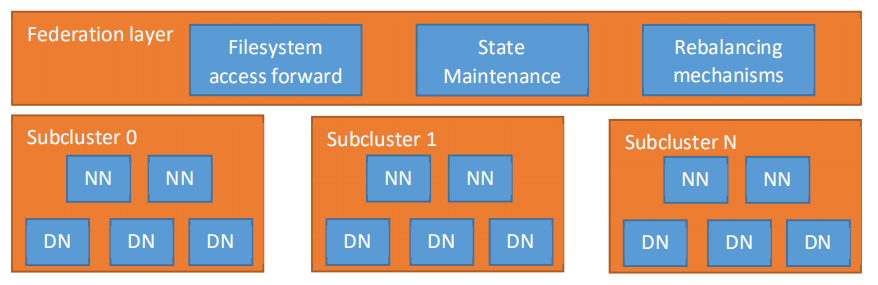
HDFS Router-based Federation

Hadoop 3.0以后添加了Router-Based Federation，添加了RPC路由层，提供了多个HDFS命名空间的联合视图，与ViewFs功能类似，不同之处在于Mount table由服务器端的路由层维护，这简化了客户端对Federated Cluster的访问。

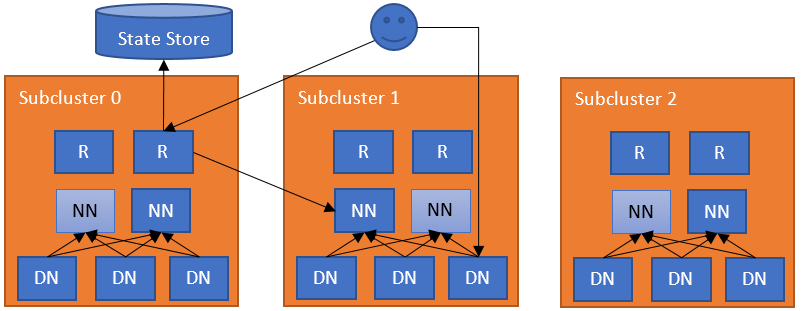
# RBF架构

RBF底层由多个namespace的子集群组成，可以是独立的hdfs集群、Federation集群或者混合集群。通常在每个NameNode节点上部署Router，向客户端提供NameNode接口的服务，Router是无状态的，其通过StateStore获取Federation状态信息，类似于VeiwFS中的MountTable。RBF为多个子集群提供了Federation Layer，如下图所示：



用户通过该Layer，可以透明访问任意的子集群，并且支持跨子集群的数据均衡，为了实现以上功能，RBF提供以下模块：

* Filesystem access Forward，在Router Component中提供和NameNode相同的接口，并根据State Store中的信息将客户端请求提交给相应的相应的子集群
* State Maintenance，State Store类似于ViewFS维护了Mount Table，及子集群的负载信息
* Rebalancing Mechanisms，在每个子集群中可以启动多个Routers，如下所示：



可以通过负载均衡降低Router的负载。

DFS客户端通过Router访问Federated FileSystem中的文件，Router从Route Table中获取哪个子集群有文件，从Membership table中获取该子集群的NameNode节点。Router将请求转发给对应的NameNode，并给客户端返回响应信息，即文件所在的DN，最后客户端直接和DN交互获取数据。

# Router功能

在HDFS中可以启动多个Router，其主要的工作分为两个部分：

* Federated Interface，向客户端提供全局的NameNode接口，并将请求转发给相应子集群的Active NameNode
* Namenode Heartbeat，在StateStore中维护NameNode信息

## **State Store**

Router基于State Store转发请求，其维护以下信息：

* SubCluster的状态，包括block访问负载、磁盘可用量、HA状态等
* 目录/文件与subcluster的映射关系，即remote Mount Table
* Rebalancer operation的状态 //有待进一步研究
* Routers的状态

### **Tables**

State Store中维护了Router中间共享的多数据实体，其包括以下几个部分：

*[MountTable, MembershipState, RouterState]*

1. Membership，在Federation中NameNodes的状态信息，包括子集群的存储数据量及节点数目。Router与一个或者多个NameNode通过心跳获取信息。这些信息用于：

* 获取子集群的Ative NameNode，并重定向查询
* 决策哪个子集群承载特定数据

由于多个Routers可以监控一个NameNode，Router使用从State Store查询的数据，自动丢弃原数据。在zk中的存储条目如下：

*[zk: localhost:2181(CONNECTED) 1] ls /hdfs-federation/MembershipState*

*[null-ns2-cmhhost1.novalocal\_8888,null-ns1-fys1.cmss.com\_8888]*

数据存储内容格式如下：



1. Mount Table，目录和子集群的映射关系，类似于ViewFS

*hdfs://tmp → hdfs://C0-1/tmp //目录tmp映射为c0-1集群中的tmp目录*

*hdfs://share → hdfs://C0-2/share*

*hdfs://user/user1 → hdfs://C0-3/user/user1*

*hdfs://user/user2 → hdfs://C0-2/user2*

子集群之间可以进行负载均衡，在负载均衡过程中需要对Mount Table添加lock，可以允许读但是不能进行写操作。在ZK中，Mount Table的Entry如下：

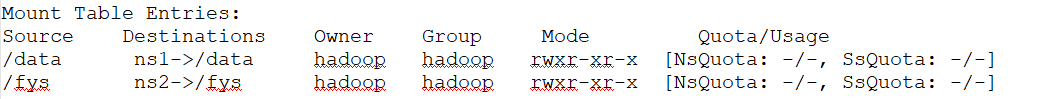
*[zk: localhost:2181(CONNECTED) 4] ls /hdfs-federation/MountTable*

*[0SLASH0data, 0SLASH0fys]*

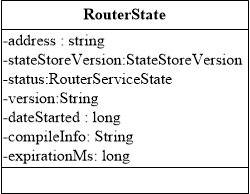
存储的数据内容格式如下：



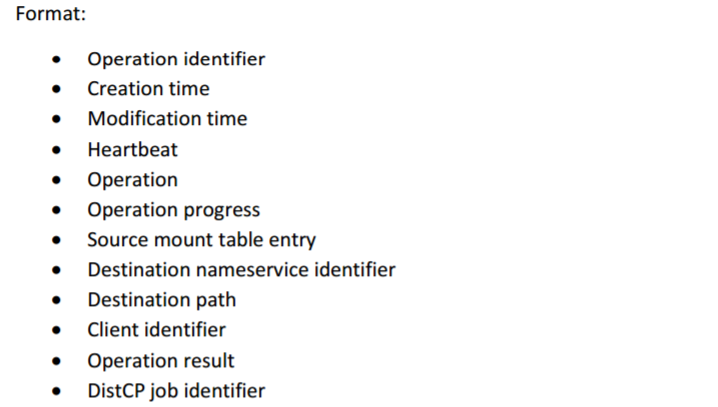
通过命令行查询如下：



1. Router State，获取Routers缓存中的状态，从而安全的进行rebalance，Router存放State Store中cached tables的版本，其数据格式如下：



1. Subcluster Rebalancer Log，为了对rebalance操作容错，将Rebalance操作存储到State Store中，Rebalance Entry的数据格式如下：



目前还未支持，需要进一步的调研进展情况，同时还需要调研Delegation tokens。

### **Implementations**

在Hadoop中提供多种State Store，是可插拔的，其配置参数：

*<property>  
 <name>dfs.federation.router.store.driver.class</name>  
 <value>*

*org.apache.hadoop.hdfs.server.federation.store.driver.impl.StateStoreZooKeeperImpl*

*</value>  
 <description>  
 \*\*\* StateStoreFileImpl，可以通过网络共享的文件，用于开发  
 \*\*\* StateStoreFileSystemImpl ，使用HDFS，用于单元测试*

*\*\*\* SQL，YARN Federation -JDBC，支持多种实现Mysql,SQLServer*

*\*\*\* StateStoreZooKeeperImpl.，默认使用Zookeeper  
 </description>  
</property>*

目前有三种实现StateStoreFileImpl、StateStoreFileSystemImpl和StateStoreZookeeperImpl，默认情况下使用Zookeeper来存储State Store信息。



### **Router Heartbeat**

Router周期性的更新StateStore中的状态，执行如下：

*synchronized void updateStateStore() {*

*String routerId = router.getRouterId();*

*....*

*if (isStoreAvailable()) {*

*RouterStore routerStore = router.getRouterStateManager();*

*try {*

*RouterState record = RouterState.newInstance(*

*routerId, router.getStartTime(), router.getRouterState());*

*StateStoreVersion stateStoreVersion = StateStoreVersion.newInstance(*

*getStateStoreVersion(MembershipStore.class),*

*getStateStoreVersion(MountTableStore.class));*

*record.setStateStoreVersion(stateStoreVersion);*

*RouterHeartbeatRequest request =*

*RouterHeartbeatRequest.newInstance(record);*

*RouterHeartbeatResponse response = routerStore.routerHeartbeat(request);*

*.....*

*}*

## Federated interface

Router接收到客户端的请求后，首先从State Store中获取访问请求相应的Subcluster，然后将请求转发给Subcluster的active NameNode，NameNode通过Router向客户端响应。

为了提高性能，可以启动多个Router（无状态），通过负载均衡器实现均衡。Router在内存中缓存Mount table及SubClusters的状态，并通过心跳更新State Store中Router的状态。

