分布式系统 作业五

19335015 陈恩婷

第一题

1. Zookeeper

- zookeeper 的 leader 选举存在两个阶段，一个是服务器启动时 leader 选举，另一个是运行过程中 leader 服务器宕机
- 有以下几个参数：
  - 服务器 ID(myid)：编号越大在选举算法中权重越大
  - 事务 ID(zxid)：值越大说明数据越新，权重越大
  - 逻辑时钟(epoch-logicalclock)：同一轮投票过程中的逻辑时钟值是相同的，每投完一次值会增加
- 选举状态：
  - LOOKING: 竞选状态
  - FOLLOWING: 随从状态，同步 leader 状态，参与投票
  - OBSERVING: 观察状态，同步 leader 状态，不参与投票
  - LEADING: 领导者状态

① 服务器启动时的 leader 选举

- 每个节点启动的时候都 LOOKING 观望状态，接下来就开始进行选举主流程。这里选取三台机器组成的集群为例。第一台服务器 server1启动时，无法进行 leader 选举，当第二台服务器 server2 启动时，两台机器可以相互通信，进入 leader 选举过程。
- 每台 server 发出一个投票，由于是初始情况，server1 和 server2 都将自己作为 leader 服务器进行投票，每次投票包含所推举的服务器myid、zxid、epoch，使用（myid，zxid）表示，此时 server1 投票为（1,0），server2 投票为（2,0），然后将各自投票发送给集群中其他机器
- 接收来自各个服务器的投票，集群中的每个服务器收到投票后，首先判断该投票的有效性，如检查是否是本轮投票（epoch）、是否来自 LOOKING 状态的服务器。
- 分别处理投票，针对每一次投票，服务器都需要将其他服务器的投票和自己的投票进行对比，对比规则如下：
  - 优先比较 epoch
  - 检查 zxid，zxid 比较大的服务器优先作为 leader
  - 如果 zxid 相同，那么就比较 myid，myid 较大的服务器作为 leader 服务器
- 统计投票，每次投票后，服务器统计投票信息，判断是都有过半机器接收到相同的投票信息。server1、server2 都统计出集群中有两台机器接受了（2,0）的投票信息，此时已经选出了 server2 为 leader 节点
- 改变服务器状态，一旦确定了 leader，每个服务器响应更新自己的状态，如果是 follower，那么就变更为 FOLLOWING，如果是 Leader，变更为 LEADING。此时 server3继续启动，直接加入变更自己为 FOLLOWING

② 运行过程中的 leader 选举

- 变更状态。leader 挂后，其他非 Oberver服务器将自身服务器状态变更为 LOOKING。
- 每个 server 发出一个投票。在运行期间，每个服务器上 zxid 可能不同。
- 处理投票。规则同启动过程。
- 统计投票。与启动过程相同。
- 改变服务器状态。与启动过程相同。

2. Redis

- slave发现自己的master变为FAIL
- 将自己记录的集群currentEpoch加1，并广播FAILOVER_AUTH_REQUEST信息
- 其他节点收到该信息，只有master响应，判断请求者的合法性，并发送FAILOVER_AUTH_ACK，对每一个epoch只发送一次ack
- 尝试failover的slave收集FAILOVER_AUTH_ACK
- 超过半数后变成新Master
- 广播通知其他集群节点。

3. MongoDB

新版本的MongoDB用Raft取代了Bully，下面介绍Raft

- follower 先增加自己的当前任期号并且转换到 Candidate 状态。
- 投票给自己并且并行地向集群中的其他服务器节点发送 RequestVote RPC。
- Candidate 会一直保持当前状态直到以下三件事情之一发生：
    - 收到过半的投票，赢得了这次的选举
    - 其他的服务器节点成为 leader
    - 一段时间之后没有任何获胜者。
- 若Candidate 赢得选举

  当一个 Candidate 获得集群中过半服务器节点针对同一个任期的投票，它就赢得了这次选举并成为 Leader 。对于同一个任期，每个服务器节点只会投给一个 Candidate ，按照先来先服务的原则。一旦 Candidate 赢得选举，就立即成为 Leader 。然后它会向其他的服务器节点发送心跳消息来确定自己的地位并阻止新的选举。

