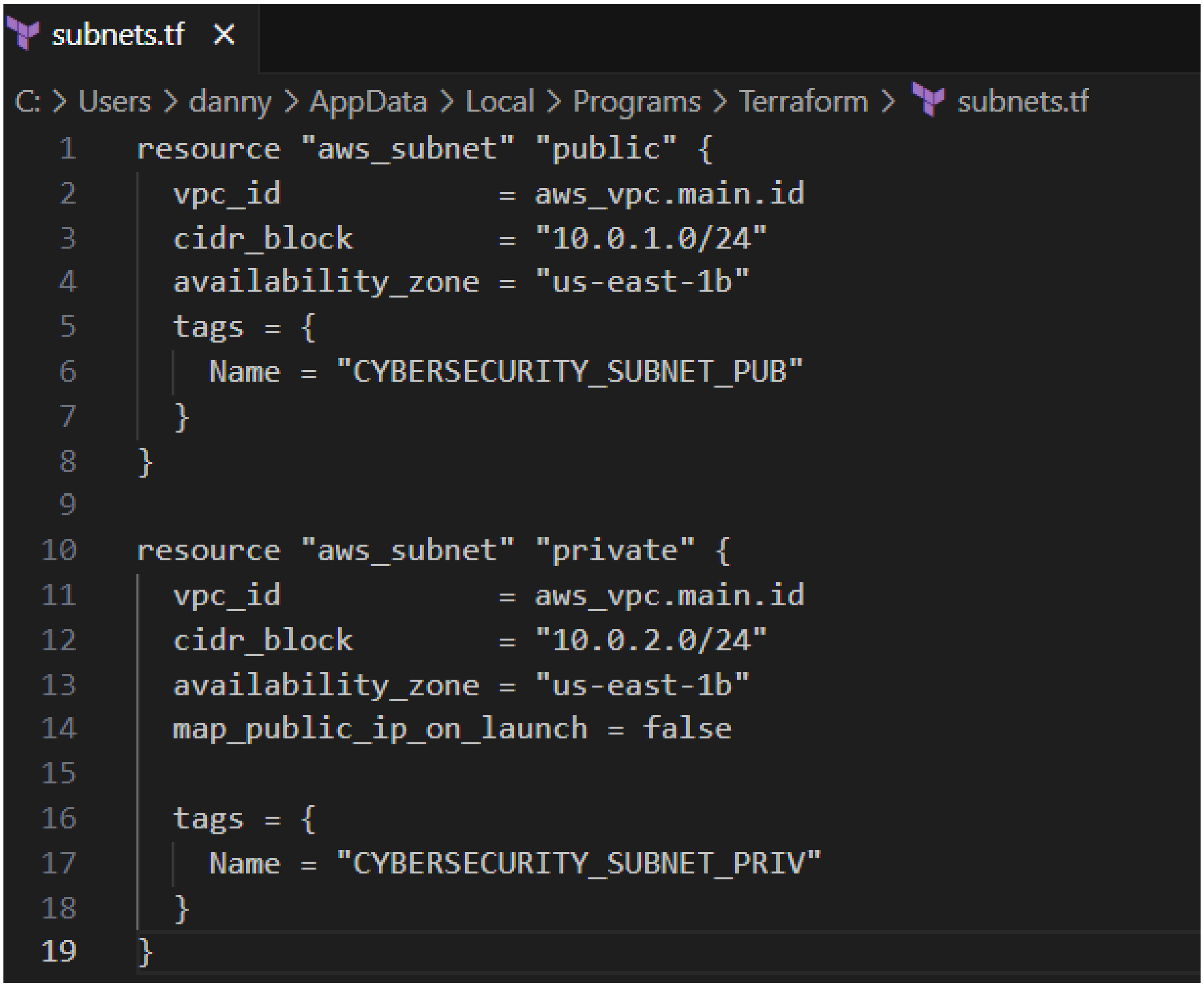
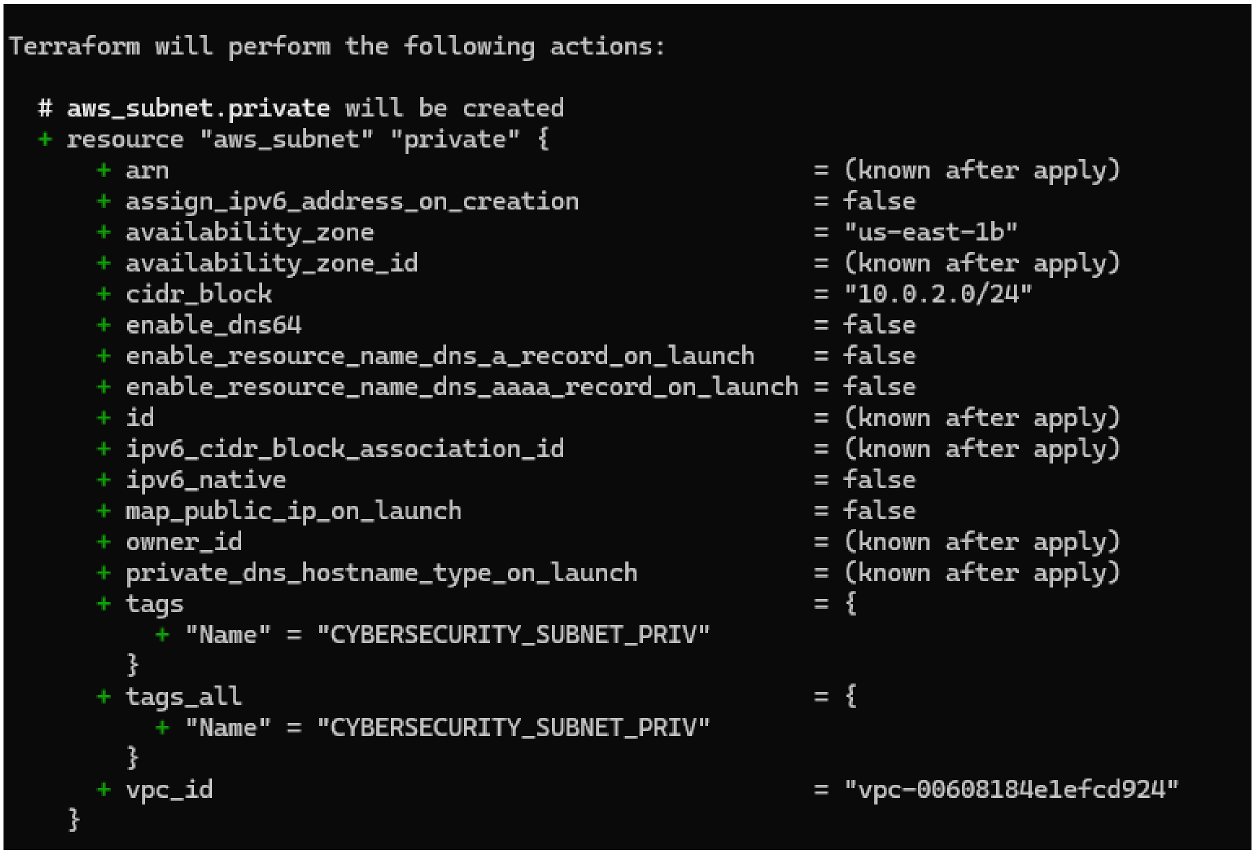
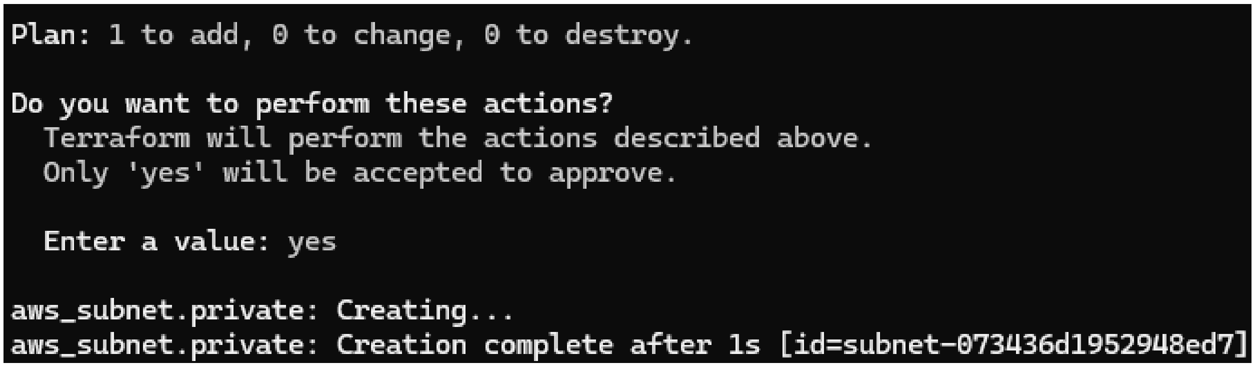
Task 1


subnets.tf
C: > Users > danny > AppData > Local > Programs > Terraform > subnets.tf

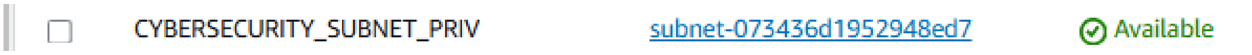
```
resource "aws_subnet" "public" {
  vpc_id            = aws_vpc.main.id
  cidr_block        = "10.0.1.0/24"
  availability_zone = "us-east-1b"
  tags = {
    Name = "CYBERSECURITY_SUBNET_PUB"
  }
}

resource "aws_subnet" "private" {
  vpc_id            = aws_vpc.main.id
  cidr_block        = "10.0.2.0/24"
  availability_zone = "us-east-1b"
  map_public_ip_on_launch = false

  tags = {
    Name = "CYBERSECURITY_SUBNET_PRIV"
  }
}
```



```
Terraform will perform the following actions:

  # aws_subnet.private will be created
  + resource "aws_subnet" "private" {
      + arn                                            = (known after apply)
      + assign_ipv6_address_on_creation                = false
      + availability_zone                              = "us-east-1b"
      + availability_zone_id                           = (known after apply)
      + cidr_block                                     = "10.0.2.0/24"
      + enable_dns64                                   = false
      + enable_resource_name_dns_a_record_on_launch    = false
      + enable_resource_name_dns_aaaa_record_on_launch = false
      + id                                             = (known after apply)
      + ipv6_cidr_block_association_id                 = (known after apply)
      + ipv6_native                                    = false
      + map_public_ip_on_launch                        = false
      + owner_id                                       = (known after apply)
      + private_dns_hostname_type_on_launch            = (known after apply)
      + tags                                           = {
          + "Name" = "CYBERSECURITY_SUBNET_PRIV"
        }
      + tags_all                                       = {
          + "Name" = "CYBERSECURITY_SUBNET_PRIV"
        }
      + vpc_id                                         = "vpc-00608184e1efcd924"
    }
```

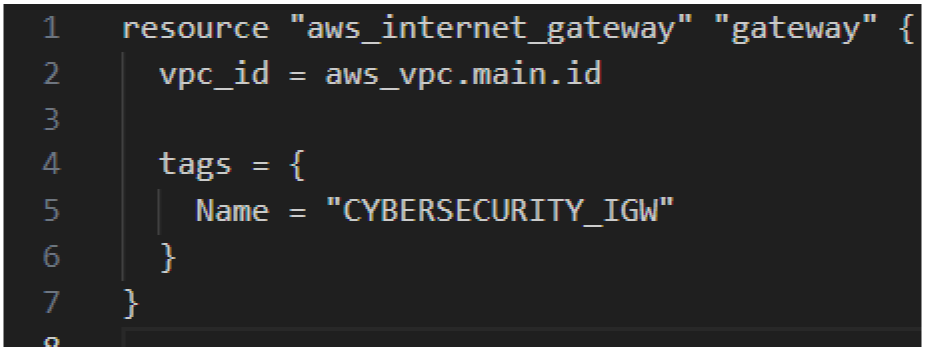


```
Plan: 1 to add, 0 to change, 0 to destroy.

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

aws_subnet.private: Creating...
aws_subnet.private: Creation complete after 1s [id=subnet-073436d1952948ed7]
```

