**Automating Infrastructure using Terraform**

**Step 1: Install and set up Terraform on your local system.**

1.1 Create a folder

***mkdir project***

***cd project***

1.2 Run the following command to download the appropriate package (make sure to get the latest version from [Terraform Versions | HashiCorp Releases](https://releases.hashicorp.com/terraform/))

***wget https://releases.hashicorp.com/terraform/1.0.10/terraform\_1.0.10\_linux\_amd64.zip***

**Step 2: Add the binary file into the bin directory**

2.1 Run the below set of commands to download, unzip, and move the terraform binary file to the **bin** directory:

***sudo apt-get install unzip***

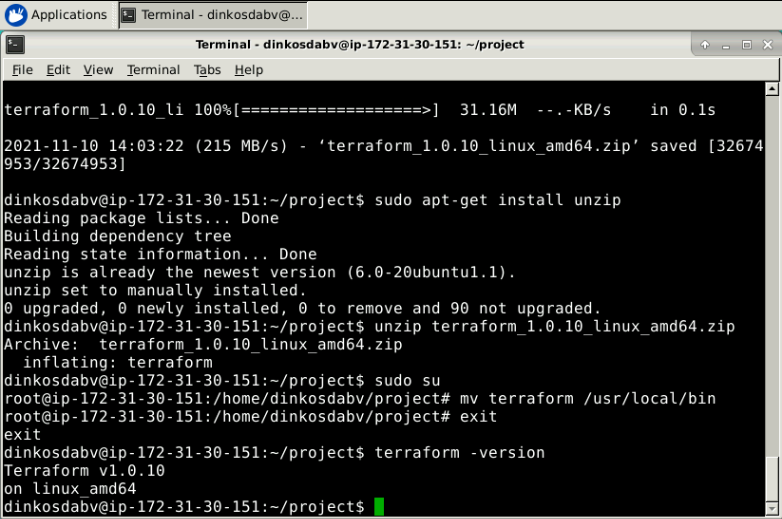
***unzip terraform\_1.0.10\_linux\_amd64.zip***

