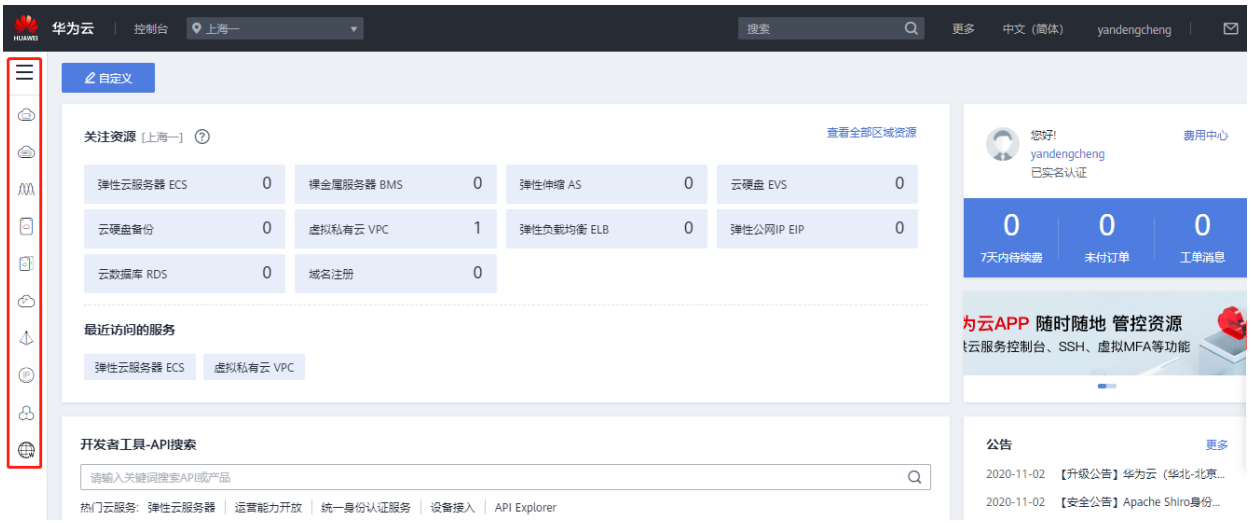
1. 购买华为云ECS

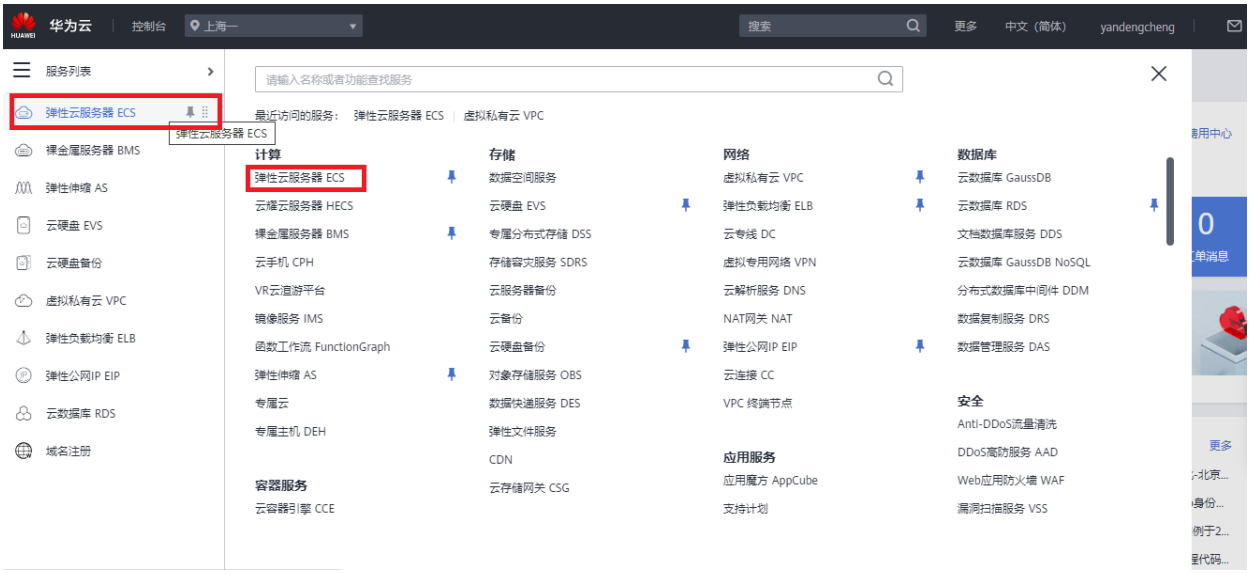
打开华为云主页（https://www.huaweicloud.com），登录华为云账号。

