Debugging etcd

Joe Betz, Google Jingyi Hu, Google

About Us

Joe Betz (@jpbetz on github)

Lead engineer for etcd at Google. etcd open source project maintainer. Active contributor to Kubernetes.

Jingyi Hu (@jingyih on github)

Software engineer at Google. Active contributor to open source etcd and Kubernetes.

Agenda

- etcd Recap
- How etcd Serves and Stores Data
- Tools of the Trade
- Debugging Approaches
- Keeping your etcd Healthy
- Q/A


KubeCon
CloudNativeCon
North America 2018
etcd Recap

etcd Recap

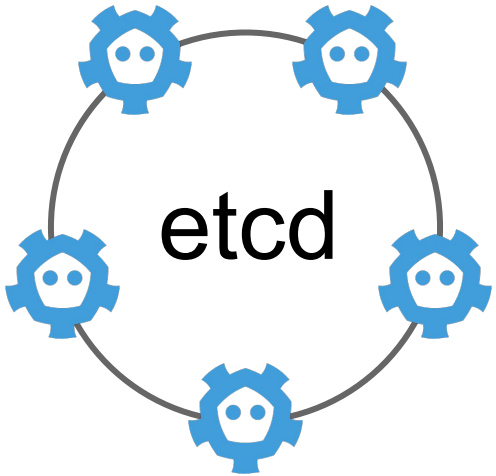
Distributed Key/Value store

“Consensus Datastore”

Reliably manage the coordination state of distributed systems

Related: Google Chubby, Apache ZooKeeper

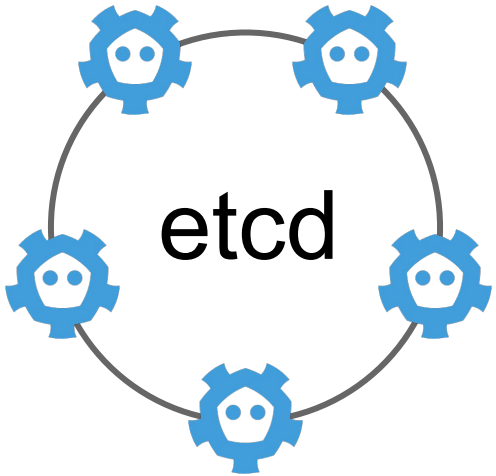
- Highly Available
- Strong consistency model
- Scalable watch mechanism
- Concurrency control primitives



etcd Recap

RAFT Consensus Algorithm

Cluster Size	Majority	Fault Tolerance
1	1	0
2	2	0
3	2	1
4	3	1
5	3	2



etcd Recap

Prometheus

“open-source monitoring system and time series database”

gRPC

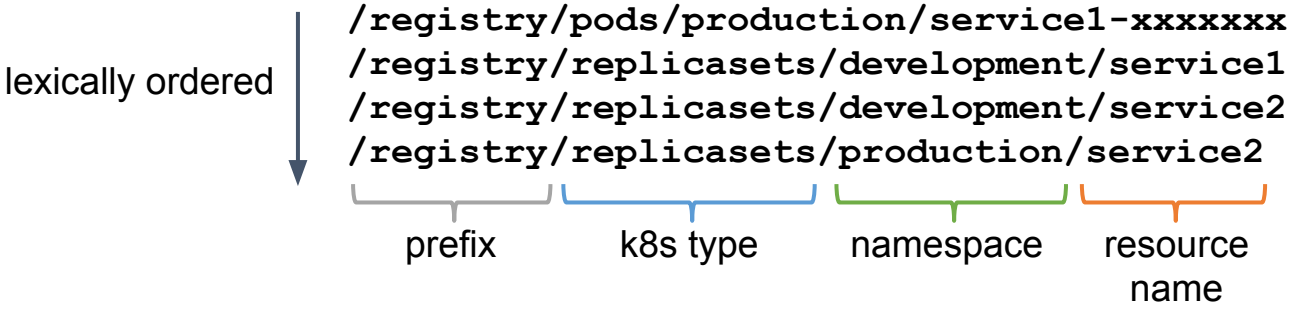
“A high performance, open-source universal RPC framework” (API Also exposed via JSON+HTTP)

BoltDB

“embedded key/value database for Go”

How etcd Serves and Stores Data

Key Space



Lexically ordered index makes “range reads” efficient: RANGE <start-key>..<end-key>

(Not all kubernetes key names are obvious, for example, nodes are keyed as “minions” for legacy reasons)

Data Serving

Request/Response Operations

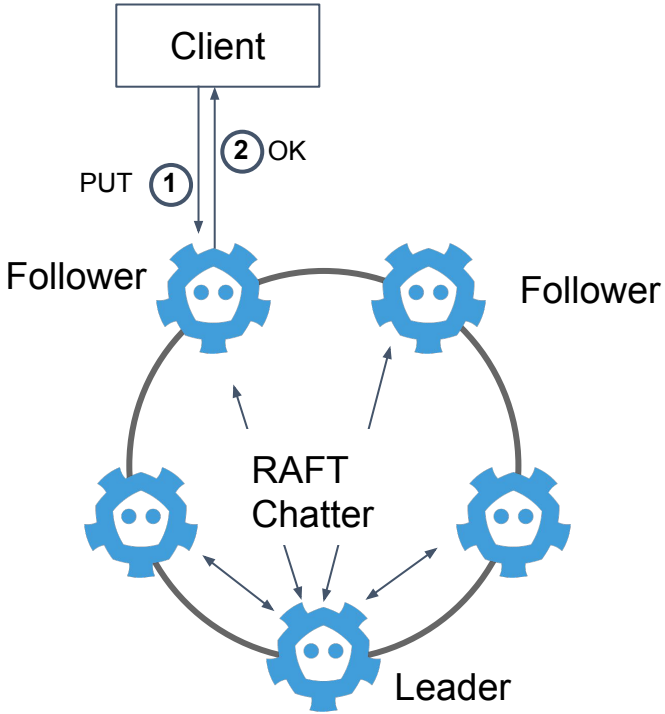
- **RANGE** <start_key>..<end_key>
- **PUT** <key> <value>
- **DELETE RANGE** <start_key>..<end_key>
- **TXN** (if <condition> then <op1, ..> else <op2, ..>)

Data Serving

Request/Response Operations

- **RANGE** <start_key>..<end_key>
- **PUT** <key> <value>
- **DELETE RANGE** <start_key>..<end_key>
- **TXN** (if <condition> then <op1, ..> else <op2, ..>)

Linearizable Consistency
a.k.a. External Consistency

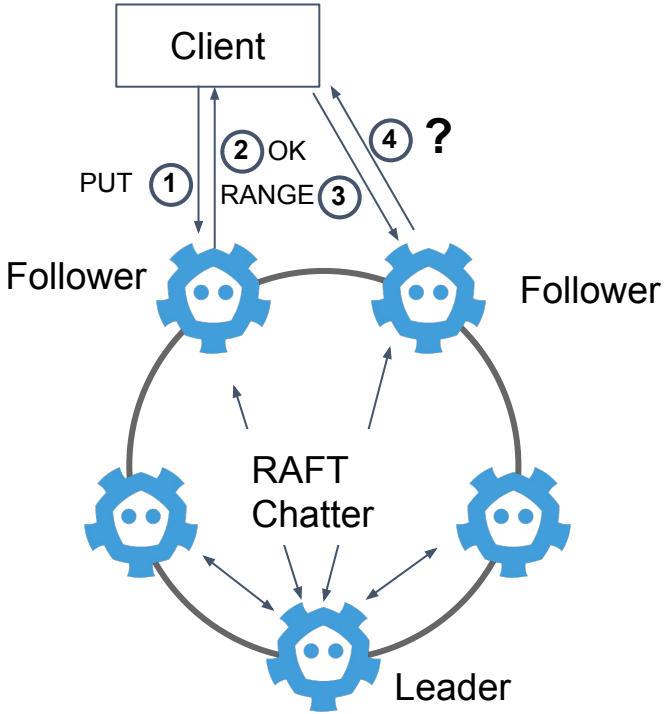


Data Serving

Request/Response Operations

- **RANGE** <start_key>..<end_key>
- **PUT** <key> <value>
- **DELETE RANGE** <start_key>..<end_key>
- **TXN** (if <condition> then <op1, ..> else <op2, ..>)

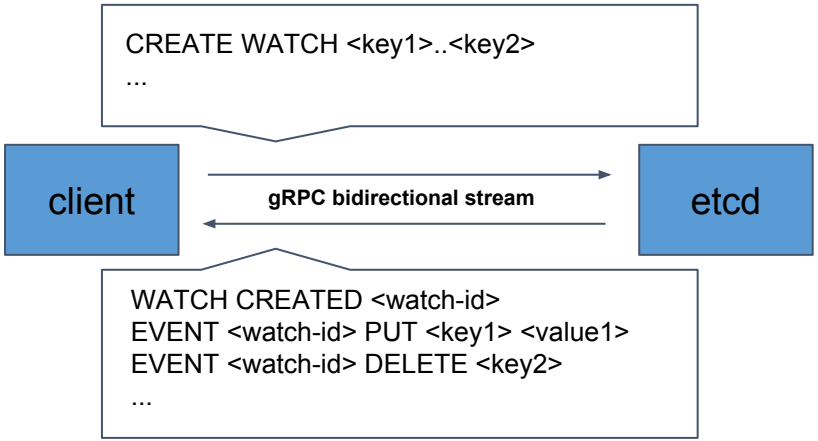
Linearizable Consistency
a.k.a. External Consistency



