

TERRAFORM ASSIGNMENT-2

Name: Vikram

Assignment: Destroy Previous EC2 & Create New EC2 with Elastic IP

TASK 1: Destroy Previous Deployment

Go to the folder where Assignment 1 Terraform files exist:

```
cd terraform-ec2
```

Destroy previous resources:

```
terraform destroy
```

Type:

yes

Result:

Previous EC2 instance is completely removed.

```
ubuntu@ip-172-31-4-222:~/terraform-ec2$ terraform destroy
aws_instance.my_ec2: Refreshing state... [id=i-01b993b2abbc6632c]

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

Terraform will perform the following actions:

# aws_instance.my_ec2 will be destroyed
- resource "aws_instance" "my_ec2" {
  - ami              = "ami-0f5fcd140e4ab7" -> null
  - arn              = "arn:aws:ec2:us-east-2:062250062838:instance/i-01b993b2abbc6632c" -> null
  - associate_public_ip_address = true -> null
  - availability_zone = "us-east-2c" -> null
  - disable_api_stop  = false -> null
  - disable_api_termination = false -> null
  - ebs_optimized     = false -> null
  - force_destroy     = false -> null
  - get_password_data  = false -> null
  - hibernation        = false -> null
  - id                = "i-01b993b2abbc6632c" -> null

Plan: 0 to add, 0 to change, 1 to destroy.

Do you really want to destroy all resources?
  Terraform will destroy all your managed infrastructure, as shown above.
  There is no undo. Only 'yes' will be accepted to confirm.

Enter a value: yes

aws_instance.my_ec2: Destroying... [id=i-01b993b2abbc6632c]
aws_instance.my_ec2: Still destroying... [id=i-01b993b2abbc6632c, 00m10s elapsed]
aws_instance.my_ec2: Still destroying... [id=i-01b993b2abbc6632c, 00m20s elapsed]
aws_instance.my_ec2: Still destroying... [id=i-01b993b2abbc6632c, 00m30s elapsed]
aws_instance.my_ec2: Still destroying... [id=i-01b993b2abbc6632c, 00m40s elapsed]
aws_instance.my_ec2: Still destroying... [id=i-01b993b2abbc6632c, 00m50s elapsed]
aws_instance.my_ec2: Destruction complete after 1m0s

Destroy complete! Resources: 1 destroyed.
```

Instances (1/2) Info

Last updated
3 minutes ago

Connect

Instance state

Actions

Launch instances

Find Instance by attribute or tag (case-sensitive)

All states

< 1 >

| | Name | Instance ID | Instance state | Instance type | Status check | Alarm status | Availability Zone | Public IPv4 |
|-------------------------------------|---------------|---------------------|----------------|---------------|-------------------|--------------|-------------------|-------------|
| <input type="checkbox"/> | terraform | i-0ca0c7a99fa1c8500 | Running | t3.small | 3/3 checks passed | View alarms | us-east-2a | ec2-18-118 |
| <input checked="" type="checkbox"/> | Terraform-EC2 | i-01b993b2abb6632c | Terminated | t3.micro | - | View alarms | us-east-2c | - |

TASK 2: Create New EC2 with Elastic IP

Create New Project Folder

```
mkdir terraform-assignment2
```

```
cd terraform-assignment2
```

```
ubuntu@ip-172-31-4-222:~$ mkdir terraform-assignment2
cd terraform-assignment2
ubuntu@ip-172-31-4-222:~/terraform-assignment2$
```

Create Terraform Configuration File

```
nano main.tf
```

Paste this complete code:

```
provider "aws" {
    region = "us-east-2"
}

resource "aws_instance" "new_ec2" {
    ami          = "ami-0f5fcd140e4ab7"
    instance_type = "t3.micro"

    tags = {
        Name = "Terraform-EC2-With-EIP"
    }
}
```

```
resource "aws_eip" "elastic_ip" {  
    instance = aws_instance.new_ec2.id  
}
```

Save & exit.

```
ubuntu@ip-172-31-4-222:~$ mkdir terraform-assignment2  
cd terraform-assignment2  
ubuntu@ip-172-31-4-222:~/terraform-assignment2$ nano main.tf
```

```
GNU nano 7.2  
provider "aws" {  
    region = "us-east-2"  
}  
  
resource "aws_instance" "new_ec2" {  
    ami          = "ami-0f5fcdafb140e4ab7"  
    instance_type = "t3.micro"  
  
    tags = {  
        Name = "Terraform-EC2-With-EIP"  
    }  
}  
  
resource "aws_eip" "elastic_ip" {  
    instance = aws_instance.new_ec2.id  
}
```

