

Project assignment task description summary:

Through infrastructure automation my organization is using more of the DevOps methodology and it want to facilitate testing, deployment and use its benefits for disaster recovery.

A centralized server for Jenkins will be set up. Terraform will be used to build an EC2 Instance which should have basic infrastructure automation software installed.

Tools required:

Terraform, AWS account with security credentials, Keypair

Expected Deliverables:

Launch an EC2 instance using Terraform

Connect to the instance

Install Jenkins, Java and Python in the instance

Introduction	3
Terraform and Jenkins Project	3
Commands to install Terraform	3
Terraform configuration	7
Creating a .tf file	7
Main.tf Terraform configuration file	7
Terraform init	9
Terraform plan	10
Terraform apply	12
Connect to EC2 instance	13
Check installed versions	15
Specifically check Jenkins status	15
Accessing Jenkins	16
Getting Public IP	16
Jenkins, GUI first steps	17

Introduction

For this assignment, the given Ubuntu Lab Environment is the basis workspace. Furthermore AWS Webservices are used through given lab credentials.

Terraform and Jenkins Project

Commands to install Terraform

The following commands are entered to install Terraform and check the installed version

- 1)
fssl: Fail silently on (location and other) server errors
curl: Transfers data to or from a server
The Command adds the private key to trust the HashiCorp package when downloading it

Code:

```
curl -fsSL https://apt.releases.hashicorp.com/gpg | sudo apt-key add -
```

Terminal:

```
sebastian@ip-172-31-33-148:~$ curl -fsSL https://apt.releases.hashicorp.com/gpg | sudo apt-key add -
OK
sebastian@ip-172-31-33-148:~$
```

- 2)
Terraform package is installed

Code:

```
sudo apt install -y terraform
```

Terminal:

```
sebastian@ip-172-31-33-148:~$ sudo apt install -y terraform
Reading package lists... Done
Building dependency tree
Reading state information... Done
terraform is already the newest version (1.1.6).
0 upgraded, 0 newly installed, 0 to remove and 142 not upgraded.
sebastian@ip-172-31-33-148:~$
```

- 3)
Directory "ProjectTerraform" for Terraform is created on the desktop

Code:

```
mkdir ProjectTerraform
```

Terminal:

```
sebastian@ip-172-31-33-148:~$ cd Desktop
sebastian@ip-172-31-33-148:~/Desktop$ mkdir ProjectTerraform
sebastian@ip-172-31-33-148:~/Desktop$
```

4)
Login to AWS EC2 through Single-Sign-On URL

DevOps Practitioner - Terminal Only AWS DevOps in AWS V2 This Lab will get reset on 25th October 2023, 7:32 AM

Current Lab - AWS Certification - Dedicated Account

Access Information Lab Details Components Log Details Usage Details

AWS Web Console **AWS API Access**

AWS Web Console

Auth URL

<https://signin.aws.amazon.com/saml>

Session Expires in: 7h 18m 9s Refresh Link

1. Session Duration is for 8 Hours. Post the session duration, all the resources will be cleaned up automatically.
2. Auth URL, enables Single-Sign-On, so the URL, will vary for each session and the same URL, will not work next time. Refresh the Access Details page if the session duration has expired to get a new Auth URL.
3. Auth URL, per session will expire if AWS Console is not accessed within 15 minutes from the time it got generated. Please use 'Refresh Link' option to get the same link working.
4. 'Refresh Link' option will not reset the session timer and it will continue to be ticking.
5. Post the session duration of 8 Hours, it will take 30 minutes to cleanup the resources created in the previous session. Until this time, a new session cannot be created.

Powered by **CORESTACK**

AWS Certification - Dedicated Account

Category: Cloud Computing
Start Date: 2023-10-10 17:34
End Date: 2023-10-19 00:07
Code: SLAWS

Amazon Web Services (AWS) offers a suite of cloud-computing services that make up an on-demand computing platform. AWS has more than 70 services, spanning a wide range, including compute, storage, networking, databases, analytics, application services, deployment, management, mobile, developer tools and tools for the Internet of Things.

TERMINATE LAB ACCESS

Terms & Conditions

5)
AWS EC2 console is opened with the Chrome Browser, EC2 is selected (in Services if not in recent)

Console Home | us-east-1

us-east-1.console.aws.amazon.com/console/home?region=us-east-1#

Update

Services

Reset to default layout Add widgets

Recently visited

- EC2
- Billing
- S3
- Key Management Service

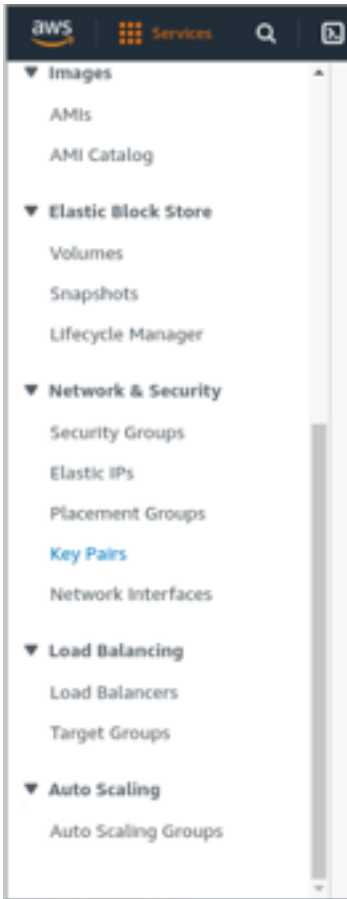
View all services

Welcome to AWS

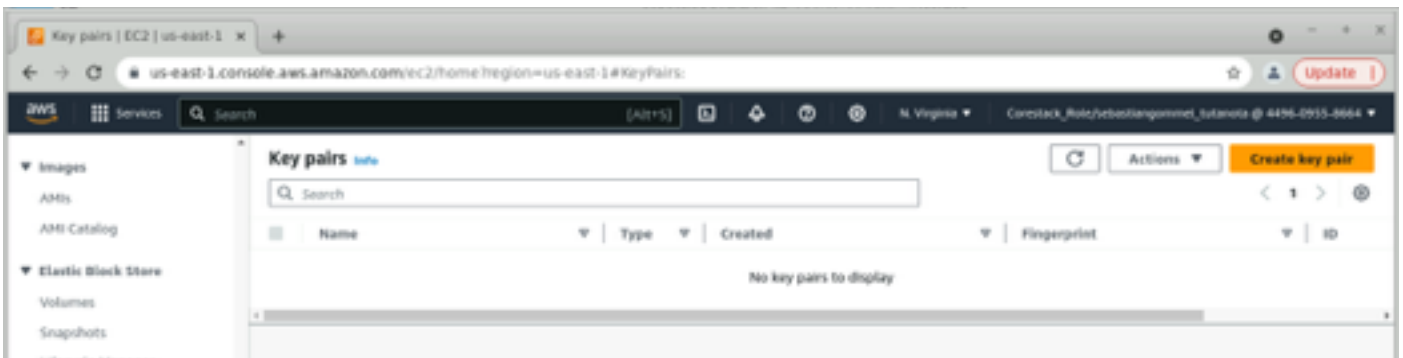
CloudShell Feedback Privacy Terms Cookie preferences

© 2023, Amazon Web Services, Inc. or its affiliates.