Data Serving

Streaming Operations

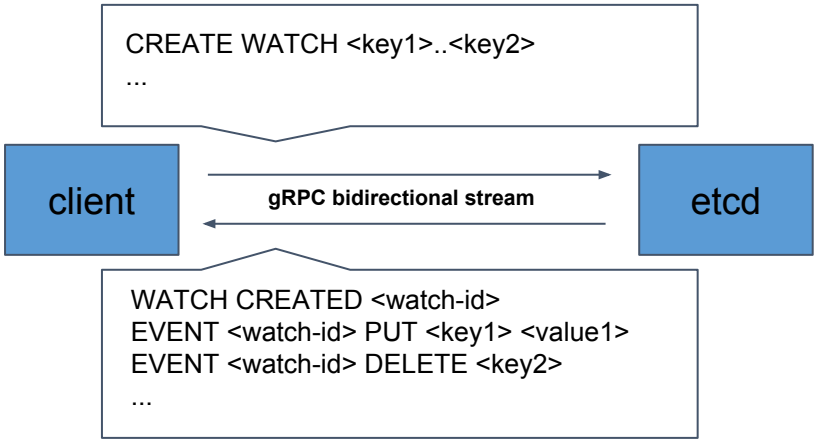
- **WATCH**



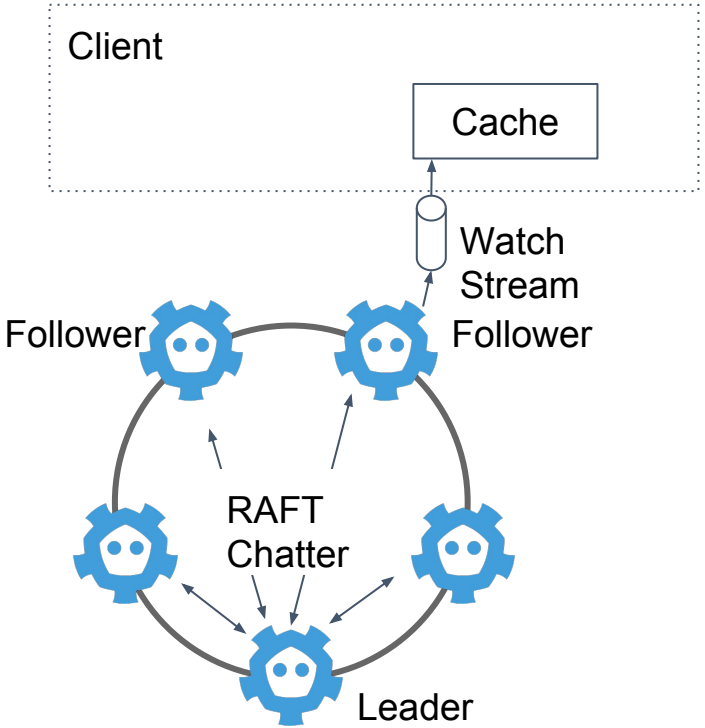
Data Serving

Streaming Operations

- **WATCH**



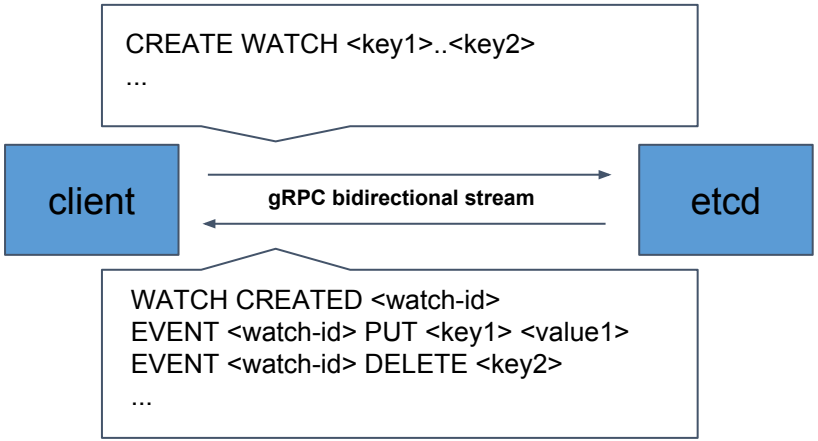
Eventual Consistency



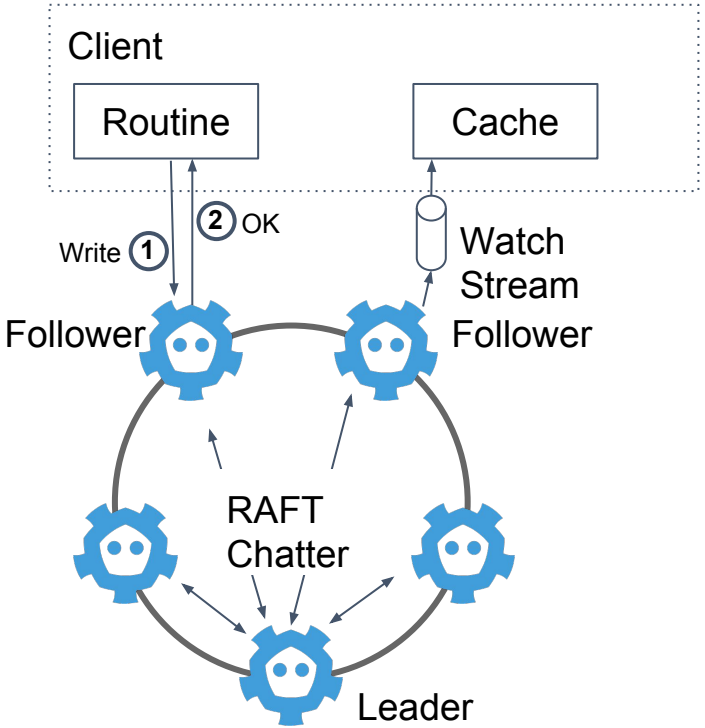
Data Serving

Streaming Operations

- **WATCH**



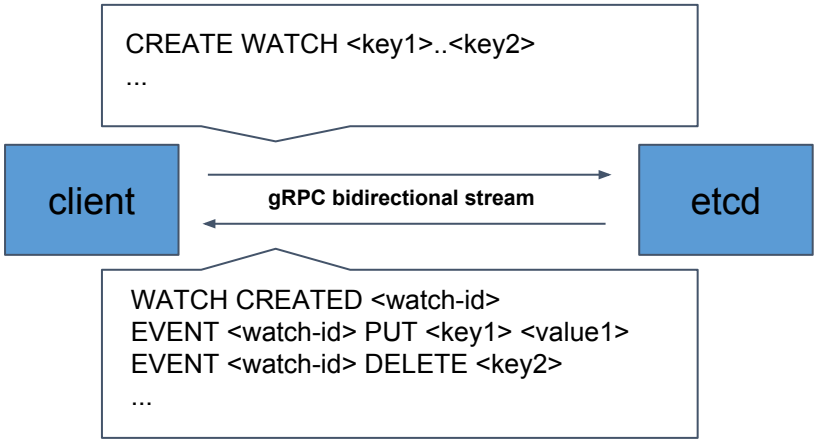
Eventual Consistency



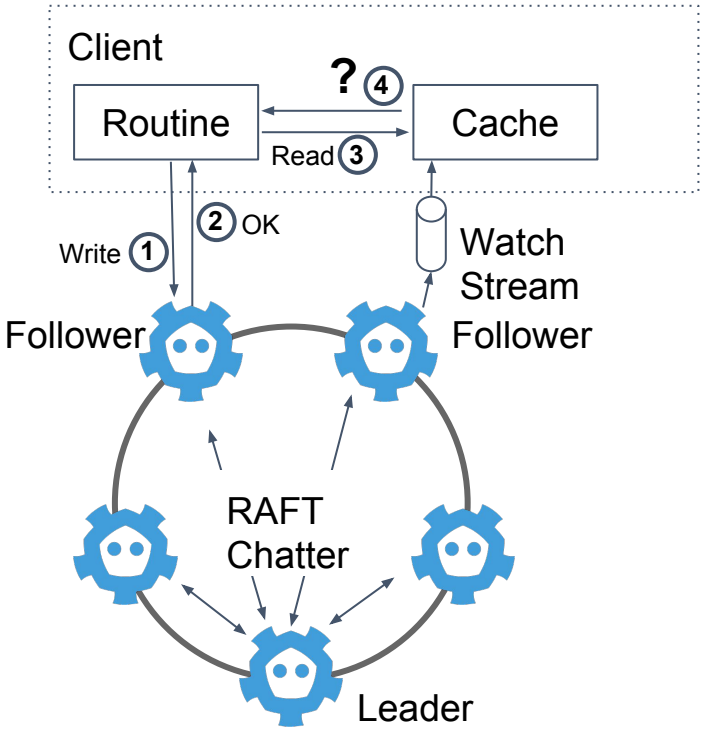
Data Serving

Streaming Operations

- **WATCH**



Eventual Consistency

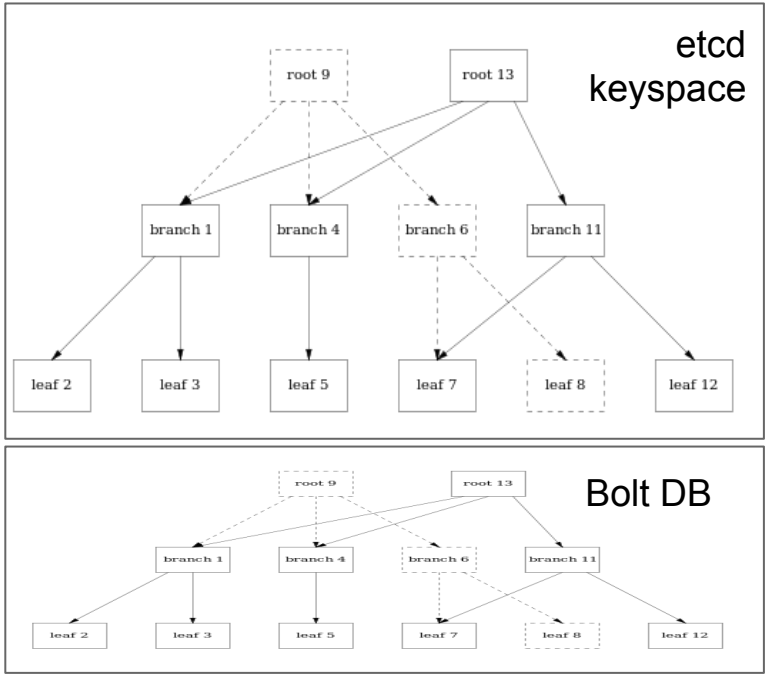


Data Storage

"Multi-version concurrency control."
Copy-on-write for all modifications.

etcd - MVCC keyspace. Values may be accessed by key+version. This is used to implement the watch operation.

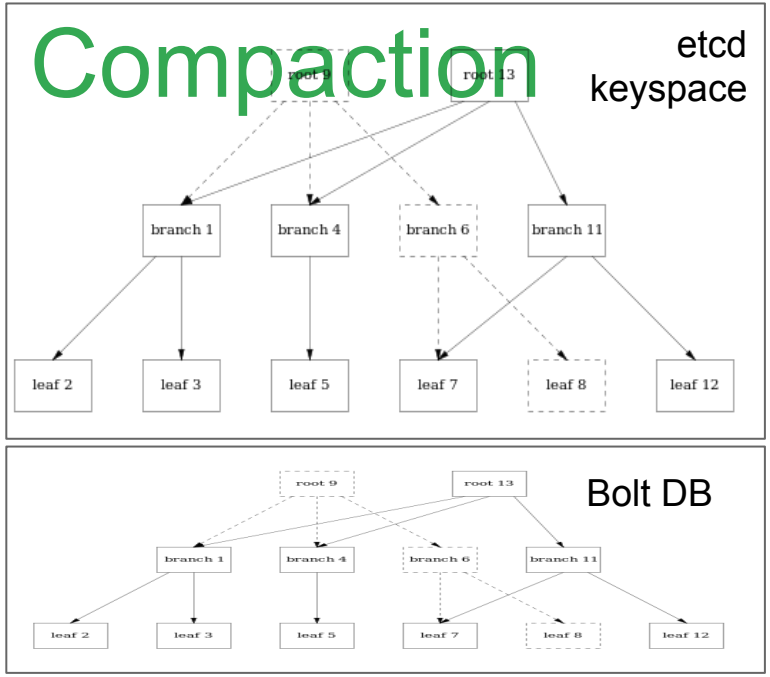
BoltDB - MVCC internally enable 1 write + N reads to be executed concurrently.



Compaction vs. Defragmentation

Compaction applies to the etcd keyspace

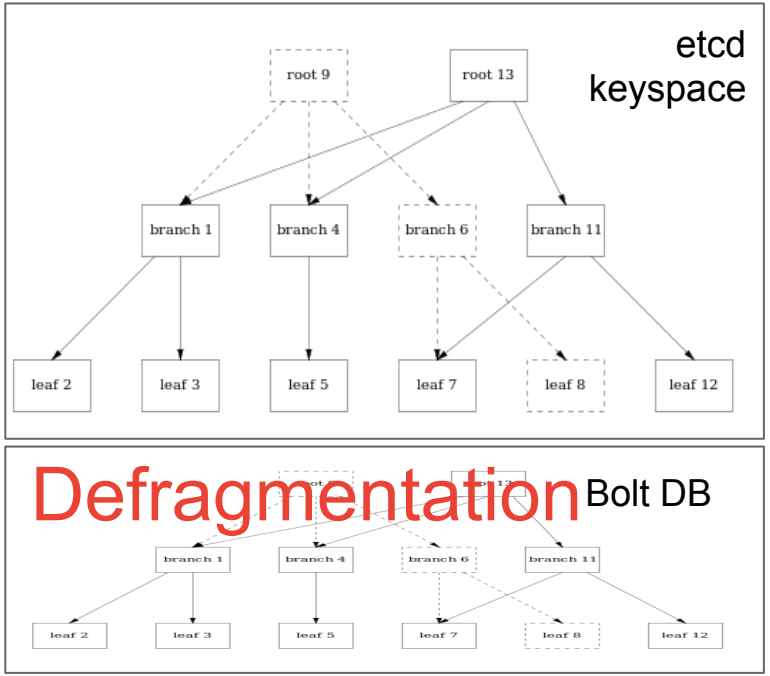
- Removes all versions of objects older than a specific revision number
- Kubernetes default policy: all data older than 5 minutes every 5 minutes
- Kube-apiserver requests compactions. etcd auto-compaction is disabled.



Compaction vs. Defragmentation

Defragmentation applies to the bolt db file

- Recovers all free space in the bolt db file.
- Only to shrink a db file as bolt does not automatically shrinks it’s file.
- Etcd will defrag and the file only if requested. This is a “stop-the-world” operation.



