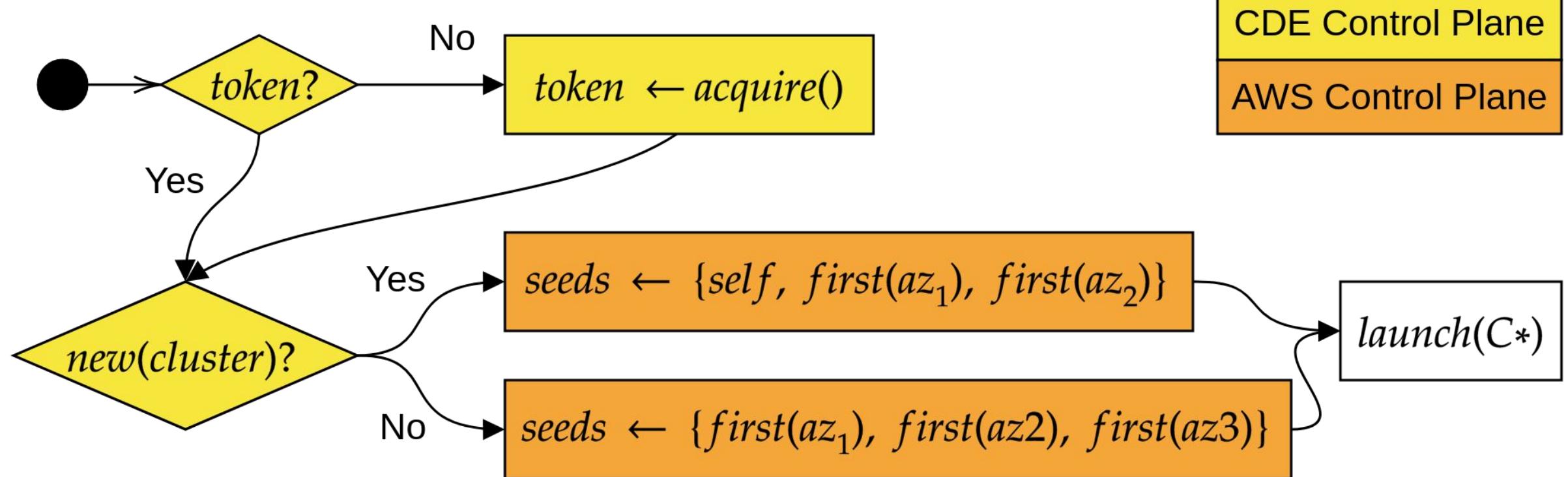
tokens

Ask CDE cluster control plane

process

Use systemd

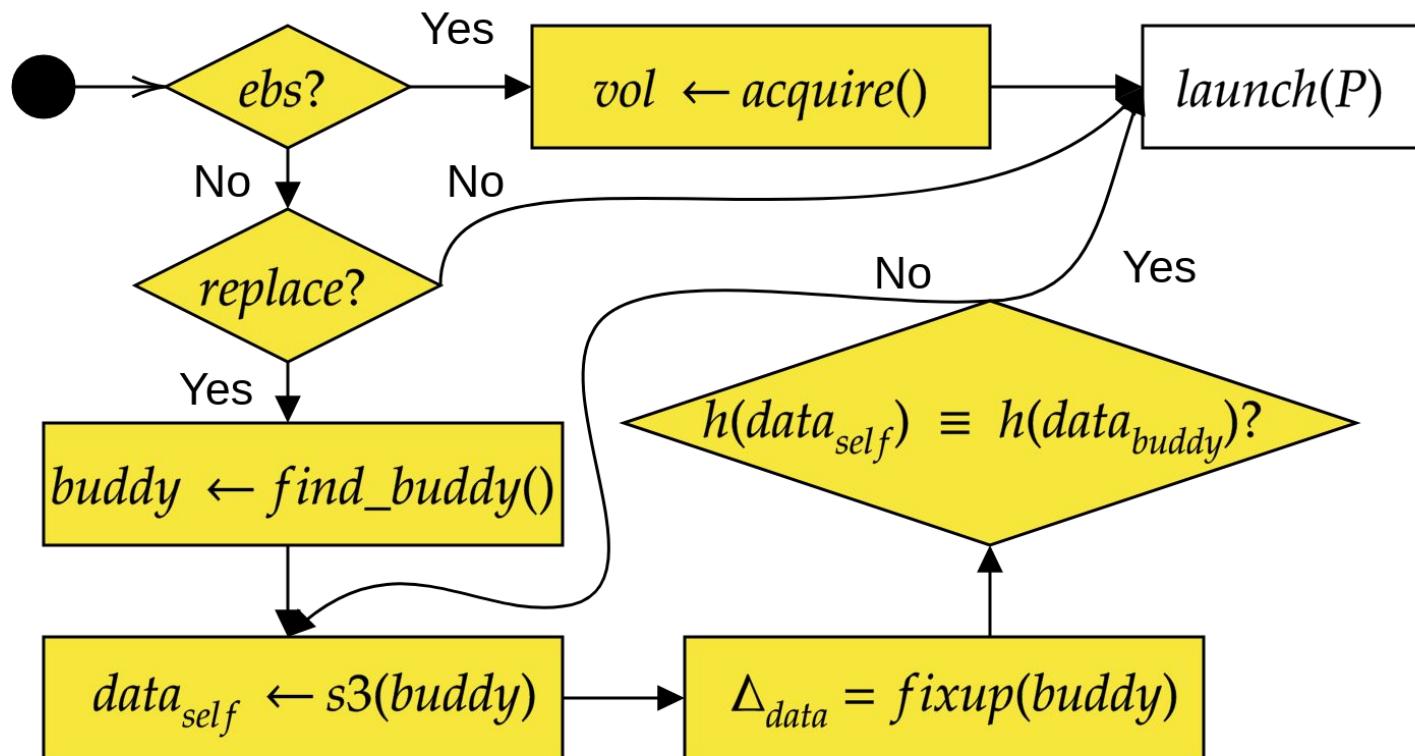
# Declarative Control Plane



R I A M

N

# Data Migration?



Avoid Cassandra  
Streaming

Use S3, Use EBS

Use direct file copy  
(sendfile)

Verify with xxHash

# Data Migration: The Numbers

$$500 \frac{\text{GiB}}{\text{node}} / 200 \frac{\text{mb}}{\text{second}} = 5.96 \frac{\text{hours}}{\text{node}}$$

$$5.96 \frac{\text{hours}}{\text{node}} * 288 \text{ node} = 2.35 \text{ months}$$

---

$$500 \text{ GiB} / 2000 \frac{\text{mb}}{\text{second}} = 0.6 \text{ hours}$$

$$0.6 \text{ hours} + 10 \frac{\text{minutes}}{\text{node}} * 288 \text{ node} = 2.02 \text{ days}$$

---

$$500 \text{ GiB} / 2000 \frac{\text{mb}}{\text{second}} = 0.6 \text{ hours}$$

$$0.6 \text{ hours} + 20 \frac{\text{minutes}}{\text{zone}} * 3 \text{ zone} = 1.596 \text{ hours}$$

---

Naive solution,  
sequential transfer

Parallel S3 download  
sequential fixup

Parallel download and  
zone wide fixup

# Process Control?

```
##/lib/systemd/system/cassandra.service

[Unit]
Description=Cassandra
# Don't ever give up on starting Cassandra
StartLimitInterval=0
StartLimitIntervalSec=1800

[Service]
RuntimeDirectory=cassandra
ExecStart=/usr/bin/cassandra_wrapper
LimitNOFILE=1048576
LimitNPROC=131072
LimitMEMLOCK=infinity
# always restart Cassandra
# If you want to stop Cassandra, either do:
# * nodetool drain
# * systemctl stop cassandra
Restart=always
RestartSec=30
StartLimitBurst=4
```

Just use systemd

Handles automatic  
restarts

Handles console logs

