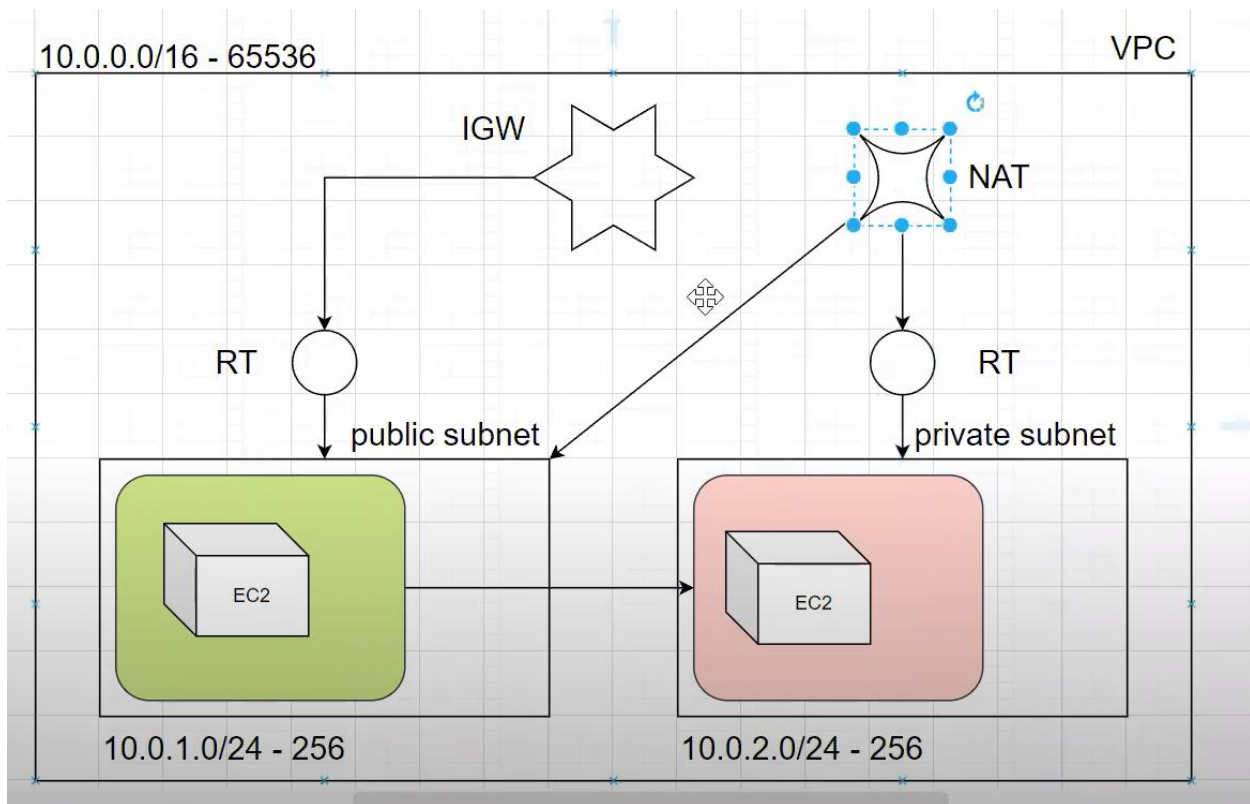


Working diagram.



