1.Launched EC2 for terraform & awscli configuration

Instances (1) Info

Find Instance by attribute or tag (case-sensitive) | All states

Name = terraformec2 | Clear filters

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
terraformec2	i-0b5a85787fe129d7d	Running	t2.micro	2/2 checks passed	View alarms	ap-south-1a	ec2-13-201-86-64.ap-s...

2.Terraform & AWS CLI installed, AWS configure Done with User ID

```
ubuntu@ip-172-31-39-183:~$ terraform version
Terraform v1.8.2
on linux_amd64
ubuntu@ip-172-31-39-183:~$ aws --version
aws-cli/2.15.45 Python/3.11.8 Linux/6.5.0-1017-aws exe/x86_64.ubuntu.22 prompt/off
ubuntu@ip-172-31-39-183:~$ aws configure
AWS Access Key ID [None]: AKIAU6GDUNP6VGSEFEMZ
AWS Secret Access Key [None]: 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
Default region name [None]: ap-south-1
Default output format [None]: json
ubuntu@ip-172-31-39-183:~$
```

3.User ID Access Key Generated in IAM Role

IAM > Users > My > Create access key

Step 1
Access key best practices & alternatives

Step 2 - *optional*
Set description tag

Step 3
Retrieve access keys

Access key best practices & alternatives Info

Avoid using long-term credentials like access keys to improve your security. Consider the following use cases and alternatives.

Use case

- (●) **Command Line Interface (CLI)**
  You plan to use this access key to enable the AWS CLI to access your AWS account.
- ( ) **Local code**
  You plan to use this access key to enable application code in a local development environment to access your AWS account.
- ( ) **Application running on an AWS compute service**
  You plan to use this access key to enable application code running on an AWS compute service like Amazon EC2, Amazon ECS, or AWS Lambda to access your AWS account.
- ( ) **Third-party service**
  You plan to use this access key to enable access for a third-party application or service that monitors or manages your AWS resources.
- ( ) **Application running outside AWS**
  You plan to use this access key to authenticate workloads running in your data center or other infrastructure outside of AWS that needs to access your AWS resources.
- ( ) **Other**
  Your use case is not listed here.

Alternatives recommended

- Use AWS CloudShell, a browser-based CLI, to run commands. Learn more
- Use the AWS CLI V2 and enable authentication through a user in IAM Identity Center. Learn more

Confirmation

[x] I understand the above recommendation and want to proceed to create an access key.

Cancel | Next

4.Demo File Created for Terraform test

```
ubuntu@ip-172-31-39-183:~$ mkdir demo
ubuntu@ip-172-31-39-183:~$ cd demo
ubuntu@ip-172-31-39-183:~/demo$ ec2.tf
ec2.tf: command not found
ubuntu@ip-172-31-39-183:~/demo$ nano ec2.tf
ubuntu@ip-172-31-39-183:~/demo$
```

5.Test File Script

```
  GNU nano 6.2                                                    ec2.tf *
resource "local_file" "example" {
        filename = "/home/ubuntu/myfile.txt"
        content = "Hello..!! \nThis is a local file created by terraform."
}
```

6.Terraform initiated in Created directory

```
ubuntu@ip-172-31-39-183:~/demo$ terraform init

Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/local...
- Installing hashicorp/local v2.5.1...
- Installed hashicorp/local v2.5.1 (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.
ubuntu@ip-172-31-39-183:~/demo$
```

7.Terrafrom Applied and Done

```
ubuntu@ip-172-31-39-183:~/demo$ terraform apply

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:

  # local_file.example will be created
  + resource "local_file" "example" {
      + content              = <<-EOT
            Hello..!!
            This is a local file created by terraform.
        EOT
      + content_base64sha256 = (known after apply)
      + content_base64sha512 = (known after apply)
      + content_md5          = (known after apply)
      + content_sha1         = (known after apply)
      + content_sha256       = (known after apply)
      + content_sha512       = (known after apply)
      + directory_permission = "0777"
      + file_permission      = "0777"
      + filename             = "/home/ubuntu/myfile.txt"
      + id                   = (known after apply)
    }

Plan: 1 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

local_file.example: Creating...
local_file.example: Creation complete after 0s [id=3e406fbffb29f44a4ac40b430826976030984372]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
ubuntu@ip-172-31-39-183:~/demo$
```

