etcd

Gopherfest
16 May 2016

Gyu-Ho Lee
CoreOS

Welcome

Thanks for coming!

Thanks for inviting me to speak!

Slides are available at:

- github.com/gyuho/presentations (https://github.com/gyuho/presentations)

Agenda

- What is etcd
- Why
- Go
- Q/A

What is etcd

etcd is ...

- Distributed key-value store
- Go
- Open source
- github.com/coreos/etcd (https://github.com/coreos/etcd)
- Google, Red Hat, EMC, Cisco, Huawei, Baidu, Alibaba...

Distributed?

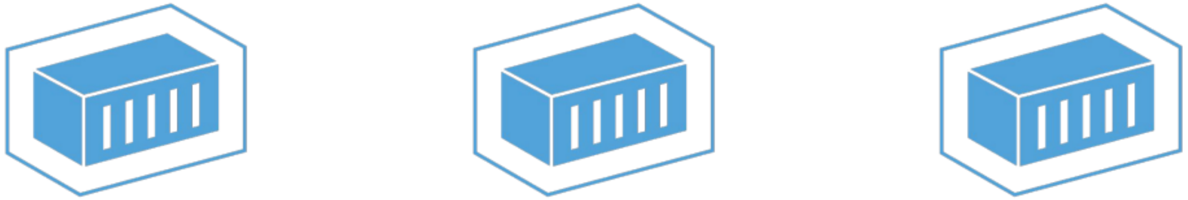
play.etcd.io (http://play.etcd.io)

Join me!

Why etcd

etcd to store configuration

Updates



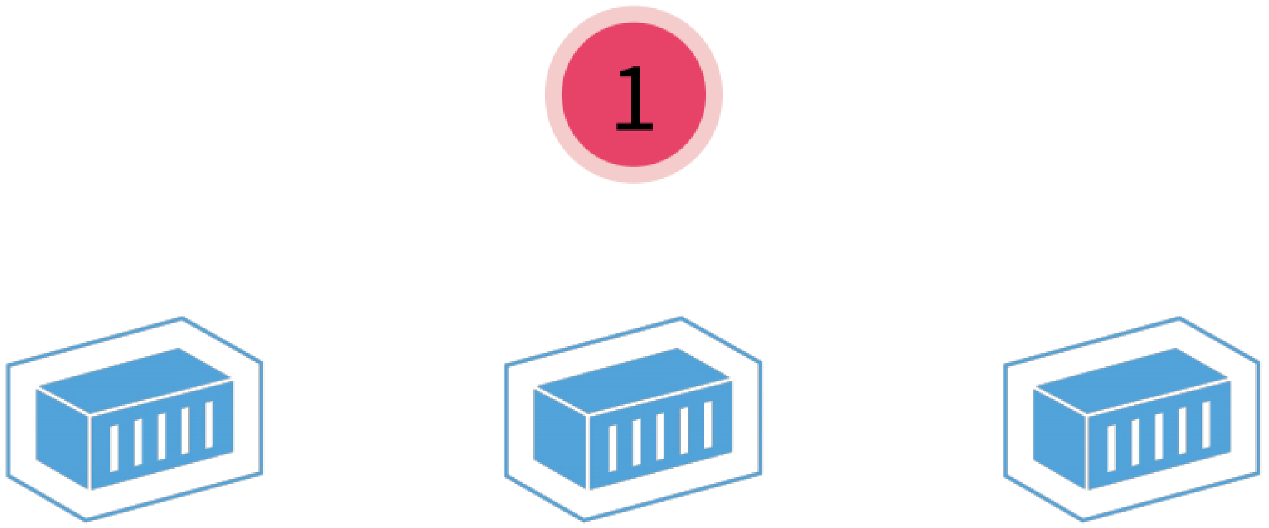
How would you update the cluster of machines?

Traditional way

- Reboot with downtime
- Manual

CoreOS updates with etcd

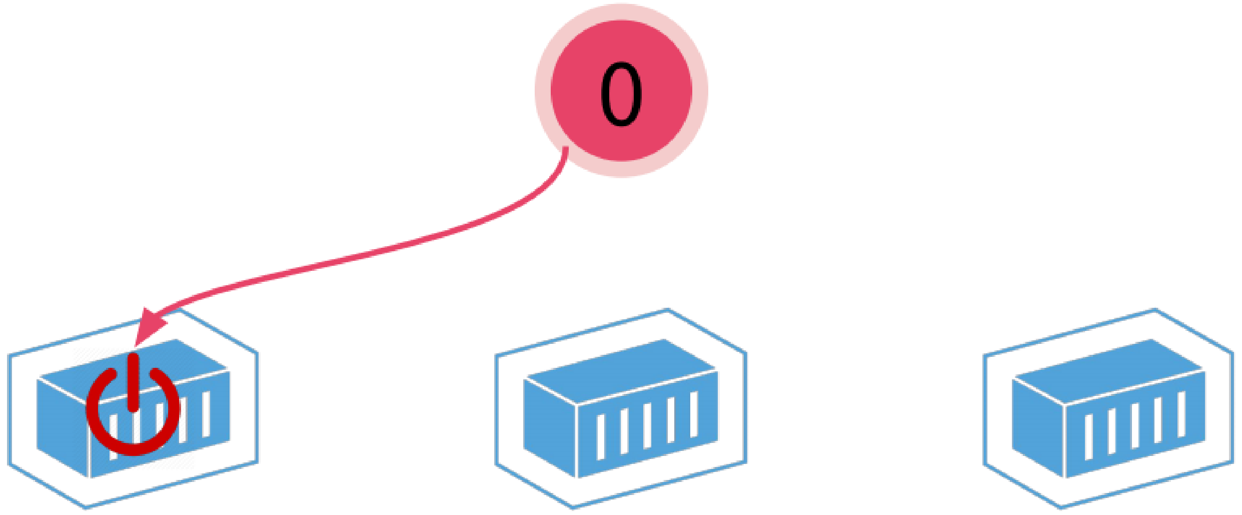
- CoreOS is an open-source Linux OS
- Automatic, No-downtime updates with etcd



Powered by etcd and github.com/coreos/locksmith (https://github.com/coreos/locksmith)

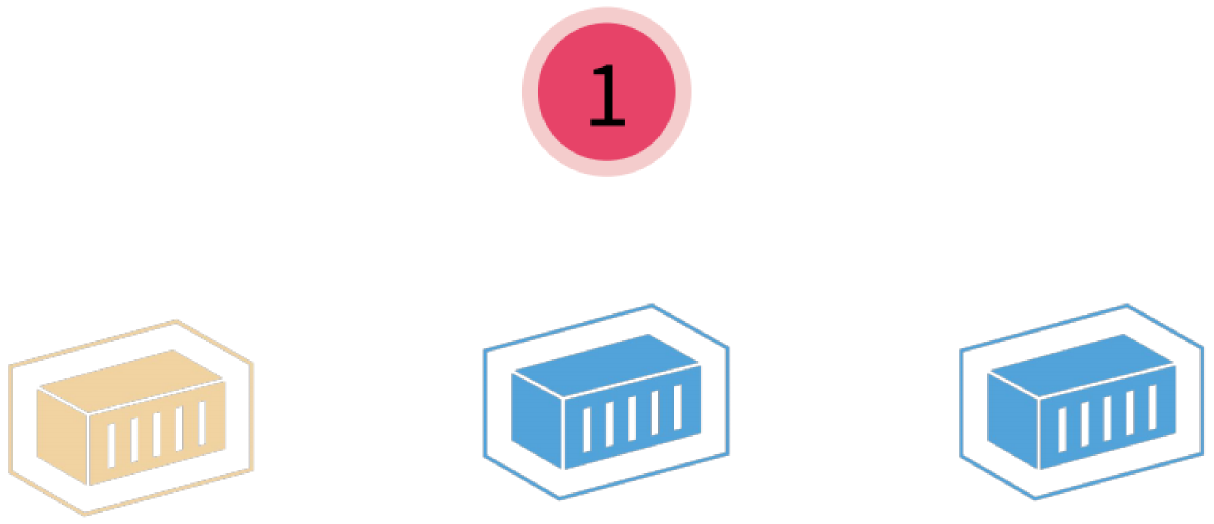
CoreOS updates with etcd

- locksmith stores semephore values in etcd
- ensure that only a subset of a cluster of machines are rebooting at any given time