It's great

# Process Control?

```
## /usr/bin/cassandra_wrapper
# OS Page Cache Loading
cd ${CASSANDRA_HOME}/data
set +e

if [ -e .happycache.gz ]; then
    happycache load
    # Delete the OS cache file even if the load fails
    rm -f .happycache.gz
fi
set -e

## Pid file for preventing multiple processes
## starting at once

PIDFILE=/run/cassandra/cassandra.pid
exec 3<> "${PIDFILE}"

if ! flock -n 3; then
    PID=`cat ${PIDFILE}`
    echo "Cassandra already running (best guess in PID:
${PID}), cannot start"
    exit 1
fi

echo $$ > ${PIDFILE}

# Start Cassandra
...
```

Reload OS page cache  
with `happycache`

Guarantee single process  
running with flocked  
pidfiles

# Proposal

seeds

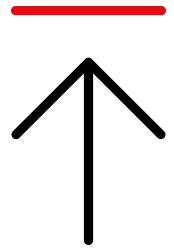
Sidecar + native cloud providers

tokens

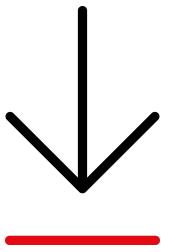
Add TokenProvider interface,  
should happen for (#13701). More  
full SSTable streaming.

process

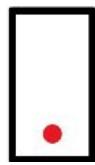
Just add a systemd unit



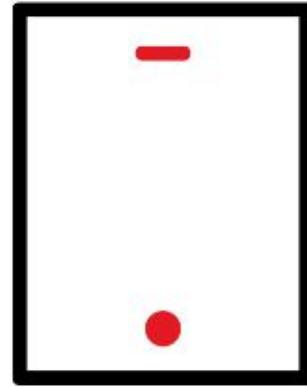
Develop



Distribute



Launch



Scale

# Scaling Cassandra

metrics	Operational insights into performance
monitor	What is happening with the cluster
integrity	Data integrity (failure, corruption)
administer	Must be easy to configure and restart

# Scaling Cassandra

metrics

Interoperable with ... JMX, but not

- statsd
- prometheus
- spectator

monitor

backup

administer

Performant export is a problem

# How does Netflix get metrics?

Netflix / spectator

Code Issues 15 Pull requests 0 Projects 0

Client library for collecting metrics.

