Defining authorisation rules with Roles and ClusterRoles

**Define authorisation rules with Roles and ClusterRoles**

Whats the difference between Roles and ClusterRoles ??

* Role is limited to a namespace (Projects/Orgs/Env)
* ClusterRole is Global

Let’s say you want to provide read only access to **instavote**, a project specific namespace to all users in the **example.org**

file: interns-role.yaml

1. apiVersion: rbac.authorization.k8s.io/v1beta1
2. kind: Role
3. metadata:
4. namespace: instavote
5. name: interns
6. rules:
7. - apiGroups: ["\*"]
8. resources: ["\*"]
9. verbs: ["get", "list", "watch"]

In order to map it to all users in **example.org**, create a RoleBinding as

Interns-rolebinding.yml

1. kind: RoleBinding
2. apiVersion: rbac.authorization.k8s.io/v1
3. metadata:
4. name: interns
5. namespace: instavote
6. subjects:
7. - kind: Group
8. name: interns
9. apiGroup: rbac.authorization.k8s.io
10. roleRef:
11. kind: Role
12. name: interns
13. apiGroup: rbac.authorization.k8s.io
14. kubectl create -f interns-role.yml
16. kubectl create -f interns-rolebinding.yml

To get information about the objects created above,

1. kubectl get roles -n instavote
2. kubectl get roles,rolebindings -n instavote
4. kubectl describe role interns
5. kubectl describe rolebinding interns

To validate the access,

1. kubectl config use-context yono-prod
2. kubectl get pods

**To switch back to admin,**

1. kubectl config use-context admin-prod

**Exercise**

Create a Role and Rolebinding for **dev** group with the authorizations defined in the table above. Once applied, test it