VPC – 1

aws VPC > Your VPCs > vpc-05edd1ff5e542309e

vpc-05edd1ff5e542309e / Mainvpc_001

Details Info

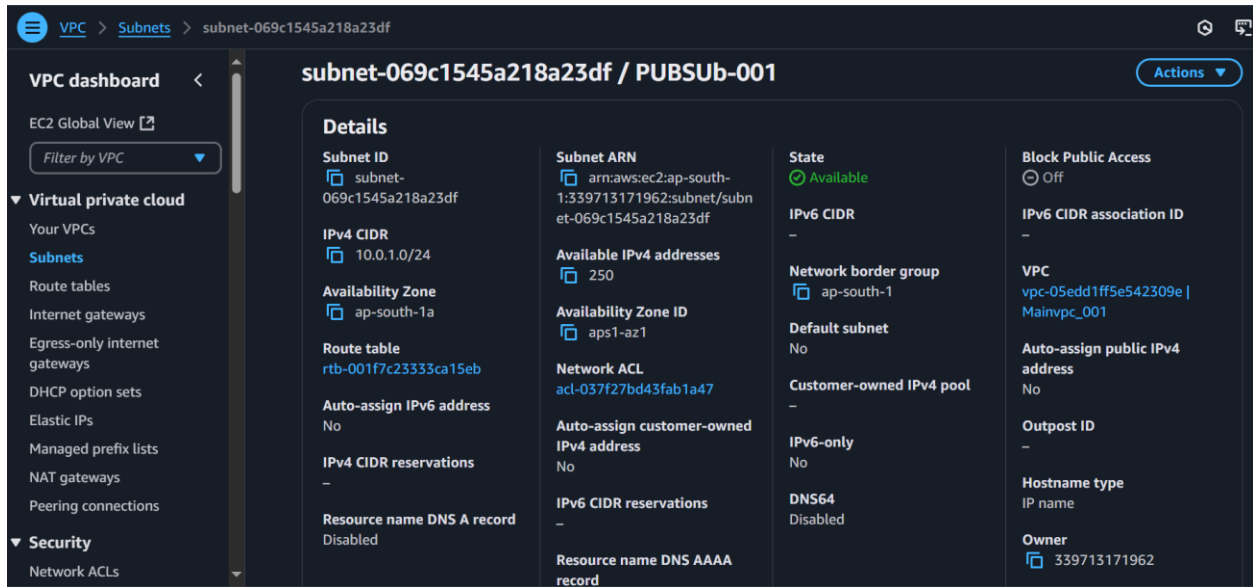
VPC ID vpc-05edd1ff5e542309e	State Available	Block Public Access Off	DNS hostnames Disabled
DNS resolution Enabled	Tenancy default	DHCP option set dopt-06eadffb7db0b7bf1	Main route table rtb-001f7c23333ca15eb
Main network ACL acl-037f27bd43fab1a47	Default VPC No	IPv4 CIDR 10.0.0.0/16	IPv6 pool -
IPv6 CIDR (Network border group) -	Network Address Usage metrics Disabled	Route 53 Resolver DNS Firewall rule groups -	Owner ID 339713171962

