

聊聊Raft协议

1. 参考资料

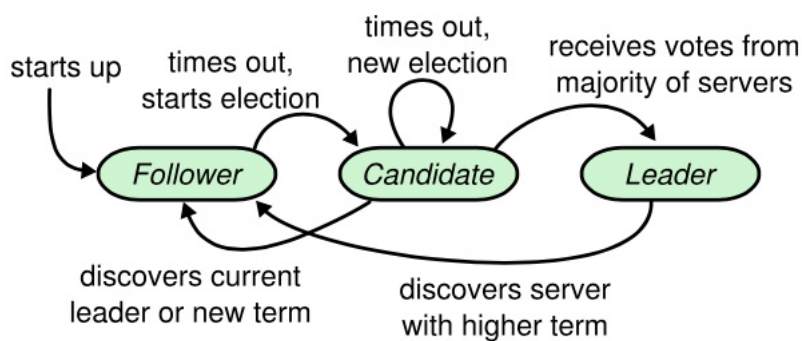
- 1.1 In Search of an Understandable Consensus Algorithm
- 1.2 寻找一种易于理解的一致性算法
- 1.3 Raft协议精解
- 1.4 Raft协议动画演示
- 1.5 goraft/raftd

The original project authors have created new raft implementations now used in etcd and InfluxDB.

goraft 的作者参与了 etcd 项目的实现，所以 goraft 是有参考价值的。

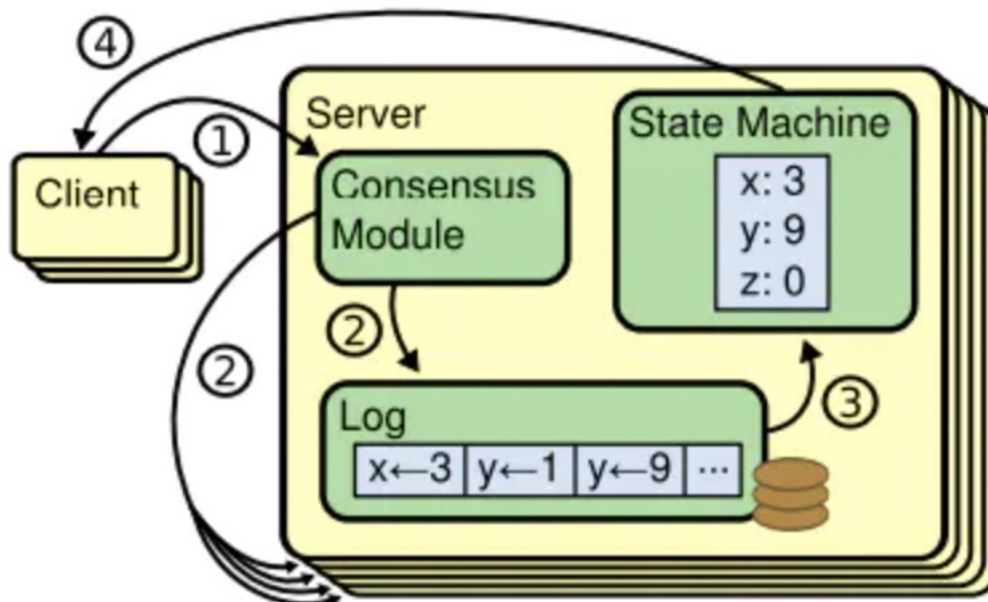
2. Node的简单介绍

2.1 node的三种状态(state)



- Leader
- Candidate
- Follower

2.2 3种不同的存储



2.2.1 Write-Ahead Log (WAL)

存储:文件

```
4f
raft:join">{"name":"2832bfa","connectionString":"http://localhost:4001"}
```

```

e
raft:nop      4f
raft:join">{"name":"3320b68","connectionString":"http://localhost:4002"}
4f
raft:join">{"name":"7bd5bdc","connectionString":"http://localhost:4003"}
e
raft:nop      29
write>{"key":"foo","value":"bar"}
29
write>{"key":"aaa","value":"bbb"}
29
write>{"key":"bbb","value":"ccc"}
29
write>{"key":"ddd","value":"eee"}
e

raft:nop      e

raft:nop      29

write>{"key":"foo","value":"bar"}
29
write>{"key":"foo","value":"bar"}

```

2.2.2 状态信息(状态信息)

commitIndex、peer等

存储:文件

2.2.3 内存数据库

存储:内存

```

// The key-value database.
type DB struct {
    data map[string]string
    mutex sync.RWMutex
}