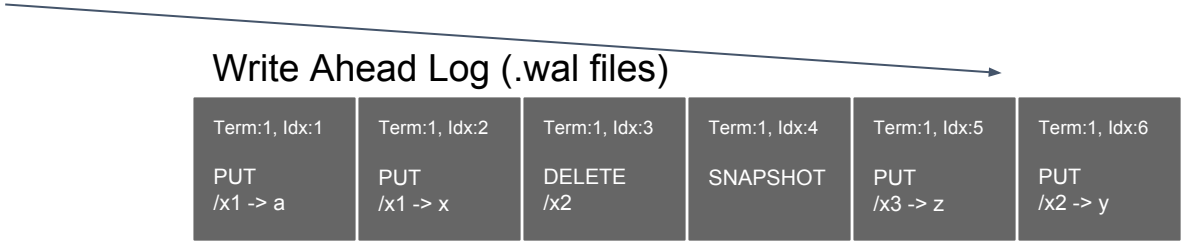
etcd “data-dir”


```
<data-dir>
└── member
    ├── snap
    │   ├── 0000000000000007-0000000000038287.snap
    │   ├── 0000000000000007-000000000003a998.snap
    │   ├── 0000000000000007-000000000003d0a9.snap
    │   ├── 0000000000000007-000000000003f7ba.snap
    │   ├── 0000000000000007-0000000000041ecb.snap
    │   └── db
    └── wal
        ├── 0000000000000004-000000000001fe18.wal
        ├── 0000000000000005-0000000000027d16.wal
        ├── 0000000000000006-000000000002fc26.wal
        ├── 0000000000000007-0000000000037b2a.wal
        └── 0000000000000008-000000000003fa1c.wal
```

How etcd Stores and Serves Data

For each write:

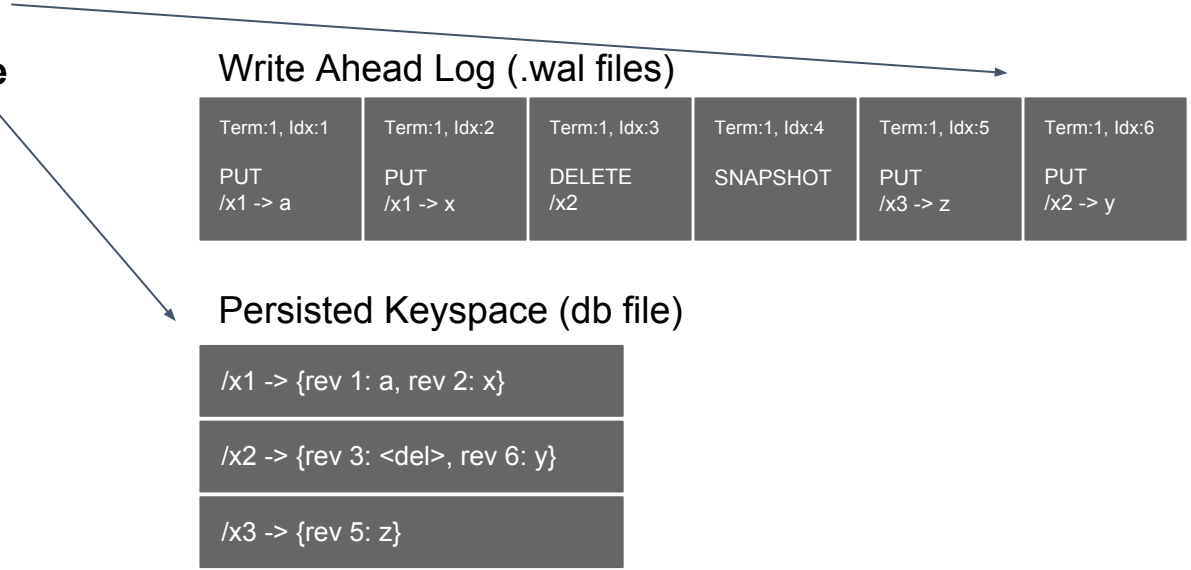
- **1. Append write to WAL**
- 2. Apply write to Keyspace



How etcd Stores and Serves Data

For each write:

- 1. Append write to WAL
- **2. Apply write to Keyspace**



How etcd Stores and Serves Data

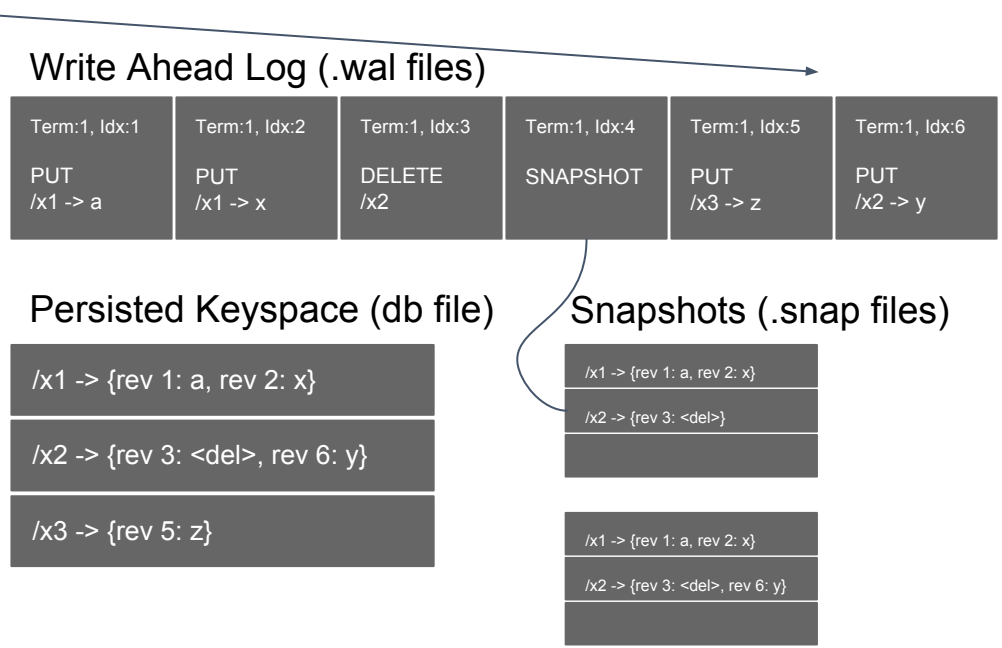
For each write:

- 1. Append write to WAL
- 2. Apply write to Keyspace

Every "--snapshot-count" writes:

- Create a snapshot file
- Record revision snapshot was created to WAL
- Remove WAL files older than the snapshot

RAFT ensures WAL log is the same on all members of an etcd cluster!




KubeCon
CloudNativeCon
North America 2018
Tools of the Trade

Tools of the Trade


```
$ ETCDCTL_API=3 etcdctl

NAME:
        etcdctl - A simple command line client for etcd3.

USAGE:
        etcdctl

VERSION:
        3.3.0

COMMANDS:
        get                     Gets the key or a range of keys
        put                     Puts the given key into the store
        del                     Removes the specified key or range of keys [key, range_end)
        txn                     Txn processes all the requests in one transaction
        compaction              Compacts the event history in etcd
        alarm disarm            Disarms all alarms
        alarm list              Lists all alarms
        defrag                  Defragments the storage of the etcd members with given endpoints
        endpoint health         Checks the healthiness of endpoints specified in `--endpoints` flag
        endpoint status         Prints out the status of endpoints specified in `--endpoints` flag
        watch                   Watches events stream on keys or prefixes
        version                 Prints the version of etcdctl
        lease grant             Creates leases
        lease revoke            Revokes leases
        lease keep-alive        Keeps leases alive (renew)
        member add              Adds a member into the cluster
        member remove           Removes a member from the cluster
...
```

Tools of the Trade


```
$ ETCDCTL_API=3 etcdctl get --prefix --keys-only /


/registry/apiregistration.k8s.io/apiservices/v1.authentication.k8s.io
/registry/apiregistration.k8s.io/apiservices/v1.authorization.k8s.io
/registry/apiregistration.k8s.io/apiservices/v1.autoscaling
/registry/apiregistration.k8s.io/apiservices/v1.batch
...
```

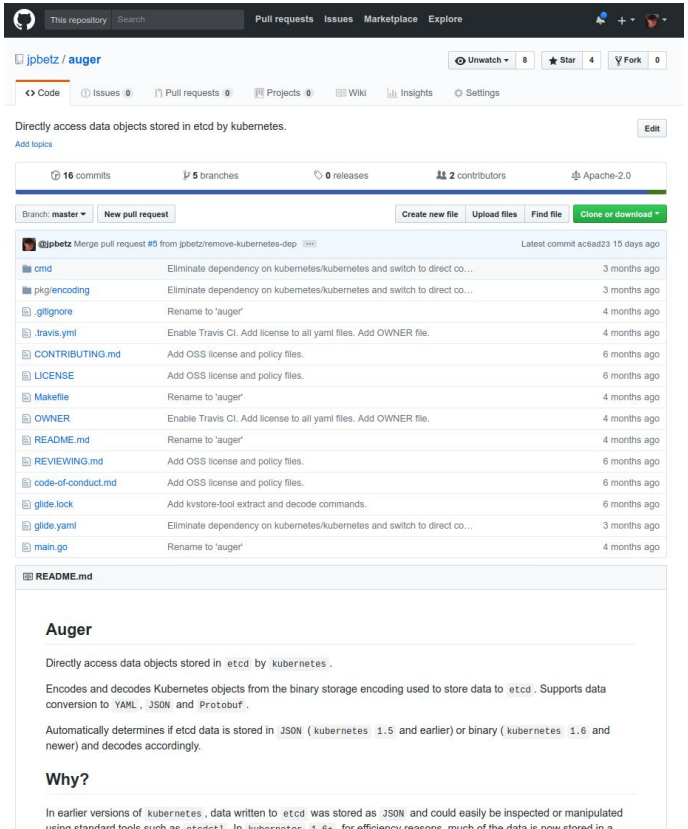
Tools of the Trade


```
$ ETCDCTL_API=3 etcdctl get /registry/pods/kube-system/kube-dns-xxxxxx-xxx

/registry/pods/kube-system/kube-dns-xxxxxx-xxx
k8s

v1Pod
kube-dns-xxxxxx-xxxkube-dns-xxxxxxxxxx-

kube-system$xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxx
k8s-appkube-dnsZ
pod-template-hash
3345330137b
kubernetes.io/created-by
{"kind":"SerializedReference","apiVersion":"v1","reference":{"kind":"ReplicaSet","namespace":"kube-system","name":"kube-dns-xxxxxxxxxx","uid":"xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx","apiVersion":"extensions","resourceVersion":"288"}
}
```

PROTOBUF

Tools of the Trade


```
$ auger --help

Inspect and analyze kubernetes objects in binary storage
encoding used with etcd 3+ and boltdb.

Usage:
  auger [command]

Available Commands:
  decode      Decode objects from the kubernetes binary key-value store encoding.
  encode      Encode objects to the kubernetes binary key-value store encoding.
  extract     Extracts kubernetes data from the boltdb '.db' files etcd persists to.
  help        Help about any command

Flags:
  -h, --help   help for auger

Use "auger [command] --help" for more information about a command.
```



github.com/jpbetz/auger

Tools of the Trade

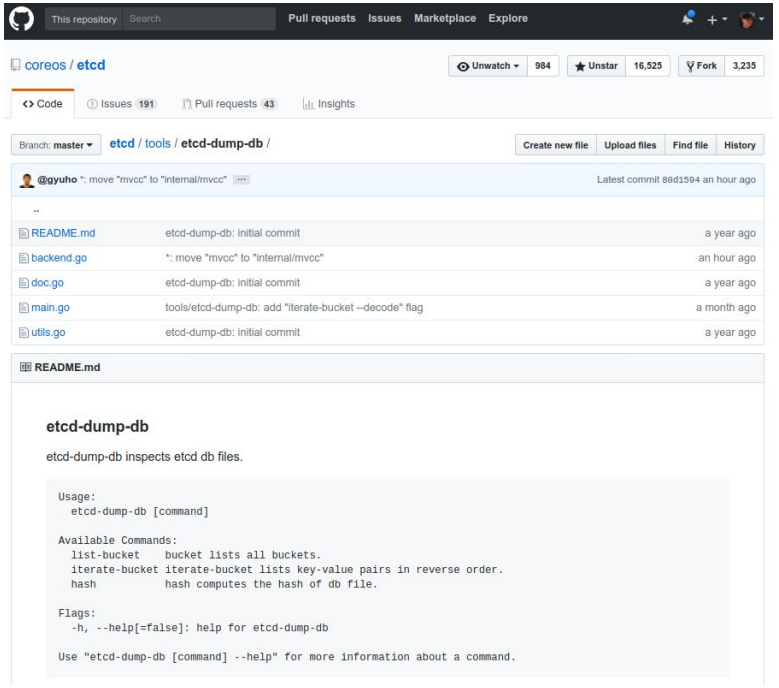

```
$ ETCDCTL_API=3 etcdctl get /registry/events/default/mypod | auger decode

 apiVersion: v1
 count: 1
 firstTimestamp: 2018-05-30T20:41:35Z
 involvedObject:
   apiVersion: v1
   fieldPath: spec.containers{mypod}
   kind: Pod
   name: mypod
   namespace: default
   resourceVersion: "30573"
   uid: xxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxx
 kind: Event
 lastTimestamp: 2018-05-30T20:41:35Z
 message: Container image "gcr.io/example/pod:1.0.0" already present on machine
 metadata:
   creationTimestamp: 2018-05-30T20:41:35Z
```

