Subhakanta Mishra
Subhamishra.in@gmail.com

Configuration Management with Ansible and Terraform

Course-End Project: Web Application Using Ansible

Objective

To create an automation script to deploy an application using Ansible and Jinja2 template.

Real-time scenario:

You have joined as a DevOps engineer in XYZ Pvt. Ltd. It is a platform where individuals can create their profile and start blogging on various topics. The application is ready to be hosted on a server. You are tasked with implementing an Ansible script to deploy this application on a remote Nginx server.

Tasks

The following tasks outline the process of deploying web application on a remote server:

1. Create an inventory file to define the remote server(s)

2. Write a YAML playbook with tasks for installing Nginx, copying web application files, deploying the Nginx configuration, and enabling the site

3. Create a directory for templates and a Jinja2 template for the Nginx configuration

4. Define variables in the playbook for application details and Nginx configuration

5. Include tasks in the playbook for installing Nginx, copying application files, deploying Nginx configuration, and enabling the Nginx site

6. Execute the playbook to deploy the web application on the remote server

Subhakanta Mishra
Subhamishra.in@gmail.com

Solution

Step 1: Configure AWS CLI with access key and secret key to establish connection remotely

 # apt-get update && apt-get install awscli -y

```
root@ip-172-31-19-44:~# apt-get update && apt-get install awscli -y
Hit:1 https://apt.releases.hashicorp.com jammy InRelease
Hit:2 https://packages.microsoft.com/repos/azure-cli jammy InRelease
Hit:3 https://packages.microsoft.com/repos/code stable InRelease
Hit:4 http://ap-south-1c.clouds.ports.ubuntu.com/ubuntu-ports jammy InRelease
Hit:5 https://baltocdn.com/helm/stable/debian all InRelease
Get:6 http://ap-south-1c.clouds.ports.ubuntu.com/ubuntu-ports jammy-updates InRelease [128 kB]
Get:7 http://ports.ubuntu.com/ubuntu-ports jammy-security InRelease [129 kB]
Hit:8 https://ppa.launchpadcontent.net/pipewire-debian/pipewire-upstream/ubuntu jammy InRelease
Get:9 http://ap-south-1c.clouds.ports.ubuntu.com/ubuntu-ports jammy-backports InRelease [127 kB]
Get:10 http://ap-south-1c.clouds.ports.ubuntu.com/ubuntu-ports jammy-updates/main arm64 Packages [2419 kB]
```

 # aws configure

```
root@ip-172-31-19-44:~# aws configure
AWS Access Key ID [****************GRPR]: AKIAZSLGDBTS7AUUGRPR
AWS Secret Access Key [****************rfJl]: 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
Default region name [None]:
Default output format [None]:
root@ip-172-31-19-44:~#
```

Step 2: Install Terraform

wget -O - https://apt.releases.hashicorp.com/gpg | sudo gpg --dearmor -o /usr/share/keyrings/hashicorp-archive-keyring.gpg

echo "deb [arch=$(dpkg --print-architecture) signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg] https://apt.releases.hashicorp.com $(grep -oP '(?<=UBUNTU_CODENAME=).*' /etc/os-release || lsb_release -cs) main" | sudo tee /etc/apt/sources.list.d/hashicorp.list

sudo apt update && sudo apt install terraform

Subhakanta Mishra
Subhamishra.in@gmail.com

Check the terraform version by using the command “terraform –version”

```
root@ip-172-31-19-44:~# terraform --version
Terraform v1.0.0
on linux_arm64

Your version of Terraform is out of date! The latest version
is 1.12.2. You can update by downloading from https://www.terraform.io/downloads.html
root@ip-172-31-19-44:~#
```

Step 3: Configure Terraform with new ssh key which will be used as key pair for launching VMs.

mkdir myproject

cd myproject

vim mykey.tf

```
provider "aws" {

region = "us-east-1"

}

resource "tls_private_key" "mykey" {

  algorithm = "RSA"

}

resource "aws_key_pair" "aws-key" {

  key_name   = "web-key"

  public_key = tls_private_key.mykey.public_key_openssh

 provisioner "local-exec" {

command = "echo '${tls_private_key.mykey.private_key_pem}' > ./web-key.pem"

}

}
```

Subhakanta Mishra
Subhamishra.in@gmail.com

```
provider "aws" {

region = "us-east-1"
}

resource "tls_private_key" "mykey" {
  algorithm = "RSA"
}


resource "aws_key_pair" "aws-key" {
  key_name   = "web-key"
  public_key = tls_private_key.mykey.public_key_openssh

 provisioner "local-exec" {

command = "echo '${tls_private_key.mykey.private_key_pem}' > ./web-key.pem"

}

}
```

Run the command "**terraform init**"

```
root@ip-172-31-19-44:~# terraform init

Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/tls...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/tls v4.1.0...
- Installed hashicorp/tls v4.1.0 (signed by HashiCorp)
- Installing hashicorp/aws v6.0.0...
- Installed hashicorp/aws v6.0.0 (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.
root@ip-172-31-19-44:~#
```

Subhakanta Mishra
Subhamishra.in@gmail.com

Run the command **“terraform apply”**

```
root@ip-172-31-19-44:~# terraform apply --auto-approve

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_key_pair.aws-key will be created
  + resource "aws_key_pair" "aws-key" {
      + arn             = (known after apply)
      + fingerprint     = (known after apply)
      + id              = (known after apply)
      + key_name        = "web-key"
      + key_name_prefix = (known after apply)
      + key_pair_id     = (known after apply)
      + key_type        = (known after apply)
      + public_key      = (known after apply)
      + region          = "us-east-1"
      + tags_all        = (known after apply)
    }

  # tls_private_key.mykey will be created
  + resource "tls_private_key" "mykey" {
      + algorithm                     = "RSA"
      + ecdsa_curve                   = "P224"
      + id                            = (known after apply)
      + private_key_openssh           = (sensitive value)
      + private_key_pem               = (sensitive value)
      + private_key_pem_pkcs8         = (sensitive value)
      + public_key_fingerprint_md5    = (known after apply)
      + public_key_fingerprint_sha256 = (known after apply)
      + public_key_openssh            = (known after apply)
      + public_key_pem                = (known after apply)
      + rsa_bits                      = 2048
    }

Plan: 2 to add, 0 to change, 0 to destroy.
tls_private_key.mykey: Creating...
tls_private_key.mykey: Creation complete after 1s [id=17c327a8ccb24ae4ca27d57b03c4e20e708670ed]
aws_key_pair.aws-key: Creating...
aws_key_pair.aws-key: Provisioning with 'local-exec'...
aws_key_pair.aws-key (local-exec): (output suppressed due to sensitive value in config)
aws_key_pair.aws-key: Creation complete after 2s [id=web-key]

Apply complete! Resources: 2 added, 0 changed, 0 destroyed.
root@ip-172-31-19-44:~#
```

Step 4: Terraform script to provision and empty sandbox, add various setting to the sandbox like VPC, security group, route table, subnets, and key pair

vim main.tf

```
resource "aws_vpc" "sl-vpc" {

cidr_block = "10.0.0.0/16"

tags = {

   Name = "sl-vpc"

 }

}

resource "aws_subnet" "sl-subnet" {

 vpc_id     = aws_vpc.sl-vpc.id

 cidr_block = "10.0.1.0/24"

 depends_on = [aws_vpc.sl-vpc]
```

Subhakanta Mishra
Subhamishra.in@gmail.com

```
  map_public_ip_on_launch = true
  tags = {
    Name = "sl-subnet"
  }
}
resource "aws_route_table" "sl-route-table" {
  vpc_id = aws_vpc.sl-vpc.id
 tags = {
    Name = "sl-route-table"
  }
}
resource "aws_route_table_association" "a" {
  subnet_id      = aws_subnet.sl-subnet.id
  route_table_id = aws_route_table.sl-route-table.id
}
resource "aws_internet_gateway" "gw" {
  vpc_id = aws_vpc.sl-vpc.id
  depends_on = [aws_vpc.sl-vpc]
  tags = {
    Name = "sl-gw"
  }
}
resource "aws_route" "sl-route" {
  route_table_id            = aws_route_table.sl-route-table.id
  destination_cidr_block    = "0.0.0.0/0"
  gateway_id = aws_internet_gateway.gw.id
}
variable "sg_ports" {
```

Subhakanta Mishra
Subhamishra.in@gmail.com

```
type = list(number)

default = [22,443,80,8080]

}

resource "aws_security_group" "sl-sg" {

 name       = "sl-sg"

 description = "Allow TLS inbound traffic and all outbound traffic"

 vpc_id = aws_vpc.sl-vpc.id

  dynamic "ingress" {

   for_each = var.sg_ports

   iterator = ports

   content{

   from_port       = ports.value

   to_port         = ports.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"]

     }

}

data "aws_ami" "myami" {

most_recent     = true

owners          = ["amazon"]

filter {

   name   = "name"

   values = ["amzn2-ami-kernel-5.10-hvm-2.0.20250610.0-x86_64-gp2"]
```

Subhakanta Mishra
Subhamishra.in@gmail.com

```
 }
}
resource "aws_instance" "myec2" {
 ami           = data.aws_ami.myami.id
 instance_type = "t2.micro"
 key_name = "web-key"
 subnet_id = aws_subnet.sl-subnet.id
 security_groups = [aws_security_group.sl-sg.id]
 tags = {
  Name = "terraform-instance"
 }
}
```

Subhakanta Mishra
Subhamishra.in@gmail.com

```
resource "aws_vpc" "sl-vpc" {

cidr_block = "10.0.0.0/16"

tags = {
    Name = "sl-vpc"
  }

}
resource "aws_subnet" "sl-subnet" {

  vpc_id     = aws_vpc.sl-vpc.id

  cidr_block = "10.0.1.0/24"

  depends_on = [aws_vpc.sl-vpc]

  map_public_ip_on_launch = true

  tags = {
    Name = "sl-subnet"
  }
}

resource "aws_route_table" "sl-route-table" {
  vpc_id = aws_vpc.sl-vpc.id

 tags = {
    Name = "sl-route-table"
  }
}

resource "aws_route_table_association" "a" {
  subnet_id      = aws_subnet.sl-subnet.id
  route_table_id = aws_route_table.sl-route-table.id
}


resource "aws_internet_gateway" "gw" {
  vpc_id = aws_vpc.sl-vpc.id
  depends_on = [aws_vpc.sl-vpc]
  tags = {
    Name = "sl-gw"
  }
}
```

Subhakanta Mishra
Subhamishra.in@gmail.com

```
resource "aws_route" "sl-route" {
  route_table_id            = aws_route_table.sl-route-table.id
  destination_cidr_block    = "0.0.0.0/0"
  gateway_id = aws_internet_gateway.gw.id

}

variable "sg_ports" {

type = list(number)

default = [22,443,80,8080]

}

resource "aws_security_group" "sl-sg" {
  name        = "sl-sg"
  description = "Allow TLS inbound traffic and all outbound traffic"
  vpc_id = aws_vpc.sl-vpc.id
   dynamic "ingress" {
    for_each = var.sg_ports
    iterator = ports
    content{
    from_port        = ports.value
    to_port          = ports.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"]


       }

}
```

Subhakanta Mishra
Subhamishra.in@gmail.com

```
data "aws_ami" "myami" {

most_recent      = true

owners           = ["amazon"]

 filter {
    name   = "name"
    values = ["amzn2-ami-kernel-5.10-hvm-2.0.20250610.0-x86_64-gp2"]
  }


}

resource "aws_instance" "myec2" {

  ami           = data.aws_ami.myami.id
  instance_type = "t2.micro"
  key_name = "web-key"
  subnet_id = aws_subnet.sl-subnet.id
  security_groups = [aws_security_group.sl-sg.id]
  tags = {
    Name = "terraform-instance"
  }

}
```

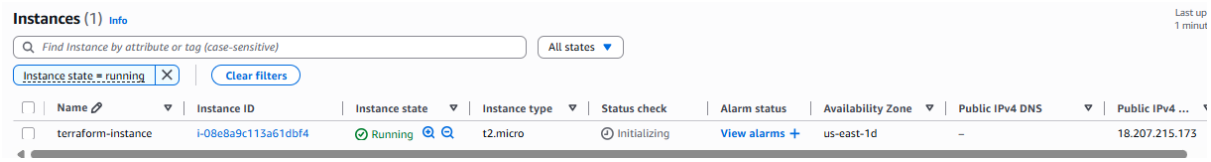
Save the file

Run the command "**terraform apply --auto-approve**"

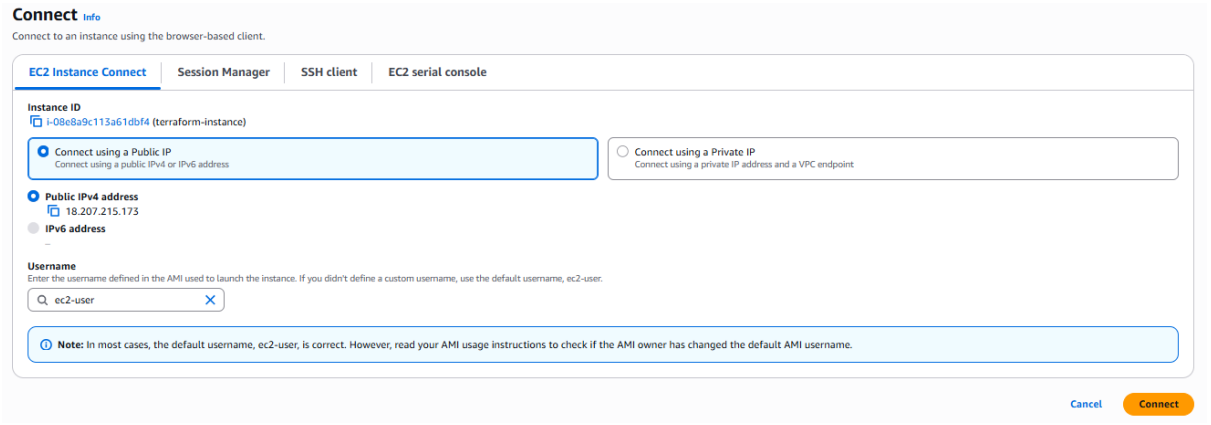
```
Plan: 8 to add, 0 to change, 0 to destroy.
aws_vpc.sl-vpc: Creating...
aws_vpc.sl-vpc: Creation complete after 4s [id=vpc-04899a3416f77437b]
aws_subnet.sl-subnet: Creating...
aws_route_table.sl-route-table: Creating...
aws_internet_gateway.gw: Creating...
aws_security_group.sl-sg: Creating...
aws_route_table.sl-route-table: Creation complete after 1s [id=rtb-0ea4f76b8a8e803d9]
aws_internet_gateway.gw: Creation complete after 1s [id=igw-04251e1bb6a1da5a2]
aws_route.sl-route: Creating...
aws_route.sl-route: Creation complete after 2s [id=r-rtb-0ea4f76b8a8e803d91080289494]
aws_security_group.sl-sg: Creation complete after 5s [id=sg-0ba91f136bee53b85]
aws_subnet.sl-subnet: Still creating... [10s elapsed]
aws_subnet.sl-subnet: Creation complete after 12s [id=subnet-03860fd499e110f23]
aws_route_table_association.a: Creating...
aws_instance.myec2: Creating...
aws_route_table_association.a: Creation complete after 1s [id=rtbassoc-00c95979cd7882c53]
aws_instance.myec2: Still creating... [10s elapsed]
aws_instance.myec2: Still creating... [20s elapsed]
aws_instance.myec2: Still creating... [30s elapsed]
aws_instance.myec2: Creation complete after 35s [id=i-08e8a9c113a61dbf4]

Apply complete! Resources: 8 added, 0 changed, 0 destroyed.
```

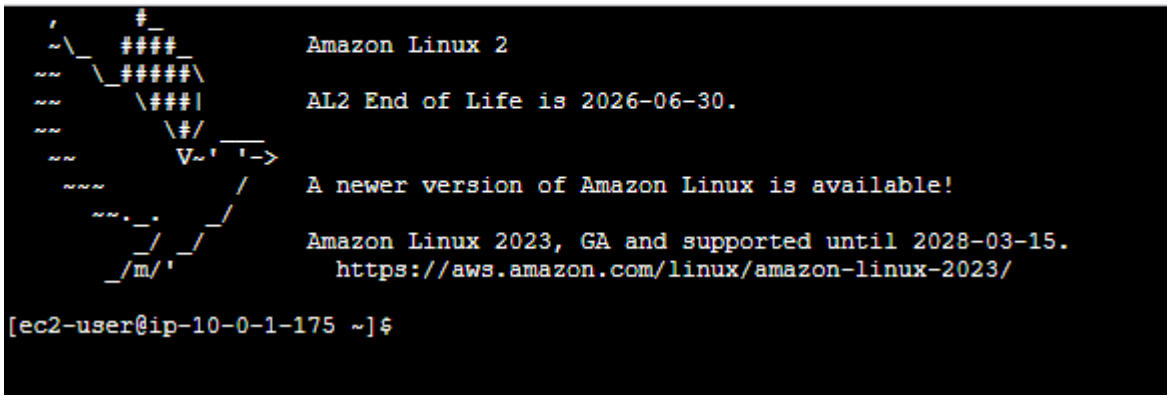
Subhakanta Mishra
Subhamishra.in@gmail.com

Login to AWS and go to EC2 instances.



Click on connect to connect the EC2 machine.





Step 5: Ansible Setup for running the playbook on above created VM

Generate ssh keys for root user on Lab terminal(Ansible Controller)

ssh-keygen

Subhakanta Mishra
Subhamishra.in@gmail.com

```
root@ip-172-31-19-44:~# 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:YUZzGUcstTfLAOyPLsT+2b2KEdx9gq/xGHyyd3xxE/o root@ip-172-31-19-44
The key's randomart image is:
+---[RSA 3072]----+
|         o.+*+    |
|        . o+o..   |
|         +. .o o  |
|        o o.. * + |
|        .S ooo * o|
|         o .o.o =.|
|        o .. = +.+|
|         o .= @.E+|
|          o+ *o+o.|
+----[SHA256]-----+
root@ip-172-31-19-44:~#
```

cat /root/.ssh/id_rsa.pub

GO to AWS EC2 server - Ansible worker

 # cd .ssh

 # echo "GIVE YOUR SSH PUBLIC KEY" >> ~/.ssh/authorized_keys

Subhakanta Mishra
Subhamishra.in@gmail.com

Connect to the EC2 machine vis ssh in lab terminal

```
root@ip-172-31-19-44:~# ssh 18.207.215.173
Warning: Permanently added '18.207.215.173' (ED25519) to the list of known hosts.
Last login: Sat Jun 21 16:31:05 2025
   ,     #_
   ~\_  ####_        Amazon Linux 2
  ~~  \_#####\
  ~~     \###|       AL2 End of Life is 2026-06-30.
  ~~       \#/ ___
   ~~       V~' '->
    ~~~         /    A newer version of Amazon Linux is available!
      ~~._.   _/
         _/ _/       Amazon Linux 2023, GA and supported until 2028-03-15.
       _/m/'           https://aws.amazon.com/linux/amazon-linux-2023/

[root@ip-10-0-1-175 ~]#
```

Step 6: Create Ansible Inventory file with IP address of AWS EC2 server

In same directory where the terraform code is, create the ansible inventory

vim myinventory

[webserver]

<public ip of worker ec2 instance>

Save the file

```
root@ip-172-31-19-44:~# vim myinventory
root@ip-172-31-19-44:~# cat myinventory
[webserver]
18.207.215.173
```

 # pwd

Copy the path of the directory

 # vim ansible.cfg

[defaults]

inventory = /root/myproject/myinventory

Save the file

Run ping command to check the connection

Validate the setup :

ansible webserver -m ping

Subhakanta Mishra
Subhamishra.in@gmail.com

```
root@ip-172-31-19-44:~# ansible -m ping webserver
[WARNING]: Platform linux on host 18.207.215.173 is using the dis
See https://docs.ansible.com/ansible-core/2.16/reference_appendic
18.207.215.173 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3.7"
    },
    "changed": false,
    "ping": "pong"
}
```

Step 7: Write the Ansible playbook.

vim playbook.yml

- name: run playbook using terraform

 hosts: webserver

 become: true

 tasks:

 - name: Install python3

  package: name=python3 state=present

 - name: Install maven

  package: name=maven state=present

 - name: Create a file

  file: path=/tmp/ansible.txt state=touch

```
- name: run playbook using terraform
  hosts: webserver
  become: true
  tasks:
  - name: Install python3
    package: name=python3 state=present
  - name: Install maven
    package: name=maven state=present
  - name: Create a file
    file: path=/tmp/ansible.txt state=touch
```

Save the file

Subhakanta Mishra
Subhamishra.in@gmail.com

Step 8: Write Terraform code to run the playbook:

vim runplaybook.tf

resource "null_resource" "run_playbook" {

provisioner "local-exec" {

command = "ansible-playbook playbook.yml"

}

}

Save the file

```
resource "null_resource" "run_playbook" {
provisioner "local-exec" {
command = "ansible-playbook playbook.yml"
}
}
```

Step 9: Execute the playbook using terraform command

terraform init

terraform apply -target null_resource.run_playbook

```
null_resource.run_playbook (local-exec): PLAY [run playbook using terraform] ************************************

null_resource.run_playbook (local-exec): TASK [Gathering Facts] ************************************************
null_resource.run_playbook (local-exec): [WARNING]: Platform linux on host 54.152.62.151 is using the discovered Python
null_resource.run_playbook (local-exec): interpreter at /usr/bin/python3.7, but future installation of another Python
null_resource.run_playbook (local-exec): interpreter could change the meaning of that path. See
null_resource.run_playbook (local-exec): https://docs.ansible.com/ansible-
null_resource.run_playbook (local-exec): core/2.16/reference_appendices/interpreter_discovery.html for more information.
null_resource.run_playbook (local-exec): ok: [54.152.62.151]

null_resource.run_playbook (local-exec): TASK [Install python3] ************************************************
null_resource.run_playbook: Still creating... [10s elapsed]
null_resource.run_playbook (local-exec): ok: [54.152.62.151]

null_resource.run_playbook (local-exec): TASK [Install maven] **************************************************
null_resource.run_playbook: Still creating... [20s elapsed]
null_resource.run_playbook: Still creating... [30s elapsed]
null_resource.run_playbook: Still creating... [40s elapsed]
null_resource.run_playbook (local-exec): changed: [54.152.62.151]

null_resource.run_playbook (local-exec): TASK [Create a file] ***************************************************
null_resource.run_playbook (local-exec): changed: [54.152.62.151]

null_resource.run_playbook (local-exec): PLAY RECAP *********************************************************************
null_resource.run_playbook (local-exec): 54.152.62.151             : ok=4    changed=2    unreachable=0    failed=0    skipped=0    rescued
=0    ignored=0

null_resource.run_playbook: Creation complete after 48s [id=4406932644251169376]

Warning: Applied changes may be incomplete

The plan was created with the -target option in effect, so some changes requested in the configuration may have been ignored and the
output values may not be fully updated. Run the following command to verify that no other changes are pending:
    terraform plan

Note that the -target option is not suitable for routine use, and is provided only for exceptional situations such as recovering from
errors or mistakes, or when Terraform specifically suggests to use it as part of an error message.

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
root@ip-172-31-21-208:~/myproject#
```