

GitChat·大数据 | 史上最详细的Hadoop环境搭建

 blog.csdn.net/GitChat/article/details/77849331

GitChat 作者：鸣宇淳

原文：[史上最详细的Hadoop环境搭建](#)

关注公众号：GitChat 技术杂谈，一本正经的讲技术

【不要错过文末彩蛋】

前言

Hadoop在大数据技术体系中的地位至关重要，Hadoop是大数据技术的基础，对Hadoop基础知识的掌握的扎实程度，会决定在大数据技术道路上走多远。

这是一篇入门文章，Hadoop的学习方法很多，网上也有很多学习路线图。本文的思路是：以安装部署Apache Hadoop2.x版本为主线，来介绍Hadoop2.x的架构组成、各模块协同工作原理、技术细节。**安装不是目的，通过安装认识Hadoop才是目的。**

本文分为五个部分、十三节、四十九步。

第一部分：Linux环境安装

Hadoop是运行在Linux，虽然借助工具也可以运行在Windows上，但是建议还是运行在Linux系统上，第一部分介绍Linux环境的安装、配置、Java JDK安装等。

第二部分：Hadoop本地模式安装

Hadoop本地模式只是用于本地开发调试，或者快速安装体验Hadoop，这部分做简单的介绍。

第三部分：Hadoop伪分布式模式安装

学习Hadoop一般是在伪分布式模式下进行。这种模式是在一台机器上各个进程上运行Hadoop的各个模块，伪分布式的意思是虽然各个模块是在各个进程上分开运行的，但是只是运行在一个操作系统上的，并不是真正的分布式。

第四部分：完全分布式安装

完全分布式模式才是生产环境采用的模式，Hadoop运行在服务器集群上，生产环境一般都会做HA，以实现高可用。

第五部分：Hadoop HA安装

HA是指高可用，为了解决Hadoop单点故障问题，生产环境一般都做HA部署。这部分介绍了如何配置Hadoop2.x的高可用，并简单介绍了HA的工作原理。

安装过程中，会穿插简单介绍涉及到的知识。希望能对大家有所帮助。

第一部分：Linux环境安装

第一步、配置Vmware NAT网络

一、Vmware网络模式介绍

参考：<http://blog.csdn.net/collection4u/article/details/14127671>

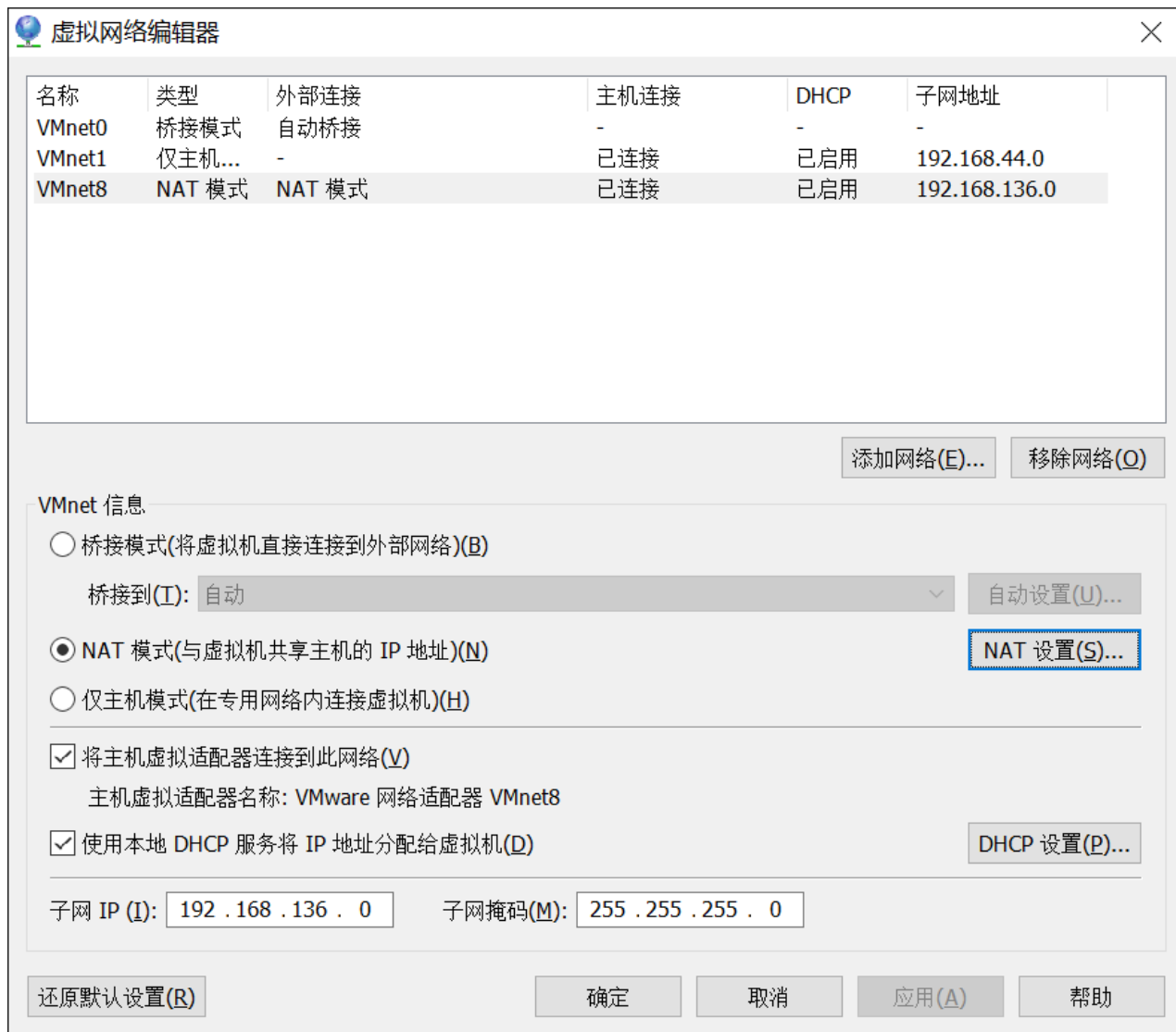
二、NAT模式配置

NAT是网络地址转换，是在宿主机和虚拟机之间增加一个地址转换服务，负责外部和虚拟机之间的通讯转接和IP转换。

我们部署Hadoop集群，这里选择NAT模式，各个虚拟机通过NAT使用宿主机的IP来访问外网。

我们的要求是集群中的各个虚拟机有固定的IP、可以访问外网，所以进行如下设置：

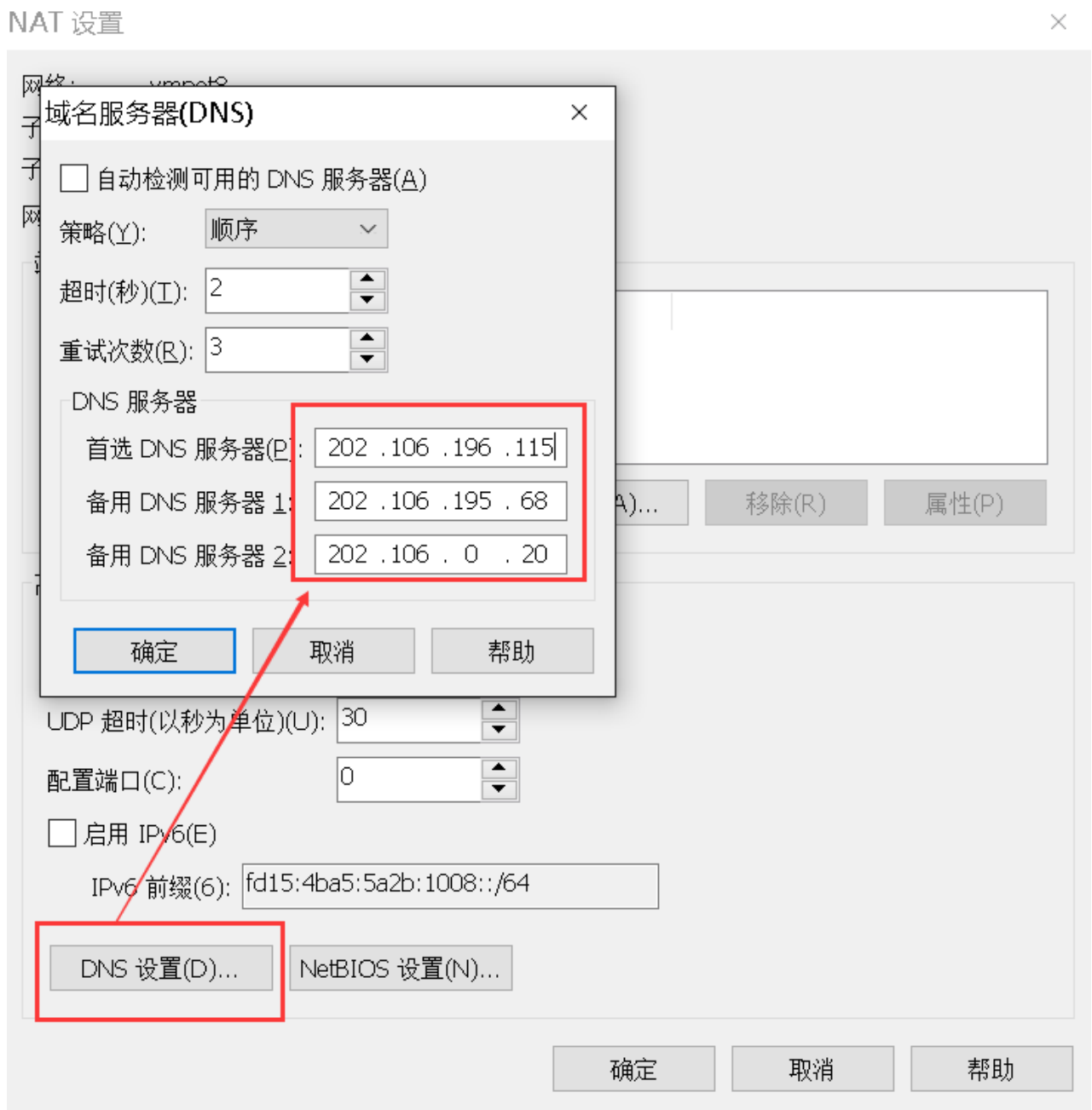
1、Vmware安装后，默认的NAT设置如下：



2、默认的设置是启动DHCP服务的，NAT会自动给虚拟机分配IP，但是我们需要将各个机器的IP固定下来，所以要取消这个默认设置。

3、为机器设置一个子网网段，默认是192.168.136网段，我们这里设置为100网段，将来各个虚拟机Ip就为 192.168.100.*。

4、点击NAT设置按钮，打开对话框，可以修改网关地址和DNS地址。这里我们为NAT指定DNS地址。



5、网关地址为当前网段里的.2地址，好像是固定的，我们不做修改，先记住网关地址就好了，后面会用到。

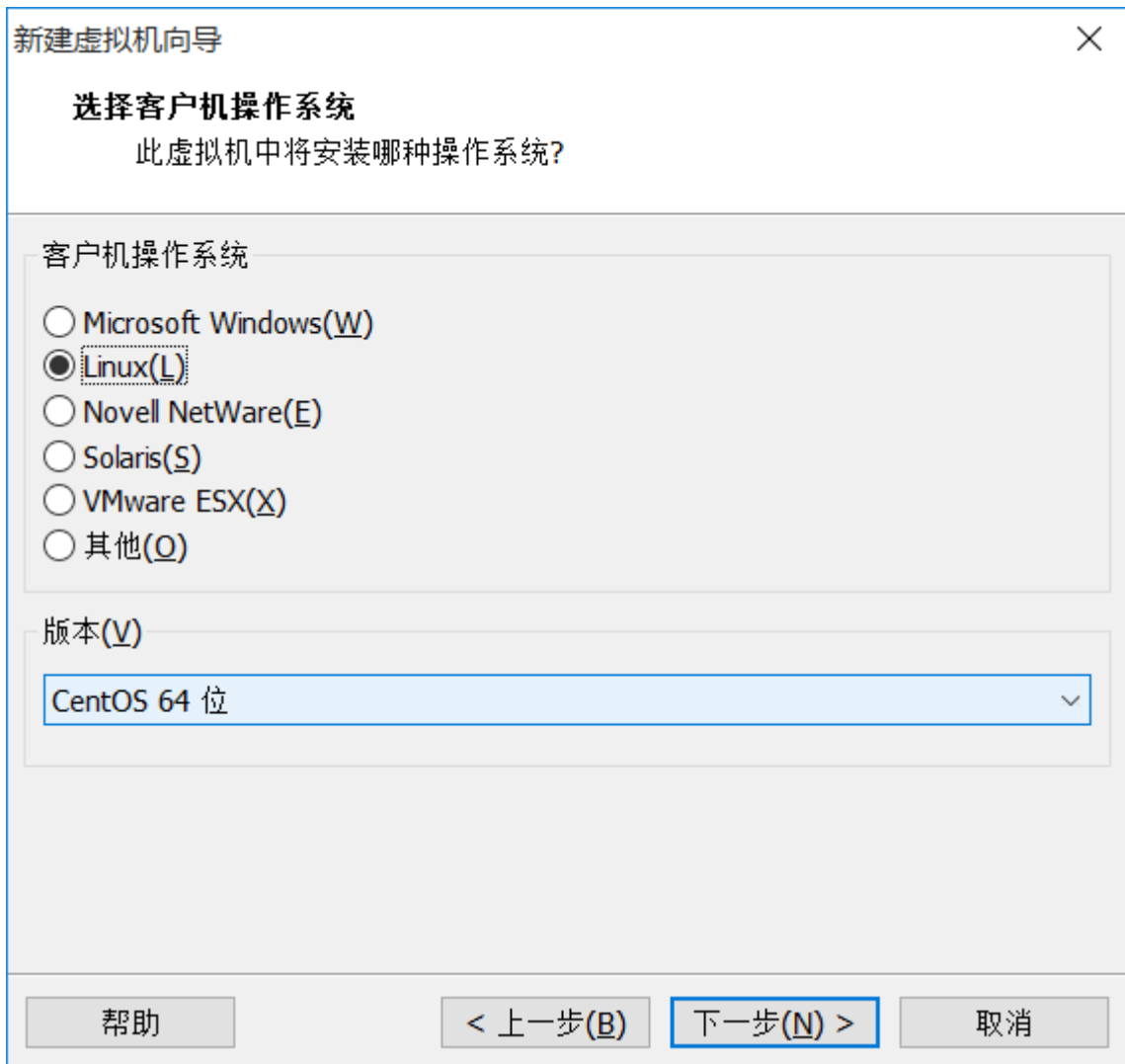
第二步、安装Linux操作系统

三、Vmware上安装Linux系统

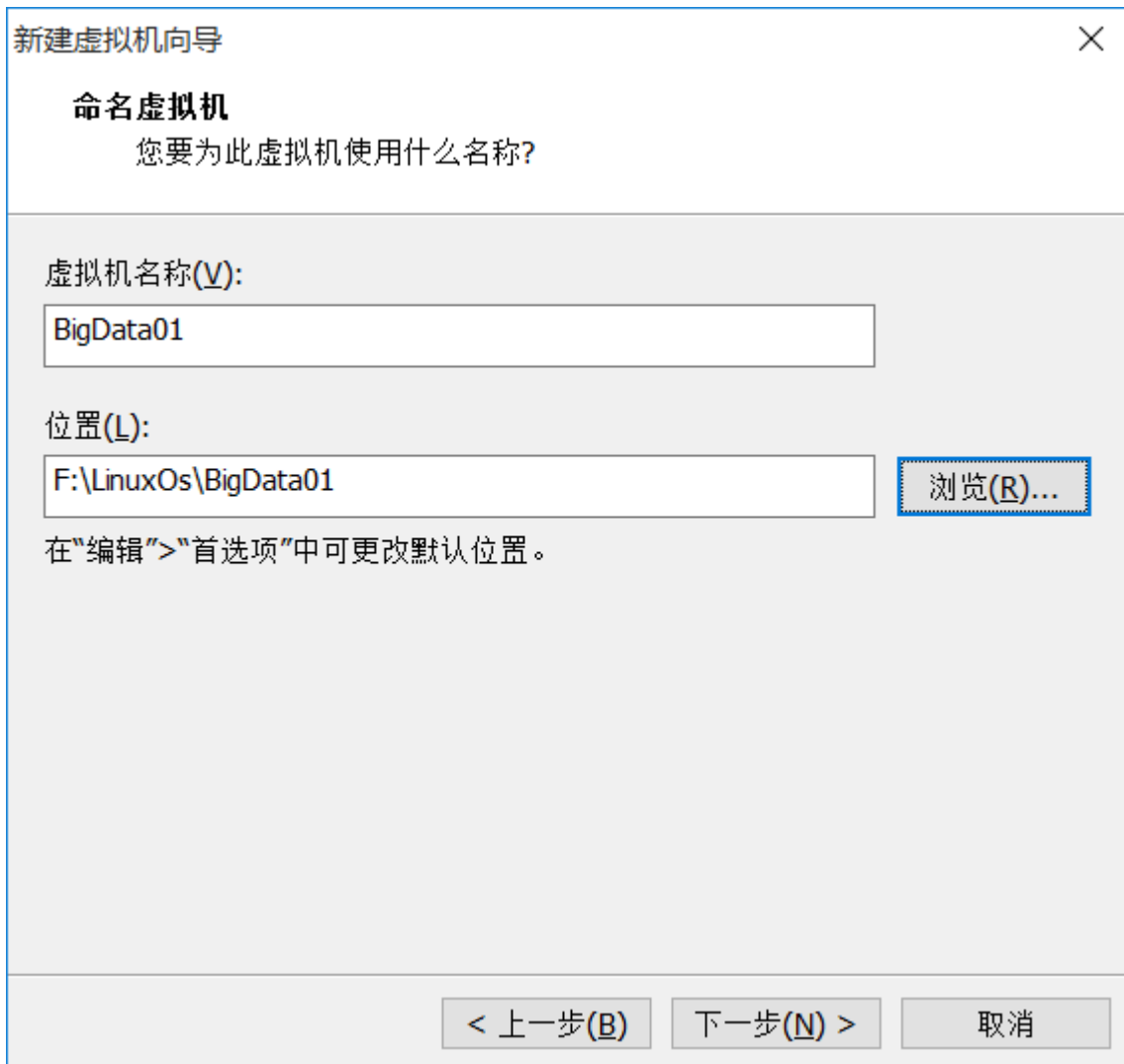
1、文件菜单选择新建虚拟机

2、选择经典类型安装，下一步。

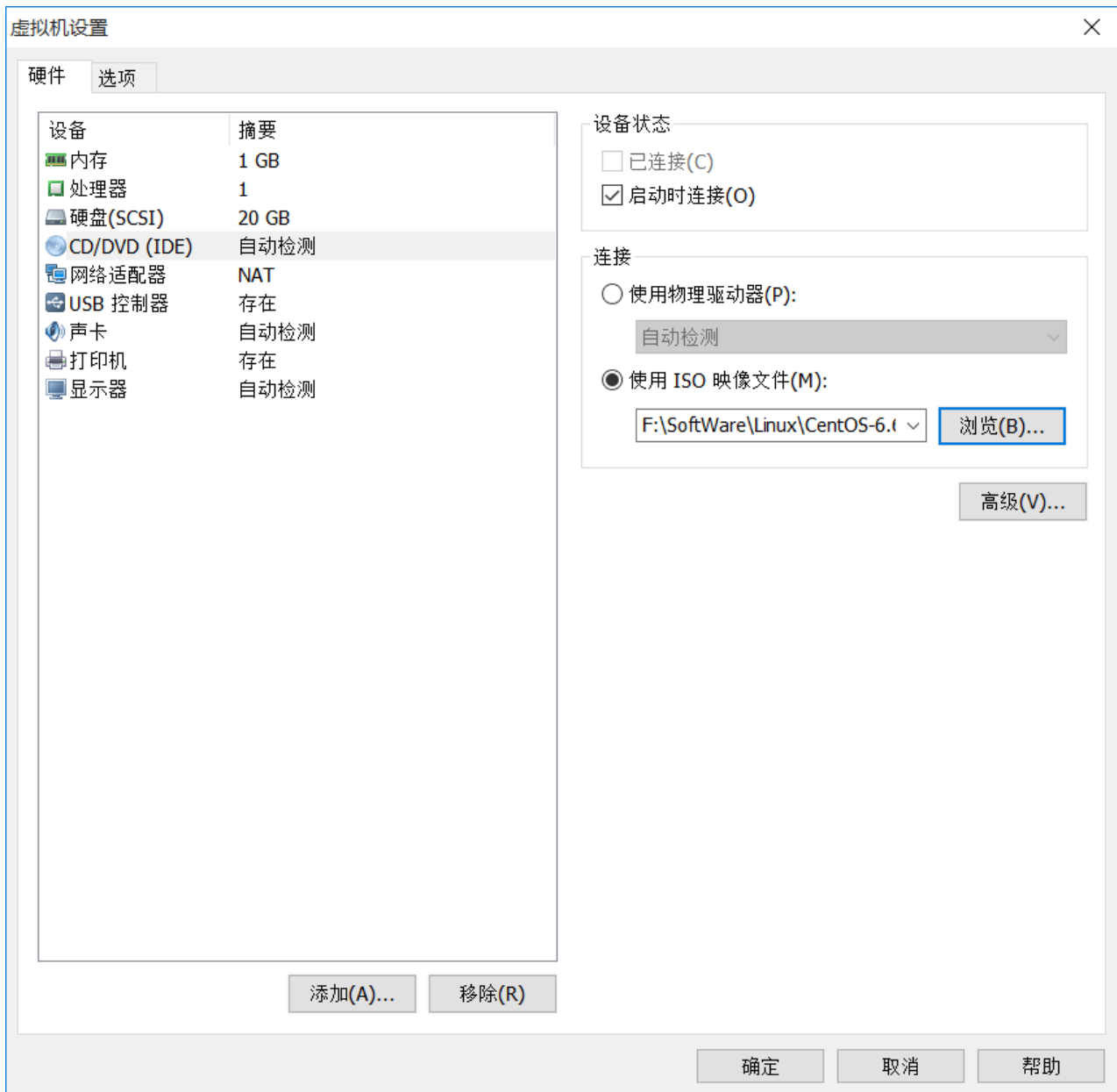
- 3、选择稍后安装操作系统，下一步。
- 4、选择Linux系统，版本选择CentOS 64位。



- 5、命名虚拟机，给虚拟机起个名字，将来显示在Vmware左侧。并选择Linux系统保存在宿主机的哪个目录下，应该一个虚拟机保存在一个目录下，不能多个虚拟机使用一个目录。



- 6、指定磁盘容量，是指定分给Linux虚拟机多大的硬盘，默认20G就可以，下一步。
- 7、点击自定义硬件，可以查看、修改虚拟机的硬件配置，这里我们不做修改。
- 8、点击完成后，就创建了一个虚拟机，但是此时的虚拟机还是一个空壳，没有操作系统，接下来安装操作系统。
- 9、点击编辑虚拟机设置，找到DVD，指定操作系统ISO文件所在位置。



10、 单击开启此虚拟机，选择第一个回车开始安装操作系统。

Welcome to CentOS 6.6!

```
Install or upgrade an existing system
Install system with basic video driver
Rescue installed system
Boot from local drive
Memory test
```

Press [Tab] to edit options

CentOS 6
Community ENTERprise Operating System



11、设置root密码。



The root account is used for administering the system. Enter a password for the root user.

