**Ansible Dynamic Inventory**



To accomplish this task, follow the steps below**:**

* Set up Lab
* Configure the **dynamic inventory**
* Run Ad hoc commands to test.

## **1- Set up Test environment**

### **1-a) Write the infrastructure code**

Here we will set up the test environment using Terraform

open git bash and move to the home directory

| cd ~ |
| --- |

Now clone the repository from GitHub

| git clone https://github.com/utrains/Ansible-Lab.git |
| --- |

now cd into **Ansible-Lab**

| cd Ansible-Lab |
| --- |

before launching the infrastructure, let's create a file to define our terraform variable values.

| code config.tfvars |
| --- |

Copy and paste the code below

| #for the istance configurations, we can use any of the values "amazon,debian,ubuntu" to configure our nodes  # to use amazon linux, ubuntu or debian. instance\_configurations = {  "master" = {  instance\_type = "t3.micro"  name = "master-instance"  user\_data = "install.sh"  ami = "amazon"  },  "node1" = {  instance\_type = "t3.micro"  name = "node1-instance"  user\_data = ""  ami = "amazon"  },  "node2" = {  instance\_type = "t4g.micro"  name = "node2-instance"  user\_data = ""  ami = "ubuntu"  } }  region = "us-east-1" keypair-name= "ansible-key" |
| --- |

Now let's go ahead and launch the infrastructure

### **1-b) Launch the Infrastructure**

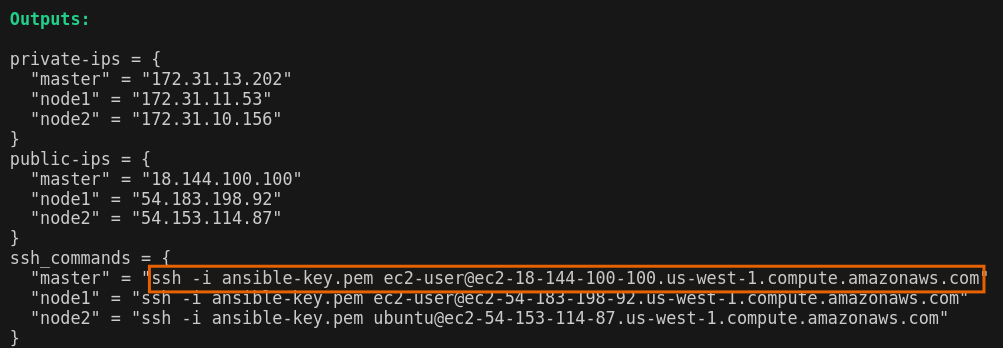
To launch the infrastructure, run the commands below

| terraform init |
| --- |

| terraform plan -var-file="config.tfvars" |
| --- |

| terraform apply -var-file="config.tfvars" --auto-approve |
| --- |

If the apply is successful, you should get the output below.



**You can see the ssh command to connect to each instance**

Now that we are done, let's **SSH** into our **master server** and configure Ansible. In my case, I used the command highlighted in the image above.

## **2- Configure the dynamic inventory file**

* SSH into your master server.

Once in your server, before we write the inventory file, we will need to configure AWS CLI on our **master server.**

* [**Here**](https://docs.google.com/document/d/1spbrSaPJWo9_i4LoI5CO5buDlUacDac7JTNeva-PuWE/edit?usp=sharing) **is the link to configure AWS CLI on your terminal**
* Once done, create the file **aws\_ec2.yml**

| sudo vi aws\_ec2.yml |
| --- |

* Add the code below to it

| plugin: aws\_ec2 regions:  - us-east-1 groups: #Dev servers group  dev: "'node1' in tags.Environment" # QA servers Group  qa: "'node2' in tags.Environment" |
| --- |

* Also, create an **ansible.cfg** and add the code below into it.

