

This repository

Search

Pull requests

Issues

Marketplace

Gist

coreos / etcd

Watch857

Star13,839

Fork2,653

<> Code

Issues131

Pull requests37

Insights

embed: Client cannot connect on restart of embedded server. #6733

New issue

Closed

prashanthpai opened this issue on Oct 26, 2016 · 25 comments



prashanthpai commented on Oct 26, 2016

+👤

I'm not quite sure if this is an issue with embed or the etcd client. Or I could be doing something I'm not supposed to.

Here's the reproducer workflow.

1. Start embedded server in standalone mode.
2. Do some client request on client endpoint. This succeeds.
3. Restart embedded server to join an existing cluster.
4. Do some client request on client endpoint. This fails.

Failure:

```
2016-10-26 11:23:20.737621 E | etcdserver/api/v2http: got unexpected response error (etcdse
2016-10-26 11:23:20.738186 I | client: etcd cluster is unavailable or misconfigured; error .
```

Setup:

```
$ etcd --version
etcd Version: 3.1.0-rc.0
Git SHA: 8334790
Go Version: go1.7.1
Go OS/Arch: linux/amd64
```

I have the following straight-forward reproducer:

```
package main

import (
    "errors"
    "log"
    "net/url"
    "os"
    "time"

    etcdclient "github.com/coreos/etcd/client"
    "github.com/coreos/etcd/embed"
    "golang.org/x/net/context"
)

const (
    myIP string = "192.168.56.26"
)

func getCfg() *embed.Config {

    cfg := embed.NewConfig()
    cfg.Name = myIP
    cfg.Dir = cfg.Name + ".etcd"
    listenClientURL, _ := url.Parse("http://" + myIP + ":2379")
    cfg.ACUrls = []url.URL{*listenClientURL}
    cfg.LCUrls = []url.URL{*listenClientURL}
    listenPeerURL, _ := url.Parse("http://" + myIP + ":2380")
    cfg.APUrls = []url.URL{*listenPeerURL}
    cfg.LPUrls = []url.URL{*listenPeerURL}

    cfg.ClusterState = embed.ClusterStateFlagNew
    cfg.InitialCluster = cfg.Name + "=" + listenPeerURL.String()
}
```

Assignees

gyuho

Labels

component/client

Projects

None yet

Milestone

v3.1.0

Notifications

Subscribe

You're not receiving notifications from this thread.

3 participants

```

    return cfg
}

func startServer(cfg *embed.Config) (*embed.Etcd, error) {
    etcd, err := embed.StartEtcd(cfg)
    if err != nil {
        return nil, err
    }

    select {
    case <-etcd.Server.ReadyNotify():
        log.Print("Etcd embedded server is ready.")
        return etcd, nil
    case <-time.After(42 * time.Second):
        return nil, errors.New("Etcd embedded server took too long to start!")
    case err := <-etcd.Err():
        return nil, err
    }
}

func invokeClientOp() error {
    ecfg := &etcdclient.Config{
        Endpoints:      []string{"http://" + myIP + ":2379"},
        Transport:      etcdclient.DefaultTransport,
        HeaderTimeoutPerRequest: 3 * time.Second,
    }
    c, err := etcdclient.New(*ecfg)
    if err != nil {
        log.Fatal(err)
    }
    eclient := etcdclient.NewKeysAPI(c)
    getOpts := &etcdclient.GetOptions{
        Quorum:    true,
        Recursive: true,
        Sort:      true,
    }
    _, err = eclient.Get(context.Background(), "blah/", getOpts)
    return err
}

func main() {
    // Start embedded etcd server
    cfg := getCfg()
    e1, err := startServer(cfg)
    if err != nil {
        log.Fatal(err)
    }

    // Do some client get op. This will fail with key not found which is expected.
    err = invokeClientOp()
    if err != nil {
        log.Print(err)
    }

    // Stop etcd server
    e1.Close()
    os.RemoveAll(cfg.Dir)
    log.Print("Etcd server stopped")

    // While this program is sleeping...
    // From another node (192.168.56.25), add this node as a member using CLI
    // etcd --listen-peer-urls http://192.168.56.25:2380 --listen-client-urls http://19
    // --advertise-client-urls http://192.168.56.25:2379 --initial-advertise-peer-
    // --initial-cluster default=http://192.168.56.25:2380
    // etcdctl --endpoint 192.168.56.25:2379 member add 192.168.56.26 http://192.168.56
    time.Sleep(30 * time.Second)

    // Start etcd server again, this time join existing cluster.
    cfg.InitialCluster = "192.168.56.26=http://192.168.56.26:2380,default=http://192.16
    cfg.ClusterState = embed.ClusterStateFlagExisting
    e2, err := startServer(cfg)
    if err != nil {
        log.Fatal(err)
    }

    // At this point, some client connections still persist...
    // watch "sudo lsof -Pan -p <pid> -i"

    err = invokeClientOp() // <--- This fails after embed server restart.
    if err != nil {
        log.Print(err)
    }

    e2.Close()
    os.RemoveAll(cfg.Dir)
}

```

```
// At this point, some client connections still persist...
// watch "sudo lsof -Pan -p <pid> -i"

time.Sleep(1000 * time.Second)
}
```

Run log:

```
[ppai@gd2-2 ~]$ ./tryembed
2016-10-26 11:22:49.194583 I | embed: listening for peers on http://192.168.56.26:2380
2016-10-26 11:22:49.195860 I | embed: listening for client requests on 192.168.56.26:2379
2016-10-26 11:22:49.202978 I | etcdserver: name = 192.168.56.26
2016-10-26 11:22:49.203604 I | etcdserver: data dir = 192.168.56.26.etcd
2016-10-26 11:22:49.203944 I | etcdserver: member dir = 192.168.56.26.etcd/member
2016-10-26 11:22:49.204267 I | etcdserver: heartbeat = 100ms
2016-10-26 11:22:49.204554 I | etcdserver: election = 1000ms
2016-10-26 11:22:49.204868 I | etcdserver: snapshot count = 10000
2016-10-26 11:22:49.205176 I | etcdserver: advertise client URLs = http://192.168.56.26:237
2016-10-26 11:22:49.205492 I | etcdserver: initial advertise peer URLs = http://192.168.56.
2016-10-26 11:22:49.205722 I | etcdserver: initial cluster = 192.168.56.26=http://192.168.5
2016-10-26 11:22:49.208369 I | etcdserver: starting member 9fbd239dbdea6f36 in cluster 7d1c
2016-10-26 11:22:49.208757 I | raft: 9fbd239dbdea6f36 became follower at term 0
2016-10-26 11:22:49.208984 I | raft: newRaft 9fbd239dbdea6f36 [peers: [], term: 0, commit:
2016-10-26 11:22:49.209300 I | raft: 9fbd239dbdea6f36 became follower at term 1
2016-10-26 11:22:49.214992 I | etcdserver: starting server... [version: 3.1.0-rc.0+git, clu
2016-10-26 11:22:49.216755 I | etcdserver/membership: added member 9fbd239dbdea6f36 [http:/
2016-10-26 11:22:49.911429 I | raft: 9fbd239dbdea6f36 is starting a new election at term 1
2016-10-26 11:22:49.913520 I | raft: 9fbd239dbdea6f36 became candidate at term 2
2016-10-26 11:22:49.915002 I | raft: 9fbd239dbdea6f36 received vote from 9fbd239dbdea6f36 a
2016-10-26 11:22:49.915680 I | raft: 9fbd239dbdea6f36 became leader at term 2
2016-10-26 11:22:49.916098 I | raft: raft.node: 9fbd239dbdea6f36 elected leader 9fbd239dbde
2016-10-26 11:22:49.917283 I | etcdserver: published {Name:192.168.56.26 ClientURLs:[http:/
2016-10-26 11:22:49.917813 I | embed: ready to serve client requests
2016-10-26 11:22:49.918386 N | embed: serving insecure client requests on 192.168.56.26:237
2016-10-26 11:22:49.918755 I | etcdserver: setting up the initial cluster version to 3.1
2016-10-26 11:22:49.919790 I | Etcd embedded server is ready.
2016-10-26 11:22:49.922500 N | etcdserver/membership: set the initial cluster version to 3.
2016-10-26 11:22:49.923071 I | etcdserver/api: enabled capabilities for version 3.1
2016-10-26 11:22:49.923688 I | 100: Key not found (/blah) [3]
2016-10-26 11:22:49.924014 I | etcdserver: skipped leadership transfer for single member cl
2016-10-26 11:22:49.927066 I | Etcd server stopped
```