***sudo su***

***mv terraform /usr/local/bin***

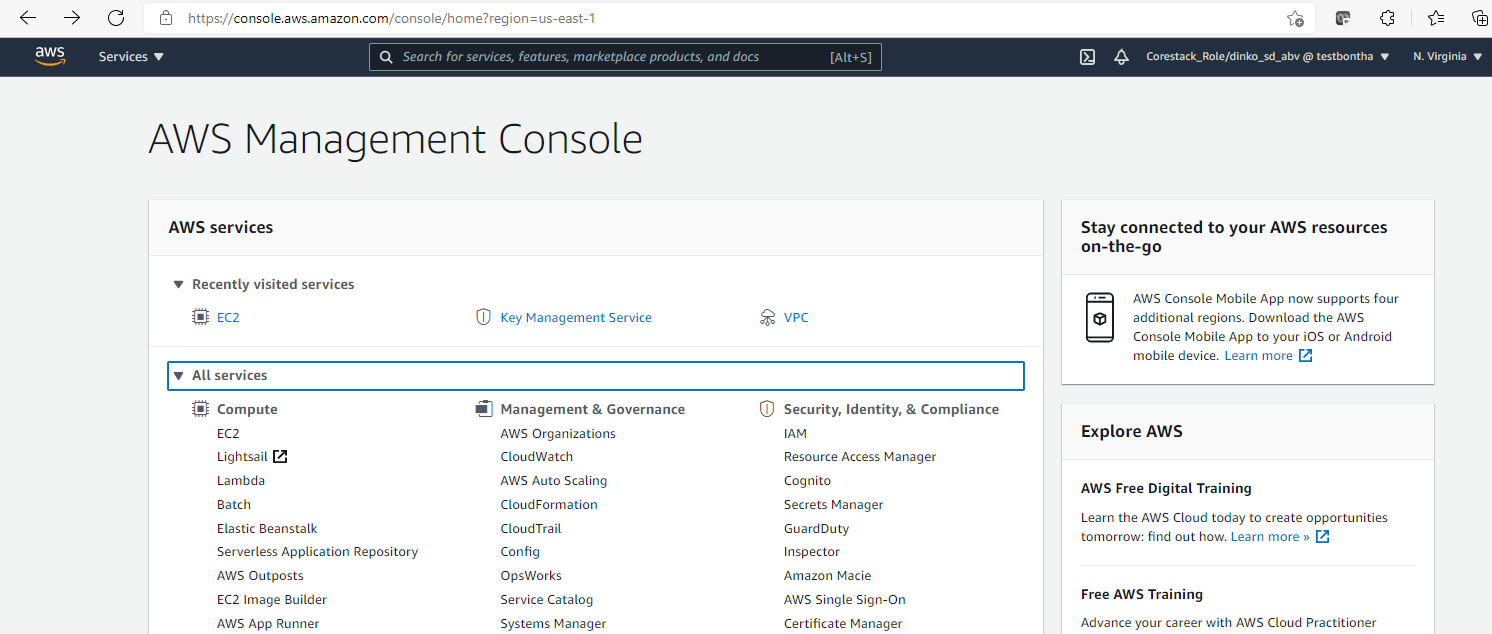
***cd ..***

***terraform -version***

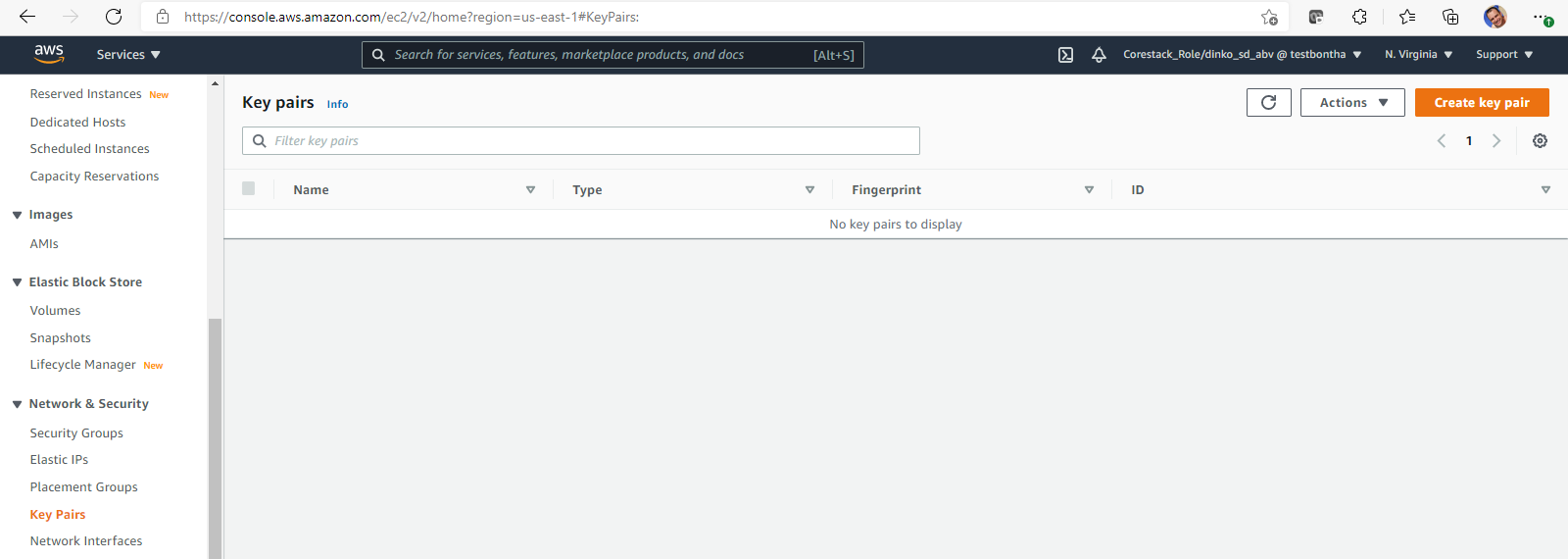


**Step 3: Create an AWS EC2 instance with Terraform**

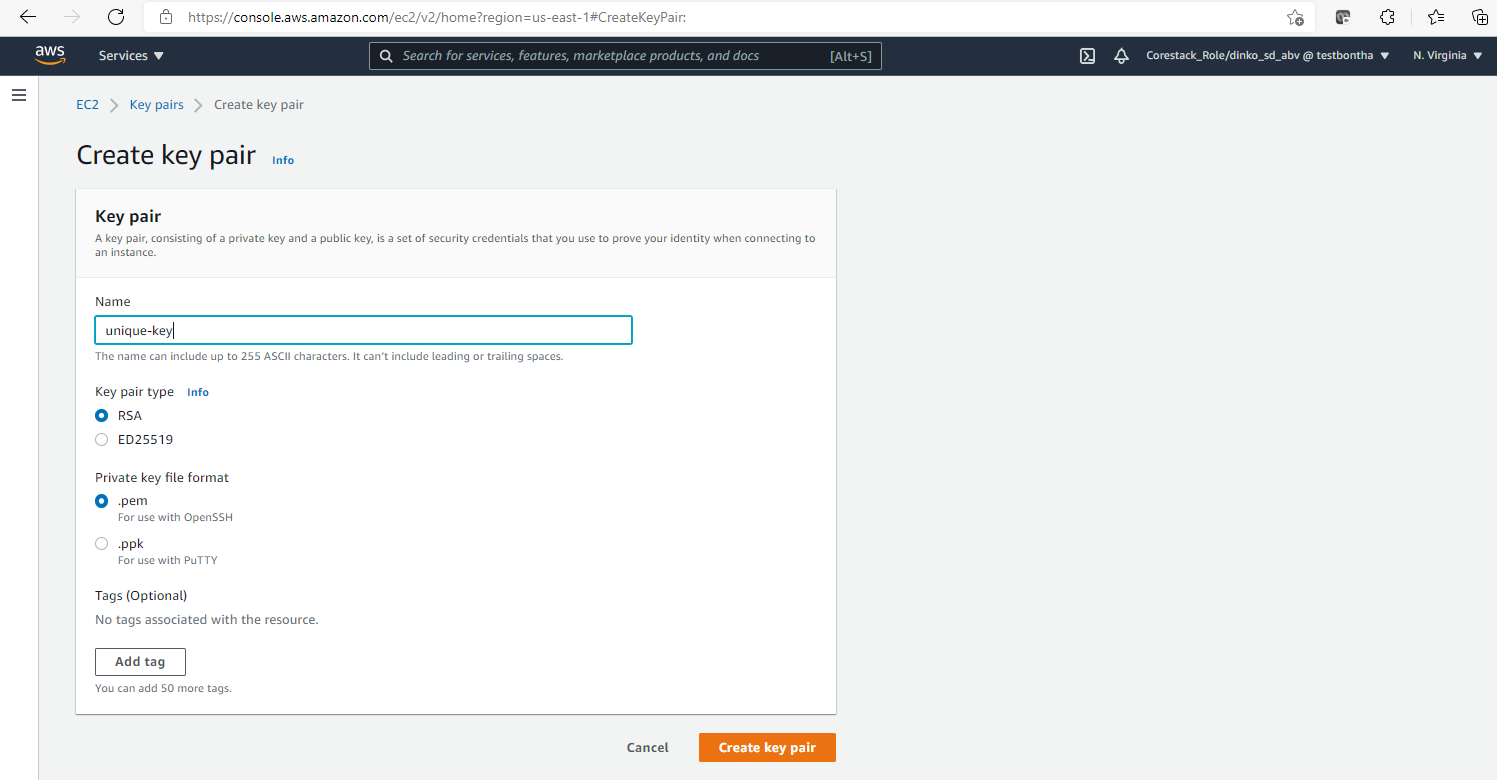
3.1 Create your AWS Keypair. For this step, login with your profile to the AWS Management Console, navigate to All Services 🡪 EC2:



On the next screen navigate to Network and Security 🡪 Key Pairs 🡪 Create key pair:



Next, give it a name and choose Private key format file “.pem”:



Click on Create key pair. It will automatically download the file to your local Downloads folder.

