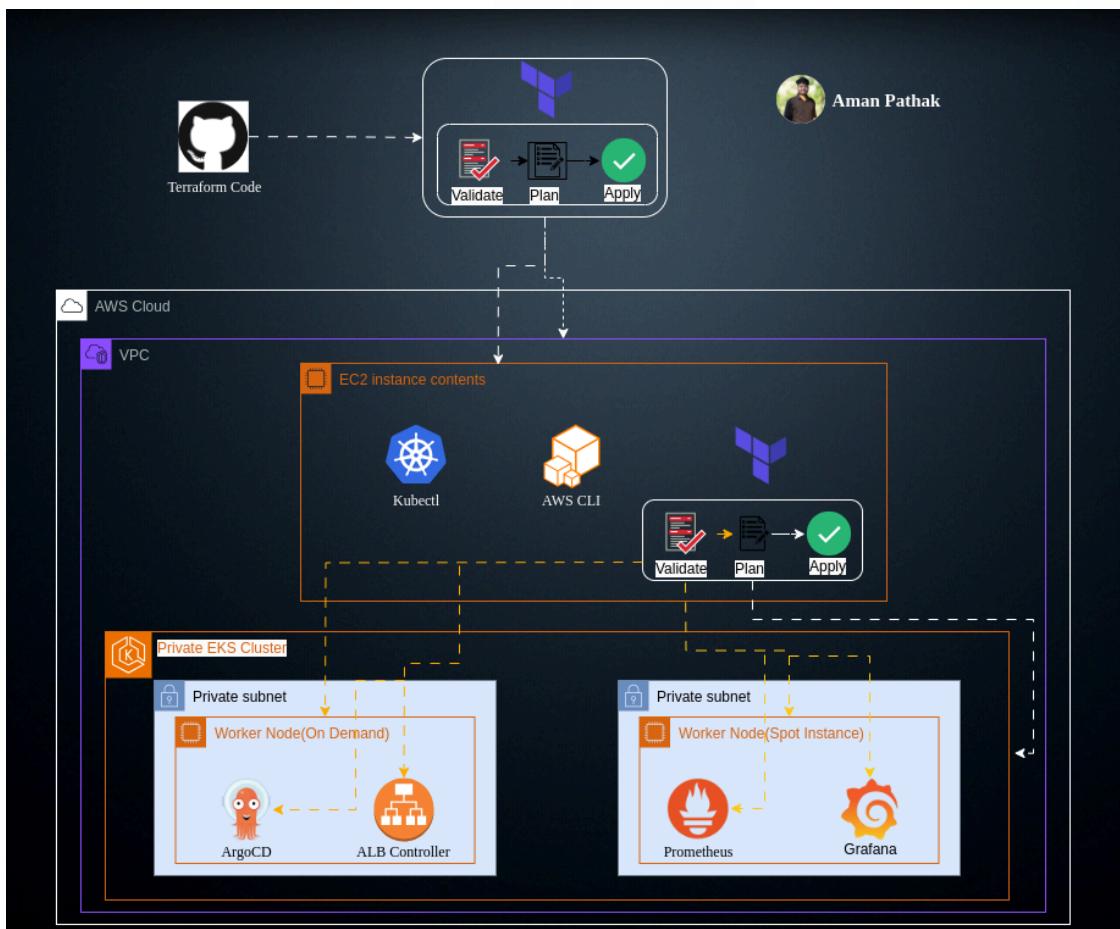


Configure ArgoCD, Prometheus, Grafana & AWS Load Balancer Controller on EKS Cluster using Terraform



Introduction

In today's DevOps-driven world, automating infrastructure deployment is crucial for maintaining efficiency and scalability. Setting up a secure and robust EKS (Elastic Kubernetes Service) cluster, complete with essential tools like ArgoCD, Prometheus, and Grafana, requires careful planning and execution. This guide will walk you through the entire process, from configuring your environment to deploying your infrastructure using Terraform, ensuring that your private EKS cluster is up and running smoothly with all the necessary resources.

Why Use Terraform for Configuring ArgoCD, Prometheus, and Other Tools?

Configuring tools like ArgoCD, Prometheus, and Grafana through Terraform offers several key advantages:

- 1. Infrastructure as Code (IaC):** Terraform allows you to define your entire infrastructure as code, making it versionable, repeatable, and easier to manage. This approach minimizes the risk of manual configuration errors and ensures consistency across environments.
- 2. Automated and Scalable Deployments:** Terraform automates the deployment process, enabling you to scale your infrastructure efficiently. Whether you're deploying in a development, staging, or production environment, Terraform ensures that all resources are provisioned and configured correctly with minimal manual intervention.
- 3. Easier Maintenance and Updates:** With Terraform, updating or modifying configurations for tools like ArgoCD and Prometheus is straightforward. You can manage changes centrally, apply updates consistently, and track modifications through version control, reducing the complexity of managing dynamic infrastructures.
- 4. Improved Security and Compliance:** By using Terraform, you can enforce security policies and compliance standards across your infrastructure. Terraform configurations can be audited, and changes can be tracked, helping to maintain a secure and compliant environment.

Prerequisites

Before diving into the deployment process, ensure that you have the following prerequisites in place:

1. **AWS Account:** You should have access to an AWS account with sufficient permissions to create and manage resources like VPCs, EC2 instances, and EKS clusters.
 2. **Terraform Installed:** Terraform must be installed on your local machine or the server from which you will manage the infrastructure.
 3. **Git:** Ensure Git is installed for cloning the necessary repositories.
 4. **AWS CLI:** The AWS CLI tool should be installed and configured with appropriate credentials for deploying resources on AWS.
 5. **Basic Knowledge of Terraform and Kubernetes:** Familiarity with Terraform and Kubernetes is essential to follow along with the steps and understand the infrastructure you're deploying.
-

Demonstration

GitHub Repository used in this demonstration-

<https://github.com/AmanPathak-DevOps/EKS-ArgoCD-AWS-LB-Controller-Terraform/tree/master>

So, I have created two branches Issue-branch and the other one is default named as master. Issue branch is to let you understand, why you can't deploy argoCD, Prometheus, and any other resources in your EKS Cluster.

This branch is 1 commit ahead of, 7 commits behind [master](#).

File	Commit Message	Time
eks	Initial Commit	yesterday
module	Initial Commit	yesterday
.gitignore	Initial Commit	yesterday
LICENSE	Initial commit	yesterday
README.md	Initial commit	yesterday

This is because your Cluster is Private. You need to be in the same network to enter your cluster and make any changes or configure something.

So, the Issue branch will create everything related to Cluster only such as EKS Cluster, Node groups, EKS Add-ons, etc. But when it tries to deploy argocd and other resources inside your cluster It will throw the error.

You can check the error in the below snippet.

```
module.eks.aws_eks.addon.eks-addons["3"]: Still creating... [20s elapsed]
module.eks.aws_eks.addon.eks-addons["0"]: Still creating... [30s elapsed]
module.eks.aws_eks.addon.eks-addons["3"]: Still creating... [30s elapsed]
module.eks.aws_eks.addon.eks-addons["2"]: Still creating... [30s elapsed]
module.eks.aws_eks.addon.eks-addons["3"]: Still creating... [40s elapsed]
module.eks.aws_eks.addon.eks-addons["0"]: Still creating... [40s elapsed]
module.eks.aws_eks.addon.eks-addons["2"]: Still creating... [40s elapsed]
module.eks.aws_eks.addon.eks-addons["2"]: Creation complete after 48s [id=dev-medium-eks-cluster:kube-proxy]
module.eks.aws_eks.addon.eks-addons["0"]: Creation complete after 50s [id=dev-medium-eks-cluster:vpc-cnt]
module.eks.aws_eks.addon.eks-addons["3"]: Still creating... [50s elapsed]
module.eks.aws_eks.addon.eks-addons["0"]: Creation complete after 51s [id=dev-medium-eks-cluster:vpc-cnt]
module.eks.aws_eks.addon.eks-addons["3"]: Creation complete after 59s [id=dev-medium-eks-cluster:aws-ebs-csi-driver]
data.aws_eks_cluster.eks-cluster: Reading...
data.aws_eks_cluster.eks-cluster: Read complete after 0s [id=dev-medium-eks-cluster]
kubernetes_namespace.argocd: Creating...
kubernetes_service_account.example: Creating...
kubernetes_service_account.example: Still creating... [10s elapsed]
kubernetes_namespace.argocd: Still creating... [10s elapsed]
kubernetes_namespace.argocd: Still creating... [20s elapsed]
kubernetes_service_account.example: Still creating... [20s elapsed]
kubernetes_service_account.example: Still creating... [30s elapsed]
kubernetes_namespace.argocd: Still creating... [30s elapsed]

Error: Post "https://16D6FBF9A7E7FC3417B0E71A310D655C.gr7.us-east-1.eks.amazonaws.com/api/v1/namespaces": dial tcp [64:ff9b:a10:8dfd]:443: i/o timeout
with kubernetes_namespace.argocd,
on argocd.tf line 1, in resource "kubernetes_namespace" "argocd":
 1: resource "kubernetes_namespace" "argocd" {
```



```
Error: Post "https://16D6FBF9A7E7FC3417B0E71A310D655C.gr7.us-east-1.eks.amazonaws.com/api/v1/namespaces/default/serviceaccounts": context deadline exceeded
with kubernetes_service_account.example,
on kubernetes.tf line 70, in resource "kubernetes_service_account" "example":
 70: resource "kubernetes_service_account" "example" {
```



```
Releasing state lock. This may take a few moments...
aman-pathak@aman:~/Projects/EKS-ArgoCD-AWS-LB-Controller-Terraform/eks$ git branch
* issue-branch
  master
```

