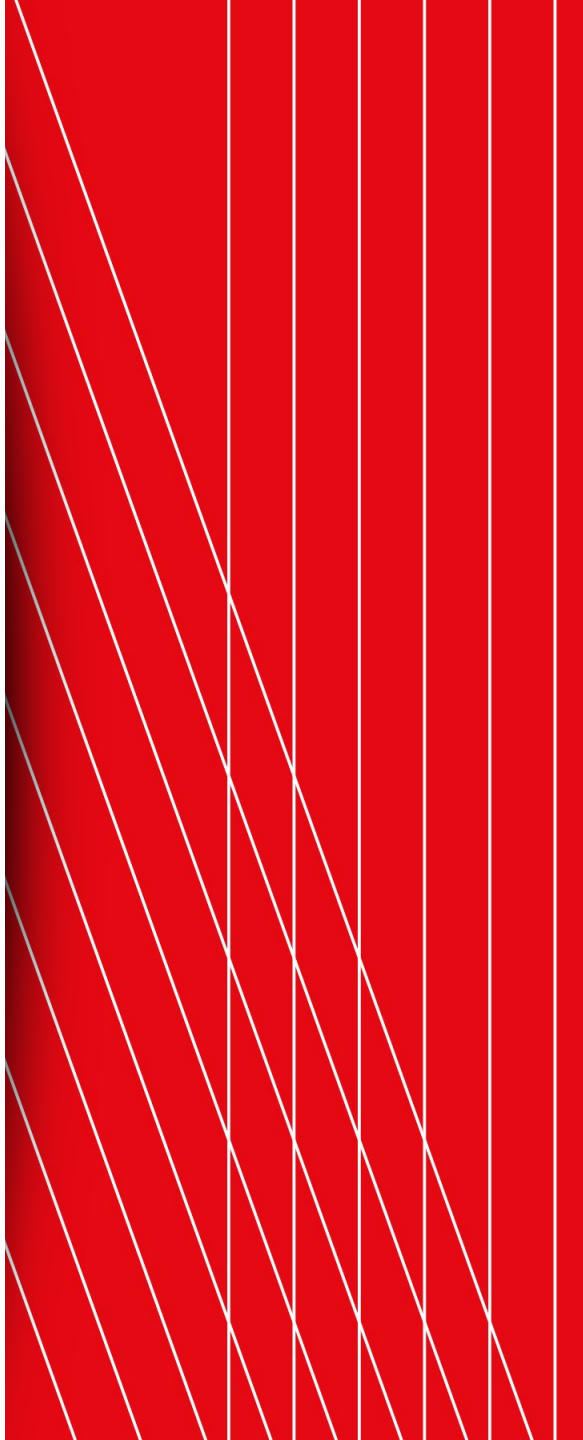
Apache C* Sidecar

let’s make C* attractive and easy to operate

Vinay Chella, Dinesh Joshi

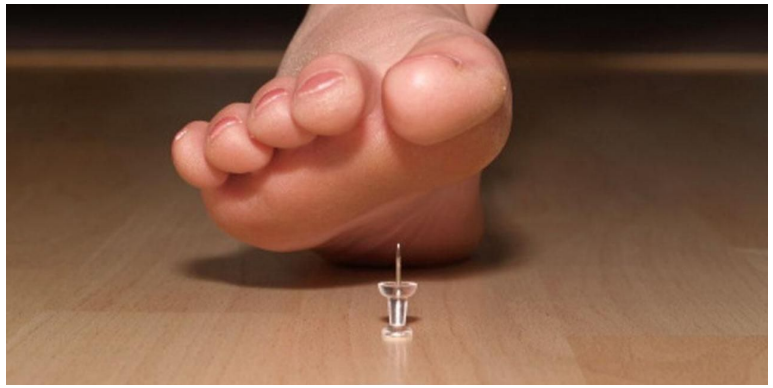


Agenda

- Operating C*
- Operating C* with a sidecar
- State of community sidecars
- Lessons learned from operating with sidecars
- Goals of C* management sidecar

Operating C*

- Bootstrap and data movement
- Configuration (files, jmx)
- Maintenance
- Monitoring/Metrics
- Backup/Restore
- Repair



Operating C*: Bootstrapping

Create a New Cluster

- Seeds
- Token assignment

Add/Remove/Replace

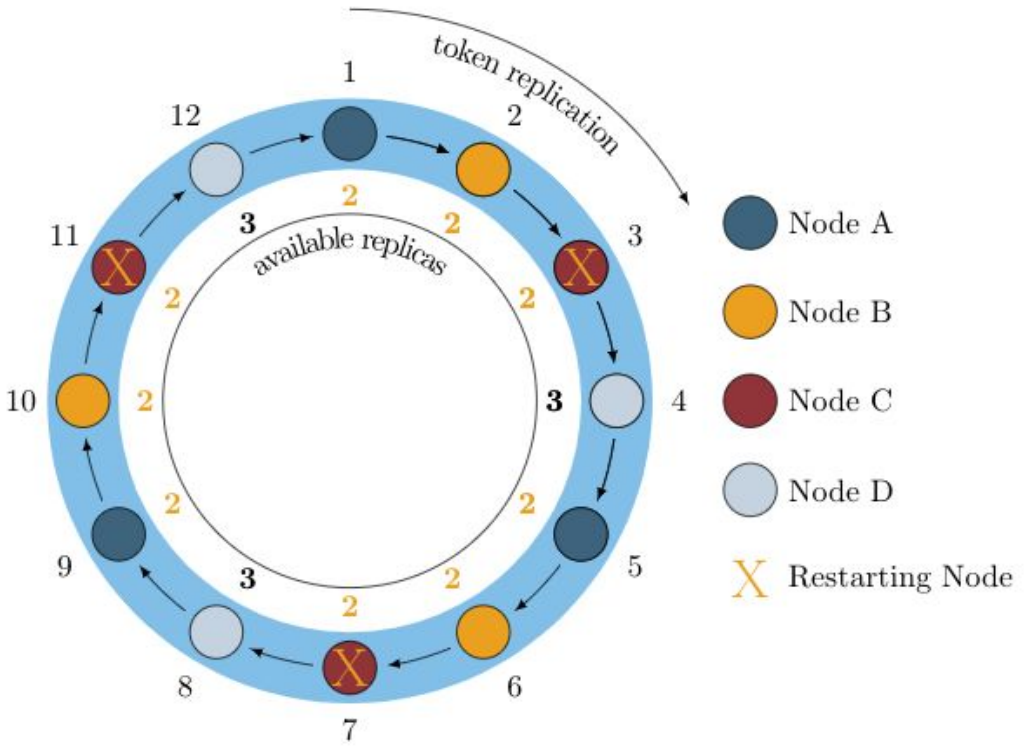
- Serial or parallel?
- Streaming?

Operating C*: Configuration

```
.
..
cassandra-env.ps1
cassandra-env.sh
cassandra-rackdc.properties
cassandra-topology.properties
cassandra.yaml
commitlog_archiving.properties
cqlshrc.sample
hotspot_compiler
jvm.options
logback-tools.xml
logback.xml
metrics-reporter-config-sample.yaml
README.txt
triggers
```

- Probably Have to Tune
  a. cassandra.yaml
  b. topology props
  c. JVM options
- May Have to Tune
  a. Logging
  b. Incremental Backup
  c. More JVM options

Operating C*: Lifecycle



Rolling Restarts (Upgrades)

- Semi-complex single node procedure
- One at a time is too slow
- Token range aware restarts?

What happens when Cassandra dies?

Ring source @ https://v2.overleaf.com/read/zchtrzskkyjb

Operating C*: Maintenance

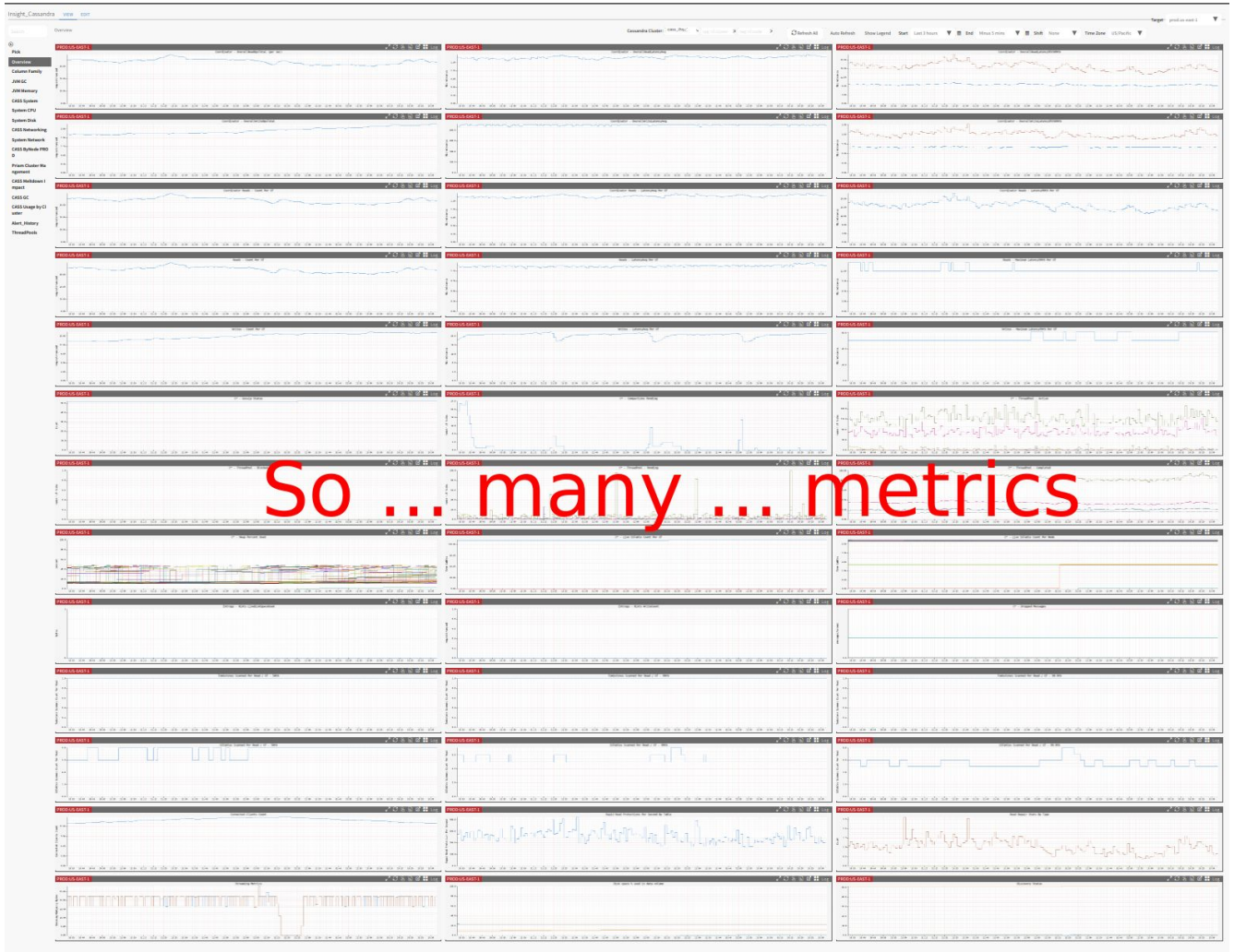
```
[redacted] ~ $ jmxterm
Welcome to JMX terminal. Type "h
$>open localhost:[redacted]
#Connection to localhost:[redacted] is
$>domains
#following domains are available
JMImplementation
ch.qos.logback.classic
com.sun.management
java.lang
java.nio
java.util.logging
org.apache.cassandra.auth
org.apache.cassandra.db
org.apache.cassandra.hints
org.apache.cassandra.internal
org.apache.cassandra.metrics
org.apache.cassandra.net
org.apache.cassandra.request
org.apache.cassandra.service
```

```
$>bean Display all 4258 possibilities? (y or n)
```

- All the Power of JMX
- … So many possibilities
  a. Many work with jmxterm/jmxsh
  b. Many only work with Java code
- What if you want to do it on all nodes?