6)
Key Pairs is selected here in Network and Security

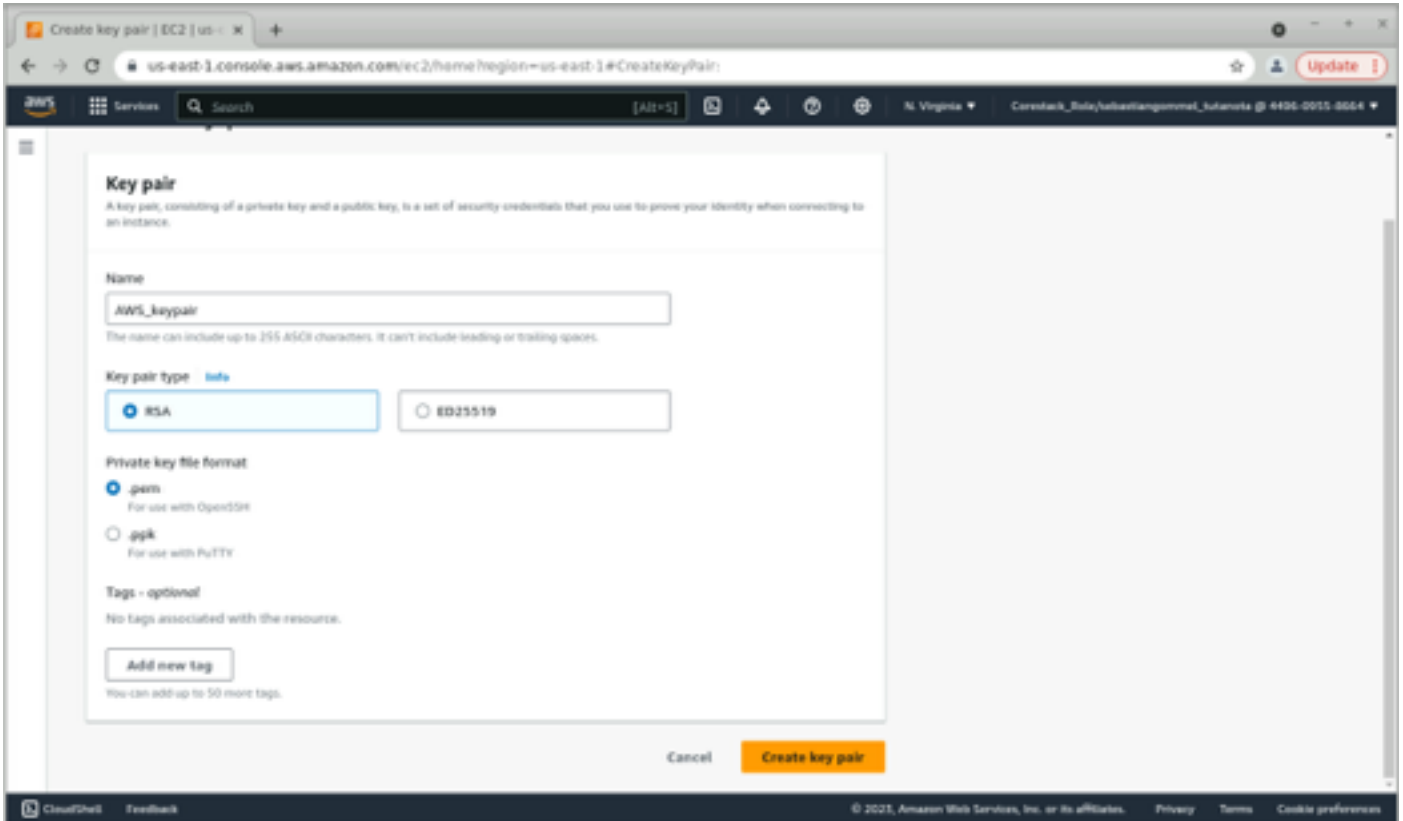


7)
Create key pair is selected

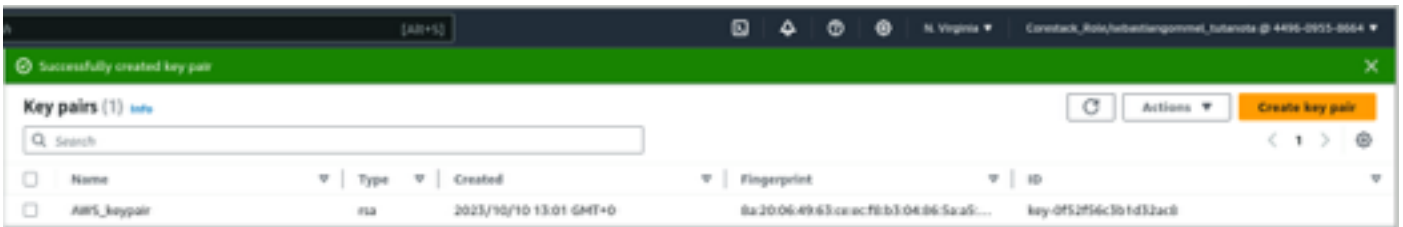


8)

Key pair configuration is set as in picture: (name = AWS_keypair..)



The Key pair has been created and is in downloads



9)

Key pair is moved to the project directory "ProjectTerraform"

Code:

```
cd /home/sebastiangommel/Downloads
```

```
ls
```

```
mv AWS_keypair.pem /home/sebastiangommel/Desktop/ProjectTerraform
```

Terminal:

```
sebastiangommel@ip-172-31-33-148:~/Desktop$ cd /home/sebastiangommel/Downloads
sebastiangommel@ip-172-31-33-148:~/Downloads$ ls
AWS_keypair.pem
sebastiangommel@ip-172-31-33-148:~/Downloads$ mv AWS_keypair.pem /home/sebastiangommel/Desktop/ProjectTerraform
sebastiangommel@ip-172-31-33-148:~/Downloads$
```

Terraform configuration

Creating a .tf file

- 1)
Change directory to project directory "ProjectTerraform"

Code:

```
cd /home/sebastiangommel/Desktop/ProjectTerraform/
```

Terminal:

```
sebastiangommel@ip-172-31-33-148:~/Downloads$ cd /home/sebastiangommel/Desktop/ProjectTerraform/
sebastiangommel@ip-172-31-33-148:~/Desktop/ProjectTerraform$
```

Main.tf Terraform configuration file

- 2)
Vim editor is used to create the main.tf file which includes the provider and region as well as the credentials for the EC2 instance.