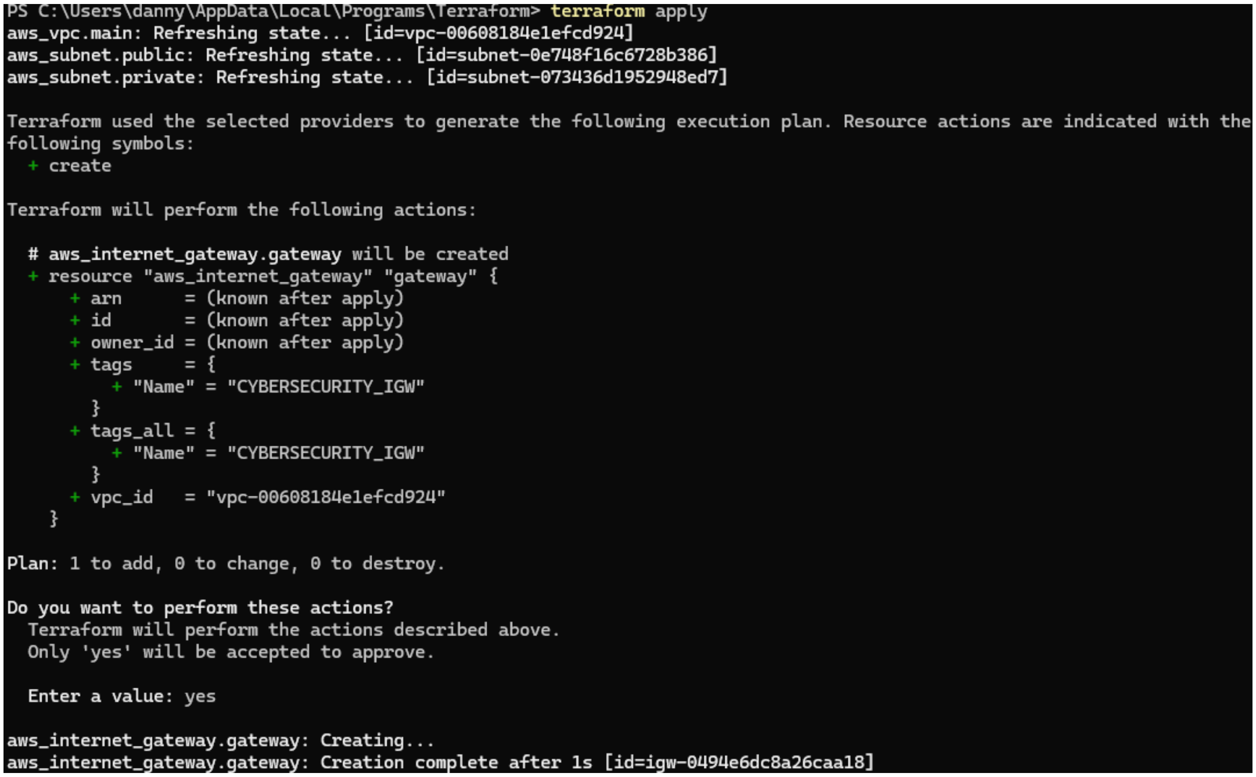


□ CYBERSECURITY_SUBNET_PRIV subnet-073436d1952948ed7 ✓ Available

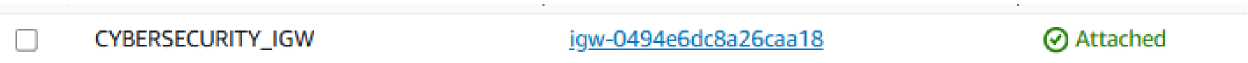
Task 2



```
resource "aws_internet_gateway" "gateway" {
  vpc_id = aws_vpc.main.id

  tags = {
    Name = "CYBERSECURITY_IGW"
  }
}
```



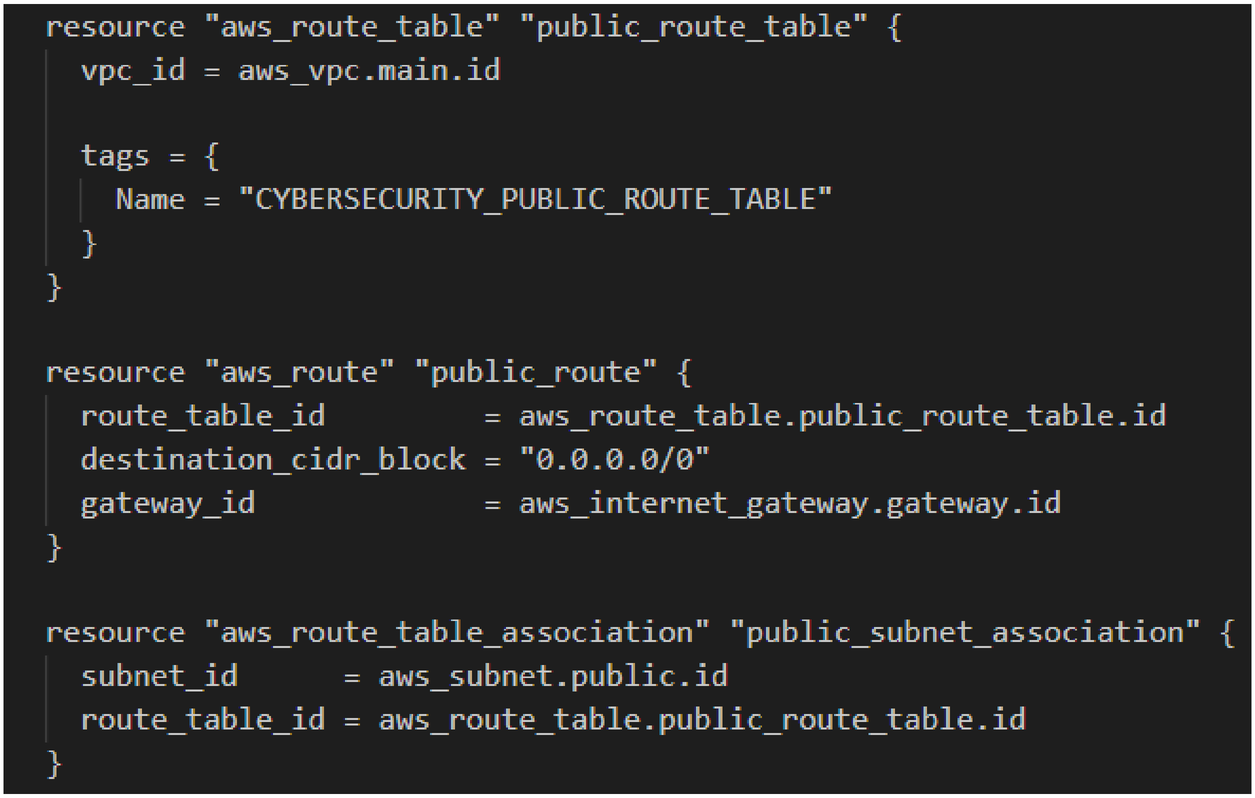
```
PS C:\Users\danny\AppData\Local\Programs\Terraform> terraform apply
aws_vpc.main: Refreshing state... [id=vpc-00608184e1efcd924]
aws_subnet.public: Refreshing state... [id=subnet-0e748f16c6728b386]
aws_subnet.private: Refreshing state... [id=subnet-073436d1952948ed7]

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_internet_gateway.gateway will be created
  + resource "aws_internet_gateway" "gateway" {
      + arn      = (known after apply)
      + id       = (known after apply)
      + owner_id = (known after apply)
      + tags     = {
          + "Name" = "CYBERSECURITY_IGW"
        }
      + tags_all = {
          + "Name" = "CYBERSECURITY_IGW"
        }
      + vpc_id   = "vpc-00608184e1efcd924"
    }

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

aws_internet_gateway.gateway: Creating...
aws_internet_gateway.gateway: Creation complete after 1s [id=igw-0494e6dc8a26caa18]
```



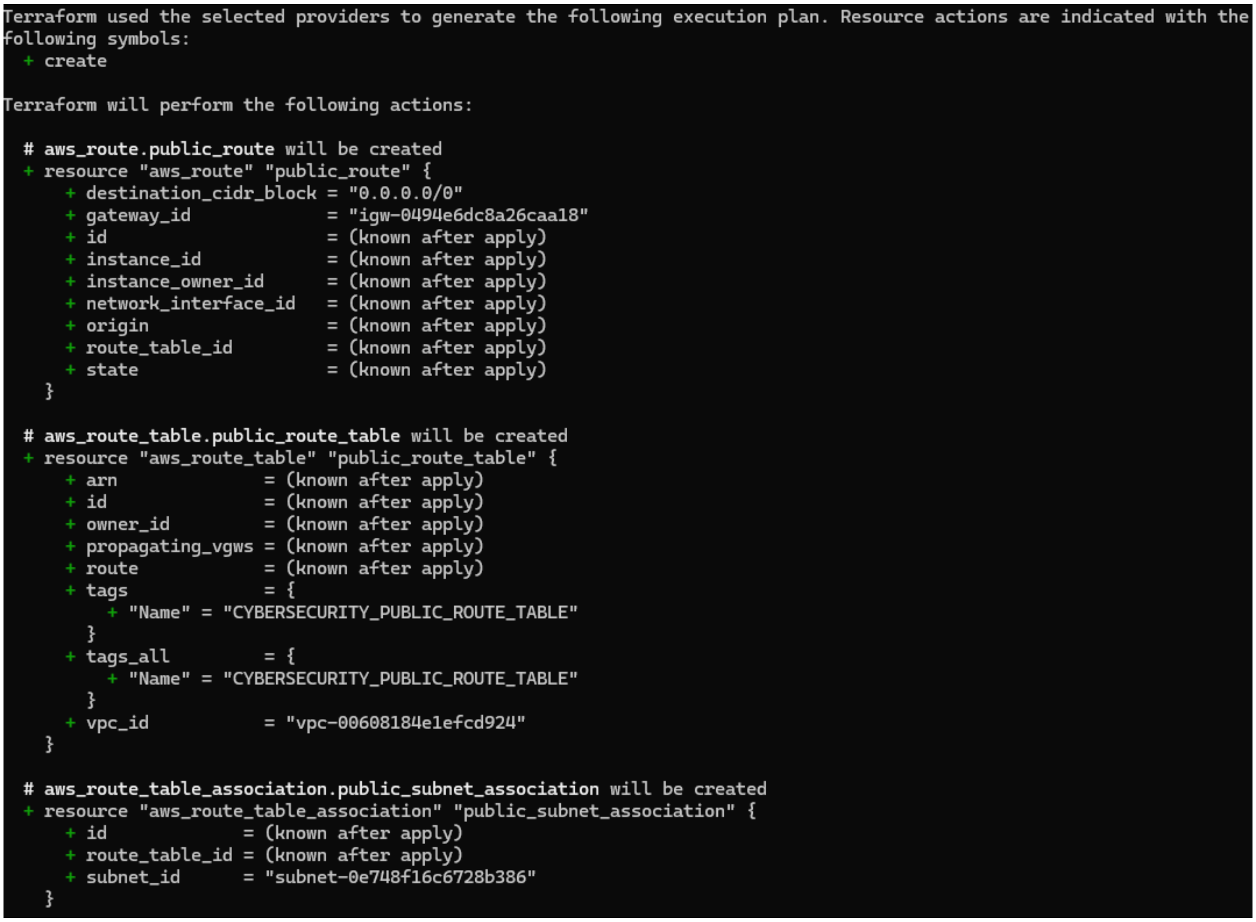
☐ CYBERSECURITY_IGW igw-0494e6dc8a26caa18 ✓ Attached

