As kubernetes [docs](https://kubernetes.io/docs/reference/access-authn-authz/authentication/) and Articles uses certificate to create or authenticate users for kubectl client. However there is one easy way to do it by using ServiceAccount. One can use ServiceAccount as a group to provide RBAC control authentication and it is very easy and descriptive. Here are the steps. All the steps i am executing is in default namespace. I am going to create a pod readonly user which can get,list,watch any pod in all namespaces.

* Create a ServiceAccount, say 'readonlyuser'.

kubectl create serviceaccount readonlyuser

* Create cluster role, say 'readonlyuser'.

kubectl create clusterrole readonlyuser --verb=get --verb=list --verb=watch --resource=pods

* Create cluster role binding, say 'readonlyuser'.

kubectl create clusterrolebinding readonlyuser --serviceaccount=default:readonlyuser --clusterrole=readonlyuser

* Now get the token from secret of ServiceAccount we have created before. we will use this token to authenticate user.

TOKEN=$(kubectl describe secrets "$(kubectl describe serviceaccount readonlyuser | grep -i Tokens | awk '{print $2}')" | grep token: | awk '{print $2}')

* Now set the credentials for the user in kube config file. I am using 'chandan' as username.

kubectl config set-credentials vikash --token=$TOKEN

* Now Create a Context say podreader. I am using my clustername 'kubernetes' here.

kubectl config set-context podreader --cluster=kubernetes --user=chandan

* Finally use the context .

kubectl config use-context podreader

And that's it. Now one can execute kubectl get pods --all-namespaces. One can also check the access by executing as given:

~ : $ kubectl auth can-i get pods --all-namespaces

yes

~ : $ kubectl auth can-i create pods

no

~ : $ kubectl auth can-i delete pods

no