# 安装（java、hadoop）

## 创建ahdoop用户

## java设置

rpm -ivh jdk-7u67-linux-x64.rpm

alternatives --install /usr/bin/java java /usr/java/jdk1.7.0\_67/jre/bin/java 100

alternatives --config java

|  |
| --- |
| vi /etc/profile  export JAVA\_HOME=/usr/java/jdk1.7.0\_67  export PATH=$JAVA\_HOME/bin:$PATH  source /etc/profile  echo $JAVA\_HOME |

## 配置hadoop

|  |
| --- |
| tar -xzvf /home/hadoop/hadoop-2.7.0.tar.gz -C /etc  vi /etc/profile  export HADOOP\_HOME=/etc/hadoop-2.7.0  export PATH=$HADOOP\_HOME/bin:$PATH  source /etc/profile  echo $PATH  echo $HADOOP\_HOME |

* 验证是否安装成功

hadoop version

## ssh免密码登录配置（可选）

|  |
| --- |
| #生成密钥  ssh-keygen -t rsa -P ""  #密钥保存路径  Enter file in which to save the key (/root/.ssh/id\_rsa): /home/hadoop/.ssh/id\_rsa  #查看密钥生成情况  ls -al /home/hadoop/.ssh/  #生成两个文件id\_rsa id\_rsa.pub  #增加公钥到authorized\_keys  touch /home/hadoop/.ssh/authorized\_keys  cat /home/hadoop/.ssh/id\_rsa.pub >> /home/hadoop/.ssh/authorized\_keys |

## 防火墙设置

vi /etc/sysconfig/iptables

service iptables restart

### HDFS

|  |
| --- |
| -A INPUT -p tcp -m tcp --dport 50010 -j ACCEPT  -A INPUT -p tcp -m tcp --dport 50075 -j ACCEPT  -A INPUT -p tcp -m tcp --dport 50475 -j ACCEPT  -A INPUT -p tcp -m tcp --dport 50020 -j ACCEPT  -A INPUT -p tcp -m tcp --dport 50070 -j ACCEPT  -A INPUT -p tcp -m tcp --dport 50470 -j ACCEPT  -A INPUT -p tcp -m tcp --dport 8020 -j ACCEPT  -A INPUT -p tcp -m tcp --dport 8485 -j ACCEPT  -A INPUT -p tcp -m tcp --dport 8480 -j ACCEPT  -A INPUT -p tcp -m tcp --dport 8019 -j ACCEPT  -A INPUT -p tcp -m tcp --dport 9000 -j ACCEPT  -A OUTPUT -p tcp -m tcp --dport 50010 -j ACCEPT  -A OUTPUT -p tcp -m tcp --dport 50075 -j ACCEPT  -A OUTPUT -p tcp -m tcp --dport 50475 -j ACCEPT  -A OUTPUT -p tcp -m tcp --dport 50020 -j ACCEPT  -A OUTPUT -p tcp -m tcp --dport 50070 -j ACCEPT  -A OUTPUT -p tcp -m tcp --dport 50470 -j ACCEPT  -A OUTPUT -p tcp -m tcp --dport 8020 -j ACCEPT  -A OUTPUT -p tcp -m tcp --dport 8485 -j ACCEPT  -A OUTPUT -p tcp -m tcp --dport 8480 -j ACCEPT  -A OUTPUT -p tcp -m tcp --dport 8019 -j ACCEPT  -A OUTPUT -p tcp -m tcp --dport 9000 -j ACCEPT |

