Hadoop Using Ansible over AWS

In this module, we shall launch hadoop cluster using ansible over AWS. Well, it might sound too sophisticated, but it is quite simple to perform.

Let us understand step by step how we can do the following task:

We shall use AWS AMI which is available on AWS.

First of all, we move to the root of aws linux.

Then, we install ansible using: **sudo amazon-linux-extras install ansible2**

```
root@ip-172-31-1-188:~
[ec2-user@ip-172-31-1-188 ~]$ sudo su -
Last login: Thu Oct 29 12:46:01 UTC 2020 on pts/0
[root@ip-172-31-1-188 ~]# sudo amazon-linux-extras install ansible2
Installing ansible
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Cleaning repos: amzn2-core amzn2extra-ansible2 amzn2extra-docker
12 metadata files removed
```

Now, we check the version of ansible if it is successfully installed.

Use the command: **ansible --version**

```
[root@ip-172-31-1-188 ~]# ansible --version
ansible 2.9.13
  config file = /etc/ansible/ansible.cfg
  configured module search path = [u'/root/.ansible/plugins/modules', u'/usr/sha
re/ansible/plugins/modules']
  ansible python module location = /usr/lib/python2.7/site-packages/ansible
  executable location = /usr/bin/ansible
  python version = 2.7.18 (default, Aug 27 2020, 21:22:52) [GCC 7.3.1 20180712 (
Red Hat 7.3.1-9)]
```

CREATING INVENTORY

Inventory is basically a file where we store all the necessary details of the system which we want to setup as a target node. The details include ip, username, password, protocol. However, the process of creating an inventory on aws is quite different from that on a local VM.

Foremost, we make the changes in the ansible.cfg file.

vim /etc/ansible/ansible.cfg

When you press enter, you will find [defaults]. Write the required text as shown in the image.

```
[root@ip-172-31-1-188 ~]# vim /etc/ansible/ansible.cfg
[root@ip-172-31-1-188 ~]#
```

```
# finds first

[defaults]
inventory = /root/ip.txt
# some basic default values...
```

Ones you save the file, create a new file, which is basically the inventory file. In that file write the following details:

Ip of the target node, username, and protocol.

```
root@ip-172-31-1-188:~
35.175.127.169 ansible_user=root ansible_connection=ssh
~
~
~
~
~
~
~
```

The noteworthy thing is that, we have not provided any security key till now. This step, of providing the security key is completely different as compared to what we do in the local VMs.

Write the following commands in the root directory:

cd .ssh

ssh-keygen

Press enter key unless you see that box-like figure appearing.

Use ls command to see the files present in the directory.

Open id_rsa.pub file using vim id_rsa.pub and copy the content as it is.

Now, go to your **target node** and follow the steps are stated:

Go to the root directory.

cd .ssh

ssh-keygen

vim authorized_keys

Paste the content that you have copied from the control node below the already written text.

```
[root@ip-172-31-1-188 ~]# cd .ssh
[root@ip-172-31-1-188 .ssh]# ls
authorized_keys  data.pem  known_hosts
[root@ip-172-31-1-188 .ssh]# ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:K9MhtMXBJ5ZVewsRMNhrnA5Xq1zYTDn0v+ZgTiJ95XY root@ip-172-31-1-188.ec2.inte
rnal
The key's randomart image is:
+---[RSA 2048]----+
|       ..=++=o   |
|       .*.o.=+   |
|      ..o+ Oooo  |
|     . o. B =o o |
|      o S* o  ...|
|       o o=   o .|
|      o o. o = =E|
|       o  . * = .|
|            . . |
+----[SHA256]-----+
[root@ip-172-31-1-188 .ssh]# ls
authorized_keys  data.pem  id_rsa  id_rsa.pub  known_hosts
[root@ip-172-31-1-188 .ssh]# vim id_rsa.pub
[root@ip-172-31-1-188 .ssh]#
```