Task3

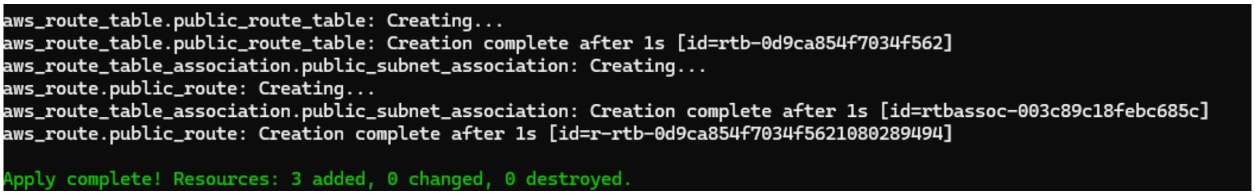
Define the route table for the Internet Gateway and associate the route table with the public subnet



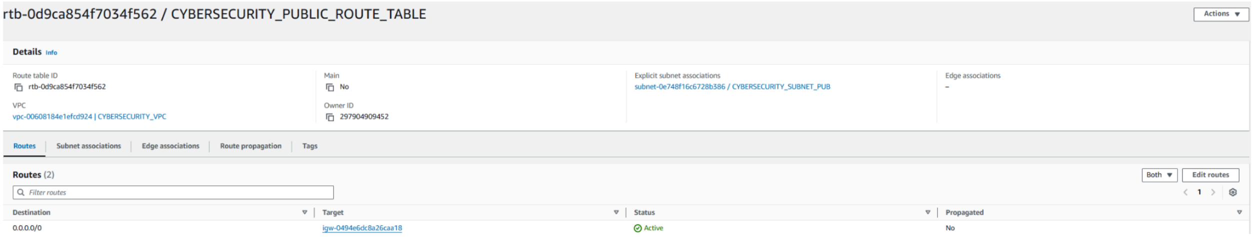
```
resource "aws_route_table" "public_route_table" {
  vpc_id = aws_vpc.main.id

  tags = {
    Name = "CYBERSECURITY_PUBLIC_ROUTE_TABLE"
  }
}

resource "aws_route" "public_route" {
  route_table_id         = aws_route_table.public_route_table.id
  destination_cidr_block = "0.0.0.0/0"
  gateway_id             = aws_internet_gateway.gateway.id
}

resource "aws_route_table_association" "public_subnet_association" {
  subnet_id      = aws_subnet.public.id
  route_table_id = aws_route_table.public_route_table.id
}
```



```
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_route.public_route will be created
  + resource "aws_route" "public_route" {
      + destination_cidr_block = "0.0.0.0/0"
      + gateway_id             = "igw-0494e6dc8a26caa18"
      + id                     = (known after apply)
      + instance_id            = (known after apply)
      + instance_owner_id      = (known after apply)
      + network_interface_id   = (known after apply)
      + origin                 = (known after apply)
      + route_table_id         = (known after apply)
      + state                  = (known after apply)
    }

  # aws_route_table.public_route_table will be created
  + resource "aws_route_table" "public_route_table" {
      + arn              = (known after apply)
      + id               = (known after apply)
      + owner_id         = (known after apply)
      + propagating_vgws = (known after apply)
      + route            = (known after apply)
      + tags             = {
          + "Name" = "CYBERSECURITY_PUBLIC_ROUTE_TABLE"
        }
      + tags_all         = {
          + "Name" = "CYBERSECURITY_PUBLIC_ROUTE_TABLE"
        }
      + vpc_id           = "vpc-00608184e1efcd924"
    }

  # aws_route_table_association.public_subnet_association will be created
  + resource "aws_route_table_association" "public_subnet_association" {
      + id             = (known after apply)
      + route_table_id = (known after apply)
      + subnet_id      = "subnet-0e748f16c6728b386"
    }
```



```
aws_route_table.public_route_table: Creating...
aws_route_table.public_route_table: Creation complete after 1s [id=rtb-0d9ca854f7034f562]
aws_route_table_association.public_subnet_association: Creating...
aws_route.public_route: Creating...
aws_route_table_association.public_subnet_association: Creation complete after 1s [id=rtbassoc-003c89c18febc685c]
aws_route.public_route: Creation complete after 1s [id=r-rtb-0d9ca854f7034f5621080289494]

Apply complete! Resources: 3 added, 0 changed, 0 destroyed.
```



rtb-0d9ca854f7034f562 / CYBERSECURITY_PUBLIC_ROUTE_TABLE

Details Info

Route table ID rtb-0d9ca854f7034f562	Main No	Explicit subnet associations subnet-0e748f16c6728b386 / CYBERSECURITY_SUBNET_PUB	Edge associations –
VPC vpc-00608184e1efcd924 | CYBERSECURITY_VPC	Owner ID 297904909452		

Routes | Subnet associations | Edge associations | Route propagation | Tags

Routes (2)

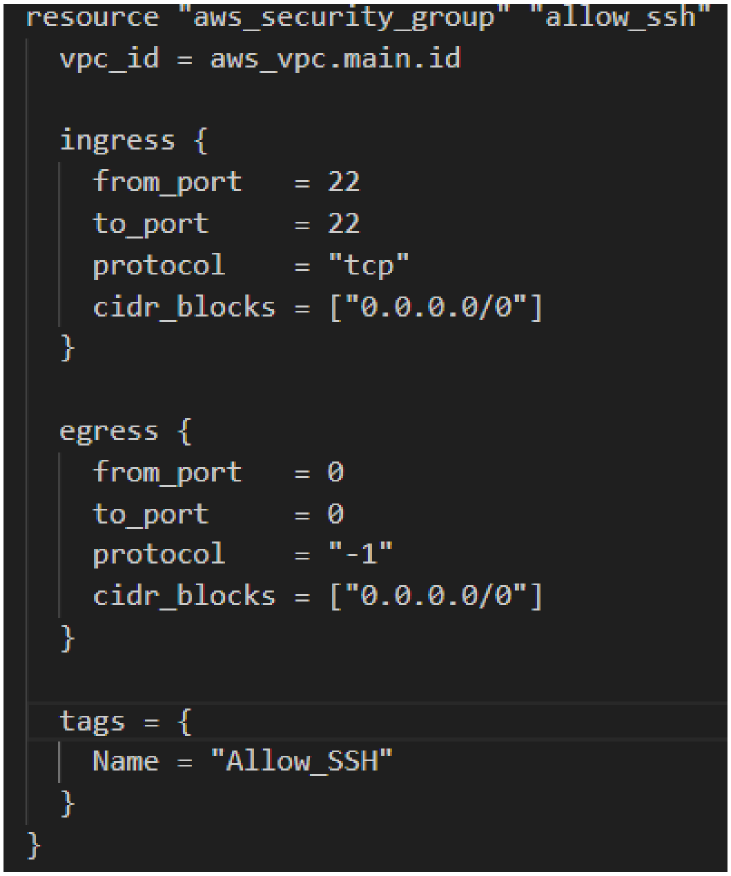
Filter routes
Both ▼
Edit routes
1

Destination	Target	Status	Propagated
0.0.0.0/0	igw-0494e6dc8a26caa18	Active	No

Task 4

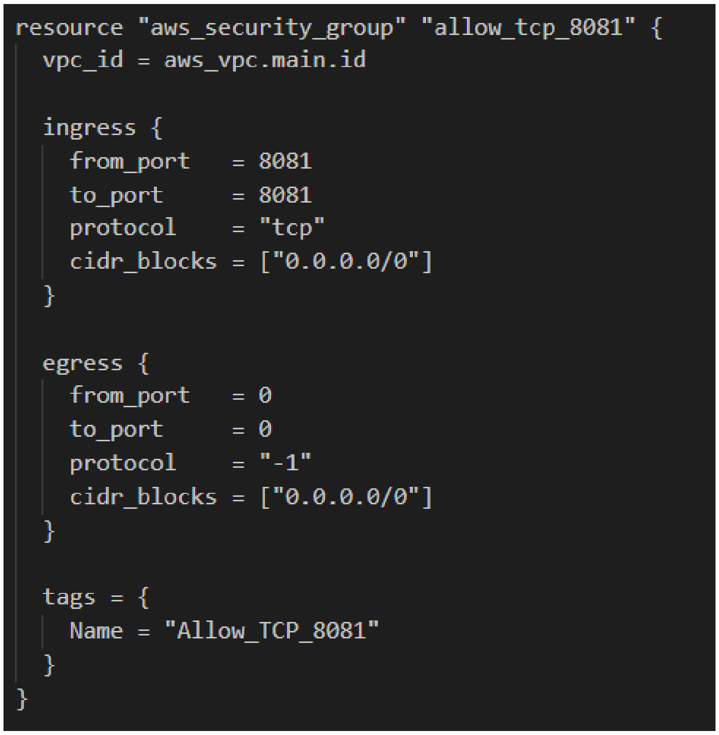
3 different security groups were built

1. Allow SSH



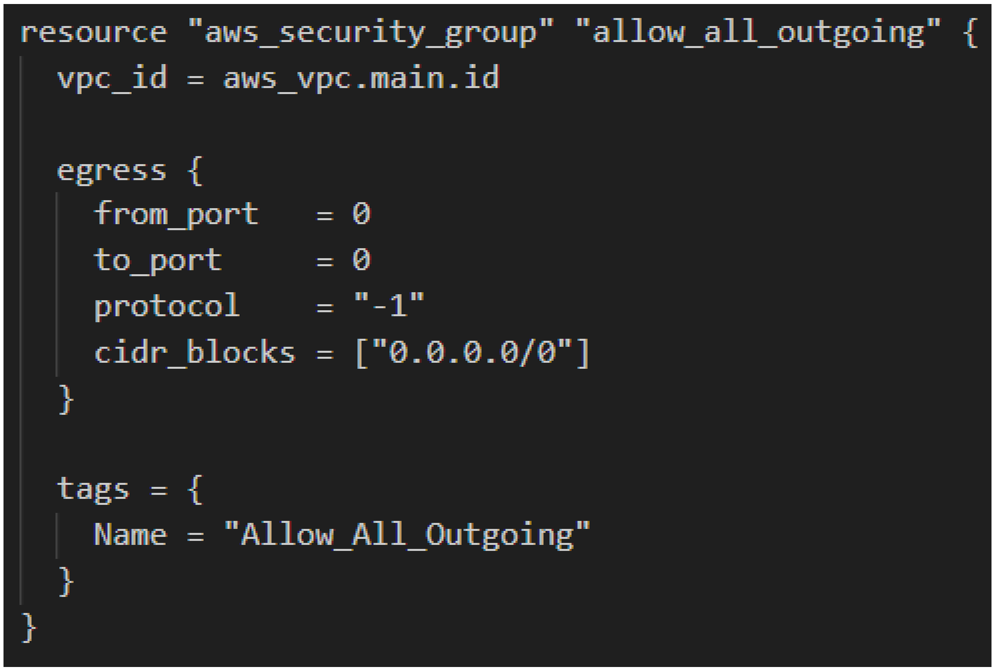
```
resource "aws_security_group" "allow_ssh"
  vpc_id = aws_vpc.main.id

  ingress {
    from_port   = 22
    to_port     = 22
    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"]
  }

  tags = {
    Name = "Allow_SSH"
  }
}
```

