



& CI/CD VPC to the EKS
Network

Connect the Cloud9
IDE & CICD VPC to the
EKS VPC

Terraform files
explanation

▼ 6. Deploy the CICD
Infrastructure

Create the CI/CD
Components

Terraform files
explanation

▼ 7. EKS Cluster Creation

Creating the EKS
Cluster

Terraform files
explanation

▼ 8. Create a customized
managed Node Group

**Creating the EKS
NodeGroup**

Test Networking

Terraform files
explanation

► 9. Enable AWS Load
Balancers on EKS

► 10. Deploy a sample
application

► 11. Private CI/CD for EKS

► 6. Extra Activities (Optional)

► 7. Using Fargate (Optional)

► Conclusion

► Cleanup

[Amazon EKS Terraform Workshop](#) > [5. Creating a private EKS Cluster with Terraform](#) > [8. Create a customized managed Node Group](#) >

Creating the EKS NodeGroup

Creating the EKS NodeGroup

Create a managed node group with a custom ami and user_data

& CI/CD VPC to the EKS Network

Connect the Cloud9 IDE & CICD VPC to the EKS VPC

Terraform files explanation

▼ 6. Deploy the CICD Infrastructure

Create the CI/CD Components

Terraform files explanation

▼ 7. EKS Cluster Creation

Creating the EKS Cluster

Terraform files explanation

▼ 8. Create a customized managed Node Group

Creating the EKS NodeGroup

Test Networking

Terraform files explanation

► 9. Enable AWS Load Balancers on EKS

► 10. Deploy a sample application

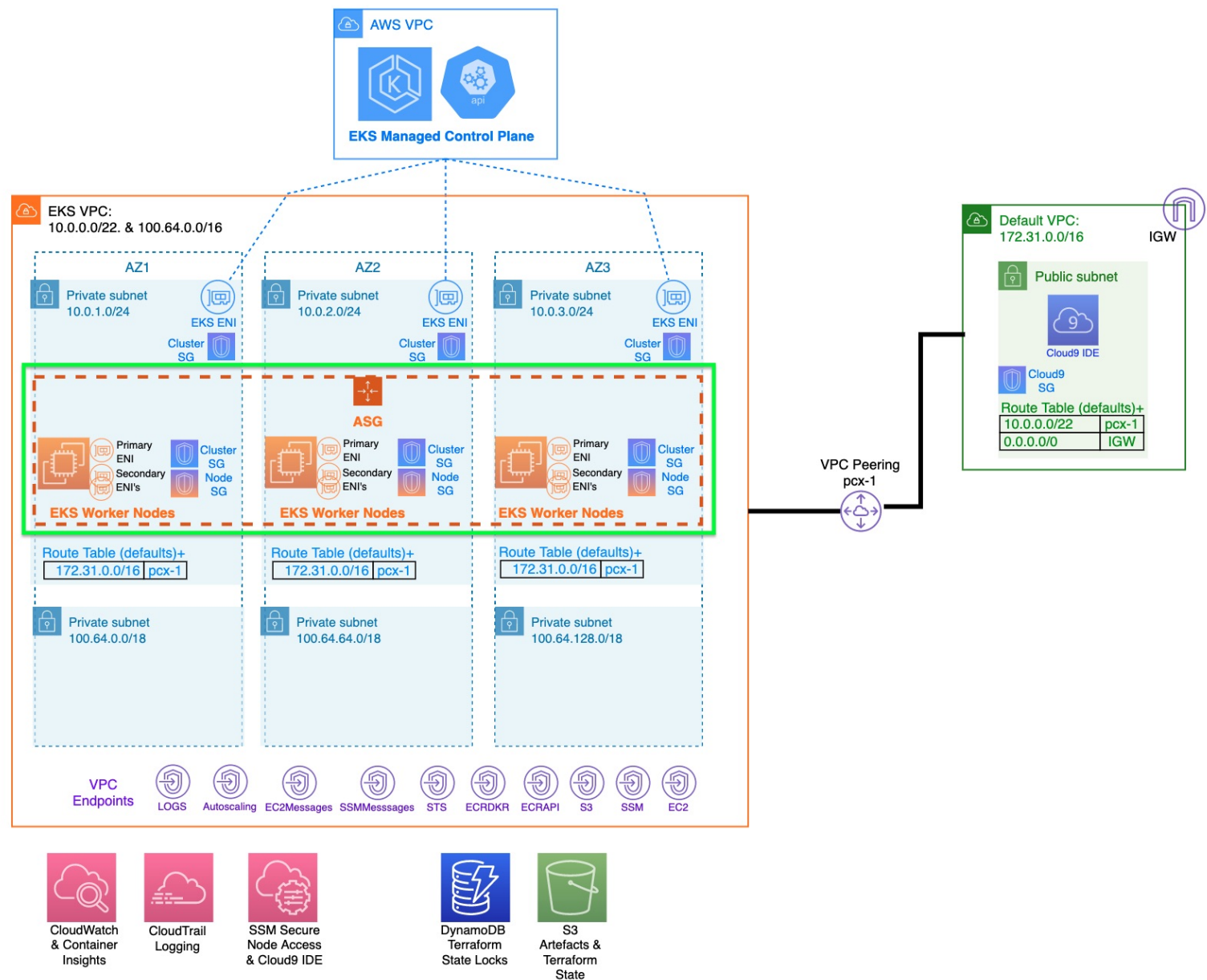
► 11. Private CI/CD for EKS

► 6. Extra Activities (Optional)

► 7. Using Fargate (Optional)

► Conclusion

► Cleanup



```
1 cd ~/environment/tfekscode/nodegroup
```

Initialize Terraform:

```
1 terraform init
```

& CI/CD VPC to the EKS Network

Connect the Cloud9 IDE & CICD VPC to the EKS VPC

Terraform files explanation

▼ 6. Deploy the CICD Infrastructure

Create the CI/CD Components

Terraform files explanation

▼ 7. EKS Cluster Creation

Creating the EKS Cluster

Terraform files explanation

▼ 8. Create a customized managed Node Group

Creating the EKS NodeGroup

Test Networking

Terraform files explanation

► 9. Enable AWS Load Balancers on EKS

► 10. Deploy a sample application

► 11. Private CI/CD for EKS

► 6. Extra Activities (Optional)

► 7. Using Fargate (Optional)

► Conclusion

► Cleanup

Validate the Terraform code:

```
1 terraform validate
```



Plan the deployment:

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



**** THE OUTPUT IS TRUNCATED FOR BREVITY INDICATED WITH '...' ****

```
