Apache　Hadoop 三种架构介绍（高可用分布式环境介绍以及安装）

1.3、高可用分布式环境搭建（适用于工作当中正式环境搭建）

实现 namenode 高可用，ResourceManager 的高可用

集群运行服务规划

	192.168.52.100	192.168.52.110	192.168.52.120
zookeeper	zk	zk	zk
HDFS	JournalNode	JournalNode	JournalNode
	NameNode	NameNode	
	ZKFC	ZKFC	
	DataNode	DataNode	DataNode
YARN		ResourceManager	ResourceManager
	NodeManager	NodeManager	NodeManager
MapReduce			JobHistoryServer

安装包解压

停止之前的 hadoop 集群的所有服务，并删除所有机器的 hadoop 安装包，

第一台机器执行以下命令

```
cd /export/servers/hadoop-2.7.5
sbin/stop-dfs.sh
```

```
sbin/stop-yarn.sh
sbin/mr-jobhistory-daemon.sh stop historyserver
```

cd /export/servers/

rm -rf hadoop-2.7.5/

```
[root@node04 servers]# rm -rf hadoop-2.7.5/
```

然后重新解压 hadoop 压缩包

解压压缩包

第一台机器执行以下命令进行解压

```
cd /export/softwares
tar -zxvf hadoop-2.7.5.tar.gz -C ../servers/
```

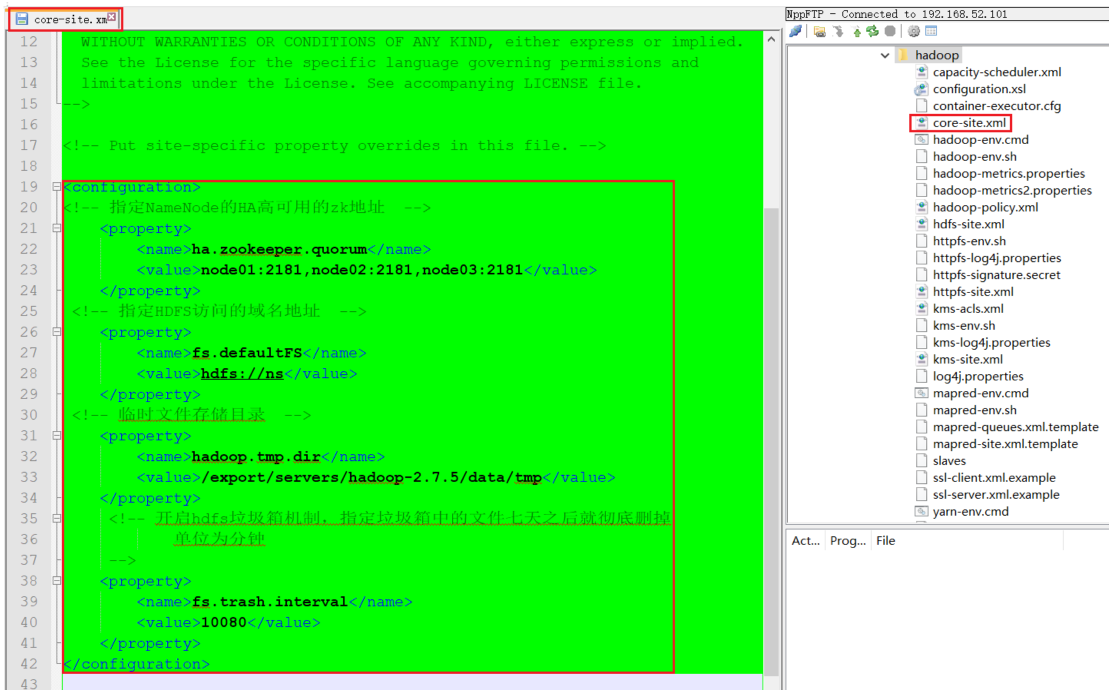
配置文件的修改

修改 core-site.xml

第一台机器执行以下命令

```
cd /export/servers/hadoop-2.7.5/etc/hadoop
vim core-site.xml
```

```
<configuration>
<!--  指定 NameNode 的 HA 高可用的 zk 地址   -->
    <property>
        <name>ha.zookeeper.quorum</name>
        <value>node01:2181,node02:2181,node03:2181</value>
    </property>
 <!--  指定 HDFS 访问的域名地址   -->
    <property>
        <name>fs.defaultFS</name>
        <value>hdfs://ns</value>
    </property>
 <!--  临时文件存储目录   -->
    <property>
        <name>hadoop.tmp.dir</name>
        <value>/export/servers/hadoop-2.7.5/data/tmp</value>
    </property>
     <!--  开启 hdfs 垃圾箱机制，指定垃圾箱中的文件七天之后就彻底删
掉
            单位为分钟
     -->
    <property>
        <name>fs.trash.interval</name>
        <value>10080</value>
    </property>
</configuration>
```



core-site.xml

```
  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>
<!-- 指定NameNode的HA高可用的zk地址  -->
    <property>
        <name>ha.zookeeper.quorum</name>
        <value>node01:2181,node02:2181,node03:2181</value>
    </property>
 <!-- 指定HDFS访问的域名地址  -->
    <property>
        <name>fs.defaultFS</name>
        <value>hdfs://ns</value>
    </property>
 <!-- 临时文件存储目录   -->
    <property>
        <name>hadoop.tmp.dir</name>
        <value>/export/servers/hadoop-2.7.5/data/tmp</value>
    </property>
     <!-- 开启hdfs垃圾箱机制，指定垃圾箱中的文件七天之后就彻底删掉
            单位为分钟
     -->
    <property>
        <name>fs.trash.interval</name>
        <value>10080</value>
    </property>
</configuration>
```

NppFTP - Connected to 192.168.52.101

- hadoop
  - capacity-scheduler.xml
  - configuration.xsl
  - container-executor.cfg
  - core-site.xml
  - hadoop-env.cmd
  - hadoop-env.sh
  - hadoop-metrics.properties
  - hadoop-metrics2.properties
  - hadoop-policy.xml
  - hdfs-site.xml
  - httpfs-env.sh
  - httpfs-log4j.properties
  - httpfs-signature.secret
  - httpfs-site.xml
  - kms-acls.xml
  - kms-env.sh
  - kms-log4j.properties
  - kms-site.xml
  - log4j.properties
  - mapred-env.cmd
  - mapred-env.sh
  - mapred-queues.xml.template
  - mapred-site.xml.template
  - slaves
  - ssl-client.xml.example
  - ssl-server.xml.example
  - yarn-env.cmd

