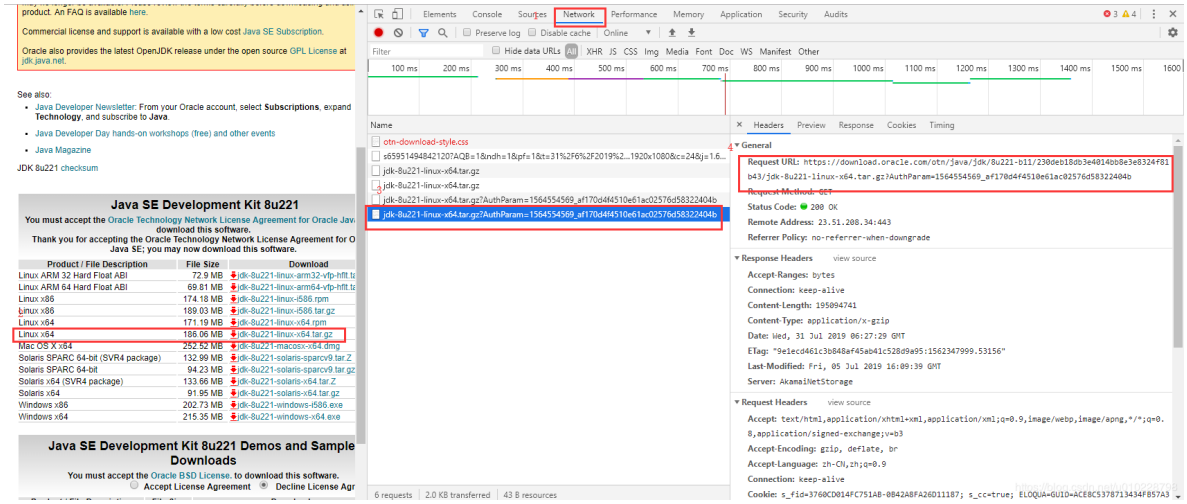
安装jdk

使用wget时会验证失败，因为需要登陆Oracle

所以可以用google浏览器桌面式安装时使用 network的方式获取下载链接




```
wget url
```

- 创建文件夹(根据自己喜好)

```
mkdir /usr/local/software/java/文件名
```

- 解压至自己的目录

```
mv 安装包路径 /usr/local/software/java
tar xvfz 安装包路径
```

- 配置环境变量

```
vi /etc/profile

export JAVA_HOME=/usr/local/sotfware/java/jdk1.8.0_231
export CLASSPATH=$JAVA_HOME/lib/
export PATH=$PATH:$JAVA_HOME/bin

source /etc/profile

java -version
```

安装Scala

从这里下载对应版本的安装包

```
wget https://downloads.lightbend.com/scala/2.13.1/scala-2.13.1.tgz
```

Other resources

You can find the installer download links for other operating systems, as well as documentation and source code archives for Scala 2.13.1 below.

Archive	System	Size
scala-2.13.1.tgz	Mac OS X, Unix, Cygwin	18.77M
scala-2.13.1.msi	Windows (msi installer)	115.13M
scala-2.13.1.zip	Windows	18.81M
scala-2.13.1.deb	Debian	582.81M
scala-2.13.1.rpm	RPM package	115.52M
scala-docs-2.13.1.txz	API docs	48.58M
scala-docs-2.13.1.zip	API docs	99.67M
scala-sources-2.13.1.tar.gz	Sources	

```
vi /etc/profile
export SCALA_HOME=/usr/local/software/scala-2.13.1
export PATH=$PATH:$SCALA_HOME/bin

source /etc/profile
scala -version
```