1,055 commits 6 branches 113 releases

Branch: master ▾ New pull request

brharrington adjust initial polling delay (#756) ...  
make cardinality limiters serializable  
updates to conform to glossary  
update to gradle 5.4.1 and spock  
remove custom reports from  
agent: support reloading changes  
cache normalized ids (#754)

codequality  
docs  
gradle/wrapper  
scripts  
**spectator-agent**  
spectator-api

```
# cassandra-env.sh patch
# Pull in any agents present in CASSANDRA_HOME

for agent_file in ${CASSANDRA_HOME}/agents/*.jar; do
    if [ -e "${agent_file}" ]; then
        base_file="${agent_file%.jar}"
        if [ -s "${base_file}.options" ]; then
            options=`cat ${base_file}.options`
            agent_file="${agent_file}=${options}"
        fi
        JVM_OPTS="$JVM_OPTS -javaagent:${agent_file}"
    fi
done

for agent_file in ${CASSANDRA_HOME}/agents/*.so; do
    if [ -e "${agent_file}" ]; then
        base_file="${agent_file%.so}"
        if [ -s "${base_file}.options" ]; then
            options=`cat ${base_file}.options`
            agent_file="${agent_file}=${options}"
        fi
        JVM_OPTS="$JVM_OPTS -agentpath:${agent_file}"
    fi
done
```