8.Demo file Tested with Output

```
ubuntu@ip-172-31-39-183:~/demo$ ls -ltr
total 8
-rw-rw-r-- 1 ubuntu ubuntu  143 May  7 16:31 ec2.tf
-rw-rw-r-- 1 ubuntu ubuntu 1640 May  7 16:33 terraform.tfstate
ubuntu@ip-172-31-39-183:~/demo$ cd ..
ubuntu@ip-172-31-39-183:~$ ls
aws  awscliv2.zip  demo  myfile.txt
ubuntu@ip-172-31-39-183:~$ cat myfile.txt
Hello..!!
This is a local file created by terraform.ubuntu@ip-172-31-39-183:~$
ubuntu@ip-172-31-39-183:~$
```

9.Creating File for EC2 Launch in two different regions

```
ubuntu@ip-172-31-39-183:~$ mkdir ec2
ubuntu@ip-172-31-39-183:~$ cd ec2/
ubuntu@ip-172-31-39-183:~/ec2$ nano ec2.tf
ubuntu@ip-172-31-39-183:~/ec2$
```

10.Terraform Script File for Two EC2 Creation

```
  GNU nano 6.2                                                    ec2.tf *
#Define AWS provider for the first region (ap-south-1) with an alias
provider "aws" {
 alias  = "ap_south_1"
 region = "ap-south-1"
}

provider "aws" {
 alias  = "us_east_1"
 region = "us-east-1"
}

resource "aws_instance" "aws1" {
  provider      = aws.ap_south_1
  ami           = "ami-013e83f579886baeb"
  instance_type = "t2.micro"
  tags = {
        Name = "aws1"
  }
}

resource "aws_instance" "aws2" {
  provider      = aws.us_east_1
  ami           = "ami-07caf09b362be10b8"
  instance_type = "t2.micro"
  tags = {
        Name = "aws2"
  }
}
```

11.Terraform Initialized in the directory

```
ubuntu@ip-172-31-39-183:~/ec2$ terraform init

Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v5.48.0...
- Installed hashicorp/aws v5.48.0 (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.
ubuntu@ip-172-31-39-183:~/ec2$
```

12.Terraform Plan check. EC2 in First Region

```
ubuntu@ip-172-31-39-183:~/ec2$ 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.aws1 will be created
  + resource "aws_instance" "aws1" {
      + ami                                  = "ami-013e83f579886baeb"
      + 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                        = "t2.micro"
```

13.EC2 Second Region Plan

```
 # aws_instance.aws2 will be created
 + resource "aws_instance" "aws2" {
     + ami                                  = "ami-07caf09b362be10b8"
     + 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                        = "t2.micro"
```

14.Terraform applied and EC2 First Region Creating

```
ubuntu@ip-172-31-39-183:~/ec2$ terraform apply

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.aws1 will be created
  + resource "aws_instance" "aws1" {
      + ami                                  = "ami-013e83f579886baeb"
      + 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                        = "t2.micro"
```