配置静态ip及DNS

```
vi  /etc/sysconfig/network-scripts/ifcfg-ens33

TYPE=Ethernet
PROXY_METHOD=none
BROWSER_ONLY=no
BOOTPROTO=static # 修改
DEFROUTE=yes
IPV4_FAILURE_FATAL=no
IPV6INIT=yes
IPV6_AUTOCONF=yes
IPV6_DEFROUTE=yes
IPV6_FAILURE_FATAL=no
IPV6_ADDR_GEN_MODE=stable-privacy
NAME=ens33
UUID=5de61f71-a87c-4961-a678-7d9a69c4704f
DEVICE=ens33
ONBOOT=yes # 修改
IPV6_PRIVACY=no
#ip
IPADDR=192.168.197.135
#网关
```

```
GATEWAY=192.168.197.2
#子网掩码
PREFIX0=24
#使用主的DNS
DNS1=8.8.8.8
#备用的DNS
DNS2=8.8.4.4
```

```
systemctl restart network.service # 生效
ping baidu.com # 测试
```

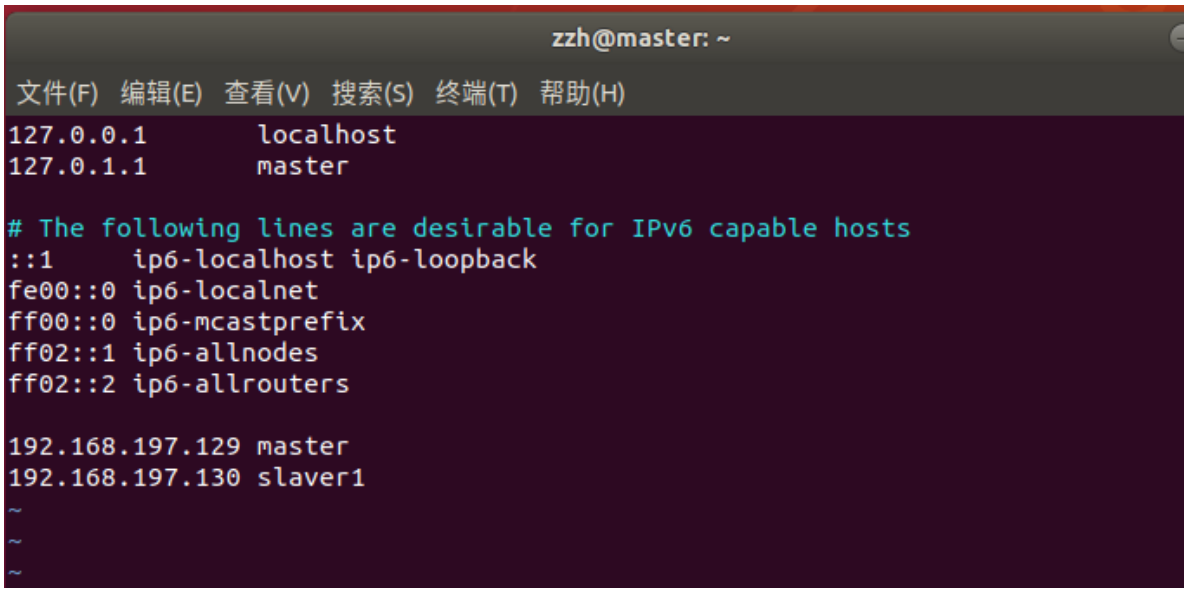
关闭防火墙

```
systemctl stop firewalld # 临时关闭防火墙
systemctl disable firewalld # 禁止开机启动
```

配置hosts文件

```
vim /etc/hosts

192.168.197.129 master
192.168.197.130 slaver1
```



添加用户账号

在所有的主机下均建立一个账号admin用来运行hadoop ，并将其添加至sudoers中

```
useradd admin    # 添加用户通过手动输入修改密码
passwd  admin  # 更改用户 admin 的密码
```

设置admin用户具有root权限 修改 /etc/sudoers 文件，找到下面一行，在root下面添加一行

```
visudo

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

用admin帐号登录，然后用命令 su - ，切换用户即可获得root权限进行操作。