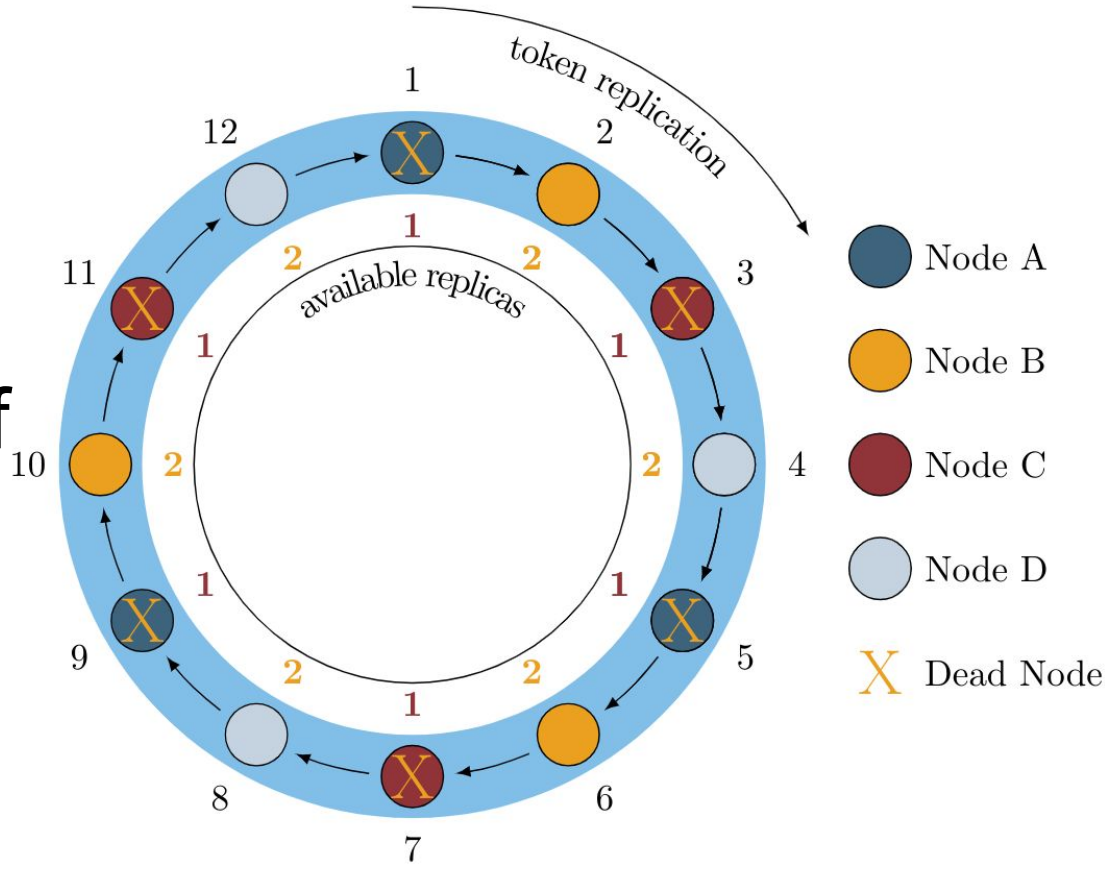
Operating C*: Monitoring

- Many Metrics (good!)
- How to Collect Them?
  - JMX … no
  - Agent!
- Which agent …



Operating C*: Ring Health

- Cassandra ring health depends on replication
- Strategies
  - Monitor replication of keyspaces
  - **Topology** Aware
  - **Maintenance** Aware



Operating C*: Backup/Restore

- What even do I need to backup!?
- Restore is legitimately tricky, do you practice?

Operating C*: Repair

"Eventually" Consistent

1. Partial Write
2. Read Repair
3. Hints play

... Nope not enough

4. Repair

Datacenter 1			Datacenter 2		
N1	N2	N3	N4	N5	N6
0	1	0	0	0	0
0	1	1	0	0	0
0	1	1	0	1	0
0	1	1	0	1	0
1	1	1	1	1	1

Sidecar: Bootstrapping

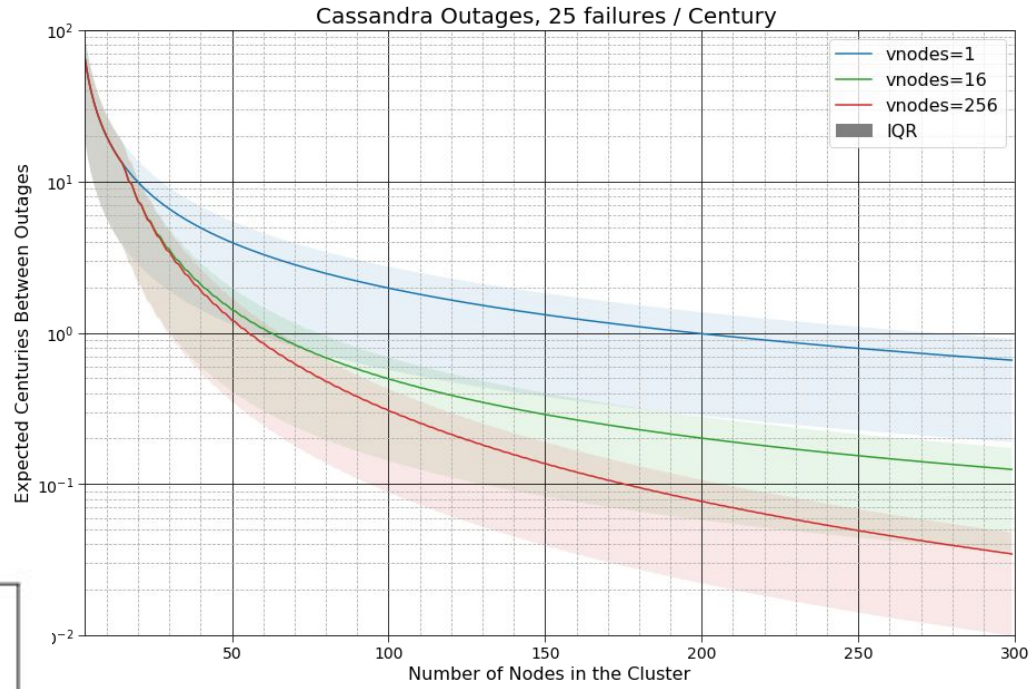
`/Priam/REST/v1/cassconfig/get_seeds`

Automatic Seed Management using ASGs/db

`/Priam/REST/v1/cassconfig/get_replaced_ip`

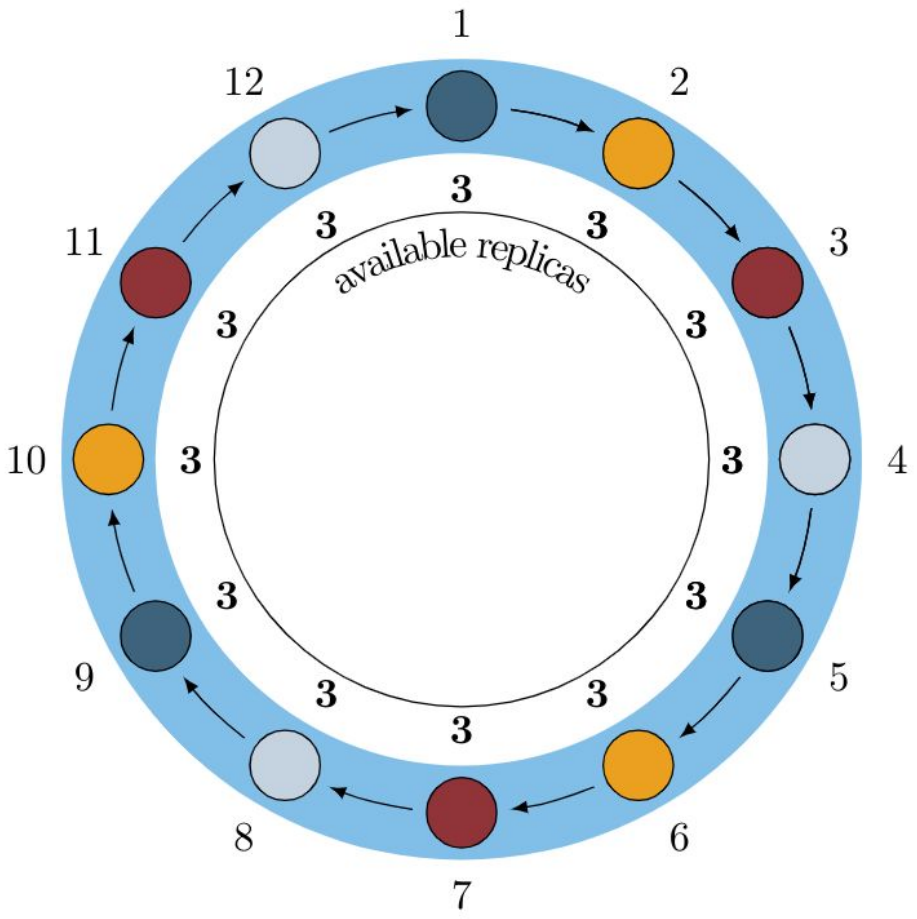
Automatic Instance Replacement

$$
\begin{aligned}
E[outages] &= \lambda_{global} \\
&= (n * \lambda_{split}) \\
&= (n * (\lambda * P(outage|failure))) \\
&= (n * (\lambda * (1 - e^{-E[T_{recovery}] * E[n_{neighbors}] * \lambda_s})))
\end{aligned}
$$