Act... Prog... File

修改 hdfs-site.xml

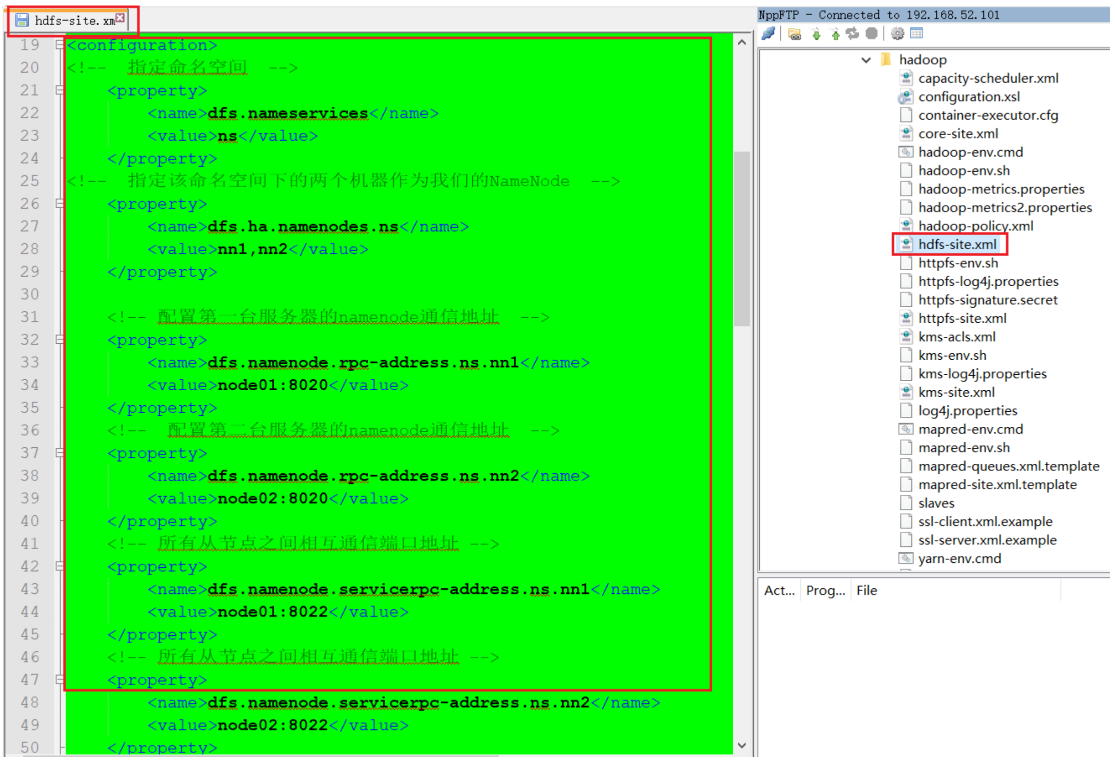
第一台机器执行以下命令

```
cd /export/servers/hadoop-2.7.5/etc/hadoop

vim hdfs-site.xml
```

```
<configuration>
<!--   指定命名空间   -->
```

```
    <property>
        <name>dfs.nameservices</name>
        <value>ns</value>
    </property>
<!--   指定该命名空间下的两个机器作为我们的 NameNode   -->
    <property>
        <name>dfs.ha.namenodes.ns</name>
        <value>nn1,nn2</value>
    </property>

    <!-- 配置第一台服务器的 namenode 通信地址   -->
    <property>
        <name>dfs.namenode.rpc-address.ns.nn1</name>
        <value>node01:8020</value>
    </property>
    <!--   配置第二台服务器的 namenode 通信地址   -->
    <property>
        <name>dfs.namenode.rpc-address.ns.nn2</name>
        <value>node02:8020</value>
    </property>
    <!-- 所有从节点之间相互通信端口地址  -->
    <property>
        <name>dfs.namenode.servicerpc-address.ns.nn1</name>
        <value>node01:8022</value>
    </property>
    <!-- 所有从节点之间相互通信端口地址  -->
    <property>
        <name>dfs.namenode.servicerpc-address.ns.nn2</name>
        <value>node02:8022</value>
    </property>

    <!-- 第一台服务器 namenode 的 web 访问地址   -->
    <property>
        <name>dfs.namenode.http-address.ns.nn1</name>
        <value>node01:50070</value>
    </property>
    <!-- 第二台服务器 namenode 的 web 访问地址   -->
    <property>
```

```
        <name>dfs.namenode.http-address.ns.nn2</name>
        <value>node02:50070</value>
    </property>

    <!-- journalNode 的访问地址，注意这个地址一定要配置 -->
    <property>
        <name>dfs.namenode.shared.edits.dir</name>

    <value>qjournal://node01:8485;node02:8485;node03:8485/ns1</value>
    </property>
    <!--   指定故障自动恢复使用的哪个 java 类 -->
    <property>
        <name>dfs.client.failover.proxy.provider.ns</name>

    <value>org.apache.hadoop.hdfs.server.namenode.ha.ConfiguredFailoverP
roxyProvider</value>
    </property>

    <!-- 故障转移使用的哪种通信机制 -->
    <property>
        <name>dfs.ha.fencing.methods</name>
        <value>sshfence</value>
    </property>

    <!-- 指定通信使用的公钥   -->
    <property>
        <name>dfs.ha.fencing.ssh.private-key-files</name>
        <value>/root/.ssh/id_rsa</value>
    </property>
    <!-- journalNode 数据存放地址   -->
    <property>
        <name>dfs.journalnode.edits.dir</name>
        <value>/export/servers/hadoop-2.7.5/data/dfs/jn</value>
    </property>
    <!-- 启用自动故障恢复功能 -->
    <property>
        <name>dfs.ha.automatic-failover.enabled</name>
        <value>true</value>
    </property>
    <!-- namenode 产生的文件存放路径  -->
    <property>
        <name>dfs.namenode.name.dir</name>
        <value>file:///export/servers/hadoop-2.7.5/data/dfs/nn/name</value>
    </property>
    <!-- edits 产生的文件存放路径  -->
    <property>
        <name>dfs.namenode.edits.dir</name>
        <value>file:///export/servers/hadoop-2.7.5/data/dfs/nn/edits</value>
    </property>
    <!-- dataNode 文件存放路径  -->
    <property>
        <name>dfs.datanode.data.dir</name>
        <value>file:///export/servers/hadoop-2.7.5/data/dfs/dn</value>
    </property>
    <!--  关闭 hdfs 的文件权限  -->
    <property>
        <name>dfs.permissions</name>
        <value>false</value>
    </property>
    <!--  指定 block 文件块的大小  -->
    <property>
        <name>dfs.blocksize</name>
        <value>134217728</value>
    </property>
</configuration>
```



```
<configuration>
<!--   指定命名空间   -->
    <property>
        <name>dfs.nameservices</name>
        <value>ns</value>
    </property>
<!--   指定该命名空间下的两个机器作为我们的NameNode   -->
    <property>
        <name>dfs.ha.namenodes.ns</name>
        <value>nn1,nn2</value>
    </property>

    <!-- 配置第一台服务器的namenode通信地址   -->
    <property>
        <name>dfs.namenode.rpc-address.ns.nn1</name>
        <value>node01:8020</value>
    </property>
    <!--   配置第二台服务器的namenode通信地址   -->
    <property>
        <name>dfs.namenode.rpc-address.ns.nn2</name>
        <value>node02:8020</value>
    </property>
    <!-- 所有从节点之间相互通信端口地址 -->
    <property>
        <name>dfs.namenode.servicerpc-address.ns.nn1</name>
        <value>node01:8022</value>
    </property>
    <!-- 所有从节点之间相互通信端口地址 -->
    <property>
        <name>dfs.namenode.servicerpc-address.ns.nn2</name>
        <value>node02:8022</value>
    </property>
```

修改 yarn-site.xml，注意 node03 与 node02 配置不同

第一台机器执行以下命令

```
cd /export/servers/hadoop-2.7.5/etc/hadoop

vim yarn-site.xml
```

```
<configuration>
<!-- Site specific YARN configuration properties -->
<!-- 是否启用日志聚合.应用程序完成后,日志汇总收集每个容器的日志,这些日志移动到文件系统,例如 HDFS. -->
<!-- 用户可以通过配置"yarn.nodemanager.remote-app-log-dir"、"yarn.nodemanager.remote-app-log-dir-suffix"来确定日志移动到的位置 -->
<!-- 用户可以通过应用程序时间服务器访问日志 -->

<!-- 启用日志聚合功能，应用程序完成后，收集各个节点的日志到一起便于查看 -->
    <property>
            <name>yarn.log-aggregation-enable</name>
            <value>true</value>
    </property>
```

