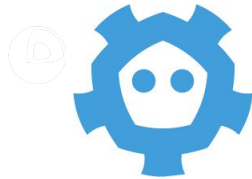
etcd - overview and future

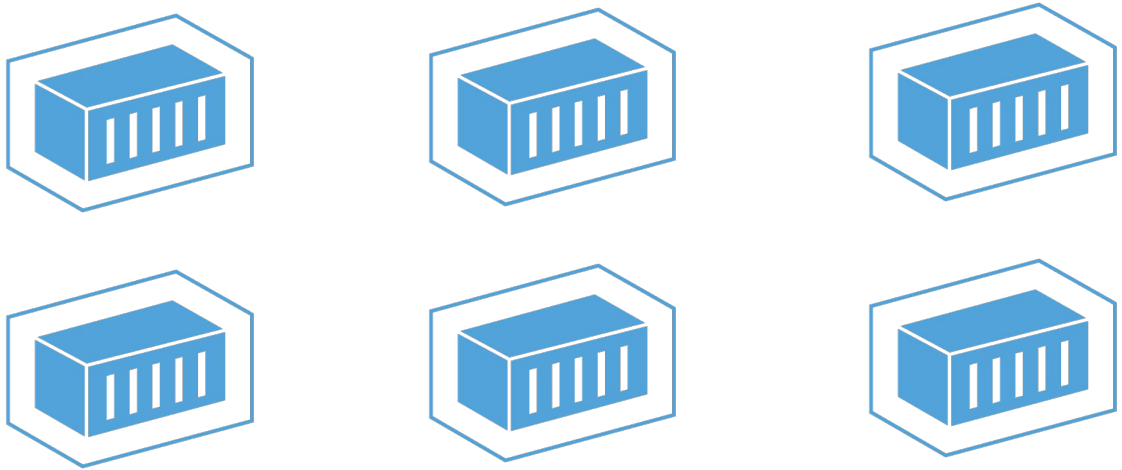
Jonathan Boulle

@baronboulle | jonathan.boulle@coreos.com

Why etcd?

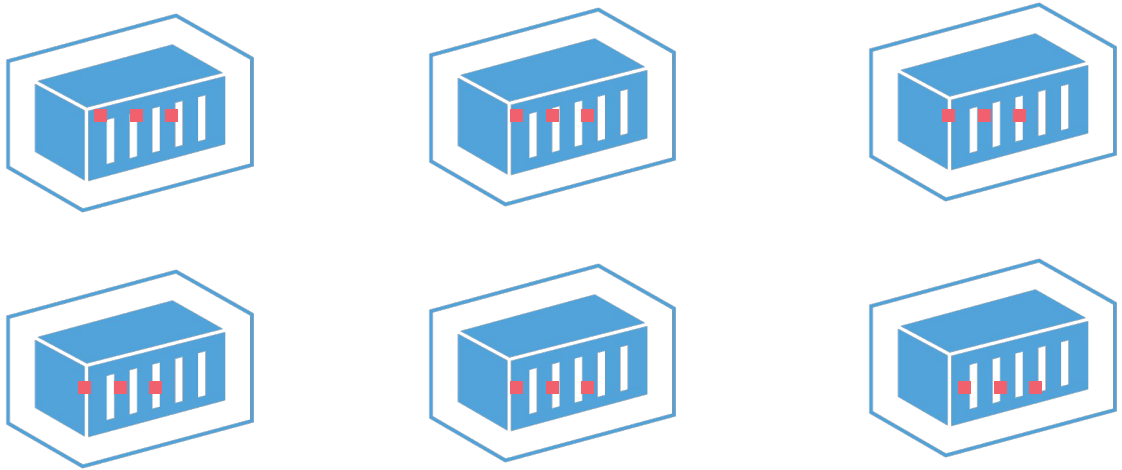


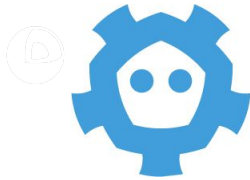
Uncoordinated Upgrades



Uncoordinated Upgrades

Unavailable



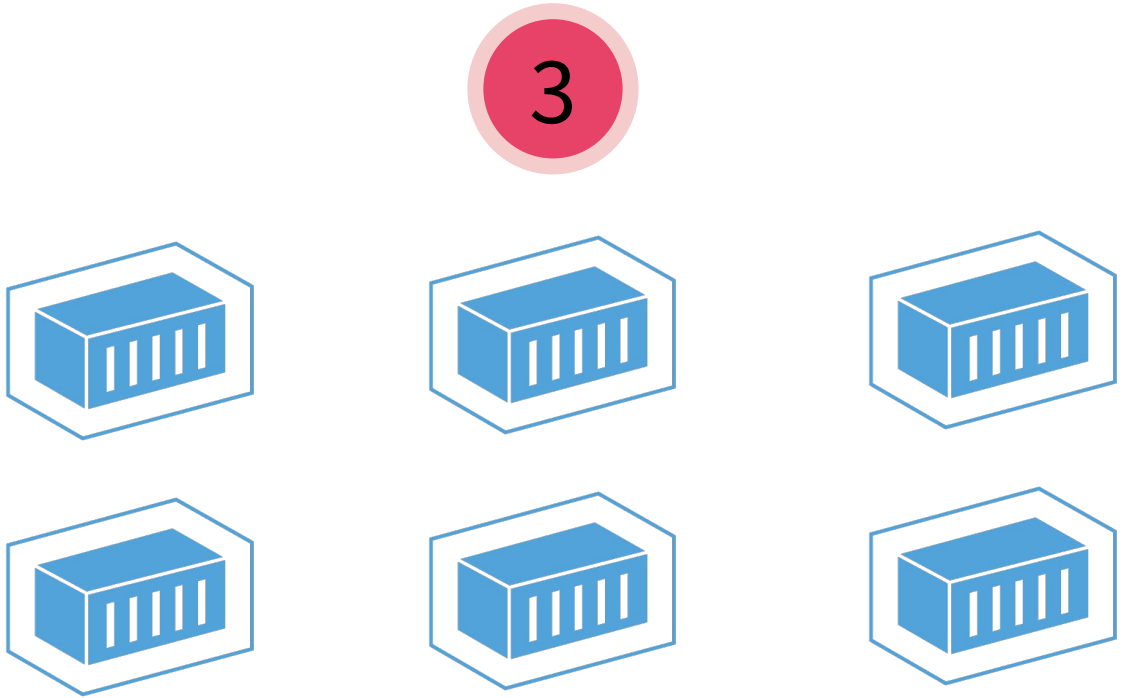


Motivation

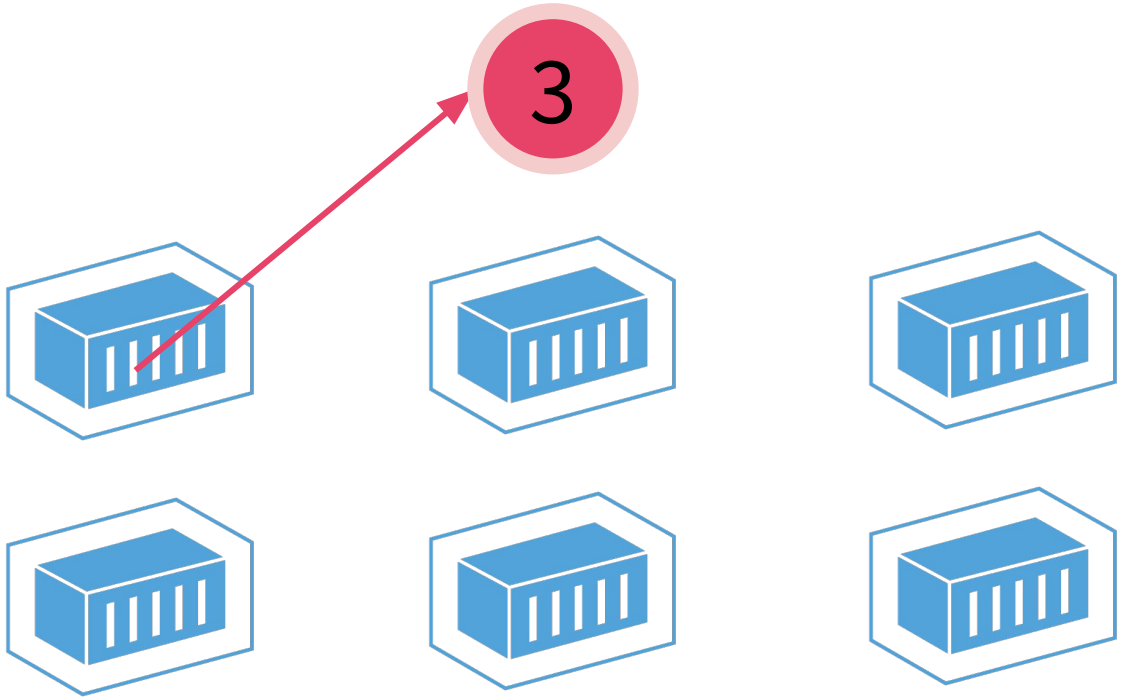
CoreOS cluster reboot lock

- Decrement a semaphore key atomically
- Reboot and wait...
- After reboot increment the semaphore key

