

# Hadoop 平台全分布式部署

作者：罗金凯

版本：V1.0

日期：2022.9.17

版本说明

版本	日期	说明
V1.0	2022.9.17	初创

# 目录

1. Hadoop 平台部署方式简介 .....	1
2. 安装器准备.....	1
2.1 平台版本.....	1
2.2 安装前系统配置.....	1
2.2.1 修改主机名.....	1
2.2.2 配置网络.....	2
2.2.3 关闭防火墙.....	3
2.2.4 关闭 selinux 服务.....	4
2.2.5 配置主机名与 ip 地址映射关系.....	4
3. Hadoop 全分布式部署 .....	5
3.1 部署用户说明.....	5
3.2 Hadoop 全分布式部署 .....	5
3.2.1 服务器信息.....	5
3.2.2 配置主机名与 ip 地址映射关系.....	5
3.2.3 上传安装包.....	7
3.2.4 为集群所有节点安装 Java .....	7
3.2.5 设置免密码登录.....	8
3.2.6 安装 Hadoop .....	10
(1) 上传安装包.....	10
(2) 解压安装包.....	10
(3) 配置环境变量.....	10
(4) 将 Master 节点 root 用户配置文件发送到其他节点.....	10
(5) 配置 hadoop-env.sh .....	10
(6) 配置 Master 节点的 core-site.xml .....	11
(7) 配置 Master 节点的 hdfs-site.xml .....	11
(6) 配置 Master 节点的 mapred-site.xml .....	12
(7) 配置 Master 节点的 yarn-site.xml .....	12
(8) 配置 Master 节点的 workers 文件.....	13
(9) 将 Master 节点配置好的安装包发送到其他节点.....	13
(10) 分别在 Master、Slave1 和 Slave2 节点创建相关目录.....	14
(11) 主节点格式化.....	14
(12) 启动集群.....	14
(13) 集群各节点进程状态.....	14
(14) 集群管理系统.....	15
(15) 作业提交集群测试.....	16

## 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. 安装器准备

### 2.1 平台版本

组件	版本
Java	1.8
Hadoop	3.3.4

### 2.2 安装前系统配置

**特别说明：**如果是部署全分布式或 HA 模式的 Hadoop 集群，本节（2.2）中所讲述的内容（2.2.1 – 2.2.5）在集群中的所有节点都要实施。

#### 2.2.1 修改主机名

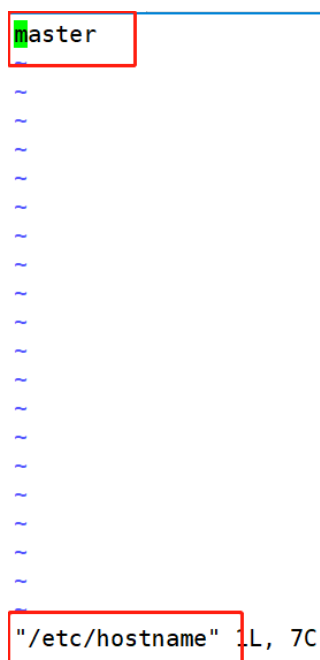
CentOS7 主机名可在 `/etc/hostname` 文件进行修改。步骤如下：

（1）vim 打开 `/etc/hostname`

```
vim /etc/hostname
```

（2）将文件中内容删除，替换为自己要改的主机名

例如，此处我们将主机名修改为 `master`，可以将 `/etc/hostname` 内容设置为如下所示。



(3) 保存退出

(4) 重启系统

## 2.2.2 配置网络

在配置网络前，需要将运行的虚拟机关闭。

为 CentOS 配置 2 张网卡：

第一张网卡为 Host-only 模式，并选择一个虚拟适配器，如下图。



第二张网卡设置为 NAT 模式，如下图所示。



然后根据选定的虚拟网卡，配置静态 ip 即可，此处不赘述。

### 2.2.3 关闭防火墙

步骤如下：

#### (1) 检查防火墙状态

```
systemctl status firewalld
```

如果状态是 “active”，则需要关闭，如下图所示。

```
[root@master ~]# systemctl status firewalld
● firewalld.service - firewalld - dynamic firewall daemon
   Loaded: loaded (/usr/lib/systemd/system/firewalld.service; enabled; vendor pre
   Active: active (running) since Mon 2022-09-05 03:31:58 EDT; 42min ago
     Docs: man:firewalld(1)
    Main PID: 742 (firewalld)
    CGroup: /system.slice/firewalld.service
            └─742 /usr/bin/python2 -Es /usr/sbin/firewalld --nofork --nopid

Sep 05 03:31:54 master systemd[1]: Starting firewalld - dynamic firewall daemon.
Sep 05 03:31:58 master systemd[1]: Started firewalld - dynamic firewall daemon.
[root@master ~]#
```

