

## Terraform Project

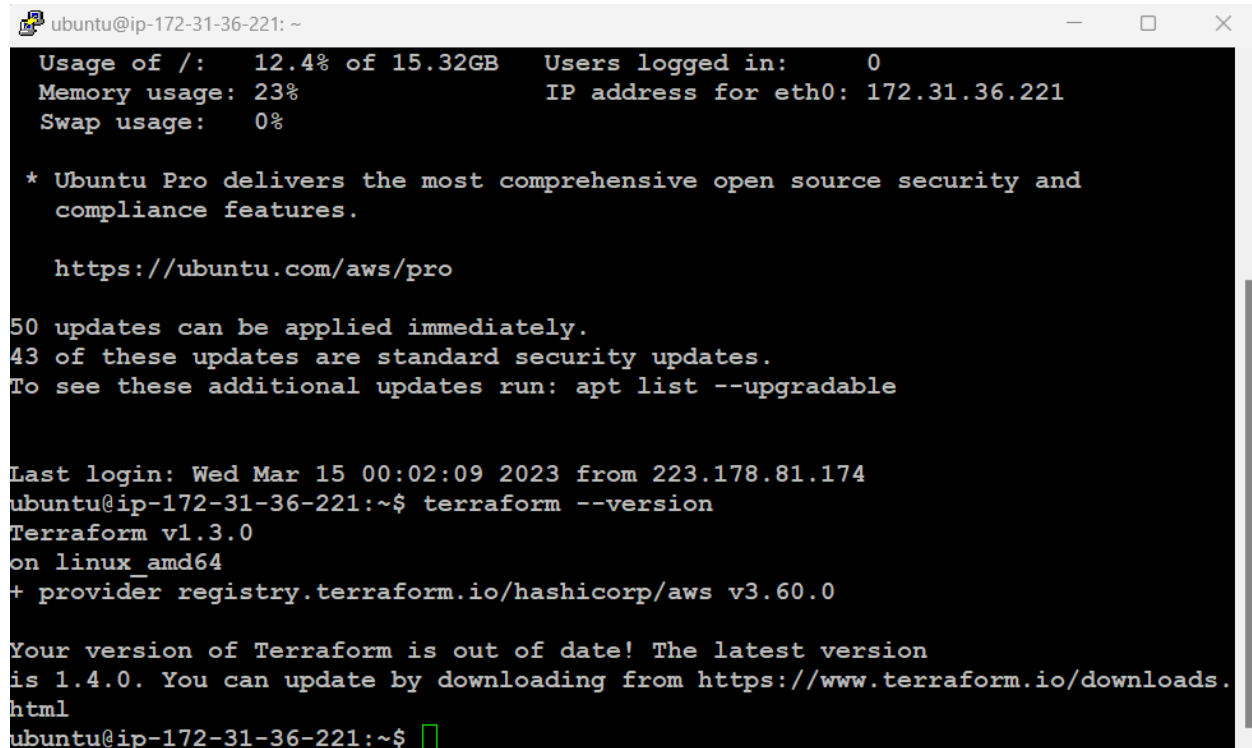
The problem statement is to create the instance using terraform

### Requirements:

- Creating Ubuntu 20.04 LTS Operating System.
- Type t2.micro
- Default vpc
- Security group ssh from anywhere
- Tag name with “my-ec2-instance”

### Solution

Installing Terraform master with ubuntu os and installing terraform version 1.3.6 for running code without bugs and aws version =3.60.0



```
ubuntu@ip-172-31-36-221: ~  
Usage of /: 12.4% of 15.32GB  Users logged in: 0  
Memory usage: 23%          IP address for eth0: 172.31.36.221  
Swap usage: 0%  
  
* Ubuntu Pro delivers the most comprehensive open source security and  
  compliance features.  
  
  https://ubuntu.com/aws/pro  
  
50 updates can be applied immediately.  
43 of these updates are standard security updates.  
To see these additional updates run: apt list --upgradable  
  
Last login: Wed Mar 15 00:02:09 2023 from 223.178.81.174  
ubuntu@ip-172-31-36-221:~$ terraform --version  
Terraform v1.3.0  
on linux_amd64  
+ provider registry.terraform.io/hashicorp/aws v3.60.0  
  
Your version of Terraform is out of date! The latest version  
is 1.4.0. You can update by downloading from https://www.terraform.io/downloads.  
html  
ubuntu@ip-172-31-36-221:~$
```