# Proposal

metrics

Allow pluggability of histogram

monitor

Official exporter agents + pluggable  
agents

integrity

HTTP metrics endpoint on sidecar

administer

# Scaling Cassandra

metrics

monitor

integrity

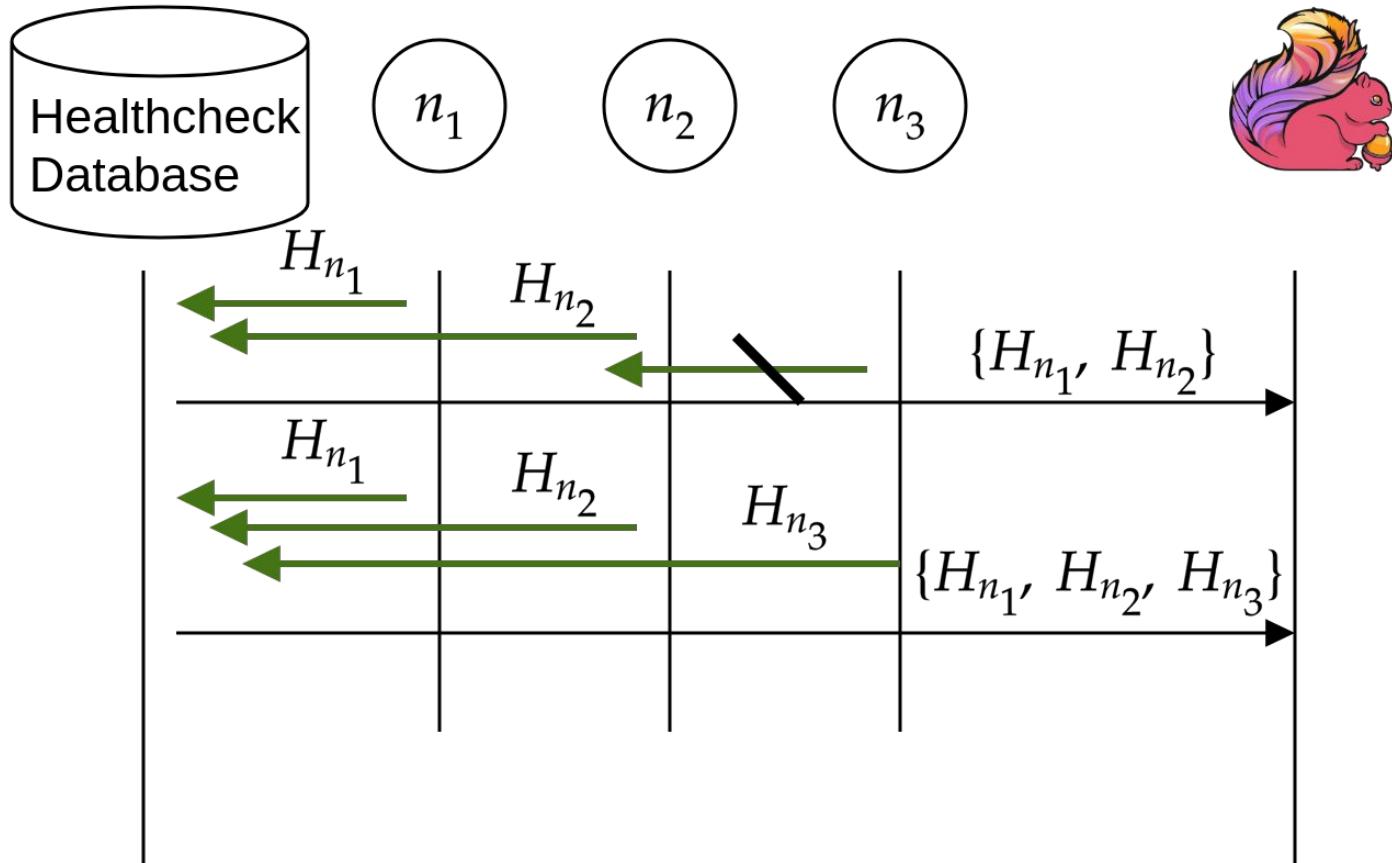
administer

No real way to know cluster view

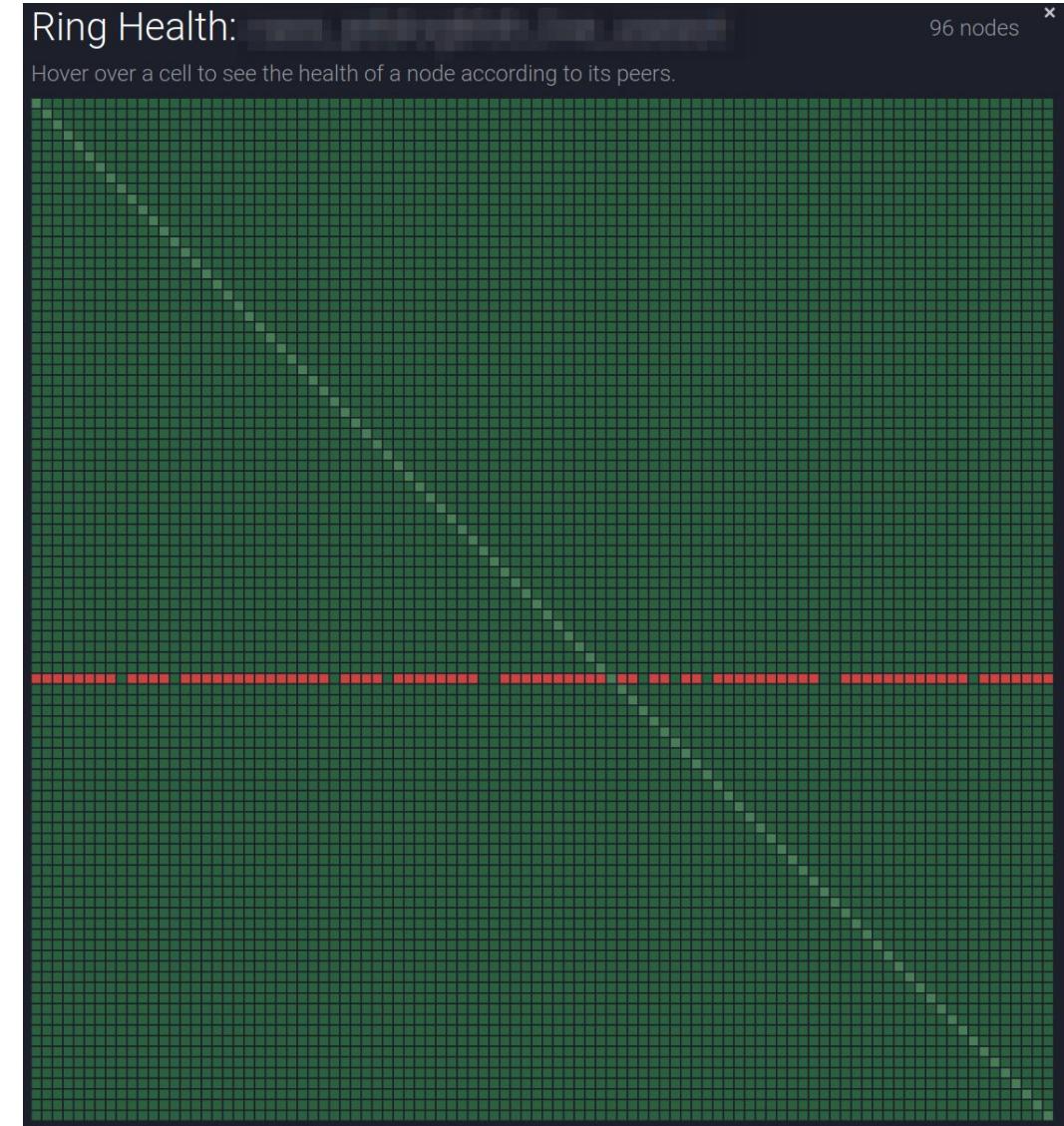
Is the ring healthy?

Where is my data located?

# How does Netflix monitor?



Cluster Health



# How does Netflix monitor?

CDE Self Service   Cassandra   Elasticsearch   Dynomite   EVCache   ZooKeeper   Admin

Cassandra Clusters » cass\_foobar

Summary   Costs   Nodes   Keyspaces   Backups   Repairs   SLOs   Maintenance Windows   Notes   Attributes   Ingress

Owner Emails: subdevteam, dev1, dev2, dev3, dev4

PagerDuty Service: subeng

Slack Channel: subscribeteam

Update

Annualized Cost: \$\$\$\$

Has Customer Impact?   Contains Critical Data?