Root Password:

Confirm:

Back

Next

12、选择Desktop，这样就会装一个Xwindow。

The default installation of CentOS is a minimum install. You can optionally select a different set of software now.

- Desktop
- Minimal Desktop
- Minimal
- Basic Server
- Database Server
- Web Server
- Virtual Host
- Software Development Workstation

Please select any additional repositories that you want to use for software installation.

- CentOS

Add additional software repositories

Modify repository

You can further customize the software selection now, or after install via the software management application.

Customize later Customize now

Back

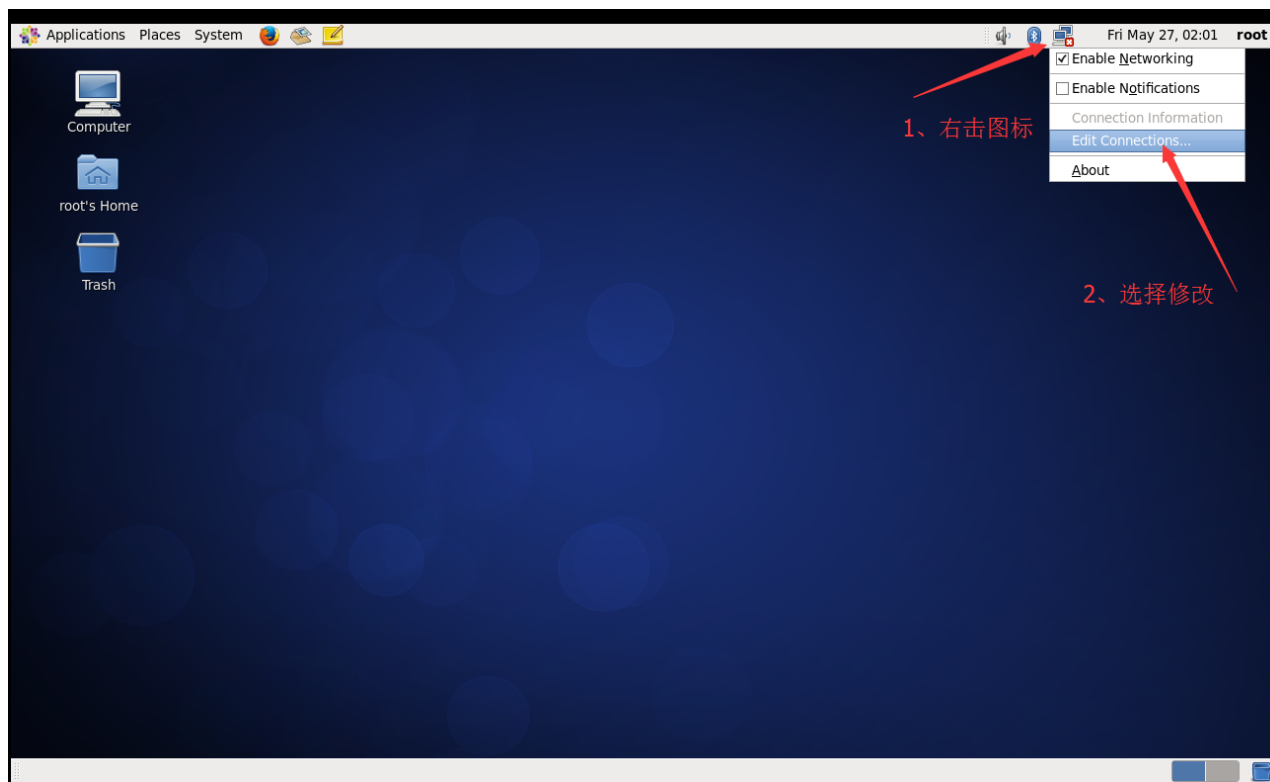
Next

13、先不添加普通用户，其他用默认的，就把Linux安装完毕了。

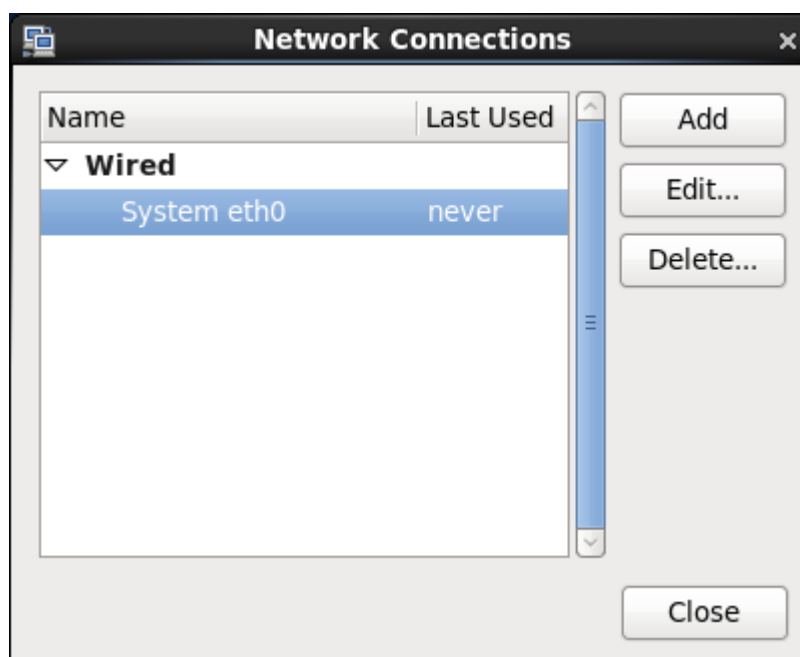
四、设置网络

因为Vmware的NAT设置中关闭了DHCP自动分配IP功能，所以Linux还没有IP，需要我们设置网络各个参数。

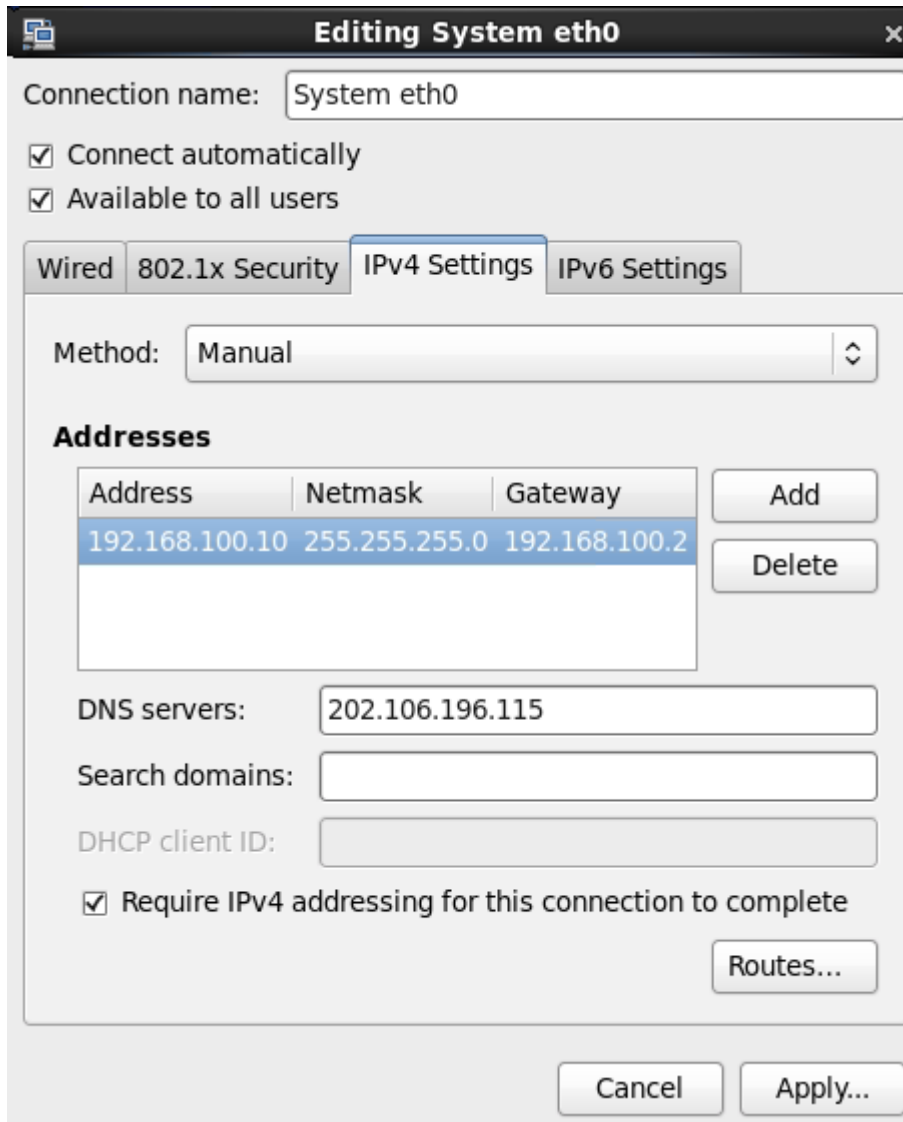
1、用root进入Xwindow，右击右上角的网络连接图标，选择修改连接。



2、网络连接里列出了当前Linux里所有的网卡，这里只有一个网卡System eth0，点击编辑。



3、配置IP、子网掩码、网关（和NAT设置的一样）、DNS等参数，因为NAT里设置网段为100.*，所以这台机器可以设置为192.168.100.10网关和NAT一致，为192.168.100.2



4、用ping来检查是否可以连接外网，如下图，已经连接成功。

```
root@localhost:~/Desktop
File Edit View Search Terminal Help
[root@localhost Desktop]# ping www.baidu.com
PING www.a.shifen.com (61.135.169.125) 56(84) bytes of data.
64 bytes from 61.135.169.125: icmp_seq=1 ttl=128 time=5.56 ms
64 bytes from 61.135.169.125: icmp_seq=2 ttl=128 time=3.32 ms
64 bytes from 61.135.169.125: icmp_seq=3 ttl=128 time=2.59 ms
64 bytes from 61.135.169.125: icmp_seq=4 ttl=128 time=2.92 ms
64 bytes from 61.135.169.125: icmp_seq=5 ttl=128 time=2.37 ms
64 bytes from 61.135.169.125: icmp_seq=6 ttl=128 time=4.87 ms
64 bytes from 61.135.169.125: icmp_seq=7 ttl=128 time=2.72 ms
64 bytes from 61.135.169.125: icmp_seq=8 ttl=128 time=3.07 ms
^C
--- www.a.shifen.com ping statistics ---
8 packets transmitted, 8 received, 0% packet loss, time 7956ms
rtt min/avg/max/mdev = 2.378/3.432/5.562/1.081 ms
[root@localhost Desktop]# ^C
[root@localhost Desktop]# █
```

五、修改Hostname

1、临时修改hostname

```
[root@localhost Desktop]# hostname bigdata-senior01.chybinmy.com
1
```

这种修改方式，系统重启后就会失效。

2、永久修改hostname

想永久修改，应该修改配置文件 /etc/sysconfig/network。

```
命令:[root@bigdata-senior01 ~] vim /etc/sysconfig/network
1
```

打开文件后,

```
NETWORKING=yes #使用网络
HOSTNAME=bigdata-senior01.chybinmy.com #设置主机名
• 1
• 2
```

六、配置Host

```
命令:[root@bigdata-senior01 ~] vim /etc/hosts
添加hosts: 192.168.100.10 bigdata-senior01.chybinmy.com
• 1
• 2
```

七、关闭防火墙

学习环境可以直接把防火墙关闭掉。

(1) 用root用户登录后, 执行查看防火墙状态。

```
[root@bigdata-senior01 hadoop]# service iptables status
1
```

(2) 用[root@bigdata-senior01 hadoop]# service iptables stop关闭防火墙, 这个是临时关闭防火墙。

```
[root@bigdata-senior01 hadoop-2.5.0]# service iptables stop
iptables: Setting chains to policy ACCEPT: filter          [ OK ]
iptables: Flushing firewall rules:                        [ OK ]
iptables: Unloading modules:                              [ OK ]
  • 1
  • 2
  • 3
  • 4
```

(3) 如果要永久关闭防火墙用。

```
[root@bigdata-senior01 hadoop]# chkconfig iptables off
1
```

关闭, 这种需要重启才能生效。

八、关闭selinux

selinux是Linux一个子安全机制, 学习环境可以将它禁用。

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ vim /etc/sysconfig/selinux
1
```

```
# This file controls the state of SELinux on the system.
# 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 these two values:
#   targeted - Targeted processes are protected,
#   mls - Multi Level Security protection.
SELINUXTYPE=targeted
  • 1
  • 2
  • 3
  • 4
  • 5
  • 6
  • 7
  • 8
  • 9
  • 10
```

```
# This file controls the state of SELinux on the system.
# 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 these two values:
#     targeted - Targeted processes are protected,
#     mls - Multi Level Security protection.
SELINUXTYPE=targeted
```

第三步、安装JDK

九、安装Java JDK

1、查看是否已经安装了java JDK。

```
[root@bigdata-senior01 Desktop]# java -version
1
```

注意：Hadoop机器上的JDK，最好是Oracle的Java JDK，不然会有一些问题，比如可能没有JPS命令。

如果安装了其他版本的JDK，卸载掉。

2、安装java JDK

(1) 去下载Oracle版本Java JDK：jdk-7u67-linux-x64.tar.gz

(2) 将jdk-7u67-linux-x64.tar.gz解压到/opt/modules目录下

```
[root@bigdata-senior01 /]# tar -zxvf jdk-7u67-linux-x64.tar.gz -C /opt/modules
1
```

(3) 添加环境变量

设置JDK的环境变量 JAVA_HOME。需要修改配置文件/etc/profile，追加

```
export JAVA_HOME="/opt/modules/jdk1.7.0_67"
export PATH=$JAVA_HOME/bin:$PATH
• 1
• 2
```

修改完毕后，执行 source /etc/profile

(4)安装后再次执行 java -version,可以看见已经安装完成。

```
[root@bigdata-senior01 /]# java -version
java version "1.7.0_67"
Java(TM) SE Runtime Environment (build 1.7.0_67-b01)
Java HotSpot(TM) 64-Bit Server VM (build 24.65-b04, mixed mode)
  • 1
  • 2
  • 3
  • 4
```

第二部分：Hadoop本地模式安装

第四步、Hadoop部署模式

Hadoop部署模式有：本地模式、伪分布模式、完全分布式模式、HA完全分布式模式。

区分的依据是NameNode、DataNode、ResourceManager、NodeManager等模块运行在几个JVM进程、几个机器上。

| 模式名称 | 各个模块占用的JVM进程数 | 各个模块运行在几个机器数上 |
|---------|---------------|---------------|
| 本地模式 | 1个 | 1个 |
| 伪分布式模式 | N个 | 1个 |
| 完全分布式模式 | N个 | N个 |
| HA完全分布式 | N个 | N个 |

第五步、本地模式部署

十、本地模式介绍

本地模式是最简单的模式，所有模块都运行与一个JVM进程中，使用的本地文件系统，而不是HDFS，本地模式主要是用于本地开发过程中的运行调试用。下载hadoop安装包后不用任何设置，默认的就是本地模式。

十一、解压hadoop后就是直接可以使用

1、创建一个存放本地模式hadoop的目录

```
[hadoop@bigdata-senior01 modules]$ mkdir /opt/modules/hadoopstandalone
1
```

2、解压hadoop文件

```
[hadoop@bigdata-senior01 modules]$ tar -zxf /opt/sofeware/hadoop-2.5.0.tar.gz -C /opt/modules/hadoopstandalone/
1
```

3、确保JAVA_HOME环境变量已经配置好

```
[hadoop@bigdata-senior01 modules]$ echo ${JAVA_HOME}
/opt/modules/jdk1.7.0_67
  • 1
  • 2
```

十二、运行MapReduce程序, 验证

我们这里用hadoop自带的wordcount例子来在本地模式下测试跑mapreduce。

1、准备mapreduce输入文件wc.input

```
[hadoop@bigdata-senior01 modules]$ cat /opt/data/wc.input
hadoop mapreduce hive
hbase spark storm
sqoop hadoop hive
spark hadoop
  • 1
  • 2
  • 3
  • 4
  • 5
```

2、运行hadoop自带的mapreduce Demo

```
[hadoop@bigdata-senior01 hadoopstandalone]$ bin/hadoop jar
share/hadoop/mapreduce/hadoop-mapreduce-examples-2.5.0.jar wordcount
/opt/data/wc.input output2
  1
```

```
[hadoop@bigdata-senior01 hadoopstandalone]$ bin/hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examp
data/wc.input output2
16/07/07 12:50:05 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... u
ere applicable
16/07/07 12:50:05 INFO Configuration.deprecation: session.id is deprecated. Instead, use dfs.metrics.sess
16/07/07 12:50:05 INFO jvm.JvmMetrics: Initializing JVM Metrics with processName=JobTracker, sessionId=
16/07/07 12:50:05 INFO input.FileInputFormat: Total input paths to process : 1
16/07/07 12:50:05 INFO mapreduce.JobSubmitter: number of splits:1
16/07/07 12:50:06 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_local1085723536_0001
16/07/07 12:50:06 WARN conf.Configuration: file:/tmp/hadoop-hadoop/mapred/staging/hadoop1085723536/.stagi
job.xml:an attempt to override final parameter: mapreduce.job.end-notification.max.retry.interval; Ignor
16/07/07 12:50:06 WARN conf.Configuration: file:/tmp/hadoop-hadoop/mapred/staging/hadoop1085723536/.stagi
job.xml:an attempt to override final parameter: mapreduce.job.end-notification.max.attempts; Ignoring.
16/07/07 12:50:06 WARN conf.Configuration: file:/tmp/hadoop-hadoop/mapred/local/localRunner/hadoop/job_lo
1085723536_0001.xml:an attempt to override final parameter: mapreduce.job.end-notification.max.retry.inte
16/07/07 12:50:06 WARN conf.Configuration: file:/tmp/hadoop-hadoop/mapred/local/localRunner/hadoop/job_lo
1085723536_0001.xml:an attempt to override final parameter: mapreduce.job.end-notification.max.attempts;
16/07/07 12:50:06 INFO mapreduce.Job: The url to track the job: http://localhost:8080/
16/07/07 12:50:06 INFO mapreduce.Job: Running job: job_local1085723536_0001
16/07/07 12:50:06 INFO mapred.LocalJobRunner: OutputCommitter set in config null
16/07/07 12:50:06 INFO mapred.LocalJobRunner: OutputCommitter is org.apache.hadoop.mapreduce.lib.output.F
16/07/07 12:50:06 INFO mapred.LocalJobRunner: Waiting for map tasks
16/07/07 12:50:06 INFO mapred.LocalJobRunner: Starting task: attempt_local1085723536_0001_m_000000_0
16/07/07 12:50:06 INFO mapred.Task: Using ResourceCalculatorProcessImpl as the
```

这里可以看到job ID中有local字样，说明是运行在本地模式下的。

3、查看输出文件

本地模式下，mapreduce的输出是输出到本地。