Code:

```
vim main.tf
```

Terminal:

```
sebastiangommel@ip-172-31-33-148:~/Desktop/ProjectTerraform$ vim main.tf
sebastiangommel@ip-172-31-33-148:~/Desktop/ProjectTerraform$
```

Furthermore the security group settings, as well as the type of resource to be executed and the commands to prepare the automation infrastructure environment for installation of Java, Python and Jenkins are executed.

- 3)
The following command rewrites the Terraform configuration file to a canonical format and style, which also improves the appearance of the next page with the code.

Code:

```
terraform fmt
```

Terminal:

```
sebastiangommel@ip-172-31-22-183:~/Desktop/ProjectTerraform$ terraform fmt
```

main.tf file

Code:

```
# Configuring the cloud provider with lab provided credentials
provider "aws" {
  region    = "us-east-1"
  access_key = "ASIAWRLWRNKECLZGHYIK"
  secret_key = "t/sHjJDHlrGYPj2VYBJBSf+IAE3unURU3wXWnD5J"
  token     = "FwoGZXlvYXdzEN7////////wEaDGts0LrZlozkSJpMkSK9AfE6qr612k6bydtmGV/
vqlxBiLeVg1nYnvY02MSwyMbGbldXSDgn8feqws5ak+oCFD8yy95CHC9Mw1sK4oR7s4WpexXnapP/
Qu+g+n1KuGWwLXHd80WRWNe9VWCwKAiBi9gO858xIS+LkmYbxsG2F9jOaZuK7fQPfTQr7cwK/
sAWmSmNxlI0HXssqu0s2MzQeKQF1QVRXyWt6jYX+BbBhmXWM6Pb3KL2z3Nylq6KHQ9pcLdC5zo4pgl/
Sxx1yid/JSpBjlt/MzfQfGX9yEXkxWXj1HTYRCTlrzOJOy+/hT/fErhrWF2tPBErNKVc9H7BF/"
}

# Security group settings
variable "ingress-rules" {
  type    = list(number)
  default = [22, 8080]
}

resource "aws_security_group" "web_traffic" {
  name        = "Allow web traffic"
  description = "SSH/Jenkins inbound, everything outbound"
  dynamic "ingress" {
    iterator = port
    for_each = var.ingress-rules
    content {
      from_port = port.value
      to_port   = port.value
      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"]
  }
}

# Type of resource to be executed
resource "aws_instance" "ec2" {
  ami           = "ami-04505e74c0741db8d"
  instance_type = "t3.micro"
  key_name      = "AWS_keypair"
  vpc_security_group_ids = [aws_security_group.web_traffic.id]

  # Remotely execute commands to install Java, Python, Jenkins
  provisioner "remote-exec" {
    inline = [
      "sudo apt update && upgrade",
      "sudo apt install -y python3.8",
    ]
  }
}
```



```

    "curl -fsSL https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key | sudo tee /usr/share/keyrings/
jenkins-keyring.asc > /dev/null",
    "echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] https://pkg.jenkins.io/debian-stable
binary/ | sudo tee /etc/apt/sources.list.d/jenkins.list > /dev/null",
    "sudo apt-get install fontconfig java-17-openjdk",
    "sudo systemctl daemon-reload",
    "sudo apt-get install jenkins"

]
}

```

```

# Type of connection to be established
connection {
  type      = "ssh"
  user      = "ubuntu"
  private_key = file("${path.module}/AWS_keypair.pem")
  host      = self.public_ip
}
}

```

Terraform init

4)
Execute terraform init command

Code:
terraform init

Terminal

```

sebastian@ip-172-31-33-148:~/Desktop/ProjectTerraform$ terraform init
Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v5.20.0...
- Installed hashicorp/aws v5.20.0 (signed by HashiCorp)

Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
sebastian@ip-172-31-33-148:~/Desktop/ProjectTerraform$

```

Terraform plan

5) Execute terraform plan command

Code:
terraform plan

Terminal

```
chastiangomez@ip-172-31-33-148:~$ ssh -t -C -o StrictHostKeyChecking=no ubuntu@ec2-54-90-64-10.us-east-2.compute.amazonaws.com:22 terraform plan
```

```
Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
```

- + create

```
Terraform will perform the following actions:
```

```
# aws_instance.ec2 will be created
+ resource "aws_instance" "ec2" {
    + ami                     = "ami-04505e74c074fdbbd"
    + arn                    = (known after apply)
    + associate_public_ip_address = (known after apply)
    + availability_zone       = (known after apply)
    + cpu_core_count         = (known after apply)
    + cpu_threads_per_core   = (known after apply)
    + disable_api_stop       = (known after apply)
    + disable_api_termination = (known after apply)
    + ebs_optimized          = (known after apply)
    + get_password_data      = false
    + host_id                = (known after apply)
    + host_resource_group_arn = (known after apply)
    + iam_instance_profile   = (known after apply)
    + id                     = (known after apply)
    + instance_initiated_shutdown_behavior = (known after apply)
    + instance_lifecycle     = (known after apply)
    + instance_state         = (known after apply)
    + instance_type          = "t3.micro"
    + ipv6_address_count     = (known after apply)
    + ipv6_addresses        = (known after apply)
    + key_name               = "RMS_keypair"
    + monitoring             = (known after apply)
    + outpost_arn            = (known after apply)
    + password_data         = (known after apply)
    + placement_group        = (known after apply)
    + placement_partition_number = (known after apply)
    + primary_network_interface_id = (known after apply)
    + private_dns            = (known after apply)
    + private_ip             = (known after apply)
```

```

+ public_dns              = (known after apply)
+ public_ip              = (known after apply)
+ secondary_private_ips  = (known after apply)
+ security_groups         = (known after apply)
+ source_dest_check       = true
+ spot_instance_request_id = (known after apply)
+ subnet_id              = (known after apply)
+ tags_all               = (known after apply)
+ tenancy                = (known after apply)
+ user_data              = (known after apply)
+ user_data_base64       = (known after apply)
+ user_data_replace_on_change = false
+ vpc_security_group_ids = (known after apply)

+ capacity_reservation_specification {
+   capacity_reservation_preference = (known after apply)

+   capacity_reservation_target {
+     capacity_reservation_id      = (known after apply)
+     capacity_reservation_resource_group_arn = (known after apply)
+   }
+ }

+ cpu_options {
+   amd_sev_snp = (known after apply)
+   core_count  = (known after apply)
+   threads_per_core = (known after apply)
+ }

+ ebs_block_device {
+   delete_on_termination = (known after apply)
+   device_name            = (known after apply)
+   encrypted              = (known after apply)
+   iops                   = (known after apply)
+   kms_key_id             = (known after apply)
+   snapshot_id            = (known after apply)
+   tags                   = (known after apply)
+   throughput             = (known after apply)
+ }

