

## Mr Cloud Book

Home Blog DevSecOps Contact About Me Testimonials

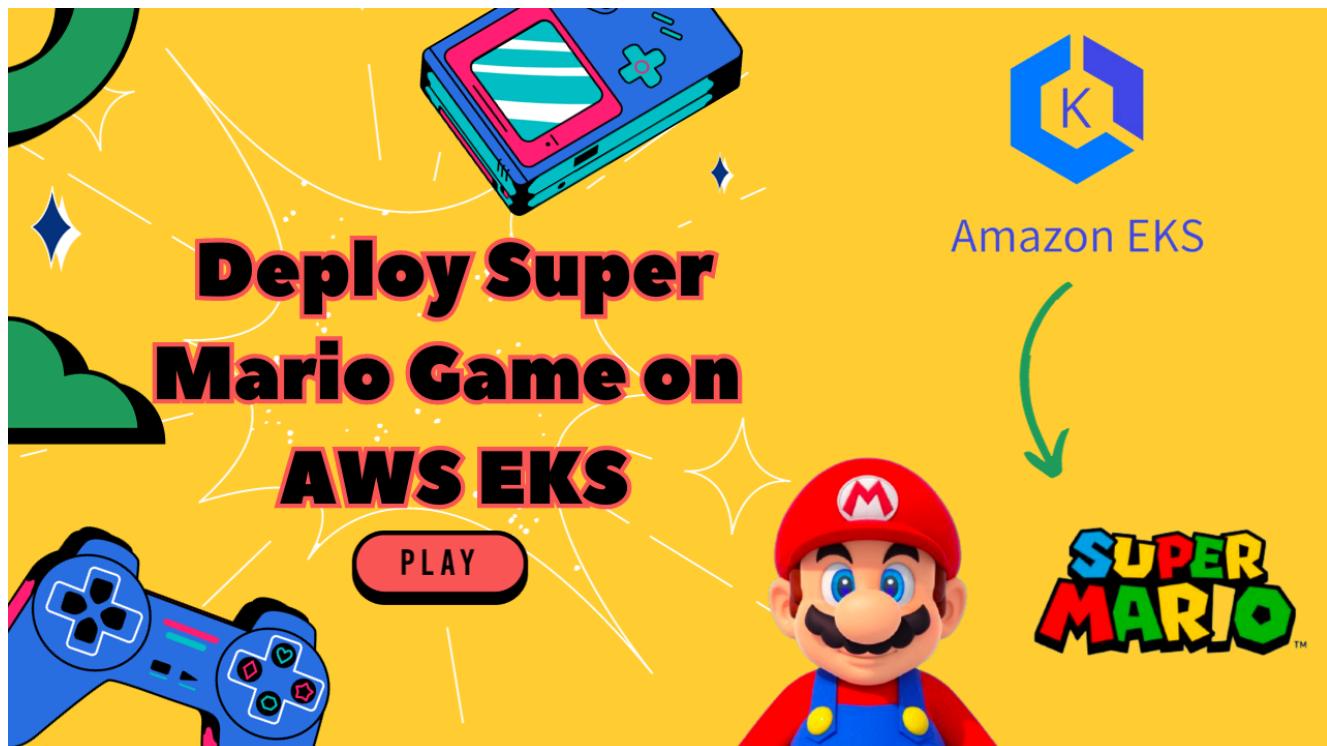
Search Blogs

DevOps

# Deploying Super Mario on Kubernetes



mrcloobook.com · 8 January 2024



Hey folks, remember the thrill of 90's gaming? Let's step back in time and relive those exciting moments! With the game deployed on Kubernetes, it's time to dive into the nostalgic world of Mario. Grab your controllers, it's game time!

Super Mario is a classic game loved by many. In this guide, we'll explore how to deploy a Super Mario game on Amazon's Elastic Kubernetes Service (EKS). Utilizing

Kubernetes, we can orchestrate the game's deployment on AWS EKS, allowing for scalability, reliability, and easy management.

## Contents [ [hide](#) ]

Prerequisites:

### LET'S DEPLOY

[STEP 1: Launch Ubuntu instance](#)

[STEP 2: Create IAM role](#)

[STEP 3: Cluster provision](#)

[Destruction :](#)

## Prerequisites:

1. An Ubuntu Instance
2. IAM role
3. Terraform should be installed on instance
4. AWS CLI and KUBECTL on Instance

## LET'S DEPLOY

### STEP 1: Launch Ubuntu instance

1. **Sign in to AWS Console:** Log in to your AWS Management Console.
2. **Navigate to EC2 Dashboard:** Go to the EC2 Dashboard by selecting “Services” in the top menu and then choosing “EC2” under the Compute section.
3. **Launch Instance:** Click on the “Launch Instance” button to start the instance creation process.
4. **Choose an Amazon Machine Image (AMI):** Select an appropriate AMI for your instance. For example, you can choose Ubuntu image.
5. **Choose an Instance Type:** In the “Choose Instance Type” step, select `t2.micro` as your instance type. Proceed by clicking “Next: Configure Instance Details.”
6. **Configure Instance Details:**

- For “Number of Instances,” set it to 1 (unless you need multiple instances).
- Configure additional settings like network, subnets, IAM role, etc., if necessary.
- For “Storage,” click “Add New Volume” and set the size to 8GB (or modify the existing storage to 8GB).
- Click “Next: Add Tags” when you’re done.

**7. Add Tags (Optional):** Add any desired tags to your instance. This step is optional, but it helps in organizing instances.

#### **8. Configure Security Group:**

- Choose an existing security group or create a new one.
- Ensure the security group has the necessary inbound/outbound rules to allow access as required.