P.S.: The EKS Cluster was configured as it was not visible in above snippet.

```

module.eks.aws_eks_node_group.on-demand-node: Still destroying... [id=dev-medium-eks-cluster:dev-medium-eks-cluster-on-demand-nodes, 2m30s elapsed]
module.eks.aws_eks_node_group.spot-node: Destruction complete after 2m31s
module.eks.aws_eks_node_group.on-demand-node: Still destroying... [id=dev-medium-eks-cluster:dev-medium-eks-cluster-on-demand-nodes, 2m40s elapsed]
module.eks.aws_eks_node_group.on-demand-node: Destruction complete after 2m42s
module.eks.aws_iam_role.eks-nodegroup-role[0]: Destroying... [id=dev-medium-eks-cluster-nodegroup-role-2377]
module.eks.aws_eks_cluster.eks[0]: Destroying... [id=dev-medium-eks-cluster]
module.eks.aws_iam_role.eks-nodegroup-role[0]: Destruction complete after 3s
module.eks.aws_eks_cluster.eks[0]: Still destroying... [id=dev-medium-eks-cluster, 10s elapsed]
module.eks.aws_iam_role.eks-nodegroup-role[0]: Still destroying... [id=dev-medium-eks-cluster, 20s elapsed]
module.eks.aws_iam_role.eks-nodegroup-role[0]: Still destroying... [id=dev-medium-eks-cluster, 30s elapsed]
module.eks.aws_iam_role.eks-nodegroup-role[0]: Still destroying... [id=dev-medium-eks-cluster, 40s elapsed]
module.eks.aws_iam_role.eks-nodegroup-role[0]: Still destroying... [id=dev-medium-eks-cluster, 50s elapsed]
module.eks.aws_iam_role.eks-nodegroup-role[0]: Still destroying... [id=dev-medium-eks-cluster, 1m0s elapsed]
module.eks.aws_iam_role.eks-nodegroup-role[0]: Still destroying... [id=dev-medium-eks-cluster, 1m10s elapsed]
module.eks.aws_iam_role.eks-nodegroup-role[0]: Still destroying... [id=dev-medium-eks-cluster, 1m20s elapsed]
module.eks.aws_iam_role.eks-nodegroup-role[0]: Still destroying... [id=dev-medium-eks-cluster, 1m30s elapsed]
module.eks.aws_iam_role.eks-nodegroup-role[0]: Still destroying... [id=dev-medium-eks-cluster, 1m40s elapsed]
module.eks.aws_iam_role.eks-nodegroup-role[0]: Still destroying... [id=dev-medium-eks-cluster, 1m50s elapsed]
module.eks.aws_iam_role.eks-nodegroup-role[0]: Still destroying... [id=dev-medium-eks-cluster, 2m0s elapsed]
module.eks.aws_iam_role.eks-nodegroup-role[0]: Still destroying... [id=dev-medium-eks-cluster, 2m10s elapsed]
module.eks.aws_iam_role.eks-nodegroup-role[0]: Still destroying... [id=dev-medium-eks-cluster, 2m20s elapsed]
module.eks.aws_iam_role.eks-nodegroup-role[0]: Still destroying... [id=dev-medium-eks-cluster, 2m30s elapsed]
module.eks.aws_iam_role.eks-nodegroup-role[0]: Still destroying... [id=dev-medium-eks-cluster, 2m40s elapsed]
module.eks.aws_iam_role.eks-nodegroup-role[0]: Still destroying... [id=dev-medium-eks-cluster, 2m50s elapsed]
module.eks.aws_iam_role.eks-nodegroup-role[0]: Destruction complete after 2m51s
module.eks.aws_iam_role.eks-cluster-role[0]: Destroying... [id=dev-medium-eks-cluster-role-2377]
module.eks.aws_subnet.private-subnet[0]: Destroying... [id=subnet-00204fcf95bcd3cd1]
module.eks.aws_subnet.private-subnet[2]: Destroying... [id=subnet-0fd9e98584aa5a3a]
module.eks.aws_subnet.private-subnet[1]: Destroying... [id=subnet-07b6dd58ba103d26f]
module.eks.aws_security_group.eks-cluster-sg: Destroying... [id=sg-0ce8e07f99d14c4a3]
module.eks.aws_iam_role.eks-cluster-role[0]: Destruction complete after 2s
module.eks.random_integer.random_suffix: Destroying... [id=2377]
module.eks.random_integer.random_suffix: Destruction complete after 0s
module.eks.aws_subnet.private-subnet[0]: Destruction complete after 3s
module.eks.aws_subnet.private-subnet[2]: Destruction complete after 3s
module.eks.aws_subnet.private-subnet[1]: Destruction complete after 4s
module.eks.aws_security_group.eks-cluster-sg: Destruction complete after 4s
module.eks.aws_vpc.vpc: Destroying... [id=vpc-0230937f800bc75d5e]
module.eks.aws_vpc.vpc: Destruction complete after 2s
Releasing state lock. This may take a few moments...

```

Destroy complete! Resources: 40 destroyed.

How to solve this?

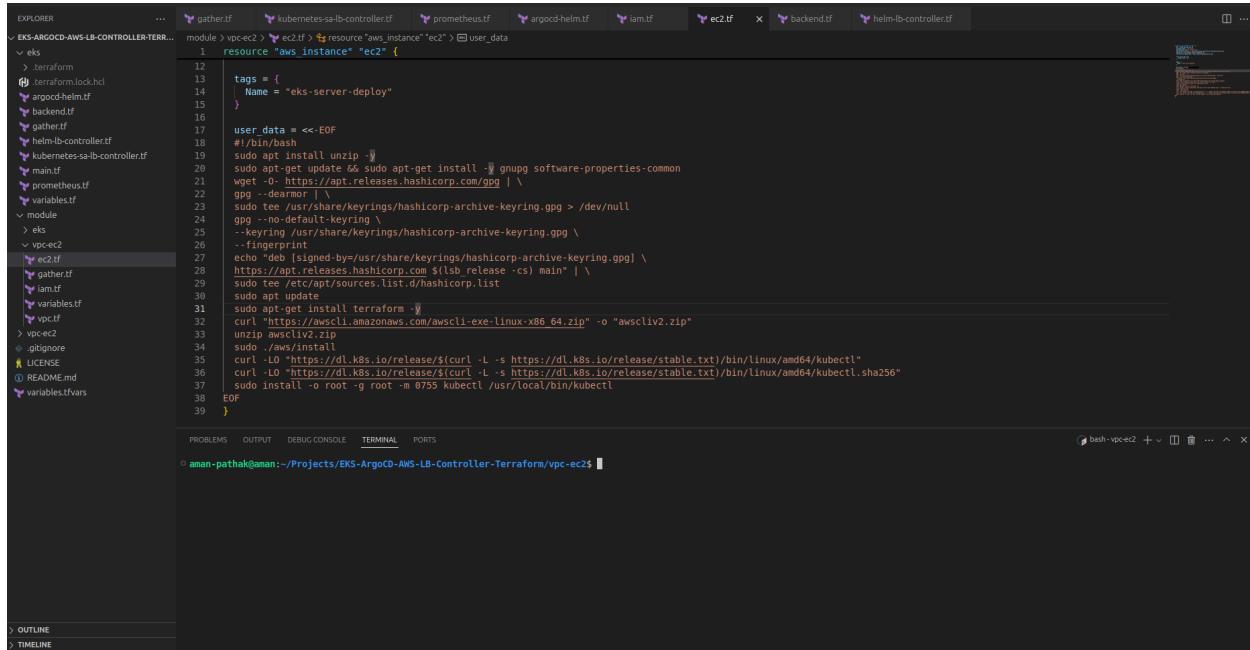
To deploy anything inside your EKS Cluster, we need an instance or server to deploy those configurations that have the same VPC.

So, we are going to deploy our VPC and one EC2 server first. Then from the ec2 server, we will deploy everything related to the EKS Cluster.

In the master branch of the repository, you will find two modules. The first module is vpc-ec2 which we need to apply first.

Commit	Message	Time Ago
AmanPathak-DevOps: File Structure updated	Updated naming	1 hour ago
AmanPathak-DevOps: Updated policies for ec2		16 minutes ago
AmanPathak-DevOps: File Structure updated		now
.gitignore	Initial Commit	19 hours ago
LICENSE	Initial commit	yesterday
README.md	Initial commit	yesterday
variables.tfvars	Added minor changes	7 hours ago

So, clone the repository with the master branch and navigate to the vpc-ec2 directory.



```
resource "aws_instance" "ec2" {
  tags = {
    Name = "eks-server-deploy"
  }

  user_data = <<EOF
#!/bin/bash
sudo apt-get update && sudo apt-get install -y gnupg software-properties-common
wget -O https://apt.releases.hashicorp.com/gpg | \
gpg --dearmor | \
sudo tee /usr/share/keyrings/hashicorp-archive-keyring.gpg > /dev/null
gpg --no-default-keyring \
--keyring /usr/share/keyrings/hashicorp-archive-keyring.gpg \
--fingerprint
echo "deb [signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg] \
https://apt.releases.hashicorp.com $(lsb_release -cs) main" | \
sudo tee /etc/apt/sources.list.d/hashicorp.list
sudo apt update
sudo apt-get install terraform -y
curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
unzip awscliv2.zip
sudo ./aws/install
curl -Lo "https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl"
curl -Lo "https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl.sha256"
sudo install -o root -g root -m 0755 kubectl /usr/local/bin/kubectl
EOF
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

aman-pathak@aman:~/Projects/EKS-ArgoCD-AWS-LB-Controller-Terraform/vpc-ec2\$

Run the below commands to deploy VPC(Other networking resources) & EC2 Server on AWS.

> **terraform init**

```
aman-pathak@aman:~/Projects/EKS-ArgoCD-AWS-LB-Controller-Terraform/vpc-ec2$ terraform init
Initializing the backend...
Successfully configured the backend "s3"! Terraform will automatically
use this backend unless the backend configuration changes.