```
su - admin # 切换admin用户
su - # 切换root用户
```

ssh免密

- ## 配置master和slaver

  - 切换root用户
  - 配置每台主机的SSH免密码登录环境
  - 在每台主机上生成公钥和私钥对

    ```
    # 对每台主机
    ssh-keygen -t rsa
    ```

  - 将slaver上的id_rsa.pub发送给master

    ```
    scp ~/.ssh/id_rsa.pub root@master:~/.ssh/id_rsa.pub.slaver1
    ```

  - 在master上，将所有公钥加载到用于认证的公钥文件authorized_key中，并查看生成的文件

    ```
    cat ~/.ssh/id_rsa.pub* >> ~/.ssh/authorized_keys
    cd .ssh
    ls
    ```

  - 将master上的公钥文件authorized_key分发给slaver

    ```
    scp ~/.ssh/authorized_keys root@slaver1:~/.ssh/
    ```

  - 最后使用SSH命令，检验是否能免密码登录。

    ```
    ssh slaver1
    ```

  - 进入admin用户，免密

```
su - admin
ssh-keygen -t rsa
ssh-copy-id master #本机也需要
ssh-copy-id slaver1
```

```


```

创建软件安装目录

root用户下

```
mkdir /usr/local/software
chown -R admin:admin /usr/local/software #赋予权限
```

安装Hadoop

- 从这里下载对应版本的安装包

Name	Last modified	Size	Description
Parent Directory		-	
current/	2019-10-20 20:44	-	
current2/	2019-09-24 06:32	-	
hadoop-2.10.0/	2019-10-29 18:22	-	
hadoop-2.7.7/	2018-07-19 18:12	-	
hadoop-2.8.5/	2018-09-18 09:13	-	
hadoop-2.9.2/	2018-11-19 21:45	-	
hadoop-3.1.3/	2019-10-20 20:44	-	
hadoop-3.2.1/	2019-09-24 06:32	-	
stable/	2019-09-24 06:32	-	
stable2/	2019-09-24 06:32	-	
readme.txt	2015-04-20 18:32	184	

```
wget http://apache.claz.org/hadoop/common/hadoop-3.2.1/hadoop-3.2.1.tar.gz
```

- 配置环境变量

```
vi /etc/profile

export HADOOP_HOME=/usr/local/software/hadoop-3.2.1
export HADOOP_CONF_DIR=$HADOOP_HOME/etc/hadoop
export PATH=$PATH:$HADOOP_HOME/bin

source /etc/profile
```

- 修改hadoop-env.sh、yarn-env.sh，添加JAVA_HOME

```
export JAVA_HOME=/usr/local/software/java/jdk1.8.0_231
```

```
# The java implementation to use. By default, this environment
# variable is REQUIRED on ALL platforms except OS X!
export JAVA_HOME=/home/zzh/software/java/jdk1.8.0_231
```

```
# export YARN_CONTAINER_RUNTIME_DOCKER_RUN_OVERRIDE_DISABLE=true

export JAVA_HOME=/home/zzh/software/java/jdk1.8.0_231
```

- 配置 core-site.xml

```
<configuration>
    <property>
        <name>fs.defaultFS</name>
        <value>hdfs://master:9000</value>
    </property>
    <property>
    <name>hadoop.tmp.dir</name>
    <value>/usr/local/software/hadoop-3.2.1/data</value>
    </property>
</configuration>
```

- 配置 hdfs-site.xml (创建好datanode namenode目录)

```
<configuration>
    <!--指定hdfs数据的冗余份数 默认是3-->
    <property>
        <name>dfs.replication</name>
        <value>2</value>
    </property>
    <property>
        <name>dfs.name.dir</name>
        <value>/usr/local/software/hadoop-3.2.1/hdfs/namenode</value>
    </property>
    <property>
        <name>dfs.data.dir</name>
        <value>/usr/local/software/hadoop-3.2.1/hdfs/datanode</value>
    </property>
</configuration>
```

- 配置 mapred-site.xml

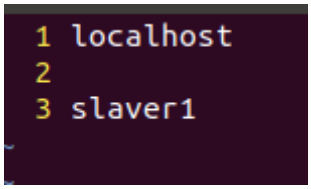
```
<configuration>
  <property>
      <name>mapreduce.framework.name</name>
      <value>yarn</value>
  </property>
   <property>
      <name>mapred.job.tracker</name>
      <value>http://master:9001</value>
  </property>
</configuration>
```

- 配置 yarn-site.xml