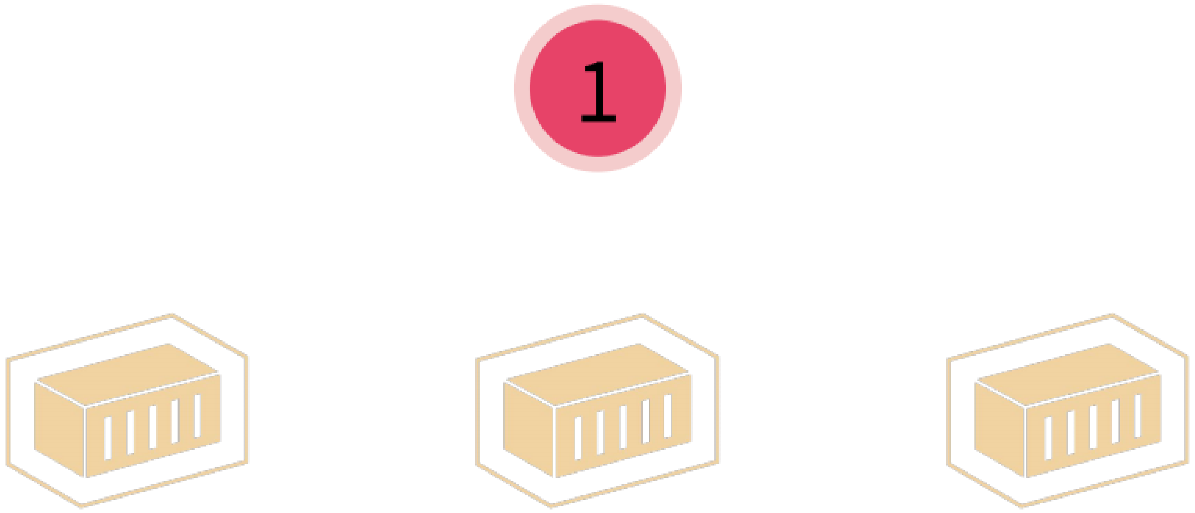


CoreOS updates with etcd

- Automatic
- No downtime



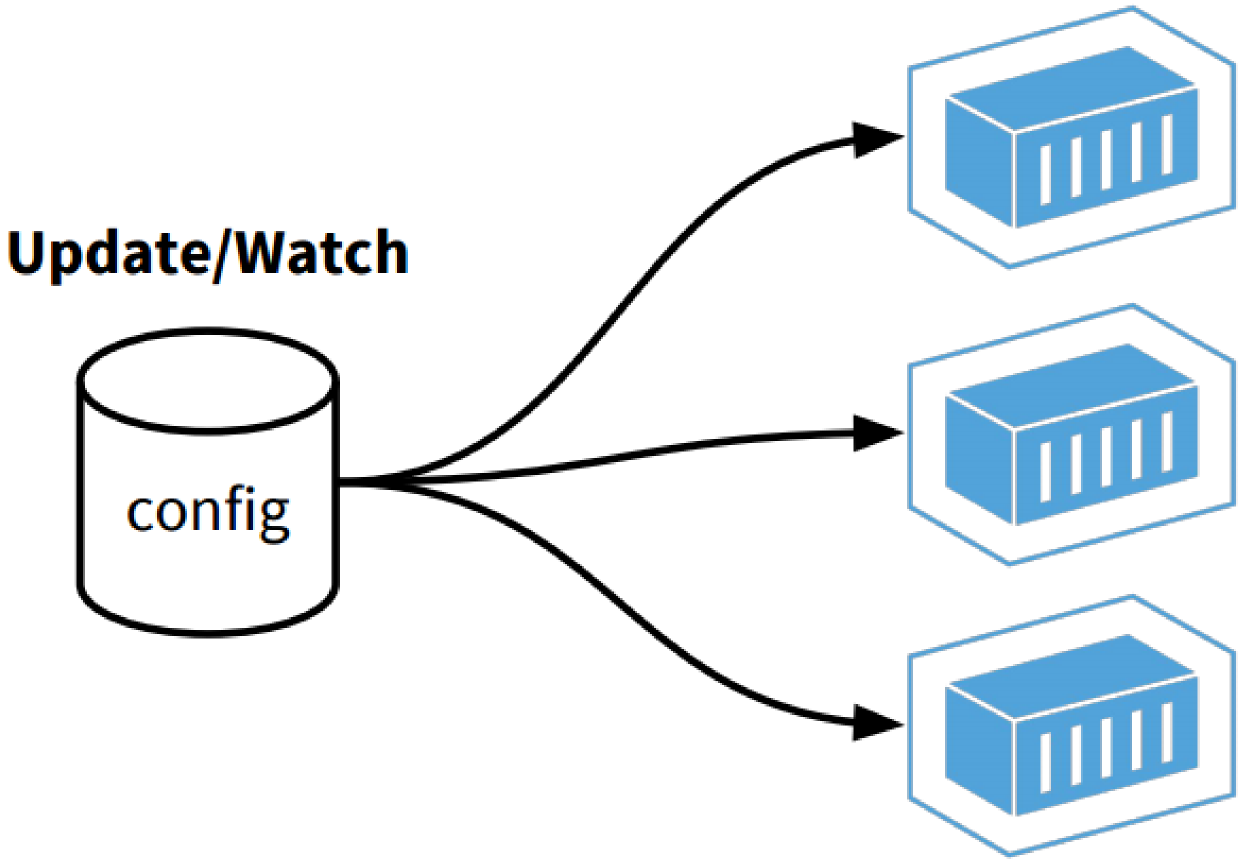
CoreOS updates with etcd



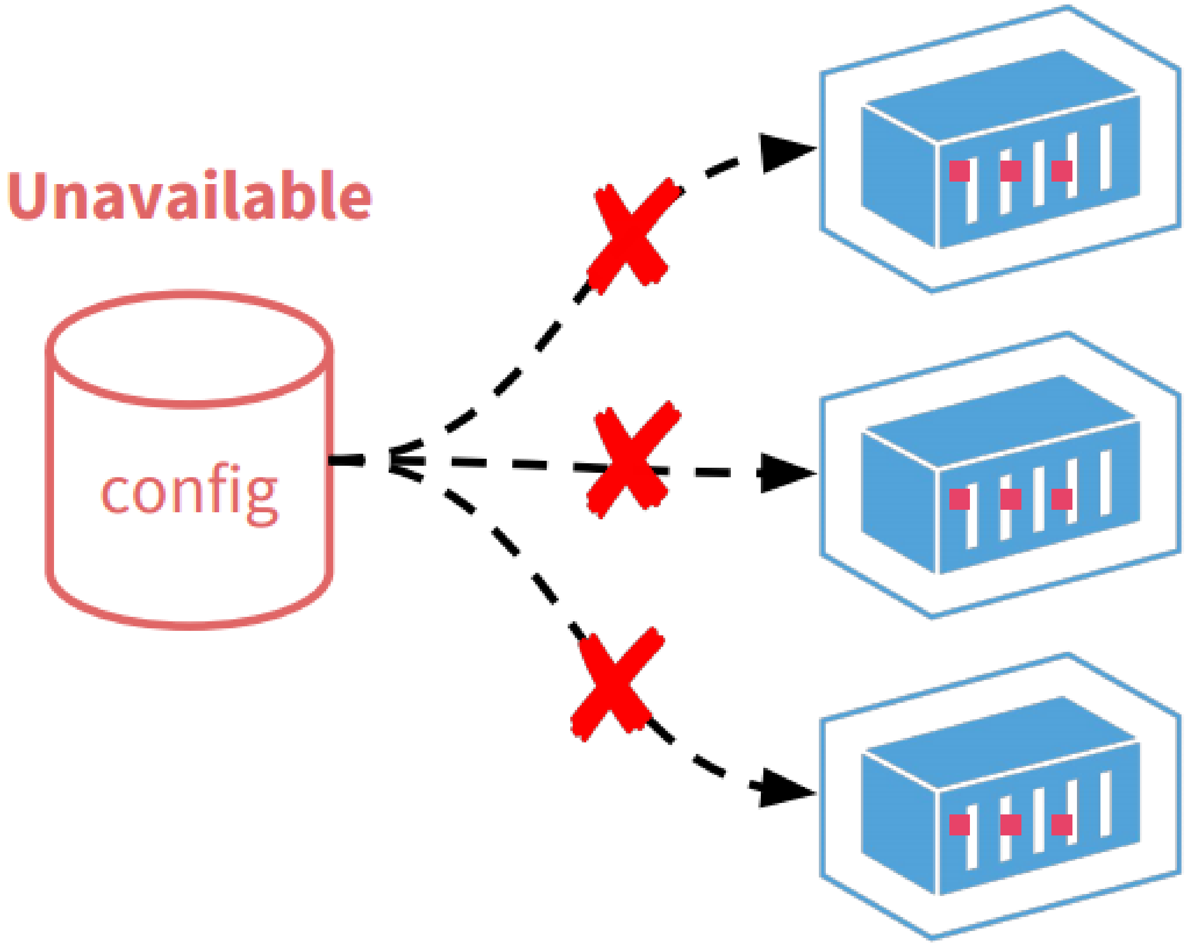
Your cluster is now secured

etcd for "critical" configuration

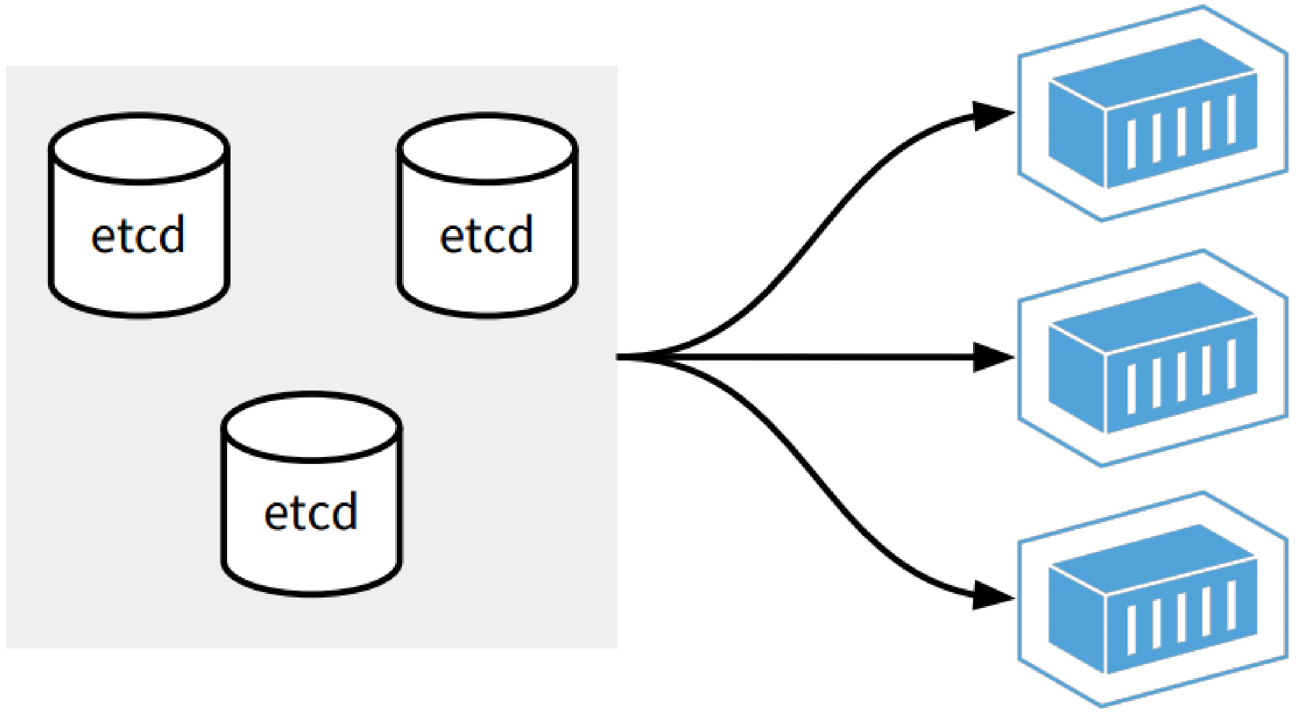
Bad practice



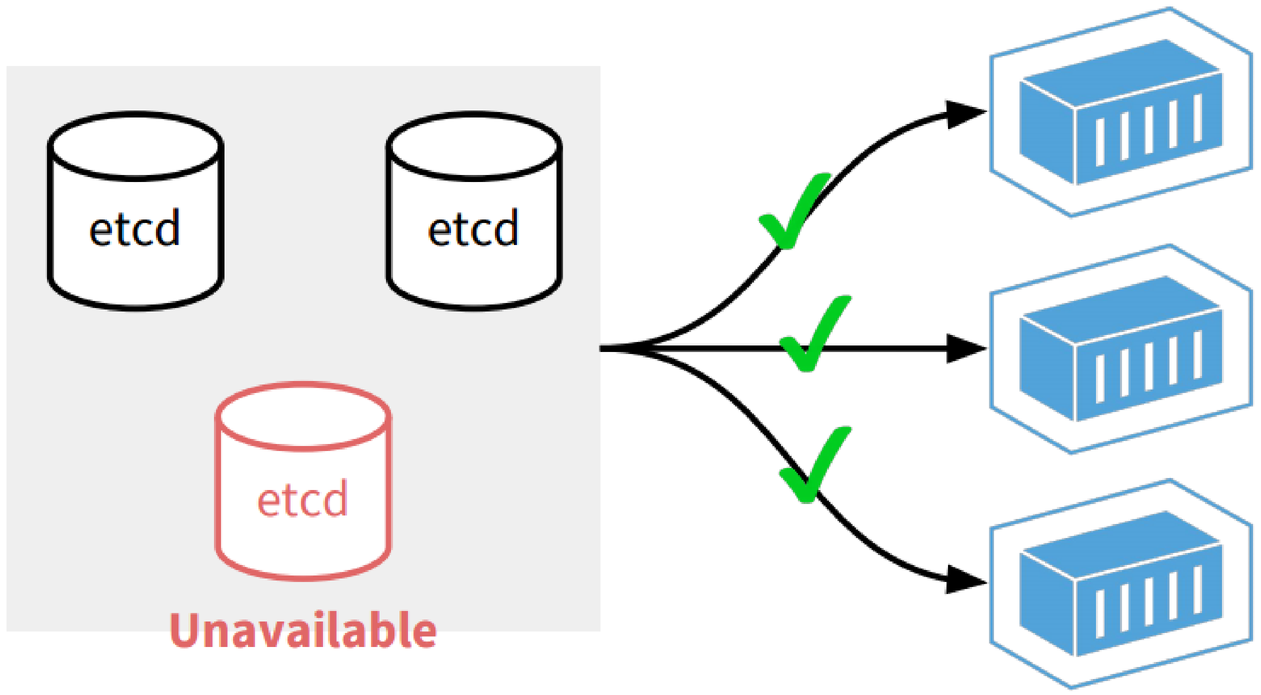
Bad practice



Good practice



Good practice



etcd

Consistent view of critical configuration

- mutual exclusive at any time for locking purpose

Highly available

- resilient to single points of failure & network partitions

Watchable

- push configuration updates to application

Why not Zookeeper or Consul?

etcd Project Status

etcd Performance

etcd v3

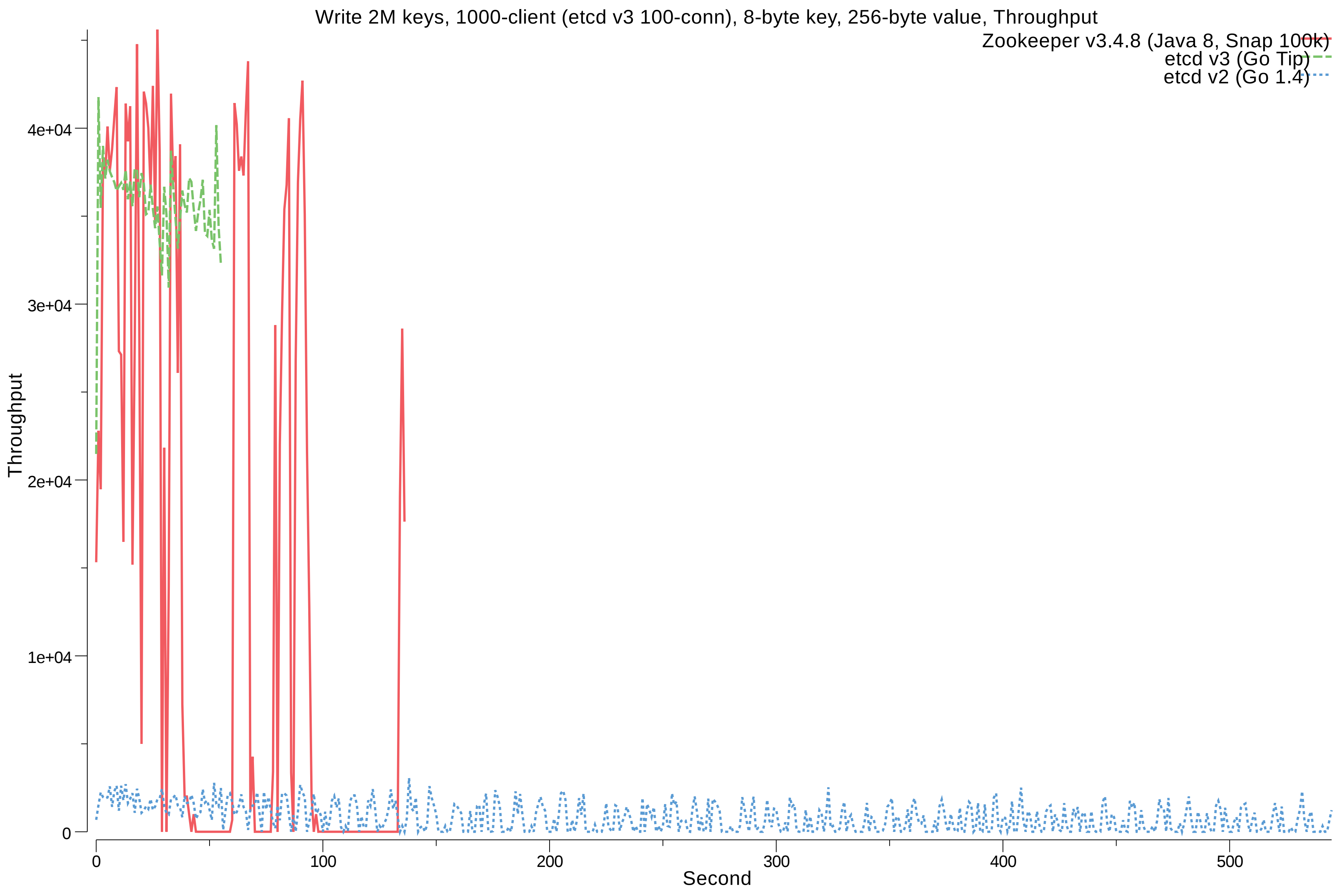
BoltDB (https://github.com/boltdb/bolt)

- B+tree disk storage
- Incremental snapshot
- vs Zookeeper snapCount 10,000

gRPC (http://www.grpc.io/)

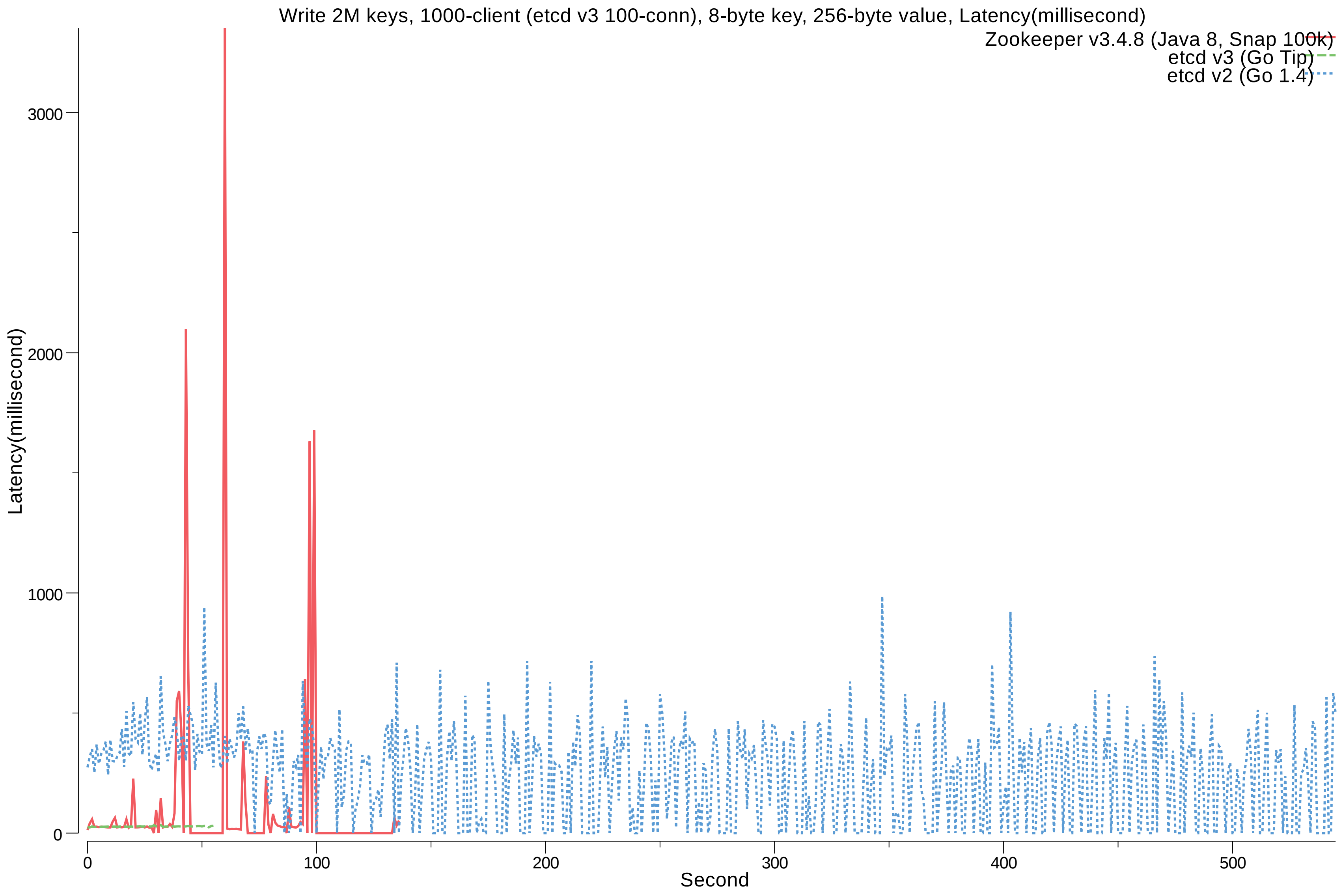
- Protocol Buffer
- HTTP/2
- streams, less TCP congestions

Throughput



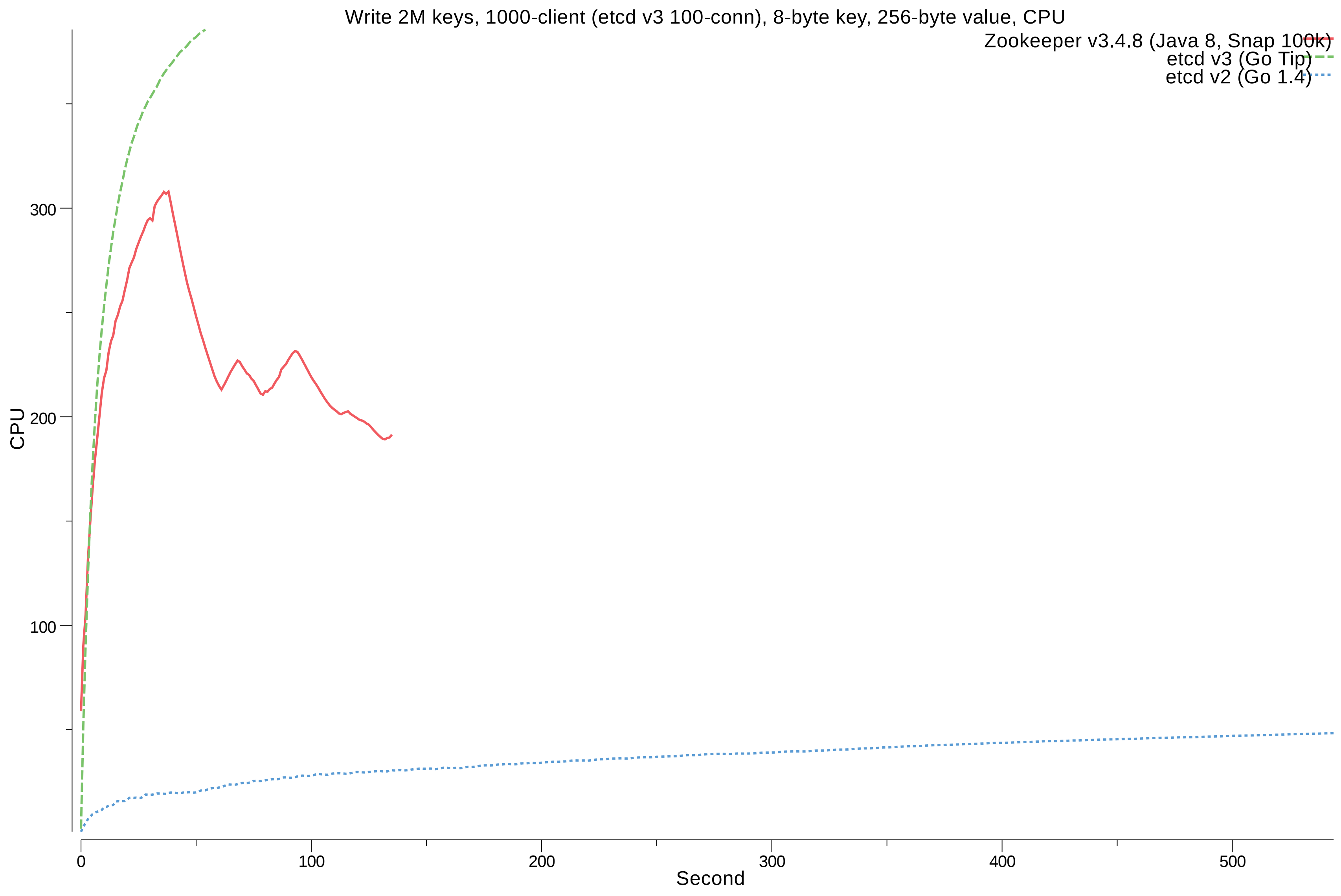
Write 2M keys, 1000-client (etcd v3 100-conn), 8-byte key, 256-byte value, Throughput
Zookeeper v3.4.8 (Java 8, Snap 100k)
etcd v3 (Go Tip)
etcd v2 (Go 1.4)
Throughput
0
1e+04
2e+04
3e+04
4e+04
Second
0
100
200
300
400
500

Latency

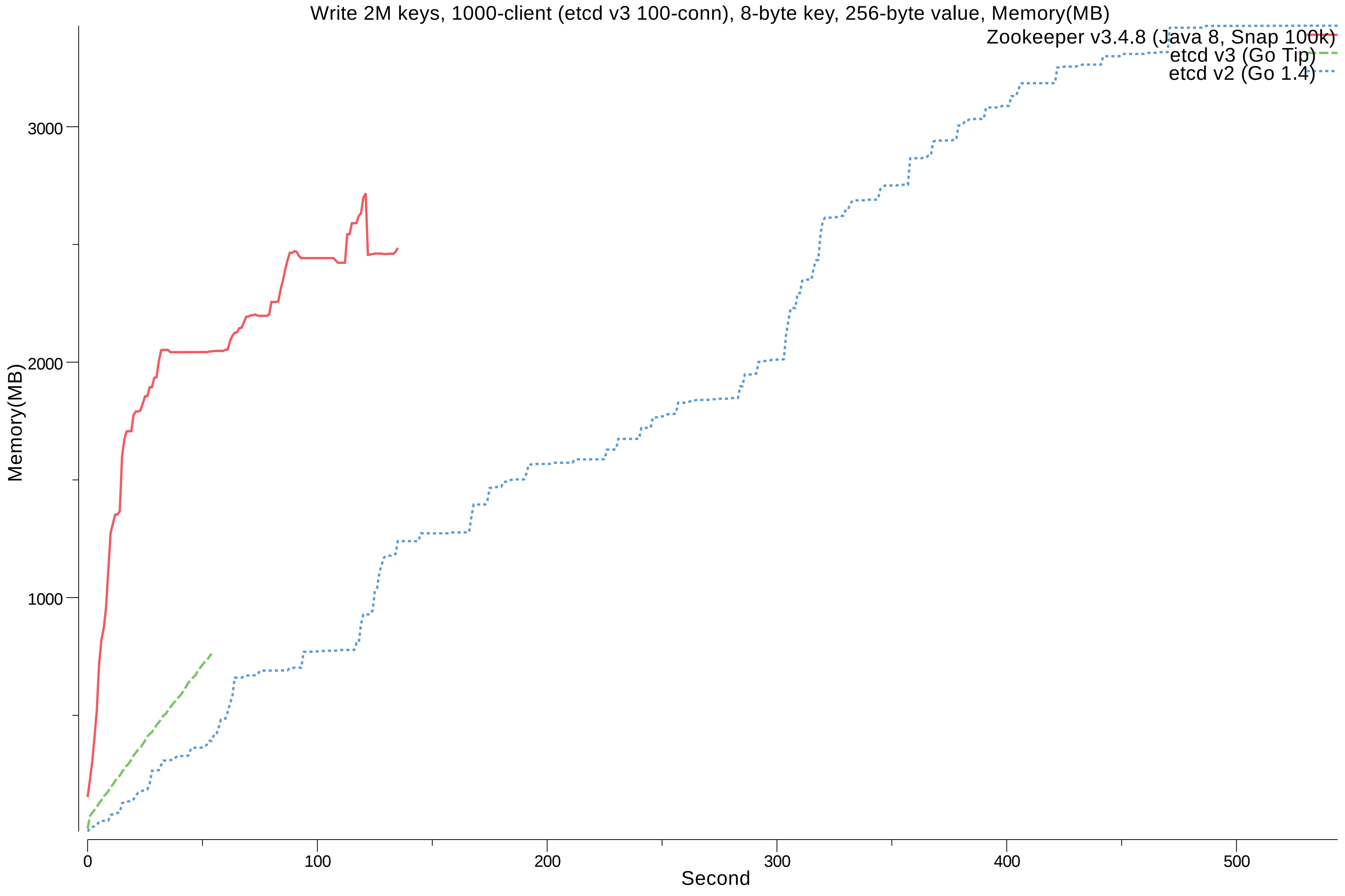


Write 2M keys, 1000-client (etcd v3 100-conn), 8-byte key, 256-byte value, Latency(millisecond)
Zookeeper v3.4.8 (Java 8, Snap 100k)
etcd v3 (Go Tip)
etcd v2 (Go 1.4)
Latency(millisecond)
3000
2000
1000
0
Second
0
100
200
300
400
500

CPU

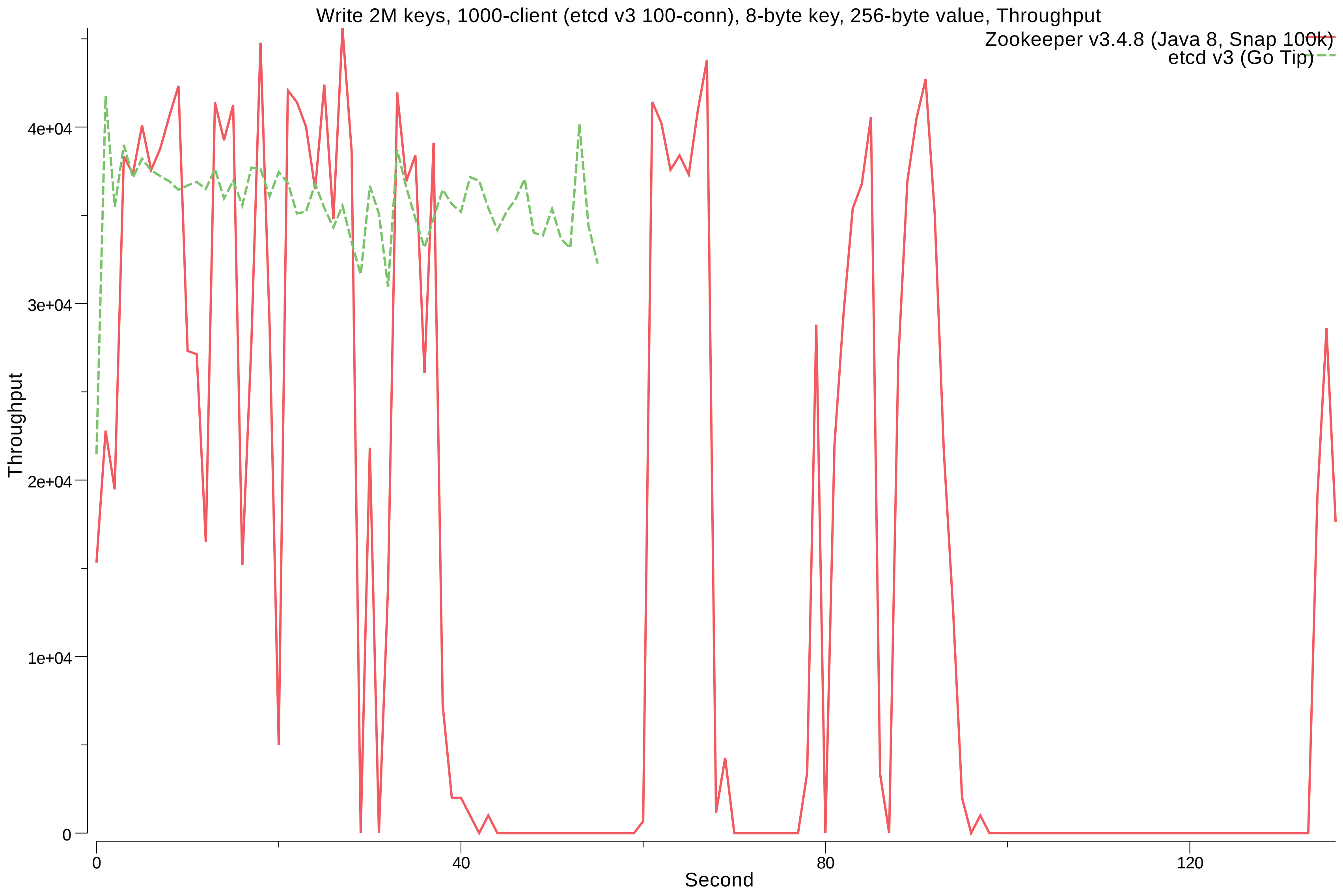

Write 2M keys, 1000-client (etcd v3 100-conn), 8-byte key, 256-byte value, CPU
Zookeeper v3.4.8 (Java 8, Snap 100k)
etcd v3 (Go Tip)
etcd v2 (Go 1.4)
CPU
300
200
100
0
100
200
300
400
500
Second

Memory

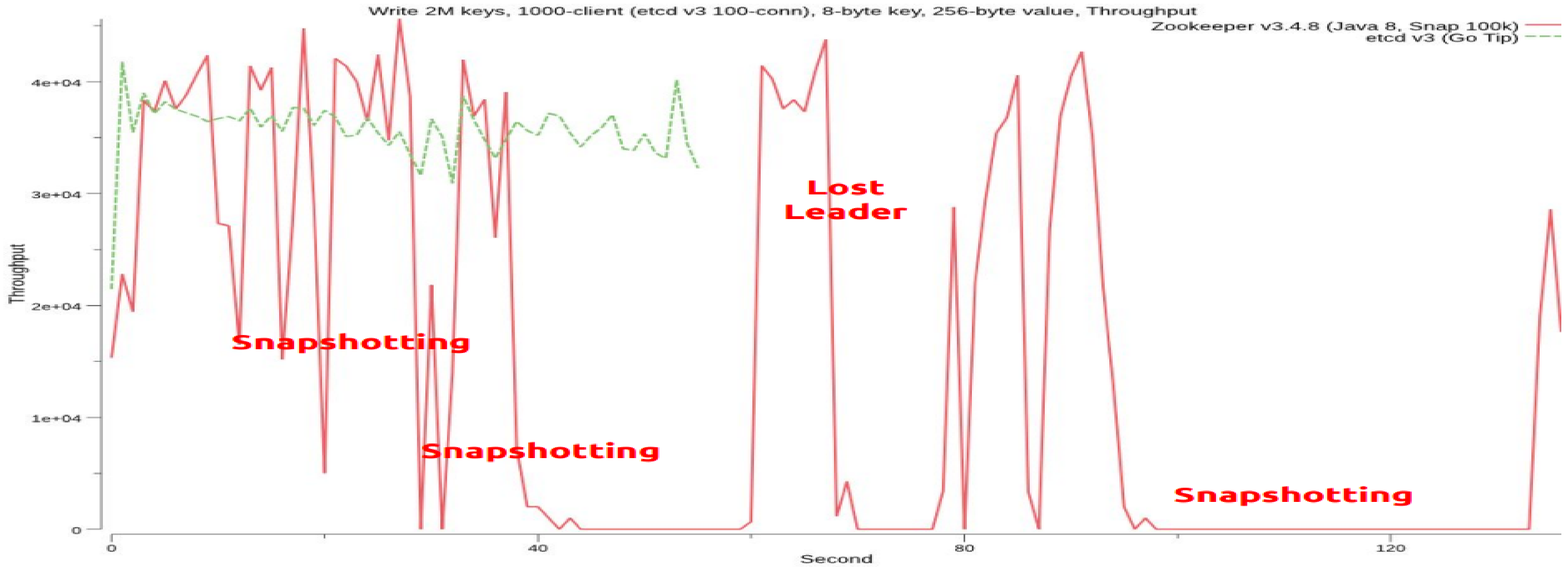

Write 2M keys, 1000-client (etcd v3 100-conn), 8-byte key, 256-byte value, Memory(MB)
Zookeeper v3.4.8 (Java 8, Snap 100k)
etcd v3 (Go Tip)
etcd v2 (Go 1.4)
Memory(MB)
3000
2000
1000
0
100
200
300
400
500
Second

etcd Reliability

Throughput


Write 2M keys, 1000-client (etcd v3 100-conn), 8-byte key, 256-byte value, Throughput
Zookeeper v3.4.8 (Java 8, Snap 100k)
etcd v3 (Go Tip)
Throughput
4e+04
3e+04
2e+04
1e+04
0
0
40
80
120
Second

etcd Reliability



Zookeeper logs

```
07:16:35 [Snapshot Thread:FileTxnSnapLog@240] - Snapshotting...
07:16:43 fsync-ing the write ahead log in SyncThread:3 took 1224ms...
07:16:46 fsync-ing the write ahead log in SyncThread:3 took 3205ms... // Snapshotting
...
07:17:14 [FastLeaderElection@818] - New election...                  // Leader Election
```

etcd Reliability

We test...

- Kill all members
- Kill majority of members
- Kill leader
- Network partition
- Network latency
- More...

etcd Reliability

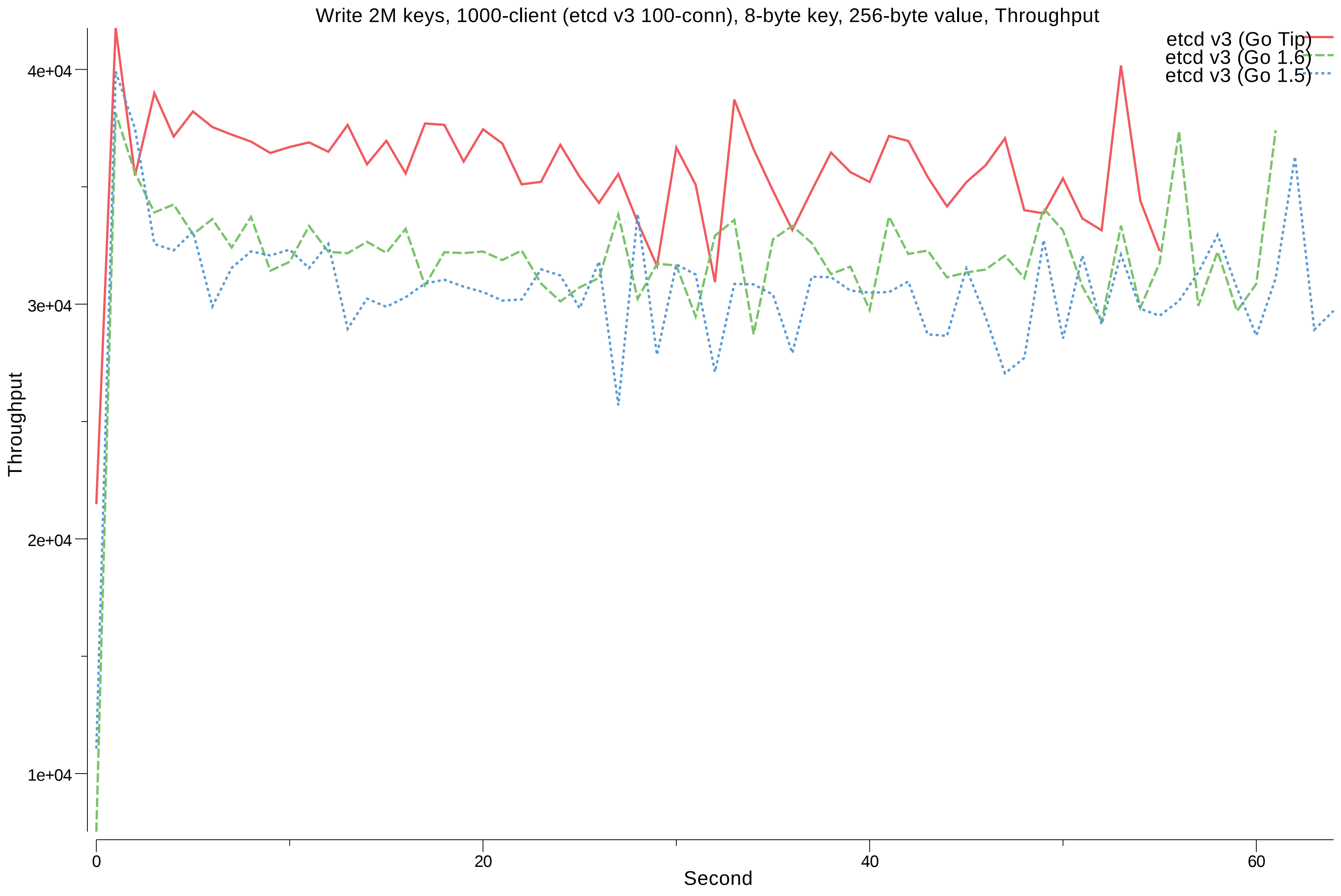
Intensive failure injection testing

- dash.etcd.io (http://dash.etcd.io/dashboard/db/functional-tests)
- >12,000 failure injections per day
- >1.5M injected for etcd v3

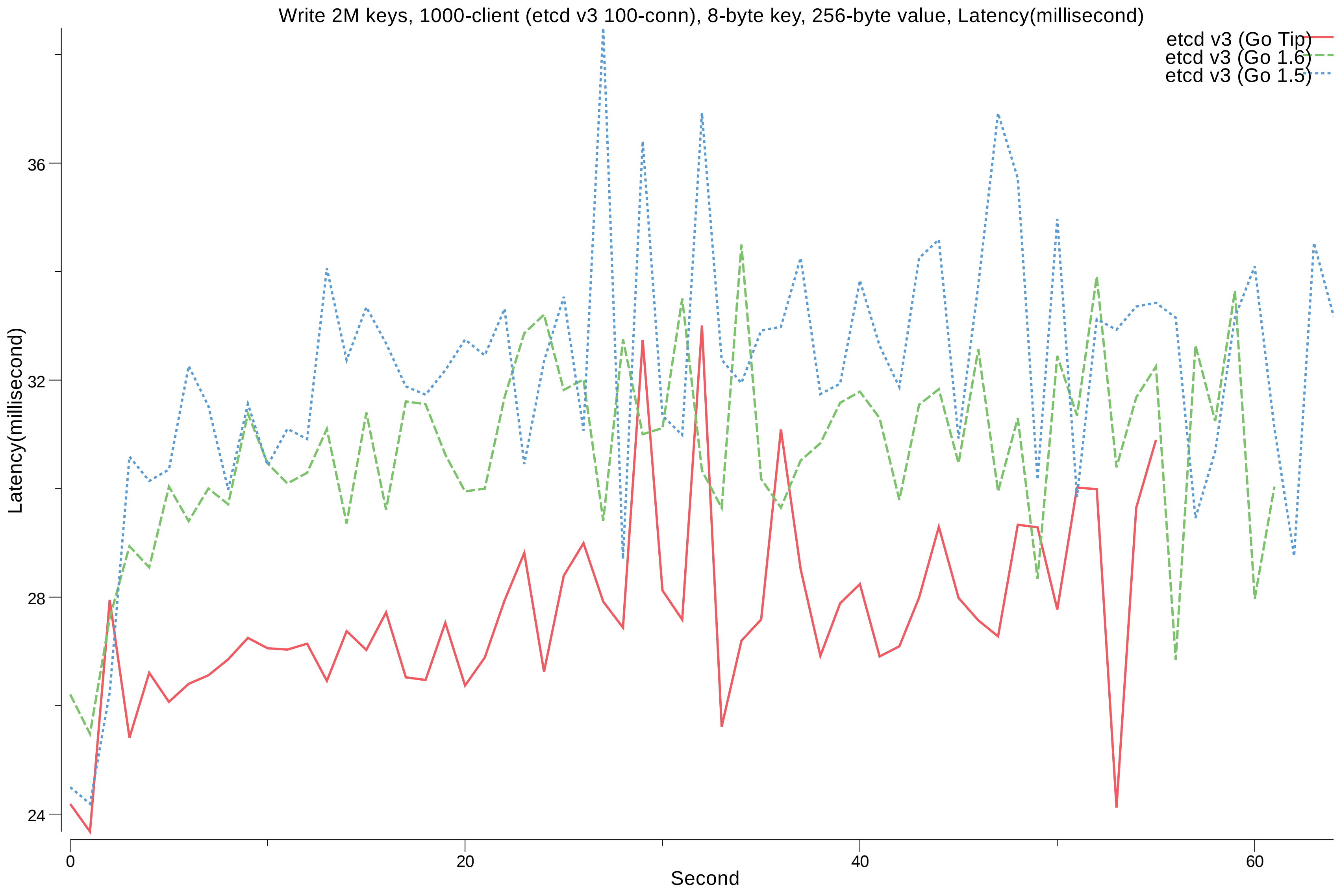
etcd Go 10 Tips

#1 Use latest Go

Throughput

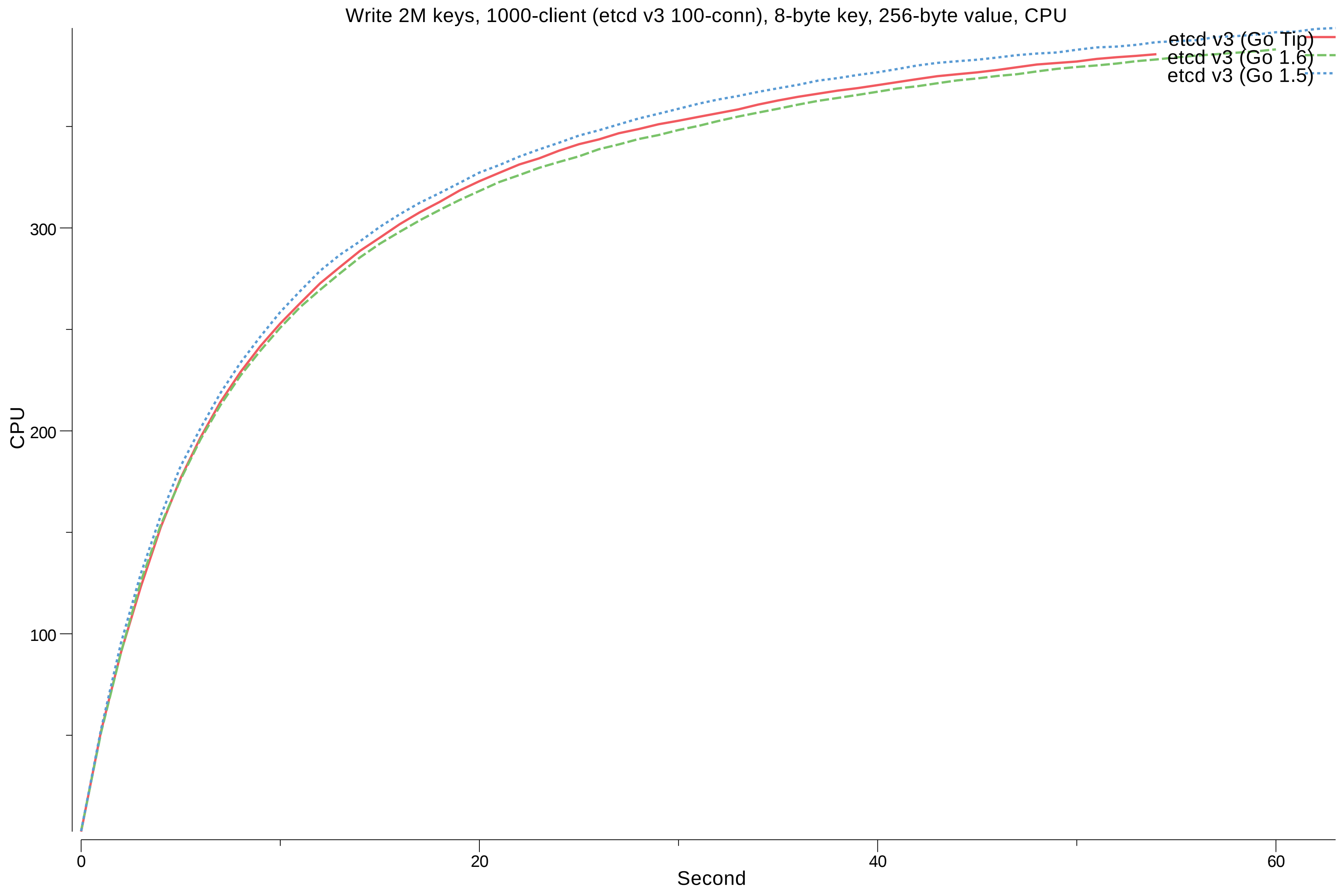

Write 2M keys, 1000-client (etcd v3 100-conn), 8-byte key, 256-byte value, Throughput
etcd v3 (Go Tip)
etcd v3 (Go 1.6)
etcd v3 (Go 1.5)
Throughput
4e+04
3e+04
2e+04
1e+04
0
20
40
60
Second

Latency

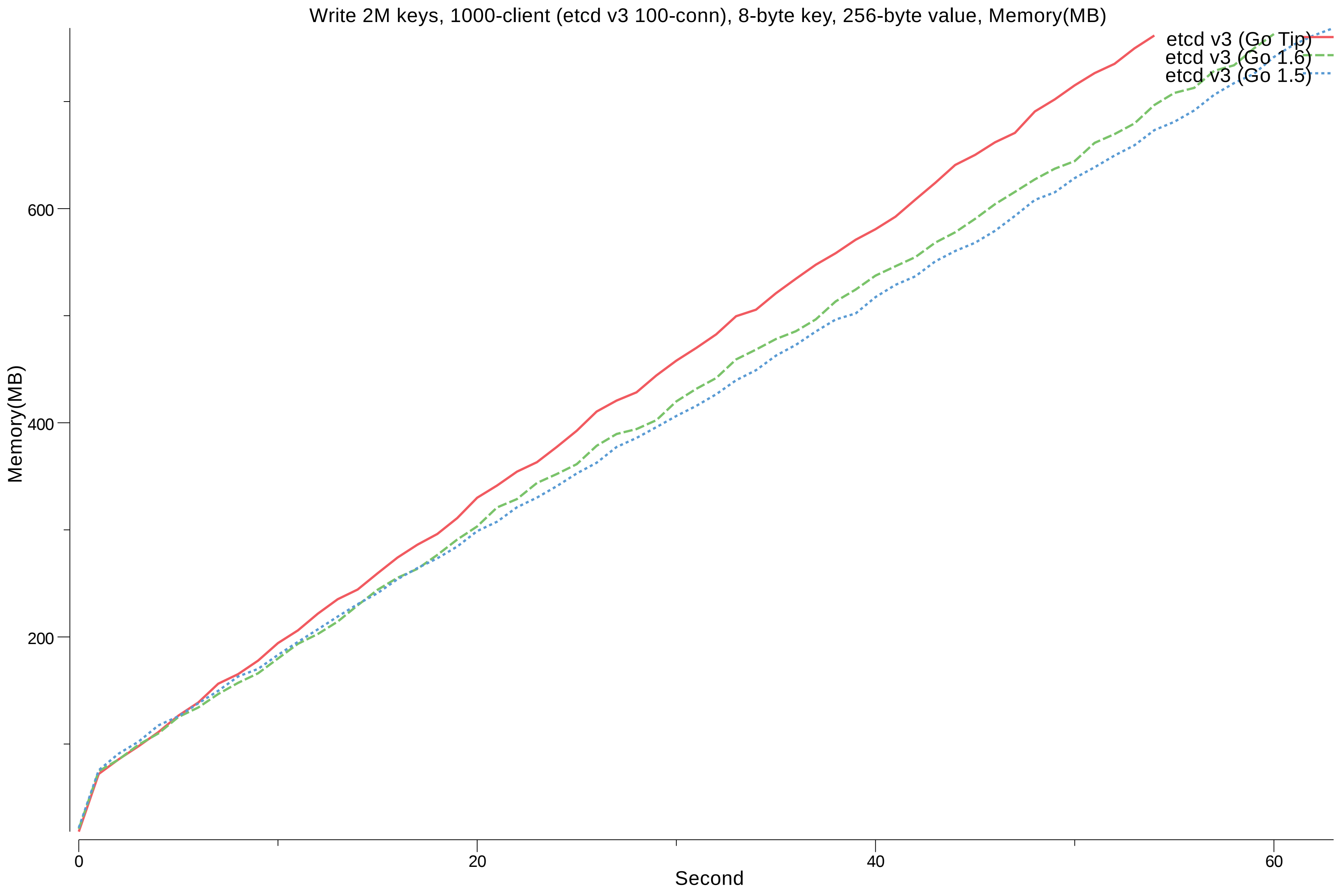


Write 2M keys, 1000-client (etcd v3 100-conn), 8-byte key, 256-byte value, Latency(millisecond)
etcd v3 (Go Tip)
etcd v3 (Go 1.6)
etcd v3 (Go 1.5)
Latency(millisecond)
36
32
28
24
0
20
40
60
Second

CPU


Write 2M keys, 1000-client (etcd v3 100-conn), 8-byte key, 256-byte value, CPU
etcd v3 (Go Tip)
etcd v3 (Go 1.6)
etcd v3 (Go 1.5)
CPU
100
200
300
Second
0
20
40
60

Memory



Write 2M keys, 1000-client (etcd v3 100-conn), 8-byte key, 256-byte value, Memory(MB)
etcd v3 (Go Tip)
etcd v3 (Go 1.6)
etcd v3 (Go 1.5)
Memory(MB)
600
400
200
0
20
40
60
Second

#2 Check slice allocation

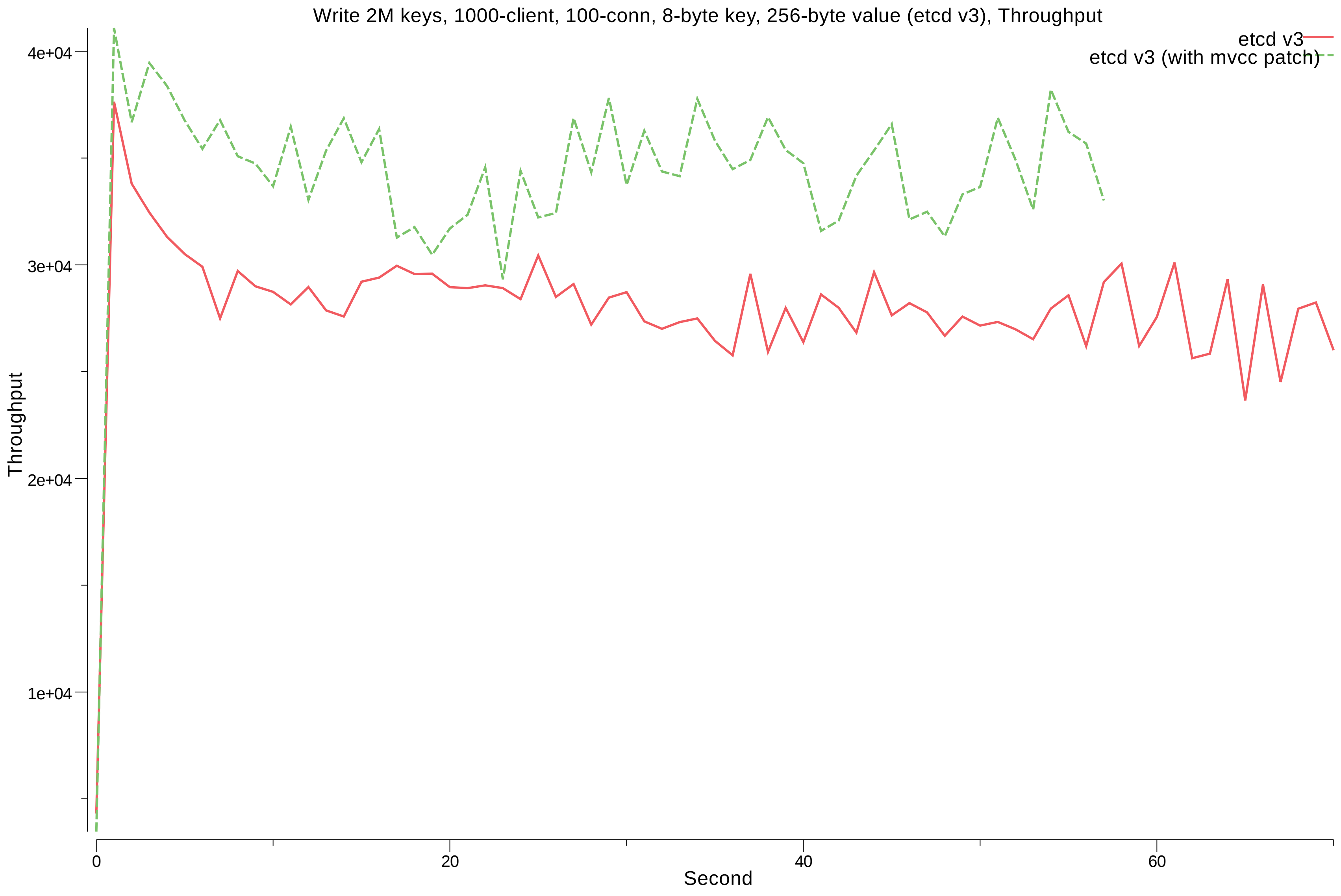
GH5238 (https://github.com/coreos/etcd/pull/5238)

```
s.changes = make([]mvccpb.KeyValue, 0, 128)
s.changes = make([]mvccpb.KeyValue, 0, 4)    // better for etcd use case
```

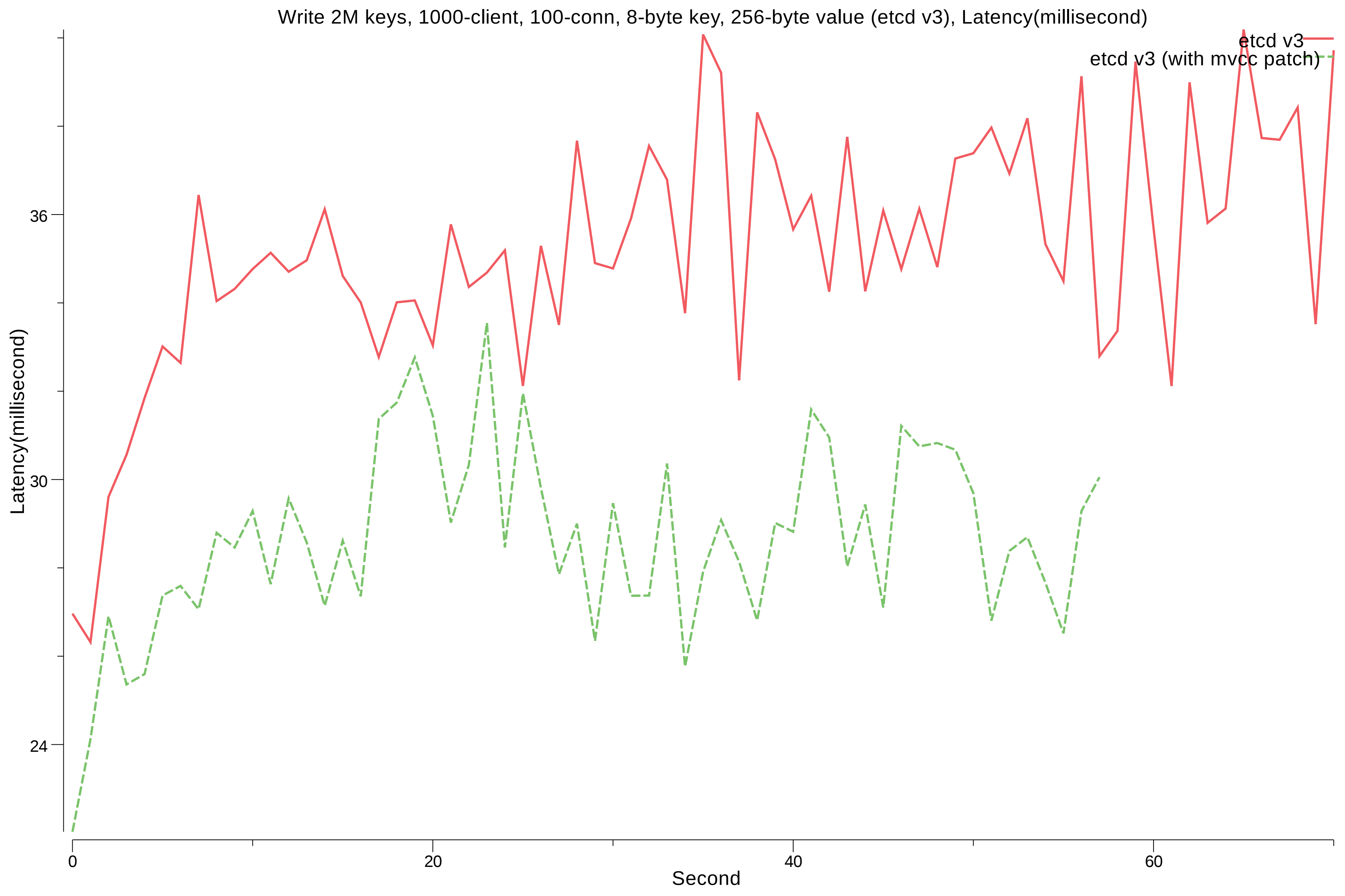
More capacity means more allocation.

Check your use case when making slice!

Throughput


Write 2M keys, 1000-client, 100-conn, 8-byte key, 256-byte value (etcd v3), Throughput
etcd v3
etcd v3 (with mvcc patch)
Throughput
4e+04
3e+04
2e+04
1e+04
0
20
40
60
Second

Latency


Write 2M keys, 1000-client, 100-conn, 8-byte key, 256-byte value (etcd v3), Latency(millisecond)
etcd v3
etcd v3 (with mvcc patch)
Latency(millisecond)
36
30
24
0
20
40
60
Second

#3 Test your code

All with "go test" command

- unit tests
- integration tests
- functional tests
- benchmarks

Use expect package for e2e tests

```
proc, _ = expect.NewExpect("etcdctl", "get", "foo")
_, err = proc.Expect("bar")  // if err != nil, found a bug!
```

github.com/coreos/etcd/pkg/expect (https://godoc.org/github.com/coreos/etcd/pkg/expect)

#4 Check goroutine leaks

Scan runtime.Stack

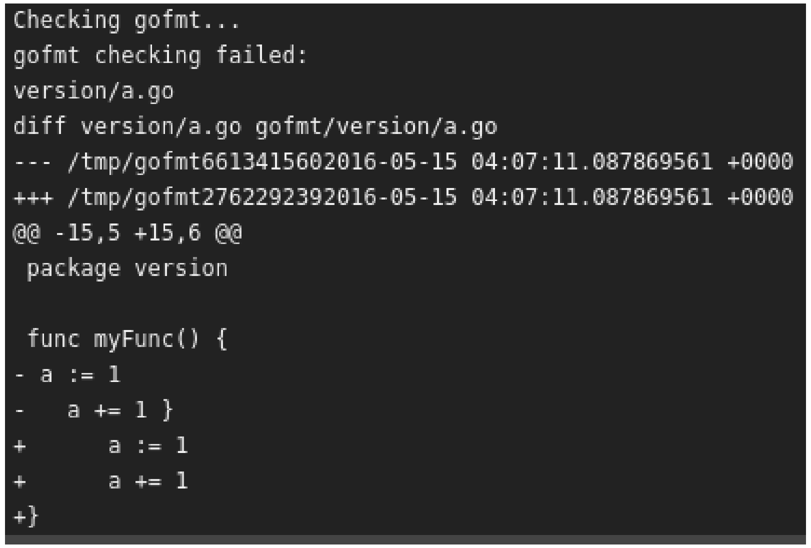
```
func TestMain(m *testing.M) {
    v := m.Run()
    if v == 0 && testutil.CheckLeakedGoroutine() {
        os.Exit(1)
    }
    os.Exit(v)
}

func TestSample(t *testing.T) {
    defer testutil.AfterTest(t)
    ...
}
```

- net/http/main_test.go (https://github.com/golang/go/blob/master/src/net/http/main_test.go)
- github.com/coreos/etcd/pkg/testutil (https://godoc.org/github.com/coreos/etcd/pkg/testutil)

Highly recommend for projects with context.Context, gRPC

#5 Always gofmt, go vet

gofmt



```
Checking gofmt...
gofmt checking failed:
version/a.go
diff version/a.go gofmt/version/a.go
--- /tmp/gofmt6613415602016-05-15 04:07:11.087869561 +0000
+++ /tmp/gofmt2762292392016-05-15 04:07:11.087869561 +0000
@@ -15,5 +15,6 @@
 package version

 func myFunc() {
- a := 1
-   a += 1 }
+      a := 1
+      a += 1
+}
```

go vet

```
log.Fatalf("hello %d", "a")
// arg "a" for printf verb %d of wrong type: string
```

#6 Write simple Go

```
ok := true
if ok == true {}  // X
if ok {}          // O
```

Don't:

```
err := l.newStream()
if err != nil {
    return err
}
return nil
```

Do:

```
return l.newStream()
```

github.com/dominikh/go-simple (https://github.com/dominikh/go-simple) by Dominik

#7 Check unused

- github.com/dominikh/go-unused (https://github.com/dominikh/go-unused) by Dominik
- Finds unused constants, variables, functions and types

Found bugs in etcd GH4955 (https://github.com/coreos/etcd/pull/4995/files)

#8 Use goword

- github.com/chzchzchz/goword (https://github.com/chzchzchz/goword) by Anthony (etcd team)

Comment checker

```
// This.
func Hello() {}  // This. (godoc-export: This -> Hello?)"
```

Spell checker

```
// Hello retuens.
func Hello() {}  // Hello retuens. (spell: retuens -> returns?)
```

#9 Document with godoc

- etcd must be easy to use
- etcd needs good documentation
- Example? etcd/clientv3 (https://godoc.org/github.com/coreos/etcd/clientv3#example-KV--Get)

#10 vendor

Problem

- etcd client package is used within etcd repo (etcdctl)
- etcd client imports gRPC and vendors it
- Project B import this etcd client package
- Project B also uses gRPC but from different import path

Now two projects has conflicting gRPC code GH566 (https://github.com/grpc/grpc-go/issues/566)

```
panic: http: multiple registrations for /debug/requests
```

#10 vendor

Solution GH4950 (https://github.com/coreos/etcd/pull/4950)

- Create symlinks inside cmd directory
- Keep vendor inside cmd directory

Now

- Still go-get-able
- No conflicts with other projects

It works, even on Windows.

Thank you

Gyu-Ho Lee
CoreOS
gyu_ho.lee@coreos.com (mailto:gyu_ho.lee@coreos.com)
https://github.com/gyuho/presentations (https://github.com/gyuho/presentations)