Initializing modules...
- vpc-ec2 in ../module/vpc-ec2
Initializing provider plugins...
- Finding hashicorp/aws versions matching "> 5.49.0"...
- Finding hashicorp/kubernetes versions matching "2.31.0"...
- Installing hashicorp/kubernetes v2.31.0...
- Installed hashicorp/kubernetes v2.31.0 (signed by HashiCorp)
- Installing hashicorp/aws v5.49.0...
- Installed hashicorp/aws v5.49.0 (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.
aman-pathak@aman:~/Projects/EKS-ArgoCD-AWS-LB-Controller-Terraform/vpc-ec2$
```

> **terraform validate**

```
▶ aman-pathak@aman:~/Projects/EKS-ArgoCD-AWS-LB-Controller-Terraform/vpc-ec2$ terraform validate
Success! The configuration is valid.

○ aman-pathak@aman:~/Projects/EKS-ArgoCD-AWS-LB-Controller-Terraform/vpc-ec2$
```

```
> terraform plan -var-file=../variables.tfvars
```

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
+ 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                    = {
    + "Env"  = "dev"
    + "Name" = "dev-medium-vpc"
}
+ tags_all                = {
    + "Env"  = "dev"
    + "Name" = "dev-medium-vpc"
}
}

Plan: 24 to add, 0 to change, 0 to destroy.

Warning: Value for undeclared variable
The root module does not declare a variable named "ondemand_instance_types" but a value was found in file "../variables.tfvars". If you meant to use this value, add a "variable" block to the configuration.

To silence these warnings, use TF_VAR_... environment variables to provide certain "global" settings to all configurations in your organization. To reduce the verbosity of these warnings, use the -compact-warnings option.

Warning: Value for undeclared variable
The root module does not declare a variable named "max_capacity_on_demand" but a value was found in file "../variables.tfvars". If you meant to use this value, add a "variable" block to the configuration.

To silence these warnings, use TF_VAR_... environment variables to provide certain "global" settings to all configurations in your organization. To reduce the verbosity of these warnings, use the -compact-warnings option.

Warning: Values for undeclared variables
In addition to the other similar warnings shown, 11 other variable(s) defined without being declared.

Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if you run "terraform apply" now.
Releasing state lock. This may take a few moments...
aman-pathak@aman:~/Projects/EKS-ArgoCD-AWS-LB-Controller-Terraform/vpc-ec2$
```

```
> terraform apply -auto-approve -var-file=../variables.tfvars
```

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
module.vpc-ec2.aws_instance.ec2: Still creating... [10s elapsed]
module.vpc-ec2.aws_nat_gateway.ngw: Still creating... [20s elapsed]
module.vpc-ec2.aws_instance.ec2: Still creating... [20s elapsed]
module.vpc-ec2.aws_nat_gateway.ngw: Still creating... [30s elapsed]
module.vpc-ec2.aws_instance.ec2: Still creating... [30s elapsed]
module.vpc-ec2.aws_instance.ec2: Creation complete after 37s [id=i-0d8a3bc509812b65b]
module.vpc-ec2.aws_nat_gateway.ngw: Still creating... [40s elapsed]
module.vpc-ec2.aws_nat_gateway.ngw: Still creating... [50s elapsed]
module.vpc-ec2.aws_nat_gateway.ngw: Still creating... [60s elapsed]
module.vpc-ec2.aws_nat_gateway.ngw: Still creating... [1m0s elapsed]
module.vpc-ec2.aws_nat_gateway.ngw: Still creating... [1m10s elapsed]
module.vpc-ec2.aws_nat_gateway.ngw: Still creating... [1m20s elapsed]
module.vpc-ec2.aws_nat_gateway.ngw: Still creating... [1m30s elapsed]
module.vpc-ec2.aws_route_table.private_rt: Creating...
module.vpc-ec2.aws_route_table.private_rt: Creation complete after 3s [id=rtb-bb3db5ef7d41bab3]
module.vpc-ec2.aws_route_table_association.private_rt_association[0]: Creating...
module.vpc-ec2.aws_route_table_association.private_rt_association[1]: Creating...
module.vpc-ec2.aws_route_table_association.private_rt_association[0]: Creating...
module.vpc-ec2.aws_route_table_association.private_rt_association[1]: Creation complete after 2s [id=rtbassoc-0a46e7018fb044c]
module.vpc-ec2.aws_route_table_association.private_rt_association[2]: Creating...
module.vpc-ec2.aws_route_table_association.private_rt_association[2]: Creation complete after 2s [id=rtbassoc-0de5af70add86803]
module.vpc-ec2.aws_route_table_association.private_rt_association[1]: Creation complete after 2s [id=rtbassoc-00599d70a94f9b8fc]

Warning: Value for undeclared variable
The root module does not declare a variable named "max_capacity_on_demand" but a value was found in file "../variables.tfvars". If you meant to use this value, add a "variable" block to the configuration.

To silence these warnings, use TF_VAR_... environment variables to provide certain "global" settings to all configurations in your organization. To reduce the verbosity of these warnings, use the -compact-warnings option.

Warning: Value for undeclared variable
The root module does not declare a variable named "min_capacity_spot" but a value was found in file "../variables.tfvars". If you meant to use this value, add a "variable" block to the configuration.

To silence these warnings, use TF_VAR_... environment variables to provide certain "global" settings to all configurations in your organization. To reduce the verbosity of these warnings, use the -compact-warnings option.

Warning: Values for undeclared variables
In addition to the other similar warnings shown, 11 other variable(s) defined without being declared.

Releasing state lock. This may take a few moments...

Apply complete! Resources: 24 added, 0 changed, 0 destroyed.
aman@aman:~/Projects/EKS-ArgoCD-AWS-LB-Controller-Terraform/vpc-ec2$ 

```

VPC & other services created through the above commands

Here are the snippets

VPC

Name	VPC ID	State	IPv4 CIDR	IPv6 CIDR	DHCP option set	Main route table
dev-medium-vpc	vpc-053477193e826f6c2	Available	10.16.0.0/16	-	dopt-0adcb6e608dea91...	rtb-03a210971559fa011

Subnets

Name	Subnet ID	State	VPC	IPv4 CIDR	IPv6 CIDR	IPv6 CIDR association ID
dev-medium-subnet-public-3	subnet-0529f20956b2ded5c	Available	vpc-053477193e826f6c2 dev...	10.16.32.0/20	-	-
dev-medium-subnet-public-1	subnet-00cb711dd4cc83bc	Available	vpc-053477193e826f6c2 dev...	10.16.0.0/20	-	-
dev-medium-subnet-private-2	subnet-00a790a9602055776	Available	vpc-053477193e826f6c2 dev...	10.16.144.0/20	-	-
dev-medium-subnet-private-1	subnet-03676bf1f1636a205e	Available	vpc-053477193e826f6c2 dev...	10.16.128.0/20	-	-
dev-medium-subnet-private-3	subnet-007d9f2dfdf45d05	Available	vpc-053477193e826f6c2 dev...	10.16.160.0/20	-	-
dev-medium-subnet-public-2	subnet-0b600ba0303c3a40f	Available	vpc-053477193e826f6c2 dev...	10.16.16.0/20	-	-

Route tables

Name	Route table ID	Explicit subnet associations	Edge associations	Main	VPC	Owner ID
dev-medium-public-route-table	rtb-06fb91ac0cf65e1e80	3 subnets	-	No	vpc-053477193e826f6c2 dev...	407622020962
dev-medium-private-route-table	rtb-0b5dfb5ef7d41bab3	3 subnets	-	No	vpc-053477193e826f6c2 dev...	407622020962
-	rtb-05a210971559fa011	-	-	Yes	vpc-053477193e826f6c2 dev...	407622020962

Internet Gateway

The screenshot shows the AWS VPC dashboard with the 'Internet gateways' section selected. It displays two internet gateways:

Name	Internet gateway ID	State	VPC ID	Owner
-	igw-035c0082b1f4d2aa5	Attached	vpc-0a40de8a185b59650	407622020962
dev-medium-igw	igw-081e78f5db8daa6a4	Attached	vpc-053477195e026f6c2 dev-medium...	407622020962

NAT Gateway

The screenshot shows the AWS VPC dashboard with the 'NAT gateways' section selected. It displays one NAT gateway:

Name	NAT gateway ID	Connectivity...	State	State message	Primary public I...	Primary private I...	Primary network...	VPC
dev-medium-ngw	nat-0e920b64e01b765	Public	Available	-	3.139.11.167	10.16.8.110	eni-09e795a350611...	vpc-053477195e026f6c2

Security Group

The screenshot shows the AWS EC2 dashboard with the 'Security Groups' section selected. It displays three security groups:

Name	Security group ID	Security group name	VPC ID	Description	Owner
eks-sg	sg-071f0bf4b1928202e	ec2-sg	vpc-053477195e026f6c2	Allow 443 from Jump Server only	407622020962
-	sg-0d644b1be403e5a0d	default	vpc-053477195e026f6c2	default VPC security group	407622020962
eks-sg	sg-0d1c9a4552d0bcfa9f	eks-sg	vpc-053477195e026f6c2	Allow 443 from Jump Server only	407622020962

EC2-Instance

The screenshot shows the AWS EC2 dashboard with the 'Instances' section selected. It displays one EC2 instance:

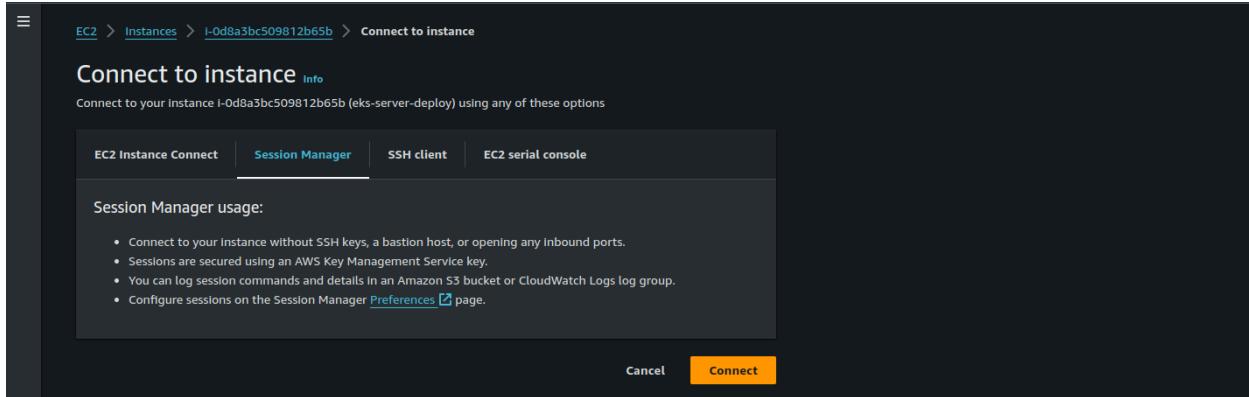
Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elastic IP
eks-server-deploy	i-0d0a3bc509812b65b	Running	t2.micro	2/2 checks passed	-	us-east-2a	ec2-18-191-116-22.us...	18.191.116.22	-

So, everything is created according to our requirements.

Now log in to the Server by selecting the created ec2 instance and click on Connect

You can log in without the Pem file as we did not follow that.

If Session Manager is struggling to connect to the instance, you can use EC2 Instance Connect to login to the server.



I am logged In to my server and switched to the Ubuntu user with the help of the below command