Tools of the Trade


```
$ auger extract --file <backup-file> --key /registry/events/default/mypod

 apiVersion: v1
 count: 1
 firstTimestamp: 2018-05-30T20:41:35Z
 involvedObject:
   apiVersion: v1
   fieldPath: spec.containers{mypod}
   kind: Pod
   name: mypod
   namespace: default
   resourceVersion: "30573"
   uid: xxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxx
 kind: Event
 lastTimestamp: 2018-05-30T20:41:35Z
 message: Container image "gcr.io/example/pod:1.0.0" already present on
machine
 metadata:
```

Tools of the Trade


```
$ etcd-dump-logs -h
Usage of ./etcd-dump-logs:
  -entry-type string
        If set, filters output by entry type. Must be one or more than one of:
        ConfigChange, Normal, Request, InternalRaftRequest,
        IRRRange, IRRPut, IRRDeleteRange, IRRTxn,
        IRRCompaction, IRRLeaseGrant, IRRLeaseRevoke
  -start-index uint
        The index to start dumping
  -start-snap string
        The base name of snapshot file to start dumping
  -stream-decoder string
        The name of an executable decoding tool, the executable must process
        hex encoded lines of binary input (from etcd-dump-logs)
        and output a hex encoded line of binary for each input line
```



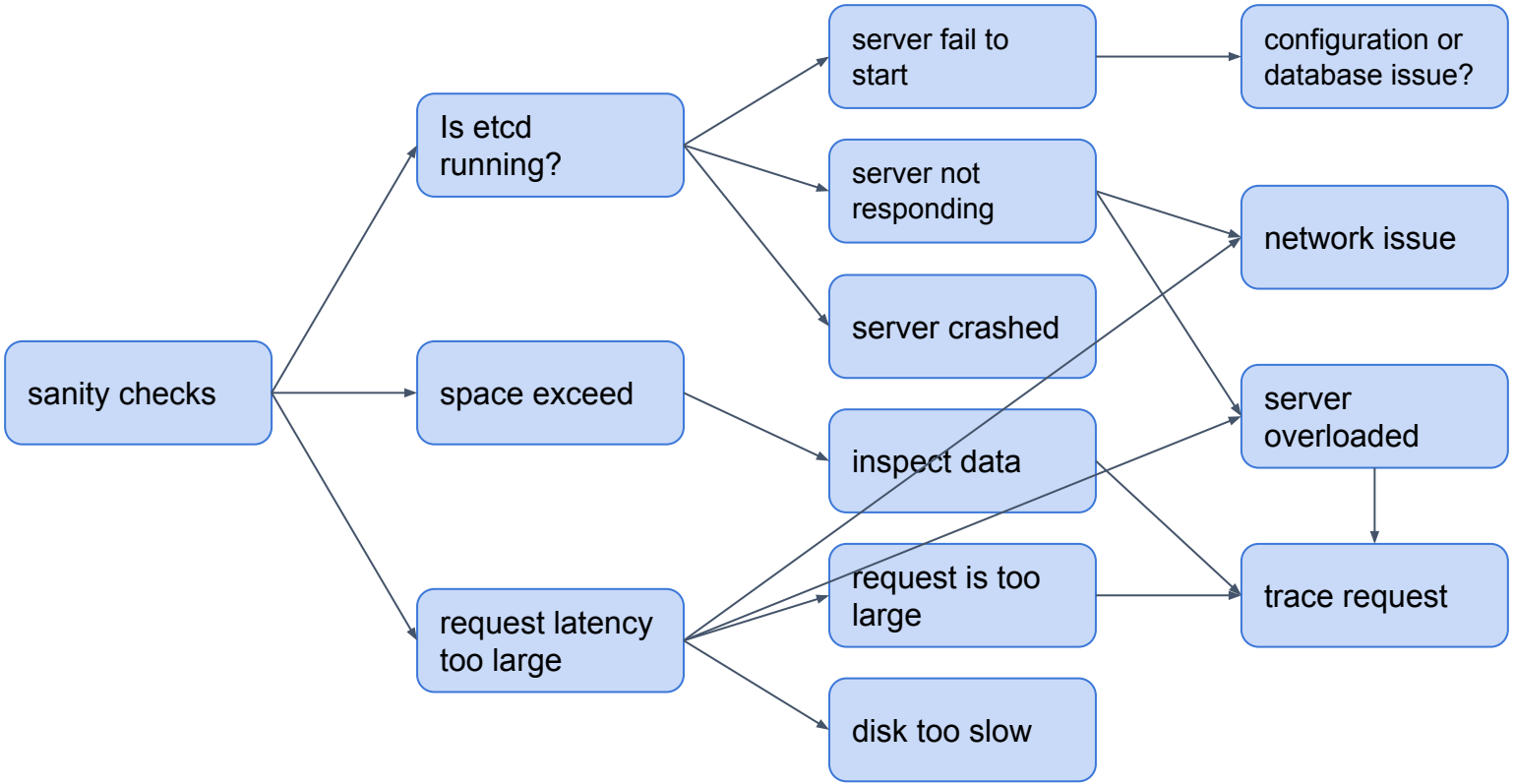
https://github.com/etcd-io/etcd/tree/master/tools/etcd-dump-logs

Tools of the Trade

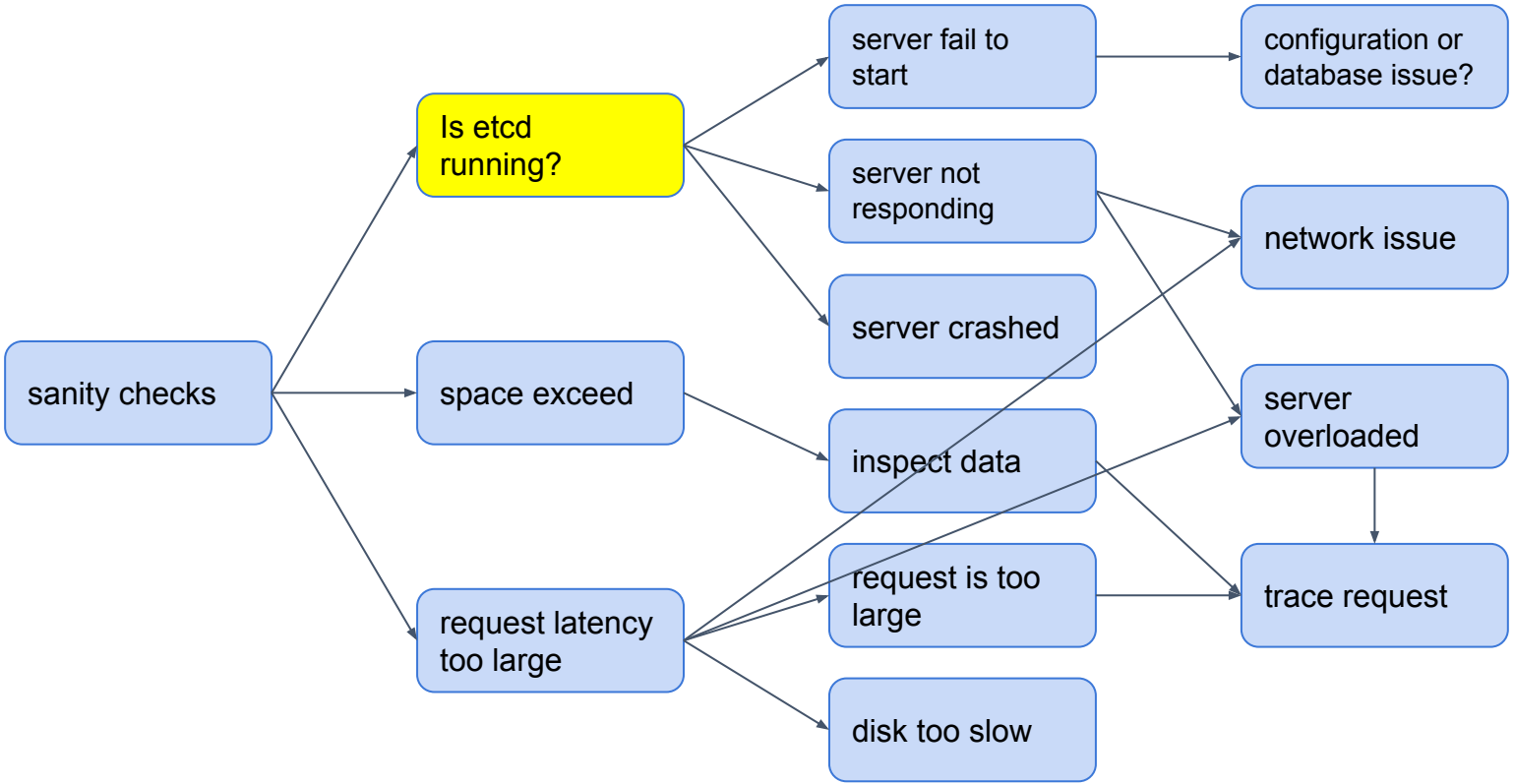

```
$ etcd-dump-logs /var/etcd/data
...
term         index     type    data
   1             1     conf    method=ConfChangeAddNode id=2
   2             2     conf    method=ConfChangeRemoveNode id=2
   2             3     conf    method=ConfChangeUpdateNode id=2
   2             4     conf    method=ConfChangeAddLearnerNode id=3
   7            13     norm    ID:8 txn:<success:<request_delete_range:<key:"a"
range_end:"k8s\000\n\025\n\002v1\022\017RangeAllocation\022#\n\022\n\000\022\000\032\000\"\000*\0002\0008\000B\000z\000\022\01310.0.0.0/1
6\032\000\032\000\"\000" > > failure:<request_delete_range:<key:"a"
range_end:"k8s\000\n\025\n\002v1\022\017RangeAllocation\022#\n\022\n\000\022\000\032\000\"\000*\0002\0008\000B\000z\000\022\01310.0.0.0/1
6\032\000\032\000\"\000" > > >
   8            14     norm    ID:9 compaction:<physical:true >
```


KubeCon
CloudNativeCon
North America 2018
Debugging Approaches

Debugging Approaches



Debugging Approaches



Debugging Approaches

- Sanity checks: is etcd running?

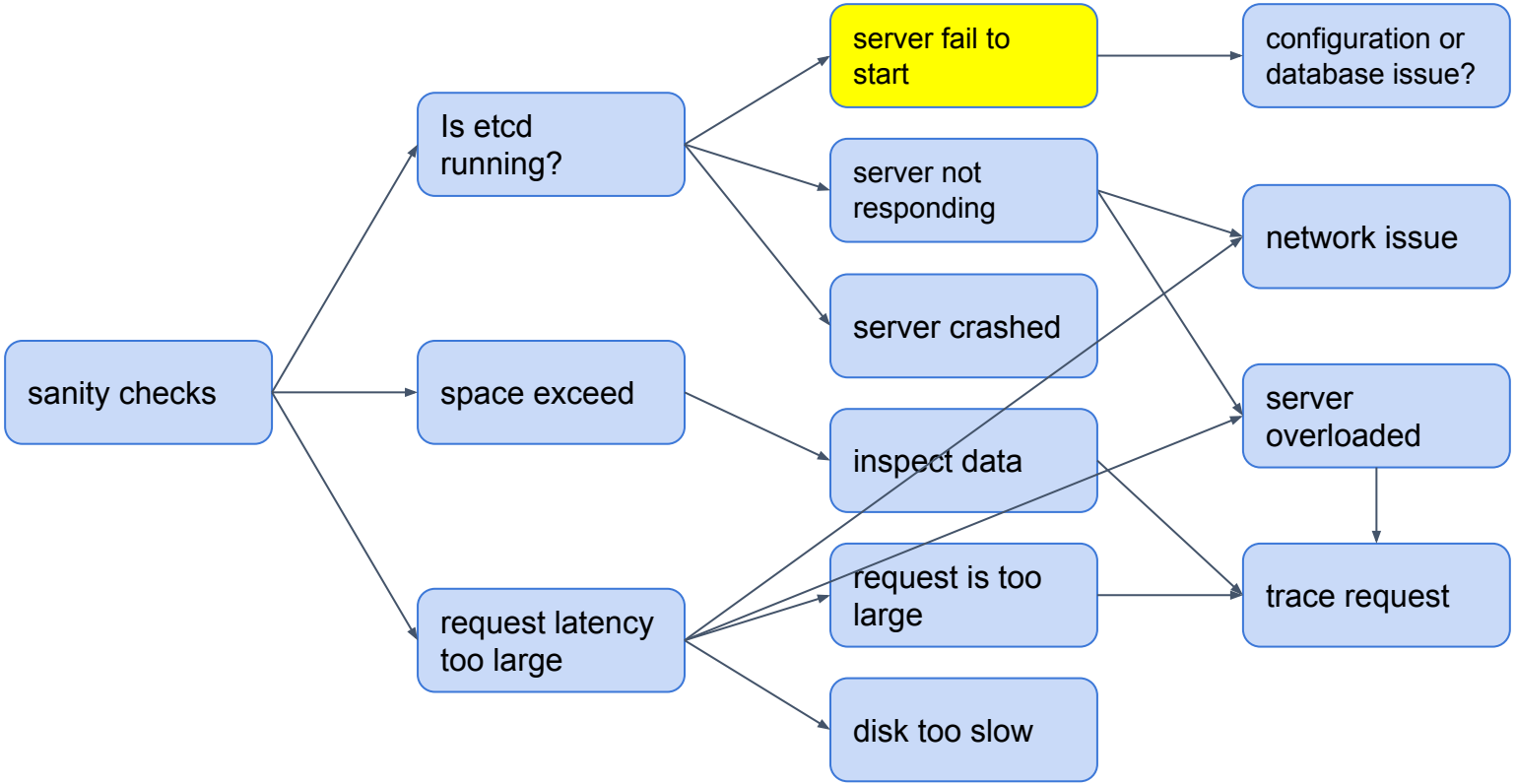
```
$ docker ps | grep etcd
CONTAINER ID        IMAGE          COMMAND              CREATED             STATUS
79c8331e02c6        .../etcd       "/bin/sh -c ..."     2 days ago          Up 2 days

$ docker ps | grep etcd
CONTAINER ID        IMAGE          COMMAND              CREATED             STATUS
Ad39be67ae27        .../etcd       "/bin/sh -c ..."     2 days ago          Exited (137) 5 seconds ago

$ kubectl get componentstatuses
NAME                 STATUS    MESSAGE              ERROR
etcd                 Healthy   {"health": "true"}

$ curl -L http://127.0.0.1:2379/health
{"health": "true"}

$ curl -L http://127.0.0.1:2379/health
Failed to connect to 127.0.0.1 port 2379: Connection refused

HTTP probe failed with statuscode: 503
```

