

Hadoop 平台部署总结

作者:	罗金凯
版本:	V1.1
日期:	2020.10.05

版本说明

版本	日期	说明
V1.0	2020.10.05	初创
V1.1	2022.8.18	修改全分布式章节内容，纠正部分错误

目录

1. Hadoop 平台部署方式简介	1
2. 平台及组件版本	1
3. Hadoop 伪分布式部署	1
3.1 root 用户部署	1
3.1.1 上传安装包	1
3.1.2 安装 java1.8	1
3.1.3 配置免密码登录	2
3.1.4 安装 hadoop2.7.4	2
3.1.5 hdfs 集群格式化	9
3.1.6 启动集群	9
3.2 Hadoop 用户部署	12
3.2.1 创建 hadoop 用户并修改权限	12
3.2.2 安装 java1.8	12
3.2.3 配置免密码登录	13
3.2.4 安装 Hadoop	13
3.2.5 创建目录	15
3.2.6 格式化 hdfs	15
3.2.7 启动集群	16
4. Hadoop 全分布式部署	17
4.1 hadoop 用户部署	17
4.1.1 服务器信息	17
4.1.2 上传安装包到 master 节点	17
4.1.3 创建 hadoop 用户并赋权	17
4.1.4 设置免密码登录	17
4.1.5 安装 Hadoop	19
4.1.6 集群管理系统	25
4.1.7 作业提交集群测试	26
4.2 root 用户部署	27
4.2.1 上传安装包	27
4.2.2 安装 Java	27
4.2.3 设置免密码登录	27
4.2.4 安装 Hadoop	28
5. Hadoop HA 部署	33
5.1 解压 Zookeeper 安装包	33
5.2 配置环境变量	33
5.3 修改 zookeeper 配置文件	33
5.4 部署 Hadoop	36

1. Hadoop 平台部署方式简介

常用的 Hadoop 平台主要有以下几种部署方式：单机模式、伪分布式模式、全分布式模式和高可用（HA）模式。

2. 平台及组件版本

组件	版本
Java	1.8
Hadoop	2.7.4
Yarn	2.6.0
Zookeeper	3.4.5
Hive	2.3.2
Flume	1.6.0
Sqoop	1.4
kafka	1.0
Spark	2.0
MySQL	5.5

3. Hadoop 伪分布式部署

3.1 root 用户部署

3.1.1 上传安装包

将 Hadoop 安装包以及 Java 安装包上传到/opt/bigdata 目录。

```
[root@localhost bigdata]# ll
total 447700
-rw-r--r--. 1 root root 266688029 Oct  5 00:22 hadoop-2.7.4.tar.gz
-rw-r--r--. 1 root root 191753373 Oct  5 00:23 jdk-8u191-linux-x64.tar.gz
[root@localhost bigdata]#
```

3.1.2 安装 java1.8

（1）解压安装包

```
[root@localhost bigdata]# tar -zxvf jdk-8u191-linux-x64.tar.gz
```

(2) 配置环境变量

- 打开当前用户环境变量配置文件

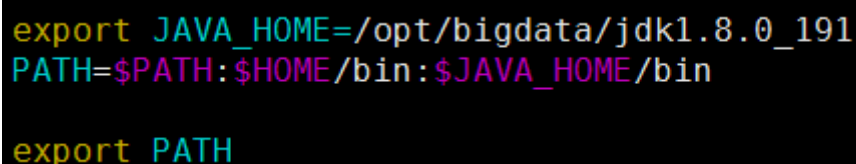
```
[root@localhost jdk1.8.0-191]# vim ~/.bash-profile
```

- 增加 java 的配置到 path

```
export JAVA_HOME=/opt/bigdata/jdk1.8.0-191
```

```
PATH=$PATH:$HOME/bin:$JAVA_HOME/bin
```

如下图所示:



```
export JAVA_HOME=/opt/bigdata/jdk1.8.0_191
PATH=$PATH:$HOME/bin:$JAVA_HOME/bin

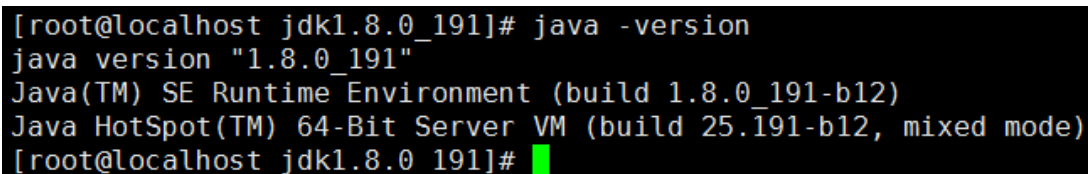
export PATH
```

- 配置生效

```
[root@localhost jdk1.8.0-191]# source ~/.bash-profile
```

(3) 验证 Java 是否安装成功

```
[root@localhost jdk1.8.0-191]# java -version
```



```
[root@localhost jdk1.8.0_191]# java -version
java version "1.8.0_191"
Java(TM) SE Runtime Environment (build 1.8.0_191-b12)
Java HotSpot(TM) 64-Bit Server VM (build 25.191-b12, mixed mode)
[root@localhost jdk1.8.0_191]#
```

3.1.3 配置免密码登录

若没有该目录, 请先执行一次 `ssh localhost`

```
[root@localhost sbin]# cd ~/.ssh/
```

会有提示, 都按回车就可以

```
[root@localhost .ssh]# ssh-keygen -t rsa
```

加入授权

```
[root@localhost .ssh]# cat ./id_rsa.pub >> ./authorized_keys
```

3.1.4 安装 hadoop2.7.4

(1) 解压安装包

```
[root@localhost bigdata]# tar -zxvf hadoop-2.7.4.tar.gz
```

(2) 配置 hadoop

- 配置 hadoop 目录到 Path 变量

```
[root@localhost hadoop-2.7.4]# vim ~/.bash-profile
```

```
export HADOOP_HOME=/opt/bigdata/hadoop-2.7.4
export JAVA_HOME=/opt/bigdata/jdk1.8.0_191
PATH=$PATH:$HOME/bin:$JAVA_HOME/bin:$HADOOP_HOME/bin
export PATH
```

让配置生效

```
[root@localhost hadoop-2.7.4]# source ~/.bash-profile
```

验证配置正确并生效:

```
[root@localhost hadoop-2.7.4]# hadoop version
```

```
[root@localhost hadoop-2.7.4]# hadoop version
Hadoop 2.7.4
Subversion https://shv@git-wip-us.apache.org/repos/asf/hadoop.git -r cd915e1e8d9d0131462a0b7301586c175728a282
Compiled by kshvachk on 2017-08-01T00:29Z
Compiled with protoc 2.5.0
From source with checksum 50b0468318b4ce9bd24dc467b7c41148
This command was run using /opt/bigdata/hadoop-2.7.4/share/hadoop/common/hadoop-common-2.7.4.jar
```

- 配置 hadoop-env.sh

将 Java 目录配置到该配置文件中。在 hadoop-env.sh 配置文件中，约 25 行处，按以下方式进行修改:

```
export JAVA_HOME=/opt/bigdata/jdk1.8.0_191
```

```
24 # The java implementation to use.
25 #export JAVA_HOME=${JAVA_HOME}
26 export JAVA_HOME=/opt/bigdata/jdk1.8.0_191
```

- 配置 core-site.xml

配置文件系统及端口号。

```
<configuration>
```

```
<property>
```

```

<name>fs.defaultFS</name>
<value>hdfs://localhost:9000</value>
</property>
<property>
  <name>hadoop.tmp.dir</name>
  <value>file:/usr/local/hadoop/tmp</value>
</property>
</configuration>

```

```

<configuration>
  <property>
    <name>fs.defaultFS</name>
    <value>hdfs://localhost:9000</value>
  </property>
  <property>
    <name>hadoop.tmp.dir</name>
    <value>file:/usr/local/hadoop/tmp</value>
  </property>
</configuration>

```

思考: 此处 localhost 配置为本机 IP(例如: 192.168.188.138)是否可行? ?