```
[hadoop@bigdata-senior01 hadoopstandalone]$ ll output2
total 4
-rw-r--r-- 1 hadoop hadoop 60 Jul  7 12:50 part-r-00000
-rw-r--r-- 1 hadoop hadoop  0 Jul  7 12:50 _SUCCESS
  • 1
  • 2
  • 3
  • 4
```

输出目录中有_SUCCESS文件说明JOB运行成功，part-r-00000是输出结果文件。

第三部分：Hadoop伪分布式模式安装

第六步、伪分布式Hadoop部署过程

十三、Hadoop所用的用户设置

1、创建一个名字为hadoop的普通用户

```
[root@bigdata-senior01 ~]# useradd hadoop
[root@bigdata-senior01 ~]# passwd hadoop
  • 1
  • 2
```

2、给hadoop用户sudo权限

```
[root@bigdata-senior01 ~]# vim /etc/sudoers
  1
```

设置权限，学习环境可以将hadoop用户的权限设置的大一些，但是生产环境一定要注意普通用户的权限限制。

```
root    ALL=(ALL)        ALL
hadoop  ALL=(root) NOPASSWD:ALL
  • 1
  • 2
```

注意：如果root用户无权修改sudoers文件，先手动为root用户添加写权限。

```
[root@bigdata-senior01 ~]# chmod u+w /etc/sudoers
  1
```

3、切换到hadoop用户

```
[root@bigdata-senior01 ~]# su - hadoop
[hadoop@bigdata-senior01 ~]$
  • 1
  • 2
```


4、创建存放hadoop文件的目录

```
[hadoop@bigdata-senior01 ~]$ sudo mkdir /opt/modules  
1
```

5、将hadoop文件夹的所有者指定为hadoop用户

如果存放hadoop的目录的所有者不是hadoop，之后hadoop运行中可能会有权限问题，那么就讲所有者改为hadoop。

```
[hadoop@bigdata-senior01 ~]# sudo chown -R hadoop:hadoop /opt/modules  
1
```

十四、解压Hadoop目录文件

1、复制hadoop-2.5.0.tar.gz到/opt/modules目录下。

2、解压hadoop-2.5.0.tar.gz

```
[hadoop@bigdata-senior01 ~]# cd /opt/modules  
[hadoop@bigdata-senior01 hadoop]# tar -zxvf hadoop-2.5.0.tar.gz  
• 1  
• 2
```

十五、配置Hadoop

1、配置Hadoop环境变量

```
[hadoop@bigdata-senior01 hadoop]# vim /etc/profile  
1
```

追加配置：

```
export HADOOP_HOME="/opt/modules/hadoop-2.5.0"  
export PATH=$HADOOP_HOME/bin:$HADOOP_HOME/sbin:$PATH  
• 1  
• 2
```

执行：source /etc/profile 使得配置生效

验证HADOOP_HOME参数：

```
[hadoop@bigdata-senior01 ~]$ echo $HADOOP_HOME  
/opt/modules/hadoop-2.5.0  
• 1  
• 2
```

2、配置 hadoop-env.sh、mapred-env.sh、yarn-env.sh文件的JAVA_HOME参数

```
[hadoop@bigdata-senior01 ~]$ sudo vim ${HADOOP_HOME}/etc/hadoop/hadoop-env.sh
1
```

修改JAVA_HOME参数为：

```
export JAVA_HOME="/opt/modules/jdk1.7.0_67"
• 1
• 2
```

3、配置core-site.xml

```
<configuration>
  <property>
    <name>fs.defaultFS</name>
    <value>hdfs://bigdata-senior01.chybinmy.com:8020</value>
  </property>
  <property>
    <name>hadoop.tmp.dir</name>
    <value>/opt/data/tmp</value>
  </property>
</configuration>
```

```
[hadoop@bigdata-senior01 ~]{HADOOP_HOME}/etc/hadoop/core-site.xml
```

(1) fs.defaultFS参数配置的是HDFS的地址。

```
<property>
<name>fs.defaultFS</name>
<value>hdfs://bigdata-senior01.chybinmy.com:8020</value>
</property>
• 1
• 2
• 3
• 4
```

(2) `hadoop.tmp.dir` 配置的是Hadoop临时目录，比如HDFS的NameNode数据默认都存放在这个目录下，查看 `*-default.xml` 等默认配置文件，就可以看到很多依赖 `${hadoop.tmp.dir}` 的配置。

默认的 `hadoop.tmp.dir` 是 `/tmp/hadoop-${user.name}`，此时有个问题就是NameNode会将HDFS的元数据存储在这个/tmp目录下，如果操作系统重启了，系统会清空/tmp目录下的东西，导致NameNode元数据丢失，是个非常严重的问题，所有我们应该修改这个路径。

创建临时目录：

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ sudo mkdir -p /opt/data/tmp
1
```

将临时目录的所有者修改为hadoop

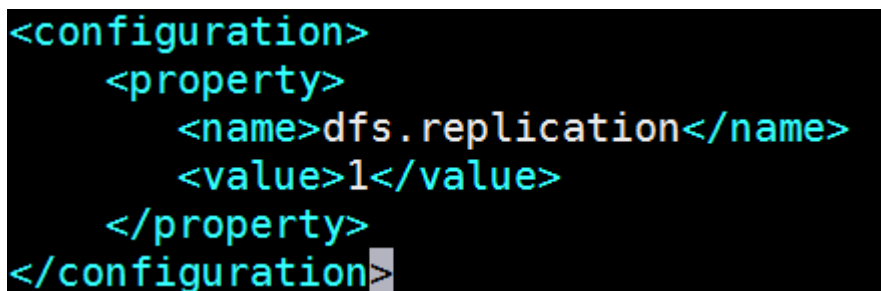
```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ sudo chown -R hadoop:hadoop /opt/data/tmp  
1
```

修改hadoop.tmp.dir

```
<property>  
  <name>hadoop.tmp.dir</name>  
  <value>/opt/data/tmp</value>  
</property>  
• 1  
• 2  
• 3  
• 4
```

十六、配置、格式化、启动HDFS

1、配置hdfs-site.xml



```
<configuration>  
  <property>  
    <name>dfs.replication</name>  
    <value>1</value>  
  </property>  
</configuration>
```

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ vim ${HADOOP_HOME}/etc/hadoop/hdfs-  
site.xml  
1
```

```
<property>  
  <name>dfs.replication</name>  
  <value>1</value>  
</property>  
• 1  
• 2  
• 3  
• 4
```

dfs.replication配置的是HDFS存储时的备份数量，因为这里是伪分布式环境只有一个节点，所以这里设置为1。

2、格式化HDFS

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ bin/hdfs namenode -format
16/07/04 17:19:31 INFO namenode.NameNode: STARTUP_MSG:
/*****
STARTUP_MSG: Starting NameNode
STARTUP_MSG: host = bigdata-senior01.chybinmy.com/127.0.0.1
STARTUP_MSG: args = [-format]
STARTUP_MSG: version = 2.5.0
STARTUP_MSG: classpath = /opt/modules/hadoop-2.5.0/etc/hadoop:/opt/modules/h
/hadoop-2.5.0/share/hadoop/common/lib/jasper-compiler-5.5.23.jar:/opt/modules/
/share/hadoop/common/lib/snappy-java-1.0.4.1.jar:/opt/modules/hadoop-2.5.0/sha
p/common/lib/protobuf-java-2.5.0.jar:/opt/modules/hadoop-2.5.0/share/hadoop/co
ommon/lib/jackson-xc-1.9.13.jar:/opt/modules/hadoop-2.5.0/share/hadoop/common/
tpcore-4.2.5.jar:/opt/modules/hadoop-2.5.0/share/hadoop/common/lib/hadoop-auth
s-codec-2.0.0-M15.jar:/opt/modules/hadoop-2.5.0/share/hadoop/common/lib/xz-1.0
:/opt/modules/hadoop-2.5.0/share/hadoop/common/lib/jettison-1.1.jar:/opt/modul
es/hadoop-2.5.0/share/hadoop/common/lib/jetty-6.1.26.jar:/opt/modules/hadoop-
op-2.5.0/share/hadoop/common/lib/commons-math3-3.1.1.jar:/opt/modules/hadoop-2
/hadoop/common/lib/jasper-runtime-5.5.23.jar:/opt/modules/hadoop-2.5.0/share/h
n/lib/asm-3.2.jar:/opt/modules/hadoop-2.5.0/share/hadoop/common/lib/jackson-ma
og4j12-1.7.5.jar:/opt/modules/hadoop-2.5.0/share/hadoop/common/lib/api-asn1-ap
```

```
[hadoop@bigdata-senior01 ~]$ hdfs namenode -format
1
```

格式化是对HDFS这个分布式文件系统里的DataNode进行分块，统计所有分块后的初始元数据的存储在NameNode中。

格式化后，查看core-site.xml里hadoop.tmp.dir（本例是/opt/data目录）指定的目录下是否有dfs目录，如果有，说明格式化成功。

注意：

1. 格式化时，这里注意hadoop.tmp.dir目录的权限问题，应该hadoop普通用户有读写权限才行，可以将/opt/data的所有者改为hadoop。

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ sudo chown -R hadoop:hadoop
/opt/data
```

2. 查看NameNode格式化后的目录。

```
[hadoop@bigdata-senior01 ~]$ ll /opt/data/tmp/dfs/name/current
1
```

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ ll /opt/data/tmp/dfs/name/current/
total 16
-rw-rw-r--. 1 hadoop hadoop 353 Jul  4 17:25 fsimage_00000000000000000000
-rw-rw-r--. 1 hadoop hadoop  62 Jul  4 17:25 fsimage_00000000000000000000.md5
-rw-rw-r--. 1 hadoop hadoop   2 Jul  4 17:25 seen_txid
-rw-rw-r--. 1 hadoop hadoop 202 Jul  4 17:25 VERSION
```

fsimage是NameNode元数据在内存满了后，持久化保存到的文件。

fsimage*.md5 是校验文件，用于校验fsimage的完整性。

seen_txid 是hadoop的版本

version文件里保存：

- namespaceID：NameNode的唯一ID。
- clusterID:集群ID, NameNode和DataNode的集群ID应该一致, 表明是一个集群。

```
#Mon Jul 04 17:25:50 CST 2016
namespaceID=2101579007
clusterID=CID-205277e6-493b-4601-8e33-c09d1d23ece4
cTime=0
storageType=NAME_NODE
blockpoolID=BP-1641019026-127.0.0.1-1467624350057
layoutVersion=-57
  • 1
  • 2
  • 3
  • 4
  • 5
  • 6
  • 7
```

3、启动NameNode

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ ${HADOOP_HOME}/sbin/hadoop-daemon.sh start
namenode
starting namenode, logging to /opt/modules/hadoop-2.5.0/logs/hadoop-hadoop-
namenode-bigdata-senior01.chybinmy.com.out
  • 1
  • 2
```

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ ${HADOOP_HOME}/sbin/hadoop-daemon.sh start namenode
starting namenode, logging to /opt/modules/hadoop-2.5.0/logs/hadoop-hadoop-namenode-bigdata-senior01.chybinmy.com.out
```

4、启动DataNode

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ ${HADOOP_HOME}/sbin/hadoop-daemon.sh start
datanode
starting datanode, logging to /opt/modules/hadoop-2.5.0/logs/hadoop-hadoop-
datanode-bigdata-senior01.chybinmy.com.out
  • 1
  • 2
```

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ ${HADOOP_HOME}/sbin/hadoop-daemon.sh start datanode
starting datanode, logging to /opt/modules/hadoop-2.5.0/logs/hadoop-hadoop-datanode-bigdata-senior01.chybinmy.com.out
```

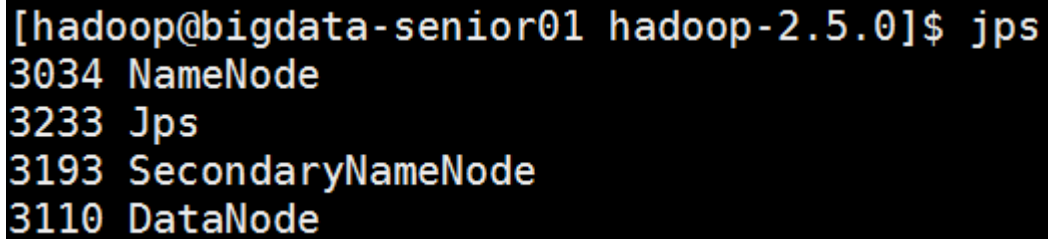
5、启动SecondaryNameNode

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ ${HADOOP_HOME}/sbin/hadoop-daemon.sh start
secondarynamenode
starting secondarynamenode, logging to /opt/modules/hadoop-2.5.0/logs/hadoop-
hadoop-secondarynamenode-bigdata-senior01.chybinmy.com.out
  • 1
  • 2
```

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ ${HADOOP_HOME}/sbin/hadoop-daemon.sh start secondarynamenode
starting secondarynamenode, logging to /opt/modules/hadoop-2.5.0/logs/hadoop-hadoop-secondarynamenode-bigdata-senior01.chybinmy.com.out
```

6、JPS命令查看是否已经启动成功，有结果就是启动成功了。

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ jps
3034 NameNode
3233 Jps
3193 SecondaryNameNode
3110 DataNode
    • 1
    • 2
    • 3
    • 4
    • 5
```

A terminal window with a black background and white text. The text shows the output of the 'jps' command: [hadoop@bigdata-senior01 hadoop-2.5.0]\$ jps, followed by four lines: 3034 NameNode, 3233 Jps, 3193 SecondaryNameNode, and 3110 DataNode.

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ jps
3034 NameNode
3233 Jps
3193 SecondaryNameNode
3110 DataNode
```

7、HDFS上测试创建目录、上传、下载文件

HDFS上创建目录

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ ${HADOOP_HOME}/bin/hdfs dfs -mkdir /demo1
1
```

上传本地文件到HDFS上

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ ${HADOOP_HOME}/bin/hdfs dfs -put
${HADOOP_HOME}/etc/hadoop/core-site.xml /demo1
    • 1
    • 2
```

读取HDFS上的文件内容

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ ${HADOOP_HOME}/bin/hdfs dfs -cat
/demo1/core-site.xml
1
```

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ ${HADOOP_HOME}/bin/hdfs dfs -cat /demo1/core-site.xml
16/07/04 17:46:29 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<!--
Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License. See accompanying LICENSE file.
-->

<!-- Put site-specific property overrides in this file. -->

<configuration>
  <property>
    <name>fs.defaultFS</name>
    <value>hdfs://bigdata-senior01.chybinmy.com:8020</value>
  </property>
  <property>
    <name>hadoop.tmp.dir</name>
    <value>/opt/data/tmp</value>
  </property>
</configuration>
```

从HDFS上下载文件到本地

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ bin/hdfs dfs -get /demo1/core-site.xml
1
```

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ ll
total 40
drwxr-xr-x. 2 hadoop hadoop 4096 Jul  2 10:03 bin
-rw-r--r--. 1 hadoop hadoop  988 Jul  4 17:48 core-site.xml
drwxr-xr-x. 3 hadoop hadoop 4096 Jul  4 15:13 etc
drwxrwxr-x. 2 hadoop hadoop 4096 Jul  4 15:23 hadoop_tmp
drwxr-xr-x. 2 hadoop hadoop 4096 Jul  2 10:03 include
drwxr-xr-x. 3 hadoop hadoop 4096 Jul  2 10:03 lib
drwxr-xr-x. 2 hadoop hadoop 4096 Jul  2 10:03 libexec
drwxrwxr-x. 2 hadoop hadoop 4096 Jul  4 17:35 logs
drwxr-xr-x. 2 hadoop hadoop 4096 Jul  2 10:03 sbin
drwxr-xr-x. 4 hadoop hadoop 4096 Aug  7  2014 share
```

十七、配置、启动YARN

1、配置mapred-site.xml

默认没有mapred-site.xml文件，但是有个mapred-site.xml.template配置模板文件。复制模板生成mapred-site.xml。

```
[hadoop@bigdata-senior01 hadoop-2.5.0]# cp etc/hadoop/mapred-site.xml.template
etc/hadoop/mapred-site.xml
1
```

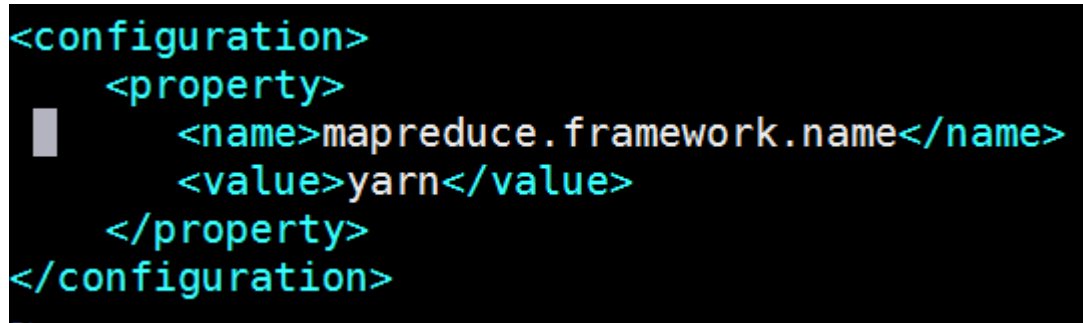
添加配置如下：

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


- 1
- 2
- 3
- 4

```

指定mapreduce运行在yarn框架上。

A screenshot of a terminal window showing XML configuration for mapreduce.framework.name. The text is as follows:

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

2、配置yarn-site.xml

添加配置如下：

```
<property>
  <name>yarn.nodemanager.aux-services</name>
  <value>mapreduce_shuffle</value>
</property>
<property>
  <name>yarn.resourcemanager.hostname</name>
  <value>bigdata-senior01.chybinmy.com</value>
</property>


- 1
- 2
- 3
- 4
- 5
- 6
- 7
- 8

```

- yarn.nodemanager.aux-services配置了yarn的默认混洗方式，选择为mapreduce的默认混洗算法。

- yarn.resourcemanager.hostname指定了Resourcemanager运行在哪个节点上。

```
<configuration>
  <property>
    <name>yarn.nodemanager.aux-services</name>
    <value>mapreduce_shuffle</value>
  </property>
  <property>
    <name>yarn.resourcemanager.hostname</name>
    <value>bigdata-senior01.chybinmy.com</value>
  </property>
</configuration>
```

3、启动Resourcemanager

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ ${HADOOP_HOME}/sbin/yarn-daemon.sh start
resourcemanager
1
```

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ ${HADOOP_HOME}/sbin/yarn-daemon.sh start resourcemanager
starting resourcemanager, logging to /opt/modules/hadoop-2.5.0/logs/yarn-hadoop-resourcemanager-bigdata-senior01.chybinmy.com.out
```

4、启动nodemanager

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ ${HADOOP_HOME}/sbin/yarn-daemon.sh start
nodemanager
1
```

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ ${HADOOP_HOME}/sbin/yarn-daemon.sh start nodemanager
starting nodemanager, logging to /opt/modules/hadoop-2.5.0/logs/yarn-hadoop-nodemanager-bigdata-senior01.chybinmy.com.out
```

5、查看是否启动成功

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ jps
3034 NameNode
4439 NodeManager
4197 ResourceManager
4543 Jps
3193 SecondaryNameNode
3110 DataNode
  • 1
  • 2
  • 3
  • 4
  • 5
  • 6
  • 7
```