### YARN

|  |
| --- |
| -A INPUT -p tcp -m tcp --dport 8032 -j ACCEPT  -A INPUT -p tcp -m tcp --dport 8030 -j ACCEPT  -A INPUT -p tcp -m tcp --dport 8031 -j ACCEPT  -A INPUT -p tcp -m tcp --dport 8033 -j ACCEPT  -A INPUT -p tcp -m tcp --dport 8088 -j ACCEPT  -A INPUT -p tcp -m tcp --dport 8040 -j ACCEPT  -A INPUT -p tcp -m tcp --dport 8042 -j ACCEPT  -A INPUT -p tcp -m tcp --dport 8041 -j ACCEPT  -A INPUT -p tcp -m tcp --dport 10020 -j ACCEPT  -A INPUT -p tcp -m tcp --dport 19888 -j ACCEPT  -A OUTPUT -p tcp -m tcp --dport 8032 -j ACCEPT  -A OUTPUT -p tcp -m tcp --dport 8030 -j ACCEPT  -A OUTPUT -p tcp -m tcp --dport 8031 -j ACCEPT  -A OUTPUT -p tcp -m tcp --dport 8033 -j ACCEPT  -A OUTPUT -p tcp -m tcp --dport 8088 -j ACCEPT  -A OUTPUT -p tcp -m tcp --dport 8040 -j ACCEPT  -A OUTPUT -p tcp -m tcp --dport 8042 -j ACCEPT  -A OUTPUT -p tcp -m tcp --dport 8041 -j ACCEPT  -A OUTPUT -p tcp -m tcp --dport 10020 -j ACCEPT  -A OUTPUT -p tcp -m tcp --dport 19888 -j ACCEPT |

### HBase

|  |
| --- |
| -A INPUT -p tcp -m tcp --dport 60000 -j ACCEPT  -A INPUT -p tcp -m tcp --dport 60010 -j ACCEPT  -A INPUT -p tcp -m tcp --dport 60020 -j ACCEPT  -A INPUT -p tcp -m tcp --dport 60030 -j ACCEPT  -A INPUT -p tcp -m tcp --dport 2181 -j ACCEPT  -A INPUT -p tcp -m tcp --dport 2888 -j ACCEPT  -A INPUT -p tcp -m tcp --dport 3888 -j ACCEPT  -A OUTPUT -p tcp -m tcp --dport 60000 -j ACCEPT  -A OUTPUT -p tcp -m tcp --dport 60010 -j ACCEPT  -A OUTPUT -p tcp -m tcp --dport 60020 -j ACCEPT  -A OUTPUT -p tcp -m tcp --dport 60030 -j ACCEPT  -A OUTPUT -p tcp -m tcp --dport 2181 -j ACCEPT  -A OUTPUT -p tcp -m tcp --dport 2888 -j ACCEPT  -A OUTPUT -p tcp -m tcp --dport 3888 -j ACCEPT |

### Hive

|  |
| --- |
| -A INPUT -p tcp -m tcp --dport 9083 -j ACCEPT  -A INPUT -p tcp -m tcp --dport 10000 -j ACCEPT  -A OUTPUT -p tcp -m tcp --dport 9083 -j ACCEPT  -A OUTPUT -p tcp -m tcp --dport 10000 -j ACCEPT |

### ZooKeeper

|  |
| --- |
| -A INPUT -p tcp -m tcp --dport 2181 -j ACCEPT  -A INPUT -p tcp -m tcp --dport 2888 -j ACCEPT  -A INPUT -p tcp -m tcp --dport 3888 -j ACCEPT  -A OUTPUT -p tcp -m tcp --dport 2181 -j ACCEPT  -A OUTPUT -p tcp -m tcp --dport 2888 -j ACCEPT  -A OUTPUT -p tcp -m tcp --dport 3888 -j ACCEPT |

# 伪分布设置

## 创建目录

|  |
| --- |
| mkdir $HADOOP\_HOME/tmp  mkdir $HADOOP\_HOME/hdfs  mkdir $HADOOP\_HOME/hdfs/name  mkdir $HADOOP\_HOME/hdfs/data  mkdir $HADOOP\_HOME/mapred  mkdir $HADOOP\_HOME/mapred/local  mkdir $HADOOP\_HOME/mapred/system  chown -R hadoop:hadoop $HADOOP\_HOME |

## etc/hadoop/core-site.xml(9000)

vi $HADOOP\_HOME/etc/hadoop/core-site.xml

|  |
| --- |
| <configuration>  <property>  <name>fs.default.name</name>  <value>hdfs://localhost:9000</value>  </property>  <property>  <name>hadoop.tmp.dir</name>  <value>/lzg/app/hadoop-2.7.0/tmp</value>  </property>  </configuration> |