Come back to the control node and type **ansible all –m ping** to see that you are successfully connected to the system.

```
[root@ip-172-31-1-188 ~]# ansible all -m ping
The authenticity of host '35.175.127.169 (35.175.127.169)' can't be established.
ECDSA key fingerprint is SHA256:HIbetGb5/OaxqhViqZVPHNyqJuIyxgeoUpemlKhT3mA.
ECDSA key fingerprint is MD5:26:31:1a:0b:c1:54:f8:cd:77:18:e3:fc:cb:c5:1c:07.
Are you sure you want to continue connecting (yes/no)? yes
[WARNING]: Platform linux on host 35.175.127.169 is using the discovered Python
interpreter at /usr/bin/python, but future installation of another Python
interpreter could change this. See https://docs.ansible.com/ansible/2.9/referen
ce_appendices/interpreter_discovery.html for more information.
35.175.127.169 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": false,
    "ping": "pong"
}
```

For configuring a hadoop cluster, we at least need 2 systems. The ip that we have already added acts as the data node. We shall use one more system and paste the content in its id_rsa.pub file and make it the host.

The *final inventory file* will look like this:

root@ip-172-31-1-188:~

```
[slave]
35.175.127.169 ansible_user=root ansible_connection=ssh

[master]
3.239.27.147 ansible_user=root ansible_connection=ssh
~
~
~
```

The *final ping* will look like:

```
[root@ip-172-31-1-188 .ssh]# ls
authorized_keys  data.pem  id_rsa  id_rsa.pub  known_hosts
[root@ip-172-31-1-188 .ssh]# vim id_rsa.pub
[root@ip-172-31-1-188 .ssh]# ansible all -m ping
[WARNING]: Platform linux on host 3.239.27.147 is using the discovered Python
interpreter at /usr/bin/python, but future installation of another Python
interpreter could change this. See https://docs.ansible.com/ansible/2.9/referen
ce_appendices/interpreter_discovery.html for more information.
3.239.27.147 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": false,
    "ping": "pong"
}
[WARNING]: Platform linux on host 35.175.127.169 is using the discovered Python
interpreter at /usr/bin/python, but future installation of another Python
interpreter could change this. See https://docs.ansible.com/ansible/2.9/referen
ce_appendices/interpreter_discovery.html for more information.
35.175.127.169 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": false,
    "ping": "pong"
}
```

Now, we shall write the playbook to configure hadoop cluster. For that, first we need jdk and hadoop file in the control node, so that it can be copied to the target node.

The playbook looks as follows:

root@ip-172-31-1-188:/t11

```
- hosts: all
  tasks:
  - name: "Sending jdk to the target node"
    copy:
            dest: "/root"
            src:  "/home/ec2-user/jdk-8u171-linux-x64.rpm"
  - name: "Sending  hadoop to the target node"
    copy:
            dest: "/root"
            src: "/home/ec2-user/hadoop-1.2.1-1.x86_64.rpm"


  - name: "installing JDK"
    command: "rpm -i jdk-8u171-linux-x64.rpm"

  - name: "Installing hadoop"
    command: "rpm -i hadoop-1.2.1-1.x86_64.rpm --force"
#Setting up the namenode
- hosts: master



  tasks:
  - name: "Creating nn directory"
    file:
            state: directory
            path: "/nn"
  - name: "Setting up hdfs-site.xml"
    template:
            dest: "/etc/hadoop/hdfs-site.xml"
            src: "masterhdfs.xml"
  - name: "Setting up core-site.xml"
    template:
            dest: "/etc/hadoop/core-site.xml"
            src: "mastercore.xml"
#Setting yp the datanode
- hosts: slave



  tasks:
  - name: "Creating dn directory"
    file:
            state: directory
            path: "/dn"
  - name: "hdfs in slave"
    template:
            dest: "/etc/hadoop/hdfs-site.xml"
            src: "slavehdfs.xml"
  - name: "core in slave"
"ans.yml" 77L, 1888C
```

The complete yml file of the playbook is available on my github.