可以看到ResourceManager、NodeManager已经启动成功了。

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ jps
3034 NameNode
4439 NodeManager
4197 ResourceManager
4543 Jps
3193 SecondaryNameNode
3110 DataNode
```

6、YARN的Web页面

YARN的Web客户端端口号是8088，通过<http://192.168.100.10:8088/>可以查看。

The screenshot shows the Hadoop YARN web interface. The browser address bar displays `192.168.100.10:8088/cluster`. The page features the Hadoop logo and the title "All Appl". On the left, there is a navigation menu under "Cluster" with options like "About", "Nodes", "Applications", and "Scheduler". The "Applications" section is expanded, showing a list of application states: NEW, NEW_SAVING, SUBMITTED, ACCEPTED, RUNNING, FINISHED, FAILED, and KILLED. The "Tools" section is also visible. The main content area displays "Cluster Metrics" with a table showing the following data:

| Apps Submitted | Apps Pending | Apps Running | Apps Completed | Containers Running | Memory Used | Memory Total | Mem Rese |
|----------------|--------------|--------------|----------------|--------------------|-------------|--------------|----------|
| 0 | 0 | 0 | 0 | 0 | 0 B | 8 GB | 0 B |

Below the table, there is a "Show 20 entries" dropdown and a table header with columns: ID, User, Name, Application Type, Queue, and StartTime. The table content is empty, displaying "No data". At the bottom of the page, there is a link for "About Apache Had".

十八、运行MapReduce Job

在Hadoop的share目录里，自带了一些jar包，里面带有一些mapreduce实例小例子，位置在 `share/hadoop/mapreduce/hadoop-mapreduce-examples-2.5.0.jar`，可以运行这些例子体验刚搭建好的Hadoop平台，我们这里来运行最经典的WordCount实例。

1、创建测试用的Input文件

创建输入目录:

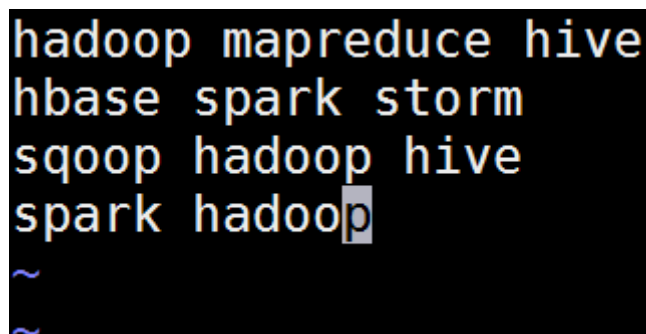
```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ bin/hdfs dfs -mkdir -p /wordcountdemo/input
```

创建原始文件:

在本地/opt/data目录创建一个文件wc.input,内容如下。

将wc.input文件上传到HDFS的/wordcountdemo/input目录中:

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ bin/hdfs dfs -put /opt/data/wc.input /wordcountdemo/input
```



```
hadoop mapreduce hive
hbase spark storm
sqoop hadoop hive
spark hadoop
```

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ bin/hdfs dfs -put /opt/data/wc.input /wordcountdemo/input
16/07/05 05:08:16 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform.
lasses where applicable
[hadoop@bigdata-senior01 hadoop-2.5.0]$ bin/hdfs dfs -ls /wordcountdemo/input
16/07/05 05:08:43 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform.
lasses where applicable
Found 1 items
-rw-r--r-- 1 hadoop supergroup          71 2016-07-05 05:08 /wordcountdemo/input/wc.input
```

2、运行WordCount MapReduce Job

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ bin/yarn jar share/hadoop/mapreduce/hadoop-mapreduce-examples-2.5.0.jar wordcount /wordcountdemo/input /wordcountdemo/output
```

```
16/07/05 05:12:02 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
16/07/05 05:12:04 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
16/07/05 05:12:05 INFO input.FileInputFormat: Total input paths to process : 1
16/07/05 05:12:05 INFO mapreduce.JobSubmitter: number of splits:1
16/07/05 05:12:05 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1467665627197_0001
16/07/05 05:12:06 INFO impl.YarnClientImpl: Submitted application application_1467665627197_0001
16/07/05 05:12:06 INFO mapreduce.Job: The url to track the job: http://localhost:8088/proxy/application_1467665627197_0001/
16/07/05 05:12:06 INFO mapreduce.Job: Running job: job_1467665627197_0001
16/07/05 05:12:17 INFO mapreduce.Job: Job job_1467665627197_0001 running in uber mode : false
16/07/05 05:12:17 INFO mapreduce.Job:  map 0% reduce 0%
16/07/05 05:12:30 INFO mapreduce.Job:  map 100% reduce 0%
16/07/05 05:12:44 INFO mapreduce.Job:  map 100% reduce 100%
```

3、查看输出结果目录

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ bin/hdfs dfs -ls /wordcountdemo/output
-rw-r--r-- 1 hadoop supergroup          0 2016-07-05 05:12 /wordcountdemo/output/_SUCCESS
-rw-r--r-- 1 hadoop supergroup        60 2016-07-05 05:12 /wordcountdemo/output/part-r-00000
```

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ bin/hdfs dfs -ls /wordcountdemo/output
16/07/05 05:14:26 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your plat
able
Found 2 items
-rw-r--r--  1 hadoop supergroup          0 2016-07-05 05:12 /wordcountdemo/output/_SUCCESS
-rw-r--r--  1 hadoop supergroup        60 2016-07-05 05:12 /wordcountdemo/output/part-r-00000
```

- output目录中有两个文件，_SUCCESS文件是空文件，有这个文件说明Job执行成功。
- part-r-00000文件是结果文件，其中-r-说明这个文件是Reduce阶段产生的结果，mapreduce程序执行时，可以没有reduce阶段，但是肯定会有map阶段，如果没有reduce阶段这个地方有是-m-。
- 一个reduce会产生一个part-r-开头的文件。
- 查看输出文件内容。

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ bin/hdfs dfs -cat
/wordcountdemo/output/part-r-00000
hadoop 3
hbase 1
hive 2
mapreduce 1
spark 2
sqoop 1
storm 1
  • 1
  • 2
  • 3
  • 4
  • 5
  • 6
  • 7
  • 8
```

结果是按照键值排好序的。

十九、停止Hadoop

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ sbin/hadoop-daemon.sh stop namenode
stopping namenode
[hadoop@bigdata-senior01 hadoop-2.5.0]$ sbin/hadoop-daemon.sh stop datanode
stopping datanode
[hadoop@bigdata-senior01 hadoop-2.5.0]$ sbin/yarn-daemon.sh stop resourcemanager
stopping resourcemanager
[hadoop@bigdata-senior01 hadoop-2.5.0]$ sbin/yarn-daemon.sh stop nodemanager
stopping nodemanager
  • 1
  • 2
  • 3
  • 4
  • 5
  • 6
  • 7
  • 8
```

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ sbin/hadoop-daemon.sh stop namenode
stopping namenode
[hadoop@bigdata-senior01 hadoop-2.5.0]$ sbin/hadoop-daemon.sh stop datanode
stopping datanode
[hadoop@bigdata-senior01 hadoop-2.5.0]$ sbin/yarn-daemon.sh stop resourcemanager
stopping resourcemanager
[hadoop@bigdata-senior01 hadoop-2.5.0]$ sbin/yarn-daemon.sh stop nodemanager
stopping nodemanager
```

二十、Hadoop各个功能模块的理解

1、HDFS模块

HDFS负责大数据的存储，通过将大文件分块后进行分布式存储方式，突破了服务器硬盘大小的限制，解决了单台机器无法存储大文件的问题，HDFS是个相对独立的模块，可以为YARN提供服务，也可以为HBase等其他模块提供服务。

2、YARN模块

YARN是一个通用的资源协同和任务调度框架，是为了解决Hadoop1.x中MapReduce里NameNode负载太大和其他问题而创建的一个框架。

YARN是个通用框架，不止可以运行MapReduce，还可以运行Spark、Storm等其他计算框架。

3、MapReduce模块

MapReduce是一个计算框架，它给出了一种数据处理的方式，即通过Map阶段、Reduce阶段来分布式地流式处理数据。它只适用于大数据的离线处理，对实时性要求很高的应用不适用。

第七步、开启历史服务

二十一、历史服务介绍

Hadoop开启历史服务可以在web页面上查看Yarn上执行job情况的详细信息。可以通过历史服务器查看已经运行完的Mapreduce作业记录，比如用了多少个Map、用了多少个Reduce、作业提交时间、作业启动时间、作业完成时间等信息。

二十二、开启历史服务

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ sbin/mr-jobhistory-daemon.sh start
historyserver、
    1
```

开启后，可以通过Web页面查看历史服务器：

<http://bigdata-senior01.chybinmy.com:19888/>

二十三、Web查看job执行历史

1、运行一个mapreduce任务

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ bin/yarn jar
share/hadoop/mapreduce/hadoop-mapreduce-examples-
2.5.0.jar wordcount /wordcountdemo/input /wordcountdemo/output1
```

- 1
- 2

2、job执行中

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 | 2 | 3 GB | 8 GB | 0 B | 2 | 8 | 0 | 1 | 0 | 0 | 0 | 0 |

Show 20 entries

| ID | User | Name | Application Type | Queue | StartTime | FinishTime | State | FinalStatus | Progress | Tracking UI |
|--------------------------------|--------|------------|------------------|---------|-------------------------------|------------|---------|-------------|----------|-------------------|
| application_1467869293970_0001 | hadoop | word count | MAPREDUCE | default | Thu, 07 Jul 2016 05:39:07 GMT | N/A | RUNNING | UNDEFINED | | ApplicationMaster |

Showing 1 to 1 of 1 entries

3、查看job历史

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 | 0 | 1 | 0 | 0 B | 8 GB | 0 B | 0 | 8 | 0 | 1 | 0 | 0 | 0 | 0 |

Show 20 entries

| ID | User | Name | Application Type | Queue | StartTime | FinishTime | State | FinalStatus | Progress | Tracking UI |
|--------------------------------|--------|------------|------------------|---------|-------------------------------|-------------------------------|----------|-------------|----------|-------------|
| application_1467869293970_0001 | hadoop | word count | MAPREDUCE | default | Thu, 07 Jul 2016 05:39:07 GMT | Thu, 07 Jul 2016 05:40:18 GMT | FINISHED | SUCCEEDED | | History |

Showing 1 to 1 of 1 entries

MapReduce Job
job_1468133958728_0001

Job Overview

Job Name: word count
 User Name: hadoop
 Queue: default
 State: SUCCEEDED
 Uberized: false
 Submitted: Sun Jul 10 14:59:56 CST 2016
 Started: Sun Jul 10 15:00:05 CST 2016
 Finished: Sun Jul 10 15:00:19 CST 2016
 Elapsed: 13sec

Diagnostics:

Average Map Time 4sec
 Average Shuffle Time 3sec
 Average Merge Time 0sec
 Average Reduce Time 0sec

| ApplicationMaster | | | |
|-------------------|------------------------------|------------------------------------|----------------------|
| Attempt Number | Start Time | Node | Logs |
| 1 | Sun Jul 10 15:00:01 CST 2016 | bigdata-senior01.chybinmy.com:8042 | logs |

| Task Type | Total | Complete |
|-----------|-------|----------|
| Map | 1 | 1 |
| Reduce | 1 | 1 |

| Attempt Type | Failed | Killed | Successful |
|--------------|--------|--------|------------|
| Maps | 0 | 0 | 1 |
| Reduces | 0 | 0 | 1 |

历史服务器的Web端口默认是19888，可以查看Web界面。

但是在上面所显示的某一个Job任务页面的最下面，Map和Reduce个数的链接上，点击进入Map的详细信息页面，再查看某一个Map或者Reduce的详细日志是看不到的，是因为没有开启日志聚集服务。

二十四、开启日志聚集

4、日志聚集介绍

MapReduce是在各个机器上运行的，在运行过程中产生的日志存在于各个机器上，为了能够统一查看各个机器的运行日志，将日志集中存放在HDFS上，这个过程就是日志聚集。

5、开启日志聚集

配置日志聚集功能：

Hadoop默认是不启用日志聚集的。在yarn-site.xml文件里配置启用日志聚集。

```

<property>
  <name>yarn.log-aggregation-enable</name>
  <value>>true</value>
</property>
<property>
  <name>yarn.log-aggregation.retain-seconds</name>
  <value>106800</value>
</property>


- 1
- 2
- 3
- 4
- 5
- 6
- 7
- 8

```

yarn.log-aggregation-enable:是否启用日志聚集功能。

yarn.log-aggregation.retain-seconds : 设置日志保留时间，单位是秒。

将配置文件分发到其他节点：

```

[hadoop@bigdata-senior01 hadoop]$ scp /opt/modules/hadoop-2.5.0/etc/hadoop/yarn-site.xml bigdata-senior02.chybinmy.com:/opt/modules/hadoop-2.5.0/etc/hadoop/
[hadoop@bigdata-senior01 hadoop]$ scp /opt/modules/hadoop-2.5.0/etc/hadoop/yarn-site.xml bigdata-senior03.chybinmy.com:/opt/modules/hadoop-2.5.0/etc/hadoop/


- 1
- 2

```

重启Yarn进程：

```

[hadoop@bigdata-senior01 hadoop-2.5.0]$ sbin/stop-yarn.sh
[hadoop@bigdata-senior01 hadoop-2.5.0]$ sbin/start-yarn.sh


- 1
- 2

```

重启HistoryServer进程：

```

[hadoop@bigdata-senior01 hadoop-2.5.0]$ sbin/mr-jobhistory-daemon.sh stop historyserver
[hadoop@bigdata-senior01 hadoop-2.5.0]$ sbin/mr-jobhistory-daemon.sh start historyserver


- 1
- 2

```

6、测试日志聚集

运行一个demo MapReduce, 使之产生日志：

```

bin/yarn jar share/hadoop/mapreduce/hadoop-mapreduce-examples-2.5.0.jar wordcount /input /output1
1

```


查看日志：

运行Job后，就可以在历史服务器Web页面查看各个Map和Reduce的日志了。

第四部分：完全分布式安装

第八步、完全布式环境部署Hadoop

完全分部式是真正利用多台Linux主机来进行部署Hadoop，对Linux机器集群进行规划，使得Hadoop各个模块分别部署在不同的多台机器上。

二十五、环境准备

1、克隆虚拟机

- Vmware左侧选中要克隆的机器，这里对原有的BigData01机器进行克隆，虚拟机菜单中，选中管理菜单下的克隆命令。
- 选择“创建完整克隆”，虚拟机名称为BigData02，选择虚拟机文件保存路径，进行克隆。
- 再次克隆一个名为BigData03的虚拟机。

2、配置网络

修改网卡名称：

在BigData02和BigData03机器上编辑网卡信息。执行`sudo vim /etc/udev/rules.d/70-persistent-net.rules`命令。因为是从BigData01机器克隆来的，所以会保留BigData01的网卡eth0，并且再添加一个网卡eth1。并且eth0的Mac地址和BigData01的地址是一样的，Mac地址不允许相同，所以要删除eth0，只保留eth1网卡，并且要将eth1改名为eth0。将修改后的eth0的mac地址复制下来，修改network-scripts文件中的HWADDR属性。

```
sudo vim /etc/sysconfig/network-scripts/ifcfg-eth0
1
```

```
hadoop@bigdata-senior01:~/Desktop
File Edit View Search Terminal Help
NETMASK=255.255.255.0
PREFIX=24
GATEWAY=192.168.100.2
DNS1=202.106.196.115
DNS2=202.106.0.20
DEFROUTE=yes
IPV4_FAILURE_FATAL=yes
IPV6INIT=no
NAME="System eth0"
HWADDR=00:0c:29:3c:2e:f7
LAST_CONNECT=1467323379
~
~
~
~
~
```

修改网络参数：

BigData02机器IP改为192.168.100.12

BigData03机器IP改为192.168.100.13

3、配置Hostname

BigData02配置hostname为 bigdata-senior02.chybinmy.com

BigData03配置hostname为 bigdata-senior03.chybinmy.com

4、配置hosts

BigData01、BigData02、BigData03三台机器hosts都配置为：

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ sudo vim /etc/hosts
192.168.100.10 bigdata-senior01.chybinmy.com
192.168.100.12 bigdata-senior02.chybinmy.com
192.168.100.13 bigdata-senior03.chybinmy.com
• 1
• 2
• 3
• 4
```

5、配置Windows上的SSH客户端

在本地Windows中的SSH客户端上添加对BigData02、BigData03机器的SSH链接。

二十六、服务器功能规划

bigdata-senior01.chybinmy.com

bigdata-senior02.chybinmy.com

bigdata-senior03.chybinmy.com

| bigdata-senior01.chybinmy.com | bigdata-senior02.chybinmy.com | bigdata-senior03.chybinmy.com |
|--------------------------------------|--------------------------------------|--------------------------------------|
| NameNode | ResourceManage | |
| DataNode | DataNode | DataNode |
| NodeManager | NodeManager | NodeManager |
| HistoryServer | | SecondaryNameNode |

二十七、在第一台机器上安装新的Hadoop

为了和之前BigData01机器上安装为分布式Hadoop区分开来，我们将BigData01上的Hadoop服务都停止掉，然后在一个新的目录/opt/modules/app下安装另外一个Hadoop。我们采用先在第一台机器上解压、配置Hadoop，然后再分发到其他两台机器上的方式来安装集群。

6、解压Hadoop目录：

```
[hadoop@bigdata-senior01 modules]$ tar -zxf /opt/sofeware/hadoop-2.5.0.tar.gz -C /opt/modules/app/
1
```

7、配置Hadoop JDK路径修改hadoop-env.sh、mapred-env.sh、yarn-env.sh文件中的JDK路径：

```
export JAVA_HOME="/opt/modules/jdk1.7.0_67"
1
```

8、配置core-site.xml

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ vim etc/hadoop/core-site.xml
1
```

```

<configuration>
  <property>
    <name>fs.defaultFS</name>
    <value>hdfs://bigdata-senior01.chybinmy.com:8020</value>
  </property>
  <property>
    <name>hadoop.tmp.dir</name>
    <value>/opt/modules/app/hadoop-2.5.0/data/tmp</value>
  </property>
</configuration>
  • 1
  • 2
  • 3
  • 4
  • 5
  • 6
  • 7
  • 8
  • 9
  • 10

```

fs.defaultFS 为NameNode的地址。

hadoop.tmp.dir 为hadoop临时目录的地址，默认情况下，NameNode和DataNode的数据文件都会存在这个目录下的对应子目录下。应该保证此目录是存在的，如果不存在，先创建。

9、配置hdfs-site.xml

```

[hadoop@bigdata-senior01 hadoop-2.5.0]$ vim etc/hadoop/hdfs-site.xml
  1

```

```

<configuration>
  <property>
    <name>dfs.namenode.secondary.http-address</name>
    <value>bigdata-senior03.chybinmy.com:50090</value>
  </property>
</configuration>
  • 1
  • 2
  • 3
  • 4
  • 5
  • 6

```

dfs.namenode.secondary.http-address是指定secondaryNameNode的http访问地址和端口号，因为在规划中，我们将BigData03规划为SecondaryNameNode服务器。

所以这里设置为：bigdata-senior03.chybinmy.com:50090

10、配置slaves

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ vim etc/hadoop/slaves
bigdata-senior01.chybinmy.com
bigdata-senior02.chybinmy.com
bigdata-senior03.chybinmy.com
  • 1
  • 2
  • 3
  • 4
```

slaves文件是指定HDFS上有哪些DataNode节点。

11、配置yarn-site.xml

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ vim etc/hadoop/yarn-site.xml
1

<property>
  <name>yarn.nodemanager.aux-services</name>
  <value>mapreduce_shuffle</value>
</property>
<property>
  <name>yarn.resourcemanager.hostname</name>
  <value>bigdata-senior02.chybinmy.com</value>
</property>
<property>
  <name>yarn.log-aggregation-enable</name>
  <value>>true</value>
</property>
<property>
  <name>yarn.log-aggregation.retain-seconds</name>
  <value>106800</value>
</property>
  • 1
  • 2
  • 3
  • 4
  • 5
  • 6
  • 7
  • 8
  • 9
  • 10
  • 11
  • 12
  • 13
  • 14
  • 15
  • 16
```

根据规则! `yarn.resourcemanager.hostname` 这个指定resourcemanager服务器指向 `bigdata-senior02.chybinmy.com`。

`yarn.log-aggregation-enable` 是配置是否启用日志聚集功能。

`yarn.log-aggregation.retain-seconds` 是配置聚集的日志在HDFS上最多保存多长时间。

12、配置mapred-site.xml

从mapred-site.xml.template复制一个mapred-site.xml文件。

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ cp etc/hadoop/mapred-site.xml.template  
etc/hadoop/mapred-site.xml
```