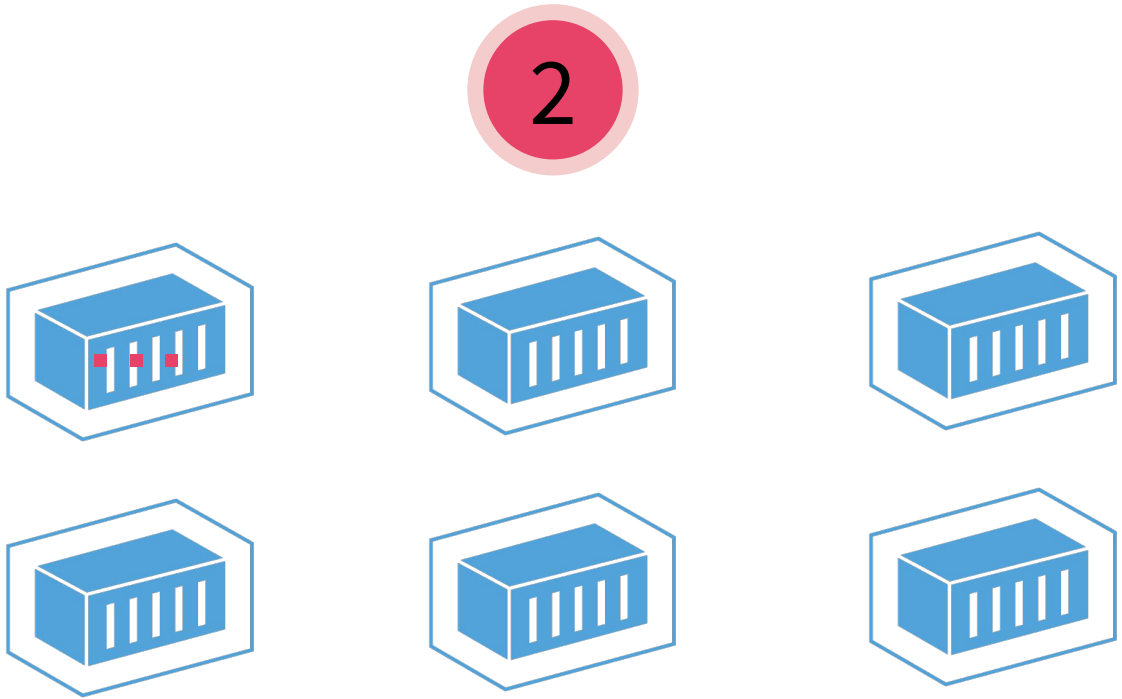
CoreOS updates coordination



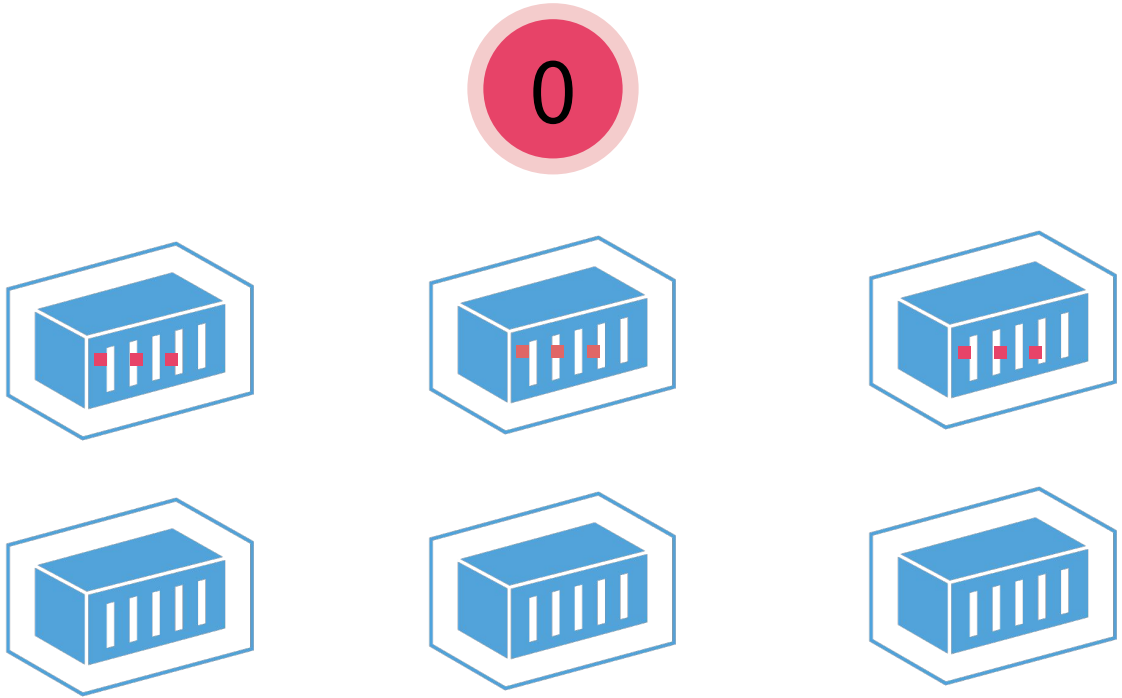
CoreOS updates coordination



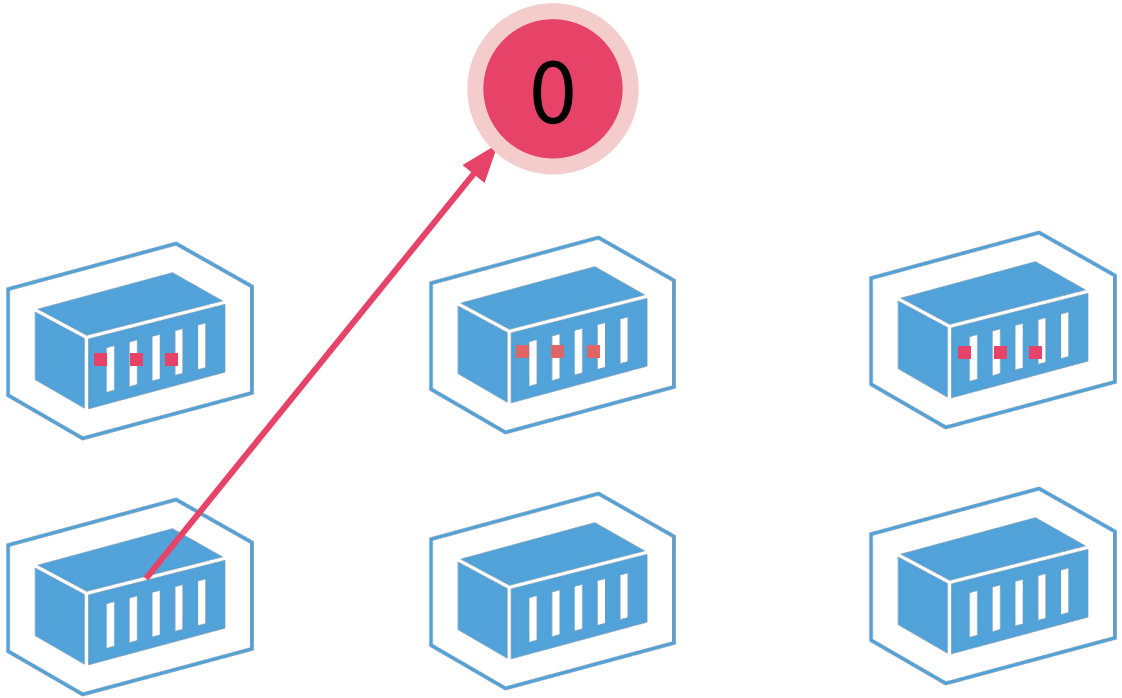
CoreOS updates coordination



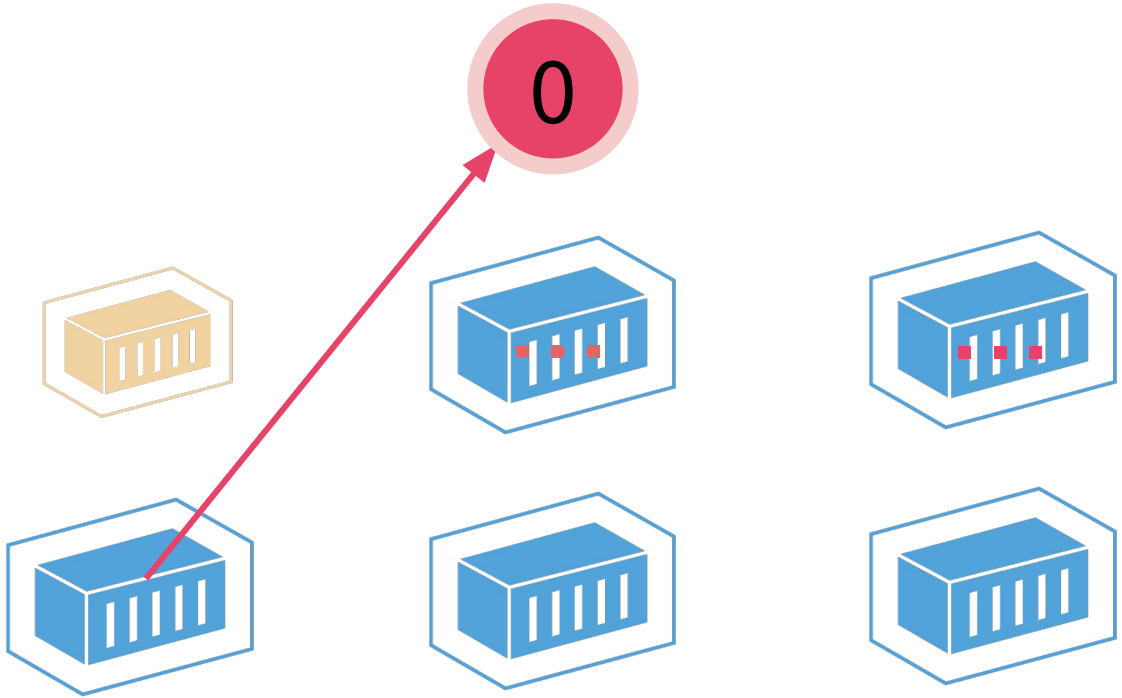
CoreOS updates coordination



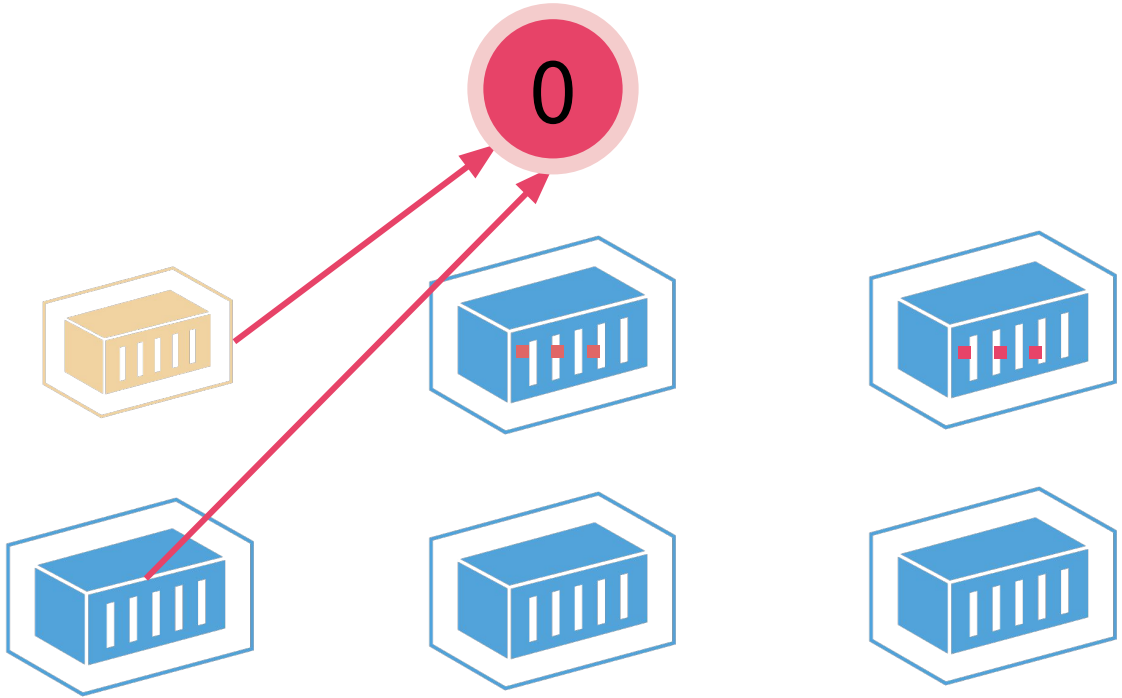
CoreOS updates coordination



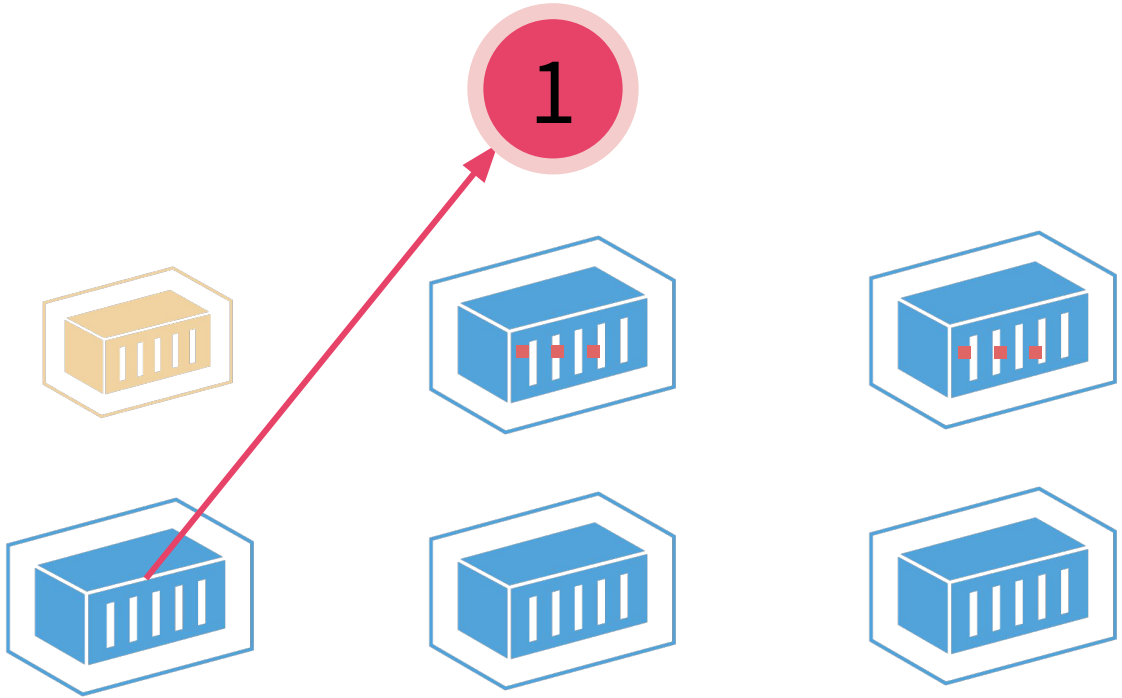
CoreOS updates coordination



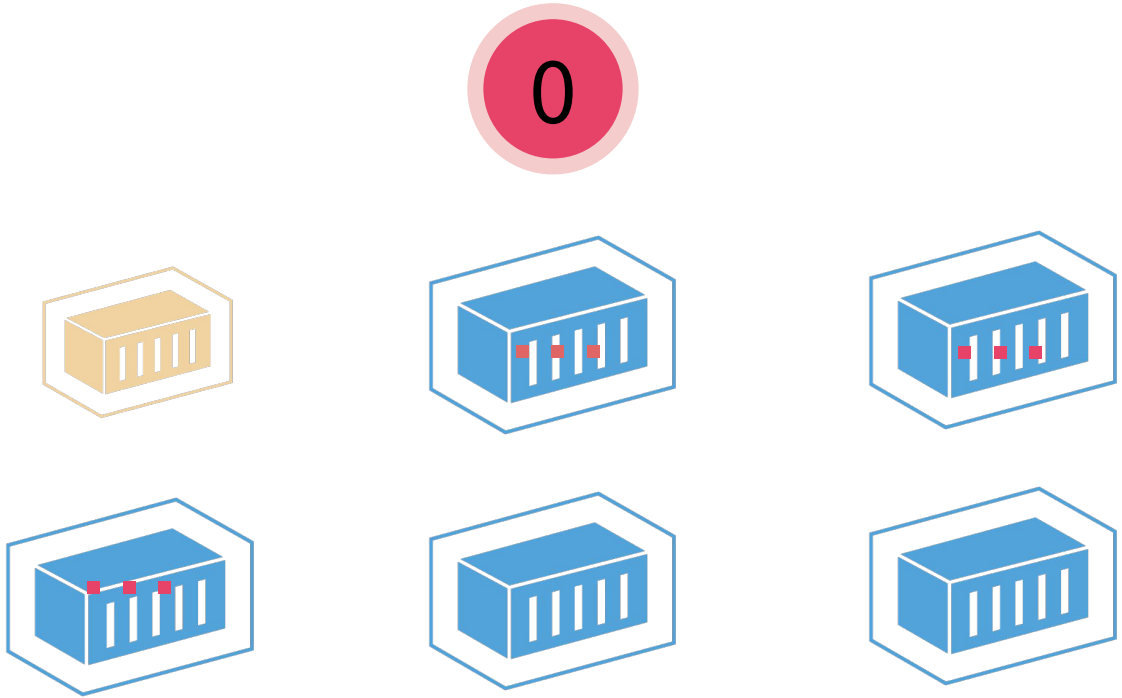
CoreOS updates coordination



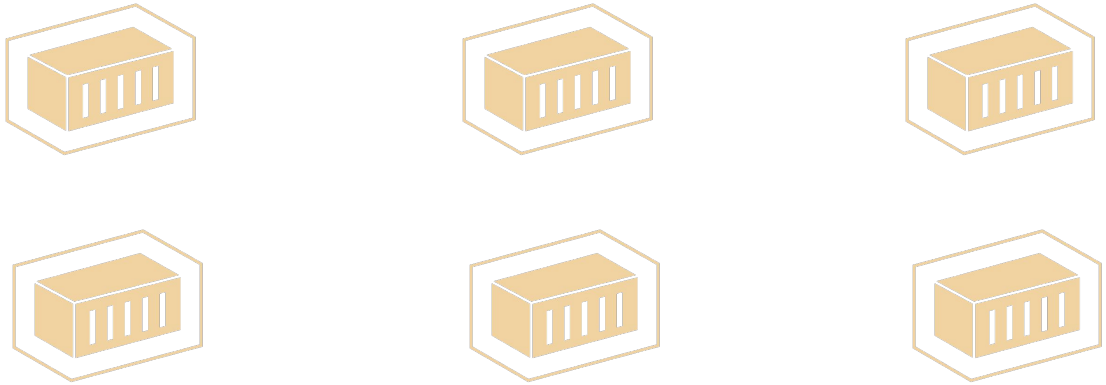
CoreOS updates coordination

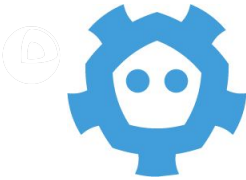


CoreOS updates coordination

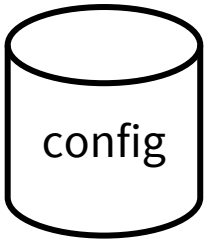


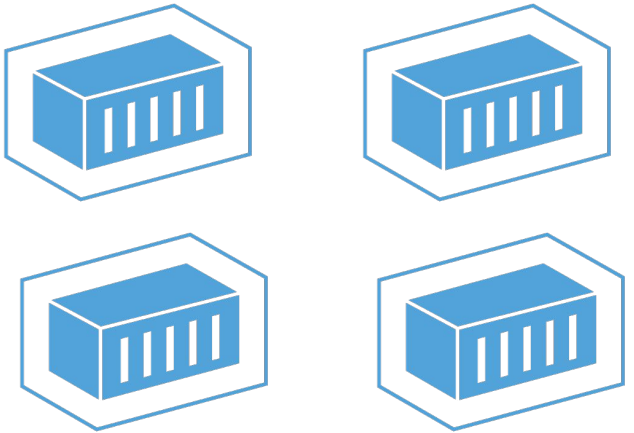
