**电 子 科 技 大 学**

**实 验 报 告**

**学生姓名指导教师：薛瑞尼**

**一、实验室名称：**

**二、实验项目名称：Simple Distributed Cache System**

**三、实验学时：**

**四、实验原理：完成一个简易分布式缓存系统**

**五、实验目的：**

**六、实验内容：**

a. Cache数据以Key-value形式存储在缓存系统节点内存中（不需要持久化）；

b. Cache数据以既定策略（round-robin或hash均可，不做限定）分布在不同节点（不考虑副本存储）；

c. 服务至少启动3个节点，不考虑节点动态变化；

i. 所有节点均提供HTTP访问入口；

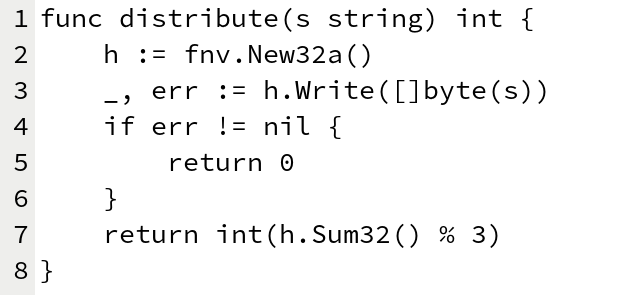
ii. 客户端读写访问可从任意节点接入，每个请求只支持一个key存取；

iii. 若数据所在存储服务器与接入服务器不同，接入服务器通过内部RPC从目标存储服务器获取数据，再返回至客户端。

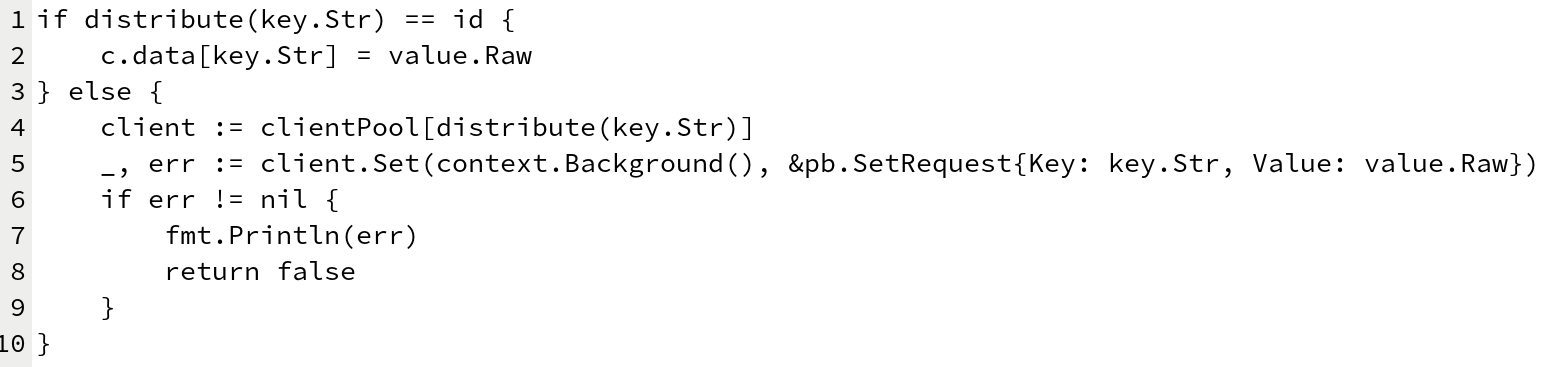
**七、实验器材（设备、元器件）：**电脑

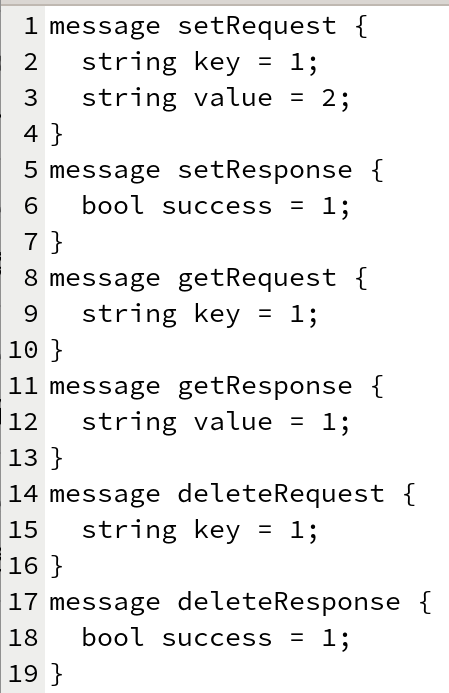
**八、实验步骤：**

由于分布式的设计，每个服务器都有自己的全局唯一id，根据id来生成对应连接的地址和判断key的存储服务器。由于采用docker compose，因此连接的地址可以设置为fmt.Sprintf("server%v:50000", i)，对应docker compose需要设置services的name为server0这样的格式。