Resource map **CIDRs** Flow logs Tags Integrations

IPv4 CIDRs

Address family	CIDR	Status
IPv4	10.0.0.0/16	Associated

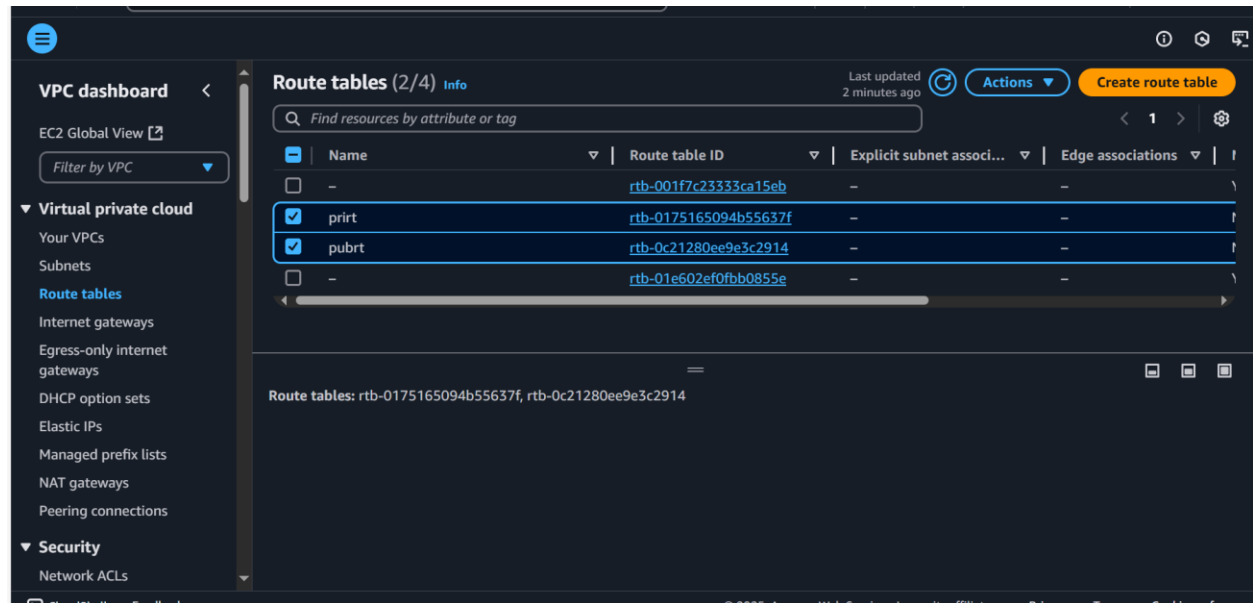
Subnet – pub



The screenshot displays the AWS VPC console interface for a specific subnet. The left sidebar shows the navigation menu with categories like 'Virtual private cloud' and 'Security'. The main content area is titled 'subnet-069c1545a218a23df / PUBSub-001'. Below the title, there are several key-value pairs organized into columns, providing detailed information about the subnet's configuration and status.

Details			
Subnet ID subnet-069c1545a218a23df	Subnet ARN arn:aws:ec2:ap-south-1:339713171962:subnet/subnet-069c1545a218a23df	State Available	Block Public Access Off
IPv4 CIDR 10.0.1.0/24	Available IPv4 addresses 250	IPv6 CIDR -	IPv6 CIDR association ID -
Availability Zone ap-south-1a	Availability Zone ID aps1-az1	Network border group ap-south-1	VPC vpc-05edd1ff5e542309e Mainvpc_001
Route table rtb-001f7c23333ca15eb	Network ACL acl-037f27bd43fab1a47	Default subnet No	Auto-assign public IPv4 address No
Auto-assign IPv6 address No	Auto-assign customer-owned IPv4 address No	Customer-owned IPv4 pool -	Outpost ID -
IPv4 CIDR reservations -	IPv6 CIDR reservations -	IPV6-only No	Hostname type IP name
Resource name DNS A record Disabled	Resource name DNS AAAA record	DNS64 Disabled	Owner 339713171962

Route tables

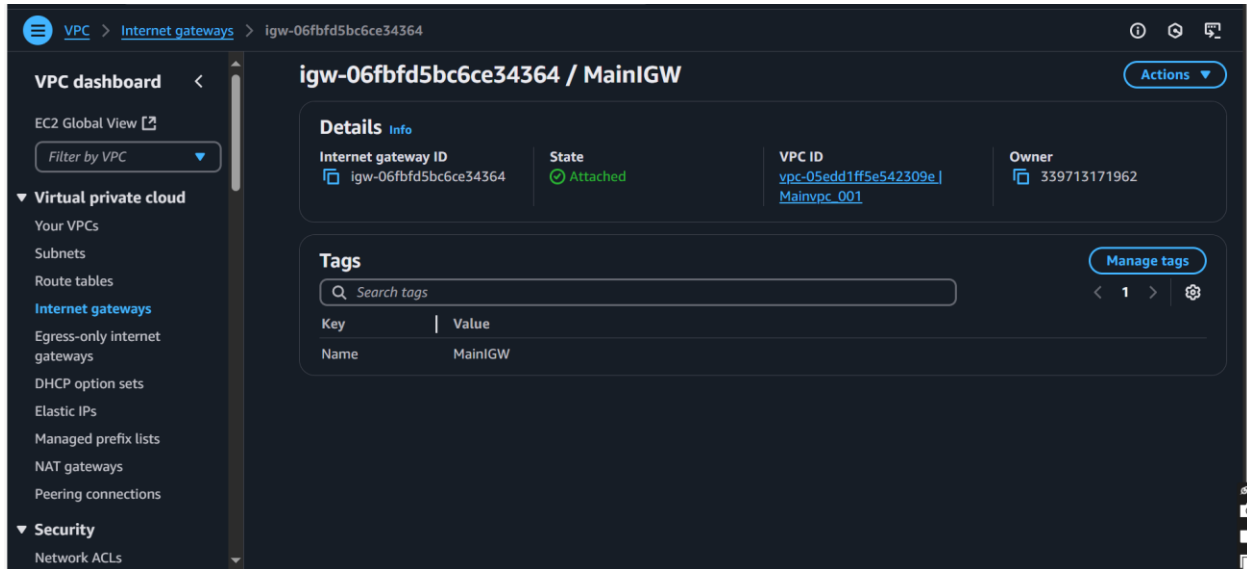


The screenshot shows the 'Route tables' page in the AWS VPC console. It features a table listing the available route tables, including their names, IDs, and associated subnets. The 'pubrt' route table is highlighted. Below the table, there is a summary of the route tables and their associated subnets.

