**Security**

1. Create a Role named "pod-reader" that allows users to perform get, watch and list on pods:
   1. kubectl create role pod-reader --verb=get --verb=list --verb=watch --resource=pods
2. Create a Role named "pod-reader1" with resourceNames specified:
   1. kubectl create role pod-reader1 --verb=get --resource=pods --resource-name=readablepod --resource-name=anotherpod
3. Create a Role named "pod-user" with apiGroups specified:
   1. kubectl create role pod-user --verb=get,list,watch --resource=replicasets.apps
4. Create a Role named " pod-user1" with subresource permissions:
   1. kubectl create role pod-user1 --verb=get,list,watch --resource=pods,pods/status
5. Create a Role named "my-component-lease-holder" with permissions to get/update a resource with a specific name:
   1. kubectl create role my-component-lease-holder --verb=get,list,watch,update --resource=lease --resource-name=my-component
6. Create a ClusterRole named "pod-reader2" that allows user to perform get, watch and list on pods:
   1. kubectl create clusterrole pod-reader2 --verb=get,list,watch --resource=pods
7. Create a ClusterRole named "pod-reader3" with resourceNames specified:
   1. kubectl create clusterrole pod-reader3 --verb=get --resource=pods --resource-name=readablepod --resource-name=anotherpod
8. Create a ClusterRole named "cluster-user" with apiGroups specified:
   1. kubectl create clusterrole cluster-user --verb=get,list,watch --resource=replicasets.apps
9. Create a ClusterRole named " cluster-user1" with subresource permissions:
   1. kubectl create clusterrole cluster-user1 --verb=get,list,watch --resource=pods,pods/status
10. Create a ClusterRole named " cluster-user2" with nonResourceURL specified:
    1. kubectl create clusterrole "cluster-user2" --verb=get --non-resource-url=/logs/\*
11. Create a ClusterRole named "monitoring" with an aggregationRule specified:
    1. kubectl create clusterrole monitoring --aggregation-rule="rbac.example.com/aggregate-to-monitoring=true"
12. Within the namespace "default", grant the permissions in the "admin" ClusterRole to a user named "jitendra":
    1. kubectl create rolebinding jitendra-admin-binding --clusterrole=admin --user=jitendra --namespace=default
13. Within the namespace "default", grant the permissions in the "view" ClusterRole to the service account in the namespace "default" named "myapp":
    1. kubectl create rolebinding myapp-view-binding --clusterrole=view --serviceaccount=default:myapp --namespace=default
14. Within the namespace "default", grant the permissions in the "view" ClusterRole to a service account in the namespace "myappnamespace" named "myapp":
    1. kubectl create rolebinding myappnamespace-myapp-view-binding --clusterrole=view --serviceaccount=myappnamespace:myapp --namespace=default
15. Across the entire cluster, grant the permissions in the "cluster-admin" ClusterRole to a user named "root"
    1. kubectl create clusterrolebinding root-cluster-admin-binding --clusterrole=cluster-admin --user=root
16. Across the entire cluster, grant the permissions in the "system:node-proxier" ClusterRole to a user named "system:kube-proxy":
    1. kubectl create clusterrolebinding kube-proxy-binding --clusterrole=system:node-proxier --user=system:kube-proxy
17. Across the entire cluster, grant the permissions in the "view" ClusterRole to a service account named "myapp" in the namespace "default":
    1. kubectl create clusterrolebinding myapp-view-binding --clusterrole=view --serviceaccount=default:myapp

**Kubeadm**

* install kubeadm
  + sudo snap install kubeadm –classic
* check version
  + kubeadm version
* upgrade Kubernetes(for knowledge only)
  + kubeadm upgrade plan (this will check the cluster to see upgrade availability)
  + sudo kubeadm upgrade apply v1.23.x

(here x is the version, we want to upgrade to)