data.aws_ssm_parameter.ca: Reading...
```

```
....
```

```
data.aws_subnet.i2: Read complete after 0s
```

```
data.aws_caller_identity.current: Read complete after 0s [id=440018911661]
```

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_eks_node_group.ng1 will be created
+ resource "aws_eks_node_group" "ng1" {
  + ami_type           = (known after apply)
  + arn                = (known after apply)
  + capacity_type      = (known after apply)
  + cluster_name       = "mycluster1"
  + disk_size          = 0
  + id                 = (known after apply)
  + instance_types     = []
  + labels             = {
    + "eks/cluster-name" = "mycluster1"
    + "eks/nodegroup-name" = "ng1-mycluster1"
  }
  + node_group_name     = "ng1-mycluster1"
  + node_group_name_prefix = (known after apply)
  + node_role_arn       = (sensitive value)
  + release_version     = (known after apply)
  + resources           = (known after apply)
  + status              = (known after apply)
  + subnet_ids          = (sensitive value)
  + tags                = {
    + "eks/cluster-name" = "mycluster1"
    + "eks/nodegroup-name" = "ng1-mycluster1"
```

& CI/CD VPC to the EKS Network

Connect the Cloud9 IDE & CICD VPC to the EKS VPC

Terraform files
explanation

▼ 6. Deploy the CICD Infrastructure

Create the CI/CD Components

Terraform files
explanation

▼ 7. EKS Cluster Creation

Creating the EKS Cluster

Terraform files
explanation

▼ 8. Create a customized managed Node Group

[Creating the EKS NodeGroup](#)

Test Networking

Terraform files
explanation

► 9. Enable AWS Load Balancers on EKS

► 10. Deploy a sample application

► 11. Private CI/CD for EKS

► 6. Extra Activities (Optional)

► 7. Using Fargate (Optional)

► Conclusion

► Cleanup

```
+ "eks/nodegroup-type"           = "managed"
+ "eksctl.cluster.k8s.io/v1alpha1/cluster-name" = "mycluster1"
}
+ tags_all                       = {
+   "eks/cluster-name"           = "mycluster1"
+   "eks/nodegroup-name"        = "ng1-mycluster1"
+   "eks/nodegroup-type"        = "managed"
+   "eksctl.cluster.k8s.io/v1alpha1/cluster-name" = "mycluster1"
}
+ version                       = (known after apply)

+ launch_template {
+   id      = (known after apply)
+   name    = "at-lt-mycluster1-ng1"
+   version = "1"
}

+ scaling_config {
+   desired_size = 2
+   max_size     = 3
+   min_size     = 1
}

+ timeouts {}
}
```

....