1

```
<configuration>  
  <property>  
    <name>mapreduce.framework.name</name>  
    <value>yarn</value>  
  </property>  
  <property>  
    <name>mapreduce.jobhistory.address</name>  
    <value>bigdata-senior01.chybinmy.com:10020</value>  
  </property>  
  <property>  
    <name>mapreduce.jobhistory.webapp.address</name>  
    <value>bigdata-senior01.chybinmy.com:19888</value>  
  </property>  
</configuration>  
• 1  
• 2  
• 3  
• 4  
• 5  
• 6  
• 7  
• 8  
• 9  
• 10  
• 11  
• 12  
• 13  
• 14
```

mapreduce.framework.name设置mapreduce任务运行在yarn上。

mapreduce.jobhistory.address是设置mapreduce的历史服务器安装在BigData01机器上。

mapreduce.jobhistory.webapp.address是设置历史服务器的web页面地址和端口号。

二十八、设置SSH无密码登录

Hadoop集群中的各个机器间会相互地通过SSH访问，每次访问都输入密码是不现实的，所以要配置各个机器间的

SSH是无密码登录的。

1、在BigData01上生成公钥

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ ssh-keygen -t rsa
1
```

一路回车，都设置为默认值，然后再当前用户的Home目录下的 `.ssh` 目录中会生成公钥文件 `(id_rsa.pub)` 和私钥文件 `(id_rsa)`。

2、分发公钥

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ ssh-copy-id bigdata-senior01.chybinmy.com
[hadoop@bigdata-senior01 hadoop-2.5.0]$ ssh-copy-id bigdata-senior02.chybinmy.com
[hadoop@bigdata-senior01 hadoop-2.5.0]$ ssh-copy-id bigdata-senior03.chybinmy.com
• 1
• 2
• 3
```

3、设置BigData02、BigData03到其他机器的无密钥登录

同样的在BigData02、BigData03上生成公钥和私钥后，将公钥分发到三台机器上。

二十九、分发Hadoop文件

1、首先在其他两台机器上创建存放Hadoop的目录

```
[hadoop@bigdata-senior02 ~]$ mkdir /opt/modules/app
[hadoop@bigdata-senior03 ~]$ mkdir /opt/modules/app
• 1
• 2
```

2、通过Scp分发

Hadoop根目录下的share/doc目录是存放的hadoop的文档，文件相当大，建议在分发之前将这个目录删除掉，可以节省硬盘空间并能提高分发的速度。

doc目录大小有1.6G。

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ du -sh /opt/modules/app/hadoop-2.5.0/share/doc
1.6G    /opt/modules/app/hadoop-2.5.0/share/doc
[hadoop@bigdata-senior01 hadoop-2.5.0]$ scp -r /opt/modules/app/hadoop-2.5.0/ bigdata-senior02.chybinmy.com:/opt/modules/app
[hadoop@bigdata-senior01 hadoop-2.5.0]$ scp -r /opt/modules/app/hadoop-2.5.0/ bigdata-senior03.chybinmy.com:/opt/modules/app
• 1
• 2
• 3
• 4
```

三十、格式NameNode

在NameNode机器上执行格式化：

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ /opt/modules/app/hadoop-2.5.0/bin/hdfs
namenode -format
1
```

注意：

如果需要重新格式化NameNode,需要先将原来NameNode和DataNode下的文件全部删除,不然会报错,NameNode和DataNode所在目录是在 `core-site.xml` 中 `hadoop.tmp.dir`、`dfs.namenode.name.dir`、`dfs.datanode.data.dir` 属性配置的。

```
<property>
  <name>hadoop.tmp.dir</name>
  <value>/opt/data/tmp</value>
</property>
<property>
  <name>dfs.namenode.name.dir</name>
  <value>file://${hadoop.tmp.dir}/dfs/name</value>
</property>
<property>
  <name>dfs.datanode.data.dir</name>
  <value>file://${hadoop.tmp.dir}/dfs/data</value>
</property>
• 1
• 2
• 3
• 4
• 5
• 6
• 7
• 8
• 9
• 10
• 11
• 12
```

因为每次格式化,默认是创建一个集群ID,并写入NameNode和DataNode的VERSION文件中(VERSION文件所在目录为`dfs/name/current`和`dfs/data/current`),重新格式化时,默认会生成一个新的集群ID,如果不删除原来的目录,会导致namenode中的VERSION文件中是新的集群ID,而DataNode中是旧的集群ID,不一致时会报错。

另一种方法是格式化时指定集群ID参数,指定为旧的集群ID。

三十一、启动集群

1、启动HDFS

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ /opt/modules/app/hadoop-2.5.0/sbin/start-
dfs.sh
1
```



```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ /opt/modules/app/hadoop-2.5.0/sbin/start-dfs.sh
16/07/14 16:00:57 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-
java classes where applicable
Starting namenodes on [bigdata-senior01.chybinmy.com]
bigdata-senior01.chybinmy.com: starting namenode, logging to /opt/modules/app/hadoop-2.5.0/logs/hadoop-hadoop-namenode
-bigdata-senior01.chybinmy.com.out
bigdata-senior03.chybinmy.com: starting datanode, logging to /opt/modules/app/hadoop-2.5.0/logs/hadoop-hadoop-datanode
-bigdata-senior03.chybinmy.com.out
bigdata-senior02.chybinmy.com: starting datanode, logging to /opt/modules/app/hadoop-2.5.0/logs/hadoop-hadoop-datanode
-bigdata-senior02.chybinmy.com.out
bigdata-senior01.chybinmy.com: starting datanode, logging to /opt/modules/app/hadoop-2.5.0/logs/hadoop-hadoop-datanode
-bigdata-senior01.chybinmy.com.out
Starting secondary namenodes [bigdata-senior03.chybinmy.com]
bigdata-senior03.chybinmy.com: starting secondarynamenode, logging to /opt/modules/app/hadoop-2.5.0/logs/hadoop-hadoop
-secondarynamenode-bigdata-senior03.chybinmy.com.out
16/07/14 16:01:16 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-
java classes where applicable
[hadoop@bigdata-senior01 hadoop-2.5.0]$ jps
4454 Jps
4122 NameNode
4245 DataNode
```

2、启动YARN

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ /opt/modules/app/hadoop-2.5.0/sbin/start-
yarn.sh
```

1

在BigData02上启动ResourceManager:

```
[hadoop@bigdata-senior02 hadoop-2.5.0]$ sbin/yarn-daemon.sh start resourcemanager
```

1

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ ^C
[hadoop@bigdata-senior01 hadoop-2.5.0]$ /opt/modules/app/hadoop-2.5.0/sbin/start-yarn.sh
starting yarn daemons
starting resourcemanager, logging to /opt/modules/app/hadoop-2.5.0/logs/yarn-hadoop-resourcemanager-bigdata-senior01.chybinmy.com.out
bigdata-senior02.chybinmy.com: starting nodemanager, logging to /opt/modules/app/hadoop-2.5.0/logs/yarn-hadoop-nodemanager-bigdata-senior02.
chybinmy.com.out
bigdata-senior03.chybinmy.com: starting nodemanager, logging to /opt/modules/app/hadoop-2.5.0/logs/yarn-hadoop-nodemanager-bigdata-senior03.
chybinmy.com.out
bigdata-senior01.chybinmy.com: starting nodemanager, logging to /opt/modules/app/hadoop-2.5.0/logs/yarn-hadoop-nodemanager-bigdata-senior01.
chybinmy.com.out
[hadoop@bigdata-senior01 hadoop-2.5.0]$ jps
4662 Jps
4609 NodeManager
4122 NameNode
4245 DataNode
```

3、启动日志服务器

因为我们规划的是在BigData03服务器上运行MapReduce日志服务，所以要在BigData03上启动。

```
[hadoop@bigdata-senior03 ~]$ /opt/modules/app/hadoop-2.5.0/sbin/mr-jobhistory-
daemon.sh start historyserver
starting historyserver, logging to /opt/modules/app/hadoop-2.5.0/logs/mapred-
hadoop-historyserver-bigda ta-senior03.chybinmy.com.out
```

- 1
- 2

```
[hadoop@bigdata-senior03 ~]$ jps
3570 Jps
3537 JobHistoryServer
3310 SecondaryNameNode
3213 DataNode
3392 NodeManager
  • 1
  • 2
  • 3
  • 4
  • 5
  • 6
```

4、查看HDFS Web页面

<http://bigdata-senior01.chybinmy.com:50070/>

5、查看YARN Web 页面

<http://bigdata-senior02.chybinmy.com:8088/cluster>

三十二、测试Job

我们这里用hadoop自带的wordcount例子来在本地模式下测试跑mapreduce。

1、准备mapreduce输入文件wc.input

```
[hadoop@bigdata-senior01 modules]$ cat /opt/data/wc.input
hadoop mapreduce hive
hbase spark storm
sqoop hadoop hive
spark hadoop
  • 1
  • 2
  • 3
  • 4
  • 5
```

2、在HDFS创建输入目录input

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ bin/hdfs dfs -mkdir /input
1
```

3、将wc.input上传到HDFS

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ bin/hdfs dfs -put /opt/data/wc.input
/input/wc.input
1
```

4、运行hadoop自带的mapreduce Demo

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ bin/yarn jar
share/hadoop/mapreduce/hadoop-mapreduce-examples-2.5.0.jar wordcount
/input/wc.input /output
```

1

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ bin/yarn jar share/hadoop/mapreduce/hadoop-mapreduce-examples-2.5.0.jar wordcount /input
/wc.input output
16/07/14 16:32:29 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java class
es where applicable
16/07/14 16:32:30 INFO client.RMProxy: Connecting to ResourceManager at bigdata-senior02.chybinmy.com/192.168.100.12:8032
16/07/14 16:32:31 INFO input.FileInputFormat: Total input paths to process : 1
16/07/14 16:32:31 INFO mapreduce.JobSubmitter: number of splits:1
16/07/14 16:32:31 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1468484139051_0003
16/07/14 16:32:32 INFO impl.YarnClientImpl: Submitted application application_1468484139051_0003
16/07/14 16:32:32 INFO mapreduce.Job: The url to track the job: http://bigdata-senior02.chybinmy.com:8088/proxy/application_1468
484139051_0003/
16/07/14 16:32:32 INFO mapreduce.Job: Running job: job_1468484139051_0003
16/07/14 16:32:43 INFO mapreduce.Job: Job job_1468484139051_0003 running in uber mode : false
16/07/14 16:32:43 INFO mapreduce.Job: map 0% reduce 0%
16/07/14 16:32:50 INFO mapreduce.Job: map 100% reduce 0%
16/07/14 16:33:01 INFO mapreduce.Job: map 100% reduce 100%
16/07/14 16:33:01 INFO mapreduce.Job: Job job_1468484139051_0003 completed successfully
16/07/14 16:33:01 INFO mapreduce.Job: Counters: 49
File System Counters
FILE: Number of bytes read=94
FILE: Number of bytes written=194511
FILE: Number of read operations=0
FILE: Number of large read operations=0
FILE: Number of write operations=0
HDFS: Number of bytes read=192
```

5、查看输出文件

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ bin/hdfs dfs -ls /output
```

Found 2 items

```
-rw-r--r--  3 hadoop supergroup          0 2016-07-14 16:36 /output/_SUCCESS
-rw-r--r--  3 hadoop supergroup        60 2016-07-14 16:36 /output/part-r-00000
  • 1
  • 2
  • 3
  • 4
```

第五部分：Hadoop HA安装

HA的意思是High Availability高可用，指当前工作中的机器宕机后，会自动处理这个异常，并将工作无缝地转移到其他备用机器上去，以此来保证服务的高可用。

HA方式安装部署才是最常见的生产环境上的安装部署方式。Hadoop HA是Hadoop 2.x中新添加的特性，包括NameNode HA 和 ResourceManager HA。因为DataNode和NodeManager本身就是被设计为高可用的，所以不用对他们进行特殊的高可用处理。

第九步、时间服务器搭建

Hadoop对集群中各个机器的时间同步要求比较高，要求各个机器的系统时间不能相差太多，不然会造成很多问题。可以配置集群中各个机器和互联网的时间服务器进行时间同步，但是在实际生产环境中，集群中大部分服务器是不能连接外网的，这时候可以在内网搭建一个自己的时间服务器（NTP服务器），集群的各个机器与这个时间服务器进行时间同步。

三十三、配置NTP服务器

我们选择第三台机器（bigdata-senior03.chybinmy.com）为NTP服务器，其他机器和这台机器进行同步。

1、检查ntp服务是否已经安装

```
[hadoop@bigdata-senior03 data]$ sudo rpm -qa | grep ntp
ntpdate-4.2.6p5-1.el6.centos.x86_64
ntp-4.2.6p5-1.el6.centos.x86_64
  • 1
  • 2
  • 3
```

显示已经安装过了ntp程序，其中 `ntpdate-4.2.6p5-1.el6.centos.x86_64` 是用来和某台服务器进行同步的，`ntp-4.2.6p5-1.el6.centos.x86_64` 是用来提供时间同步服务的。

2、修改配置文件ntp.conf

```
[hadoop@bigdata-senior03 data]$ vim /etc/ntp.conf
1
```

启用restrict,修改网段

```
restrict 192.168.100.0 mask 255.255.255.0 nomodify notrap
```

将这行的注释去掉，并且将网段改为集群的网段，我们这里是100网段。

注释掉server域名配置

```
#server 0.centos.pool.ntp.org iburst
#server 1.centos.pool.ntp.org iburst
#server 2.centos.pool.ntp.org iburst
#server 3.centos.pool.ntp.org iburst
  • 1
  • 2
  • 3
  • 4
```

是时间服务器的域名，这里不需要连接互联网，所以将他们注释掉。

修改

```
server 127.127.1.0
```

```
fudge 127.127.1.0 stratum 10
```

3、修改配置文件ntpd

```
[hadoop@bigdata-senior03 ~]$ sudo vim /etc/sysconfig/ntpd
1
```

添加一行配置：SYNC_CLOCK=yes

```
# Drop root to id 'ntp:ntp' by default.
SYNC_HWCLOCK=yes
OPTIONS="-u ntp:ntp -p /var/run/ntpd.pid -g"
```

4、启动ntp服务

```
[hadoop@bigdata-senior03 ~]$ sudo chkconfig ntpd on
1
```

这样每次机器启动时，ntp服务都会自动启动。

三十四、配置其他机器的同步

切换到root用户进行配置通过crontab进行定时同步：

```
[root@bigdata-senior02 hadoop]# crontab -e
*/10 * * * * /usr/sbin/ntpdate bigdata-senior03.chybinmy.com
[root@bigdata-senior02 hadoop]# crontab -e
*/10 * * * * /usr/sbin/ntpdate bigdata-senior03.chybinmy.com
• 1
• 2
• 3
• 4
```

三十五、测试同步是否有效

1、查看目前三台机器的时间

```
[hadoop@bigdata-senior03 ~]$ date "+%Y-%m-%d %H:%M:%S"
2016-09-23 16:43:56
[hadoop@bigdata-senior02 ~]$ date "+%Y-%m-%d %H:%M:%S"
2016-09-23 16:44:08
[hadoop@bigdata-senior01 data]$ date "+%Y-%m-%d %H:%M:%S"
2016-09-23 16:44:18
• 1
• 2
• 3
• 4
• 5
• 6
```

2、修改bigdata-senior01上的时间

将时间改为一个以前的时间：

```
[hadoop@bigdata-senior01 data]$ sudo date -s '2016-01-01 00:00:00'
Fri Jan  1 00:00:00 CST 2016
[hadoop@bigdata-senior01 data]$ date "+%Y-%m-%d %H:%M:%S"
2016-01-01 00:00:05
• 1
• 2
• 3
• 4
```

等10分钟，看是否可以实现自动同步，将bigdata-senior01上的时间修改为和bigdata-senior03上的一致。

3、查看是否自动同步时间

```
[hadoop@bigdata-senior01 data]$ date "+%Y-%m-%d %H:%M:%S"  
2016-09-23 16:54:36  
• 1  
• 2
```

可以看到bigdata-senior01上的时间已经实现自动同步了。

第十步、Zookeeper分布式机器部署

三十六、zookeeper说明

Zookeeper在Hadoop集群中的作用。

Zookeeper是分布式管理协作框架，Zookeeper集群用来保证Hadoop集群的高可用，（高可用的含义是：集群中就算有一部分服务器宕机，也能保证正常地对外提供服务。）

Zookeeper保证高可用的原理。

Zookeeper集群能够保证NameNode服务高可用的原理是：Hadoop集群中有两个NameNode服务，两个NameNode都定时地给Zookeeper发送心跳，告诉Zookeeper我还活着，可以提供服务，某一个时间只有一个是Action状态，另外一个Standby状态，一旦Zookeeper检测不到Action NameNode发送来的心跳后，就切换到Standby状态的NameNode上，将它设置为Action状态，所以集群中总有一个可用的NameNode，达到了NameNode的高可用目的。

Zookeeper的选举机制。

Zookeeper集群也能保证自身的高可用，保证自身高可用的原理是，Zookeeper集群中的各个机器分为Leader和Follower两个角色，写入数据时，要先写入Leader，Leader同意写入后，再通知Follower写入。客户端读取数据时，因为数据都是一样的，可以从任意一台机器上读取数据。

这里Leader角色就存在单点故障的隐患，高可用就是解决单点故障隐患的。Zookeeper从机制上解决了Leader的单点故障问题，Leader是哪一台机器是不固定的，Leader是选举出来的。选举流程是，集群中任何一台机器发现集群中没有Leader时，就推荐自己为Leader，其他机器来同意，当超过一半数的机器同意它为Leader时，选举结束，所以Zookeeper集群中的机器数据必须是奇数。这样就算当Leader机器宕机后，会很快选举出新的Leader，保证了Zookeeper集群本身的高可用。

写入高可用。

集群中的写入操作都是先通知Leader，Leader再通知Follower写入，实际上当超过一半的机器写入成功后，就认为写入成功了，所以就算有些机器宕机，写入也是成功的。

读取高可用。

zookeeper客户端读取数据时，可以读取集群中的任何一个机器。所以部分机器的宕机并不影响读取。

zookeeper服务器必须是奇数台，因为zookeeper有选举制度，角色有：领导者、跟随者、观察者，选举的目的是保证集群中数据的一致性。

三十七、安装zookeeper

我们这里在BigData01、BigData02、BigData03三台机器上安装zookeeper集群。

1、解压安装包

在BigData01上安装解压zookeeper安装包。

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ tar -zxf /opt/software/zookeeper-3.4.8.tar.gz -C /opt/modules/
1
```

2、修改配置

拷贝conf下的zoo_sample.cfg副本，改名为zoo.cfg。zoo.cfg是zookeeper的配置文件：

```
[hadoop@bigdata-senior01 zookeeper-3.4.8]$ cp conf/zoo_sample.cfg conf/zoo.cfg
1
```

dataDir属性设置zookeeper的数据文件存放的目录：

```
dataDir=/opt/modules/zookeeper-3.4.8/data/zData
```

指定zookeeper集群中各个机器的信息：

```
server.1=bigdata-senior01.chybinmy.com:2888:3888
server.2=bigdata-senior02.chybinmy.com:2888:3888
server.3=bigdata-senior03.chybinmy.com:2888:3888
• 1
• 2
• 3
```

server后面的数字范围是1到255，所以一个zookeeper集群最多可以有255个机器。

```
dataDir=/opt/modules/zookeeper-3.4.8/data/zData
# the port at which the clients will connect
clientPort=2181
# the maximum number of client connections.
# increase this if you need to handle more clients
#maxClientCnxns=60
#
# Be sure to read the maintenance section of the
# administrator guide before turning on autopurge.
#
# http://zookeeper.apache.org/doc/current/zookeepe
#
# The number of snapshots to retain in dataDir
#autopurge.snapRetainCount=3
# Purge task interval in hours
# Set to "0" to disable auto purge feature
#autopurge.purgeInterval=1
server.1=bigdata-senior01.chybinmy.com:2888:3888
server.2=bigdata-senior02.chybinmy.com:2888:3888
server.3=bigdata-senior03.chybinmy.com:2888:3888
```

3、创建myid文件

在dataDir所指定的目录下创建一个名为myid的文件，文件内容为server点后面的数字。

```
[hadoop@bigdata-senior01 zookeeper-3.4.8]$ vim conf/zoo.cfg
[hadoop@bigdata-senior01 zookeeper-3.4.8]$ touch data/zData/myid
[hadoop@bigdata-senior01 zookeeper-3.4.8]$ echo 1 >> data/zData/myid
[hadoop@bigdata-senior01 zookeeper-3.4.8]$ cat data/zData/myid
1
```

4、分发到其他机器

```
[hadoop@bigdata-senior01 zookeeper-3.4.8]$ scp -r /opt/modules/zookeeper-3.4.8
bigdata-senior02.chybinmy.com:/opt/modules
[hadoop@bigdata-senior01 zookeeper-3.4.8]$ scp -r /opt/modules/zookeeper-3.4.8
bigdata-senior03.chybinmy.com:/opt/modules
• 1
• 2
```

5、修改其他机器上的myid文件