Debugging Approaches

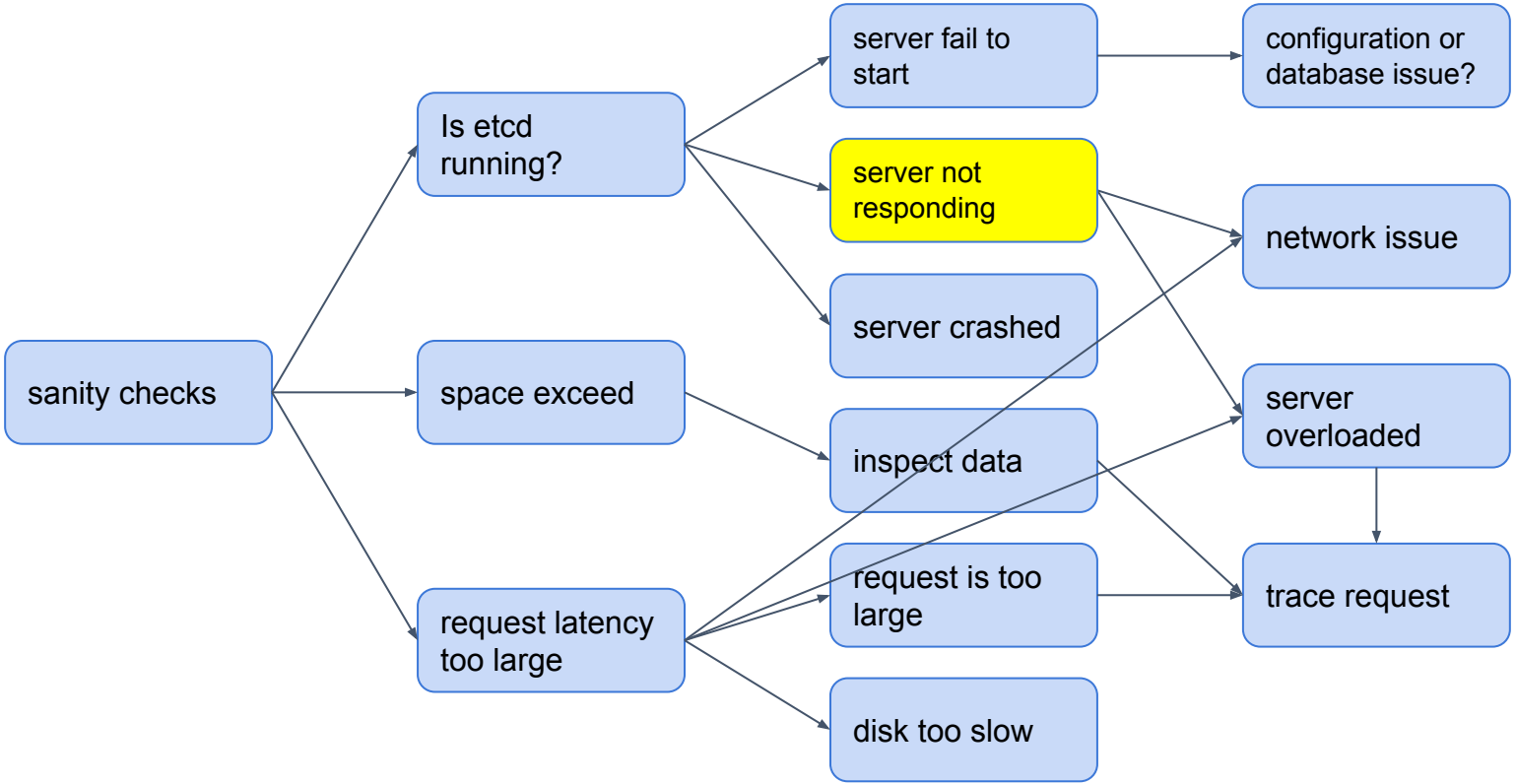
- Sanity checks: is etcd running?

```
$ ETCDCTL_API=3 etcdctl --write-out=table endpoint status
+----------------+------------------+---------+---------+-----------+-----------+------------+
|    ENDPOINT    |        ID        | VERSION | DB SIZE | IS LEADER | RAFT TERM | RAFT INDEX |
+----------------+------------------+---------+---------+-----------+-----------+------------+
| 127.0.0.1:2379 | 3dad195a8fe24bd1 | 3.1.11  | 5.7 MB  | false     |         7 |     260921 |
+----------------+------------------+---------+---------+-----------+-----------+------------+


$ ETCDCTL_API=3 etcdctl --write-out=table member list
+------------------+---------+--------------------+----------------------------+-----------------------+
|        ID        | STATUS  |        NAME        |         PEER ADDRS         |     CLIENT ADDRS      |
+------------------+---------+--------------------+----------------------------+-----------------------+
| 9654975ed4a2f3f  | started | etcd-10.127.240.162 | https://10.127.240.162:2380 | http://127.0.0.1:2379 |
| 241738ddc3b07bb9 | started | etcd-10.127.240.163 | https://10.127.240.163:2380 | http://127.0.0.1:2379 |
| 3dad195a8fe24bd1 | started | etcd-10.127.240.161 | https://10.127.240.161:2380 | http://127.0.0.1:2379 |
+------------------+---------+--------------------+----------------------------+-----------------------+
```