```
# null_resource.annotate will be created
+ resource "null_resource" "annotate" {
+   id      = (known after apply)
+   triggers = {}
}

# null_resource.gen_cluster_auth will be created
+ resource "null_resource" "gen_cluster_auth" {
+   id      = (known after apply)
+   triggers = {}
}
```

Plan: 4 to add, 0 to change, 0 to destroy.

Changes to Outputs:

```
+ config-map-aws-auth = "local.config-map-aws-auth"
+ kubeconfig          = "local.kubeconfig"
```

Saved the plan to: tfplan

& CI/CD VPC to the EKS Network

Connect the Cloud9 IDE & CID VPC to the EKS VPC

Terraform files explanation

▼ 6. Deploy the CID Infrastructure

Create the CI/CD Components

Terraform files explanation

▼ 7. EKS Cluster Creation

Creating the EKS Cluster

Terraform files explanation

▼ 8. Create a customized managed Node Group

[Creating the EKS NodeGroup](#)

Test Networking

Terraform files explanation

▶ 9. Enable AWS Load Balancers on EKS

▶ 10. Deploy a sample application

▶ 11. Private CI/CD for EKS

▶ 6. Extra Activities (Optional)

▶ 7. Using Fargate (Optional)

▶ Conclusion

▶ Cleanup

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

- A Launch template
- A NodeGroup using the launch template above
- A null resource (this will auth us to the cluster)

Build the environment:

```
1 terraform apply tfplan
```



**** THE OUTPUT IS TRUNCATED FOR BREVITY INDICATED WITH '...' ****

```
aws_launch_template.lt-ng1: Creating...
aws_launch_template.lt-ng1: Creation complete after 0s [id=lt-0bc35815d22910c0d]
aws_eks_node_group.ng1: Creating...
aws_eks_node_group.ng1: Still creating... [10s elapsed]
....
aws_eks_node_group.ng1: Still creating... [1m50s elapsed]
aws_eks_node_group.ng1: Creation complete after 2m0s [id=mycluster1:ng1-mycluster1]
null_resource.annotate: Creating...
null_resource.gen_cluster_auth: Creating...
null_resource.gen_cluster_auth: Provisioning with 'local-exec'...
null_resource.gen_cluster_auth (local-exec): Executing: ["/bin/bash" "-c" "          ./c9-auth.sh\n          echo \"*****\""]
null_resource.annotate: Provisioning with 'local-exec'...
null_resource.gen_cluster_auth (local-exec): Executing: ["/bin/bash" "-c" "          az1=$(echo eu-west-1a)\n          az2=$(echo eu-west-1b)"]
null_resource.gen_cluster_auth (local-exec): C9_PID is 8487e9e186ab4c20b39264c36ebd4a38
null_resource.gen_cluster_auth (local-exec): local auth
null_resource.annotate (local-exec): eu-west-1a eu-west-1b eu-west-1c subnet-07d95eb90a2d0f5aa subnet-037153414a2da552e subnet-03825ee07e5c9abae
null_resource.annotate (local-exec): Annotate nodes .....
null_resource.annotate (local-exec): CLUSTER is mycluster1
null_resource.annotate (local-exec): tfid is 4e05298ec6bc96e1
null_resource.annotate (local-exec): NAME                               CREATED AT
null_resource.annotate (local-exec): eniconfigs.crd.k8s.amazonaws.com 2023-06-27T09:27:21Z
null_resource.annotate (local-exec): securitygrouppolicies.vpcresources.k8s.aws 2023-06-27T09:27:23Z
null_resource.annotate (local-exec): Descr EC2 instance i-0b28c94d508c3b9df ...
null_resource.annotate (local-exec): subnet subnet-07d95eb90a2d0f5aa zone eu-west-1a
null_resource.annotate (local-exec): subnet subnet-037153414a2da552e zone eu-west-1b
null_resource.annotate (local-exec): subnet subnet-03825ee07e5c9abae zone eu-west-1c
null_resource.annotate (local-exec): eu-west-1a
```

& CI/CD VPC to the EKS Network

Connect the Cloud9 IDE & CICD VPC to the EKS VPC

Terraform files explanation

▼ 6. Deploy the CICD Infrastructure

Create the CI/CD Components

Terraform files explanation

▼ 7. EKS Cluster Creation

Creating the EKS Cluster

Terraform files explanation

▼ 8. Create a customized managed Node Group

[Creating the EKS NodeGroup](#)

Test Networking

Terraform files explanation

► 9. Enable AWS Load Balancers on EKS

► 10. Deploy a sample application

► 11. Private CI/CD for EKS

► 6. Extra Activities (Optional)

► 7. Using Fargate (Optional)

► Conclusion

► Cleanup