<< added this node as a member here, from another node >>

```
2016-10-26 11:23:19.929231 I | embed: listening for peers on http://192.168.56.26:2380
2016-10-26 11:23:19.931325 I | embed: listening for client requests on 192.168.56.26:2379
2016-10-26 11:23:19.947651 I | etcdserver: name = 192.168.56.26
2016-10-26 11:23:19.949391 I | etcdserver: data dir = 192.168.56.26.etcd
2016-10-26 11:23:19.950760 I | etcdserver: member dir = 192.168.56.26.etcd/member
2016-10-26 11:23:19.952069 I | etcdserver: heartbeat = 100ms
2016-10-26 11:23:19.952864 I | etcdserver: election = 1000ms
2016-10-26 11:23:19.952916 I | etcdserver: snapshot count = 10000
2016-10-26 11:23:19.952972 I | etcdserver: advertise client URLs = http://192.168.56.26:237
2016-10-26 11:23:19.975016 I | etcdserver: starting member a781f7cffc56c33d in cluster ebeb
2016-10-26 11:23:19.975918 I | raft: a781f7cffc56c33d became follower at term 0
2016-10-26 11:23:19.976345 I | raft: newRaft a781f7cffc56c33d [peers: [], term: 0, commit:
2016-10-26 11:23:19.976811 I | raft: a781f7cffc56c33d became follower at term 1
2016-10-26 11:23:19.983299 I | rafthttp: started HTTP pipelining with peer b3e1648d87d338a3
2016-10-26 11:23:19.983740 I | rafthttp: starting peer b3e1648d87d338a3...
2016-10-26 11:23:19.984040 I | rafthttp: started HTTP pipelining with peer b3e1648d87d338a3
2016-10-26 11:23:19.992463 I | rafthttp: started peer b3e1648d87d338a3
2016-10-26 11:23:19.993282 I | rafthttp: added peer b3e1648d87d338a3
2016-10-26 11:23:19.993532 I | etcdserver: starting server... [version: 3.1.0-rc.0+git, clu
2016-10-26 11:23:19.993960 I | rafthttp: started streaming with peer b3e1648d87d338a3 (writ
2016-10-26 11:23:19.995267 I | rafthttp: started streaming with peer b3e1648d87d338a3 (writ
2016-10-26 11:23:19.996670 I | rafthttp: started streaming with peer b3e1648d87d338a3 (stre
2016-10-26 11:23:19.998071 I | rafthttp: started streaming with peer b3e1648d87d338a3 (stre
2016-10-26 11:23:19.998537 I | rafthttp: peer b3e1648d87d338a3 became active
2016-10-26 11:23:19.999146 I | rafthttp: established a TCP streaming connection with peer b
2016-10-26 11:23:19.999404 I | rafthttp: established a TCP streaming connection with peer b
2016-10-26 11:23:19.999637 I | rafthttp: established a TCP streaming connection with peer b
2016-10-26 11:23:20.004842 I | rafthttp: established a TCP streaming connection with peer b
2016-10-26 11:23:20.696763 I | raft: a781f7cffc56c33d [term: 1] received a MsgVote message
2016-10-26 11:23:20.698083 I | raft: a781f7cffc56c33d became follower at term 19
2016-10-26 11:23:20.699525 I | raft: a781f7cffc56c33d [logterm: 0, index: 0, vote: 0] voted
2016-10-26 11:23:20.709926 I | raft: raft.node: a781f7cffc56c33d elected leader b3e1648d87d
2016-10-26 11:23:20.720827 I | etcdserver/membership: added member b3e1648d87d338a3 [http:/
2016-10-26 11:23:20.723911 N | etcdserver/membership: set the initial cluster version to 3.
2016-10-26 11:23:20.726797 I | etcdserver/membership: added member a781f7cffc56c33d [http:/
2016-10-26 11:23:20.731452 I | etcdserver: published {Name:192.168.56.26 ClientURLs:[http:/
2016-10-26 11:23:20.732256 I | embed: ready to serve client requests
2016-10-26 11:23:20.733617 N | embed: serving insecure client requests on 192.168.56.26:237
```

```

2016-10-26 11:23:20.736240 I | Etcd embedded server is ready.
2016-10-26 11:23:20.737621 E | etcdserver/api/v2http: got unexpected response error (etcdse
2016-10-26 11:23:20.738186 I | client: etcd cluster is unavailable or misconfigured; error
2016-10-26 11:23:20.738573 I | etcdserver: skipped leadership transfer for stopping non-lea
2016-10-26 11:23:20.739292 I | rafthttp: stopped HTTP pipelining with peer b3e1648d87d338a3
2016-10-26 11:23:20.739681 I | rafthttp: stopping peer b3e1648d87d338a3...
2016-10-26 11:23:20.740195 I | rafthttp: closed the TCP streaming connection with peer b3e1
2016-10-26 11:23:20.740683 I | rafthttp: stopped streaming with peer b3e1648d87d338a3 (writ
2016-10-26 11:23:20.741261 I | rafthttp: closed the TCP streaming connection with peer b3e1
2016-10-26 11:23:20.742010 I | rafthttp: stopped streaming with peer b3e1648d87d338a3 (writ
2016-10-26 11:23:20.742807 I | rafthttp: stopped HTTP pipelining with peer b3e1648d87d338a3
2016-10-26 11:23:20.743796 W | rafthttp: lost the TCP streaming connection with peer b3e164
2016-10-26 11:23:20.744797 I | rafthttp: stopped streaming with peer b3e1648d87d338a3 (stre
2016-10-26 11:23:20.745224 W | rafthttp: lost the TCP streaming connection with peer b3e164
2016-10-26 11:23:20.745637 I | rafthttp: stopped streaming with peer b3e1648d87d338a3 (stre
2016-10-26 11:23:20.746033 I | rafthttp: stopped peer b3e1648d87d338a3

```

Connections don't seem to cleanup

```

$ sudo lsof -Pan -p 6003 -i
COMMAND  PID USER  FD   TYPE DEVICE SIZE/OFF NODE NAME
tryembed 6003 ppai   23u   IPv4 69520      0t0  TCP 192.168.56.26:57400->192.168.56.26:2379
tryembed 6003 ppai   24u   IPv4 69521      0t0  TCP 192.168.56.26:2379->192.168.56.26:57400

