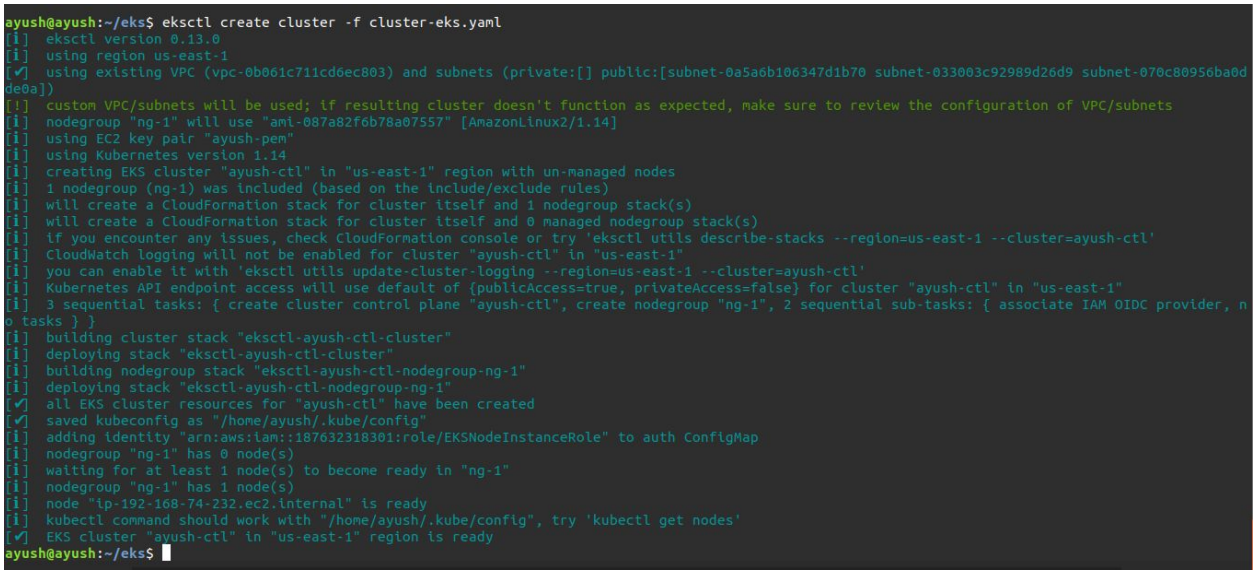
Day 1 Exercise

1. **Create eks cluster using eksctl**
   During creation, Specify
   - **Cluster name**
   - **Kubernetes version**
   - **Control plane role**
   - **Subnets for Control Plane**
   - **Control Plane security Group**
   - **Add tag: owner, purpose on Control Plane**
   - **Node Group Name**
   - **Node Instance Role**
   - **Subnets for Node Group**
   - **Node Instance SSH key pair**
   - **Node Instance Security Group**
   - **Node Instance Instance Type**
   - **Node Instance Disk**
   - **Add tag: owner, purpose on Node Group**
   - **Node Group Size: min, max**



```yaml
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig

metadata:
  name: ayush-ctl
  region: us-east-1

iam:
  withOIDC: true
  serviceRoleARN: "arn:aws:iam::187632318301:role/eksServiceRole"

vpc:
  id: vpc-0b061c711cd6ec803
  #  securityGroups: sg-07f4ba3693fc6a2f8
  cidr: "192.168.0.0/16"
  subnets:
    public:
      us-east-1a:
        id: subnet-0a5a6b106347d1b70
      us-east-1b:
        id: subnet-033003c92989d26d9
      us-east-1c:
        id: subnet-070c80956ba0dde0a
```

```yaml
nodeGroups:
  - name: ng-1
    instanceType: t3.medium
    minSize: 1
    desiredCapacity: 1
    maxSize: 2
    volumeSize: 20
    availabilityZones: ["us-east-1a", "us-east-1b", "us-east-1c"]
    iam:
      instanceProfileARN: "arn:aws:iam::187632318301:instance-profile/EKSNodeInstanceRole"
    securityGroups:
      withShared: true
      withLocal: true
      attachIDs: ['sg-002c8a7b43bacc21c']
    ssh:
      allow: true
      publicKeyName: 'ayush-pem'
    tags:
      'Owner': 'ayush'
```

```
ayush@ayush:~/eks$ kubectl get nodes
NAME                                STATUS   ROLES    AGE   VERSION
ip-192-168-75-227.ec2.internal   Ready    <none>   63s   v1.14.8-eks-b8860f
ayush@ayush:~/eks$ vim namespace.yml
ayush@ayush:~/eks$ kubectl apply -f namespace.yml
namespace/dev created
namespace/qa created
namespace/prod created
namespace/ayush created
ayush@ayush:~/eks$ kubectl apply -f pod.yml -n ayush
pod/nginx created
ayush@ayush:~/eks$ kubectl get pods -n ayush
NAME    READY   STATUS    RESTARTS   AGE
nginx   1/1     Running   0          28s
ayush@ayush:~/eks$
```