2. Allow TCP on port 8081

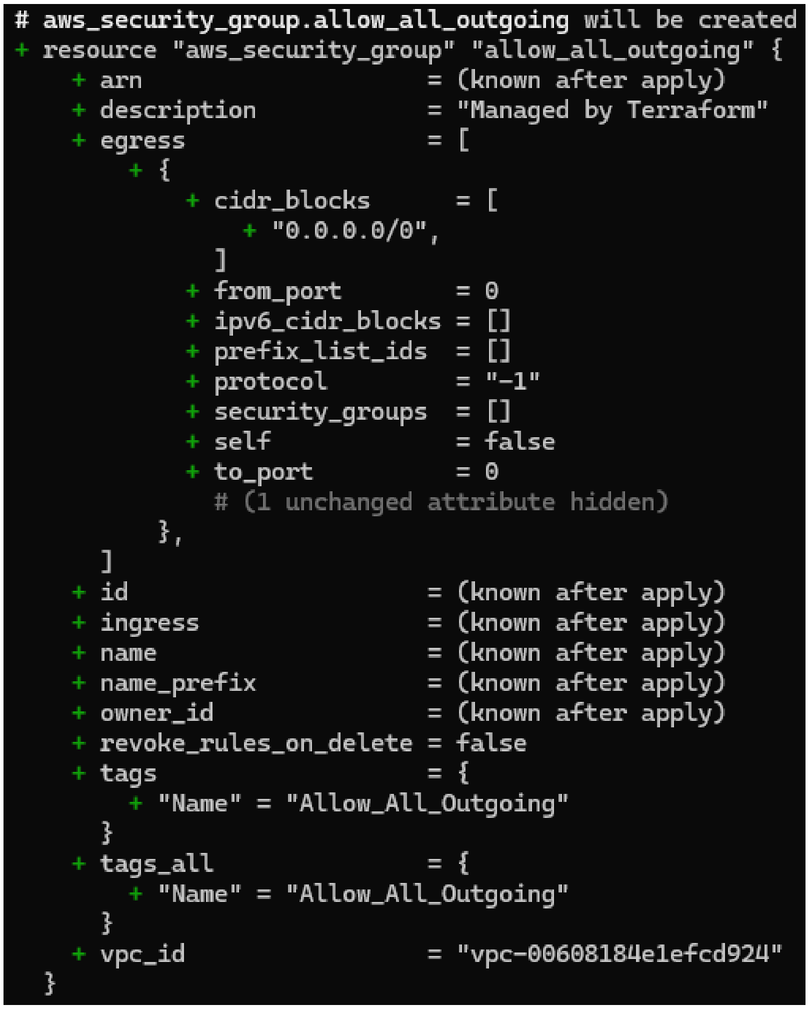


```
resource "aws_security_group" "allow_tcp_8081" {
  vpc_id = aws_vpc.main.id

  ingress {
    from_port   = 8081
    to_port     = 8081
    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"]
  }

  tags = {
    Name = "Allow_TCP_8081"
  }
}
```

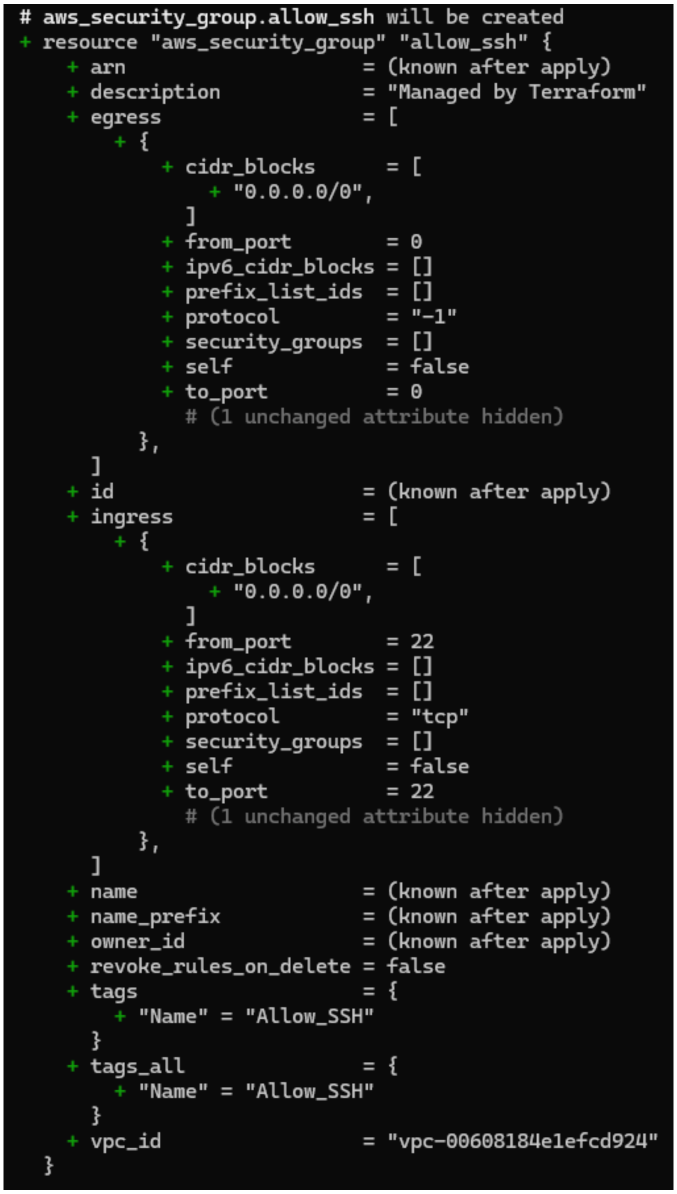
3. Allow All Outgoing Traffic



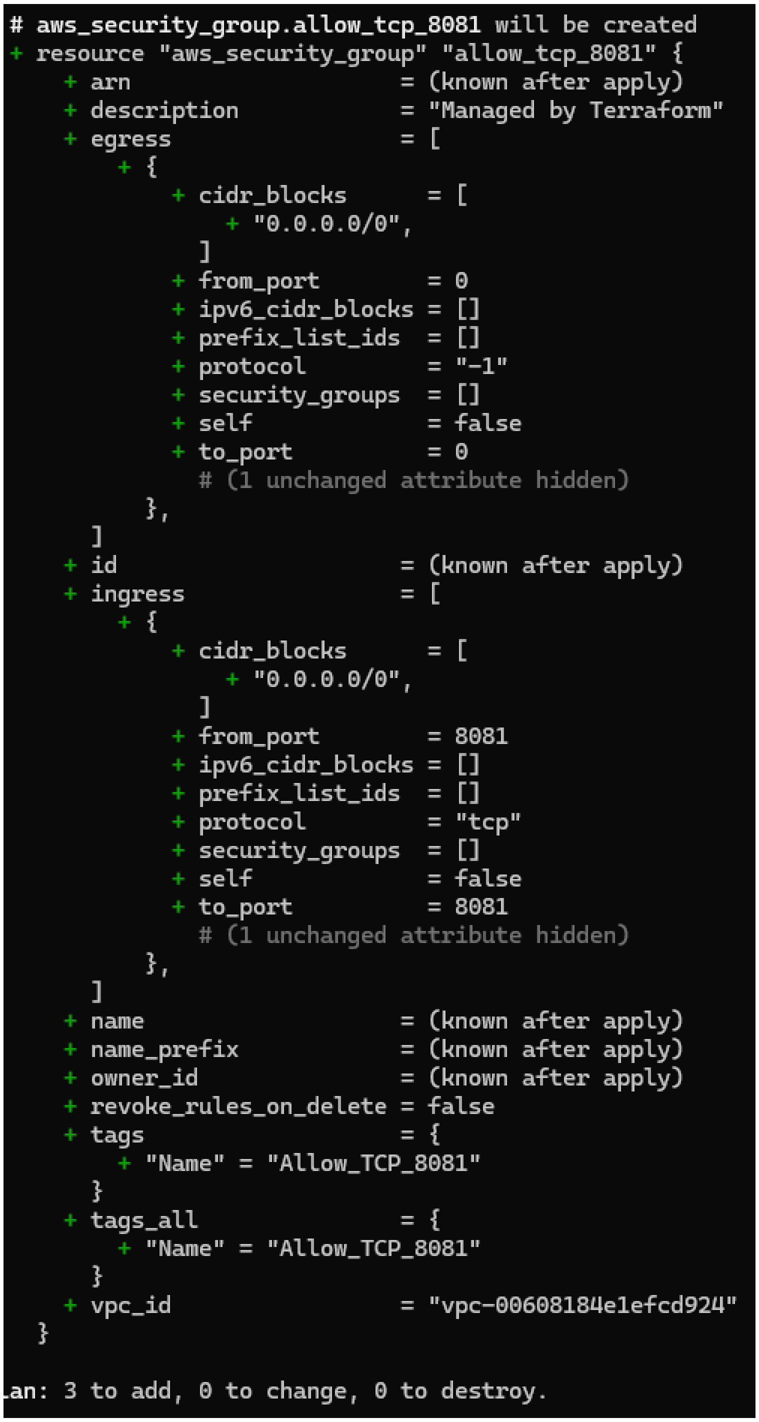
```
resource "aws_security_group" "allow_all_outgoing" {
  vpc_id = aws_vpc.main.id

  egress {
    from_port   = 0
    to_port     = 0
    protocol    = "-1"
    cidr_blocks = ["0.0.0.0/0"]
  }

  tags = {
    Name = "Allow_All_Outgoing"
  }
}
```



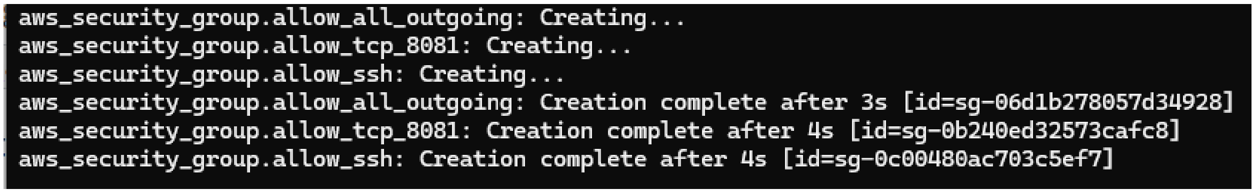
```
# aws_security_group.allow_all_outgoing will be created
+ resource "aws_security_group" "allow_all_outgoing" {
    + arn                    = (known after apply)
    + description            = "Managed by Terraform"
    + egress                 = [
        + {
            + cidr_blocks      = [
                + "0.0.0.0/0",
              ]
            + from_port        = 0
            + ipv6_cidr_blocks = []
            + prefix_list_ids  = []
            + protocol         = "-1"
            + security_groups  = []
            + self             = false
            + to_port          = 0
              # (1 unchanged attribute hidden)
          },
      ]
    + id                     = (known after apply)
    + ingress                = (known after apply)
    + name                   = (known after apply)
    + name_prefix            = (known after apply)
    + owner_id               = (known after apply)
    + revoke_rules_on_delete = false
    + tags                   = {
        + "Name" = "Allow_All_Outgoing"
      }
    + tags_all               = {
        + "Name" = "Allow_All_Outgoing"
      }
    + vpc_id                 = "vpc-00608184e1efcd924"
  }
```



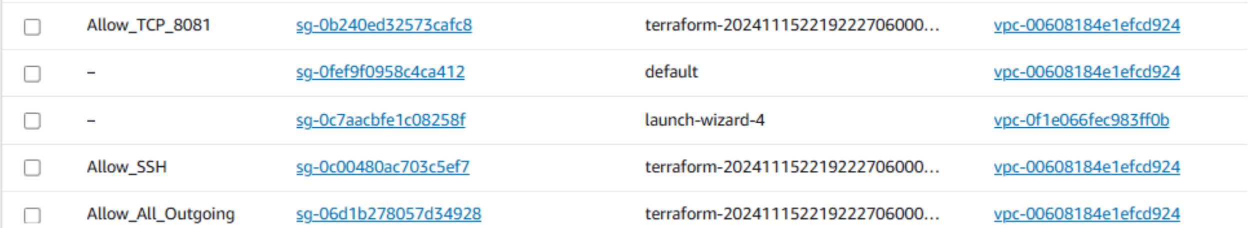
```
# aws_security_group.allow_ssh will be created
+ resource "aws_security_group" "allow_ssh" {
    + arn                    = (known after apply)
    + description            = "Managed by Terraform"
    + egress                 = [
        + {
            + cidr_blocks      = [
                + "0.0.0.0/0",
              ]
            + from_port        = 0
            + ipv6_cidr_blocks = []
            + prefix_list_ids  = []
            + protocol         = "-1"
            + security_groups  = []
            + self             = false
            + to_port          = 0
              # (1 unchanged attribute hidden)
          },
      ]
    + id                     = (known after apply)
    + ingress                = [
        + {
            + cidr_blocks      = [
                + "0.0.0.0/0",
              ]
            + from_port        = 22
            + ipv6_cidr_blocks = []
            + prefix_list_ids  = []
            + protocol         = "tcp"
            + security_groups  = []
            + self             = false
            + to_port          = 22
              # (1 unchanged attribute hidden)
          },
      ]
    + name                   = (known after apply)
    + name_prefix            = (known after apply)
    + owner_id               = (known after apply)
    + revoke_rules_on_delete = false
    + tags                   = {
        + "Name" = "Allow_SSH"
      }
    + tags_all               = {
        + "Name" = "Allow_SSH"
      }
    + vpc_id                 = "vpc-00608184e1efcd924"
  }
```



```
# aws_security_group.allow_tcp_8081 will be created
+ resource "aws_security_group" "allow_tcp_8081" {
    + arn                    = (known after apply)
    + description            = "Managed by Terraform"
    + egress                 = [
        + {
            + cidr_blocks      = [
                + "0.0.0.0/0",
              ]
            + from_port        = 0
            + ipv6_cidr_blocks = []
            + prefix_list_ids  = []
            + protocol         = "-1"
            + security_groups  = []
            + self             = false
            + to_port          = 0
              # (1 unchanged attribute hidden)
          },
      ]
    + id                     = (known after apply)
    + ingress                = [
        + {
            + cidr_blocks      = [
                + "0.0.0.0/0",
              ]
            + from_port        = 8081
            + ipv6_cidr_blocks = []
            + prefix_list_ids  = []
            + protocol         = "tcp"
            + security_groups  = []
            + self             = false
            + to_port          = 8081
              # (1 unchanged attribute hidden)
          },
      ]
    + name                   = (known after apply)
    + name_prefix            = (known after apply)
    + owner_id               = (known after apply)
    + revoke_rules_on_delete = false
    + tags                   = {
        + "Name" = "Allow_TCP_8081"
      }
    + tags_all               = {
        + "Name" = "Allow_TCP_8081"
      }
    + vpc_id                 = "vpc-00608184e1efcd924"
  }

an: 3 to add, 0 to change, 0 to destroy.
```