Name	Route table ID	Explicit subnet associ...	Edge associations
-	rtb-001f7c23333ca15eb	-	-
prirt	rtb-0175165094b55637f	-	-
pubrt	rtb-0c21280ee9e3c2914	-	-
-	rtb-01e602ef0fbb0855e	-	-

Route tables: rtb-0175165094b55637f, rtb-0c21280ee9e3c2914

Internet Gateway



The screenshot displays the AWS Management Console interface for an Internet Gateway. The left sidebar shows the navigation menu with categories like VPC dashboard, Virtual private cloud, and Security. The main content area is titled "igw-06bfd5bc6ce34364 / MainIGW". It includes a "Details" section with fields for Internet gateway ID, State (Attached), VPC ID, and Owner. Below this is a "Tags" section with a search bar and a table showing a single tag with the key "Name" and value "MainIGW".

VPC dashboard < EC2 Global View [↗](#)

Filter by VPC [▼](#)

▼ **Virtual private cloud**

- Your VPCs
- Subnets
- Route tables
- Internet gateways**
- Egress-only internet gateways
- DHCP option sets
- Elastic IPs
- Managed prefix lists
- NAT gateways
- Peering connections

▼ **Security**

- Network ACLs

igw-06bfd5bc6ce34364 / MainIGW [Actions](#) ▼

Details [Info](#)

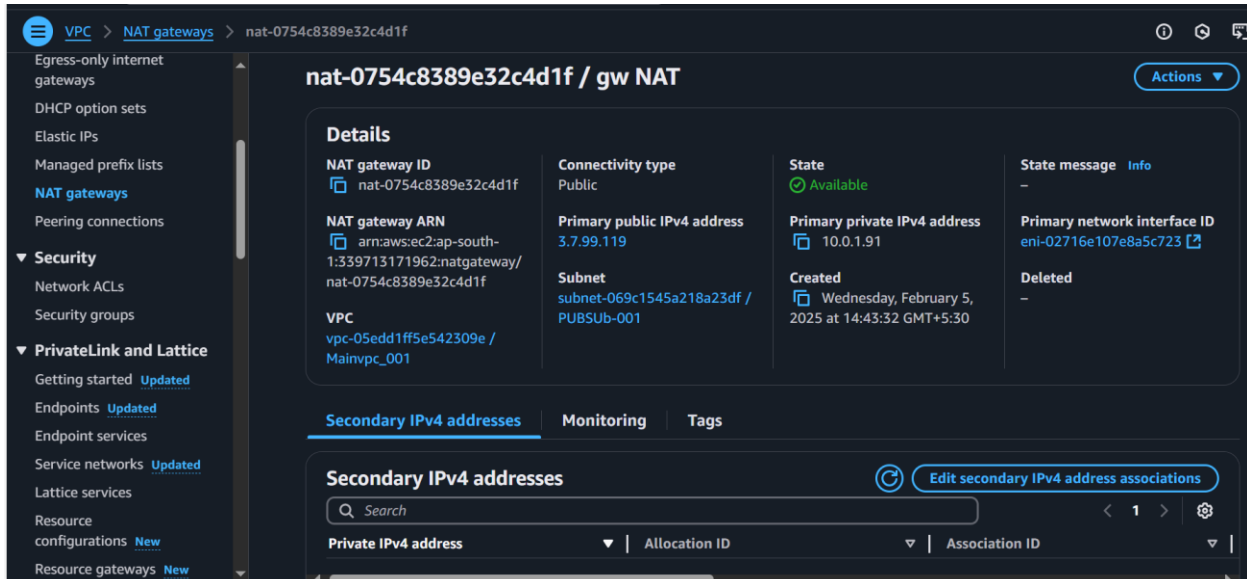
Internet gateway ID igw-06bfd5bc6ce34364	State Attached	VPC ID vpc-05edd1ff5e542309e / Mainvpc_001	Owner 339713171962
--	--------------------------	--	--

Tags [Manage tags](#)