CoreOS updates coordination

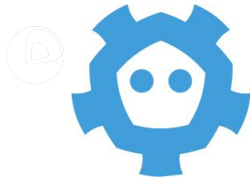




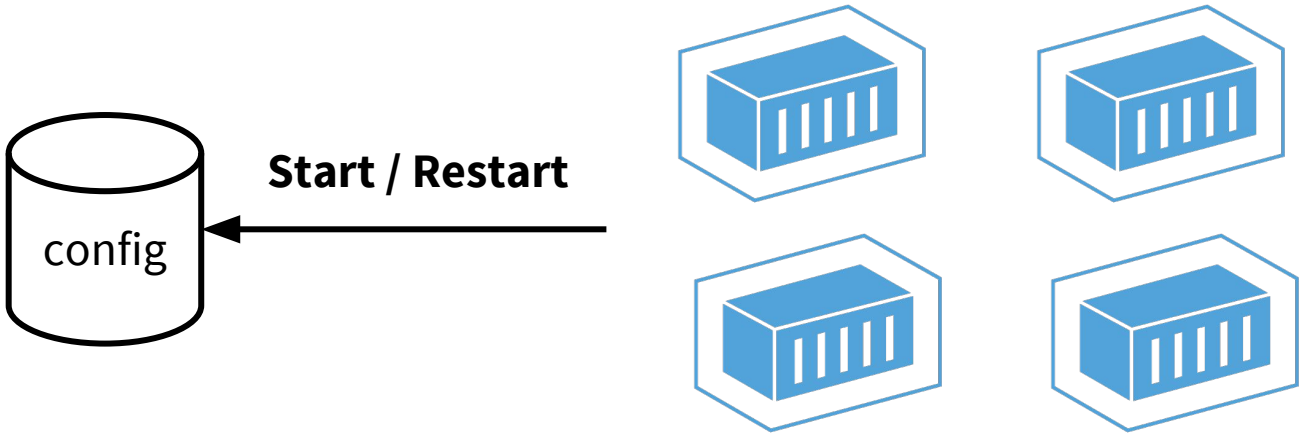
Store Application Configuration



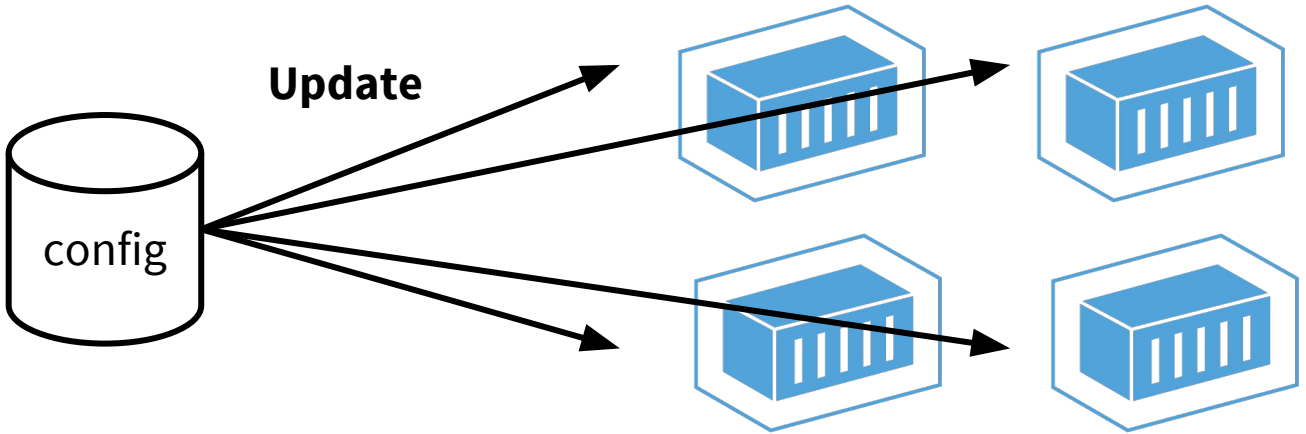




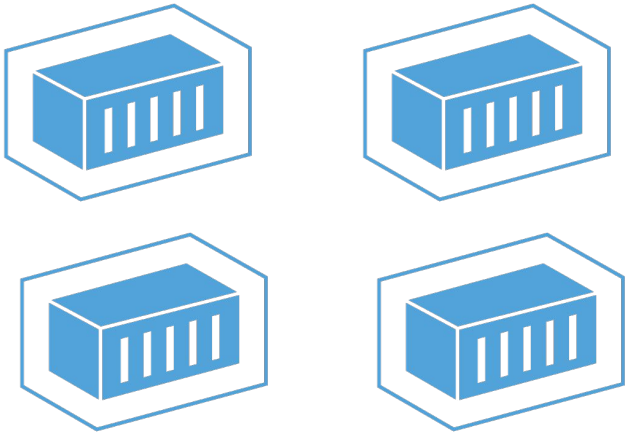
Store Application Configuration

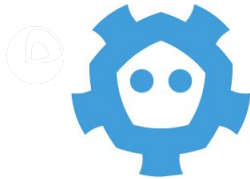


Store Application Configuration



Store Application Configuration





Requirements

Strong Consistency

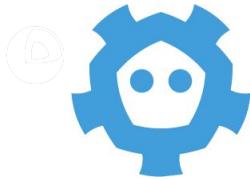
- mutual exclusive at any time for locking purpose

Highly Available

- resilient to single points of failure & network partitions

Watchable

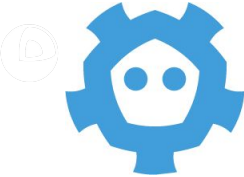
- push configuration updates to application



Requirements