```
null_resource.annotate (local-exec): created eu-west-1a-pod-netconfig.yaml
null_resource.annotate (local-exec): eu-west-1b
null_resource.annotate (local-exec): created eu-west-1b-pod-netconfig.yaml
null_resource.annotate (local-exec): eu-west-1c
null_resource.annotate (local-exec): created eu-west-1c-pod-netconfig.yaml
null_resource.annotate (local-exec): apply the CRD eu-west-1a
null_resource.annotate (local-exec): eniconfig.crd.k8s.amazonaws.com/eu-west-1a-pod-netconfig created
null_resource.annotate (local-exec): apply the CRD eu-west-1b
null_resource.annotate (local-exec): eniconfig.crd.k8s.amazonaws.com/eu-west-1b-pod-netconfig created
null_resource.annotate (local-exec): apply the CRD eu-west-1c
null_resource.gen_cluster_auth (local-exec): Role ARN: arn:aws:iam::666763910423:role/eksworkshop-admin
null_resource.annotate (local-exec): eniconfig.crd.k8s.amazonaws.com/eu-west-1c-pod-netconfig created
null_resource.annotate (local-exec): pause 20s before annotate
null_resource.gen_cluster_auth (local-exec): Warning: resource configmaps/aws-auth is missing the kubectl.kubernetes.io/latest
null_resource.gen_cluster_auth (local-exec): configmap/aws-auth configured
null_resource.gen_cluster_auth: Still creating... [10s elapsed]
null_resource.annotate: Still creating... [10s elapsed]
null_resource.gen_cluster_auth (local-exec): Name:      aws-auth
null_resource.gen_cluster_auth (local-exec): Namespace: kube-system
null_resource.gen_cluster_auth (local-exec): Labels:      <none>
null_resource.gen_cluster_auth (local-exec): Annotations: <none>

null_resource.gen_cluster_auth (local-exec): Data
null_resource.gen_cluster_auth (local-exec): ====
null_resource.gen_cluster_auth (local-exec): mapRoles:
null_resource.gen_cluster_auth (local-exec): ----
null_resource.gen_cluster_auth (local-exec): - groups:
null_resource.gen_cluster_auth (local-exec):   - system:bootstrappers
null_resource.gen_cluster_auth (local-exec):   - system:nodes
null_resource.gen_cluster_auth (local-exec):   rolearn: arn:aws:iam::666763910423:role/4e05298ec6bc96e1-eks-nodegroup-NodeInstance
null_resource.gen_cluster_auth (local-exec):   username: system:node:{{EC2PrivateDNSName}}

null_resource.gen_cluster_auth (local-exec): mapUsers:
null_resource.gen_cluster_auth (local-exec): ----
null_resource.gen_cluster_auth (local-exec): - userarn: arn:aws:iam::666763910423:role/eksworkshop-admin
null_resource.gen_cluster_auth (local-exec):   username: admin
null_resource.gen_cluster_auth (local-exec):   groups:
null_resource.gen_cluster_auth (local-exec):     - system:masters

null_resource.gen_cluster_auth (local-exec): BinaryData
null_resource.gen_cluster_auth (local-exec): ====