Search tags

Key	Value
Name	MainIGW

NAT Gateway



The screenshot displays the AWS Management Console interface for a NAT Gateway. The left sidebar shows the navigation menu with categories like Egress-only internet gateways, DHCP option sets, Elastic IPs, Managed prefix lists, NAT gateways, Peering connections, Security, PrivateLink and Lattice, and Resource gateways. The main content area is titled "nat-0754c8389e32c4d1f / gw NAT". It includes a "Details" section with fields for NAT gateway ID, Connectivity type, State (Available), NAT gateway ARN, Primary public IPv4 address, Primary private IPv4 address, Created, State message, Primary network interface ID, and Deleted. Below this is a "Secondary IPv4 addresses" section with a search bar and a table showing a single address with the key "Private IPv4 address" and value "Allocation ID".

nat-0754c8389e32c4d1f / gw NAT [Actions](#) ▼

Details

NAT gateway ID nat-0754c8389e32c4d1f	Connectivity type Public	State Available	State message Info -
NAT gateway ARN arn:aws:ec2:ap-south-1:339713171962:natgateway/nat-0754c8389e32c4d1f	Primary public IPv4 address 3.7.99.119	Primary private IPv4 address 10.0.1.91	Primary network interface ID eni-02716e107e8a5c723 ↗
VPC vpc-05edd1ff5e542309e / Mainvpc_001	Subnet subnet-069c1545a218a23df / PUBSub-001	Created Wednesday, February 5, 2025 at 14:43:32 GMT+5:30	Deleted -

Secondary IPv4 addresses [Monitoring](#) [Tags](#)

Secondary IPv4 addresses [Edit secondary IPv4 address associations](#)

Search

Private IPv4 address	Allocation ID	Association ID

Security Groups

The screenshot displays the AWS Management Console interface for a Security Group. The breadcrumb navigation shows the path: VPC > Security Groups > sg-03fe218d5e1655168 - allow_tls. The left-hand navigation pane lists various AWS services, with 'Security' and 'PrivateLink and Lattice' expanded. The main content area is titled 'sg-03fe218d5e1655168 - allow_tls' and includes an 'Actions' button. Below the title, a 'Details' section provides key information in a grid:

Security group name allow_tls	Security group ID sg-03fe218d5e1655168	Description Allow TLS inbound traffic and all outbound traffic	VPC ID vpc-05edd1ff5e542309e
Owner 339713171962	Inbound rules count 1 Permission entry	Outbound rules count 0 Permission entries	

Below the details, there are tabs for 'Inbound rules', 'Outbound rules', 'Sharing - new', 'VPC associations - new', and 'Tags'. The 'Inbound rules' tab is active, showing a table with one rule. The table has columns for Name, Security group rule ID, IP version, Type, and Protocol. The first rule is 'sg-03fe218d5e1655168 - allow_tls' with an ID of 'sg-03fe218d5e1655168' and a protocol of 'TCP'.

Terraform console Output:-

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.myeip will be created
+ resource "aws_eip" "myeip" {
  + 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)
  + public_dns         = (known after apply)
  + public_ip          = (known after apply)
  + public_ipv4_pool   = (known after apply)
  + tags               = {
    + "name" = "example name "
  }
  + tags_all           = {
    + "name" = "example name "
  }
  + vpc                = (known after apply)
}
```

```
# aws_instance.Ec2_1 will be created
+ resource "aws_instance" "Ec2_1" {
```

```
# aws_instance.Ec2_1 will be created
+ resource "aws_instance" "Ec2_1" {
  + ami                    = "00bb6a80f01f03502"
  + 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)
  + enable_primary_ipv6    = (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"
  + ipv6_address_count     = (known after apply)
  + ipv6_addresses        = (known after apply)
  + key_name               = (known after apply)
  + monitoring             = (known after apply)
  + outpost_arn            = (known after apply)
  + password_data          = (known after apply)
  + placement_group        = (known after apply)
  + placement_partition_number = (known after apply)
  + primary_network_interface_id = (known after apply)
  + private_dns            = (known after apply)
  + private_ip             = (known after apply)
  + public_dns             = (known after apply)
  + public_ip              = (known after apply)
  + secondary_private_ips  = (known after apply)
  + security_groups        = (known after apply)
```