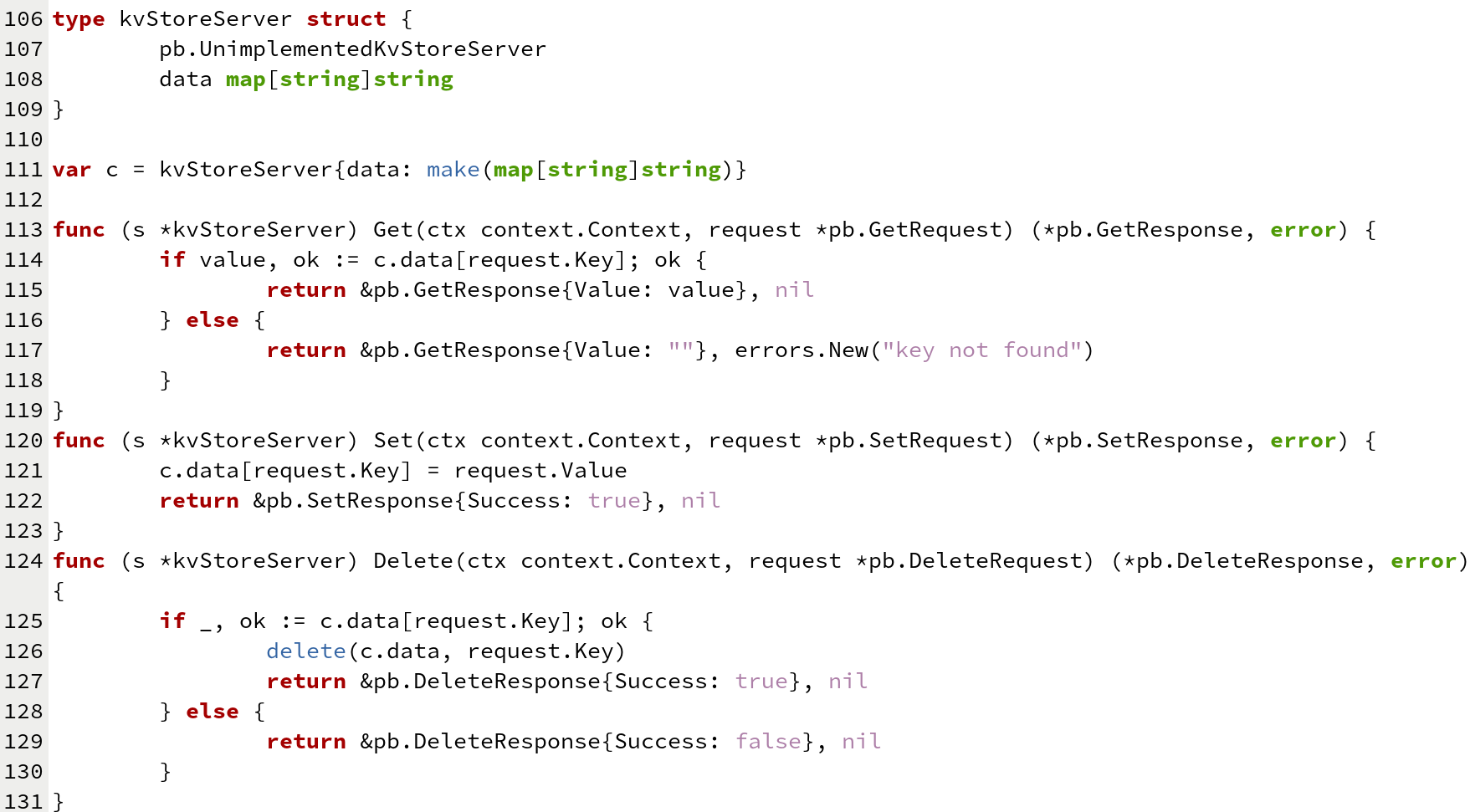
对于key要分布在不同的节点，采用hash的方式决定存储的节点。使用快速的FNV-1a算法生成hash，取余后就是对应的节点id。

服务器先开启自身的grpc服务端，然后连接其他所有的服务器，生成的client保存在var clientPool = make(map[int]pb.KvStoreClient)中。最后才是开启http服务，这样的顺序可以避免收到http请求时grpc还没有初始化完成的情况。

对于每个key，先根据hash计算出对应的服务器，若是本机就直接查询，否则发送grpc请求到对应服务器。

grpc定义为：

grpc调用部分为：



**九、实验数据及结果分析：**

**十、实验结论：**这是一个简单的分布式系统，对于理解分布式和rpc很有帮助。

**十一、总结及心得体会：**

**十二、对本实验过程及方法、手段的改进建议：**

**报告评分：**

**指导教师签字：**