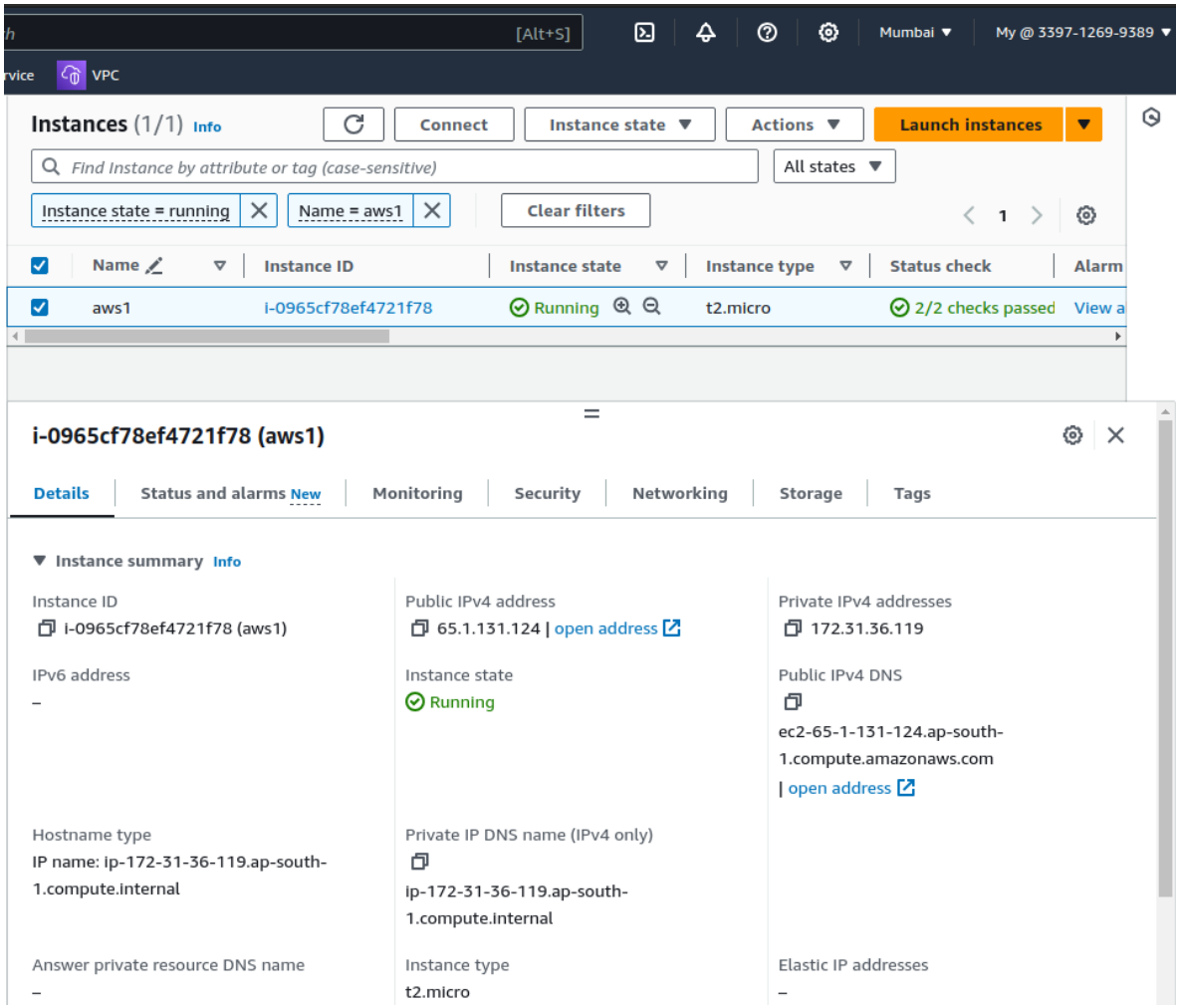
15.EC2 Second Region Creating

```
# aws_instance.aws2 will be created
+ resource "aws_instance" "aws2" {
    + ami                                  = "ami-07caf09b362be10b8"
    + 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                        = "t2.micro"
```

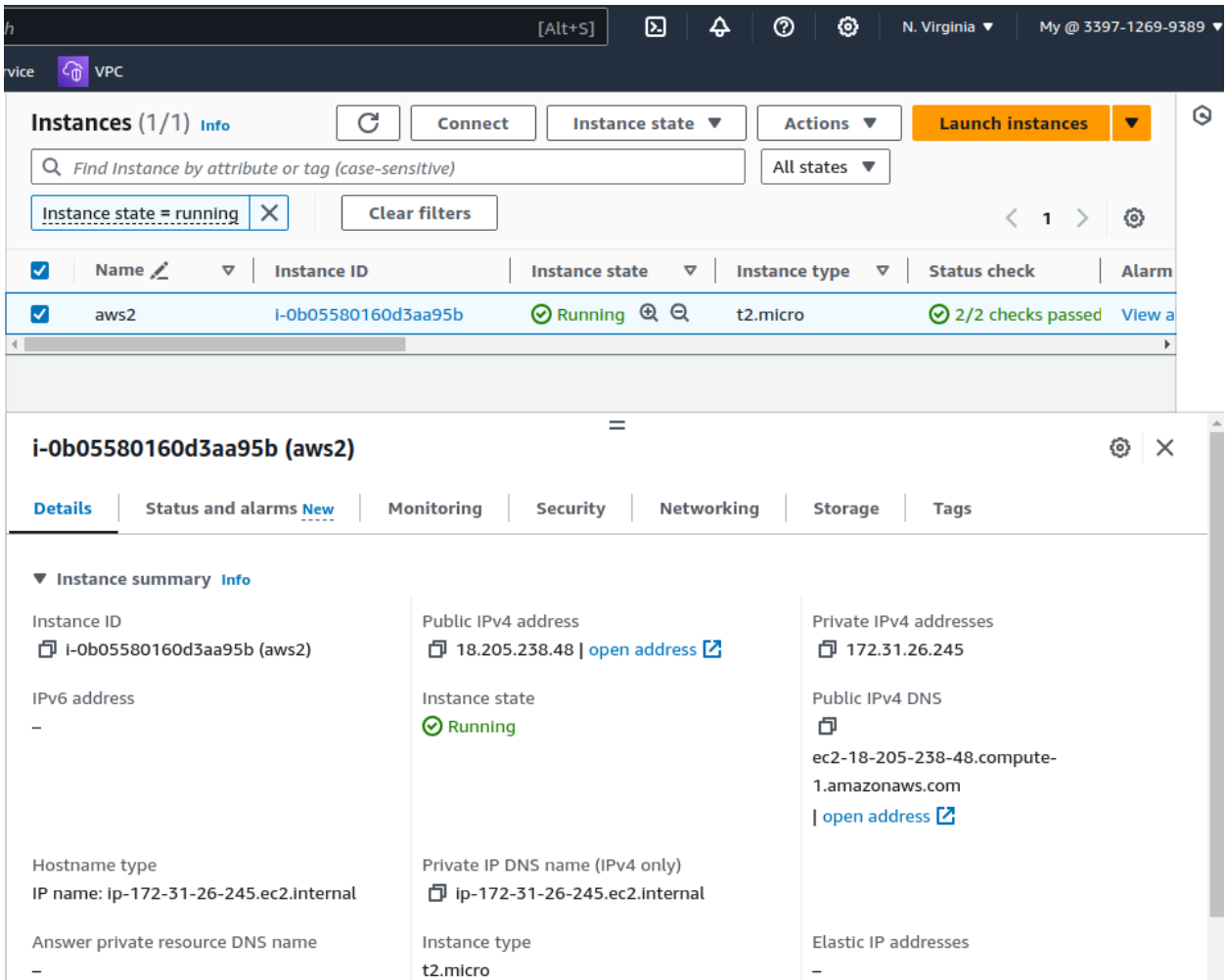
16.Two EC2 created in Two different Regions

```
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

aws_instance.aws1: Creating...
aws_instance.aws2: Creating...
aws_instance.aws1: Still creating... [10s elapsed]
aws_instance.aws2: Still creating... [10s elapsed]
aws_instance.aws1: Still creating... [20s elapsed]
aws_instance.aws2: Still creating... [20s elapsed]
aws_instance.aws1: Still creating... [30s elapsed]
aws_instance.aws2: Still creating... [30s elapsed]
aws_instance.aws1: Creation complete after 32s [id=i-0965cf78ef4721f78]
aws_instance.aws2: Creation complete after 35s [id=i-0b05580160d3aa95b]

Apply complete! Resources: 2 added, 0 changed, 0 destroyed.
ubuntu@ip-172-31-39-183:~/ec2$
```