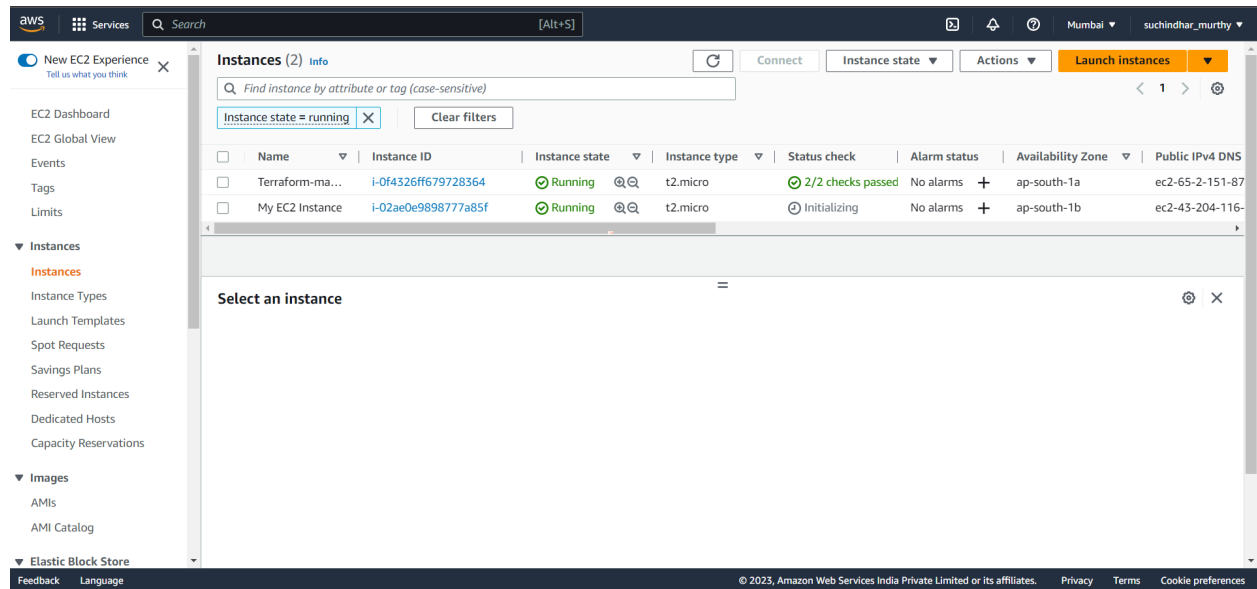
Terraform.lock.hcl which locks the aws version in system.

```
ubuntu@ip-172-31-36-221: ~  
-rw----- 1 ubuntu ubuntu 167 Mar 15 00:10 .bash_history  
ubuntu@ip-172-31-36-221:~$ cat .terraform.lock.hcl  
# This file is maintained automatically by "terraform init".  
# Manual edits may be lost in future updates.  
  
provider "registry.terraform.io/hashicorp/aws" {  
  version      = "3.60.0"  
  constraints = "3.60.0"  
  hashes = [  
    "h1:vwRjnpZOFWd1bFb2WX10JM2zNEEVyRLc8cBwkxCX1AE=",  
    "zh:01323eedb8f006c8f9fffdcf23b449625b1446c1e43b8454e4a40a7461193661",  
    "zh:03513ffdae205832be480b30d332b47a573e48623390e8f9f833141c8ceccb6a",  
    "zh:47611a8b361ced9a3b58b9868be2004677cf4ea0d04cfb5f54c6ae95e997e7c7",  
    "zh:9a7e80c2a2ed0f2e59b05e27374daafd64785161546ed40f4db11048fbc78a7",  
    "zh:9e809746c4fdaa4214700e81a67b35f02afc1f2873591b0360c473cfd7193877",  
    "zh:a009d48e4ebcf78e24af9299c6a8664e0375411b4f16d5d0d7c7454b12052c10",  
    "zh:adc910f48f5ddc402e7653e70429d150d61bee5190aba7495575303aba6ca6c8",  
    "zh:b702e219532bc09be58f8a30cb3239626ffc9bc0e42b44497b0644f9ecc657b5",  
    "zh:bc50d787593e714acb54d65e8df026490a968e54d2184496efda7ba07c211836",  
    "zh:bd74e3b1c815d5a9c710cb5c55f2d5f6742471a23e63f924fd3a6493f384cd43",  
    "zh:fa7eb23bcf4c01f93d74c509c0e9b039148f43424c3b4ce64619af17ee12265c",  
  ]  
}  
ubuntu@ip-172-31-36-221:~$
```