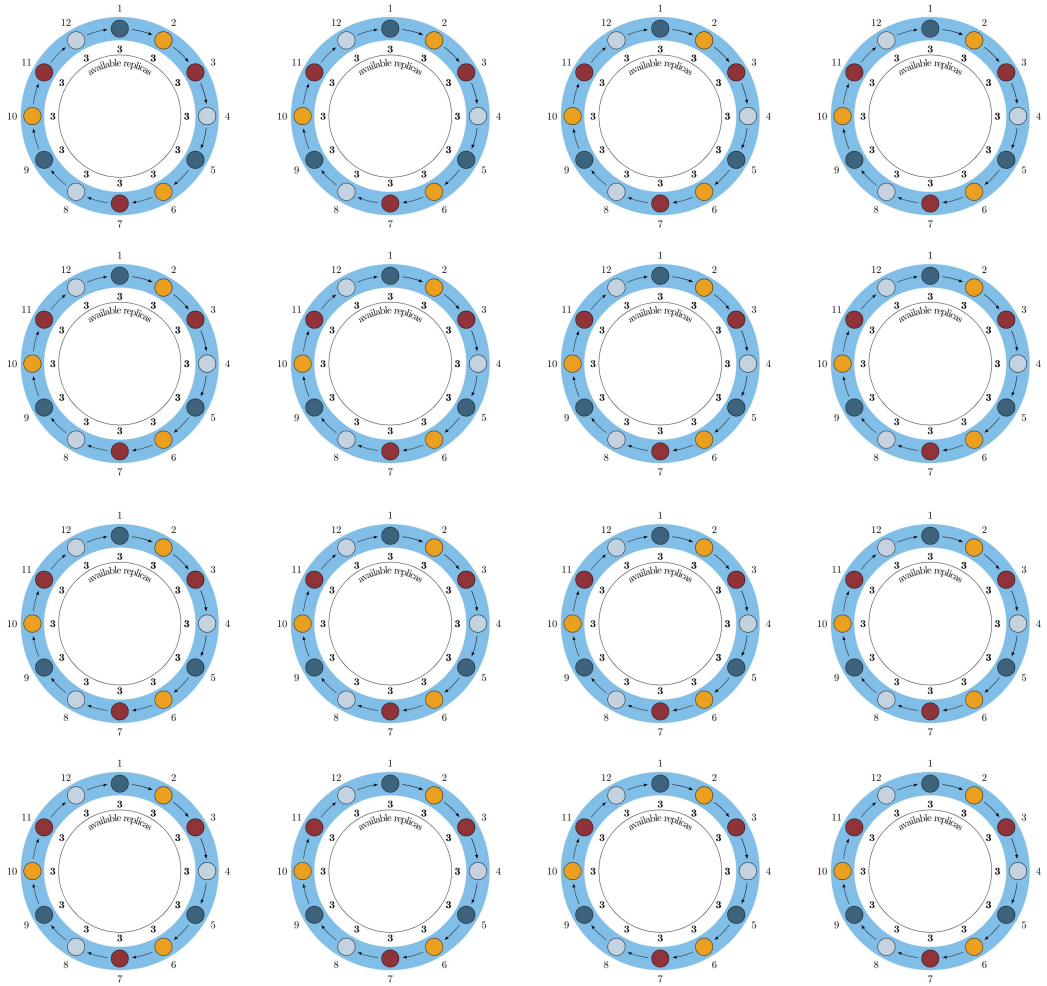


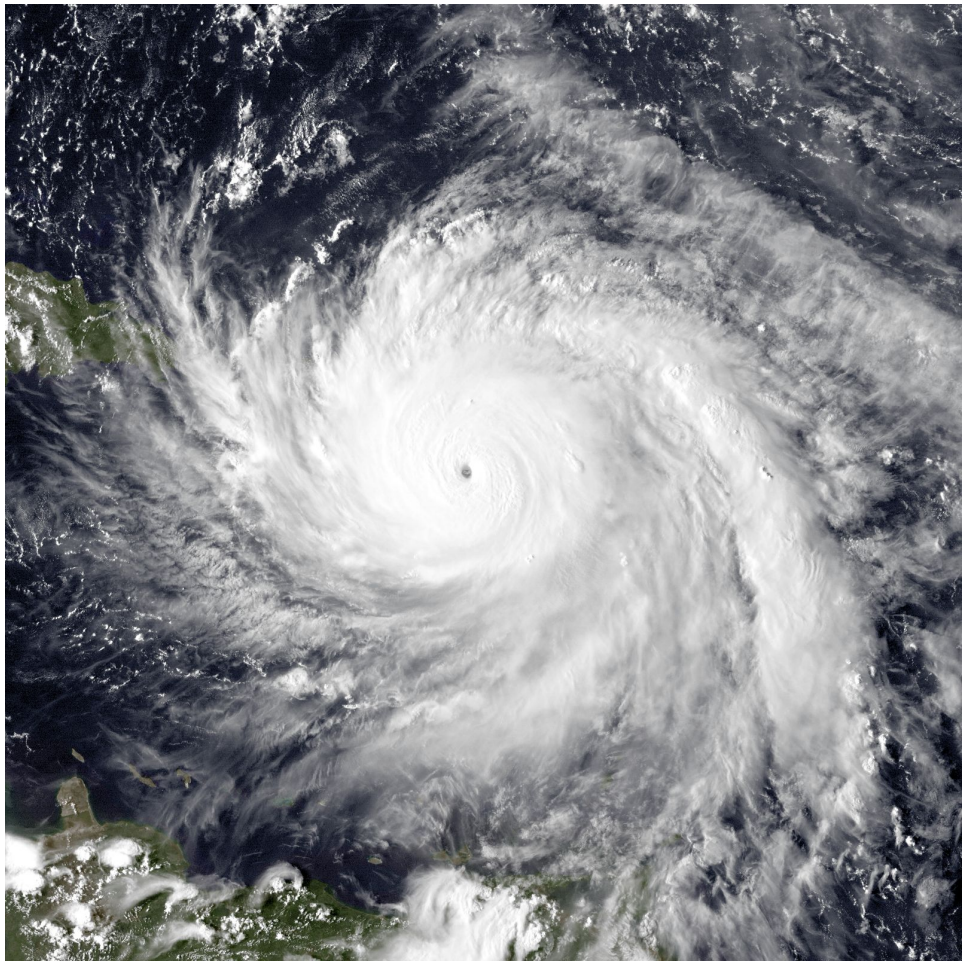
Equation+Graph from “Cassandra Availability with Virtual Nodes” by Joey Lynch and Josh Snyder

Operating C* In General



Operating C* In General





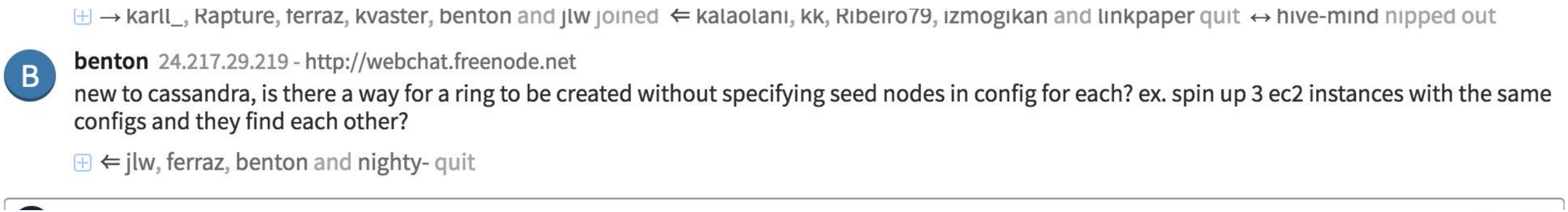
What is needed to Operate C*?

Separate solutions for ...

- Bootstrap and data movement
- Maintenance
- Configuration (files, jmx)
- Monitoring/Metrics
- Backup/Restore
- Repair

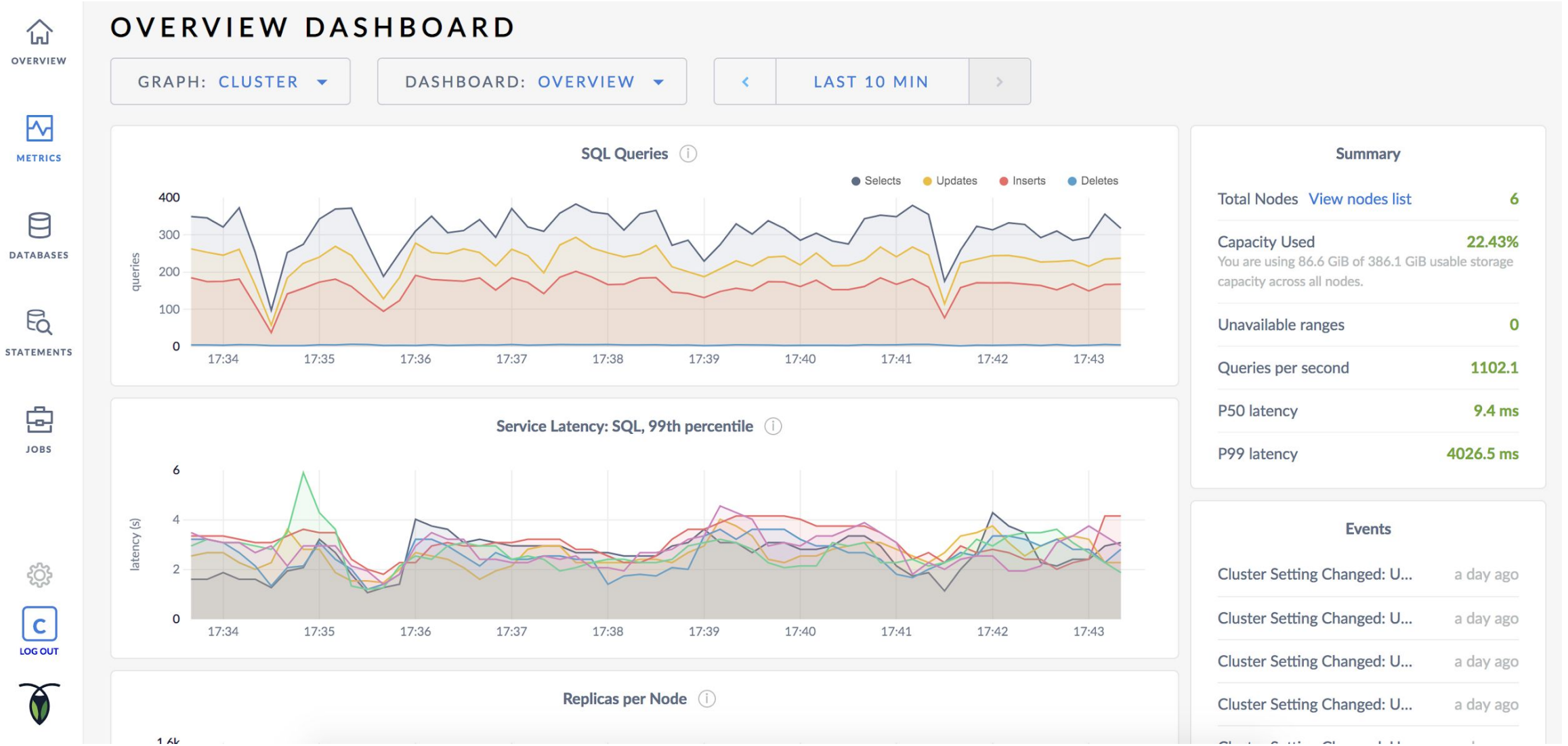
We need better tools!

