DVS Technologies Aws & Devops

Compiled and Scrutinized by
Mr. Shaan Shaik
(Senior DevOps Lead)

Words To The Students

Though we have taken utmost efforts to present you this book error free, but still it may contain some errors or mistakes. Students are encouraged to bring, if there are any mistakes or errors in this document to our notice. So that it may be rectified in the next edition of this document.

"Suppressing your doubts is Hindering your growth".

We urge you to work hard and make use of the facilities we are providing to you, because there is no substitute for hard work. We wish you all the best for your future.

"The grass isn't greener on the other side; the grass is greener where you water it."

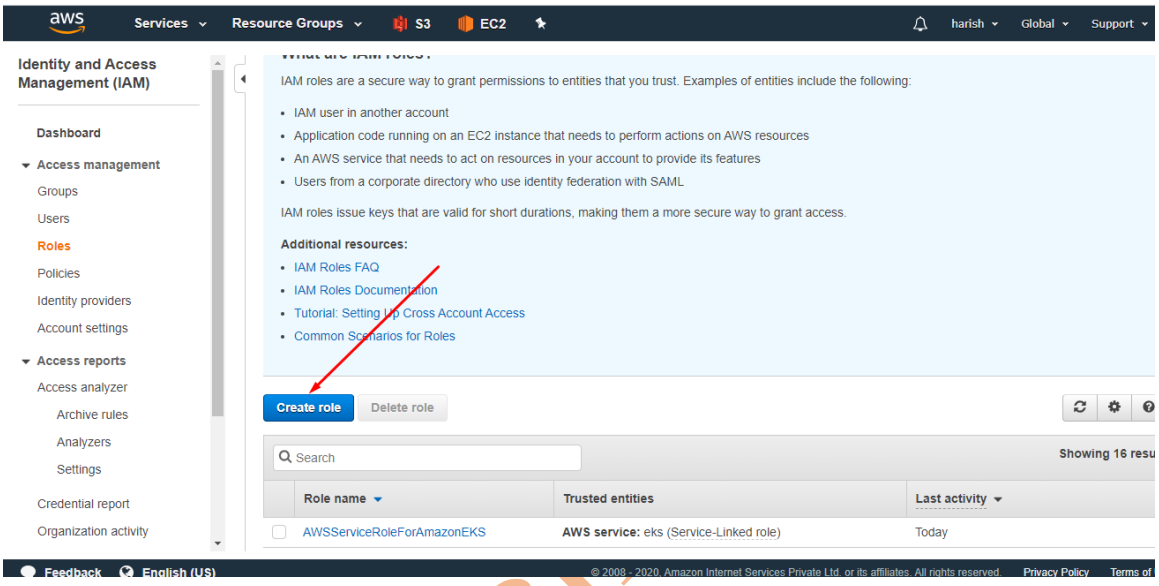
You and your suggestions are valuable to us; Help us to serve you better. In case of any suggestions, grievance, or complaints, please feel free to write us your suggestions, grievance and feedback on the following

Dvs.training@gmail.com

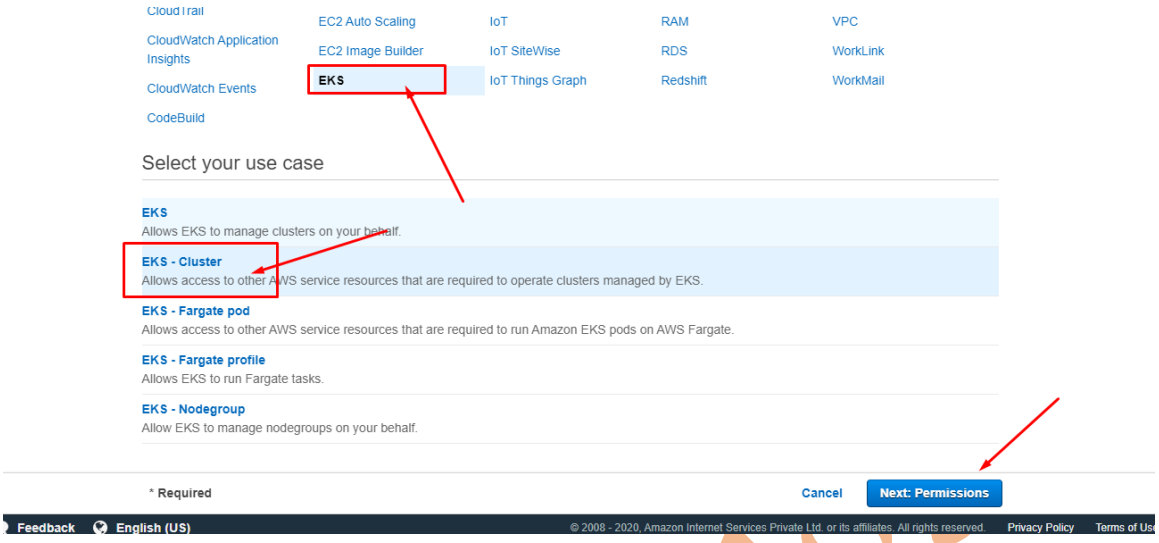
DVS Technologies Aws & Devops

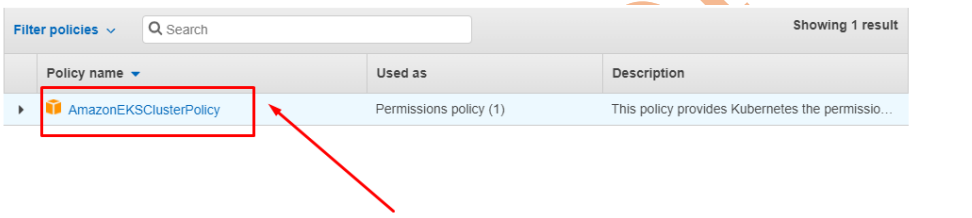
1. Installation and configuration

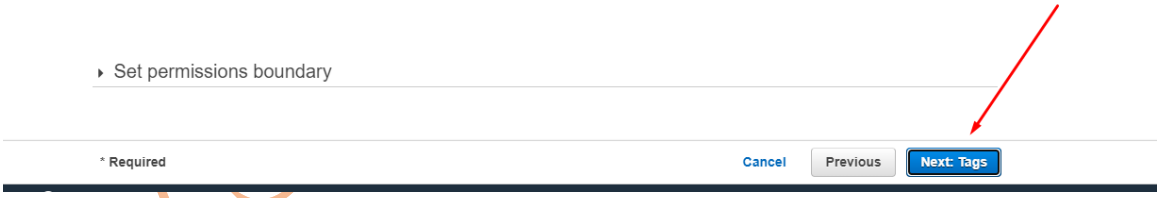
Role Creation:



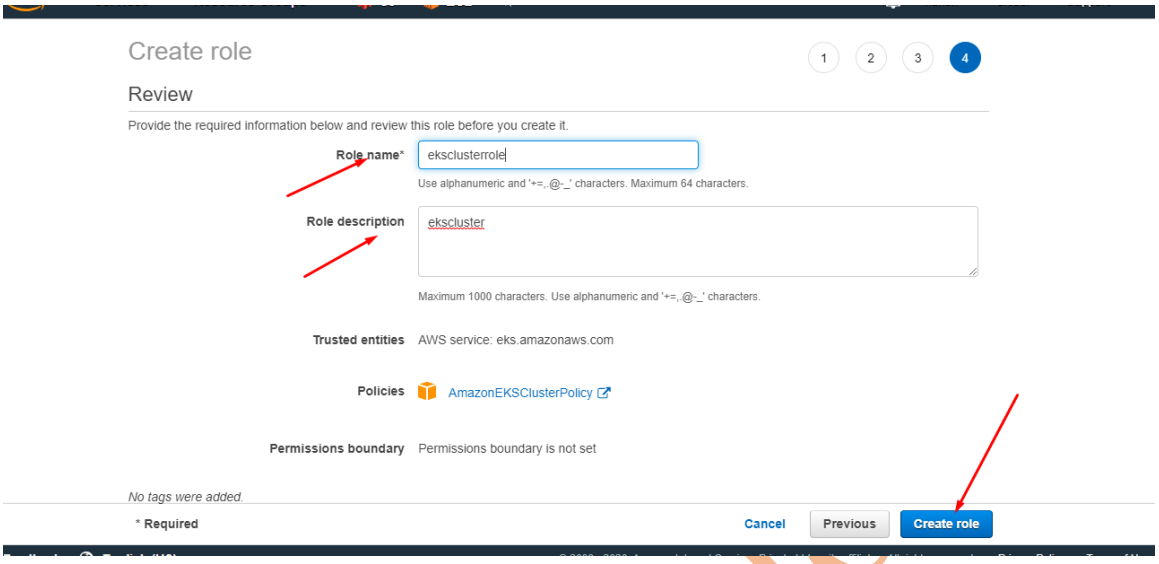
DVS Technologies Aws & Devops

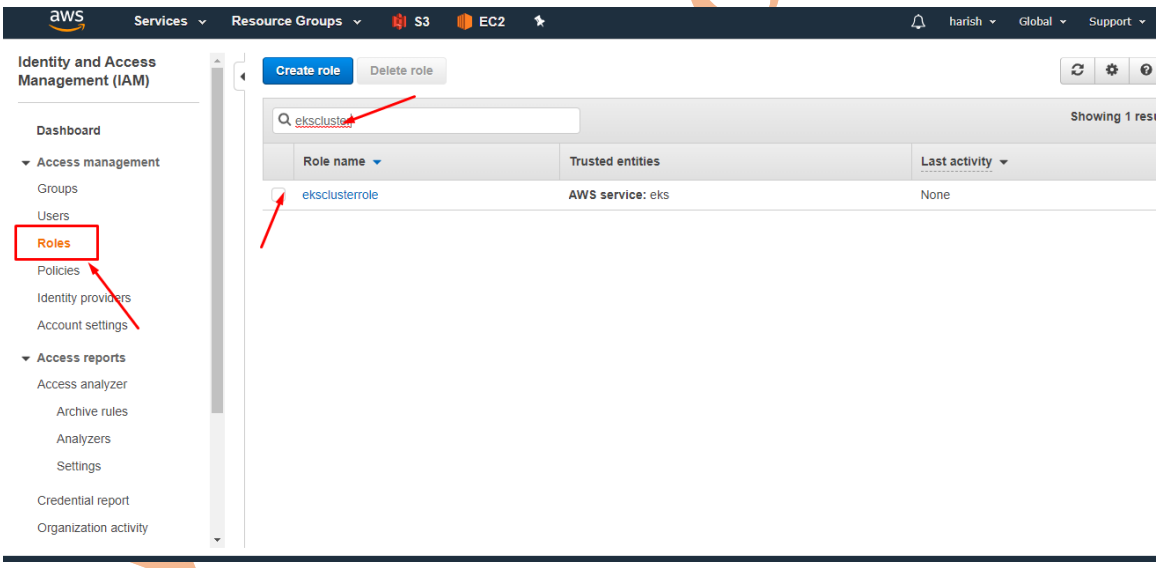




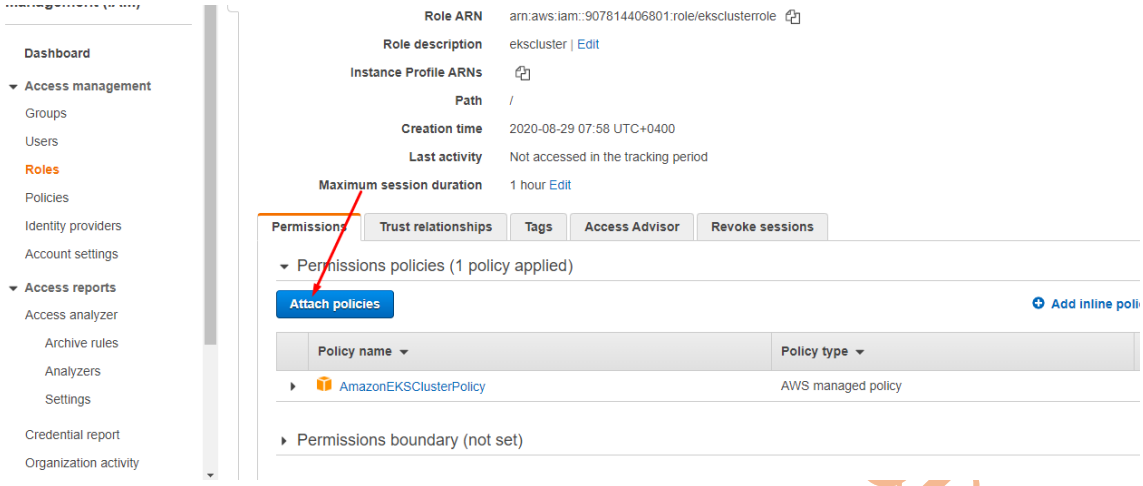


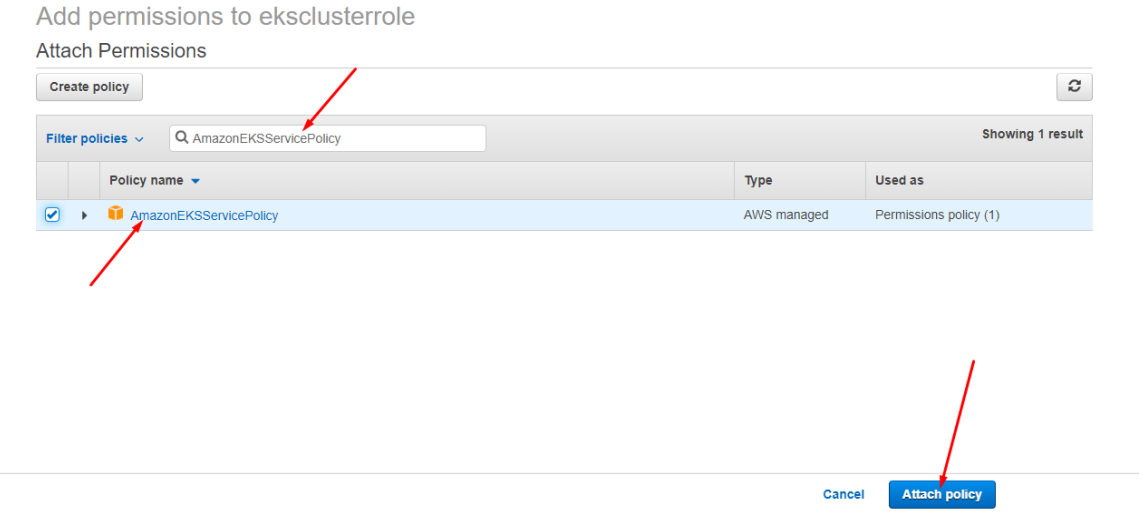
DVS Technologies Aws & Devops



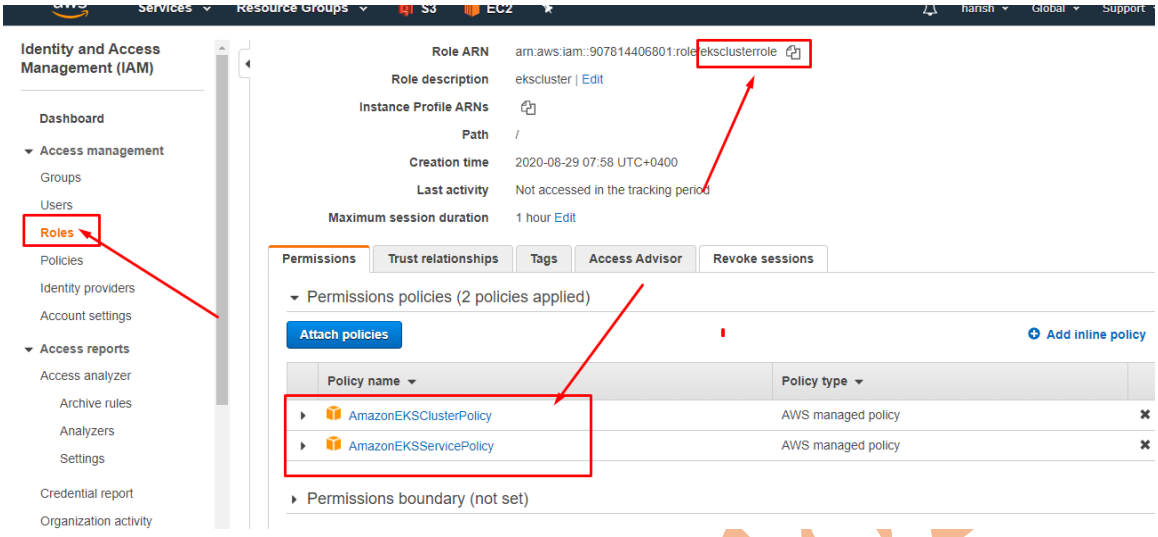


DVS Technologies Aws & Devops

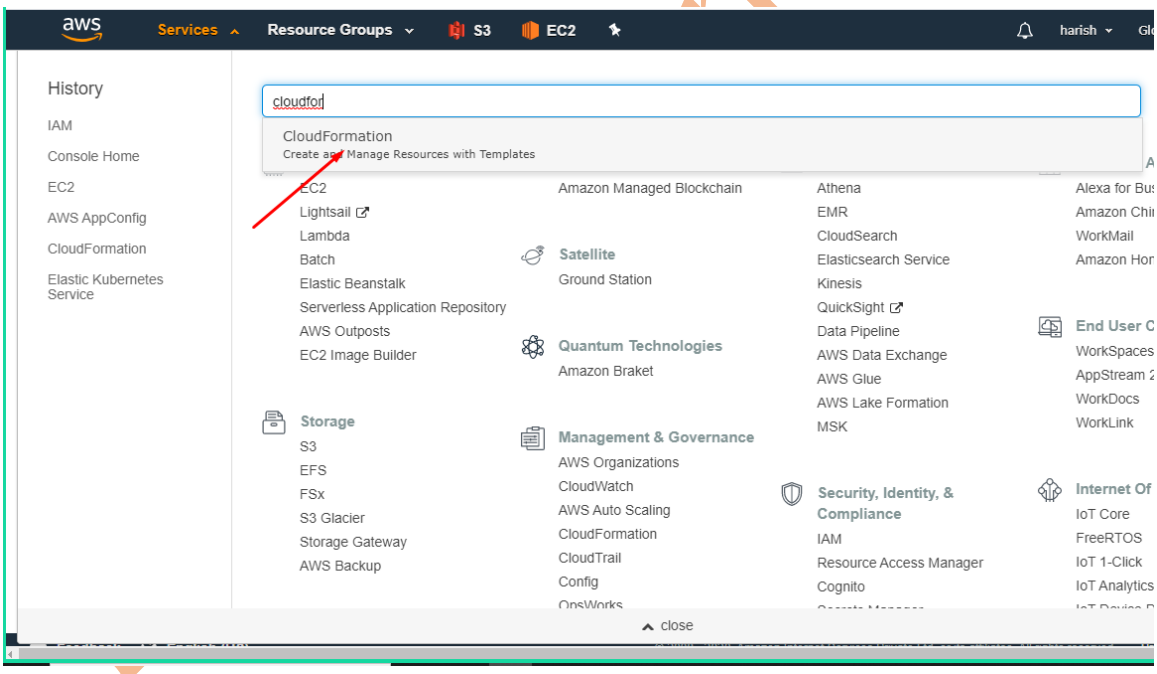




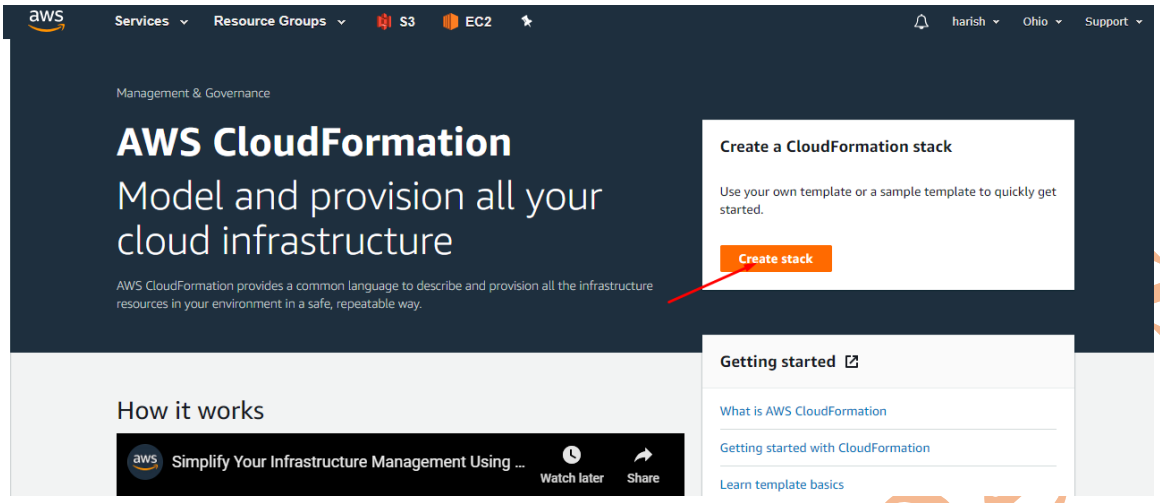
DVS Technologies Aws & Devops

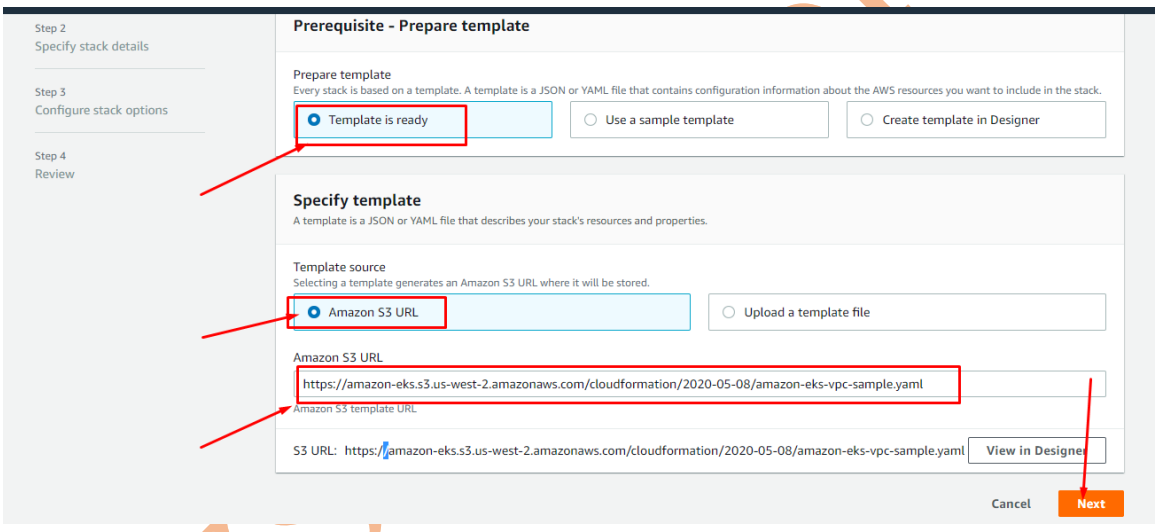


VPC Creation:

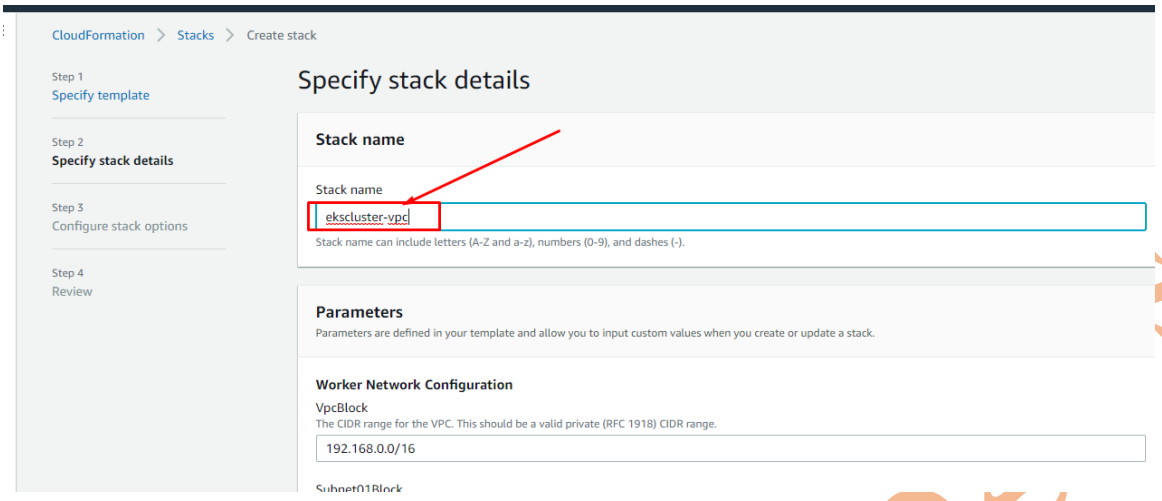


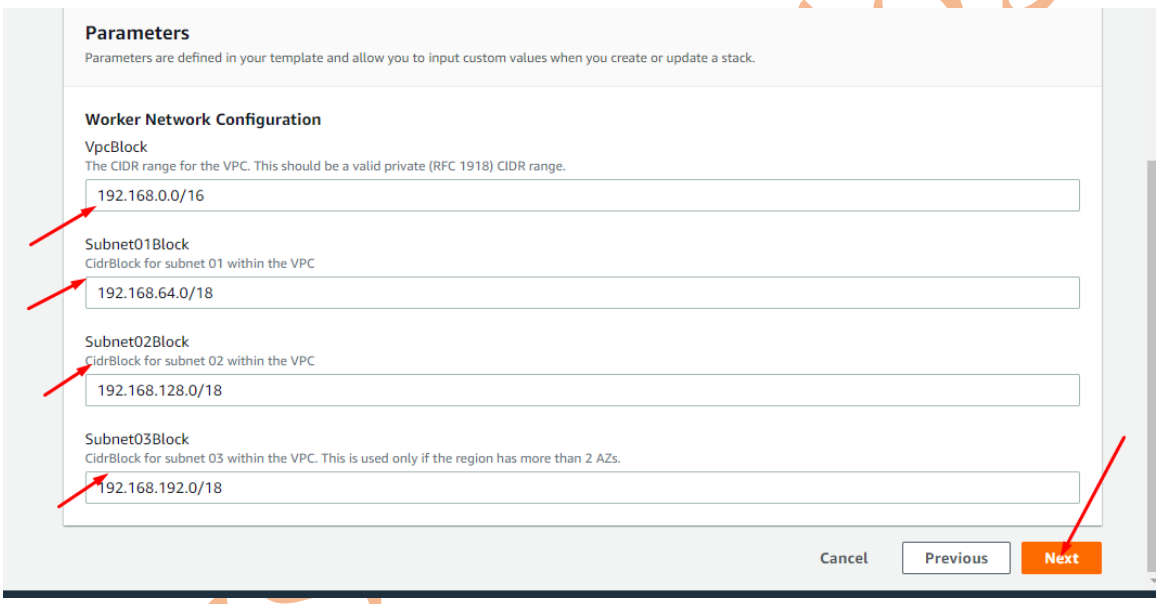
DVS Technologies Aws & Devops

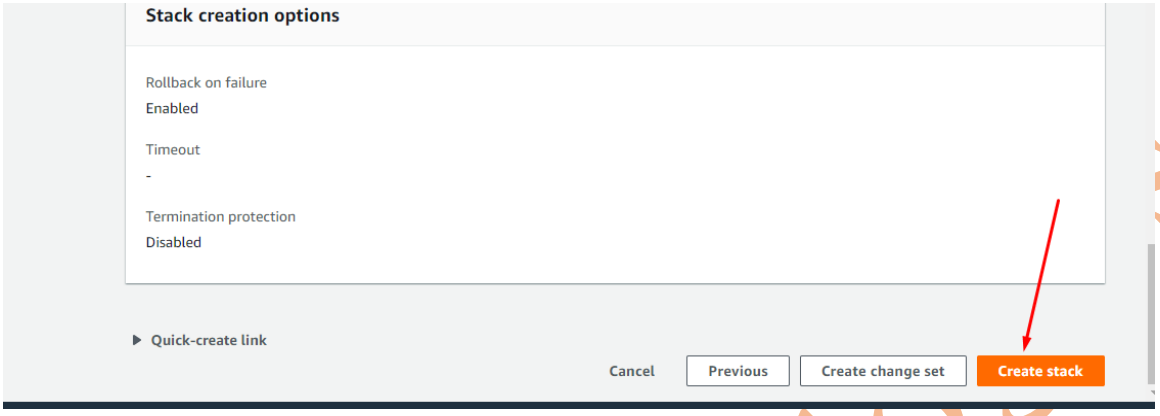




DVS Technologies Aws & Devops

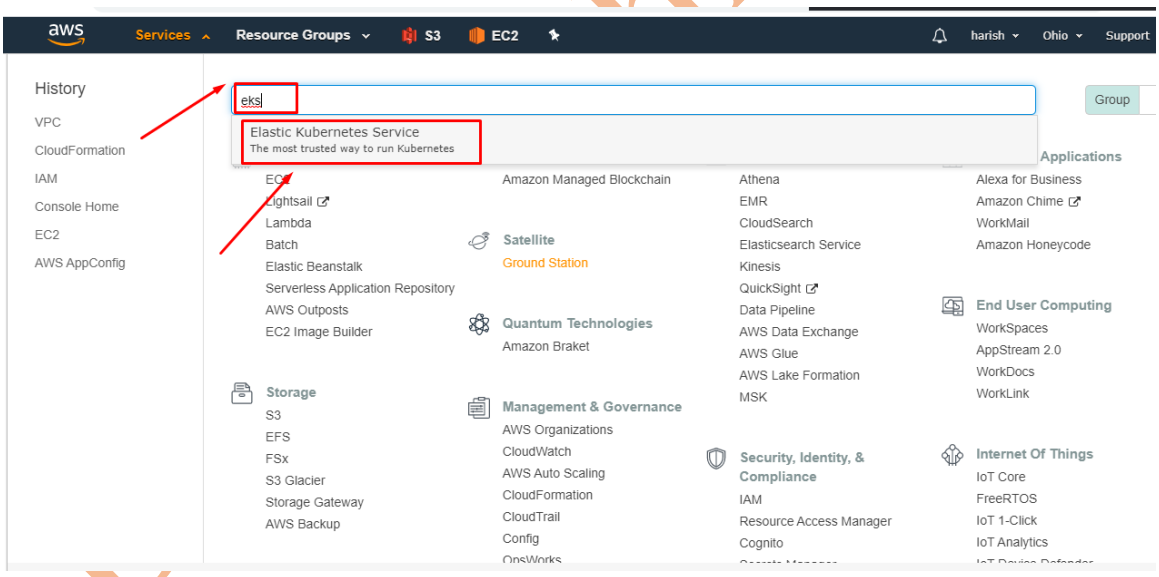




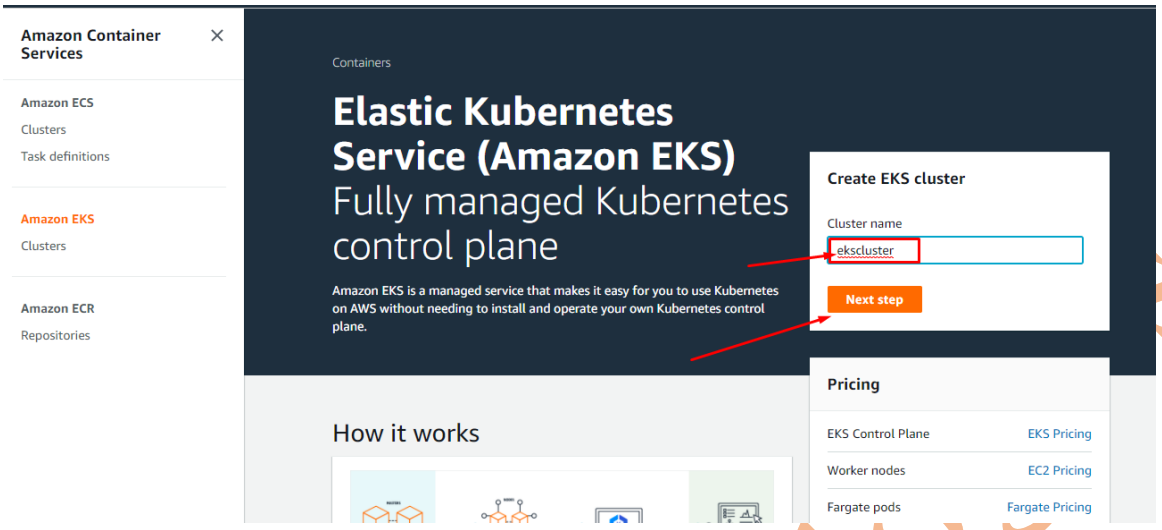


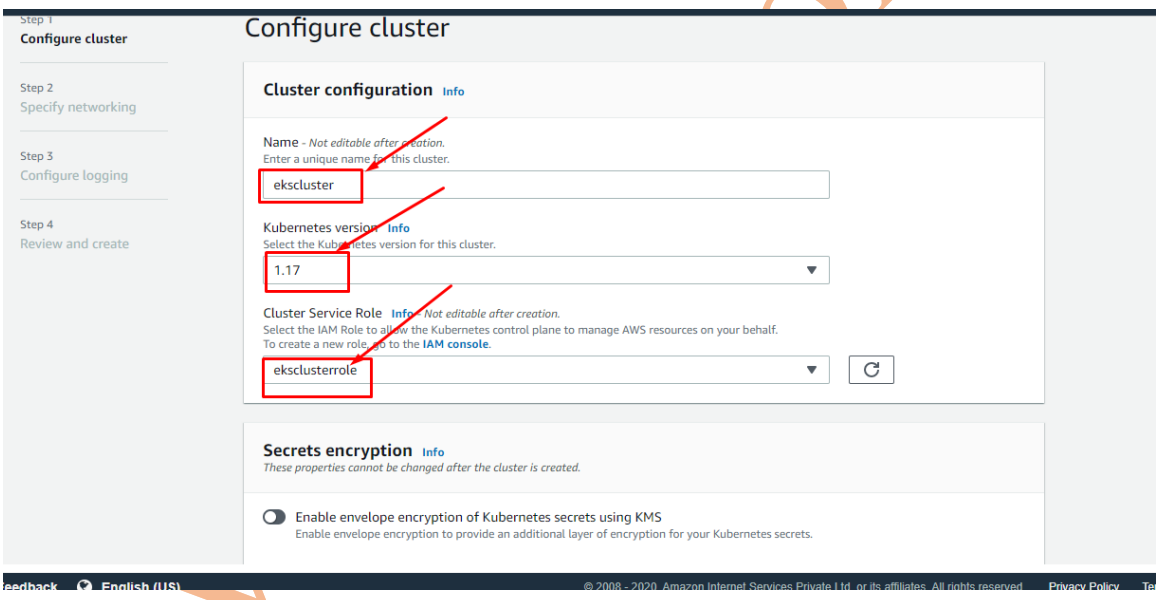
Note: Wait Till you get your VPC

Let's create our EKS cluster now:

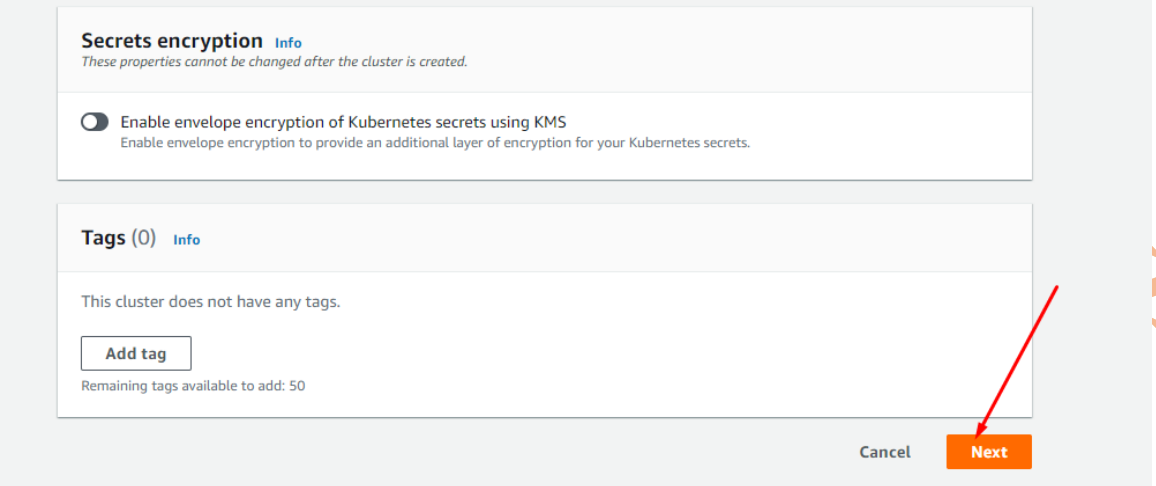


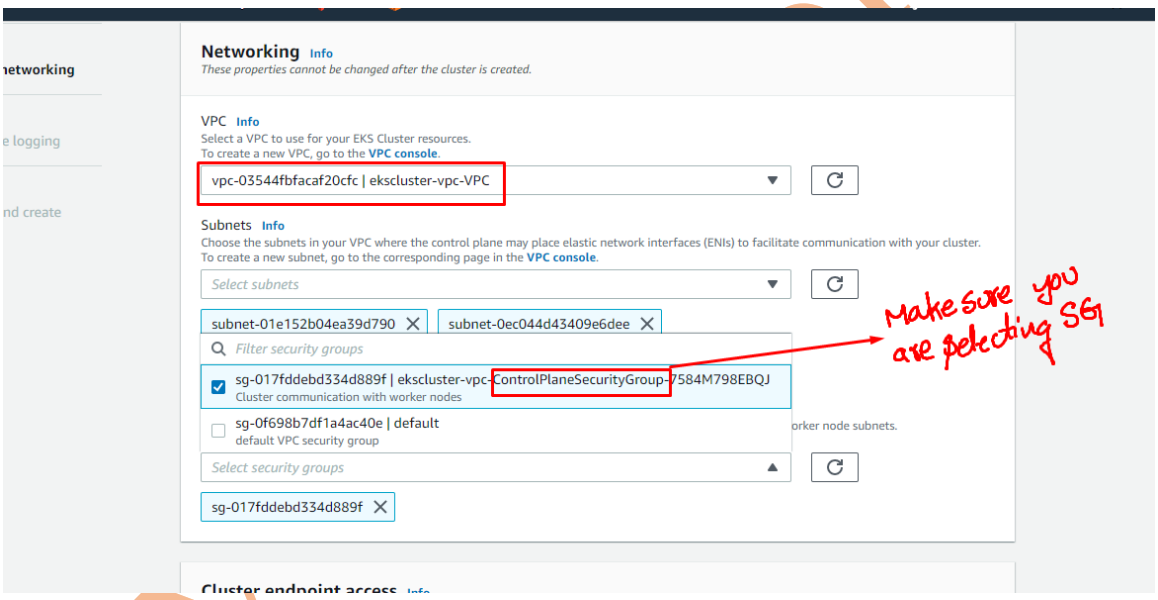
DVS Technologies Aws & Devops

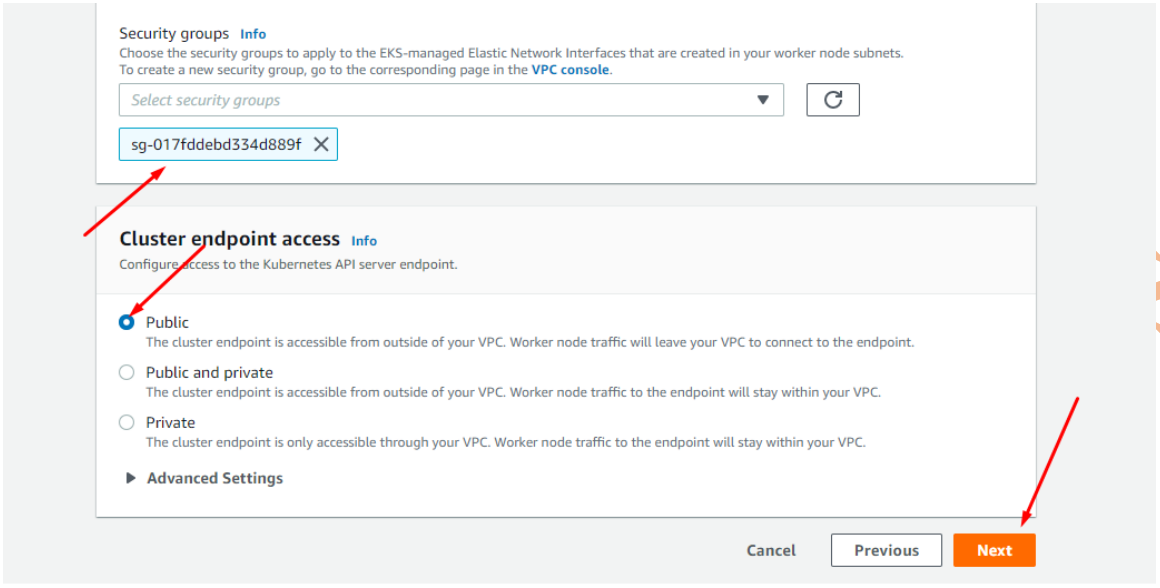


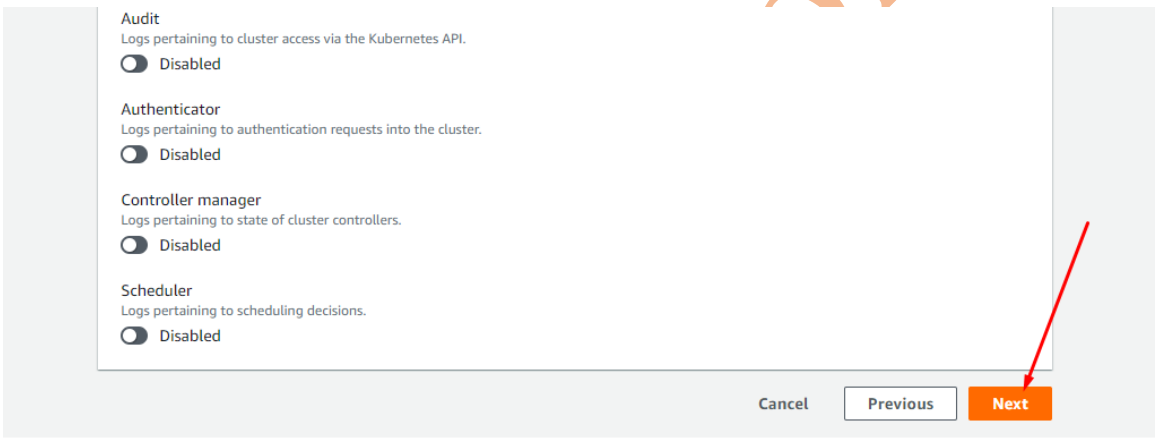


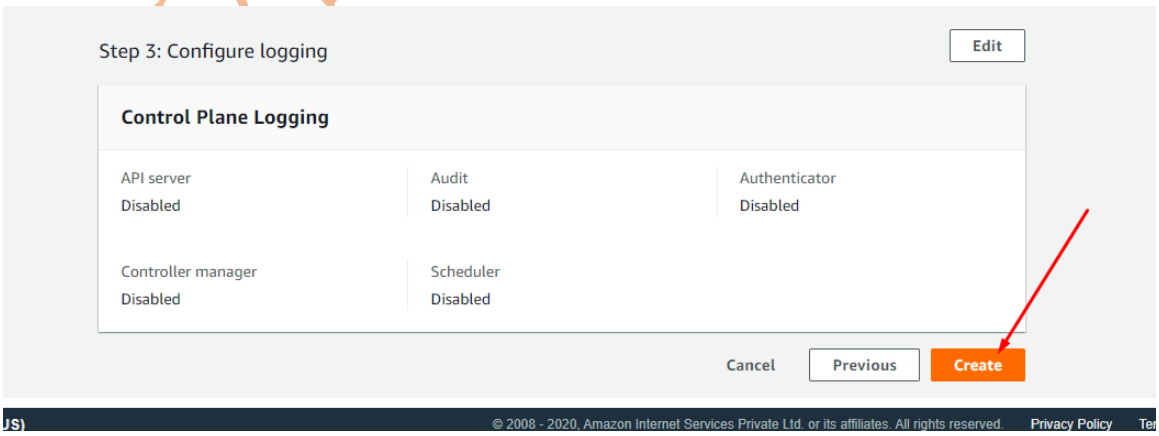
DVS Technologies Aws & Devops

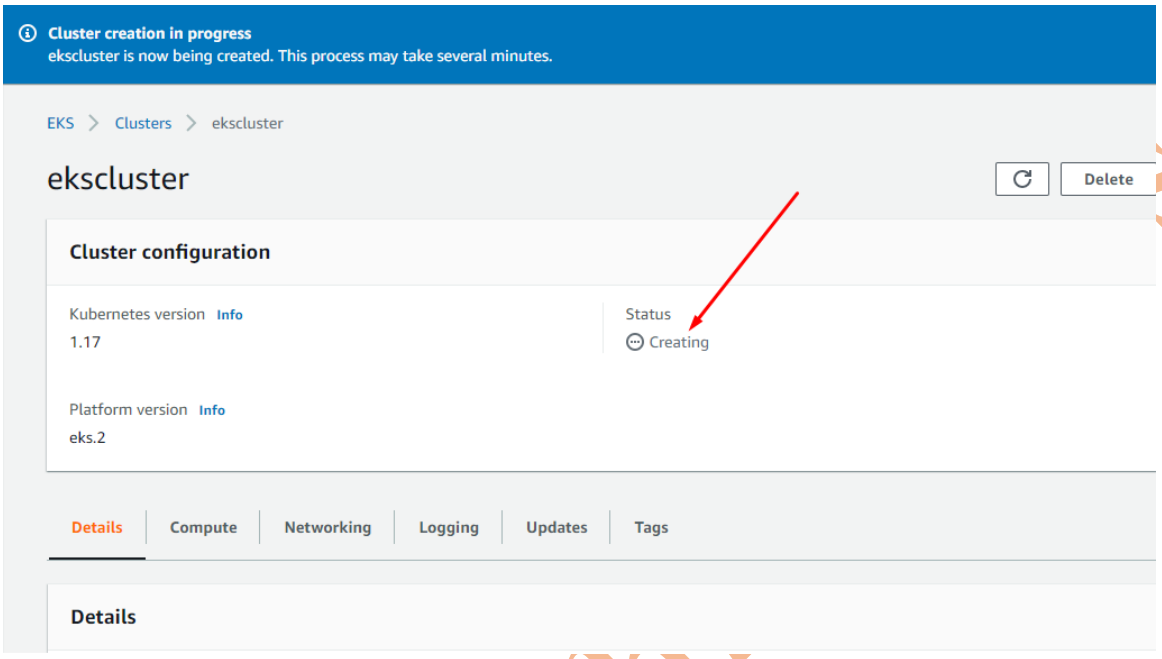





Security groups Info
Choose the security groups to apply to the EKS-managed Elastic Network Interfaces that are created in your worker node subnets.
To create a new security group, go to the corresponding page in the VPC console.
Select security groups
sg-017fddebd334d889f
Cluster endpoint access Info
Configure access to the Kubernetes API server endpoint.
Public
The cluster endpoint is accessible from outside of your VPC. Worker node traffic will leave your VPC to connect to the endpoint.
Public and private
The cluster endpoint is accessible from outside of your VPC. Worker node traffic to the endpoint will stay within your VPC.
Private
The cluster endpoint is only accessible through your VPC. Worker node traffic to the endpoint will stay within your VPC.
Advanced Settings
Cancel
Previous
Next

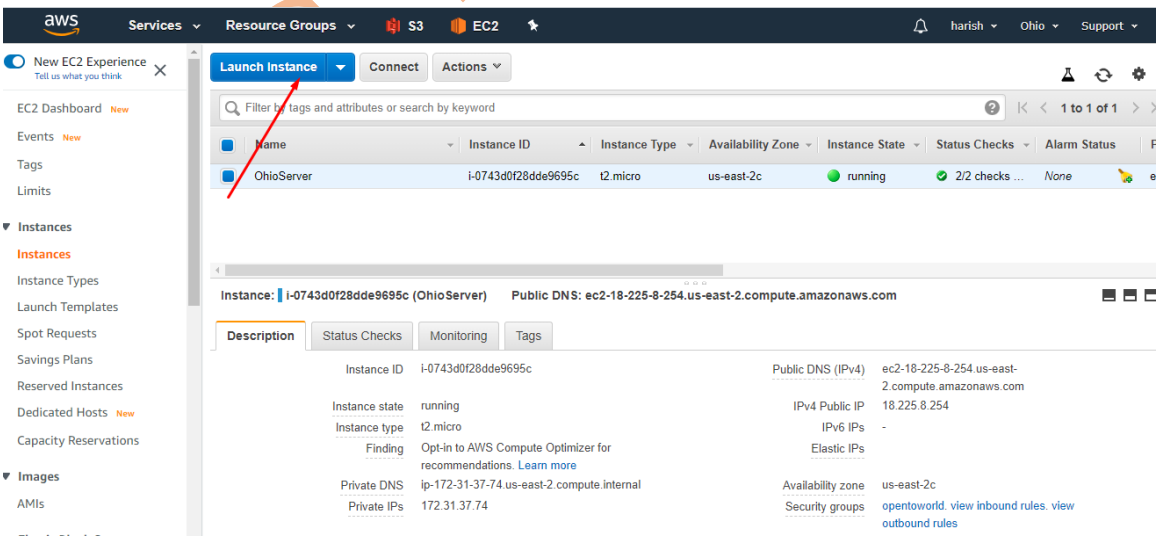

Audit
Logs pertaining to cluster access via the Kubernetes API.
Disabled
Authenticator
Logs pertaining to authentication requests into the cluster.
Disabled
Controller manager
Logs pertaining to state of cluster controllers.
Disabled
Scheduler
Logs pertaining to scheduling decisions.
Disabled
Cancel
Previous
Next


Step 3: Configure logging
Edit
Control Plane Logging
API server
Disabled
Audit
Disabled
Authenticator
Disabled
Controller manager
Disabled
Scheduler
Disabled
Cancel
Previous
Create
© 2008 - 2020, Amazon Internet Services Private Ltd. or its affiliates. All rights reserved.
Privacy Policy

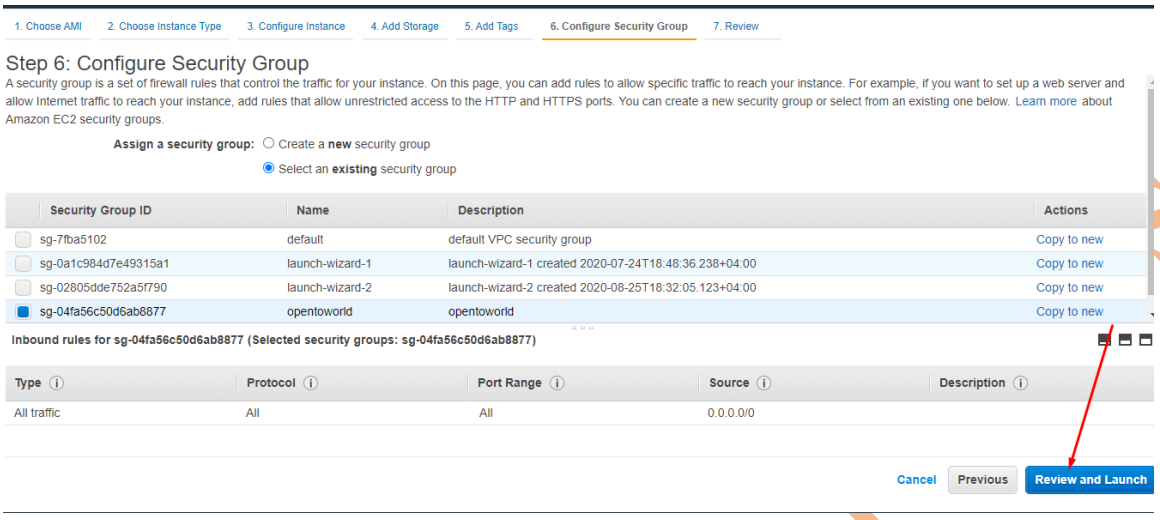


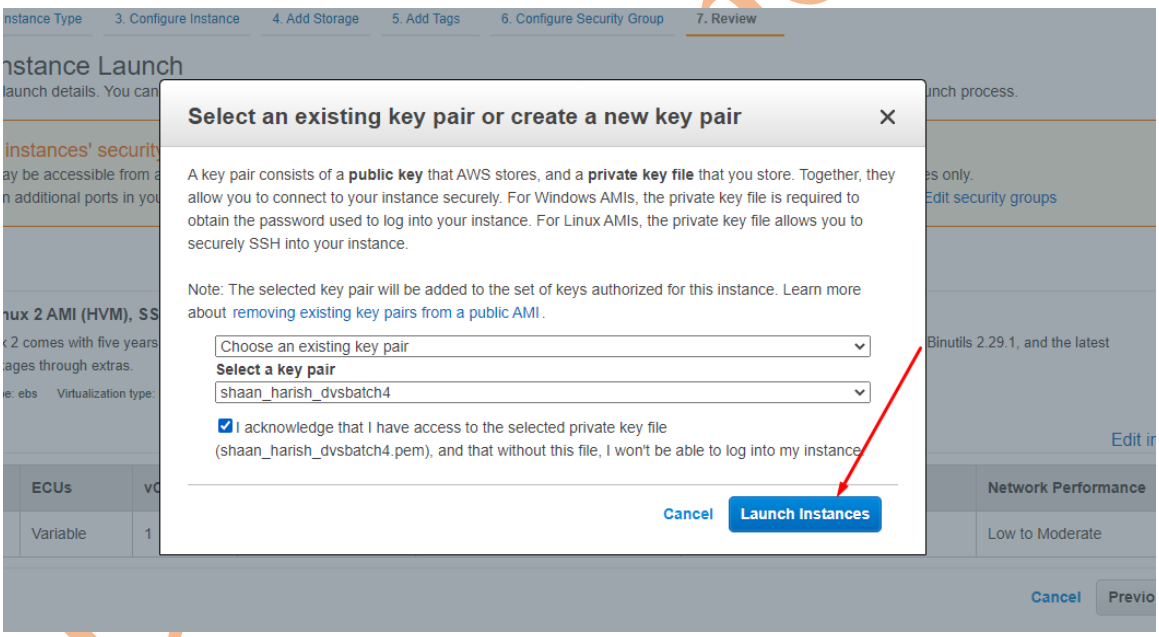
Wait Till it is available

Create & Configure your workstation as below:

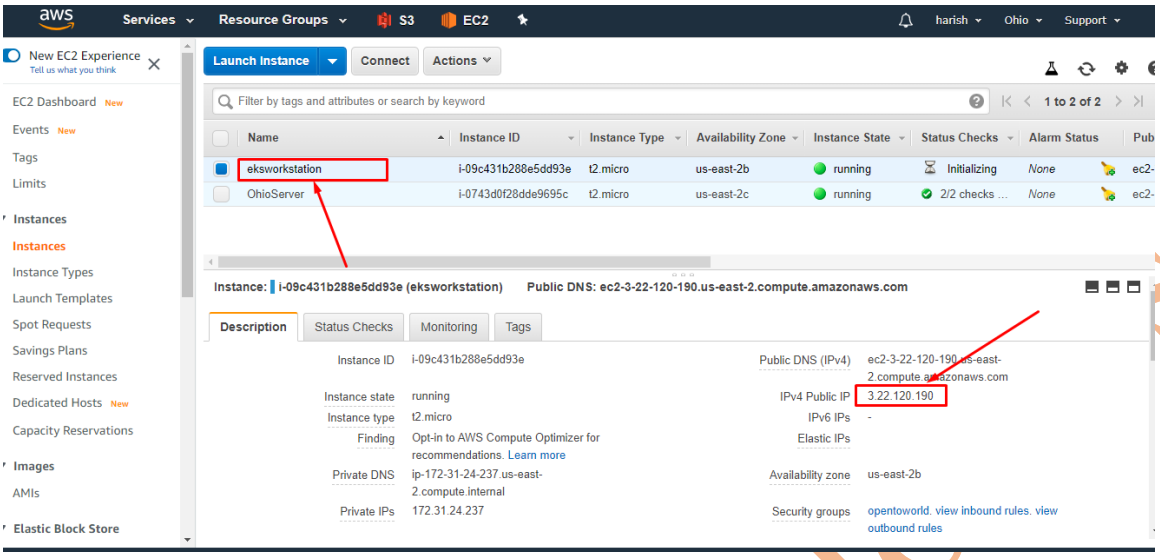


DVS Technologies Aws & Devops





DVS Technologies Aws & Devops




```
https://aws.amazon.com/amazon-linux-2/
7 package(s) needed for security, out of 14 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-24-237 ~]$ sudo hostnamectl set-hostname eksworkstation
[ec2-user@ip-172-31-24-237 ~]$ bash
[ec2-user@eksworkstation ~]$
```

Step1:
curl -o kubectl https://amazon-eks.s3.us-west-2.amazonaws.com/1.17.9/2020-08-04/bin/linux/amd64/kubectl
chmod +x ./kubectl
mkdir -p $HOME/bin && cp ./kubectl $HOME/bin/kubectl && export PATH=$PATH:$HOME/bin
echo 'export PATH=$PATH:$HOME/bin' >> ~/.bashrc
kubectl version --short --client

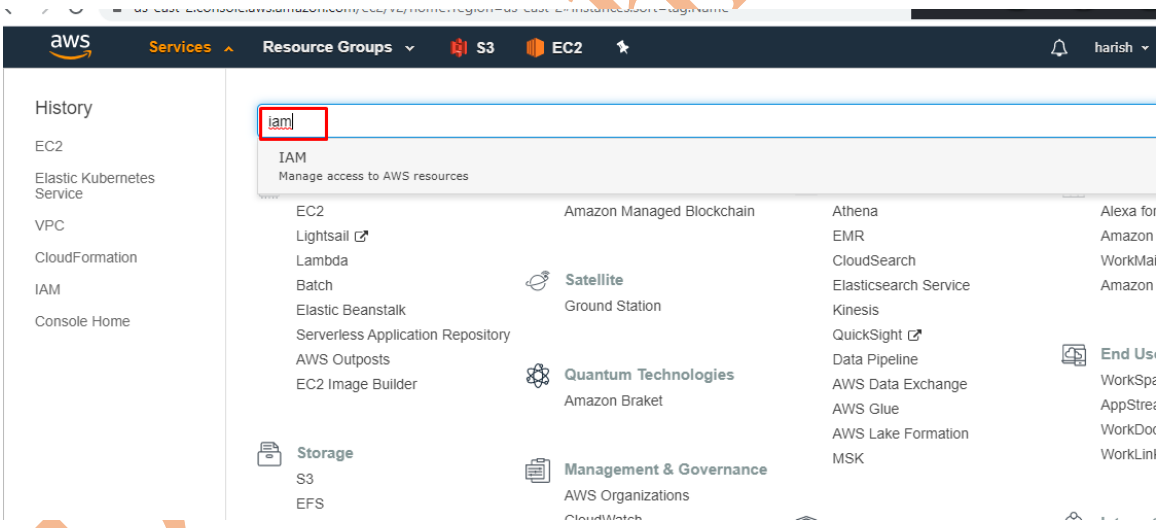
Step2:
Now inorder to access our cluster we should make sure that we are having aws-iam-authenticator
ref:
https://docs.aws.amazon.com/eks/latest/userguide/install-aws-iam-authenticator.html
https://github.com/kubernetes-sigs/aws-iam-authenticator

curl -o aws-iam-authenticator https://amazon-eks.s3.us-west-2.amazonaws.com/1.17.9/2020-08-04/bin/linux/amd64/aws-iam-authenticator
chmod +x ./aws-iam-authenticator
mkdir -p $HOME/bin && cp ./aws-iam-authenticator $HOME/bin/aws-iam-authenticator && export PATH=$PATH:$HOME/bin
aws-iam-authenticator help
A tool to authenticate to Kubernetes using AWS IAM credentials