```
[hadoop@bigdata-senior02 zookeeper-3.4.8]$ echo 2 > /opt/modules/zookeeper-3.4.8/data/zData/myid
[hadoop@bigdata-senior02 zookeeper-3.4.8]$ cat /opt/modules/zookeeper-3.4.8/data/zData/myid
2
• 1
• 2
• 3
```

```
[hadoop@bigdata-senior03 ~]$ echo 3 > /opt/modules/zookeeper-3.4.8/data/zData/myid
[hadoop@bigdata-senior03 ~]$ cat /opt/modules/zookeeper-3.4.8/data/zData/myid
3
• 1
• 2
• 3
```

6、启动zookeeper

需要在各个机器上分别启动zookeeper。

```
[hadoop@bigdata-senior01 zookeeper-3.4.8]$ bin/zkServer.sh start
[hadoop@bigdata-senior02 zookeeper-3.4.8]$ bin/zkServer.sh start
[hadoop@bigdata-senior03 zookeeper-3.4.8]$ bin/zkServer.sh start
• 1
• 2
• 3
```

```
[hadoop@bigdata-senior02 zookeeper-3.4.8]$ bin/zkServer.sh start
ZooKeeper JMX enabled by default
Using config: /opt/modules/zookeeper-3.4.8/bin/../conf/zoo.cfg
Starting zookeeper ... STARTED
[hadoop@bigdata-senior02 zookeeper-3.4.8]$ jps
4384 QuorumPeerMain
3218 NodeManager
3387 ResourceManager
3113 DataNode
4416 Jps
```

三十八、zookeeper命令

进入zookeeper Shell

在zookeeper根目录下执行 bin/zkCli.sh 进入zk shell模式。

zookeeper 很像是一个小型的文件系统，/是根目录，下面的所有节点都叫zNode。

进入zk shell 后输入任意字符，可以列出所有的zookeeper命令

```
[zk: localhost:2181(CONNECTED) 2] /
ZooKeeper -server host:port cmd args
  connect host:port
  get path [watch]
  ls path [watch]
  set path data [version]
  rmr path
  delquota [-n|-b] path
  quit
  printwatches on|off
  create [-s] [-e] path data acl
  stat path [watch]
  close
  ls2 path [watch]
  history
  listquota path
  setAcl path acl
  getAcl path
  sync path
  redo cmdno
  addauth scheme auth
  delete path [version]
  setquota -n|-b val path
```

查询zNode上的数据：get /zookeeper

创建一个zNode：create /znode1 “demodata”

列出所有子zNode：ls /

```
[zk: localhost:2181(CONNECTED) 0] ls /
[znode1, zookeeper]
```

删除znode：rmr /znode1

退出shell模式：quit

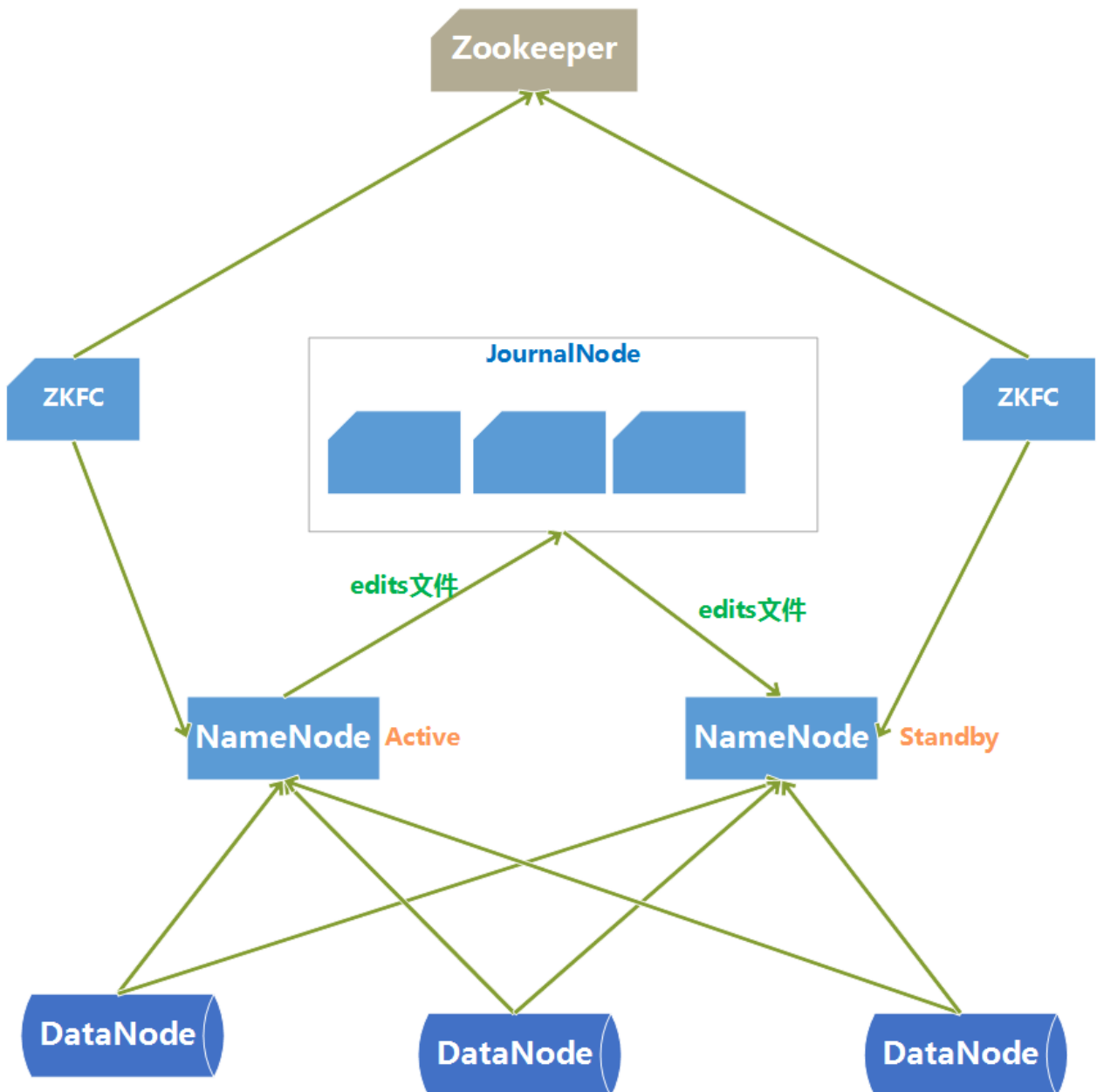
第十一步、Hadoop 2.x HDFS HA 部署

三十九、HDFS HA原理

单NameNode的缺陷存在单点故障的问题，如果NameNode不可用，则会导致整个HDFS文件系统不可用。所以需要设计高可用的HDFS（Hadoop HA）来解决NameNode单点故障的问题。解决的方法是在HDFS集群中设置多个NameNode节点。但是一旦引入多个NameNode，就有一些问题需要解决。

- HDFS HA需要保证的四个问题：
 - 保证NameNode内存中元数据数据一致，并保证编辑日志文件的安全性。
 - 多个NameNode如何协作
 - 客户端如何能正确地访问到可用的那个NameNode。
 - 怎么保证任意时刻只能有一个NameNode处于对外服务状态。
- 解决方法
 - 对于保证NameNode元数据的一致性和编辑日志的安全性，采用Zookeeper来存储编辑日志文件。
 - 两个NameNode一个是Active状态的，一个是Standby状态的，一个时间点只能有一个Active状态的NameNode提供服务，两个NameNode上存储的元数据是实时同步的，当Active的NameNode出现问题时，通过Zookeeper实时切换到Standby的NameNode上，并将Standby改为Active状态。
 - 客户端通过连接一个Zookeeper的代理来确定当时哪个NameNode处于服务状态。

四十、HDFS HA架构图



- HDFS HA架构中有两台NameNode节点，一台是处于活动状态（Active）为客户端提供服务，另外一台处于热备份状态（Standby）。
- 元数据文件有两个文件：fsimage和edits，备份元数据就是备份这两个文件。JournalNode用来实时从Active NameNode上拷贝edits文件，JournalNode有三台也是为了实现高可用。
- Standby NameNode不对外提供元数据的访问，它从Active NameNode上拷贝fsimage文件，从JournalNode上拷贝edits文件，然后负责合并fsimage和edits文件，相当于SecondaryNameNode的作用。最终目的是保证Standby NameNode上的元数据信息和Active NameNode上的元数据信息一致，以实现热备份。
- Zookeeper来保证在Active NameNode失效时及时将Standby NameNode修改为Active状态。

- ZKFC (失效检测控制) 是Hadoop里的一个Zookeeper客户端, 在每一个NameNode节点上都启动一个ZKFC进程, 来监控NameNode的状态, 并把NameNode的状态信息汇报给Zookeeper集群, 其实就是在Zookeeper上创建了一个Znode节点, 节点里保存了NameNode状态信息。当NameNode失效后, ZKFC检测到报告给Zookeeper, Zookeeper把对应的Znode删除掉, Standby ZKFC发现没有Active状态的NameNode时, 就会用shell命令将自己监控的NameNode改为Active状态, 并修改Znode上的数据。
Znode是个临时的节点, 临时节点特征是客户端的连接断了后就会把znode删除, 所以当ZKFC失效时, 也会导致切换NameNode。
- DataNode会将心跳信息和Block汇报信息同时发给两台NameNode, DataNode只接受Active NameNode发来的文件读写操作指令。

四十一、搭建HDFS HA 环境

1、服务器角色规划

| bigdata-senior01.chybinmy.com | bigdata-senior01.chybinmy.com | bigdata-senior01.chybinmy.com |
|--------------------------------------|--------------------------------------|--------------------------------------|
| NameNode | NameNode | |
| Zookeeper | Zookeeper | Zookeeper |
| DataNode | DataNode | DataNode |
| | ResourceManage | ResourceManage |
| NodeManager | NodeManager | NodeManager |

2、创建HDFS HA 版本Hadoop程序目录

在bigdata01、bigdata02、bigdata03三台机器上分别创建目录/opt/modules/hadooph/用来存放Hadoop HA环境。

```
[hadoop@bigdata-senior01 modules]$ mkdir /opt/modules/hadooph
1
```

3、新解压Hadoop 2.5.0

```
[hadoop@bigdata-senior01 ~]$ tar -zxf /opt/software/hadoop-2.5.0.tar.gz -C /opt/modules/hadooph/
1
```

4、配置Hadoop JDK路径

修改hadoop-env.sh、mapred-env.sh、yarn-env.sh文件中的JDK路径

```
export JAVA_HOME="/opt/modules/jdk1.7.0_67"
```

- 1
- 2

5、配置hdfs-site.xml

```

<?xml version="1.0" encoding="UTF-8"?>
<configuration>
  <property>

    <name>dfs.nameservices</name>
    <value>ns1</value>
  </property>
  <property>

    <name>dfs.ha.namenodes.ns1</name>
    <value>nn1,nn2</value>
  </property>
  <property>

    <name>dfs.namenode.rpc-address.ns1.nn1</name>
    <value>bigdata-senior01.chybinmy.com:8020</value>
  </property>
  <property>

    <name>dfs.namenode.rpc-address.ns1.nn2</name>
    <value>bigdata-senior02.chybinmy.com:8020</value>
  </property>
  <property>

    <name>dfs.namenode.http-address.ns1.nn1</name>
    <value>bigdata-senior01.chybinmy.com:50070</value>
  </property>
  <property>

    <name>dfs.namenode.http-address.ns1.nn2</name>
    <value>bigdata-senior02.chybinmy.com:50070</value>
  </property>
  <property>

    <name>dfs.namenode.shared.edits.dir</name>
    <value>qjournal://bigdata-senior01.chybinmy.com:8485;bigdata-
senior02.chybinmy.com:8485;bigdata-senior03.chybinmy.com:8485/ns1</value>
  </property>
  <property>

    <name>dfs.journalnode.edits.dir</name>
    <value>/opt/modules/hadoophadoop/hadoop-2.5.0/tmp/data/dfs/jn</value>
  </property>
  <property>

    <name>dfs.client.failover.proxy.provider.ns1</name>
    <value>org.apache.hadoop.hdfs.server.namenode.ha.ConfiguredFailoverProxyProvider</v
  </property>
  <property>

    <name>dfs.ha.fencing.methods</name>
    <value>sshfence</value>
  </property>
  <property>

    <name>dfs.ha.fencing.ssh.private-key-files</name>
    <value>/home/hadoop/.ssh/id_rsa</value>
  </property>

```

</property>
</configuration>

- 1
- 2
- 3
- 4
- 5
- 6
- 7
- 8
- 9
- 10
- 11
- 12
- 13
- 14
- 15
- 16
- 17
- 18
- 19
- 20
- 21
- 22
- 23
- 24
- 25
- 26
- 27
- 28
- 29
- 30
- 31
- 32
- 33
- 34
- 35
- 36
- 37
- 38
- 39
- 40
- 41
- 42
- 43
- 44
- 45
- 46
- 47
- 48
- 49
- 50
- 51
- 52
- 53
- 54
- 55
- 56
- 57

6、配置core-site.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
  <property>

    <name>fs.defaultFS</name>
    <value>hdfs://ns1</value>
  </property>
  <property>

    <name>hadoop.tmp.dir</name>
    <value>/opt/modules/hadoophadoop-2.5.0/data/tmp</value>
  </property>
</configuration>


- 1
- 2
- 3
- 4
- 5
- 6
- 7
- 8
- 9
- 10
- 11
- 12
- 13
- 14

```

`hadoop.tmp.dir` 设置hadoop临时目录地址，默认时，NameNode和DataNode的数据存在这个路径下。

7、配置slaves文件

```
bigdata-senior01.chybinmy.com
bigdata-senior02.chybinmy.com
bigdata-senior03.chybinmy.com


- 1
- 2
- 3

```

8、分发到其他节点

分发之前先将share/doc目录删除，这个目录中是帮助文件，并且很大，可以删除。

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ scp -r /opt/modules/hadoophadoop bigdata-senior02.chybinmy.com:/opt/modules
[hadoop@bigdata-senior01 hadoop-2.5.0]$ scp -r /opt/modules/hadoophadoop bigdata-senior03.chybinmy.com:/opt/modules


- 1
- 2

```

9、启动HDFS HA集群

三台机器分别启动Journalnode。

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ sbin/hadoop-daemon.sh start journalnode
[hadoop@bigdata-senior02 hadoop-2.5.0]$ sbin/hadoop-daemon.sh start journalnode
[hadoop@bigdata-senior03 hadoop-2.5.0]$ sbin/hadoop-daemon.sh start journalnode
  • 1
  • 2
  • 3
```

jps命令查看是否启动。

10、启动Zookeeper

在三台节点上启动Zookeeper：

```
[hadoop@bigdata-senior01 zookeeper-3.4.8]$ bin/zkServer.sh start
[hadoop@bigdata-senior02 zookeeper-3.4.8]$ bin/zkServer.sh start
[hadoop@bigdata-senior03 zookeeper-3.4.8]$ bin/zkServer.sh start
  • 1
  • 2
  • 3
```

11、格式化NameNode

在第一台上进行NameNode格式化：

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ bin/hdfs namenode -format
1
```

在第二台NameNode上：

```
[hadoop@bigdata-senior02 hadoop-2.5.0]$ bin/hdfs namenode -bootstrapStandby
1
```

12、启动NameNode

在第一台、第二台上启动NameNode：

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ sbin/hadoop-daemon.sh start namenode
[hadoop@bigdata-senior02 hadoop-2.5.0]$ sbin/hadoop-daemon.sh start namenode
  • 1
  • 2
```

查看HDFS Web页面，此时两个NameNode都是standby状态。

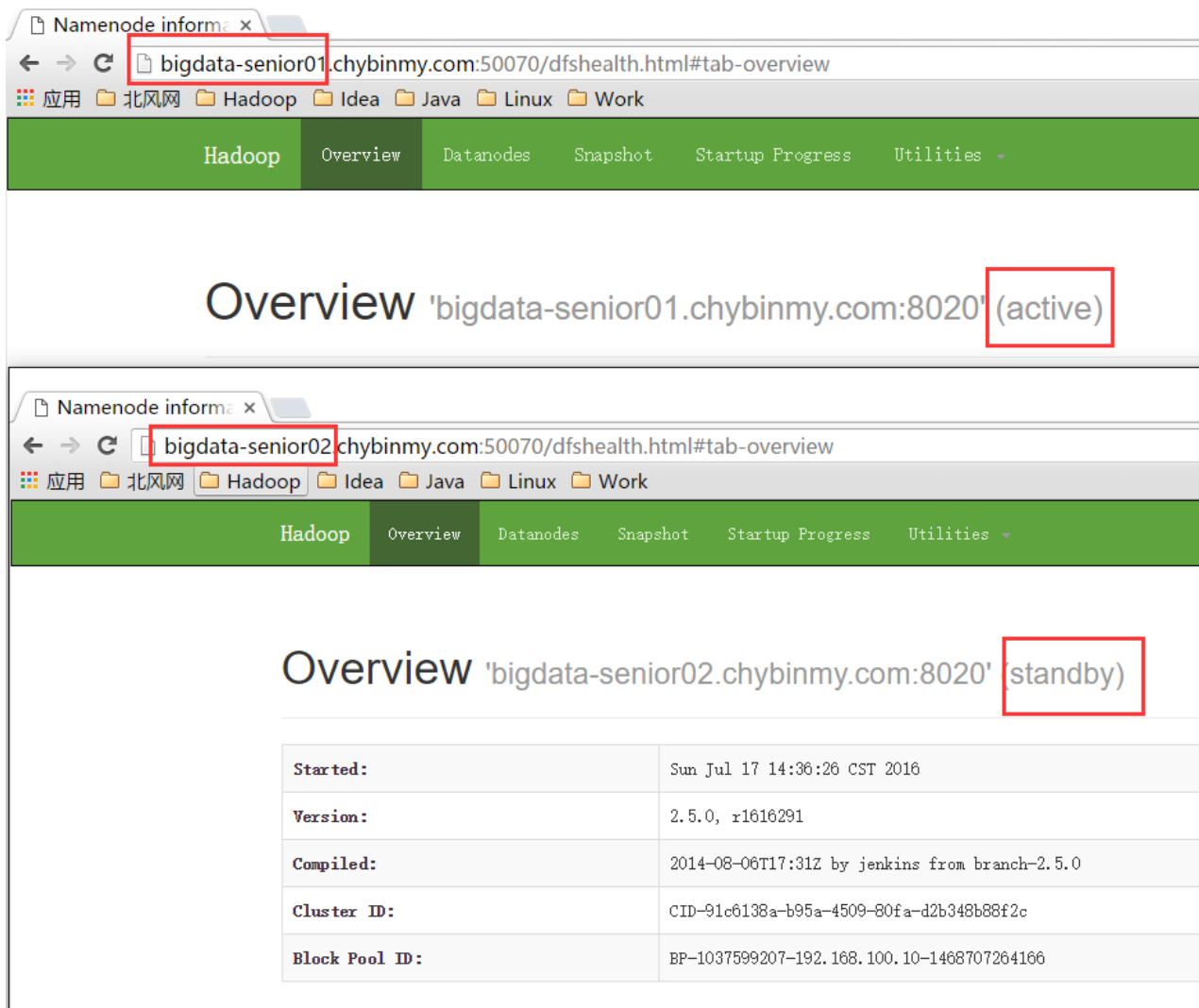
切换第一台为active状态：

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ bin/hdfs haadmin -transitionToActive nn1
1
```

可以添加上forcemanual参数，强制将一个NameNode转换为Active状态。

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ bin/hdfs haadmin -transitionToActive -forcemanual nn1  
1
```