登录后点击**控制台**，进入华为云控制台。

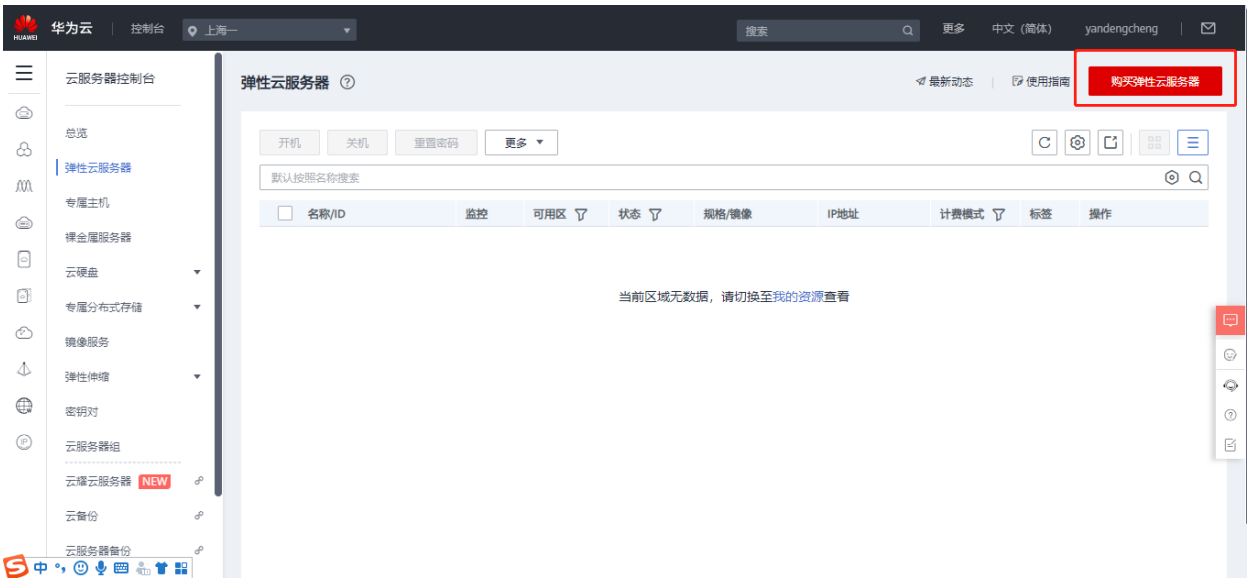
在控制台页面，华为云相关云服务菜单在左侧导航栏中。



选择**弹性云服务器ECS**。



选择**购买弹性云服务器**。

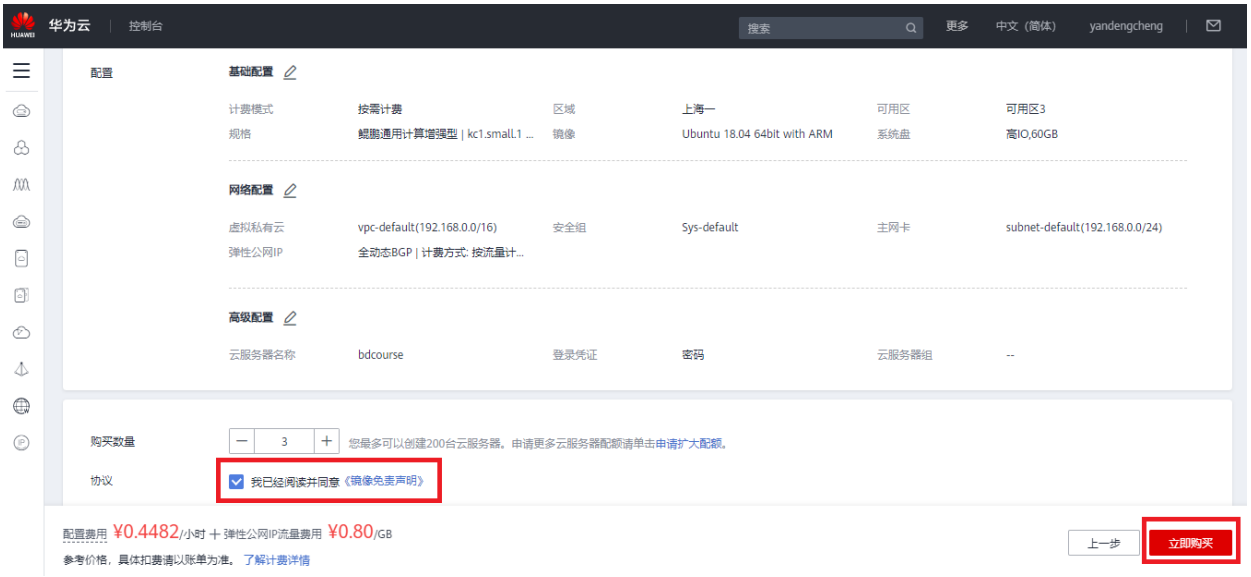


进行基础配置，**严格按照下图规格**选择ECS虚拟机 计费模式 、 CPU架构 、 镜像 和 系统盘 。

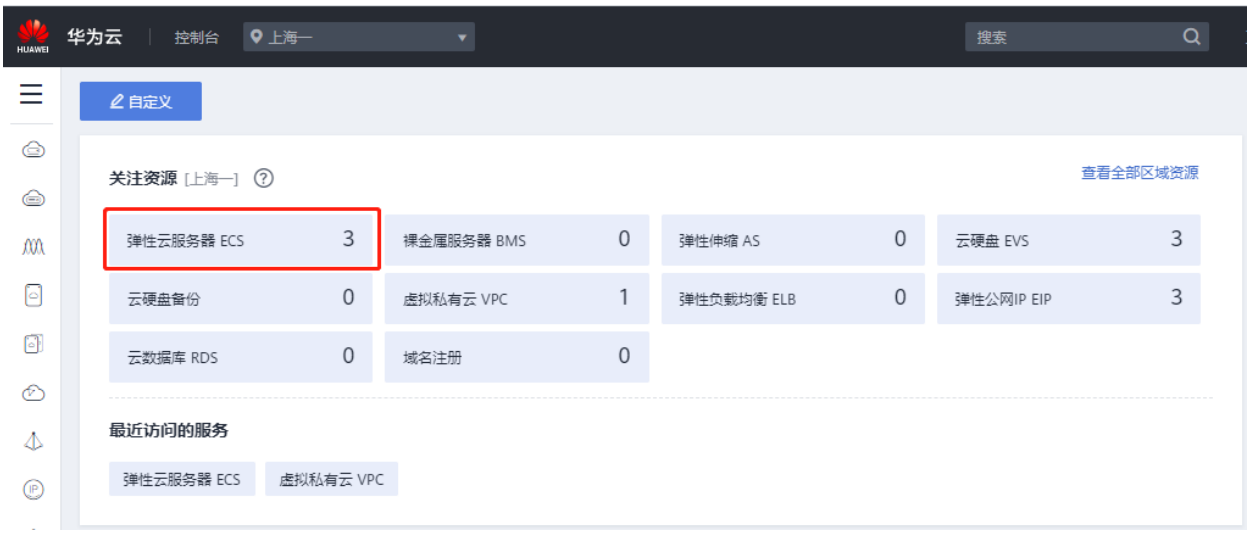
进行网络配置，选择ECS虚拟机的`网络`、`安全组`和`IP`。

进行高级配置，配置ECS虚拟机的`账号`和`备份`（云备份`暂不购买`）等。

最后核对配置信息，并确认购买。



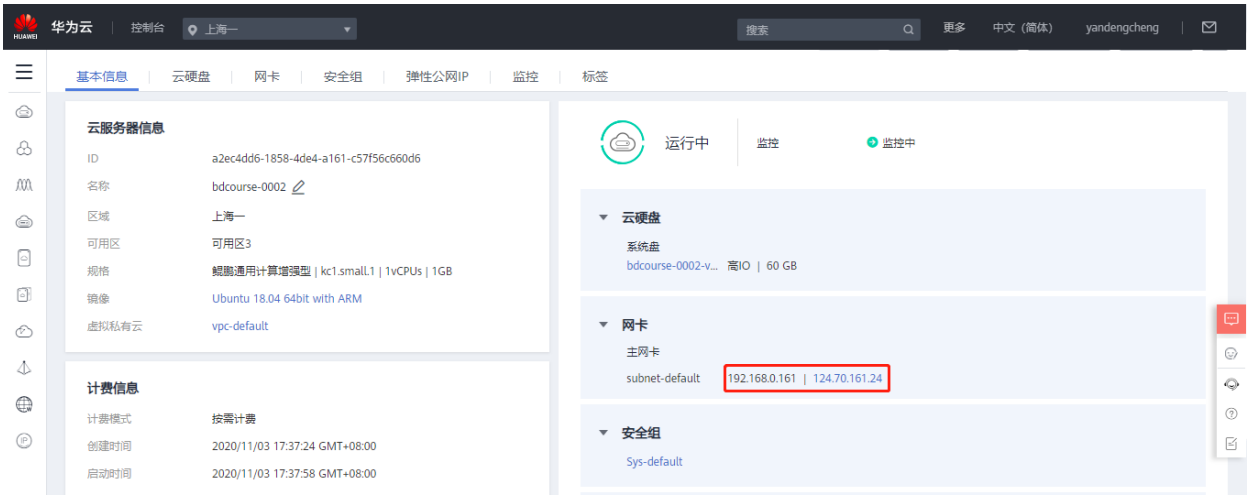
在**控制台**中查看已购买的ECS虚拟机。



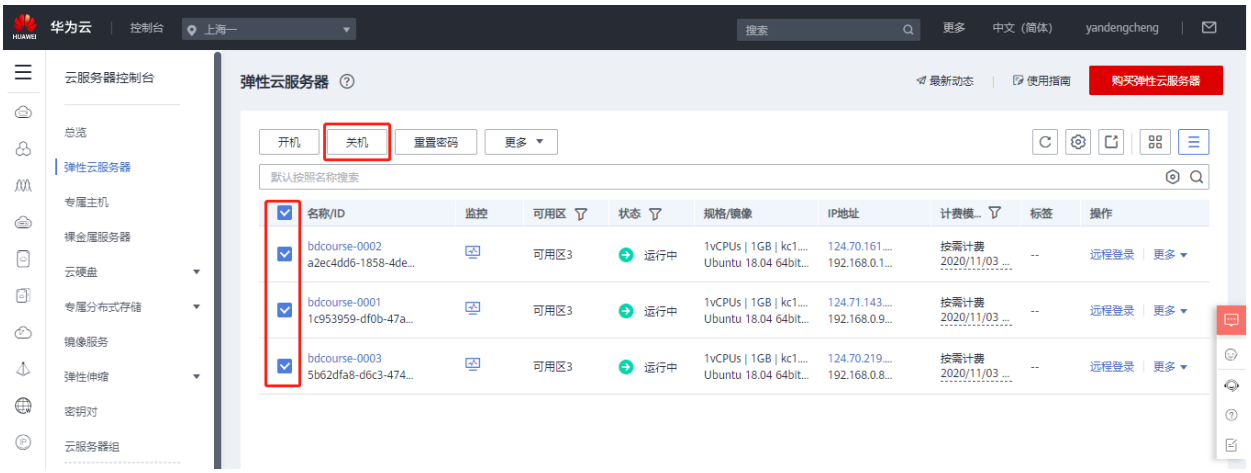
在虚拟主机列表中点击虚拟主机名称，进入虚拟主机详情页面，分别记下各台虚拟主机的**公网IP**和**内网IP**（192开头的IP）。**后面各台主机之间使用内网IP进行通信，内网流量免费。** 例如刚申请的三台ECS虚拟机的IP分别是:

- bdcourse-0001: 192.168.0.95 | 124.71.143.16

- bdcourse-0002: 192.168.0.161 | 124.70.161.24
- bdcourse-0003: 192.168.0.8 | 124.70.219.238



注意：当不使用的时候，请将虚拟机关机，以节省费用。



2. Hadoop分布式系统安装配置

2.0 基本信息

注意：

- `\$`符号开头的是shell命令，执行的时候不要复制前面的`\$`符号；
- `#`号开头的是注释内容，用于简单解释命令用途等；
- 不带`或`#`号开头的内容一般是需要编辑的文本内容，使用vi进行编辑，具体使用方法参考鸟哥的Linux私房菜-vim编辑器

序号	虚拟机	内网IP	外网IP
1	bdcourse-0001	192.168.0.95	124.71.143.16

序号	虚拟机	内网IP	外网IP
2	bdcourse-0002	192.168.0.161	124.70.161.24
3	bdcourse-0003	192.168.0.8	124.70.219.238

对上表中的**三台虚拟机分别**执行下面的命令，修改hosts信息

```
$ vi /etc/hosts

# 添加如下内容
192.168.0.95   bdcourse-0001
192.168.0.161  bdcourse-0002
192.168.0.8    bdcourse-0003

# 注意，需要删除该文件中下面的这些行
127.0.1.1       bdcourse-0001   bdcourse-0001
127.0.1.1       bdcourse-0002   bdcourse-0002
127.0.1.1       bdcourse-0003   bdcourse-0003
```

禁止系统启动时在/etc/hosts添加127.0.1.1 localhost localhost

```
# 修改manage_etc_hosts: false，然后重启虚拟机
$ vi /etc/cloud/cloud.cfg
```

2.1 配置master节点免密登录slave节点

- 将`bdcourse-0001`选定为master节点，master节点主要安装HDFS的NamaNode守护进程和YARN的ResourceManager守护进程等；
- 将`bdcourse-0001`、`bdcourse-0002`、`bdcourse-0003`选定为slave节点（注意`bdcourse-0001`同时作为master和slave节点），slave节点主要安装HDFS的DataNode守护进程和YARN的NodeManager守护进程。

