Set up a K8s cluster: multi-node Kubernetes cluster using the `kubeadm` tool

multi-node Kubernetes cluster using the `kubeadm` tool

1. Installing kubeadm: 사전 준비 : All Nodes

- system 구성 확인 및 설정

```
# 2 GB or more of RAM per machine (any less will leave little room for your apps).
# 2 CPUs or more.
lscpu
free -h

# Full network connectivity between all machines in the cluster (public or private network is fine).
# Unique hostname, MAC address, and product_uuid for every node. See here for more details.
hostname
ip addr show dev ens32 | grep inet
cat /etc/hosts
...
10.0.20.50   master
10.0.20.51   node1
10.0.20.52   node2


# Certain ports are open on your machines. See here for more details.
# ufw status
# systemctl disable ufw.service
# systemctl stop ufw.service
# systemctl status ufw

# Swap disabled. You MUST disable swap in order for the kubelet to work properly
swapoff -a && sed -i '/swap/s/^/#/' /etc/fstab
```

- **IPv4를 포워딩하여 iptables가 bridge된 traffic을 보게 하기**

```
# br_netfilter 모듈을 로드
cat <<EOF | sudo tee /etc/modules-load.d/k8s.conf
overlay
br_netfilter
EOF

modprobe overlay
modprobe br_netfilter

# bridge taffic 보게 커널 파라메터 수정
# 필요한 sysctl 파라미터를 /etc/sysctl.d/conf 파일에 설정하면, 재부팅 후에도 값이 유지된다.
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

# 재부팅하지 않고 sysctl 파라미터 적용하기
sysctl --system
```

2. Installing a container runtime : All Nodes

- 참고: https://kubernetes.io/ko/docs/setup/production-environment/tools/kubeadm/install-kubeadm/#installing-runtime
- 컨테이너 런타임은 컨테이너 실행을 담당하는 소프트웨어

▼ docker install

```
apt update
apt install -y ca-certificates curl gnupg lsb-release

curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg
echo "deb [arch=$(dpkg --print-architecture) signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] https://download.docke
apt update
apt install -y docker-ce docker-ce-cli containerd.io docker-compose-plugin

docker -v

systemctl status docker

usermod -a -G docker user
id user
```

- cre-docker install.

  cre-docker 설치 메뉴얼

  ▼ 설치

```
wget https://github.com/Mirantis/cri-dockerd/releases/download/v0.2.5/cri-dockerd-0.2.5.amd64.tgz

tar -xf cri-dockerd-0.2.5.amd64.tgz
cp cri-dockerd/cri-dockerd /usr/bin/
chmod +x /usr/bin/cri-dockerd

# systemctl 데몬 시작파일 구성
cat <<"EOF" > /usr/lib/systemd/system/cri-docker.service
[Unit]
Description=CRI Interface for Docker Application Container Engine
Documentation=https://docs.mirantis.com
After=network-online.target firewalld.service docker.service
Wants=network-online.target
Requires=cri-docker.socket

[Service]
Type=notify

ExecStart=/usr/bin/cri-dockerd --network-plugin=cni --pod-infra-container-image=registry.aliyuncs.com/google_containers/pause

ExecReload=/bin/kill -s HUP $MAINPID
TimeoutSec=0
RestartSec=2
Restart=always

StartLimitBurst=3

StartLimitInterval=60s

LimitNOFILE=infinity
LimitNPROC=infinity
LimitCORE=infinity

TasksMax=infinity
Delegate=yes
KillMode=process

[Install]
WantedBy=multi-user.target

EOF

# socket 파일 생성
cat <<"EOF" > /usr/lib/systemd/system/cri-docker.socket
[Unit]
Description=CRI Docker Socket for the API
PartOf=cri-docker.service

[Socket]
ListenStream=%t/cri-dockerd.sock
SocketMode=0660
SocketUser=root
SocketGroup=docker

[Install]
WantedBy=sockets.target

EOF

#cri-docker 시작
systemctl daemon-reload
```

```
systemctl enable --now cri-docker
systemctl status cri-docker
...
<CTRL>+<c>
```

3. Installing kubeadm : ALL

- 모든 머신에 다음 패키지들을 설치한다.
  - `kubeadm`: 클러스터를 부트스트랩하는 명령이다.
  - `kubelet`: 클러스터의 모든 머신에서 실행되는 파드와 컨테이너 시작과 같은 작업을 수행하는 컴포넌트이다.
  - `kubectl`: 클러스터와 통신하기 위한 커맨드 라인 유틸리티이다.

▼ **Installing kubeadm, kubelet and kubectl**

```
# Update the apt package index and install packages needed to use the Kubernetes apt repository:
apt-get update
apt-get install -y apt-transport-https ca-certificates curl

# Download the Google Cloud public signing key:
curl -fsSLo /usr/share/keyrings/kubernetes-archive-keyring.gpg https://packages.cloud.google.com/apt/doc/apt-key.gpg

# Add the Kubernetes apt repository:
echo "deb [signed-by=/usr/share/keyrings/kubernetes-archive-keyring.gpg] https://apt.kubernetes.io/ kubernetes-xenial main" | sudo

# Update apt package index, install kubelet, kubeadm and kubectl, and pin their version:
apt-get update
apt-get install -y kubelet kubeadm kubectl
apt-mark hold kubelet kubeadm kubectl
```

4. Creating a cluster with kubeadm

- **Initializing your control-plane node**

```
# control-plaine 컴포넌트 구성
kubeadm init --pod-network-cidr=192.168.0.0/16 --cri-socket unix:///var/run/cri-dockerd.sock
...
[addons] Applied essential addon: kube-proxy

Your Kubernetes control-plane has initialized successfully!


# Kubectl을 명령 실행 허용하려면 kubeadm init 명령의 실행결과 나온 내용을 동작해야 함
mkdir -p $HOME/.kube
cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
chown $(id -u):$(id -g) $HOME/.kube/config


# root 권한으로 worker node join을 위한 명령어. 별도 저장해둠.
cat > token.join
kubeadm join 10.0.2.50:6443 --token 2whvdj...qbbib \
  --discovery-token-ca-cert-hash sha256:7125...78570b57
<CTRL>+<d>
```

- **Installing a Pod network add-on**
  - 참조: https://kubernetes.io/docs/concepts/cluster-administration/networking/#how-to-implement-the-kubernetes-networking-model
  - **Calico :** https://projectcalico.docs.tigera.io/getting-started/kubernetes/quickstart

```
# CNI(컨테이너 네트워크 인터페이스) 기반 Pod 네트워크 추가 기능을 배포해야 Pod가 서로 통신할 수 있습니다. 네트워크를 설치하기 전에 클러스터 DNS(CoreDNS)가 시
kubectl get nodes
NAME       STATUS     ROLES           AGE     VERSION
master     NotReady   control-plane   7m31s   v1.24.3
```

```
# Pod 네트워크가 호스트 네트워크와 겹치지 않도록 주의해야함.
# Calico 설치
kubectl create -f https://projectcalico.docs.tigera.io/manifests/tigera-operator.yaml
kubectl create -f https://projectcalico.docs.tigera.io/manifests/custom-resources.yaml

# node 초기화 될때까지 기다림
watch kubectl get pods -n calico-system

kubectl get pods -n calico-system
NAME                                       READY   STATUS    RESTARTS   AGE
calico-kube-controllers-657d56796-nrxbx   0/1     Pending   0          75s
calico-node-w2n8f                          1/1     Running   0          75s
calico-typha-5d9f7ffbf4-mx846              1/1     Running   0          75s


# 다시 확인
kubectl get nodes
NAME                 STATUS   ROLES           AGE   VERSION
master.example.com   Ready    control-plane   15m   v1.24.3
```

5. Joining nodes

- 노드는 워크로드(컨테이너 및 포드 등)가 실행되는 곳입니다. 클러스터에 새 노드를 추가하려면 각 시스템에 대해 다음을 수행합니다.
  - SSH to the machine
  - Become root (e.g. `sudo su -`)
  - Install a runtime if needed
  - Run the command that was output by `kubeadm init`

```
kubeadm join 10.0.2.50:6443 --token 2whv...bbib --discovery-token-ca-cert-hash sha256:712538d8a5a...6a5078570b57 --cri-socket unix
```

6. 기본 구성

- kubectl 명령어 자동 완성 설정
- https://kubernetes.io/docs/reference/kubectl/cheatsheet/

```
apt-get install bash-completion -y
source <(kubectl completion bash) # setup autocomplete in bash into the current shell, bash-completion package should be installed
echo "source <(kubectl completion bash)" >> ~/.bashrc # add autocomplete permanently to your bash shell.
```

- user 사용자가 kubectl 명령 실행 가능하도록 설정

```
mkdir -p ~user/.kube
cp -i /etc/kubernetes/admin.conf ~user/.kube/config
chown user:user ~user/.kube/config
```