```
<configuration>

<!-- Site specific YARN configuration properties -->
    <property>
        <name>yarn.nodemanager.aux-services</name>
        <value>mapreduce_shuffle</value>
    </property>
    <property>
        <name>yarn.resourcemanager.hostname</name>
        <value>master</value>
    </property>
</configuration>
```

- 修改workers(hadoop2.x的是slavers)， 配置slaver节点的ip或hostname



- 将配置好的hadoop-3.2.1文件分发给所有slaver

```
scp -r /usr/local/software/hadoop-3.2.1/
admin@slaver1:/usr/local/software/
```

scp目标服务器注意写法（用户名@主机ip）

- 进入bin目录并初始化

```
cd /usr/local/software/hadoop-3.2.1/bin
./hadoop namenode -format
```

- 启动hadoop集群

```
cd /usr/local/software/hadoop-3.2.1/sbin
start-dfs.sh
start-yarn.sh
```

安装conda

- 由于ubuntu自带了python3.6，就不用安装了，可以安装pip

```
sudo apt isntall python3-pip
```

- 选择适合自己的版本，用wget命令下载

```
wget -c https://repo.continuum.io/miniconda/Miniconda3-latest-Linux-x86_64.sh
```

这里选择的是`latest-Linux`版本，所以下载的程序会随着python的版本更新而更新

-
```
bash Miniconda3-latest-Linux-x86_64.sh #运行
```

- 换清华源

```
vi ~/.condarc

channels:
  - https://mirrors.ustc.edu.cn/anaconda/pkgs/main/
  - https://mirrors.ustc.edu.cn/anaconda/cloud/conda-forge/
  - https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/free/
  - defaults
show_channel_urls: true
```

- 创建环境

```
cd /usr/local/software/miniconda3/bin
./conda create -n env_name list of packages

# ps: conda create -n my_env numpy
```

在这里，`-n env_name` 设置环境的名称（`-n` 是指名称），而 `list of packages` 是要安装在环境中的包的列表。

创建环境时，可以指定要安装在环境中的 Python 版本。这在你同时使用 Python 2.x 和 Python 3.x 中的代码时很有用。

```
conda create -n env_name python=3.6
```

- 进入环境

```
cd /usr/local/software/miniconda3/bin/


#在 Linux 上使用下面进入环境:
source activate my_env
#在 Windows 上使用下面进入环境:
activate my_env
```

可以使用 `conda list` 检查安装的包

```
# 安装包的命令
conda install package_name
```

- 离开环境

```
#在 Linux 上请键入:
source deactivate。
#在 Windows 上键入:
deactivate
```

> 参考链接： https://blog.csdn.net/abc13526222160/article/details/84624334

Jupyter运行conda环境

安装jupyter

```
pip3 install jupyter # 安装命令

jupyter notebook --generate-config # 在当前用户路径 ~/创建 .jupyter目录
```

- 进入python环境获取密匙

```
from notebook.auth import passwd
passwd() # 'sha1:0a07349659f1:85120e097cbc4e3ccb91c7936640fb8be95d1537'
```

- 进入.jupyter目录下的 jupyter_notebook_config.py 文件

```
vi ~/.jupyter/ jupyter_notebook_config.py
```

```
c.NotebookApp.ip='*'                    # 就是设置所有ip皆可访问
c.NotebookApp.password = u'sha:ce...'        # 刚才复制的那个密文'
c.NotebookApp.open_browser = False          # 禁止自动打开浏览器
c.NotebookApp.port =8888                  #随便指定一个端口
c.NotebookApp.notebook_dir = u''            # 指定工作目录
```

- 进入待激活环境