```

 prashanthpai referenced this issue in [gluster/glusterd2](#) on Oct 26, 2016

Embed etcd #148

Merged



prashanthpai commented on Oct 26, 2016

+ 

This issue is likely to be due to the active TCP connection:

```

$ lsof -np 7051
COMMAND  PID USER  FD   TYPE    DEVICE  SIZE/OFF      NODE NAME
tryembed 7051 ppai   cwd    DIR      253,0    4096 8923701 /home/ppai
tryembed 7051 ppai   rtd    DIR      253,0      252     96 /
tryembed 7051 ppai   txt    REG      253,0 28062088 8928091 /home/ppai/tryembed
tryembed 7051 ppai   mem    REG      253,0 2089496 8498154 /usr/lib64/libc-2.23.
tryembed 7051 ppai   mem    REG      253,0 142312 8498182 /usr/lib64/libpthread
tryembed 7051 ppai   mem    REG      253,0 172080 8466806 /usr/lib64/ld-2.23.so
tryembed 7051 ppai    0u    CHR     136,1      0t0      4 /dev/pts/1
tryembed 7051 ppai    1u    CHR     136,1      0t0      4 /dev/pts/1
tryembed 7051 ppai    2u    CHR     136,1      0t0      4 /dev/pts/1
tryembed 7051 ppai    3u   unix 0xffff880035cb4400      0t0  76113 type=DGRAM
tryembed 7051 ppai    4u  a_inode          0,11      0  7036 [eventpoll]
tryembed 7051 ppai   23u   IPv4     76547      0t0      TCP 192.168.56.26:57830->
tryembed 7051 ppai   30u   IPv4     76548      0t0      TCP 192.168.56.26:etcd-cl

```

I couldn't find anything in the client API to close connections. Tried again by using `Client.Sync()` and setting `KeepAlive` to 0. It's still the same.



gyuhho commented on Oct 26, 2016

Member

+ 

@prashanthpai Do you have logs in the new server that you are adding?

And please note that if you add 1 server to single-node cluster, the cluster loses its quorum and triggers leader election. I recommend to start with at least 3-node.



prashanthpai commented on Oct 26, 2016

+ 

@gyuhho It's emedded etcd server, so I guess the logs are the stdout output you see above.

The 3 node quorum comment is valid but not so relevant here. This issue is reproducible only when `etcdclient` API is used programmatically. I don't see any issue when I use `etcdctl`.

Further, in my two node reproducer cluster, the health was good and the first node was leader.



prashanthpai commented on Oct 26, 2016

+ 

I'll see if I can find the log of the server from which the member was added.



prashanthpai commented on Oct 26, 2016



I put a sleep before shutting down embedded etcd server the second time in example code above.

```
err = invokeClientOp() // <--- This fails after embed server restart.
if err != nil {
    log.Print(err)
}

time.Sleep(30 * time.Second)

e2.Close()
os.RemoveAll(cfg.Dir)
```

Node 1 (192.168.56.25) logs:

```
[ppai@gd2-1 ~]$ etcd --listen-peer-urls http://192.168.56.25:2380 --listen-client-urls http
2016-10-26 20:02:34.701621 I | etcdmain: etcd Version: 3.1.0-rc.0
2016-10-26 20:02:34.703472 I | etcdmain: Git SHA: 8334790
2016-10-26 20:02:34.703848 I | etcdmain: Go Version: go1.7.1
2016-10-26 20:02:34.704112 I | etcdmain: Go OS/Arch: linux/amd64
2016-10-26 20:02:34.704400 I | etcdmain: setting maximum number of CPUs to 1, total number
2016-10-26 20:02:34.704686 W | etcdmain: no data-dir provided, using default data-dir ./def
2016-10-26 20:02:34.705029 I | embed: listening for peers on http://192.168.56.25:2380
2016-10-26 20:02:34.705404 I | embed: listening for client requests on 192.168.56.25:2379
2016-10-26 20:02:34.709825 I | etcdserver: name = default
2016-10-26 20:02:34.710383 I | etcdserver: data dir = default.etcd
2016-10-26 20:02:34.710908 I | etcdserver: member dir = default.etcd/member
2016-10-26 20:02:34.711586 I | etcdserver: heartbeat = 100ms
2016-10-26 20:02:34.711898 I | etcdserver: election = 1000ms
2016-10-26 20:02:34.712153 I | etcdserver: snapshot count = 10000
2016-10-26 20:02:34.712418 I | etcdserver: advertise client URLs = http://192.168.56.25:237
2016-10-26 20:02:34.712632 I | etcdserver: initial advertise peer URLs = http://192.168.56.
2016-10-26 20:02:34.712898 I | etcdserver: initial cluster = default=http://192.168.56.25:2
2016-10-26 20:02:34.720584 I | etcdserver: starting member b3e1648d87d338a3 in cluster ebeb
2016-10-26 20:02:34.721844 I | raft: b3e1648d87d338a3 became follower at term 0
2016-10-26 20:02:34.722110 I | raft: newRaft b3e1648d87d338a3 [peers: [], term: 0, commit:
2016-10-26 20:02:34.722507 I | raft: b3e1648d87d338a3 became follower at term 1
2016-10-26 20:02:34.727882 I | etcdserver: starting server... [version: 3.1.0-rc.0, cluster
2016-10-26 20:02:34.730488 I | etcdserver/membership: added member b3e1648d87d338a3 [http:/
2016-10-26 20:02:34.823449 I | raft: b3e1648d87d338a3 is starting a new election at term 1
2016-10-26 20:02:34.823902 I | raft: b3e1648d87d338a3 became candidate at term 2
2016-10-26 20:02:34.824141 I | raft: b3e1648d87d338a3 received vote from b3e1648d87d338a3 a
2016-10-26 20:02:34.824391 I | raft: b3e1648d87d338a3 became leader at term 2
2016-10-26 20:02:34.824590 I | raft: raft.node: b3e1648d87d338a3 elected leader b3e1648d87d
2016-10-26 20:02:34.825363 I | etcdserver: setting up the initial cluster version to 3.1
2016-10-26 20:02:34.826130 I | etcdserver: published {Name:default ClientURLs:[http://192.1
2016-10-26 20:02:34.826479 E | etcdmain: forgot to set Type=notify in systemd service file?
2016-10-26 20:02:34.826674 I | embed: ready to serve client requests
2016-10-26 20:02:34.827134 N | embed: serving insecure client requests on 192.168.56.25:237
2016-10-26 20:02:34.828469 N | etcdserver/membership: set the initial cluster version to 3.
2016-10-26 20:02:34.829704 I | etcdserver/api: enabled capabilities for version 3.1
```