此时从web 页面就看到第一台已经是active状态了。



13、配置故障自动转移

利用zookeeper集群实现故障自动转移，在配置故障自动转移之前，要先关闭集群，不能在HDFS运行期间进行配置。

关闭NameNode、DataNode、JournalNode、zookeeper

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ sbin/hadoop-daemon.sh stop namenode
[hadoop@bigdata-senior01 hadoop-2.5.0]$ sbin/hadoop-daemon.sh stop datanode
[hadoop@bigdata-senior01 hadoop-2.5.0]$ sbin/hadoop-daemon.sh stop journalnode
[hadoop@bigdata-senior01 hadoop-2.5.0]$ cd ../../zookeeper-3.4.8/
[hadoop@bigdata-senior01 zookeeper-3.4.8]$ bin/zkServer.sh stop
[hadoop@bigdata-senior02 hadoop-2.5.0]$ sbin/hadoop-daemon.sh stop namenode
[hadoop@bigdata-senior02 hadoop-2.5.0]$ sbin/hadoop-daemon.sh stop datanode
[hadoop@bigdata-senior02 hadoop-2.5.0]$ sbin/hadoop-daemon.sh stop journalnode
[hadoop@bigdata-senior02 hadoop-2.5.0]$ cd ../../zookeeper-3.4.8/
[hadoop@bigdata-senior02 zookeeper-3.4.8]$ bin/zkServer.sh stop
[hadoop@bigdata-senior03 hadoop-2.5.0]$ sbin/hadoop-daemon.sh stop datanode
[hadoop@bigdata-senior03 hadoop-2.5.0]$ sbin/hadoop-daemon.sh stop journalnode
[hadoop@bigdata-senior03 hadoop-2.5.0]$ cd ../../zookeeper-3.4.8/
[hadoop@bigdata-senior03 zookeeper-3.4.8]$ bin/zkServer.sh stop
```

- 1
- 2
- 3
- 4
- 5
- 6
- 7
- 8
- 9
- 10
- 11
- 12
- 13
- 14

修改hdfs-site.xml

```
<property>
  <name>dfs.ha.automatic-failover.enabled</name>
  <value>>true</value>
</property>
```

- 1
- 2
- 3
- 4

修改core-site.xml

```
<property>
  <name>ha.zookeeper.quorum</name>
  <value>bigdata-senior01.chybinmy.com:2181,bigdata-
senior02.chybinmy.com:2181,bigdata-senior03.chybinmy.com:2181</value>
</property>
```

- 1
- 2
- 3
- 4

将hdfs-site.xml和core-site.xml分发到其他机器

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ scp /opt/modules/hadoopha/hadoop-2.5.0/etc/hadoop/hdfs-site.xml bigdata-senior02.chybinmy.com:/opt/modules/hadoopha/hadoop-2.5.0/etc/hadoop/
[hadoop@bigdata-senior01 hadoop-2.5.0]$ scp /opt/modules/hadoopha/hadoop-2.5.0/etc/hadoop/hdfs-site.xml bigdata-senior03.chybinmy.com:/opt/modules/hadoopha/hadoop-2.5.0/etc/hadoop/
[hadoop@bigdata-senior01 hadoop-2.5.0]$ scp /opt/modules/hadoopha/hadoop-2.5.0/etc/hadoop/core-site.xml bigdata-senior02.chybinmy.com:/opt/modules/hadoopha/hadoop-2.5.0/etc/hadoop/
[hadoop@bigdata-senior01 hadoop-2.5.0]$ scp /opt/modules/hadoopha/hadoop-2.5.0/etc/hadoop/core-site.xml bigdata-senior03.chybinmy.com:/opt/modules/hadoopha/hadoop-2.5.0/etc/hadoop/
```

- 1
- 2
- 3
- 4

启动zookeeper

三台机器启动zookeeper

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ /opt/modules/zookeeper-3.4.8/bin/zkServer.sh start
1
```

创建一个zNode

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ cd /opt/modules/hadoopha/hadoop-2.5.0/
[hadoop@bigdata-senior01 hadoop-2.5.0]$ bin/hdfs zkfc -formatZK
```

- 1
- 2

```
16/07/17 15:55:59 INFO zookeeper.ClientCnxn: Session establishment complete on server bigdat
nid = 0x255f7d759e20000, negotiated timeout = 5000
16/07/17 15:55:59 INFO ha.ActiveStandbyElector: Session connected.
16/07/17 15:55:59 INFO ha.ActiveStandbyElector: Successfully created /hadoop-ha/ns1 in ZK.
16/07/17 15:55:59 INFO zookeeper.ZooKeeper: Session: 0x255f7d759e20000 closed
16/07/17 15:55:59 INFO zookeeper.ClientCnxn: EventThread shut down
```

在Zookeeper上创建一个存储namenode相关的节点。

14、启动HDFS、JournalNode、zkfc

启动NameNode、DataNode、JournalNode、zkfc

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ sbin/start-dfs.sh
1
```

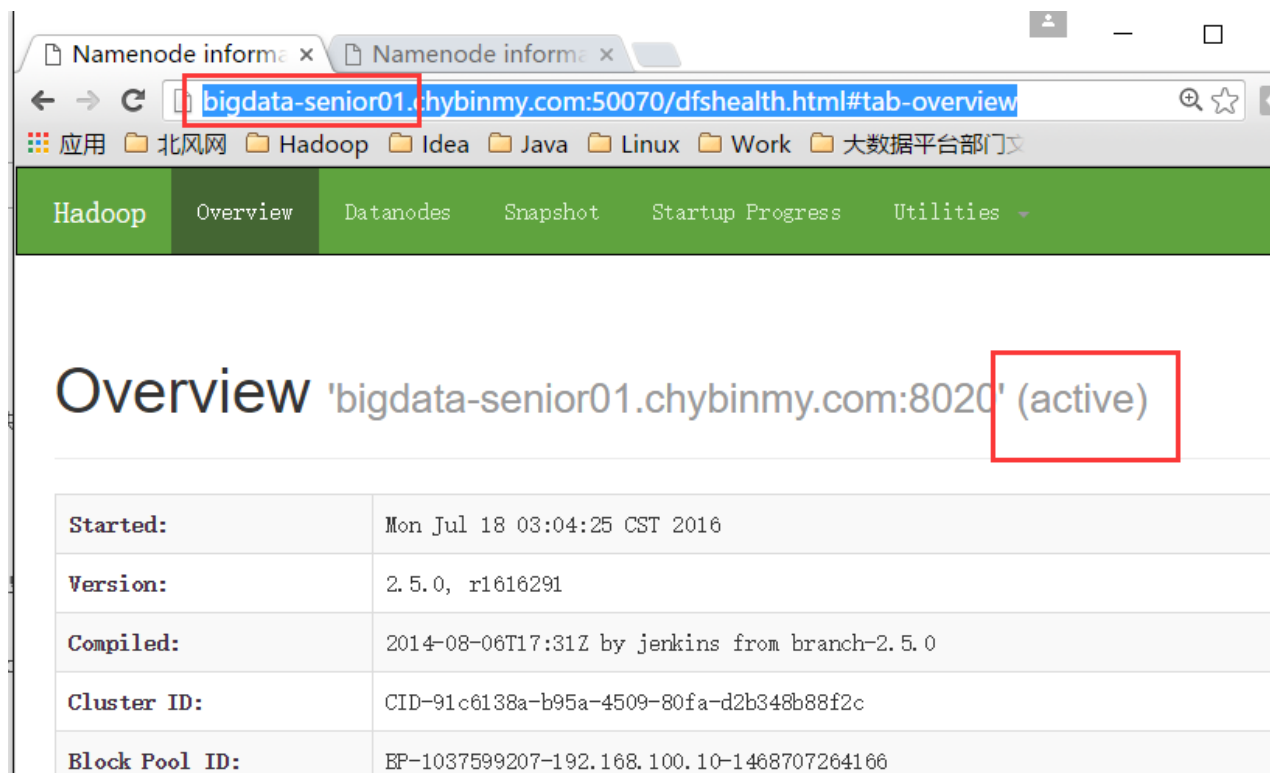
zkfc只针对NameNode监听。

四十二、测试HDFS HA

1、测试故障自动转移和数据是否共享

在nn1上上传文件

目前bigdata-senior01节点上的NameNode是Active状态的。



```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ bin/hdfs dfs -put /opt/data/wc.input /  
1
```

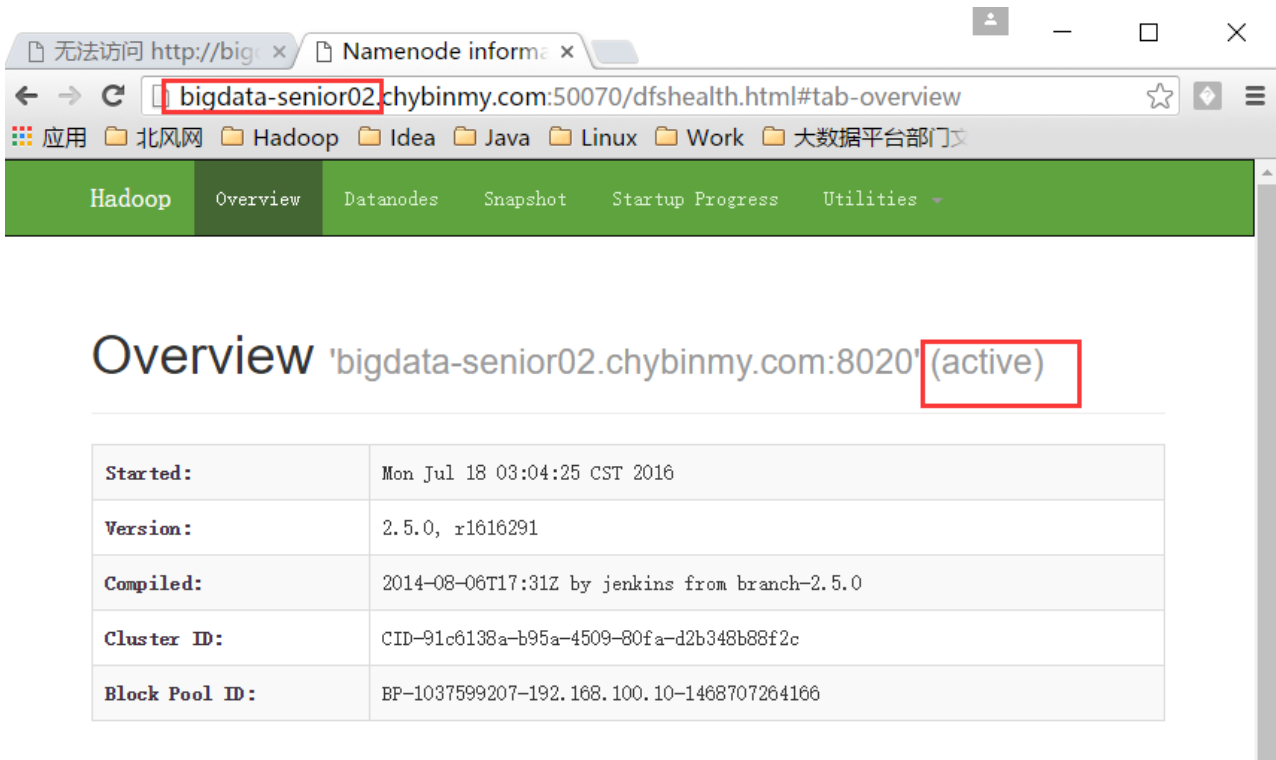


将nn1上的NameNode进程杀掉

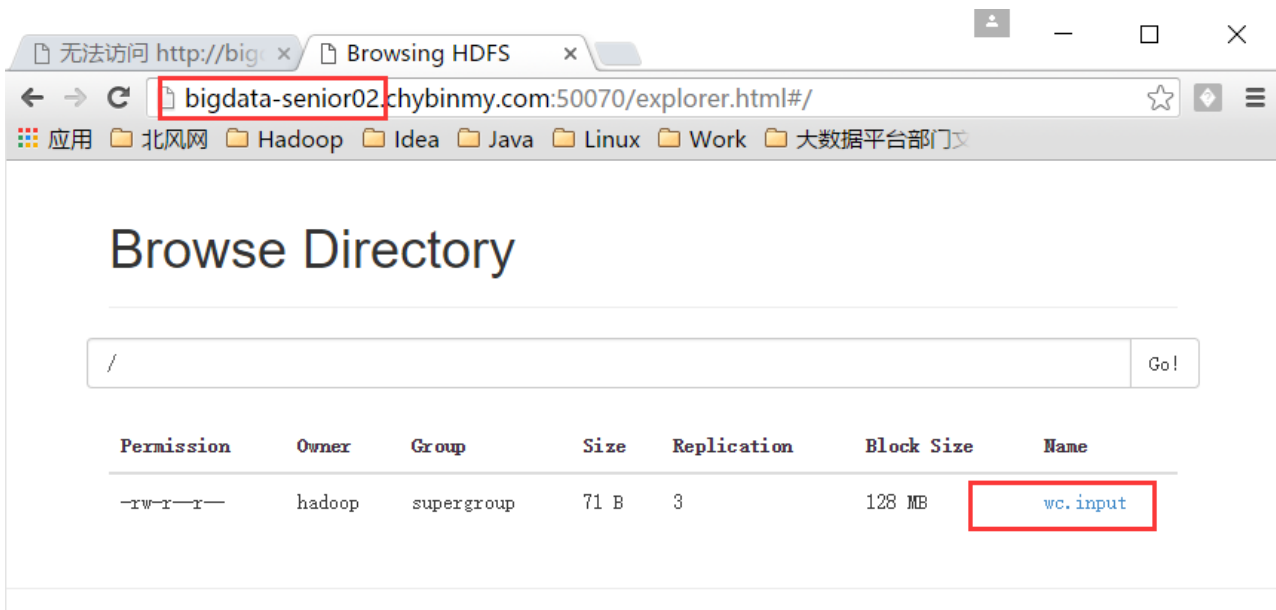
```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ kill -9 3364  
1
```

nn1上的namenode已经无法访问了。

查看nn2是否是Active状态



在nn2上查看是否看见文件



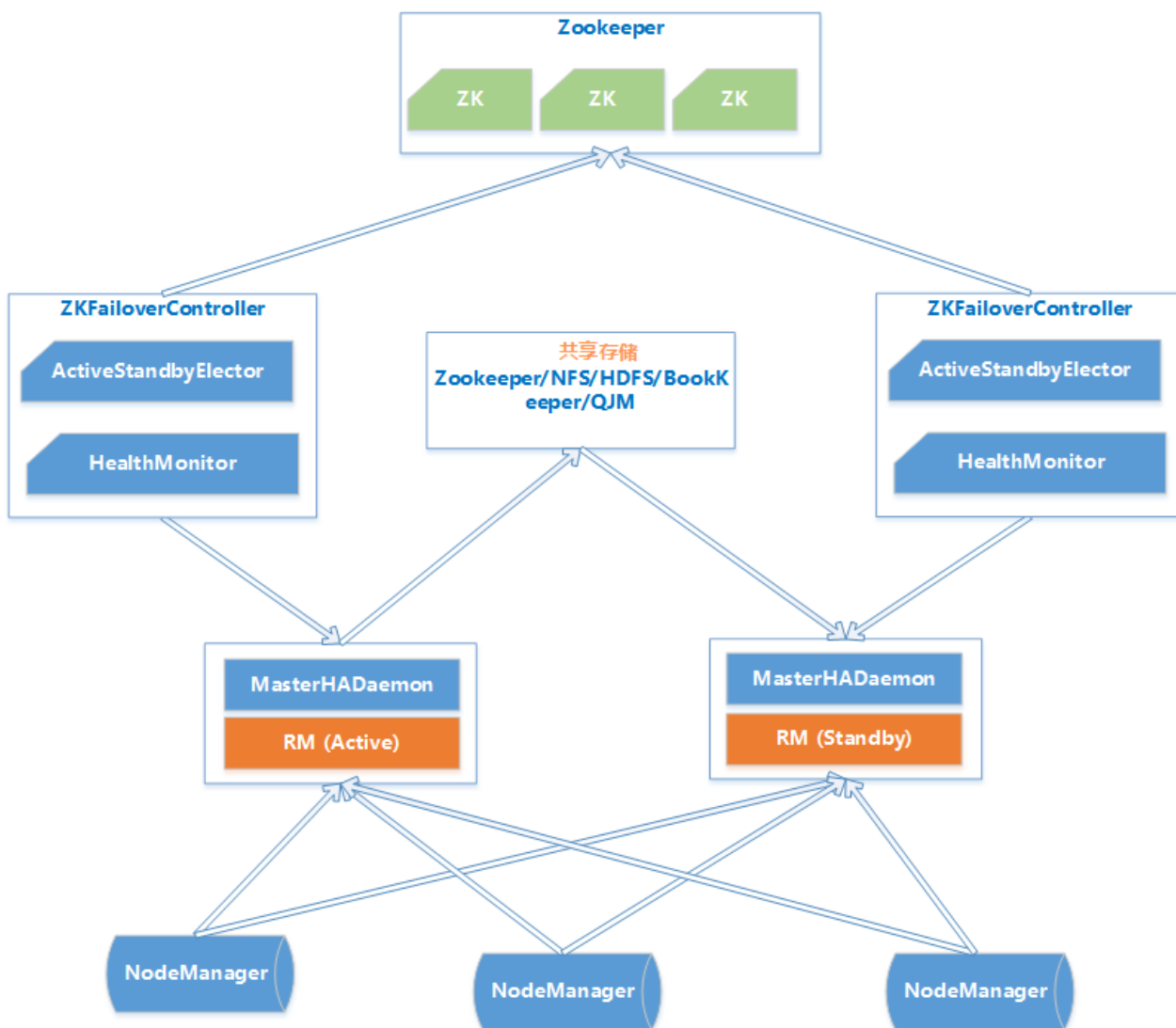
经以上验证，已经实现了nn1和nn2之间的文件同步和故障自动转移。

第十二步、Hadoop 2.x YARN HA 部署

四十三、YARN HA原理

Hadoop2.4版本之前，ResourceManger也存在单点故障的问题，也需要实现HA来保证ResourceManger的高可用性。

ResourceManager从记录着当前集群的资源分配情况和JOB的运行状态，YARN HA 利用 Zookeeper等共享存储介质来存储这些信息来达到高可用。另外利用Zookeeper来实现 ResourceManager自动故障转移。



- MasterHADAemon：控制RM的 Master的启动和停止，和RM运行在一个进程中，可以接收外部RPC命令。
- 共享存储：Active Master将信息写入共享存储，Standby Master读取共享存储信息以保持和Active Master同步。
- ZKFailoverController：基于Zookeeper实现的切换控制器，由ActiveStandbyElector和HealthMonitor组成，ActiveStandbyElector负责与Zookeeper交互，判断所管理的Master是进入Active还是Standby；HealthMonitor负责监控Master的活动健康情况，是个监视器。
- Zookeeper：核心功能是维护一把全局锁控制整个集群上只有一个Active的ResourceManager。

四十四、搭建YARN HA环境

1、服务器角色规划

**bigdata-
senior01.chyblnmy.com**

**bigdata-
senior01.chyblnmy.com**

**bigdata-
senior01.chyblnmy.com**

NameNode

NameNode

Zookeeper

Zookeeper

Zookeeper

DataNode

DataNode

DataNode

ResourceManage

ResourceManage

NodeManager

NodeManager

NodeManager

2、修改配置文件yarn-site.xml

```

<?xml version="1.0" encoding="UTF-8"?>
<configuration>
  <property>
    <name>yarn.nodemanager.aux-services</name>
    <value>mapreduce_shuffle</value>
  </property>
  <property>
    <name>yarn.log-aggregation-enable</name>
    <value>true</value>
  </property>
  <property>
    <name>yarn.log-aggregation.retain-seconds</name>
    <value>106800</value>
  </property>
  <property>

    <name>yarn.resourcemanager.ha.enabled</name>
    <value>true</value>
  </property>
  <property>

    <name>yarn.resourcemanager.cluster-id</name>
    <value>yarn-cluster</value>
  </property>
  <property>

    <name>yarn.resourcemanager.ha.rm-ids</name>
    <value>rm12,rm13</value>
  </property>
  <property>

    <name>yarn.resourcemanager.hostname.rm12</name>
    <value>bigdata-senior02.chybinmy.com</value>
  </property>
  <property>

    <name>yarn.resourcemanager.hostname.rm13</name>
    <value>bigdata-senior03.chybinmy.com</value>
  </property>
  <property>

    <name>yarn.resourcemanager.zk-address</name>
    <value>bigdata-senior01.chybinmy.com:2181,bigdata-
senior02.chybinmy.com:2181,bigdata-senior03.chybinmy.com:2181</value>
  </property>
  <property>

    <name>yarn.resourcemanager.recovery.enabled</name>
    <value>true</value>
  </property>
  <property>

    <name>yarn.resourcemanager.store.class</name>

<value>org.apache.hadoop.yarn.server.resourcemanager.recovery.ZKRMStateStore</value>