After writing the playbook, before running it, we shall check the syntax using the following command:

```
[root@ip-172-31-1-188 t11]# ansible-playbook --syntax-check ans.yml

playbook: ans.yml
```

If the name of your yml file is displayed, it means there is no error in the code.

Now, we shall create hdfs-site.xml and core-site.xml files for master and slave nodes.

```
[root@ip-172-31-1-188 t11]# ls
ans.yml  mastercore.xml  masterhdfs.xml  slavecore.xml  slavehdfs.xml
[root@ip-172-31-1-188 t11]#
```

The content of the core file of master and slave is same, so we shall write it ones and copy it. We write the following content in mastercore.xml

root@ip-172-31-1-188:/t11

```
<configuration>
<property>
<name>fs.default.name</name>
<value>hdfs://3.239.175.46:8001</value>
</property>
</configuration>
~
~
~
~
~
```

Copy the file in slave core file using**: cp mastercore.xml slavecore.xml**

Now, we write the content in slavehdfs.xml:

root@ip-172-31-1-188:/t11

```
<configuration>
<property>
<name>dfs.data.dir</name>
<value>{{ slave }}</value>
</property>
</configuration>
~
~
~
~
```

The content of masterhdfs.xml is as follows:

root@ip-172-31-1-188:/t11

```
<configuration>
<property>
<name>dfs.name.dir</name>
<value>{{ master }} </value>
</property>
</configuration>
~
~
~
```

The output is as follows:

```
root@ip-172-31-1-188:/t11
[root@ip-172-31-1-188 t11]# ansible-playbook ans.yml

PLAY [all] ***********************************************************************************************************************************************

TASK [Gathering Facts] ***********************************************************************************************************************************
[WARNING]: Platform linux on host 3.239.175.46 is using the discovered Python interpreter at /usr/bin/python, but future installation of another Python
this. See https://docs.ansible.com/ansible/2.9/reference_appendices/interpreter_discovery.html for more information.
ok: [3.239.175.46]
[WARNING]: Platform linux on host 3.235.199.96 is using the discovered Python interpreter at /usr/bin/python, but future installation of another Python
this. See https://docs.ansible.com/ansible/2.9/reference_appendices/interpreter_discovery.html for more information.
ok: [3.235.199.96]

TASK [Sending jdk to the target node] ********************************************************************************************************************
ok: [3.235.199.96]
ok: [3.239.175.46]

TASK [Sending  hadoop to the target node] ****************************************************************************************************************
ok: [3.239.175.46]
ok: [3.235.199.96]

TASK [installing JDK] ************************************************************************************************************************************
[WARNING]: Consider using the yum, dnf or zypper module rather than running 'rpm'.  If you need to use command because yum, dnf or zypper is insufficie
to this command task or set 'command_warnings=False' in ansible.cfg to get rid of this message.
changed: [3.235.199.96]
changed: [3.239.175.46]

TASK [Installing hadoop] *********************************************************************************************************************************
changed: [3.235.199.96]
changed: [3.239.175.46]

PLAY [master] ********************************************************************************************************************************************

TASK [Gathering Facts] ***********************************************************************************************************************************
ok: [3.239.175.46]

TASK [Creating nn directory] *****************************************************************************************************************************
ok: [3.239.175.46]

TASK [Setting up hdfs-site.xml] **************************************************************************************************************************
changed: [3.239.175.46]

TASK [Setting up core-site.xml] **************************************************************************************************************************
changed: [3.239.175.46]

PLAY [slave] *********************************************************************************************************************************************

TASK [Gathering Facts] ***********************************************************************************************************************************
ok: [3.235.199.96]

TASK [Creating dn directory] *****************************************************************************************************************************
ok: [3.235.199.96]
```

Now, we can use jps to check if the datanode and masternode is active.

```
[root@ip-172-31-8-133 ~]# jps
24208 DataNode
24268 Jps
```

Similarly, check the jps of the master node. Don't forget to format the master node ones before starting the service.

Hence, we successfully complete deploying hadoop using automation via ansible over AWS ;-)