<< this is where I added second node as member >>

```
[ppai@gd2-1 ~]$ etcdctl --endpoint 192.168.56.25:2379 member add 192.168.56.26 http://192.1
Added member named 192.168.56.26 with ID 87575c769a05665a to cluster
```

```
ETCD_NAME="192.168.56.26"
ETCD_INITIAL_CLUSTER="192.168.56.26=http://192.168.56.26:2380,default=http://192.168.56.25:
ETCD_INITIAL_CLUSTER_STATE="existing"
```

```
2016-10-26 20:02:48.280494 I | etcdserver/membership: added member 87575c769a05665a [http:/
2016-10-26 20:02:48.282261 I | rafthttp: starting peer 87575c769a05665a...
2016-10-26 20:02:48.283612 I | rafthttp: started HTTP pipelining with peer 87575c769a05665a
2016-10-26 20:02:48.288712 I | rafthttp: started streaming with peer 87575c769a05665a (writ
2016-10-26 20:02:48.290598 I | rafthttp: started peer 87575c769a05665a
2016-10-26 20:02:48.290976 I | rafthttp: added peer 87575c769a05665a
2016-10-26 20:02:48.293302 I | rafthttp: started streaming with peer 87575c769a05665a (writ
2016-10-26 20:02:48.293914 I | rafthttp: started streaming with peer 87575c769a05665a (stre
2016-10-26 20:02:48.295068 I | rafthttp: started streaming with peer 87575c769a05665a (stre
2016-10-26 20:02:48.824095 W | raft: b3e1648d87d338a3 stepped down to follower since quorum
```

```

2016-10-26 20:02:48.824572 I | raft: b3e1648d87d338a3 became follower at term 2
2016-10-26 20:02:48.824857 I | raft: raft.node: b3e1648d87d338a3 lost leader b3e1648d87d338
2016-10-26 20:02:50.324820 I | raft: b3e1648d87d338a3 is starting a new election at term 2
2016-10-26 20:02:50.325187 I | raft: b3e1648d87d338a3 became candidate at term 3
2016-10-26 20:02:50.325377 I | raft: b3e1648d87d338a3 received vote from b3e1648d87d338a3 a
2016-10-26 20:02:50.325580 I | raft: b3e1648d87d338a3 [logterm: 2, index: 33] sent vote req
2016-10-26 20:02:52.124603 I | raft: b3e1648d87d338a3 is starting a new election at term 3
2016-10-26 20:02:52.125007 I | raft: b3e1648d87d338a3 became candidate at term 4
2016-10-26 20:02:52.125250 I | raft: b3e1648d87d338a3 received vote from b3e1648d87d338a3 a
2016-10-26 20:02:52.125480 I | raft: b3e1648d87d338a3 [logterm: 2, index: 33] sent vote req
2016-10-26 20:02:53.291457 W | rafthttp: health check for peer 87575c769a05665a could not c
2016-10-26 20:02:53.624056 I | raft: b3e1648d87d338a3 is starting a new election at term 4
2016-10-26 20:02:53.624762 I | raft: b3e1648d87d338a3 became candidate at term 5
2016-10-26 20:02:53.625054 I | raft: b3e1648d87d338a3 received vote from b3e1648d87d338a3 a
2016-10-26 20:02:53.625289 I | raft: b3e1648d87d338a3 [logterm: 2, index: 33] sent vote req
2016-10-26 20:02:55.523963 I | raft: b3e1648d87d338a3 is starting a new election at term 5
2016-10-26 20:02:55.524760 I | raft: b3e1648d87d338a3 became candidate at term 6
2016-10-26 20:02:55.525160 I | raft: b3e1648d87d338a3 received vote from b3e1648d87d338a3 a
2016-10-26 20:02:55.525791 I | raft: b3e1648d87d338a3 [logterm: 2, index: 33] sent vote req
2016-10-26 20:02:56.723686 I | raft: b3e1648d87d338a3 is starting a new election at term 6
2016-10-26 20:02:56.724630 I | raft: b3e1648d87d338a3 became candidate at term 7
2016-10-26 20:02:56.725118 I | raft: b3e1648d87d338a3 received vote from b3e1648d87d338a3 a
2016-10-26 20:02:56.725466 I | raft: b3e1648d87d338a3 [logterm: 2, index: 33] sent vote req
2016-10-26 20:02:58.023983 I | raft: b3e1648d87d338a3 is starting a new election at term 7
2016-10-26 20:02:58.024549 I | raft: b3e1648d87d338a3 became candidate at term 8
2016-10-26 20:02:58.024898 I | raft: b3e1648d87d338a3 received vote from b3e1648d87d338a3 a
2016-10-26 20:02:58.025231 I | raft: b3e1648d87d338a3 [logterm: 2, index: 33] sent vote req
2016-10-26 20:02:58.429087 W | rafthttp: health check for peer 87575c769a05665a could not c
2016-10-26 20:02:59.423812 I | raft: b3e1648d87d338a3 is starting a new election at term 8
2016-10-26 20:02:59.424180 I | raft: b3e1648d87d338a3 became candidate at term 9
2016-10-26 20:02:59.424397 I | raft: b3e1648d87d338a3 received vote from b3e1648d87d338a3 a
2016-10-26 20:02:59.424611 I | raft: b3e1648d87d338a3 [logterm: 2, index: 33] sent vote req
2016-10-26 20:03:01.023386 I | raft: b3e1648d87d338a3 is starting a new election at term 9
2016-10-26 20:03:01.023921 I | raft: b3e1648d87d338a3 became candidate at term 10
2016-10-26 20:03:01.024232 I | raft: b3e1648d87d338a3 received vote from b3e1648d87d338a3 a
2016-10-26 20:03:01.024556 I | raft: b3e1648d87d338a3 [logterm: 2, index: 33] sent vote req
2016-10-26 20:03:02.323768 I | raft: b3e1648d87d338a3 is starting a new election at term 10
2016-10-26 20:03:02.324756 I | raft: b3e1648d87d338a3 became candidate at term 11
2016-10-26 20:03:02.325546 I | raft: b3e1648d87d338a3 received vote from b3e1648d87d338a3 a
2016-10-26 20:03:02.326385 I | raft: b3e1648d87d338a3 [logterm: 2, index: 33] sent vote req
2016-10-26 20:03:03.292882 W | rafthttp: health check for peer 87575c769a05665a could not c
2016-10-26 20:03:03.423489 I | raft: b3e1648d87d338a3 is starting a new election at term 11
2016-10-26 20:03:03.424236 I | raft: b3e1648d87d338a3 became candidate at term 12
2016-10-26 20:03:03.424981 I | raft: b3e1648d87d338a3 received vote from b3e1648d87d338a3 a
2016-10-26 20:03:03.425485 I | raft: b3e1648d87d338a3 [logterm: 2, index: 33] sent vote req
2016-10-26 20:03:04.723337 I | raft: b3e1648d87d338a3 is starting a new election at term 12
2016-10-26 20:03:04.723801 I | raft: b3e1648d87d338a3 became candidate at term 13
2016-10-26 20:03:04.724070 I | raft: b3e1648d87d338a3 received vote from b3e1648d87d338a3 a
2016-10-26 20:03:04.724283 I | raft: b3e1648d87d338a3 [logterm: 2, index: 33] sent vote req
2016-10-26 20:03:06.023278 I | raft: b3e1648d87d338a3 is starting a new election at term 13
2016-10-26 20:03:06.024478 I | raft: b3e1648d87d338a3 became candidate at term 14
2016-10-26 20:03:06.025231 I | raft: b3e1648d87d338a3 received vote from b3e1648d87d338a3 a
2016-10-26 20:03:06.025507 I | raft: b3e1648d87d338a3 [logterm: 2, index: 33] sent vote req
2016-10-26 20:03:07.123391 I | raft: b3e1648d87d338a3 is starting a new election at term 14
2016-10-26 20:03:07.124105 I | raft: b3e1648d87d338a3 became candidate at term 15
2016-10-26 20:03:07.124446 I | raft: b3e1648d87d338a3 received vote from b3e1648d87d338a3 a
2016-10-26 20:03:07.124779 I | raft: b3e1648d87d338a3 [logterm: 2, index: 33] sent vote req
2016-10-26 20:03:08.293436 W | rafthttp: health check for peer 87575c769a05665a could not c
2016-10-26 20:03:08.723197 I | raft: b3e1648d87d338a3 is starting a new election at term 15
2016-10-26 20:03:08.723675 I | raft: b3e1648d87d338a3 became candidate at term 16
2016-10-26 20:03:08.723933 I | raft: b3e1648d87d338a3 received vote from b3e1648d87d338a3 a
2016-10-26 20:03:08.724117 I | raft: b3e1648d87d338a3 [logterm: 2, index: 33] sent vote req
2016-10-26 20:03:10.624528 I | raft: b3e1648d87d338a3 is starting a new election at term 16
2016-10-26 20:03:10.625076 I | raft: b3e1648d87d338a3 became candidate at term 17
2016-10-26 20:03:10.625454 I | raft: b3e1648d87d338a3 received vote from b3e1648d87d338a3 a
2016-10-26 20:03:10.625821 I | raft: b3e1648d87d338a3 [logterm: 2, index: 33] sent vote req
2016-10-26 20:03:11.605093 I | rafthttp: peer 87575c769a05665a became active
2016-10-26 20:03:11.605522 I | rafthttp: established a TCP streaming connection with peer 8
2016-10-26 20:03:11.605971 I | rafthttp: established a TCP streaming connection with peer 8
2016-10-26 20:03:11.608536 I | rafthttp: established a TCP streaming connection with peer 8
2016-10-26 20:03:11.620157 I | rafthttp: established a TCP streaming connection with peer 8
2016-10-26 20:03:11.723644 I | raft: b3e1648d87d338a3 is starting a new election at term 17
2016-10-26 20:03:11.724208 I | raft: b3e1648d87d338a3 became candidate at term 18
2016-10-26 20:03:11.724509 I | raft: b3e1648d87d338a3 received vote from b3e1648d87d338a3 a
2016-10-26 20:03:11.724920 I | raft: b3e1648d87d338a3 [logterm: 2, index: 33] sent vote req
2016-10-26 20:03:11.728096 I | raft: b3e1648d87d338a3 received vote from 87575c769a05665a a
2016-10-26 20:03:11.728332 I | raft: b3e1648d87d338a3 [quorum:2] has received 2 votes and 0
2016-10-26 20:03:11.728538 I | raft: b3e1648d87d338a3 became leader at term 18
2016-10-26 20:03:11.728897 I | raft: raft.node: b3e1648d87d338a3 elected leader b3e1648d87d