3.2 Prepare a new terraform file for execution.

Going back to the console of your local system, navigate to your project folder and create a new Terraform file for execution:

***vi <name of the file>.tf***

Configure the following script:

***terraform {***

***required\_providers {***

***aws = {***

***source = "hashicorp/aws"***

***version = "~> 3.27"***

***}***

***}***

***required\_version = ">= 0.14.9"***

***}***

***provider "aws" {***

***profile = "<your AWS profile>"***

***region = "<AWS region to be used>"***

***access\_key = "<your AWS access key>"***

***secret\_key = "<your AWS secret key>"***

***token = "<your AWS token>"***

***}***

***resource "aws\_instance" "<name of your future instance>" {***

***ami = "<the ami you have chosen for your instance to create>"***

***instance\_type = "<type of instance>"***

***key\_name = "<the name of your Keypair>"***

***vpc\_security\_group\_ids = [aws\_security\_group.<ID of your security group>.name]***

***tags = {***

***Name = "<name of your future instance>"***

***}***

***}***

***resource "aws\_security\_group" "<ID of your security group>" {***

***name = "<name of the AWS security group to be created>"***

***ingress {***

***from\_port = 22***

***to\_port = 22***

***protocol = "tcp"***

***cidr\_blocks = ["0.0.0.0/0"]***

***}***

***ingress {***

***from\_port = 443***

***to\_port = 443***

***protocol = "tcp"***

***cidr\_blocks = ["0.0.0.0/0"]***

***}***

***ingress {***

***from\_port = 8080***

***to\_port = 8080***

***protocol = "tcp"***

***cidr\_blocks = ["0.0.0.0/0"]***

***}***

***egress {***

***from\_port = 0***

***to\_port = 0***

***protocol = -1***

***cidr\_blocks = ["0.0.0.0/0"]***

***}***

***tags = {***

***Name = "<name of the AWS security group to be created>"***

***}***

***}***

We want to configure also an outputs file to give us the ID and public IP address of the instance, which will be used further.

***vi outputs.tf***

Configure the following script:

***output "instance\_id" {***

***description = "ID of the EC2 instance:"***

***value = aws\_instance.<name of your future instance>.id***

***}***

***output "instance\_public\_ip" {***

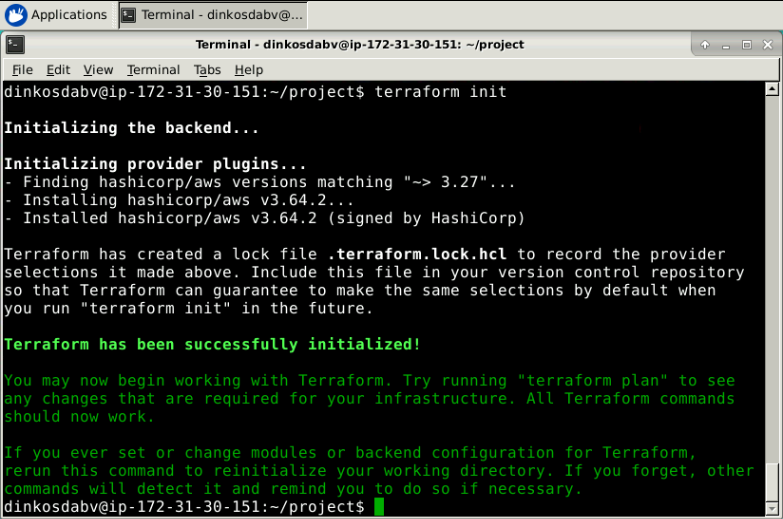
***description = "EC2 instance public IP:"***