**9. Review and Launch:** Review the configuration details. Ensure everything is set as desired.

#### **10. Select Key Pair:**

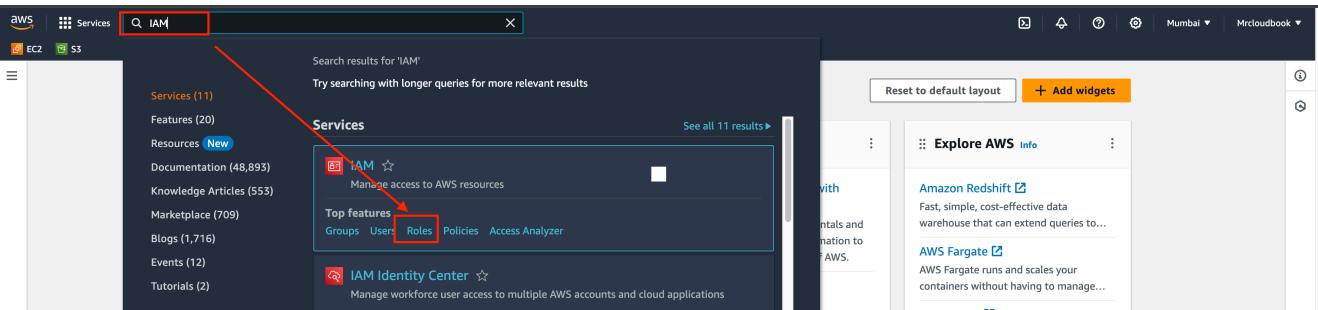
- Select “Choose an existing key pair” and choose the key pair from the dropdown.
- Acknowledge that you have access to the selected private key file.
- Click “Launch Instances” to create the instance.

**11. Access the EC2 Instance:** Once the instance is launched, you can access it using the key pair and the instance’s public IP or DNS.

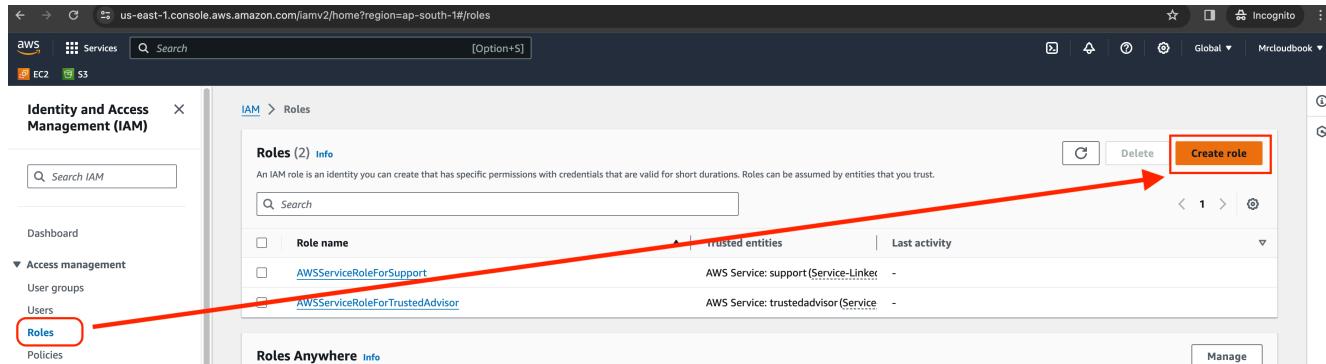
Ensure you have necessary permissions and follow best practices while configuring security groups and key pairs to maintain security for your EC2 instance.

## **STEP 2: Create IAM role**

Search for IAM in the search bar of AWS and click on roles.

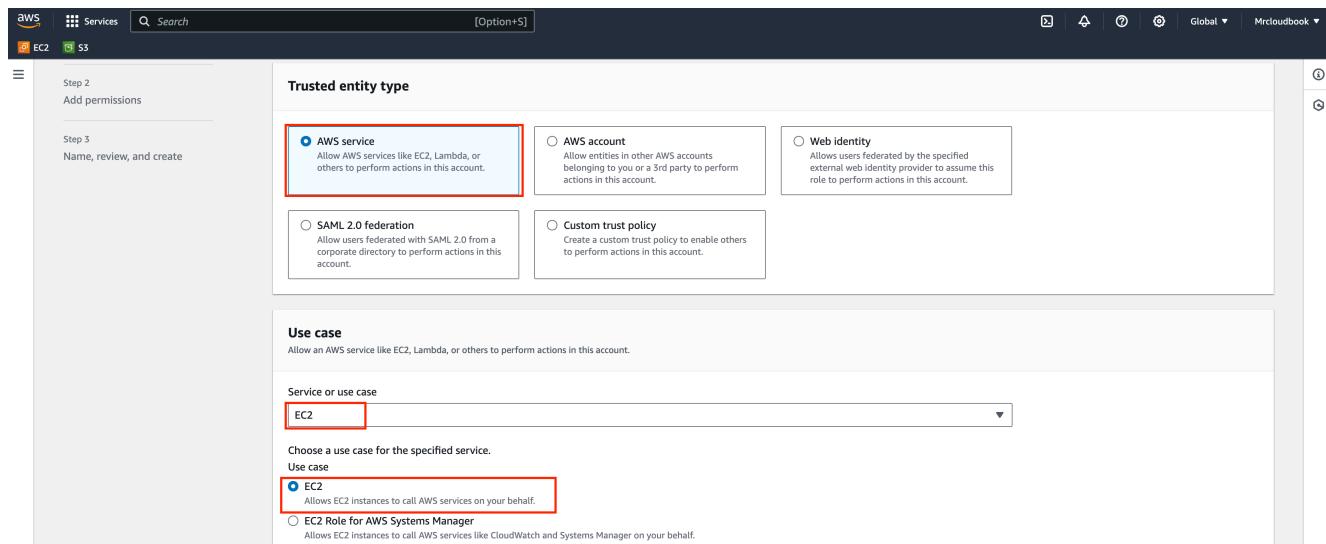


Click on Create Role



Select entity type as AWS service

Use case as EC2 and click on Next.