```

Node 1 (192.168.56.26) logs - embedded server:

```

[ppai@gd2-2 ~]$ ./tryembed
2016-10-26 20:02:40.504031 I | embed: listening for peers on http://192.168.56.26:2380
2016-10-26 20:02:40.504818 I | embed: listening for client requests on 192.168.56.26:2379

```



```

2016-10-26 20:02:40.506736 I | etcdserver: name = 192.168.56.26
2016-10-26 20:02:40.507141 I | etcdserver: data dir = 192.168.56.26.etcd
2016-10-26 20:02:40.507451 I | etcdserver: member dir = 192.168.56.26.etcd/member
2016-10-26 20:02:40.507717 I | etcdserver: heartbeat = 100ms
2016-10-26 20:02:40.507967 I | etcdserver: election = 1000ms
2016-10-26 20:02:40.508328 I | etcdserver: snapshot count = 10000
2016-10-26 20:02:40.508548 I | etcdserver: advertise client URLs = http://192.168.56.26:237
2016-10-26 20:02:40.508812 I | etcdserver: initial advertise peer URLs = http://192.168.56.
2016-10-26 20:02:40.509167 I | etcdserver: initial cluster = 192.168.56.26=http://192.168.5
2016-10-26 20:02:40.512804 I | etcdserver: starting member 9fbd239dbdea6f36 in cluster 7d1c
2016-10-26 20:02:40.513453 I | raft: 9fbd239dbdea6f36 became follower at term 0
2016-10-26 20:02:40.513778 I | raft: newRaft 9fbd239dbdea6f36 [peers: [], term: 0, commit:
2016-10-26 20:02:40.514073 I | raft: 9fbd239dbdea6f36 became follower at term 1
2016-10-26 20:02:40.524541 I | etcdserver: starting server... [version: 3.1.0-rc.0+git, clu
2016-10-26 20:02:40.527367 I | etcdserver/membership: added member 9fbd239dbdea6f36 [http:/
2016-10-26 20:02:41.516571 I | raft: 9fbd239dbdea6f36 is starting a new election at term 1
2016-10-26 20:02:41.518190 I | raft: 9fbd239dbdea6f36 became candidate at term 2
2016-10-26 20:02:41.519327 I | raft: 9fbd239dbdea6f36 received vote from 9fbd239dbdea6f36 a
2016-10-26 20:02:41.520259 I | raft: 9fbd239dbdea6f36 became leader at term 2
2016-10-26 20:02:41.521198 I | raft: raft.node: 9fbd239dbdea6f36 elected leader 9fbd239dbde
2016-10-26 20:02:41.523791 I | etcdserver: setting up the initial cluster version to 3.1
2016-10-26 20:02:41.524009 I | etcdserver: published {Name:192.168.56.26 ClientURLs:[http:/
2016-10-26 20:02:41.524035 I | embed: ready to serve client requests
2016-10-26 20:02:41.524914 N | embed: serving insecure client requests on 192.168.56.26:237
2016-10-26 20:02:41.529051 I | Etcd embedded server is ready.
2016-10-26 20:02:41.535469 N | etcdserver/membership: set the initial cluster version to 3.
2016-10-26 20:02:41.537787 I | etcdserver/api: enabled capabilities for version 3.1
2016-10-26 20:02:41.538741 I | 100: Key not found (/blah) [3]
2016-10-26 20:02:41.539280 I | etcdserver: skipped leadership transfer for single member cl
2016-10-26 20:02:41.542552 I | Etcd server stopped

```

<< this is where I add this node from other node >>

```

2016-10-26 20:03:11.545016 I | embed: listening for peers on http://192.168.56.26:2380
2016-10-26 20:03:11.545899 I | embed: listening for client requests on 192.168.56.26:2379
2016-10-26 20:03:11.557615 I | etcdserver: name = 192.168.56.26
2016-10-26 20:03:11.558117 I | etcdserver: data dir = 192.168.56.26.etcd
2016-10-26 20:03:11.558464 I | etcdserver: member dir = 192.168.56.26.etcd/member
2016-10-26 20:03:11.558719 I | etcdserver: heartbeat = 100ms
2016-10-26 20:03:11.558976 I | etcdserver: election = 1000ms
2016-10-26 20:03:11.559311 I | etcdserver: snapshot count = 10000
2016-10-26 20:03:11.560066 I | etcdserver: advertise client URLs = http://192.168.56.26:237
2016-10-26 20:03:11.565671 I | etcdserver: starting member 87575c769a05665a in cluster ebeb
2016-10-26 20:03:11.567248 I | raft: 87575c769a05665a became follower at term 0
2016-10-26 20:03:11.568354 I | raft: newRaft 87575c769a05665a [peers: [], term: 0, commit:
2016-10-26 20:03:11.568699 I | raft: 87575c769a05665a became follower at term 1
2016-10-26 20:03:11.576393 I | rafthttp: started HTTP pipelining with peer b3e1648d87d338a3
2016-10-26 20:03:11.577708 I | rafthttp: starting peer b3e1648d87d338a3...
2016-10-26 20:03:11.577985 I | rafthttp: started HTTP pipelining with peer b3e1648d87d338a3
2016-10-26 20:03:11.588289 I | rafthttp: started peer b3e1648d87d338a3
2016-10-26 20:03:11.596152 I | rafthttp: added peer b3e1648d87d338a3
2016-10-26 20:03:11.597978 I | etcdserver: starting server... [version: 3.1.0-rc.0+git, clu
2016-10-26 20:03:11.598685 I | rafthttp: started streaming with peer b3e1648d87d338a3 (writ
2016-10-26 20:03:11.600274 I | rafthttp: peer b3e1648d87d338a3 became active
2016-10-26 20:03:11.600988 I | rafthttp: established a TCP streaming connection with peer b
2016-10-26 20:03:11.601652 I | rafthttp: started streaming with peer b3e1648d87d338a3 (writ
2016-10-26 20:03:11.602080 I | rafthttp: established a TCP streaming connection with peer b
2016-10-26 20:03:11.602350 I | rafthttp: started streaming with peer b3e1648d87d338a3 (stre
2016-10-26 20:03:11.602801 I | rafthttp: started streaming with peer b3e1648d87d338a3 (stre
2016-10-26 20:03:11.607389 I | rafthttp: established a TCP streaming connection with peer b
2016-10-26 20:03:11.608760 I | rafthttp: established a TCP streaming connection with peer b
2016-10-26 20:03:11.721614 I | raft: 87575c769a05665a [term: 1] received a MsgVote message
2016-10-26 20:03:11.721928 I | raft: 87575c769a05665a became follower at term 18
2016-10-26 20:03:11.722197 I | raft: 87575c769a05665a [logterm: 0, index: 0, vote: 0] voted
2016-10-26 20:03:11.725032 I | raft: raft.node: 87575c769a05665a elected leader b3e1648d87d
2016-10-26 20:03:11.727482 I | etcdserver/membership: added member b3e1648d87d338a3 [http:/
2016-10-26 20:03:11.728129 N | etcdserver/membership: set the initial cluster version to 3.
2016-10-26 20:03:11.728590 I | etcdserver/membership: added member 87575c769a05665a [http:/
2016-10-26 20:03:11.730272 I | etcdserver: published {Name:192.168.56.26 ClientURLs:[http:/
2016-10-26 20:03:11.731022 I | embed: ready to serve client requests
2016-10-26 20:03:11.733890 N | embed: serving insecure client requests on 192.168.56.26:237
2016-10-26 20:03:11.736611 I | Etcd embedded server is ready.
2016-10-26 20:03:11.745538 I | 100: Key not found (/blah) [5]