Maintenance: None   Backup: Finished   Repair: Started - 97%

Last 1 hour

US-EAST-1   EU-WEST-1   US-WEST-2

Coordinator Reads		
Last 1h		
AVG		
<b>944.1 K/s</b>		
MIN 920.1 K/s	MAX 985.3 K/s	TOTAL 3.39 M

Coordinator Writes		
Last 1h		
AVG		
<b>1.5 M/s</b>		
MIN 1.4 M/s	MAX 1.5 M/s	TOTAL 88.5 M

Coordinator Read Latency 99th		
Last 1h		
AVG		
<b>5.5 ms</b>		
MIN 4.6 ms	MAX 7.2 ms	

Coordinator Write Latency 99th		
Last 1h		
AVG		
<b>1.2 ms</b>		
MIN 1.2 ms	MAX 1.3 ms	

Data Size		
TOTAL		
<b>258 TB</b>		

Data Explorer   Metrics   Cluster Health

N

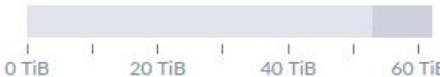
# How do other DBs do this?

OVERVIEW

## CLUSTER OVERVIEW

### Capacity Usage

0.0%



USED CAPACITY

6.4 GiB

USABLE CAPACITY

62.0 TiB

### Node Status

18

0

1

LIVE NODES

SUSPECT NODES

DEAD NODES

### Replication Status

399

0

0

TOTAL RANGES

UNDER-REPLICATED RANGES

UNAVAILABLE RANGES

METRICS

DATABASES

STATEMENTS

JOB

VIEW: NODE LIST ▾

### Dead Nodes

ID

ADDRESS

DEAD SINCE

n8

2 months ago

### Live Nodes

ID

ADDRESS

UPTIME

REPLICAS

CPUS

CAPACITY USAGE

MEM USAGE

VERSION

n1

[REDACTED]

5 months

70

16

0%

508.7 MiB

3.4 TiB

31%

37.9 GiB

120.0 GiB

v2.1.6

n2

[REDACTED]

5 months

69

16

0%

365.8 MiB

3.4 TiB

31%

37.5 GiB

120.0 GiB

v2.1.6

n3

[REDACTED]