#### (2) 如果防火墙处于开启状态，则关闭防火墙

```
systemctl stop firewalld
```

#### (3) 禁止防火墙开机启动

```
systemctl disable firewalld
```

```
[root@master ~]# systemctl disable firewalld
Removed symlink /etc/systemd/system/multi-user.target.wants/firewalld.service.
Removed symlink /etc/systemd/system/dbus-org.fedoraproject.FirewallD1.service.
[root@master ~]#
```

#### (4) 检查确认防火墙状态

```
systemctl status firewalld
```

```
[root@master ~]# systemctl status firewalld
● firewalld.service - firewalld - dynamic firewall daemon
   Loaded: loaded (/usr/lib/systemd/system/firewalld.service; disabled; vendor preset: enabled)
   Active: inactive (dead)
     Docs: man:firewalld(1)

Sep 05 03:31:54 master systemd[1]: Starting firewalld - dynamic firewall daemon...
Sep 05 03:31:58 master systemd[1]: Started firewalld - dynamic firewall daemon.
Sep 05 04:15:46 master systemd[1]: Stopping firewalld - dynamic firewall daemon...
Sep 05 04:15:47 master systemd[1]: Stopped firewalld - dynamic firewall daemon.
```

## 2.2.4 关闭 selinux 服务

关闭 selinux 服务步骤如下:

- (1) 查看 selinux 运行状态

执行命令: `getenforce`

```
[root@master ~]# getenforce
Enforcing
[root@master ~]#
```

显示为 Enforcing 状态则需要关闭。

- (2) 打开编辑 `/etc/selinux/config` 文件

`vim /etc/selinux/config`

- (3) 修改配置

```
# This file controls the state of SELinux
# SELINUX= can take one of these three values:
#     enforcing - SELinux security policy is enforced
#     permissive - SELinux prints warnings instead of enforcing
#     disabled  No SELinux policy is loaded
SELINUX=disabled
# SELINUXTYPE= can take one of three values:
#     targeted - Targeted processes are protected
#     minimum - Modification of targeted policy
#     mls - Multi Level Security protection
SELINUXTYPE=targeted
```

- (4) 保存退出文件编辑
- (5) 重启系统
- (6) 查看 selinux 运行状态

```
[root@master ~]# getenforce
Disabled
[root@master ~]#
```

## 2.2.5 配置主机名与 ip 地址映射关系

步骤如下:

(1) 打开/etc/hosts 文件

```
vim /etc/hosts
```

(2) 在文件末尾添加以下内容

```
192.168.56.2    master

127.0.0.1      localhost localhost.localdomain
::1           localhost localhost.localdomain

192.168.56.2    master
```

(3) 保存退出

### 3. Hadoop 全分布式部署

#### 3.1 部署用户说明

本节中所讲述的 Hadoop 全分布式集群部署，将统一使用 root 用户进行操作。

#### 3.2 Hadoop 全分布式部署

##### 3.2.1 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	192.168.56.2	Master
Slave1	192.168.56.3	Slave1
Slave2	192.168.56.4	Slave2

##### 3.2.2 配置主机名与 ip 地址映射关系

配置主机名与 ip 地址映射关系步骤如下：

(1) 打开 Master 节点的/etc/hosts 文件

```
vim /etc/hosts
```

(2) 在文件末尾添加以下内容

```
192.168.56.2    Master
192.168.56.3    Slave1
```



192.168.56.4    Slave2

```
127.0.0.1    localhost localhost.
::1         localhost localhost.

192.168.56.2    Master
192.168.56.3    Slave1
192.168.56.4    Slave2
```