```



prashanthpai commented on Oct 26, 2016



Just out of curiosity, I attached gdb to the process that has embedded etcd server and client in them, then from gdb I manually closed the residual/leaked socket fds and things worked just fine after that.



prashanthpai commented on Oct 26, 2016



So, what's also worked for me in the above example is this change w.r.t turning on `DisableKeepAlives` :

```
var DefaultTransport etcdclient.CancelableTransport = &http.Transport{
    Proxy: http.ProxyFromEnvironment,
    Dial: (&net.Dialer{
        Timeout: 30 * time.Second,
        KeepAlive: 0,
    }).Dial,
    TLSHandshakeTimeout: 10 * time.Second,
    DisableKeepAlives: true,
}

....

ecfg := &etcdclient.Config{
    Endpoints: []string{"http://" + myIP + ":2379"},
    Transport: DefaultTransport,
    HeaderTimeoutPerRequest: 3 * time.Second,
}
```



gyuhho commented on Oct 27, 2016 • edited

Member



@prashanthpai

From another node (192.168.56.25), add this node as a member using CLI

1. You first need to start the cluster that this node is added to.
2. And start this embedded etcd server with correct configuration (correct `initial-cluster` flag)

```
2016-10-26 11:23:20.737621 E | etcdserver/api/v2http: got unexpected response error (etcdse
2016-10-26 11:23:20.738186 I | client: etcd cluster is unavailable or misconfigured; error
```

indicates that it is misconfigured



prashanthpai commented on Oct 27, 2016



Yeah I've done exactly the same. The two node cluster works flawlessly when CLI is used. It also works using etcd client API package **as long as only one instance of client** is present in the process space and **embedded etcd isn't restarted**.

```
$ etcdctl --endpoint 192.168.56.26:2379 member list
b3e1648d87d338a3: name=default peerURLs=http://192.168.56.25:2380 clientURLs=http://192.168
eafab8344b329ec3: name=192.168.56.26 peerURLs=http://192.168.56.26:2380 clientURLs=http://1

$ etcdctl --endpoint 192.168.56.26:2379 cluster-health
member b3e1648d87d338a3 is healthy: got healthy result from http://192.168.56.25:2379
member eafab8344b329ec3 is healthy: got healthy result from http://192.168.56.26:2379
cluster is healthy

$ etcdctl --endpoint 192.168.56.26:2379 get /whatever
Error: 100: Key not found (/whatever) [5]
```

Also works using client API:

```
package main

import (
    "log"
    "time"

    etcdclient "github.com/coreos/etcd/client"
    "golang.org/x/net/context"
)

func main() {
```



```

ecfg := &etcdclient.Config{
    Endpoints:      []string{"http://192.168.56.26:2379"},
    Transport:      etcdclient.DefaultTransport,
    HeaderTimeoutPerRequest: 3 * time.Second,
}
c, err := etcdclient.New(*ecfg)
if err != nil {
    log.Fatal(err)
}
eclient := etcdclient.NewKeysAPI(c)
getOpts := &etcdclient.GetOptions{
    Quorum:    true,
    Recursive: true,
    Sort:      true,
}
if err != nil {
    log.Print(err)
}
_, err = eclient.Get(context.Background(), "blah/", getOpts)
log.Print(err)
}

```

```

$ ./testit
2016/10/26 23:13:38 100: Key not found (/blah) [5]

```

In my original post, I commented out the first call to client API and doing just that makes the second call succeed.

```

// Do some client get op. This will fail with key not found which is expected.
//     err = invokeClientOp()
//     if err != nil {
//         log.Print(err)
//     }

```

I can assure you that the cluster isn't misconfigured.



gyuho commented on Oct 27, 2016 • edited

Member + 🗨️

I am confused.

So your question is

```
err = invokeClientOp() // <--- This fails after embed server restart.
```

?

But this comment [#6733 \(comment\)](#) shows the same client error, which is expected because it will take some time for the new member to catch up the cluster data.

Or this error?

```
2016-10-26 11:23:20.738186 I | client: etcd cluster is unavailable or misconfigured; error #0: client:
etcd member http://192.168.56.26:2379 has no leader
```

Since you are adding one member to single-node, the leader-lost is expected. You might want to wait a few seconds before you send client requests to the new cluster.

Your log on adding a new member looks fine here [#6733 \(comment\)](#). The new member was successfully added.

I tried your code and also works in my side as well.

Please let me know if you have more questions.

Thanks.



prashanthpai commented on Oct 27, 2016

+ 🗨️

Thanks for trying it out. I'm sorry if I wasn't clear earlier.

I call `invokeClientOp()` twice, once before embedded etcd restart and once after in the same process. The cluster is always empty with no keys. The error I expect from **both calls** to `invokeClientOp()` is:

2016/10/26 23:13:38 100: Key not found (/blah) [5]

But this is what I'm getting for the second call to `invokeClientOp()` (after etcd restart):

2016-10-26 11:23:20.738186 I | client: etcd cluster is unavailable or misconfigured; error #0: client:

Since you are adding one member to single-node, the leader-lost is expected.
You might want to wait a few seconds before you send client requests to the new cluster.

I tried that too by putting in a sleep of 30 seconds before issuing the second call to `invokeClientOp()`

I'm almost certain that this problem is because the client or the server, one of them fail to break down all the TCP connections (could be an internal grpc thing). Even after `e1.Close()` which should result in etcd server shutting down all client connections, I see two TCP connections that persist:

To be more clear, I'll give you even simpler - just a single node reproducer to demo the leak:

single.go

```
func main() {
    // Start embedded etcd server
    cfg := getCfg()
    e, err := startServer(cfg)
    if err != nil {
        log.Fatal(err)
    }

    err = invokeClientOp()
    if err != nil {
        log.Print(err)
    }

    // Stop etcd server
    e.Close()
    os.RemoveAll(cfg.Dir)
    log.Print("Etcd server stopped")
    time.Sleep(3000 * time.Second)

    // At this point, some client connections still persist...
    // lsof -np <pid>
}
```

When you run the above snippet, you'll still see TCP connections open:

```
[ppai@gd2-2 ~]$ lsof -np `pidof single` | grep TCP
single 1001 ppai 23u IPv4 18792 0t0 TCP 192.168.56.26:41256->1
single 1001 ppai 24u IPv4 18793 0t0 TCP 192.168.56.26:etcd-cli
```

Let me know if you'd like to have further details. Thanks.



gyuhoo commented on Oct 28, 2016

Member + 🗨️

So when this client error happens

2016-10-26 11:23:20.738186 I | client: etcd cluster is unavailable or misconfigured; error #0: client: etcd member <http://192.168.56.26:2379> has no leader

does etcd log show that there was a valid leader before this timestamp? This log tells that there was no leader at the time of client request.