Community needs

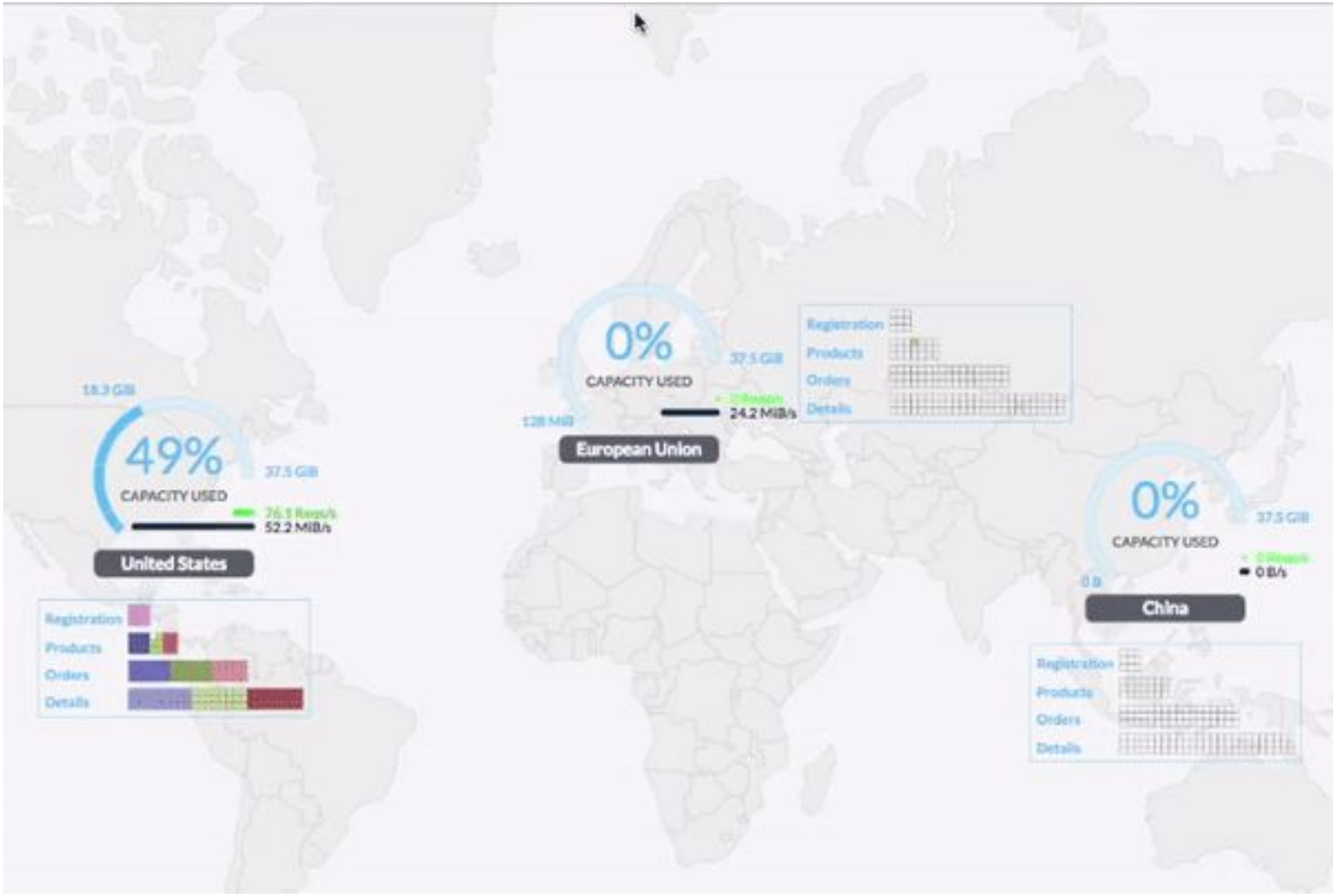


Current state of the art?

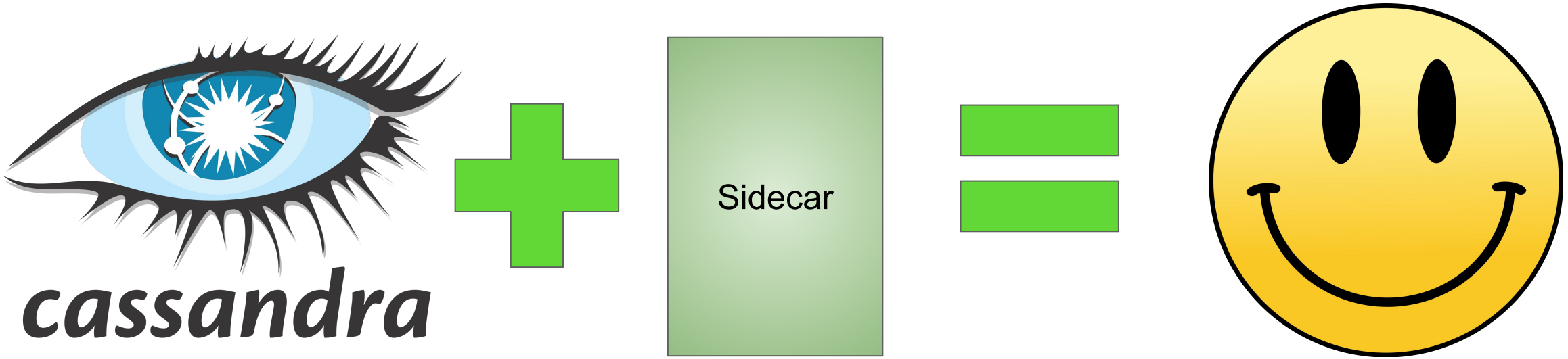
CockroachDB



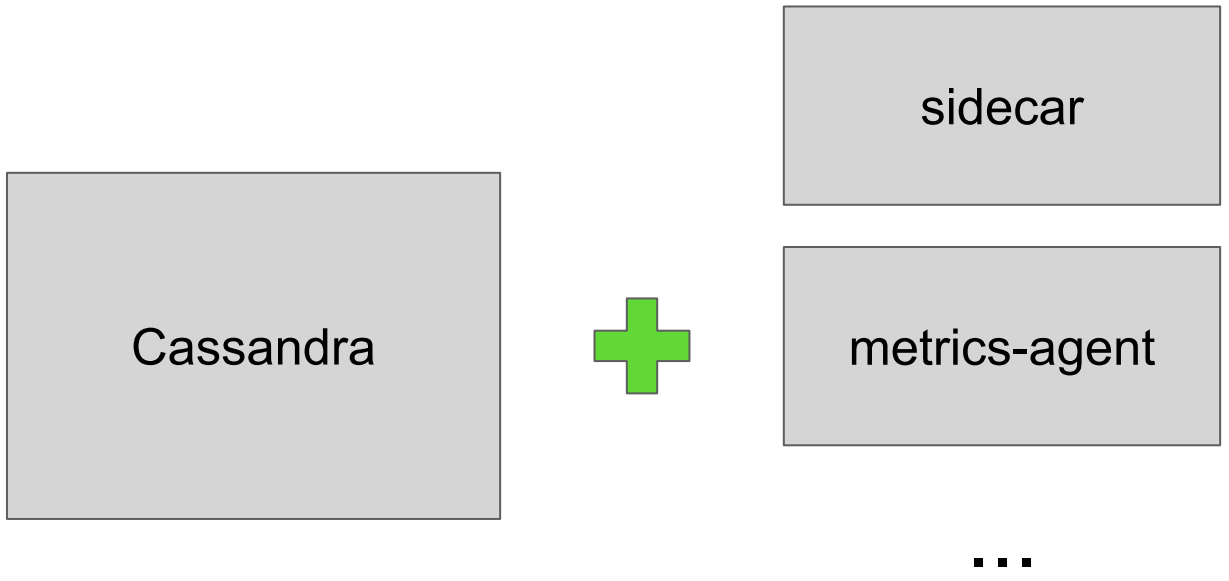
CockroachDB



Operating C* with Sidecar(s)



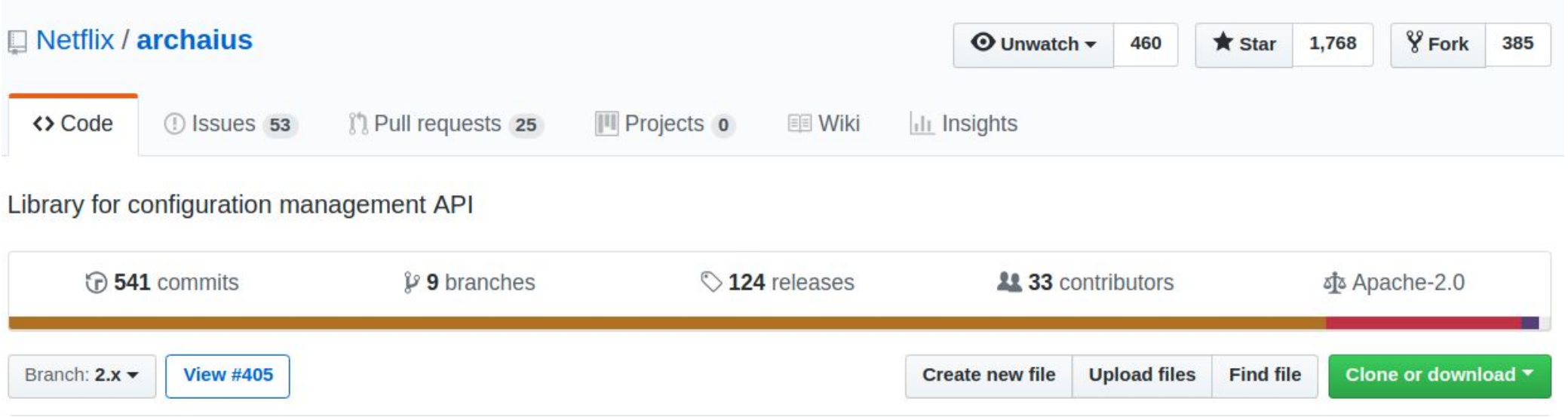
What's a Sidecar?



Sidecars Live Outside Main Daemon Scope

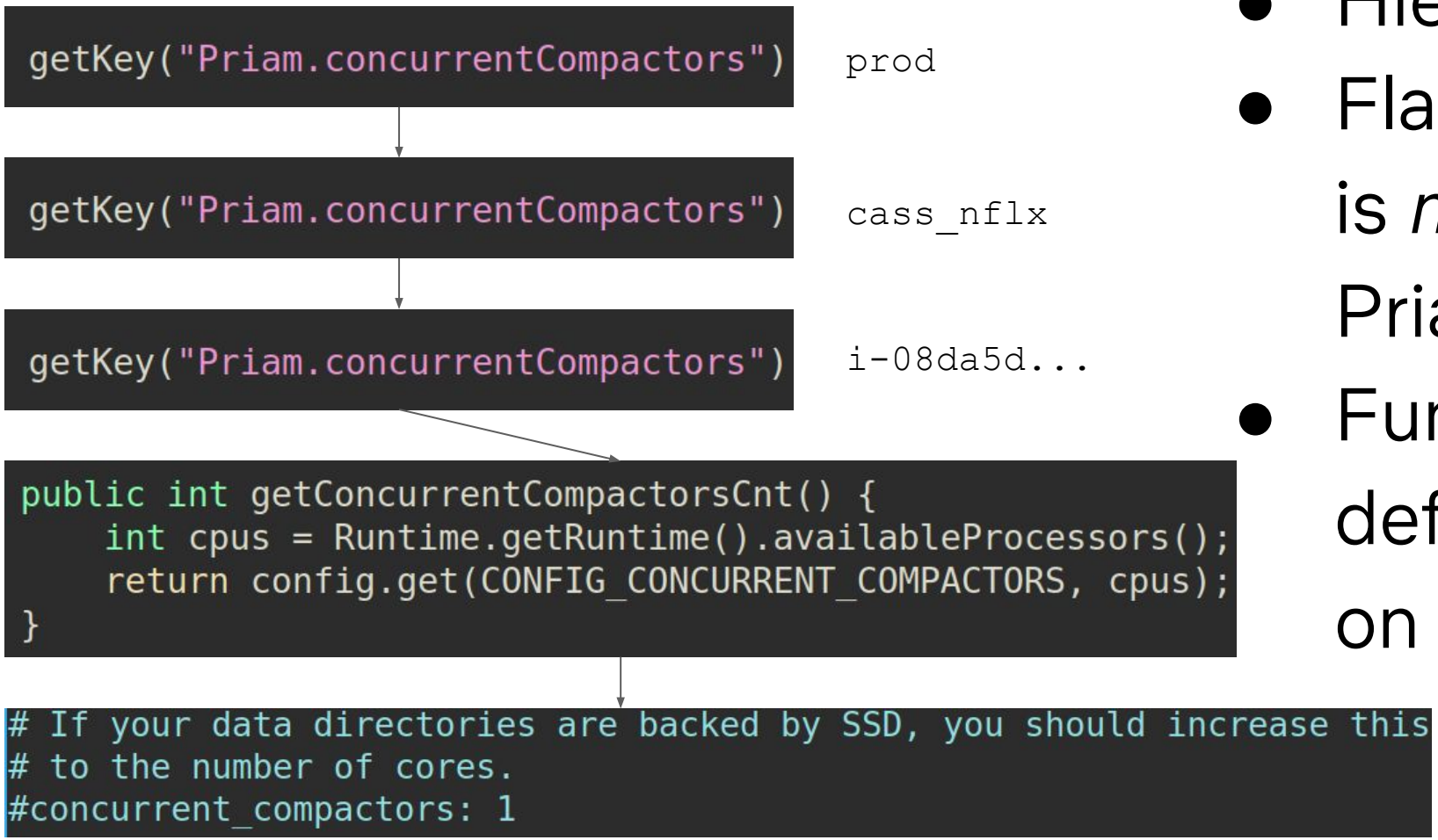
- Often built for a specific purpose
- Typically a different OS process