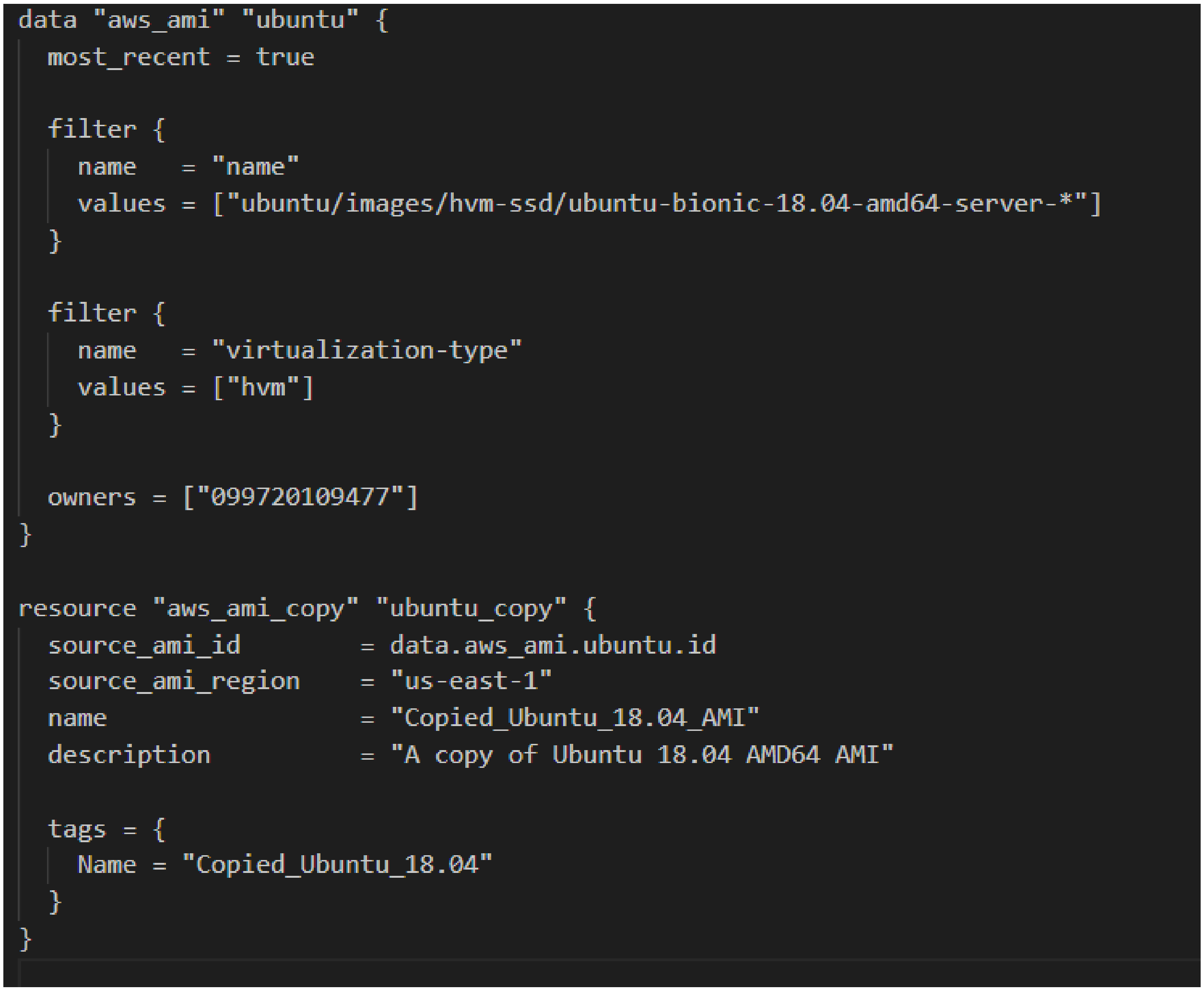
```
aws_security_group.allow_all_outgoing: Creating...
aws_security_group.allow_tcp_8081: Creating...
aws_security_group.allow_ssh: Creating...
aws_security_group.allow_all_outgoing: Creation complete after 3s [id=sg-06d1b278057d34928]
aws_security_group.allow_tcp_8081: Creation complete after 4s [id=sg-0b240ed32573cafc8]
aws_security_group.allow_ssh: Creation complete after 4s [id=sg-0c00480ac703c5ef7]
```



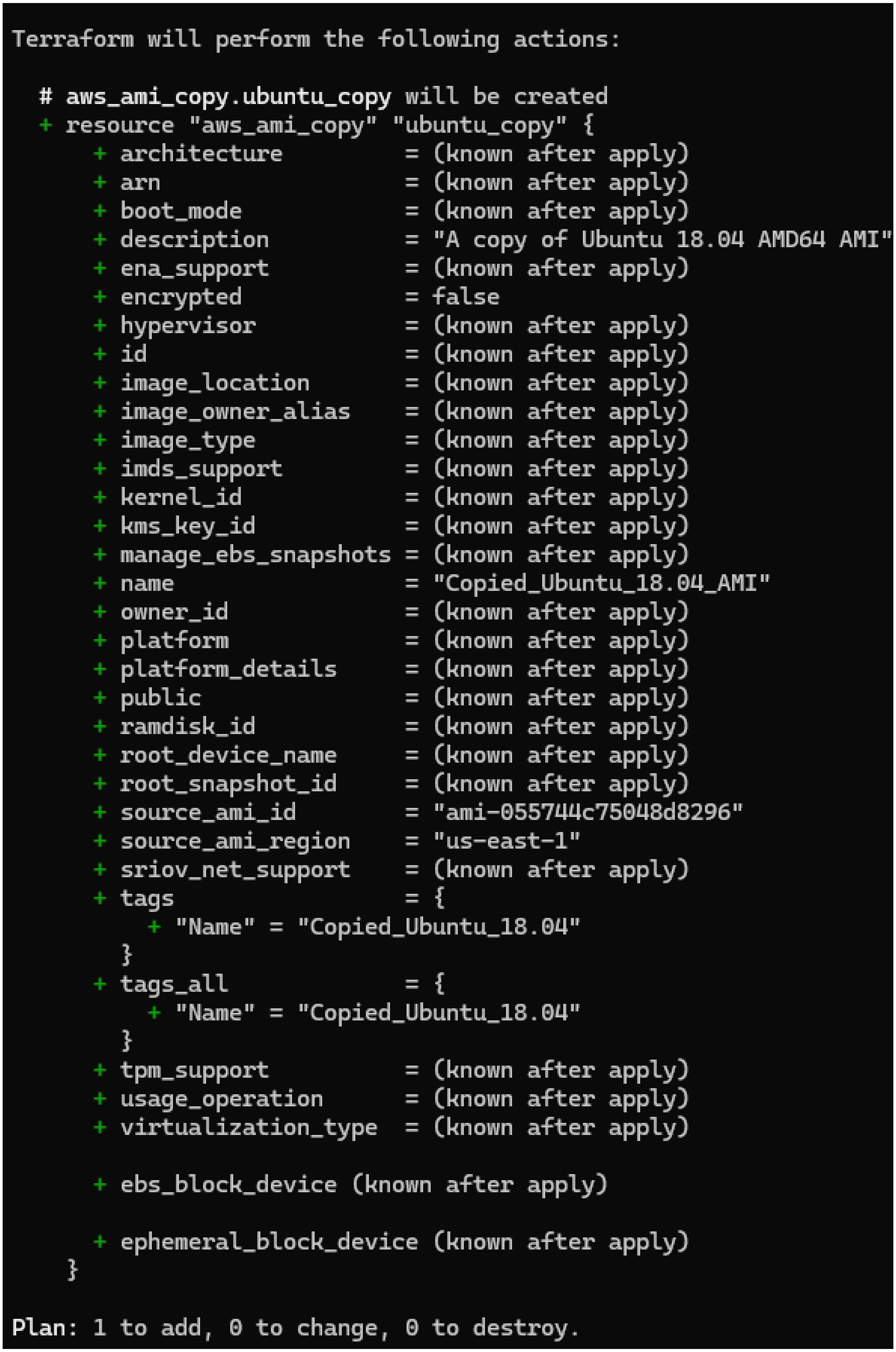
□	Allow_TCP_8081	sg-0b240ed32573cafc8	terraform-202411152219222706000...	vpc-00608184e1efcd924
□	–	sg-0fef9f0958c4ca412	default	vpc-00608184e1efcd924
□	–	sg-0c7aacbfe1c08258f	launch-wizard-4	vpc-0f1e066fec983ff0b
□	Allow_SSH	sg-0c00480ac703c5ef7	terraform-202411152219222706000...	vpc-00608184e1efcd924
□	Allow_All_Outgoing	sg-06d1b278057d34928	terraform-202411152219222706000...	vpc-00608184e1efcd924

Task 5

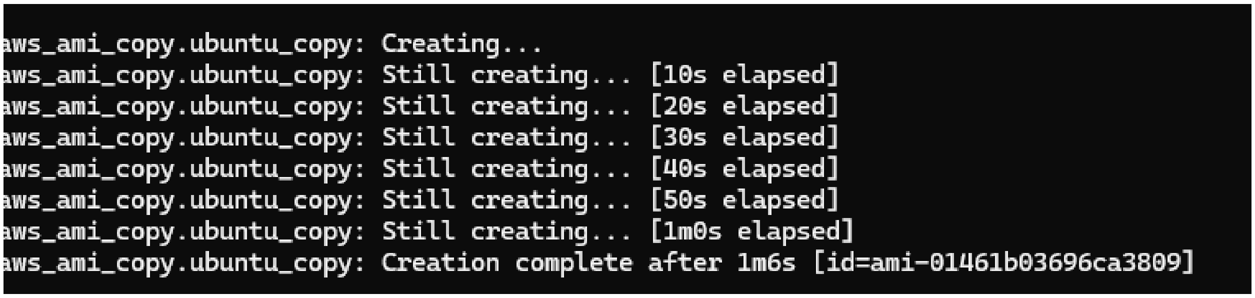
Use the aws_ami Data Source to Fetch the Ubuntu 18.04 AMI
The aws_ami data source helps to look up an existing AMI based on specific filters.
Then, copy the AMI fetched from the aws_ami data source:



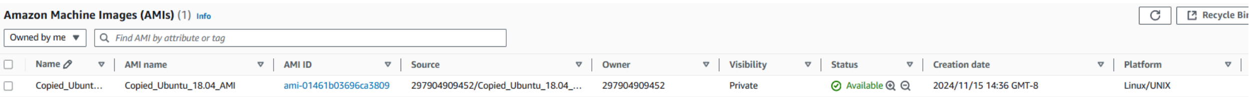
```
data "aws_ami" "ubuntu" {
  most_recent = true

  filter {
    name   = "name"
    values = ["ubuntu/images/hvm-ssd/ubuntu-bionic-18.04-amd64-server-*"]
  }

  filter {
    name   = "virtualization-type"
    values = ["hvm"]
  }

  owners = ["099720109477"]
}

resource "aws_ami_copy" "ubuntu_copy" {
  source_ami_id         = data.aws_ami.ubuntu.id
  source_ami_region     = "us-east-1"
  name                  = "Copied_Ubuntu_18.04_AMI"
  description           = "A copy of Ubuntu 18.04 AMD64 AMI"

  tags = {
    Name = "Copied_Ubuntu_18.04"
  }
}
```



```
Terraform will perform the following actions:

  # aws_ami_copy.ubuntu_copy will be created
  + resource "aws_ami_copy" "ubuntu_copy" {
      + architecture          = (known after apply)
      + arn                   = (known after apply)
      + boot_mode             = (known after apply)
      + description           = "A copy of Ubuntu 18.04 AMD64 AMI"
      + ena_support           = (known after apply)
      + encrypted             = false
      + hypervisor            = (known after apply)
      + id                    = (known after apply)
      + image_location        = (known after apply)
      + image_owner_alias     = (known after apply)
      + image_type            = (known after apply)
      + imds_support          = (known after apply)
      + kernel_id             = (known after apply)
      + kms_key_id            = (known after apply)
      + manage_ebs_snapshots  = (known after apply)
      + name                  = "Copied_Ubuntu_18.04_AMI"
      + owner_id              = (known after apply)
      + platform              = (known after apply)
      + platform_details      = (known after apply)
      + public                = (known after apply)
      + ramdisk_id            = (known after apply)
      + root_device_name      = (known after apply)
      + root_snapshot_id      = (known after apply)
      + source_ami_id         = "ami-055744c75048d8296"
      + source_ami_region     = "us-east-1"
      + sriov_net_support     = (known after apply)
      + tags                  = {
          + "Name" = "Copied_Ubuntu_18.04"
        }
      + tags_all              = {
          + "Name" = "Copied_Ubuntu_18.04"
        }
      + tpm_support           = (known after apply)
      + usage_operation       = (known after apply)
      + virtualization_type   = (known after apply)

      + ebs_block_device (known after apply)

      + ephemeral_block_device (known after apply)
    }

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



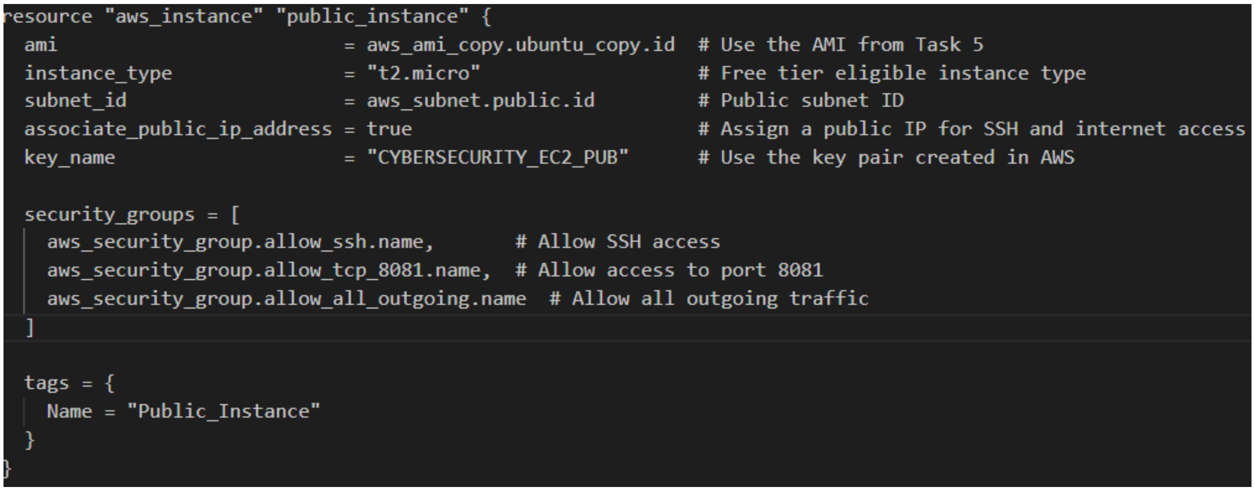
```
aws_ami_copy.ubuntu_copy: Creating...
aws_ami_copy.ubuntu_copy: Still creating... [10s elapsed]
aws_ami_copy.ubuntu_copy: Still creating... [20s elapsed]
aws_ami_copy.ubuntu_copy: Still creating... [30s elapsed]
aws_ami_copy.ubuntu_copy: Still creating... [40s elapsed]
aws_ami_copy.ubuntu_copy: Still creating... [50s elapsed]
aws_ami_copy.ubuntu_copy: Still creating... [1m0s elapsed]
aws_ami_copy.ubuntu_copy: Creation complete after 1m6s [id=ami-01461b03696ca3809]
```



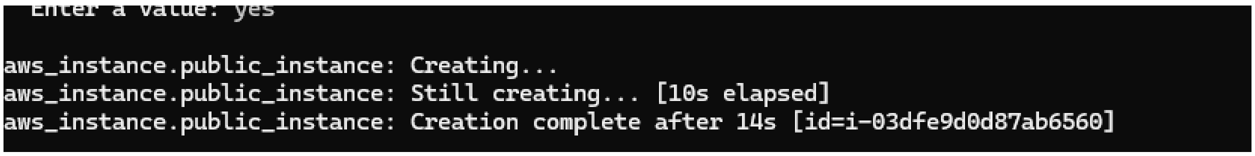
Amazon Machine Images (AMIs) (1) Info

□	Name	AMI name	AMI ID	Source	Owner	Visibility	Status	Creation date	Platform
□	Copied_Ubunt...	Copied_Ubuntu_18.04_AMI	ami-01461b03696ca3809	297904909452/Copied_Ubuntu_18.04_...	297904909452	Private	✓ Available	2024/11/15 14:36 GMT-8	Linux/UNIX

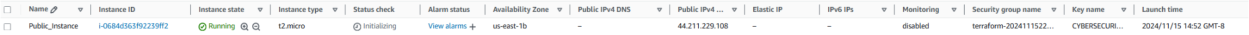
Task 6



```
resource "aws_instance" "public_instance" {
  ami                         = aws_ami_copy.ubuntu_copy.id  # Use the AMI from Task 5
  instance_type               = "t2.micro"                   # Free tier eligible instance type
  subnet_id                   = aws_subnet.public.id         # Public subnet ID
  associate_public_ip_address = true                         # Assign a public IP for SSH and internet access
  key_name                    = "CYBERSECURITY_EC2_PUB"      # Use the key pair created in AWS

  security_groups = [
    aws_security_group.allow_ssh.name,       # Allow SSH access
    aws_security_group.allow_tcp_8081.name,  # Allow access to port 8081
    aws_security_group.allow_all_outgoing.name  # Allow all outgoing traffic
  ]

  tags = {
    Name = "Public_Instance"
  }
}
```

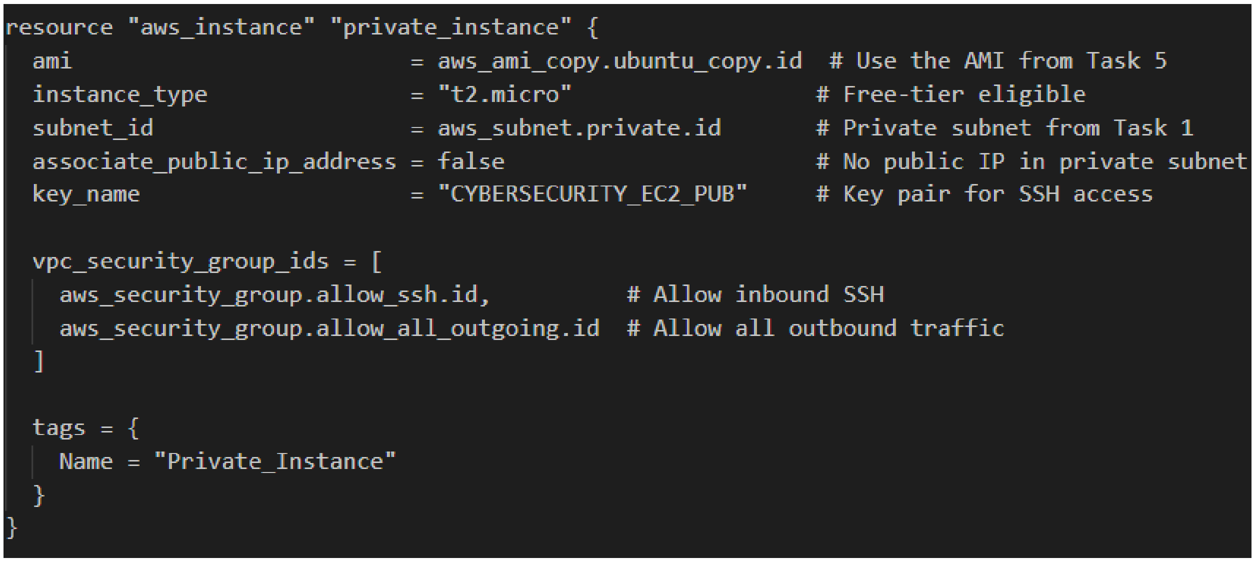


```
  Enter a value: yes

aws_instance.public_instance: Creating...
aws_instance.public_instance: Still creating... [10s elapsed]
aws_instance.public_instance: Creation complete after 14s [id=i-03dfe9d0d87ab6560]
```

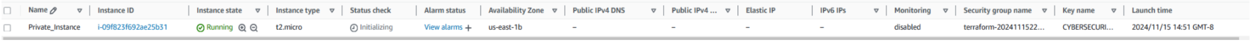


□	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elastic IP	IPv6 IPs	Monitoring	Security group name	Key name	Launch time
□	Public_Instance	i-0684d363f92239ff2	✓ Running	t2.micro	Initializing	View alarms +	us-east-1b	–	44.211.229.108	–	–	disabled	terraform-2024111522...	CYBERSECURI...	2024/11/15 14:52 GMT-8

Task 7



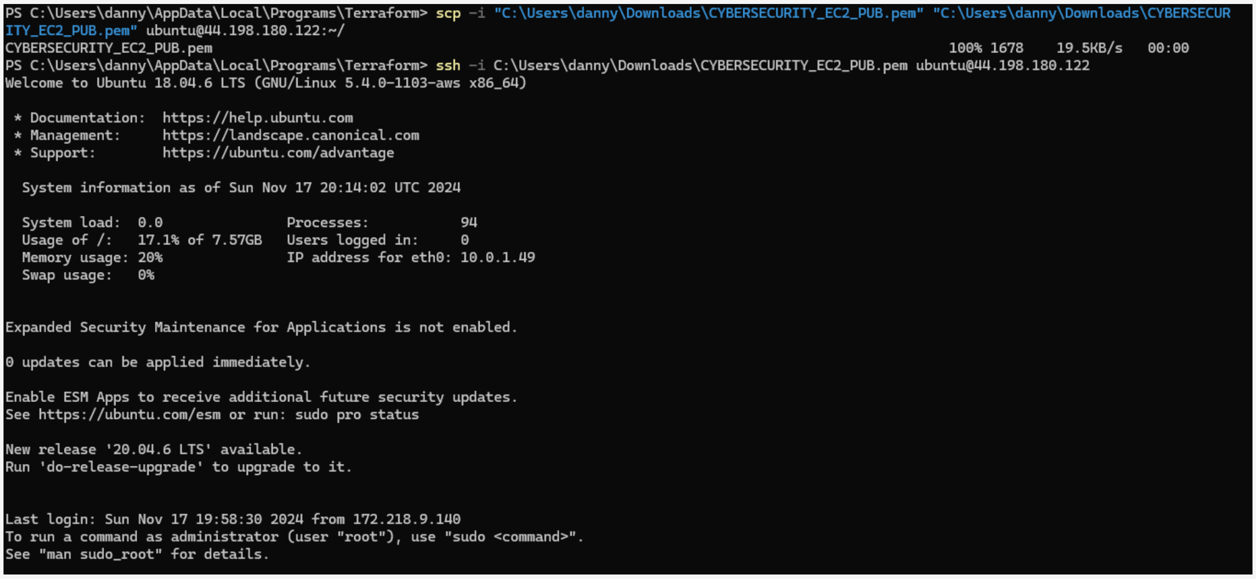
```
resource "aws_instance" "private_instance" {
  ami                         = aws_ami_copy.ubuntu_copy.id  # Use the AMI from Task 5
  instance_type               = "t2.micro"                  # Free-tier eligible
  subnet_id                   = aws_subnet.private.id       # Private subnet from Task 1
  associate_public_ip_address = false                       # No public IP in private subnet
  key_name                    = "CYBERSECURITY_EC2_PUB"     # Key pair for SSH access