I'm almost certain that this problem is because the client or the server, one of them fail to break down all the TCP connections (could be an internal grpc thing).

I don't understand how the second client call with new TCP connection can fail from previous client request and receive the `has no leader` error. And your code uses v2 client, so grpc should not be related.

you'll still see TCP connections open:

So you mention that if you set `DisableKeepAlive: true`, the problem does not exist, right? I think that's the solution you need? v3 client has `Close` method to close all the connections, but I don't think we have that for v2 client?

/cc @xiang90



prashanthpai commented on Oct 28, 2016



does etcd log show that there was a valid leader before this timestamp?

Yes. The etcd server log does show that there was a leader. This is also confirmed by listing members using `etcdctl` command.

I don't understand how the second client call with new TCP connection can fail from previous client request and receive the has no leader error. And your code uses v2 client, so grpc should not be related.

That's exactly the mystery :) Do note that all this happens in the same process space. If etcd were a standalone external process, I won't see this issue as client connections terminate when the etcd process is restarted.

Although the v2 client may not have a `Close()` API, it's reasonable to expect the embedded etcd server to kill all client connections on shutdown, right? I had a glance at the code which is meant to close all client connections, couldn't figure where it was leaking connection.

So you mention that if you set `DisableKeepAlive: true`, the problem does not exist, right?

Correct.

think that's the solution you need? v3 client has `Close` method to close all the connections, but I don't think we have that for v2 client?

Using either `DisableKeepAlive: true` or using `clientv3` works as a workaround. Unfortunately, the default value of `DisableKeepAlive` is false. We use `libkv` which in turn uses v2 client as of today. Eventually everyone will move to v3, but until then, setting `DisableKeepAlive: true` in other libraries is going to be very hard.



gyuho commented on Nov 2, 2016

Member



@prashanthpai Could you file an issue to <https://github.com/docker/libkv>?

`DisableKeepAlive: true` is the solution that works for you.

And please let us know if you find any other issues with etcd.

Thanks.



gyuho closed this on Nov 2, 2016



prashanthpai commented on Nov 2, 2016



@gyuho I'm ok with closing of this issue. But it's still a workaround and not the solution I was looking for. v2 clients will probably be there for some time and leaking TCP connections doesn't do any good, with or without this issue. Moreover, at least theoretically, `DisableKeepAlive: true` will slow things down. If I knew etcd internals, I would've probably dugged further into this. It would be great if you could take another look into this whenever possible.

Thanks.



gyuho commented on Nov 3, 2016

Member



@prashanthpai I don't think there's not much we can do in etcd side?

Start embedded server in standalone mode.
Do some client request on client endpoint. This succeeds.
Restart embedded server to join an existing cluster.
Do some client request on client endpoint. This fails.

Do some client request on client endpoint. This fails. because it takes time to elect a leader and takes time to replicate the data to the newly joined member. Which is all expected.

I don't think the cause is the lingering TCP connection. And I am not sure how `DisableKeepAlive: true` resolved your issue because I am not familiar with libkv. And the code you provided works fine in my side.

@xiang90 Did we have ever face this issue where keep-alived TCP connection errors client?



prashanthpai commented on Nov 3, 2016



This fails. because it takes time to elect a leader and takes time to replicate the data to the newly joined member. Which is all expected.

I'm sorry. I don't understand this. The data is replicated to new member correctly. I have double tested this by running query from new instance of clients and it works fine. I am only facing the issue when **it's the same old v2 client** (inside same process) which talks to a reconfigured member (embedded etcd server) **at the same old endpoint** (inside same process). To put it bluntly, the client fails to recognise that the new etcd server at the old endpoint belongs to a different cluster now - regardless of how long I wait or if I call Sync. I'm assuming this is because stopping embedded etcd server does not cut all client TCP connections (can be seen open).

And I am not sure how `DisableKeepAlive: true` resolved your issue because I am not familiar with libkv.

libkv is irrelevant here. The issue is seen and easily reproducible without libkv and using etcd v2 client directly.

And the code you provided works fine in my side.

You sure ? I have a super easy reproducer (code in original post up there) which consistently fails. Are you able to see the lingering TCP connection on your side after stopping embedded etcd server ?

Thanks for your help @gyuho



gyuho commented on Nov 3, 2016

Member



@prashanthpai Last time, it worked fine with some sleep time before launching client request.

Will give another try.

I might have misread the logs.

Reopening.

gyuho reopened this on Nov 3, 2016

gyuho added this to the **v3.1.0** milestone on Nov 3, 2016

gyuho added the `component/client` label on Nov 3, 2016

xiang90 modified the milestone: **v3.2.0-maybe**, **v3.1.0** on Nov 4, 2016



xiang90 commented on Nov 4, 2016

Contributor



I moved this out of 3.1. It does not seem to block our 3.1 release.



gyuho commented on Nov 5, 2016

Member



@prashanthpai One thing I notice

```
// While this program is sleeping...
// From another node (192.168.56.25), add this node as a member using CLI
// etcd --listen-peer-urls http://192.168.56.25:2380 --listen-client-urls http://192.168.56.25:2379
```

```
// --advertise-client-urls http://192.168.56.25:2379 --initial-advertise-peer-
// --initial-cluster default=http://192.168.56.25:2380
```

If you are adding this node, `--initial-cluster` should include `192.168.56.26` but your command only includes single node. Can you double-check on this?



gyuho commented on Nov 5, 2016

Member + 🗨️

nvm. You are doing the other way... I was confused.

🚩 gyuho modified the milestone: **v3.1.0, v3.2.0-maybe** on Nov 5, 2016

👤 gyuho self-assigned this on Nov 5, 2016



gyuho commented on Nov 8, 2016

Member + 🗨️

@prashanthpai I checked the code base and confirmed that this is all expected.

1. There was a client request with keep-alive connection to the old node. Unless you create a new transport, the client request to the same port will re-use that keep-alive connection to the old listener, which is why the client requests to the newly-created node with the same port fails with error `client: etcd cluster is unavailable or misconfigured; error #0: client: etcd member http://localhost:12379 has no leader`.
2. We do not recommend to launch multiple `embed.Etcd` instances in one process. It can mess up the other node with conflicting configurations.

So, I would disable keep-alive in client request as we discussed. And separate embedded etcds into independent processes.