## etc/hadoop/hdfs-site.xml

vi $HADOOP\_HOME/etc/hadoop/hdfs-site.xml

|  |
| --- |
| <configuration>  <property>  <name>dfs.name.dir</name>  <value>/lzg/app/hadoop-2.7.0/hdfs/name</value>  <description>namenode存储路径</description>  </property>  <property>  <name>dfs.data.dir</name>  <value>/lzg/app/hadoop-2.7.0/hdfs/data</value>  <description>datanode存储路径</description>  </property>  <property>  <name>dfs.replication</name>  <value>1</value>  </property>  </configuration> |

## etc/hadoop/mapred-site.xml(9001)

vi $HADOOP\_HOME/etc/hadoop/mapred-site.xml

|  |
| --- |
| <configuration>  <property>  <name>mapred.job.tracker</name>  <value>hdfs://localhost:9001</value>  </property>  <property>  <name>mapred.local.dir</name>  <value>/lzg/app/hadoop-2.7.0/mapred/local</value>  <description>存储mapred自己使用的路径</description>  </property>  <property>  <name>mapred.system.dir</name>  <value>/lzg/app/hadoop-2.7.0/mapred/system</value>  <description>存储mapred系统级别的路径，可以共享</description>  </property>  </configuration> |

## etc/hadoop/yarn-site.xml

vi $HADOOP\_HOME/etc/hadoop/yarn-site.xml

|  |
| --- |
| <configuration>  <property>  <name>yarn.nodemanager.aux-services</name>  <value>mapreduce\_shuffle</value>  </property>  </configuration> |

## hadoop里增加java\_home

vi /HADOOP\_HOME/etc/hadoop/hadoop-env.sh

## 格式化

|  |
| --- |
| $HADOOP\_HOME/bin/hadoop namenode –format |

## 启动、关闭服务

|  |
| --- |
| sh - hadoop -C '$HADOOP\_HOME/sbin/start-all.sh'  sh $HADOOP\_HOME/sbin/stop-all.sh  su - hadoop -c '$HADOOP\_HOME/sbin/start-all.sh'  su - hadoop -c '$HADOOP\_HOME/sbin/stop-all.sh'  jps  #能看到5个线程  $HADOOP\_HOME/bin/hadoop dfsadmin -report |

# java接口

## 读数据

### 读数据流程图



### URL读取文件（需要设置UrlStreamHandlerFactory）

|  |
| --- |
| //使用java识别hdfs协议  URL.*setURLStreamHandlerFactory*(**new** FsUrlStreamHandlerFactory());  InputStream in = **null**;  **try** {  in = **new** URL("hdfs://192.168.1.149:9000/lzg/test2.txt").openStream();  **int** read = 0;  **byte** buffer[] = **new** **byte**[1024];  **while**((read=in.read(buffer))!=-1){  System.***out***.println(**new** String(buffer));  }  } **catch** (Exception e) {  e.printStackTrace();  **if**(in !=**null** ){  in.close();  }  } |

### IOUtils读取（需要设置UrlStreamHandlerFactory）

|  |
| --- |
| //使用java识别hdfs协议  URL.*setURLStreamHandlerFactory*(**new** FsUrlStreamHandlerFactory());  InputStream in = **null**;  **try** {  in = **new** URL("hdfs://192.168.1.149:9000/lzg/test2.txt").openStream();  IOUtils.*copyBytes*(in, System.***out***, 4096, **false**);  //FSDataInputStream可以移动读取位置  in.seek(0);  IOUtils.*copyBytes*(in, System.***out***, 4096, **false**);  } **catch** (Exception e) {  IOUtils.*closeStream*(in);  } |

### FileSystem api

|  |
| --- |
| Configuration config = **new** Configuration();  FileSystem fs = FileSystem.*get*(URI.*create*("hdfs://192.168.1.149:9000/lzg/test2.txt"), config);  InputStream in = **null**;  **try** {  in = fs.open(**new** Path("hdfs://192.168.1.149:9000/lzg/test2.txt"));  IOUtils.*copyBytes*(in, System.***out***, 4096, **false**);  //FSDataInputStream可以移动读取位置  //高开销操作，需要慎重使用，  in.seek(0);  IOUtils.*copyBytes*(in, System.***out***, 4096, **false**);  } **catch** (Exception e) {  IOUtils.*closeStream*(in);  } |