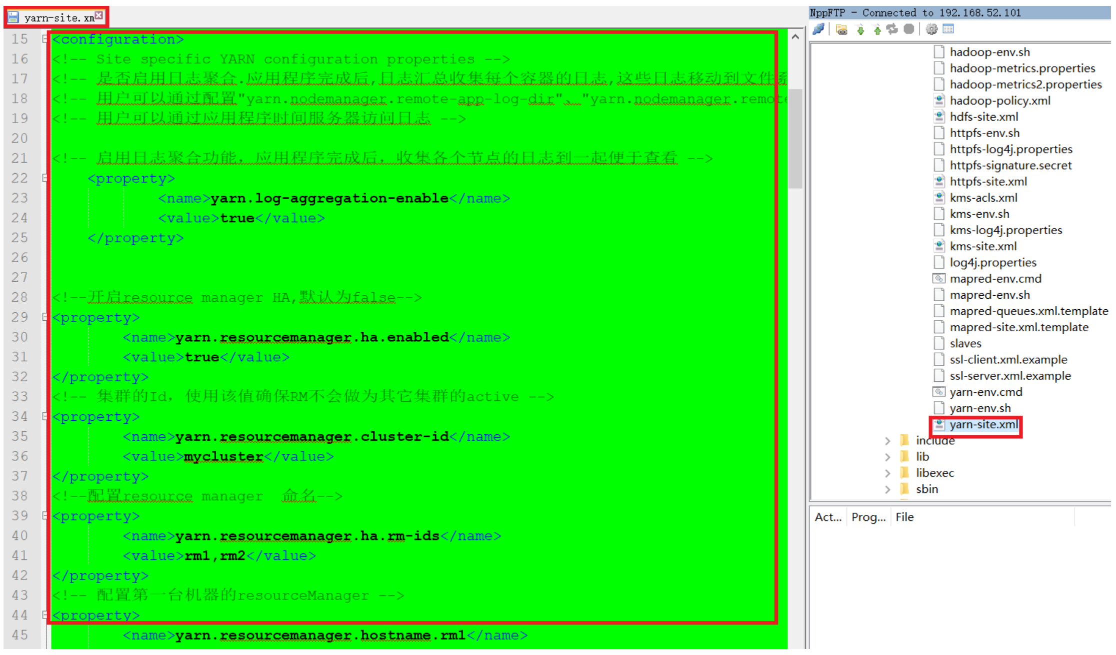
```
<!--开启 resource manager HA,默认为 false-->
<property>
        <name>yarn.resourcemanager.ha.enabled</name>
        <value>true</value>
</property>
<!-- 集群的 Id，使用该值确保 RM 不会做为其它集群的 active -->
<property>
        <name>yarn.resourcemanager.cluster-id</name>
        <value>mycluster</value>
</property>
<!--配置 resource manager  命名-->
<property>
        <name>yarn.resourcemanager.ha.rm-ids</name>
        <value>rm1,rm2</value>
</property>
<!-- 配置第一台机器的 resourceManager -->
<property>
        <name>yarn.resourcemanager.hostname.rm1</name>
        <value>node03</value>
</property>
<!-- 配置第二台机器的 resourceManager -->
<property>
        <name>yarn.resourcemanager.hostname.rm2</name>
        <value>node02</value>
</property>

<!-- 配置第一台机器的 resourceManager 通信地址  -->
<property>
        <name>yarn.resourcemanager.address.rm1</name>
        <value>node03:8032</value>
</property>
<property>
        <name>yarn.resourcemanager.scheduler.address.rm1</name>
        <value>node03:8030</value>
</property>
<property>
```

```
            <name>yarn.resourcemanager.resource-
tracker.address.rm1</name>
            <value>node03:8031</value>
</property>
<property>
            <name>yarn.resourcemanager.admin.address.rm1</name>
            <value>node03:8033</value>
</property>
<property>
            <name>yarn.resourcemanager.webapp.address.rm1</name>
            <value>node03:8088</value>
</property>

<!-- 配置第二台机器的 resourceManager 通信地址 -->
<property>
            <name>yarn.resourcemanager.address.rm2</name>
            <value>node02:8032</value>
</property>
<property>
            <name>yarn.resourcemanager.scheduler.address.rm2</name>
            <value>node02:8030</value>
</property>
<property>
            <name>yarn.resourcemanager.resource-
tracker.address.rm2</name>
            <value>node02:8031</value>
</property>
<property>
            <name>yarn.resourcemanager.admin.address.rm2</name>
            <value>node02:8033</value>
</property>
<property>
            <name>yarn.resourcemanager.webapp.address.rm2</name>
            <value>node02:8088</value>
</property>
<!--开启 resourcemanager 自动恢复功能-->
<property>
            <name>yarn.resourcemanager.recovery.enabled</name>
```

```
              <value>true</value>
</property>
<!--在 node1 上配置 rm1,在 node2 上配置 rm2,注意：一般都喜欢把配置好
的文件远程复制到其它机器上，但这个在 YARN 的另一个机器上一定要修
改，其他机器上不配置此项-->
     <property>
          <name>yarn.resourcemanager.ha.id</name>
          <value>rm1</value>
            <description>If we want to launch more than one RM in single node,
we need this configuration</description>
     </property>

          <!--用于持久存储的类。尝试开启-->
<property>
              <name>yarn.resourcemanager.store.class</name>

<value>org.apache.hadoop.yarn.server.resourcemanager.recovery.ZKRMState
Store</value>
</property>
<property>
              <name>yarn.resourcemanager.zk-address</name>
              <value>node01:2181,node02:2181,node03:2181</value>
              <description>For multiple zk services, separate them with
comma</description>
</property>
<!--开启 resourcemanager 故障自动切换，指定机器-->
<property>
              <name>yarn.resourcemanager.ha.automatic-
failover.enabled</name>
              <value>true</value>
              <description>Enable automatic failover; By default, it is enabled
only when HA is enabled.</description>
</property>
<property>
              <name>yarn.client.failover-proxy-provider</name>

<value>org.apache.hadoop.yarn.client.ConfiguredRMFailoverProxyProvider<
/value>
```

```
</property>
<!-- 允许分配给一个任务最大的 CPU 核数，默认是 8 -->
<property>
            <name>yarn.nodemanager.resource.cpu-vcores</name>
            <value>4</value>
</property>
<!-- 每个节点可用内存,单位 MB -->
<property>
            <name>yarn.nodemanager.resource.memory-mb</name>
            <value>512</value>
</property>
<!-- 单个任务可申请最少内存，默认 1024MB -->
<property>
            <name>yarn.scheduler.minimum-allocation-mb</name>
            <value>512</value>
</property>
<!-- 单个任务可申请最大内存，默认 8192MB -->
<property>
            <name>yarn.scheduler.maximum-allocation-mb</name>
            <value>512</value>
</property>
<!--多长时间聚合删除一次日志 此处-->
<property>
            <name>yarn.log-aggregation.retain-seconds</name>
            <value>2592000</value><!--30 day-->
</property>
<!--时间在几秒钟内保留用户日志。只适用于如果日志聚合是禁用的-->
<property>
            <name>yarn.nodemanager.log.retain-seconds</name>
            <value>604800</value><!--7 day-->
</property>
<!--指定文件压缩类型用于压缩汇总日志-->
<property>
            <name>yarn.nodemanager.log-aggregation.compression-
type</name>
            <value>gz</value>
</property>
<!-- nodemanager 本地文件存储目录-->
```

```
<property>
		<name>yarn.nodemanager.local-dirs</name>
		<value>/export/servers/hadoop-2.7.5/yarn/local</value>
</property>
<!-- resourceManager  保存最大的任务完成个数  -->
<property>
		<name>yarn.resourcemanager.max-completed-applications</name>
		<value>1000</value>
</property>
<!-- 逗号隔开的服务列表，列表名称应该只包含 a-zA-Z0-9_,不能以数字开始-->
<property>
		<name>yarn.nodemanager.aux-services</name>
		<value>mapreduce_shuffle</value>
</property>

<!--rm 失联后重新链接的时间-->
<property>
		<name>yarn.resourcemanager.connect.retry-interval.ms</name>
		<value>2000</value>
</property>
</configuration>
```



```
<configuration>
<!-- Site specific YARN configuration properties -->
<!-- 是否启用日志聚合.应用程序完成后,日志汇总收集每个容器的日志,这些日志移动到文件系
<!-- 用户可以通过配置"yarn.nodemanager.remote-app-log-dir"、"yarn.nodemanager.remote
<!-- 用户可以通过应用程序时间服务器访问日志 -->

<!-- 启用日志聚合功能，应用程序完成后，收集各个节点的日志到一起便于查看 -->
	<property>
			<name>yarn.log-aggregation-enable</name>
			<value>true</value>
	</property>


<!--开启resource manager HA,默认为false-->
<property>
		<name>yarn.resourcemanager.ha.enabled</name>
		<value>true</value>
</property>
<!-- 集群的Id，使用该值确保RM不会做为其它集群的active -->
<property>
		<name>yarn.resourcemanager.cluster-id</name>
		<value>mycluster</value>
</property>
<!--配置resource manager  命名-->
<property>
		<name>yarn.resourcemanager.ha.rm-ids</name>
		<value>rm1,rm2</value>
</property>
<!-- 配置第一台机器的resourceManager -->
<property>
		<name>yarn.resourcemanager.hostname.rm1</name>
```