```

```

    + volume_id = (known after apply)
    + volume_size = (known after apply)
    + volume_type = (known after apply)
  }

  + enclave_options {
    + enabled = (known after apply)
  }

  + ephemeral_block_device {
    + device_name = (known after apply)
    + no_device = (known after apply)
    + virtual_name = (known after apply)
  }

  + instance_market_options {
    + market_type = (known after apply)

    + spot_options {
      + instance_interruption_behavior = (known after apply)
      + max_price = (known after apply)
      + spot_instance_type = (known after apply)
      + valid_until = (known after apply)
    }
  }
}

```

```

+ maintenance_options {
+   auto_recovery = (known after apply)
+ }

+ metadata_options {
+   http_endpoint      = (known after apply)
+   http_protocol_ipv6 = (known after apply)
+   http_put_response_hop_limit = (known after apply)
+   http_tokens         = (known after apply)
+   instance_metadata_tags = (known after apply)
+ }

```

```

+ network_interface {
+   delete_on_termination = (known after apply)
+   device_index           = (known after apply)
+   network_card_index     = (known after apply)
+   network_interface_id   = (known after apply)
+ }

+ private_dns_name_options {
+   enable_resource_name_dns_a_record   = (known after apply)
+   enable_resource_name_dns_aaaa_record = (known after apply)
+   hostname_type                       = (known after apply)
+ }

+ root_block_device {
+   delete_on_termination = (known after apply)
+   device_name            = (known after apply)
+   encrypted              = (known after apply)
+   iops                   = (known after apply)
+   kms_key_id             = (known after apply)
+   tags                   = (known after apply)
+   throughput             = (known after apply)
+   volume_id              = (known after apply)
+   volume_size            = (known after apply)
+   volume_type            = (known after apply)
+ }
}

# aws_security_group.web_traffic will be created
+ resource "aws_security_group" "web_traffic" {
+   arn          = (known after apply)
+   description   = "SSH/Jenkins inbound, everything outbound"
+   egress        = [
+     {
+       cidr_blocks = [
+         "0.0.0.0/0",
+       ]
+       description = ""
+       from_port   = 0

```

```

+   ipv6_cidr_blocks = []
+   prefix_list_ids  = []
+   protocol         = "-1"
+   security_groups  = []
+   self             = false
+   to_port          = 0
+     },
+   ],
+   id          = (known after apply)
+   ingress     = [
+     {
+       cidr_blocks = [
+         "0.0.0.0/0",
+       ]
+       description = ""
+       from_port   = 22
+       ipv6_cidr_blocks = []
+       prefix_list_ids = []
+       protocol     = "tcp"
+       security_groups = []
+       self         = false
+       to_port      = 22
+     },
+     {
+       cidr_blocks = [
+         "0.0.0.0/0",
+       ]
+       description = ""
+       from_port   = 8080
+       ipv6_cidr_blocks = []
+       prefix_list_ids = []
+       protocol     = "tcp"
+       security_groups = []
+       self         = false
+       to_port      = 8080
+     },
+   ],
+   name          = "Allow web traffic"
+   name_prefix   = (known after apply)
+   owner_id      = (known after apply)
+   revoke_rules_on_delete = false
+   tags_all      = (known after apply)
+   vpc_id        = (known after apply)
+ }

```

Plan: 2 to add, 0 to change, 0 to destroy.

Note: You didn't use the `-out` option to save this plan, so Terraform can't guarantee to take exactly these actions if you run `"terraform apply"` now.

sebastian@ip-172-21-33-148:~/Desktop/ProjectTerraform\$

Terraform apply

6)

Execute terraform apply command

Yes has to typed in after it is prompted to confirm that the actions will be performed.

Code:

terraform apply

```
    + description      = ""
    + from_port        = 22
    + ipv6_cidr_blocks = []
    + prefix_list_ids  = []
    + protocol         = "tcp"
    + security_groups  = []
    + self              = false
    + to_port          = 22
  },
  + {
    + cidr_blocks      = [
      + "0.0.0.0/0",
    ]
    + description      = ""
    + from_port        = 8080
    + ipv6_cidr_blocks = []
    + prefix_list_ids  = []
    + protocol         = "tcp"
    + security_groups  = []
    + self              = false
    + to_port          = 8080
  },
  + name              = "Allow web traffic"
  + name_prefix       = (known after apply)
  + owner_id          = (known after apply)
  + revoke_rules_on_delete = false
  + tags_all          = (known after apply)
  + vpc_id            = (known after apply)
}

Plan: 2 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

Enter a value: yes
```