```
> sudo su ubuntu
```

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

ubuntu@ip-10-16-10-26:~$ var/snap/amazon-ssm-agent/7993$ cd
ubuntu@ip-10-16-10-26:~$
```

We need to install a few tools as our pre-requisites. We installed it through user data in the Terraform script

Run the below command to validate whether it's installed or not. If not, wait for 5-10 minutes,

```
> terraform version
aws --version
kubectl version
```

```
ubuntu@ip-10-16-10-26:~$ terraform version
Terraform v0.9.5
on linux_amd64
ubuntu@ip-10-16-10-26:~$ aws --version
aws-cli/2.17.42 Python/3.11.9 Linux/6.5.0-1024-aws exe/x86_64.ubuntu.22
ubuntu@ip-10-16-10-26:~$ kubectl version
Client Version: v1.26.2
Kustomize Version: v5.4.2
The connection to the server localhost:8080 was refused - did you specify the right host or port?
ubuntu@ip-10-16-10-26:~$
```

Once all the tools are installed then, we need to configure AWS CLI as we need to deploy our infrastructure over AWS Cloud.

Run the below command to configure CLI and make sure you have sufficient permission with your credentials. For demonstration, you can utilize Administration Access user keys

P.S.: Don't use the same access keys as you will have your keys in your AWS Account

```
> aws configure
```

```
Session ID: Adorable-Aman-ed12c7a0whlbgmrvnbt7ha53b Instance ID: i-0d8a3bc509812b65b
Terminate
ubuntu@ip-10-16-10-26:~$ aws configure
AWS Access Key [None]: e250Pn3MKHtAMpwB0i/epCunsoommqqZflfqhk
AWS Secret Access Key [None]: ez50Pn3MKHtAMpwB0i/epCunsoommqqZflfqhk
Default region name [None]: us-east-2
Default output format [None]: json
ubuntu@ip-10-16-10-26:~$
```

Now, clone the same repository where our Terraform code is present for EKS

```
> git clone
https://github.com/AmanPathak-DevOps/EKS-ArgoCD-AWS-LB-Controller-Terraform.git
```

```
Session ID: Adorable-Aman-ed12c7a0whlbgmrvnbt7ha53b Instance ID: i-0d8a3bc509812b65b
Terminate
ubuntu@ip-10-16-10-26:~$ git clone https://github.com/AmanPathak-DevOps/EKS-ArgoCD-AWS-LB-Controller-Terraform.git
Cloning into 'EKS-ArgoCD-AWS-LB-Controller-Terraform'...
Username for 'https://github.com': AmanPathak-DevOps
Password for 'https://AmanPathak-DevOps@github.com':
remote: Enumerating objects: 85, done.
remote: Counting objects: 100% (85/85), done.
remote: Compressing objects: 100% (80/80), done.
remote: Total 85 (delta 40), reused 64 (delta 23), pack-reused 0 (from 0)
Resolving deltas: 100% (85/85), 19.61 KiB | 1.73 MiB/s, done.
Resolving deltas: 100% (85/85), done.
ubuntu@ip-10-16-10-26:~$ git clone https://github.com/AmanPathak-DevOps/EKS-ArgoCD-AWS-LB-Controller-Terraform.git
ubuntu@ip-10-16-10-26:~$ ls
EKS-ArgoCD-AWS-LB-Controller-Terraform
ubuntu@ip-10-16-10-26:~$
```

Now, we are ready to deploy our EKS Cluster, and other tools through Terraform. Navigate to the eks directory and run the below commands to deploy it

```
> terraform init
```

```
Session ID: Adorable-Aman-ed12c7a0whlbgmrvnbt7ha53b Instance ID: i-0d8a3bc509812b65b
Terminate
ubuntu@ip-10-16-10-26:~/EKS-ArgoCD-AWS-LB-Controller-Terraform/eks$ terraform init
Initializing the backend...
Successfully configured the backend "s3"! Terraform will automatically
use this backend unless the backend configuration changes.
Initializing modules...
- eks in ./module/eks
Initializing provider plugins...
- Finding latest version of hashicorp/tls...
- Finding hashicorp/helm versions matching "> 2.10.0"...
- Finding hashicorp/aws versions matching "> 5.49.0"...
- Finding hashicorp/kubernetes versions matching "2.31.0"...
- Finding latest version of hashicorp/random...
  Installing hashicorp/random v3.6.2...
  Installed hashicorp/random v3.6.2 (signed by HashiCorp)
  Installing hashicorp/tls v4.0.5...
  Installed hashicorp/tls v4.0.5 (signed by HashiCorp)
  Installing hashicorp/helm v2.10.1...
  Installed hashicorp/helm v2.10.1 (signed by HashiCorp)
  Installing hashicorp/aws v5.49.0...
  Installed hashicorp/aws v5.49.0 (signed by HashiCorp)
  - Installing hashicorp/kubernetes v2.31.0...
  - Installed hashicorp/kubernetes v2.31.0 (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-10-16-10-26:~/EKS-ArgoCD-AWS-LB-Controller-Terraform/eks$
```

```
> terraform validate
```

```
Session ID: Adorable-Aman-ed12c7a0whlbgmrvnbt7ha53b Instance ID: i-0d8a3bc509812b65b
Terminate
ubuntu@ip-10-16-10-26:~/EKS-ArgoCD-AWS-LB-Controller-Terraform/eks$ terraform validate
Success! The configuration is valid.
ubuntu@ip-10-16-10-26:~/EKS-ArgoCD-AWS-LB-Controller-Terraform/eks$
```

```
> terraform plan -var-file=../variables.tfvars
```

```
Session ID: Adorable-Aman-edf27uwfhblqgmrwth7hzf3b
Instance ID: i-0d8a5bc509812b65b
Terminate



```

}
module.eks.aws_iam_role_policy_attachment.eks AMAZONWorkerNodePolicy[0] will be created
+ resource "aws_iam_role_policy_attachment" "eks AMAZONWorkerNodePolicy" {
+ id = (known after apply)
+ role_arn = "arn:aws:iam::aws:policy/AmazonEKSWorkerNodePolicy"
+ policy_arn = (known after apply)
}

module.eks.aws_iam_role_policy_attachment.eks-oidc-policy-attach will be created
+ resource "aws_iam_role_policy_attachment" "eks-oidc-policy-attach" {
+ id = (known after apply)
+ role_arn = (known after apply)
+ policy_arn = "eks-oidc"
}

module.eks.random_integer.random_suffix will be created
resource "random_integer" "random_suffix" {
+ id = (known after apply)
+ max = 9999
+ min = 9999
+ result = (Known after apply)
}

Plan: 25 to add, 0 to change, 0 to destroy.

Warning: Value for undeclared variable
The root module does not declare a variable named "ec2-sg" but a value was found in file "../variables.tfvars". If you meant to use this value, add a "variable" block to the configuration.
To silence these warnings, use TF_VAR... environment variables to provide certain "global" settings to all configurations in your organization. To reduce the verbosity of these warnings, use the -compact-warnings option.

Warning: Value for undeclared variable
The root module does not declare a variable named "ec2-name" but a value was found in file "../variables.tfvars". If you meant to use this value, add a "variable" block to the configuration.
To silence these warnings, use TF_VAR... environment variables to provide certain "global" settings to all configurations in your organization. To reduce the verbosity of these warnings, use the -compact-warnings option.

Warning: Values for undeclared variables
In addition to the other similar warnings shown, 3 other variable(s) defined without being declared.

Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if you run "terraform apply" now.
ubuntu@ip-10-16-10-26:~/EKS-ArgoCD-AWS-LB-Controller-Terraform/eks$
```


```

```
> terraform apply -auto-approve -var-file=../variables.tfvars
```

```
Session ID: Adorable-Aman-edf27uwfhblqgmrwth7hzf3b
Instance ID: i-0d8a5bc509812b65b
Terminate

module.eks.aws_eks_addon.eks-addons["2"]: Still creating... [5s elapsed]
module.eks.aws_eks_addon.eks-addons["3"]: Still creating... [5s elapsed]
module.eks.aws_eks_addon.eks-addons["3"]: Still creating... [5s elapsed]
module.eks.aws_eks_addon.eks-addons["3"]: Creation complete after 55s [id=dev-medium-eks-cluster:aws-ebs-csi-driver]
module.eks.aws_eks_addon.eks-addons["2"]: Creation complete after 55s [id=dev-medium-eks-cluster:kube-proxy]
module.eks.aws_eks_addon.eks-addons["3"]: Creation complete after 55s [id=dev-medium-eks-cluster:vpc-cni]
data.aws_eks_cluster.eks-cluster: Reading...
data.aws_eks_cluster.eks-cluster: Read complete after 0s [id=dev-medium-eks-cluster]
kubernetes_service_account.lb-controller: Creating...
kubernetes_service_account.lb-controller: Creation complete after 0s [id=default/aws-load-balancer-controller]
helm.release.aws_load_balancer_controller: Still creating... [10s elapsed]
helm.release.aws_load_balancer_controller: Still creating... [10s elapsed]
helm.release.aws_load_balancer_controller: Creation complete after 10s [id=aws-load-balancer-controller]
helm.release.argo: Creating...
helm.release.argo: Still creating... [10s elapsed]
helm.release.argo: Still creating... [10s elapsed]
helm.release.argo: Still creating... [30s elapsed]
helm.release.argo: Creation complete after 39s [id=argood]
helm.release.prometheus-helm: Creating...
helm.release.prometheus-helm: Still creating... [50s elapsed]
helm.release.prometheus-helm: Still creating... [20s elapsed]
helm.release.prometheus-helm: Still creating... [30s elapsed]
helm.release.prometheus-helm: Still creating... [40s elapsed]
helm.release.prometheus-helm: Still creating... [50s elapsed]
helm.release.prometheus-helm: Creation complete after 1m11s [id=prometheus]

Warning: Value for undeclared variable
The root module does not declare a variable named "ec2-name" but a value was found in file "../variables.tfvars". If you meant to use this value, add a "variable" block to the configuration.
To silence these warnings, use TF_VAR... environment variables to provide certain "global" settings to all configurations in your organization. To reduce the verbosity of these warnings, use the -compact-warnings option.

Warning: Value for undeclared variable
The root module does not declare a variable named "ec2-sg" but a value was found in file "../variables.tfvars". If you meant to use this value, add a "variable" block to the configuration.
To silence these warnings, use TF_VAR... environment variables to provide certain "global" settings to all configurations in your organization. To reduce the verbosity of these warnings, use the -compact-warnings option.

Warning: Values for undeclared variables
In addition to the other similar warnings shown, 3 other variable(s) defined without being declared.

Apply complete! Resources: 25 added, 0 changed, 0 destroyed.
ubuntu@ip-10-16-10-26:~/EKS-ArgoCD-AWS-LB-Controller-Terraform/eks$
```

**EKS Cluster & other services created through the above commands
Here are the snippets**

EKS Cluster

The screenshot shows the AWS EKS Clusters page. At the top, there's a header with 'Clusters' and a sub-header 'Clusters (1) Info'. Below that is a search bar with 'Filter clusters'. On the right, there are buttons for 'Delete' and 'Add cluster'. The main table has columns: Cluster name, Status, Kubernetes version, Support period, Upgrade policy, Created, Provider, and Actions. One row is shown for 'dev-medium-eks-cluster', which is Active, running Kubernetes version 1.29, with support ending on March 23, 2025, and created 16 minutes ago by EKS.

Node Groups

This screenshot shows the details for the 'dev-medium-eks-cluster'. The top navigation includes 'Clusters' and the specific cluster name. On the right are buttons for 'Delete cluster' and 'Upgrade version'. A note about standard support ending on March 23, 2025, is displayed, along with an 'Upgrade now' button. The main content area has tabs for 'Overview', 'Resources', 'Compute' (which is selected), 'Networking', 'Add-ons', 'Access', 'Observability', 'Upgrade insights', 'Update history', and 'Tags'. Under 'Compute', the 'Nodes (2) Info' section lists two nodes: 'ip-10-16-147-226.us-east-2.compute.internal' (t3a.medium, created 4 minutes ago, ready) and 'ip-10-16-157-126.us-east-2.compute.internal' (t3a.large, created 4 minutes ago, ready). The 'Node groups (2) Info' section shows two groups: 'dev-medium-eks-cluster-on-demand-nodes' and 'dev-medium-eks-cluster-spot-nodes'.

To validate whether the other Kubernetes resources have been created or not. We need to update the kubeconfig on our same ec2 server
Run the below command to update the kubeconfig

```
> aws eks update-kubeconfig --name dev-medium-eks-cluster
--region us-east-2
```

A terminal session on an EC2 instance (Session ID: Adorable-Aman-estjz7owhjlegmrvutb7hf3h, Instance ID: i-0d8a3bc509812b65b) executing the command 'aws eks update-kubeconfig --name dev-medium-eks-cluster --region us-east-2'. The output shows the new context 'arn:aws:eks:us-east-2:240622020962:cluster/dev-medium-eks-cluster' added to the kubeconfig file.

Run the below command to validate whether it's our cluster or not

```
> kubectl get nodes
```

A terminal session on an EC2 instance (Session ID: Adorable-Aman-estjz7owhjlegmrvutb7hf3h, Instance ID: i-0d8a3bc509812b65b) executing 'kubectl get nodes'. The output lists two nodes: 'ip-10-16-147-226.us-east-2.compute.internal' and 'ip-10-16-157-126.us-east-2.compute.internal', both in a 'Ready' state.

Check whether the resources are running in argocd namespace or not

> kubectl get all -n argocd

```
Session ID: Adorable-Aman-ed2c2whhbgmuvtb7hjz30l Instance ID: i-0d8a3bc509812b65b Terminate
ubuntu@ip-10-16-10-26:~/EKS-ArgoCD-AWS-LB-Controller-Terraform/eks$ kubectl get all -n argocd
NAME                                     READY   STATUS    RESTARTS   AGE
pod/argocd-application-controller-0      1/1    Running   0          4m32s
pod/argocd-applicationset-controller-0   1/1    Running   0          4m33s
pod/argocd-dex-server-6d8bcbf4db-4c8st  1/1    Running   0          4m33s
pod/argocd-notifications-controller-7df6fd8fd4-wzsjs 1/1    Running   0          4m33s
pod/argocd-redis-57bd77ab7d-sqgwx    1/1    Running   0          4m33s
pod/argocd-repo-server-8476665c5b-cubcb 1/1    Running   0          4m33s
pod/argocd-server-84d0774fb8-bhtmv    1/1    Running   0          4m33s

NAME          TYPE        CLUSTER-IP      EXTERNAL-IP      PORT(S)          AGE
service/argocd-applicationset-controller ClusterIP   172.28.100.9   <none>           80/TCP           4m33s
service/argocd-dex-server                 ClusterIP   172.28.100.78  <none>           53/TCP           4m33s
service/argocd-redis                     ClusterIP   172.28.140.141 <none>           6379/TCP         4m33s
service/argocd-repo-server                ClusterIP   172.28.100.123 <none>           8081/TCP         4m33s
service/argocd-server                    LoadBalancer 172.28.232.62 a6360afb259146559274c90b23d69be-105956856.us-east-2.elb.amazonaws.com 80:31297/TCP, 443:31884/TCP 4m33s

NAME          READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/argocd-applicationset-controller 1/1     1           1           4m33s
deployment.apps/argocd-dex-server              1/1     1           1           4m33s
deployment.apps/argocd-notifications-controller 1/1     1           1           4m33s
deployment.apps/argocd-redis                  1/1     1           1           4m33s
deployment.apps/argocd-repo-server             1/1     1           1           4m33s
deployment.apps/argocd-server                 1/1     1           1           4m33s

NAME          DESIRED  CURRENT  READY   AGE
replicaset.apps/argocd-applicationset-controller-765cfb7674 1       1       1       4m33s
replicaset.apps/argocd-dex-server-6d8bcbf4db 1       1       1       4m33s
replicaset.apps/argocd-notifications-controller-7df6fd8fd4 1       1       1       4m33s
replicaset.apps/argocd-redis-57bd77ab7d 1       1       1       4m33s
replicaset.apps/argocd-repo-server-8476665c5b 1       1       1       4m33s
replicaset.apps/argocd-server-84d0774fb8 1       1       1       4m33s

NAME          READY   AGE
statefulset.apps/argocd-application-controller 1/1     4m33s
ubuntu@ip-10-16-10-26:~/EKS-ArgoCD-AWS-LB-Controller-Terraform/eks$
```