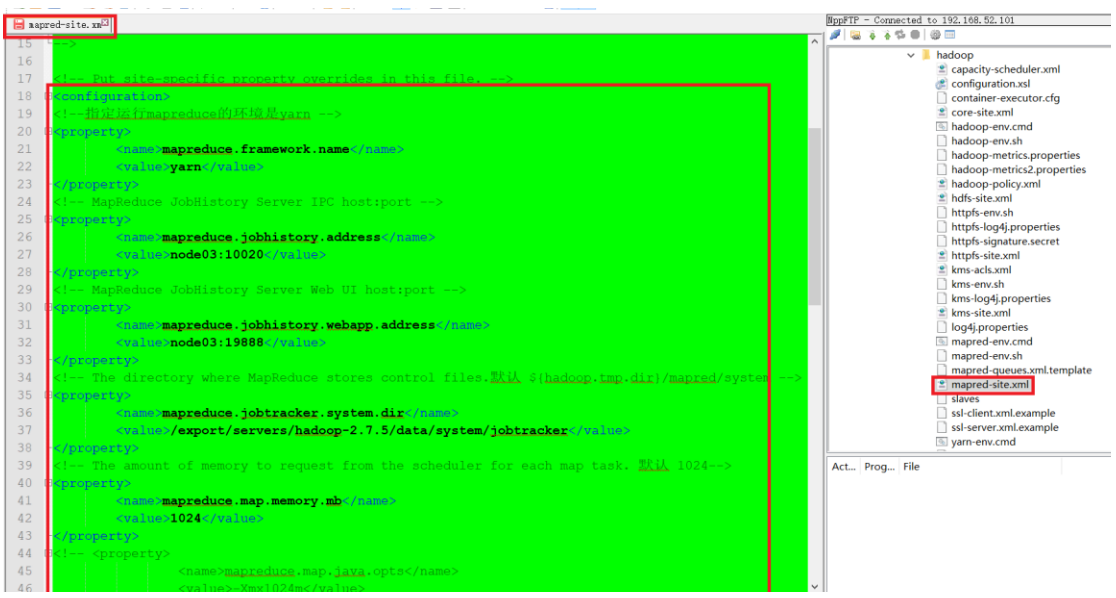
修改 mapred-site.xml

```
cd /export/servers/hadoop-2.7.5/etc/hadoop
vim mapred-site.xml
```

```
<configuration>
<!--指定运行 mapreduce 的环境是 yarn -->
<property>
		<name>mapreduce.framework.name</name>
		<value>yarn</value>
</property>
<!-- MapReduce JobHistory Server IPC host:port -->
<property>
		<name>mapreduce.jobhistory.address</name>
		<value>node03:10020</value>
</property>
<!-- MapReduce JobHistory Server Web UI host:port -->
<property>
		<name>mapreduce.jobhistory.webapp.address</name>
		<value>node03:19888</value>
</property>
<!-- The directory where MapReduce stores control files. 默认 ${hadoop.tmp.dir}/mapred/system -->
<property>
		<name>mapreduce.jobtracker.system.dir</name>
		<value>/export/servers/hadoop-2.7.5/data/system/jobtracker</value>
</property>
<!-- The amount of memory to request from the scheduler for each map task. 默认 1024-->
<property>
		<name>mapreduce.map.memory.mb</name>
		<value>1024</value>
</property>
<!-- <property>
				<name>mapreduce.map.java.opts</name>
				<value>-Xmx1024m</value>
		</property> -->
```

```
<!-- The amount of memory to request from the scheduler for each reduce task.
默认  1024-->
<property>
        <name>mapreduce.reduce.memory.mb</name>
        <value>1024</value>
</property>
<!-- <property>
                <name>mapreduce.reduce.java.opts</name>
                <value>-Xmx2048m</value>
        </property> -->
<!--  用于存储文件的缓存内存的总数量，以兆字节为单位。默认情况下，
分配给每个合并流 1MB，给个合并流应该寻求最小化。默认值 100-->
<property>
        <name>mapreduce.task.io.sort.mb</name>
        <value>100</value>
</property>

<!-- <property>
        <name>mapreduce.jobtracker.handler.count</name>
        <value>25</value>
        </property>-->
<!--  整理文件时用于合并的流的数量。这决定了打开的文件句柄的数量。
默认值 10-->
<property>
        <name>mapreduce.task.io.sort.factor</name>
        <value>10</value>
</property>
<!--  默认的并行传输量由 reduce 在 copy(shuffle)阶段。默认值 5-->
<property>
        <name>mapreduce.reduce.shuffle.parallelcopies</name>
        <value>25</value>
</property>
<property>
        <name>yarn.app.mapreduce.am.command-opts</name>
        <value>-Xmx1024m</value>
</property>
<!-- MR AppMaster 所需的内存总量。默认值 1536-->
<property>
```

```
                <name>yarn.app.mapreduce.am.resource.mb</name>
                <value>1536</value>
</property>
<!-- MapReduce 存储中间数据文件的本地目录。目录不存在则被忽略。默认值${hadoop.tmp.dir}/mapred/local-->
<property>
                <name>mapreduce.cluster.local.dir</name>
                <value>/export/servers/hadoop-2.7.5/data/system/local</value>
</property>
</configuration>
```