在master节点（`bdcourse-0001`节点）上执行如下命令，实现master节点免密登录其他slave节点，因为集群启动命令在master节点上执行，由master节点远程到slave主机上去启动相应的守护进程（如DataNode和NodeManager守护进程）。

```
# 生成公私钥对，选择默认选项
$ ssh-keygen -t rsa -P ""

# 免密登录slave节点
$ ssh-copy-id -i ~/.ssh/id_rsa.pub root@bdcourse-0001
$ ssh-copy-id -i ~/.ssh/id_rsa.pub root@bdcourse-0002
$ ssh-copy-id -i ~/.ssh/id_rsa.pub root@bdcourse-0003

# 尝试从bdcourse-0001登录bdcourse-0001，bdcourse-0002，bdcourse-0003
# 如果不需要密码能正常登录，则配置成功
$ ssh root@bdcourse-0001
$ exit
$ ssh root@bdcourse-0002
$ exit
$ ssh root@bdcourse-0003
$ exit
```

2.2 安装JDK

安装主机：

- bdcourse-0001
- bdcourse-0002
- bdcourse-0003

```
$ apt-get install openjdk-8-jdk

# 配置JAVA_HOME环境变量，很多java程序需要使用该环境变量
# 首先查看java安装位置
# 下面命令输出/usr/lib/jvm/java-8-openjdk-arm64/jre/bin/java
# 后面使用/usr/lib/jvm/java-8-openjdk-arm64作为JAVA_HOME环境变量
$ update-alternatives --config java

$ vi /etc/profile
# 在文件结尾添加如下内容
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-arm64
export JRE_HOME=$JAVA_HOME/jre
export CLASSPATH=$JAVA_HOME/lib:$JRE_HOME/lib:$CLASSPATH
export PATH=$JAVA_HOME/bin:$JRE_HOME/bin:$PATH

# 使环境变量生效
$ source /etc/profile
```

2.3 Hadoop安装环境准备

安装主机：

- bdcourse-0001
- bdcourse-0002
- bdcourse-0003

创建相关文件夹

```
# -p 选项表示如果父目录不存在则创建之
$ mkdir -p /data/hadoop/tmp
```

更新环境变量

```
$ vi /etc/profile

# 更新如下环境变量
export HADOOP_HOME=/opt/hadoop
# 设置Hadoop动态库查找路径
# 否则spark运行报Unable to load native-hadoop library for your platform
export LD_LIBRARY_PATH=$HADOOP_HOME/lib/native/
export PATH=$HADOOP_HOME/bin:$HADOOP_HOME/sbin:$JAVA_HOME/bin:$JRE_HOME/bin:$PATH

$ source /etc/profile
```

2.4 安装Hadoop

可以先在bdcourse-0001上安装、配置Hadoop，然后将安装文件复制到各slave节点上，保证所有节点的配置文件一致。

`2.4` 节和 `2.5` 节中的命令仅在 `bdcourse-0001` 上执行， `2.6` 节再将 `bdcourse-0001` 上安装配置好的hadoop安装包复制到其他节点上。

```
# 从USTC源下载Hadoop 2.7
$ wget http://mirrors.ustc.edu.cn/apache/hadoop/common/hadoop-2.7.7/hadoop-2.7.7.tar.gz
# 解压
$ tar -xzvf hadoop-2.7.7.tar.gz
# 移动到安装目录，软件包的安装可以都放在/opt目录下
$ mv hadoop-2.7.7 /opt/
# 建立软链接，便于版本更新
$ ln -s /opt/hadoop-2.7.7 /opt/hadoop
```

2.5 配置Hadoop

`hadoop-env.sh` 配置

`hadoop-env.sh` 中配置Hadoop启动时加载的环境变量。

```
vi ${HADOOP_HOME}/etc/hadoop/hadoop-env.sh
# export JAVA_HOME=${JAVA_HOME}换成下面
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-arm64
```

Hadoop deamon配置

Hadoop的Java配置包括两部分，一部分是只读默认配置（在相应jar文件中，如 `${HADOOP_HOME}/share/hadoop/common/hadoop-common-2.7.7.jar` 解压之后包含 `core-default.xml` ，在 `${HADOOP_HOME}/share/doc/hadoop/hadoop-project-dist/hadoop-common/core-default.xml` 有示例模板）：

