Kubernetes Setup Using Kubeadm In AWS EC2 Ubuntu Servers Container-D As Runtime

=======

Prerequisite:

=============

3 - Ubuntu Serves

1 - Manager (4GB RAM , 2 Core) t2.medium

2 - Workers (1 GB, 1 Core) t2.micro

Note: Open Required Ports In AWS Security Groups. For now we will open All trafic.

==========COMMON FOR MASTER & SLAVES START ====

1) Switch to root user

sudo su -

2) Turn Off Swap Space

swapoff -a

sed -i '/ swap / s/^\(.\*\)$/#\1/g' /etc/fstab

3) Install packages.To install Kubernetes and containerd run these commands:

apt update -y

apt install -y apt-transport-https -y

curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | apt-key add -

cat <<EOF >/etc/apt/sources.list.d/kubernetes.list

deb http://apt.kubernetes.io/ kubernetes-xenial main

EOF

apt update -y

apt install -y kubelet kubeadm containerd kubectl

# apt-mark hold will prevent the package from being automatically upgraded or removed.

apt-mark hold kubelet kubeadm kubectl containerd

4) Configure Containerd.Load the necessary modules for Containerd:

cat <<EOF | sudo tee /etc/modules-load.d/containerd.conf

overlay

br\_netfilter

EOF

modprobe overlay

modprobe br\_netfilter

5) Setup the required kernel parameters

cat <<EOF | sudo tee /etc/sysctl.d/99-kubernetes-cri.conf

net.bridge.bridge-nf-call-iptables = 1

net.bridge.bridge-nf-call-ip6tables = 1

net.ipv4.ip\_forward = 1

EOF

sysctl --system

6) Configure containerd:

mkdir -p /etc/containerd

containerd config default | sudo tee /etc/containerd/config.toml

systemctl restart containerd

7) Start and enable kubelet service

# Enable and start kubelet service

systemctl daemon-reload

systemctl start kubelet

systemctl enable kubelet.service

===========In Master ====================

# Initialize Kubernates master by executing below commond.

kubeadm init

IF Error

sudo kubeadm init --cri-socket /run/containerd/containerd.sock

#exit as root user & exeucte as normal ubuntu user

exit

mkdir -p $HOME/.kube

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

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

# To verify, if kubectl is working or not, run the following command.

kubectl get pods -o wide -n kube-system

#You will notice from the previous command, that all the pods are running except one: ‘core-dns’. For resolving this we will install a # pod network. To install the weave pod network, run the following command:

kubectl apply -f "https://cloud.weave.works/k8s/net?k8s-version=$(kubectl version | base64 | tr -d '\n')"

kubectl get nodes

kubectl get pods

kubectl get pods --all-namespaces

# Get token

kubeadm token create --print-join-command

=========================================

NOde Side:

Add Worker Machines to Kubernates Master

Copy kubeadm join token from and execute in Worker Nodes to join to cluster

kubectl commonds has to be executed in master machine.

Check Nodes

=============

kubectl get nodes

Deploy Sample Application

==========================

kubectl run nginx-demo --image=nginx --port=80

kubectl expose pod nginx-demo --port=80 --type=NodePort

Get Node Port details

=====================

kubectl get services