For permission policy select Administrator Access (Just for learning purpose), click Next.

The screenshot shows the 'Add permissions' step in the IAM 'Create role' wizard. A red arrow points from the previous step's 'Add permissions' link to this screen. The 'Permissions policies' section lists two policies: 'AdministratorAccess' (selected) and 'AdministratorAccess-Amplify'. The 'AdministratorAccess' policy is described as 'AWS managed - job function' and 'Provides full access to AWS services an...'. The 'AdministratorAccess-Amplify' policy is described as 'AWS managed' and 'Grants account administrative permis...'. The 'Policy name' field contains 'AdministratorAccess'.

Provide a Name for Role and click on Create role.

The screenshot shows the 'Role details' step in the IAM 'Create role' wizard. The 'Role name' field is filled with 'MARIO'. The 'Description' field contains 'Allows EC2 instances to call AWS services on your behalf.' The 'Step 1: Select trusted entities' section shows a JSON trust policy:

```

1  "Version": "2012-10-17",
2   "Statement": [
3     {
4       "Effect": "Allow",
5       "Action": [
6         "sts:AssumeRole"
7       ],
8       "Principal": {
9         "Service": [
10           "ec2.amazonaws.com"
11         ]
12       }
13     }
14   ]
15 ]
16 ]

```

Role is created.

The screenshot shows the 'Roles' list in the IAM console. A green banner at the top says '(+) Role MARIO created.' The 'MARIO' role is listed in the table, highlighted with a red box. The table columns are 'Role name', 'Trusted entities', and 'Last activity'. The 'Role name' column shows 'MARIO', 'AWSRoleForSupport', 'AWSRoleForTrustedAdvisor', and 'MARIO'. The 'Trusted entities' column shows 'AWS Service: support (Service-Linker)', 'AWS Service: trustedadvisor (Service)', and 'AWS Service: ec2'.

Now Attach this role to Ec2 instance that we created earlier, so we can provision cluster from that instance.

Go to EC2 Dashboard and select the instance.

Click on Actions -> Security -> Modify IAM role.

The screenshot shows the AWS EC2 Instances page. A single instance named 'Test' is listed as 'Running'. In the 'Actions' dropdown menu, the 'Modify IAM role' option is highlighted with a red box.

Select the Role that created earlier and click on Update IAM role.

The screenshot shows the 'Modify IAM role' dialog box. The 'Instance ID' is set to 'i-0c92d797a5813a128 (Test)'. The 'IAM role' dropdown is set to 'MARIO', which is highlighted with a red box. The 'Update IAM role' button at the bottom is also highlighted with a red box.

Connect the instance to Mobaxtreme or Putty

## STEP 3: Cluster provision

Now clone this Repo.

The screenshot shows a terminal window with the command 'git clone https://github.com/Aj7Ay/k8s-mario.git' entered. The entire command line is highlighted with a red box.

```
root@ip-172-31-41-147:/home/ubuntu#
root@ip-172-31-41-147:/home/ubuntu# git clone https://github.com/Aj7Ay/k8s-mario.git
Cloning into 'k8s-mario'...
remote: Enumerating objects: 32, done.
remote: Counting objects: 100% (32/32), done.
remote: Compressing objects: 100% (30/30), done.
remote: Total 32 (delta 8), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (32/32), 8.56 KiB / 1.43 MiB/s, done.
Resolving deltas: 100% (8/8), done.
root@ip-172-31-41-147:/home/ubuntu# ls
k8s-mario
root@ip-172-31-41-147:/home/ubuntu#
```

change directory



```
cd k8s-mario
```



Provide the executable permission to script.sh file, and run it.



```
sudo chmod +x script.sh  
./script.sh
```



```
root@ip-172-31-41-147:/home/ubuntu# ls  
k8s-mario  
root@ip-172-31-41-147:/home/ubuntu# cd k8s-mario/  
root@ip-172-31-41-147:/home/ubuntu/k8s-mario# ls  
EKS-TF deployment.yaml script.sh service.yaml  
root@ip-172-31-41-147:/home/ubuntu/k8s-mario# chmod +x script.sh  
root@ip-172-31-41-147:/home/ubuntu/k8s-mario# ls  
EKS-TF deployment.yaml script.sh service.yaml  
root@ip-172-31-41-147:/home/ubuntu/k8s-mario# ./script.sh
```

This script will install Terraform, AWS cli, Kubectl, Docker.

Check versions



```
docker --version  
aws --version  
kubectl version --client  
terraform --version
```



```
root@ip-172-31-41-147:/home/ubuntu/k8s-mario# 
root@ip-172-31-41-147:/home/ubuntu/k8s-mario# aws --version
aws-cli/2.14.5 Python/3.11.6 Linux/6.2.0-1012-aws exe/x86_64.ubuntu.22 prompt/off
root@ip-172-31-41-147:/home/ubuntu/k8s-mario#
root@ip-172-31-41-147:/home/ubuntu/k8s-mario# terraform --version
Command 'terraform' not found, did you mean:
  command 'terraform' from snap terraform (1.6.5)
See 'snap info <snapname>' for additional versions.
root@ip-172-31-41-147:/home/ubuntu/k8s-mario# terraform --version
Terraform v1.6.5
on linux_amd64
root@ip-172-31-41-147:/home/ubuntu/k8s-mario#
root@ip-172-31-41-147:/home/ubuntu/k8s-mario# kubectl version --client
Client Version: v1.28.4
Kustomize Version: v5.0.4-0.20230601165947-6ce0bf390ce3
root@ip-172-31-41-147:/home/ubuntu/k8s-mario#
```

Now change directory into the EKS-TF

Run Terraform init

NOTE: Don't forgot to change the s3 bucket name in the backend.tf file

```
cd EKS-TF
terraform init
```

```
root@ip-172-31-41-147:/home/ubuntu/k8s-mario#  
root@ip-172-31-41-147:/home/ubuntu/k8s-mario# ls  
EKS-TF aws awscliv2.zip deployment.yaml kubectl script.sh service.yaml  
root@ip-172-31-41-147:/home/ubuntu/k8s-mario# cd EKS-TF/  
root@ip-172-31-41-147:/home/ubuntu/k8s-mario/EKS-TF#  
root@ip-172-31-41-147:/home/ubuntu/k8s-mario/EKS-TF#  
root@ip-172-31-41-147:/home/ubuntu/k8s-mario/EKS-TF# ls  
backend.tf main.tf provider.tf  
root@ip-172-31-41-147:/home/ubuntu/k8s-mario/EKS-TF# terraform init
```

Initializing the backend...