- core-default.xml
- hdfs-default.xml
- yarn-default.xml
- mapred-default.xml

另一部分是特定站点配置：

- etc/hadoop/core-site.xml
- etc/hadoop/hdfs-site.xml
- etc/hadoop/yarn-site.xml
- etc/hadoop/mapred-site.xml

core-site.xml配置

```
$ vi ${HADOOP_HOME}/etc/hadoop/core-site.xml

# 删除原有的<configuration></configuration>及其之间的内容，添加如下内容
<configuration>
    <property>
        <name>fs.defaultFS</name>
        <value>hdfs://bdcourse-0001:9000</value>
    </property>

    <property>
        <name>hadoop.tmp.dir</name>
        <value>/data/hadoop/tmp</value>
    </property>
</configuration>
```

hdfs-site.xml配置

```
$ vi ${HADOOP_HOME}/etc/hadoop/hdfs-site.xml

# 删除原有的<configuration></configuration>及其之间的内容，添加如下内容
<configuration>
    <property>
        <name>dfs.namenode.http-address</name>
        <value>bdcourse-0001:50070</value>
    </property>

    <property>
        <name>dfs.namenode.https-address</name>
        <value>bdcourse-0001:50470</value>
    </property>

    <property>
        <name>dfs.namenode.secondary.http-address</name>
        <value>bdcourse-0002:50090</value>
    </property>

    <property>
        <name>dfs.namenode.secondary.https-address</name>
        <value>bdcourse-0002:50091</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>

    <property>
        <name>dfs.datanode.data.dir.perm</name>
        <value>700</value>
    </property>

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

    <property>
        <name>dfs.blocksize</name>
        <value>32m</value>
    </property>
</configuration>
```

yarn-site.xml配置

```
$ vi ${HADOOP_HOME}/etc/hadoop/yarn-site.xml

# 删除原有的<configuration></configuration>及其之间的内容，添加如下内容
<configuration>
    <property>
        <name>yarn.resourcemanager.hostname</name>
        <value>bdcourse-0001</value>
    </property>

    <property>
        <name>yarn.resourcemanager.address</name>
        <value>${yarn.resourcemanager.hostname}:8032</value>
    </property>

    <property>
        <name>yarn.resourcemanager.webapp.address</name>
        <value>${yarn.resourcemanager.hostname}:8088</value>
    </property>

    <property>
        <name>yarn.resourcemanager.webapp.https.address</name>
        <value>${yarn.resourcemanager.hostname}:8090</value>
    </property>

    <property>
        <name>yarn.resourcemanager.admin.address</name>
        <value>${yarn.resourcemanager.hostname}:8033</value>
    </property>

    <property>
        <name>yarn.scheduler.minimum-allocation-mb</name>
        <value>1024</value>
    </property>

    <property>
        <name>yarn.scheduler.maximum-allocation-mb</name>
        <value>8192</value>
    </property>

    <property>
        <name>yarn.scheduler.minimum-allocation-vcores</name>
        <value>1</value>
    </property>

    <property>
        <name>yarn.scheduler.maximum-allocation-vcores</name>
        <value>8</value>
    </property>

    <property>
        <name>yarn.nodemanager.resource.memory-mb</name>
        <value>16384</value>
    </property>

    <property>
        <name>yarn.nodemanager.resource.cpu-vcores</name>
        <value>8</value>
    </property>

    <property>
        <name>yarn.nodemanager.aux-services</name>
        <value>mapreduce_shuffle</value>
    </property>

    <property>
        <name>yarn.nodemanager.pmem-check-enabled</name>
        <value>false</value>
    </property>

    <property>
        <name>yarn.nodemanager.vmem-check-enabled</name>
        <value>false</value>
    </property>
</configuration>
```

mapred-site.xml配置

```
$ vi ${HADOOP_HOME}/etc/hadoop/mapred-site.xml

<configuration>
    <property>
        <name>mapreduce.framework.name</name>
        <value>yarn</value>
    </property>
    <property>
        <name>mapreduce.jobhistory.address</name>
        <value>bdcourse-0001:10020</value>
    </property>
    <property>
        <name>mapreduce.jobhistory.webapp.address</name>
        <value>bdcourse-0001:19888</value>
    </property>
</configuration>
```

slaves

```
$ vi ${HADOOP_HOME}/etc/hadoop/slaves

# 删除原有的localhost，添加如下内容
bdcourse-0001
bdcourse-0002
bdcourse-0003
```

2.6 复制master节点配置好的Hadoop安装包到slave节点

仅在`bdcourse-0001`上执行

```
$ scp -r /opt/hadoop-2.7.7 root@bdcourse-0002:/opt/hadoop-2.7.7
$ ssh root@bdcourse-0002
$ ln -s /opt/hadoop-2.7.7 /opt/hadoop
$ exit

$ scp -r /opt/hadoop-2.7.7 root@bdcourse-0003:/opt/hadoop-2.7.7
$ ssh root@bdcourse-0003
$ ln -s /opt/hadoop-2.7.7 /opt/hadoop
$ exit
```

2.7 启动Hadoop集群

2.7.1 初始化NameNode并手动启动Hadoop集群

初始化NameNode

仅在`bdcourse-0001`上执行一次。

```
$ hdfs namenode -format
```

启动HDFS和YARN服务

仅在`bdcourse-0001`上执行

```
# 启动HDFS
$ ${HADOOP_HOME}/sbin/start-dfs.sh
# 查看HDFS集群状态
$ hdfs dfsadmin -report

# 启动YARN
$ ${HADOOP_HOME}/sbin/start-yarn.sh
# 查看YARN集群状态
$ yarn node -list -all

# 启动MapReduce
$ ${HADOOP_HOME}/sbin/mr-jobhistory-daemon.sh start historyserver

# 查看启动的进程
$ jps
```