```
service activate test
```

- 安装nb_conda

```
conda install nb_conda
```

- 添加虚拟环境至notebook

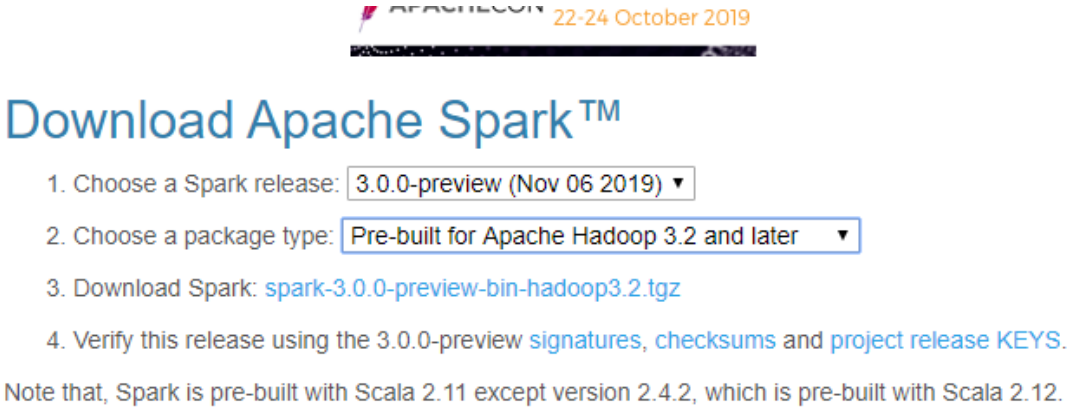
```
python -m ipykernel install --user --name 虚拟环境名 --display-name "显示环境名"
#python -m ipykernel install --user --name test --display-name "python test"
```

- 在待激活环境里启动notebook

```
jupyter notebook
```

安装Spark

- 从这里下载对应版本的安装包

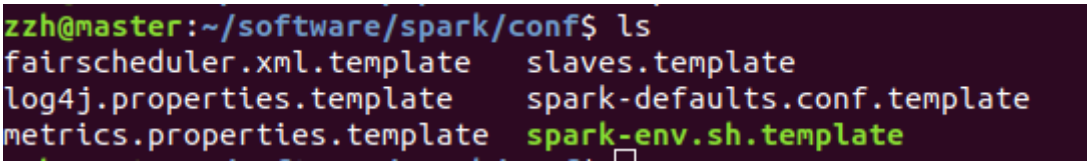



```
wget https://www-eu.apache.org/dist/spark/spark-3.0.0-preview/spark-3.0.0-preview-bin-hadoop3.2.tgz
```

- 解压后进入conf目录

```
cd /usr/local/software/spark/conf
```

在该目录下，看到很多文件都是以template结尾的，这是因为spark给我们提供的是模板配置文件，我们可以先拷贝一份，然后将.template给去掉，变成真正的配置文件后再编辑。



- 配置spark-env.sh，该文件包含spark的各种运行环境

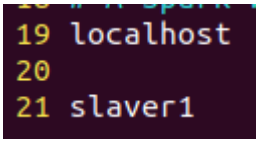
```
cp spark-env.sh.template spark-env.sh
vi spark-env.sh

export SCALA_HOME=/usr/local/software/scala-2.13.1
export JAVA_HOME=/usr/local/software/java/jdk1.8.0_231
export HADOOP_INSTALL=/usr/local/software/hadoop-3.2.1
export HADOOP_CONF_DIR=$HADOOP_INSTALL/etc/hadoop
SPARK_MASTER_IP=master # 得用ip格式
SPARK_MASTER_HOST=master # 得用ip格式
SPARK_LOCAL_IP=master
SPARK_LOCAL_DIRS=/usr/local/software/spark-3.0.0-preview-bin-hadoop3.2
SPARK_DRIVER_MEMORY=2G
```

- 配置slaves文件

