Greeshmika Korrapati - U00932594

Cloud Computing Project -2

1.1

Commands used:

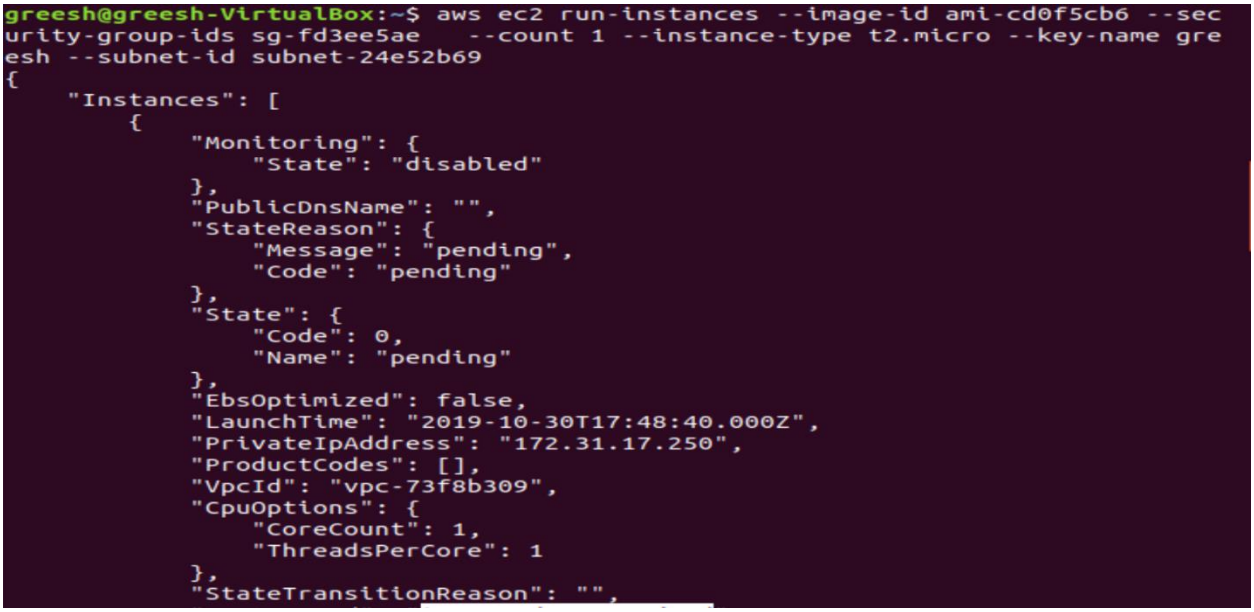
aws ec2 run-instances --image-id ami-cd0f5cb6 --security-group-ids sg-fd3ee5ae --count-1 --instance-type t2.micro --key-name greesh --subnet-id subnet-24e52b69

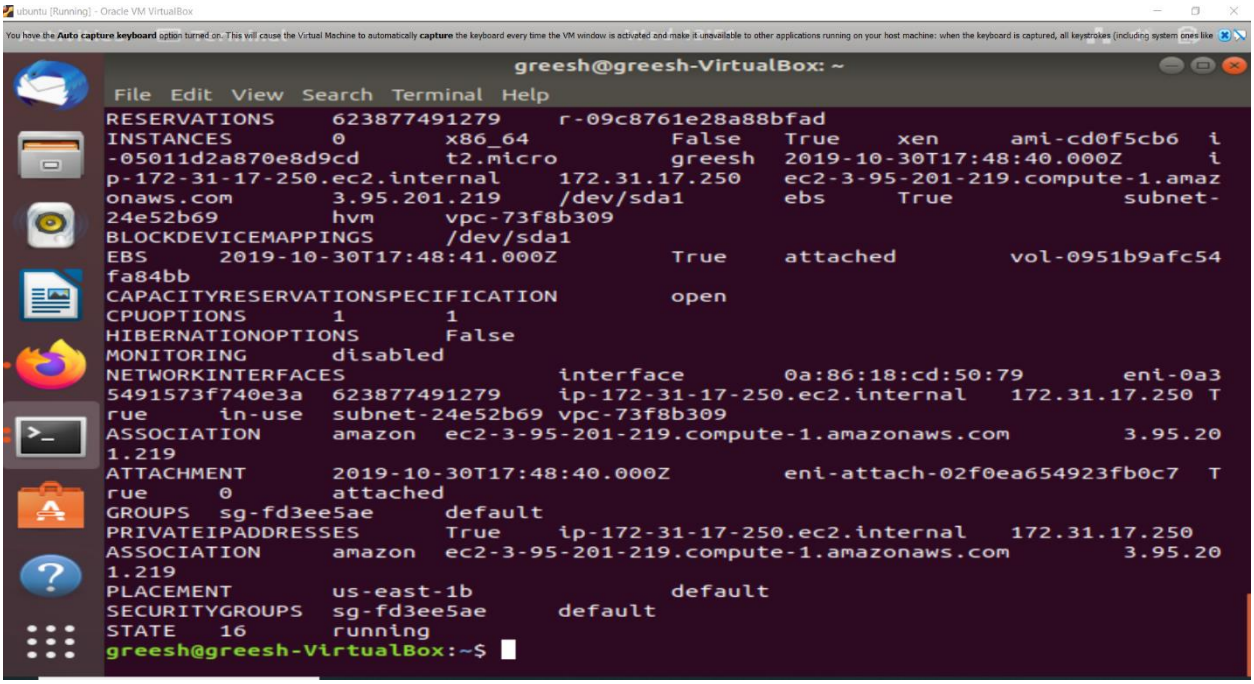
aws ec2 describe-instances

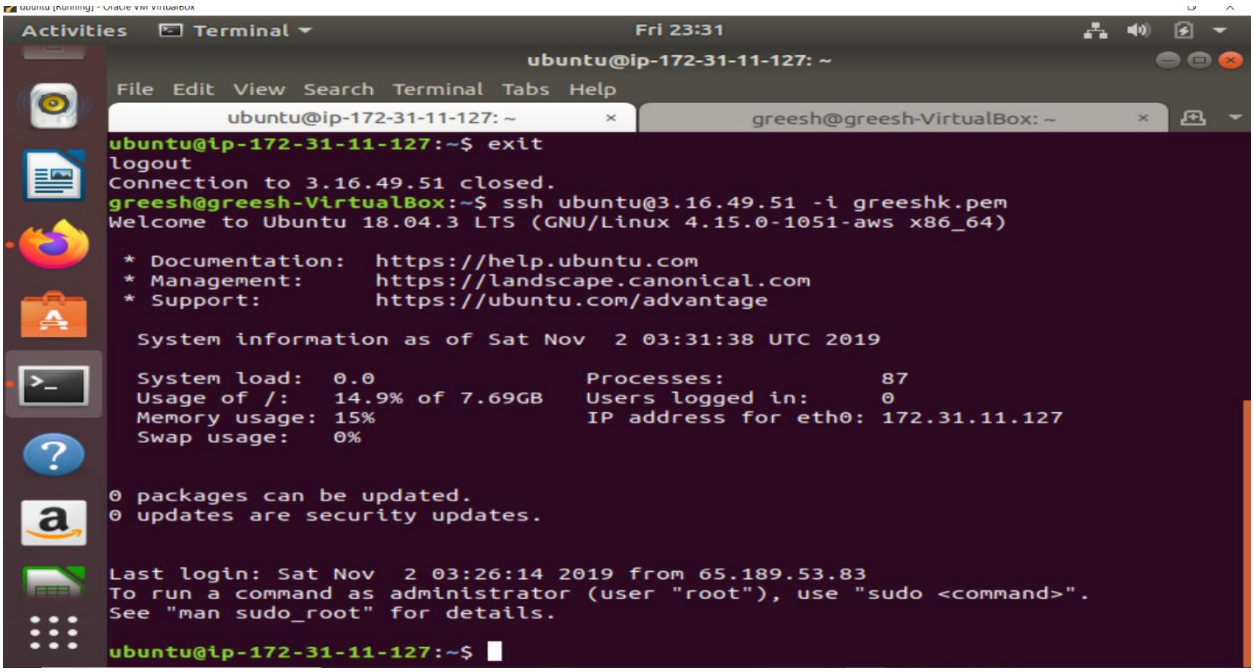
ssh ubuntu@3.16.49.51 -i greeshk.pem

aws ec2 terminate-instances --instance-ids i- 05011d2a870e8d9cd

successfully created instance and connected to ssh







1.2

```
import boto3
import time
import sys

ec2 = boto3.resource('ec2')
client = boto3.client('ec2')
def create_instance(num_instances):
```

```
    return ec2.create_instances(ImageId='ami-0d5d9d301c853a04a',
                                InstanceType='t2.micro',
                                MinCount=1,
                                MaxCount=num_instances)
def print_instanceid(instanceid):
    instancewait = client.get_waiter('instance_running')
    instancecreated=[]
    for id in instanceid:
        instancecreated.append(id)
        instancewait.wait(InstanceIds=[id])
    print(instancecreated)
    return

if __name__ == '__main__':

    print("creating instances")
    newinstance=create_instance(3)
    instanceid=(i.id for i in newinstance)
    print_instanceid(instanceid)
```

1.3

```
import boto3
import botocore

s3=boto3.resource('s3')
client=boto3.client('s3')
required_files=client.list_objects_v2(Bucket = 'wsu2017fall')

#print(required_files)

frombucket=s3.Bucket('wsu2017fall')
tobucket=s3.Bucket('greeshb')

for i in frombucket.objects.all():
    keys=i.key
    print("Keys in wsu2017fall: ",keys)
print("--------------------")

def reading_bucket(buckettoread,keysin):
      if (required_files.get('Contents', None) is not None):
         print("Reading files")
         for i in frombucket.objects.all():

          try:
            filedata=i.get()['Body'].read().decode()
            print(filedata)

          except:
            print("Directory is empty")
```

```
      else:
        print("Bucket is empty")
      return filedata
def copying_files():
 if(required_files.get('Contents',None)is not None):
  for f in required_files.get('Contents',None):
    for filekeys in frombucket.objects.all():
        sourcebucket={'Bucket':'wsu2017fall',
                      'Key':filekeys}

        try:
          s3.meta.client.copy(sourcebucket,tobucket,filekeys)

        except:
          pass
  print("copied file successfully to {}".format(tobucket))
 else:
    print("Bucket is empty")

 return
for i in tobucket.objects.all():

    print("keys in my bucket: ",i.key)
print("--------------------")

if __name__ == '__main__':

    lines_data=reading_bucket(frombucket,keys)
    copying_files()
```

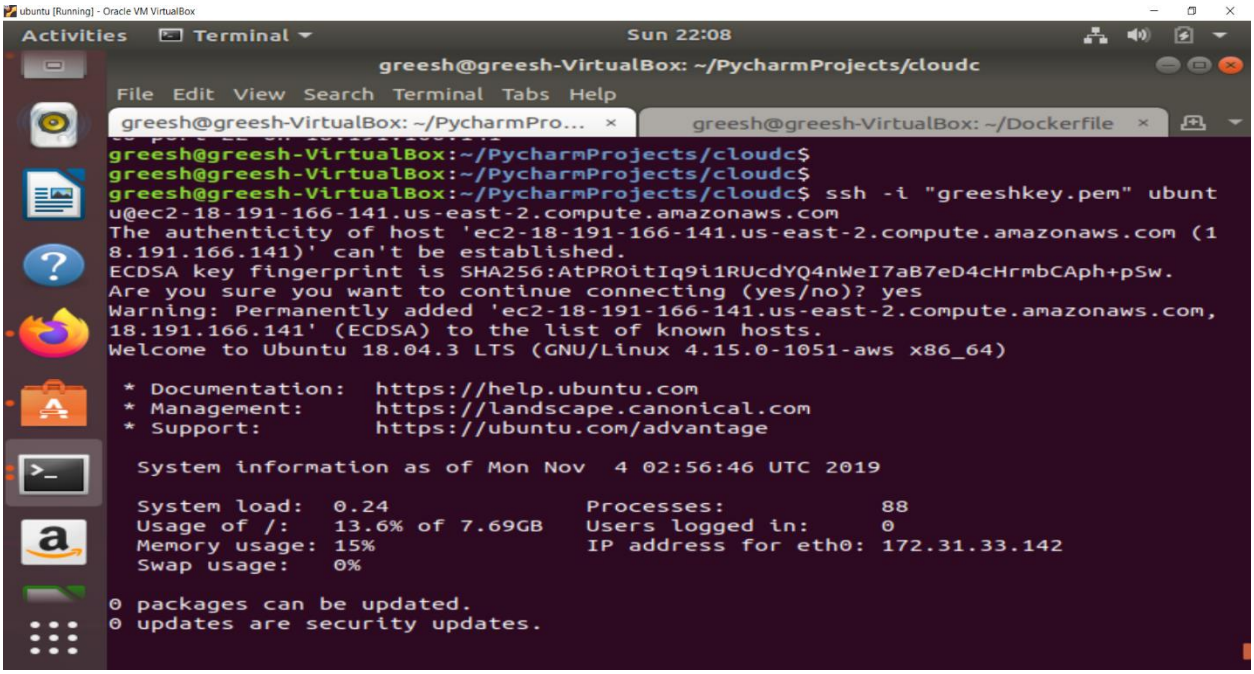
1.4

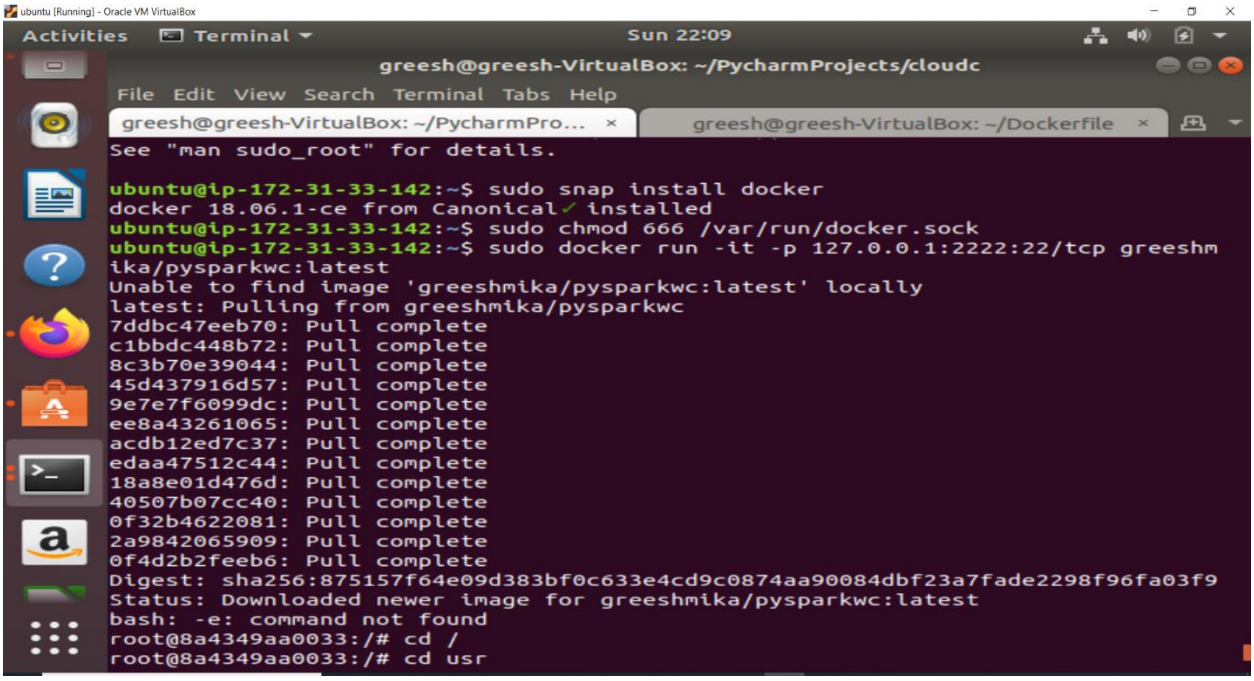
```
FROM ubuntu

RUN apt-get update && apt-get install -y \
    apt-utils \
    openssh-server \
    default-jre && \
    python3 && \
    apt-get install -y  python3 python3-setuptools && \
    apt-get clean && \
    ln -s /usr/bin/python3 /usr/bin/python && \
    apt-get install -y python-pip \
    pip install py4j \
    pip -V \
    python -V \
    python --version


RUN apt-get install -y curl
RUN echo -e "Installations done successfully" >>~/.bashrc
```