(3) 保存退出

(4) 将 Master 节点上的映射文件发送到集群其他所有节点 (Slave1、Slave2)

```
scp /etc/hosts Slave1:/etc/
```

```
scp /etc/hosts Slave2:/etc/
```

```
[root@Master ~]# scp /etc/hosts Slave1:/etc/
The authenticity of host 'slave1 (192.168.56.3)' can't be established.
ECDSA key fingerprint is SHA256:RHpPkL80A+fiN34N5Cj47lXH6lH6VlnCWcHPxd+zH4.
ECDSA key fingerprint is MD5:28:9e:3b:0c:7f:77:25:c0:a9:1c:71:e2:55:66:03:4c.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'slave1,192.168.56.3' (ECDSA) to the list of known hosts.
root@slave1's password:
hosts                                     100% 219   221.3KB/s   00:00
[root@Master ~]#
```

```
[root@Master ~]# scp /etc/hosts Slave2:/etc/
The authenticity of host 'slave2 (192.168.56.4)' can't be established.
ECDSA key fingerprint is SHA256:Nho5qNnsQsEUKTAPSS7Fksoz1aIHUHLI73XNiK4q8JM.
ECDSA key fingerprint is MD5:46:c4:83:39:de:1f:a1:bc:a3:0d:d4:98:2c:0b:e6:f3.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'slave2,192.168.56.4' (ECDSA) to the list of known hosts.
root@slave2's password:
hosts                                     100% 219   55.3KB/s   00:00
[root@Master ~]#
```

(5) 分别打开 Slave1 和 Slave2 节点的/etc/hosts 文件，确认内容是否正确

```
[root@Slave1 ~]# cat /etc/hosts
```

```
[root@Slave1 ~]# cat /etc/hosts
127.0.0.1    localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6

192.168.56.2    Master
192.168.56.3    Slave1
192.168.56.4    Slave2
[root@Slave1 ~]#
```

```
[root@Slave2 ~]# cat /etc/hosts
```

```
[root@Slave2 ~]# cat /etc/hosts
127.0.0.1    localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6

192.168.56.2    Master
192.168.56.3    Slave1
192.168.56.4    Slave2
[root@Slave2 ~]#
```

### 3.2.3 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]#
```

### 3.2.4 为集群所有节点安装 Java

**特别说明：** 集群所有节点都需要安装 JDK。

（1）在 Master 节点上解压安装 JDK 并配置环境变量

打开环境变量文件，将以下内容配置到文件中。

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

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
```

对环境变量生效，手动使用 source 命令执行环境变量文件。

```
[root@Master root]source ~/.bash-profile
```

然后即可验证 Master 节点上 Java 是否成功，在命令行输入 “java -  
version” 命令，确认是否可以查看 java 版本。

```
[root@Master root]java -version
```

```
[root@Master bigdata_root]# 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@Master bigdata_root]#
```

如果查看 Java 版本提示 “java 命令找不到” 则需要检查 Java 环境变量配置。

（2）将 Master 节点的 Java 整个目录分别发送到 Slave1 和 Slave2 节点

```
scp -r jdk1.8.0_191/ Slave1:/opt/bigdata-root/
```

```
scp -r jdk1.8.0_191/ Slave2:/opt/bigdata-root/
```

### (3) 分别配置 Slave1 和 Slave2 节点的 Java 环境变量

此处有 2 种操作方法，一是复制 Master 节点上环境变量内容，分别打开 Slave1 和 Slave2 节点的环境变量配置文件，并将复制的内容粘贴到其中；二是可以直接将 Master 节点的环境变量配置文件发送到 Slave1 和 Slave2 节点。（由于 3 个节点安装的相同版本的系统，软件等环境相同，因此采用第二种方式会更便捷）

在 Master 上执行以下操作，将配置文件发送到 Slave1 和 Slave2。

```
[root@Slave1 opt]# scp ~/.bash_profile Slave1:/root/
```

```
[root@Slave1 opt]# scp ~/.bash_profile Slave2:/root/
```

```
[root@Master bigdata_root]# scp ~/.bash_profile Slave1:/root/
root@slave1's password:
.bash_profile
100% 327 210.4KB/s 00:00
[root@Master bigdata_root]#

[root@Master bigdata_root]# scp ~/.bash_profile Slave2:/root/
root@slave2's password:
.bash_profile
100% 327 215.4KB/s 00:00
[root@Master bigdata_root]#
```