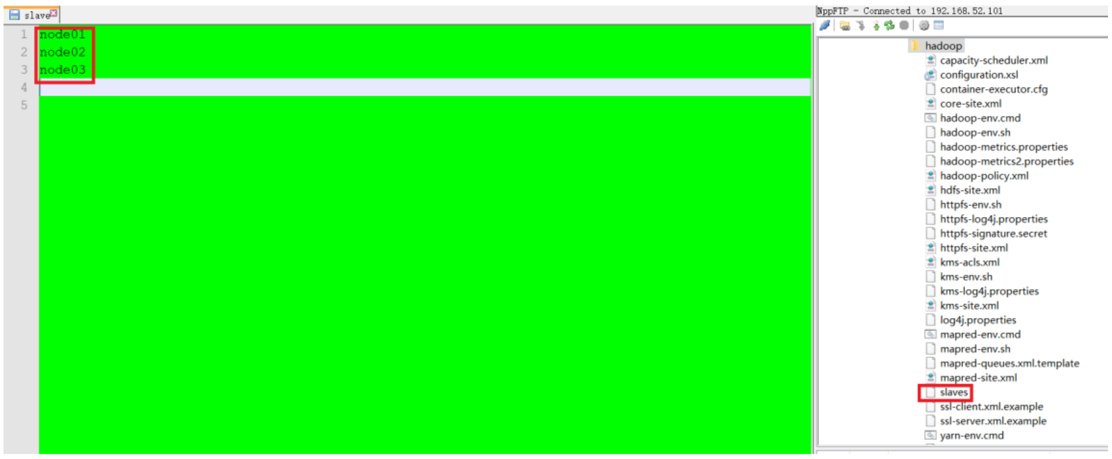
```
-->

<!-- Put site-specific property overrides in this file. -->
<configuration>
<!--指定运行mapreduce的环境是yarn -->
<property>
        <name>mapreduce.framework.name</name>
        <value>yarn</value>
</property>
<!-- MapReduce JobHistory Server IPC host:port -->
<property>
        <name>mapreduce.jobhistory.address</name>
        <value>node03:10020</value>
</property>
<!-- MapReduce JobHistory Server Web UI host:port -->
<property>
        <name>mapreduce.jobhistory.webapp.address</name>
        <value>node03:19888</value>
</property>
<!-- The directory where MapReduce stores control files.默认 ${hadoop.tmp.dir}/mapred/system -->
<property>
        <name>mapreduce.jobtracker.system.dir</name>
        <value>/export/servers/hadoop-2.7.5/data/system/jobtracker</value>
</property>
<!-- The amount of memory to request from the scheduler for each map task. 默认 1024-->
<property>
        <name>mapreduce.map.memory.mb</name>
        <value>1024</value>
</property>
<!-- <property>
                <name>mapreduce.map.java.opts</name>
                <value>-Xmx1024m</value>
```

修改 slaves

第一台机器执行以下命令

```
cd /export/servers/hadoop-2.7.5/etc/hadoop

vim slaves
```

```
node01
node02
node03
```


slaves

```
node01
node02
node03
```

NppFTP - Connected to 192.168.52.101

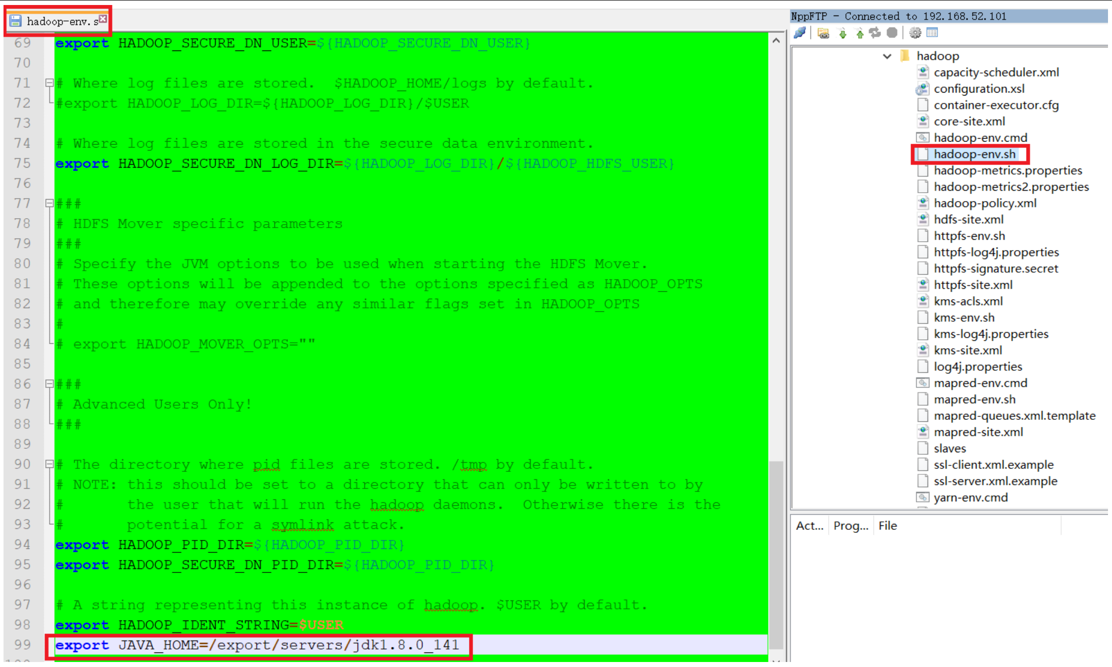
- hadoop
  - capacity-scheduler.xml
  - configuration.xsl
  - container-executor.cfg
  - core-site.xml
  - hadoop-env.cmd
  - hadoop-env.sh
  - hadoop-metrics.properties
  - hadoop-metrics2.properties
  - hadoop-policy.xml
  - hdfs-site.xml
  - httpfs-env.sh
  - httpfs-log4j.properties
  - httpfs-signature.secret
  - httpfs-site.xml
  - kms-acls.xml
  - kms-env.sh
  - kms-log4j.properties
  - kms-site.xml
  - log4j.properties
  - mapred-env.cmd
  - mapred-env.sh
  - mapred-queues.xml.template
  - mapred-site.xml
  - slaves
  - ssl-client.xml.example
  - ssl-server.xml.example
  - yarn-env.cmd

修改 hadoop-env.sh

第一台机器执行以下命令

```
cd /export/servers/hadoop-2.7.5/etc/hadoop

vim hadoop-env.sh
```

```
export JAVA_HOME=/export/servers/jdk1.8.0_141
```


hadoop-env.s

```
export HADOOP_SECURE_DN_USER=${HADOOP_SECURE_DN_USER}

# Where log files are stored.  $HADOOP_HOME/logs by default.
#export HADOOP_LOG_DIR=${HADOOP_LOG_DIR}/$USER

# Where log files are stored in the secure data environment.
export HADOOP_SECURE_DN_LOG_DIR=${HADOOP_LOG_DIR}/${HADOOP_HDFS_USER}

###
# HDFS Mover specific parameters
###
# Specify the JVM options to be used when starting the HDFS Mover.
# These options will be appended to the options specified as HADOOP_OPTS
# and therefore may override any similar flags set in HADOOP_OPTS
#
# export HADOOP_MOVER_OPTS=""

###
# Advanced Users Only!
###

# The directory where pid files are stored. /tmp by default.
# NOTE: this should be set to a directory that can only be written to by
#       the user that will run the hadoop daemons.  Otherwise there is the
#       potential for a symlink attack.
export HADOOP_PID_DIR=${HADOOP_PID_DIR}
export HADOOP_SECURE_DN_PID_DIR=${HADOOP_PID_DIR}

# A string representing this instance of hadoop. $USER by default.
export HADOOP_IDENT_STRING=$USER
export JAVA_HOME=/export/servers/jdk1.8.0_141
```