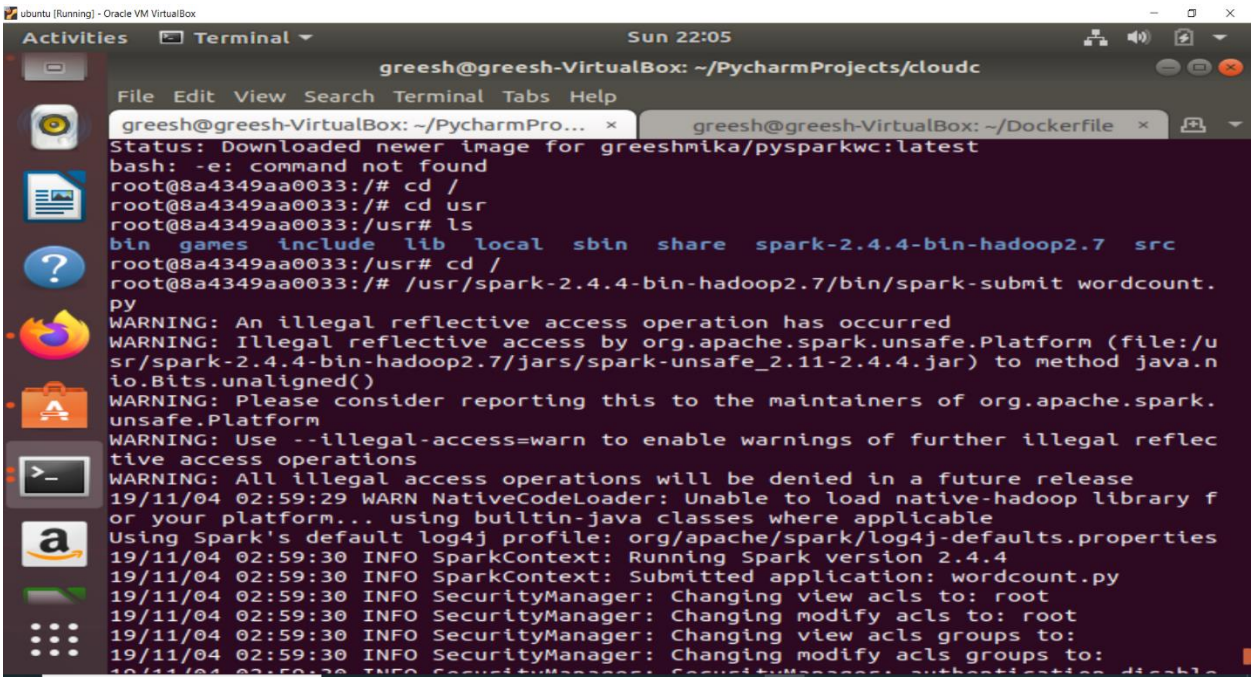
```
#RUN update-rc.d ssh defaults && ssh-keygen -f ~/.ssh/id_rsa -t rsa  -
N '' && cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
#COPY authorized_keys / ~/.ssh/authorized_keys
RUN ssh-keygen -A
RUN mkdir /var/run/sshd && chmod 755 /var/run/sshd
#RUN apt-get install -y curl
RUN curl https://www-us.apache.org/dist/spark/spark-2.4.4/spark-2.4.4-
bin-hadoop2.7.tgz >spark-2.4.4.tgz
RUN tar -xvf spark-2.4.4.tgz -C /usr/ && rm spark-2.4.4.tgz
COPY greesh.txt /
COPY wordcount.py /
RUN service ssh start
EXPOSE 22
```