5 months

70

16

0%

412.9 MiB

3.4 TiB

31%

38.2 GiB

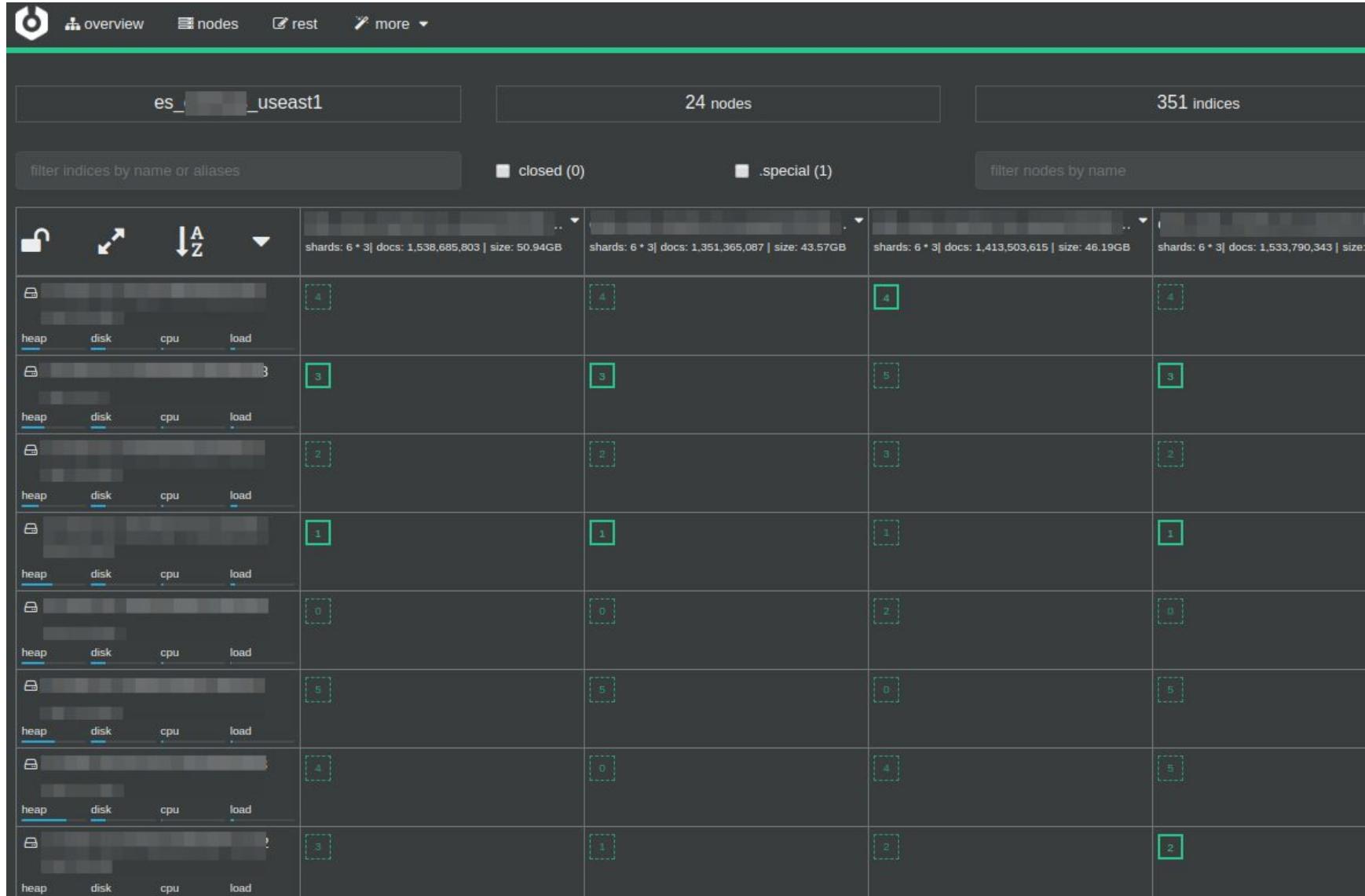
120.0 GiB

v2.1.6

CockroachDB  
out of the box



# How do other DBs do this?



Elasticsearch  
plugs in  
Cerebro



# Proposal

metrics

Ship out of the box health and status dashboards

monitor

JSON metrics/status may be sufficient

integrity

Sidecar may be a great place for this

# Scaling Cassandra

metrics

Nodes and clusters constantly failing

monitor

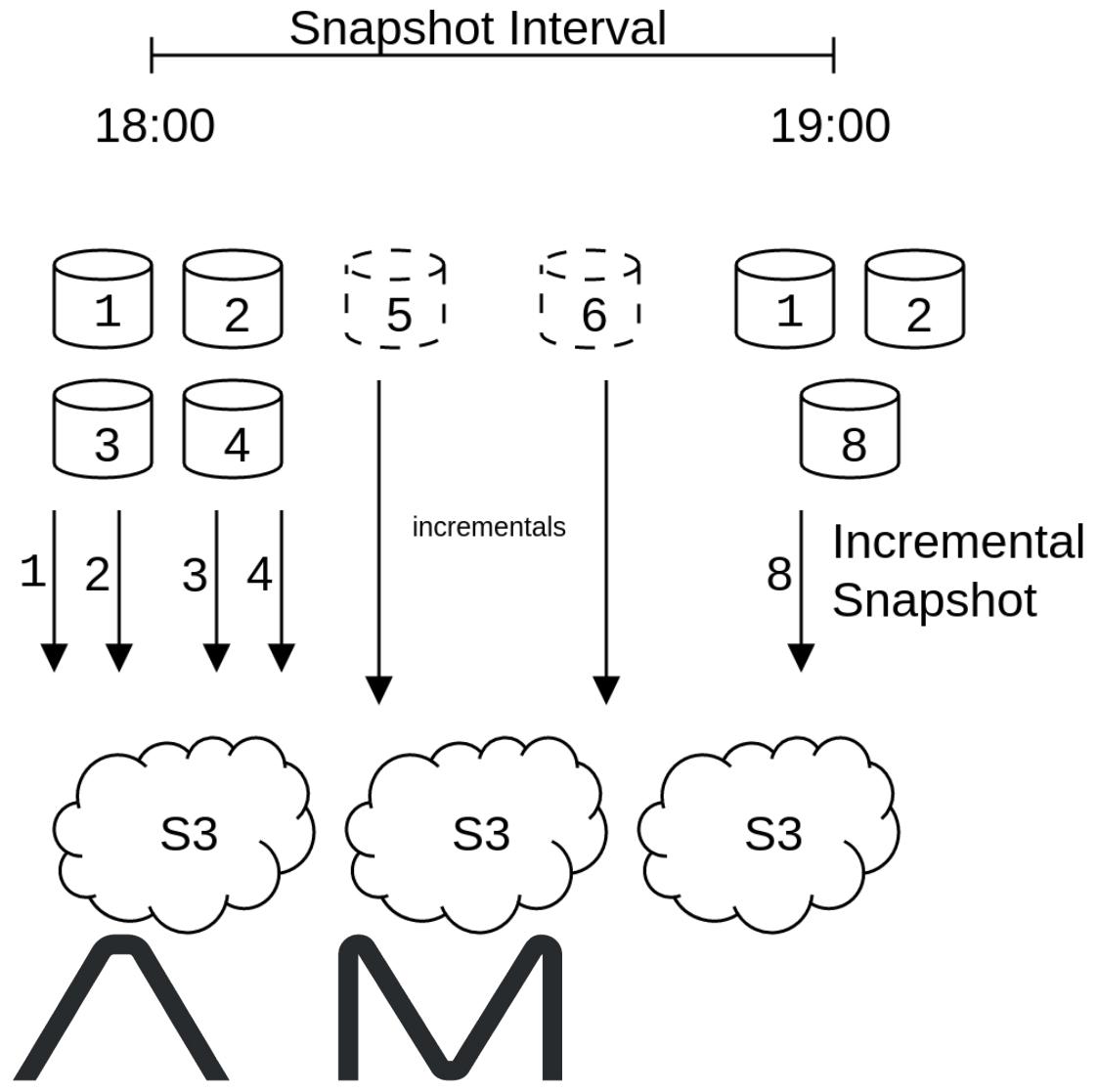
Lack critical backup, restore and repair scheduling functionality