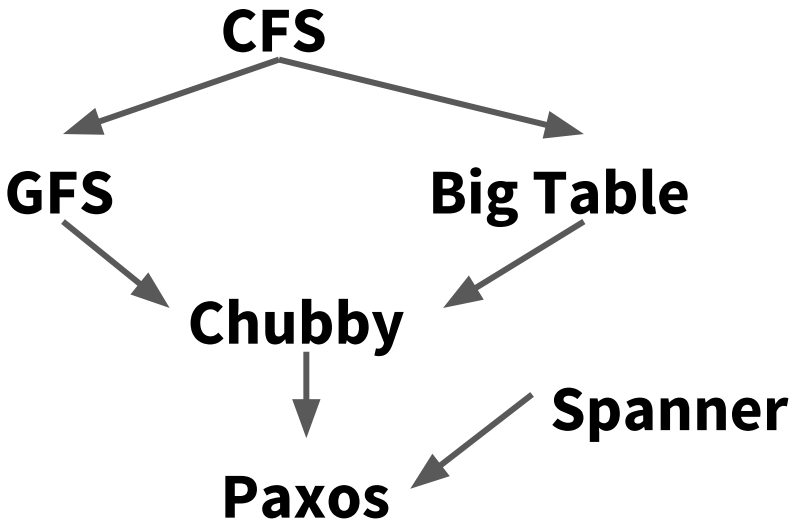
CAP

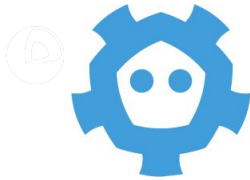
- Consistency, Availability, Partition Tolerance: choose 2
- We want CP
- We want something like Paxos



Common problem

Google - “All” infrastructure relies on Paxos



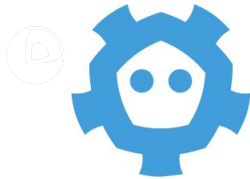


Common problem

Amazon - Replicated log powers ec2

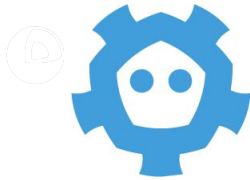
Microsoft - Boxwood powers storage infrastructure

Hadoop - ZooKeeper is the heart of the ecosystem



COMMON PROBLEM

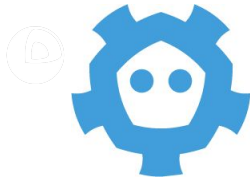
#GIFEE and Cloud Native Solution



10,000 Stars on Github

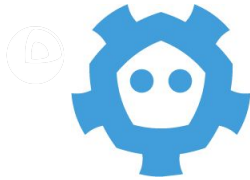
250 contributors

Google, Red Hat, EMC, Cisco, Huawei, Baidu, Alibaba...



