Impala负载均衡方案

概述

Impala分为是三个组件，statestored/catalogd和impalad，其中statestored和catalogd是单点的，没有高可用的需求，因为这两个实例是无状态的，本身不存储任何数据，例如catalogd的数据存储在第三方数据库(例如mysql中)，statestore的数据全都存储在内存中，可以通过简单的主备的方式来实现高可用，本文最后会提到。正常情况下只有master提供服务，slave只是运行状态但是不接受任何请求，当master出现问题之后再slave提升为master提供服务。

而对于impalad节点，每一个节点都可以提供jdbc和thrift等服务，并且对于连接到该impalad的查询作为coordinator节点（需要消耗一定的内存和CPU）存在，为了保证每一个节点的负载的平衡需要对于这些impalad做一下均衡，负载均衡分为四层负载均衡和七层负载均衡，前者是针对运输层的，后者是针对应用层的，区别在于前者不需要了解应用协议，只需要对传输层收到的IP数据包进行转发，而后者需要了解应用协议的，而对于impalad这种SQL服务器，就需要使用SQL协议的代理，所以七层代理对于impalad是有点不切实际的。

下面以haproxy作为四层代理服务器来说明如何对impalad节点进行load balance。官方推荐的代理方案参见该文档。

除了本文档提到的使用 load-balancing proxy server外，最简单的方案莫过于使用DNS做负载均衡，但是DNS的性能一般，所以这里我们按照官方的建议使用haproxy实现四层的负载均衡，相对于通常的负载均衡的实现，这里我们还需要拷贝kerberos的支持。

impalad负载均衡

首先下载haproxy：http://www.haproxy.org/download/1.6/src/haproxy-1.6.10.tar.gz，这里下载的源码，安装非常方便，使用如下命令：

```
make TARGET=generic
```

构建完成之后会在当前目录下生成haproxy可执行文件，然后关键的是对haproxy进行配置，可以参考如下配置文件：

```
cat etc/haproxy.cfg
global
    log 127.0.0.1 local0
    uid 71488
    gid 1003
    deamon
    pidfile /path/to/haproxy/pid/haproxy.pid
    maxconn 65536

defaults
    backlog 2048
    balance roundrobin
    log global
    mode tcp
    stats enable
    stats refresh 5s
    retries 3
    timeout connect 120s
    timeout client 600s
    timeout server 600s

listen impala_ha
    bind 0.0.0.0:8006
    mode tcp

    balance roundrobin
    server impala1 hadoop461.lt.server.org:21050 check
    server impala2 hadoop462.lt.server.org:21050 check
    server impala3 hadoop463.lt.server.org:21050 check
    server impala4 hadoop464.lt.server.org:21050 check
    server impala5 hadoop465.lt.server.org:21050 check
```

这里只配置了一个负载均衡代理impala的hs2服务，监听在本机的8006端口，代理模式为tcp，也就是四层代理，使用roundrobin的方式轮询后端服务器，这里使用了五台后端impalad节点，分别转发到impalad的hive server服务，除了对这个服务进行负载均衡，还可以对其他的服务进行负载均衡，

只需要添加一个listen配置就可以了。还需要注意的是uid和gid分别是当前的用户id和组id。

配置好配置文件之后，启直接启动haproxy：

```
./haproxy -f ./etc/haproxy.cfg
```

此时haproxy如果没出现什么问题就会以daemon的方式启动，此时通过beline或者jdbc代码就可以通过访问haproxy_host:8006来访问impala了。

kerberos配置

但是对于配置了kerberos认证的集群，还需要额外的处理，因为对于开启kerberos的impala使用的url格式为：jdbc:hive2://haproxy_host:8006/default;principal=impala/${hostname}@realm;而一般情况下不同的impalad节点使用相同的impala.keytab，但是使用不同的impala principal，例如 hadoop461.lt.server.org使用的principal是impala/hadoop461.lt.server.org@realm，而hadoop462.lt.server.org使用的principal是impala/hadoop462.lt.server.org@realm，由于在创建impala连接的时候只能在url中指定一个principal的配置，这样就导致创建连接的时候会出现null异常（应该是空指针了）。

所以我们需要做的是如果将不同的impalad识别的principal设置成相同的，在impalad的参数中存在两个关于principal的：-principal和-be_principal，前者设置的是外部连接使用的principal，也就是url中需要填的，后者是impalad和其它节点通信使用的principal，因此可以通过如下的处理方式修改principal：

- 创建一个新的proxy.keytab，假设它的principal是proxy/haproxy_host@realm.
- 执行如下操作分别将不同impalad使用的的impala.keytab合并成一个keytab，这样使用同一个keytab可以对应两个principal，分别是：proxy/haproxy_host@realm和impala/${hostname}@realm

  ```
  ktutil
  ktutil:  rkt proxy.keytab
  ktutil:  rkt impala.keytab
  ktutil:  wkt proxy_impala.keytab
  ktutil:  quit
  ```

- 然后将合并之后的proxy_impala.keytab分别拷贝到对应的impalad机器上，通常需要将其设置为400，只有当前用户可读，防止其他用户使用该keytab非法访问。
- 分别重启每一个impalad节点，使用如下的kerberos配置参数：

  ```
  --principal=impala/${hostname}@realm
  --be_principal=proxy/haproxy_host@realm
  --keytab_file=path_to_proxy_impala.keytab
  ```

重新创建到proxy服务器的jdbc连接，It works！

总结

最后，haproxy本身又是一个单点服务，可以在它之上再做一个高可用配置，类似于statestored和catalogd服务，他们的需求都是主备配置，所有的服务由主节点提供，当主节点挂了之后备节点提升为主节点服务，这种工作通常使用keepalived完成。

本文介绍了impala集群所有服务的高可用方案，尤其是impalad配置高可用服务的流程。