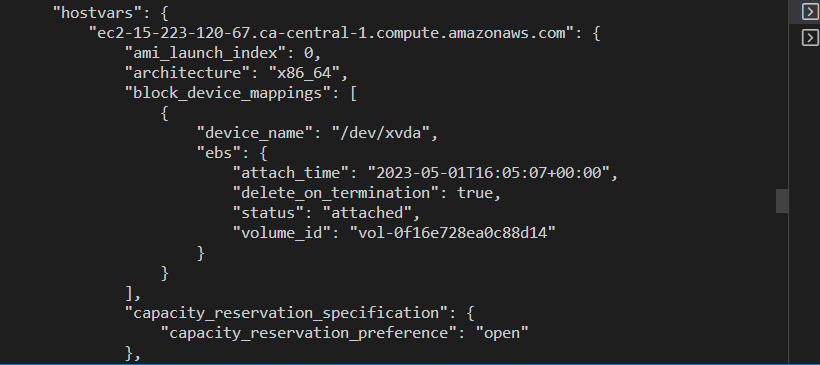
| sudo vi ansible.cfg |
| --- |

* Add the code below inside and save

| [defaults] host\_key\_checking = False inventory=aws\_ec2.yml interpreter\_python=auto\_silent localhost\_warning=false deprecation\_warnings = False private\_key\_file = /home/ec2-user/ansible-key.pem |
| --- |

* Let's go ahead and test if the inventory has been configured.

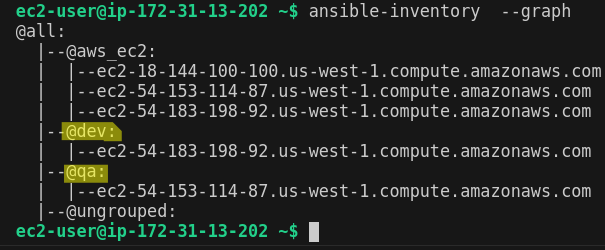
| ansible-inventory --list |
| --- |



* Let's run the Ansible graph and see the various groups.

To get the graph, run the command below:

| ansible-inventory --graph |
| --- |



The highlighted groups are the servers with environment tag **dev** and **QA,** respectively.

In the next phase, we will run some Ad-hoc commands to test the dynamic inventory.

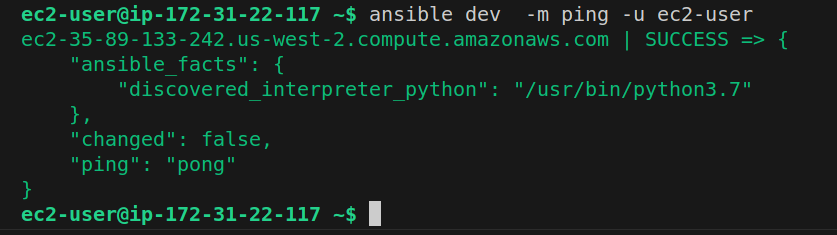
## **3- Test the inventory**

In this section, we will run Ad hoc commands to access our nodes.

### **3-a) Run the ping command**

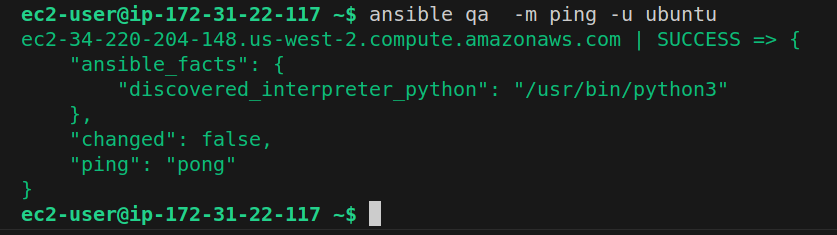
Let's ping all the dev servers

| ansible dev -m ping -u ec2-user |
| --- |



Now let's ping the QA servers:

| ansible qa -m ping -u ubuntu |
| --- |

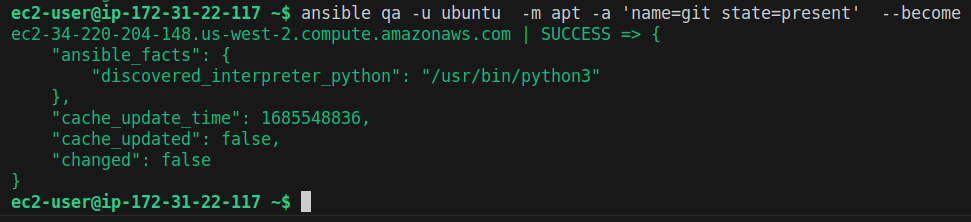


In the next test, we will install **Git**

### **3-b) Install git**

Here we will install Git just on the **qa** servers. To do that, run the command below:

| ansible qa -u ubuntu -m apt -a 'name=git state=present' --become |
| --- |



You just installed a package (Git) on your **qa servers**. For verification, SSH into the **qa servers** and type the following command to get the Git version: **git version**

