Kubernetes cluster Setup on AWS Using Kubeadm and Containerd

Prerequisites

- A compatible Linux hosts: 2 GB or more of RAM per machine and 2 CPUs or more
- 3 - Ubuntu 20.04 LTS Serves: 1x Manager (4GB RAM, 2 vCPU)t2.medium type, 2x Workers (1 GB, 1 Core) t2.micro type
- Full network connectivity between all machines in the cluster
- Unique hostname for each host. Change hostname of the machines using hostnamectl. For master nodes, run`hostnamectl set-hostname master`. For slaves, run `hostnamectl set-hostname slave-01` `hostnamectl set-hostname slave-02`
- Certain ports are open on your machines(https://kubernetes.io/docs/reference/ports-and-protocols/)

On Master Node :-

6443/tcp for Kubernetes API Server

2379-2380 for etcd server client API

6783/tcp,6784/udp for Weavenet CNI

10248-10260 for Kubelet API, Kube-scheduler, Kube-controller-manager, Read-Only Kubelet API, Kubelet health

80,8080,443 Generic Ports

30000-32767 for NodePort Services

On Worker Node :-

6783/tcp,6784/udp for Weavenet CNI

10248-10260 for Kubelet API etc

30000-32767 for NodePort Services

Run on all nodes of the cluster as root user

Disable Swap :-

swapoff -a

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

Install Containerd :-

- wget https://github.com/containerd/containerd/releases/download/v1.6.16/containerd-1.6.16-linux-amd64.tar.gz
- tar Cxzvf /usr/local containerd-1.6.16-linux-amd64.tar.gz
- wget https://raw.githubusercontent.com/containerd/containerd/main/containerd.service
- mkdir -p /usr/local/lib/systemd/system
- mv containerd.service /usr/local/lib/systemd/system/containerd.service
- systemctl daemon-reload
- systemctl enable --now containerd

Install Runc :-

- wget https://github.com/opencontainers/runc/releases/download/v1.1.4/runc.amd64
- install -m 755 runc.amd64 /usr/local/sbin/runc

Install CNI :-

- wget https://github.com/containernetworking/plugins/releases/download/v1.2.0/cni-plugins-linux-amd64-v1.2.0.tgz
- mkdir -p /opt/cni/bin
- tar Cxzvf /opt/cni/bin cni-plugins-linux-amd64-v1.2.0.tgz

Install CRICTL :-

- VERSION="v1.26.0" # check latest version in /releases page
- wget https://github.com/kubernetes-sigs/cri-tools/releases/download/$VERSION/crictl-$VERSION-linux-amd64.tar.gz
- sudo tar zxvf crictl-$VERSION-linux-amd64.tar.gz -C /usr/local/bin
- rm -f crictl-$VERSION-linux-amd64.tar.gz

- cat <<EOF | sudo tee /etc/crictl.yaml
- runtime-endpoint: unix:///run/containerd/containerd.sock
- image-endpoint: unix:///run/containerd/containerd.sock
- timeout: 2
- debug: false
- pull-image-on-create: false
- EOF

Forwarding IPv4 and letting iptables see bridged traffic

https://kubernetes.io/docs/setup/production-environment/container-runtimes/#forwarding-ipv4-and-letting-iptables-see-bridged-traffic

- cat <<EOF | sudo tee /etc/modules-load.d/k8s.conf
- overlay
- br_netfilter
- EOF
- sudo modprobe overlay
- sudo modprobe br_netfilter
- cat <<EOF | sudo tee /etc/sysctl.d/k8s.conf
- net.bridge.bridge-nf-call-iptables = 1
- net.bridge.bridge-nf-call-ip6tables = 1
- net.ipv4.ip_forward = 1
- EOF
- sudo sysctl --system
- sysctl net.bridge.bridge-nf-call-iptables net.bridge.bridge-nf-call-ip6tables net.ipv4.ip_forward
- modprobe br_netfilter
- sysctl -p /etc/sysctl.conf

INSTALL Kubeadm ,Kubectl and Kubelet :-

- apt-get update && sudo apt-get install -y apt-transport-https curl
- curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo apt-key add -
- cat <<EOF | sudo tee /etc/apt/sources.list.d/kubernetes.list
- deb https://apt.kubernetes.io/ kubernetes-xenial main
- EOF
- apt update -y
- apt install -y kubelet kubeadm kubectl
- sudo apt-mark hold kubelet kubeadm kubectl
- sudo apt-mark hold kubelet kubeadm kubectl

Run on Master Node and follow the instructions

- kubeadm config images pull
- kubeadm init

Install any CNI plugin . We will use weavenet :-

kubectl apply -f https://github.com/weaveworks/weave/releases/download/v2.8.1/weave-daemonset-k8s.yaml

Run on Worker Nodes .

Run the join command obtained from kubeadm init output on all Workers nodes. Example

kubeadm join \

192.168.56.2:6443 --token ... --discovery-token-ca-cert-hash sha256 . . . .

Test the SETUP :-

- kubectl get nodes
- kubectl get pods -A

Run a Demo App :-

- kubectl run nginx --image=nginx --port=80
- kubectl expose pod nginx --port=80 --type=NodePort

- https://kubernetes.io/docs/setup/production-environment/tools/kubeadm/install-kubeadm/
- https://kubernetes.io/docs/setup/production-environment/container-runtimes/#containerd
- https://kubernetes.io/docs/setup/production-environment/tools/kubeadm/create-cluster-kubeadm/
- https://www.mirantis.com/blog/how-install-kubernetes-kubeadm/
- https://www.mankier.com/1/kubeadm-init
- https://kubernetes.io/docs/setup/production-environment/container-runtimes/#docker
- https://github.com/containerd/containerd/blob/main/docs/getting-started.md
- https://kubernetes.io/docs/reference/networking/ports-and-protocols/
- https://www.weave.works/docs/net/latest/kubernetes/kube-addon/#install
- https://github.com/skooner-k8s/skooner
- https://www.weave.works/docs/net/latest/kubernetes/kube-addon/#eks
- https://github.com/kubernetes-sigs/cri-tools/blob/master/docs/crictl.md
- https://www.mankier.com/1/kubeadm-init