null_resource.gen_cluster_auth (local-exec): Events: <none>
null_resource.gen_cluster_auth (local-exec): *****
null_resource.gen_cluster_auth: Creation complete after 10s [id=2959057410039000718]
null_resource.annotate: Still creating... [20s elapsed]
null_resource.annotate (local-exec): Found 2 nodes to annotate of 2
null_resource.annotate (local-exec): ip-10-0-1-180.eu-west-1.compute.internal eu-west-1a
null_resource.annotate (local-exec): kubectl annotate node ip-10-0-1-180.eu-west-1.compute.internal k8s.amazonaws.com/eniconfig.crd.k8s.amazonaws.com/eu-west-1a-pod-netconfig created
null_resource.annotate: Still creating... [30s elapsed]
```

& CI/CD VPC to the EKS Network

Connect the Cloud9 IDE & CID VPC to the EKS VPC

Terraform files
explanation

▼ 6. Deploy the CID Infrastructure

Create the CI/CD Components

Terraform files
explanation

▼ 7. EKS Cluster Creation

Creating the EKS Cluster

Terraform files
explanation

▼ 8. Create a customized managed Node Group

[Creating the EKS NodeGroup](#)

Test Networking

Terraform files
explanation

► 9. Enable AWS Load Balancers on EKS

► 10. Deploy a sample application

► 11. Private CI/CD for EKS

► 6. Extra Activities (Optional)

► 7. Using Fargate (Optional)

► Conclusion

► Cleanup

```
null_resource.annotate (local-exec): node/ip-10-0-1-180.eu-west-1.compute.internal annotated
null_resource.annotate (local-exec): ip-10-0-2-60.eu-west-1.compute.internal eu-west-1b
null_resource.annotate (local-exec): kubectl annotate node ip-10-0-2-60.eu-west-1.compute.internal k8s.amazonaws.com/eniCon
null_resource.annotate (local-exec): node/ip-10-0-2-60.eu-west-1.compute.internal annotated
null_resource.annotate (local-exec): Should see coredns on 100.64.x.y addresses .....
null_resource.annotate (local-exec): kubectl get pods -A -o wide | grep coredns
null_resource.annotate: Creation complete after 32s [id=5968836339862921197]
```

Apply complete! Resources: 4 added, 0 changed, 0 destroyed.

Outputs:

```
config-map-aws-auth = "local.config-map-aws-auth"
kubeconfig = "local.kubeconfig"
```

Check the custom software install

Our user_data.tf resource boot strapped our node into the cluster and installed the SSM agent.

You can check the SSM agent has worked by looking in the console for

Systems Manager then Fleet Manager

You should see the two worker node instances listed, as well as your Cloud9 IDE instance.

& CI/CD VPC to the EKS Network

Connect the Cloud9 IDE & CID VPC to the EKS VPC

Terraform files explanation

▼ 6. Deploy the CID Infrastructure

Create the CI/CD Components

Terraform files explanation

▼ 7. EKS Cluster Creation

Creating the EKS Cluster

Terraform files explanation

▼ 8. Create a customized managed Node Group

[Creating the EKS NodeGroup](#)

Test Networking

Terraform files explanation

► 9. Enable AWS Load Balancers on EKS

► 10. Deploy a sample application

► 11. Private CI/CD for EKS

► 6. Extra Activities (Optional)

► 7. Using Fargate (Optional)

► Conclusion

► Cleanup

The screenshot shows the AWS Systems Manager console. At the top, there's a navigation bar with the AWS logo, 'Services' dropdown, a search bar containing 'systems manager', and user information. Below the navigation bar, the breadcrumb trail reads 'AWS Systems Manager > Fleet Manager'. There are two tabs: 'Managed instances' (active) and 'Settings'. The main content area is titled 'Managed instances' and includes a search bar. Below the search bar is a table with three columns: 'Instance ID', 'Instance name', and 'SSM Agent status'. The table lists three instances, with the first one selected. To the right of the table, there are two dropdown menus: 'Instance actions' and 'Account management'. The 'Instance actions' menu is open, showing options like 'Connect', 'Start session', 'Admin tools', 'View file system', 'View performance counters', 'Manage users and groups', 'Instance settings', 'Reset password', 'Change IAM role in EC2 console', and 'Deregister this managed instance'. The 'Account management' dropdown shows the 'SSM Agent version' as '3.0.161.0'.

Instance ID	Instance name	SSM Agent status
i-07ac2822ccd4a043f	mycluster1-ng1	Online
i-02427e1b288721984	aws-cloud9-eks-terraform-f14252bc6a1e4ea3b2a8c2261f7b60c4	Online
i-0e93464e6f3ca91e1	mycluster1-ng1	Online

You can start a SSM session and login to the node if required.

Select a node, Instance actions and then Start session

This provides a more secure way to access worker nodes compared with allowing ssh based access. It also enables other Systems Manager capabilities such as automation, inventory collection and patching.

[Previous](#)

[Next](#)