配置项说明:

配置项	作用及说明	默认值
fs.defaultFS	The name of the default file system. A URI whose scheme and authority determine the FileSystem implementation. The uri's scheme determines the config property (fs.SCHEME.impl) naming the FileSystem implementation class. The uri's authority is used to determine the host, port, etc. for a filesystem.	file:///
hadoop.tmp.dir	A base for other temporary	/tmp/hadoop-\${user.name}

	directories 该配置可不配，使用默认值	
--	------------------------------------	--

core-site.xml 所有参数官方说明：

<https://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-common/core-default.xml#>

说明：hadoop2.7.4 版本的 core-site.xml 文件中共 334 个配置项，绝大部分配置项都有默认值。

● 配置 hdfs-site.xml

```
<configuration>
  <property>
    <name>file.replication</name>
    <value>1</value>
  </property>
  <property>
    <name>dfs.namenode.name.dir</name>
    <value>file:///data/hadoop/hdfs/nn</value>
  </property>
  <property>
    <name>dfs.datanode.data.dir</name>
    <value>file:///data/hadoop/hdfs/dn</value>
  </property>
  <property>
    <name>dfs.namenode.checkpoint.dir</name>
    <value>file:///data/hadoop/hdfs/snn</value>
  </property>
</configuration>
```



```

<configuration>
  <property>
    <name>file.replication</name>
    <value>1</value>
  </property>

  <property>
    <name>dfs.namenode.name.dir</name>
    <value>file:///data/hadoop/hdfs/nn</value>
  </property>
  <property>
    <name>dfs.datanode.data.dir</name>
    <value>file:///data/hadoop/hdfs/dn</value>
  </property>
  <property>
    <name>dfs.namenode.checkpoint.dir</name>
    <value>file:///data/hadoop/hdfs/snn</value>
  </property>
</configuration>

```

以上各配置项目录，如果配置了，则必须在启动 hadoop 时手动创建目录。

配置项说明：

配置项	作用及说明	默认值
file.replication	Replication factor 可不配置，使用默认值	1
dfs.namenode.name.dir	Determines where on the local filesystem the DFS name node should store the name table(fsimage). If this is a comma-delimited list of directories then the name table is replicated in all of the directories, for redundancy.	file://\${hadoop.tmp.dir}/dfs/name
dfs.datanode.data.dir	Determines where on the local filesystem an DFS data node should store its blocks. If this is a comma-delimited list of directories, then data will be stored in all named	file://\${hadoop.tmp.dir}/dfs/data

	<p>directories, typically on different devices. The directories should be tagged with corresponding storage types</p> <p>(([SSD]/[DISK]/[ARCHIVE]/[RAM_DISK]) for HDFS storage policies.</p> <p>The default storage type will be DISK if the directory does not have a storage type tagged explicitly.</p> <p>Directories that do not exist will be created if local filesystem permission allows.</p>	
dfs.namenode.checkpoint.dir	<p>Determines where on the local filesystem the DFS secondary name node should store the temporary images to merge. If this is a comma-delimited list of directories then the image is replicated in all of the directories for redundancy.</p>	<p>file://\${hadoop.tmp.dir}/dfs/namesecondary</p>

hdfs-site.xml 所有配置项官方说明:

<https://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-hdfs/hdfs-default.xml>

● 配置 mapred-site.xml

```
<configuration>
```

```
  <property>
```

```
    <name>mapreduce.framework.name</name>
```

```
    <value>yarn</value>
```

```
  </property>
```

```
</configuration>
```

各配置项说明:

配置项	作用及说明	默认值
mapreduce.framework.name	The runtime framework for executing MapReduce jobs. Can be one of local, classic or yarn. 该配置也可以不配, 使用默认值	local

如果 mapreduce.framework.name 配置项配置为“yarn”, 那么必须继续配置 yarn-site.xml。

● yarn-site.xml 配置

```
<configuration>
```

```
<property>
```

```
<name>yarn.resourcemanager.hostname</name>
```

```
<value>localhost</value>
```

```
</property>
```

```
<property>
```

```
<name>yarn.nodemanager.aux-services</name>
```

```
<value>mapreduce_shuffle</value>
```

```
</property>
```

```
</configuration>
```

各配置项说明:

配置项	作用及说明	默认值
yarn.resourcemanager.hostname	The hostname of the RM. 指定 ResourceManager 的地址	0.0.0.0
yarn.nodemanager.aux-services	A comma separated list of services where service name should only contain a-zA-Z0-9_ and can not start with numbers NodeManager 上运行的附属服务。需配置成 mapreduce_shuffle, 才可运行 MapReduce 程序	无

yarn.nodemanager.local-dirs	List of directories to store localized files in. An application's localized file directory will be found in: \${yarn.nodemanager.local-dirs}/usercache/\${user}/appcache/application_\${appid}. Individual containers' work directories, called container_\${contid}, will be subdirectories of this.	\${hadoop.tmp.dir}/nm-local-dir
-----------------------------	---	---------------------------------

● 创建相关配置目录

```
[root@localhost jdk1.8.0-191]# mkdir -p /data/hadoop/hdfs/dn
[root@localhost jdk1.8.0-191]# mkdir -p /data/hadoop/hdfs/nn
[root@localhost jdk1.8.0-191]# mkdir -p /data/hadoop/hdfs/snn
[root@localhost jdk1.8.0-191]# mkdir -p /data/hadoop/yarn/nm
```