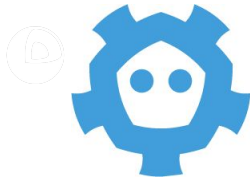
THE HEART OF CLOUD NATIVE

Kubernetes, Cloud Foundry's Diego,
Docker's SwarmKit, many others



ETCD KEY VALUE STORE

Fully Replicated, Highly Available, Consistent

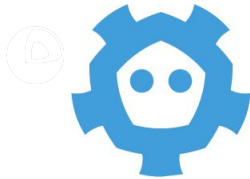


Key-value Operations

PUT(foo, bar), GET(foo), DELETE(foo)

Watch(foo)

CAS(foo, bar, bar1)

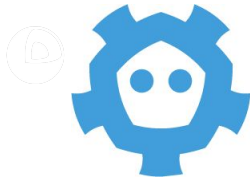


DEMO

play.etcd.io

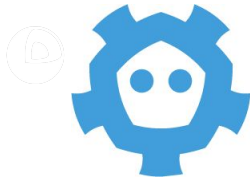
etcd Operationality

Runtime Reconfiguration
Point-in-time Backup
Extensive Metrics



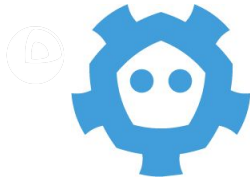
ETCD v3

Successor of etcd v2



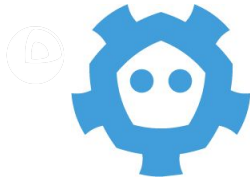
ETCD v3

Better Performance



ETCD v3

Massively Scalable



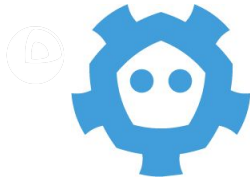
ETCD v3

More Efficient & Powerful APIs

gRPC Based API

~4x Faster vs JSON

HTTP/2 Improves Efficiency



Multi-Version

```
Put(foo, bar)

Put(foo, bar1)

Put(foo, bar2)

Get(foo) -> bar2
```

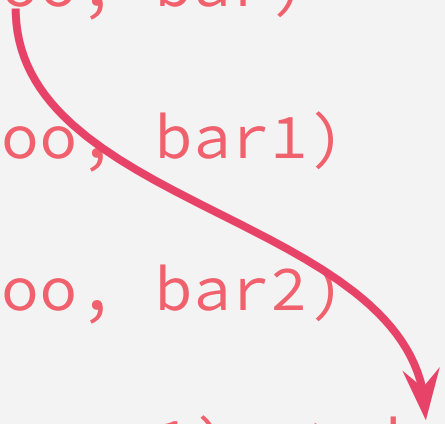
Multi-Version

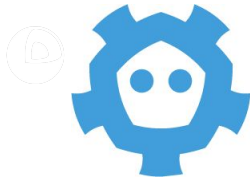
```
Put(foo, bar)

Put(foo, bar1)

Put(foo, bar2)

Get(foo, 1) -> bar
```





Mini-Transactions

```
Tx.If(
Compare(Value("foo"), ">", "bar"),
Compare(Version("foo"), "=", 2),
...
).Then(
Put("ok","true")...
).Else(
   Put("ok","false")...
).Commit()
```

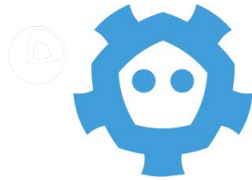
Leases

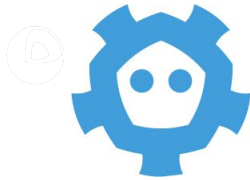
```
l = CreateLease(15 * second)

Put(foo, bar, l)

l.KeepAlive()

l.Revoke()
```

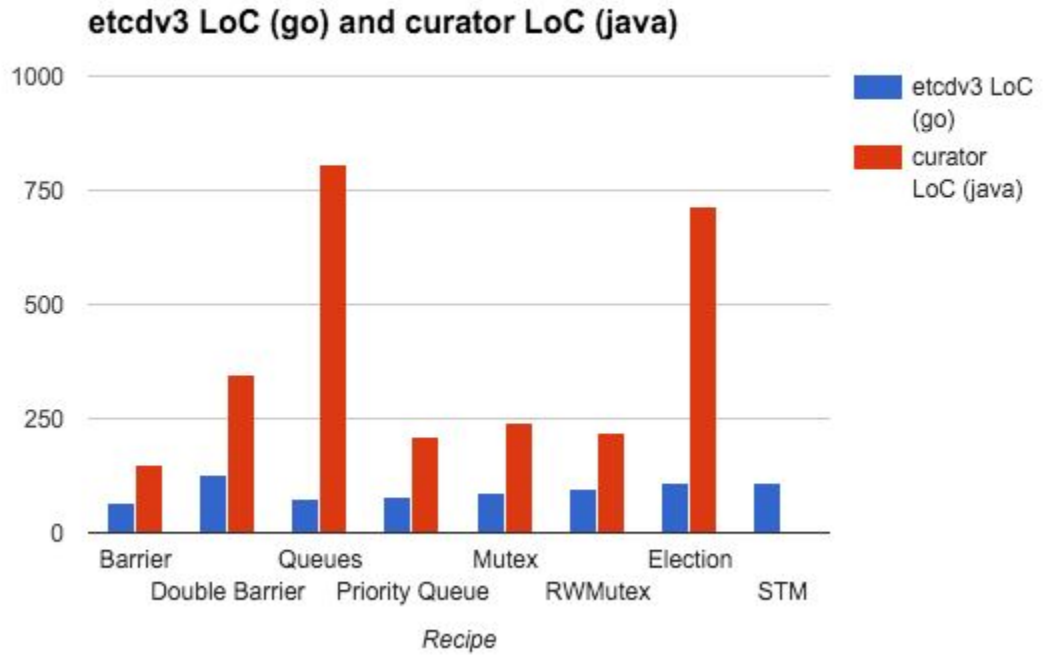


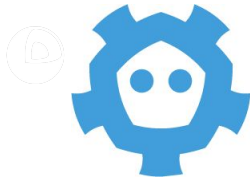


Streaming Watch

```
w = Watch(foo)
for {
r = w.Recv()
  print(r.Event) // PUT
  print(r.KV) // foo,bar
}
```

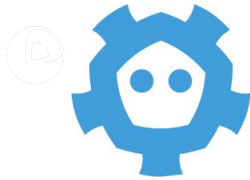
Synchronization LoC





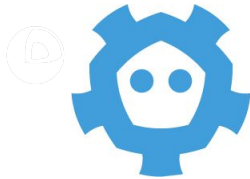
ETCD v2

machine coordination -> O(10k)



ETCD v3

app/container coordination -> O(1M)



Reliability

99% at small scale is easy

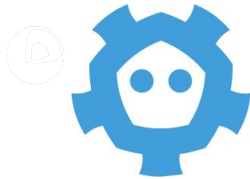
- Failure is infrequent and human manageable

99% at large scale is not enough

- Not manageable by humans

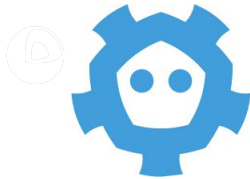
99.99% at large scale

- Reliable systems at bottom layer



HOW DO WE ACHIEVE RELIABILITY

WAL, Snapshots, Testing



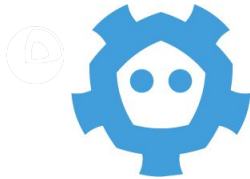
Write Ahead Log

Append only

- Simple is good

Rolling CRC protected

- Storage & OSes can be unreliable



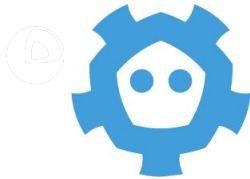
Snapshots

Torturing DBs for Fun and Profit (OSDI2014)

- The simpler database is safer
- LMDB was the winner

Boltdb an append only B+Tree

- A simpler LMDB written in Go

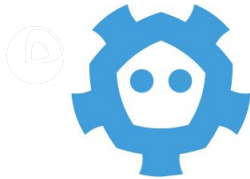


Testing Clusters Failure

Inject failures into running clusters

White box runtime checking

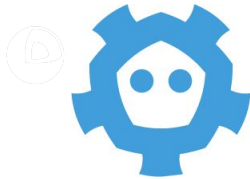
- Hash state of the system
- Progress of the system



Testing Cluster Health with Failures

Issue lock operations across cluster

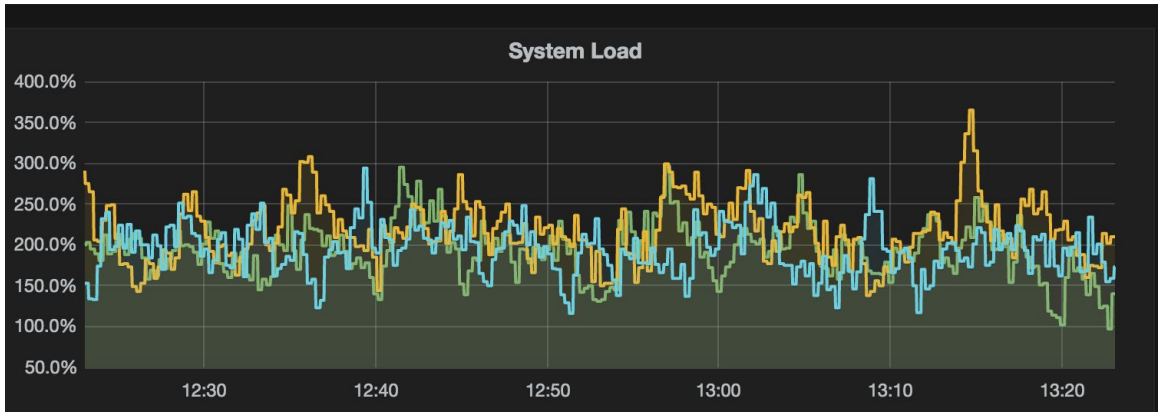
Ensure the correctness of client library



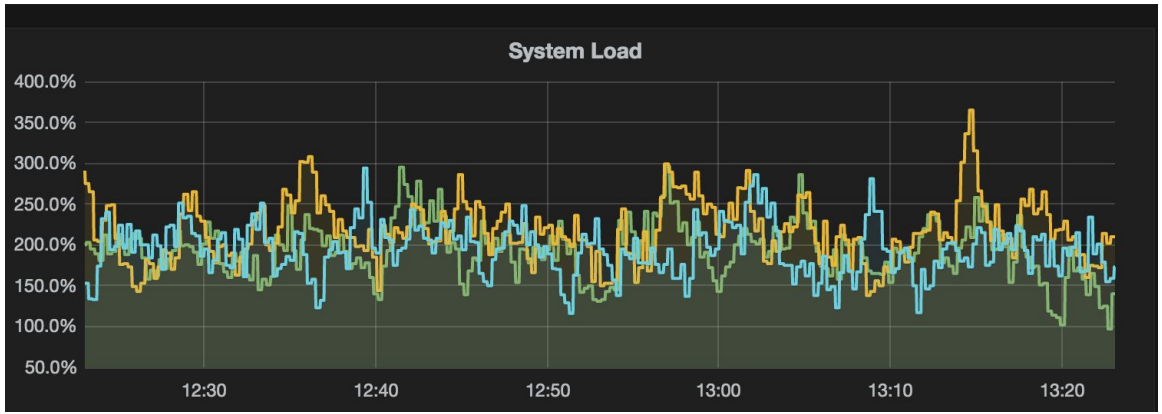
TESTING CLUSTER

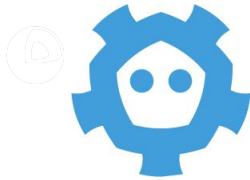
dash.etcd.io

Punishing Functional Tests



Punishing Functional Tests



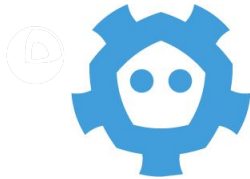


etcd/raft Reliability

Designed for testability and flexibility

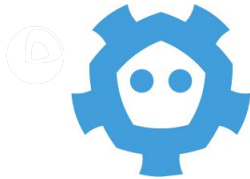
Used by large scale db systems and others

- Cockroachdb, TiKV, Dgraph



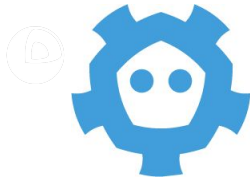
etcd vs others

Do one thing



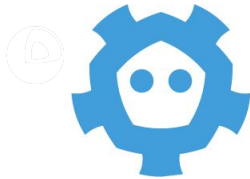
etcd vs others

Only do the One Thing



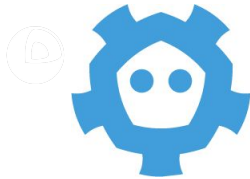
etcd vs others

Do it Really Well



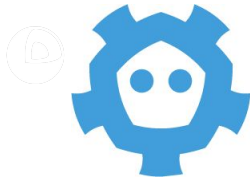
etcd Reliability

Do it Really Well



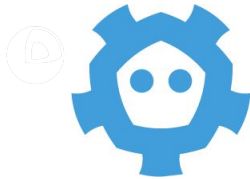
ETCD v3.0 BETA

Efficient and Scalable



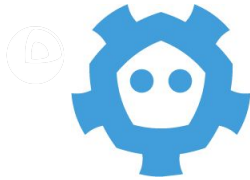
BETA AVAILABLE TODAY

github.com/coreos/etcd



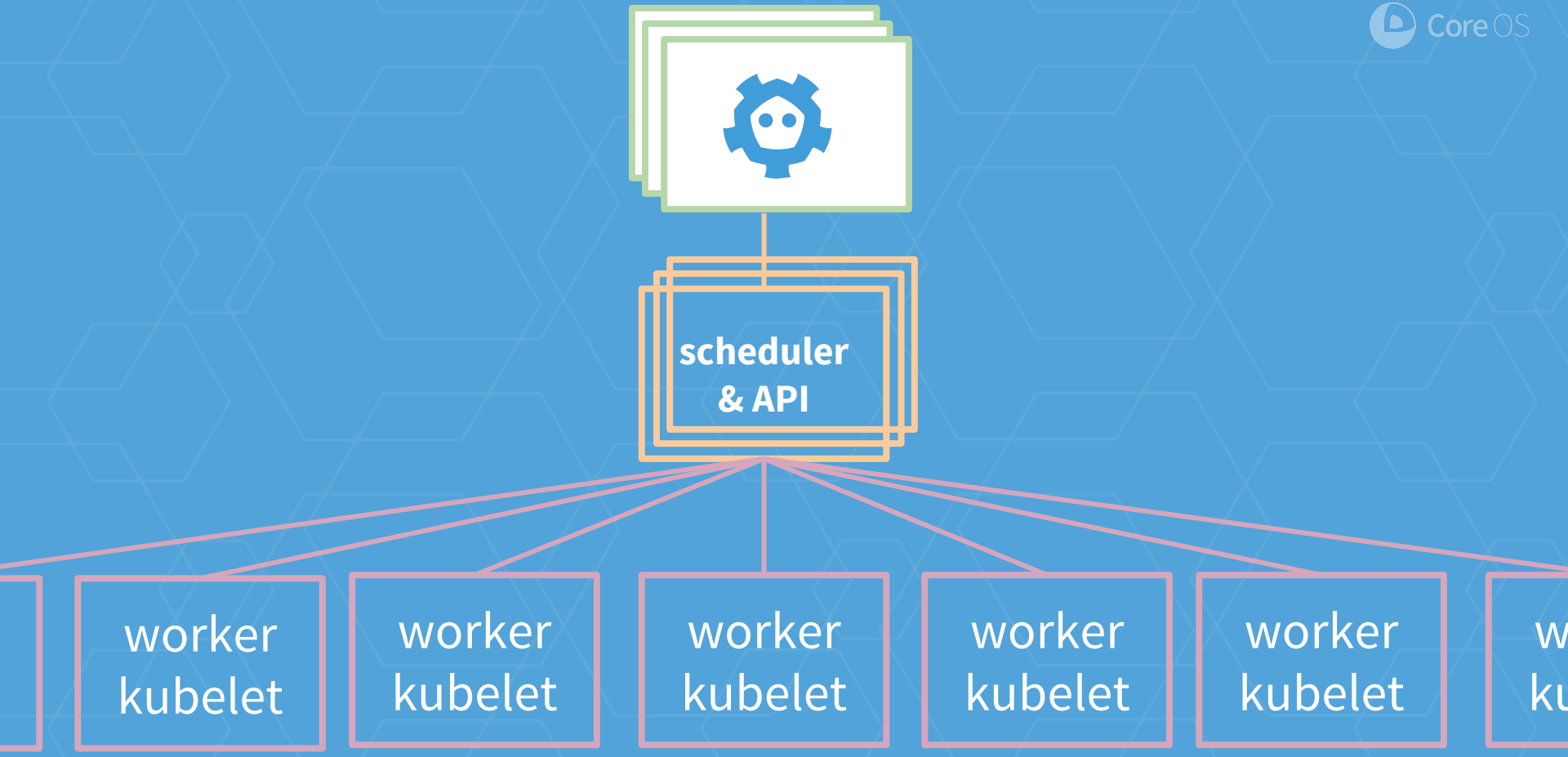
FUTURE WORK

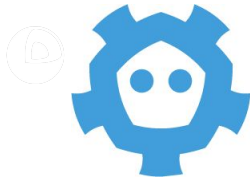
Proxy, Caching, Watch Coalescing, Secondary Index



ETCD and KUBERNETES

The Data Store


scheduler
& API
worker
kubelet
worker
kubelet
worker
kubelet
worker
kubelet
worker
kubelet

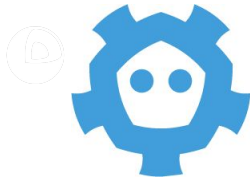


etcd and Kubernetes

- Kubernetes currently uses the V2 API

- Work very actively in process to migrate to V3

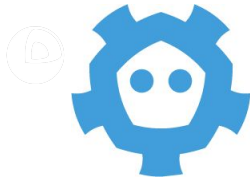
- Opt-in currently, default in future



etcd v3 and Kubernetes

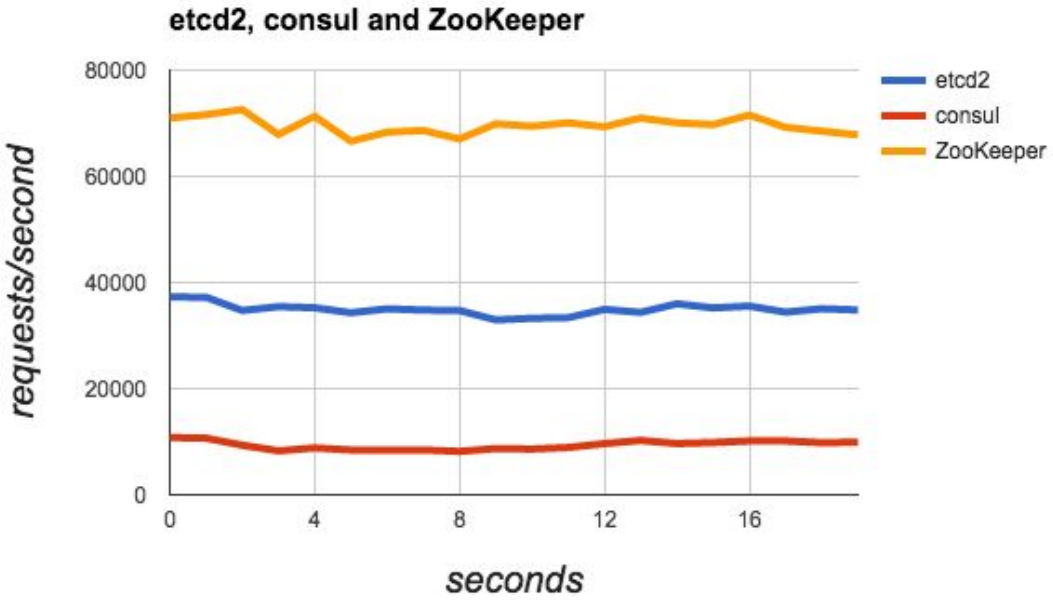
- Follow along:
https://github.com/kubernetes/kubernetes/issues/22448

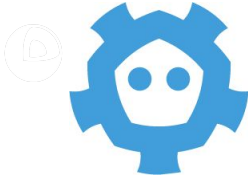
- Try it out!