Successfully configured the backend "s3"! Terraform will automatically use this backend unless the backend configuration changes.

Initializing provider plugins...

- Finding hashicorp/aws versions matching "~> 5.0"...
- Installing hashicorp/aws v5.29.0...
- Installed hashicorp/aws v5.29.0 (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.

```
root@ip-172-31-41-147:/home/ubuntu/k8s-mario/EKS-TF#
```

Now run `terraform validate` and `terraform plan`

terraform validate  
terraform plan



```
root@ip-172-31-41-147:/home/ubuntu/k8s-mario/EKS-TF#
root@ip-172-31-41-147:/home/ubuntu/k8s-mario/EKS-TF# terraform validate
Success! The configuration is valid.

root@ip-172-31-41-147:/home/ubuntu/k8s-mario/EKS-TF# terraform plan
data.aws_iam_policy_document.assume_role: Reading...
data.aws_vpc.default: Reading...
data.aws_iam_policy_document.assume_role: Read complete after 0s [id=3552664922]
data.aws_vpc.default: Read complete after 0s [id=vpc-0f6bdd74ced5c07c0]
data.aws_subnets.public: Reading...
data.aws_subnets.public: Read complete after 0s [id=ap-south-1]

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_cluster.example will be created
+ resource "aws_eks_cluster" "example" {
  + arn          = (known after apply)
  + certificate_authority = (known after apply)
  + cluster_id    = (known after apply)
  + created_at    = (known after apply)
  + endpoint      = (known after apply)
  + id           = (known after apply)
  + identity      = (known after apply)
  + name          = "EKS_CLOUD"
  + platform_version = (known after apply)
  + role_arn      = (known after apply)
  + status         = (known after apply)
  + tags_all      = (known after apply)
  + version        = (known after apply)
```

Now Run terraform apply to provision cluster.



**terraform apply --auto-approve**



```
root@ip-172-31-41-147:/home/ubuntu/k8s-mario/EKS-TF#
root@ip-172-31-41-147:/home/ubuntu/k8s-mario/EKS-TF# terraform apply --auto-approve
data.aws_vpc.default: Reading...
data.aws_iam_policy_document.assume_role: Reading...
data.aws_iam_policy_document.assume_role: Read complete after 0s [id=3552664922]
data.aws_vpc.default: Read complete after 0s [id=vpc-0f6bdd74ced5c07c0]
data.aws_subnets.public: Reading...
data.aws_subnets.public: Read complete after 0s [id=ap-south-1]

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_cluster.example will be created
+ resource "aws_eks_cluster" "example" {
  + arn          = (known after apply)
  + certificate_authority = (known after apply)
  + cluster_id    = (known after apply)
  + created_at    = (known after apply)
  + endpoint      = (known after apply)
  + id           = (known after apply)
  + identity      = (known after apply)
  + name          = "EKS_CLOUD"
  + platform_version = (known after apply)
  + role_arn      = (known after apply)
  + status         = (known after apply)
  + tags_all      = (known after apply)
  + version        = (known after apply)
```