***value = aws\_instance.<name of your future instance>.public\_ip***

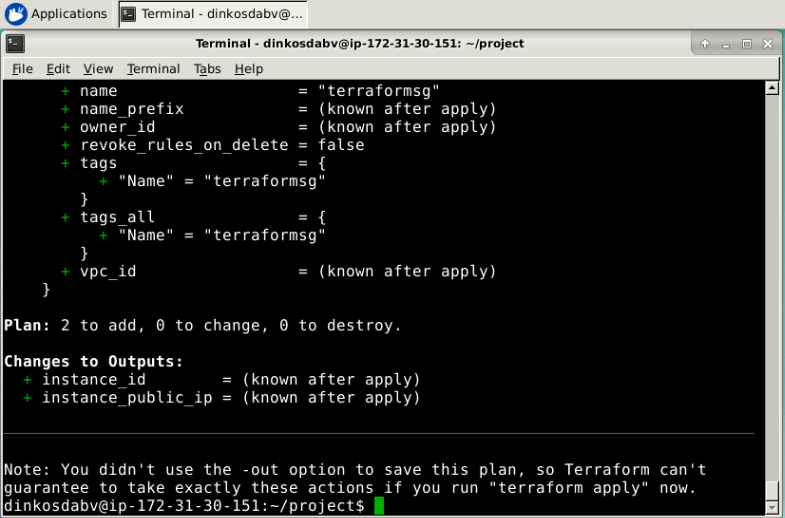
***}***

3.3 Run the next commands to create a new EC2 instance:

***terraform init***

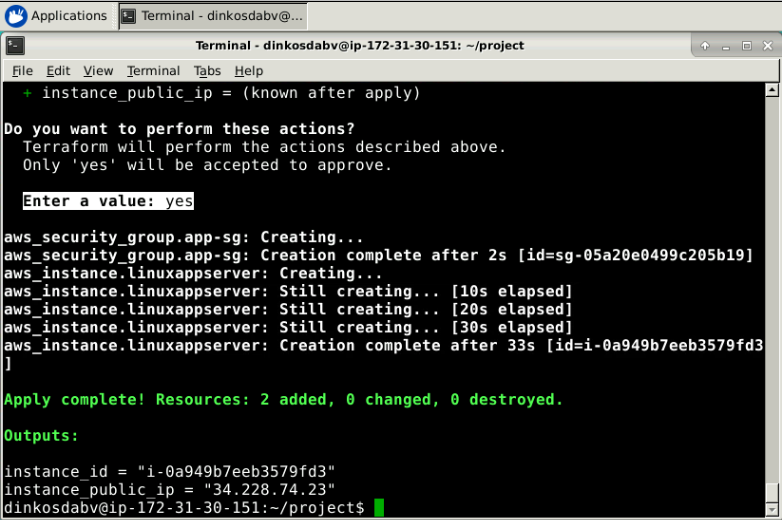


***terraform plan***

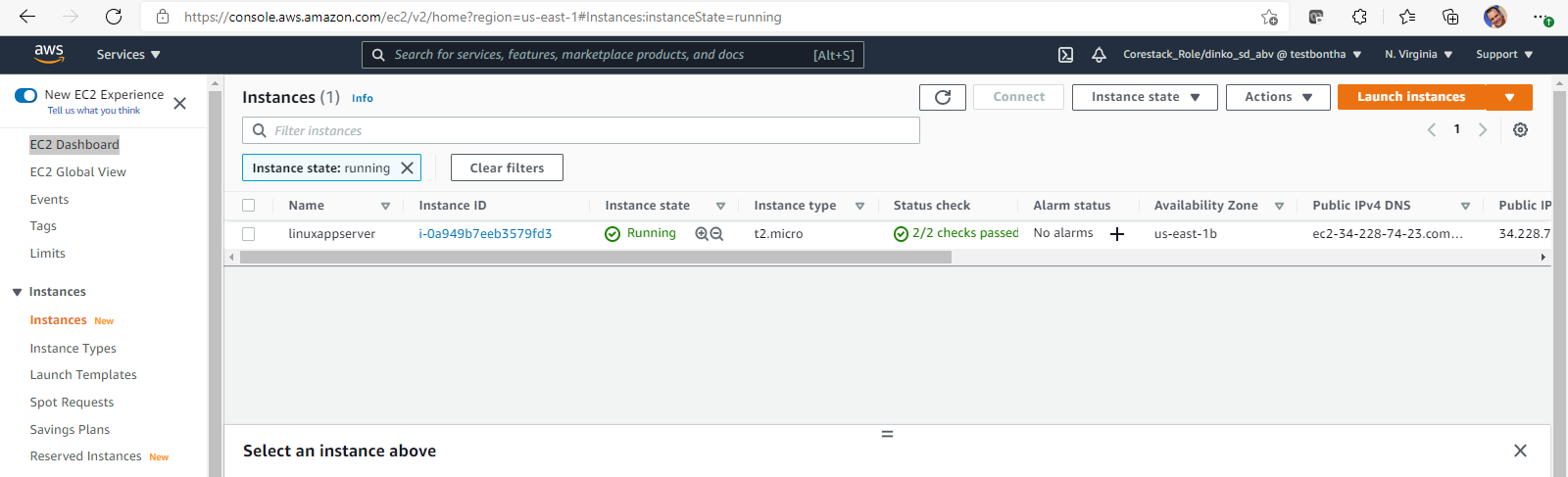


***terraform apply***

When prompted enter a “yes” value. The result is as follows:



Navigate to your AWS Management Console 🡪 All services 🡪 Compute 🡪 EC2 🡪 EC2 Dashboard to review your newly created instance.

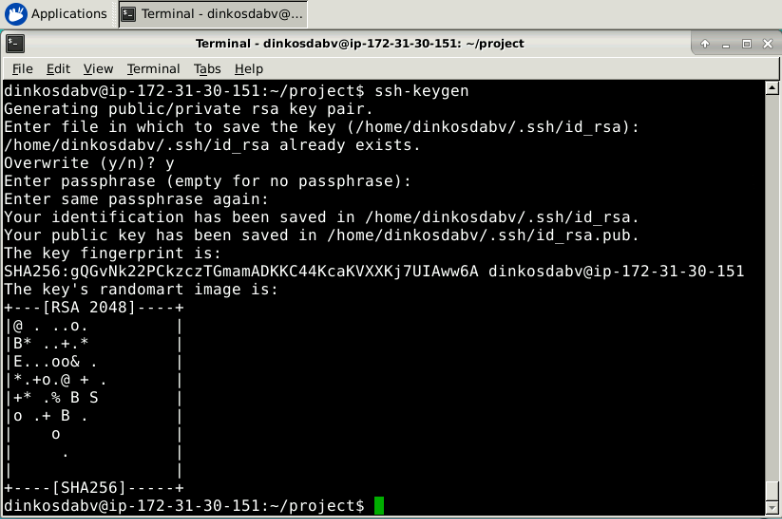


**Step 4. Establish connectivity to your AWS EC2 instance**

Before we proceed with Ansible execution, we want to make sure there is connectivity to our newly created EC2 instance. For this purpose, run the following command in your local system:

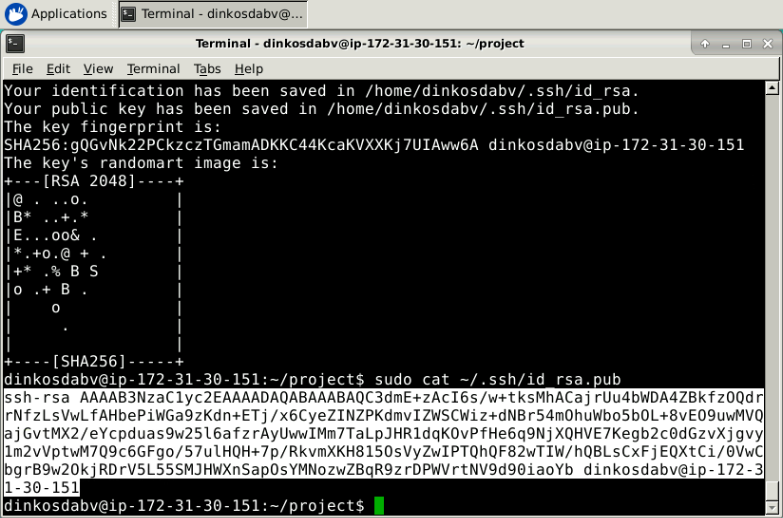
***ssh-keygen***

When prompted, push “Enter”:

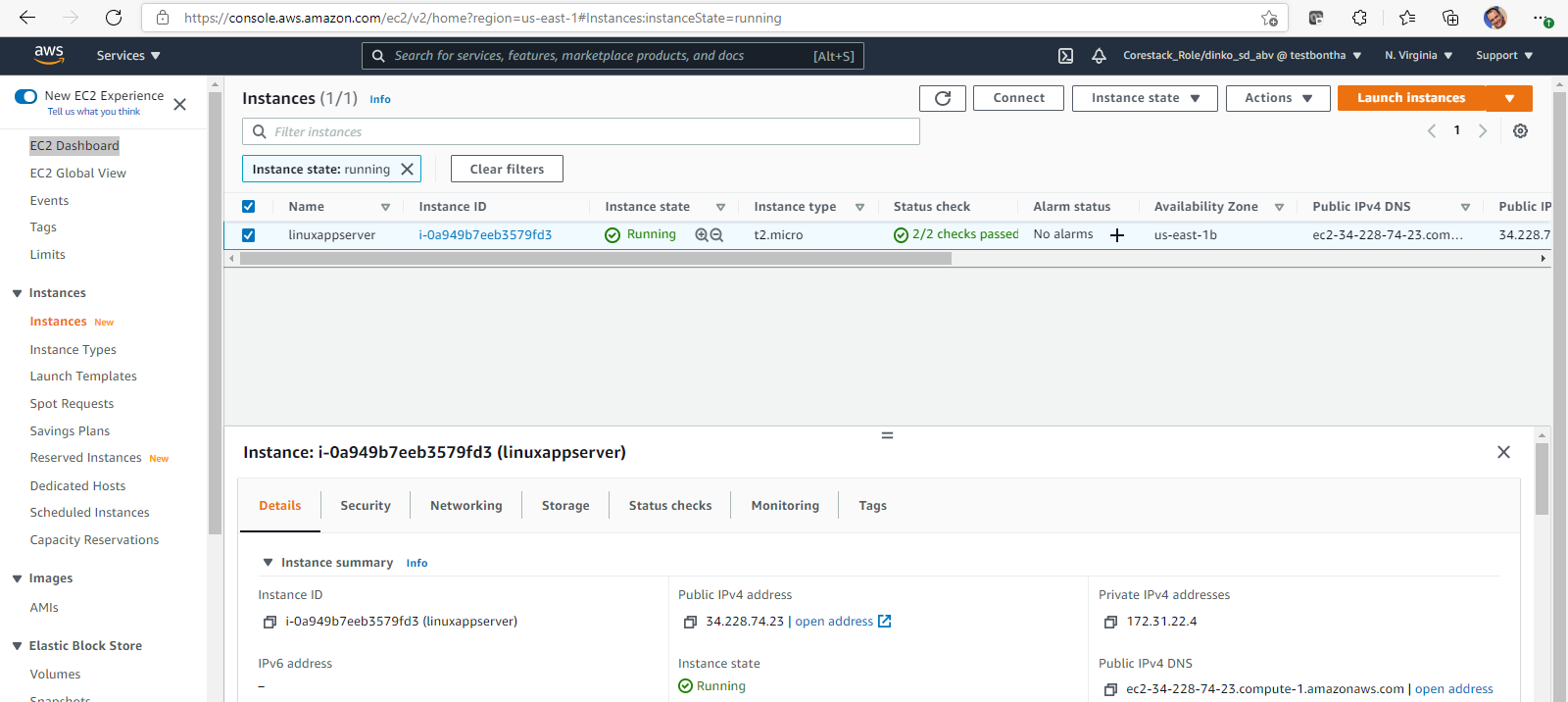


Execute the following command and copy the key:

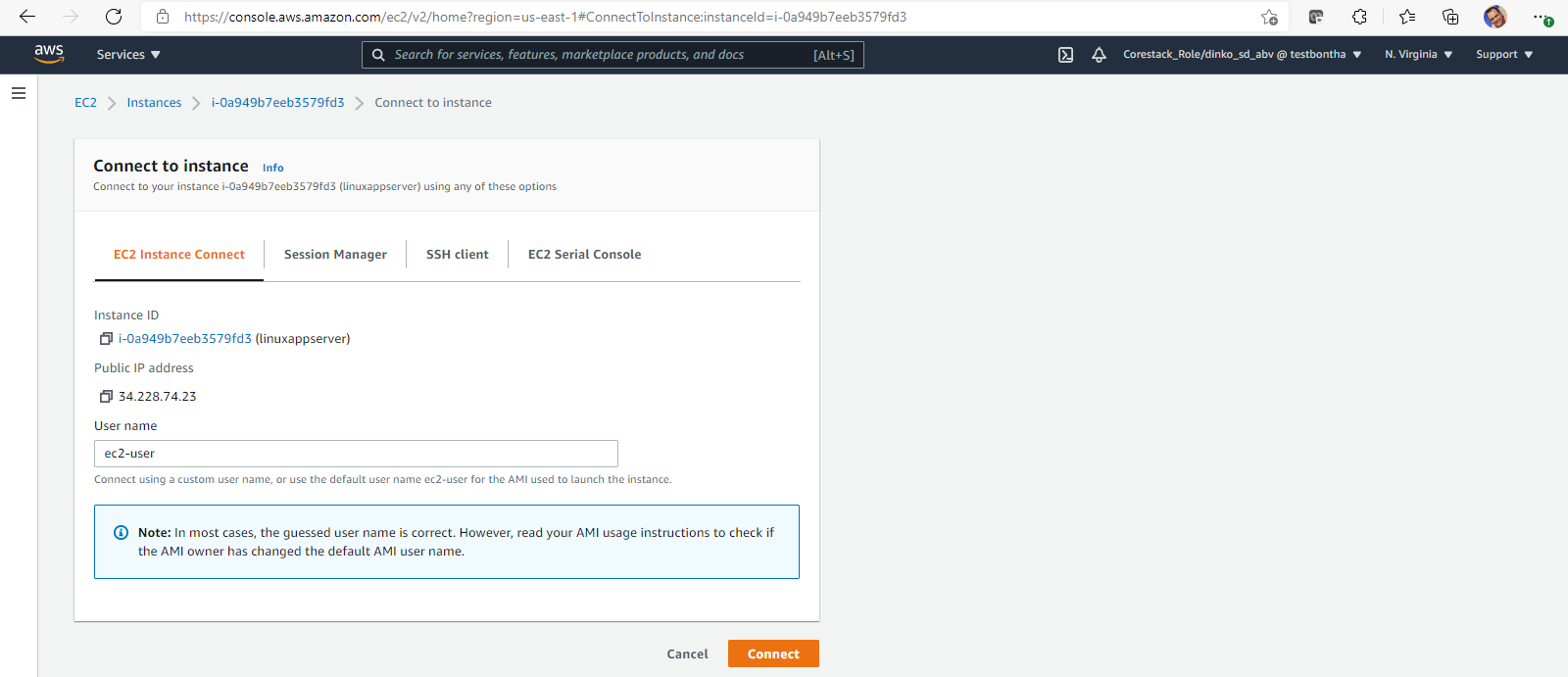
***sudo cat ~/.ssh/id\_rsa.pub***



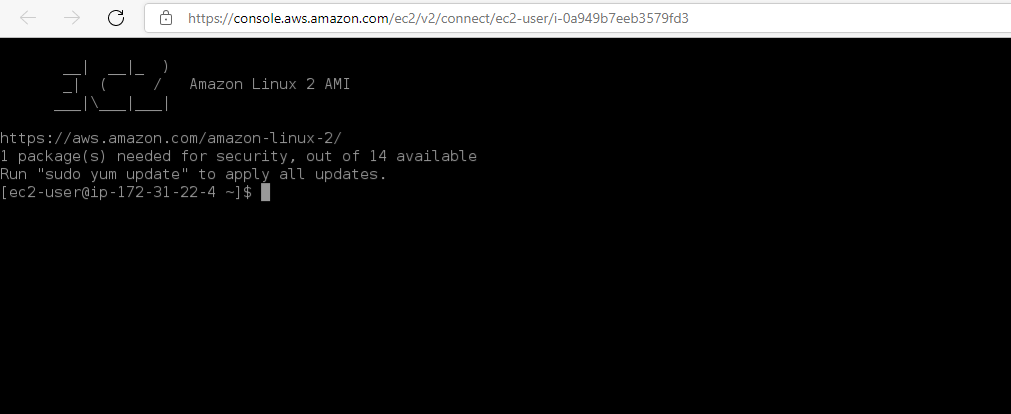
Ensure the EC2 instance allows connection from local system. For this purpose, go back to the AWS Console and connect to the instance:



On the next screen, provide a user (or use the default):



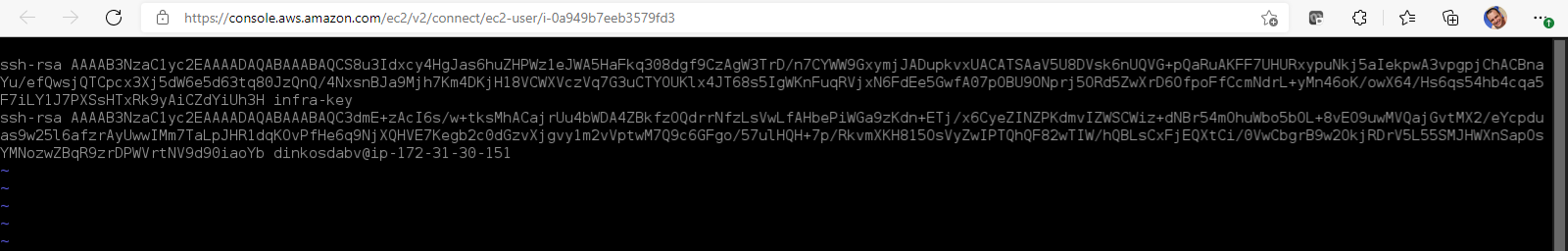
Click “Connect”. A new console tab will be loaded with your instance:



Run the following command:

***sudo vi ~/.ssh/authorized\_keys***

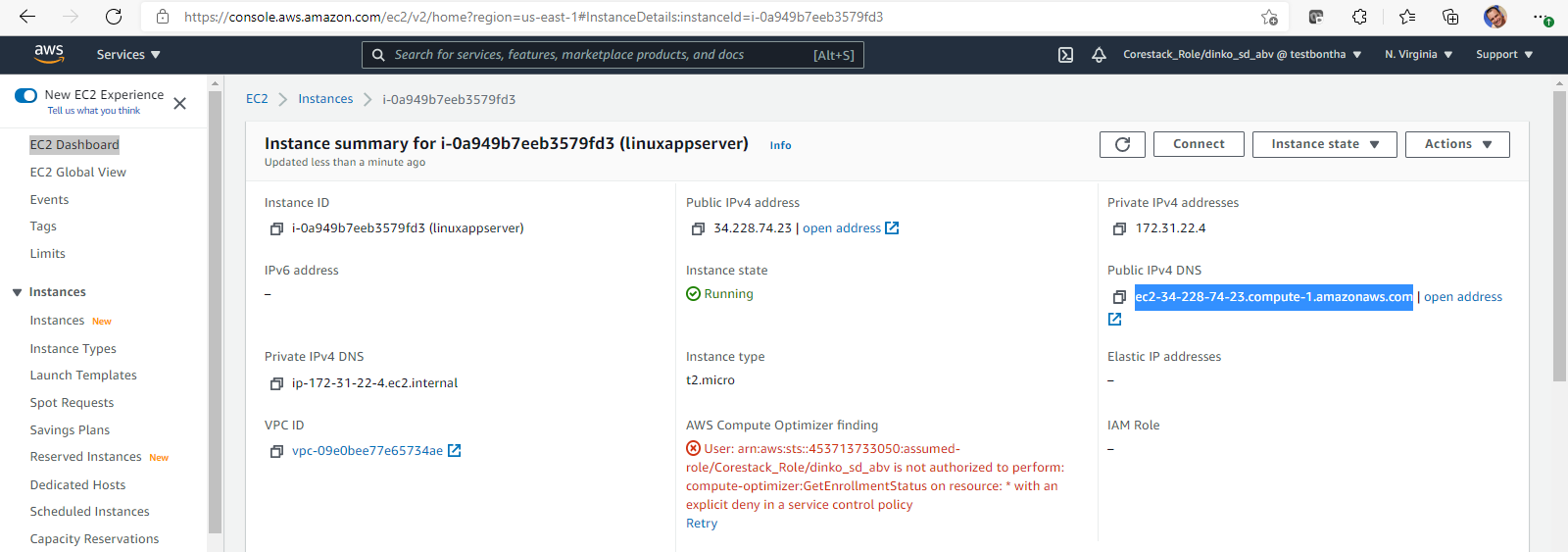
Copy the SSH key of your local system in this file (ESC + wq! to write and exit the file):



