2022.12.07 CKA(Certified Kubernetes Administrator)

2022.06.25 (PSI Secure Browser)

URL:

https://training.linuxfoundation.org/bridge-migration-2021/

- 30

Exam

Exam Preparation Checklist

Agree to Global Candidate Agreement Read Now

Verify Name Status: Done

Select Platform Platform: Ubuntu 20.04

Schedule an Exam

Exam Date:

December 07, 2022 - 01:30PM

Asia/Seoul

Check System Requirements Status: System Requirements Checked

Get Candidate Handbook Read Now

Read the Important Instructions Read the Important Instructions

Take Exam

CKA Schedule 30

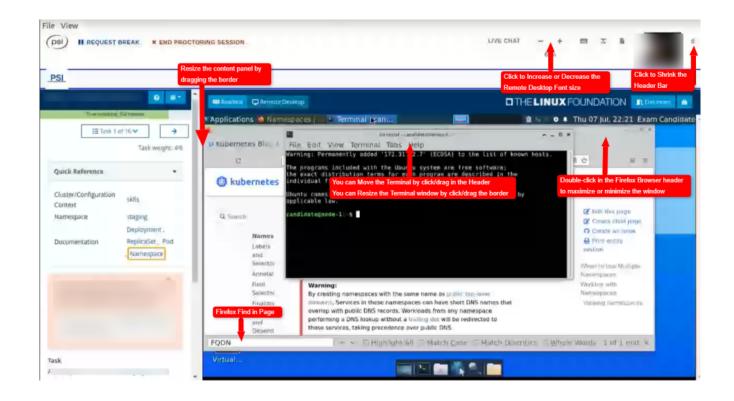
URL
https://trainingportal.linuxfoundation.org/learn/course/certified-kubernetes-administrator-cka/exam/exam

TAKE EXAM PSI
Cotana, (zoon, bluejean), VPN 7

CHAT

#
- ()
(, HDMI
CHAT)
- , (/)
-

가



14

Flag

URL:

https://docs.linuxfoundation.org/tc-docs/certification/tips-ck a-and-ckad#adjusting-font-and-windows-in-the-examui

&

가 Firefox Kubernetes.io Document

YAML

Ctrl+Shift+C , V 가 Copy&Paste &

PSI 가

Firefox

CHAT 가 가 가

100 66 (2/3) **Pass** 24 24 가

Fail Reschedule

Flag

가 /

Kubernetes 1.24 + cri-docker Installation (kubeadm)

Kubernetes

18.04

Controll Node (node01), Worker Node (node02, node03) 3EA root 가 user(worker) sudo

(Pre-Option)

```
sudo kill -9 $(lsof -t /var/lib/dpkg/lock)
sudo kill -9 $(lsof -t /var/lib/apt/lists/lock)
sudo kill -9 $(lsof -t /var/cache/apt/archives/lock)
sudo rm /var/lib/apt/lists/lock
sudo rm /var/cache/apt/archives/lock
sudo rm /var/lib/dpkg/lock
sudo dpkg --configure -a
```

(Require) SWAPOFF

```
# SWAP-On 가
sudo swapoff /swap.img
sudo sed -i -e '/swap.img/d' /etc/fstab
```

() Container Runtime Install

```
k8s 1.24 ( 2022/05 )
k8s dockershim
cri-docker 가 k8s
```

Pre-Setting - All node

```
curl -fsSL https://get.docker.com -o get-docker.sh
sudo sh get-docker.sh
sudo systemctl enable --now docker && sudo systemctl status
docker --no-pager
sudo usermod -aG docker worker
sudo docker container ls
# cri-docker Install
VER=$(curl
https://api.github.com/repos/Mirantis/cri-dockerd/releases/lat
est|grep tag_name | cut -d '"' -f 4|sed 's/v//g')
echo $VER
waet
https://github.com/Mirantis/cri-dockerd/releases/download/v${V
ER}/cri-dockerd-${VER}.amd64.tgz
tar xvf cri-dockerd-${VER}.amd64.tgz
sudo mv cri-dockerd/cri-dockerd /usr/local/bin/
```

```
# cri-docker Version Check
cri-dockerd --version
waet
https://raw.githubusercontent.com/Mirantis/cri-dockerd/master/
packaging/systemd/cri-docker.service
wget
https://raw.githubusercontent.com/Mirantis/cri-dockerd/master/
packaging/systemd/cri-docker.socket
               cri-docker.socket cri-docker.service
sudo
        mν
/etc/systemd/system/
sudo sed -i -e 's,/usr/bin/cri-dockerd,/usr/local/bin/cri-
dockerd,' /etc/systemd/system/cri-docker.service
sudo systemctl daemon-reload
sudo systemctl enable cri-docker.service
sudo systemctl enable --now cri-docker.socket
# cri-docker Active Check
sudo systemctl restart docker && sudo systemctl restart cri-
docker
sudo systemctl status cri-docker.socket --no-pager
# Docker cgroup Change Require to Systemd
sudo mkdir /etc/docker
cat <<EOF | sudo tee /etc/docker/daemon.json</pre>
{
  "exec-opts": ["native.cgroupdriver=systemd"],
  "log-driver": "json-file",
  "log-opts": {
    "max-size": "100m"
  },
  "storage-driver": "overlay2"
}
E0F
sudo systemctl restart docker && sudo systemctl restart cri-
docker
sudo docker info | grep Cgroup
# Kernel Forwarding
```

```
cat <<EOF | sudo tee /etc/modules-load.d/k8s.conf
br netfilter
E0F
cat <<EOF | sudo tee /etc/sysctl.d/k8s.conf
net.bridge.bridge-nf-call-ip6tables = 1
net.bridge.bridge-nf-call-iptables = 1
E0F
sudo sysctl --system
Packages Install - All node
sudo apt-get update
sudo apt-get install -y apt-transport-https ca-certificates
curl
    curl -fsSLo /usr/share/keyrings/kubernetes-archive-
sudo
keyring.gpg
https://packages.cloud.google.com/apt/doc/apt-key.gpg
echo "deb [signed-by=/usr/share/keyrings/kubernetes-archive-
keyring.gpg] https://apt.kubernetes.io/ kubernetes-xenial
main" | sudo tee /etc/apt/sources.list.d/kubernetes.list
# Update
sudo apt-get update
# k8s
sudo apt-get install -y kubelet kubeadm kubectl
#
kubectl version --short
Client Version: v1.24.3
```

Kustomize Version: v4.5.4

sudo apt-mark hold kubelet kubeadm kubectl

#

k8s Init - Controller Node (Node01)

```
# Controller Node
sudo kubeadm config images pull --cri-socket unix:///run/cri-
dockerd.sock
sudo kubeadm init --ignore-preflight-errors=all --pod-network-
cidr=192.168.0.0/16
                               --apiserver-advertise-
address=211.115.207.136 --cri-socket /var/run/cri-dockerd.sock
## --apiserver-advertise-address=203.248.23.161 -> Controller
    IP.
               join Command
                      211.115.207.136:6443
kubeadm
             ioin
                                                   --token
r8qfco.6s7f60dns4vgwcc0 \
                        --discovery-token-ca-cert-hash
sha256:7b0f82be076748e67f8615eab0b86a61317bac397f94b2921810231
ab14afdcc
# kubeadm root
                                가
mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
# Token
kubeadm token create --print-join-command
           ( Ready , Running )
kubectl get nodes -o wide
kubectl get pod -A
# Worker Node
                kubectl
                                  admin.conf
sudo
                            /etc/kubernetes/admin.conf
               SCD
worker@node02:/home/worker/admin.conf
                            /etc/kubernetes/admin.conf
sudo
               SCD
worker@node03:/home/worker/admin.conf
```