Completed in 10mins

```

Plan: 8 to add, 0 to change, 0 to destroy.
aws_iam_role.example1: Creating...
aws_iam_role.example: Creating...
aws_iam_role.example: Creation complete after 1s [id=eks-cluster-cloud]
aws_iam_role_policy_attachment.example AMAZONEKSClusterPolicy: Creating...
aws_iam_role.example1: Creation complete after 1s [id=eks-node-group-cloud]
aws_iam_role_policy_attachment.example AMAZONEC2ContainerRegistryReadOnly: Creating...
aws_iam_role_policy_attachment.example AMAZONEKSWorkerNodePolicy: Creating...
aws_iam_role_policy_attachment.example AMAZONEKS_CNI_Policy: Creating...
aws_iam_role_policy_attachment.example AMAZONEKS_CNT_Policy: Creation complete after 1s [id=eks-cluster-cloud-20231205133150541700000001]
aws_eks_cluster.example: Creating...
aws_iam_role_policy_attachment.example AMAZONEC2ContainerRegistryReadOnly: Creation complete after 1s [id=eks-node-group-cloud-20231205133150682600000002]
aws_iam_role_policy_attachment.example AMAZONEKSWorkerNodePolicy: Creation complete after 1s [id=eks-node-group-cloud-20231205133150786300000003]
aws_iam_role_policy_attachment.example AMAZONEKS_CNT_Policy: Creation complete after 1s [id=eks-node-group-cloud-20231205133150926900000004]
aws_eks_cluster.example: Still creating... [10s elapsed]
aws_eks_cluster.example: Still creating... [20s elapsed]
aws_eks_cluster.example: Still creating... [30s elapsed]
aws_eks_cluster.example: Still creating... [40s elapsed]
aws_eks_cluster.example: Still creating... [50s elapsed]
aws_eks_cluster.example: Still creating... [1m0s elapsed]
aws_eks_cluster.example: Still creating... [1m10s elapsed]
aws_eks_cluster.example: Still creating... [1m20s elapsed]
aws_eks_cluster.example: Still creating... [1m30s elapsed]
aws_eks_cluster.example: Still creating... [1m40s elapsed]
aws_eks_cluster.example: Still creating... [1m50s elapsed]
aws_eks_cluster.example: Still creating... [2m0s elapsed]
aws_eks_cluster.example: Still creating... [2m10s elapsed]
aws_eks_cluster.example: Still creating... [2m20s elapsed]
aws_eks_cluster.example: Still creating... [2m30s elapsed]
aws_eks_cluster.example: Still creating... [2m40s elapsed]
aws_eks_cluster.example: Still creating... [2m50s elapsed]
aws_eks_cluster.example: Still creating... [3m0s elapsed]
aws_eks_cluster.example: Still creating... [3m10s elapsed]
aws_eks_cluster.example: Still creating... [3m20s elapsed]
aws_eks_cluster.example: Still creating... [3m30s elapsed]
aws_eks_cluster.example: Still creating... [3m40s elapsed]
aws_eks_cluster.example: Still creating... [3m50s elapsed]
aws_eks_cluster.example: Still creating... [4m0s elapsed]
aws_eks_cluster.example: Still creating... [4m10s elapsed]
aws_eks_cluster.example: Still creating... [4m20s elapsed]
aws_eks_cluster.example: Still creating... [4m30s elapsed]
aws_eks_cluster.example: Still creating... [4m40s elapsed]
aws_eks_cluster.example: Still creating... [4m50s elapsed]
aws_eks_cluster.example: Still creating... [5m0s elapsed]
aws_eks_cluster.example: Still creating... [5m10s elapsed]
aws_eks_cluster.example: Still creating... [5m20s elapsed]
aws_eks_cluster.example: Still creating... [5m30s elapsed]
aws_eks_cluster.example: Still creating... [5m40s elapsed]
aws_eks_cluster.example: Still creating... [5m50s elapsed]
aws_eks_cluster.example: Still creating... [6m0s elapsed]
aws_eks_cluster.example: Creation complete after 6m10s [id=EKS_CLOUD]
aws_eks_node_group.example: Creating...
aws_eks_node_group.example: Still creating... [10s elapsed]
aws_eks_node_group.example: Still creating... [20s elapsed]
aws_eks_node_group.example: Still creating... [30s elapsed]
aws_eks_node_group.example: Still creating... [40s elapsed]
aws_eks_node_group.example: Still creating... [50s elapsed]
aws_eks_node_group.example: Still creating... [1m0s elapsed]
aws_eks_node_group.example: Still creating... [1m10s elapsed]
aws_eks_node_group.example: Still creating... [1m20s elapsed]
aws_eks_node_group.example: Still creating... [1m30s elapsed]
aws_eks_node_group.example: Still creating... [1m40s elapsed]
aws_eks_node_group.example: Still creating... [1m50s elapsed]
aws_eks_node_group.example: Still creating... [2m0s elapsed]
aws_eks_node_group.example: Still creating... [2m10s elapsed]
aws_eks_node_group.example: Creation complete after 2m18s [id=EKS_CLOUD:Node-cloud]

Apply complete! Resources: 8 added, 0 changed, 0 destroyed.
root@ip-172-31-41-147:/home/ubuntu/k8s-mario/EKS-TF# 
```

## Update the Kubernetes configuration

Make sure change your desired region



```
aws eks update-kubeconfig --name EKS_CLOUD --region ap-south-1
```



```

root@ip-172-31-41-147:/home/ubuntu/k8s-mario/EKS-TF#
root@ip-172-31-41-147:/home/ubuntu/k8s-mario/EKS-TF# aws eks update-kubeconfig --name EKS_CLOUD --region ap-south-1
Added new context arn:aws:eks:ap-south-1:672618677785:cluster/EKS_CLOUD to /root/.kube/config
root@ip-172-31-41-147:/home/ubuntu/k8s-mario/EKS-TF# 
```

Now change directory back to k8s-mario



```
cd ..
```



Let's apply the deployment and service

## Deployment



```
kubectl apply -f deployment.yaml  
#to check the deployment  
kubectl get all
```



```
root@ip-172-31-41-147:/home/ubuntu/k8s-mario/EKS-TF#  
root@ip-172-31-41-147:/home/ubuntu/k8s-mario/EKS-TF#  
root@ip-172-31-41-147:/home/ubuntu/k8s-mario/EKS-TF# cd ..  
root@ip-172-31-41-147:/home/ubuntu/k8s-mario#  
root@ip-172-31-41-147:/home/ubuntu/k8s-mario# ls -l  
total 107168  
drwxr-xr-x 3 root root 4096 Dec 5 13:30 EKS-TF  
drwxr-xr-x 3 root docker 4096 Nov 30 22:22 aws  
-rw-r--r-- 1 root docker 59832742 Dec 5 13:28 awscliv2.zip  
-rw-r--r-- 1 root root 387 Dec 5 13:26 deployment.yaml  
-rw-r--r-- 1 root docker 49885184 Dec 5 13:28 kubectl  
-rwxr-xr-x 1 root root 933 Dec 5 13:28 script.sh  
-rw-r--r-- 1 root root 180 Dec 5 13:26 service.yaml  
root@ip-172-31-41-147:/home/ubuntu/k8s-mario#  
root@ip-172-31-41-147:/home/ubuntu/k8s-mario# kubectl apply -f deployment.yaml  
deployment.apps/mario-deployment created  
root@ip-172-31-41-147:/home/ubuntu/k8s-mario# kubectl get all  
NAME READY STATUS RESTARTS AGE  
pod/mario-deployment-78cbc65cb-9n547 1/1 Running 0 11s  
pod/mario-deployment-78cbc65cb-stxkr 1/1 Running 0 11s  
  
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE  
service/kubernetes ClusterIP 10.100.0.1 <none> 443/TCP 7m6s  
  
NAME READY UP-TO-DATE AVAILABLE AGE  
deployment.apps/mario-deployment 2/2 2 2 11s  
  
NAME DESIRED CURRENT READY AGE  
replicaset.apps/mario-deployment-78cbc65cb 2 2 2 11s  
root@ip-172-31-41-147:/home/ubuntu/k8s-mario#
```