  vpc_security_group_ids = [
    aws_security_group.allow_ssh.id,           # Allow inbound SSH
    aws_security_group.allow_all_outgoing.id  # Allow all outbound traffic
  ]

  tags = {
    Name = "Private_Instance"
  }
}
```



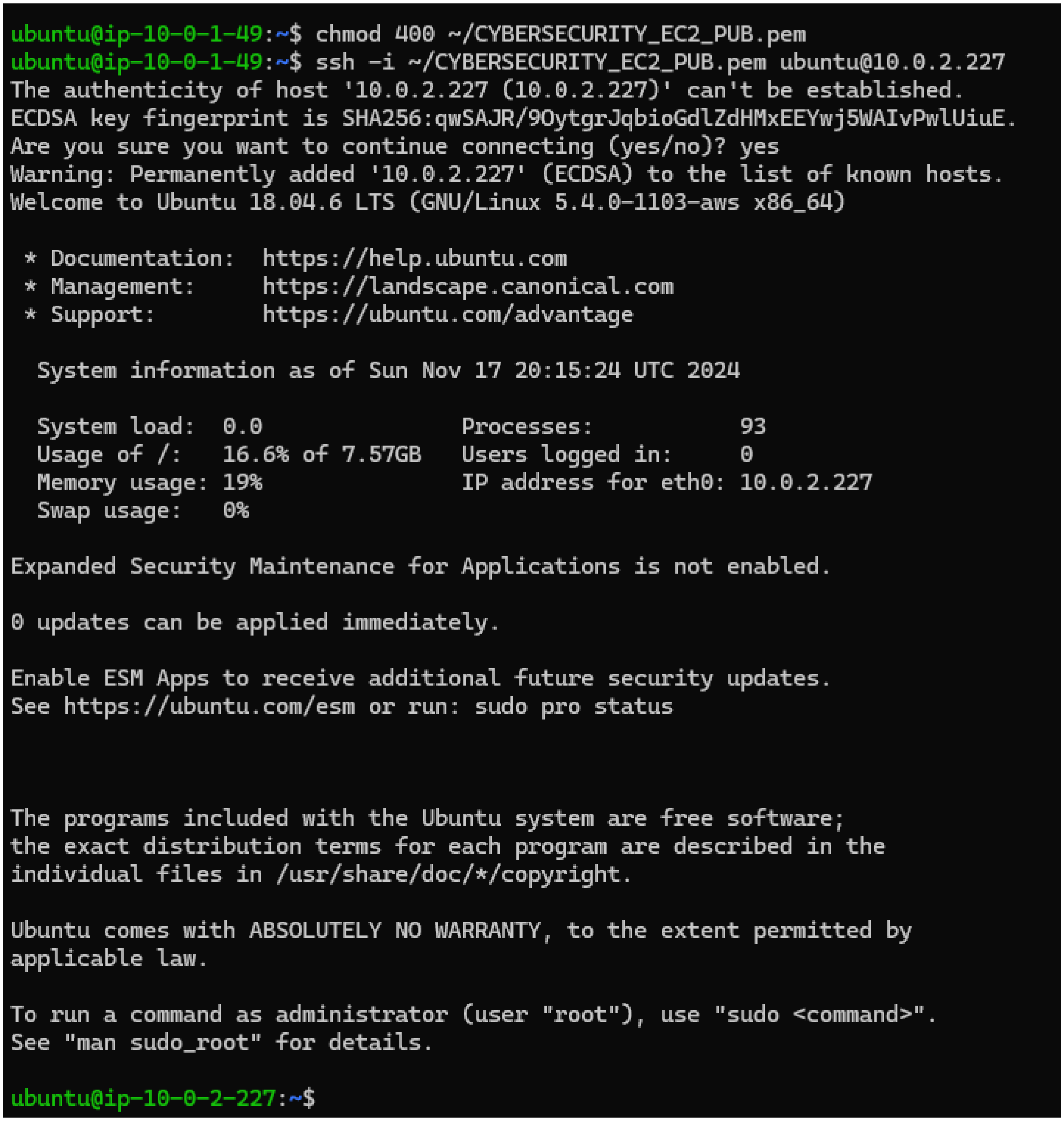
□	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elastic IP	IPv6 IPs	Monitoring	Security group name	Key name	Launch time
□	Private_Instance	i-09f823f692ae25b31	Running	t2.micro	Initializing	View alarms +	us-east-1b	–	–	–	–	disabled	terraform-2024111522...	CYBERSECURI...	2024/11/15 14:51 GMT-8

Task 8

Use Method 1 to connect the private instance through the public instance



```
PS C:\Users\danny\AppData\Local\Programs\Terraform> scp -i "C:\Users\danny\Downloads\CYBERSECURITY_EC2_PUB.pem" "C:\Users\danny\Downloads\CYBERSECUR
ITY_EC2_PUB.pem" ubuntu@44.198.180.122:~/
CYBERSECURITY_EC2_PUB.pem                                                                                     100% 1678    19.5KB/s   00:00
PS C:\Users\danny\AppData\Local\Programs\Terraform> ssh -i C:\Users\danny\Downloads\CYBERSECURITY_EC2_PUB.pem ubuntu@44.198.180.122
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 5.4.0-1103-aws x86_64)

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

  System information as of Sun Nov 17 20:14:02 UTC 2024

  System load:  0.0               Processes:           94
  Usage of /:   17.1% of 7.57GB   Users logged in:     0
  Memory usage: 20%               IP address for eth0: 10.0.1.49
  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

New release '20.04.6 LTS' available.
Run 'do-release-upgrade' to upgrade to it.


Last login: Sun Nov 17 19:58:30 2024 from 172.218.9.140
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.
```

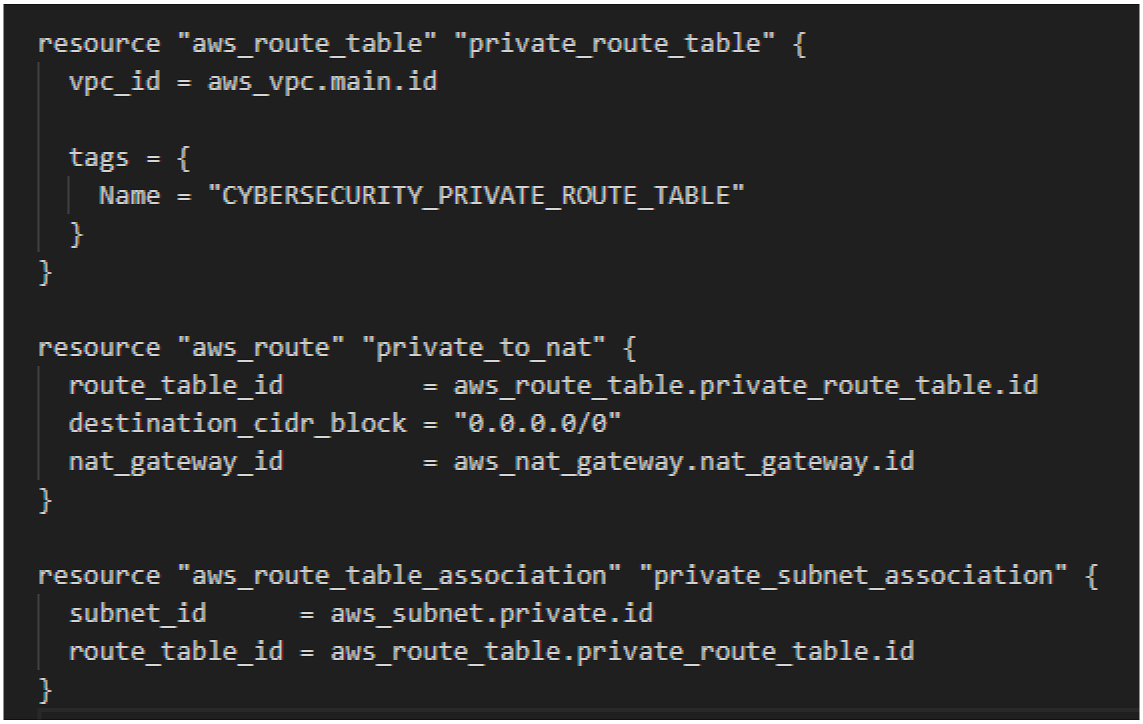


```
ubuntu@ip-10-0-1-49:~$ chmod 400 ~/CYBERSECURITY_EC2_PUB.pem
ubuntu@ip-10-0-1-49:~$ ssh -i ~/CYBERSECURITY_EC2_PUB.pem ubuntu@10.0.2.227
The authenticity of host '10.0.2.227 (10.0.2.227)' can't be established.
ECDSA key fingerprint is SHA256:qwSAJR/9OytgrJqbioGdlZdHMxEEYwj5WAIvPwlUiuE.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.0.2.227' (ECDSA) to the list of known hosts.
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 5.4.0-1103-aws x86_64)

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

  System information as of Sun Nov 17 20:15:24 UTC 2024

  System load:  0.0               Processes:           93
  Usage of /:   16.6% of 7.57GB   Users logged in:     0
  Memory usage: 19%               IP address for eth0: 10.0.2.227
  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 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.