```
cp slaves.template slaves
vi slaves

slaver1
```



- 将配置好的spark-2.3.1文件分发给所有slaver

```
scp -r /usr/local/software/spark/ admin@slaver1:/usr/local/software/
```

- 启动spark集群
  - 切换用户admin
  - 进入hadoop目录 ，在该目录下，启动 hadoop 文件管理系统 HDFS以及启动 hadoop 任务管理器 YARN。

    ```
    cd /usr/local/software/hadoop-3.2.1/sbin
    start-dfs.sh
    start-yarn.sh
    ```

  - 启动spark

    ```
    cd /usr/local/software/spark/sbin
    ./start-all.sh
    ```

- 查看Spark集群信息
  - 使用jps命令
  - 查看spark管理界面，在浏览器中输入：http://master:8080
- 运行 spark-shell，可以进入 Spark 的 shell 控制台
- 停止运行集群

  停止集群时，运行sbin/stop-all.sh停止Spark集群，运行sbin/stop-dfs.sh来关闭hadoop 文件管理系统 HDFS，最后运行sbin/stop-yarn.sh来关闭hadoop 任务管理器 YARN。

Jupyter里运行PySpark

用findSpark包

```
# 需要配置好SPARK_HOME环境变量
import findspark
findspark.init()

# 下面再使用pyspark函数
```

参考链接

https://blog.csdn.net/qq_15349687/article/details/82748074

踩坑

root@zzh-0: Permission denied (publickey,password)

解决办法，复制导入公钥就可以了，SSH链接需要使用公钥认证：

切换到ssh目录：cd ~/.ssh/

```
ssh-keygen -t rsa -P "" (回车)
cat $HOME/.ssh/id_rsa.pub >> $HOME/.ssh/authorized_keys
```

再次启动HDFS就可以了

```
sbin/start-dfs.sh
```

参考链接： https://yq.aliyun.com/articles/695939

but there is no HDFS_NAMENODE_USER defined. Aborting operation

```
root@zzh-1:/usr/local/software/hadoop/hadoop-3.1.3# ./sbin/start-all.sh
Starting namenodes on [zzh-1]
ERROR: Attempting to operate on hdfs namenode as root
ERROR: but there is no HDFS_NAMENODE_USER defined. Aborting operation.
Starting datanodes
ERROR: Attempting to operate on hdfs datanode as root
ERROR: but there is no HDFS_DATANODE_USER defined. Aborting operation.
Starting secondary namenodes [zzh-1]
ERROR: Attempting to operate on hdfs secondarynamenode as root
ERROR: but there is no HDFS_SECONDARYNAMENODE_USER defined. Aborting operation.
WARNING: YARN_CONF_DIR has been replaced by HADOOP_CONF_DIR. Using value of YARN
_CONF_DIR.
Starting resourcemanager
ERROR: Attempting to operate on yarn resourcemanager as root
ERROR: but there is no YARN_RESOURCEMANAGER_USER defined. Aborting operation.
Starting nodemanagers
ERROR: Attempting to operate on yarn nodemanager as root
ERROR: but there is no YARN_NODEMANAGER_USER defined. Aborting operation.
root@zzh-1:/usr/local/software/hadoop/hadoop-3.1.3#
```

在start-dfs.sh和stop-dfs.sh中：

```
HDFS_DATANODE_USER=root
HADOOP_SECURE_DN_USER=hdfs
HDFS_NAMENODE_USER=root
HDFS_SECONDARYNAMENODE_USER=root
```

```
#!/usr/bin/env bash
HDFS_DATANODE_USER=root
HADOOP_SECURE_DN_USER=hdfs
HDFS_NAMENODE_USER=root
HDFS_SECONDARYNAMENODE_USER=root
# Licensed to the Apache Software Foundation (ASF) under one or more
```

在start-yarn.sh和stop-yarn.sh中：

```
YARN_RESOURCEMANAGER_USER=root
HADOOP_SECURE_DN_USER=yarn
YARN_NODEMANAGER_USER=root
```

```
#!/usr/bin/env bash
YARN_RESOURCEMANAGER_USER=root
HADOOP_SECURE_DN_USER=yarn
YARN_NODEMANAGER_USER=root
# Licensed to the Apache Software Foundation (ASF) under one or more
```

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

解决方案关闭SELINUX

```
-- 关闭SELINUX
vim /etc/selinux/config

-- 注释掉
#SELINUX=enforcing
#SELINUXTYPE=targeted
– 添加
SELINUX=disabled
```

[https://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html]: