

Using Terraform to
create the Terraform
state bucket

Terraform files
explanation

▼ 3. Setting up the Network

**Using Terraform to
create VPC and other
Network related
resources**

Terraform files
explanation

▼ 4. IAM Roles and Policies for EKS

Using Terraform to
create the IAM Roles
and Policies for EKS

Terraform files
explanation

► 5. Linking the Cloud9 IDE & CI/CD VPC to the EKS Network

▼ 6. Deploy the CICD Infrastructure

Create the CI/CD
Components

Terraform files
explanation

► 7. EKS Cluster Creation

► 8. Create a customized managed Node Group

► 9. Enable AWS Load Balancers on EKS

► 10. Deploy a sample application

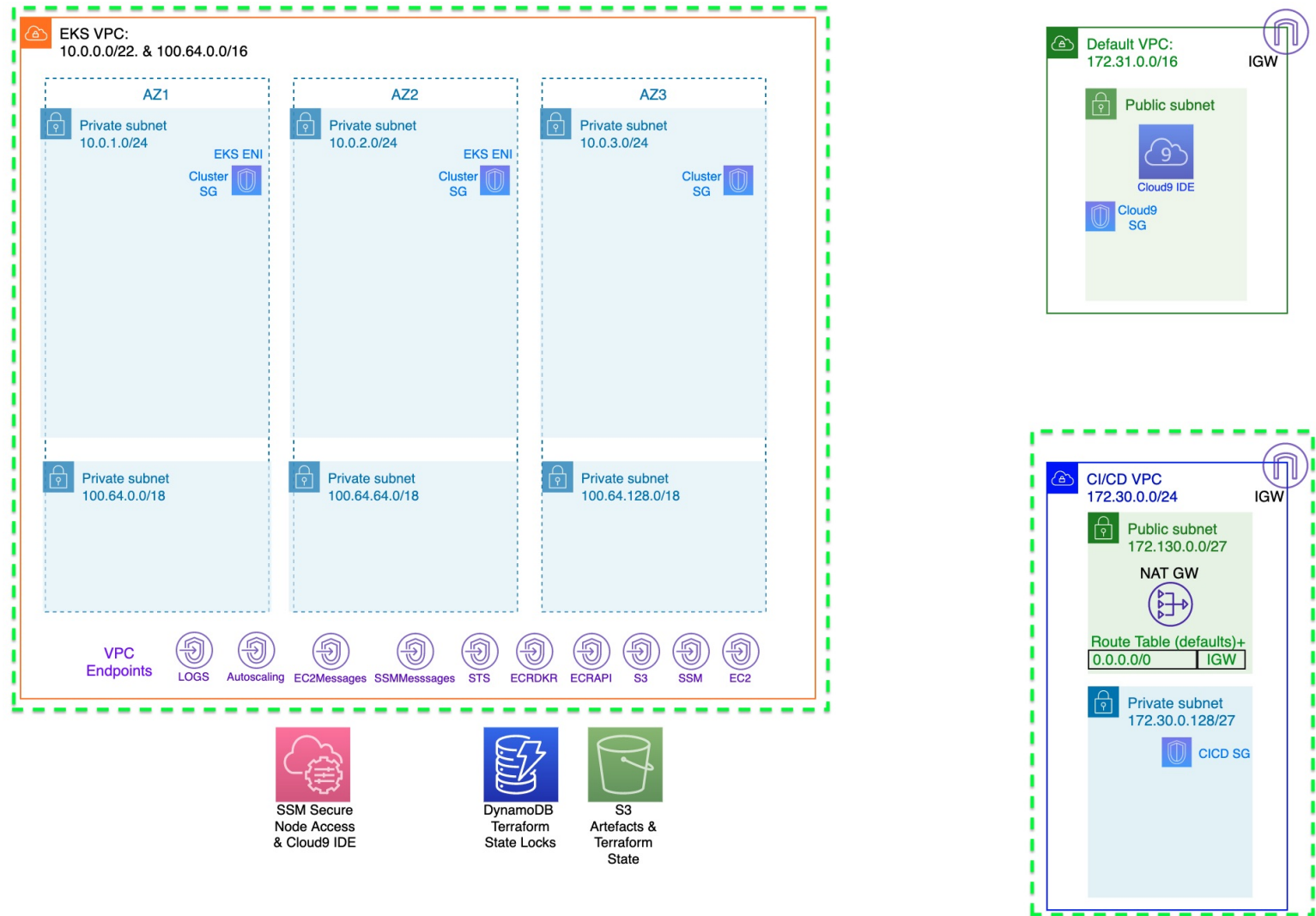
► 11. Private CI/CD for EKS


[Amazon EKS Terraform Workshop](#) > [5. Creating a private EKS Cluster with Terraform](#) > [3. Setting up the Network](#) >
Using Terraform to create VPC and other Network related resources

Using Terraform to create VPC and other Network related resources

From the Cloud9 IDE we will next build the main networking components for our EKS cluster

This diagram shows the EKS VPC and CI/CD VPC we will build in this section:



 **Disclaimer:** For production workloads you should expand the default and CI/CD VPC's to use multiple subnets in two or three availability zones.