```

+ public_dns                = (known after apply)
+ public_ip                 = (known after apply)
+ secondary_private_ips     = (known after apply)
+ security_groups           = (known after apply)
+ source_dest_check         = true
+ spot_instance_request_id  = (known after apply)
+ subnet_id                 = (known after apply)
+ tags_all                  = (known after apply)
+ tenancy                   = (known after apply)
+ user_data                 = (known after apply)
+ user_data_base64         = (known after apply)
+ user_data_replace_on_change = false
+ vpc_security_group_ids    = (known after apply)

+ capacity_reservation_specification (known after apply)

+ cpu_options (known after apply)

+ ebs_block_device (known after apply)

+ enclave_options (known after apply)

+ ephemeral_block_device (known after apply)

+ instance_market_options (known after apply)

+ maintenance_options (known after apply)

+ metadata_options (known after apply)

+ network_interface (known after apply)

+ private_dns_name_options (known after apply)

+ root_block_device (known after apply)
}

```

```

# aws_instance.Ec2_2 will be created
+ resource "aws_instance" "Ec2_2" {
+   ami                = "00bb6a80f01f03502"
+   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)
+   enable_primary_ipv6 = (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"
+   ipv6_address_count   = (known after apply)
+   ipv6_addresses       = (known after apply)
+   key_name             = (known after apply)
+   monitoring           = (known after apply)
+   outpost_arn          = (known after apply)
+   password_data        = (known after apply)
+   placement_group      = (known after apply)
+   placement_partition_number = (known after apply)
+   primary_network_interface_id = (known after apply)
+   private_dns          = (known after apply)
+   private_ip           = (known after apply)
+   public_dns           = (known after apply)
+   public_ip            = (known after apply)
+   secondary_private_ips = (known after apply)

```

```

+ source_dest_check           = true
+ spot_instance_request_id    = (known after apply)
+ subnet_id                   = (known after apply)
+ tags_all                    = (known after apply)
+ tenancy                     = (known after apply)
+ user_data                   = (known after apply)
+ user_data_base64           = (known after apply)
+ user_data_replace_on_change = false
+ vpc_security_group_ids      = (known after apply)

+ capacity_reservation_specification (known after apply)

+ cpu_options (known after apply)

+ ebs_block_device (known after apply)

+ enclave_options (known after apply)

+ ephemeral_block_device (known after apply)

+ instance_market_options (known after apply)

+ maintenance_options (known after apply)

+ metadata_options (known after apply)

+ network_interface (known after apply)

+ private_dns_name_options (known after apply)

+ root_block_device (known after apply)
}

# aws_internet_gateway.gw will be created
+ resource "aws_internet_gateway" "gw" {
+   arn = (known after apply)

```

```

+   "Name" = "MainIGW"
+ }
+   tags_all = {
+     "Name" = "MainIGW"
+   }
+   vpc_id = (known after apply)
+ }

# aws_nat_gateway.NatTF will be created
+ resource "aws_nat_gateway" "NatTF" {
+   allocation_id           = (known after apply)
+   association_id          = (known after apply)
+   connectivity_type       = "public"
+   id                      = (known after apply)
+   network_interface_id    = (known after apply)
+   private_ip              = (known after apply)
+   public_ip               = (known after apply)
+   secondary_private_ip_address_count = (known after apply)
+   secondary_private_ip_addresses = (known after apply)
+   subnet_id               = (known after apply)
+   tags                    = {
+     "Name" = "gw NAT"
+   }
+   tags_all                = {
+     "Name" = "gw NAT"
+   }
+ }

# aws_route_table.priort will be created
+ resource "aws_route_table" "priort" {
+   arn = (known after apply)
+   id = (known after apply)
+   owner_id = (known after apply)
+   propagating_vgws = (known after apply)
+   route = [
+     {

```

```

+ route          = [
+   + {
+     + cidr_block          = "10.0.2.0/24"
+     + gateway_id          = (known after apply)
+     # (11 unchanged attributes hidden)
+   },
+ ]
+ tags            = {
+   + "Name" = "prirt"
+ }
+ tags_all        = {
+   + "Name" = "prirt"
+ }
+ vpc_id          = (known after apply)
}

# aws_route_table.pubrt will be created
+ resource "aws_route_table" "pubrt" {
+   arn              = (known after apply)
+   id               = (known after apply)
+   owner_id         = (known after apply)
+   propagating_vgws = (known after apply)
+   route            = [
+     + {
+       + cidr_block          = "10.0.1.0/24"
+       + gateway_id          = (known after apply)
+       # (11 unchanged attributes hidden)
+     },
+   ]
+   tags              = {
+     + "Name" = "pubrt"
+   }
+   tags_all          = {
+     + "Name" = "pubrt"
+   }
+   vpc_id            = (known after apply)
}

```