NppFTP - Connected to 192.168.52.101

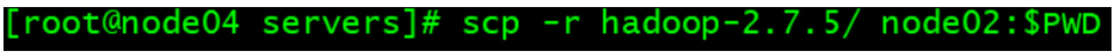
- hadoop
  - capacity-scheduler.xml
  - configuration.xsl
  - container-executor.cfg
  - core-site.xml
  - hadoop-env.cmd
  - hadoop-env.sh
  - hadoop-metrics.properties
  - hadoop-metrics2.properties
  - hadoop-policy.xml
  - hdfs-site.xml
  - httpfs-env.sh
  - httpfs-log4j.properties
  - httpfs-signature.secret
  - httpfs-site.xml
  - kms-acls.xml
  - kms-env.sh
  - kms-log4j.properties
  - kms-site.xml
  - log4j.properties
  - mapred-env.cmd
  - mapred-env.sh
  - mapred-queues.xml.template
  - mapred-site.xml
  - slaves
  - ssl-client.xml.example
  - ssl-server.xml.example
  - yarn-env.cmd

集群启动过程

将第一台机器的安装包发送到其他机器上

第一台机器执行以下命令:

```
cd /export/servers

scp -r hadoop-2.7.5/ node02:$PWD

scp -r hadoop-2.7.5/ node03:$PWD
```



```
[root@node04 servers]# scp -r hadoop-2.7.5/ node02:$PWD
```

三台机器上共同创建目录

三台机器执行以下命令

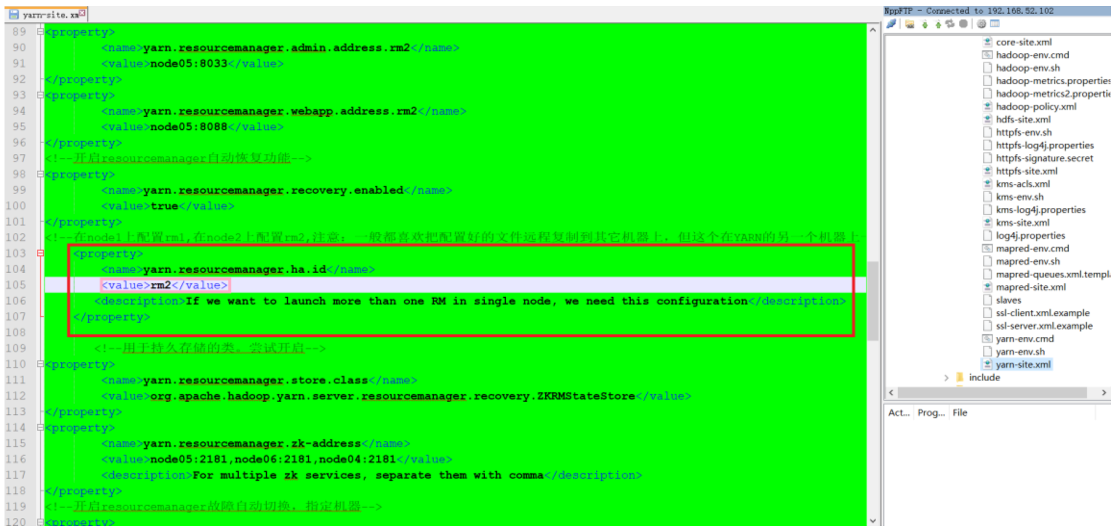
```
mkdir -p /export/servers/hadoop-2.7.5/data/dfs/nn/name
mkdir -p /export/servers/hadoop-2.7.5/data/dfs/nn/edits
mkdir -p /export/servers/hadoop-2.7.5/data/dfs/nn/name
mkdir -p /export/servers/hadoop-2.7.5/data/dfs/nn/edits
```

更改 node02 的 rm2

第二台机器执行以下命令

```
cd /export/servers/hadoop-2.7.5/etc/hadoop

vim  yarn-site.xml
```

```
<!--在 node3 上配置 rm1,在 node2 上配置 rm2,注意：一般都喜欢把配置好的文件远程复制到其它机器上，
但这个在 YARN 的另一个机器上一定要修改，其他机器上不配置此项
注意我们现在有两个 resourceManager   第三台是 rm1    第二台是 rm2
这个配置一定要记得去 node02 上面改好

-->
    <property>
        <name>yarn.resourcemanager.ha.id</name>
        <value>rm2</value>
         <description>If we want to launch more than one RM in single node, we need this configuration</description>
    </property>
```



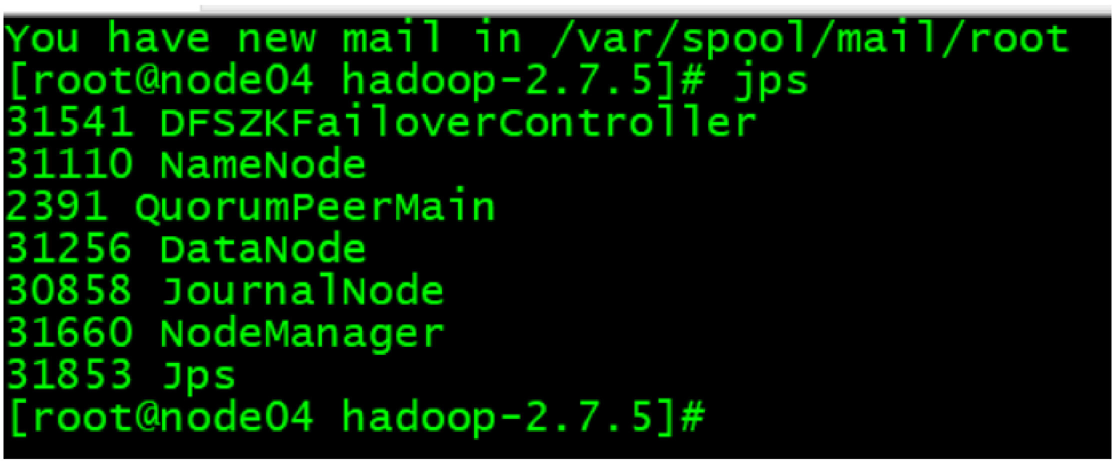
```xml
<property>
        <name>yarn.resourcemanager.admin.address.rm2</name>
        <value>node05:8033</value>
</property>
<property>
        <name>yarn.resourcemanager.webapp.address.rm2</name>
        <value>node05:8088</value>
</property>
<!--开启resourcemanager自动恢复功能-->
<property>
        <name>yarn.resourcemanager.recovery.enabled</name>
        <value>true</value>
</property>
<!--在node1上配置rm1,在node2上配置rm2,注意：一般都喜欢把配置好的文件远程复制到其它机器上，但这个在YARN的另一个机器上
    <property>
        <name>yarn.resourcemanager.ha.id</name>
        <value>rm2</value>
       <description>If we want to launch more than one RM in single node, we need this configuration</description>
    </property>

       <!--用于持久存储的类。尝试开启-->
<property>
        <name>yarn.resourcemanager.store.class</name>
        <value>org.apache.hadoop.yarn.server.resourcemanager.recovery.ZKRMStateStore</value>
</property>
<property>
        <name>yarn.resourcemanager.zk-address</name>
        <value>node05:2181,node06:2181,node04:2181</value>
        <description>For multiple zk services, separate them with comma</description>
</property>
<!--开启resourcemanager故障自动切换，指定机器-->
<property>
```