2.7.2 配置systemd services启动Hadoop（可选）

Linux系统启动的时候会启动一批守护进程（服务），早期的Linux版本一般采用`init`进程来启动服务，所有服务都是`init`进程的子进程。但是`init`只能串行启动，速度慢。现在一般采用Systemd，它是一组命令，对系统所有资源进行管理，系统服务（Service）只是其中一种。

分别在对应主机安装对应服务：

- hdfs-nn
- hdfs-dn
- yarn-rm
- yarn-nm
- mr-hs

所有services放在`/opt/services`目录下，复制到Hadoop集群各台主机上。

```
$ mkdir -p /opt/services
```

因为systemd启动时不包含`/etc/profile`配置的环境变量，所以为每个服务配置一个启动脚本，以`.run.sh`结尾，放置于`/opt/services`目录下。

hdfs-nn

仅需要在master节点启动

```
$ vi /opt/services/hdfs-nn.run.sh

#!/bin/bash

source /etc/profile
```

```
start() {
  /opt/hadoop/sbin/hadoop-daemon.sh start namenode
}

stop() {
  /opt/hadoop/sbin/hadoop-daemon.sh stop namenode
}

case "$1" in
  start)
    start
    ;;
  stop)
    stop
    ;;
  *)
esac

$ vi /opt/services/hdfs-nn.service

[Unit]
Description=HDFS namenode service
After=network.target

[Service]
User=root
Group=root
Type=forking
WorkingDirectory=/opt/hadoop
ExecStart=/opt/services/hdfs-nn.run.sh start
ExecStop=/opt/services/hdfs-nn.run.sh stop
Restart=always
RestartSec=10

[Install]
WantedBy=multi-user.target

# 修改文件属性，使其可执行
$ chmod u+x /opt/services/hdfs-nn.run.sh
# 建立软连接，Systemd 默认从目录/etc/systemd/system/读取配置文件
$ ln -s /opt/services/hdfs-nn.service /etc/systemd/system/hdfs-nn.service
# 重新加载服务的配置文件
$ systemctl daemon-reload
# 设置开机启动（通过在/etc/systemd/system/multi-user.target.wants/创建软连接实现）
$ systemctl enable hdfs-nn
# 启动服务
$ systemctl start hdfs-nn
# 查看服务状态
$ systemctl status hdfs-nn
# 停止服务
$ systemctl stop hdfs-nn
```

hdfs-dn

需要在所有slave节点启动

```
$ vi /opt/services/hdfs-dn.run.sh

#!/bin/bash

source /etc/profile

start() {
  /opt/hadoop/sbin/hadoop-daemon.sh start datanode
}

stop() {
  /opt/hadoop/sbin/hadoop-daemon.sh stop datanode
}

case "$1" in
  start)
    start
    ;;
  stop)
    stop
    ;;
  *)
esac

$ vi /opt/services/hdfs-dn.service

[Unit]
Description=HDFS datanode service
After=network.target

[Service]
User=root
Group=root
Type=forking
WorkingDirectory=/opt/hadoop
ExecStart=/opt/services/hdfs-dn.run.sh start
ExecStop=/opt/services/hdfs-dn.run.sh stop
Restart=always
RestartSec=10

[Install]
WantedBy=multi-user.target

# 修改文件属性，使其可执行
$ chmod u+x /opt/services/hdfs-dn.run.sh
# 建立软连接，Systemd 默认从目录/etc/systemd/system/读取配置文件
$ ln -s /opt/services/hdfs-dn.service /etc/systemd/system/hdfs-dn.service
# 重新加载服务的配置文件
$ systemctl daemon-reload
# 设置开机启动（通过在/etc/systemd/system/multi-user.target.wants/创建软连接实现）
$ systemctl enable hdfs-dn
# 启动服务
$ systemctl start hdfs-dn
# 查看服务状态
$ systemctl status hdfs-dn
```

```
# 停止服务
$ systemctl stop hdfs-dn
```