然后分别将 Slave1 和 Slave2 节点上的配置文件生效。

在 Slave1 节点执行: [root@Slave1 opt]# source ~/.bash\_profile

在 Slave2 节点执行: [root@Slave2 opt]# source ~/.bash\_profile

最后，分别在 Slave1 和 Slave2 节点上查看 java 版本信息。

```
[root@Slave1 opt]# 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@Slave1 opt]#
```

```
[root@Slave2 bigdata_root]# 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@Slave2 bigdata_root]#
```

至此，集群三台节点的 JDK 安装配置结束。

### 3.2.5 设置免密码登录

#### (1) 分别在 Master、Slave1 和 Slave2 节点生成密钥对

在 Master 节点生成密钥对:

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

在 Slave1 节点生成密钥对:

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

在 Slave2 节点生成秘钥对:

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

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

在 Master 节点上执行以下操作:

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

在 Slave1 节点上执行以下操作:

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

在 Slave2 节点上执行以下操作:

```
[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。特别说明: 如果节点或者节点之间在此前没有进行过 ssh 登录, 第一次进行登录时需要输入一个 “yes”。

从 Master 节点分别远程登录到 Slave1 和 Slave2 节点:

```
[root@Master .ssh]# ssh Slave1
```

```
[root@Master .ssh]# ssh Slave2
```

```
[root@Master .ssh]# ssh Slave1
Last login: Sun Sep 18 06:50:52 2022 from master
[root@Slave1 ~]#
```

```
[root@Master .ssh]# ssh Slave2
Last login: Sun Sep 18 05:08:07 2022 from 192.168.56.1
[root@Slave2 ~]#
```

从 Slave1 节点分别远程登录到 Master 和 Slave2 节点:

```
[root@Slave1 .ssh]# ssh Master
```

```
[root@Slave1 .ssh]# ssh Slave2
```

```
[root@Slave1 .ssh]# ssh Master
Last login: Sun Sep 18 06:22:57 2022 from slave1
[root@Master ~]#
```

```
[root@Slave1 .ssh]# ssh Slave2
Last login: Sun Sep 18 06:52:39 2022 from master
[root@Slave2 ~]#
```

从 Slave2 节点分别远程登录到 Master 和 Slave1 节点:

```
[root@Slave2 .ssh]# ssh Master
```

```
[root@Slave2 .ssh]# ssh Slave1
```

```
[root@Slave2 .ssh]# ssh Master
Last login: Sun Sep 18 06:30:43 2022 from slave2
[root@Master ~]#
```

```
[root@Slave2 .ssh]# ssh Slave1
The authenticity of host 'slave1 (192.168.56.3)' can't be established.
ECDSA key fingerprint is SHA256:RHpPkL80A+fiN34N5Cj47LXH6LH6VllnCwCHPxd+zh4.
ECDSA key fingerprint is MD5:28:9e:3b:0c:7f:77:25:c0:a9:1c:71:e2:55:66:03:4c.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'slave1,192.168.56.3' (ECDSA) to the list of known hosts.
Last login: Sun Sep 18 06:51:48 2022 from master
[root@Slave1 ~]#
```

### 3.2.6 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-3.3.4.tar.gz
```

#### (3) 配置环境变量

```
[root@Master hadoop-3.3.4]# vim ~/.bash_profile
```

```
export HADOOP_HOME=/opt/bigdata-root/hadoop-3.3.4
```

```
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
```

```
export HADOOP_HOME=/opt/bigdata_root/hadoop-3.3.4
```