ubuntu@ip-10-0-2-227:~$
```

Task 9

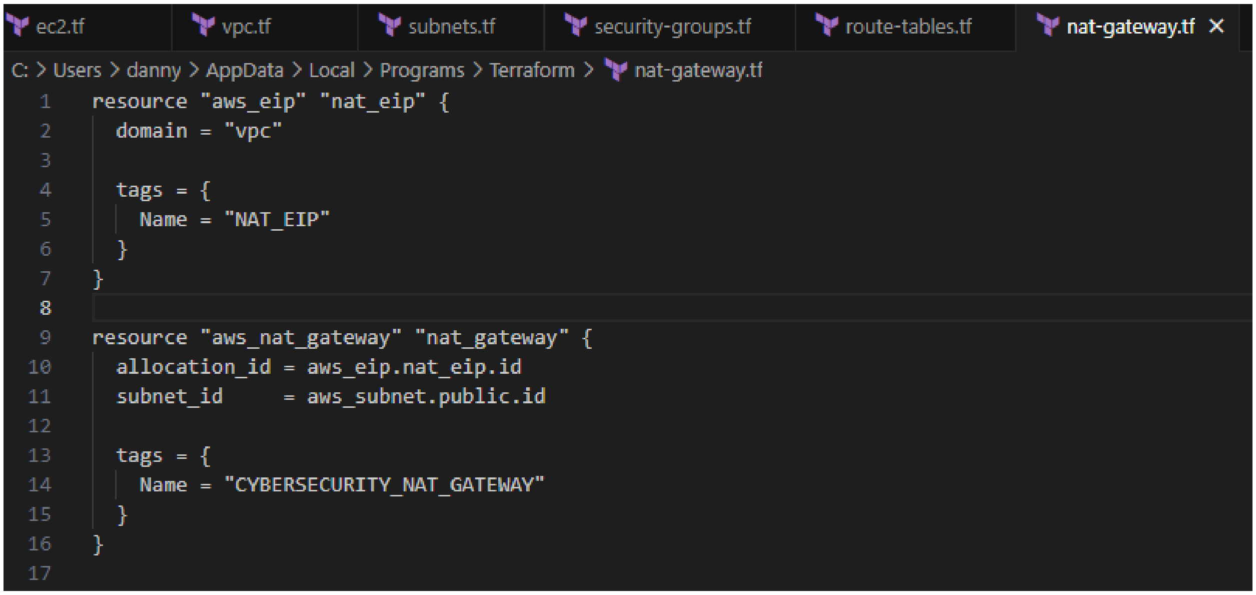
Update **route-tables.tf**

1. Add a Route Table for the Private Subnet
2. Add a Route to the NAT Gateway
3. Associate the Private Subnet with the Private Route Table



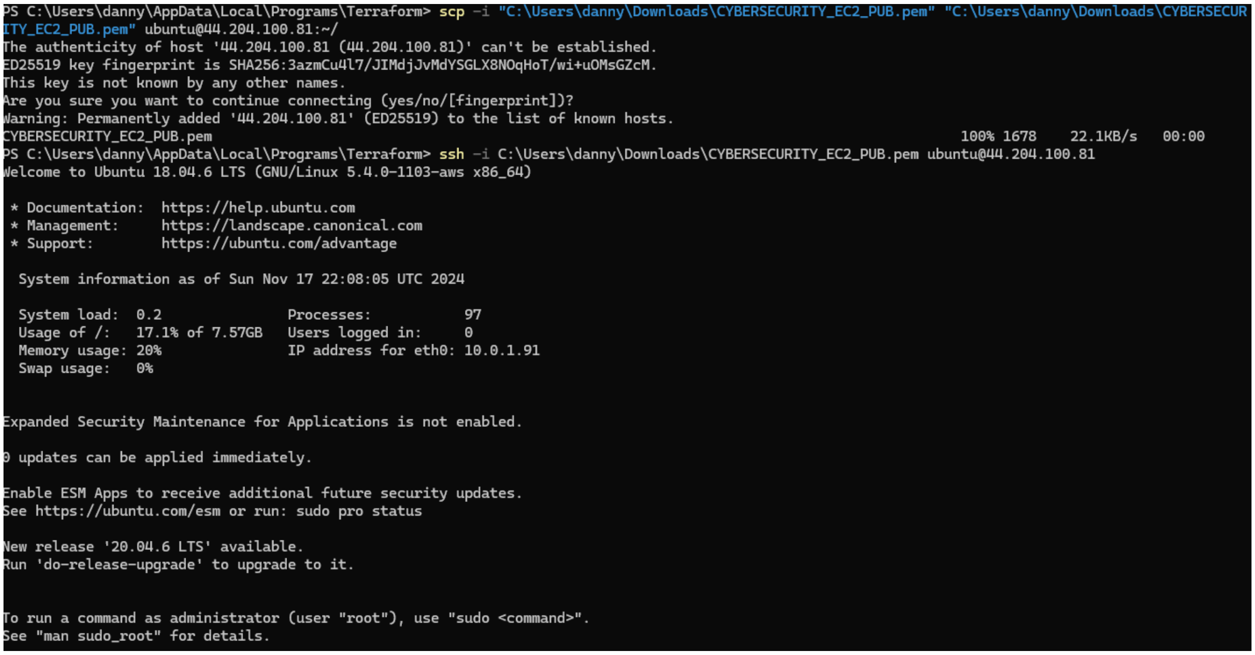
```
resource "aws_route_table" "private_route_table" {
  vpc_id = aws_vpc.main.id

  tags = {
    Name = "CYBERSECURITY_PRIVATE_ROUTE_TABLE"
  }
}

resource "aws_route" "private_to_nat" {
  route_table_id         = aws_route_table.private_route_table.id
  destination_cidr_block = "0.0.0.0/0"
  nat_gateway_id         = aws_nat_gateway.nat_gateway.id
}

resource "aws_route_table_association" "private_subnet_association" {
  subnet_id      = aws_subnet.private.id
  route_table_id = aws_route_table.private_route_table.id
}
```

Create a new file named **nat-gateway.tf**
Add **Elastic IP** and **NAT Gateway**

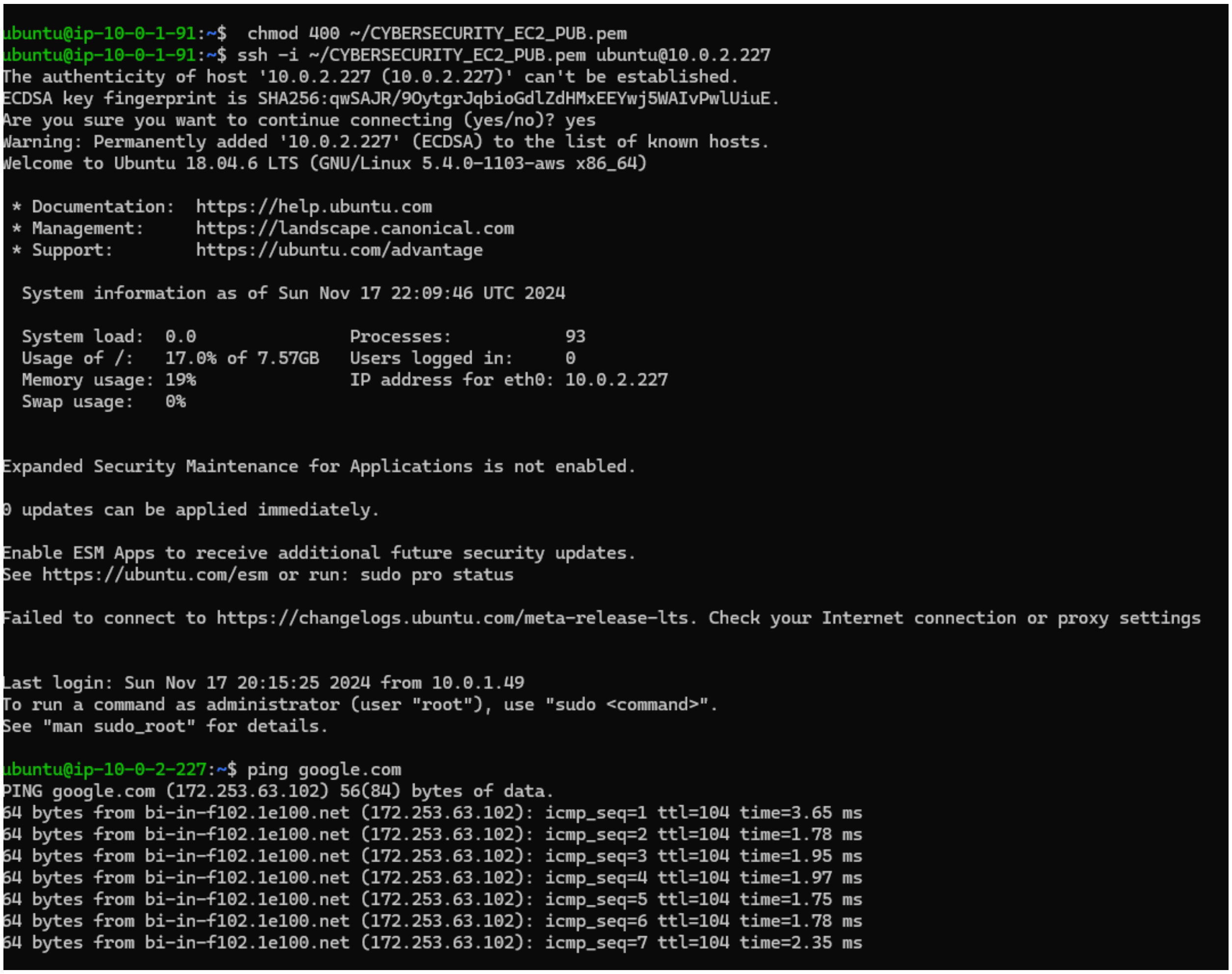

ec2.tf
vpc.tf
subnets.tf
security-groups.tf
route-tables.tf
nat-gateway.tf ✕
C: > Users > danny > AppData > Local > Programs > Terraform > nat-gateway.tf

```
resource "aws_eip" "nat_eip" {
  domain = "vpc"

  tags = {
    Name = "NAT_EIP"
  }
}

resource "aws_nat_gateway" "nat_gateway" {
  allocation_id = aws_eip.nat_eip.id
  subnet_id     = aws_subnet.public.id

  tags = {
    Name = "CYBERSECURITY_NAT_GATEWAY"
  }
}

```

Since I applied the change, 2 new instances have respawned.



```
PS C:\Users\danny\AppData\Local\Programs\Terraform> scp -i "C:\Users\danny\Downloads\CYBERSECURITY_EC2_PUB.pem" "C:\Users\danny\Downloads\CYBERSECUR
ITY_EC2_PUB.pem" ubuntu@44.204.100.81:~/
The authenticity of host '44.204.100.81 (44.204.100.81)' can't be established.
ED25519 key fingerprint is SHA256:3azmCu4l7/JIMdjJvMdYSGLX8NOqHoT/wi+uOMsGZcM.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])?
Warning: Permanently added '44.204.100.81' (ED25519) to the list of known hosts.
CYBERSECURITY_EC2_PUB.pem                                                                                    100% 1678    22.1KB/s   00:00
PS C:\Users\danny\AppData\Local\Programs\Terraform> ssh -i C:\Users\danny\Downloads\CYBERSECURITY_EC2_PUB.pem ubuntu@44.204.100.81
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 5.4.0-1103-aws x86_64)

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

  System information as of Sun Nov 17 22:08:05 UTC 2024

  System load:  0.2               Processes:           97
  Usage of /:   17.1% of 7.57GB   Users logged in:     0
  Memory usage: 20%               IP address for eth0: 10.0.1.91
  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

New release '20.04.6 LTS' available.
Run 'do-release-upgrade' to upgrade to it.


To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.
```



```
ubuntu@ip-10-0-1-91:~$  chmod 400 ~/CYBERSECURITY_EC2_PUB.pem
ubuntu@ip-10-0-1-91:~$ ssh -i ~/CYBERSECURITY_EC2_PUB.pem ubuntu@10.0.2.227
The authenticity of host '10.0.2.227 (10.0.2.227)' can't be established.
ECDSA key fingerprint is SHA256:qwSAJR/9OytgrJqbioGdlZdHMxEEYwj5WAIvPwlUiuE.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.0.2.227' (ECDSA) to the list of known hosts.
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 5.4.0-1103-aws x86_64)

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

  System information as of Sun Nov 17 22:09:46 UTC 2024

  System load:  0.0               Processes:           93
  Usage of /:   17.0% of 7.57GB   Users logged in:     0
  Memory usage: 19%               IP address for eth0: 10.0.2.227
  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

Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your Internet connection or proxy settings


Last login: Sun Nov 17 20:15:25 2024 from 10.0.1.49
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-10-0-2-227:~$ ping google.com
PING google.com (172.253.63.102) 56(84) bytes of data.
64 bytes from bi-in-f102.1e100.net (172.253.63.102): icmp_seq=1 ttl=104 time=3.65 ms
64 bytes from bi-in-f102.1e100.net (172.253.63.102): icmp_seq=2 ttl=104 time=1.78 ms
64 bytes from bi-in-f102.1e100.net (172.253.63.102): icmp_seq=3 ttl=104 time=1.95 ms
64 bytes from bi-in-f102.1e100.net (172.253.63.102): icmp_seq=4 ttl=104 time=1.97 ms
64 bytes from bi-in-f102.1e100.net (172.253.63.102): icmp_seq=5 ttl=104 time=1.75 ms
64 bytes from bi-in-f102.1e100.net (172.253.63.102): icmp_seq=6 ttl=104 time=1.78 ms
64 bytes from bi-in-f102.1e100.net (172.253.63.102): icmp_seq=7 ttl=104 time=2.35 ms
```

You can see the private instance can ping Google.com