```

# aws_route_table_association.prirtass will be created
+ resource "aws_route_table_association" "prirtass" {
+   id              = (known after apply)
+   route_table_id = (known after apply)
+   subnet_id       = (known after apply)
+ }

# aws_route_table_association.pubrtass will be created
+ resource "aws_route_table_association" "pubrtass" {
+   id              = (known after apply)
+   route_table_id = (known after apply)
+   subnet_id       = (known after apply)
+ }

# aws_security_group.allow_tls will be created
+ resource "aws_security_group" "allow_tls" {
+   arn              = (known after apply)
+   description      = "Allow TLS inbound traffic and all outbound traffic"
+   egress           = (known after apply)
+   id               = (known after apply)
+   ingress          = (known after apply)
+   name             = "allow_tls"
+   name_prefix      = (known after apply)
+   owner_id         = (known after apply)
+   revoke_rules_on_delete = false
+   tags             = {
+     + "Name" = "allow_tls"
+   }
+   tags_all         = {
+     + "Name" = "allow_tls"
+   }
+   vpc_id           = (known after apply)
+ }

# aws_subnet.prirsub will be created

```



```

# aws_subnet.prisub will be created
+ resource "aws_subnet" "prisub" {
  + arn                                     = (known after apply)
  + assign_ipv6_address_on_creation        = false
  + availability_zone                     = "ap-south-2a"
  + 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" = "PRISUB-001"
  }
  + tags_all                              = {
    + "Name" = "PRISUB-001"
  }
  + vpc_id                                = (known after apply)
}

# aws_subnet.pubsub will be created
+ resource "aws_subnet" "pubsub" {
  + arn                                     = (known after apply)
  + assign_ipv6_address_on_creation        = false
  + availability_zone                     = "ap-south-1a"
  + availability_zone_id                  = (known after apply)
  + cidr_block                            = "10.0.1.0/24"
  + enable_dns64                          = false
  + enable_resource_name_dns_a_record_on_launch = false
  + enable_resource_name_dns_aaaa_record_on_launch = false

```

```

    + "Name" = "PUBSUB-001"
  }
  + tags_all                              = {
    + "Name" = "PUBSUB-001"
  }
  + vpc_id                                = (known after apply)
}

# aws_vpc.Mainvpc will be created
+ resource "aws_vpc" "Mainvpc" {
  + arn                                     = (known after apply)
  + cidr_block                             = "10.0.0.0/16"
  + default_network_acl_id                 = (known after apply)
  + default_route_table_id                = (known after apply)
  + default_security_group_id              = (known after apply)
  + dhcp_options_id                       = (known after apply)
  + enable_dns_hostnames                   = (known after apply)
  + enable_dns_support                     = true
  + enable_network_address_usage_metrics = (known after apply)
  + id                                    = (known after apply)
  + instance_tenancy                       = "default"
  + ipv6_association_id                   = (known after apply)
  + ipv6_cidr_block                       = (known after apply)
  + ipv6_cidr_block_network_border_group = (known after apply)
  + main_route_table_id                   = (known after apply)
  + owner_id                             = (known after apply)
  + tags                                  = {
    + "Name" = "Mainvpc-001"
  }
  + tags_all                              = {
    + "Name" = "Mainvpc-001"
  }
}

# aws_vpc_security_group_ingress_rule.allow_tls_ipv4 will be created
+ resource "aws_vpc_security_group_ingress_rule" "allow_tls_ipv4" {

```

```
# aws_vpc_security_group_ingress_rule.allow_tls_ipv4 will be created
+ resource "aws_vpc_security_group_ingress_rule" "allow_tls_ipv4" {
  + arn                = (known after apply)
  + cidr_ipv4          = "10.0.0.0/16"
  + from_port          = 22
  + id                 = (known after apply)
  + ip_protocol         = "tcp"
  + security_group_id   = (known after apply)
  + security_group_rule_id = (known after apply)
  + tags_all            = {}
  + to_port             = 22
}
```

Plan: 14 to add, 0 to change, 0 to destroy.