Last Output of Apply Command:

```
aws_instance.ec2 (remote-exec): (Reading database ... 80%
aws_instance.ec2 (remote-exec): (Reading database ... 85%
aws_instance.ec2 (remote-exec): (Reading database ... 90%
aws_instance.ec2 (remote-exec): (Reading database ... 95%
aws_instance.ec2 (remote-exec): (Reading database ... 100%
aws_instance.ec2 (remote-exec): (Reading database ... 63895 files and directories currently installed.)
aws_instance.ec2 (remote-exec): Preparing to unpack .../libpython3.8-3.8.10-0ubuntu1-20.04.8_amd64.deb ...
Progress: 13% [#####]
aws_instance.ec2 (remote-exec): [#####]Unpacking libpython3.8:amd64 (3.8.10-0ubuntu1-20.04.8) over (3.8.10-0ubuntu1-20.04.1) ...
Progress: 14% [#####]
aws_instance.ec2 (remote-exec): [#####]Preparing to unpack .../python3.8-3.8.10-0ubuntu1-20.04.8_amd64.deb ...
Progress: 14% [#####]
aws_instance.ec2 (remote-exec): [#####]Unpacking python3.8 (3.8.10-0ubuntu1-20.04.8) over (3.8.10-0ubuntu1-20.04.1) ...
Progress: 19% [#####]
aws_instance.ec2 (remote-exec): [#####]Preparing to unpack .../libpython3.8-stdlib:amd64 (3.8.10-0ubuntu1-20.04.8) over (3.8.10-0ubuntu1-20.04.1) ...
Progress: 24% [#####]
aws_instance.ec2 (remote-exec): [#####]Unpacking libpython3.8-stdlib:amd64 (3.8.10-0ubuntu1-20.04.8) over (3.8.10-0ubuntu1-20.04.1) ...
Progress: 29% [#####]
aws_instance.ec2 (remote-exec): [#####]Preparing to unpack .../python3.8-minimal-3.8.10-0ubuntu1-20.04.8_amd64.deb ...
Progress: 33% [#####]
aws_instance.ec2 (remote-exec): [#####]Unpacking python3.8-minimal (3.8.10-0ubuntu1-20.04.8) over (3.8.10-0ubuntu1-20.04.1) ...
Progress: 38% [#####]
aws_instance.ec2 (remote-exec): [#####]Preparing to unpack .../libpython3.8-minimal-3.8.10-0ubuntu1-20.04.8_amd64.deb ...
Progress: 43% [#####]
aws_instance.ec2 (remote-exec): [#####]Unpacking libpython3.8-minimal:amd64 (3.8.10-0ubuntu1-20.04.8) over (3.8.10-0ubuntu1-20.04.1) ...
Progress: 48% [#####]
aws_instance.ec2 (remote-exec): [#####]
aws_instance.ec2 (remote-exec): [#####]Setting up libpython3.8-minimal:amd64 (3.8.10-0ubuntu1-20.04.8) ...
Progress: 57% [#####]
aws_instance.ec2 (remote-exec): [#####]Setting up python3.8-minimal (3.8.10-0ubuntu1-20.04.8) ...
Progress: 62% [#####]
aws_instance.ec2 (remote-exec): [#####]
aws_instance.ec2 (remote-exec): [#####]Setting up libpython3.8-stdlib:amd64 (3.8.10-0ubuntu1-20.04.8) ...
Progress: 67% [#####]
aws_instance.ec2 (remote-exec): [#####]Setting up python3.8 (3.8.10-0ubuntu1-20.04.8) ...
Progress: 76% [#####]
aws_instance.ec2 (remote-exec): [#####]
aws_instance.ec2 (remote-exec): [#####]Setting up libpython3.8:amd64 (3.8.10-0ubuntu1-20.04.8) ...
Progress: 81% [#####]
aws_instance.ec2 (remote-exec): [#####]Processing triggers for libc-bin (2.31-0ubuntu9.2) ...
Progress: 86% [#####]
aws_instance.ec2 (remote-exec): [#####]Processing triggers for man-db (2.9.1-1) ...
Progress: 95% [#####]
aws_instance.ec2 (remote-exec): [#####]Processing triggers for mime-support (3.0.4ubuntu1) ...
Progress: 95% [#####]

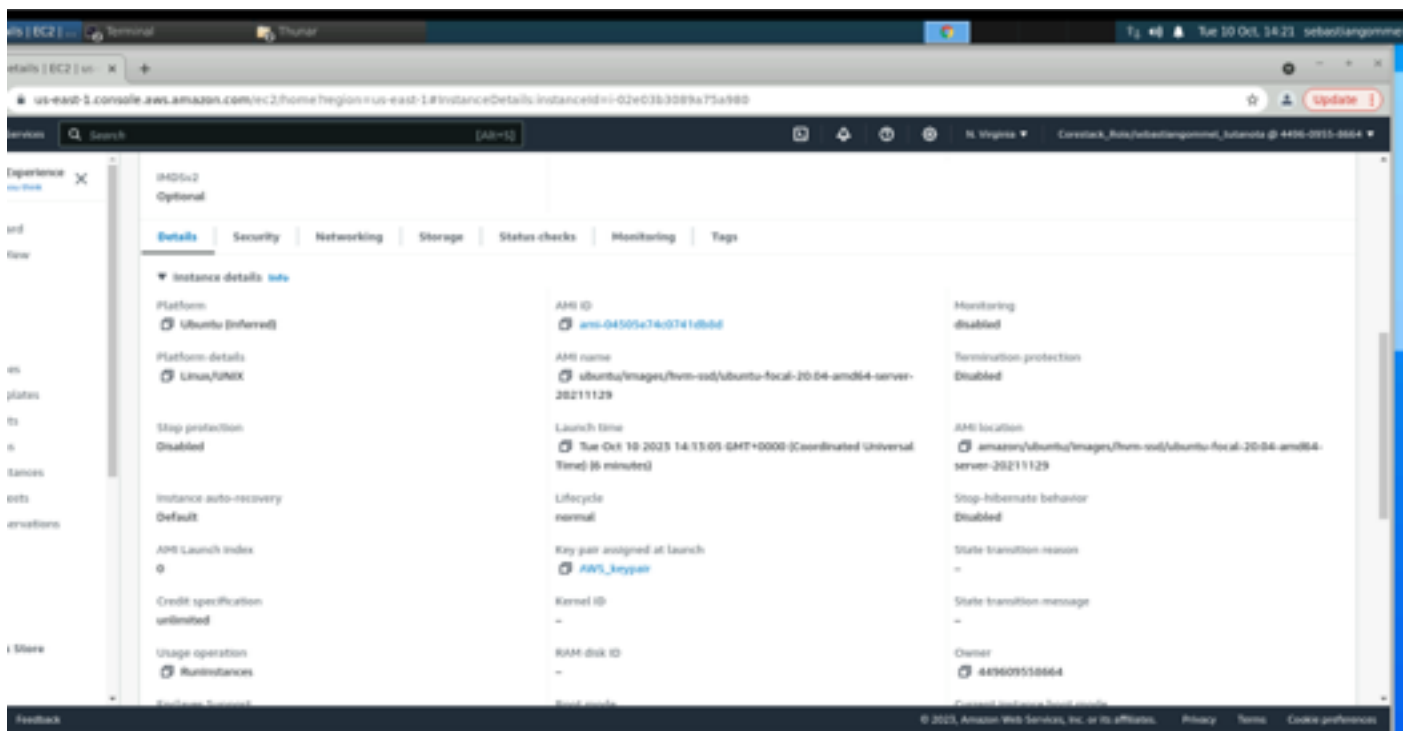
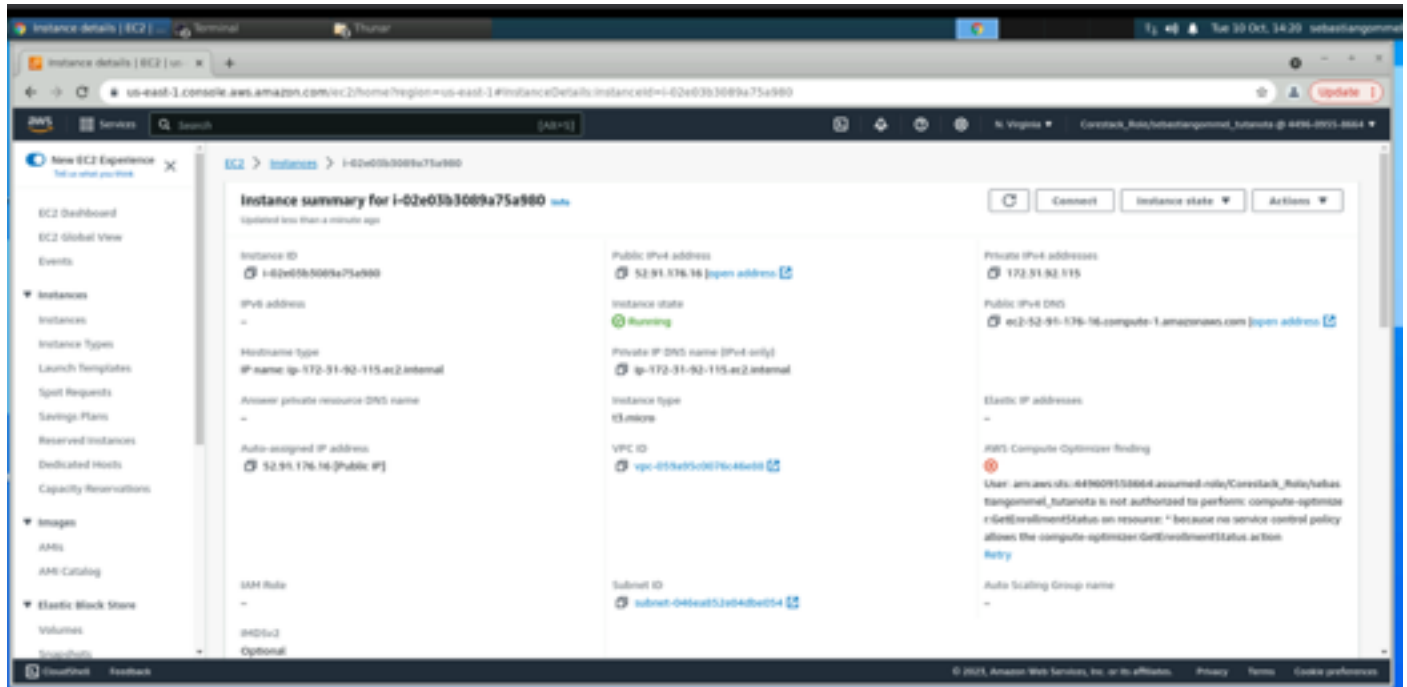
aws_instance.ec2: Still creating... [50s elapsed]
aws_instance.ec2: Creation complete after 55s [id=i-02e03b3809a75a980]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
sebastiangommel@ip-172-31-33-146:~/Desktop/ProjectTerraform$
```

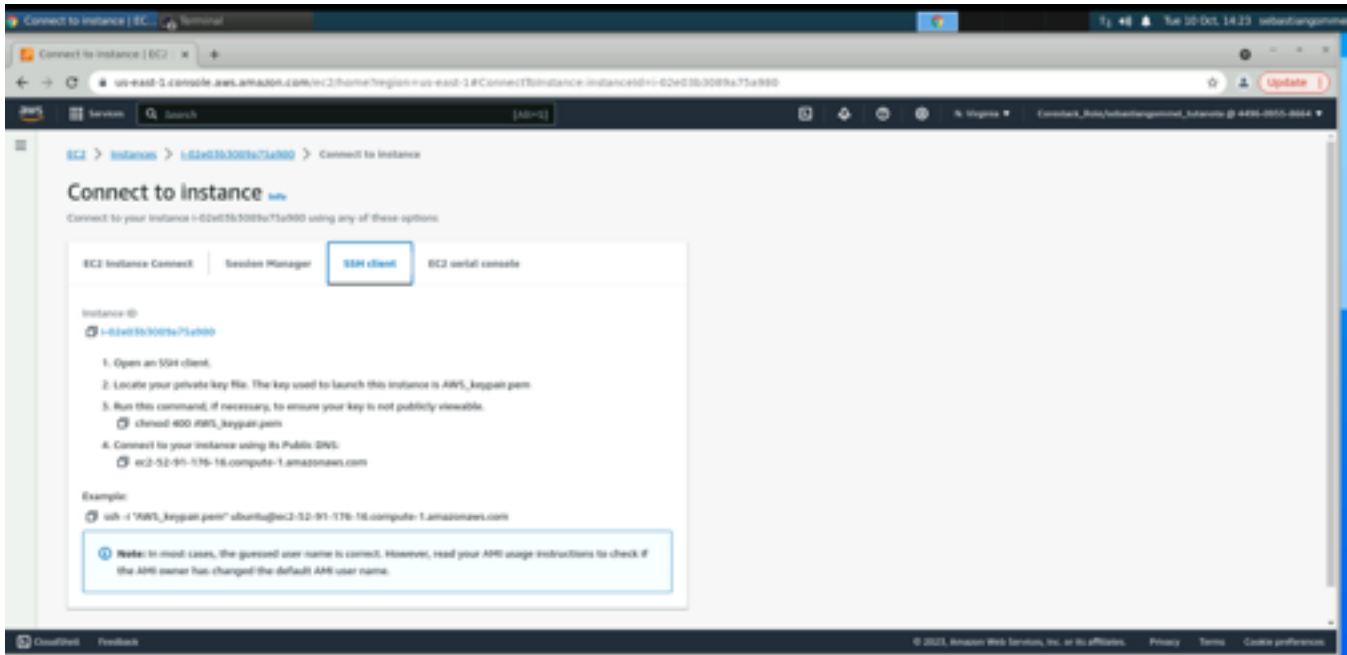
Connect to EC2 instance

7)
Connecting to EC2 instance

Get information from AWS, by going to AWS GUI to instances and clicking on the instance ID.
Here Connect is clicked on the top right of the AWS GUI



- 8)
Click on SSH client to copy the relevant SSH information (after having clicked connect):



- 9)
Back to the terminal, the permission for the pem file is changed here first before running the ssh command, as indicated in the step 3 of the AWS console

Code:

chmod 400 AWS_keypair.pem

Terminal:

```
sebastian@sebastian-laptop:~/Desktop/ProjectTerraform$ chmod 400 AWS_keypair.pem
```

- 10)
The next step is to connect to the EC2 instance with the ssh information copied from AWS GUI

Code:

ssh -i "AWS_keypair.pem" ubuntu@ec2-52-91-176-16.compute-1.amazonaws.com

Terminal:

```
sebastian@sebastian-laptop:~/Desktop/ProjectTerraform$ ssh -i "AWS_keypair.pem" ubuntu@ec2-52-91-176-16.compute-1.amazonaws.com
The authenticity of host 'ec2-52-91-176-16.compute-1.amazonaws.com (52.91.176.16)' can't be established.
ECDSA key fingerprint is SHA256:G3PF8gQ/nV8BAfP8XdyM3o/fVnLMSj2pCzxbw2wY.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-52-91-176-16.compute-1.amazonaws.com,52.91.176.16' (ECDSA) to the list of known hosts.
Welcome to Ubuntu 20.04.3 LTS (GNU/Linux 5.11.0-1022-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

System information as of Tue Oct 10 14:43:28 UTC 2023

System load:  0.0               Processes:    103
Usage of /:   21.5% of 7.69GB   Users logged in: 0
Memory usage: 23%              IP4 address for ens5: 172.31.92.115
Swap usage:   0%

258 updates can be applied immediately.
187 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

New release '22.04.3 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Tue Oct 10 14:11:32 2023 from 10.234.25.220
ubuntu@ip-172-31-92-115:~$
```

Check installed versions

11)

Code:

python3 -- version

Terminal:

```
ubuntu@ip-172-31-92-115:~$ python3 --version
Python 3.8.10
ubuntu@ip-172-31-92-115:~$
```

12)

Code:

java -showversion

Terminal:

```
ubuntu@ip-172-31-92-115:~$ java -showversion
openjdk version "11.0.20.1" 2023-08-24
OpenJDK Runtime Environment (build 11.0.20.1+1-post-Ubuntu-0ubuntu120.04)
OpenJDK 64-Bit Server VM (build 11.0.20.1+1-post-Ubuntu-0ubuntu120.04, mixed mode, sharing)
ubuntu@ip-172-31-92-115:~$
```

13)

Code:

jenkins - - version

Terminal:

```
ubuntu@ip-172-31-92-115:~$ jenkins --version
2.414.2
ubuntu@ip-172-31-92-115:~$
```

Specifically check Jenkins status

14)

Code:

sudo systemctl status jenkins

Terminal:

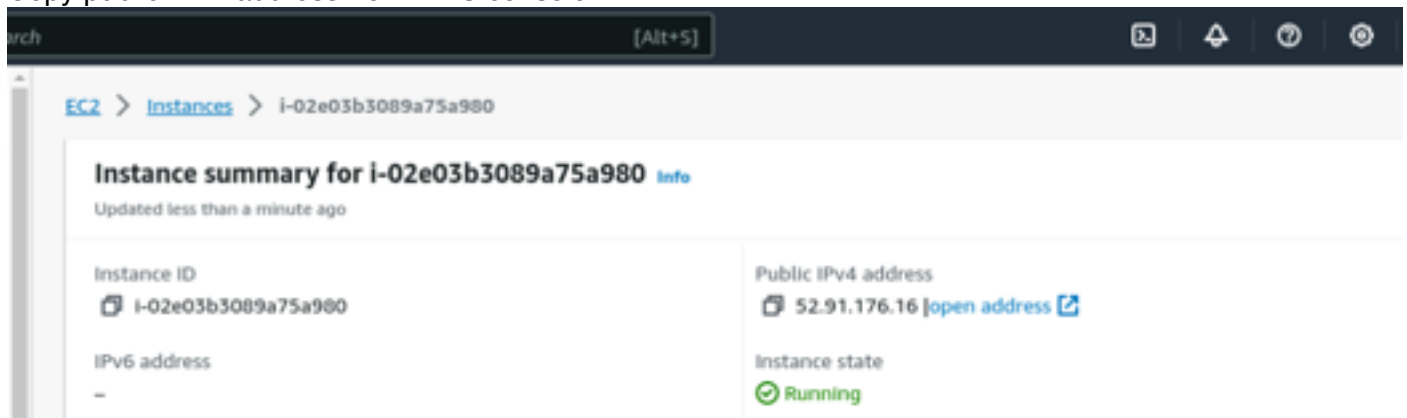
```
ubuntu@ip-172-31-92-115:~$ sudo systemctl status jenkins
● jenkins.service - Jenkins Continuous Integration Server
   Loaded: loaded (/lib/systemd/system/jenkins.service; enabled; vendor preset: enabled)
   Active: active (running) since Tue 2023-10-10 14:55:18 UTC; 2min 34s ago
     Main PID: 5553 (java)
       Tasks: 36 (limit: 1116)
      Memory: 312.6M
    CGroup: /system.slice/jenkins.service
            └─5553 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war --webroot=/var/cache/jenkins/war --httpPort=8080

Oct 10 14:54:32 ip-172-31-92-115 jenkins[5553]: a4af8b7c73d4e6fab98758d4c9a6c1a
Oct 10 14:54:32 ip-172-31-92-115 jenkins[5553]: This may also be found at: /var/lib/jenkins/secrets/initialAdminPassword
Oct 10 14:54:32 ip-172-31-92-115 jenkins[5553]: *****
Oct 10 14:54:32 ip-172-31-92-115 jenkins[5553]: *****
Oct 10 14:54:32 ip-172-31-92-115 jenkins[5553]: *****
Oct 10 14:55:18 ip-172-31-92-115 jenkins[5553]: 2023-10-10 14:55:18.269+0000 [id=20] INFO jenkins.InitReactorRunner$1#onAttained: Completed initialization
Oct 10 14:55:18 ip-172-31-92-115 jenkins[5553]: 2023-10-10 14:55:18.297+0000 [id=22] INFO hudson.lifecycle.Lifecycle#onReady: Jenkins is fully initialized
Oct 10 14:55:18 ip-172-31-92-115 systemd[1]: Started Jenkins Continuous Integration Server.
Oct 10 14:55:18 ip-172-31-92-115 jenkins[5553]: 2023-10-10 14:55:18.465+0000 [id=46] INFO h.m.DownloadService$Downloadable#load: Obtained the metadata
Oct 10 14:55:18 ip-172-31-92-115 jenkins[5553]: 2023-10-10 14:55:18.466+0000 [id=46] INFO hudson.util.Retrier#start: Performed the action check
lines 1-19/19 (END)
```