2. Authentication Management

a. Add new 2 IAM user into the cluster

```
apiVersion: v1
data:
  mapRoles: |
    - groups:
      - system:bootstrappers
      - system:nodes
      rolearn: arn:aws:iam::187632318301:role/EKSNodeInstanceRole
      username: system:node:{{EC2PrivateDNSName}}
  mapUsers: |
    - userarn: arn:aws:iam::187632318301:user/diksha.tomar@tothenew.com
      username: diksha
      groups:
        - system:master
    - userarn: arn:aws:iam::187632318301:user/yash.khandelwal@tothenew.com
      username: yash
      groups:
        - system:master
kind: ConfigMap
metadata:
  creationTimestamp: "2020-03-05T10:16:27Z"
  name: aws-auth
  namespace: kube-system
  resourceVersion: "2636"
  selfLink: /api/v1/namespaces/kube-system/configmaps/aws-auth
  uid: 60035bd7-5eca-11ea-ba7b-029b3c7efe85
```

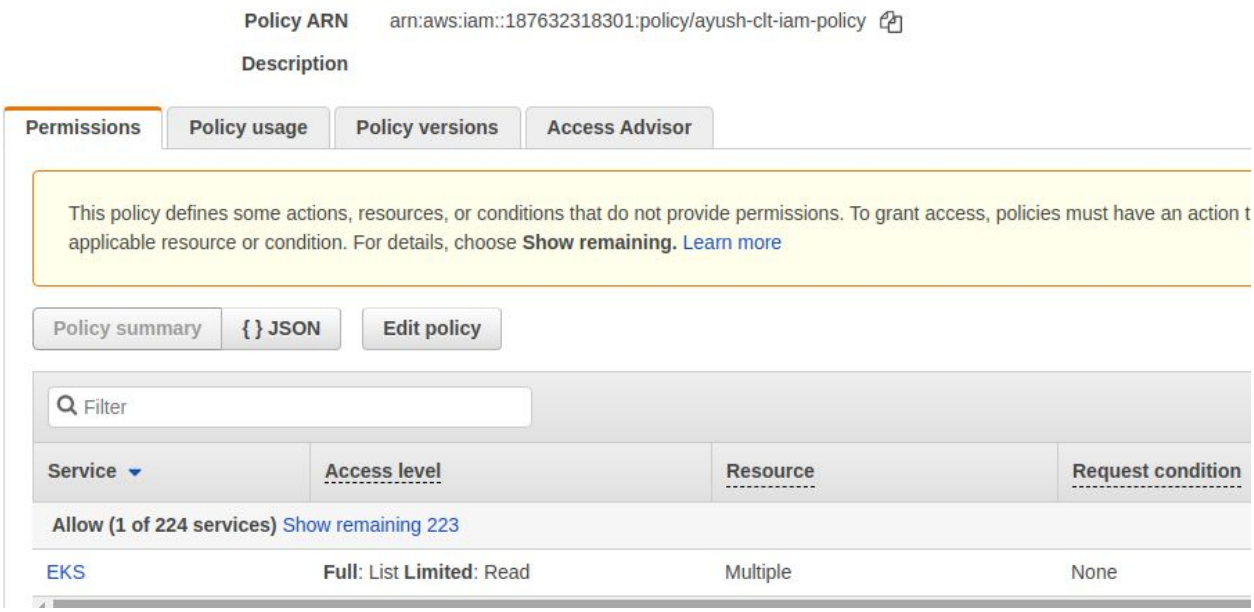
```
diksha@diksha:~$ kubectl get svc
NAME         TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
kubernetes   ClusterIP   10.100.0.1   <none>        443/TCP   83m
diksha@diksha:~$ kubectl get nodes
No resources found.
diksha@diksha:~$ kubectl get nodes
NAME                                STATUS   ROLES    AGE     VERSION
ip-192-168-75-227.ec2.internal   Ready    <none>   3m34s   v1.14.8-eks-b8860f
diksha@diksha:~$ kubectl get pods -n ayush
NAME    READY   STATUS    RESTARTS   AGE
nginx   1/1     Running   0          74s
diksha@diksha:~$
```

```
yash@yash-khandelwal:~$ kubectl get svc
NAME         TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
kubernetes   ClusterIP   10.100.0.1   <none>        443/TCP   96m
yash@yash-khandelwal:~$ kubectl get nodes
NAME                                STATUS   ROLES    AGE     VERSION
ip-192-168-75-227.ec2.internal   Ready    <none>   5m56s   v1.14.8-eks-b8860f
yash@yash-khandelwal:~$ kubectl get pods -n ayush
NAME    READY   STATUS    RESTARTS   AGE
nginx   1/1     Running   0          3m19s
yash@yash-khandelwal:~$
```