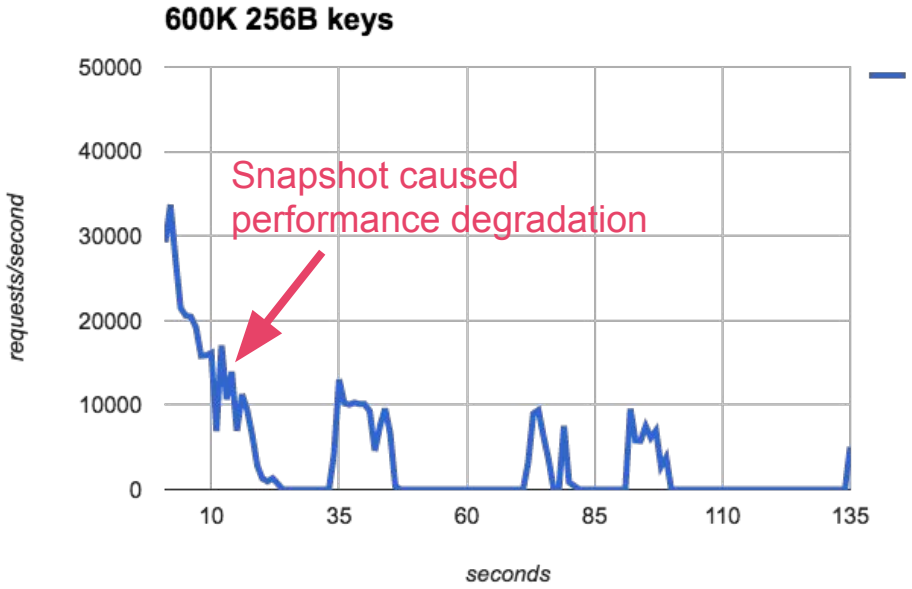
etcd v3 will support Kubernetes
as it scales to 5.000 nodes and beyond