```

2.3 节点之间的通讯&重要的名词解释



Leader --> Follower

```

{
  "Term": 17,
  "PrevLogIndex": 26,
  "PrevLogTerm": 17,
  "CommitIndex": 26,
  "LeaderName": "2832bfa",
  "Entries": [{
    "Index": 27,
    "Term": 17,
    "CommandName": "write",
    "Command": "eyJrZXkiOiJhYWUiLCJ2YWx1ZSI6ImJiYiJ9Cg=="
  }]
}

```

名词解释

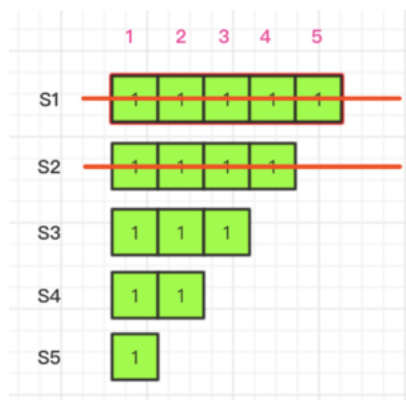
3. Leader Election(选举过程)

注意:不需要集群中节点的数量是奇数 可以是4、8个, 都没关系

3.1 什么时候开始选举?

3.2 投票相关

看参考资料1.4 的演示动画



这个时候, 谁可能被选为leader, 注意: raft协议中这样的要求

candidate's log is at least as up-to-date as receiver's log then vote

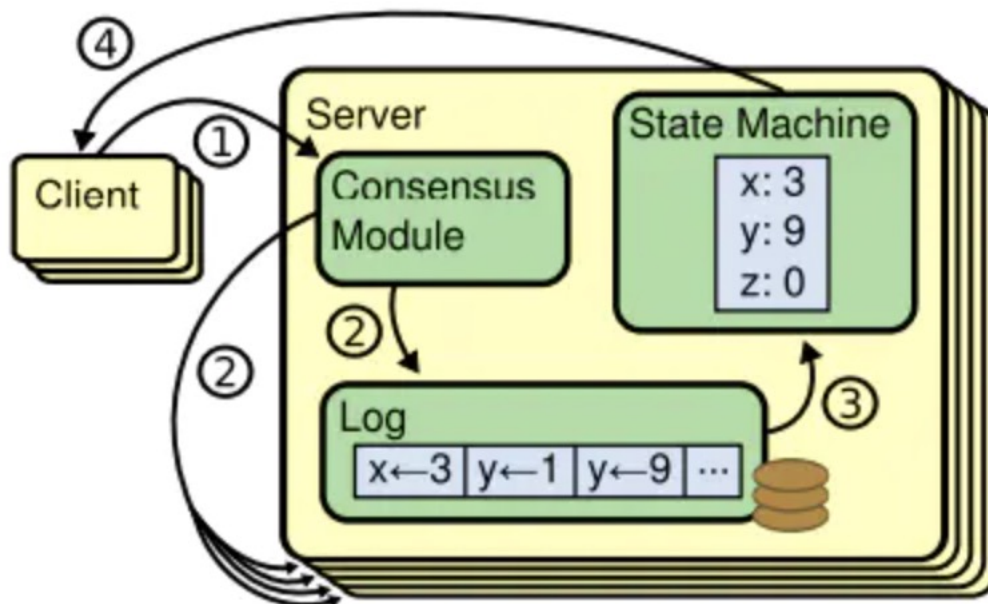
解释起来, 就是必须同时满足以下2个条件, 才会给candidate投票

- `candidate.LastLogTerm >= receiver.LastLogTerm`
- `candidate.LastLogIndex >= receiver.LastLogIndex`

3.3 当选&当选后的一系列动作

4. Log Replication(数据写入)

聊聊RAFT的一个实现(3)-commit



Step0. Client 发出 WriteCommand

Step1. 先写 Leader 的Log

Step2. 在通过AppendRequest, 写 Follower 的Log

Step3. 执行 Leader 的Commit(写内存数据库)

Step4. 执行 Follower 的Commit

Step5. 给 Client 返回结果

注意: 写入动作, 只能由 Leader 来发起

5. 数据读取&watch

5.1 默认consul有3种一致性模型

- default
- consistent
- stale

默认情况下，consul server(follower) 不提供数据查询，仅转发请求给consul server(leader) [consistency](#)

其中consistent模式是强一致性的，其它两种模式都不能保证强一致性。用stale模式可以提高吞吐能力，当然数据短时间内可能会有不一致问题

5.2 default 和 consistent 模式的区别

5.3 如何使用stale模型

```
curl -v 'http://dev1:8500/v1/health/service/es?dc=dc1&passing=1&stale'
```

5.4 watch是怎么回事？

[玩转CONSUL\(1\)-WATCH机制探究](#)