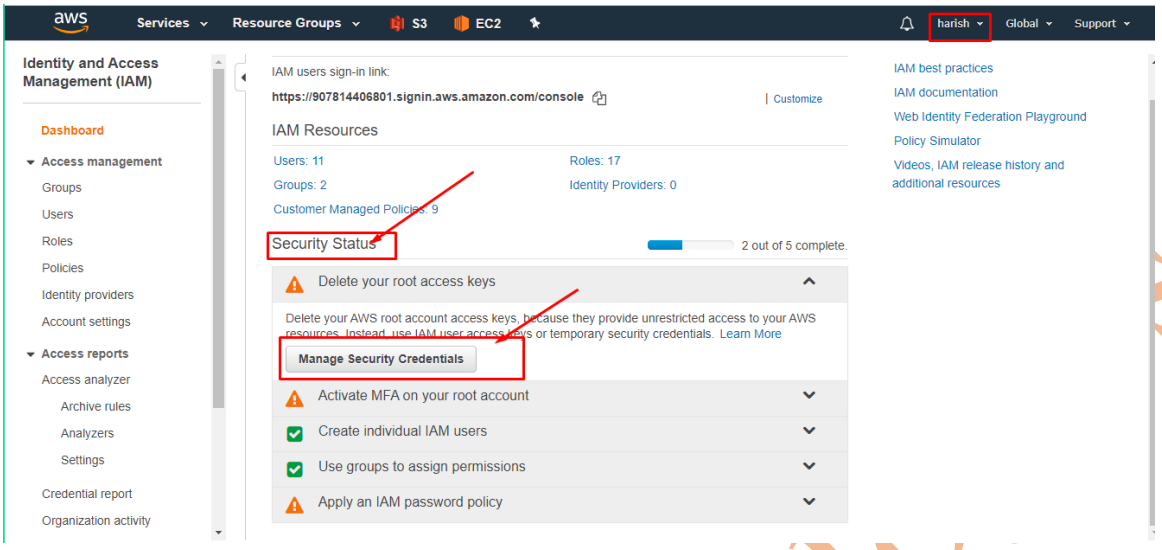
Step3:
Install and configure latest PIP package

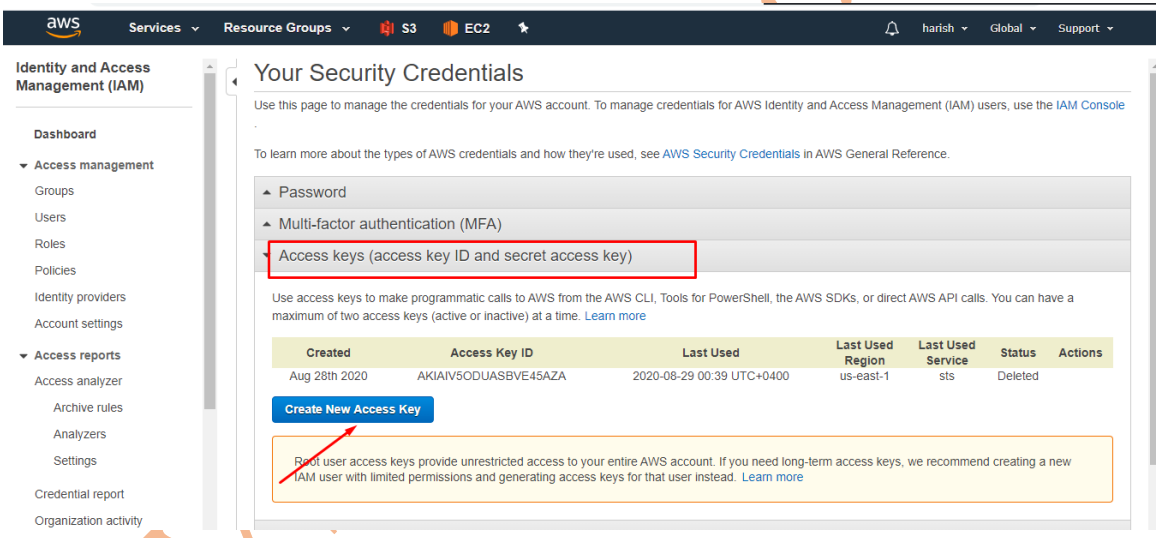
curl -O https://bootstrap.pypa.io/get-pip.py
python get-pip.py
pip install awscli --upgrade

NOTE: IF YOU ARE CREATING YOUR CLUSTER AS ROOT USER THEN YOU HAVE TO USE THE ROOT TOKEN FOR CONFIGURING YOU EKS CLUSTER LIKE BELOW

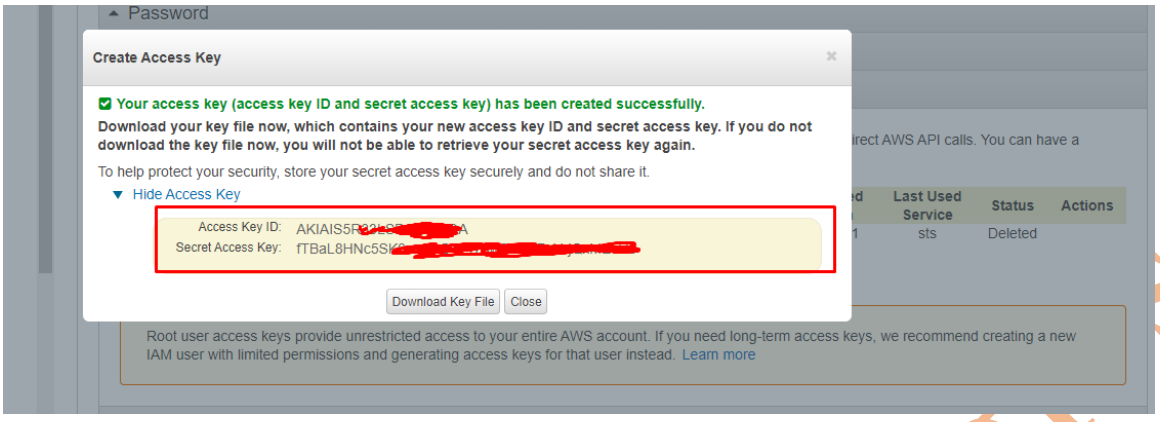


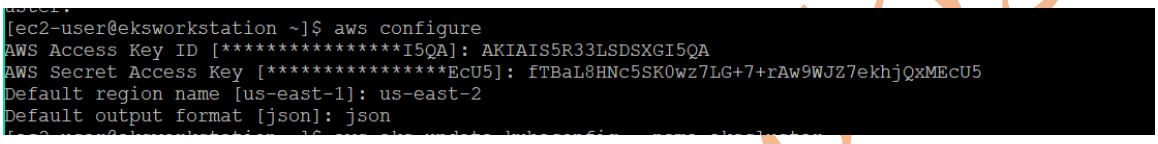
DVS Technologies Aws & Devops

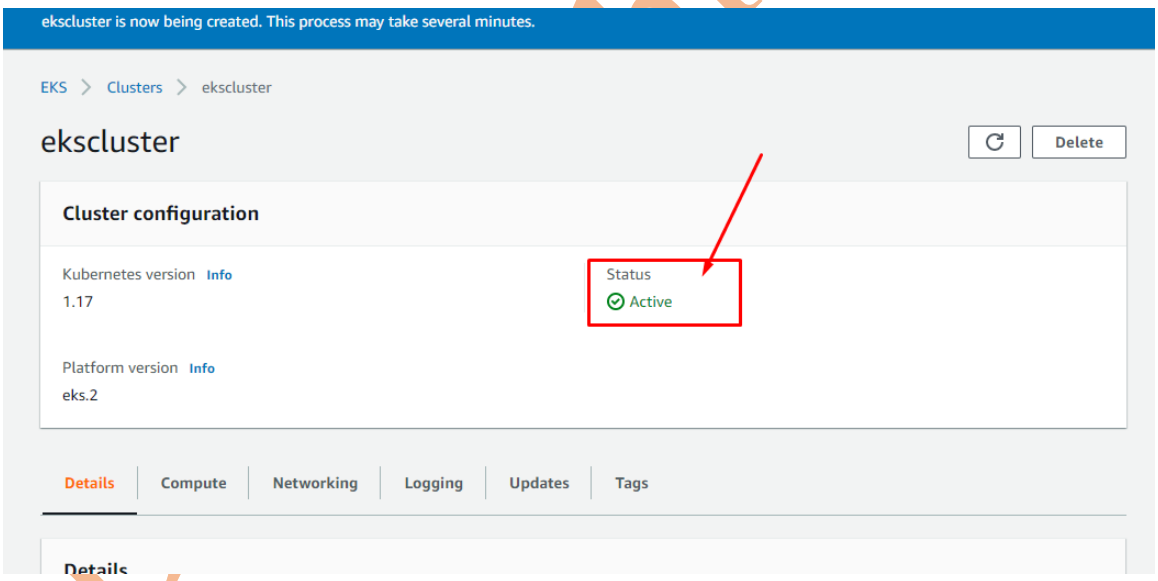




DVS Technologies Aws & Devops

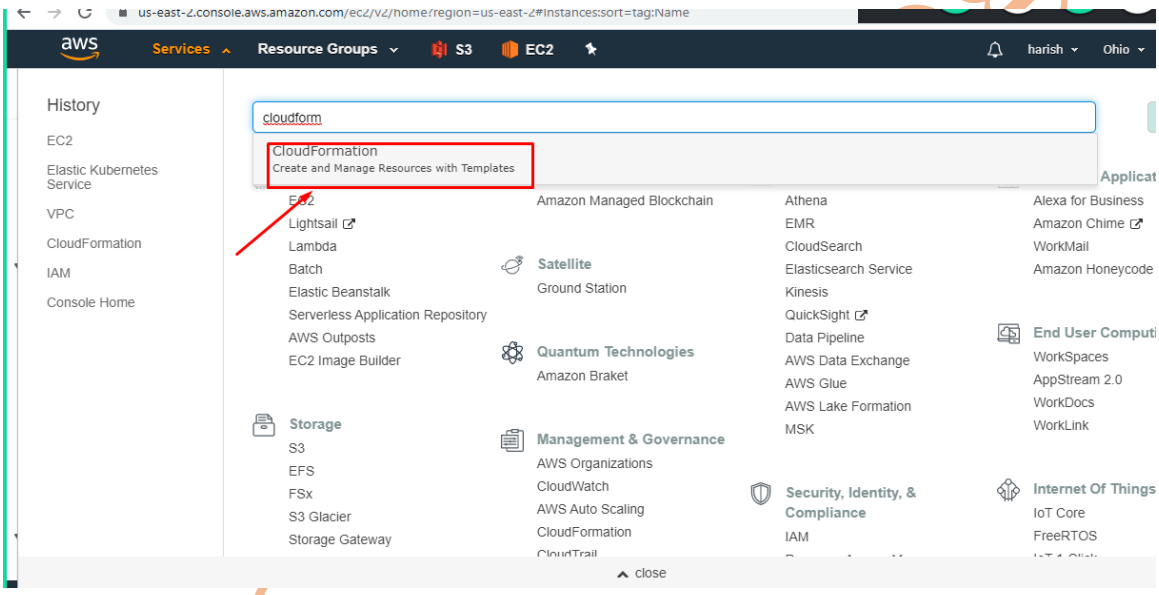


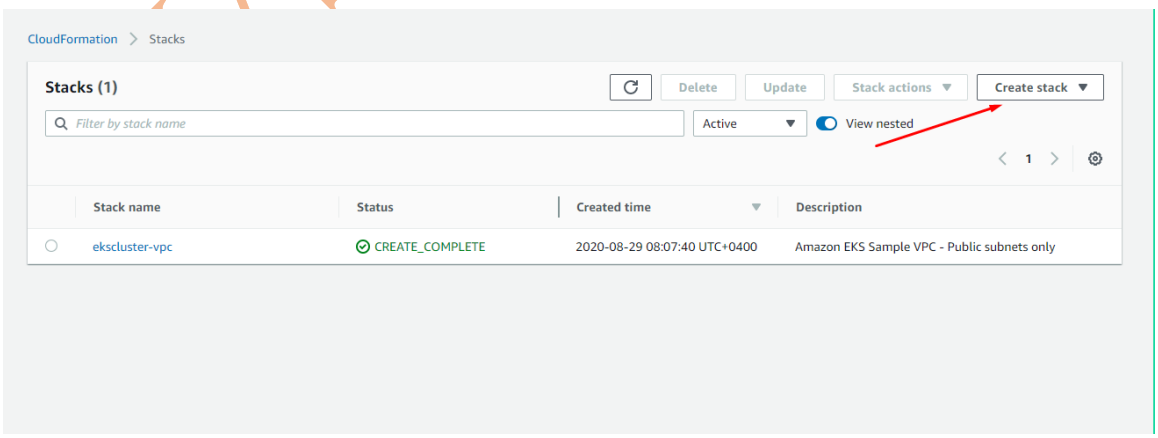




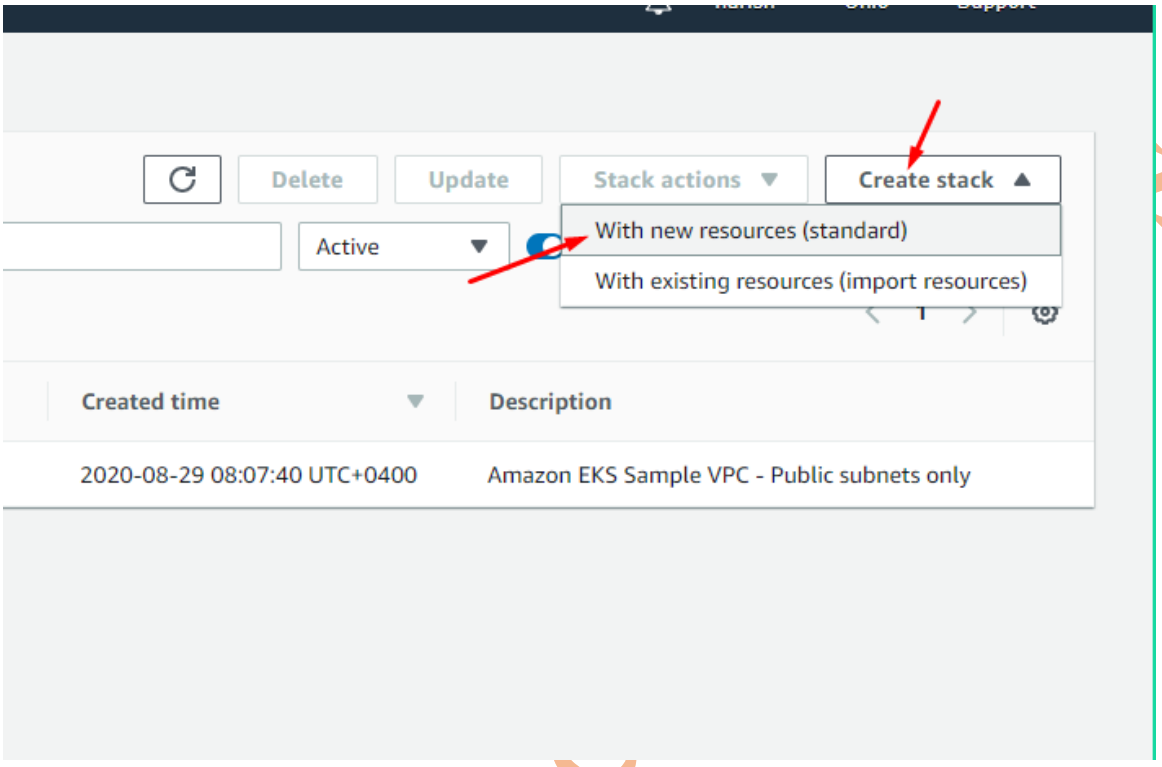
```
[ec2-user@eksworkstation ~]$ aws eks update-kubeconfig --name ekscluster
Updated context arn:aws:eks:us-east-2:907814406801:cluster/ekscluster in /home/ec2-user/.kube/config
[ec2-user@eksworkstation ~]$ kubectl cluster-info
Kubernetes master is running at https://FA9307D5C48FA56F3C194E2BF4967C7B.yl4.us-east-2.eks.amazonaws.com
CoreDNS is running at https://FA9307D5C48FA56F3C194E2BF4967C7B.yl4.us-east-2.eks.amazonaws.com/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy

To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.
[ec2-user@eksworkstation ~]$
```

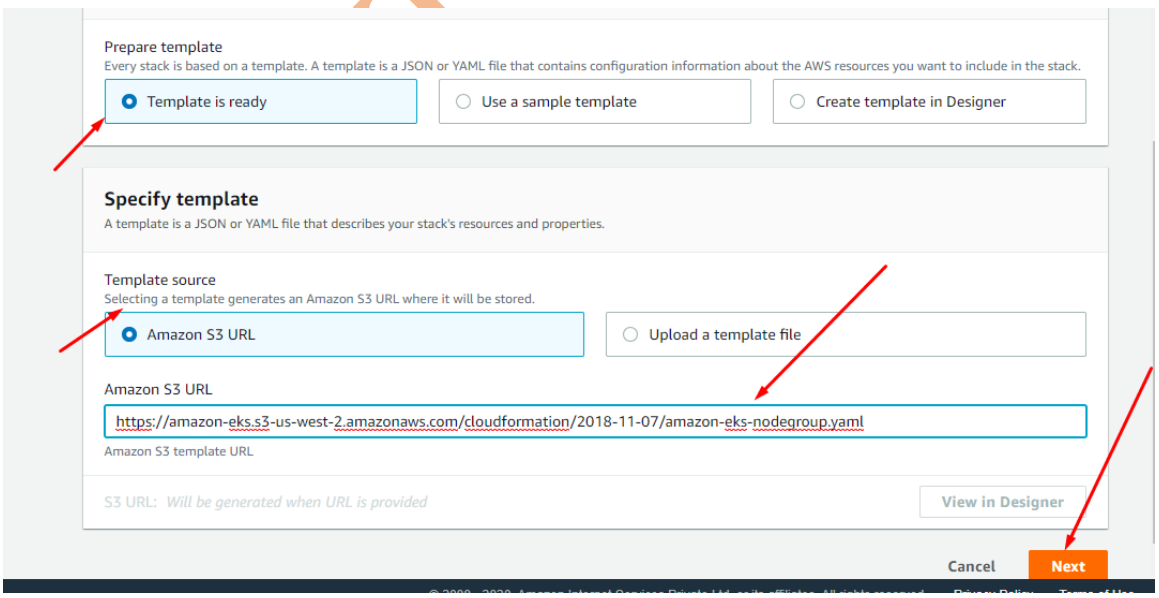
Provisioning Worker Nodes :