Initialize Terraform

terraform init

```
ubuntu@ip-172-31-4-222:~/terraform-assignment2$ terraform init  
Initializing the backend...  
Initializing provider plugins...  
- Finding latest version of hashicorp/aws...  
- Installing hashicorp/aws v6.27.0...  
- Installed hashicorp/aws v6.27.0 (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-172-31-4-222:~/terraform-assignment2$
```

```
ubuntu@ip-172-31-4-222:~/terraform-assignment2$ terraform validate
Success! The configuration is valid.
```

```
ubuntu@ip-172-31-4-222:~/terraform-assignment2$
```

Apply Terraform Configuration

[terraform apply](#)

Type: yes

```
ubuntu@ip-172-31-4-222:~/terraform-assignment2$ 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_eip.elastic_ip will be created
+ resource "aws_eip" "elastic_ip" {
+   allocation_id      = (known after apply)
+   arn                 = (known after apply)
+   association_id     = (known after apply)
+   carrier_ip         = (known after apply)
+   customer_owned_ip  = (known after apply)
+   domain              = (known after apply)
+   id                  = (known after apply)
+   instance            = (known after apply)
+   ipam_pool_id        = (known after apply)
+   network_border_group = (known after apply)
+   network_interface   = (known after apply)
+   private_dns         = (known after apply)
+   private_ip          = (known after apply)
+   ptr_record          = (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
```

```
aws_instance.new_ec2: Creating...
```

```
aws_instance.new_ec2: Still creating... [00m10s elapsed]
```

```
aws_instance.new_ec2: Creation complete after 12s [id=i-043705974a254af64]
```

```
aws_eip.elastic_ip: Creating...
```

```
aws_eip.elastic_ip: Creation complete after 2s [id=eipalloc-086410904857b2574]
```

```
Apply complete! Resources: 2 added, 0 changed, 0 destroyed.
```

```
ubuntu@ip-172-31-4-222:~/terraform-assignment2$
```

```
i-0ca0c7a99fa1c8500 (terraform)
```

Verification

Check EC2 instance:

[terraform show](#)

```
ubuntu@ip-172-31-4-222:~/terraform-assignment2$ terraform show
# aws_eip.elastic_ip:
resource "aws_eip" "elastic_ip" {
  allocation_id      = "eipalloc-086410904857b2574"
  arn                 = "arn:aws:ec2:us-east-2:062250062838:elastic-ip/eipalloc-086410904857b2574"
  association_id      = "eipassoc-03b01e5ee1c55c470"
  carrier_ip          = null
  customer_owned_ip   = null
  customer_owned_ipv4_pool = null
  domain              = "vpc"
  id                  = "eipalloc-086410904857b2574"
  instance            = "i-043705974a254af64"
  network_border_group = "us-east-2"
  network_interface    = "eni-006e43a7728927b75"
  private_dns          = "ip-172-31-42-40.us-east-2.compute.internal"
  private_ip           = "172.31.42.40"
  ptr_record           = null
  public_dns           = "ec2-52-14-109-253.us-east-2.compute.amazonaws.com"
  public_ip            = "52.14.109.253"
  public_ipv4_pool     = "amazon"
  region              = "us-east-2"
  tags_all             = {}
}

# aws_instance.new_ec2:
resource "aws_instance" "new_ec2" {
  ami = "ami-0f5fcd9b140e4ab7"
}
```

Go to AWS Console → EC2 → Instances

You will see:

- ✓ New EC2 instance
- ✓ Elastic IP attached to the instance

The screenshot shows the AWS Management Console interface. At the top, there's a search bar and navigation tabs. The main content area is titled 'Instances (1/2)' and shows a table of EC2 instances. The table has columns for Name, Instance ID, Instance state, Instance type, Status check, Alarm status, Availability Zone, and Public IPv4. Two instances are listed: 'terraform' (Running, t3.small) and 'Terraform-EC2-With-EIP' (Initializing, t3.micro). The details for 'Terraform-EC2-With-EIP' are expanded, showing its private IP, public IP, and Elastic IP address.

| Name | Instance ID | Instance state | Instance type | Status check | Alarm status | Availability Zone | Public IPv4 |
|------------------------|---------------------|----------------|---------------|-------------------|--------------|-------------------|-------------|
| terraform | i-0ca0c7a99fa1c8500 | Running | t3.small | 3/3 checks passed | View alarms | us-east-2a | ec2-18-118 |
| Terraform-EC2-With-EIP | i-043705974a254af64 | Initializing | t3.micro | Initializing | View alarms | us-east-2c | ec2-52-14-1 |

i-043705974a254af64 (Terraform-EC2-With-EIP)

| | | |
|---|--|--|
| Hostname type IP name: ip-172-31-42-40.us-east-2.compute.internal | Private IP DNS name (IPv4 only) ip-172-31-42-40.us-east-2.compute.internal | Elastic IP addresses 52.14.109.253 [Public IP] |
|---|--|--|

Conclusion

Successfully:

- ✓ Destroyed previous deployment
- ✓ Created new EC2 instance
- ✓ Allocated & attached Elastic IP using Terraform