Sidecar: Configuration



- Hierarchy: Environment -> Cluster -> Node
- Flat namespace that is *merged* to provide Priam config

Sidecar: Configuration

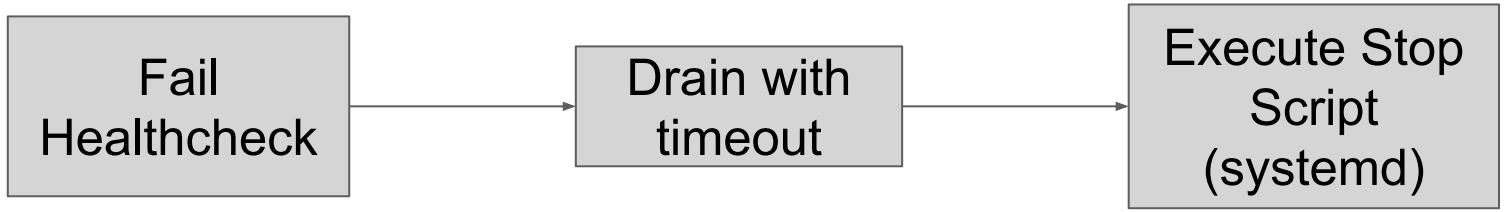


- Hierarchy
- Flat namespace that is *merged* to provide Priam config
- Functions for defaults (e.g. based on cpu)

Sidecar: Lifecycle

```
curl http://127.0.0.1:8080/Priam/REST/healthcheck
```


```
{
  "isBootstrapping": false,
  "isCassandraProcessAlive": true,
  "shouldCassandraBeAlive": true,
  "lastAttemptedStartTime": 1536170471979,
  "isGossipActive": true,
  "isNativeTransportActive": true,
  "isRequiredDirectoriesExist": true,
  "isYmlWritten": true,
  "isHealthy": true,
  "isHealthyOverride": true,
  "restoreStatus": {}
}
```


```
curl http://127.0.0.1:8080/Priam/REST/v1/cassadmin/stop
```

Sidecar: Lifecycle

```
curl http://127.0.0.1:8080/Priam/REST/v1/cassadmin/start
```

Rolling Restarts (Upgrades)

- Cluster automation is now much easier

What happens when Cassandra dies?

- Continuous health monitoring and supervision (OOM)
- Priam + systemd + jvmkill[1] == pretty good

[1] https://github.com/airlift/jvmkill

Sidecar: Maintenance

```
public abstract class IClusterManagement<T> extends Task {
```

```
/**
 * Timer to be used for compaction interval.
 *
 * @param config {@link IConfiguration} to get configuration details from priam.
 * @return the timer to be used for compaction interval  from {@link IConfiguration#getCompactionCronExpression()}
 */
public static TaskTimer getTimer(IConfiguration config) throws Exception {
    return CronTimer.getCronTimer(Task.COMPACTION.name(), config.getCompactionCronExpression());
}
```

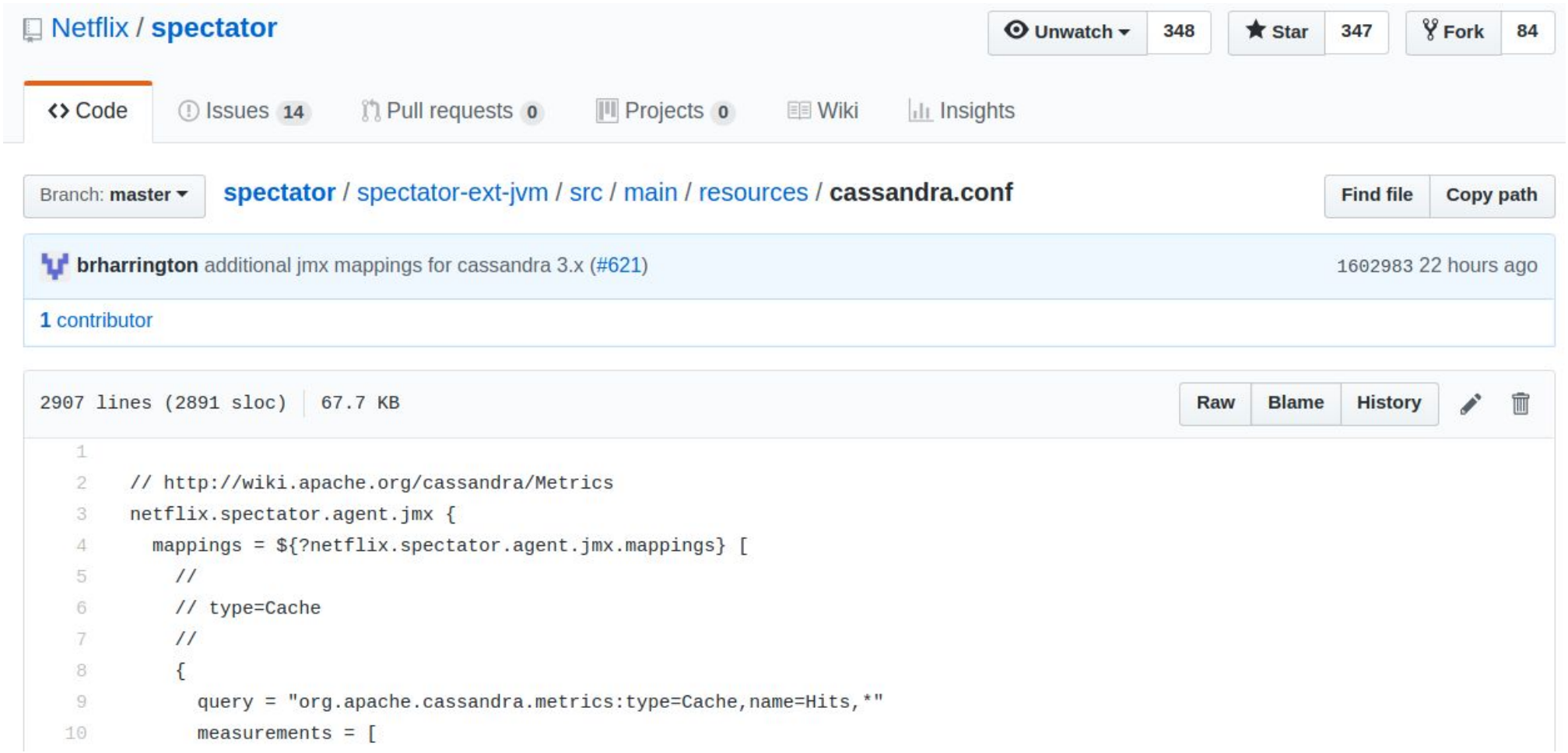
- JMX methods on cron
- Can add arbitrary tasks like compactions, flushes, etc

Sidecar: Maintenance

```
$> grep "Path(" CassandraAdmin.java
@Path("/v1/cassadmin")
    @Path("/start")
    @Path("/stop")
    @Path("/refresh")
    @Path("/info")
    @Path("/partitioner")
    @Path("/ring/{id}")
    @Path("/flush")
    @Path("/compact")
    @Path("/cleanup")
    @Path("/repair")
    @Path("/version")
    @Path("/disablegossip")
    @Path("/enablegossip")
    @Path("/gossipinfo")
    @Path("/move")
    @Path("/drain")
    @Path("/decompress")
```

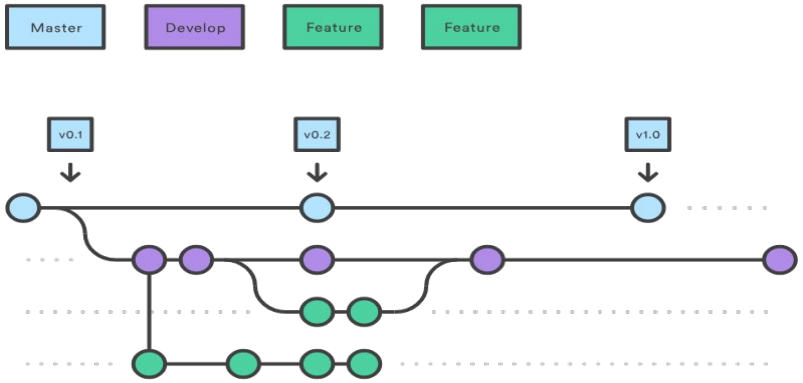
- Sidecar provides JMX over HTTP
  - Cleanup
  - Invoke complex JMX methods using curl
  - Many of these are better done scheduled (e.g. repair, compaction, flushes, etc)

Sidecar: Monitoring

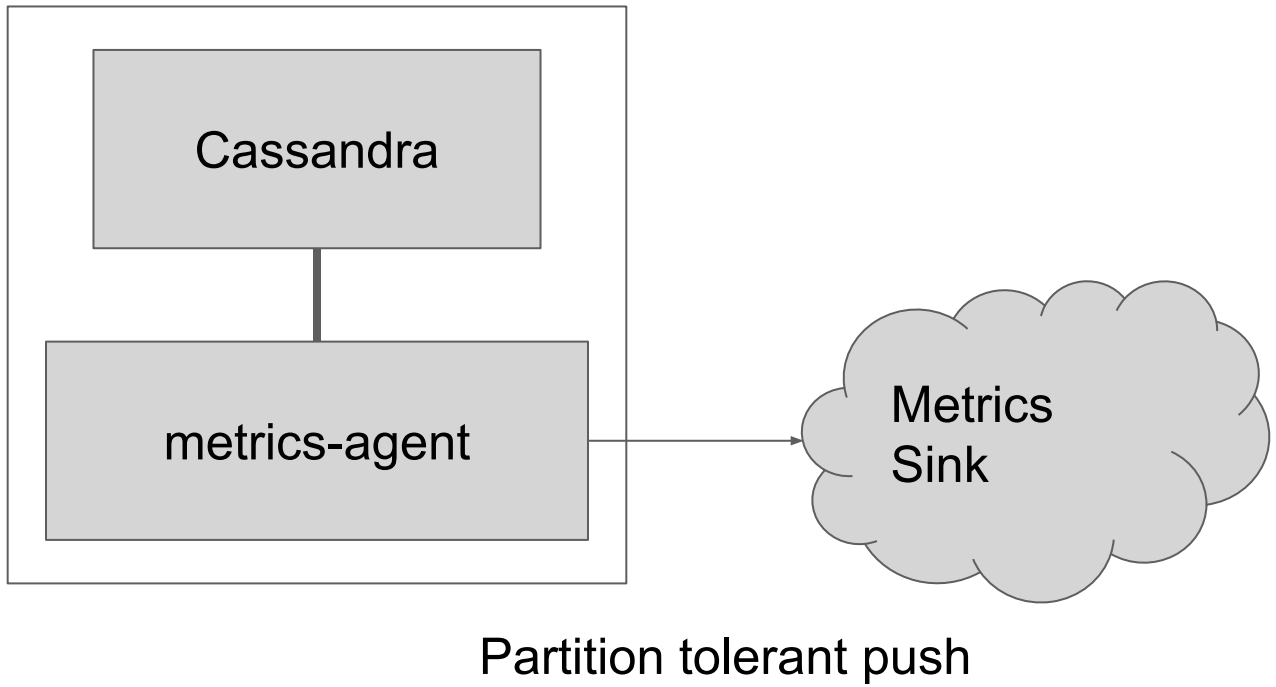


Lessons learned from Existing sidecars

- Specific to deployments
- Not easy to use externally
- Multi-cloud?
- ~3 different implementations of everything
- Version compatibility