  </property>
</configuration>

```

- 1

- 2
- 3
- 4
- 5
- 6
- 7
- 8
- 9
- 10
- 11
- 12
- 13
- 14
- 15
- 16
- 17
- 18
- 19
- 20
- 21
- 22
- 23
- 24
- 25
- 26
- 27
- 28
- 29
- 30
- 31
- 32
- 33
- 34
- 35
- 36
- 37
- 38
- 39
- 40
- 41
- 42
- 43
- 44
- 45
- 46
- 47
- 48
- 49
- 50
- 51
- 52
- 53
- 54
- 55
- 56

3、分发到其他机器

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ scp /opt/modules/hadoopha/hadoop-2.5.0/etc/hadoop/yarn-site.xml bigdata-senior02.chybinmy.com:/opt/modules/hadoopha/hadoop-2.5.0/etc/hadoop/
[hadoop@bigdata-senior01 hadoop-2.5.0]$ scp /opt/modules/hadoopha/hadoop-2.5.0/etc/hadoop/yarn-site.xml bigdata-senior03.chybinmy.com:/opt/modules/hadoopha/hadoop-2.5.0/etc/hadoop/\
  • 1
  • 2
```

4、启动

在bigdata-senior01上启动yarn：

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ sbin/start-yarn.sh
1
```

在bigdata-senior02、bigdata-senior03上启动resource manager：

```
[hadoop@bigdata-senior02 hadoop-2.5.0]$ sbin/yarn-daemon.sh start resourcemanager
[hadoop@bigdata-senior03 hadoop-2.5.0]$ sbin/yarn-daemon.sh start resourcemanager
  • 1
  • 2
```

启动后各个节点的进程。

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ jps
5538 Jps
4402 NameNode
3658 JournalNode
5388 NodeManager
2073 QuorumPeerMain
3806 DFSZKFailoverController
3468 DataNode
```

```
[hadoop@bigdata-senior02 hadoop-2.5.0]$ jps
2640 NameNode
4314 NodeManager
2073 QuorumPeerMain
2709 DataNode
4742 Jps
2894 DFSZKFailoverController
4475 ResourceManager
2805 JournalNode
```

```
[hadoop@bigdata-senior03 hadoop-2.5.0]$ jps
3339 Jps
2359 DataNode
2451 JournalNode
3273 ResourceManager
3137 NodeManager
2079 QuorumPeerMain
```

Web客户端访问bigdata02机器上的resourcemanager正常，它是active状态的。

<http://bigdata-senior02.chybinmy.com:8088/cluster>

访问另外一个resourcemanager，因为他是standby,会自动跳转到active的resourcemanager。

<http://bigdata-senior03.chybinmy.com:8088/cluster>

四十五、测试YARN HA

5、运行一个mapreduce job

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ bin/yarn jar
share/hadoop/mapreduce/hadoop-mapreduce-examples-2.5.0.jar wordcount /wc.input
/input
1
```

6、在job运行过程中，将Active状态的resourcemanager进程杀掉。

```
[hadoop@bigdata-senior02 hadoop-2.5.0]$ kill -9 4475
1
```

7、观察另外一个resourcemanager是否可以自动接替。

bigdata02的resourcemanager Web客户端已经不能访问，bigdata03的resourcemanager已经自动变为active状态。

8、观察job是否可以顺利完成。

而mapreduce job 也能顺利完成，没有因为resourcemanager的意外故障而影响运行。

经过以上测试，已经验证YARN HA 已经搭建成功。

第十三步、HDFS Federation 架构部署

四十六、HDFS Federation 的使用原因

1、单个NameNode节点的局限性

命名空间的限制。

NameNode上存储着整个HDFS上的文件的元数据，NameNode是部署在一台机器上的，因为单个机器硬件的限制，必然会限制NameNode所能管理的文件个数，制约了数据量的增长。

数据隔离问题。

整个HDFS上的文件都由一个NameNode管理，所以一个程序很有可能会影响到整个HDFS上的程序，并且权限控制比较复杂。

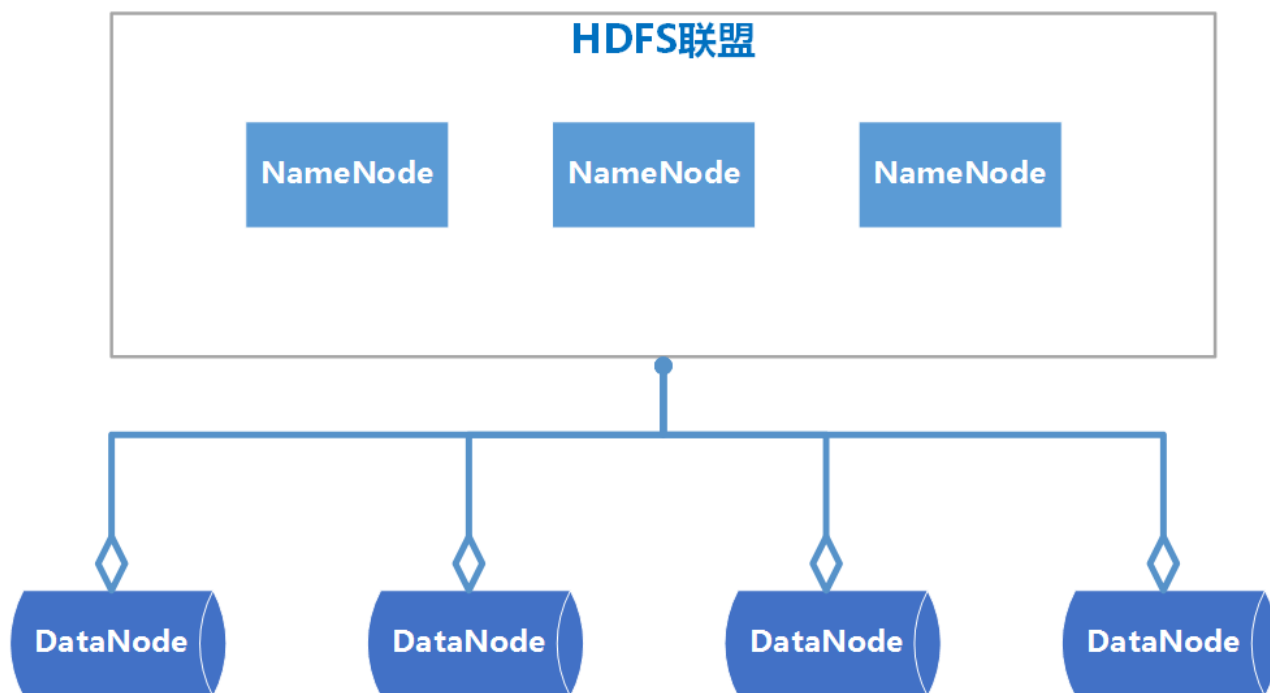
性能瓶颈。

单个NameNode时HDFS文件系统的吞吐量受限于单个NameNode的吞吐量。因为NameNode是个JVM进程，JVM进程所占用的内存很大时，性能会下降很多。

2、HDFS Federation介绍

HDFS Federation是可以在Hadoop集群中设置多个NameNode，不同于HA中多个NameNode是完全一样的，是多个备份，Federation中的多个NameNode是不同的，可以理解为将一个NameNode切分为了多个NameNode，每一个NameNode只负责管理一部分数据。HDFS Federation中的多个NameNode共用DataNode。

四十七、HDFS Federation的架构图



四十八、HDFS Federation搭建

1、服务器角色规划

| | | |
|--------------------------------------|--------------------------------------|--------------------------------------|
| bigdata-senior01.chybinmy.com | bigdata-senior01.chybinmy.com | bigdata-senior01.chybinmy.com |
| NameNode1 | NameNode2 | NameNode3 |
| ResourceManage | | |

| bigdata-senior01.chybinmy.com | bigdata-senior01.chybinmy.com | bigdata-senior01.chybinmy.com |
|--------------------------------------|--------------------------------------|--------------------------------------|
| DataNode | DataNode | DataNode |
| NodeManager | NodeManager | NodeManager |

2、创建HDFS Federation 版本Hadoop程序目录

在bigdata01上创建目录/opt/modules/hadoopfederation /用来存放Hadoop Federation环境。

```
[hadoop@bigdata-senior01 modules]$ mkdir /opt/modules/hadoopfederation
1
```

3、新解压Hadoop 2.5.0

```
[hadoop@bigdata-senior01 ~]$ tar -zxf /opt/sofeware/hadoop-2.5.0.tar.gz -C
/opt/modules/hadoopfederation/
1
```

4、配置Hadoop JDK路径

修改hadoop-env.sh、mapred-env.sh、yarn-env.sh文件中的JDK路径。

```
export JAVA_HOME="/opt/modules/jdk1.7.0_67"
```

5、配置hdfs-site.xml

```

<configuration>
<property>
<!--配置三台NameNode -->
    <name>dfs.nameservices</name>
    <value>ns1,ns2,ns3</value>
</property>
<property>
<!--第一台NameNode的机器名和rpc端口, 指定了NameNode和DataNode通讯用的端口号 -->
    <name>dfs.namenode.rpc-address.ns1</name>
    <value>bigdata-senior01.chybinmy.com:8020</value>
</property>
<property>
<!--第一台NameNode的机器名和rpc端口, 备用端口号 -->
    <name>dfs.namenode.serviceerpc-address.ns1</name>
    <value>bigdata-senior01.chybinmy.com:8022</value>
</property>
<property>
<!--第一台NameNode的http页面地址和端口号 -->
    <name>dfs.namenode.http-address.ns1</name>
    <value>bigdata-senior01.chybinmy.com:50070</value>
</property>
<property>
<!--第一台NameNode的https页面地址和端口号 -->
    <name>dfs.namenode.https-address.ns1</name>
    <value>bigdata-senior01.chybinmy.com:50470</value>
</property>

<property>
    <name>dfs.namenode.rpc-address.ns2</name>
    <value>bigdata-senior02.chybinmy.com:8020</value>
</property>
<property>
    <name>dfs.namenode.serviceerpc-address.ns2</name>
    <value>bigdata-senior02.chybinmy.com:8022</value>
</property>
<property>
    <name>dfs.namenode.http-address.ns2</name>
    <value>bigdata-senior02.chybinmy.com:50070</value>
</property>
<property>
    <name>dfs.namenode.https-address.ns2</name>
    <value>bigdata-senior02.chybinmy.com:50470</value>
</property>

<property>
    <name>dfs.namenode.rpc-address.ns3</name>
    <value>bigdata-senior03.chybinmy.com:8020</value>
</property>
<property>
    <name>dfs.namenode.serviceerpc-address.ns3</name>
    <value>bigdata-senior03.chybinmy.com:8022</value>
</property>
<property>
    <name>dfs.namenode.http-address.ns3</name>
    <value>bigdata-senior03.chybinmy.com:50070</value>
</property>
<property>
    <name>dfs.namenode.https-address.ns3</name>

```



```
<value>bigdata-senior03.chybinmy.com:50470</value>  
</property>
```

```
</configuration>
```

- 1
- 2
- 3
- 4
- 5
- 6
- 7
- 8
- 9
- 10
- 11
- 12
- 13
- 14
- 15
- 16
- 17
- 18
- 19
- 20
- 21
- 22
- 23
- 24
- 25
- 26
- 27
- 28
- 29
- 30
- 31
- 32
- 33
- 34
- 35
- 36
- 37
- 38
- 39
- 40
- 41
- 42
- 43
- 44
- 45
- 46
- 47
- 48
- 49
- 50
- 51
- 52
- 53
- 54
- 55

- 56
- 57
- 58
- 59
- 60
- 61
- 62
- 63

6、配置core-site.xml

```
<configuration>
<property>
  <name>hadoop.tmp.dir</name>
  <value>/opt/modules/hadoopha/hadoop-2.5.0/data/tmp</value>
</property>
</configuration>
```

- 1
- 2
- 3
- 4
- 5
- 6

hadoop.tmp.dir设置hadoop临时目录地址，默认时，NameNode和DataNode的数据存在这个路径下。

7、配置slaves文件

```
bigdata-senior01.chybinmy.com
bigdata-senior02.chybinmy.com
bigdata-senior03.chybinmy.com
```

- 1
- 2
- 3

8、配置yarn-site.xml

```

<configuration>
  <property>
    <name>yarn.nodemanager.aux-services</name>
    <value>mapreduce_shuffle</value>
  </property>
  <property>
    <name>yarn.resourcemanager.hostname</name>
    <value>bigdata-senior02.chybinmy.com</value>
  </property>
  <property>
    <name>yarn.log-aggregation-enable</name>
    <value>true</value>
  </property>
  <property>
    <name>yarn.log-aggregation.retain-seconds</name>
    <value>106800</value>
  </property>
</configuration>

```

- 1
- 2
- 3
- 4
- 5
- 6
- 7
- 8
- 9
- 10
- 11
- 12
- 13
- 14
- 15
- 16
- 17
- 18

9、分发到其他节点

分发之前先将share/doc目录删除，这个目录中是帮助文件，并且很大，可以删除。

```

[hadoop@bigdata-senior01 hadoop-2.5.0]$ scp -r /opt/modules/
/opt/modules/hadoopfederation bigdata-senior02.chybinmy.com:/opt/modules
[hadoop@bigdata-senior01 hadoop-2.5.0]$ scp -r /opt/modules/hadoopfederation
bigdata-senior03.chybinmy.com:/opt/modules

```

- 1
- 2

10、格式化NameNode

在第一台上进行NameNode格式化。

```

[hadoop@bigdata-senior01 hadoop-2.5.0]$ bin/hdfs namenode -format -clusterId
hadoop-federation-clusterId

```

1

这里一定要指定一个集群ID，使得多个NameNode的集群ID是一样的，因为这三个NameNode在同一个集群中，这里集群ID为hadoop-federation-clusterId。

在第二台NameNode上。

```
[hadoop@bigdata-senior02 hadoop-2.5.0]$ bin/hdfs namenode -format -clusterId
hadoop-federation-clusterId
1
```

在第二台NameNode上。

```
[hadoop@bigdata-senior03 hadoop-2.5.0]$ bin/hdfs namenode -format -clusterId
hadoop-federation-clusterId
1
```

11、启动NameNode

在第一台、第二台、第三台机器上启动NameNode：

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ sbin/hadoop-daemon.sh start namenode
[hadoop@bigdata-senior02 hadoop-2.5.0]$ sbin/hadoop-daemon.sh start namenode
[hadoop@bigdata-senior03 hadoop-2.5.0]$ sbin/hadoop-daemon.sh start namenode
• 1
• 2
• 3
```

启动后，用jps命令查看是否已经启动成功。

查看HDFS Web页面，此时三个NameNode都是standby状态。

[Namenode informa x](#)
[Namenode informa x](#)
[Namenode informa x](#)

[senior01.chybinmy.com:50070/dfshealth.html#tab-overview](#)

[Hadoop](#)
[Idea](#)
[Java](#)
[Linux](#)
[Work](#)
[大数据部门](#)
[VWare](#)
[Hive](#)

Hadoop
[Overview](#)
[Datanodes](#)
[Snapshot](#)
[Startup Progress](#)
[Utilities](#)

Overview 'bigdata-senior01.chybinmy.com:8020' (active)

| | |
|-----------------------|--|
| Started: | Sun Sep 25 22:22:15 CST 2016 |
| Version: | 2.5.0, r1616291 |
| Compiled: | 2014-08-06T17:31Z by jenkins from branch-2.5.0 |
| Cluster ID: | hadoop-federation-clusterId |
| Block Pool ID: | BP-1649265296-192.168.100.10-1474813121909 |

[Namenode informa x](#)
[Namenode informa x](#)
[Namenode informa x](#)

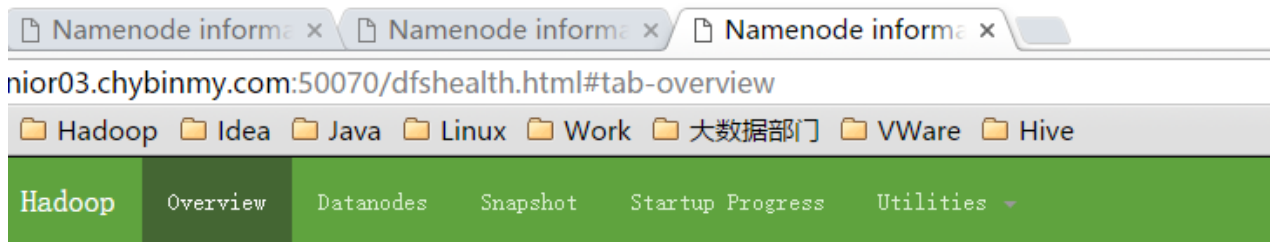
[senior02.chybinmy.com:50070/dfshealth.html#tab-overview](#)

[Hadoop](#)
[Idea](#)
[Java](#)
[Linux](#)
[Work](#)
[大数据部门](#)
[VWare](#)
[Hive](#)

Hadoop
[Overview](#)
[Datanodes](#)
[Snapshot](#)
[Startup Progress](#)
[Utilities](#)

Overview 'bigdata-senior02.chybinmy.com:8020' (active)

| | |
|-----------------------|--|
| Started: | Sun Sep 25 22:22:29 CST 2016 |
| Version: | 2.5.0, r1616291 |
| Compiled: | 2014-08-06T17:31Z by jenkins from branch-2.5.0 |
| Cluster ID: | hadoop-federation-clusterId |
| Block Pool ID: | BP-31463253-192.168.100.12-1474813259565 |



Overview 'bigdata-senior03.chybinmy.com:8020' (active)

| | |
|-----------------------|--|
| Started: | Sun Sep 25 22:22:42 CST 2016 |
| Version: | 2.5.0, r1616291 |
| Compiled: | 2014-08-06T17:31Z by jenkins from branch-2.5.0 |
| Cluster ID: | hadoop-federation-clusterId |
| Block Pool ID: | BP-1481416476-192.168.100.13-1474813302641 |

12、启动DataNode

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ sbin/hadoop-daemon.sh start datanode
[hadoop@bigdata-senior02 hadoop-2.5.0]$ sbin/hadoop-daemon.sh start datanode
[hadoop@bigdata-senior03 hadoop-2.5.0]$ sbin/hadoop-daemon.sh start datanode
  • 1
  • 2
  • 3
```

启动后，用jps命令确认DataNode进程已经启动成功。

四十九、测试HDFS Federation

1、修改core-site.xml

在bigdata-senior01机器上,修改core-site.xml文件，指定连接的NameNode是第一台NameNode。

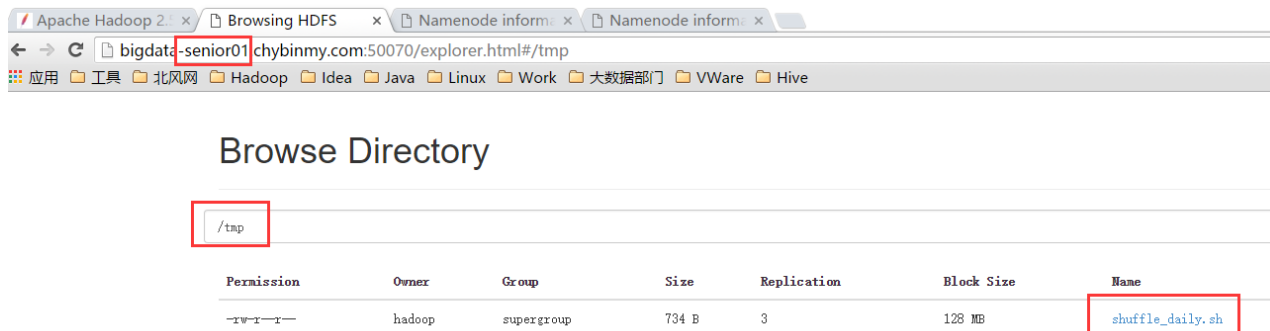
```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ vim etc/hadoop/core-site.xml
```

```
<configuration>
  <property>
    <name>fs.defaultFS</name>
    <value>hdfs://bigdata-senior01.chybinmy.com:8020</value>
  </property>
  <property>
    <name>hadoop.tmp.dir</name>
    <value>/opt/modules/hadoopfederation/hadoop-2.5.0/data/tmp</value>
  </property>
</configuration>
```

2、在bigdate-senior01上传一个文件到HDFS

```
[hadoop@bigdata-senior01 hadoop-2.5.0]$ bin/hdfs dfs -mkdir /tmp
[hadoop@bigdata-senior01 hadoop-2.5.0]$ bin/hdfs dfs -put ~/shuffle_daily.sh
/tmp/shuffle_daily.sh
  • 1
  • 2
```

3、查看HDFS文件



可以看到，刚才的文件只上传到了bigdate-senior01机器上的NameNode上了，并没有上传到其他NameNode上去。

这样，在HDFS的客户端，可以指定要上传到哪个NameNode上，从而来达到划分NameNode的目的。

后记

这篇文章的操作步骤并不是工作中标准的操作流程，如果在成百上千的机器全部这样安装会被累死，希望读者可以通过文章中一步步地安装，从而初步了解到Hadoop的组成部分，协助过程等，这对于Hadoop的深入使用有很大的帮助。

实录：《鸣宇淳：搭建Hadoop学习环境实战解析》