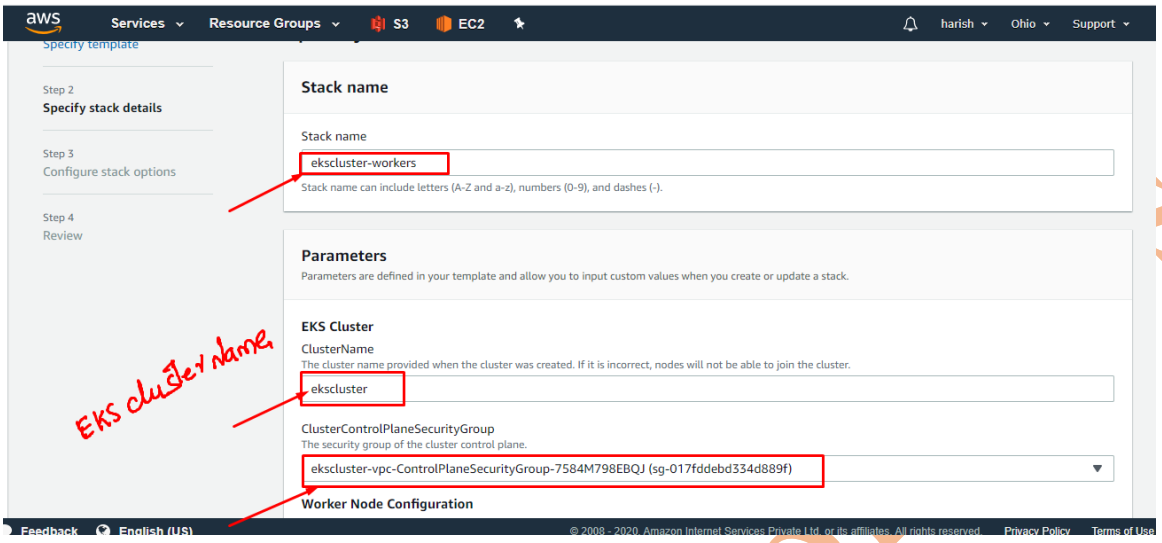


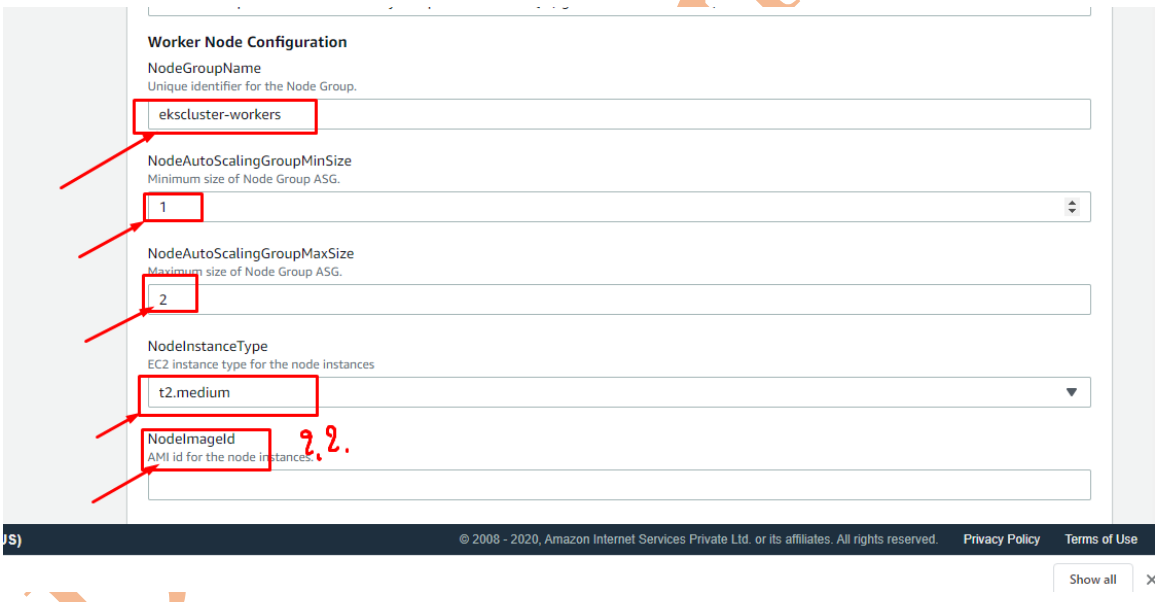


DVS Technologies Aws & Devops

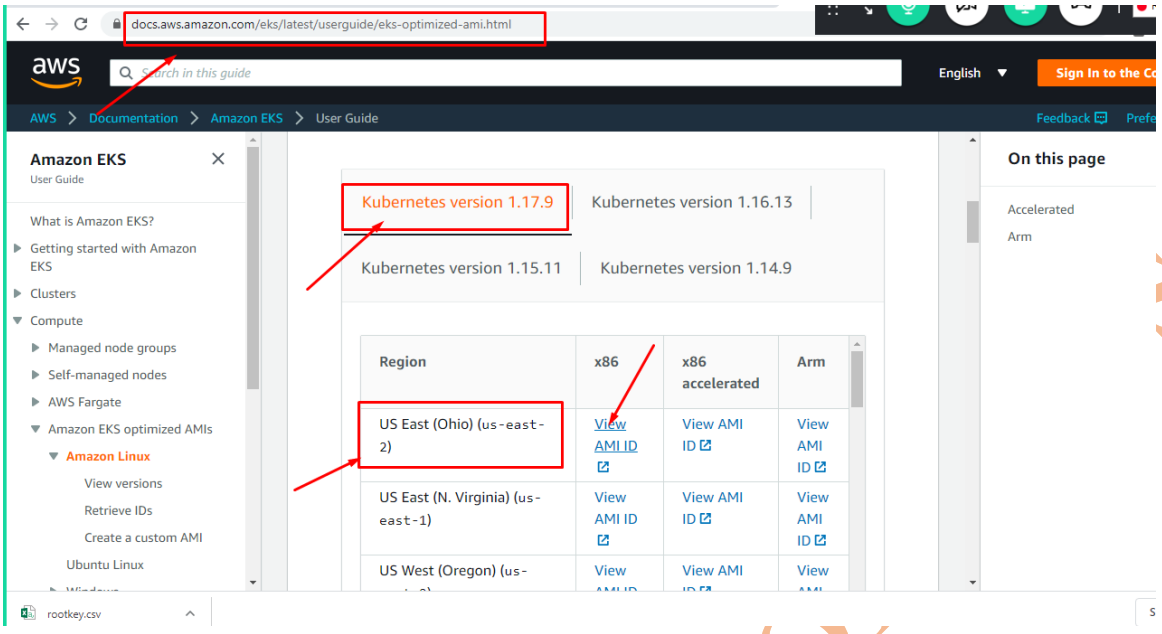




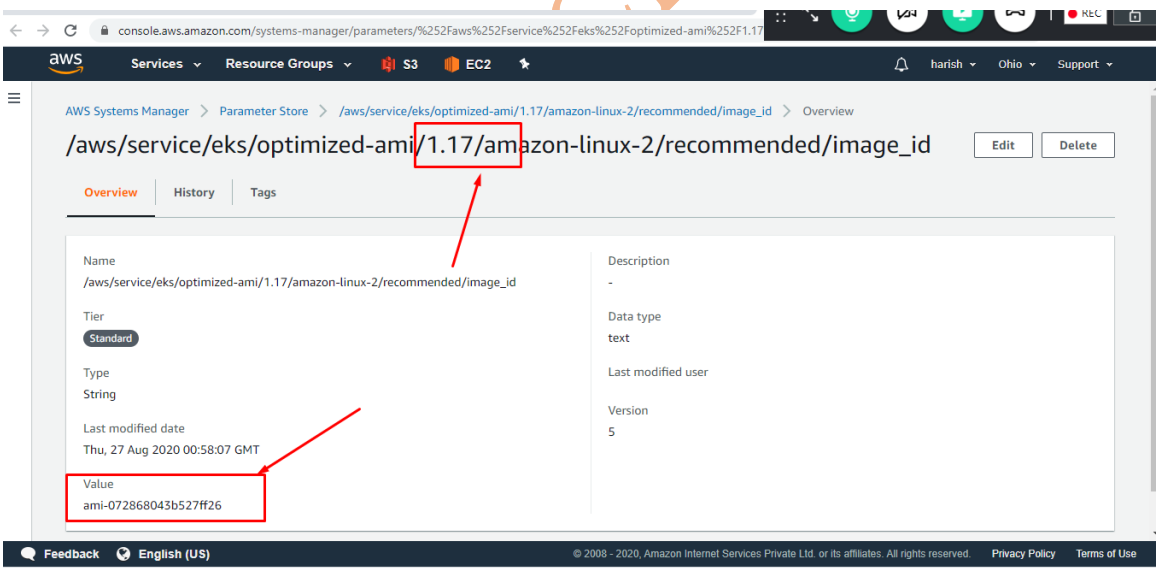




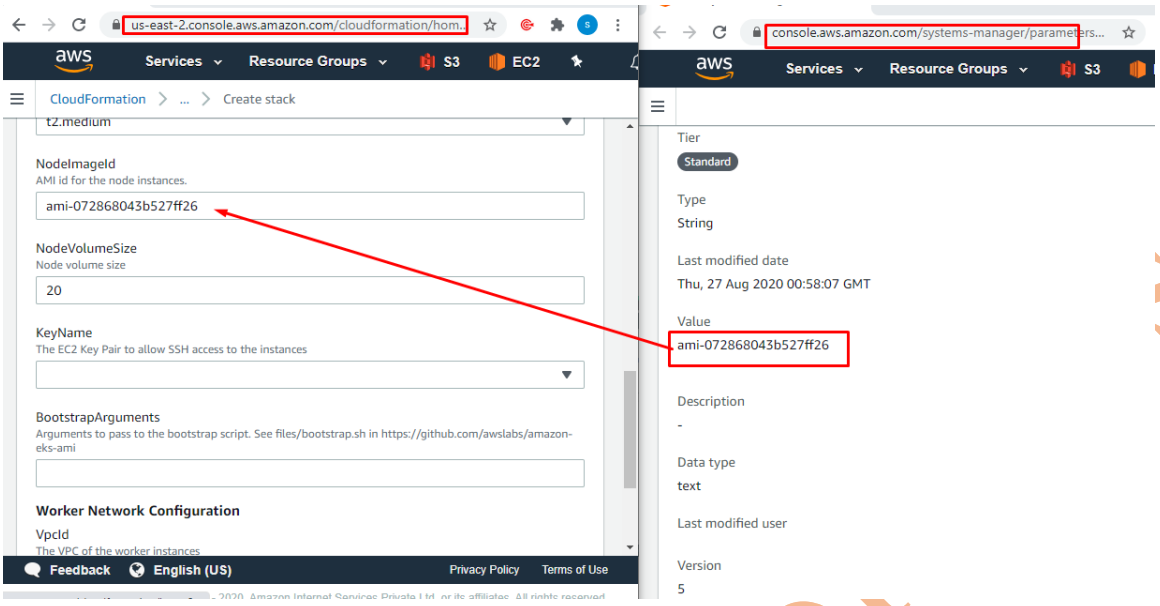
Follow the below site to get the right AMI for the workernode configuration:

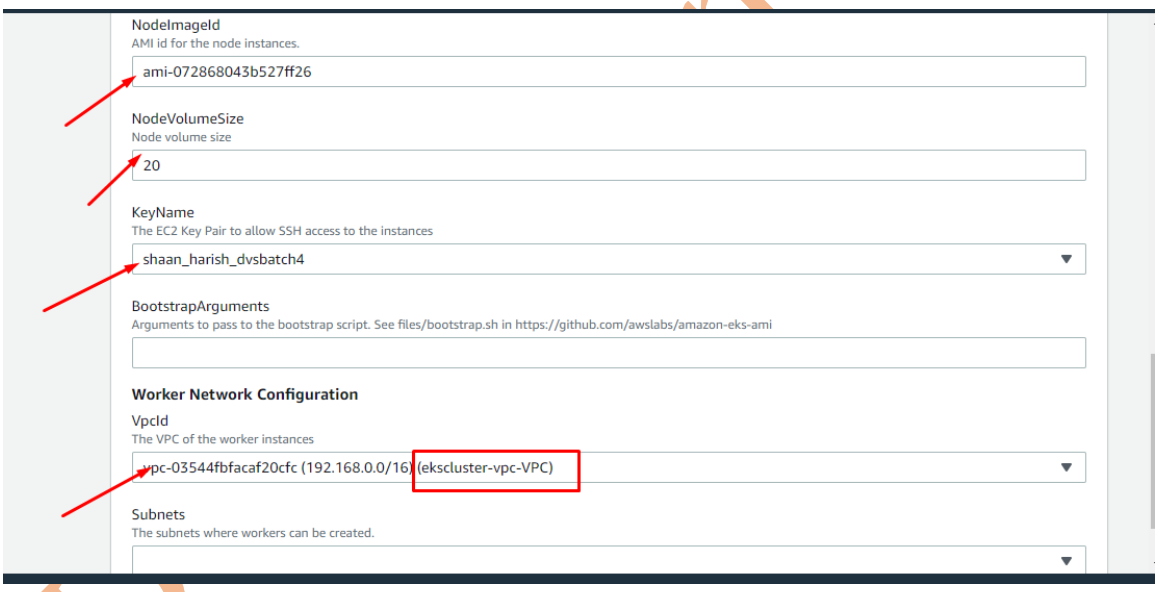


NOTE: Since I am using OHIO opting for this ami, please select the ami id as per your selected region

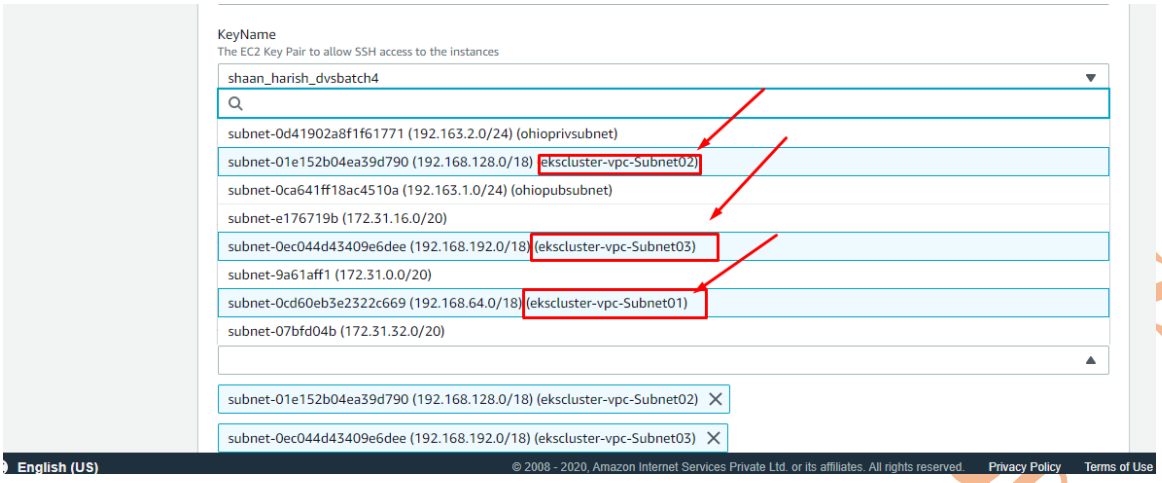


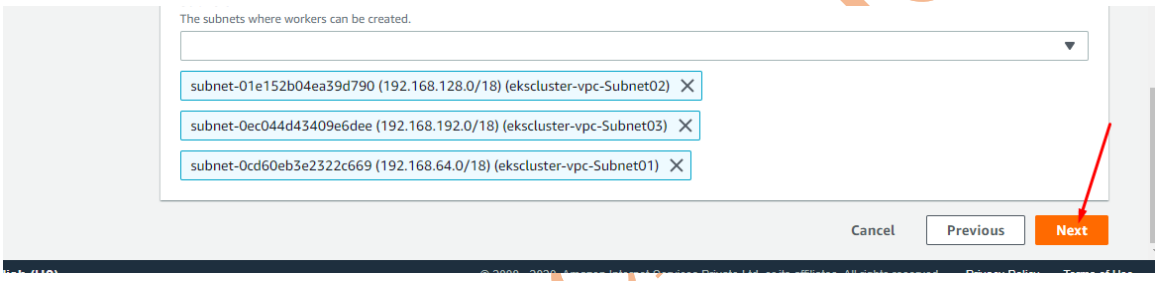
DVS Technologies Aws & Devops

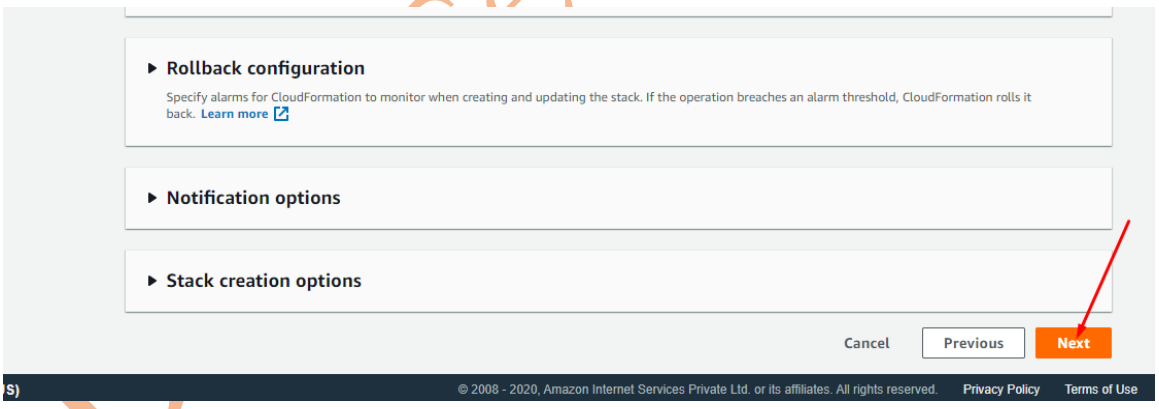


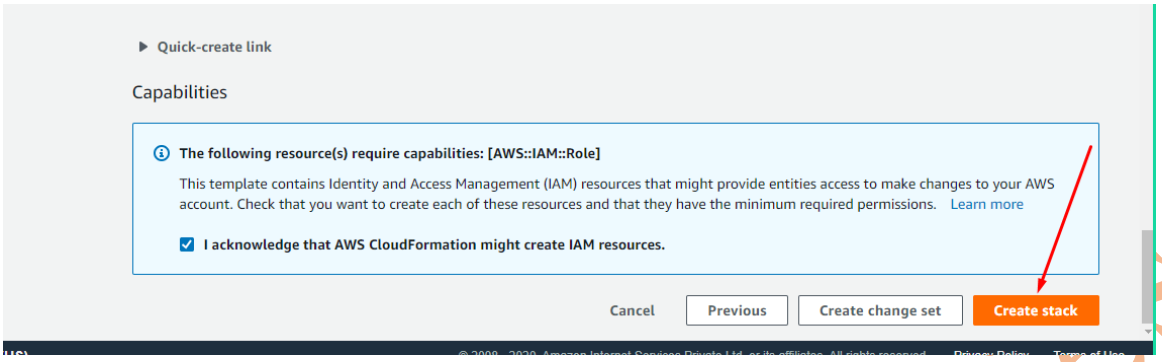


DVS Technologies Aws & Devops

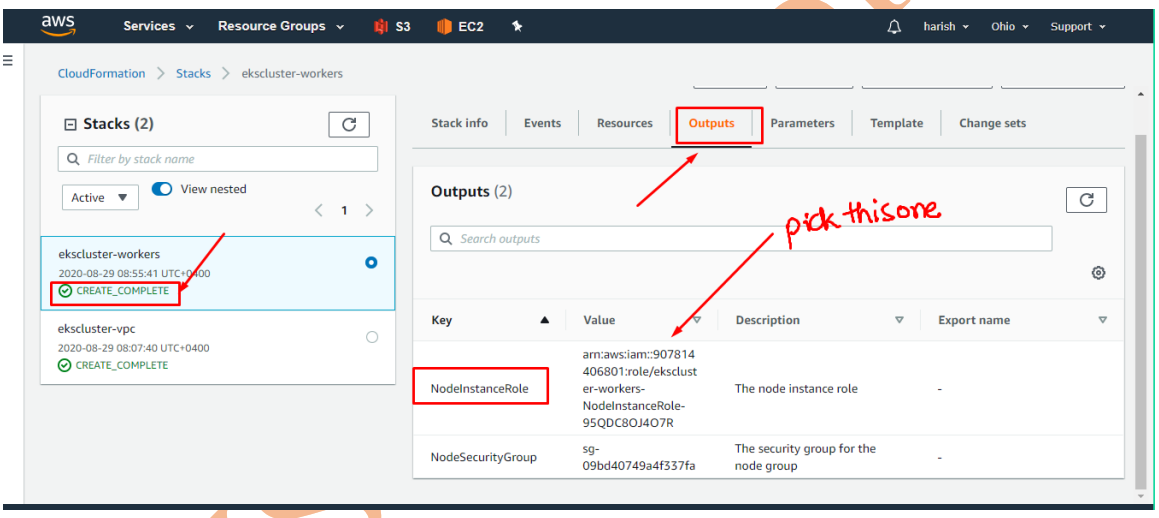








Once stack is done, make sure that you are copying the node ARN like below.
arn:aws:iam::907814406801:role/ekscluster-workers-NodeInstanceRole-95QDC8OJ4O7R

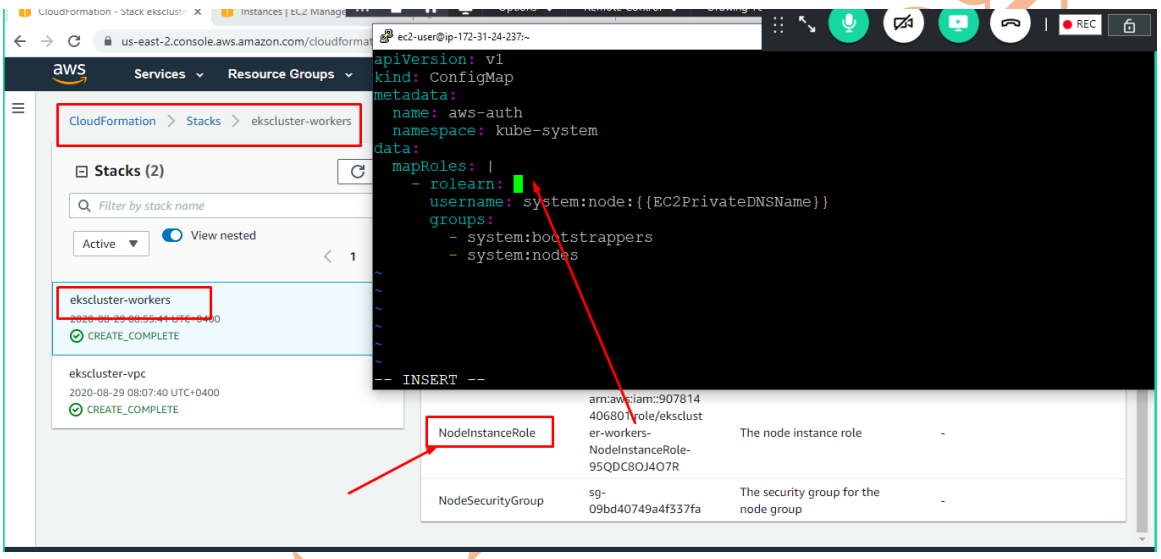


Now perform the below in our workstation server

curl -O https://amazon-eks.s3-us-west-2.amazonaws.com/cloudformation/2018-11-07/aws-auth-cm.yaml

DVS Technologies Aws & Devops

```
[ec2-user@eksworkstation ~]$ curl -O https://amazon-eks.s3-us-west-2.amazonaws.com/cloudformation/2018-11-07/aws-auth-cm
yamlcurl -O https://amazon-eks.s3-us-west-2.amazonaws.com/cloudformation/2018-11-07/aws-auth-cm.yaml
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   317    0   317    0     0   1035      0 --:--:-- --:--:-- --:--:--  1035
100   282  100   282    0     0   3098      0 --:--:-- --:--:-- --:--:--  3098
[ec2-user@eksworkstation ~]$ ls -l aws-auth-cm.yaml
-rw-rw-r-- 1 ec2-user ec2-user 282 Aug 29 05:01 aws-auth-cm.yaml
[ec2-user@eksworkstation ~]$ cat aws-auth-cm.yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: aws-auth
  namespace: kube-system
data:
  mapRoles: |
    - rolearn: <ARN of instance role (not instance profile)>
      username: system:node:{{EC2PrivateDNSName}}
      groups:
        - system:bootstrappers
        - system:nodes
[ec2-user@eksworkstation ~]$
```




```
[ec2-user@eksworkstation ~]$ cat aws-auth-cm.yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: aws-auth
  namespace: kube-system
data:
  mapRoles: |
    - rolearn: arn:aws:iam::907814406801:role/ekscluster-workers-NodeInstanceRole-95QDC8OJ4O7R
      username: system:node:{{EC2PrivateDNSName}}
      groups:
        - system:bootstrappers
        - system:nodes
[ec2-user@eksworkstation ~]$
```

```
      - system:nodes
[ec2-user@eksworkstation ~]$ kubectl apply -f aws-auth-cm.yaml
configmap/aws-auth created
[ec2-user@eksworkstation ~]$ kubectl get nodes
NAME                                           STATUS     ROLES    AGE   VERSION
ip-192-168-191-192.us-east-2.compute.internal   NotReady   <none>   3s    v1.17.9-eks-4c6976
ip-192-168-205-24.us-east-2.compute.internal    NotReady   <none>   2s    v1.17.9-eks-4c6976
[ec2-user@eksworkstation ~]$
```

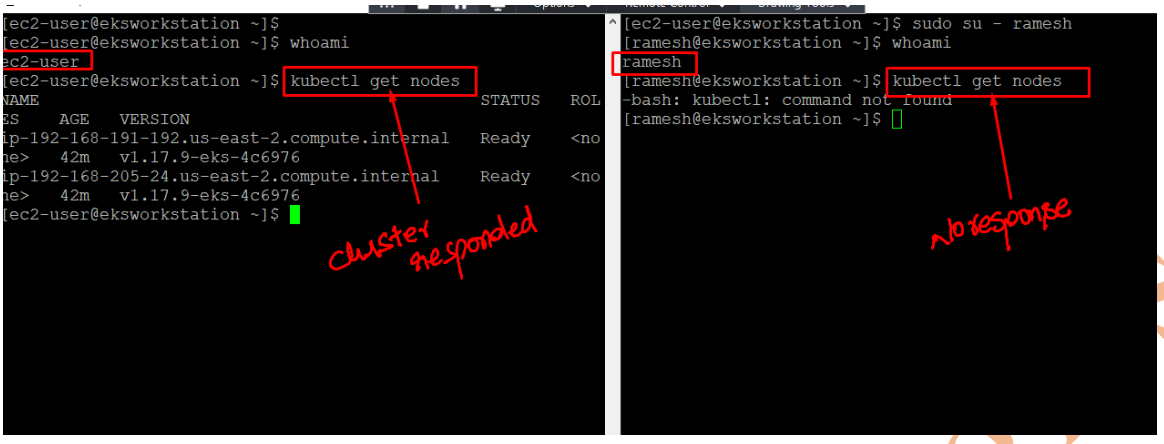
```
^C[ec2-user@eksworkstation ~]$ kubectl get nodes
NAME                                           STATUS   ROLES    AGE   VERSION
ip-192-168-191-192.us-east-2.compute.internal   Ready    <none>   36s   v1.17.9-eks-4c6976
ip-192-168-205-24.us-east-2.compute.internal    Ready    <none>   35s   v1.17.9-eks-4c6976
[ec2-user@eksworkstation ~]$
```

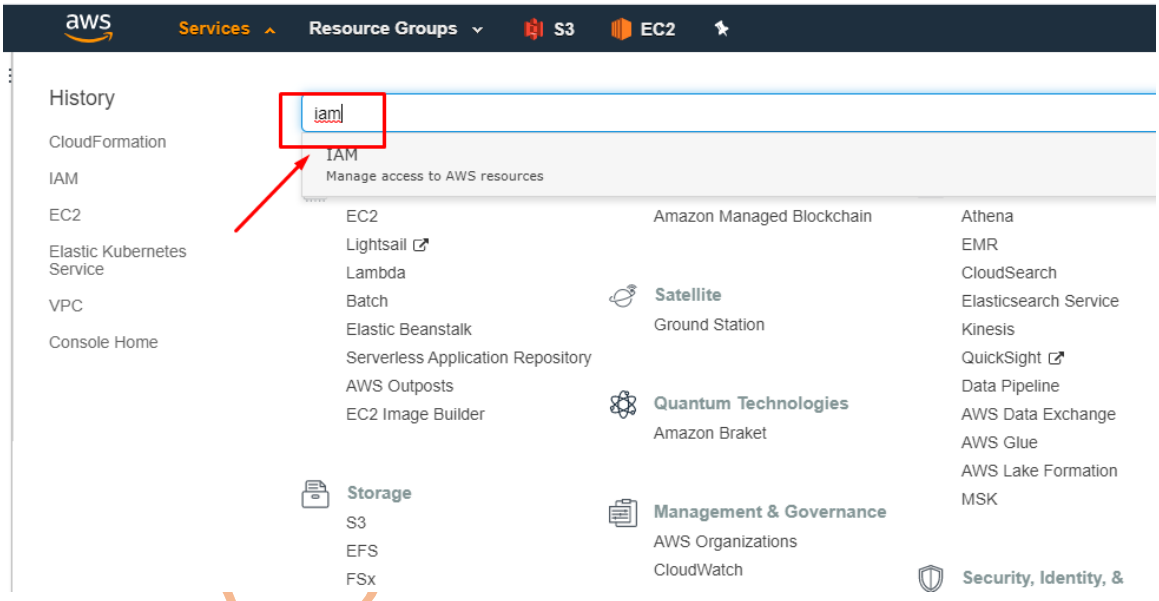
2. IAM Authentication

Let's create a user called "ramesh" & try to perform the cluster acitvities.

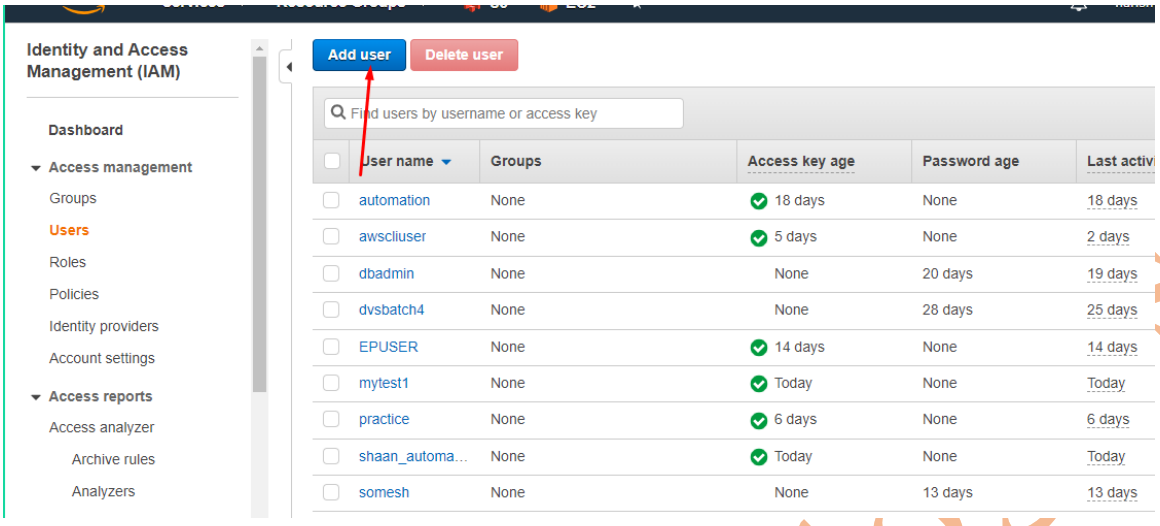
```
[ec2-user@eksworkstation ~]$ id -a ramesh
id: ramesh: no such user
[ec2-user@eksworkstation ~]$ sudo useradd ramesh
[ec2-user@eksworkstation ~]$ id -a ramesh
uid=1001(ramesh) gid=1001(ramesh) groups=1001(ramesh)
[ec2-user@eksworkstation ~]$
```

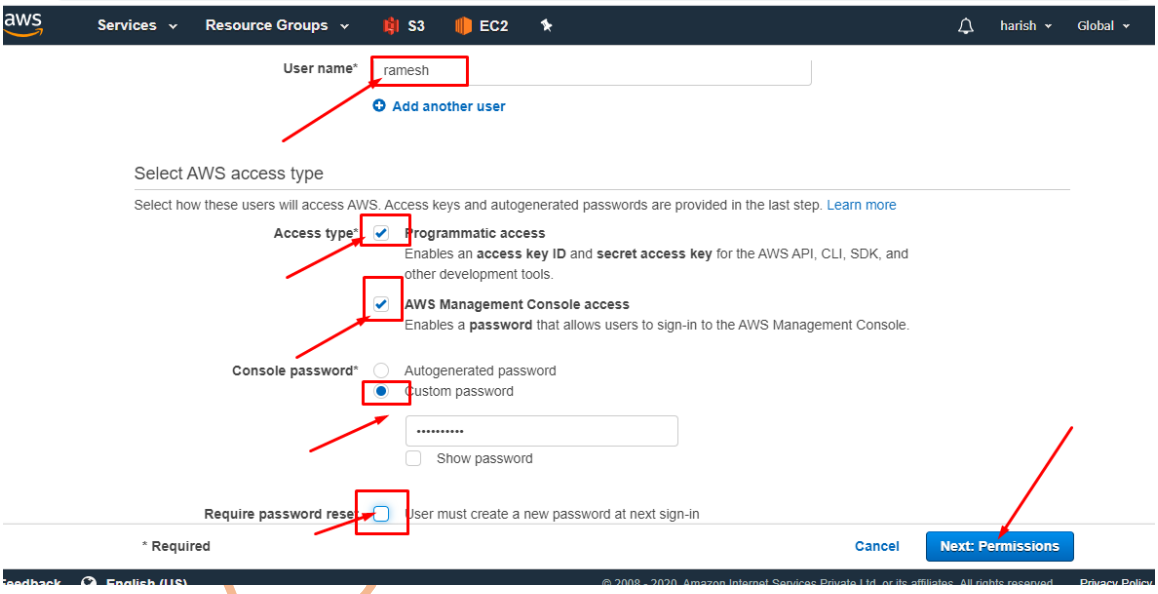
DVS Technologies Aws & Devops

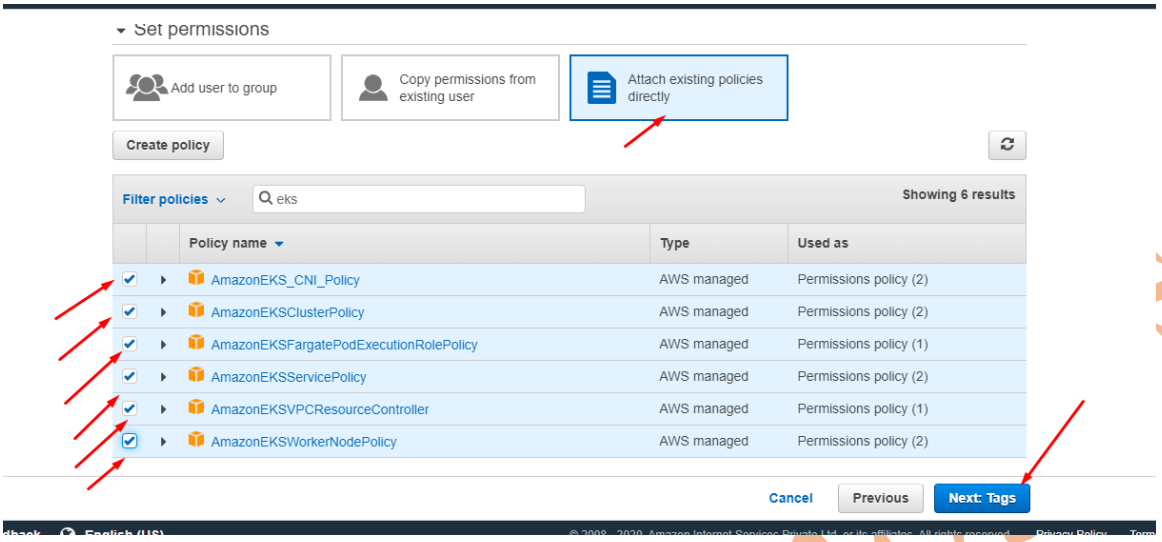


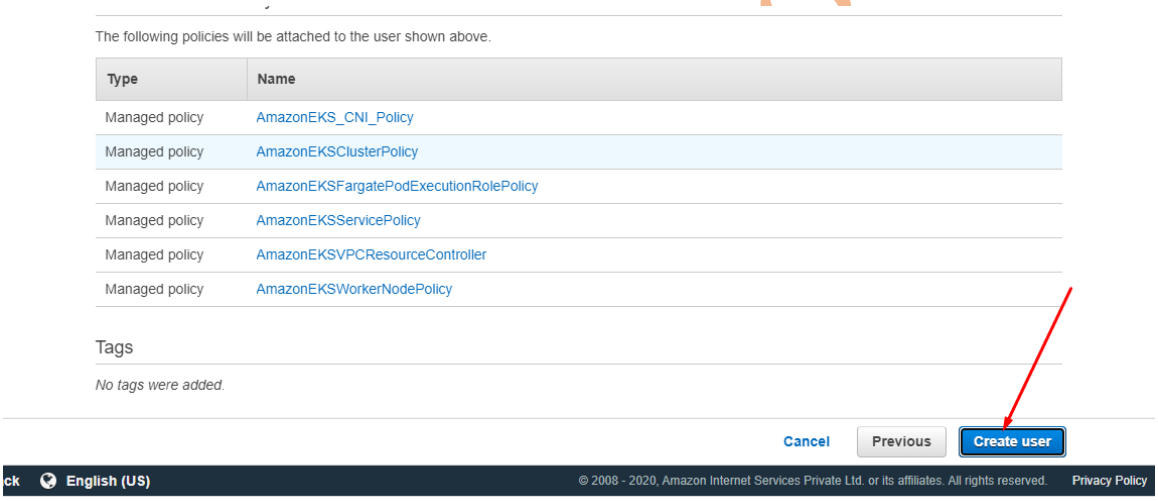


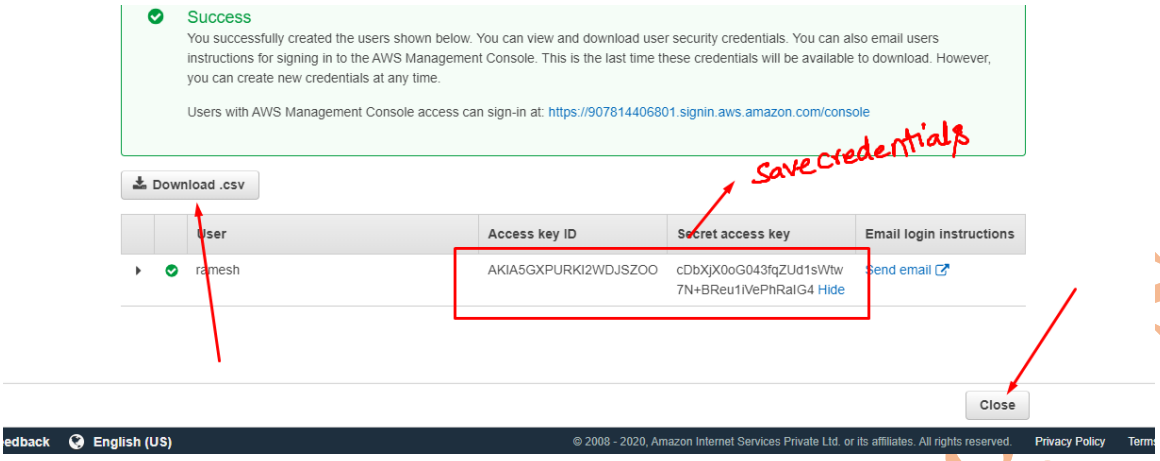
DVS Technologies Aws & Devops






Set permissions
Add user to group
Copy permissions from existing user
Attach existing policies directly
Create policy
Filter policies
eks
Showing 6 results
Policy name
Type
Used as
AmazonEKS_CNI_Policy
AWS managed
Permissions policy (2)
AmazonEKSClusterPolicy
AWS managed
Permissions policy (2)
AmazonEKSFargatePodExecutionRolePolicy
AWS managed
Permissions policy (1)
AmazonEKSServicePolicy
AWS managed
Permissions policy (2)
AmazonEKSVPCResourceController
AWS managed
Permissions policy (1)
AmazonEKSWorkerNodePolicy
AWS managed
Permissions policy (2)
Cancel
Previous
Next: Tags


The following policies will be attached to the user shown above.
Type
Name
Managed policy
AmazonEKS_CNI_Policy
Managed policy
AmazonEKSClusterPolicy
Managed policy
AmazonEKSFargatePodExecutionRolePolicy
Managed policy
AmazonEKSServicePolicy
Managed policy
AmazonEKSVPCResourceController
Managed policy
AmazonEKSWorkerNodePolicy
Tags
No tags were added.
Cancel
Previous
Create user
English (US)
© 2008 - 2020, Amazon Internet Services Private Ltd. or its affiliates. All rights reserved.
Privacy Policy