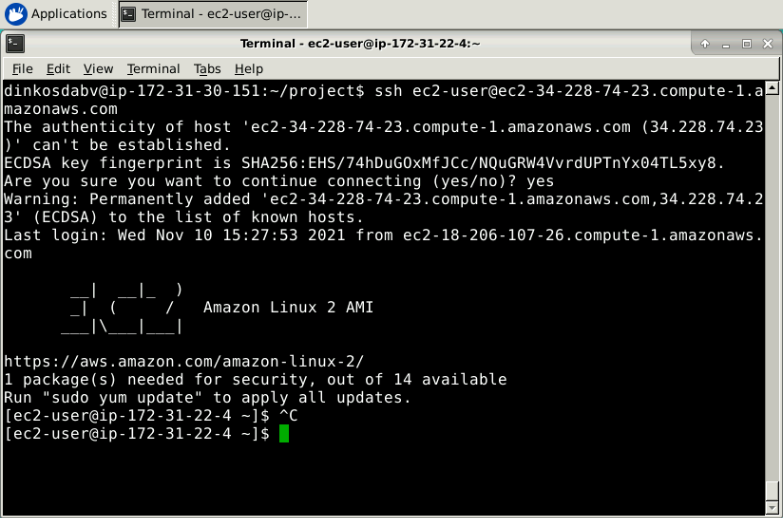
Go back to the console of your local system and run:

***ssh <EC2 user>@<EC2 public DNS>***

The command looks like this “ssh [ec2-user@ec2-34-228-74-23.compute-1.amazonaws.com](mailto:ec2-user@ec2-34-228-74-23.compute-1.amazonaws.com)”). When prompted, provide yes to permanently add the IP to the list of known hosts. You can copy the same from your instance’s summary:



Connectivity is established between local system and the newly created EC2 instance:



Next is to prepare the instance for the upcoming installation of Jenkins, Java and Python using Ansible. For this purpose, we want to execute the following commands to ensure we have the packages we need:

***sudo yum update***

When prompted enter “yes”.

***sudo amazon-linux-extras install epel –y***

This is a package required by Jenkins. Exit the instance:

***exit***

**Step 5. Install Jenkins, Java and Python using Ansible.**

For this step, we want to ensure that Ansible is installed on our local system. The required steps are in the attached file:



Next, we want to establish connectivity between Ansible controller and the EC2 instance:

***sudo vi /etc/ansible/hosts***

At the bottom of the file, insert the following line:

***[all]***

***<name of host> ansible\_host=<EC2 public DNS> ansible\_user=<your EC2 user> ansible\_ssh\_private\_key\_file=~/.ssh/id\_rsa ansible\_python\_interpreter=/usr/bin/python2***

Moderate the version of python interpreter according to your needs. An example of the command looks like the following:

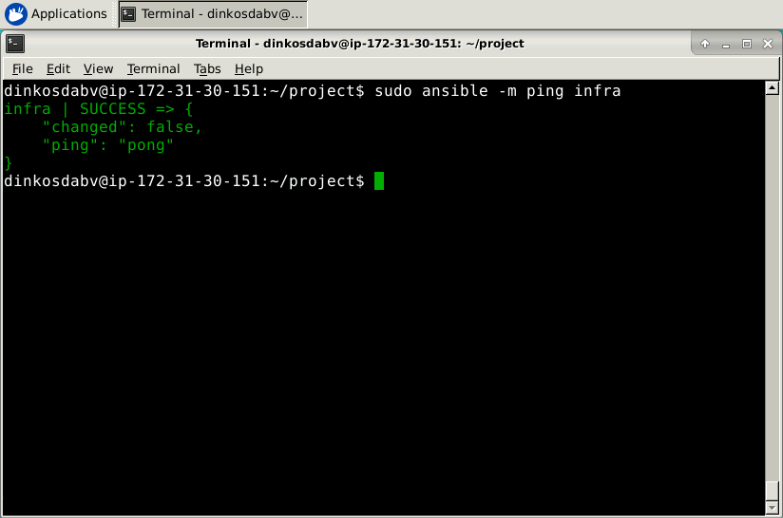
***[all]***

***infra ansible\_host=ec2-34-228-74-23.compute-1.amazonaws.com ansible\_user=ec2-user ansible\_ssh\_private\_key\_file=~/.ssh/id\_rsa ansible\_python\_interpreter=/usr/bin/python2***

Now make sure the connection is working using the following command:

***sudo ansible -m ping <name of host>***

When prompted provide “yes”. The end result is as displayed:



Finally, deploy Jenkins, Java and Python using Ansible. For this purpose, configure the Ansible “yml” file:

***vi <name of file>.yml***

Configure the following script:

***---***

***- hosts: <name of host>***

***remote\_user: <EC2 user>***

***gather\_facts: no***

***become: true***

***tasks:***

***- name: Install Java***

***yum:***

***name: java-1.8.0-openjdk-devel***

***state: present***

***update\_cache: yes***

***- name: Install Python***

***yum:***

***name: python2***

***state: present***

***update\_cache: yes***

***- name: Get Jenkins***

***get\_url:***

***url: http://pkg.jenkins-ci.org/redhat-stable/jenkins.repo***

***dest: /etc/yum.repos.d/jenkins.repo***

***- name: Get Jenkins Key***

***rpm\_key:***

***state: present***

***key: https://pkg.jenkins.io/redhat/jenkins.io.key***

***- name: Install Jenkins***

***yum:***

***name: jenkins***

***state: present***

***update\_cache: yes***

***- name: Start Jenkins***

***systemd:***

***name: jenkins***

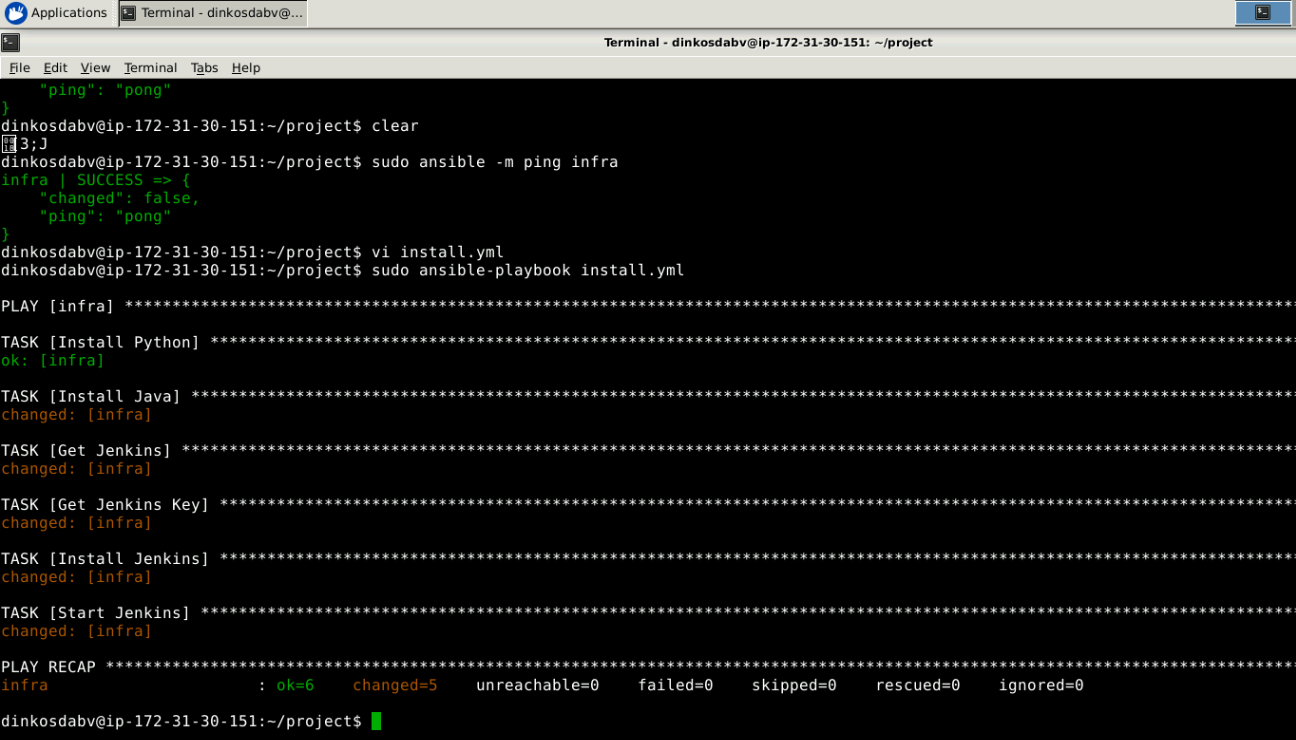
***state: started***

***enabled: true***

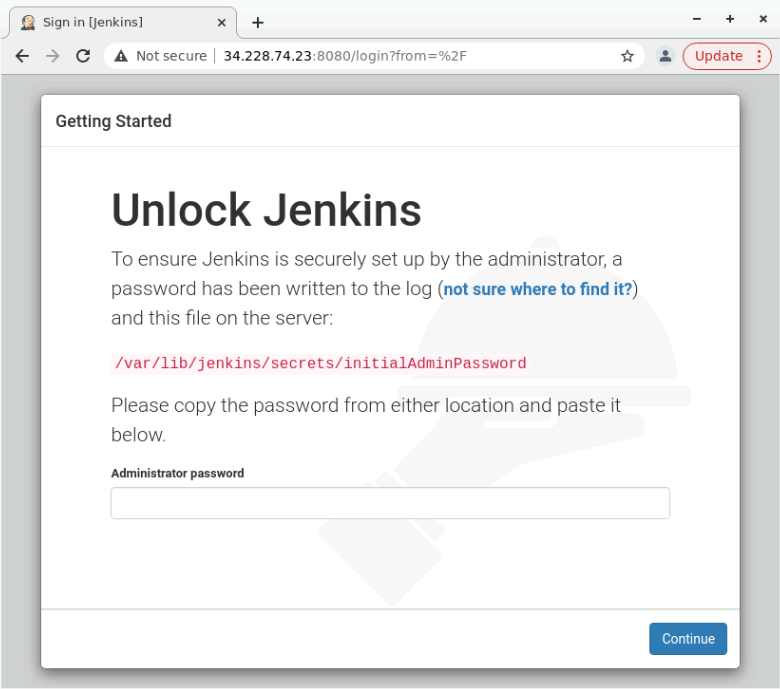
Save the file. Execute Ansible via the following command:

***sudo ansible-playbook <name of file>.yml***

The end result looks like this:

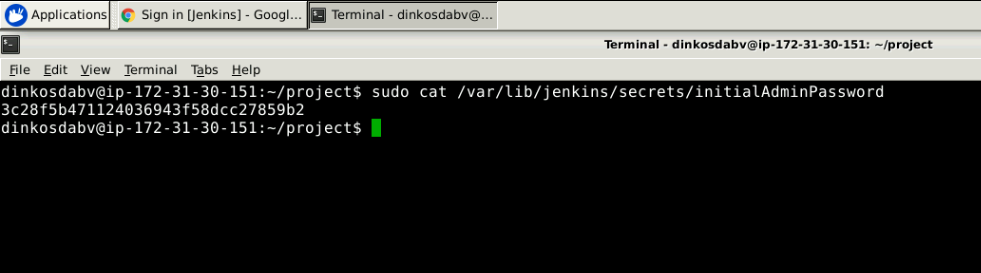


Finally, ensure Jenkins is correctly installed. For this purpose, copy the EC2 instance public IP address and enter it in a new browser tab (<EC2 instance public IP>:8080 e.g. <http://34.228.74.23:8080/>):



Get the Jenkins password via the following command:

***sudo cat /var/lib/jenkins/secrets/initialAdminPassword***



Copy the same to unlock Jenkins.