3.1.5 hdfs 集群格式化

```
[root@localhost jdk1.8.0-191]# hdfs namenode -format
```

```
20/10/05 05:15:30 INFO metrics.TopMetrics: NNTop conf: dfs.namenode.top.num.users = 10
20/10/05 05:15:30 INFO metrics.TopMetrics: NNTop conf: dfs.namenode.top.windows.minutes = 1,5,25
20/10/05 05:15:30 INFO namenode.FSNamesystem: Retry cache on namenode is enabled
20/10/05 05:15:30 INFO namenode.FSNamesystem: Retry cache will use 0.03 of total heap and retry cache entry expiry time is 600000 millis
20/10/05 05:15:30 INFO util.GSet: Computing capacity for map NameNodeRetryCache
20/10/05 05:15:30 INFO util.GSet: VM type = 64-bit
20/10/05 05:15:30 INFO util.GSet: 0.029999999329447746% max memory 966.7 MB = 297.0 KB
20/10/05 05:15:30 INFO util.GSet: capacity = 2^15 = 32768 entries
20/10/05 05:15:34 INFO namenode.FSImage: Allocated new BlockPoolId: BP-1138813009-127.0.0.1-1601889330965
20/10/05 05:15:35 INFO common.Storage: Storage directory /data/hadoop/hdfs/nn has been successfully formatted.
20/10/05 05:15:35 INFO namenode.FSImageFormatProtobuf: Saving image file /data/hadoop/hdfs/nn/current/fsimage.ckpt_00000000000000000000 using no compression
20/10/05 05:15:37 INFO namenode.FSImageFormatProtobuf: Image file /data/hadoop/hdfs/nn/current/fsimage.ckpt_00000000000000000000 of size 321 bytes saved in 1 seconds.
20/10/05 05:15:37 INFO namenode.NNStorageRetentionManager: Going to retain 1 images with txid >= 0
20/10/05 05:15:37 INFO util.ExitUtil: Exiting with status 0
20/10/05 05:15:37 INFO namenode.NameNode: SHUTDOWN_MSG:
/*****
SHUTDOWN_MSG: Shutting down NameNode at localhost/127.0.0.1
*****/
[root@localhost jdk1.8.0-191]#
```

3.1.6 启动集群

(1) 使用 yarn 作为集群资源管理系统时启动方式

```
[root@localhost sbin]# ./start-all.sh
```

启动日志:

```
[root@localhost sbin]# ./start-all.sh
```

This script is Deprecated. Instead use start-dfs.sh and start-yarn.sh

Starting namenodes on [localhost]

The authenticity of host 'localhost (::1)' can't be established.

ECDSA key fingerprint is SHA256:IqszgY4nj7DfEiLxAWw4UIk9Pe1Q6zGKGIPszMp4Fc8.

ECDSA key fingerprint is MD5:b9:07:26:5e:69:f6:48:3e:9d:07:b8:66:4c:a4:89:51.

Are you sure you want to continue connecting (yes/no)? yes

localhost: Warning: Permanently added 'localhost' (ECDSA) to the list of known hosts.

root@localhost's password:

localhost: starting namenode, logging to /opt/bigdata/hadoop-2.7.4/logs/hadoop-root-namenode-localhost.localdomain.out

root@localhost's password:

localhost: starting datanode, logging to /opt/bigdata/hadoop-2.7.4/logs/hadoop-root-datanode-localhost.localdomain.out

Starting secondary namenodes [0.0.0.0]

The authenticity of host '0.0.0.0 (0.0.0.0)' can't be established.

ECDSA key fingerprint is SHA256:IqszgY4nj7DfEiLxAWw4UIk9Pe1Q6zGKGIPszMp4Fc8.

ECDSA key fingerprint is MD5:b9:07:26:5e:69:f6:48:3e:9d:07:b8:66:4c:a4:89:51.

Are you sure you want to continue connecting (yes/no)? yes

0.0.0.0: Warning: Permanently added '0.0.0.0' (ECDSA) to the list of known hosts.

root@0.0.0.0's password:

0.0.0.0: starting secondarynamenode, logging to /opt/bigdata/hadoop-2.7.4/logs/hadoop-root-secondarynamenode-localhost.localdomain.out

starting yarn daemons

starting resourcemanager, logging to /opt/bigdata/hadoop-2.7.4/logs/yarn-root-resourcemanager-localhost.localdomain.out

root@localhost's password:

localhost: starting nodemanager, logging to /opt/bigdata/hadoop-2.7.4/logs/yarn-root-nodemanager-localhost.localdomain.out

使用 jps 命令查看启动的进程:

```
[root@localhost sbin]# jps
```

```
[root@localhost sbin]# jps
4144 NodeManager
4321 Jps
3670 DataNode
3581 NameNode
3869 SecondaryNameNode
4014 ResourceManager
[root@localhost sbin]#
```

(2) 使用 local 方式作为集群资源管理系统时启动方式

```
[root@localhost sbin]# ./start-dfs.sh
```

启动日志:

```
[root@localhost sbin]# ./start-dfs.sh
Starting namenodes on [localhost]
root@localhost's password:
localhost: starting namenode, logging to /opt/bigdata/hadoop-2.7.4/logs/hadoop-root-namenode-localhost.localdomain.out
root@localhost's password:
localhost: starting datanode, logging to /opt/bigdata/hadoop-2.7.4/logs/hadoop-root-datanode-localhost.localdomain.out
Starting secondary namenodes [0.0.0.0]
root@0.0.0.0's password:
0.0.0.0: starting secondarynamenode, logging to /opt/bigdata/hadoop-2.7.4/logs/hadoop-root-secondarynamenode-localhost.localdomain.out
```

使用 jps 命令查看启动进程:

```
[root@localhost sbin]# jps
```

```
[root@localhost sbin]# jps
8673 SecondaryNameNode
8786 Jps
8387 NameNode
8495 DataNode
[root@localhost sbin]#
```

3.2 Hadoop 用户部署

3.2.1 创建 hadoop 用户并修改权限

- 创建 hadoop 用户

```
[root@localhost sbin]# useradd -m hadoop
```

- 修改 hadoop 用户密码

```
[root@localhost sbin]# passwd hadoop
```

- 修改 hadoop 用户权限

修改/etc/sudoers 文件，确认文件中以下内容：

```
[root@localhost sbin]# vim /etc/sudoers
```

```
## Allow root to run any commands anywhere
root    ALL=(ALL)        ALL

## Allows members of the 'sys' group to run networking, software,
## service management apps and more.
# %sys ALL = NETWORKING, SOFTWARE, SERVICES, STORAGE, DELEGATING,

## Allows people in group wheel to run all commands
%wheel  ALL=(ALL)        ALL
```

- 将 hadoop 用户加入 root 用户组

```
[hadoop@localhost ~]$ usermod -g root Hadoop
```

3.2.2 安装 java1.8

- (1) 解压安装包

```
[hadoop@localhost ~]$ tar -zxvf jdk-8u191-linux-x64.tar.gz
```

- (2) 配置环境变量

```
[hadoop@localhost hadoop-2.7.4]$ vim ~/.bash-profile
```

```
export JAVA_HOME=/opt/bigdata-hd/jdk1.8.0_191
```

```
# User specific environment and startup programs
export JAVA_HOME=/opt/bigdata_hd/jdk1.8.0_191
export HADOOP_HOME=/opt/bigdata_hd/hadoop-2.7.4

PATH=$PATH:$HOME/.local/bin:$HOME/bin:$JAVA_HOME/bin:$HADOOP_HOME/bin
export PATH
```

- (3) 配置生效

```
[hadoop@localhost hadoop-2.7.4]$ source ~/.bash_profile
```

3.2.3 配置免密码登录

```
[hadoop@localhost ssh]$ cd ~/.ssh/
```

```
[hadoop@localhost .ssh]$ ssh-keygen -t rsa
```

```
[hadoop@localhost .ssh]$ cat id_rsa.pub >> authorized_keys
```

3.2.4 安装 Hadoop

(1) 解压安装包

```
[hadoop@localhost bigdata-hd]$ tar -zxvf hadoop-2.7.4.tar.gz
```

(2) 配置 `hadoop-env.sh`

修改文件的 25 行位置，将 java 的主目录配置到文件中。

```
export JAVA_HOME=/opt/bigdata-hd/jdk1.8.0-191
```