Execute the below commands as ec2-user::

```
[ec2-user@eksworkstation ~]$ sudo cp -r ./bin/ /home/ramesh/ -f
[ec2-user@eksworkstation ~]$ sudo cp .local /home/ramesh/ -rf
[ec2-user@eksworkstation ~]$ sudo cp ~/.bashrc /home/ramesh/ -rf
[ec2-user@eksworkstation ~]$
```

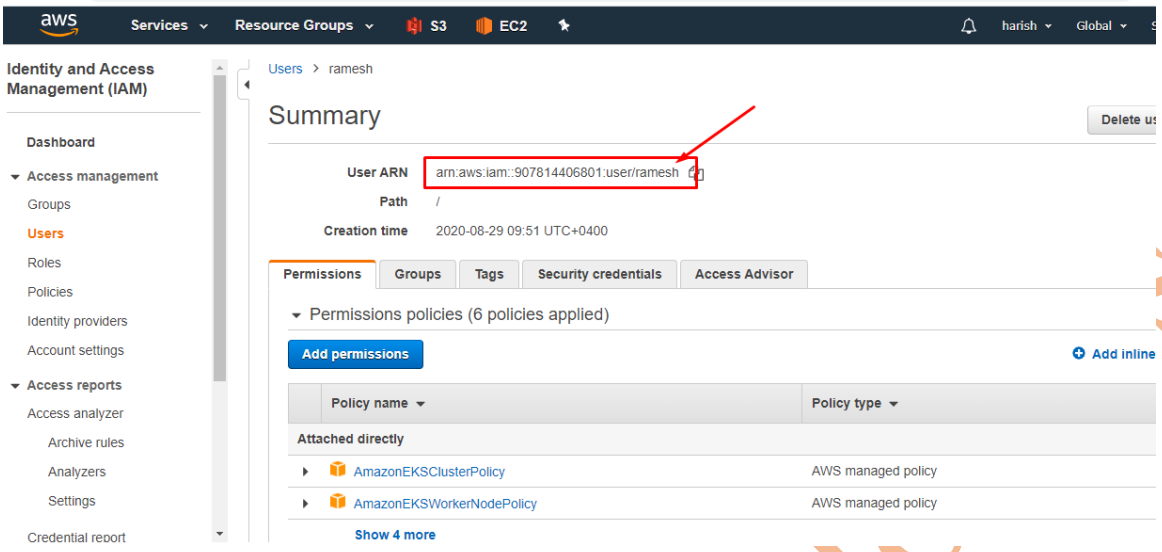
[[As Ramesh User]]

```
[ramesh@eksworkstation ~]$ aws configure
AWS Access Key ID [None]: AKIA5GXPURKI2WDJSZOO
AWS Secret Access Key [None]: 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
Default region name [None]: us-east-2
Default output format [None]: json
[ramesh@eksworkstation ~]$ kubectl version --short
Client Version: v1.17.9-eks-4c6976
The connection to the server localhost:8080 was refused - did you specify the right host or port?
[ramesh@eksworkstation ~]$
```

[[As ec2-user or root User]]

perform the below as ec2-user or root user

DVS Technologies Aws & Devops




```
[ec2-user@eksworkstation ~]$ cat aws-auth-cm.yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: aws-auth
  namespace: kube-system
data:
  mapRoles: |
    - rolearn: arn:aws:iam::907814406801:role/ekscluster-workers-NodeInstanceRole-95QDC8OJ4O7R
      username: system:node:{{EC2PrivateDNSName}}
      groups:
        - system:bootstrappers
        - system:nodes
```

Before adding user info

```
        - system:nodes
[ec2-user@eksworkstation ~]$ vi aws-auth-cm.yaml
[ec2-user@eksworkstation ~]$ cat aws-auth-cm.yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: aws-auth
  namespace: kube-system
data:
  mapRoles: |
    - rolearn: arn:aws:iam::907814406801:role/ekscluster-workers-NodeInstanceRole-95QDC8OJ4O7R
      username: system:node:{{EC2PrivateDNSName}}
      groups:
        - system:bootstrappers
        - system:nodes
  mapUsers: |
    - userarn: arn:aws:iam::907814406801:user/ramesh
      username: ramesh
      groups:
        - system:masters
[ec2-user@eksworkstation ~]$
```

After adding user info

[ec2-user@eksworkstation ~]$ cat aws-auth-cm.yaml
apiVersion: v1

```
kind: ConfigMap
metadata:
  name: aws-auth
  namespace: kube-system
data:
  mapRoles: |
    - rolearn: arn:aws:iam::907814406801:role/ekscluster-workers-NodeInstanceRole-95QDC8OJ4O7R
      username: system:node:{{EC2PrivateDNSName}}
      groups:
        - system:bootstrappers
        - system:nodes
  mapUsers: |
    - userarn: arn:aws:iam::907814406801:user/ramesh
      username: ramesh
      groups:
        - system:masters
```

```
[ec2-user@eksworkstation ~]$ kubectl apply -f aws-auth-cm.yaml
configmap/aws-auth configured
[ec2-user@eksworkstation ~]$
```

[[As ramesh User]]

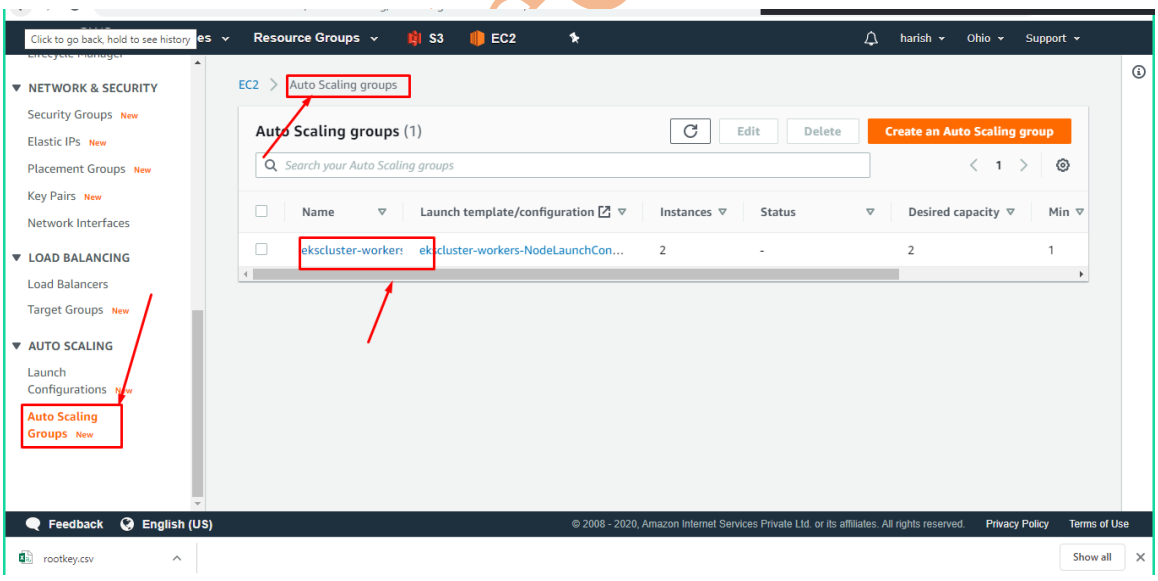
```
[ramesh@eksworkstation ~]$ whoami
ramesh
[ramesh@eksworkstation ~]$ aws eks update-kubeconfig --name ekscluster
Added new context arn:aws:eks:us-east-2:907814406801:cluster/ekscluster to /home/ramesh/.kube/config
[ramesh@eksworkstation ~]$ kubectl get nodes
NAME                                           STATUS   ROLES    AGE   VERSION
ip-192-168-191-192.us-east-2.compute.internal   Ready    <none>   55m   v1.17.9-eks-4c6976
ip-192-168-205-24.us-east-2.compute.internal    Ready    <none>   55m   v1.17.9-eks-4c6976
[ramesh@eksworkstation ~]$
```

3. Configuring Autoscalling

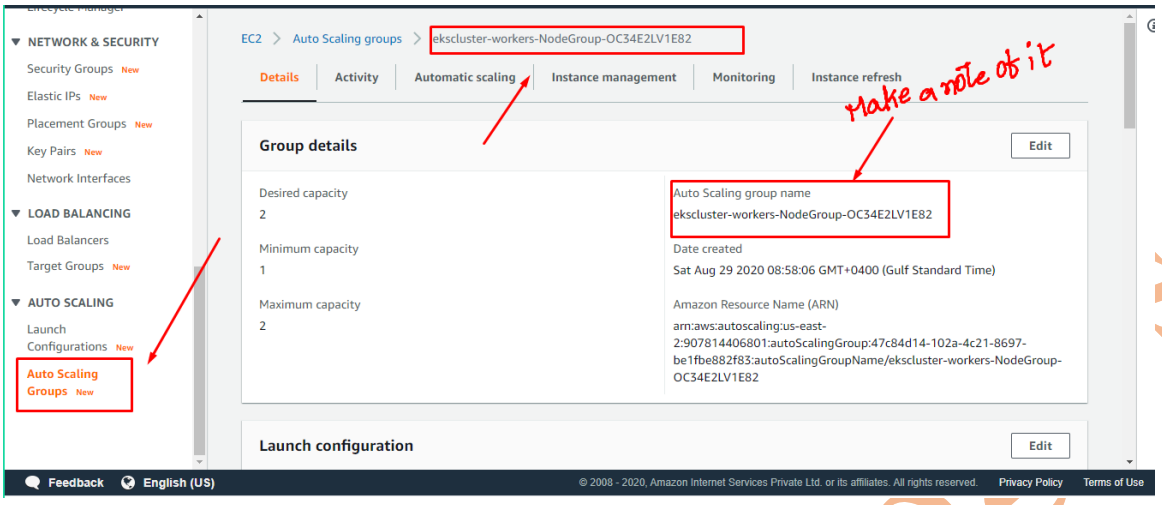
```
[ec2-user@eksworkstation ~]$
[ec2-user@eksworkstation ~]$ kubectl create deploy myapp --image=nginx
deployment.apps/myapp created
[ec2-user@eksworkstation ~]$ kubectl scalce deploy myapp --replicas=3
```

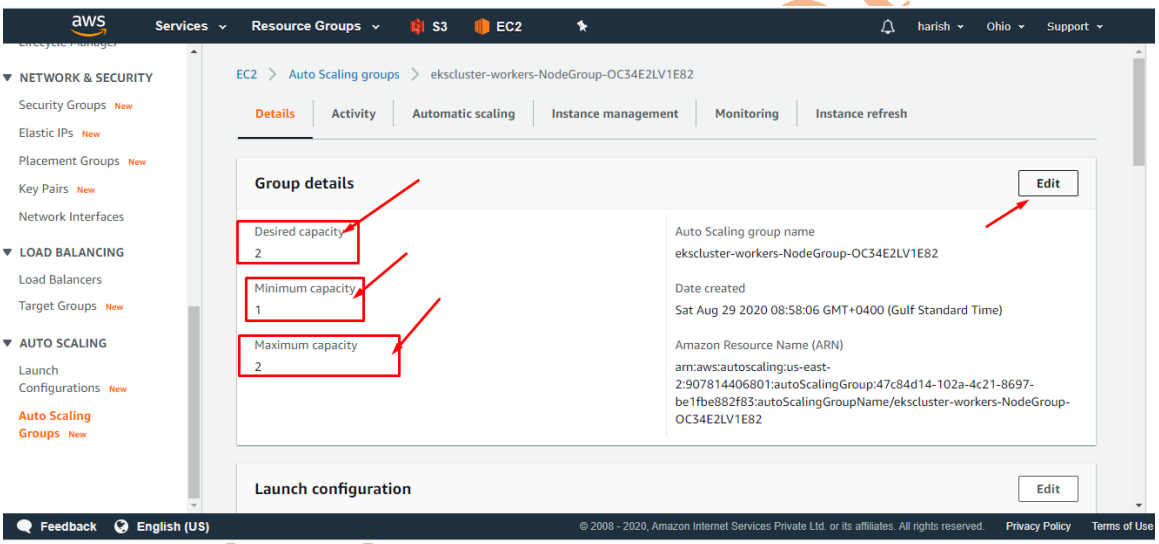
```
[ec2-user@eksworkstation ~]$ kubectl scale deploy myapp --replicas=3
deployment.apps/myapp scaled
[ec2-user@eksworkstation ~]$ kubectl expose deploy myapp --port 80 --type LoadBalancer
service/myapp exposed
[ec2-user@eksworkstation ~]$ kubectl get all -l app=myapp
NAME                         READY   STATUS    RESTARTS   AGE
pod/myapp-57c9b8fc4-74vds    1/1     Running   0          32s
pod/myapp-57c9b8fc4-7spdb    1/1     Running   0          32s
pod/myapp-57c9b8fc4-hfl9k    1/1     Running   0          50s

NAME            TYPE           CLUSTER-IP      EXTERNAL-IP                                                              P
ORT(S)        AGE
service/myapp   LoadBalancer   10.100.250.89   a403534ac5d4b44d486b923b9d6bfa97-1634315458.us-east-2.elb.amazonaws.com   8
0:30833/TCP   9s

NAME                    READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/myapp   3/3     3            3           50s

NAME                               DESIRED   CURRENT   READY   AGE
replicaset.apps/myapp-57c9b8fc4    3         3         3       50s
[ec2-user@eksworkstation ~]$
```

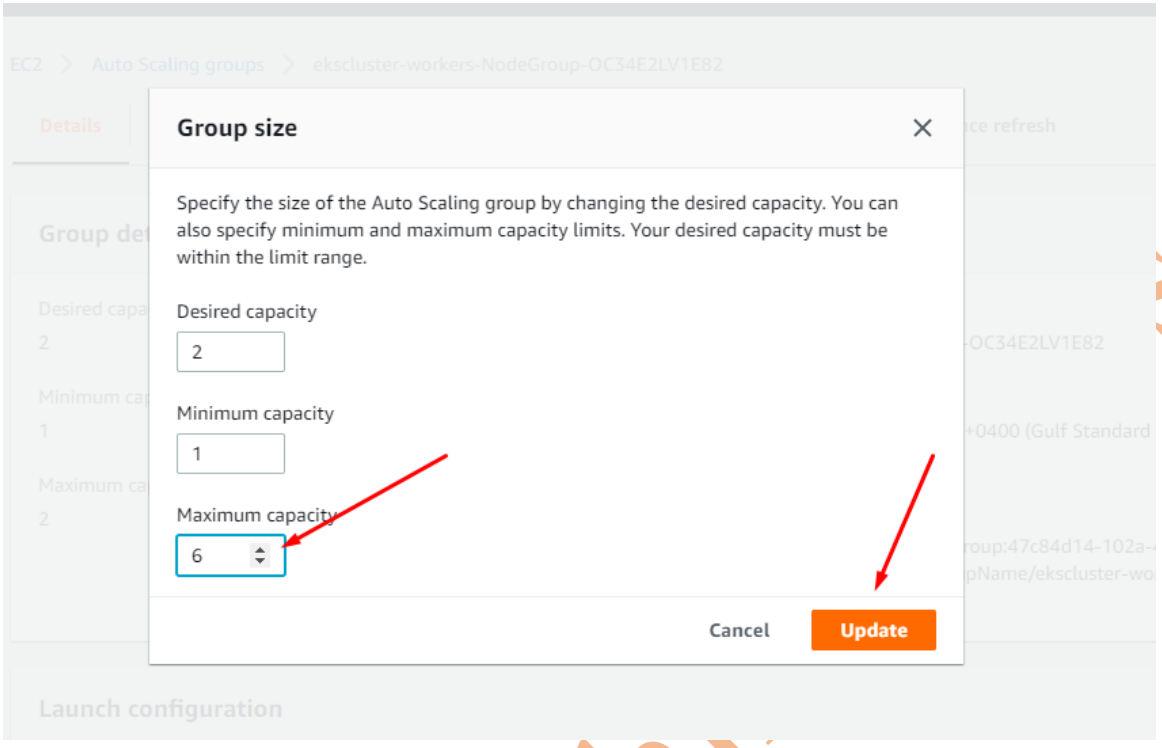
Whenever there is a huge load on my pod in the cluster how my cluster's worker nodes get expanded automatically. In order to achieve this do the below.