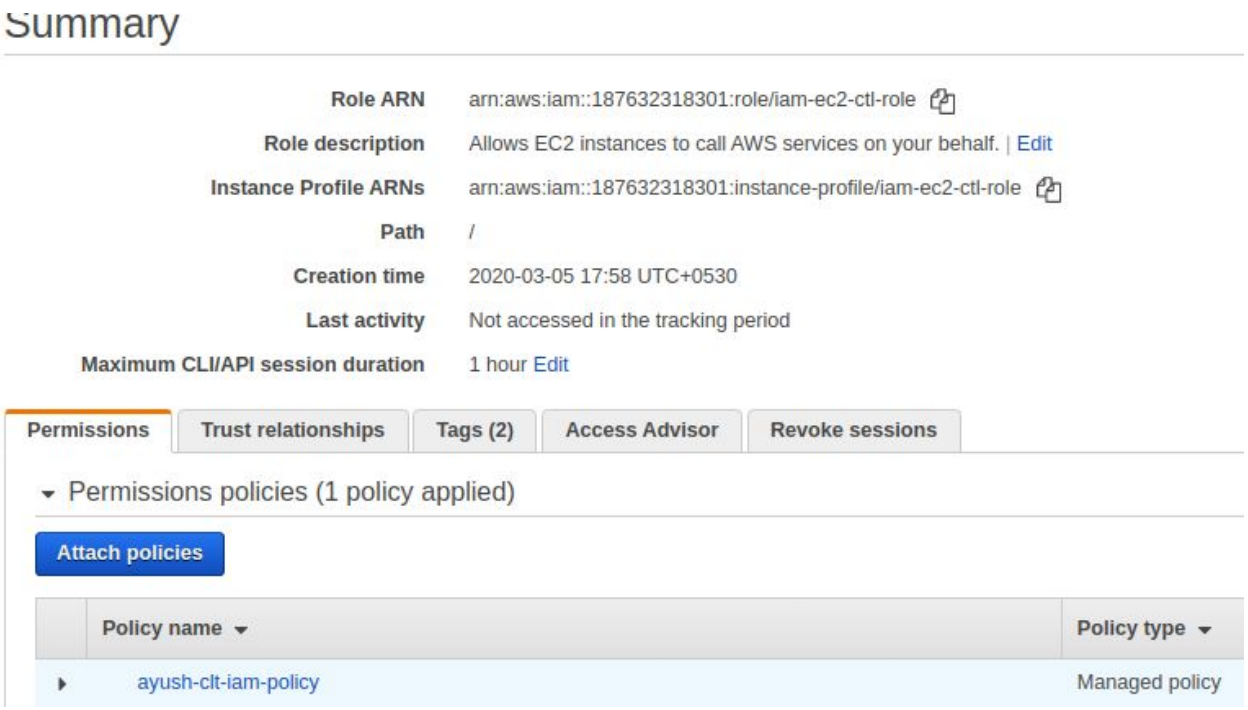
b. Enable a EC2 server to access Cluster master API without using access/secret key

Create policy:

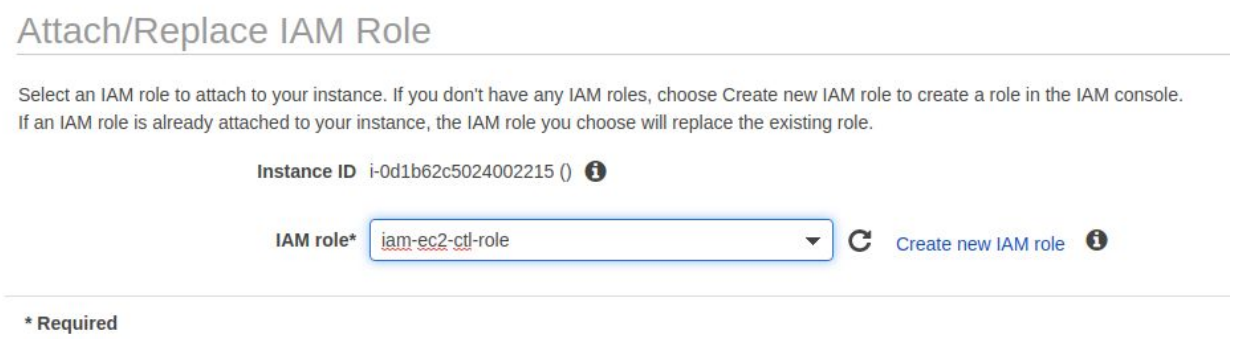
```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "eks:ListFargateProfiles",
                "eks:DescribeNodegroup",
                "eks:ListNodegroups",
                "eks:DescribeFargateProfile",
                "eks:ListTagsForResource",
                "eks:DescribeUpdate",
                "eks:ListUpdates",
                "eks:DescribeCluster"
            ],
            "Resource": "arn:aws:eks:us-east-1:187632318301:cluster/ayush-ctl"
        },
        {
            "Sid": "VisualEditor1",
            "Effect": "Allow",
            "Action": "eks:ListClusters",
            "Resource": "*"
        }
    ]
}
```


Policy ARN
arn:aws:iam::187632318301:policy/ayush-clt-iam-policy
Description
Permissions
Policy usage
Policy versions
Access Advisor
This policy defines some actions, resources, or conditions that do not provide permissions. To grant access, policies must have an action t
applicable resource or condition. For details, choose Show remaining. Learn more
Policy summary
{ } JSON
Edit policy
Filter
Service
Access level
Resource
Request condition
Allow (1 of 224 services) Show remaining 223
EKS
Full: List Limited: Read
Multiple
None

Create role:


Summary
Role ARN
arn:aws:iam::187632318301:role/iam-ec2-ctl-role
Role description
Allows EC2 instances to call AWS services on your behalf. | Edit
Instance Profile ARNs
arn:aws:iam::187632318301:instance-profile/iam-ec2-ctl-role
Path
/
Creation time
2020-03-05 17:58 UTC+0530
Last activity
Not accessed in the tracking period
Maximum CLI/API session duration
1 hour Edit
Permissions
Trust relationships
Tags (2)
Access Advisor
Revoke sessions
Permissions policies (1 policy applied)
Attach policies
Policy name
Policy type
ayush-clt-iam-policy
Managed policy

Attach role to ec2 instance:



Ssh into iam and get access to this cluster

```
[ec2-user@ip-172-31-28-142 ~]$ aws eks describe-cluster --name ayush-ctl --region us-east-1
{
    "cluster": {
        "status": "ACTIVE",
        "endpoint": "https://8040EDD6F05333462E6F66D0C81BC62A.gr7.us-east-1.eks.amazonaws.com",
        "logging": {
            "clusterLogging": [
                {
                    "enabled": false,
                    "types": [
                        "api",
                        "audit",
                        "authenticator",
                        "controllerManager",
                        "scheduler"
                    ]
                }
            ]
        },
        "name": "ayush-ctl",
        "tags": {
            "owner": "ayush",
            "purpose": "aws-test-cli",
            "Name": "ayush-ctl"
        },
        "certificateAuthority": {
            "data": "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUN5RENDQWJDZ0F3SUJBZ0lCQURBTkJna3Foa2lHOXcwQkFR
GRHVnpNQjRYRFRJd01ETXdOVEV3TVRFd09Wb1hEVE13TURNd016RXdNVEV3T1Zvd0ZURVRNQkVHQTFVRQpBeE1LYTNWaVpYSnVaWFJsY3pDQ0
DQVFvQ2dnRUJBTVNNCmplbjh4WmRMTGptOHY5WVV2T1FFMUs2QS9GUFlpRVU1ZHVMRlJhNHNRRC9TaUxDMW1tOGJjc3Z2dWJsVndDc0YKU1JY
2E4dlhOcXkxaEBoY1lENWo3M2o3S2ZvBoNwSHU4MqpVRHbcaUxobHoyMjEDTURJYWpPdEqwc1M2VEV2dkdKam5iRTZkT1BGU2VDS1E4U0p1eC