Now let's apply the service

## Service



```
kubectl apply -f service.yaml  
kubectl get all
```



```
root@ip-172-31-41-147:/home/ubuntu/k8s-mario#  
root@ip-172-31-41-147:/home/ubuntu/k8s-mario# kubectl apply -f service.yaml  
service/mario-service created  
root@ip-172-31-41-147:/home/ubuntu/k8s-mario# kubectl get all  
NAME READY STATUS RESTARTS AGE  
pod/mario-deployment-78cbc65cb-9n547 1/1 Running 0 47s  
pod/mario-deployment-78cbc65cb-stxkr 1/1 Running 0 47s  
  
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE  
service/kubernetes ClusterIP 10.100.0.1 <none> 443/TCP 7m42s  
service/mario-service LoadBalancer 10.100.119.16 afe22fd27c11b42c0a7c486290725ad8-21137843.ap-south-1.elb.amazonaws.com 80:30746/TCP 14s  
  
NAME READY UP-TO-DATE AVAILABLE AGE  
deployment.apps/mario-deployment 2/2 2 2 47s  
  
NAME DESIRED CURRENT READY AGE  
replicaset.apps/mario-deployment-78cbc65cb 2 2 2 47s  
root@ip-172-31-41-147:/home/ubuntu/k8s-mario#
```

Now let's describe the service and copy the LoadBalancer Ingress



```
kubectl describe service mario-service
```



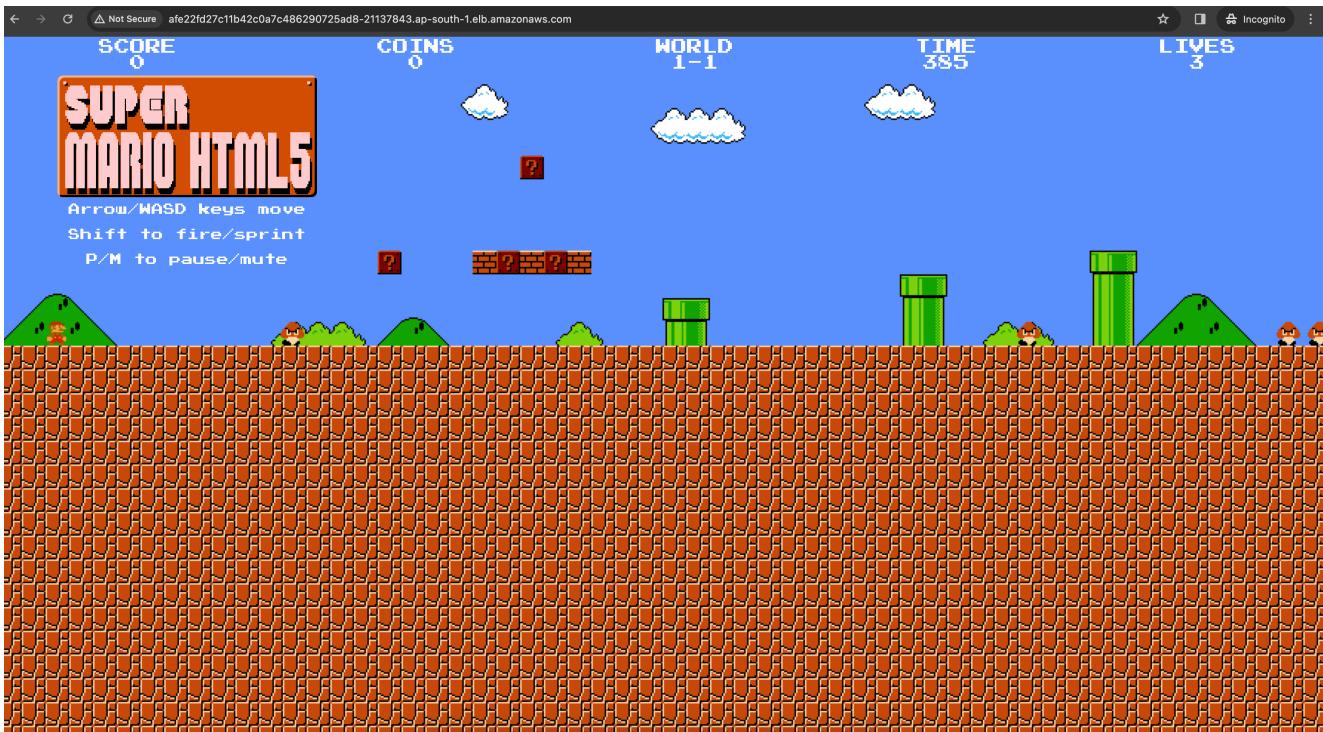
```

root@ip-172-31-41-147:/home/ubuntu/k8s-mario#
root@ip-172-31-41-147:/home/ubuntu/k8s-mario#
root@ip-172-31-41-147:/home/ubuntu/k8s-mario# kubectl describe service mario-service
Name: AJAY MR CLOUD BOOK      mario-service
Namespace:                      default
Labels:                          <none>
Annotations:                     <none>
Selector:                        app=mario
Type:                            LoadBalancer
IP Family Policy:               SingleStack
IP Families:                    IPv4
IP:                             10.100.119.16
IPs:                            10.100.119.16
LoadBalancer Ingress:           afe22fd27c11b42c0a7c486290725ad8-21137843.ap-south-1.elb.amazonaws.com
Port:                           <unset> 80/TCP
TargetPort:                      80/TCP
NodePort:                        <unset> 30746/TCP
Endpoints:                      172.31.8.149:80,172.31.9.206:80
Session Affinity:                None
External Traffic Policy:        Cluster
Events:
  Type   Reason     Age   From            Message
  ----  -----     --   --   --   -----
Normal  EnsuringLoadBalancer  86s  service-controller  Ensuring load balancer
Normal  EnsuredLoadBalancer  83s  service-controller  Ensured load balancer
root@ip-172-31-41-147:/home/ubuntu/k8s-mario# 