Node Join - Worker Node (Node02, Node03)

```
# Worker Node Join. --cri-docker sorket init
```

```
kubeadm join --token <token> <controlplane-
sudo
host>:<controlplane-port> --ignore-preflight-errors=all --cri-
socket unix:///var/run/cri-dockerd.sock
   sudo
         kubeadm
                   join [Controller
                                        IP1:6443 --token
3nvrgw.33k750dq9klm5omi --discovery-token-ca-cert-hash
sha256:b10158bcea37aae0e92ed6b68b4dd1e8213623cc7d406e77eef55fe
6196fe346 --cri-socket /var/run/cri-dockerd.sock
# Join
sudo kubeadm join [Controller IP]:6443 --token
r8qfco.6s7f60dns4vqwcc0 \
                        --discovery-token-ca-cert-hash
sha256:7b0f82be076748e67f8615eab0b86a61317bac397f94b2921810231
ab14afdcc --cri-socket /var/run/cri-dockerd.sock
# Check
systemctl status kubelet
# worker
          kubectl
                                 가
cd ~
mkdir -p $HOME/.kube
sudo cp -i ./admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
 가
      node
                kubectl get nodes -o wide
Pod Network Install - Controller Node (Node01)
             CNI Plugin
                                      CoreDNS 가 Pending
#
         kube-system-proxy
# CNI Mode
# Check - kube-proxy
                              . kube-proxy
                                            Node
kubectl get pod -A
kubectl logs -f pod/kube-proxy-[name] -n kube-system
             - Linux
                           iptables Mode
# Log
kubectl logs -f pod/kube-proxy-6dqp8 -n kube-system | grep
mode
I0805 06:32:48.444077
                           1 server others.go:578] "Unknown
```

```
proxy mode, assuming iptables proxy" proxyMode=""
I0805 06:32:48.469028
                               1 server others.go:213] "kube-
proxy running in dual-stack mod
" ipFamily=IPv4
## Pod Network Plugin Install
                                   CKA
                                                    Callico
Plugin
Calico Network IPIP , VXLAN
                                         IPVS
kubectl
                               create
                                                             - f
https://projectcalico.docs.tigera.io/manifests/tigera-operator
.yaml
curl
https://projectcalico.docs.tigera.io/manifests/custom-resource
s.yaml -0
kubectl create -f custom-resources.yaml
kubectl get pods -A
## Control-Plane ( Controller ) Node
                                            Taint (
            calico-controller-pod가 Pending
   )
## Taint Check
kubectl describe node node01 | grep Taints
kubectl taint nodes --all node-role.kubernetes.io/master-
kubectl taint nodes node01 node-role.kubernetes.io/control-
plane: NoSchedule-
## PoD check
kubectl get pod -A
#kubectl describe pods/calico-kube-controllers-657d56796-xxxxx
-n calico-system
## Calicoctl
               Status
cd /usr/local/bin
sudo
                              curl
https://github.com/projectcalico/calico/releases/download/v3.2
3.3/calicoctl-linux-amd64 -o calicoctl
sudo chmod +x calicoctl
```

```
## CNI Type Check
calicoctl get ippool -o wide

## Block Check
sudo calicoctl ipam show --show-blocks

## BGP Protocol Check
sudo calicoctl node status

## Node Endpoint Check
calicoctl get workloadendpoint -A
```

Rejoin or Reset

(Trouble)

```
# All Node
sudo systemctl stop kubelet
sudo kubeadm reset -f --cri-socket /var/run/cri-dockerd.sock
sudo rm -rf ~/.kube
sudo rm -rf /root/.kube
sudo rm -rf /var/lib/etcd
sudo rm -rf /etc/kubernetes
```

Calico Network

```
# ALL Node
kubectl delete -f custom-resources.yaml
kubectl delete -f tigera-operator.yaml
sudo rm -rf /var/run/calico/
sudo rm -rf /var/lib/calico/
sudo rm -rf /etc/cni/net.d/
sudo rm -rf /var/lib/cni/
sudo rm -rf /opt/cni
sudo reboot
```

```
CRI-O
             가
CRI-O?
# CRI-0 OCI
                                      (CRI) .
가
                           Docker
                                       Kubernetes 1.24
Docker CRI(
                                    ) shim
 (Docker Engine ) CRI-0
                                가
        ( VM )
Controller Server : 1EA
Worker Server : 1EA
05
Ubuntu 20.04 Server Minimal
https://github.com/cri-o/cri-o/blob/main/install.md#supported-
versions
                : https://tech.hostway.co.kr/2022/02/06/418/
Traffic Setup
# .conf
# Controller / Worker
cat <<EOF | sudo tee /etc/modules-load.d/crio.conf</pre>
overlay
br netfilter
E0F
sudo modprobe overlay
```

```
sudo modprobe br netfilter
#
         sysctl
# Controller / Worker
cat <<EOF | sudo tee /etc/sysctl.d/99-kubernetes-cri.conf
net.bridge.bridge-nf-call-iptables = 1
net.ipv4.ip forward
net.bridge.bridge-nf-call-ip6tables = 1
E0F
sudo sysctl --system
cgroup driver
# CRI-0
                  systemd cgroup
                                 , /etc/crio/crio.conf
cgroupfs cgroup
     /etc/crio/crio.conf.d/02-cgroup-manager.conf
 (drop-in)
[crio.runtime]
conmon cgroup = "pod"
cgroup manager = "cgroupfs"
CRI-O
# Controller / Worker
sudo -i
export OS=xUbuntu 20.04 # OS
export VERSION=1.19 # cri-o
                                                          "deb
echo
https://download.opensuse.org/repositories/devel:/kubic:/libco
ntainers:/stable/$0S/
/etc/apt/sources.list.d/devel:kubic:libcontainers:stable.list
echo
http://download.opensuse.org/repositories/devel:/kubic:/libcon
```

```
o:$VERSION.list
curl
                                                            - L
https://download.opensuse.org/repositories/devel:kubic:libcont
ainers:stable:cri-o:$VERSION/$OS/Release.key | apt-key add -
curl
https://download.opensuse.org/repositories/devel:/kubic:/libco
ntainers:/stable/$0S/Release.key | apt-key add -
sudo apt-get update
sudo apt-get install cri-o cri-o-runc
sudo systemctl daemon-reload
sudo systemctl enable crio --now
sudo systemctl status crio
Controller Node Initialize
# docker
                                                          CRT-
                                     1
0
# Controller
kubeadm init --cri-socket=/var/run/crio/crio.sock --ignore-
preflight-errors=all --pod-network-cidr=192.168.0.0/16
apiserver-advertise-address=203.248.23.192
# Worker
kubeadm
                        203.248.23.192:6443
              join
                                                     --token
9oy7rn.qtw04nfd8ga417p4
                             --discovery-token-ca-cert-hash
sha256:26218e0c80320b7c23735916c130fc48f644b26314212d917969553
ec0651256
```

/etc/apt/sources.list.d/devel:kubic:libcontainers:stable:cri-

tainers:/stable:/cri-o:/\$VERSION/\$0S/

[Network] K8S Overlay Network (IPIP -> VXLAN)

K8S Overlay Network

```
IPIP -> VXIAN
                                     POD가
   )
( pod
                                              가
                                                    )
Calico IP-IP Network
                              VXI.AN
Node: Controller/Worker01/Worker02
## Controller
# Mode
             IPIPMODE
calicoctl get ippool -o wide
NAME
                         CIDR
                                                     IPIPMODE
                                             NAT
VXLANMODE
             DISABLED
                         DISABLEBGPEXPORT
                                               SELECTOR.
default-ipv4-ippool
                      192.168.0.0/16
                                       true
                                              Always
                                                          Never
           false
false
 all()
      Manifest YAML
kubectl delete -f calico.yml
## Contoller / Worker
# 가
                       tunl0
                                     가
sudo rm -rf /var/run/calico/
sudo rm -rf /var/lib/calico/
sudo rm -rf /etc/cni/net.d/
sudo rm -rf /var/lib/cni/
sudo reboot
```

```
## Controller
# Manifest. calico.yaml
                       VXLAN
 livenessProbe:
           exec:
             command:
             - /bin/calico-node
             - -felix-live
             # - -bird-live
                            // VXLAN bird(BGP)
           periodSeconds: 10
            initialDelaySeconds: 10
           failureThreshold: 6
           timeoutSeconds: 10
          readinessProbe:
           exec:
             command:
              - /bin/calico-node
             - -felix-ready
            # - -bird-ready //
           # Enable IPIP
            - name: CALICO IPV4POOL IPIP
             value: "Never" // Always --> Never
           # Enable or Disable VXLAN on the default IP pool.
            - name: CALICO_IPV4P00L_VXLAN
             value: "Always"
                                     // Never --> Always
kind: ConfigMap
apiVersion: v1
metadata:
  name: calico-config
  namespace: kube-system
data:
  # Typha is disabled.
  typha service name: "none"
  # Configure the backend to use.
  calico backend: "vxlan"
                                 // "bird" --> "vxlan"
```

```
#
kubectl apply -f calico.yaml
# Calico Node
                      Readv
kubectl get nodes -o wide -A
# Calico Pod
                . kube-system PoD
                                        가
kubectl get pod -o wide -A
# Calico Type . BIRD
sudo calicoctl node status
Calico process is running.
The BGP backend process (BIRD) is not running.
# Network
              VXLANMODE 가
calicoctl get ippool -o wide
NAME
                         CIDR
                                             NAT
                                                     IPIPMODE
VXLANMODE
                          DISABLEBGPEXPORT
             DISABLED
                                               SELECTOR
default-ipv4-ippool
                           192.168.0.0/16
                                                true
                                                         Never
                             false
Always
               false
 all()
            tunl0
                          가
                                                   가
                                    vxlan
#
                                               가
        vxlan
#
hostway@controller:~$ route -n
Kernel IP routing table
Destination
                                 Genmask
                 Gateway
                                                   Flags Metric
     Use Iface
Ref
0.0.0.0
                                0.0.0.0
                                                 UG
                10.10.10.1
                                                       0
                                                              0
0 ens18
10.10.10.0
                0.0.0.0
                                255.255.255.0
                                                 U
                                                       0
                                                              0
0 ens18 // External (SNAT)
172.17.0.0
                0.0.0.0
                                255.255.0.0
                                                 U
                                                       0
                                                              0
0 docker0 // Container Runtime Bridge
192.168.5.0
                192.168.5.0
                                255.255.255.192 UG
                                                       0
                                                              0
0 vxlan.calico
                // Worker01
192.168.30.64
                192.168.30.64
```

0 vxlan.calico

192.168.49.0

192.168.49.1

0 *

// Worker02

// Controller

0.0.0.0

0.0.0.0

255.255.255.192 UG

255.255.255.192 U

255.255.255.255 UH

vxlan

0

0

0

0

0

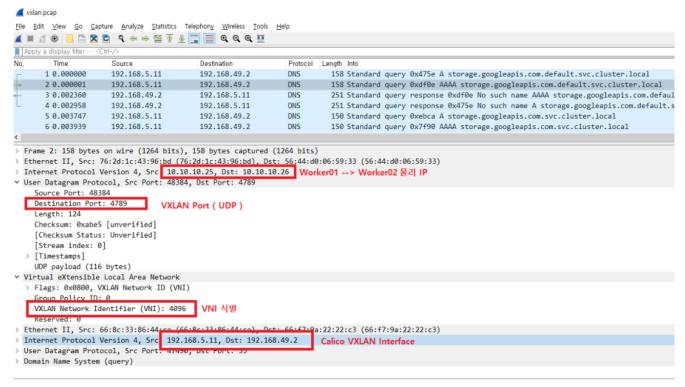
0

```
0 cali09ae4a7064b
                    // Node(Worker01)가
                                            GW
192.168.49.2
                                255.255.255.255 UH
                0.0.0.0
                                                      0
                                                             0
0 cali1fdac863dc5
                 // Node(Worker02)가
                                           GW
# Worker
hostway@controller:~$ ip nei | grep vxlan
              dev vxlan.calico lladdr
192.168.5.0
                                           66:8c:33:86:44:ce
PERMANENT
192.168.30.64 dev vxlan.calico lladdr 66:fb:72:20:22:a1
PERMANENT
# VXLAN Traffic Port
                       UDP
                  0 0.0.0.0:4789
                                            0.0.0.0:*
           0
udp
# PoD
hostway@controller:~$ kubectl create deployment sampleos
image=gcr.io/google-samples/kubernetes-bootcamp:v1
replicas=3
deployment.apps/sampleos created
hostway@controller:~$ kubectl get pod -o wide
NAME
                             READY
                                     STATUS
                                                RESTARTS
                                                           AGE
ΙP
                           NOMINATED NODE
                                            READINESS GATES
                NODE
sampleos-646dc9654b-8xjw9
                             1/1
                                     Running
                                                           45s
                                                0
192.168.5.11
               worker01
                           <none>
                                            <none>
sampleos-646dc9654b-gxn75
                             1/1
                                     Running
                                                0
                                                           45s
192.168.5.10
               worker01
                                            <none>
                           <none>
sampleos-646dc9654b-snkxg
                                     Runnina
                                                           45s
                             1/1
192.168.30.75
               worker02
                           <none>
                                            <none>
# VXLAN
// Controller
1) worker01
              worker02
                                 POD
                                       Ping
hostway@controller:~$
                                kubectl
                                                           -it
                                               exec
sampleos-646dc9654b-8xjw9 -- ping 192.168.30.75
PING 192.168.30.75: 56 data bytes
64 bytes from 192.168.30.75: icmp seq=0 ttl=115 time=92.124 ms
64 bytes from 192.168.30.75: icmp seq=1 ttl=115 time=79.735 ms
64 bytes from 192.168.30.75: icmp seg=2 ttl=115 time=79.233 ms
```

2) tcpdump

sudo tcpdump -i ens18 -w vxlan.pcap

3) Wireshark . UDP



[] CentOS 7 Kubernetes Install

CentOS 7 Kubernetes

OS : CentOS 7.6.1810 Minimal

Account : root - SNAT IP

Controller: 10.10.10.237 SSH:4223 Worker-01: 10.10.10.204 SSH:4224 Worker-02: 10.10.10.190 SSH:4225

```
# root
                  . sudo
useradd -d /home/username username
echo "password" | passwd username --stdin
#
           su
chmod 700 /usr/bin/su
# sudoer
               wheel
                                가
sed -ie '/wheel/s/$/\:username/' /etc/group
# Timezone
sudo timedatectl set-timezone Asia/Seoul
# SWAP OFF
sudo swapoff -a
sudo sed -i -e '/swap/d' /etc/fstab
# firewalld off
sudo systemctl stop firewalld && sudo systemctl disable
firewalld
# Selinux
setenforce 0
sudo sed -i 's/SELINUX=enforcing/SELINUX=disabled/g'
/etc/selinux/config
# Hostname
sudo hostnamectl set-hostname controller
sudo hostnamectl set-hostname worker-01
sudo hostnamectl set-hostname worker-02
## Controller / Worker
#curl -s https://get.docker.com | sudo sh
curl -fsSL https://get.docker.com -o get-docker.sh
sudo sh get-docker.sh
## Check
sudo docker -v
sudo docker ps -a
```

```
## Controller / Worker
sudo mkdir /etc/docker
cat <<EOF | sudo tee /etc/docker/daemon.json</pre>
{
  "exec-opts": ["native.cgroupdriver=systemd"],
  "log-driver": "json-file",
  "log-opts": {
    "max-size": "100m"
  "storage-driver": "overlay2"
}
E0F
## Docker enable && restart
sudo systemctl enable docker
sudo systemctl daemon-reload
sudo systemctl restart docker
## Packages Repo
sudo cat <<EOF | sudo tee /etc/yum.repos.d/kubernetes.repo</pre>
[kubernetes]
name=Kubernetes
baseurl=https://packages.cloud.google.com/yum/repos/kubernetes
-el7-x86 64
enabled=1
gpgkey=https://packages.cloud.google.com/yum/doc/yum-key.gpg
https://packages.cloud.google.com/yum/doc/rpm-package-key.gpg
E0F
## Install
            install -v kubelet kubeadm
                                                 kubectl
sudo vum
disableexcludes=kubernetes
```

Controller Init

```
# Regular User Privileges
  mkdir -p $HOME/.kube
  sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
  sudo chown $(id -u):$(id -g) $HOME/.kube/config
# Network Plugin Setting ( Calico )
curl
https://projectcalico.docs.tigera.io/manifests/calico.yaml -0
kubectl apply -f calico.yaml
# System Namespace ( kube-system ) check. CoreDNS 가
kubectl get pods -o wide -A
NAMESPACE
              NAME
                                                        READY
STATUS
            RESTARTS
                          AGE
                                     ΙP
                                                         NODE
NOMINATED NODE
                  READINESS GATES
kube-system
               calico-kube-controllers-7c845d499-p85pm
                                                           1/1
                                 192.168.49.3
                        3m6s
Running
                                                  controller
           0
<none>
                  <none>
kube-system
              calico-node-fnm2q
                                                           1/1
                                 10.10.10.237
                        3m6s
                                                  controller
Running
           0
<none>
                  <none>
              coredns-64897985d-cgvml
kube-system
                                                           1/1
                        5m41s
                                 192.168.49.2
                                                  controller
Running
           0
<none>
                  <none>
kube-system
              coredns-64897985d-vdckf
                                                           1/1
                                 192.168.49.1
                                                  controller
Running
                        5m42s
           0
<none>
                  <none>
```

1/1

1/1

1/1

1/1

1/1

controller

controller

controller

controller

etcd-controller

<none>

<none>

<none>

<none>

kube-proxy-nn5zn

5m54s

kube-apiserver-controller

kube-controller-manager-controller

5m54s

5m42s

kube-scheduler-controller

6 m

10.10.10.237

10.10.10.237

10.10.10.237

10.10.10.237

kube-system

kube-system

kube-system

kube-system

kube-system

0

0

0

Running

Running

Running

Running <none>

<none>

<none>

<none>

```
10.10.10.237 controller
Running
           0
                       5m54s
<none>
                  <none>
     ) Multi NIC
                   가
                               INTERNAL-IP
# (
            가
                                K8S
                                               NIC
                                                      IP 가
INTERNAL-IP
   INTERNAL-IP
                 Init
kubeadm --apiserver-advertise-address
                                             ΙP
cat << EOF | sudo tee /etc/default/kubelet
KUBELET EXTRA ARGS='--node-ip $(hostname -I | cut -d ' ' -f2)'
E0F
sudo systemctl daemon-reload
sudo systemctl restart kubelet
kubectl cluster-info
Worker Join
# Worker-01 Woker-02 Node
                             User Privileges
sudo
                           /etc/kubernetes//admin.conf
              scp
username@10.10.10.204:/home/username/admin.conf
                           /etc/kubernetes//admin.conf
sudo
              scp
username@10.10.10.190:/home/username/admin.conf
# Worker
mkdir -p $HOME/.kube
sudo cp -i ./admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
#
    Worker
               kubeadm
                                Join
                            10.10.10.237:6443 -- token
sudo
        kubeadm
                   join
jgocer.fu65ql39kdod5qi0 \
                        --discovery-token-ca-cert-hash
sha256:3cb85267e89913d7865d219922daaa8fc6e788dd2be0e2f80fae271
76e2dfe3b
kubeadm token create --print-join-command
```

```
# Check
kubectl get nodes -o wide
            STATUS
                      ROLES
                                            AGE
                                                  VERSION
INTERNAL-IP
              EXTERNAL-IP
                                                  KERNEL-
                           OS - IMAGE
                CONTAINER-RUNTIME
VERSION
controller
            Ready control-plane, master
                                            16m
                                                 v1.23.5
10.10.10.237 <none>
                                 CentOS Linux 7 (Core)
3.10.0-1062.el7.x86 64 docker://20.10.14
worker-01
            Ready
                                            55s
                                                  v1.23.5
                      <none>
10.10.10.204
                                 CentOS Linux 7 (Core)
                <none>
3.10.0-1062.el7.x86 64 docker://20.10.14
worker-02
         NotReady <none>
                                            38s
                                                 v1.23.5
10.10.10.190
              <none>
                                 CentOS Linux 7 (Core)
3.10.0-1062.el7.x86 64 docker://20.10.14
# Check Pod Create
kubectl run hello --image=nginx --dry-run=client -o yaml |
kubectl apply -f-
pod/hello created
[myungin.baek@controller ~]$ kubectl get pods -o wide
     RFADY
                                 RESTARTS
                    STATUS
                                              AGF
                                                      ΤP
             NOMINATED NODE
                               READINESS GATES
NODE
                                           192.168.171.1
hello 1/1
                                     42s
                Running
                         0
worker-01
          <none>
                             <none>
```

[] CNI - Calico Plugin

: CNI Calico Network

#1 (controller, worker)

CNI (Container Network Interface)

CNCF .

```
Plugin
Calico Network?
                  (L3)
vRouter
Kubernetes
                    Network
                                        CNI
                                                       Network
Plugin
                             Document
                                                 URL
https://projectcalico.docs.tigera.io/reference/
Non-overlay
# Direct
- BGP(Border Gateway Protocol) BIRD
  Pod
              Pod
                                                   가
                Calico Pod
  Node
                             BGP
                                           Peer
                                                  ( ex:
Overlay Network
Workload IP( ex:
                              )
                            (Encaptulation)
                                                           (L2)
                           Node
                                  ΙP
                                         가
                                                       POD
      가
ΙP
# IP in IP
              (Default)
                          Direct
                     tunl0(tunneling)
ΙP
  가
                가
    Direct
                       BGP (BIRD)
          (IPVS)
Node
Calico Routing
```

Kubenet

CNI

Network

Kubernetes

```
# VXLAN
- 가
                               IP in IP
 . ( ex: Azure )
      Calico
                                                  가
                             BGP
VXLAN
              Node
L2
        UDP
IP in IP
                        가
# Cross-subnet
                                                   가
    가
                     가
                             )
       /
(
# WireGuard
Calico
                                              가
                 가
Calicoctl
Controller
             Calico Network
Host
                 kubectl plugin
# Host
$ cd /usr/local/bin
                                  curl
$
               sudo
https://github.com/projectcalico/calico/releases/download/v3.2
2.1/calicoctl-linux-amd64 -o calicoctl
$ sudo chmod +x calicoctl
# Check
Calico 가
              Network Pool Block
$ sudo calicoctl ipam show --show-blocks
----+
                              | IPS TOTAL | IPS IN USE |
| GROUPING |
             CIDR
IPS FREE
```

```
----+
| IP Pool | 192.168.0.0/16 | 65536 | 5 (0%)
65531 (100%) |
| Block
      | 192.168.136.0/26 |
                          64 | 4 (6%)
                                     l 60
(94\%)
     | 192.168.153.192/26 | 64 | 1 (2%)
                                      1 63
+-----
----+
BGP
$ sudo calicoctl node status
Calico process is running.
IPv4 BGP status
+----+---+---+
----+
| PEER ADDRESS | PEER TYPE | STATE | SINCE |
INF0
+-----+----+-----
----+
Established |
+----+---+----+
Block
$ route -n | egrep "tun|cali|\*"
192.168.136.0 0.0.0.0
                       255.255.255.192 U
                                        0
      0 *
192.168.136.1 0.0.0.0
                    255.255.255.255 UH
                                   0
                                        0
0 calibc6c3028870
192.168.136.2 0.0.0.0
                    255.255.255.255 UH
                                   0
                                        0
0 calid6edae09645
192.168.136.3 0.0.0.0
                    255.255.255.255 UH
                                        0
                                   0
0 calic6bfd11bfbe
192.168.153.192 203.248.23.215 255.255.255.192 UG
                                   0
                                        0
0 tunl0
  Pod가 calicxxxxxx
```

System(default) Namespace

가

- A

# colicectl got workloadendneint A								
<pre>\$ calicoctl get workloadendpoint -A NAMESPACE WORKLOAD NETWORKS INTERFACE</pre>	NODE							
kube-system calico-kube-controllers-56fcbf9d6b-nlqg2 controller 192.168.136.2/32 calid6edae09645								
	ıser-							
kube-system coredns-64897985d-vbpn4 controller 192.168.136.1/32 calibc6c3028870								
Calico Veth type(Pair) .								
\$ ip -br -c link show type veth								
calibc6c3028870@if3 UP ee:ee:ee:e	e:ee							
<pre><broadcast,multicast,up,lower_up></broadcast,multicast,up,lower_up></pre>								
calid6edae09645@if4 UP ee:ee:ee:e	e:ee							
<pre><broadcast, lower_up="" multicast,="" up,=""></broadcast,></pre>	0.00							
<pre>calic6bfd11bfbe@if4 UP</pre>	e:ee							
CDNOADCAST, MOETICAST, OF , LOWER_OF >								
Calico Management Pod								
Daemon Pod .								
Controller Worker Node Pod 가 .								
\$ kubectl get pods -o wide -n kube-system								
	ATUS							
RESTARTS AGE IP NODE calico-kube-controllers-56fcbf9d6b-nlqg2 1/1 Running	ı 0							
30m 192.168.136.2 user-controller	, 0							
calico-node-8cts6 1/1 Running	a 0							
30m 10.0.2.15 user-controller	•							
calico-node-mb9n6 1/1 Running	g 0							

Calico DB datastore etcd

user-worker

29m 10.0.2.15

\$ kubectl get pods -o wide -n kube-system | grep -i etcd etcd-user-controller 1/1 Running 0 user-controller 10.0.2.15 39m

Calico Felix

```
Pod kube-proxy
etcd Pod Network
kube-proxy 7 iptables / ipvs Mode
iptables ipvs

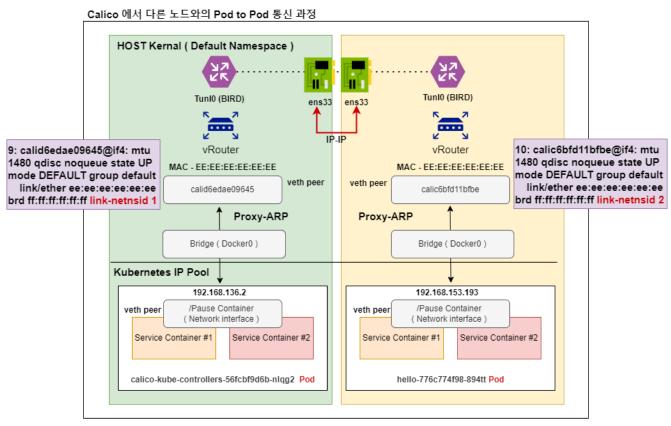
☐ IPVS = Hash

$ sudo iptables -t nat -S | grep -i cali
$ sudo iptables -t filter -S | grep -i cali
```

Networking

IP in IP Networking

Controller Node Worker Node Pod



Controller Node Worker Node

- 1) Controller 192.168.136.2 Pod Worker 192.168.153.193
 Pod
 2) Controller Pod (veth) Pair Host calico
 (veth) ARP
- 3) Host Calico Worker Pod

```
ARP
4)
        Calico
                          link-local (
                                                            )
    HOST 가 BIRD
                            Worker
5) Controller Calico
                                                ARP Proxy
                                 vRouter
         Worker
6) BIRD
         Tunl0 --> Host
                                                       Pod
   가
7)
             SNAT ( MASQUERADE )
                                            tunl0
Felix
H0ST
       ens33
Packet Check
             ( Controllor POD <---> Worker POD )
#
                                                    Ping
$ kubectl get pod -o wide
NAME
                         READY
                                 STATUS
                                           RESTARTS
                                                      AGE
                                                            ΙP
                 NOMINATED NODE
                                  READINESS GATES
NODE
hello-776c774f98-894tt 1/1
                                   Running
                                               0
                                                           13d
192.168.153.193
                 user-worker
                                   <none>
                                                    <none>
hί
                          1/1
                                   Running
                                               0
                                                           13d
192.168.136.5 user-controller
                                   <none>
                                                    <none>
# Worker POD --> Container POD.
                                         Ping
                                                        Pod
  Host
                   PTD
$ sudo nsenter -t 225201 -n ping 192.168.136.5
64 bytes from 192.168.136.5: icmp seq=627 ttl=62 time=0.709 ms
64 bytes from 192.168.136.5: icmp seq=628 ttl=62 time=0.675 ms
64 bytes from 192.168.136.5: icmp seq=629 ttl=62 time=0.727 ms
64 bytes from 192.168.136.5: icmp_seq=630 ttl=62 time=0.797 ms
64 bytes from 192.168.136.5: icmp seq=631 ttl=62 time=0.887 ms
                             . IPIP
# Controller
                                          API
            , API
$ sudo tcpdump -i enp0s8 -nn proto 4 -w test.pcap
# Wireshark
1) POD IP
                               ICMP
```

Apply a display filter ··· < Cttl-/>							
Vo.	Time	Course	Doctination	Protocol	Longth Info		
b	1 0.000000	192.168.153.193	192.168.136.5	ICMP	118 Echo (ping) request id=0x000b, seq=229/58624, ttl=63 (reply in 2)		
-	2 0.000217	192.168.136.5	192.168.153.193	ICMP	118 Echo (ping) reply id=0x000b, seq=229/58624, ttl=63 (request in 1)		
	3 1.001428	192.108.133.193	192.108.130.3	TCMF	тто сспо (ртпg) request та-охооов, seq-гоо/обобо, сст-оо (repту тп 4)		
	4 1.001611	192.168.136.5	192.168.153.193	ICMP	118 Echo (ping) reply id=0x000b, seq=230/58880, ttl=63 (request in 3)		
	5 2.002647	192.168.153.193	192.168.136.5	ICMP	118 Echo (ping) request id=0x000b, seq=231/59136, ttl=63 (reply in 6)		
	6 2.002801	192.168.136.5	192.168.153.193	ICMP	118 Echo (ping) reply id=0x000b, seq=231/59136, ttl=63 (request in 5)		
	7 2 027576	402 460 463 403	402 400 430 5	TCMD	440 5 1 / 1		
Fra	me 1: 118 bvte	es on wire (944 bits).	118 bytes captured (9	944 bits)			
Eth	ernet II, Src:	PcsCompu bc:85:3a (08	:00:27:bc:85:3a), Dst	t: PcsCompu	u 39:ce:bd (08:00:27:39:ce:bd)		
Internet Protocol Version 4, Src: 203.248.23.215, Dst: 203.248.23.214							
Int	Internet Protocol Version 4, Src: 192.168.153.193, Dst: 192.168.136.5						
Int	ernet Control	Message Protocol					

2) MAC Controller Worker Node API IP

Controller

3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq codel state UP group default qlen 1000

link/ether 08:00:27:39:ce:bd brd ff:ff:ff:ff:ff

inet 203.248.23.214/25 brd 203.248.23.255 scope global enp0s8

Worker

3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000

link/ether 08:00:27:bc:85:3a brd ff:ff:ff:ff:ff
inet 203.248.23.215/25 brd 203.248.23.255 scope global
enp0s8

3) IPv4 Protocol 2

Outer IP POD

Inner IP 2

```
No.
         Time
                       Source
                                              Destination
                                                                     |Protocol | Length | Info
       10.000000 192.168.153.193 192.168.136.5 ICMP 118 Echo (ping) request id=0x000b, seq=229/58624,

    2 0.000217
    192.168.136.5
    192.168.153.193

    3 1.001428
    192.168.153.193
    192.168.136.5

                                                                     ICMP
                                                                                118 Echo (ping) reply id=0x000b, seq=229/58624,
                                                                     ICMP
                                                                                118 Echo (ping) request id=0x000b, seq=230/58880,
       4 1.001611 192.168.136.5 192.168.153.193 ICMP 5 2.002647 192.168.153.193 192.168.136.5 ICMP 6 2.002801 192.168.136.5 192.168.153.193 ICMP
                                                                                118 Echo (ping) reply id=0x000b, seq=230/58880,
                                                                                 118 Echo (ping) request id=0x000b, seq=231/59136,
                                                                                118 Echo (ping) reply id=0x000b, seq=231/59136,
                                                                                                        > Frame 1: 118 bytes on wire (944 bits), 118 bytes captured (944 bits)
> Ethernet II, Src: PcsCompu_bc:85:3a (08:00:27:bc:85:3a) Dct: PcsCo
                                                                               39:ce:bd (08:00:27:39:ce:bd)
✓ Internet Protocol Version 4, Src: 203.248.23.215, Dst: 203.248.23.214
     0100 .... = Version: 4
     .... 0101 = Header Length: 20 bytes (5)
   > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
     Total Length: 104
     Identification: 0xf14e (61774)
   > Flags: 0x40, Don't fragment
     ...0 0000 0000 0000 = Fragment Offset: 0
     Time to Live: 63
    Protocol: IPIP (4)
     Header Checksum: 0x82a5 [validation disabled]
     [Header checksum status: Unverified]
     Source Address: 203.248.23.215
    Destination Address: 203.248.23.214
> Internet Protocol Version 4, Src: 192.168.153.193, Dst: 192.168.136.5
> Internet Control Message Protocol
```

Outer IP 가 InnerIP

IP-IP Protocol

	•								
No.	Time	Source	Destination	Protocol	Length Into				
_►	1 0.000000	192.168.153.193	192.168.136.5	ICMP	118 Echo	(ping) request	id=0x000k		
\vdash	2 0.000217	192.168.136.5	192.168.153.193	ICMP	118 Echo	(ping) reply	id=0x000k		
	3 1.001428	192.168.153.193	192.168.136.5	ICMP	118 Echo	(ping) request	id=0x000k		
	4 1.001611	192.168.136.5	192.168.153.193	ICMP	118 Echo	(ping) reply	id=0x000k		
	5 2.002647	192.168.153.193	192.168.136.5	ICMP	118 Echo	(ping) request	id=0x000k		
	6 2.002801	192.168.136.5	192.168.153.193	ICMP	118 Echo	(ping) reply	id=0x000k		
	7 7 007576	402 400 453 403	402 4C0 43C F	TCMD	440 5 1	/ • • • •			
> Fra	me 1: 118 bytes	on wire (944 bits),	118 bytes captured (9	944 bits)					
> Eth	> Ethernet II, Src: PcsCompu_bc:85:3a (08:00:27:bc:85:3a), Dst: PcsCompu_39:ce:bd (08:00:27:39:ce:bd)								
> Int	ernet Protocol	Version 4, Src: 203.2	48.23.215. Dst: 203.2	48.23.214	_				
∨ Int	ernet Protocol	Version 4, Src: 192.1	.68.153.193, Dst: 192.	168.136.5					
6	0100 = Ver	sion: 4			4				
0101 = Header Length: 20 bytes (5)									
> Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)									
Total Length: 84									
Identification: 0x7c24 (31780)									
> Flags: 0x40, Don't fragment									
0 0000 0000 0000 = Fragment Offset: 0									
Time to Live: 63									
Protocol: ICMP (1)									
Header Checksum: 0x1c6d [validation disabled]									
[Header checksum status: Unverified]									
Source Address: 192.168.153.193									
Destination Address: 192.168.136.5									
> Internet Control Message Protocol									

NO,	IIme	Source	Destination	Protocol	Lengtn Into					
	1 0.000000	192.168.153.193	192.168.136.5	ICMP	118 Echo (ping) request	id=0x0				
4	2 0.000217	192.168.136.5	192.168.153.193	ICMP	118 Echo (ping) reply	id=0x0				
	3 1.001428	192.168.153.193	192.168.136.5	ICMP	118 Echo (ping) request	id=0x0				
	4 1.001611	192.168.136.5	192.168.153.193	ICMP	118 Echo (ping) reply	id=0x0				
	5 2.002647	192.168.153.193	192.168.136.5	ICMP	118 Echo (ping) request	id=0x0				
	6 2.002801	192.168.136.5	192.168.153.193	ICMP	118 Echo (ping) reply	id=0x0				
	7 7 007576	400 400 453 403	400 400 400 5	TCMD	440 5 1 / • \					
>	Frame 1: 118 bytes	on wire (944 bits),	118 bytes captured (9	944 bits)						
>	> Ethernet II, Src: PcsCompu_bc:85:3a (08:00:27:bc:85:3a), Dst: PcsCompu_39:ce:bd (08:00:27:39:ce:bd)									
>	> Internet Protocol Version 4, Src: 203.248.23.215, Dst: 203.248.23.214									
>	> Internet Protocol Version 4, Src: 192.168.153.193, Dst: 192.168.136.5									
~	✓ Internet Control Message Protocol									
	Type: 8 (Echo (ping) request)									
	Code: 0									
	Chackeums 0x771c [connact]									
	[Checksum Status: Good]									
	Identifier (BE): 11 (0x000b)									
	Identifier (LE): 2816 (0x0b00)									
	Sequence Number (BE): 229 (0x00e5)									
	Sequence Number (LE): 58624 (0xe500)									
	[Response frame: 2]									
	Timestamp from icmp data: Apr 26, 2022 17:24:45.000000000 대한민국 표준시									
	[Timestamp from icmp data (relative): 0.979217000 seconds]									
	> Data (48 bytes)	. , ,		_						
	, , ,									

Network Overlay : https://ikcoo.tistory.com/117

[CKA] #2. Pod - 1

[CKA] #2. Pod (1)

CKA 가 URL

Pod?

Kubernetes

```
, Pod
                                              Container
                  가
Pod
     Network
                            /pause
  Pod
                               가
                                         , /pause 가
         Pod
                Network
Pod Create
                                    dry run
#
          yaml
$ kubectl run hello --image=nginx --dry-run=client -o yaml
apiVersion: v1
kind: Pod
metadata:
  creationTimestamp: null
  labels:
    run: hello
  name: hello
spec:
  containers:
  - image: nginx
    name: hello
    resources: {}
  dnsPolicy: ClusterFirst
  restartPolicy: Always
status: {}
                                              (apply) -
#
                     | (
                           )
kubectl run hello2 --image=nginx --dry-run=client -o yaml |
kubectl apply -f -
kubectl get pods -o wide
# nodeName
              가
                                         가 .
                                pod
       Taint
          nodeName
                       가
vaml
$ kubectl run hi --image=nginx --dry-run=client -o yaml >
hi.yaml
apiVersion: v1
kind: Pod
```

```
metadata:
  creationTimestamp: null
  labels:
    run: hi
  name: hi
spec:
  containers:
  - image: nginx
    name: hi
    resources: {}
  dnsPolicy: ClusterFirst
  restartPolicy: Always
  nodeName: user-controller
                                ## Node
status: {}
        YAML
                     Pod
#
$ kubectl create -f hi.yaml
$ kubectl get pods -o wide
NAME
                           READY
                                    STATUS
                                               RESTARTS
                                                            AGE
ΙP
                                                      READINESS
                  NODE
                                     NOMINATED NODE
GATES
hello-776c774f98-894tt
                            1/1
                                    Running
                                                            18h
                                                0
192.168.153.193 user-worker
                                    <none>
                                                     <none>
hί
                          1/1
                                   Running
                                              0
                                                          3m10s
192.168.136.5 user-controller
                                    <none>
                                                     <none>
Pod status
$ kubectl describe pod hi
# Pod
                     . -- kubectl
                                    arg
$ kubectl exec -it hi -- /bin/bash
root@hi:/#
               Pod
                                         . deployment
#
                --replicas
```

\$ kubectl create deployment web --image=nginx --replicas=3

```
deployment.apps/web created
$ kubectl get pods -o wide | grep -i web
web-76b56fd968-c2pk9
                            1/1
                                    Running
                                                0
                                                            11s
192.168.153.217
                  user-worker
                                    <none>
                                                     <none>
web-76b56fd968-chr4w
                                    Running
                                                0
                                                            11s
192.168.136.6
                  user-controller
                                    <none>
                                                     <none>
web-76b56fd968-mmdfn
                            1/1
                                    Running
                                                0
                                                            11s
192.168.153.218
                  user-worker
                                    <none>
                                                     <none>
Pod log
# Pod info
kubectl describe pod hi
# Pod log
kubectl logs hi
# journal Log(kubelet)
sudo journalctl -u kubelet
#
          Log
hi POD
           Container
/pause
             가 Pod
                                      . nignx
                                                     가
$ sudo docker ps -a | grep -i hi
d507d8b298c3
                                                    "/docker-
                  nginx
entrvpoint..."
                       26
                            hours
                                    ago
                                               Uр
                                                         hours
k8s hi hi default 6a1464a1-0fea-4ff8-a5c6-426afe281173 0
                                                      "/pause"
                  k8s.gcr.io/pause:3.6
d8fb1a992247
26
        hours
                                         Uр
                                                 26
                                                         hours
                    ago
k8s POD hi default 6a1464a1-0fea-4ff8-a5c6-426afe281173 0
# nginx Service Container info
$ sudo docker inspect d507d8b298c3
$ sudo docker logs d507d8b298c3
# Pod Network info
$ sudo docker inspect d8fb1a992247
$ sudo docker logs d8fb1a992247
```

가

Container

```
$ sudo docker exec -it d507d8b298c3 ls
bin
      docker-entrypoint.d
                            home
                                  media
                                          proc sbin
                                                     tmp
     docker-entrypoint.sh
                           lib
boot
                                  mnt
                                          root srv
                                                     usr
dev
                            lib64
                                          run sys
     etc
                                  opt
                                                     var
$ sudo docker exec -it d507d8b298c3 /bin/bash
root@hi:/#
                          가
#
             Net
                                         nsenter
ip
$ $ sudo docker exec -it d507d8b298c3 ip addr
OCI runtime exec failed: exec failed: container linux.go:380:
starting container process caused: exec: "ip": executable file
not found in $PATH: unknown
       PID
$ sudo docker inspect --format '{{ .State.Pid }}' d507d8b298c3
1244489
                     PID
                                           (pod namespace가
nsenter
                   가
 )
$ sudo nsenter -t 1244489 -n ip addr
1: lo: <LOOPBACK, UP, LOWER UP> mtu 65536 qdisc noqueue state
UNKNOWN group default glen 1000
   link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid lft forever preferred lft forever
2: tunl@NONE: <NOARP> mtu 1480 qdisc noop state DOWN group
default glen 1000
   link/ipip 0.0.0.0 brd 0.0.0.0
4: eth0@if12: <BROADCAST,MULTICAST,UP,LOWER UP> mtu 1480 gdisc
noqueue state UP group default
    link/ether c6:3d:04:5d:80:82 brd ff:ff:ff:ff:ff:ff link-
netnsid 0
    inet 192.168.136.5/32 scope global eth0
       valid lft forever preferred lft forever
```

Pod delete

Pod

```
# Pod Delete. replicas 7 .

$ kubectl delete deployment web
deployment.apps "web" deleted
$ kubectl get pods -o wide | grep -i web
-

Pod : https://kubernetes.io/docs/concepts/workloads/pods/
Pod Networking :
https://www.digitalocean.com/community/tutorials/how-to-inspec
t-kubernetes-networking
```

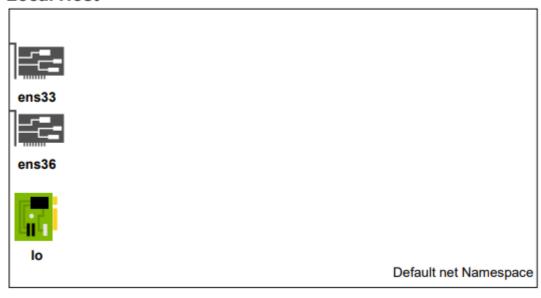
[] Network Namespace

: CentOS 7.6.1810

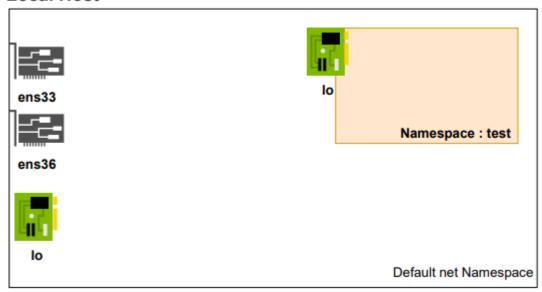
: root

Network Namespace

Default Network Namespace Check



Create Network Namespace



lo 가

test 가 Namespace

\$ ip netns add test

\$ ip netns
test

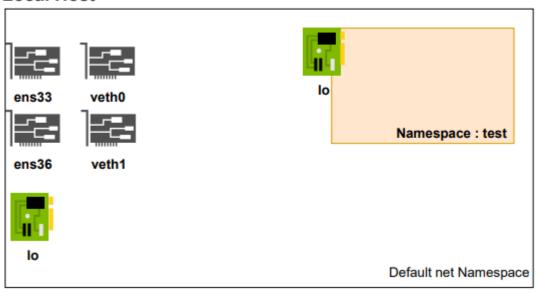
Check

PID 가 lsns

\$ lsns -t net
PID USER TYPE COMMAND

1 root net /usr/lib/systemd/systemd --switched-root -system --deserialize 22

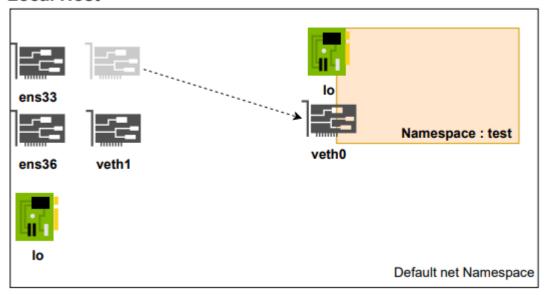
Namespace Network 1 – 가



```
가
     Network Namespace
가
        veth
                H0ST <--->
veth
                         가
# HOST 가
                                   . veth type
                                                          pair
                                                   peer
$ ip link add veth0 type veth peer name veth1
# HOST veth0/veth1
                         2
                              가
                                           가
$ ip -br -c addr
lo
                 UNKNOWN
                                127.0.0.1/8
ens33
                 UP
                                211.239.150.48/23
                                192.168.0.2/24
ens36
                 UP
veth1@veth0
                 DOWN
veth0@veth1
                 DOWN
```

Namespace Network 2 – 가

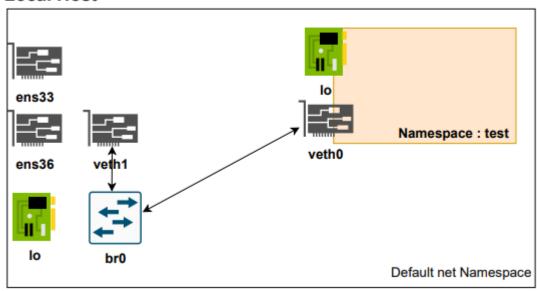
veth0@if4



```
가
                                                    test
# veth0
                  test Namespace
                                   Set
$ ip link set veth0 netns test
#
        HOST veth0
                         test namespace
$ ip -br -c addr
                               127.0.0.1/8
lo
                UNKNOWN
ens33
                UP
                               211.239.150.48/23
ens36
                UP
                               192.168.0.2/24
veth1@if5
                DOWN
test namespace
                 가
                                         veth0
# test namespace netns exec
$ ip netns exec test ip -br addr
lo
                DOWN
```

Namespace Network 3 - bridge

DOWN



```
test namespace veth0
                                            가
H0ST
                               veth1
  DOWN
      가
                                ΙP
                                (bridge)
          HOST
                                         가
# Check
      가
           ) yum install -y bridge-utils-1.5-9.el7.x86 64
$ (
$ btctl show
bridge name bridge id
                                             STP enabled
interfaces
        가
                           . br0
                                               HOST
# Bridge Create && Check
$ ip link add br0 type bridge
$ brctl show
bridge name bridge id
                                             STP enabled
interfaces
br0
              8000.000000000000
                                     no
```

127.0.0.1/8

가

UNKNOWN

\$ ip -br -c addr

lo

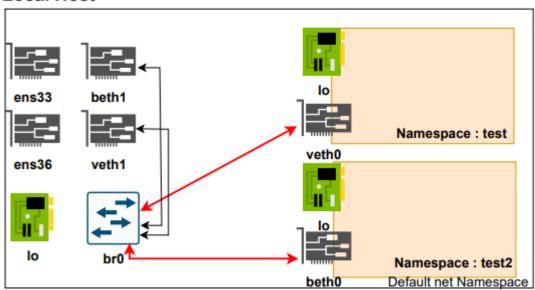
```
ens33
                                211.239.150.48/23
ens36
                                192.168.0.2/24
                 UP
veth1@if5
                 DOWN
br0
                 DOWN
           br0 vethx
# HOST
                 veth1
                         Host
                                br0
$ ip link set veth1 master br0
# check
          bridge
                   veth1 가
$ brctl show
bridge name
                                                  STP enabled
                   bridge id
interfaces
br0
                8000.46df623e69e4
                                                        veth1
                                        no
가
                                                    , IP
                         ΙP
ifconfig
                 net-util
                                          ip
                                                             ΙP
# netns exec
                          test Namespace
                                            veth0
          UP
$ ip netns exec test ip addr add 10.10.10.2/24 dev veth0
$ ip netns exec test ip link set veth0 up
#
      host
             veth1
                              bridge
                                                 up
$ ip link set br0 up
$ ip link set veth1 up
# UP check
                                가
                                       UP
$ ip -br -c addr
lo
                                127.0.0.1/8
                 UNKNOWN
ens33
                 UP
                                211.239.150.48/23
ens36
                                192.168.0.2/24
                 UP
veth1@if5
                 UP
br0
                 UP
                            UP
# test namespace
```

UP

```
$ ip netns exec test ip link
1: lo: <LOOPBACK> mtu 65536 qdisc noop state DOWN mode DEFAULT
group default glen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
5: veth0@if4: <BROADCAST,MULTICAST,UP,LOWER UP> mtu 1500 gdisc
noqueue state UP mode DEFAULT group default glen 1000
    link/ether f2:1c:09:d4:47:fc brd ff:ff:ff:ff:ff:ff link-
netnsid 0
# lo
             가 DOWN
                                                        가
  . UP
           UNKNOWN
$ ip netns exec test ip link set dev lo up
$ ip netns exec test ip a
1: lo: <LOOPBACK, UP, LOWER UP> mtu 65536 qdisc noqueue state
UNKNOWN group default glen 1000
# Check
$ ip netns exec test ping 127.0.0.1
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
64 bytes from 127.0.0.1: icmp seq=1 ttl=64 time=0.063 ms
64 bytes from 127.0.0.1: icmp seq=2 ttl=64 time=0.058 ms
# Check 2
Host
                      ΙP
Gateway
       Routing 가
                                                         ip
                                     ΙP
#
$ ip addr add 10.10.10.200/24 dev br0
# test
                   veth0
                                     Ping
$ ping 10.10.10.2
 ping 10.10.10.2 -c 2
PING 10.10.10.2 (10.10.10.2) 56(84) bytes of data.
64 bytes from 10.10.10.2: icmp seq=1 ttl=64 time=0.073 ms
64 bytes from 10.10.10.2: icmp seq=2 ttl=64 time=0.071 ms
--- 10.10.10.2 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 999ms
```

Namespace Network 4 –

Local Host



```
가
test namespace
      가
                                         가
# test2 namespace 가 beth0/beth1
# IP
      test <---> test2
                                         Ping
ip netns add test2
ip link add beth0 type veth peer name beth1
ip link set beth0 netns test2
ip link set beth1 master br0
ip netns exec test2 ip addr add 10.10.10.3/24 dev beth0
ip netns exec test2 ip link set beth0 up
ip netns exec test2 ip link set dev lo up
ip link set beth1 up
# test2 namespace
$ ip netns
test2 (id: 1)
test (id: 0)
```

\$ brctl show

beth1@if8

bridge name bridge id STP enabled

interfaces

br0 8000.2e0e64ccb0e5 no beth1

veth1

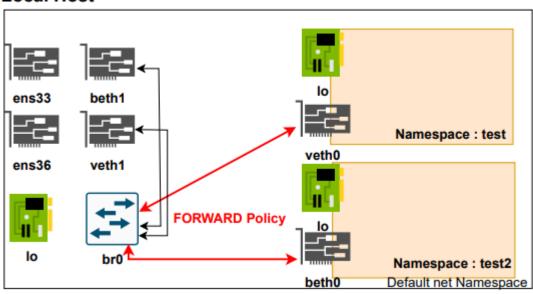
test namespace veth0(10.10.10.2) Ping ip netns exec test2 ping 10.10.10.2 -c 2 PING 10.10.10.2 (10.10.10.2) 56(84) bytes of data. 64 bytes from 10.10.10.2: icmp_seq=1 ttl=64 time=0.112 ms 64 bytes from 10.10.10.2: icmp_seq=2 ttl=64 time=0.076 ms

--- 10.10.10.2 ping statistics ---

UP

2 packets transmitted, 2 received, 0% packet loss, time 1000ms rtt min/avg/max/mdev = 0.076/0.094/0.112/0.018 ms

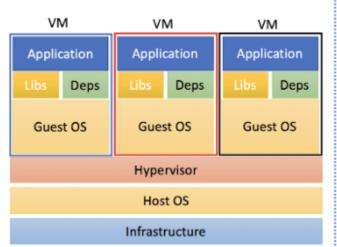
Local Host

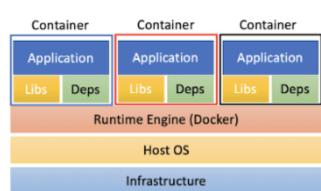


? HOST lo () , Host

```
NAT
ip4
               FORWARD
                              HOST
      HOST iptables FORWARD
                                          ACCEPT
$ iptables -nL | grep -i forward
Chain FORWARD (policy DROP)
#
$ iptables --policy FORWARD ACCEPT
$ iptables -nL | grep -i forward
Chain FORWARD (policy ACCEPT)
$ service iptables save (
                                           0S
                                                              )
             ip4v.forward
echo 1 > /proc/sys/net/ipv4/ip_forward
sysctl --system
# check
#
```

l VM Container





Container

```
(LXC) LXC
Docker)
              HOST
                                            Namespace
        cgroup
                                       가
#
       namespace
                    Host
                                                        가
                     가
H0ST
      Linux
                                . VM
pid
user
uts
ipc
mnt
net
#cgroup
                               Host
Memory
```

CPU
Network
Device
I/O

Host

Windows OS . - Container Host

가

\mathbf{VM}

- VM Host Hypervisor 가 OS - Host , 가 Linux/Windows/Other Guest OS OS .

[CKA] #1.

: [CKA] #1.

Kubenertes

7 CKA
kubeadm .

(VM)
Controller Server : 1EA
Worker Server : 1EA

OS
Ubuntu 20.04 Server Minimal

SWAP
sudo swapoff /swap.img
sudo sed -i -e '/swap.img/d' /etc/fstab

. (regular user) sudo

```
sudo hostnamectl set-hostname controller
sudo hostnamectl set-hostname worker
```

Traffic Setup

Container Runtime

```
# POD
CKA Docker
This is a second of the controller of the control
```

cgroup

```
# cgroup .

OS cgroup systemd , docker, kubelet
```

```
cgroupfs 가
```

```
## Controller / Worker
sudo mkdir /etc/docker
cat <<EOF | sudo tee /etc/docker/daemon.json</pre>
{
  "exec-opts": ["native.cgroupdriver=systemd"],
  "log-driver": "json-file",
  "log-opts": {
    "max-size": "100m"
  },
  "storage-driver": "overlay2"
E0F
## Docker enable && restart
sudo systemctl enable docker
sudo systemctl daemon-reload
sudo systemctl restart docker
## Docker cgroup driver
                                     cgroupfs systemd
sudo docker info | grep -i cgroup
 Cgroup Driver: systemd
Cgroup Version: 1
#
                 kebe
## Controller / Worker
sudo apt-get update
sudo apt-get install -y apt-transport-https ca-certificates
curl
sudo curl -fsSLo /usr/share/keyrings/kubernetes-archive-
keyring.gpg
https://packages.cloud.google.com/apt/doc/apt-key.gpg
```

echo "deb [signed-by=/usr/share/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

Kube InitIalize.

kubectl get nodes

NAME

VERSION

Controller Node init --cri-socket: kubeadm socket 가 --pod-network-cidr : pod network CoreDNS Service --apiserver-advertise-address=<ip-address> : Controller API ## Controller. ΙP APT (Advertise) sudo kubeadm init --ignore-preflight-errors=all --pod-networkcidr=192.168.0.0/16--apiserver-advertiseaddress=203.248.23.192 # init 가 (regular user) + sudo 1) cluster Your Kubernetes control-plane has initialized successfully! To start using your cluster, you need to run the following as a regular user: mkdir -p \$HOME/.kube sudo cp -i /etc/kubernetes/admin.conf \$HOME/.kube/config sudo chown \$(id -u):\$(id -g) \$HOME/.kube/config ## Check

STATUS

ROLES

AGE

user1-controller NotReady control-plane,master 6m28s
v1.23.5

2) pod network

Network Plugin

You should now deploy a pod network to the cluster.

Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:

https://kubernetes.io/docs/concepts/clusteradministration/addons/

Pod Network

CoreDNS 가

(Pending)

kubectl get pods --all-namespaces

NAMESPACE		NAME		READY
STATUS	REST	ARTS	AGE	
kube-syst	em	coredi	ns-64897985d-9sj9j	0/1
Pending	0		12m	
kube-system		coredns-64897985d-zfl8q		0/1
Pending	0		12m	
kube-syst	em	etcd-	user1-controller	1/1
Running	0		12m	
kube-system		kube-a	apiserver-user1-controller	1/1
Running	0		12m	
kube-system		kube-	controller-manager-user1-controller	1/1
Running	0		12m	
kube-system		kube-	proxy-g5xdv	1/1
Running	0		12m	
kube-system		kube-s	scheduler-user1-controller	1/1
Running	0		12m	

Pod Network Plugin Install

CKA Callico Plugin

curl

https://projectcalico.docs.tigera.io/manifests/calico.yaml -0
kubectl apply -f calico.yaml
kubectl get nodes

Check

, coredns status 가 Running

```
kubectl get pods --all-namespaces
NAMESPACE
             NAME
                                                       READY
STATUS
          RESTARTS
                    AGE
              calico-kube-controllers-56fcbf9d6b-bnxz5
kube-system
                                                         0/1
Pending
kube-system
             calico-node-khp2h
                                                         0/1
Init:2/3
                     20s
kube-system
             coredns-64897985d-9si9i
                                                         0/1
Pending
                     22m
          0
kube-system
             coredns-64897985d-zfl8a
                                                         0/1
Pending
                    22m
          0
kube-system etcd-user1-controller
                                                         1/1
                     22m
Running
          0
Multi NIC
             가
                            INTERNAL-IP
            가
                                                NIC
                                K8S
                                                      IP 가
INTERNAL-IP
   INTERNAL-IP
                 Init
kubeadm --apiserver-advertise-address
                                             ΙP
# INTERNAL-IP 가 10.0.2.15 ( Calico Network Default )
$ kubectl get nodes -o wide
NAME
                    STATUS
                              ROLES
                                                        AGE
                                                  OS-IMAGF
             INTERNAL-IP
VERSION
                               EXTERNAL-IP
                     CONTAINER-RUNTIME
KERNEL-VERSION
user-controller
                    Ready
                              control-plane, master 44h
                                        Ubuntu 20.04.1 LTS
v1.23.5 10.0.2.15
                         <none>
5.4.0-64-generic docker://20.10.14
user-worker
                    Ready < none>
                                                        44h
          10.0.2.15
                                        Ubuntu 20.04.1 LTS
v1.23.5
                         <none>
5.4.0-64-generic docker://20.10.14
# Controller.
cat << EOF | sudo tee /etc/default/kubelet</pre>
KUBELET EXTRA ARGS='--node-ip $(hostname -I | cut -d ' ' -f2)'
E0F
sudo systemctl daemon-reload
sudo systemctl restart kubelet
kubectl cluster-info
```

```
# Worker.
cat << EOF | sudo tee /etc/default/kubelet</pre>
KUBELET EXTRA ARGS='--node-ip $(hostname -I | cut -d ' ' -f2)'
F0F
sudo systemctl daemon-reload
sudo systemctl restart kubelet
         Internal-IP 가 advertise
# Check
$ kubectl get nodes -o wide
                   STATUS
NAME
                             ROLES
                                                      AGE
VERSION
           INTERNAL-IP
                                EXTERNAL-IP
                                                OS-IMAGE
                   CONTAINER-RUNTIME
KERNEL-VERSION
                   Ready control-plane, master 45h
user-controller
v1.23.5 203.248.23.214
                                        Ubuntu 20.04.1 LTS
                          <none>
5.4.0-64-generic docker://20.10.14
user-worker
                   Ready
                                                      44h
                             <none>
v1.23.5 203.248.23.215
                                        Ubuntu 20.04.1 LTS
                          <none>
5.4.0-64-generic docker://20.10.14
Worker
            Controller
                        Join
Then you can join any number of worker nodes by running the
following on each as root:
root
         Worker
                     kebeadm
                                            Controller
/etc/kebenertes/admin.conf Worker
# Controller
sudo
                          /etc/kubernetes//admin.conf
              scp
vagrant@203.248.23.193:/home/vagrant/admin.conf
# Worker
mkdir -p $HOME/.kube
sudo cp -i ./admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
kubeadm
                  203.248.23.192:6443 --token
             ioin
wy11vq.bk2rze7g9lilg2d9 \
                       --discovery-token-ca-cert-hash
```

sha256:f7bc17bb974c804821b21427d500cb96615f66c1fd88cb53c023d8b

2c598d3f7

가 가 ignore kubeadm join 203.248.23.192:6443 --token sudo wy11vg.bk2rze7g9lilg2d9 --ignore-preflight-errors=all discovery-token-ca-cert-hash sha256:f7bc17bb974c804821b21427d500cb96615f66c1fd88cb53c023d8b 2c598d3f7

This node has joined the cluster:

- * Certificate signing request was sent to apiserver and a response was received.
- * The Kubelet was informed of the new secure connection details.

Run 'kubectl get nodes' on the control-plane to see this node

join the cluster.								
### Check								
Worker pod 가								
kubectl get nodes								
NAME STATUS ROLES	AGE							
VERSION Ween't controller Dondy control plane meeter	22							
<pre>user1-controller Ready control-plane,master v1.23.5</pre>	33m							
user1-worker Ready <none></none>	84s							
v1.23.5								
kubactl got node all namacnaces								
kubectl get podsall-namespaces NAMESPACE NAME								
STATUS RESTARTS AGE								
kube-system calico-kube-controllers-56fcbf9d6b-bnxz5	5 1/1							
Running 0 11m	1/1							
<pre>kube-system calico-node-khp2h Running 0 11m</pre>	1/1							
kube-system calico-node-skdjl	1/1							
Running 0 2m3s								
kube-system coredns-64897985d-9sj9j	1/1							
Running 0 33m	1/1							
kube-system coredns-64897985d-zfl8q	/							

kube-system	etcd-user1-controller	1/1
Running 0	33m	
kube-system	kube-apiserver-userl-controller	1/1
Running 0	33m	
kube-system	kube-controller-manager-user1-controller	1/1
Running 0	33m	
kube-system	kube-proxy-g5xdv	1/1
Running 0	33m	
kube-system	kube-proxy-m6ztf	1/1
Running 0	2m3s	
kube-system	kube-scheduler-user1-controller	1/1
Running 0	33m	

(Trouble)

```
# All Node
sudo systemctl stop kubelet
sudo kubeadm reset -f

sudo rm -rf ~/.kube
sudo rm -rf /root/.kube
sudo rm -rf /var/lib/etcd
```

Network Plugin Status

Kubernetes Auto Complation

```
# alias Tab
echo '' >>~/.bashrc
echo 'source <(kubectl completion bash)' >>~/.bashrc
echo 'alias k=kubectl' >>~/.bashrc
echo 'complete -F __start_kubectl k' >>~/.bashrc

. ~/.bashrc

# Check
## Tab
k get nodes -o wide
kubectl get nodes -o wide
```