HBase报告

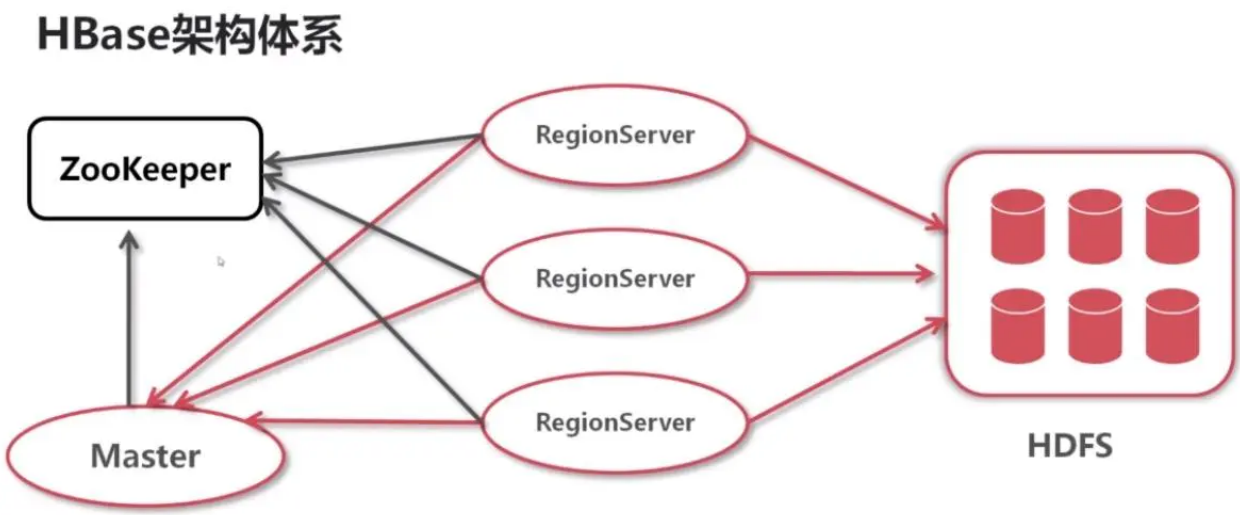
一. 原理介绍

- 介绍

  HBase是一个分布式的、面向列的开源数据库，HBase在Hadoop之上提供了类似于Bigtable的能力。HBase是Apache的Hadoop项目的子项目。HBase不同于一般的关系数据库，它是一个适合于非结构化数据存储的数据库。另一个不同的是HBase基于列的而不是基于行的模式。

- 关系型数据库的缺陷
  - 无法支持列可变
  - 不支持分布式事务
  - 对于长文本、音频等二进制数据性能差
- 特性
  - 强读写一致，但是不是“最终一致性”的数据存储，这使得它非常适合高速的计算聚合
  - 自动分片，通过Region分散在集群中，当行数增长的时候，Region也会自动的切分和再分配
  - 自动的故障转移
  - Hadoop/HDFS集成，和HDFS开箱即用，不用太麻烦的衔接
  - 丰富的“简洁，高效”API，Thrift/REST API，Java API
  - 块缓存，布隆过滤器，可以高效的列查询优化
  - 操作管理，Hbase提供了内置的web界面来操作，还可以监控JMX指标
- 缺陷
  - 不适合处理数据量少的数据集
  - 不支持RDBMS中的辅助索引、静态类型的列、事务等特性
  - 硬件资源需求大

