echo '-----BEGIN CERTIFICATE-----

-----END CERTIFICATE-----' >> CA.crt

MIIEpAIBAAKCAQEAtf4dOPw0bNGc. . . . .

echo '-----BEGIN RSA PRIVATE KEY-----

MIIEpAIBAAKCAQEAtf4dOPw0bNGc. . . . .

-----END RSA PRIVATE KEY-----' >> CA.key

kubectl config set-context --current --namespace=production

export KUBECONFIG=/root/anand/config

export KUBECONFIG=/root/anand/config

cd ${HOME}/.kube

openssl genrsa -out anand.key 2048

openssl req -new -key anand.key -out anand.csr -subj "/CN=anand/O=development"

kubectl config view #To find where is root CA and Key

openssl x509 -req -in anand.csr -CA ${HOME}/.minikube/ca.crt -CAkey ${HOME}/.minikube/ca.key -CAcreateserial -out anand.crt -days 45

To add the user in the Kubeconfig file, we can execute below command (set-credentials). Please make sure that you provide the correct path to the private key and the certificate of anand.

kubectl config set-credentials anand --client-certificate ${HOME}/.kube/anand.crt --client-key ${HOME}/.kube/anand.key

kubectl config view #Verify User details in the kubeconfig as below.

users:

- name: anand

user:

client-certificate: anand.crt

client-key: anand.key

The next step is to add a context in the config file, that will allow this user (anand) to access the development namespace in the cluster.

kubectl config set-context anand-context --cluster=minikube --namespace=development --user=anand

kubectl config view #Verify User context details as below.

- context:

cluster: minikube

namespace: development

user: anand

name: anand-context

$ kubectl get pods --context=anand-context

Error from server (Forbidden): pods is forbidden:

User "anand" cannot list resource "pods" in API group "" in the namespace "development"

kind: Role

apiVersion: rbac.authorization.k8s.io/v1

metadata:

name: payroll-role

namespace: payroll

rules:

- apiGroups: ["", "extensions", "apps"] # "" indicates the core API group for eacho use '' not "".

resources: ["pods", "deployments", "replicasets"]

verbs: ["get", "update", "list", "create", "delete"]

kind: RoleBinding

apiVersion: rbac.authorization.k8s.io/v1

metadata:

name: anand-RoleBinding

namespace: payroll

subjects:

- kind: User

name: anand

apiGroup: rbac.authorization.k8s.io

roleRef:

kind: Role

name: payroll-role

apiGroup: rbac.authorization.k8s.io

kubectl get pods --context=DevUser-context

========================MULTIPLE NAMESPACES===========================

ku create ns payroll-dev

ku create ns payroll-staging

kind: Role

apiVersion: rbac.authorization.k8s.io/v1

metadata:

name: payroll-dev-role

namespace: payroll-dev

rules:

- apiGroups: ["", "extensions", "apps"] # "" indicates the core API group

resources: ["pods", "deployments", "replicasets"]

verbs: ["get", "update", "list", "create", "delete"]

kind: Role

apiVersion: rbac.authorization.k8s.io/v1

metadata:

name: payroll-staging-role

namespace: payroll-staging

rules:

- apiGroups: ["", "extensions", "apps"] # "" indicates the core API group

resources: ["pods", "deployments", "replicasets"]

verbs: ["get", "update", "list", "create", "delete"]

kind: RoleBinding

apiVersion: rbac.authorization.k8s.io/v1

metadata:

name: anand-RoleBinding-payroll-dev

namespace: payroll-dev

subjects:

- kind: User

name: anand

apiGroup: rbac.authorization.k8s.io

roleRef:

kind: Role

name: payroll-dev-role

apiGroup: rbac.authorization.k8s.io

kind: RoleBinding

apiVersion: rbac.authorization.k8s.io/v1

metadata:

name: anand-RoleBinding-payroll-staging

namespace: payroll-staging

subjects:

- kind: User

name: anand

apiGroup: rbac.authorization.k8s.io

roleRef:

kind: Role

name: payroll-staging-role

apiGroup: rbac.authorization.k8s.io

=================================================

kind: ClusterRole

apiVersion: rbac.authorization.k8s.io/v1

metadata:

name: payroll-cluster-wide-role

rules:

- apiGroups: ["", "extensions", "apps"] # "" indicates the core API group

resources: ["pods", "deployments", "replicasets"]

verbs: ["get", "update", "list", "create", "delete"]

kind: RoleBinding

apiVersion: rbac.authorization.k8s.io/v1

metadata:

name: anand-Cluster-Role-Binding

namespace: payroll

subjects:

- kind: User

name: anand

apiGroup: rbac.authorization.k8s.io

roleRef:

kind: ClusterRole

name: payroll-cluster-wide-role

apiGroup: rbac.authorization.k8s.io

kind: RoleBinding

apiVersion: rbac.authorization.k8s.io/v1

metadata:

name: anand-Cluster-Role-Binding

namespace: payroll-dev

subjects:

- kind: User

name: anand

apiGroup: rbac.authorization.k8s.io

roleRef:

kind: ClusterRole

name: payroll-cluster-wide-role

apiGroup: rbac.authorization.k8s.io

kind: RoleBinding

apiVersion: rbac.authorization.k8s.io/v1

metadata:

name: anand-Cluster-Role-Binding

namespace: payroll-staging

subjects:

- kind: User

name: anand

apiGroup: rbac.authorization.k8s.io

roleRef:

kind: ClusterRole

name: payroll-cluster-wide-role

apiGroup: rbac.authorization.k8s.io

=======CLUSTER-ADMIN--ROLE===================

apiVersion: rbac.authorization.k8s.io/v1

kind: ClusterRole

metadata:

name: new-cluster-admin-role

rules:

- apiGroups: ["\*"]

resources: ["\*"]

verbs: ["\*"]

---

kind: ClusterRoleBinding

apiVersion: rbac.authorization.k8s.io/v1

metadata:

name: ClusterRole-Anand

subjects:

- kind: User

name: anand

apiGroup: rbac.authorization.k8s.io

roleRef:

kind: ClusterRole

name: new-cluster-admin-role

apiGroup: rbac.authorization.k8s.io

---

kind: Role

apiVersion: rbac.authorization.k8s.io/v1

metadata:

namespace: rbac-test

name: pod-reader

rules:

- apiGroups: [""] # "" indicates the core API group

resources: ["pods"]

verbs: ["list","get","watch"]

- apiGroups: ["extensions","apps"]

resources: ["deployments"]

verbs: ["get", "list", "watch"]