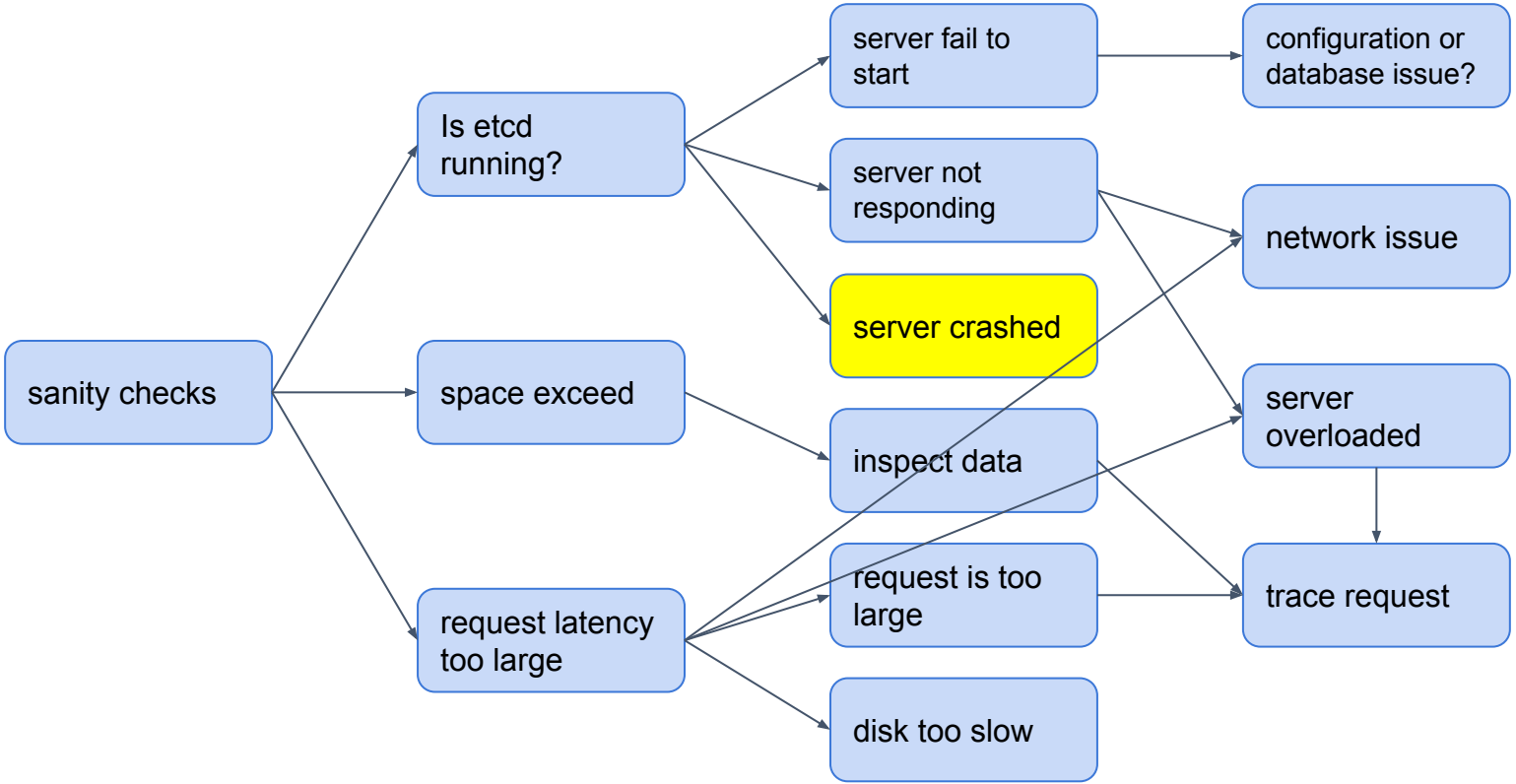
Debugging Approaches



Debugging Approaches



Debugging Approaches



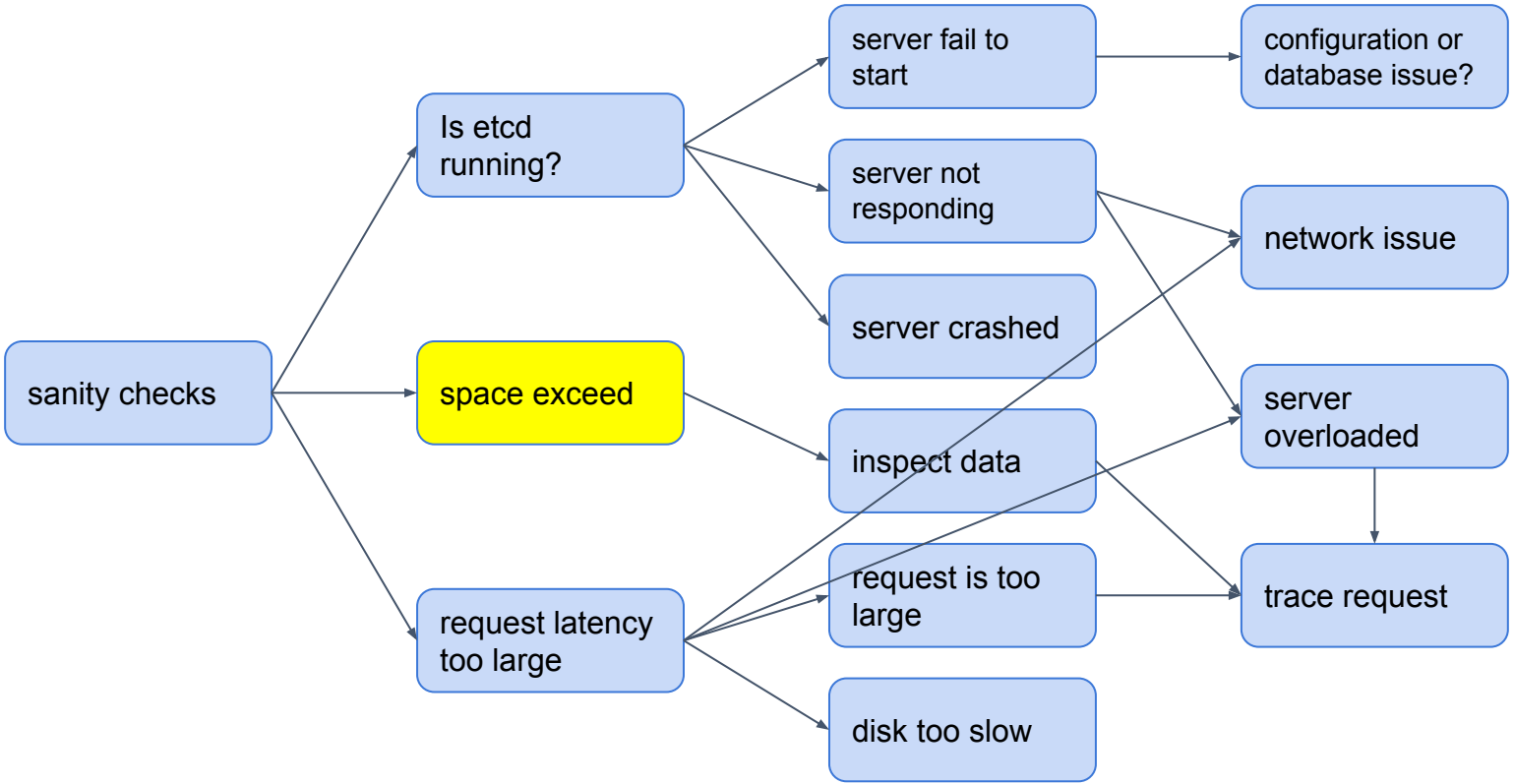
Debugging Approaches

- Sanity checks: is etcd running?

```
$ grep "[CE] |" etcd.log

2017-11-14 23:30:34.340030 E | rafthttp: failed to read 5137d09ebac61b82 on stream MsgApp v2 (context canceled)
2017-11-14 23:30:34.340454 E | rafthttp: failed to read 72f26c9f9da79ea7 on stream Message (context canceled)
2017-11-14 23:31:03.130335 E | rafthttp: failed to read 5137d09ebac61b82 on stream MsgApp v2 (unexpected EOF)
2017-11-14 23:31:30.694572 C | mvcc/backend: cannot commit tx (write agent-1/etcd.data/member/snap/db: file too
large)
```

Example: freelist corruption
https://github.com/etcd-io/bbolt/pull/67

Debugging Approaches

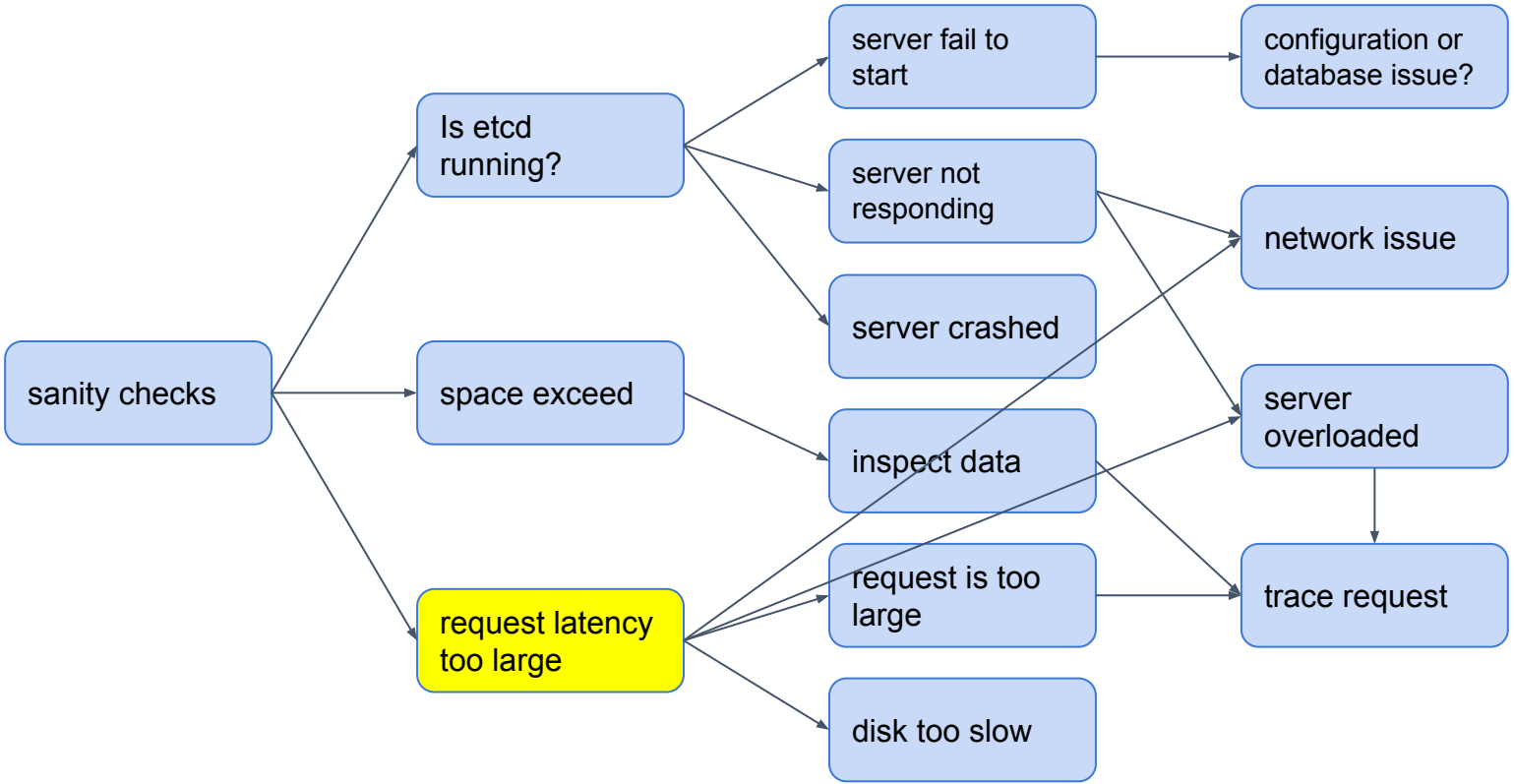


Debugging Approaches

- Sanity checks: space quota exceeded?

```
$ ETCDCTL_API=3 etcdctl --write-out=table endpoint status
+----------------+------------------+---------+---------+-----------+-----------+------------+
|    ENDPOINT    |        ID        | VERSION | DB SIZE | IS LEADER | RAFT TERM | RAFT INDEX |
+----------------+------------------+---------+---------+-----------+-----------+------------+
| 127.0.0.1:2379 | 3dad195a8fe24bd1 | 3.1.11  | 8.0 GB  | false     |         7 |     260921 |
+----------------+------------------+---------+---------+-----------+-----------+------------+


Alternatively, look at 'db' file in the snapshot directory
$ sudo ls -la ${path to etcd data dir}/member/snap/


$ ETCDCTL_API=3 etcdctl alarm list
memberID:3dad195a8fe24bd1 alarm:NOSPACE


Error message:
"etcdserver: mvcc: database space exceeded"
```

Debugging Approaches



Debugging Approaches

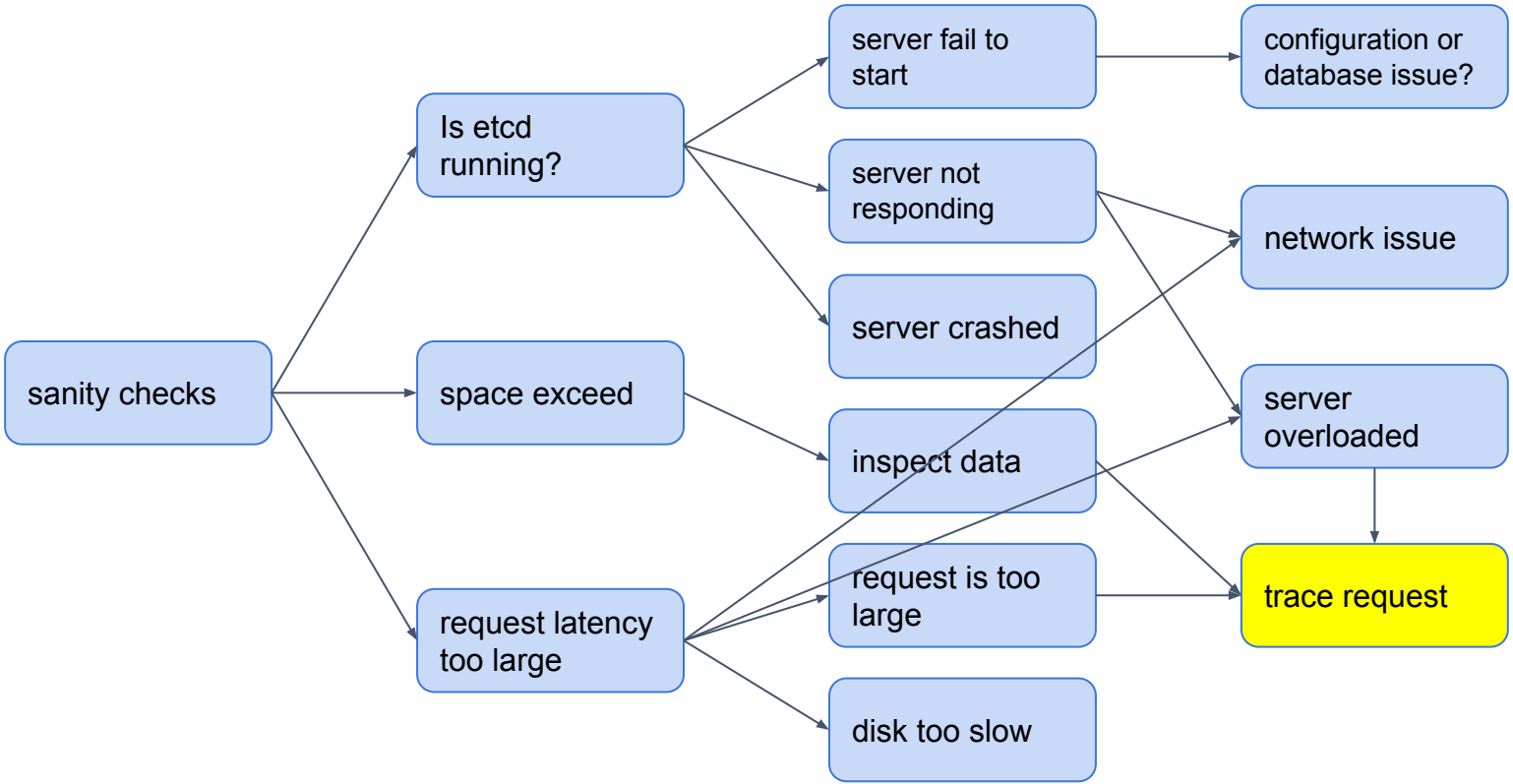
- Sanity checks: request latency too large

```
$ ETCDCTL_API=3 etcdctl --write-out=table endpoint status

Failed to get the status of endpoint 127.0.0.1:2379 (context deadline exceeded)



$ grep “apply entries took too long” etcd.log

...
2018-10-15 20:54:02.963571 W | etcdserver: apply entries took too long [12.66726791s for 1 entries]
2018-10-15 20:54:02.963617 W | etcdserver: avoid queries with large range/delete range!
...
```

Debugging Approaches

- Sanity checks: request latency too large
  - Request size too large?
    - Trace request
  - Server overloaded?
    - Check server resource utilization: CPU starvation, memory swapping
    - Trace request
  - Disk performance
    - /metrics endpoint
    - two disk related metrics:
      - wal_fsync_duration_seconds
      - backend_commit_duration_seconds
  - Networking
    - Could cause slow apply and frequent leader election