Deploying the Network

```
1 cd ~/environment/tfekscode/net
```

Initialize Terraform:

```
1 terraform init
```

Initializing the backend...

Initializing provider plugins...

- Reusing previous version of hashicorp/kubernetes from the dependency lock file
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v4.63.0
- Using previously-installed hashicorp/kubernetes v2.17.0

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.

Validate the Terraform code

```
1 terraform validate
```

Success! The configuration is valid.

Plan the deployment:

- terraform state
 - Using Terraform to create the Terraform state bucket
 - Terraform files explanation
- ▼ 3. Setting up the Network
 - [Using Terraform to create VPC and other Network related resources](#)
 - Terraform files explanation
- ▼ 4. IAM Roles and Policies for EKS
 - Using Terraform to create the IAM Roles and Policies for EKS
 - Terraform files explanation
- 5. Linking the Cloud9 IDE & CI/CD VPC to the EKS Network
- ▼ 6. Deploy the CICD Infrastructure
 - Create the CI/CD Components
 - Terraform files explanation
- 7. EKS Cluster Creation
- 8. Create a customized managed Node Group
- 9. Enable AWS Load Balancers on EKS
- 10. Deploy a sample application
- 11. Private CI/CD for EKS

```
1 terraform plan -out tfplan
```



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.eipalloc-052dd24eaa93ed064 will be created
+ resource "aws_eip" "eipalloc-052dd24eaa93ed064" {
  + allocation_id      = (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)
  + network_border_group = (known after apply)
  + network_interface  = (known after apply)
  + private_dns        = (known after apply)
  + private_ip         = (known after apply)
  + public_dns         = (known after apply)
  + public_ip          = (known after apply)
  + public_ipv4_pool    = "amazon"
  + tags_all           = (known after apply)
  + vpc                 = true

  + timeouts {}
}
```

****OUTPUT TRUNCATED FOR BREVITY****

```
# aws_vpc_ipv4_cidr_block_association.vpc-cidr-assoc will be created
+ resource "aws_vpc_ipv4_cidr_block_association" "vpc-cidr-assoc" {
  + cidr_block = "100.64.0.0/16"
  + id        = (known after apply)
  + vpc_id    = (known after apply)

  + timeouts {}
}
```

Plan: 62 to add, 0 to change, 0 to destroy.

Terraform state

Using Terraform to
create the Terraform
state bucket

Terraform files
explanation

▼ 3. Setting up the Network

**Using Terraform to
create VPC and other
Network related
resources**

Terraform files
explanation

▼ 4. IAM Roles and Policies for EKS

Using Terraform to
create the IAM Roles
and Policies for EKS

Terraform files
explanation

► 5. Linking the Cloud9 IDE & CI/CD VPC to the EKS Network

▼ 6. Deploy the CICD Infrastructure

Create the CI/CD
Components

Terraform files
explanation

► 7. EKS Cluster Creation

► 8. Create a customized managed Node Group

► 9. Enable AWS Load Balancers on EKS

► 10. Deploy a sample application

► 11. Private CI/CD for EKS

terraform state	Using Terraform to create the Terraform state bucket	Terraform files explanation
▼ 3. Setting up the Network	Using Terraform to create VPC and other Network related resources	Terraform files explanation
▼ 4. IAM Roles and Policies for EKS	Using Terraform to create the IAM Roles and Policies for EKS	Terraform files explanation
► 5. Linking the Cloud9 IDE & CI/CD VPC to the EKS Network		
▼ 6. Deploy the CICD Infrastructure	Create the CI/CD Components	Terraform files explanation
► 7. EKS Cluster Creation		
► 8. Create a customized managed Node Group		
► 9. Enable AWS Load Balancers on EKS		
► 10. Deploy a sample application		
► 11. Private CI/CD for EKS		

Saved the plan to: tfplan

To perform exactly these actions, run the following command to apply:
terraform apply "tfplan"

You can see from the plan the following resources will be created - open the corresponding files to see the Terraform HCL code that details the configuration:

- A VPC (**vpc-cluster.tf**).
- A secondary VPC CIDR block (**aws_vpc_ipv4_cidr_block_association__vpc-cidr-assoc.tf**).
- Various VPE Endpoints (**vpce.tf**).
- Subnets (**subnets-eks.tf**).
- Route Tables (**aws_route_table__rtb-*.tf**).
- Route Table Associations (**aws_route_table_association__rtbassoc.tf***).
- Security Groups (**aws_security_group__allnodes-sg.tf** & **aws_security_group__cluster-sg.tf**).
- NAT Gateway (**aws_nat_gateway__eks-cicd.tf**).

There are also Terraform file to setup the VPC and subnets used by CodeBuild part of the CICD pipeline

- The VPC (**aws_vpc__eks-cicd.tf**) and it's associated:
- Subnets (**aws_subnet__eks-cicd.tf***).
- Security groups (**aws_security_group__sg-eks-cicd.tf**).
- Route tables (**aws_route_table__private1.tf** & **aws_route_table__public1.tf**).
- Route Table Associations (**aws_route_table_association__private1.tf** & **aws_route_table_association__public1.tf**).
- Internet Gateway (**aws_internet_gateway__eks-cicd.tf**).
- A NAT Gateway (**aws_eip__eipalloc-cicd-natgw.tf**).

Build the Network environment (**note this will take a few minutes**):