下面是Router的minamal配置：

*<property>*

*<name>dfs.nameservices</name>*

*<value>ns1,ns2,ns-fed</value>*

*</property>*

*<property>*

*<name>dfs.ha.namenodes.ns-fed</name>*

*<value>r1,r2</value>*

*</property>*

*<property>*

*<name>dfs.namenode.rpc-address.ns-fed.r1</name>*

*<value>fys1.cmss.com:8888</value>*

*</property>*

*<property>*

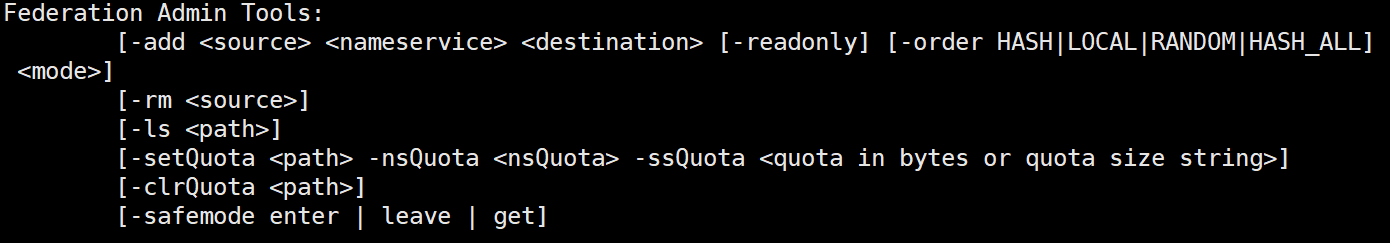
*<name>dfs.namenode.rpc-address.ns-fed.r2</name>*

*<value>cmhhost1.novalocal:8888</value>*

*</property>*

### **2.2.1 Mount Table的挂载**

Router启动后，通过dfsRouterAdmin挂载目录，执行命令的格式如下：



挂载目录的执行命令如下：

*$ HADOOP\_HOME/bin/hdfs dfsrouteradmin -add /data/app1 ns2 /data/app1*

执行序列图如下所示：



RouterAdminServer接收到客户端请求后，调用StateStoreService中的MountTableStore将挂载点写入到StateStore中。

### 2.2.2 文件查询流程

客户端发送请求：

*bin/hdfs dfs -ls hdfs://ns-fed*

请求中指定全局命名空间ns-fed，将请求转发到r1或者r2，RouterRpcServer接收到客户端请求，其实现接口ClientProtocol，同时将请求转发到NamenodeRpcServer，其类图如下所示：



获取文件信息的步骤如下：



1. 客户端向RouterRpcServer发送文件查询请求，协议接口ClientProtocol
2. RouterRpcServer接收到请求后，通过FileSubclusterResolver从StateStore中获取文件所属子集群，数据保存在MountTable
3. 调用ActieNameNodeResolver，从StateStore Membership中子集群的Active NameNode
4. RouterRpcClient与上述步骤获取的Active NameNode交互，获取文件信息HdfsFileStatus
5. RouterRpcServer将文件信息返回给DFSClient

## **NameNode Heartbeat**

Router通过心跳机制从NameNode（一般部署本主机）中获取统计信息，包括HA状态、负载信息等，并保存到StateStore中。执行类为NamenodeHeartbeatService，其类图如下所示：



核心方法为NamenodeHeartbeatService#updateState

*private void updateState() {*

*NamenodeStatusReport report = getNamenodeStatusReport();*

*....*

*try {*

*if (!resolver.registerNamenode(report)) {*

*.....*

*}*

通过MembershipNameNodeResolver#registerNameNode写入到State Store中

MembershipStoreImpl#namenodeHeartbeat，其核心代码如下

*@Override*

*public NamenodeHeartbeatResponse namenodeHeartbeat(*

*NamenodeHeartbeatRequest request) throws IOException {*

*MembershipState record = request.getNamenodeMembership();*

*......*

*boolean status = getDriver().put(record, true, false);*

*......*

*}*

为了解决操作，可以将本Service嵌入到Namenode上。

# Subcluster Rebalancer

Federation Layer可以基于Mount table进行集群之间的负载均衡，自动更新mount table，子集群的负载包括：

* RPC请求量
* 子集群的数据量
* Mount Table中的Entry数目

RBF提供Subcluster Rebalancer工具来均衡子集群之间的负载，使用Distcp进行子集群之间数据的迁移，并更新State Store。

1. Distcp，Rebalancer使用Distcp进行子集群之间的数据移动，Distcp支持目录中部分文件的复制，Relalancer会监控其进度和异常
2. Rebalance时，进行数据的更新，为了支持该操作增加文件设置Precondition及Mount table Entry lock机制
3. Fault tolerance，为了容错，在State Store中存放操作的状态
4. Limits，设置移动数据的大小和文件数目
5. Rebalance操作的步骤

* 在Reblancer日志中保存Mount Point
* 检查mount point文件树下是否最近被修改
* Distcp进行数据迁移
* Distcp是否成功
* 在目标NameService路径下创建新的Mount point
* Mount Point加锁
* 所有Router更新Mount point
* 验证原Mount Point的目录是否有文件
* 删除old service path下的文件
* 解锁

Route-based HDFS Federation:

https://issues.apache.org/jira/browse/HDFS-10467

HDFS RBF的全局配额管理方案：

https://issues.apache.org/jira/browse/HDFS-12934

https://blog.csdn.net/androidlushangderen/article/details/78942285