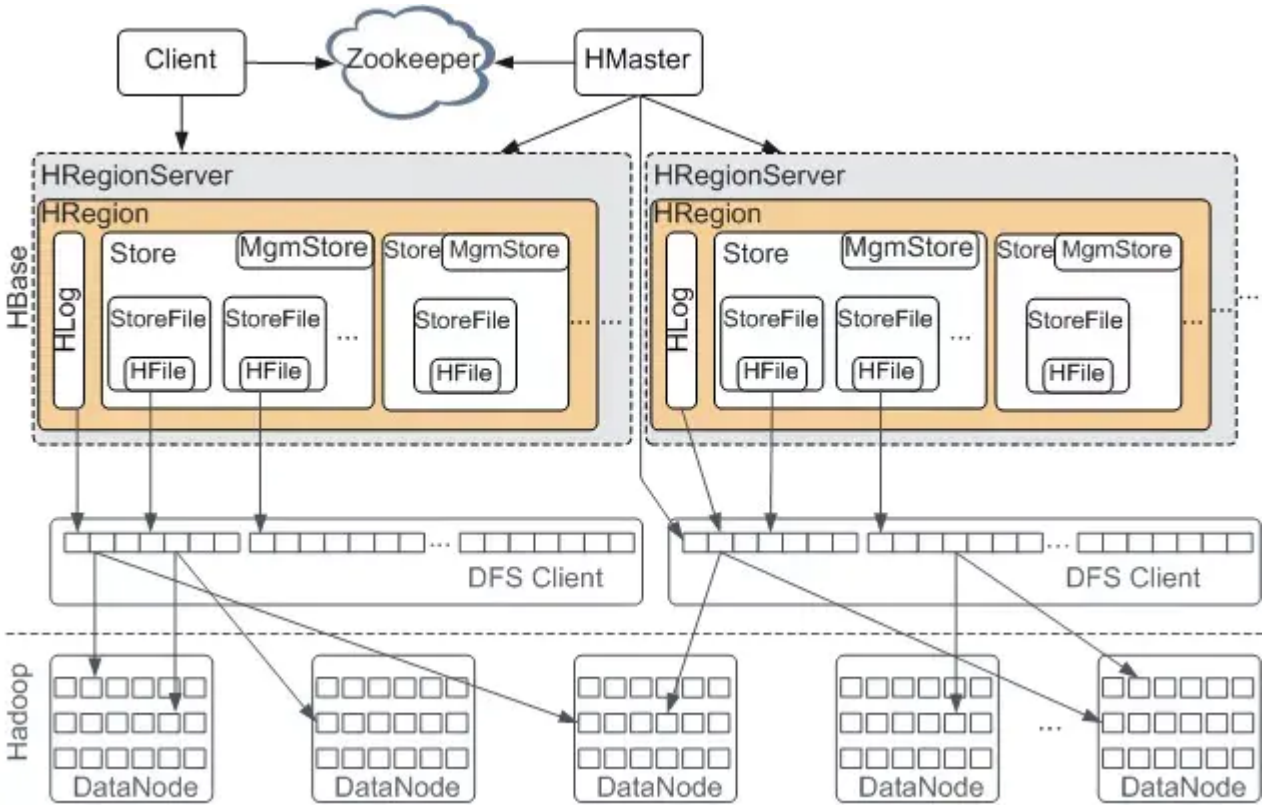
二. 架构详解

- 

  - `Zookeeper`：作为分布式的协调。`RegionServer`也会把自己的信息写到`ZooKeeper`中。

- `HDFS`：Hbase运行的底层文件系统
- `RegionServer`：理解为数据节点，存储数据的。
- `Master RegionServer`：实时的向Master报告信息。Master知道全局的RegionServer运行情况，可以控制RegionServer的故障转移和Region的切分。

- **架构细化**



- `HMaster`：`Master Server`的实现，负责监控集群中的`RegionServer`实例
- `HRegionServer`：是`RegionServer`的实现，服务和管理`Regions`，集群中`RegionServer`运行在`DataNode`
- `Regions`：代表`table`，有多个`Store`(列簇)，`Store`有一个`Memstore`和多个`StoreFiles(HFiles)`，`StoreFiles`的底层是`Block`
- `Zookeeper`：作为`Master`的HA解决方案

- **存储设计**

在`Hbase`中，表被分割成多个更小的块然后分散的存储在不同的服务器上，这些小块叫做`Regions`，存放`Regions`的地方叫做`RegionServer`。`Master`进程负责处理不同的`RegionServer`之间的`Region`的分发。在`Hbase`实现中`HRegionServer`和`HRegion`类代表`RegionServer`和`Region`。`HRegionServer`除了包含一些`HRegions`之外，还处理两种类型的文件用于数据存储

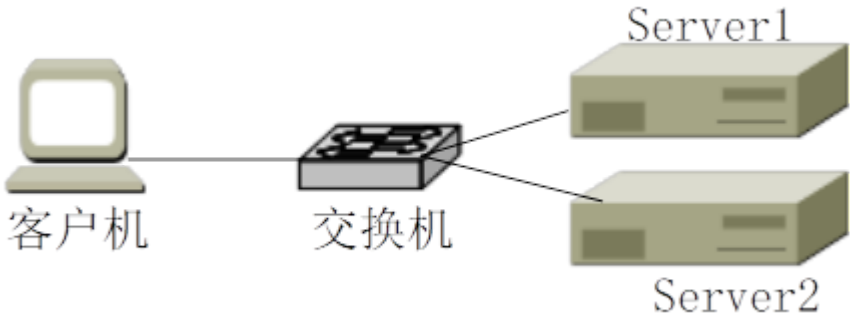
  - HLog： 预写日志文件，也叫做WAL(write-ahead log)
  - HFile： 真实的数据存储文件

- 占用端口的部分情况
  - zookeeper
    - 2181 ---- 对客户端提供服务
    - 2888 ---- 集群内通信，只有`leader`监听
    - 3888 ---- 选举`leader`使用
  - HBase
    - 16010 ---- 集群监控

- HDFS
  - 50070 ---- 集群监控

三. 实验环境及内容

- 环境



- 任务
  - 熟悉HBase工作原理
  - 在master上安装HBase中的Master、Zookeeper、RegionServer
  - 在node1上安装RegionServer

四.基础配置

配置	server1	server2
hostname	node1	master
root_passswd	toor	toor
group_name	hadoop	hadoop
user_name	hdfs	hdfs
user_passwd	toor	toor
ip	192.168.0.101	192.168.0.102

五. 环境搭建

1. 关闭防火墙
   - 临时关闭

     ```
     systemctl stop firewalld
     ```

   - 禁止开机启动

```
systemctl disable firewalld
```

2. 修改`selinux`属性(需重启)

```
vim /etc/selinux/config
SELINUX=disabled
```

3. 安装辅助软件
   - Net-tool（网络工具)

     ```
     apt-get install -y net-tool.x86_64
     ```

   - Lrzsz(文件传输)

     ```
     apt-get install -y lrzsz.x86_64
     ```

   - ssh服务

     ```
     apt-get install openssh-server
     ```

4. 安装集群软件
   - Jdk1.8
     - 解压

       ```
       tar -zxvf jdk-1.8.0.tar.gz
       ```

     - 修改环境变量（hdfs用户)

       ```
       vim ~/.bashrc

       export JAVA_HOME=/usr/local/jdk1.8
       export CLASSPATH=.:$JAVA_HOME/lib/dt.jar:$JAVA_HOME/lib/tools.jar
       export PATH=$PATH:$JAVA_HOME/bin
       ```

     - 生效

       ```
       source ~/.bashrc
       ```

     - 验证安装是否成功

       ```
       Java -version
       ```

     - 发送jdk到子节点

```
MASTER : scp -r /usr/local/jdk1.8.0 hdfs@node1:~
NODE1 : sudo mv /usr/local/jdk1.8.0
```

    - 照同样方式修改子节点的`~/.bashrc`文件

- Hadoop2.7.5

```
tar -zxvf hadoop-2.7.5.tar.gz
```

- Zookeeper3.4.13

```
tar -zxvf zookeeper-3.4.14.tar.gz
```

5. 安装Hadoop集群准备工作
    - 修改主机名（root）

```
vi /etc/hostname
#修改主机名(master/node1)
reboot
```

    - ssh免密登录(使用hdfs用户)
        - 添加公匙到公匙证书列表中

```
ssh-keygen -t rsa -P '' -f ~/.ssh/id_rsa
```

        - 在子节点新建.ssh文件夹

```
mkdir ~/.ssh
```

        - 将master公匙证书添加到其他节点中

```
scp ~/.ssh/authorized_keys hdfs@node1:~/.ssh/
```

    - 配置/etc/hosts

```
192.168.0.102 master
192.168.0.101 node1
```

6. 在master上配置hadoop2.7.5
    - 修改配置文件
        - core-site.xml

```
vim /usr/local/hadoop/hadoop-2.7.5/etc/hadoop/core-site.xml
```

```
<configuration>
    <property>
        <name>hadoop.tmp.dir</name>
        <value>/usr/local/hadoop/hadoop-2.7.5/tmp</value>
    </property>
    <property>
        <name>fs.defaultFS</name>
        <value>hdfs://master:9000</value>
    </property>
    <property>
        <name>io.file.buffer.size</name>
        <value>4096</value>
    </property>
</configuration>
```

- hdfs-site.xml

```
vim /usr/local/hadoop/hadoop-2.7.5/etc/hadoop/hdfs-site.xml
```

```
<configuration>
    <property>
        <name>dfs.nameservices</name>
        <value>master-hadoop-cluster</value>
    </property>

    <property>
        <name>dfs.namenode.secondary.http-address</name>
        <value>master:50090</value>
    </property>

    <property>
        <name>dfs.namenode.name.dir</name>
        <value>file:///usr/local/hadoop/hadoop-2.7.5/dfs/name</value>
    </property>

    <property>
        <name>dfs.datanode.data.dir</name>
        <value>file:///usr/local/hadoop/hadoop-2.7.5/dfs/data</value>
    </property>

    <property>
        <name>dfs.replication</name>
        <value>3</value>
    </property>

    <property>
        <name>dfs.webhdfs.enabled</name>
        <value>true</value>
    </property>
</configuration>
```

- mapred-site.xml

```
cp /usr/local/hadoop/hadoop-2.7.5/etc/hadoop/mapred-site.xml.template
/usr/local/hadoop/hadoop-2.7.5/etc/hadoop/mapred-site.xml
```

```
vim /usr/local/hadoop/hadoop-2.7.5/etc/hadoop/mapred-site.xml
```

```
<configuration>
    <property>
        <name>mapreduce.framework.name</name>
        <value>yarn</value> </property>
    <property>
        <name>mapreduce.jobtracker.http.address</name>
        <value>master:50030</value> </property>
    <property>
        <name>mapreduce.jobhistory.address</name>
        <value>master:10020</value>
    </property>
    <property>
        <name>mapreduce.jobhistory.webapp.address</name>
        <value>master:19888</value>
    </property>
</configuration>
```

- yarn-site.xml

```
vim /usr/local/hadoop/hadoop-2.7.5/etc/hadoop/yarn-site.xml
```

```
<configuration>
    <property>
        <name>yarn.nodemanager.aux-services</name>
        <value>mapreduce_shuffle</value>
    </property>

    <property>
        <name>yarn.resourcemanager.address</name>
        <value>master:8032</value>
    </property>

    <property>
        <name>yarn.resourcemanager.scheduler.address</name>
        <value>master:8030</value>
    </property>

    <property>
        <name>yarn.resourcemanager.resource-tracker.address</name>
        <value>master:8031</value>
    </property>

    <property>
```

```
        <name>yarn.resourcemanager.admin.address</name>
        <value>master:8033</value>
    </property>

    <property>
        <name>yarn.resourcemanager.webapp.address</name>
        <value>master:8088</value>
    </property>
<configuration>
```

- 修改slaves文件

```
/usr/local/hadoop/hadoop-2.7.5/etc/hadoop/slaves
```

```
node1
```

- 修改相应文件中的JAVA_HOME

```
vim /usr/local/hadoop/hadoop-2.7.5/etc/hadoop/hadoop-env.sh
```

```
export JAVA_HOME=/usr/local/jdk1.8.0
```

```
vim /usr/local/hadoop/hadoop-2.7.5/etc/hadoop/yarn-env.sh
```

```
export JAVA_HOME=/usr/local/jdk1.8.0
```

- 分享配置给server2

```
scp -r /hadoop/ server2:/
```

- 分享hadoop给node1

```
scp -r /usr/local/hadoop/hadoop-2.7.5/ hdfs@node1:/usr/local/hadoop/
scp -r /usr/local/hadoop/hadoop-2.7.5/ hdfs@node2:/usr/local/hadoop/
```

- 分别在master和node1配置环境变量

```
vim ~/.bashrc
```

```
export JAVA_HOME=/opt/SoftWare/jdk1.8.0
export CLASSPATH=.:$JAVA_HOME/lib/dt.jar:/
export HADOOP_HOME=/opt/SoftWare/hadoop/hadoop-2.7.5
export PATH=$PATH:$JAVA_HOME/bin:$HADOOP_HOME/bin:$HADOOP_HOME/sbin
```

- 重启

```
source ~/.bashrc
```

- 格式化HDFS

```
hdfs namenode -format
```

- 启动

```
./start-all.sh
```

7. 配置zookeeper集群
   - 修改配置文件
     - `/zookeeper/conf` 目录下

       ```
       cp zoo_sample.cfg zoo.cfg
       ```

       修改

       ```
       server.1=master:2888:3888
       server.2=node1:2888:3888
       dataLogDir=/opt/SoftWare/zookeeper-3.4.14/logs
       dataDir=/opt/SoftWare/zookeeper-3.4.14/dataDir
       ```

     - 创建文件夹

       ```
       mkdir -p /usr/local/hadoop/zookeeper-3.4.10/dataDir
       mkdir -p /usr/local/hadoop/zookeeper-3.4.10/logs
       ```

     - 新建`myid`文件

       ```
       vim /usr/local/hadoop/zookeeper-3.4.14/dataDir/myid
       ```

   - 分享至node1

     ```
     scp -r zookeeper-3.4.14/ hdfs@node1:/usr/local/hadoop/
     ```

   - 修改myid文件内容，在master节点中修改内容为 1 ，在node1中为 2
   - 配置环境变量（hdfs用户）

     ```
     vim ~/.bashrc
     ```

```
export JAVA_HOME=/opt/SoftWare/jdk1.8.0
export CLASSPATH=.:$JAVA_HOME/lib/dt.jar:/ $JAVA_HOME/lib/tool.jar
export HADOOP_HOME=/opt/SoftWare/hadoop/hadoop-2.7.5
export ZOOKEEPER_HOME=/opt/SoftWare/zookeeper-3.4.14
PATH=$PATH:$JAVA_HOME/bin:$HADOOP_HOME/bin:$HADOOP_HOME/sbin:$ZOOKEEPER_HOME/bin
```

```
source ~/.bashrc
```

- 启动zookeeper（各节点分别执行）

```
./zkServer.sh start  #启动
```

- 查看状态

```
zkServer.sh status
```

8. 安装Hbase
   - 解压

```
tar -zxvf hbase-1.2.6.tar.gz
```

   - 在终端输入以下命令，新建目录。

```
mkdir -p /var/local/hadoop/hbase-1.2.6/dataDir
mkdir -p /var/local/hadoop/hbase-1.2.6/logs
```

   - 配置文件
     - `hbase-1.2.6/conf/hbase-env.sh`

```
export HBASE_MANAGES_ZK=false
export HBASE_LOG_DIR=/usr/local/hadoop/hbase-1.2.6/logs
export JAVA_HOME=/usr/local/jdk1.8.0
export HADOOP_HOME=/usr/local/hadoop/hadoop-2.7.5
export HBASE_HOME=/usr/local/hadoop/hbase-1.2.6
export HBASE_CLASSPATH=/usr/local/hadoop/hadoop-2.7.5/etc/hadoop
```

     - `hbase-1.2.6/conf/hbase-site.xml`

```
!vim  $HBASE_HOME/conf/hbase-site.xml
<property>
    <name>hbase.rootdir</name>
    <value>hdfs://master:9000/hbase</value>
</property>

<property>
    <name>hbase.cluster.distributed</name>
    <value>true</value>
```

```
</property>

<property>
    <name>hbase.zookeeper.property.clientPort</name>
    <value>2181</value>
</property>

<property>
    <name>hbase.zookeeper.quorum</name>
    <value>master,node1</value>
</property>

<property>
    <name>hbase.zookeeper.property.dataDir</name>
    <value>/opt/SoftWare/zookeeper-3.4.14/dataDir</value>
</property>

<!-- HMaster.... -->
<property>
    <name>hbase.master.info.bindAddress</name>
    <value>master</value>
    <description>HBase Master Web UI0.0.0.0</description>
</property>

<property>
    <name>hbase.master.info.port</name>
    <value>16010</value>
    <description>HBase Master Web UIWeb UI</description>
</property>
<!-- HRegionServer.... -->
<property>
    <name>hbase.regionserver.port</name><value>16020</value>
    <description>The port the HBase RegionServer binds to.</description>
</property>

<property>
    <name>hbase.regionserver.info.port</name>
    <value>16030</value>
    <description>The port for the HBase RegionServer web UI</description>
</property>

<property>
    <name>hbase.regionserver.info.bindAddress</name>
    <!--下面这项需要在node1节点和node2节点上改为node1和node2 -->
    <value>master</value>
    <description>The address for the HBase RegionServer web UI</description>
</property>
```

- `hbase-1.2.6/conf/regionservers`

```
node1
```

- 同步配置

```
scp -r /var/local/hadoop/hbase-1.1.7/ hdfs@node1:/var/local/hadoop
```

- 在node1节点中修改hbase配置文件"hbase-site.xml"中的最后一个"property"标签
- 配置各节点环境变量

```
export JAVA_HOME=/opt/SoftWare/jdk1.8.0
export CLASSPATH=.:$JAVA_HOME/lib/dt.jar:/ $JAVA_HOME/lib/tool.jar
export HADOOP_HOME=/opt/SoftWare/hadoop/hadoop-2.7.5
export HBASE_HOME=/opt/SoftWare/hadoop/hbase-1.2.6
export ZOOKEEPER_HOME=/opt/SoftWare/zookeeper-3.4.14
PATH=$PATH:$JAVA_HOME/bin:$HADOOP_HOME/bin:$HADOOP_HOME/sbin:$ZOOKEEPER_HOME/
bin: $HBASE_HOME/bin
```

- 启动hbase

  在master节点，使用hdfs用户，在终端输入"start-hbase", 然后在各个节点输入jps查看进程，可以在master节点上看到HMaster进程，在两个子节点上看到HRegionServer进程。

```
cd $HBASE_HOME/bin
./start-hbase.sh

hbase shell #查看运行状态
status
```

五. 参考资料

- https://www.jianshu.com/p/b23800d9b227
- 搭建hbase https://www.cnblogs.com/huhongy/p/10953647.html
- https://blog.csdn.net/zyy_2018/article/details/79576252
- https://github.com/stumbleupon/asynchbase