Hadoop3.0 版后，对每一个模块的操作用户做了严格限定，Hadoop2.X 版本中默认是设定为 root 用户，而 Hadoop3.X 版本可以允许用户进行设置，在 hadoop-env.sh 文件中再添加以下语句。

```
export HDFS_NAMENODE_USER=root
export HDFS_SECONDARYNAMENODE_USER=root
export HDFS_DATANODE_USER=root
export YARN_RESOURCEMANAGER_USER=root
export YARN_NODEMANAGER_USER=root
```

```
export HDFS_NAMENODE_USER=root
export HDFS_SECONDARYNAMENODE_USER=root
export HDFS_DATANODE_USER=root
export YARN_RESOURCEMANAGER_USER=root
export YARN_NODEMANAGER_USER=root

###
# Registry DNS specific parameters
###
# For privileged registry DNS, user to run as after dropping privileges
# This will replace the hadoop.id.str Java property in secure mode.
# export HADOOP_REGISTRYDNS_SECURE_USER=yarn

# Supplemental options for privileged registry DNS
# By default, Hadoop uses jsvc which needs to know to launch a
# server jvm.
# export HADOOP_REGISTRYDNS_SECURE_EXTRA_OPTS="-jvm server"
export JAVA_HOME=/opt/bigdata_root/jdk1.8.0_191
export HADOOP_HOME=/opt/bigdata_root/hadoop-3.3.4
```

保存退出即可，此文件不需要使用 source 命令执行。

## (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>dfs.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>
```

**特别说明：**在 Hadoop2.X 版本中，文件备份参数是 “file.replication”，而在 Hadoop3.X 版本中统一更改为 “dfs.replication”。

#### (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 workers 文件

\$HADOOP\_HOME/etc/hadoop/目录下有一个 workers 文件，专门用于配置 Hadoop 集群的工作节点列表，需要将整个集群中作为工作节点的主机名配置到该文件中。

使用 vim 命令打开 \$HADOOP\_HOME/etc/hadoop/workers 文件，将以下内容替换原来的内容。

Master

Slave1

Slave2

**特别说明：**Hadoop2.X 版本中，该文件名字为 “slaves”，而在 Hadoop3.X 版本中，该文件统一更名为 “workers”。

## (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。

从 Master 节点将配置好的 Hadoop 整个目录发送到 Slave1 节点：

```
[root@Master bigdata-root]# scp -r hadoop-3.3.4
```

Slave1: /opt/bigdata-root/

```
[root@Master bigdata_root]# ll
total 866440
drwxr-xr-x 11 1024 1024      227 Sep  6 23:22 hadoop-3.3.4
-rw-r--r--  1 root root 695457782 Aug 28 09:40 hadoop-3.3.4.tar.gz
drwxr-xr-x  7  10 143      245 Oct  6 2018 jdk1.8.0_191
-rw-r--r--  1 root root 191753373 Nov 13 2018 jdk-8u191-linux-x64.tar.gz
-rw-r--r--  1 root root 15752 Jun  1 04:29 package-lock.json
-rw-r--r--  1 root root 65 Jun 30 07:48 tk.csv
[root@Master bigdata_root]# scp -r hadoop-3.3.4 Slave1:/opt/bigdata_root/
```

从 Master 节点将配置好的 Hadoop 整个目录发送到 Slave2 节点：

```
[root@Master bigdata-root]# scp -r hadoop-3.3.4
```

Slave2: /opt/bigdata-root/



### (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) 主节点格式化

在 Master 节点上执行格式化操作

```
[root@Master hadoop]hdfs namenode -format
```

### (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) 集群各节点进程状态

分别在 Master、Slave1 和 Slave3 三个节点上执行 “jps” 命令, 查看各个节点有哪些进程存在, 并解释各节点存在查询到的进程的原因, 此小节作为思考, 后续讲解补充。

## (14) 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。

### A. HDFS 集群

访问 <http://MASTER-IP:9870>

#### ● 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).

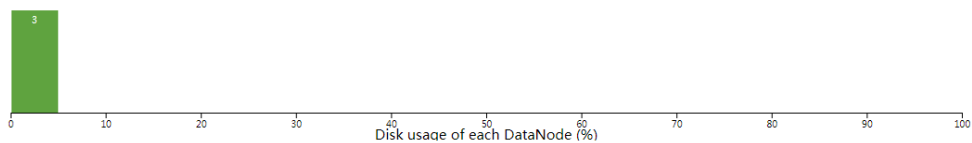
Heap Memory used 38.83 MB of 46.38 MB Heap Memory. Max Heap Memory is 966.69 MB.

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

## B. 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	3	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	3	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

## (15) 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-3.3.4
/share/hadoop/mapreduce/hadoop-mapreduce-examples-3.3.4.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	3	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

