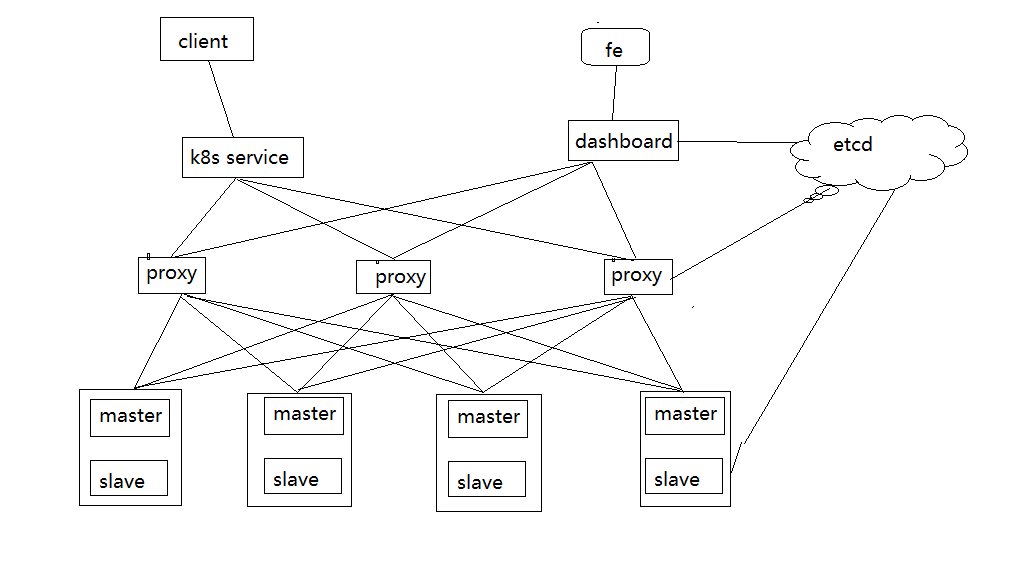
容器平台codis简介

codis的基本架构



容器平台基于k8s对codis做了一些改进，在这里我们来对codis的架构做一个详细的解析：

1. k8s service这个是基于容器平台的一个特性所做的负载均衡，主要是运用kubernetes的原始特性，其原始策略使用轮询以及session粘和性保证请求可以均衡的分发到每一个proxy上，而且保持了每一个连接固定的由一个proxy处理；
2. dashboard 主要用于管理集群的各个组件以及包括对集群的扩容，缩容请求的处理等等。
3. fe 主要用于将融客户端管理集群和发送扩容、缩容、数据迁移等相关操作命令；
4. etcd 整个集群的信息都存储在etcd集群中，集群的所有相关信息都有etcd负责存储；
5. proxy 这里容器平台对proxy做了容器化处理，结合k8s集群的自动扩缩容特性，可以更简洁的对proxy进行扩缩容操作；
6. master与slave的切换，主要采用脚本的方式实现，测试完全可以实现主从自动切换的目的。

主要做了如下改造，对dashboard做了自动删除功能，proxy添加自删除功能，group组master，slave，主动切换功能。

增加slave自动rdb备份功能，主从切换之后自动判断slave进行rdb备份，rdb数据储存到共享网盘中

客户端连接proxy采用容器平台特有的service代理功能，service代理采用轮询与session粘和性来分发客户端请求。

容器平台添加亲和性选项，确保每个group中的slave和master不在同一个物理服务器上。

考虑到各种场景，容器平台codis提供了完备的容灾方案，尽最大力度保证数据的安全性。

当redis的内存超过设置的maxmemory限制时，当接下来继续添加新数据时，我们这里需要制定默认的策略。redis可以只为命令返回错误，但是这样将占用更多的内存，或者我们在每次添加新数据时，回收掉一些旧的数据以避免内存限制。

redis内存碎片率:

内存碎片率 = used\_memory\_rss/used\_memory

跟踪内存碎片率对理解redis实例的资源性能是非常重要的。内存碎片率稍大于1是合理的，这个值表示内存碎片率比较低，也说明redis没有发生内存交换。但是如果大于1.5，那就说明redis消耗了实际物理内存的150%，其中50%是内存碎片率。若是内存碎片率低于1的话，说明redis内存分配超出了物理内存，操作系统正在进行内存交换，这样回引起显著的响应延迟。