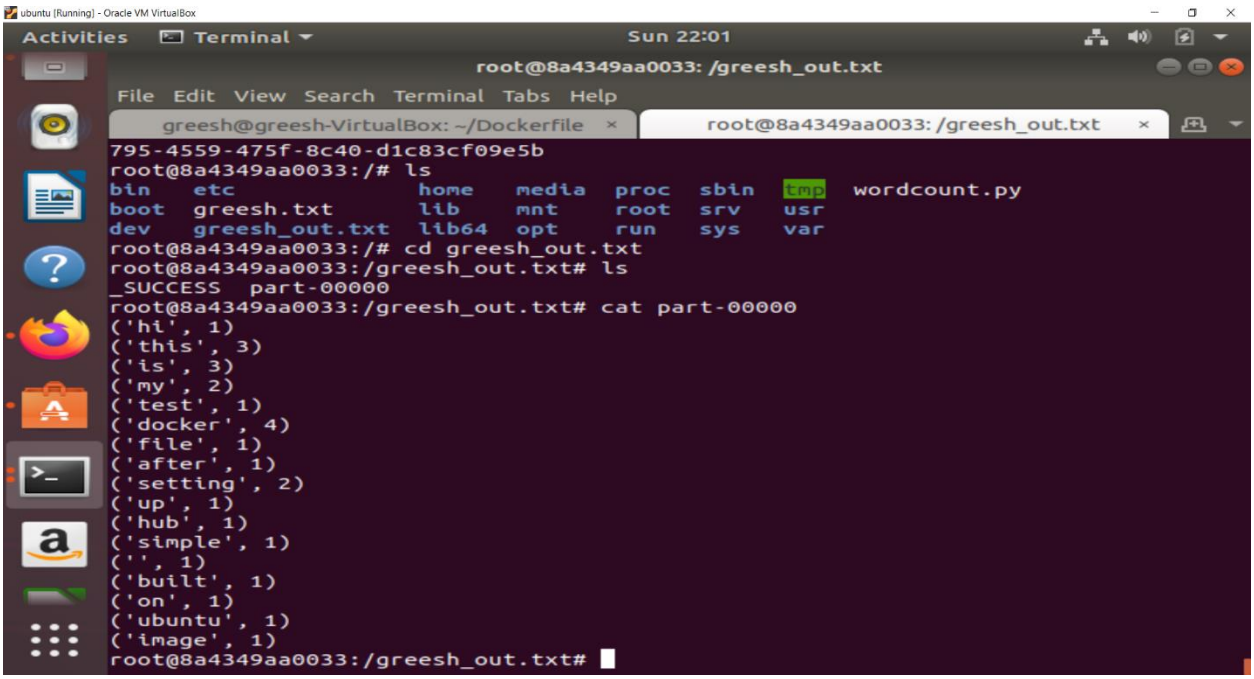
Commands used:

```
    docker build -t greeshmika/pysparkwc.latest .
    docker run -it -d -p 22 greeshmika/pysparkwc:latest
    sudo docker push Greeshmika/pysparkwc:latest
    ssh -I "greeshkey.pem" ubuntu@ec2-18-191-166-141.us-east-
    2.copute.amazonaws.com
    sudo snap install docker
    sudo chmod 666 /var/run/docker.sock
  sudo docker run -it -p 127.0.0.1:2222:22/tcp
 greeshmika/pysparkwc:latest
    usr/spark-2.4.4-bin-hadoop-2.7/bin/spark-submit wordcount.py
    cd greesh_out.txt
    cat part-00000
```