(3) 配置 `core-site.xml`

```
<configuration>
```

```
  <property>
```

```
    <name>fs.defaultFS</name>
```

```
    <value>hdfs://localhost:9000</value>
```

```
  </property>
```

```
  <property>
```

```
    <name>hadoop.tmp.dir</name>
```

```
    <value>file:/usr/local/hadoop-hd/tmp</value>
```

```
  </property>
```

```
</configuration>
```



```
<configuration>
  <property>
    <name>fs.defaultFS</name>
    <value>hdfs://localhost:9000</value>
  </property>
  <property>
    <name>hadoop.tmp.dir</name>
    <value>file:/usr/local/hadoop_hd/tmp</value>
  </property>
</configuration>
```

(4) 配置 hdfs-site.xml

```
<configuration>

  <property>

    <name>file.replication</name>

    <value>1</value>

  </property>

  <property>

    <name>dfs.namenode.name.dir</name>

    <value>file:/data/hadoop_hd/hdfs/nn</value>

  </property>

  <property>

    <name>dfs.datanode.data.dir</name>

    <value>file:/data/hadoop_hd/hdfs/dn</value>

  </property>

  <property>

    <name>dfs.namenode.checkpoint.dir</name>

    <value>file:/data/hadoop_hd/hdfs/snn</value>

  </property>

</configuration>
```

```
<configuration>
  <property>
    <name>file.replication</name>
    <value>1</value>
  </property>
  <property>
    <name>dfs.namenode.name.dir</name>
    <value>file:/data/hadoop_hd/hdfs/nn</value>
  </property>
  <property>
    <name>dfs.datanode.data.dir</name>
    <value>file:/data/hadoop_hd/hdfs/dn</value>
  </property>
  <property>
    <name>dfs.namenode.checkpoint.dir</name>
    <value>file:/data/hadoop_hd/hdfs/snn</value>
  </property>
</configuration>
```

(5) 配置 mapred-site.xml

伪分布式方式部署 Hadoop 时，mapred-site.xml 文件中关于 mapreduce.framework.name 配置项可以直接使用默认值（local）。

3.2.5 创建目录

```
[root@localhost data]# mkdir -p /home/hadoop/hadoop_hd/tmp
[root@localhost data]# mkdir -p /home/hadoop/hadoop_hd/hdfs/nn
[root@localhost data]# mkdir -p /home/hadoop/hadoop_hd/hdfs/dn
[root@localhost data]# mkdir -p /home/hadoop/hadoop_hd/hdfs/snn
```

特别注意：设置目录要求 hadoop 用户具备对应的操作权限。

3.2.6 格式化 hdfs

```
[hadoop@localhost hadoop]$ hdfs namenode -format
```

```
20/10/06 00:47:34 INFO metrics.TopMetrics: NNTop conf: dfs.namenode.top.window.num.buckets = 10
20/10/06 00:47:34 INFO metrics.TopMetrics: NNTop conf: dfs.namenode.top.num.users = 10
20/10/06 00:47:34 INFO metrics.TopMetrics: NNTop conf: dfs.namenode.top.windows.minutes = 1,5,25
20/10/06 00:47:34 INFO namenode.FSNamesystem: Retry cache on namenode is enabled
20/10/06 00:47:34 INFO namenode.FSNamesystem: Retry cache will use 0.03 of total heap and retry c
ache entry expiry time is 600000 millis
20/10/06 00:47:34 INFO util.GSet: Computing capacity for map NameNodeRetryCache
20/10/06 00:47:34 INFO util.GSet: VM type = 64-bit
20/10/06 00:47:34 INFO util.GSet: 0.029999999329447746% max memory 966.7 MB = 297.0 KB
20/10/06 00:47:34 INFO util.GSet: capacity = 2^15 = 32768 entries
20/10/06 00:47:34 INFO namenode.FSImage: Allocated new BlockPoolId: BP-942029196-127.0.0.1-160195
9654200
20/10/06 00:47:34 INFO common.Storage: Storage directory /home/hadoop/hadoop_hd/hdfs/nn has been
successfully formatted.
20/10/06 00:47:34 INFO namenode.FSImageFormatProtobuf: Saving image file /home/hadoop/hadoop_hd/h
dfs/nn/current/fsimage.ckpt_000000000000000000 using no compression
20/10/06 00:47:34 INFO namenode.FSImageFormatProtobuf: Image file /home/hadoop/hadoop_hd/hdfs/nn/
current/fsimage.ckpt_000000000000000000 of size 322 bytes saved in 0 seconds.
20/10/06 00:47:34 INFO namenode.NNStorageRetentionManager: Going to retain 1 images with txid >=
0
20/10/06 00:47:34 INFO util.ExitUtil: Exiting with status 0
20/10/06 00:47:34 INFO namenode.NameNode: SHUTDOWN_MSG:
/*****
SHUTDOWN_MSG: Shutting down NameNode at localhost/127.0.0.1
*****/
[hadoop@localhost hadoop]$
```

3.2.7 启动集群

```
[hadoop@localhost sbin]$ ./start-dfs.sh
```

使用 jps 命令，查看启动的 java 进程：

```
[hadoop@localhost sbin]$ jps
```

```
[hadoop@localhost sbin]$ jps
2352 NameNode
2449 DataNode
2744 Jps
2635 SecondaryNameNode
```

4. Hadoop 全分布式部署

4.1 hadoop 用户部署

4.1.1 服务器信息

创建 3 个 CentOS7 虚拟机系统，用于搭建全分布式 Hadoop 集群，各服务器 IP 以及在集群中分配的角色如下表。

节点	IP 地址	主机名
Master	192.168.188.129	Master
Slave1	192.168.188.130	Slave1
Slave2	192.168.188.131	Slave2

4.1.2 上传安装包到 master 节点

将 Java、Hadoop 安装包上传到 Master 节点。

4.1.3 创建 hadoop 用户并赋权

```
[root@Master ~]# useradd hadoop
```

```
[root@Master ~]# passwd hadoop
```

修改/etc/sudoers 文件

```
[root@Master ~]# usermod -g root hadoop
```

4.1.4 设置免密码登录

(1) 在 Master 节点生成秘钥

- 进入 ~/.ssh 目录

```
[hadoop@Master bigdata]$ cd ~/.ssh/
```

- 生成秘钥，并加入授权文件末尾

```
[hadoop@Master .ssh]$ ssh-keygen -t rsa
```

```
[hadoop@Master .ssh]$ cat id_rsa.pub >> authorized_keys
```

- 验证是否生效

```
[hadoop@Master .ssh]$ ssh localhost
```

如果不需要输入密码登录成功，则表示配置成功。

(2) 在 Slave1 节点生成秘钥并发送到 Master 节点

- 进入 .ssh 目录

```
[hadoop@Slave1 bigdata]$ cd ~/.ssh/
```

- 生成 rsa 类型公钥

```
[hadoop@Slave1 .ssh]$ ssh-keygen -t rsa
```

- 将 Slave1 生成的公钥发送到 Master 节点

```
[hadoop@Slave1 .ssh]$ scp id_rsa.pub
```

```
Master: /home/Hadoop/.ssh/id_rsa-slave1.pub
```

- 在 Master 节点将 id_rsa-slave1.pub 内容加入 authorized_keys 文件

```
[hadoop@Master .ssh]$ cat id_rsa-slave1.pub >>
```

```
authorized_keys
```

(3) 在 Slave2 节点生成秘钥并发送到 Master 节点

- 进入 .ssh 目录

```
[hadoop@Slave2 bigdata]$ cd ~/.ssh/
```

- 生成 rsa 类型公钥

```
[hadoop@Slave2 .ssh]$ ssh-keygen -t rsa
```

- 将 Slave2 生成的公钥发送到 Master 节点

```
[hadoop@Slave2 .ssh]$ scp id_rsa.pub
```

```
Master: /home/hadoop/.ssh/id_rsa-slave2.pub
```

- 在 Master 节点将 id_rsa-slave2.pub 内容合并到 authorized_keys

```
[hadoop@Master .ssh]$ cat id_rsa-slave2.pub >>
```

```
authorized_keys
```

(4) 将 Master 节点的 authorized_keys 文件发送到 Slave1 和 Slave2

```
[hadoop@Master .ssh]$ scp authorized_keys
```

```
Slave1: /home/hadoop/.ssh/
```

```
[hadoop@Master .ssh]$ scp authorized_keys
```

Slave2: /home/hadoop/.ssh/

(5) 测试免密码登录

- 在 Master 节点使用 ssh 命令登录 Slave1 和 Slave2 节点

```
[hadoop@Master .ssh]$ ssh Slave1
```

```
[hadoop@Master .ssh]$ ssh Slave2
```

- 在 Slave1 节点使用 ssh 命令登录 Master 和 Slave2 节点

```
[hadoop@Slave1 .ssh]$ ssh Master
```

```
[hadoop@Slave1 .ssh]$ ssh Slave2
```

- 在 Slave2 节点使用 ssh 命令登录 Master 和 Slave1 节点

```
[hadoop@Slave2 .ssh]$ ssh Master
```

```
[hadoop@Slave2 .ssh]$ ssh Slave1
```

如果以上各命令执行时无需输入登录密码，则表示免密码登录设置成功。

由于全分布式部署 Hadoop 集群，集群的每一台节点都要进行对应的配置，因此以下的配置先在主节点（Master）配置，后发送到工作节点（Slaves）。

4.1.5 安装 Hadoop

(1) 配置 `hadoop-env.sh`

在该文件中配置 JAVA_HOME 变量，在文件 25 行处修改变量值。

```
[hadoop@Master hadoop]$ vim hadoop-env.sh
```

```
export JAVA_HOME=/opt/bigdata/jdk1.8.0_191
```

```
24 # The java implementation to use.
25 #export JAVA_HOME=${JAVA_HOME}
26 export JAVA_HOME=/opt/bigdata/jdk1.8.0_191
```

(2) 配置 `core-site.xml`

- 进入 hadoop 配置文件目录

```
[hadoop@Master hadoop]$ cd /opt/bigdata/hadoop-2.7.4/etc/hadoop
```

- 打开 `core-site.xml` 并进行配置

```
<configuration>
```

```
<property>
  <name>fs.defaultFS</name>
  <value>hdfs://Master:9000</value>
</property>
<property>
  <name>hadoop.tmp.dir</name>
  <value>file:/usr/local/data/hadoop/tmp</value>
</property>
</configuration>
```

提醒：fs.defaultFS 配置的值必须为主节点的主机名或 IP 地址

```
<configuration>
  <property>
    <name>fs.defaultFS</name>
    <value>hdfs://localhost:9000</value>
  </property>
  <property>
    <name>hadoop.tmp.dir</name>
    <value>file:/usr/local/data/hadoop/tmp</value>
  </property>
</configuration>
```

(3) 配置 hdfs-site.xml

```
<configuration>

  <property>

    <name>file.replication</name>

    <value>1</value>

  </property>

  <property>

    <name>dfs.namenode.name.dir</name>

    <value>/data/hadoop/hdfs/nn</value>

  </property>

  <property>
```

```
    <name>dfs.datanode.data.dir</name>
    <value>/data/hadoop/hdfs/dn</value>
  </property>
  <property>
    <name>dfs.namenode.checkpoint.dir</name>
    <value>/data/hadoop/hdfs/nnc</value>
  </property>
</configuration>
```

```
<configuration>
  <property>
    <name>file.replication</name>
    <value>2</value>
  </property>
  <property>
    <name>dfs.namenode.name.dir</name>
    <value>/data/hadoop/hdfs/nn</value>
  </property>
  <property>
    <name>dfs.datanode.data.dir</name>
    <value>/data/hadoop/hdfs/dn</value>
  </property>
  <property>
    <name>dfs.namenode.checkpoint.dir</name>
    <value>/data/hadoop/hdfs/nnc</value>
  </property>
</configuration>
```

(4) 配置 mapred-site.xml

```
<configuration>

  <property>

    <name>mapreduce.framework.name</name>

    <value>yarn</value>

  </property>

</configuration>
```



```
<configuration>
  <property>
    <name>mapreduce.framework.name</name>
    <value>yarn</value>
  </property>
</configuration>
```

(5) 配置 yarn-site.xml

```
<configuration>

  <property>

    <name>yarn.resourcemanager.hostname</name>

    <value>Master</value>

  </property>

  <property>

    <name>yarn.nodemanager.aux-services</name>

    <value>mapreduce_shuffle</value>

  </property>

</configuration>
```

```
<configuration>
  <property>
    <name>yarn.resourcemanager.hostname</name>
    <value>localhost</value>
  </property>
  <property>
    <name>yarn.nodemanager.aux_services</name>
    <value>mapreduce_shuffle</value>
  </property>
</configuration>
```

(6) 配置 slaves

把集群各节点配置到 slaves 文件中。

```
Master
Slave1
Slave2
```

(7) 将 Master 节点配置好的安装包发送到其他节点

```
[hadoop@Master bigdata]$ scp -r hadoop-2.7.4 Slave1:/opt/bigdata/
```

```
[hadoop@Master bigdata]$ scp -r hadoop-2.7.4 Slave2:/opt/bigdata/
```

(8) 创建目录并赋权

(1) 创建目录

```
[hadoop@Master sbin]$ sudo mkdir -p /data/hadoop/hdfs/nn
```

```
[hadoop@Master sbin]$ sudo mkdir -p /data/hadoop/hdfs/dn
```

```
[hadoop@Master sbin]$ sudo mkdir -p /data/hadoop/hdfs/nnc
```

```
[hadoop@Master sbin]$ sudo chown -R hadoop:root /data/hadoop/hdfs/nn
```

```
[hadoop@Master sbin]$ sudo chown -R hadoop:root /data/hadoop/hdfs/dn
```

```
[hadoop@Master sbin]$ sudo chown -R hadoop:root /data/hadoop/hdfs/nnc
```

(2) 修改目录权限

● 修改/usr/local/data 目录权限

```
[hadoop@Master sbin]$ sudo mkdir /usr/local/data
```

```
[hadoop@Master sbin]$ sudo chown -R hadoop:root /usr/local/data/
```

● 修改 hadoop 安装主目录权限

```
[hadoop@Master /]$ sudo chown -R hadoop:root /opt
```

备注：hadoop 安装在/opt/bigdata/目录下。

(9) 关闭防火墙

从 CentOS 7.0 默认使用的是 firewall 作为防火墙。

检查防火墙状态：

```
[hadoop@Slave1 logs]$ sudo systemctl status firewalld
```

关闭防火墙：

```
[hadoop@Slave1 logs]$ sudo systemctl stop firewalld
```

禁止开机启动：

```
[hadoop@Slave1 logs]$ sudo systemctl disable firewalld
```

(10) 启动集群

在主节点 (Master) 执行以下 2 条命令:

```
[hadoop@Master sbin]$ ./start-dfs.sh
```

```
[hadoop@Master sbin]$ ./start-yarn.sh
```

(11) 验证集群

在 Master 节点执行 `./start-dfs.sh` 命令后, 主节点以及从节点会启动 NameNode 和 DataNode 进程。在主节点 (Master) 将同时启动 NameNode 和 DataNode 两个进程, 而从节点都只会启动一个 DataNode 进程。

在 Master 节点执行 `./start-yarn.sh` 命令后, 主节点会继续启动 ResourceManager 和 NodeManager 两个进程。而从节点会启动 NodeManager 进程。

最后, 集群启动以后, 各节点所启动的进程如下所示。

(1) 主节点 (Master)

```
[hadoop@Master sbin]$ jps
9042 ResourceManager
9491 Jps
8597 NameNode
8696 DataNode
9144 NodeManager
[hadoop@Master sbin]$
```

(2) 从节点 (Slave)

```
[hadoop@Slave1 logs]$ jps
4230 Jps
4025 NodeManager
3898 DataNode
[hadoop@Slave1 logs]$
```

```
[hadoop@Slave2 logs]$ jps
3735 Jps
3434 DataNode
3551 NodeManager
[hadoop@Slave2 logs]$
```

4.1.6 集群管理系统

在虚拟机中部署 Hadoop 集群，需要关闭 linux 的防火墙，windows 系统中才可以访问。

(1) HDFS 集群

访问 <http://MASTER-IP:50070>

● HDFS 系统首页

Hadoop	Overview	Datanodes	Datanode Volume Failures	Snapshot	Startup Progress	Utilities -
--------	----------	-----------	--------------------------	----------	------------------	-------------

Overview 'Master:9000' (active)

Started:	Sat Oct 10 09:27:26 EDT 2020
Version:	2.7.4, rcd915e1e8d9d0131462a0b7301586c175728a282
Compiled:	2017-08-01T00:29Z by kshvachk from branch-2.7.4
Cluster ID:	CID-51691e70-9e7c-44fa-9261-ad152c3ac5bc
Block Pool ID:	BP-1397824087-192.168.188.129-1602050116898

Summary

Security is off.

Safemode is off.

1 files and directories, 0 blocks = 1 total filesystem object(s).

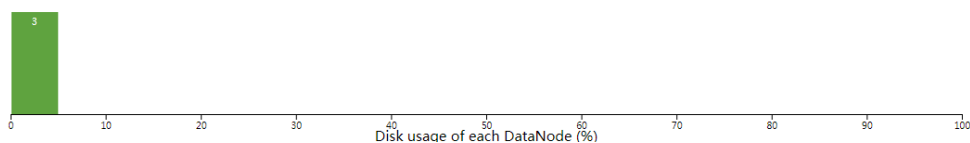
Heap Memory used 38.83 MB of 46.38 MB Heap Memory. Max Heap Memory is 966.69 MB.

Non Heap Memory used 42.13 MB of 43 MB Committed Non Heap Memory. Max Non Heap Memory is -1 B.

● HDFS 集群节点页面

Datanode Information

Datanode usage histogram




In operation

Node	Last contact	Admin State	Capacity	Used	Non DFS Used	Remaining	Blocks	Block pool used	Failed Volumes	Version
Slave1:50010 (192.168.188.130:50010)	2	In Service	7.99 GB	12 KB	2.14 GB	5.85 GB	0	12 KB (0%)	0	2.7.4
Master:50010 (192.168.188.129:50010)	0	In Service	7.99 GB	12 KB	3.43 GB	4.55 GB	0	12 KB (0%)	0	2.7.4
Slave2:50010 (192.168.188.131:50010)	1	In Service	7.99 GB	12 KB	2.14 GB	5.85 GB	0	12 KB (0%)	0	2.7.4

(2) yarn 集群

访问地址: <http://MASTER-IP:8088>

● Yarn 集群首页



Logged in as: dr.who

All Applications

Cluster

- About
- Nodes
- Node Labels
- Applications
- NEW
- NEW SAVING
- SUBMITTED
- ACCEPTED
- RUNNING
- FINISHED
- FAILED
- KILLED
- Scheduler

Tools

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	VCores Used	VCores Total	VCores Reserved	Active Nodes	Decommissioned Nodes	Lost Nodes	Unhealthy Nodes	Rebooted Nodes
0	0	0	0	0	0 B	24 GB	0 B	0	24	0	2	0	0	0	0

Scheduler Metrics


Scheduler Type	Scheduling Resource Type	Minimum Allocation	Maximum Allocation
Capacity Scheduler	[MEMORY]	<memory:1024, vCores:1>	<memory:8192, vCores:8>

Show 20 entries

ID	User	Name	Application Type	Queue	StartTime	FinishTime	State	FinalStatus	Progress	Tracking UI	Blacklisted Nodes
No data available in table											

Showing 0 to 0 of 0 entries

● Yarn 集群节点页面



Logged in as: dr.who

Nodes of the cluster

Cluster

- About
- Nodes
- Node Labels
- Applications
- NEW
- NEW SAVING
- SUBMITTED
- ACCEPTED
- RUNNING
- FINISHED
- FAILED
- KILLED
- Scheduler

Tools

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	VCores Used	VCores Total	VCores Reserved	Active Nodes	Decommissioned Nodes	Lost Nodes	Unhealthy Nodes	Rebooted Nodes
0	0	0	0	0	0 B	24 GB	0 B	0	24	0	2	0	0	0	0

Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum Allocation	Maximum Allocation
Capacity Scheduler	[MEMORY]	<memory:1024, vCores:1>	<memory:8192, vCores:8>

Show 20 entries


Node Labels	Rack	Node State	Node Address	Node HTTP Address	Last health-update	Health-report	Containers	Mem Used	Mem Avail	VCores Used	VCores Avail	Version
/default-rack		RUNNING	Master:43677	Master:8042	Sat Oct 10 10:06:30 -0400 2020		0	0 B	8 GB	0	8	2.7.4
/default-rack		RUNNING	Slave2:35189	Slave2:8042	Sat Oct 10 10:06:23 -0400 2020		0	0 B	8 GB	0	8	2.7.4
/default-rack		RUNNING	Slave1:37193	Slave1:8042	Sat Oct 10 10:06:24 -0400 2020		0	0 B	8 GB	0	8	2.7.4

Showing 1 to 3 of 3 entries

4.1.7 作业提交集群测试

使用 hadoop 自带的计算圆周率的例子，提交到集群。执行以下命令：

```
[hadoop@Master sbin]$ yarn jar /opt/bigdata/hadoop-2.7.4
/share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.4.jar pi 20 50
```



Logged in as: dr.who

All Applications

Cluster

- About
- Nodes
- Node Labels
- Applications
- NEW
- NEW SAVING
- SUBMITTED
- ACCEPTED
- RUNNING
- FINISHED
- FAILED
- KILLED
- Scheduler

Tools

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	VCores Used	VCores Total	VCores Reserved	Active Nodes	Decommissioned Nodes	Lost Nodes	Unhealthy Nodes	Rebooted Nodes
1	0	1	0	1	2 GB	24 GB	0 B	1	24	0	2	0	0	0	0

Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum Allocation	Maximum Allocation
Capacity Scheduler	[MEMORY]	<memory:1024, vCores:1>	<memory:8192, vCores:8>

Show 20 entries

ID	User	Name	Application Type	Queue	StartTime	FinishTime	State	FinalStatus	Progress	Tracking UI	Blacklisted Nodes
application_1602336498930_0001	hadoop	QuasiMonteCarlo	MAPREDUCE	default	Sat Oct 10 22:15:20 +0800 2020	N/A	ACCEPTED	UNDEFINED		ApplicationMaster	0

Showing 1 to 1 of 1 entries

4.2 root 用户部署

4.2.1 上传安装包

将 java、hadoop 安装包上传到 Master 节点某个目录（例如：
/opt/bigdata-root）。

```
[root@Master bigdata_root]# ll
total 447700
-rw-rw-r--. 1 root root 266688029 Oct  6 04:22 hadoop-2.7.4.tar.gz
-rw-rw-r--. 1 root root 191753373 Oct  6 04:22 jdk-8u191-linux-x64.tar.gz
[root@Master bigdata_root]#
```

4.2.2 安装 Java

```
export JAVA_HOME=/opt/bigdata-root/jdk1.8.0_191
```

```
PATH=$PATH:$HOME/bin:$JAVA_HOME/bin
```

```
# User specific environment and startup programs
export JAVA_HOME=/opt/bigdata_root/jdk1.8.0_191
PATH=$PATH:$HOME/bin:$JAVA_HOME/bin

export PATH
```

4.2.3 设置免密码登录

注意本节内容和 4.1.4 章节内容的区别。

（1）在各节点生成密钥对

```
Master: [root@Master .ssh]# ssh-keygen -t rsa
```

```
Slave1: [root@Slave1 .ssh]# ssh-keygen -t rsa
```

```
Slave2: [root@Slave2 .ssh]# ssh-keygen -t rsa
```

（2）在 Master, Slave1 和 Slave2 分别复制公钥到授权文件

```
[root@Master .ssh]# ssh-copy-id -i id_rsa.pub Master
```

```
[root@Slave1 .ssh]# ssh-copy-id -i id_rsa.pub Master
```

```
[root@Slave2 .ssh]# ssh-copy-id -i id_rsa.pub Master
```

（3）将 Master 节点的 authorized_keys 文件发送到其他节点

```
[root@Master .ssh]# scp authorized_keys Slave1:/root/.ssh/
```

```
[root@Master .ssh]# scp authorized_keys Slave2:/root/.ssh/
```

（4）验证免密码登录

在每一个节点尝试使用 ssh 远程登录到其他节点，验证免密码登录是否生

效。

4.2.4 安装 Hadoop

(1) 上传安装包

上传安装包到/opt/bigdata-root 目录。

(2) 解压安装包

```
[root@Master bigdata-root]# tar -zxvf hadoop-2.7.4.tar.gz
```

(3) 配置环境变量

```
[root@Master hadoop-2.7.4]# vim ~/.bash-profile
```

```
export HADOOP_HOME=/opt/bigdata-root/hadoop-2.7.4
```

```
PATH=$PATH:$HOME/bin:$JAVA_HOME/bin:$HADOOP_HOME/bin
```

(4) 将 Master 节点 root 用户配置文件发送到其他节点

```
[root@Master ~]# scp ~/.bash-profile Slave1:/root/
```

```
[root@Master ~]# scp ~/.bash-profile Slave2:/root/
```

(5) 配置 hadoop-env.sh

```
export JAVA_HOME=/opt/bigdata-root/jdk1.8.0-191
```

(6) 配置 Master 节点的 core-site.xml

```
<property>
  <name>fs.defaultFS</name>
  <value>hdfs://Master:9000</value>
</property>
<property>
  <name>hadoop.tmp.dir</name>
  <value>file:///usr/local/data/hadoop/tmp</value>
```

```
</property>
```

(7) 配置 Master 节点的 hdfs-site.xml

```
<property>
  <name>file.replication</name>
  <value>1</value>
</property>
<property>
  <name>dfs.namenode.name.dir</name>
  <value>file:///data/hadoop/hdfs/nn</value>
</property>
<property>
  <name>dfs.datanode.data.dir</name>
  <value>file:///data/hadoop/hdfs/dn</value>
</property>
<property>
  <name>dfs.namenode.checkpoint.dir</name>
  <value>file:///data/hadoop/hdfs/nnc</value>
</property>
```

(6) 配置 Master 节点的 mapred-site.xml

```
<property>
  <name>mapreduce.framework.name</name>
  <value>yarn</value>
</property>
```

(7) 配置 Master 节点的 yarn-site.xml

```
<property>
  <name>yarn.resourcemanager.hostname</name>
  <value>Master</value>
```



```
</property>

<property>

    <name>yarn.nodemanager.aux-services</name>

    <value>mapreduce_shuffle</value>

</property>
```

(8) 配置 Master 节点的 slaves

使用 vim 命令打开 slaves 文件，将以下内容替换原来的内容。

```
Master
Slave1
Slave2
```

(9) 将 Master 节点配置好的安装包发送到其他节点

可参照前文相关内容，也可以先将配置好 hadoop 包打包再发送。

(10) 分别在 Master、Slave1 和 Slave2 节点创建相关目录

在 Master 节点创建目录:

```
[root@Master ~]# mkdir -p /data/hadoop/hdfs/nn
[root@Master ~]# mkdir -p /data/hadoop/hdfs/dn
[root@Master ~]# mkdir -p /data/hadoop/hdfs/nnc
```

在 Slave1 节点创建目录:

```
[root@Slave1 ~]# mkdir -p /data/hadoop/hdfs/nn
[root@Slave1 ~]# mkdir -p /data/hadoop/hdfs/dn
[root@Slave1 ~]# mkdir -p /data/hadoop/hdfs/nnc
```

在 Slave2 节点创建目录:

```
[root@Slave2 ~]# mkdir -p /data/hadoop/hdfs/nn
[root@Slave2 ~]# mkdir -p /data/hadoop/hdfs/dn
```

```
[root@Slave2 ~]# mkdir -p /data/hadoop/hdfs/nnc
```

(11) 关闭所有节点的防火墙

关闭 Master 节点的防火墙:

```
[root@Master hadoop]# systemctl stop firewalld
```

```
[root@Master hadoop]# systemctl disable firewalld
```

关闭 Slave1 节点的防火墙:

```
[root@Slave1 hadoop]# systemctl stop firewalld
```

```
[root@Slave1 hadoop]# systemctl disable firewalld
```

关闭 Slave2 节点的防火墙:

```
[root@Slave2 hadoop]# systemctl stop firewalld
```

```
[root@Slave2 hadoop]# systemctl disable firewalld
```