## 写数据

### 写数据流程图



### 创建目录

|  |
| --- |
| Configuration config = **new** Configuration();  FileSystem fs = FileSystem.*get*(URI.*create*("hdfs://"+*ip*+":9000/"), config);  **try** {  fs.mkdirs(**new** Path("hdfs://"+*ip*+":9000/lzg/test1"));  } **catch** (Exception e) {  e.printStackTrace();  } |

### 新加文件

|  |
| --- |
| Configuration config = **new** Configuration();  FileSystem fs = FileSystem.*get*(URI.*create*("hdfs://"+*ip*+":9000/lzg/test3.txt"), config);  FSDataOutputStream out = **null**;  FileInputStream in = **null**;  **try** {  in = **new** FileInputStream("d:\\log.txt");  out = fs.create(**new** Path("hdfs://"+*ip*+":9000/lzg/test3.txt"),**new** Progressable() {  @Override  **public** **void** progress() {  System.***out***.println(".........");  }  });  IOUtils.*copyBytes*(in, out, 4096, **true**);    } **catch** (Exception e) {  e.printStackTrace();  IOUtils.*closeStream*(in);  } |

### 查询文件

|  |
| --- |
| Configuration config = **new** Configuration();  FileSystem fs = FileSystem.*get*(URI.*create*("hdfs://"+*ip*+":9000/"), config);  **try** {  //判断文件是否存在  System.***out***.println(fs.exists(**new** Path("hdfs://"+*ip*+":9000/lzg")));  //获取文件的各种属性  FileStatus fileStatus = fs.getFileStatus(**new** Path("hdfs://"+*ip*+":9000/lzg"));  System.***out***.println(fileStatus);  // fileStatus{  // path=hdfs://192.168.1.251:9000/lzg;  // isDirectory=true;  // modification\_time=1436101193932;  // access\_time=0;  // owner=hadoop;  // group=supergroup;  // permission=rwxrwxrwx;  // isSymlink=false  // }    //获取目录下的所有文件  FileStatus[] fileStatusArray = fs.listStatus(**new** Path("hdfs://"+*ip*+":9000/lzg"));  **for**(FileStatus file : fileStatusArray){  System.***out***.println(file.getPath());  }    } **catch** (Exception e) {  e.printStackTrace();  } |

### 删除文件

|  |
| --- |
| Configuration config = **new** Configuration();  FileSystem fs = FileSystem.*get*(URI.*create*("hdfs://"+*ip*+":9000/"), config);  **try** {  fs.delete(**new** Path("hdfs://"+*ip*+":9000/lzg/test3.txt"), **true**);  } **catch** (Exception e) {  e.printStackTrace();  } |

FSDataOutputStream FileSystem.create(Path p);

FSDataOutputStream FileSystem.append(Path p);

# 常用命令

|  |  |  |
| --- | --- | --- |
| 命令 | 含义 | 备份 |
| hadoop fsck / -files -blocks | 文件系统中各个文件的块 |  |
| hadoop fs -help | 对文件操作的所有命令 |  |
|  |  |  |
|  |  |  |
|  |  |  |

# 文件操作命令

|  |  |  |
| --- | --- | --- |
| 命令 | 含义 | 备份 |
| hadoop fs -mkdir /lzg | 新建目录 |  |
| hadoop fs -rm /lzg/test2.txt | 删除文件 |  |
| hadoop fs -copyFromLocal /home/hadoop/test2.txt /lzg | 从本地复制到hdfs |  |
| hadoop fs -copyToLocal /lzg/test2.txt /home/hadoop/test3.txt | 从hdfs复制到本地 |  |
| hadoop fs -copyFromLocal /home/hadoop/test2.txt hdfs://localhost:9000/lzg/aa.txt | 从本地复制到hdfs |  |
|  |  |  |