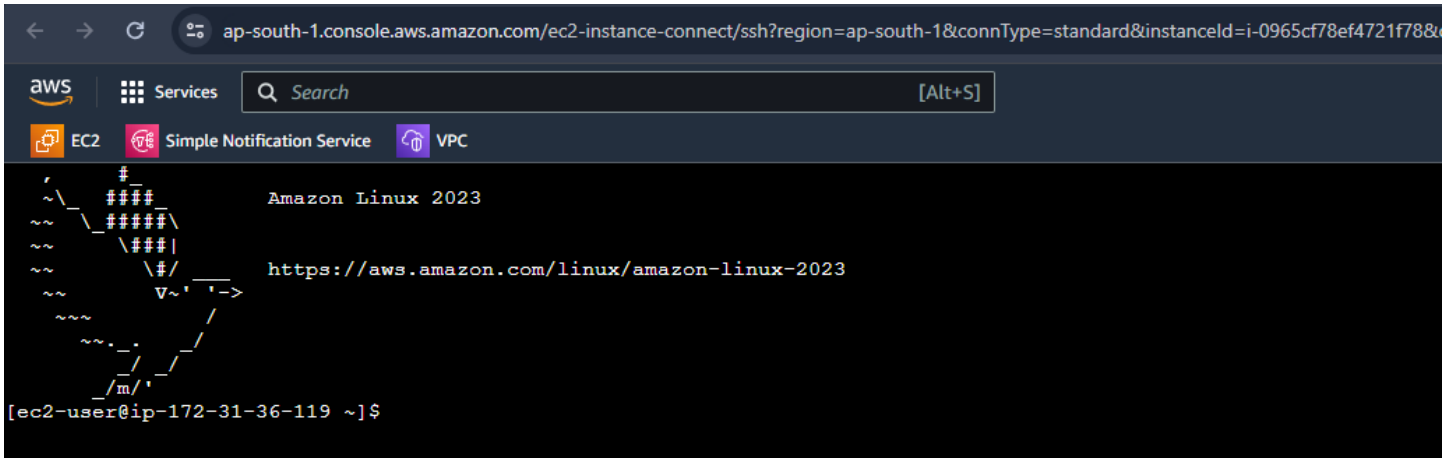
17.EC2 Created in First Region and Running



18.EC2 Created in Second Region and Running



19.EC2 Launched in First Region



20.EC2 Launched in Second Region