Check whether the resources are running in Prometheus namespace or not

> kubectl get all -n prometheus

```
Session ID: Adorable-Aman-ed2c2whhbgmuvtb7hjz30l Instance ID: i-0d8a3bc509812b65b Terminate
ubuntu@ip-10-16-10-26:~/EKS-ArgoCD-AWS-LB-Controller-Terraform/eks$ kubectl get all -n prometheus
NAME                                     READY   STATUS    RESTARTS   AGE
pod/alertmanager-prometheus-kube-prometheus-alertmanager-0 2/2    Running   0          4m49s
pod/prometheus-grafana-6cc4498afc-9p4fd 3/3    Running   0          4m52s
pod/prometheus-kube-prometheus-operator-76647ff58db-t58k2 1/1    Running   0          4m52s
pod/prometheus-kube-prometheus-prometheus-7579b-59j24 1/1    Running   0          4m52s
pod/prometheus-prometheus-kube-prometheus-prometheus-0 2/2    Running   0          4m49s
pod/prometheus-prometheus-node-exporter-5ngj7 1/1    Running   0          4m52s
pod/prometheus-prometheus-node-exporter-64rgd 1/1    Running   0          4m52s

NAME          TYPE        CLUSTER-IP      EXTERNAL-IP      PORT(S)          AGE
service/alertmanager-operated ClusterIP   None           <none>           9093/TCP, 9094/TCP, 9094/UDP 4m50s
service/prometheus-grafana LoadBalancer  172.28.168.21 a345375fb3c34543d490e55622e7afca-89423726.us-east-2.elb.amazonaws.com 80:3249/TCP 4m53s
service/prometheus-kube-prometheus-alertmanager ClusterIP   172.28.105.191 <none>           9093/TCP, 8880/TCP 4m53s
service/prometheus-kube-prometheus-operator ClusterIP   172.28.18.259 <none>           441:441/TCP 4m53s
service/prometheus-kube-prometheus-prometheus LoadBalancer 172.28.105.37 90d7eeb944f274e5d8289dd7489e9175-1643275437.us-east-2.elb.amazonaws.com 9090:31872/TCP, 8080:32243/TCP 4m53s
service/prometheus-kube-state-metrics ClusterIP   172.28.131.125 <none>           8080/TCP 4m53s
service/prometheus-operated ClusterIP   None           <none>           9099/TCP 4m49s
service/prometheus-prometheus-node-exporter ClusterIP   172.28.199.114 <none>           9100/TCP 4m53s

NAME          DESIRED  CURRENT  READY   UP-TO-DATE   AVAILABLE   NODE SELECTOR   AGE
daemonset.apps/prometheus-prometheus-node-exporter 2       2       2       2           2           kubernetes.io/os:linux 4m53s

NAME          READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/prometheus-grafana 1/1     1           1           4m53s
deployment.apps/prometheus-kube-prometheus-operator 1/1     1           1           4m53s
deployment.apps/prometheus-kube-state-metrics 1/1     1           1           4m53s

NAME          DESIRED  CURRENT  READY   AGE
replicaset.apps/prometheus-grafana-6cc4498afc 1       1       1       4m53s
replicaset.apps/prometheus-kube-prometheus-operator-76647ff58db 1       1       1       4m53s
replicaset.apps/prometheus-kube-state-metrics-5b787f97eb 1       1       1       4m53s

NAME          READY   AGE
statefulset.apps/alertmanager-prometheus-kube-prometheus-alertmanager 1/1     4m50s
statefulset.apps/prometheus-prometheus-kube-prometheus-prometheus 1/1     4m49s
ubuntu@ip-10-16-10-26:~/EKS-ArgoCD-AWS-LB-Controller-Terraform/eks$
```

Check whether the AWS Load Balancer controller pods are running in a aws-loadbalancer-controller namespace or not

> kubectl get all -n aws-loadbalancer-controller

```
Session ID: Adorable-Aman-ed2c2whhbgmuvtb7hjz30l Instance ID: i-0d8a3bc509812b65b Terminate
ubuntu@ip-10-16-10-26:~/EKS-ArgoCD-AWS-LB-Controller-Terraform/eks$ kubectl get all -n aws-loadbalancer-controller
NAME                                     READY   STATUS    RESTARTS   AGE
pod/aws-load-balancer-controller-7ch85ff90d7-5cvdp 1/1    Running   0          9m26s
pod/aws-load-balancer-controller-7ch85ff90d7-9g5bf 1/1    Running   0          9m26s

NAME          TYPE        CLUSTER-IP      EXTERNAL-IP      PORT(S)          AGE
service/aws-load-balancer-webhook-service ClusterIP   172.28.224.219 <none>           443/TCP 9m26s

NAME          READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/aws-load-balancer-controller 2/2     2           2           9m26s

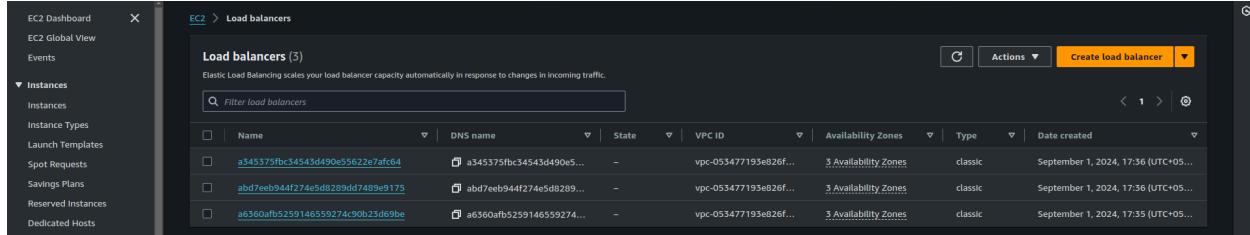
NAME          DESIRED  CURRENT  READY   AGE
replicaset.apps/aws-load-balancer-controller-7ch85ff90d7 2       2       2       9m26s
ubuntu@ip-10-16-10-26:~/EKS-ArgoCD-AWS-LB-Controller-Terraform/eks$
```

So, we have configured the AWS Load Balancer Controller, ArgoCD, Prometheus, and Grafana through Terraform.

According to the terraform script, we have updated the service type for ArgoCD, Prometheus, and Grafana from ClusterIP to LoadBalacer.

Hence, navigate to Load Balancer on your AWS account.

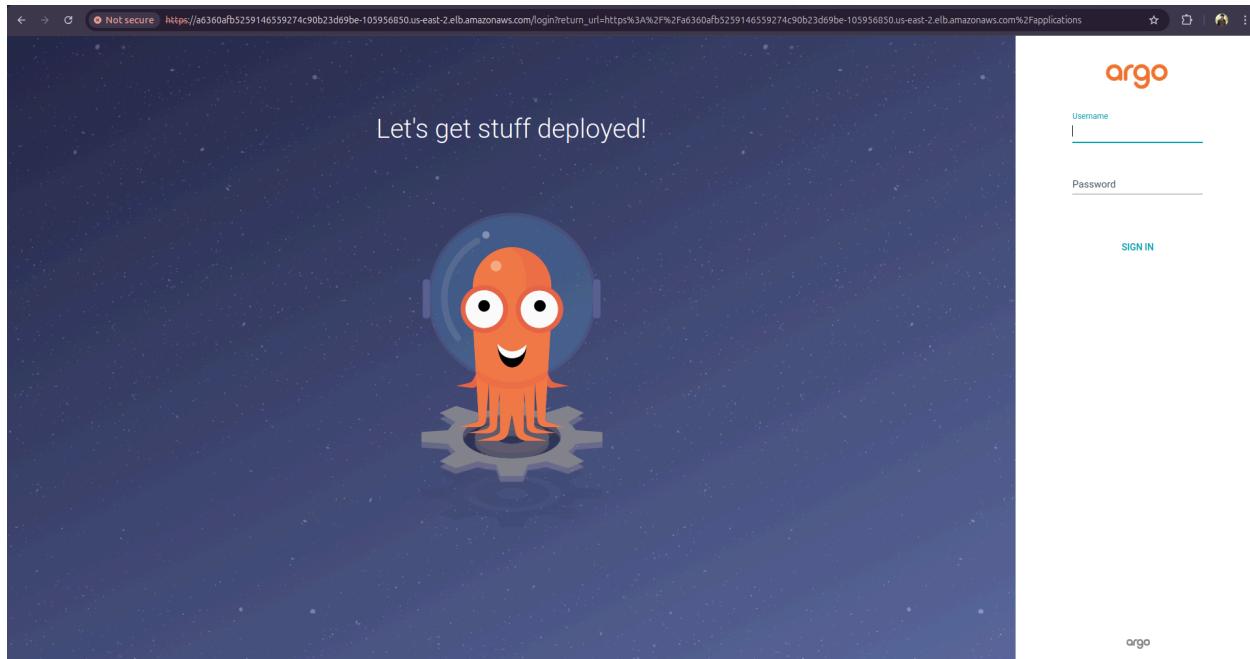
You can check which LB is for which tool through the kubectl command that we ran in the above steps.



The screenshot shows the AWS EC2 Load Balancers page. On the left, there's a sidebar with options like EC2 Dashboard, EC2 Global View, Events, Instances, Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, and Dedicated Hosts. The main area is titled 'Load balancers (3)' and contains a table with columns: Name, DNS name, State, VPC ID, Availability Zones, Type, and Date created. The table lists three load balancers:

Name	DNS name	State	VPC ID	Availability Zones	Type	Date created
a345375fb34543d490e5622e7afc64	a345375fb34543d490e5...	-	vpc-053477193e826f...	3 Availability Zones	classic	September 1, 2024, 17:36 (UTC+05...
abd7eeb944f274e5d8289dd7489e9175	abd7eeb944f274e5d8289...	-	vpc-053477193e826f...	3 Availability Zones	classic	September 1, 2024, 17:36 (UTC+05...
a6360afb5259146559274c90b23d69be	a6360afb5259146559274...	-	vpc-053477193e826f...	3 Availability Zones	classic	September 1, 2024, 17:35 (UTC+05...

So, let's try to access argoCD first. Copy the DNS and paste it into your favorite browser



The screenshot shows a web browser window with the URL https://a6360afb5259146559274c90b23d69be-105956850.us-east-2.elb.amazonaws.com/login?return_url=https%3A%2F%2Fa6360afb5259146559274c90b23d69be-105956850.us-east-2.elb.amazonaws.com%2Fapplications. The page has a dark background with a central orange cartoon character of an octopus wearing headphones. The text 'Let's get stuff deployed!' is displayed above the character. To the right, there's a sign-in form with fields for 'Username' and 'Password', and a 'SIGN IN' button. The Argo logo is visible at the top right.