```

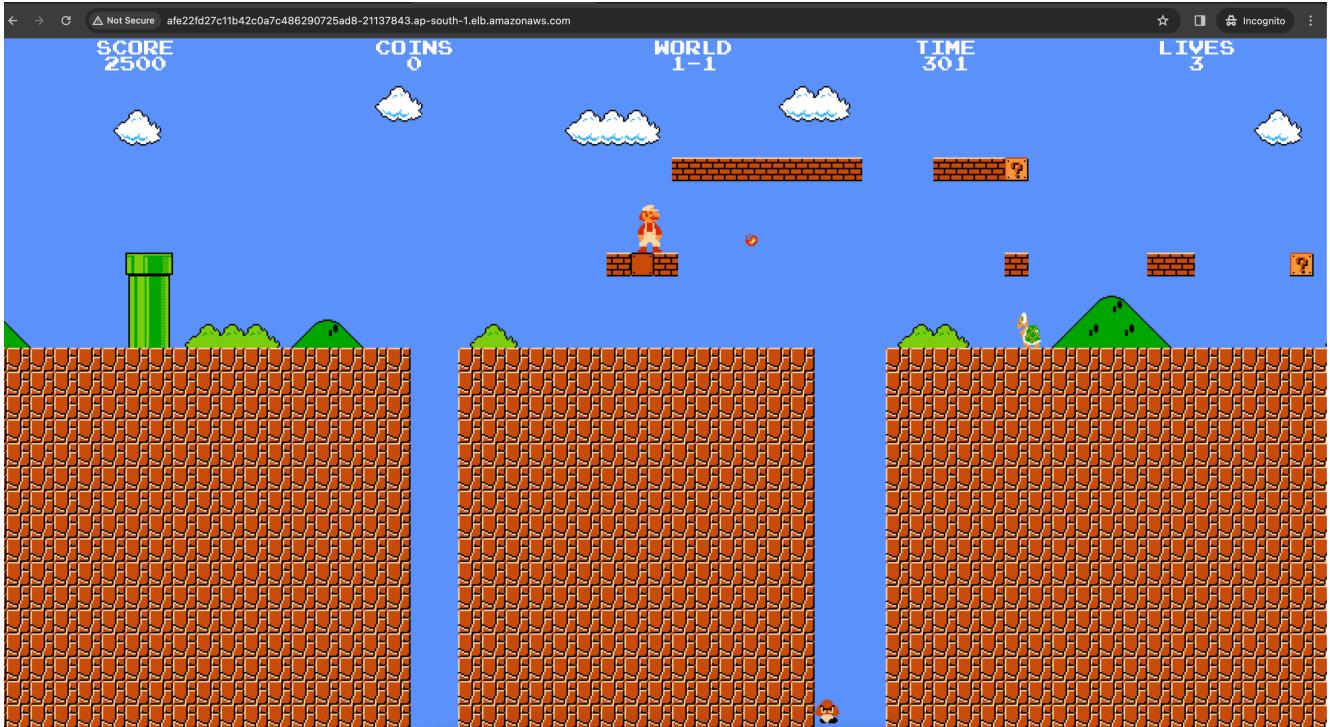
Paste the ingress link in a browser and you will see the Mario game.

Let's Go back to 1985 and play the game like children.



This is official image by MR CLOUD BOOK

You can check in Docker-hub as well sevenajay/mario:latest



## Destruction :

Let's remove the service and deployment first

```
kubectl get all  
kubectl delete service mario-service  
kubectl delete deployment mario-deployment
```

Let's Destroy the cluster

```
terraform destroy --auto-approve
```

```
root@ip-172-31-41-147:/home/ubuntu/k8s-mario/EKS-TF# terraform destroy --auto-approve
data.aws_iam_policy_document.assume_role: Reading...
aws_iam_role.example1: Refreshing state... [id=eks-node-group-cloud]
data.aws_vpc.default: Reading...
data.aws_iam_policy_document.assume_role: Read complete after 0s [id=3552664922]
aws_iam_role.example: Refreshing state... [id=eks-cluster-cloud]
data.aws_vpc.default: Read complete after 0s [id=vpc-0f6bdd74ced5c07c0]
data.aws_subnets.public: Reading...
data.aws_subnets.public: Read complete after 0s [id=ap-south-1]
aws_iam_role_policy_attachment.example AMAZON_EKS_CNI_Policy: Refreshing state... [id=eks-node-group-cloud-20231205133150926900000004]
aws_iam_role_policy_attachment.example AMAZON_EKSWorkerNodePolicy: Refreshing state... [id=eks-node-group-cloud-20231205133150786300000003]
aws_iam_role_policy_attachment.example AMAZON_ECSContainerRegistryReadOnly: Refreshing state... [id=eks-node-group-cloud-20231205133150682600000002]
aws_iam_role_policy_attachment.example AMAZON_EKSClusterPolicy: Refreshing state... [id=eks-cluster-cloud-20231205133150541700000001]
aws_eks_cluster.example: Refreshing state... [id=EKS_CLOUD]
aws_eks_node_group.example: Refreshing state... [id=EKS_CLOUD:Node-cloud]
```