```

```
i9WV2NQdWhYQm1wampPV0NqL2NuaEFSSTZpeC9ic0NyU2JsCk5tZ2pTU1VVMUhsL0lwK1B4dUZ3bVRjSVBUelQzMy9CN2VZNmNESEFjb0lnZjk
EeWpmY0RsSTd2S0JPRmNsRkN0RGFmYVVUMHVpbTJGRVJOdGRIQ1FXV3hEaDZPT3V2dndkWQp5YTZEYzFlOEZwUUVzdGU3eFNOaWRvbWZpSEhwM
z0KLS0tLS1FTkQgQ0VSVElGSUNBVEUtLS0tLQo="
        },
        "roleArn": "arn:aws:iam::187632318301:role/eksServiceRole",
        "resourcesVpcConfig": {
            "vpcId": "vpc-0b061c711cd6ec803",
            "subnetIds": [
                "subnet-0a5a6b106347d1b70",
                "subnet-033003c92989d26d9",
                "subnet-070c80956ba0dde0a"
            ],
            "securityGroupIds": [
                "sg-0d7caff2076db4339"
            ],
            "clusterSecurityGroupId": "sg-0a4c6eb86073dd3ae",
            "endpointPublicAccess": true,
            "endpointPrivateAccess": false
        },
        "platformVersion": "eks.9",
        "version": "1.14",
        "arn": "arn:aws:eks:us-east-1:187632318301:cluster/ayush-ctl",
        "identity": {
            "oidc": {
                "issuer": "https://oidc.eks.us-east-1.amazonaws.com/id/8040EDD6F05333462E6F66D0C81BC62A"
            }
        },
        "createdAt": 1583402568.016
    }
}
```

3. Eksctl command to terminate the stack

```
ayush@ayush:~/eks$ eksctl delete cluster -f cluster-eks.yaml
[ℹ]  eksctl version 0.13.0
[ℹ]  using region us-east-1
[ℹ]  deleting EKS cluster "ayush-ctl"
[ℹ]  deleted 0 Fargate profile(s)
[✓]  kubeconfig has been updated
[ℹ]  cleaning up LoadBalancer services
[ℹ]  3 sequential tasks: { 3 parallel sub-tasks: { delete nodegroup "ng-3", delete nodegroup "ng-2", delete nodegroup "ng-1" }, delete IAM OIDC provid
er, delete cluster control plane "ayush-ctl" [async] }
[ℹ]  will delete stack "eksctl-ayush-ctl-nodegroup-ng-3"
[ℹ]  waiting for stack "eksctl-ayush-ctl-nodegroup-ng-3" to get deleted
[ℹ]  will delete stack "eksctl-ayush-ctl-nodegroup-ng-1"
[ℹ]  waiting for stack "eksctl-ayush-ctl-nodegroup-ng-1" to get deleted
[ℹ]  will delete stack "eksctl-ayush-ctl-nodegroup-ng-2"
[ℹ]  waiting for stack "eksctl-ayush-ctl-nodegroup-ng-2" to get deleted
[!]  retryable error (RequestError: send request failed
caused by: Post https://cloudformation.us-east-1.amazonaws.com/: EOF) from cloudformation/DescribeStacks - will retry after delay of 41.457008ms
[!]  retryable error (RequestError: send request failed
caused by: Post https://cloudformation.us-east-1.amazonaws.com/: EOF) from cloudformation/DescribeStacks - will retry after delay of 99.732044ms
[ℹ]  will delete stack "eksctl-ayush-ctl-cluster"
[✓]  all cluster resources were deleted
```