integrity

No standard way to configure and restart nodes in a cluster

# How does Netflix backup?

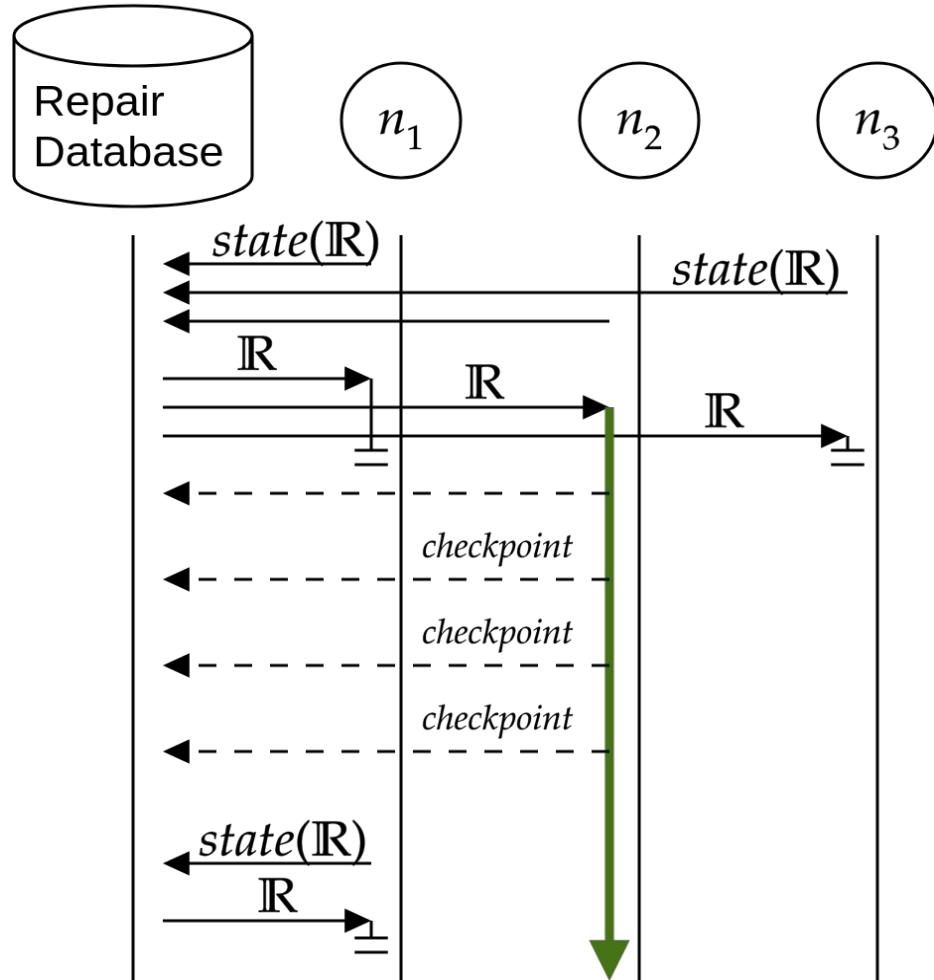
- Point in time
  - ◆ Incremental snapshot
  - ◆ Optimal performance
- Plugins for different clouds