Sidecar: Monitoring

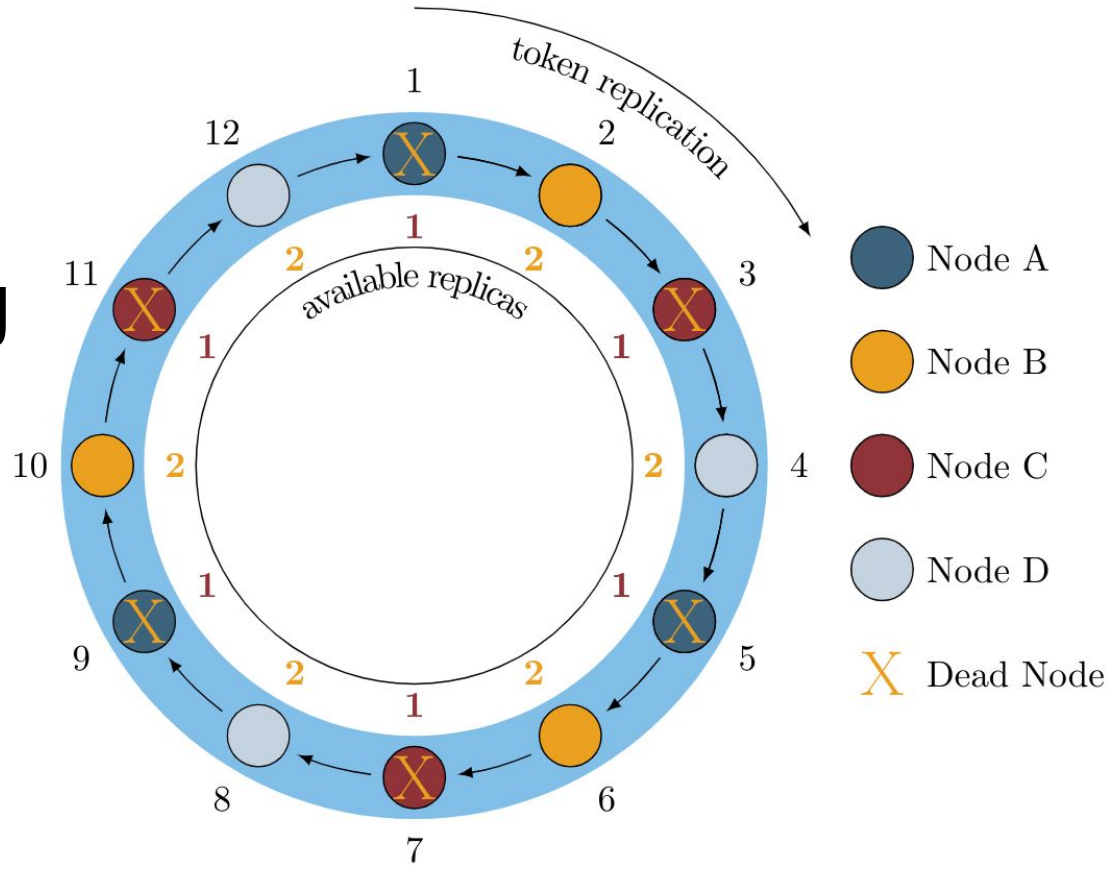


Partition tolerant push

- Sidecar agent = scalable
- Non agent JMX metric export … not even once
- Still a sidecar, just not Priam

Sidecar: Ring Health

- Sidecar monitoring local node
- Can also look at the ring
  - Gossip state
- Can
  - Ask a few Priams
  - Export info to streaming system



Sidecar: Backup/Restore

```
$> ls backup
AbstractBackup.java             BackupStatusMgr.java            IBackupStatusMgr.java     parallel
AbstractBackupPath.java         BackupVerification.java         IFileSystemContext.java   RangeReadInputStream.java
BackupFileSystemAdapter.java    BackupVerificationResult.java   IIncrementalBackup.java   SnapshotBackup.java
BackupFileSystemContext.java    CommitLogBackup.java            IMessageObserver.java     Status.java
BackupMetadata.java             CommitLogBackupTask.java        IncrementalBackup.java
BackupRestoreException.java     FileSnapshotStatusMgr.java      IncrementalMetaData.java
BackupRestoreUtil.java          IBackupFileSystem.java          MetaData.java
```

- Backup on cron schedule
  - Snapshot
- Parallel incrementals
- *Verification*
- Pluggable with code (S3, GCS, etc…)

Sidecar: Backup/Restore

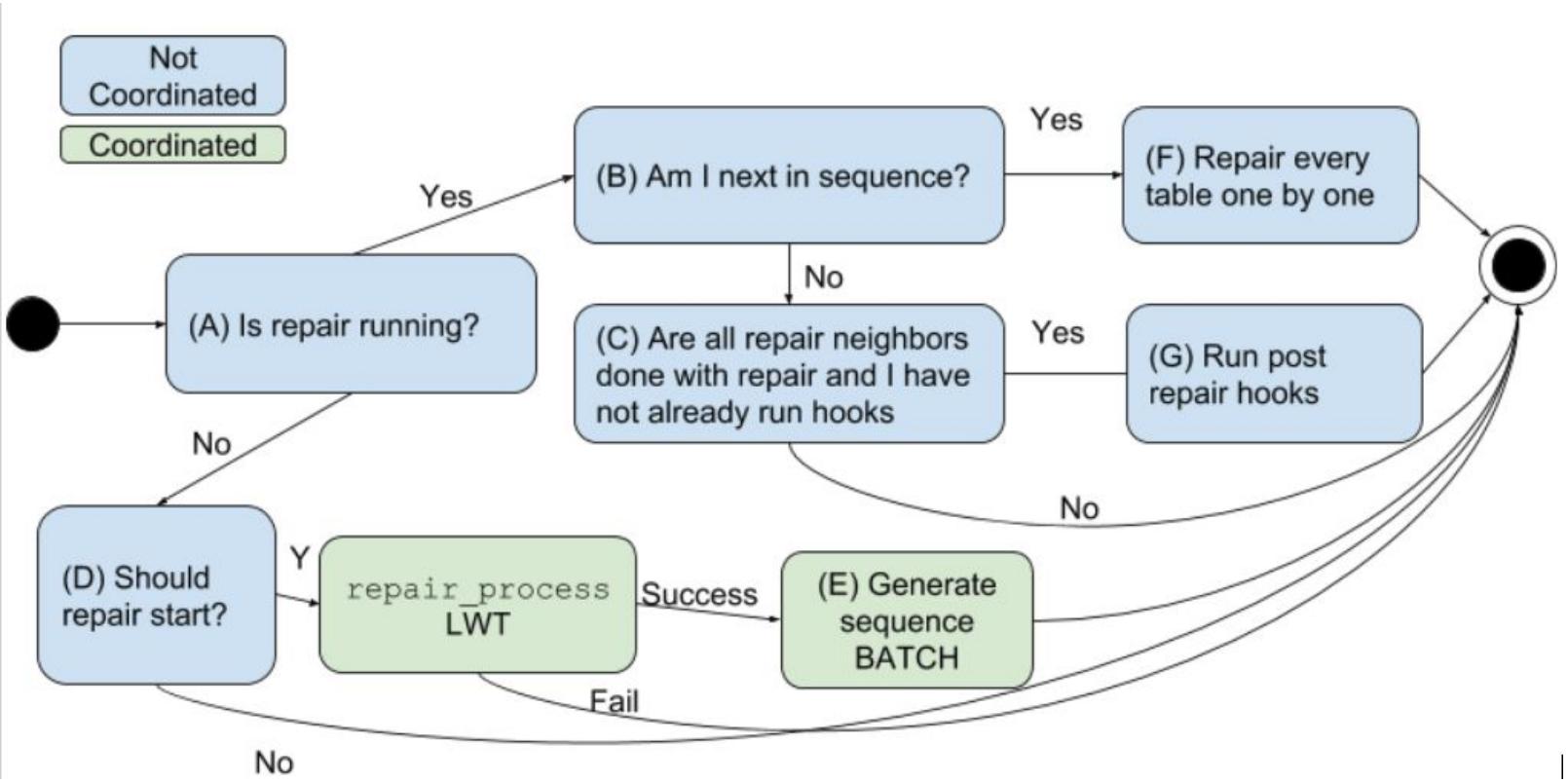
```
$> cd backupv2
$> ls
ColumnfamilyResult.java  MetaFileInfo.java     MetaFileReader.java         PrefixGenerator.java
FileUploadResult.java    MetaFileManager.java  MetaFileWriterBuilder.java
```

- Similar to cassandra-mirror[1]
- Keeps track of what files have already been uploaded
  - Unlocks *minute* level snapshot backups
  - Only uploads files once
- Continuous, point in time, self healing, eventually consistent

1: https://github.com/hashbrowncipher/cassandra-mirror

Sidecar: Repair

- Always on
- Just works
- In Cassandra itself? [1]