# hadoop端口使用

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 组件 | 节点 | 默认端口 | 配置 | 用途说明 |
| HDFS | DataNode | 50010 | dfs.datanode.address | datanode服务端口，用于数据传输 |
| HDFS | DataNode | 50075 | dfs.datanode.http.address | http服务的端口 |
| HDFS | DataNode | 50475 | dfs.datanode.https.address | https服务的端口 |
| HDFS | DataNode | 50020 | dfs.datanode.ipc.address | ipc服务的端口 |
| HDFS | NameNode | 50070 | dfs.namenode.http-address | http服务的端口 |
| HDFS | NameNode | 50470 | dfs.namenode.https-address | https服务的端口 |
| HDFS | NameNode | 8020 | fs.defaultFS | 接收Client连接的RPC端口，用于获取文件系统metadata信息。 |
| HDFS | journalnode | 8485 | dfs.journalnode.rpc-address | RPC服务 |
| HDFS | journalnode | 8480 | dfs.journalnode.http-address | HTTP服务 |
| HDFS | ZKFC | 8019 | dfs.ha.zkfc.port | ZooKeeper FailoverController，用于NN HA |
| YARN | ResourceManager | 8032 | yarn.resourcemanager.address | RM的applications manager(ASM)端口 |
| YARN | ResourceManager | 8030 | yarn.resourcemanager.scheduler.address | scheduler组件的IPC端口 |
| YARN | ResourceManager | 8031 | yarn.resourcemanager.resource-tracker.address | IPC |
| YARN | ResourceManager | 8033 | yarn.resourcemanager.admin.address | IPC |
| YARN | ResourceManager | 8088 | yarn.resourcemanager.webapp.address | http服务端口 |
| YARN | NodeManager | 8040 | yarn.nodemanager.localizer.address | localizer IPC |
| YARN | NodeManager | 8042 | yarn.nodemanager.webapp.address | http服务端口 |
| YARN | NodeManager | 8041 | yarn.nodemanager.address | NM中container manager的端口 |
| YARN | JobHistory Server | 10020 | mapreduce.jobhistory.address | IPC |
| YARN | JobHistory Server | 19888 | mapreduce.jobhistory.webapp.address | http服务端口 |
| HBase | Master | 60000 | hbase.master.port | IPC |
| HBase | Master | 60010 | hbase.master.info.port | http服务端口 |
| HBase | RegionServer | 60020 | hbase.regionserver.port | IPC |
| HBase | RegionServer | 60030 | hbase.regionserver.info.port | http服务端口 |
| HBase | HQuorumPeer | 2181 | hbase.zookeeper.property.clientPort | HBase-managed ZK mode，使用独立的ZooKeeper集群则不会启用该端口。 |
| HBase | HQuorumPeer | 2888 | hbase.zookeeper.peerport | HBase-managed ZK mode，使用独立的ZooKeeper集群则不会启用该端口。 |
| HBase | HQuorumPeer | 3888 | hbase.zookeeper.leaderport | HBase-managed ZK mode，使用独立的ZooKeeper集群则不会启用该端口。 |
| Hive | Metastore | 9083 | /etc/default/hive-metastore中export PORT=<port>来更新默认端口 |  |
| Hive | HiveServer | 10000 | /etc/hive/conf/hive-env.sh中export HIVE\_SERVER2\_THRIFT\_PORT=<port>来更新默认端口 |  |
| ZooKeeper | Server | 2181 | /etc/zookeeper/conf/zoo.cfg中clientPort=<port> | 对客户端提供服务的端口 |
| ZooKeeper | Server | 2888 | /etc/zookeeper/conf/zoo.cfg中server.x=[hostname]:nnnnn[:nnnnn]，标蓝部分 | follower用来连接到leader，只在leader上监听该端口。 |
| ZooKeeper | Server | 3888 | /etc/zookeeper/conf/zoo.cfg中server.x=[hostname]:nnnnn[:nnnnn]，标蓝部分 | 用于leader选举的。只在electionAlg是1,2或3(默认)时需要。 |

# 访问地址

http://10.20.71.204:50070/

http://10.20.71.204:8088/ ResourceManager

http://10.20.71.204:8042/ NodeManager - Node

http://192.168.1.149:50070/