The username is admin but it needs the password which we don't know. So, we need to run a command on our ec2 server to get the password for the login

```
kubectl -n argocd get secret argocd-initial-admin-secret -o jsonpath='{.data.password}' | base64 -d
```

```

Session ID: Adorable-Aman-ed1jz7uhjbllgmvutb7hzf3bl
Instance ID: i-0d8a3bc509812b65b
Terminate

ubuntu@ip-10-16-10-26:~/EKS-ArgoCD-AWS-LB-Controller-Terraform/eks$ kubectl -n argocd get secret argocd-initial-admin-secret -o jsonpath='{.data.password}' | base64 -d
F0fbFBxsEJHjwHMubuntu@ip-10-16-10-26:~/EKS-ArgoCD-AWS-LB-Controller-Terraform/eks$ 

```

It is accessible

The screenshot shows the Argo CD application management interface. On the left, there's a sidebar with icons for Applications, Settings, User Info, and Documentation. The main area has tabs for '+ NEW APP', 'SYNC APPS', and 'REFRESH APPS'. A search bar says 'Search applications...'. Below it is a large circular icon with three stacked rectangles. The text 'No applications available to you just yet' is centered, followed by 'Create new application to start managing resources in your cluster' and a 'CREATE APPLICATION' button.

Now, let's try to access Prometheus. Copy the DNS and paste it into your favorite browser with port 9090.

It is accessible

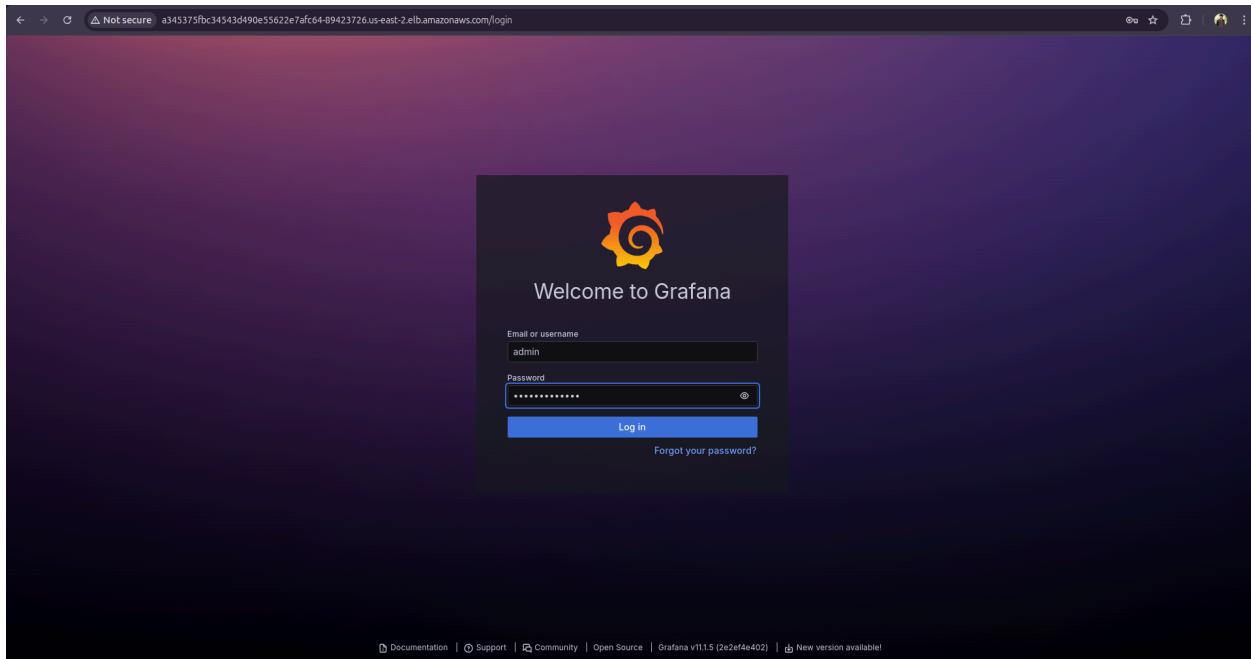
The screenshot shows the Prometheus UI with three tables of targets:

- Targets**: Shows one target for 'serviceMonitor/prometheus/prometheus-kube-prometheus-alertmanager/0' (1/1 up). Endpoint: http://10.16.149.175:9093/metrics. State: UP. Labels include container: 'alertmanager', endpoint: 'http-metrics', instance: '10.16.149.175:9093', job: 'prometheus-kube-prometheus-alertmanager', namespace: 'prometheus', pod: 'alertmanager-prometheus-kube-prometheus-alertmanager-0', and service: 'prometheus-kube-prometheus-alertmanager'.
- serviceMonitor/prometheus/prometheus-kube-prometheus-alertmanager/1 (1/1 up)**: Shows one target for 'serviceMonitor/prometheus/prometheus-kube-prometheus-alertmanager/1'. Endpoint: http://10.16.149.175:8080/metrics. State: UP. Labels include container: 'config-reloader', endpoint: 'reload-endpoint', instance: '10.16.149.175:8080', job: 'prometheus-kube-prometheus-alertmanager', namespace: 'prometheus', pod: 'alertmanager-prometheus-kube-prometheus-alertmanager-0', and service: 'prometheus-kube-prometheus-alertmanager'.
- serviceMonitor/prometheus/prometheus-kube-prometheus-apiserver/0 (2/2 up)**: Shows two targets for 'serviceMonitor/prometheus/prometheus-kube-prometheus-apiserver/0'. Endpoint: https://10.16.168.159:metrics. State: UP. Labels include endpoint: 'https', instance: '10.16.168.159:443', job: 'apiserver', namespace: 'default', and service: 'kube-apiserver'. Another target is listed below it with similar details.

In the end, let's try to access our Grafana dashboard. Copy the DNS and paste it into your favorite browser

It is accessible.

The username is admin and the password is prom-operator for the Grafana login



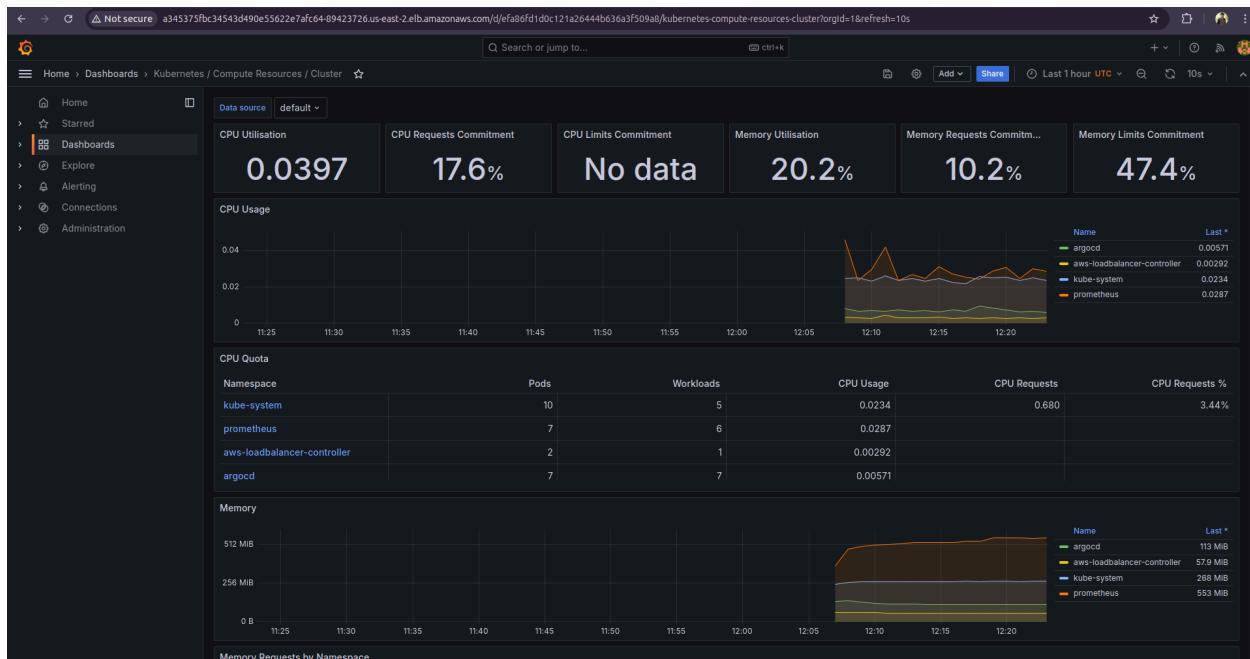
After login, the Grafana dashboard

A screenshot of the Grafana dashboard after logging in. The URL in the address bar is 'a345375fbc34543d490e55622e7afc64-89423726.us-east-2.elb.amazonaws.com/?orgId=1'. The dashboard has a dark theme. On the left is a sidebar with a 'Home' section containing links for 'Starred', 'Dashboards', 'Explore', 'Alerting', 'Connections', and 'Administration'. Below this is a 'Dashboards' section with 'Starred dashboards' and 'Recently viewed dashboards'. The main content area is titled 'Welcome to Grafana'. It features three cards: 'Basic' (with a sub-section 'TUTORIAL DATA SOURCE AND DASHBOARDS Grafana fundamentals'), 'COMPLETE Add your first data source' (with a link 'Learn how in the docs'), and 'COMPLETE Create your first dashboard' (with a link 'Learn how in the docs'). To the right of these cards is a 'Need help?' section with links for 'Documentation', 'Tutorials', 'Community', and 'Public Slack'. Below this is a 'Latest from the blog' section. The first post is 'Meet us at PromCon' (published Aug 30) which discusses the Prometheus 3.0 release. The second post is 'Visualize CockroachDB in Grafana: Introducing the CockroachDB Enterprise data source' (published Aug 29) which announces the addition of CockroachDB as a data source. The browser's standard navigation and search bars are visible at the top.