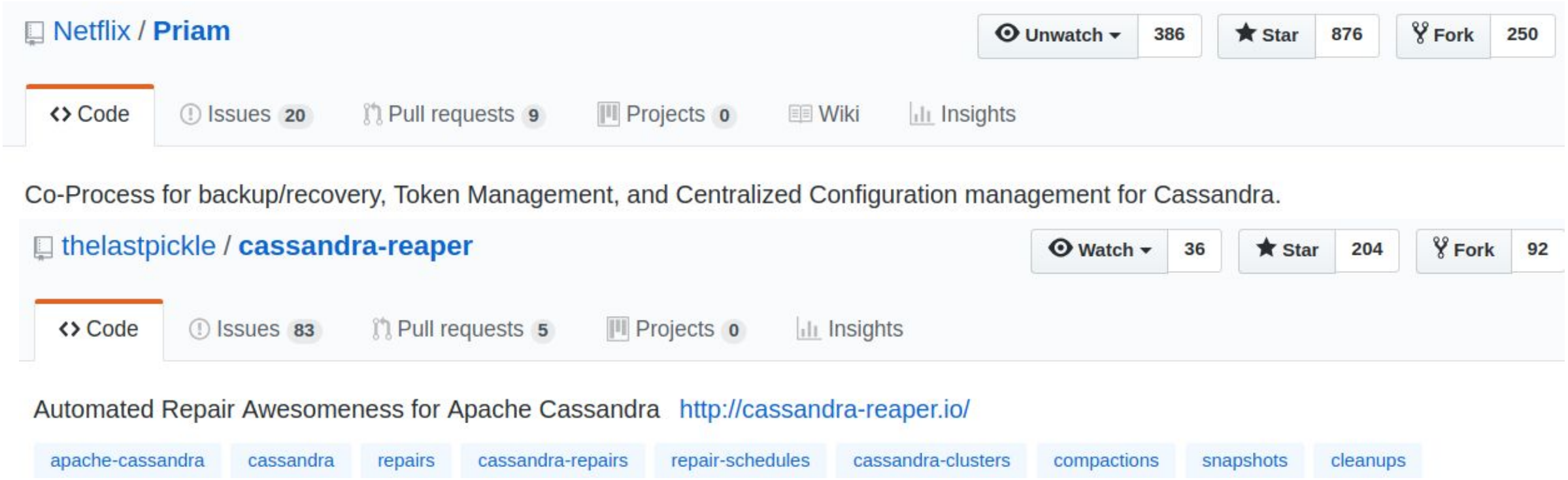
1: https://issues.apache.org/jira/browse/CASSANDRA-14346

State of Community Sidecars

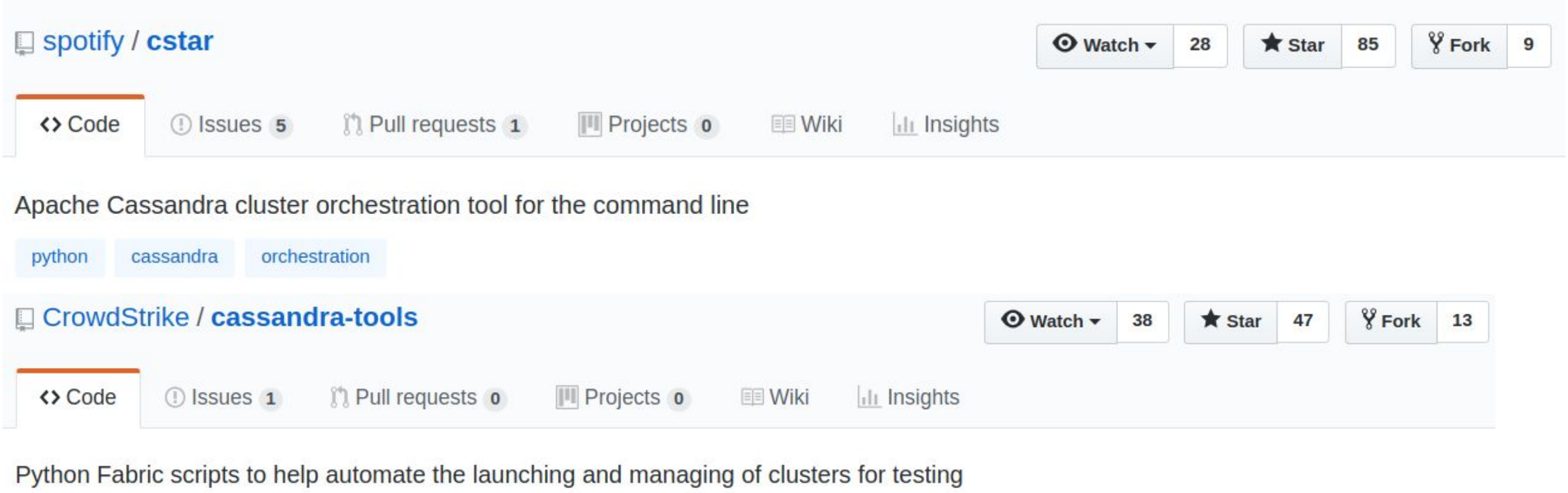
State of Community Sidecars

Everyone builds their own

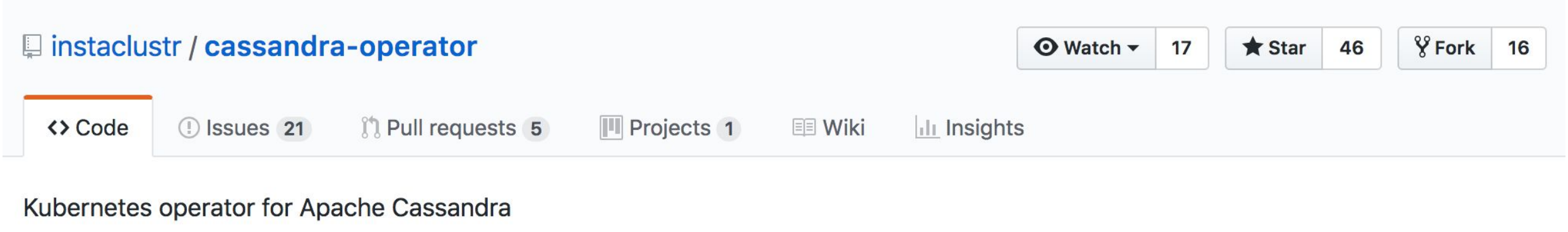
State of Community Sidecars

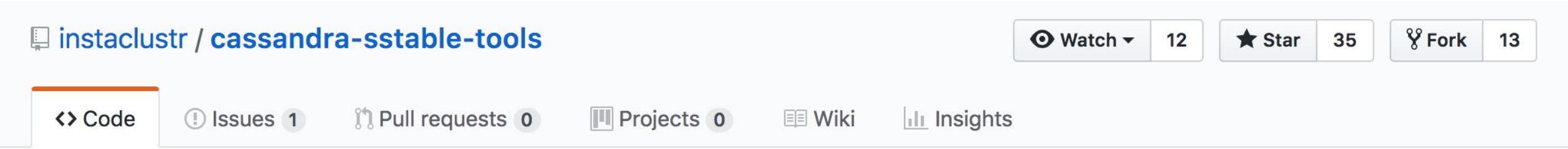


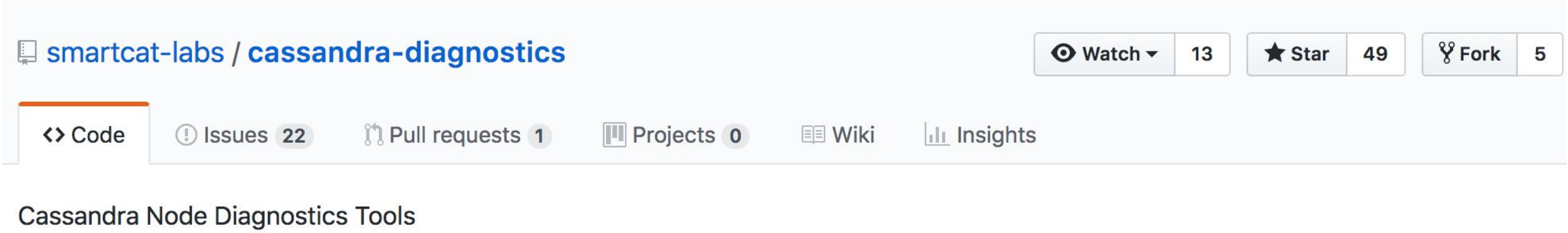
State of Community Tooling



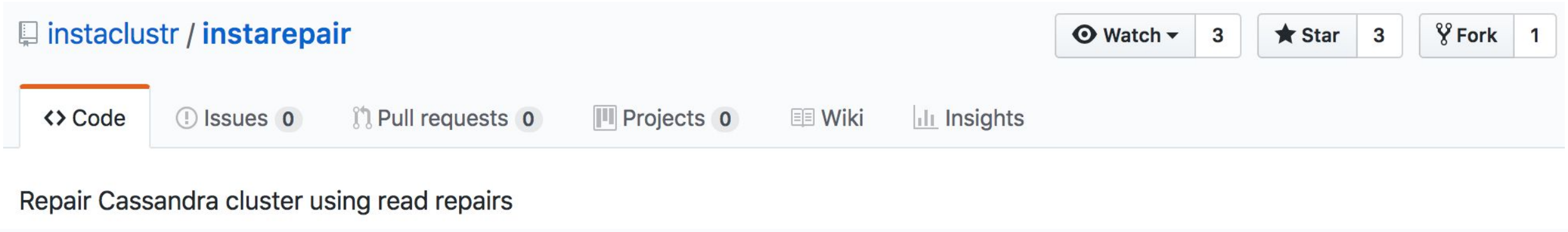
State of Community Tooling 2

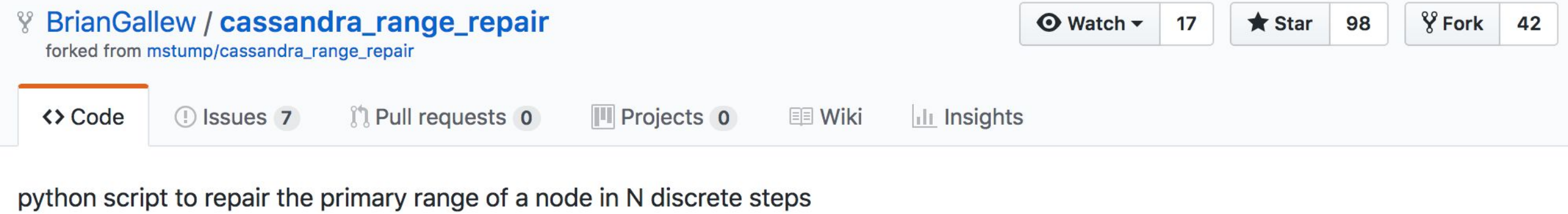




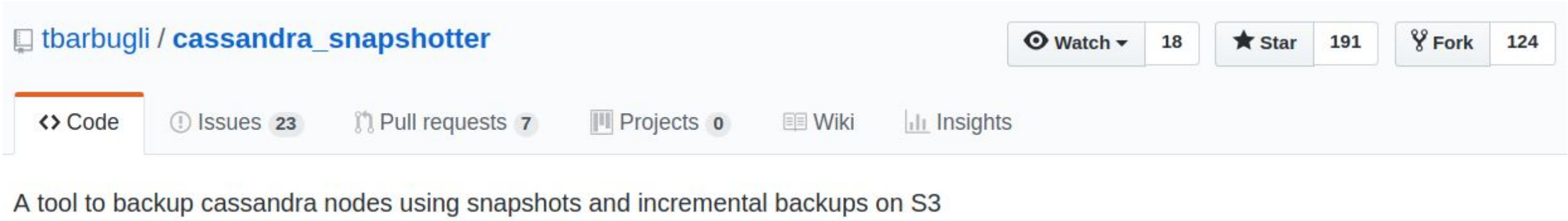


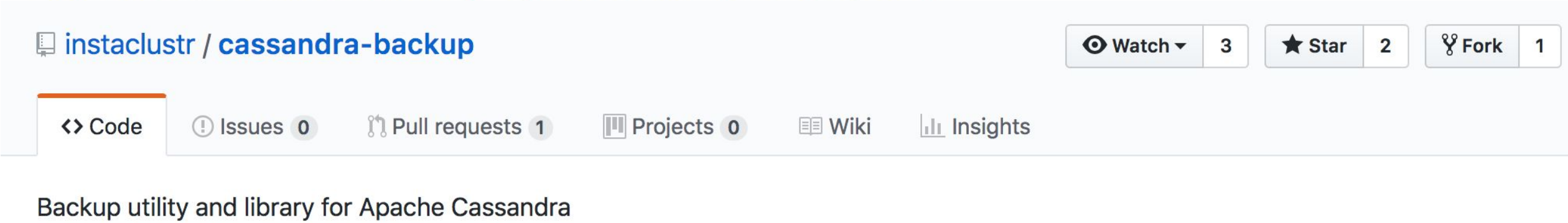
State of Community Tooling 3

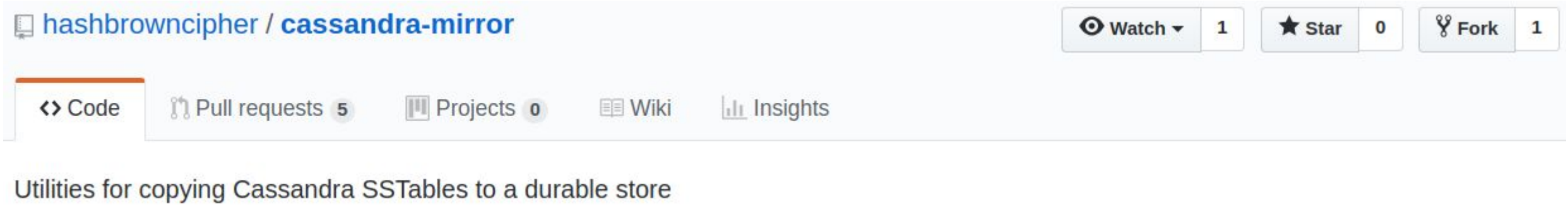




State of Community Backup







And that’s not even all of ‘em … but do they solve our problems?