Creating the instance according to the problem statement ubuntu 20.04 instance and security-group

```
root@ip-172-31-47-14: ~  
terraform {  
  required_version = ">=1.3.0"  
  
  required_providers {  
    aws = {  
      source = "hashicorp/aws"  
      version = "3.60.0"  
    }  
  }  
}  
  
provider "aws" {  
  region = "ap-south-1"  
  profile = "default"  
}  
  
resource "aws_instance" "My-Ec2-Instance" {  
  ami           = "ami-0caf778a172362f1c"  
  instance_type = "t2.micro"  
  key_name      = "terraform"  
  vpc_security_group_ids = [  
    aws_security_group.allow_ssh.id  
  ]  
  tags = {  
    Name = "My EC2 Instance"  
  }  
}  
  
resource "aws_security_group" "allow_ssh" {  
  name_prefix = "allow_ssh"  
  ingress {  
    description = "SSH access"  
    from_port   = 22  
    to_port     = 22  
    protocol    = "tcp"  
    cidr_blocks = ["0.0.0.0/0"]  
  }  
}
```

The instance has been created



```
ubuntu@ip-172-31-1-114: ~$  
Welcome to Ubuntu 20.04.5 LTS (GNU/Linux 5.15.0-1028-aws x86_64)  
  
* Documentation:  https://help.ubuntu.com  
* Management:    https://landscape.canonical.com  
* Support:       https://ubuntu.com/advantage  
  
System information as of Sat Mar 18 06:19:52 UTC 2023  
  
System load:  0.0          Processes:            98  
Usage of /:   20.4% of 7.57GB   Users logged in:     0  
Memory usage: 21%          IPv4 address for eth0: 172.31.1.114  
Swap usage:   0%  
  
Expanded Security Maintenance for Applications is not enabled.  
  
0 updates can be applied immediately.  
  
Enable ESM Apps to receive additional future security updates.  
See https://ubuntu.com/esm or run: sudo pro status  
  
The list of available updates is more than a week old.  
To check for new updates run: sudo apt update  
  
The programs included with the Ubuntu system are free software;  
the exact distribution terms for each program are described in the  
individual files in /usr/share/doc/*/copyright.  
  
Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by  
applicable law.  
  
To run a command as administrator (user "root"), use "sudo <command>".  
See "man sudo_root" for details.
```

And security group is also created **ssh 22 from anywhere** and created with a **default VPC**

aws

Services

Search

[Alt+S]

Mumbai

suchindhar\_murthy

New EC2 Experience

EC2 Dashboard

EC2 Global View

Events

Tags

Limits

Instances

Instance Types

Launch Templates

Spot Requests

Savings Plans

Reserved Instances

Dedicated Hosts

Capacity Reservations

Images

AMIs

AMI Catalog

Elastic Block Store

Instances (1/2) Info

Find instance by attribute or tag (case-sensitive)

