KUBERNETES INSTALATION

## K8s Installations

* Single Node Installations
  + minikube
  + kind
* On-prem installations
  + kube-admin
* k8s as a Service
  + AKS
  + EKS
  + GKE
* Playground (for learning):

<https://labs.play-with-k8s.com/>

## Installing k8s cluster on ubuntu vms

* Create 3 ubuntu vms which are accesible to each other with atlest 2 vCPUS and 4 GB RAM
* Installation method (kubeadm) which is something we will be using in on-premises k8s.
* In this 3 machines give a permition in security group as “ open all traffic “

### Steps

* In 3 machines
* Install docker on all nodes

$ curl -fsSL https://test.docker.com -o test-docker.sh

$ sh test-docker.sh

* Put a name
* Master
* Node1
* Node2

$ sudo usermod 777 /var/run/docker.sock

$ sudo chmod –aG docker ububtu

* Exit and relogin
* Check for docker

$ docker info

* Showing docker version and some info also
* Install CRI-Dockerd

<https://github.com/Mirantis/cri-dockerd>

* Run the below commands as root user in all the nodes

# Run these commands as root

###Install GO###

wget <https://storage.googleapis.com/golang/getgo/installer_linux>

chmod +x ./installer\_linux

./installer\_linux

source ~/.bash\_profile

git clone <https://github.com/Mirantis/cri-dockerd.git>

cd cri-dockerd

mkdir bin

go build -o bin/cri-dockerd

mkdir -p /usr/local/bin

install -o root -g root -m 0755 bin/cri-dockerd /usr/local/bin/cri-dockerd

cp -a packaging/systemd/\* /etc/systemd/system

sed -i -e 's,/usr/bin/cri-dockerd,/usr/local/bin/cri-dockerd,' /etc/systemd/system/cri-docker.service

systemctl daemon-reload

systemctl enable cri-docker.service

systemctl enable --now cri-docker.socket

* Installing kubadm, kubectl, kubelet

<https://kubernetes.io/docs/setup/production-environment/tools/kubeadm/install-kubeadm/>

$ sudo apt-get update

$ sudo apt-get install -y apt-transport-https ca-certificates curl

$ curl -fsSL https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-archive-keyring.gpg

$ echo "deb [signed-by=/etc/apt/keyrings/kubernetes-archive-keyring.gpg] https://apt.kubernetes.io/ kubernetes-xenial main" | sudo tee /etc/apt/sources.list.d/kubernetes.list

$ sudo apt-get update

sudo apt-get install -y kubelet kubeadm kubectl

sudo apt-mark hold kubelet kubeadm kubectl

* Now create a cluster from a master node
* Where will run this command that is master node
* use the command

$ kubeadm init --pod-network-cidr "10.244.0.0/16" --cri-socket "unix:///var/run/cri-dockerd.sock"

After run this command we need to configure the kubectl cluster

Exit the root user execute the these 3 commands

$ mkdir -p $HOME/.kube

$ sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config

$ sudo chown $(id -u):$(id -g) $HOME/.kube/config

$ kubectl get nodes

$ kubectl get nodes –w

These nodes are not ready because of its doesn’t have network policy

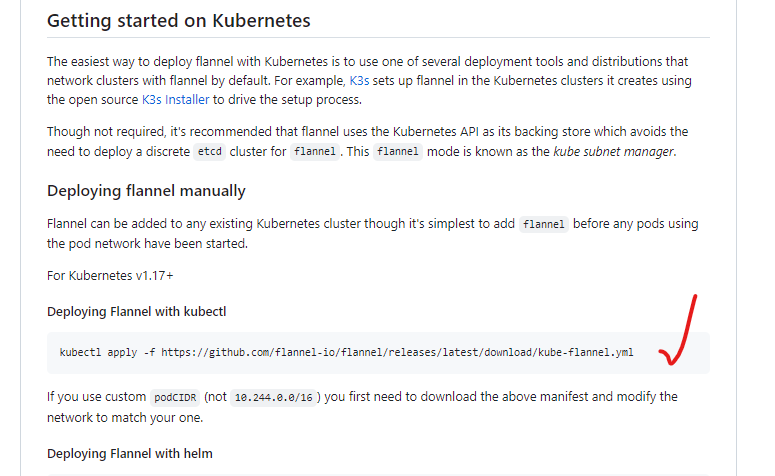
For that network policy run this command

* Setup kubeconfig
* install plannel

<https://kubernetes.io/docs/concepts/cluster-administration/addons/>

<https://github.com/flannel-io/flannel#deploying-flannel-manually>

after open this scrolldown



$ kubectl apply -f <https://github.com/flannel-io/flannel/releases/latest/download/kube-flannel.yml>

$ kubectl get nodes

$ kubectl get nodes –w (watch)

It will show nodes are ready To need to add nodes in this cluster run this command

Run this command as a root user

Run 4 lines at a time, after run this command exit as a regular user

$ kubeadm join 172.31.92.205:6443 --token mpoihs.d7eowfdydxbhu16h \

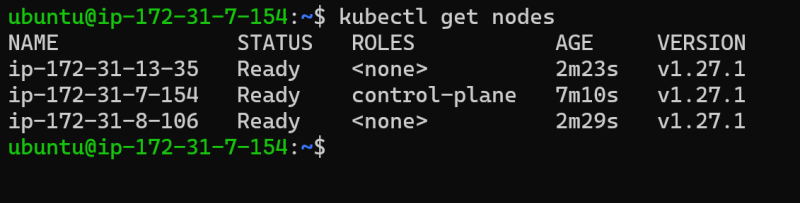
--cri-socket "unix:///var/run/cri-dockerd.sock" \

--discovery-token-ca-cert-hash sha256:88e39fceb35cb59431ee648ef92c3f38a3fb3d3c230ada4968ca333cf664ad23

$ kubectl get nodes

$ kubectl get nodes –w

It will show 3 nodes ready state



This is kubernetes cluster