There are a lot of dashboards already imported. You can click any of them to get insight about the EKS Cluster resources accordingly.

The screenshot shows the Grafana interface with a dark theme. On the left, there is a sidebar with navigation links: Home, Starred, Dashboards (which is selected and highlighted in orange), Explore, Alerting, Connections, and Administration. The main area is titled "Dashboards" and contains a search bar and a "Filter by tag" dropdown. Below these are several dashboard cards, each with a thumbnail icon and a title. To the right of each card is a "Tags" section containing one or more colored boxes with labels like "alertmanager-mixin", "coredns", "dns", "etcd-mixin", "kubernetes-mixin", etc. A "New" button is located in the top right corner of the main dashboard list area.

Here is one of them



So, we have completed our demonstration for today.
Hope you learn something new today.
To prevent the cost of cloud. Don't forget to destroy all the resources.

To do that, we need to run EKS Cluster first.

```
> terraform destroy -auto-approve -var-file=../variables.tfvars
```

```
Session ID: Adorable-Aman-ed3c7uwhhiegmo.tb7uz50b  
Instance ID: i-0d8a3bc509812b65b  
Terminate  
  
module.eks.aws_eks_cluster.eks[0]: Still destroying... [id=dev-medium-eks-cluster, 40s elapsed]  
module.eks.aws_eks_cluster.eks[0]: Still destroying... [id=dev-medium-eks-cluster, 50s elapsed]  
module.eks.aws_eks_cluster.eks[0]: Still destroying... [id=dev-medium-eks-cluster, 60s elapsed]  
module.eks.aws_eks_cluster.eks[0]: Still destroying... [id=dev-medium-eks-cluster, 1m10s elapsed]  
module.eks.aws_eks_cluster.eks[0]: Still destroying... [id=dev-medium-eks-cluster, 1m20s elapsed]  
module.eks.aws_eks_cluster.eks[0]: Still destroying... [id=dev-medium-eks-cluster, 1m30s elapsed]  
module.eks.aws_eks_cluster.eks[0]: Still destroying... [id=dev-medium-eks-cluster, 1m40s elapsed]  
module.eks.aws_eks_cluster.eks[0]: Still destroying... [id=dev-medium-eks-cluster, 1m50s elapsed]  
module.eks.aws_eks_cluster.eks[0]: Still destroying... [id=dev-medium-eks-cluster, 2m0s elapsed]  
module.eks.aws_eks_cluster.eks[0]: Still destroying... [id=dev-medium-eks-cluster, 2m10s elapsed]  
module.eks.aws_eks_cluster.eks[0]: Still destroying... [id=dev-medium-eks-cluster, 2m20s elapsed]  
module.eks.aws_eks_cluster.eks[0]: Still destroying... [id=dev-medium-eks-cluster, 2m30s elapsed]  
module.eks.aws_eks_cluster.eks[0]: Still destroying... [id=dev-medium-eks-cluster, 2m40s elapsed]  
module.eks.aws_eks_cluster.eks[0]: Destruction complete after 2m50s  
module.eks.aws_iam_role.eks-cluster-role[0]: Destroying... [id=dev-medium-eks-cluster-role-4949]  
module.eks.aws_iam_role.eks-cluster-role[0]: Destruction complete after 0s  
module.eks.aws_random_integer.random_suffix: Destroying... [id=eks-random-suffix]  
module.eks.random_integer.random_suffix: Destruction complete after 0s  
  
Warning: Value for undeclared variable  
The root module does not declare a variable named "ec2-iam-instance-profile" but a value was found in file "../variables.tfvars". If you meant to use this value, add a "variable" block to the configuration.  
To silence these warnings, use TF_VAR... environment variables to provide certain "global" settings to all configurations in your organization. To reduce the verbosity of these warnings, use the -compact-warnings option.  
  
Warning: Value for undeclared variable  
The root module does not declare a variable named "ec2-iam-role-policy" but a value was found in file "../variables.tfvars". If you meant to use this value, add a "variable" block to the configuration.  
To silence these warnings, use TF_VAR... environment variables to provide certain "global" settings to all configurations in your organization. To reduce the verbosity of these warnings, use the -compact-warnings option.  
  
Warning: Values for undeclared variables  
In addition to the other similar warnings shown, 3 other variable(s) defined without being declared.  
  
Warning: Helm uninstall returned an information message  
These resources were kept due to the resource policy:  
[CustomResourceDefinition] applications.argoproj.io  
[CustomResourceDefinition] applicationsets.argoproj.io  
[CustomResourceDefinition] appprojects.argoproj.io  
  
Destroy complete! Resources: 25 destroyed.  
ubuntu@ip-10-16-10-20:~/EKS-ArgoCD-AWS-LB-Controller-Terraform/eks$
```

Now, destroy the vpc and an ec2 server. To do it, run the below command on your local server

```
> terraform destroy -auto-approve -var-file=../variables.tfvars
```

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS bash-vpc.ec2 + v ☰ ... ▾ x  
module.vpc-ec2.aws_instance.ec2: Destruction complete after 54s  
module.vpc-ec2.aws_iam_instance_profile.ec2-instance-profile: Destroying... [id=ec2-ssm-instance-profile]  
module.vpc-ec2.aws_security_group.ec2-sg: Destroying... [id=sg-071f08f4b1928202]  
module.vpc-ec2.aws_nat_gateway.ngw: Still destroying... [id=nat-0e920b64e401eb765, 50s elapsed]  
module.vpc-ec2.aws_internet_gateway.igw: Still destroying... [id=igw-081e78f5db8daa64, 50s elapsed]  
module.vpc-ec2.aws_iam_instance_profile.ec2-instance-profile: Destruction complete after 1s  
module.vpc-ec2.aws_iam_role.role: Destroying... [id=ec2-ssm-role]  
module.vpc-ec2.aws_security_group.ec2-sg: Destruction complete after 2s  
module.vpc-ec2.aws_iam_role.role: Destruction complete after 2s  
module.vpc-ec2.aws_nat_gateway.ngw: Still destroying... [id=nat-0e920b64e401eb765, 1m0s elapsed]  
module.vpc-ec2.aws_internet_gateway.igw: Still destroying... [id=igw-081e78f5db8daa64, 1m0s elapsed]  
module.vpc-ec2.aws_internet_gateway.igw: Destruction complete after 1m4s  
module.vpc-ec2.aws_nat_gateway.ngw: Destruction complete after 1m10s  
module.vpc-ec2.aws_endpoints.endpoint[0]: Destroying... [id=endpoints-05edaa00bd-0b-fcc9]  
module.vpc-ec2.aws_subnet_public.subnet[0]: Destroying... [id=subnet-00cb71ddd8cc3bc]  
module.vpc-ec2.aws_subnet_public.subnet[1]: Destroying... [id=subnet-0b600ba393c3a40f]  
module.vpc-ec2.aws_subnet_public.subnet[2]: Destroying... [id=subnet-0529fb28936b2ded1c]  
module.vpc-ec2.aws_subnet_public.subnet[0]: Destruction complete after 1s  
module.vpc-ec2.aws_eip.ngw-eip: Destruction complete after 2s  
module.vpc-ec2.aws_subnet_public.subnet[1]: Destruction complete after 2s  
module.vpc-ec2.aws_subnet_public.subnet[2]: Destruction complete after 3s  
module.vpc-ec2.aws_vpc.vpc: Destroying... [id=vpc-053477193e826f6c2]  
module.vpc-ec2.aws_vpc.vpc: Destruction complete after 1s  
  
Warning: Value for undeclared variable  
The root module does not declare a variable named "endpoint-public-access" but a value was found in file "../variables.tfvars". If you meant to use this value, add a "variable" block to the configuration.  
To silence these warnings, use TF_VAR... environment variables to provide certain "global" settings to all configurations in your organization. To reduce the verbosity of these warnings, use the -compact-warnings option.  
  
Warning: Value for undeclared variable  
The root module does not declare a variable named "desired_capacity_on_demand" but a value was found in file "../variables.tfvars". If you meant to use this value, add a "variable" block to the configuration.  
To silence these warnings, use TF_VAR... environment variables to provide certain "global" settings to all configurations in your organization. To reduce the verbosity of these warnings, use the -compact-warnings option.  
  
Warning: Values for undeclared variables  
In addition to the other similar warnings shown, 11 other variable(s) defined without being declared.  
Releasing state lock. This may take a few moments...  
Destroy complete! Resources: 24 destroyed.
```

I would recommend you to read the eks terraform files to get a better understanding of the resources. As in the repo, everything is straightforward if you worked on Terraform already.

Conclusion

In this comprehensive guide, we've explored how to deploy a private EKS cluster on AWS and configure essential Kubernetes tools such as ArgoCD, Prometheus, and Grafana using Terraform. By following these steps, you can efficiently manage your infrastructure and ensure that your applications are running smoothly in a secure, scalable environment. Remember to clean up your resources after the demonstration to avoid unnecessary costs. Continuous learning and hands-on practice are key to mastering these DevOps practices, so keep experimenting with different configurations and tools to enhance your skills.



Follow for more

Join Discord Community: <https://lnkd.in/dsEdxpst>

Follow on GitHub: <https://github.com/AmanPathak-DevOps/>

Follow on LinkedIn- <https://www.linkedin.com/in/aman-devops/>