启动 HDFS 过程

node01 机器执行以下命令

```
cd  /export/servers/hadoop-2.7.5
bin/hdfs zkfc -formatZK
sbin/hadoop-daemons.sh start journalnode
bin/hdfs namenode -format
bin/hdfs namenode -initializeSharedEdits -force
sbin/start-dfs.sh
```

jps



```
You have new mail in /var/spool/mail/root
[root@node04 hadoop-2.7.5]# jps
31541 DFSZKFailoverController
31110 NameNode
2391 QuorumPeerMain
31256 DataNode
30858 JournalNode
31660 NodeManager
31853 Jps
[root@node04 hadoop-2.7.5]#
```

node02 上面执行

```
cd   /export/servers/hadoop-2.7.5

bin/hdfs namenode -bootstrapStandby

sbin/hadoop-daemon.sh start namenode
```

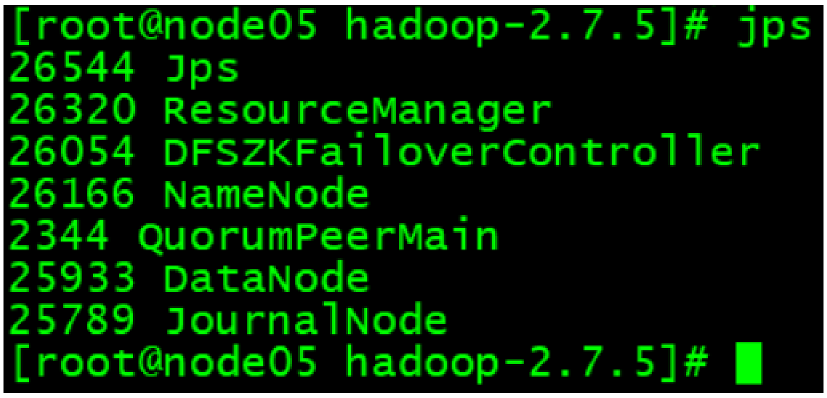
启动 yarn 过程

node03 上面执行

```
cd   /export/servers/hadoop-2.7.5

sbin/start-yarn.sh
```

node02 上执行

```
cd   /export/servers/hadoop-2.7.5

sbin/start-yarn.sh
```

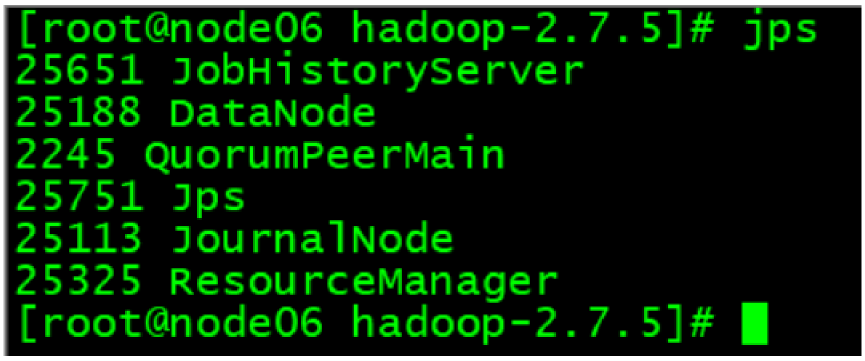


```
[root@node05 hadoop-2.7.5]# jps
26544 Jps
26320 ResourceManager
26054 DFSZKFailoverController
26166 NameNode
2344 QuorumPeerMain
25933 DataNode
25789 JournalNode
[root@node05 hadoop-2.7.5]#
```

查看 resourceManager 状态

node03 上面执行

```
cd   /export/servers/hadoop-2.7.5
bin/yarn rmadmin -getServiceState rm1
```



```
[root@node06 hadoop-2.7.5]# jps
25651 JobHistoryServer
25188 DataNode
2245 QuorumPeerMain
25751 Jps
25113 JournalNode
25325 ResourceManager
[root@node06 hadoop-2.7.5]# 
```

node02 上面执行

```
cd   /export/servers/hadoop-2.7.5
bin/yarn rmadmin -getServiceState rm2
```

node03 启动 jobHistory

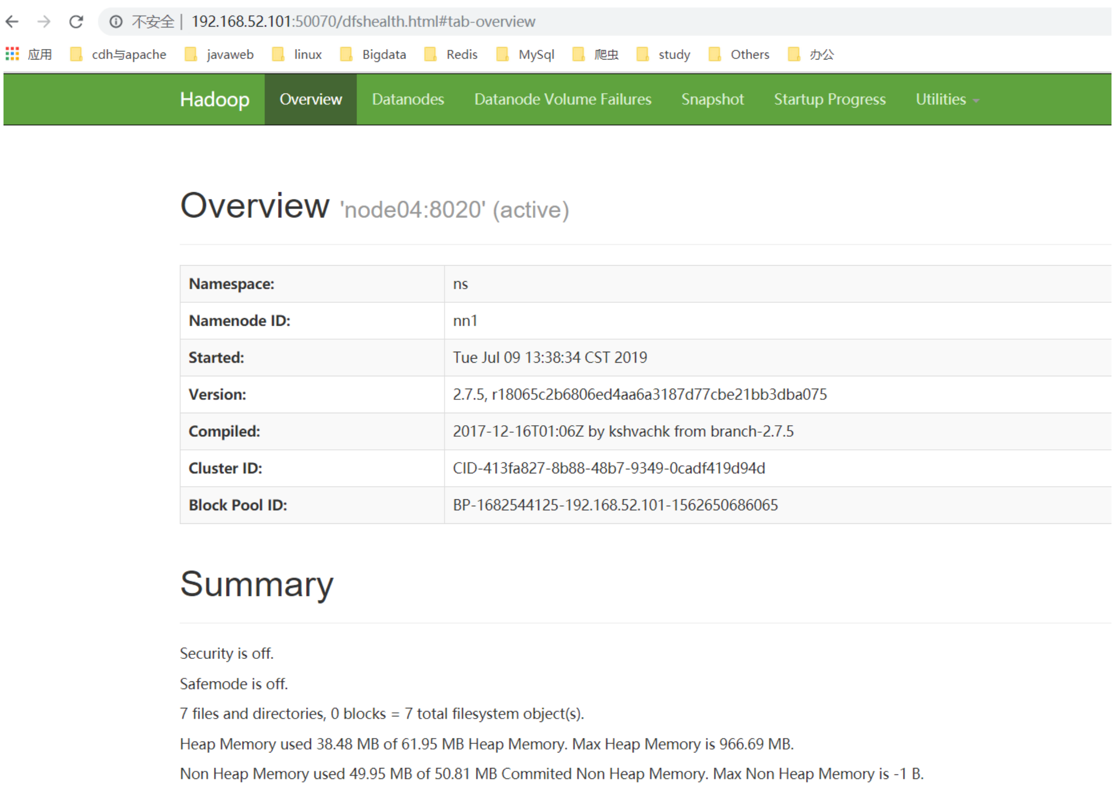
node03 机器执行以下命令启动 jobHistory

```
cd /export/servers/hadoop-2.7.5
sbin/mr-jobhistory-daemon.sh start historyserver
```

hdfs 状态查看

node01 机器查看 hdfs 状态

192.168.52.100:50070/dfshealth.html#tab-overview



不安全 | 192.168.52.101:50070/dfshealth.html#tab-overview
应用 cdh与apache javaweb linux Bigdata Redis MySql 爬虫 study Others 办公
Hadoop Overview Datanodes Datanode Volume Failures Snapshot Startup Progress Utilities

Overview 'node04:8020' (active)

Namespace:	ns
Namenode ID:	nn1
Started:	Tue Jul 09 13:38:34 CST 2019
Version:	2.7.5, r18065c2b6806ed4aa6a3187d77cbe21bb3dba075
Compiled:	2017-12-16T01:06Z by kshvachk from branch-2.7.5
Cluster ID:	CID-413fa827-8b88-48b7-9349-0cadf419d94d
Block Pool ID:	BP-1682544125-192.168.52.101-1562650686065

Summary

Security is off.