Debugging Approaches



Debugging Approaches

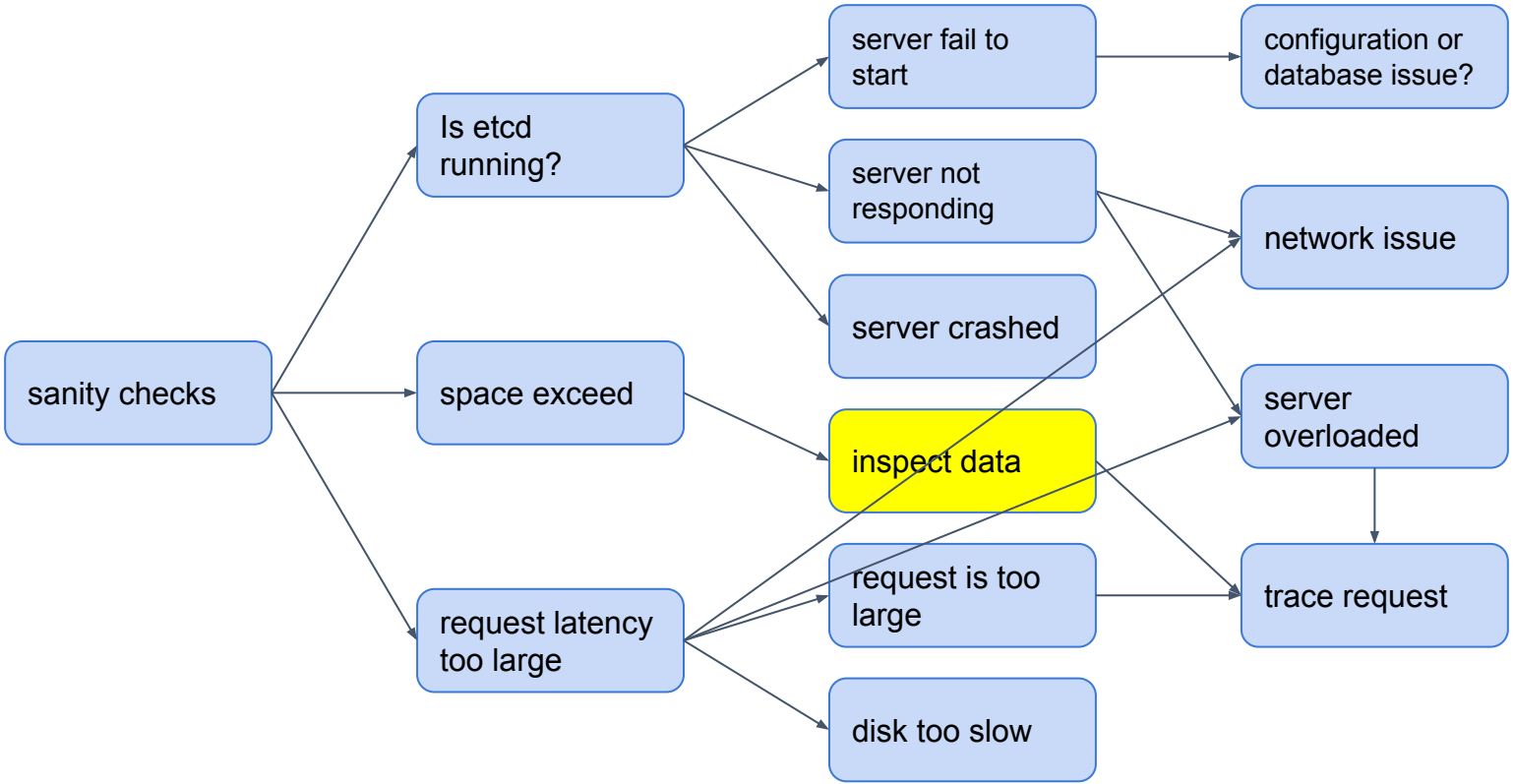
- Trace request

```
$ less kube-apiserver.log | grep "total time" -B 5 -A 5
...
I1016 00:39:03.152718       1 trace.go:76] Trace[2046021670]: "List /apis/batch/v1/jobs" (started:
2018-10-16 00:38:29.832824845 +0000 UTC m=+380.058697937) (total time: 33.319846272s):
Trace[2046021670]: [21.225676154s] [21.225669191s] Listing from storage done
Trace[2046021670]: [33.319842654s] [11.741150944s] Writing http response done (320186 items)
I1016 00:39:03.152947       1 wrap.go:42] GET /apis/batch/v1/jobs: (33.322082585s) 200
[[kube-controller-manager/v1.9.6 (linux/amd64)
kubernetes/cb15136/system:serviceaccount:kube-system:cronjob-controller] [::1]:42464]
...
```

pkg/controller/cronjob/cronjob_controller.go

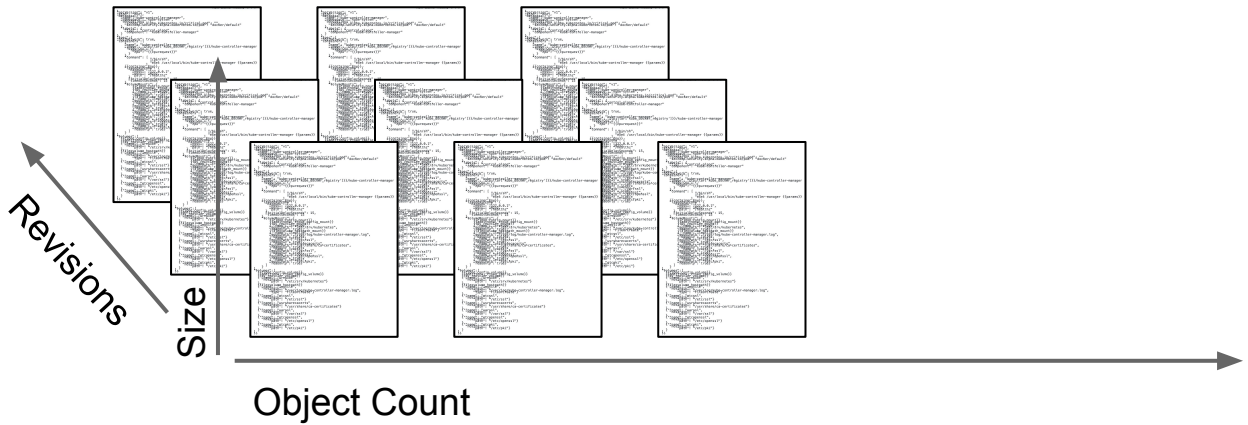
```
func (jm *CronJobController) syncAll() {
      …
      jl, err := jm.kubeClient.BatchV1().Jobs(metav1.NamespaceAll).List(metav1.ListOptions{})     <-
listing from etcd directly, w/o pagination
      …
}
```

Debugging Approaches



Debugging Approaches

- Inspect data



Debugging Approaches

- Inspect data

Space Used ~= # of Objects x Size per Object x Uncompacted Revisions

Proportional to update rate!

Debugging Approaches

- Inspect data
  - Workload can increase data volume
    - Each write creates a new object version.
    - Can be further amplified if workload increase DB fragmentation.
  - Data volume can increase workload
    - Latency of range read (listing) increases with the count of objects returned.
    - Expensive operations could increase latency or even timeout other request.

```
I1016 00:39:03.152718       1 trace.go:76] Trace[2046021670]: "List /apis/batch/v1/jobs" (started:
2018-10-16 00:38:29.832824845 +0000 UTC m=+380.058697937) (total time: 33.319846272s):
Trace[2046021670]: [21.225676154s] [21.225669191s] Listing from storage done
Trace[2046021670]: [21.57869171s] [353.015556ms] Self-linking done
Trace[2046021670]: [33.319842654s] [11.741150944s] Writing http response done (320186 items)
I1016 00:39:03.152947       1 wrap.go:42] GET /apis/batch/v1/jobs: (33.332082585s) 200
[[kube-controller-manager/v1.9.6 (linux/amd64)
kubernetes/cb15136/system:serviceaccount:kube-system:cronjob-controller] [::1]:42464]
```


KubeCon
CloudNativeCon
North America 2018
Keeping your etcd
Healthy

Keeping your etcd Healthy

- Monitoring
  - etcd uses Prometheus for metrics reporting.
  - /metrics endpoint
  - Example grafana dashboard

Keeping your etcd Healthy

- Use officially maintained versions

etcd-dev mailing list, Sept. 6, 2018:

```
If you run etcd in production, please read!

A couple recent issue report on github for both etcd and Kubernetes github have
highlighted the fact that some older versions of etcd contain defects severe enough that
we should avoid running them in production, including a data corruption bug.  Also, with
Kubernetes deprecating etcd 2.x support this year and the officially maintained etcd
versions being 3.1+,

The minimum recommended versions of etcd to run in production are:

3.1.11+
3.2.10+
3.3.0+
```

Keeping your etcd Healthy

- Backup your etcd
  - For disaster recovery purpose.
  - Per backup check
    - ETCDCTL_API=3 etcdctl snapshot status (ONLY for etcdctl v3.3.10+, v3.2.25+, v3.1.20+).
    - bbolt check
  - Regularly validate restoration from the backup files.
- Upgrades
  - Recommend upgrading to officially maintained etcd versions.
  - Refer to Documentation/upgrades for upgrade process.
- Downgrades
  - Currently, only possible if backup the entire etcd data before upgrading.
  - Ongoing: etcd downgrade support for 1 minor version. https://github.com/etcd-io/etcd/issues/9306

How to get involved

- Contact:
  - Email: etcd-dev@googlegroups.com
  - IRC: #etcd IRC channel on freenode.org
  - Community meeting: 11:00 PST Tuesday Monthly. https://github.com/etcd-io/etcd#community-meetings
- Issues and PRs: https://github.com/etcd-io/etcd
- CONTRIBUTING! https://github.com/etcd-io/etcd/blob/master/CONTRIBUTING.md

Thanks!

Joe Betz, Google Jingyi Hu, Google


KubeCon
CloudNativeCon
North America 2018

Extra Slides

Inspecting Load and Data

Load ~= Request Volume x Response object count x Response object size

- Check kube-apiserver logs for high latency or timed out requests
- Check /var/log/etcd.log for slow operation warnings (“entries took too long...”)
- Check WAL log with etcd-dump-logs
- Check etcd object counts with auger or etcdctl

Keeping your etcd Healthy

Object count quotas? Rate limits?

How do prevent accidental (or deliberate) misuse from crashing control planes?


KubeCon
CloudNativeCon
North America 2018
Tools of the Trade

Tools of the Trade

- **etcdctl** - etcd CLI
- **auger** - data inspection
- **etcd-dump-logs** - RAFT log inspection

Tools of the Trade


```
$ ETCDCTL_API=3 etcdctl

NAME:
        etcdctl - A simple command line client for etcd3.

USAGE:
        etcdctl

VERSION:
        3.3.0

COMMANDS:
        get                     Gets the key or a range of keys
        put                     Puts the given key into the store
        del                     Removes the specified key or range of keys [key, range_end)
        txn                     Txn processes all the requests in one transaction
        compaction              Compacts the event history in etcd
        alarm disarm            Disarms all alarms
        alarm list              Lists all alarms
        defrag                  Defragments the storage of the etcd members with given endpoints
        endpoint health         Checks the healthiness of endpoints specified in `--endpoints` flag
        endpoint status         Prints out the status of endpoints specified in `--endpoints` flag
        watch                   Watches events stream on keys or prefixes
        version                 Prints the version of etcdctl
        lease grant             Creates leases
        lease revoke            Revokes leases
        lease keep-alive        Keeps leases alive (renew)
        member add              Adds a member into the cluster
        member remove           Removes a member from the cluster
...
```

Tools of the Trade


```
$ ETCDCTL_API=3 etcdctl get --prefix --keys-only /

/registry/apiregistration.k8s.io/apiservices/v1.

/registry/apiregistration.k8s.io/apiservices/v1.authentication.k8s.io

/registry/apiregistration.k8s.io/apiservices/v1.authorization.k8s.io

/registry/apiregistration.k8s.io/apiservices/v1.autoscaling

/registry/apiregistration.k8s.io/apiservices/v1.batch

...
```

Tools of the Trade


```
$ ETCDCTL_API=3 etcdctl get /registry/pods/kube-system/kube-dns-xxxxxx-vwh

/registry/pods/kube-system/kube-dns-xxxxxx-vwh
k8s

v1Pod
kube-dns-xxxxxx-vwhzwkube-dns-778977457c-                    kube-system"*$df815185-022f-11e8-96ed-42010a8000922
k8s-appkube-dnsZ
pod-template-hash
3345330137b
kubernetes.io/created-by
{"kind":"SerializedReference","apiVersion":"v1","reference":{"kind":"ReplicaSet","namespace":"kube-system","name":"kube-dns-77
8977457c","uid":"df6419c0-022f-11e8-96ed-42010a800092","apiVersion":"extensions","resourceVersion":"288"}}
b.
*scheduler.alpha.kubernetes.io/critical-podj_

...
```