Instance state = running

Clear filters

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
<input type="checkbox"/>	Terraform-ma...	i-0f4326ff679728364	Running	t2.micro	2/2 checks passed	No alarms	ap-south-1a	ec2-65-2-151-87
<input checked="" type="checkbox"/>	My EC2 Instance	i-02ae0e9898777a85f	Running	t2.micro	2/2 checks passed	No alarms	ap-south-1b	ec2-43-204-116-

Instance: i-02ae0e9898777a85f (My EC2 Instance)

Instance summary Info

Instance ID	Public IPv4 address	Private IPv4 addresses
i-02ae0e9898777a85f (My EC2 Instance)	43.204.116.157   open address	172.31.1.114
IPv6 address	Instance state	Public IPv4 DNS
-	Running	ec2-43-204-116-157.ap-south-1.compute.amazonaws.com   open address
Hostname type	Private IP DNS name (IPv4 only)	Elastic IP addresses
IP name: ip-172-31-1-114.ap-south-1.compute.internal	ip-172-31-1-114.ap-south-1.compute.internal	-
Answer private resource DNS name	Instance type	
-	t2.micro	

aws

Services

Search

[Alt+S]

Mumbai

suchindhar\_murthy

New EC2 Experience

EC2 Dashboard

EC2 Global View

Events

Tags

Limits

Instances

Instance Types

Launch Templates

Spot Requests

Savings Plans

Reserved Instances

Dedicated Hosts

Capacity Reservations

Images

AMIs

AMI Catalog

Elastic Block Store

Details

Security group name	Security group ID	Description	VPC ID
allow_ssh2023031806035589890000001	sg-03e6dec68f2fae983	Managed by Terraform	vpc-0fd1a520eeb91c637
Owner	Inbound rules count	Outbound rules count	
57305077619	1 Permission entry	0 Permission entries	

Inbound rules

Outbound rules

Tags

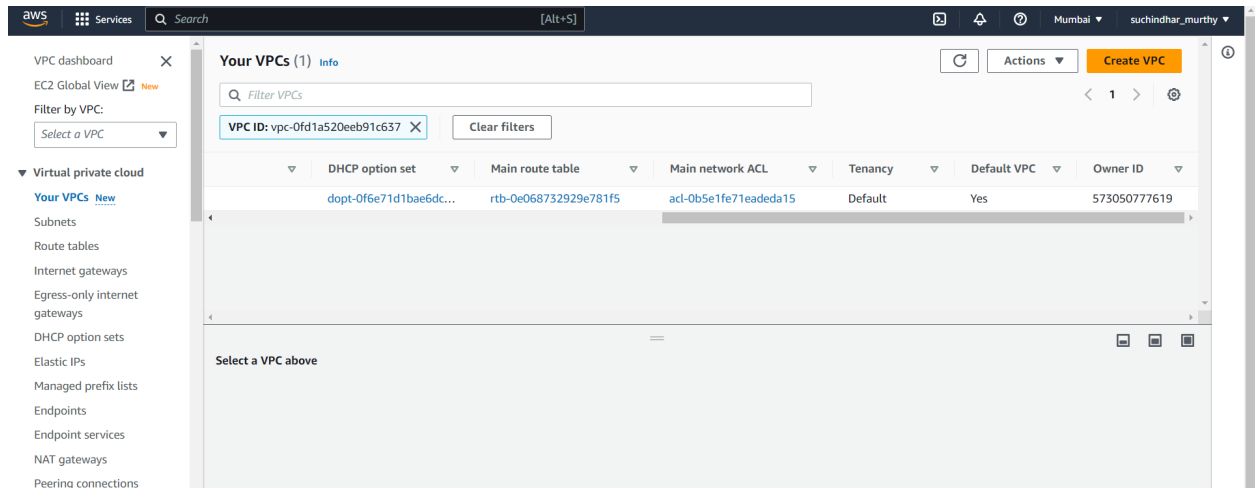
You can now check network connectivity with Reachability Analyzer