Safemode is off.

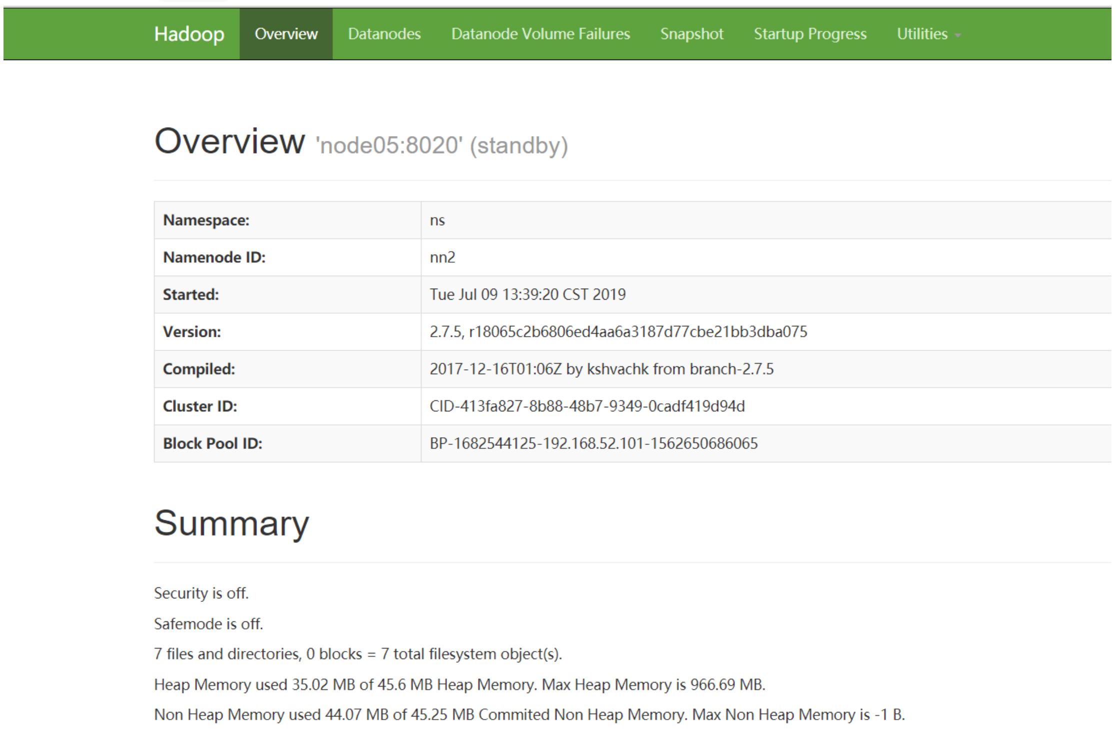
7 files and directories, 0 blocks = 7 total filesystem object(s).

Heap Memory used 38.48 MB of 61.95 MB Heap Memory. Max Heap Memory is 966.69 MB.

Non Heap Memory used 49.95 MB of 50.81 MB Commited Non Heap Memory. Max Non Heap Memory is -1 B.

node02 机器查看 hdfs 状态

192.168.52.110:50070/dfshealth.html#tab-overview



Hadoop Overview Datanodes Datanode Volume Failures Snapshot Startup Progress Utilities

Overview 'node05:8020' (standby)

Namespace:	ns
Namenode ID:	nn2
Started:	Tue Jul 09 13:39:20 CST 2019
Version:	2.7.5, r18065c2b6806ed4aa6a3187d77cbe21bb3dba075
Compiled:	2017-12-16T01:06Z by kshvachk from branch-2.7.5
Cluster ID:	CID-413fa827-8b88-48b7-9349-0cadf419d94d
Block Pool ID:	BP-1682544125-192.168.52.101-1562650686065

Summary

Security is off.

Safemode is off.

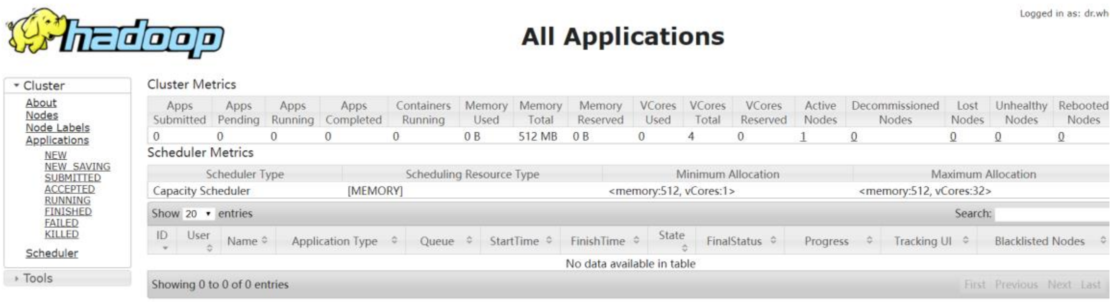
7 files and directories, 0 blocks = 7 total filesystem object(s).

Heap Memory used 35.02 MB of 45.6 MB Heap Memory. Max Heap Memory is 966.69 MB.

Non Heap Memory used 44.07 MB of 45.25 MB Commited Non Heap Memory. Max Non Heap Memory is -1 B.

yarn 集群访问查看

192.168.52.120:8088/cluster



Logged in as: dr.wh

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	512 MB	0 B	0	4	0	1	0	0	0	0

Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum Allocation	Maximum Allocation
Capacity Scheduler	[MEMORY]	<memory:512, vCores:1>	<memory:512, vCores:32>

Show 20 entries Search:

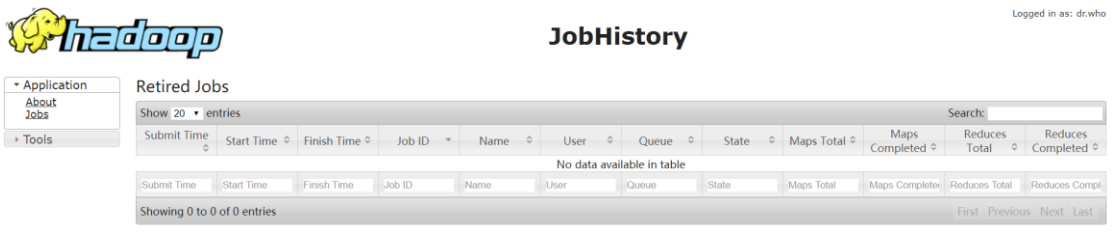
ID	User	Name	Application Type	Queue	StartTime	FinishTime	State	FinalStatus	Progress	Tracking UI	Blacklisted Nodes
No data available in table											

Showing 0 to 0 of 0 entries First Previous Next Last

历史任务浏览界面

页面访问：

192.168.52.120:19888/jobhistory



Logged in as: dr.who

JobHistory

Application
- About
- Jobs

Tools

Retired Jobs

Show 20 entries Search:

Submit Time	Start Time	Finish Time	Job ID	Name	User	Queue	State	Maps Total	Maps Completed	Reduces Total	Reduces Completed
No data available in table											
Submit Time	Start Time	Finish Time	Job ID	Name	User	Queue	State	Maps Total	Maps Complete	Reduces Total	Reduces Compl

Showing 0 to 0 of 0 entries First Previous Next Last