yarn-rm

仅需要在master节点启动

```
$ vi /opt/services/yarn-rm.run.sh

#!/bin/bash

source /etc/profile

start() {
  /opt/hadoop/sbin/yarn-daemon.sh start resourcemanager
}

stop() {
  /opt/hadoop/sbin/yarn-daemon.sh stop resourcemanager
}

case "$1" in
  start)
    start
    ;;
  stop)
    stop
    ;;
  *)
esac

$ vi /opt/services/yarn-rm.service

[Unit]
Description=YARN resource manager
After=network.target

[Service]
User=root
Group=root
Type=forking
WorkingDirectory=/opt/hadoop
ExecStart=/opt/services/yarn-rm.run.sh start
ExecStop=/opt/services/yarn-rm.run.sh stop
Restart=always
RestartSec=10

[Install]
WantedBy=multi-user.target

# 修改文件属性，使其可执行
$ chmod u+x /opt/services/yarn-rm.run.sh
# 建立软连接，Systemd 默认从目录/etc/systemd/system/读取配置文件
$ ln -s /opt/services/yarn-rm.service /etc/systemd/system/yarn-rm.service
# 重新加载服务的配置文件
```

```
$ systemctl daemon-reload
# 设置开机启动（通过在/etc/systemd/system/multi-user.target.wants/创建软连接实现）
$ systemctl enable yarn-rm
# 启动服务
$ systemctl start yarn-rm
# 查看服务状态
$ systemctl status yarn-rm
# 停止服务
$ systemctl stop yarn-rm
```

yarn-nm

需要在所有slave节点启动

```
$ vi /opt/services/yarn-nm.run.sh

#!/bin/bash

source /etc/profile

start() {
  /opt/hadoop/sbin/yarn-daemon.sh start nodemanager
}

stop() {
  /opt/hadoop/sbin/yarn-daemon.sh stop nodemanager
}

case "$1" in
  start)
    start
    ;;
  stop)
    stop
    ;;
  *)
esac

$ vi /opt/services/yarn-nm.service

[Unit]
Description=YARN node manager
After=network.target

[Service]
User=root
Group=root
Type=forking
WorkingDirectory=/opt/hadoop
ExecStart=/opt/services/yarn-nm.run.sh start
ExecStop=/opt/services/yarn-nm.run.sh stop
Restart=always
RestartSec=10

[Install]
```

```
WantedBy=multi-user.target

# 修改文件属性，使其可执行
$ chmod u+x /opt/services/yarn-nm.run.sh
# 建立软连接，Systemd 默认从目录/etc/systemd/system/读取配置文件
$ ln -s /opt/services/yarn-nm.service /etc/systemd/system/yarn-nm.service
# 重新加载服务的配置文件
$ systemctl daemon-reload
# 设置开机启动（通过在/etc/systemd/system/multi-user.target.wants/创建软连接实现）
$ systemctl enable yarn-nm
# 启动服务
$ systemctl start yarn-nm
# 查看服务状态
$ systemctl status yarn-nm
# 停止服务
$ systemctl stop yarn-nm
```

mr-hs

仅需在master节点启动

```
$ vi /opt/services/mr-hs.run.sh

#!/bin/bash

source /etc/profile

start() {
  /opt/hadoop/sbin/mr-jobhistory-daemon.sh start historyserver
}

stop() {
  /opt/hadoop/sbin/mr-jobhistory-daemon.sh stop historyserver
}

case "$1" in
  start)
    start
    ;;
  stop)
    stop
    ;;
  *)
esac

$ vi /opt/services/mr-hs.service

[Unit]
Description=MapReduce JobHistoryServer
After=network.target

[Service]
User=root
Group=root
Type=forking
```

```
WorkingDirectory=/opt/hadoop
ExecStart=/opt/services/mr-hs.run.sh start
ExecStop=/opt/services/mr-hs.run.sh stop
Restart=always
RestartSec=10

[Install]
WantedBy=multi-user.target

# 修改文件属性，使其可执行
$ chmod u+x /opt/services/mr-hs.run.sh
# 建立软连接，Systemd 默认从目录/etc/systemd/system/读取配置文件
$ ln -s /opt/services/mr-hs.service /etc/systemd/system/mr-hs.service
# 重新加载服务的配置文件
$ systemctl daemon-reload
# 设置开机启动（通过在/etc/systemd/system/multi-user.target.wants/创建软连接实现）
$ systemctl enable mr-hs
# 启动服务
$ systemctl start mr-hs
# 查看服务状态
$ systemctl status mr-hs
# 停止服务
$ systemctl stop mr-hs
```

在bdcourse-0002和bdcourse-0003主机启动相应的服务

```
$ ln -s /opt/services/hdfs-dn.service /etc/systemd/system/hdfs-dn.service
$ ln -s /opt/services/yarn-nm.service /etc/systemd/system/yarn-nm.service
$ systemctl daemon-reload
$ systemctl enable hdfs-dn
$ systemctl enable yarn-nm
$ systemctl start hdfs-dn
$ systemctl start yarn-nm
```

2.8 示例测试

注意，有可能需要更换ECS配置到2vCPU 4GB才能运行。

仅在`bdcourse-0001`上执行