	Puppet/ Chef	Terraform	CRON	Fabric / SSH in a for loop	Reaper	Priam	Jenkins	Sensu/ Nagios
Bootstrap	No	Yes	No	Maybe	No	Yes	Maybe	No
Maintenance	Yes	Maybe	Maybe	Yes	No	Yes	Yes	No
Configuration	Yes	Maybe	No	Maybe	No	Yes	Maybe	No
Monitoring	Maybe	No	Maybe	Maybe	No	Yes	Maybe	Yes
Backup	Maybe	No	Maybe	Maybe	Maybe	Yes	Yes	No
Repair	No	No	Sorta	Yes	Yes	Soon	Yes	No
Easy to Use	Yes	No*	Yes	Yes	Yes	No	Maybe	Yes
Multi-Cloud	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes
Does it Scale	Yes	Yes	Yes	Sorta	Yes*	Yes	Sorta*	Yes

Oops.

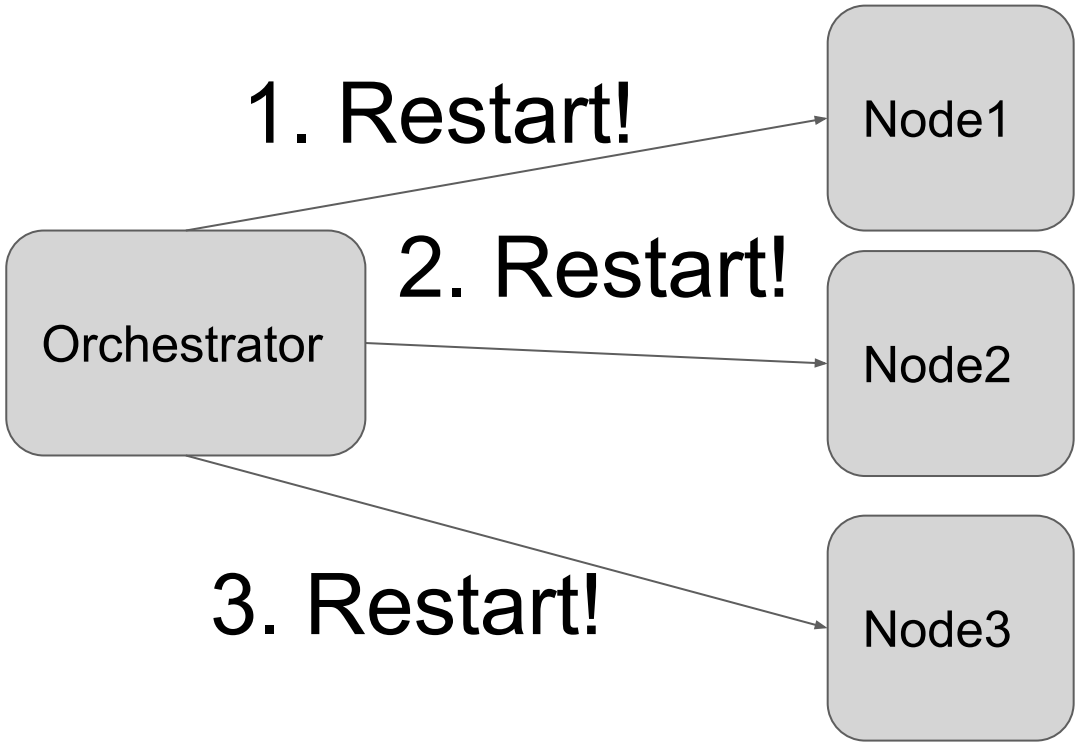
Lessons Learned

Community has operated Cassandra using

aforementioned sidecars for **years**

… We have learned a few things

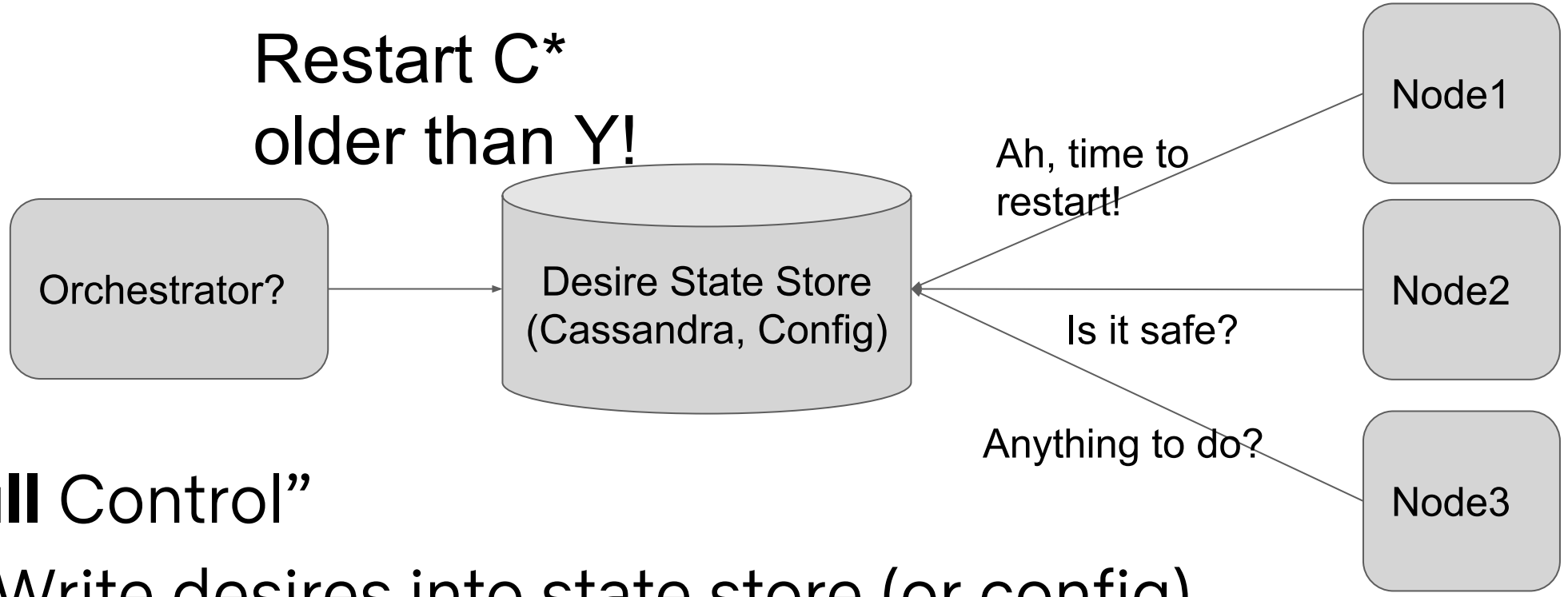
Lesson Learned: Control Plane



"Push Control"

- Relies on unreliable communication (ssh < http)
- Hard to make eventually consistent
- Hard to guarantee safety
- Very **easy** to implement

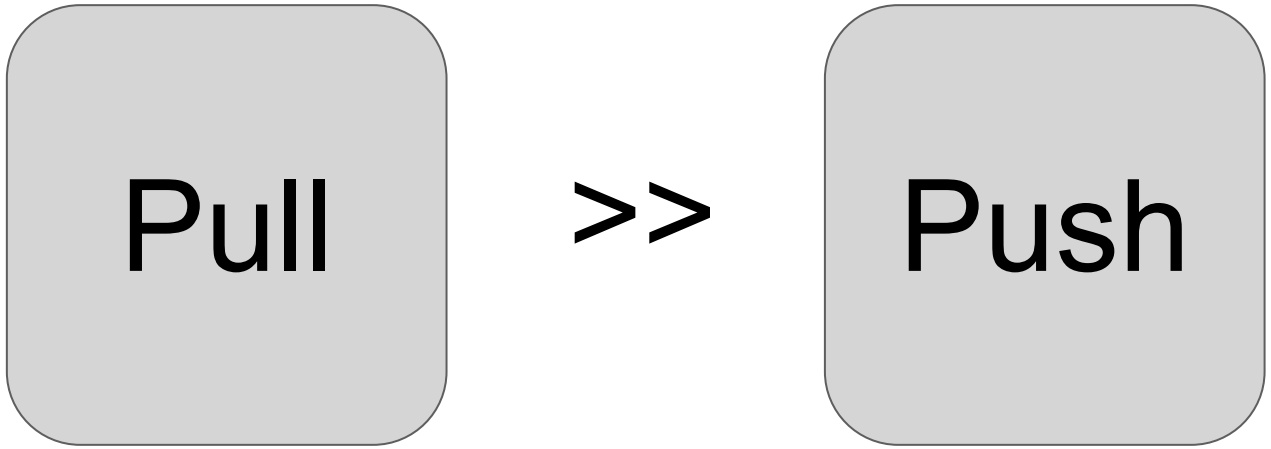
Lesson Learned: Control Plane



"**Pull** Control"

- Write desires into state store (or config)
- Nodes coordinate through state store as needed
- Tradeoff: Hard to implement

Lesson Learned: Control Plane



Goals of C* management process

1. Easy to Use

Goals of C* management process

2. Solves or eases most common problems

Goals of C* management process

3. Pluggable

Goals of C* management process

4. Scalable

C* Management Process design

Unopinionated,
pluggable & extensible

More info?

Management Process:
https://issues.apache.org/jira/browse/CASSANDRA-14395

apache / cassandra-sidecar

Code | Pull requests 1 | Actions | Security | Insights

Sidecar for Apache Cassandra https://cassandra.apache.org/

database java cassandra

7 commits **1** branch **0** releases

Branch: **master** | **New pull request** | **Create**

tolbertam and **dineshjoshi** Read config from sidecar.config System Property instead of classpath

Can it scale?

Currently deployed in Netflix ecosystem on thousands of

C* nodes

What else is coming?

- Bulk command tools
- More @ Cassandra CIP-1
- CDC processor?

Thank You.