- 若其他节点成为Leader

  在等待投票期间，Candidate 可能会收到另一个声称自己是 leader 的服务器节点发来的 AppendEntries RPC 。如果这个 Leader 的任期号（包含在RPC中）不小于 candidate 当前的任期号，那么 Candidate 会承认该 Leader 的合法地位并回到 Follower 状态。 如果 RPC 中的任期号比自己的小，那么 Candidate 就会拒绝这次的 RPC 并且继续保持 Candidate 状态。

- 若没有获胜者

  如果有多个 follower 同时成为 candidate ，那么选票可能会被瓜分以至于没有 candidate 赢得过半的投票。当这种情况发生时，每一个候选人都会超时，然后通过增加当前任期号来开始一轮新的选举。然而，如果没有其他机制的话，该情况可能会无限重复。

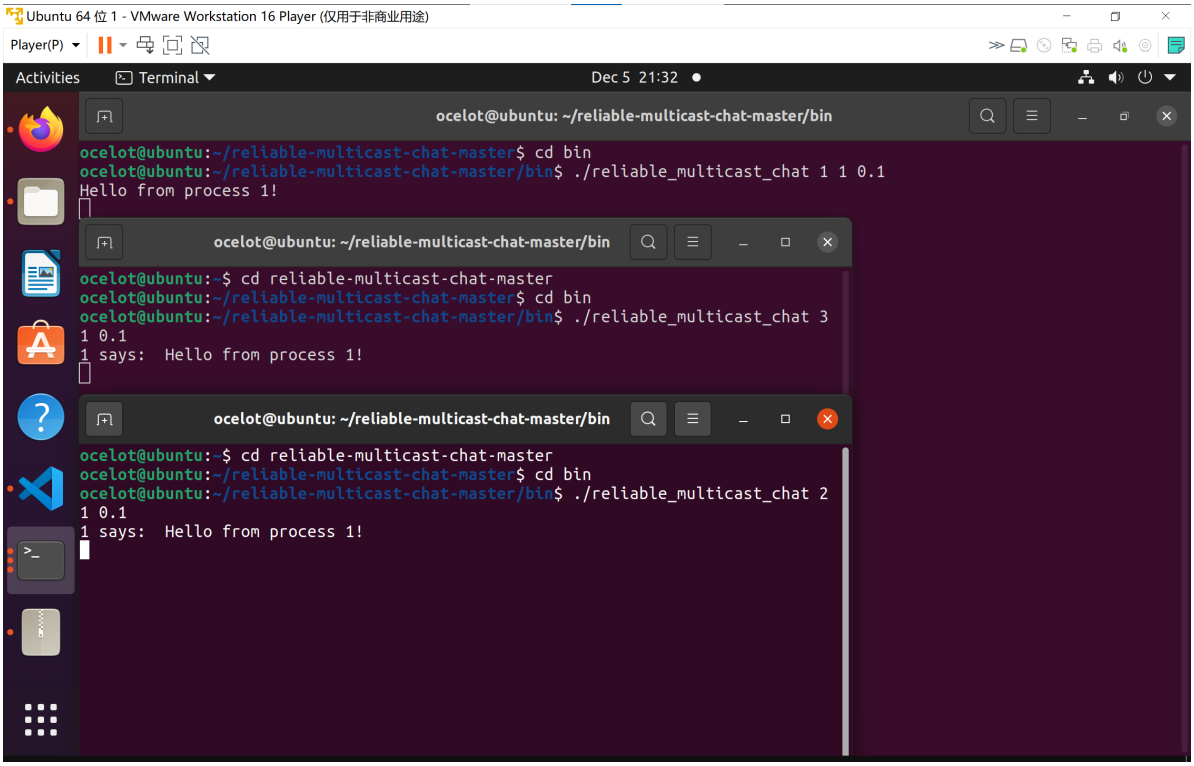
4. Cassandra

> Cassandra is a peer-to-peer distributed database that runs on a cluster of homogeneous nodes. Cassandra has been architected from the ground up to handle large volumes of data while providing high availability. Cassandra provides high write and read throughput. **A Cassandra cluster has no special nodes i.e. the cluster has no masters, no slaves or elected leaders**. This enables Cassandra to be highly available while having no single point of failure.

Cassandra中不存在主从节点和leader节点的概念，无进行选举。

第二题

我选择的是https://github.com/daeyun/reliable-multicast-chat上的软件，直接下载整个项目，找到其中的可执行文件并运行：



如图所示，成功运行了可靠多播。