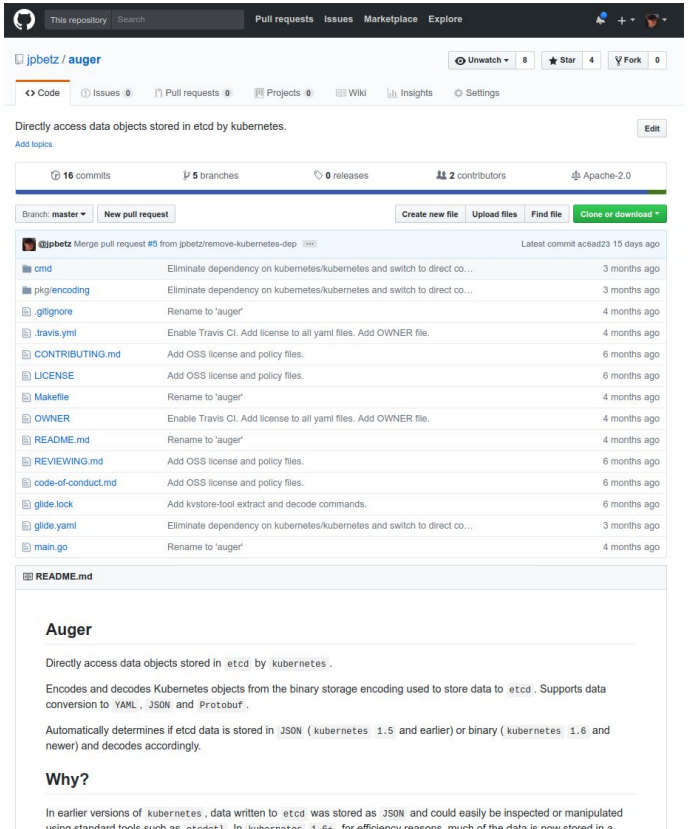
PROTOBUF

Tools of the Trade


```
$ auger --help

Inspect and analyze kubernetes objects in binary storage
encoding used with etcd 3+ and boltdb.

Usage:
  auger [command]

Available Commands:
  decode      Decode objects from the kubernetes binary key-value store encoding.
  encode      Encode objects to the kubernetes binary key-value store encoding.
  extract     Extracts kubernetes data from the boltdb '.db' files etcd persists to.
  help        Help about any command

Flags:
  -h, --help   help for auger

Use "auger [command] --help" for more information about a command.
```

Source at github.com/jpbetz/auger

Contributions welcome!



Tools of the Trade


```
$ auger extract -f $DB_FILE

/registry/apiregistration.k8s.io/apiservices/v1.
/registry/apiregistration.k8s.io/apiservices/v1.authentication.k8s.io
/registry/apiregistration.k8s.io/apiservices/v1.authorization.k8s.io
/registry/apiregistration.k8s.io/apiservices/v1.autoscaling
/registry/apiregistration.k8s.io/apiservices/v1.batch
/registry/apiregistration.k8s.io/apiservices/v1.networking.k8s.io
/registry/apiregistration.k8s.io/apiservices/v1.storage.k8s.io
/registry/apiregistration.k8s.io/apiservices/v1beta1.apiextensions.k8s.io
/registry/apiregistration.k8s.io/apiservices/v1beta1.apps
/registry/apiregistration.k8s.io/apiservices/v1beta1.authentication.k8s.io
/registry/apiregistration.k8s.io/apiservices/v1beta1.authorization.k8s.io
...
```

Tools of the Trade

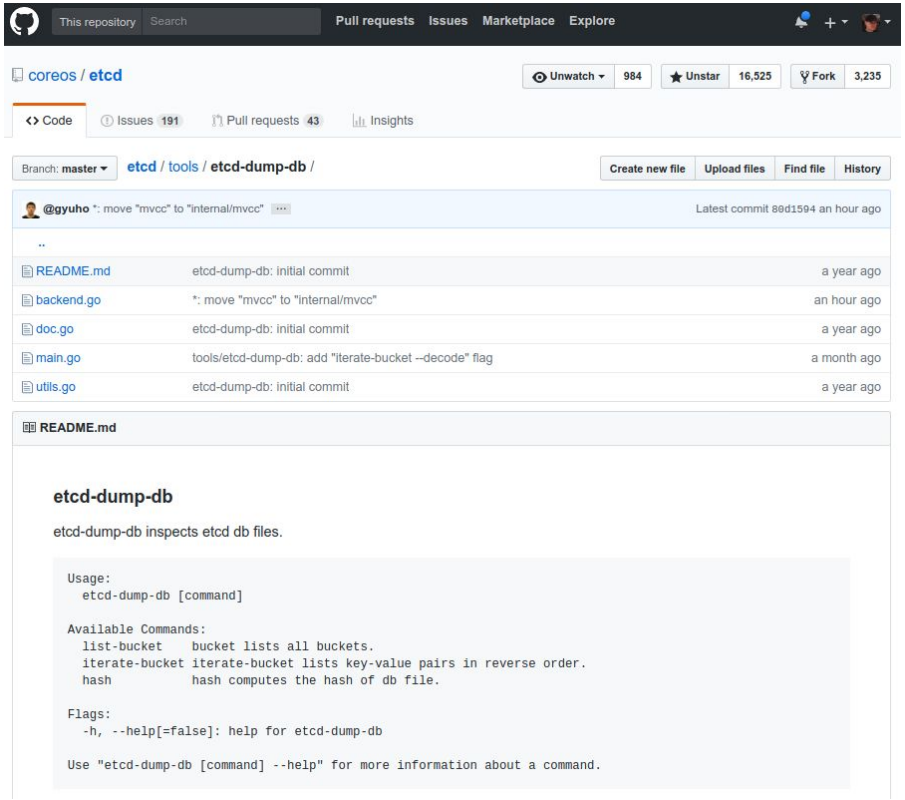

```
$ auger extract -f $DB_FILE --fields key,value-size

/registry/apiregistration.k8s.io/apiservices/v1. 590
/registry/apiregistration.k8s.io/apiservices/v1.authentication.k8s.io 665
/registry/apiregistration.k8s.io/apiservices/v1.authorization.k8s.io 662
/registry/apiregistration.k8s.io/apiservices/v1.autoscaling 635
/registry/apiregistration.k8s.io/apiservices/v1.batch 617
/registry/apiregistration.k8s.io/apiservices/v1.networking.k8s.io 653
/registry/apiregistration.k8s.io/apiservices/v1.storage.k8s.io 644
/registry/apiregistration.k8s.io/apiservices/v1beta1.apiextensions.k8s.io 676
/registry/apiregistration.k8s.io/apiservices/v1beta1.apps 628
/registry/apiregistration.k8s.io/apiservices/v1beta1.authentication.k8s.io 679
/registry/apiregistration.k8s.io/apiservices/v1beta1.authorization.k8s.io 676
...
```

Tools of the Trade


```
$ auger extract -f $DB_FILE --fields all-versions-value-size,version-count,key | sort -n

174930 42 /registry/minions/gke-demo-default-pool-912fd0f4-vw4p
...
590 1 /registry/apiregistration.k8s.io/apiservices/v1.
665 1 /registry/apiregistration.k8s.io/apiservices/v1.authentication.k8s.io
```

Tools of the Trade


```
$ etcd-dump-logs -h
Usage of ./etcd-dump-logs:
  -data-dir string

  -start-index uint
        The index to start dumping
  -start-snap string
        The base name of snapshot file to start dumping
```



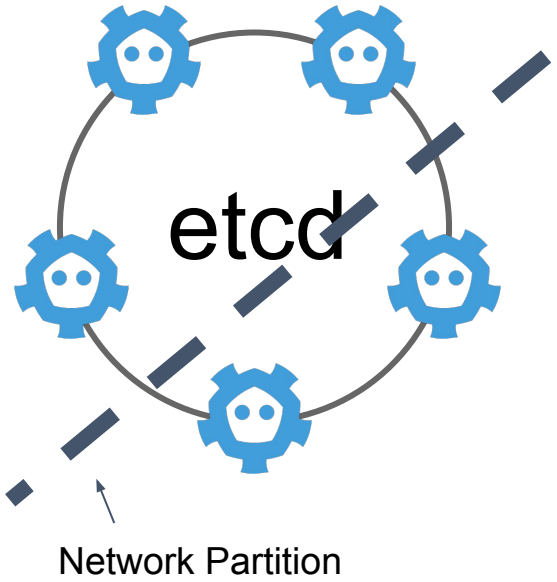
Tools of the Trade


```
$ etcd-dump-logs /var/etcd/data
...
   2        930433      norm    method=SYNC time="2018-01-26 20:08:55.64006588 +0000 UTC"
   2        930434      norm    header:<ID:5163765266442295869 > range:<key:"/registry/configmaps/kube-system/"
range_end:"/registry/configmaps/kube-system0" >
   2        930435      norm    header:<ID:5163765266442295870 > range:<key:"/registry/services/endpoints/kube-system/"
range_end:"/registry/services/endpoints/kube-system0" >
   2        930436      norm    header:<ID:5163765266442295871 > range:<key:"/registry/persistentvolumeclaims/kube-system/"
range_end:"/registry/persistentvolumeclaims/kube-system0" >
   2        930437      norm    header:<ID:5163765266442295872 > range:<key:"/registry/pods/kube-system/"
range_end:"/registry/pods/kube-system0" >
   2        930438      norm    header:<ID:5163765266442295873 > range:<key:"/registry/controllers/kube-system/"
range_end:"/registry/controllers/kube-system0" >
   2        930439      norm    header:<ID:5163765266442295874 > range:<key:"/registry/secrets/kube-system/"
range_end:"/registry/secrets/kube-system0" >
…
```

etcd Recap

RAFT Terms:

- Partition Tolerance
- Leader Election (Leader, Followers, …)
- Quorum



Deep dive: etcd
Wednesday, December 12 • 11:40am - 12:15pm
https://sched.co/JAo2

The Life of a Kubernetes Watch Event
Thursday, December 13 • 4:30pm - 5:05pm
https://sched.co/GrUX

How shall I help with etcd development

- Contact:
  - Email: etcd-dev@googlegroups.com
  - IRC: #etcd IRC channel on freenode.org
  - Community meeting: 11:00 PST Tuesday Biweely https://github.com/etcd-io/etcd#community-meetings
- Issues and PRs: https://github.com/etcd-io/etcd
- CONTRIBUTING! https://github.com/etcd-io/etcd/blob/master/CONTRIBUTING.md

Scalability

Can etcd scale horizontally? No, RAFT global consistency and high availability at the cost of funneling all operations through a leader.

Limits:

- 4 GB total data limit (enforced default), 8 GB supported [--quota-backend-bytes]
- 1.5 MB object limit (enforced default) [--max-request-bytes]
- ~50k watchers
- ~200 writes/s per client connection, ~20k writes/s total
- ~500 reads/s per client connection, ~50k reads/s total

Based on etcd 3.2+, GCE n1-standard-2 machine type with 7.5 GB memory, 2x CPUs

Components are docker containers in the Master Kubelet


```
$ kubectl get nodes
NAME                          STATUS           ROLES    AGE     VERSION
kubernetes-master             Ready            <none>   5h58m   v1.13.0-alpha.0.2315+b11211ed8cfbf5-dirty
kubernetes-minion-group-6xj5  Ready            <none>   5h58m   v1.13.0-alpha.0.2315+b11211ed8cfbf5-dirty
kubernetes-minion-group-9pq9  Ready            <none>   5h58m   v1.13.0-alpha.0.2315+b11211ed8cfbf5-dirty
kubernetes-minion-group-c9sx  Ready            <none>   5h58m   v1.13.0-alpha.0.2315+b11211ed8cfbf5-dirty

$ docker ps
CONTAINER ID  IMAGE                     COMMAND  CREATED      STATUS      NAMES
77a9877ea212  k8s.gcr.io/etcd           "..."    4 hours ago  Up 4 hours  k8s_etcd-container_etcd-server...
554af5042894  c667a020c3ca              "..."    4 hours ago  Up 4 hours  k8s_kube-scheduler...
ec9263161265  99b428320f67              "..."    4 hours ago  Up 4 hours  k8s_kube-apiserver...
a18e666fa976  a4512aa017c1              "..."    4 hours ago  Up 4 hours  k8s_kube-controller-manager...

$ kubectl get componentstatuses
NAME                 STATUS      MESSAGE            ERROR
controller-manager   Healthy     ok
scheduler            Healthy     ok
etcd                 Healthy   {"health":true}

$ curl http://localhost:${COMPONENT_PORT}/healthz
Ok

$ curl http://localhost:${COMPONENT_PORT}/metrics
...
```