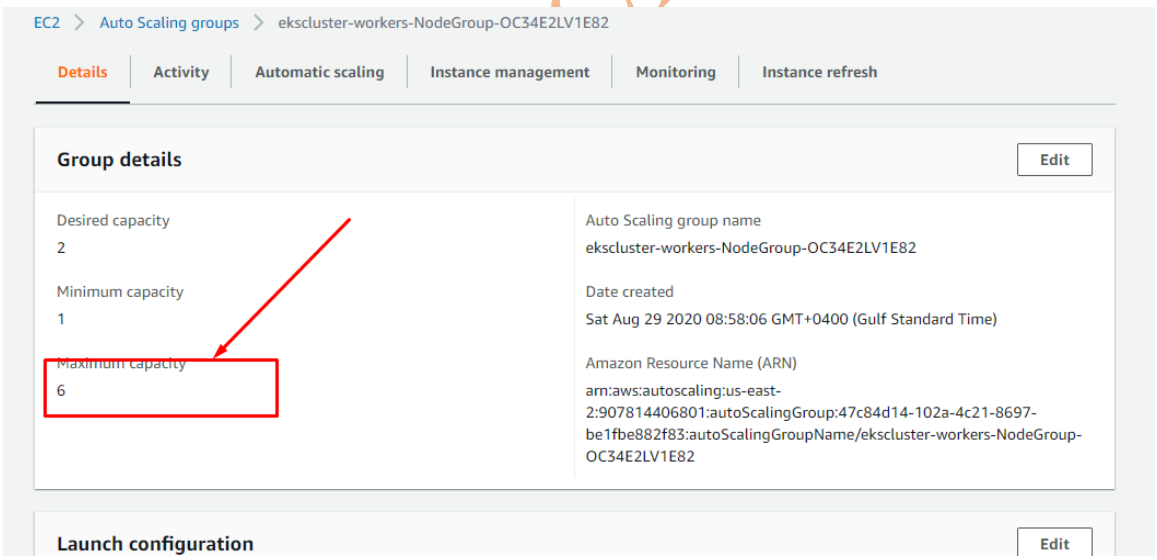


DVS Technologies Aws & Devops



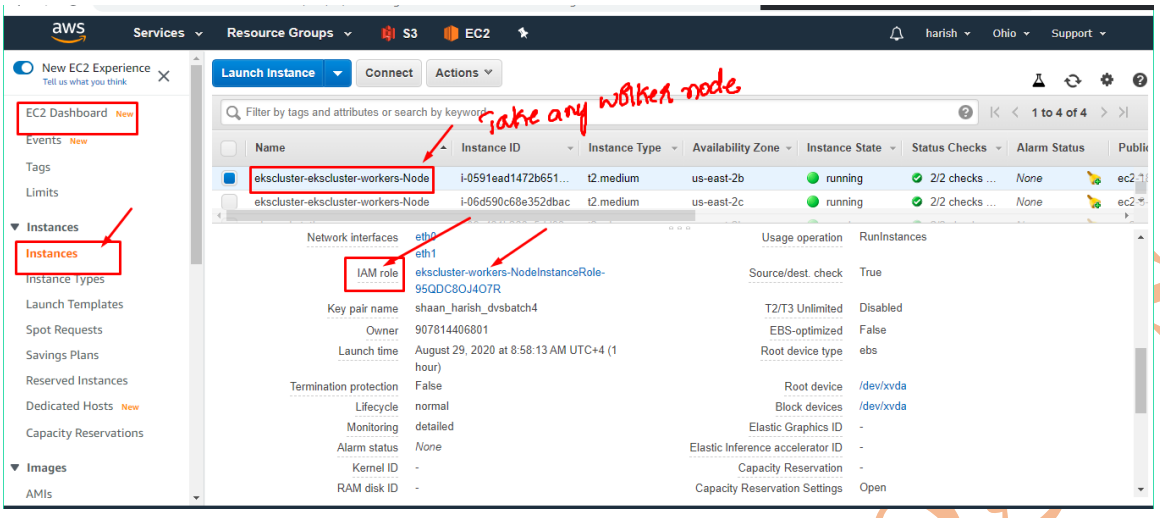


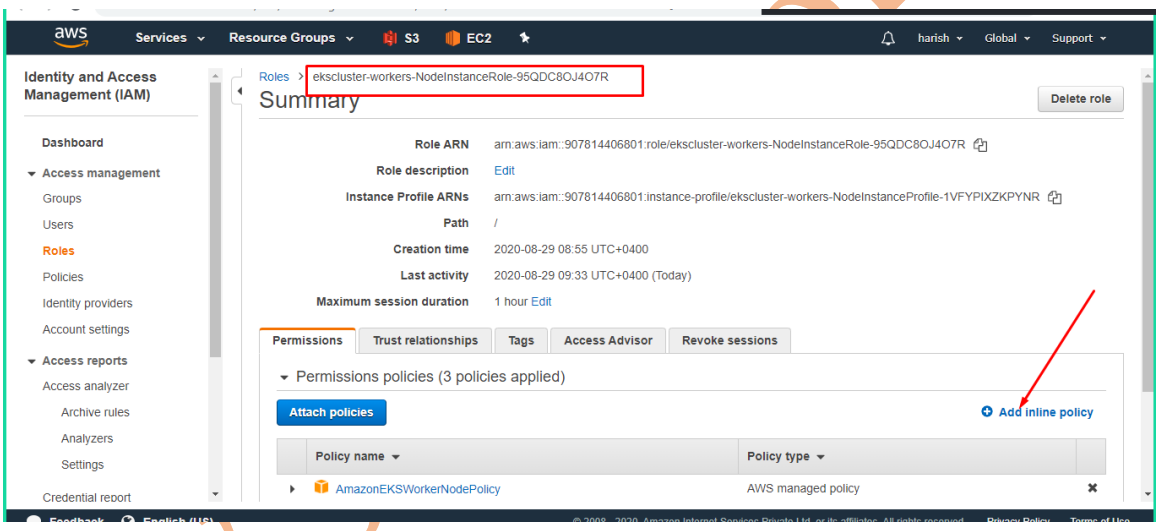


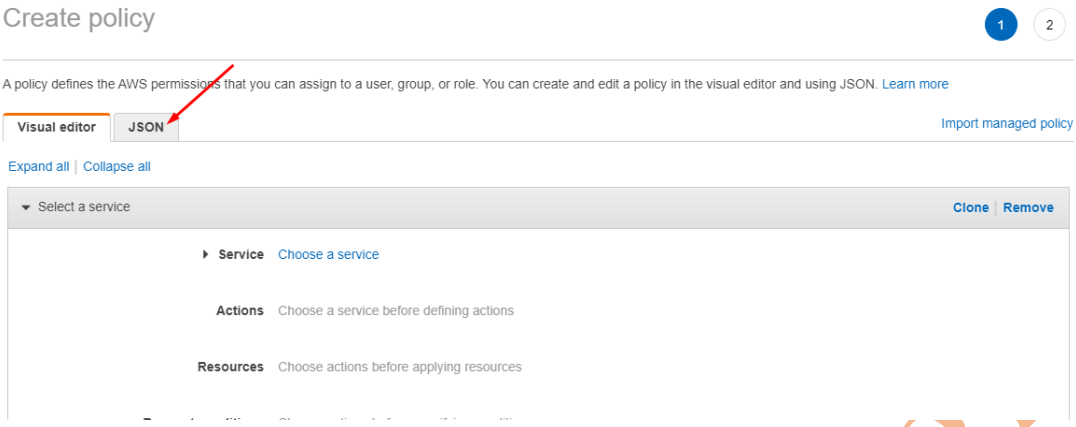


We need to create one inline policy for our role like below.

DVS Technologies Aws & Devops

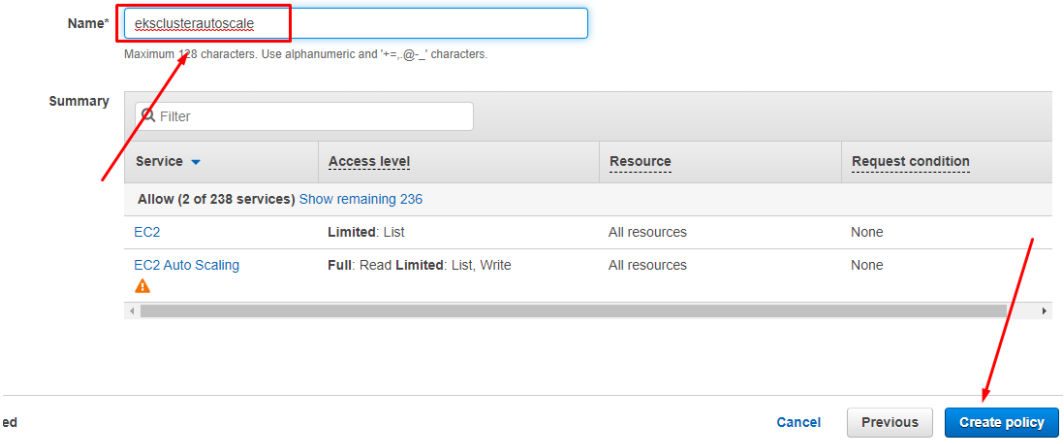


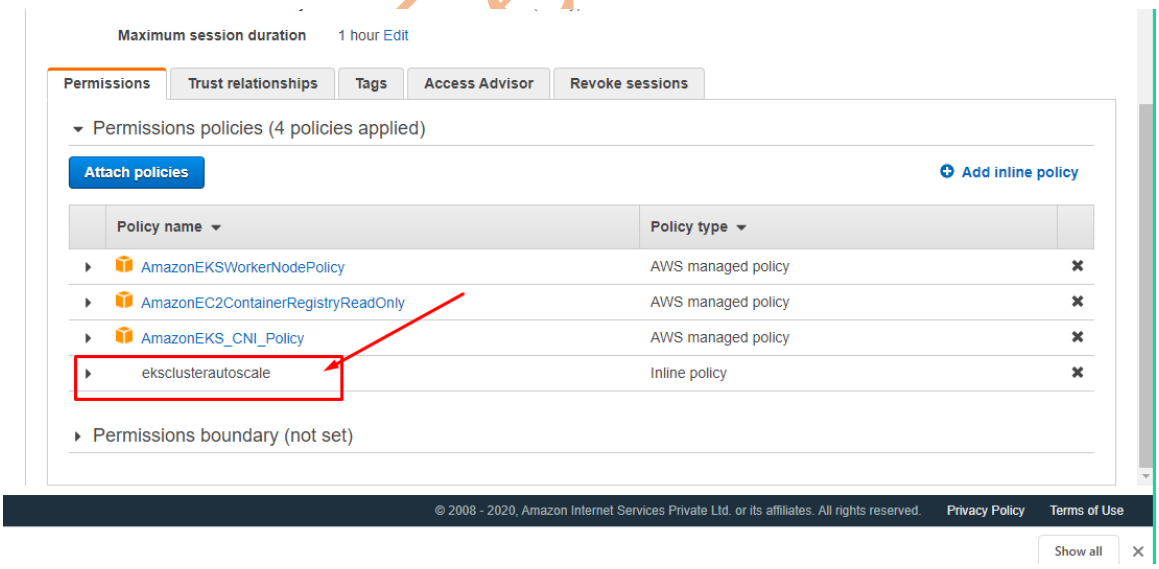




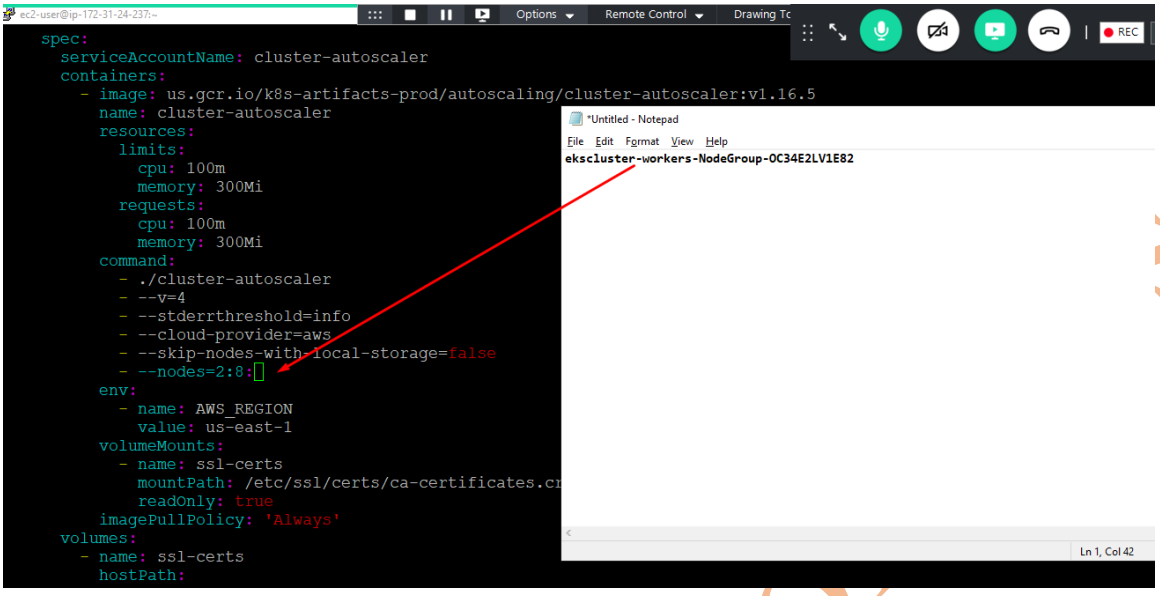

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Allow",
         "Action": [
            "autoscaling:DescribeAutoScalingGroups",
            "autoscaling:DescribeAutoScalingInstances",
            "autoscaling:DescribeInstances",
            "autoscaling:DescribeLaunchConfigurations",
            "autoscaling:DescribeTags",
            "autoscaling:SetDesiredCapacity",
            "autoscaling:TerminateInstanceInAutoScalingGroup",
            "ec2:DescribeLaunchTemplateVersions",
```

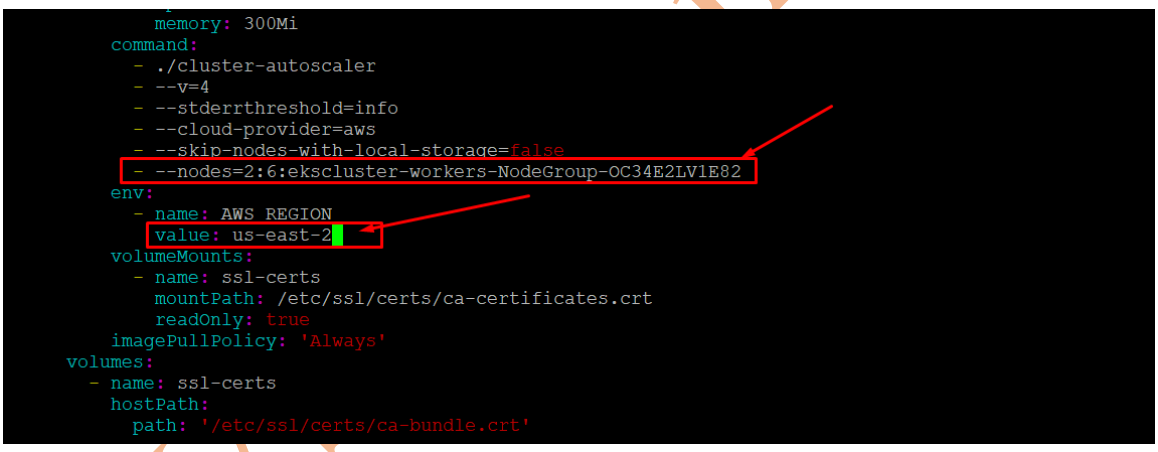
```
            "ec2:DescribeInstanceTypes"
        ],
        "Resource": "*"
    }
  ]
}
```





Now lets go to our workstation machine & do the below.





[ec2-user@eksworkstation ~]$ cat autoscaleconfiguration.yaml
apiVersion: v1
kind: ServiceAccount
metadata:
labels:
k8s-addon: cluster-autoscaler.addons.k8s.io
k8s-app: cluster-autoscaler
name: cluster-autoscaler
namespace: kube-system

```
apiVersion: rbac.authorization.k8s.io/v1beta1
kind: ClusterRole
metadata:
  name: cluster-autoscaler
  labels:
    k8s-addon: cluster-autoscaler.addons.k8s.io
    k8s-app: cluster-autoscaler
rules:
  - apiGroups: ['']
    resources: ['events', 'endpoints']
    verbs: ['create', 'patch']
  - apiGroups: ['']
    resources: ['pods/eviction']
    verbs: ['create']
  - apiGroups: ['']
    resources: ['pods/status']
    verbs: ['update']
  - apiGroups: ['']
    resources: ['endpoints']
    resourceNames: ['cluster-autoscaler']
    verbs: ['get', 'update']
  - apiGroups: ['']
    resources: ['nodes']
    verbs: ['watch', 'list', 'get', 'update']
  - apiGroups: ['']
    resources:
      [
        'pods',
        'services',
        'replicationcontrollers',
        'persistentvolumeclaims',
        'persistentvolumes',
      ]
    verbs: ['watch', 'list', 'get']
  - apiGroups: ['extensions']
    resources: ['replicasets', 'daemonsets']
    verbs: ['watch', 'list', 'get']
  - apiGroups: ['policy']
    resources: ['poddisruptionbudgets']
    verbs: ['watch', 'list']
  - apiGroups: ['apps']
    resources: ['statefulsets', 'daemonsets', 'apps', 'replicasets']
    verbs: ['watch', 'list', 'get']
  - apiGroups: ['storage.k8s.io']
```

```
    resources: ['storageclasses', 'csinodes']
    verbs: ['watch', 'list', 'get']
  - apiGroups: ['batch', 'extensions']
    resources: ['jobs']
    verbs: ['get', 'list', 'watch', 'patch']

---
apiVersion: rbac.authorization.k8s.io/v1beta1
kind: Role
metadata:
  name: cluster-autoscaler
  namespace: kube-system
  labels:
    k8s-addon: cluster-autoscaler.addons.k8s.io
    k8s-app: cluster-autoscaler
rules:
  - apiGroups: ['']
    resources: ['configmaps']
    verbs: ['create']
  - apiGroups: ['']
    resources: ['configmaps']
    resourceNames: ['cluster-autoscaler-status']
    verbs: ['delete', 'get', 'update']

---
apiVersion: rbac.authorization.k8s.io/v1beta1
kind: ClusterRoleBinding
metadata:
  name: cluster-autoscaler
  labels:
    k8s-addon: cluster-autoscaler.addons.k8s.io
    k8s-app: cluster-autoscaler
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: cluster-autoscaler
subjects:
  - kind: ServiceAccount
    name: cluster-autoscaler
    namespace: kube-system

---
apiVersion: rbac.authorization.k8s.io/v1beta1
kind: RoleBinding
```

```
metadata:
  name: cluster-autoscaler
  namespace: kube-system
  labels:
    k8s-addon: cluster-autoscaler.addons.k8s.io
    k8s-app: cluster-autoscaler
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: Role
  name: cluster-autoscaler
subjects:
  - kind: ServiceAccount
    name: cluster-autoscaler
    namespace: kube-system

---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: cluster-autoscaler
  namespace: kube-system
  labels:
    app: cluster-autoscaler
spec:
  replicas: 1
  selector:
    matchLabels:
      app: cluster-autoscaler
  template:
    metadata:
      labels:
        app: cluster-autoscaler
    spec:
      serviceAccountName: cluster-autoscaler
      containers:
        - image: us.gcr.io/k8s-artifacts-prod/autoscaling/cluster-autoscaler:v1.16.5
          name: cluster-autoscaler
          resources:
            limits:
              cpu: 100m
              memory: 300Mi
            requests:
              cpu: 100m
              memory: 300Mi
```

```
      command:
        - ./cluster-autoscaler
        - --v=4
        - --stderrthreshold=info
        - --cloud-provider=aws
        - --skip-nodes-with-local-storage=false
        - --nodes=2:6:ekscluster-workers-NodeGroup-OC34E2LV1E82
      env:
        - name: AWS_REGION
          value: us-east-2
      volumeMounts:
        - name: ssl-certs
          mountPath: /etc/ssl/certs/ca-certificates.crt
          readOnly: true
      imagePullPolicy: 'Always'
    volumes:
    - name: ssl-certs
      hostPath:
        path: '/etc/ssl/certs/ca-bundle.crt'
```

[ec2-user@eksworkstation ~]$

```
[ec2-user@eksworkstation ~]$
[ec2-user@eksworkstation ~]$ kubectl apply -f autoscaleconfiguration.yaml
serviceaccount/cluster-autoscaler created
clusterrole.rbac.authorization.k8s.io/cluster-autoscaler created
role.rbac.authorization.k8s.io/cluster-autoscaler created
clusterrolebinding.rbac.authorization.k8s.io/cluster-autoscaler created
rolebinding.rbac.authorization.k8s.io/cluster-autoscaler created
deployment.apps/cluster-autoscaler created
[ec2-user@eksworkstation ~]$
```

Testing the cluster Autoscalling:

```
[ec2-user@eksworkstation ~]$ kubectl get pods -l app=myapp
NAME                     READY   STATUS    RESTARTS   AGE
myapp-57c9b8fc4-74vds    1/1     Running   0          15m
myapp-57c9b8fc4-7spdb    1/1     Running   0          15m
myapp-57c9b8fc4-hfl9k    1/1     Running   0          16m
[ec2-user@eksworkstation ~]$ kubectl scale deploy myapp --replicas=40
deployment.apps/myapp scaled
[ec2-user@eksworkstation ~]$ kubectl get deploy
NAME    READY   UP-TO-DATE   AVAILABLE   AGE
myapp   17/40   40           17          16m
[ec2-user@eksworkstation ~]$
```

```
[ec2-user@eksworkstation ~]$ kubectl get nodes
NAME                                          STATUS   ROLES    AGE   VERSION
ip-192-168-191-192.us-east-2.compute.internal   Ready    <none>   77m   v1.17.9-eks-4c6976
ip-192-168-205-24.us-east-2.compute.internal    Ready    <none>   77m   v1.17.9-eks-4c6976
[ec2-user@eksworkstation ~]$
```

Wait for some time then check for your worker nodes

Finally :

```
[ec2-user@eksworkstation ~]$ kubectl get nodes
NAME                                          STATUS   ROLES    AGE   VERSION
ip-192-168-191-192.us-east-2.compute.internal   Ready    <none>   78m   v1.17.9-eks-4c6976
ip-192-168-205-24.us-east-2.compute.internal    Ready    <none>   78m   v1.17.9-eks-4c6976
ip-192-168-76-248.us-east-2.compute.internal    Ready    <none>   31s   v1.17.9-eks-4c6976
[ec2-user@eksworkstation ~]$
```

Removing nodes from the autoscalling by reducing the replica count:

```
[ec2-user@eksworkstation ~]$ kubectl scale deploy myapp --replicas=1
deployment.apps/myapp scaled
[ec2-user@eksworkstation ~]$ kubectl get nodes
NAME                                          STATUS   ROLES    AGE   VERSION
ip-192-168-191-192.us-east-2.compute.internal   Ready    <none>   79m   v1.17.9-eks-4c6976
ip-192-168-205-24.us-east-2.compute.internal    Ready    <none>   79m   v1.17.9-eks-4c6976
ip-192-168-76-248.us-east-2.compute.internal    Ready    <none>   59s   v1.17.9-eks-4c6976
[ec2-user@eksworkstation ~]$
```

Please wait for some time so that your third node will be automatically get delete from the cluster

```
[ec2-user@eksworkstation ~]$ kubectl get nodes
NAME                                          STATUS   ROLES    AGE   VERSION
ip-192-168-191-192.us-east-2.compute.internal   Ready    <none>   96m   v1.17.9-eks-4c6976
ip-192-168-76-248.us-east-2.compute.internal    Ready    <none>   18m   v1.17.9-eks-4c6976
[ec2-user@eksworkstation ~]$
```