(12) 启动集群

在 Master 节点 hadoop 的 sbin 目录执行:

```
[root@Master sbin]# ./start-dfs.sh
```

```
[root@Master sbin]# ./start-yarn.sh
```

(13) 验证集群

- 访问 hdfs 集群管理页面

<http://Master-IP:50070>

- 访问 yarn 集群管理页面

<http://Master-IP:8088>

- 提交测试任务到集群

```
yarn jar share/hadoop/mapreduce/hadoop-mapreduce-examples-  
2.7.4.jar pi 10 10
```

```
[root@Master hadoop-2.7.4]# yarn jar share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.4.jar pi 1 1
Number of Maps = 1
Samples per Map = 1
Wrote input for Map #0
Starting Job
20/10/14 11:49:34 INFO client.RMProxy: Connecting to ResourceManager at Master/192.168.188.129:8032
20/10/14 11:49:41 INFO input.FileInputFormat: Total input paths to process : 1
20/10/14 11:49:47 INFO mapreduce.JobSubmitter: number of splits:1
20/10/14 11:49:51 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1602689486088_0002
20/10/14 11:49:52 INFO impl.YarnClientImpl: Submitted application application_1602689486088_0002
20/10/14 11:49:52 INFO mapreduce.Job: The url to track the job: http://Master:8088/proxy/application_1602689486088_0002/
20/10/14 11:49:52 INFO mapreduce.Job: Running job: job_1602689486088_0002
20/10/14 11:58:59 INFO mapreduce.Job: Job job_1602689486088_0002 running in uber mode : false
20/10/14 11:59:00 INFO mapreduce.Job: map 0% reduce 0%
20/10/14 11:59:31 INFO mapreduce.Job: map 100% reduce 0%
20/10/14 11:59:40 INFO mapreduce.Job: map 100% reduce 100%
20/10/14 11:59:53 INFO mapreduce.Job: Job job_1602689486088_0002 completed successfully
20/10/14 11:59:54 INFO mapreduce.Job: Counters: 49
  File System Counters
    FILE: Number of bytes read=28
    FILE: Number of bytes written=242093
    FILE: Number of read operations=0
    FILE: Number of large read operations=0
    FILE: Number of write operations=0
    HDFS: Number of bytes read=261
    HDFS: Number of bytes written=215
    HDFS: Number of read operations=7
    HDFS: Number of large read operations=0
```

5. Hadoop HA 部署

5.1 解压 Zookeeper 安装包

```
[root@Master bigdata-root]# tar -zxvf zookeeper-3.4.5.tar.gz
```

5.2 配置环境变量

Master、Slave1 和 Slave2 节点都进行以下配置。

打开环境变量配置文件:

```
[root@Master zookeeper-3.4.5]# vim ~/.bash_profile
```

在配置文件中添加以下语句:

```
export ZOOKEEPER_HOME=/opt/bigdata-root/zookeeper-3.4.5
```

同时将 Zookeeper 的 bin 目录加入 Path:

```
PATH=$PATH: $HOME/bin: $JAVA_HOME/bin: $HADOOP_HOME/bin: $ZOOKEEPER_HOME/  
bin
```

配置生效:

```
[root@Master zookeeper-3.4.5]# source ~/.bash_profile
```

5.3 修改 zookeeper 配置文件

Zookeeper 的配置文件主要为 zoo.cfg, 按以下方式修改:

(1) 打开 zoo.cfg 文件

```
[root@Master conf]# cp zoo-sample.cfg zoo.cfg
```

```
[root@Master conf]# vim zoo.cfg
```

(2) 修改 DataDir 的值

```
dataDir=/opt/bigdata-root/zookeeper-3.4.5/dataDir
```

```
# the directory where the snapshot is stored.  
# do not use /tmp for storage, /tmp here is just  
# example sake.  
#dataDir=/tmp/zookeeper  
dataDir=/opt/bigdata_root/zookeeper-3.4.5/dataDir
```

(3) 添加 zookeeper 日志目录 (可选)

```
dataLogDir=/opt/bigdata-root/zookeeper-3.4.5/dataLogDir
```

```
# 配置zookeeper的日志目录，不配置时默认与dataDir相同
dataLogDir=/opt/bigdata_root/zookeeper-3.4.5/dataLogDir
```

(4) 指定 zookeeper 集群主机和端口

```
server.1=Master:2888:3888
server.2=Slave1:2888:3888
server.3=Slave2:2888:3888
```

```
# 不同的zookeeper服务器的自身标识，作为集群的一部分，
# 每一台服务器应该知道其他服务器的信息
server.1=Master:2888:3888
server.2=Slave1:2888:3888
server.3=Slave2:2888:3888
```

(5) 创建上文配置的目录

```
[root@Master zookeeper-3.4.5]# mkdir /opt/bigdata-root/zookeeper-
3.4.5/dataDir
```

```
[root@Master zookeeper-3.4.5]# mkdir /opt/bigdata-root/zookeeper-
3.4.5/dataLogDir
```

(6) 创建并编辑 myid

在 dataDir 目录内创建 myid 文件，并将“1”写入 myid 文件。

```
[root@Master dataDir]# touch myid
[root@Master dataDir]# echo 1 > myid
```

(7) 拷贝配置好的 zookeeper 到其他节点

为了便于发送，先将配置好的 zookeeper 目录打包再进行发送。

打包：

```
[root@Master bigdata-root]# tar -zcvf zookeeper-3.4.5.tar.gz  
zookeeper-3.4.5
```

发送:

```
[root@Master bigdata-root]# scp zookeeper-3.4.5.tar.gz  
Slave1: /opt/bigdata-root/  
[root@Master bigdata-root]# scp zookeeper-3.4.5.tar.gz  
Slave2: /opt/bigdata-root/
```

(8) 修改 Slave1 和 Slave2 节点的 myid 内容

myid 文件是作为当前机器在 zookeeper 集群的标识, 因此需要将 Slave1 和 Slave2 节点的 myid 进行修改。

- 修改 Slave1 节点的 myid:
将 dataDir 目录下的 myid 内容修改为 2
- 修改 Slave2 节点的 myid
将 dataDir 目录下的 myid 内容修改为 3

(9) 启动 Zookeeper 集群

- 启动集群

在 Master、Slave1 和 Slave2 节点分别执行以下命令:

```
./zkServer.sh start
```

- 查看集群状态

在 Master、Slave1 和 Slave2 节点分别执行以下语句:

```
./zkServer.sh status
```

Zookeeper 集群启动以后, 会有一个 leader 节点, 其他的都为 follower 节点, 如下图。

```
[root@Slave1 bin]# ./zkServer.sh status  
JMX enabled by default  
Using config: /opt/bigdata_root/zookeeper-3.4.5/bin/../conf/zoo.cfg  
Mode: leader
```

```
[root@Master bin]# ./zkServer.sh status
JMX enabled by default
Using config: /opt/bigdata_root/zookeeper-3.4.5/bin/../conf/zoo.cfg
Mode: follower
```

```
[root@Slave2 bin]# ./zkServer.sh status
JMX enabled by default
Using config: /opt/bigdata_root/zookeeper-3.4.5/bin/../conf/zoo.cfg
Mode: follower
```

5.4 部署 Hadoop

<https://www.jianshu.com/p/51bea945e7b7>