ubuntu [Running] - Oracle VM VirtualBox
Activities
Terminal
Sun 22:09
greesh@greesh-VirtualBox: ~/PycharmProjects/cloudc
File Edit View Search Terminal Tabs Help
greesh@greesh-VirtualBox: ~/PycharmPro...
greesh@greesh-VirtualBox: ~/Dockerfile
See "man sudo_root" for details.
ubuntu@ip-172-31-33-142:~$ sudo snap install docker
docker 18.06.1-ce from Canonical✓ installed
ubuntu@ip-172-31-33-142:~$ sudo chmod 666 /var/run/docker.sock
ubuntu@ip-172-31-33-142:~$ sudo docker run -it -p 127.0.0.1:2222:22/tcp greeshmika/pysparkwc:latest
Unable to find image 'greeshmika/pysparkwc:latest' locally
latest: Pulling from greeshmika/pysparkwc
7ddbc47eeb70: Pull complete
c1bbdc448b72: Pull complete
8c3b70e39044: Pull complete
45d437916d57: Pull complete
9e7e7f6099dc: Pull complete
ee8a43261065: Pull complete
acdb12ed7c37: Pull complete
edaa47512c44: Pull complete
18a8e01d476d: Pull complete
40507b07cc40: Pull complete
0f32b4622081: Pull complete
2a9842065909: Pull complete
0f4d2b2feeb6: Pull complete
Digest: sha256:875157f64e09d383bf0c633e4cd9c0874aa90084dbf23a7fade2298f96fa03f9
Status: Downloaded newer image for greeshmika/pysparkwc:latest
bash: -e: command not found
root@8a4349aa0033:/# cd /
root@8a4349aa0033:/# cd usr


ubuntu [Running] - Oracle VM VirtualBox
Activities
Terminal
Sun 22:05
greesh@greesh-VirtualBox: ~/PycharmProjects/cloudc
File Edit View Search Terminal Tabs Help
greesh@greesh-VirtualBox: ~/PycharmPro...
greesh@greesh-VirtualBox: ~/Dockerfile
Status: Downloaded newer image for greeshmika/pysparkwc:latest
bash: -e: command not found
root@8a4349aa0033:/# cd /
root@8a4349aa0033:/# cd usr
root@8a4349aa0033:/usr# ls
bin games include lib local sbin share spark-2.4.4-bin-hadoop2.7 src
root@8a4349aa0033:/usr# cd /
root@8a4349aa0033:/# /usr/spark-2.4.4-bin-hadoop2.7/bin/spark-submit wordcount.py
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by org.apache.spark.unsafe.Platform (file:/usr/spark-2.4.4-bin-hadoop2.7/jars/spark-unsafe_2.11-2.4.4.jar) to method java.nio.Bits.unaligned()
WARNING: Please consider reporting this to the maintainers of org.apache.spark.unsafe.Platform
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
19/11/04 02:59:29 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
19/11/04 02:59:30 INFO SparkContext: Running Spark version 2.4.4
19/11/04 02:59:30 INFO SparkContext: Submitted application: wordcount.py
19/11/04 02:59:30 INFO SecurityManager: Changing view acls to: root
19/11/04 02:59:30 INFO SecurityManager: Changing modify acls to: root
19/11/04 02:59:30 INFO SecurityManager: Changing view acls groups to:
19/11/04 02:59:30 INFO SecurityManager: Changing modify acls groups to:



Docker hub image link:

https://cloud.docker.com/repository/docker/greeshmika/pysparkwc

2.1

```
import boto3
import paramiko
import time
import socket
import sys

ec2 = boto3.resource('ec2')
client = boto3.client('ec2')
cp=paramiko.client.SSHClient()
cp.load_system_host_keys()


def create_instance(num_instances):
    return ec2.create_instances(ImageId='ami-0d5d9d301c853a04a',
                                KeyName="greeshkey",
                                InstanceType='t2.micro',
                                MinCount=1,
                                MaxCount=num_instances)
def print_instanceid(instanceid):
    instancewait = client.get_waiter('instance_running')
    instancecreated = []
    for id in instanceid:
        instancecreated.append(id)
        instancewait.wait(InstanceIds=[id])
    print(instancecreated)
```

```
    return

def ipaddress(instanceId):
   instances_created=client.describe_instances(InstanceIds
=[instanceId])
   ip=instances_created['Reservations'][0]['Instances'][0]
['NetworkInterfaces'][0] ['Association']['PublicIp']
   return ip
def status_monitor(newinst):
  while(True):
    for inst in newinstance:
        pemkey=paramiko.RSAKey.from_private_key_file("greeshkey.pem")
        #print(pemkey)
        cp.set_missing_host_key_policy(paramiko.AutoAddPolicy())
        print("passed")
        print(ipaddress(inst.id))
        cp.connect(hostname=ipaddress(inst.id),
username="ubuntu",pkey=pemkey)
        print("connected to ssh")
        stdin,stdout,stderr =cp.exec_command('top -bn1 | grep Cpu')
        print(inst.id ,'\t',stdout.readlines())
    time.sleep(10)

  return



if __name__ == '__main__':
    print("creatinginstances")
    newinstance = create_instance(2)
    instanceid = (i.id for i in newinstance)
    print_instanceid(instanceid)
    status_monitor(newinstance)
```

2.2

```
import boto3
import paramiko
import time
import socket
import sys

ec2 = boto3.resource('ec2')
client = boto3.client('ec2')
cp=paramiko.client.SSHClient()
cp.load_system_host_keys()
errorlog=[]

def create_instance(num_instances):
    return ec2.create_instances(ImageId='ami-0d5d9d301c853a04a',
                                KeyName="greeshkey",
                                InstanceType='t2.micro',
                                MinCount=1,
```

```
                                    MaxCount=num_instances)
def print_instanceid(instanceid):
    instancewait = client.get_waiter('instance_running')
    instancecreated = []
    for id in instanceid:
        instancecreated.append(id)
        instancewait.wait(InstanceIds=[id])
    print(instancecreated)
    return

def ipaddress(instanceId):
   instances_created=client.describe_instances(InstanceIds
=[instanceId])
   ip=instances_created['Reservations'][0]['Instances'][0]
['NetworkInterfaces'][0] ['Association']['PublicIp']
   return ip

def paramiko_monitor(newinstance):
   try:
    for inst in newinstance:
        pemkey=paramiko.RSAKey.from_private_key_file("greeshkey.pem")
        cp.set_missing_host_key_policy(paramiko.AutoAddPolicy())
        print("key file is extracted")
        cp.connect(hostname=ipaddress(inst.id),
username="ubuntu",pkey=pemkey)
        print("connected to ssh")
        stdin,stdout,stderr =cp.exec_command('sudo apt-get update &&
sudo apt-get install -y docker.io && sudo docker run -d -t ubuntu sh')
        print("Docker Installation started")
        print("Docker run command is successfull")
        for instaid in stdout:
            print(inst.id,'\t',ipaddress(inst.id),'\t',instaid)
   except:
       errorlog.append(inst.id)
       for i in errorlog:
           print(i)

   docker_monitor(newinstance)
   return

def docker_monitor(newinstance):
  while(True):
    for inst in newinstance:
        pemkey =
paramiko.RSAKey.from_private_key_file("greeshkey.pem")
        cp.connect(hostname=ipaddress(inst.id), username="ubuntu",
pkey=pemkey)
        print("connected to ssh")
        stdin, stdout, stderr = cp.exec_command('sudo docker ps | grep
ubuntu')
        for docid in stdout:
           doc_container=docid.split()[0]
```

```
            stdin, stdout, stderr = cp.exec_command('sudo docker exec
{} top -bn1 | grep Cpu'.format(doc_container))
         for k in stdout:
            print('{} \t {} \t {}'.format(inst.id,doc_container,k))
     time.sleep(10)

   return



if __name__ == '__main__':
     print("creating instances")
     newinstance = create_instance(2)
     instanceid = (i.id for i in newinstance)
     print_instanceid(instanceid)
     paramiko_monitor(newinstance)
```