For reference, here's the full code that I used to debug

```
package main

import (
    "fmt"
    "log"
    "net/url"
    "os"
    "reflect"
    "time"

    v2client "github.com/coreos/etcd/client"
    "github.com/coreos/etcd/embed"
    "golang.org/x/net/context"
)

func main() {
    func() {
        log.Print("Starting name1")
        cfg1 := getCfg("name1", ":12379", ":12380", true)
        srv1, err := startServer(cfg1)
        if err != nil {
            log.Fatal(err)
        }
        log.Print("Started name1")

        // comment this out to prevent creating keep-alive connection to ':12379'
        if err = invokeClientOp(":12379", true); err != nil {
            log.Printf("EXPECTED FAIL: %v %v", reflect.TypeOf(err), err)
        }

        // Stop etcd server
        srv1.Close()
        os.RemoveAll(cfg1.Dir)
        log.Print("Stopped name1", <-srv1.Err())

        *srv1 = embed.Etcd{}
        srv1 = nil
        log.Print("Sleeping...")
        time.Sleep(5 * time.Second)
    }()

    log.Print("Starting name2")
}
```

```

cfg2 := getCfg("name2", ":22379", ":22380", true)
srv2, err := startServer(cfg2)
if err != nil {
    log.Fatal(err)
}
log.Print("Started name2")

defer func() {
    log.Println("STOPPING srv2")
    srv2.Server.HardStop()
    srv2.Close()
    os.RemoveAll(cfg2.Dir)
    log.Println("STOPPED srv2", <-srv2.Err())
}()

log.Print("Sleeping...")
time.Sleep(5 * time.Second)

func() {
    // etcdctl --endpoints http://localhost:22379 member add name3 http://localhost:123
    log.Print("Adding name3 to name2")
    c, err := v2client.New(v2client.Config{
        Endpoints: []string{"http://" + "localhost" + ":22379"},
    })
    if err != nil {
        log.Fatal(err)
    }
    _, err = v2client.NewMembersAPI(c).Add(context.TODO(), "http://"+"localhost"+":1238
    if err != nil {
        log.Fatal(err)
    }
    log.Print("Added name3 to name2")
}()

func() {
    log.Print("Starting name3 AGAIN")
    cfg3 := getCfg("name3", ":12379", ":12380", false) // change 12379 to 12385 and it
    cfg3.InitialCluster += ",name2=http://localhost:22380"

    srv3, err := startServer(cfg3)
    if err != nil {
        log.Fatal(err)
    }
    log.Print("Started name3 AGAIN")

    time.Sleep(5 * time.Second)
    if srv2.Server.Leader().String() != srv3.Server.Leader().String() {
        log.Fatal("leader has not been elected")
    } else {
        fmt.Println("srv2 leader:", srv2.Server.Leader().String())
        fmt.Println("srv3 leader:", srv3.Server.Leader().String())
    }

    defer func() {
        log.Println("STOPPING srv3")
        srv3.Server.HardStop()
        srv3.Close()
        os.RemoveAll(cfg3.Dir)
        log.Println("STOPPED srv3", <-srv3.Err())
    }()

    log.Println("STARTING invokeClientOp with QGET to name3")
    err = invokeClientOp(":12379", true)
    if err != nil {
        te, ok := err.(v2client.Error)
        if !ok {
            log.Printf("UNEXPECTED FAIL? %v %q", reflect.TypeOf(err), err.Error())
        } else {
            log.Printf("EXPECTED ERROR: %v", te)
        }
    }
    time.Sleep(time.Second)

    log.Println("STARTING invokeClientOp without QGET to name3")
    err = invokeClientOp(":12379", false)
    if err != nil {
        te, ok := err.(v2client.Error)
        if !ok {
            log.Printf("UNEXPECTED FAIL? %v %q", reflect.TypeOf(err), err.Error())
        } else {
            log.Printf("EXPECTED ERROR: %v", te)
        }
    }
    time.Sleep(time.Second)

    log.Println("STARTING invokeClientOp with QGET to name3")
    err = invokeClientOp(":12379", true)

```



```

    if err != nil {
        te, ok := err.(v2client.Error)
        if !ok {
            log.Printf("UNEXPECTED FAIL? %v %q", reflect.TypeOf(err), err.Error())
        } else {
            log.Printf("EXPECTED ERROR: %v", te)
        }
    }

    time.Sleep(time.Second)
    log.Print("DONE!")
}()
}

func getCfg(name, cport, pport string, new bool) *embed.Config {
    cfg := embed.NewConfig()

    cfg.Name = name
    cfg.Dir = name + ".etcd"

    listenClientURL, _ := url.Parse("http://" + "localhost" + cport)
    cfg.ACUrls = []url.URL{*listenClientURL}
    cfg.LCUrls = []url.URL{*listenClientURL}

    listenPeerURL, _ := url.Parse("http://" + "localhost" + pport)
    cfg.APUrls = []url.URL{*listenPeerURL}
    cfg.LPUrls = []url.URL{*listenPeerURL}

    cfg.ClusterState = embed.ClusterStateFlagNew
    if !new {
        cfg.ClusterState = embed.ClusterStateFlagExisting
    }
    cfg.InitialCluster = name + "=" + listenPeerURL.String()

    return cfg
}

func startServer(cfg *embed.Config) (*embed.Etcd, error) {
    etcd, err := embed.StartEtcd(cfg)
    if err != nil {
        return nil, err
    }
    select {
    case <-etcd.Server.ReadyNotify():
        log.Print("embedded server", cfg.Name, "is ready")
        return etcd, nil
    case err := <-etcd.Err():
        return nil, err
    }
}

func invokeClientOp(cport string, quorum bool) error {
    c, err := v2client.New(v2client.Config{
        Endpoints: []string{"http://" + "localhost" + cport},
    })
    if err != nil {
        log.Fatal(err)
    }
    _, err = v2client.NewKeysAPI(c).Get(context.Background(), "mykey", &v2client.GetOptions{
        Quorum: quorum,
    })
    return err
}

/*
embed/serve.go, line 58

func (sctx *serveCtx) serve(s *etcdserver.EtcdServer, tlscfg *tls.Config, handler http.Handler) {
    fmt.Println("[DEBUG] STARTED serve on", s.Cfg.Name)
    defer fmt.Println("[DEBUG] DONE serve on", s.Cfg.Name)

    http1 := m.Match(cmux.HTTP1())

    go func() {
        fmt.Println("[DEBUG] STARTED srvhttp.Serve", s.Cfg.Name)
        err := srvhttp.Serve(http1)
        fmt.Println("[DEBUG] DONE srvhttp.Serve", s.Cfg.Name, err, http1.Close())
        errc <- err
    }()

    etcdserver/api/v2http/client.go, line 145

```

```
func (h *keysHandler) ServeHTTP(w http.ResponseWriter, r *http.Request) {
    if !allowMethod(w, r.Method, "HEAD", "GET", "PUT", "POST", "DELETE") {
        return
    }
    srv, ok := h.server.(*etcdserver.EtcdServer)
    if !ok {
        panic("got non *etcdserver.EtcdServer")
    }
    fmt.Println("[DEBUG] STARTED (h *keysHandler) ServeHTTP", srv.Cfg.Name)
    defer func() {
        fmt.Println("[DEBUG] DONE (h *keysHandler) ServeHTTP", srv.Cfg.Name)
    }()
}
*/
```

 **gyuho** closed this on Nov 8, 2016



xiang90 commented on Nov 8, 2016

Contributor + 🗨️

@gyuho @prashanthpai

Or you can create different transporter for different clients. Do not reuse the old transport which still maintains a good connection to the stopped member.



prashanthpai commented on Nov 14, 2016

+ 🗨️

@gyuho @xiang90 Thanks guys Using a different transport each time works for me.



prashanthpai commented on Nov 14, 2016 • edited

+ 🗨️

There seems to be a one line simple fix in etcd for this. Let me know what you guys think of this:

```
diff --git a/client/client.go b/client/client.go
index f9131b4..e5b53fa 100644
--- a/client/client.go
+++ b/client/client.go
@@ -159,6 +159,7 @@ func (cfg *Config) checkRedirect() CheckRedirectFunc {
    type CancelableTransport interface {
        http.RoundTripper
        CancelRequest(req *http.Request)
+       CloseIdleConnections()
    }

    type CheckRedirectFunc func(via int) error
```

More on **CloseIdleConnections()** [here](#)

Calling **CloseIdleConnections()** after I'm done with client allowed me to reuse same transport in a new instance of the client. This worked for me. @gyuho Please give it a try.



Write Preview

AA B i “ < > ↺ ⋮ ⋮ ⋮ ↶ @ 📌

Leave a comment

Attach files by dragging & dropping, [selecting them](#), or pasting from the clipboard.

📄 Styling with Markdown is supported

Comment



