\*\* [ sudo su ] \*\*

**First install k3s in the intended Karmada server:**

export K3S\_KUBECONFIG\_MODE="644"  
export INSTALL\_K3S\_EXEC=" --disable servicelb --disable traefik"  
export KUBECONFIG="~/.kube/config:/etc/rancher/k3s/k3s.yaml"

export INSTALL\_K3S\_CHANNEL="v1.30.4+k3s1"  
curl -sfL https://get.k3s.io | sh -

—-------------------------------------------------------------------------

From official repo:

> git clone <https://github.com/karmada-io/karmada>

> cd karmada

make kubectl-karmada

[if make not found: apt install make]

[ go version 1.23 install:

Delete previous: sudo rm -rf /usr/local/go

wget https://dl.google.com/go/go1.23.7.linux-amd64.tar.gz

export GOPATH=$HOME/go

export PATH=$PATH:$GOROOT/bin:$GOPATH/bin

source ~/.bashrc ]

> sudo mv \_output/bin/linux/amd64/kubectl-karmada /usr/local/bin/

> sudo kubectl karmada version

—-------------------------------------------------------------------------------

By default, k3s stores its kubeconfig at /etc/rancher/k3s/k3s.yaml instead of ~/.kube/config

cat /etc/rancher/k3s/k3s.yaml > ~/.kube/config

chmod 600 ~/.kube/config

[if previously exists] kubectl delete ns karmada-cluster karmada-system

sudo kubectl karmada init

To check logs: sudo journalctl -u k3s -f

systemctl status k3s

**To get actual ip**

ip a | grep inet

**TO JOIN KARMADA SERVER AS HOST CLUSTER:**

> MEMBER\_CLUSTER\_NAME=$(kubectl config current-context)

> kubectl karmada --kubeconfig /etc/karmada/karmada-apiserver.config \

join ${MEMBER\_CLUSTER\_NAME} \

--cluster-kubeconfig=/etc/rancher/k3s/k3s.yaml

The VM of itself where karmada server initiated as a cluster master:

> kubectl --kubeconfig /etc/karmada/karmada-apiserver.config patch cluster default --type=merge -p '{"spec":{"apiEndpoint":"<https://192.168.70.22:6443>"}}'

1 This changes the API server endpoint that Karmada uses to communicate with the default cluster. Karmada will now try to reach the Kubernetes API server at 192.168.70.22:6443 instead of the previous endpoint(like self-loop localhost)

**To check member clusters**

> kubectl --kubeconfig /etc/karmada/karmada-apiserver.config get clusters

By default, k3s stores its kubeconfig at /etc/rancher/k3s/k3s.yaml instead of ~/.kube/config

**TO JOIN REMOTE CLUSTER:**

> Install k3s in remote [say mdr4]

copy mdr4 kubeconfig:

copy [cat /etc/rancher/k3s/k3s.yaml] from mdr4   
 and paste it to /etc/karmada/mdr4-kubeconfig in mdr1 [karmada server VM] and

**Update the ip written inside as mdr4’s IP**

**Run on the VM where karmada server initialized[mdr1]:**

> kubectl karmada --kubeconfig /etc/karmada/karmada-apiserver.config \

join mdr4 \

--cluster-kubeconfig=/etc/karmada/mdr4-kubeconfig

**Delete Cluster:**  
> kubectl --kubeconfig /etc/karmada/karmada-apiserver.config delete clusters default

**Argocd install in karmada:**

kubectl create namespace argocd

kubectl apply -n argocd -f https://raw.githubusercontent.com/argoproj/argo-cd/stable/manifests/install.yaml

Get password:

kubectl -n argocd get secret argocd-initial-admin-secret -o jsonpath="{.data.password}" | base64 -d; echo

Change clusterip to nodeport:

kubectl edit svc argocd-server -n argocd [for nano edit: press i. Save: wq!]

kubectl config get-contexts

Argocd CLI installation:  
> ARGOCD\_VERSION=$(curl --silent "https://api.github.com/repos/argoproj/argo-cd/releases/latest" | grep '"tag\_name"' | sed -E 's/.\*"([^"]+)".\*/\1/')

> curl -sSL -o /tmp/argocd-${ARGOCD\_VERSION} [https://github.com/argoproj/argo-cd/releases/download/${ARGOCD\_VERSION}/argocd-linux-amd64](https://github.com/argoproj/argo-cd/releases/download/$%7BARGOCD_VERSION%7D/argocd-linux-amd64)

> chmod +x /tmp/argocd-${ARGOCD\_VERSION}

sudo mv /tmp/argocd-${ARGOCD\_VERSION} /usr/local/bin/argocd

argocd version --client

> kubectl config get-contexts --kubeconfig /etc/karmada/karmada-apiserver.config

Output should be:

CURRENT NAME CLUSTER AUTHINFO NAMESPACE

\* karmada-apiserver karmada-apiserver karmada-admin

Argocd login:

argocd login "$ARGOCD\_SERVER" --username "$ARGOCD\_USERNAME" --password "$ARGOCD\_PASSWORD" --insecure

Example:

argocd login 192.168.70.25:30613 --username admin --password Nx1r7GYlglb0hN9M --insecure

export KUBECONFIG=/etc/karmada/karmada-apiserver.config

kubectl config get-contexts -o name

argocd cluster add karmada-apiserver

**DEPLOYMENTS**

**\*Deployments will be performed through argocd\***

**But make sure that each and every resource(yamls) must have corresponding propagationpolicy.yamls in the /templates sub-chart.**

**Check the propagationpolicy.yaml, pvcpropagationpolicy.yaml, servicepropagationpolicy.yaml in this repository. Say, you have serviceaccount resource but don’t have a propagation policy(propagation-serviceaccount.yaml) for this, then issues will arise.**

//////

First deploy the deployment:

kubectl --kubeconfig=/etc/karmada/karmada-apiserver.config apply -f deployment.yaml

kubectl --kubeconfig=/etc/karmada/karmada-apiserver.config get deployments -A

At this moment you will find No pods created, Deploy-Ready: 0/2, kubectl get futile

Now apply propagation policy:

kubectl --kubeconfig=/etc/karmada/karmada-apiserver.config apply -f propagationpolicy.yaml

kubectl --kubeconfig=/etc/karmada/karmada-apiserver.config get propagationpolicy -A

Here to note: resourceSelectors in propagation policy specs should be identical to

the name of Deployment and Service (metadata).

Now you will find deployment done:

kubectl --kubeconfig=/etc/karmada/karmada-apiserver.config get deployments -A

Also kubectl get pods & deployment works now too from clusters

deploy service and Apply propagation policy for service:

kubectl --kubeconfig=/etc/karmada/karmada-apiserver.config apply -f nginx-nodeport-service.yam

kubectl --kubeconfig=/etc/karmada/karmada-apiserver.config apply -f nginx-service-propagation.yaml

kubectl --kubeconfig=/etc/karmada/karmada-apiserver.config get svc

also works kubectl get svc from clusters

**If two many files issue**

**sudo sysctl -w fs.inotify.max\_user\_watches=2099999999 sudo sysctl -w fs.inotify.max\_user\_instances=2099999999 sudo sysctl -w fs.inotify.max\_queued\_events=2099999999**

Argocd GUI creds: admin / -g9LGICe-MVbiYA1

Refrences: https://github.com/ShantoOR/express3.git