Run Reachability Analyzer

Inbound rules (1/1)

Filter security group rules

	Name	Security group rule...	IP version	Type	Protocol	Port range
<input checked="" type="checkbox"/>	-	sgr-045415a4c40e967...	IPv4	SSH	TCP	22



## Main.tf file

### # terraform block

```
terraform {
  required_version = "~>1.3.0"
```

```
  required_providers {
    aws = {
      source = "hashicorp/aws"
      version = "3.60.0"
    }
  }
}
```

### #provider block

```
provider "aws" {
  region = "ap-south-1"
  profile = "default"
}
```

### #resource block

```
resource "aws_instance" "My-Ec2-Instance" {
  ami          = "ami-0caf778a172362f1c"
  instance_type = "t2.micro"
  key_name     = "terraform"
  vpc_security_group_ids = [
    aws_security_group.allow_ssh.id
  ]
  tags = {
```

```

    Name = "My EC2 Instance"
  }
}

resource "aws_security_group" "allow_ssh" {
  name_prefix = "allow_ssh"
  ingress {
    description = "SSH access"
    from_port   = 22
    to_port     = 22
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }
}

```

## Commands

**Terraform init** to initialize the terraform

```

root@ip-172-31-47-14:~# terraform init

Initializing the backend...

Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v3.60.0

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-172-31-47-14:~# █

```

## Terraform fmt to format the main file

```
root@ip-172-31-47-14:~# terraform init

Initializing the backend...

Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v3.60.0

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-172-31-47-14:~# terraform fmt
main.tf
root@ip-172-31-47-14:~#
```

## Terraform validate to check the syntax

```
# Manual edits may be lost in future updates.

provider "registry.terraform.io/hashicorp/aws" {
  version      = "3.60.0"
  constraints = "3.60.0"
  hashes = [
    "h1:vwRjnpZOFwDlbFb2WX10JM2zNEEVyRLc8cBwKxCK1AE=",
    "zh:01323eedb8f006c8f9ffdfc23b449625b1446c1e43b8454e4a40a7461193661",
    "zh:03513ffdae205832be480b30d332b47a573e48623390e8f9f833141c8ceccb6a",
    "zh:47611a8b361ced9a3b58b9868be2004677cf4ea0d04c5f54c6ae95e997e7c7",
    "zh:9a7e80c2a2ed0f2e59b05e27374daaafd64785161546ed40f4db11048fbc78a7",
    "zh:9e809746c4fd4a214700e81a67b35f02afc1f2873591b0360c473cfd7193877",
    "zh:a009d48e4ebcf78e24af9299c6a8664e0375411b4f16d5d0d7c7454b12052c10",
    "zh:adc910f48f5ddc402e7653e70429d150d61bee5190aba7495575303aba6ca6c8",
    "zh:b702e219532bc09be58f8a30cb3239626ffc9bc0e42b44497b0644f9ecc657b5",
    "zh:bc50d787593e714acb54d65e8df026490a968e54d2184496efda7ba07c211836",
    "zh:bd74e3b1c815d5a9c710cb5c55f2d5f6742471a23e63f924fd3a6493f384cd43",
    "zh:fa7eb23bcf4c01f93d74c509c0e9b039148f43424c3b4ce64619af17ee12265c",
  ]
}

root@ip-172-31-47-14:~# terraform init

Initializing the backend...

Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v3.60.0

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-172-31-47-14:~# terraform fmt
main.tf
root@ip-172-31-47-14:~# terraform validate
Success! The configuration is valid.

root@ip-172-31-47-14:~#
```