After 10mins Resources that are provisioned will be removed.

```
aws_eks_node_group.example: Destruction complete after 9m22s
aws_iam_role_policy_attachment.example AMAZON_EKS_CNI_Policy: Destroying... [id=eks-node-group-cloud-20231205133150926900000004]
aws_iam_role_policy_attachment.example AMAZON_EKSWorkerNodePolicy: Destroying... [id=eks-node-group-cloud-20231205133150786300000003]
aws_iam_role_policy_attachment.example AMAZON_ECSContainerRegistryReadOnly: Destroying... [id=eks-node-group-cloud-20231205133150682600000002]
aws_eks_cluster.example: Destroying... [id=EKS_CLOUD]
aws_iam_role_policy_attachment.example AMAZON_ECSContainerRegistryReadOnly: Destruction complete after 0s
aws_iam_role_policy_attachment.example AMAZON_EKSWorkerNodePolicy: Destruction complete after 0s
aws_iam_role_policy_attachment.example AMAZON_EKS_CNI_Policy: Destruction complete after 0s
aws_iam_role.example1: Destroying... [id=eks-node-group-cloud]
aws_iam_role.example1: Destruction complete after 1s AJAY MR CLOUD BOOK
aws_eks_cluster.example: Still destroying... [id=EKS_CLOUD, 10s elapsed]
aws_eks_cluster.example: Still destroying... [id=EKS_CLOUD, 20s elapsed]
aws_eks_cluster.example: Still destroying... [id=EKS_CLOUD, 30s elapsed]
aws_eks_cluster.example: Still destroying... [id=EKS_CLOUD, 40s elapsed]
aws_eks_cluster.example: Still destroying... [id=EKS_CLOUD, 50s elapsed]
aws_eks_cluster.example: Still destroying... [id=EKS_CLOUD, 1m0s elapsed]
aws_eks_cluster.example: Still destroying... [id=EKS_CLOUD, 1m10s elapsed]
aws_eks_cluster.example: Still destroying... [id=EKS_CLOUD, 1m20s elapsed]
aws_eks_cluster.example: Still destroying... [id=EKS_CLOUD, 1m30s elapsed]
aws_eks_cluster.example: Still destroying... [id=EKS_CLOUD, 1m40s elapsed]
aws_eks_cluster.example: Destruction complete after 1m42s
aws_iam_role_policy_attachment.example AMAZON_EKSClusterPolicy: Destroying... [id=eks-cluster-cloud-20231205133150541700000001]
aws_iam_role_policy_attachment.example AMAZON_EKSClusterPolicy: Destruction complete after 1s
aws_iam_role.example: Destroying... [id=eks-cluster-cloud]
aws_iam_role.example: Destruction complete after 0s

Destroy complete! Resources: 8 destroyed.
root@ip-172-31-41-147:/home/ubuntu/k8s-mario/EKS-TF#
```

Thank you for joining this nostalgic journey to the 90s! We hope you enjoyed rekindling your love for gaming with the deployment of the iconic Mario game on Kubernetes. Embracing the past while exploring new technologies is a true testament to the timeless allure of classic games. Until next time, keep gaming and reliving those fantastic memories! 🎮🎮.



**Ajay Kumar Yegireddi** is a DevSecOps Engineer and System Administrator, with a passion for sharing real-world DevSecOps projects and tasks. **Mr. Cloud Book**, provides hands-on tutorials and practical insights to help others master DevSecOps tools and workflows. Content is designed to bridge the gap between development, security, and operations, making complex concepts easy to understand for both beginners and professionals.

# Comments

## 8 responses to “Deploying Super Mario on Kubernetes”

**harun**

11 January 2024

if you want to apply terraform destroy command. you must run it in the EKS\_TF folder.  
good project thank you..

[Reply](#)**isaac Ambi**

13 January 2024

I am so excited that I found the channel, every single detail is explained very much

[Reply](#)**mrcloudbook.com**

14 January 2024

glad it helped

[Reply](#)**DP**

1 February 2024

Hello Mr cloud book

Content and projects are really helpful. Can we also have E2E implementation projects on Azure cloud using azure portal services and Azure DevOps

Thanks,  
DP

[Reply](#)

---



**Ajay**  
6 February 2024

Hi,

It was a great guide to get EKS deployed in AWS. We were able to see the Mario game running by hitting the loadbalancer ingress.

However when we see the EKS cluster in AWS Console there are no deployments and services seen in there.

Thanks,  
Ajay V

[Reply](#)

---



**david prabhakar**  
22 March 2024

hi bro just now i completed this simple project thanks a lot and i am facing some issues while setup k8s i gone through many tutorials if possible can you help me . this is my insta id "davidprabhakar67"

thank you.

[Reply](#)

---



**mehul**  
5 May 2024

IS IT OK TO POST THIS PROJECT ON LINKEDIN IN UK? I MEAN COMPLIANCE WITH REGION?

[Reply](#)

---

**mrcloudbook.com**

8 May 2024

Sure you can post

[Reply](#)

## Leave a Reply

Your email address will not be published. Required fields are marked \*

Comment \*

Name \*

Email \*

Website

Save my name, email, and website in this browser for the next time I comment.

I'm not a robot

reCAPTCHA  
[Privacy](#) - [Terms](#)

[Post Comment](#)

Uncategorized

How to Automate  
Incident Response :  
How Q Developer  
Helped Me Automate a  
Daily Pain Point

22 July 2025

AI

How to Run Docker  
Model Runner on  
Ubuntu 24.04

11 July 2025

AI, DevOps

How to Install docker-ai  
on Ubuntu 24.04

15 June 2025

## Upskill with Ajay: DevSecOps Mastery

Join Mr Cloud book to master DevSecOps through real-world projects. Learn CI/CD, security integration, automation, and more, gaining hands-on skills for industry-level challenges.



## Important Links

[Privacy Policy](#)

[Terms & Conditions](#)

[Contact](#)

## Resources

[Blog](#)

[YouTube Channel](#)

