# #Day-12

Today It's about RBAC in kubeenetes :-

Firstly RBAC means Role based Access Control, As the name suggest it is used to give the appropriate access according to the roles allocated people. Similar to RBAC in Azure cloud and AWS cloud.

For Example. If you take a teaching institute, there will be lot of people who are ideally do different work according to roles thy have.

principal - manages Everything in college.

Subject1 Faculty - teaches Subject1

Subject2 faculty - teaches Subject2

Exactly. kubeenetes will provide provision to enhance Security using the features of RBAC. This will eliminate unauthorized usage of kubeenetes Services. If the RBAC is not there in kubeenetes, anyone can delete/modify/update the cluster. This should not be done in prod.

RBAC Broadly divided as follows:

$$\begin{tikzcd}
\text{RBAC} \arrow[r] \arrow[dr] & \text{users} \\
 & \text{Services Accounts.}
\end{tikzcd}$$

In Real time developer, QA, Devops, Kubernetes admin Should have different Roles there by differents levels of access to cluster.

**USERS:-** This is Just like IAM users in AWS or AAD users in Azure. This used to get users access to cluster,
→ user management.

→ We cannot create users directly on kubernetes. We can use Something called identity provider. which will authenticate user to access cluster.

These Identity provider may be AWS IAM. Azure AD, LDAP, SSO etc. There is OAuth to authenticate users (needs to integrate with Kubernetes) to access cluster.

So on high note, Kubernetes won't do any kind of user management. It's the identity provider who has to do.

SERVICE ACCOUNTS :- These are just like kubernetes objects (pod, Deploy, SVC etc). It's an yaml file to be ran on cluster to create Service accounts.

Service Account is needed because think of pod should be able to access various things such as Secrets / configmap, it is possible through Service accounts.

To manage RBAC we have

1) Service Accounts / users

2) Roles / Cluster Role

3) Role binding / cluster binding

Note: There is a default service account that is associated with each pod to establish Communication between control plane and pod.

1) A role will be created with details using Role.yaml

2) Attaching the role to service account or user is called Role binding using a file.

3) If it is constained to namespace then we will call it as role and role binding.

4) If it is at cluster level access then it is cluster role and Cluster role binding.

5) There will be default roles present in Kubernetes. We can make use of them if it matches same access levels.

## Demo:-

### Role creation and Binding:-

1) Create a test namespace.

```
# kubectl create ns test
```

2) Create a service account using below file:

```
apiVersion: v1
kind: ServiceAccount
metadata:
   name: myaccount
   namespace: test
```

3) Create the Service Account with name myaccount.

```
# kubectl apply -f sa.yaml
```

4) Now create Role which has access Rules.

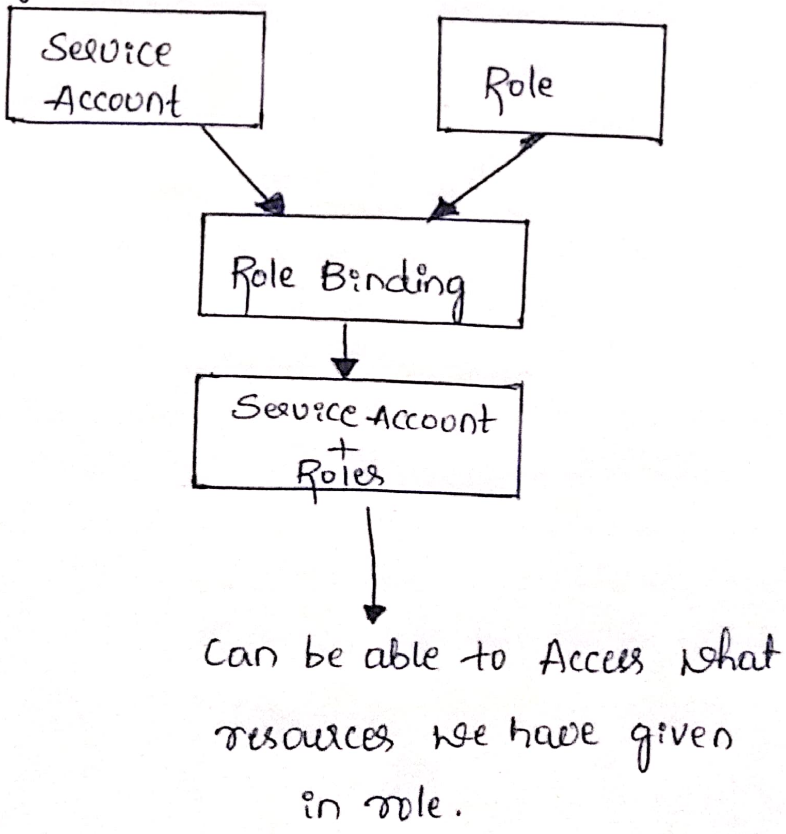
```
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
   namespace: test
   name: testadmin
rules:
- apiGroups: ["*"]
  resources: ["*"]
  verbs: ["*"]
```

→ This roles having access to everything in the cluster. (yet not attached)

5) Now create the file for role binding using below text.

```
apiVersion: rbac.authorization.k8s.io/V1
kind: RoleBinding
metadata:
    name: testadminbinding
    namespace: test
                  ↳ Namespace
Subjects:
 - kind: ServiceAccount
   name: myaccount  ──────→  Service Account
   apiGroup: ""                  Created
roleRef:
   kind: Role
   name: testadmin  ──────→  Role we Created.
   apiGroup: ""
```

This how we can create Service account, Roles and Role bindings.



| Authentication | Vs | Authorization |
|---|---|---|
| - It Authenticates User againt creds whether correct user logged in or out. | | - It authorizes the user have a Specific user or not against the policies/ Roles. |