## Terraform plan to view the main file plans

```
root@ip-172-31-14-14:~# terraform plan
+ 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.allow_ssh will be created
+ resource "aws_security_group" "allow_ssh" {
+   arn                = (known after apply)
+   description        = "Managed by Terraform"
+   egress              = (known after apply)
+   id                 = (known after apply)
+   ingress             = [
+     {
+       cidr_blocks     = [
+         "0.0.0.0/0",
+       ]
+       description     = "SSH access"
+       from_port       = 22
+       ipv6_cidr_blocks = []
+       prefix_list_ids = []
+       protocol        = "tcp"
+       security_groups = []
+       self            = false
+       to_port         = 22
+     },
+   ]
+   name               = (known after apply)
+   name_prefix        = "allow_ssh"
+   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.
root@ip-172-31-14-14:~#
```

**Terraform apply - - auto approve** - It is used to creates the application what i have provided in the main.tf.

```
root@ip-172-31-14-14:~# terraform apply
+ volume_id           = (known after apply)
+ volume_size         = (known after apply)
+ volume_type         = (known after apply)
}
}

# aws_security_group.allow_ssh will be created
+ resource "aws_security_group" "allow_ssh" {
+   arn                = (known after apply)
+   description        = "Managed by Terraform"
+   egress              = (known after apply)
+   id                 = (known after apply)
+   ingress             = [
+     {
+       cidr_blocks     = [
+         "0.0.0.0/0",
+       ]
+       description     = "SSH access"
+       from_port       = 22
+       ipv6_cidr_blocks = []
+       prefix_list_ids = []
+       protocol        = "tcp"
+       security_groups = []
+       self            = false
+       to_port         = 22
+     },
+   ]
+   name               = (known after apply)
+   name_prefix        = "allow_ssh"
+   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
```

**Terraform destroy - - auto approve** it is used to destroy the application what i have provided in the main.tf and creates backup as a tfstate file



```
Plan: 0 to add, 0 to change, 2 to destroy.
aws_instance.My-Ec2-Instance: Destroying... [id=i-02ae0e9898777a85f]
aws_instance.My-Ec2-Instance: Still destroying... [id=i-02ae0e9898777a85f, 10s elapsed]
aws_instance.My-Ec2-Instance: Still destroying... [id=i-02ae0e9898777a85f, 20s elapsed]
aws_instance.My-Ec2-Instance: Still destroying... [id=i-02ae0e9898777a85f, 30s elapsed]
aws_instance.My-Ec2-Instance: Destruction complete after 40s
aws_security_group.allow_ssh: Destroying... [id=sg-03e6dec68f2fae983]
aws_security_group.allow_ssh: Destruction complete after 0s
```

```
Destroy complete! Resources: 2 destroyed.
root@ip-172-31-47-14:~# ls -lrta
total 56
-rw-r--r-- 1 root root 148 Aug 17 2015 .profile
-rw-r--r-- 1 root root 3106 Apr 9 2018 .bashrc
drwxr-xr-x 23 root root 4096 Mar 18 05:51 ..
drwx----- 2 root root 4096 Mar 18 05:51 .ssh
drwx----- 3 root root 4096 Mar 18 05:51 snap
drwxr-xr-x 2 root root 4096 Mar 18 05:55 .terraform.d
drwxr-xr-x 3 root root 4096 Mar 18 05:56 .terraform
-rw-r--r-- 1 root root 1106 Mar 18 05:56 .terraform.lock.hcl
-rw-r--r-- 1 root root 693 Mar 18 06:23 main.tf
-rw----- 1 root root 3012 Mar 18 06:23 .viminfo
-rw-r--r-- 1 root root 5441 Mar 18 06:23 terraform.tfstate.backup
-rw-r--r-- 1 root root 181 Mar 18 06:24 terraform.tfstate
drwx----- 6 root root 4096 Mar 18 06:24 .
root@ip-172-31-47-14:~#
```

```
root@ip-172-31-47-14:~# cat terraform.tfstate
{
  "version": 4,
  "terraform_version": "1.3.9",
  "serial": 12,
  "lineage": "13f1b03a-68b8-740e-95a4-5450016a7335",
  "outputs": {},
  "resources": [],
  "check_results": null
}
root@ip-172-31-47-14:~#
```