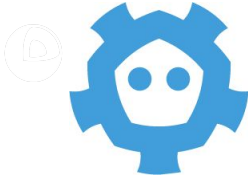
Performance 1K keys



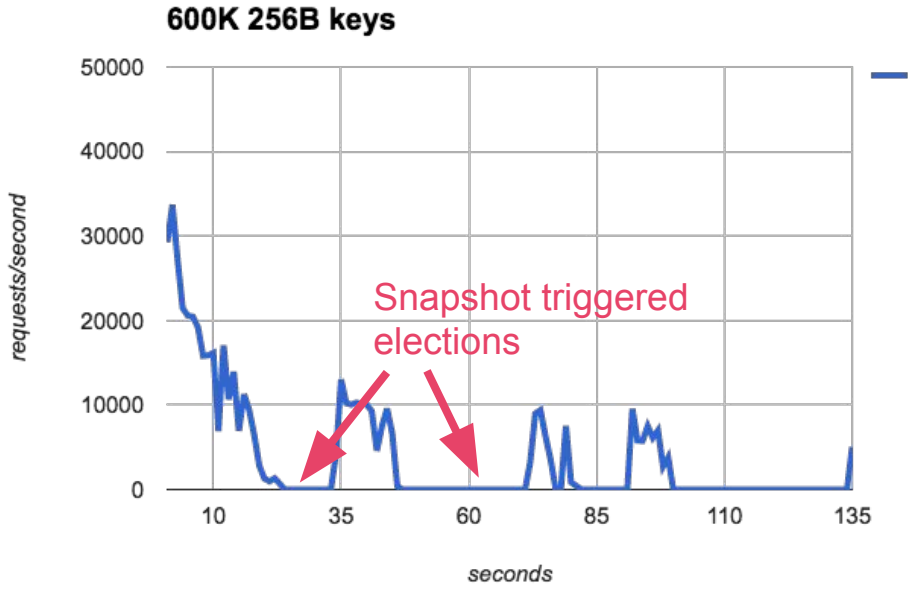


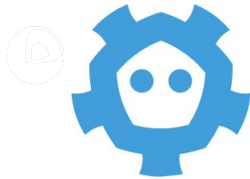
Performance etcd2 - 600K keys





Performance etcd2 - 600K keys

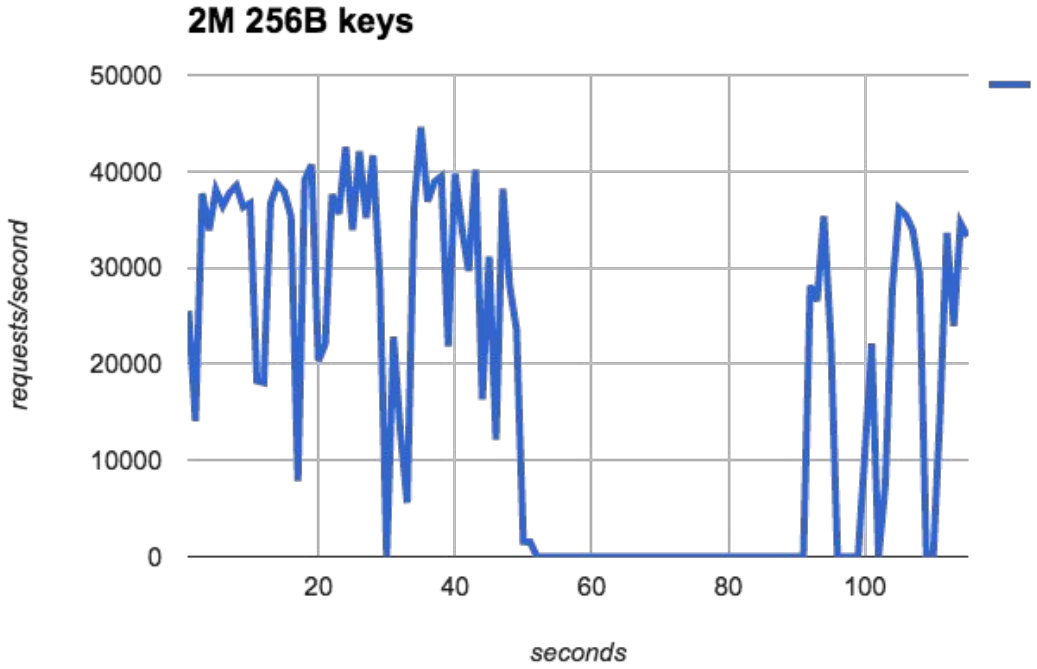




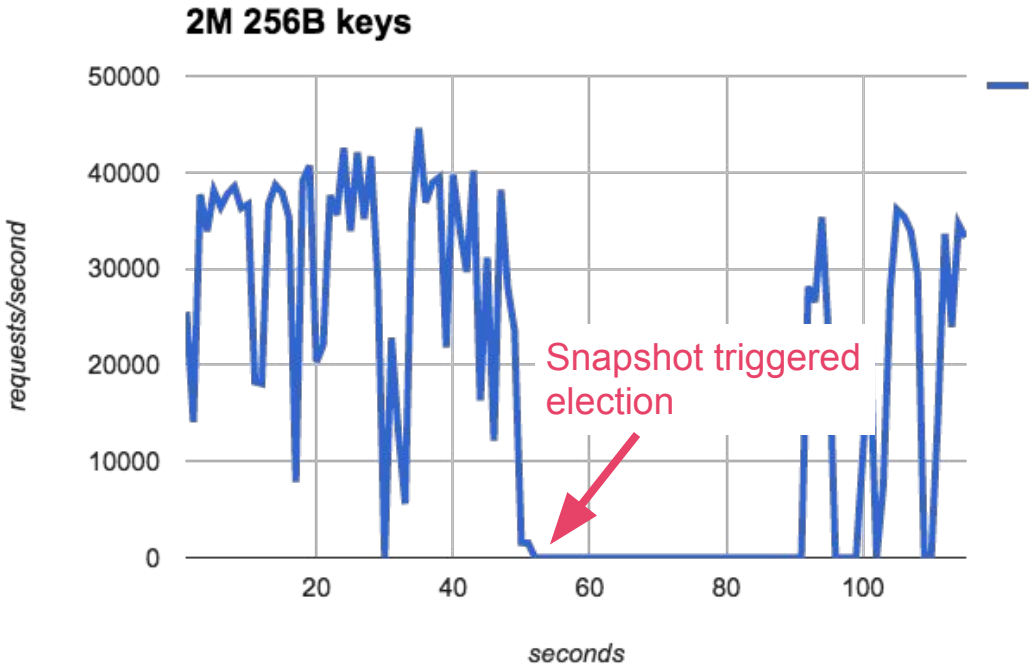
ZooKeeper Performance

Non-blocking full snapshot
Efficient memory management

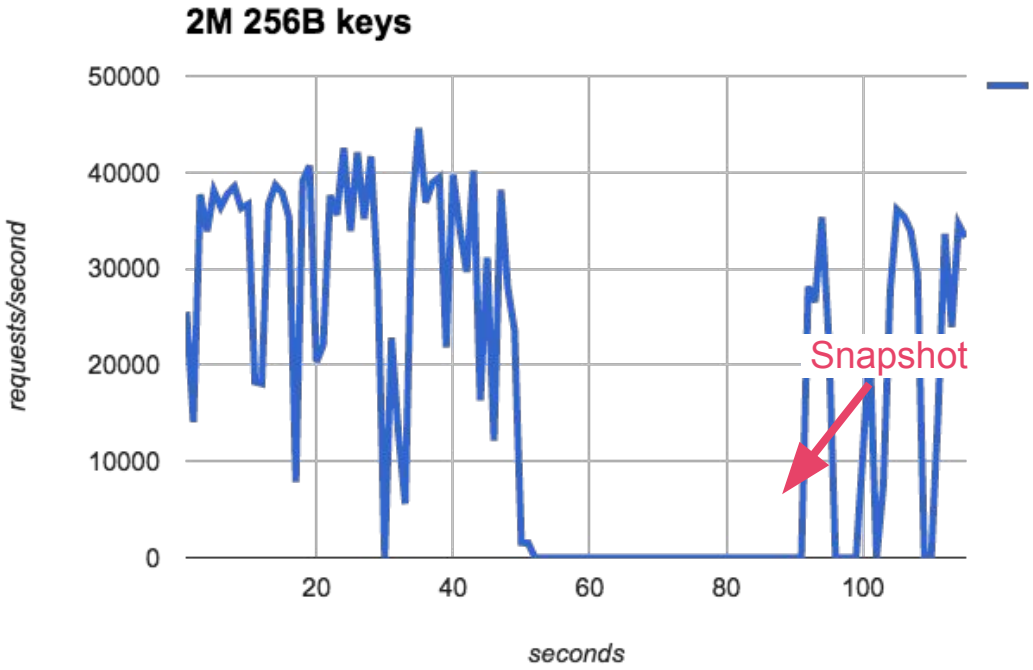
Performance ZooKeeper default



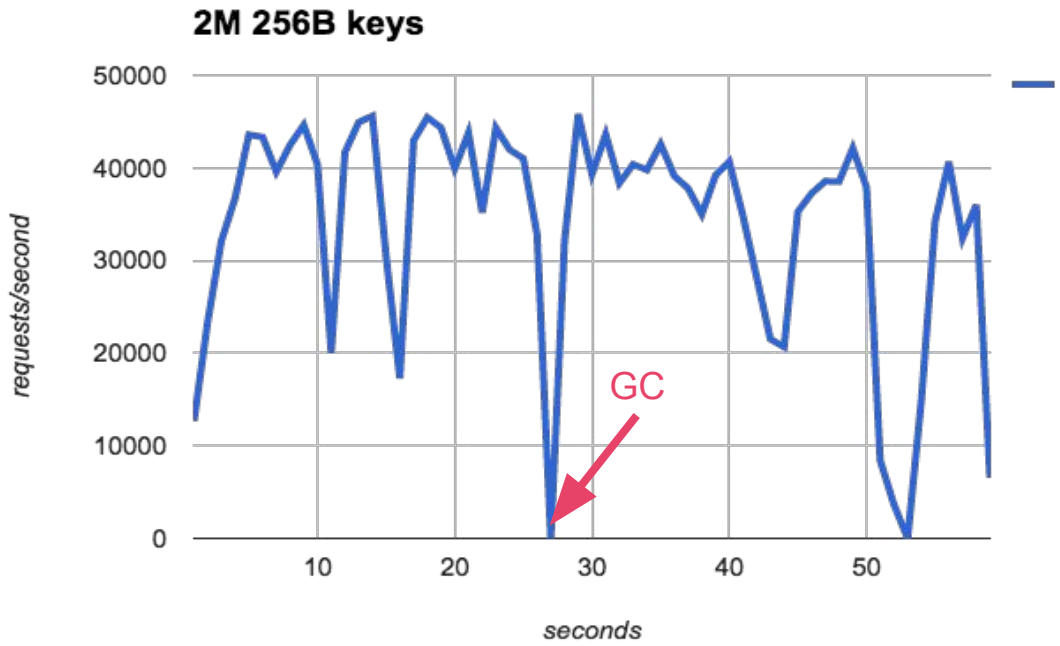
Performance ZooKeeper default

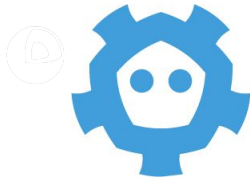


Performance ZooKeeper default



Performance ZooKeeper snapshot disabled

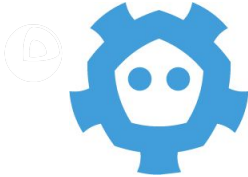




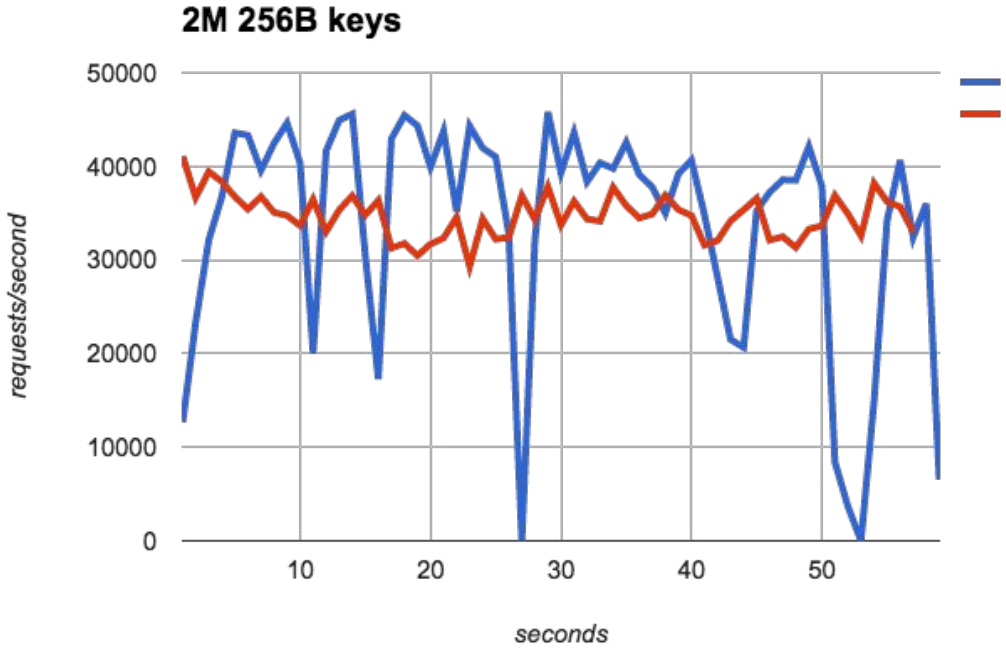
Reliable Performance

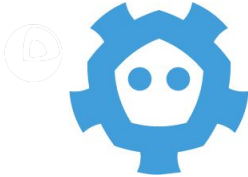
- Similar to ZooKeeper with snapshot disabled
  - Incremental snapshot

- No Garbage Collection Pauses
  - Off-heap storage

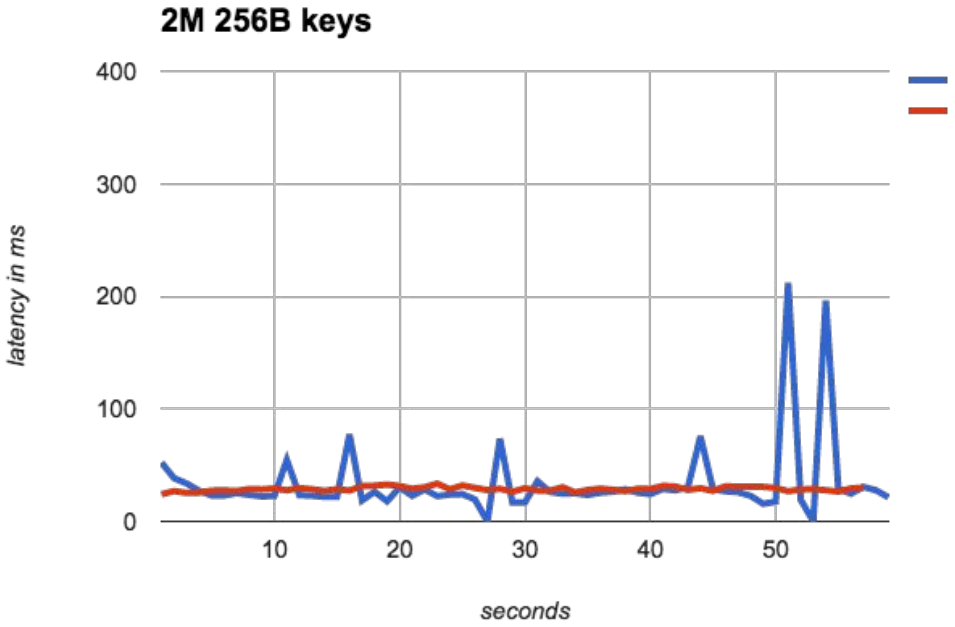


Performance etcd3 /ZooKeeper snapshot disabled

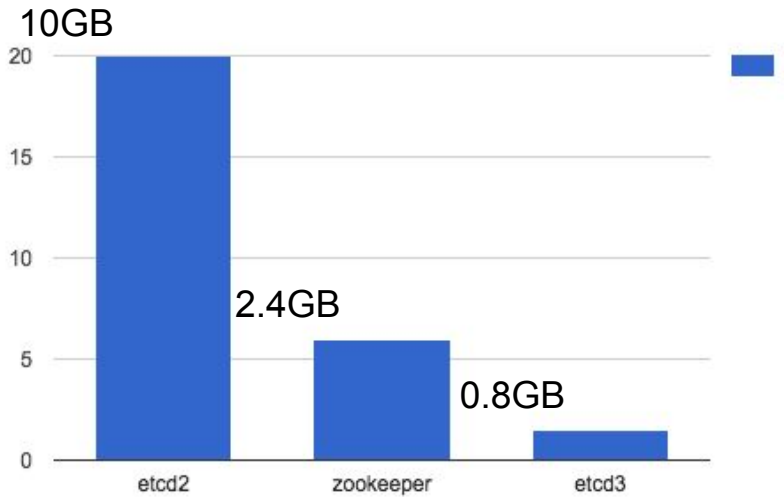


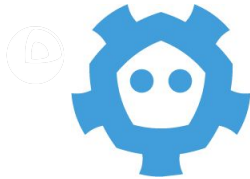


Performance etcd3 /ZooKeeper snapshot disabled



Memory 512MB data - 2M 256B keys





GET INVOLVED

github.com/coreos/etcd