Accessing Jenkins

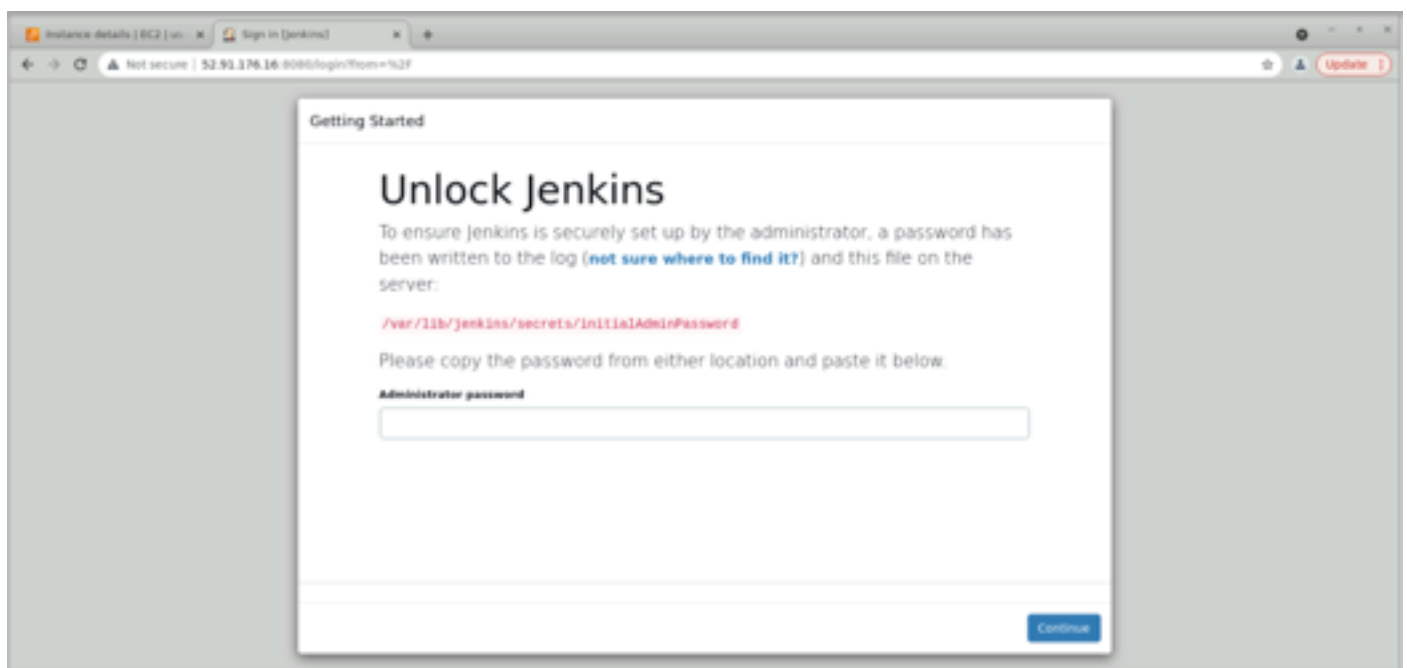
Getting Public IP

- 1) Copy public IPv4 address from AWS console



—> 52.91.176.16:8080

- 2) Ip is entered in chrome web browser with jenkins standard port 8080
52.91.176.16:8080



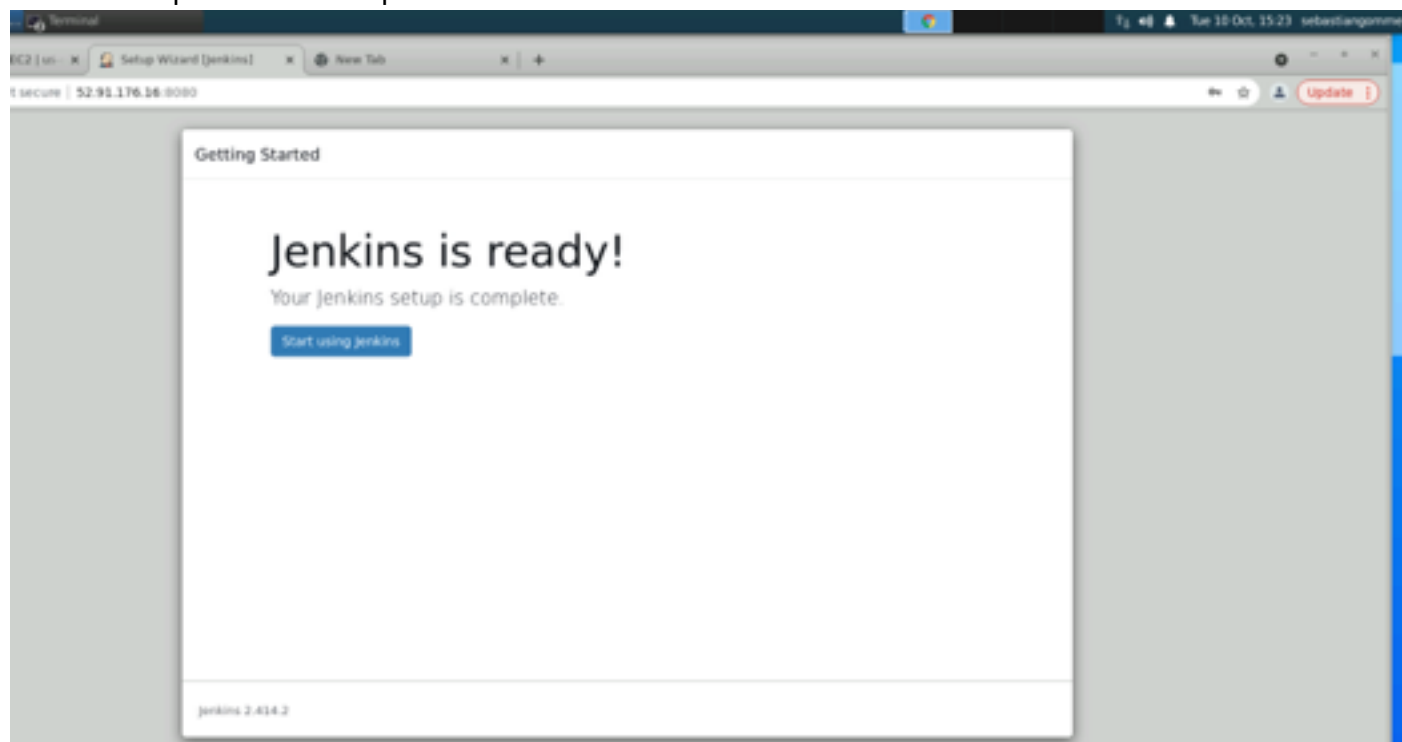
5)
The first User is created

The screenshot shows the Jenkins Setup Wizard interface in a web browser. The page is titled "Getting Started" and "Create First Admin User". It contains several input fields: "Username" with the value "admin", "Password" and "Confirm password" both with masked characters "*****", "Full name" with the value "Sebastian Gommel", and "E-mail address" with the value "sebastiangommel@nutanix.com". At the bottom, there is a "Skip and continue as admin" link and a "Save and Continue" button. The Jenkins version "jenkins 2.434.2" is displayed in the bottom left corner.

6)
Click Save and Finish

The screenshot shows the Jenkins Setup Wizard interface in a web browser. The page is titled "Getting Started" and "Instance Configuration". It features a "jenkins URL:" label and an input field containing "http://52.91.176.16:8080/". Below this, there is explanatory text about the Jenkins URL and a note that the proposed default value is not saved yet. At the bottom, there is a "Not now" link and a "Save and Finish" button. The Jenkins version "jenkins 2.434.2" is displayed in the bottom left corner.

Jenkins setup has been completed



Jenkins is now ready to use through the EC2 instance