```
1 terraform apply tfplan
```



```
aws_vpc.cluster: Creating...
aws_vpc.vpc-cicd: Creating...
aws_eip.eipalloc-052dd24eaa93ed064: Creating...
aws_eip.eipalloc-052dd24eaa93ed064: Creation complete after 1s [id=eipalloc-00641bfb571ab7079]
```

terraform state

Using Terraform to
create the Terraform
state bucket

Terraform files
explanation

▼ 3. Setting up the Network

**Using Terraform to
create VPC and other
Network related
resources**

Terraform files
explanation

▼ 4. IAM Roles and Policies for EKS

Using Terraform to
create the IAM Roles
and Policies for EKS

Terraform files
explanation

► 5. Linking the Cloud9 IDE & CI/CD VPC to the EKS Network



Create the CI/CD
Components

Terraform files
explanation

► 7. EKS Cluster Creation

► 8. Create a customized managed Node Group

► 9. Enable AWS Load Balancers on EKS

► 10. Deploy a sample application

► 11. Private CI/CD for EKS

aws_vpc.vpc-cidr: Creation complete after 1s [id=vpc-08d451043288a349a]

****OUTPUT TRUNCATED FOR BREVITY****

aws_vpc_endpoint.vpce-elb: Still creating... [1m0s elapsed]
aws_vpc_endpoint.vpce-sts: Creation complete after 1m1s [id=vpce-0c7e0ee39613074ed]
aws_vpc_endpoint.vpce-elb: Creation complete after 1m1s [id=vpce-01bc258615b5951c1]
aws_vpc_endpoint.vpce-ecrdr: Still creating... [1m10s elapsed]
aws_vpc_endpoint.vpce-ecrdr: Creation complete after 1m12s [id=vpce-048c20519113e5883]

Apply complete! Resources: 62 added, 0 changed, 0 destroyed.

The above Creates the VPC we will use for EKS and CID. Note several of the parameters are captured at Outputs, these will be used in later stages of the build.

Examine the results in AWS console. Look for new VPC's, Subnets etc.

You can also see the SSM paramters being stored at this point of the build with:

```
1 aws ssm describe-parameters --query Parameters[].Name | jq -r .[]
```



```
/workshop/tf-eks/cidr-vpc  
/workshop/tf-eks/cluster-name
```

```
/workshop/tf-eks/ia  
/workshop/tf-eks/keyid  
/workshop/tf-eks/rtb-isol  
/workshop/tf-eks/rtb-priv1  
/workshop/tf-eks/sub-priv1  
/workshop/tf-eks/sub-priv2  
/workshop/tf-eks/bucket-name  
/workshop/tf-eks/cidr-cidr  
/workshop/tf-eks/keyarn  
/workshop/tf-eks/net-cluster-sg  
/workshop/tf-eks/region  
/workshop/tf-eks/rtb-priv2  
/workshop/tf-eks/rtb-priv3  
/workshop/tf-eks/sub-isol1  
/workshop/tf-eks/sub-isol2  
/workshop/tf-eks/sub-isol3  
/workshop/tf-eks/allnodes-sg  
/workshop/tf-eks/eks-vpc
```



/workshop/tf-eks/sub-p1
/workshop/tf-eks/sub-priv3

To examine the valke of the EKS version we are using you can use:

```
1 aws ssm get-parameter --name /workshop/tf-eks/eks-version --query Parameter.Value --output text
```



1.24

[Previous](#)

[Next](#)

Terraform state

Using Terraform to
create the Terraform
state bucket

Terraform files
explanation

▼ 3. Setting up the Network

**Using Terraform to
create VPC and other
Network related
resources**

Terraform files
explanation

▼ 4. IAM Roles and Policies

create the IAM Roles
and Policies for EKS

Terraform files
explanation

► 5. Linking the Cloud9 IDE & CI/CD VPC to the EKS Network

▼ 6. Deploy the CICD Infrastructure

Create the CI/CD
Components

Terraform files
explanation

► 7. EKS Cluster Creation

► 8. Create a customized managed Node Group

► 9. Enable AWS Load Balancers on EKS

► 10. Deploy a sample application

► 11. Private CI/CD for EKS