R I ▲ M



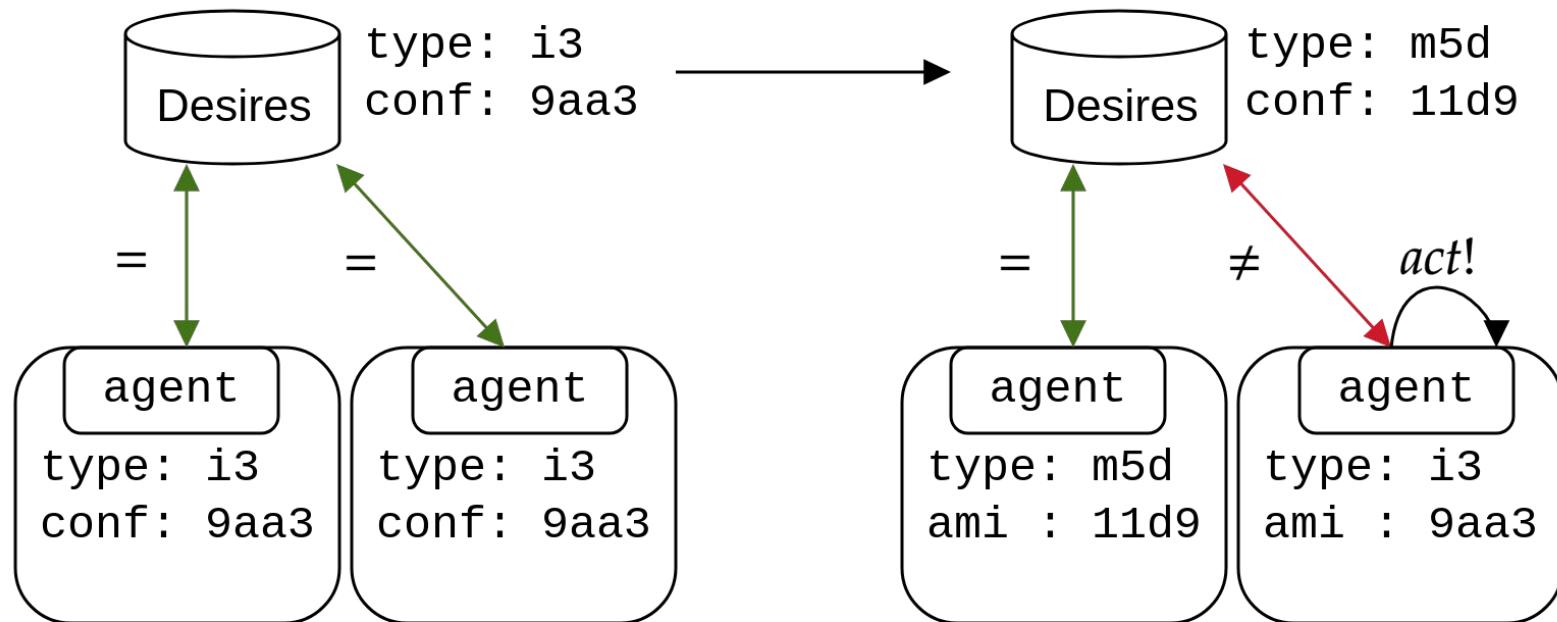
# How does Netflix repair?



- Distributed Control Plane
- Every node follows **repair state machine (#14346)**



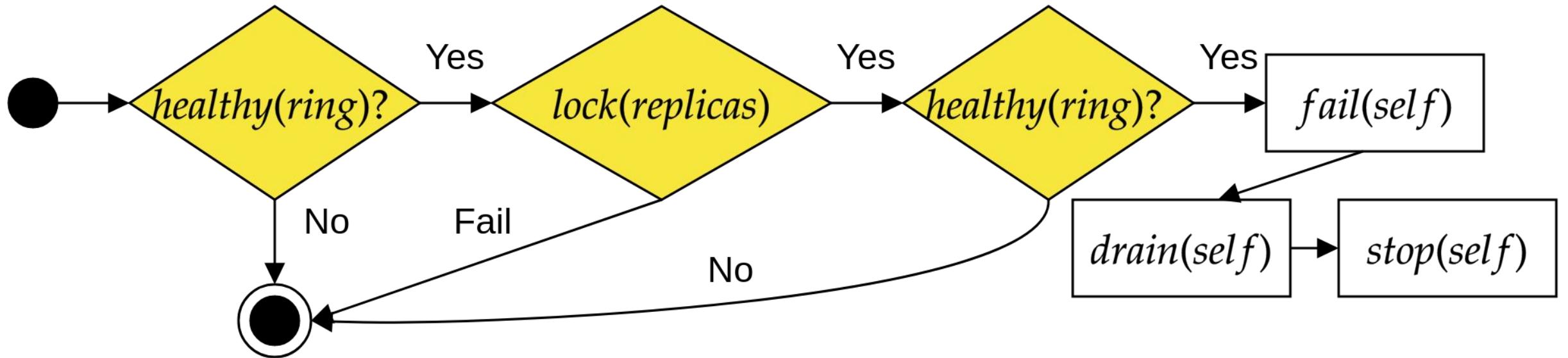
# How does Netflix administer?



Declarative

Desire based

# Distributed State Machines



# Proposal

metrics

Add native cloud backup plugins

monitor

Continue investing in Apache  
Cassandra Sidecar project

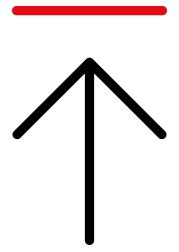
integrity

Backups

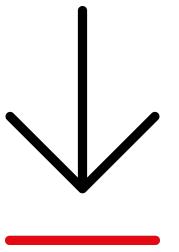
Repairs

Restarts

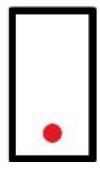
administer



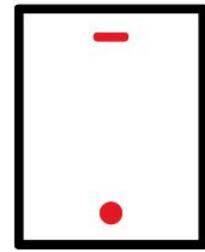
Develop



Package



Starting



Scaling

**Our users need better solutions**

**We need the community's help!**

**With targeted investment we can  
solve this problem**

# Discussion?