

2022.12.07 CKA(Certified Kubernetes Administrator)

2022.06.25 (PSI Secure Browser)

;

URL :

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

—

30

Exam

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

TAKE EXAM

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

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

CHAT

-
-
- ()
(, HDMI)
CHAT)
- , (/)
-
가

&

PSI

가

Firefox

CHAT
가

가
가

100 66 (2/3)
24

Pass

24 가

Fail Reschedule

Flag

가

/

Kubernetes 1.24 + cri-docker Installation (kubernetes)

Kubernetes

18.04

Control 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 -s https://api.github.com/repos/Mirantis/cri-dockerd/releases/latest|grep tag_name | cut -d '"' -f 4|sed 's/v//g')
echo $VER
wget https://github.com/Mirantis/cri-dockerd/releases/download/v${VER}/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

wget
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
sudo mv cri-docker.socket cri-docker.service
/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
{
  "exec-opts": ["native.cgroupdriver=systemd"],
  "log-driver": "json-file",
  "log-opts": {
    "max-size": "100m"
  },
  "storage-driver": "overlay2"
}
EOF

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
EOF
```

```
cat <<EOF | sudo tee /etc/sysctl.d/k8s.conf
net.bridge.bridge-nf-call-ip6tables = 1
net.bridge.bridge-nf-call-iptables = 1
EOF
```

```
sudo sysctl --system
```

Packages Install - All node

```
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
```

```
# 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  
kubeadm join 211.115.207.136:6443 --token r8gfco.6s7f60dns4vgwcc0 \ --discovery-token-ca-cert-hash sha256:7b0f82be076748e67f8615eab0b86a61317bac397f94b2921810231ab14afdcc
```

```
# 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 scp /etc/kubernetes/admin.conf worker@node02:/home/worker/admin.conf  
sudo scp /etc/kubernetes/admin.conf worker@node03:/home/worker/admin.conf
```

Node Join - Worker Node (Node02 , Node03)

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



```

sudo kubeadm join --token <token> <controlplane-
host>:<controlplane-port> --ignore-preflight-errors=all --cri-
socket unix:///var/run/cri-dockerd.sock
# sudo kubeadm join [Controller IP]:6443 --token
3nvrwg.33k750dq9klm5omi --discovery-token-ca-cert-hash
sha256:b10158bcea37aae0e92ed6b68b4dd1e8213623cc7d406e77eef55fe
6196fe346 --cri-socket /var/run/cri-dockerd.sock

# Join
sudo kubeadm join [Controller IP]:6443 --token
r8gfco.6s7f60dns4vgwcc0 \
--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
3          .
kubectl get pod -A
kubectl logs -f pod/kube-proxy-[name] -n kube-system

# Log          - Linux          iptables Mode          .
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
```

```
Plugin      ,      CKA      Callico
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 -O
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 -L
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
```

[CKA] #2. Pod - 1

[CKA] #2. Pod (1)

CKA 가 URL .

Pod ?

Kubernetes

Pod Network , Pod Container
Pod 가 /pause
Pod Network , /pause 가

Pod Create

yaml dry run

```
$ 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 pod 가 .  
Taint .  
yaml nodeName 가 .  
$ 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	IP	READY	STATUS	RESTARTS	AGE
			NOMINATED	NODE	READINESS
hello-776c774f98-894tt	192.168.153.193	1/1	Running	0	18h
hi	192.168.136.5	1/1	Running	0	3m10s

Pod status

```
$ kubectl describe pod hi
```

```
# Pod . -- kubectl arg --
```

```
$ kubectl exec -it hi -- /bin/bash  
root@hi:/#
```

Pod

```
# --replicas . deployment
```

```
$ 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	1/1	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 . nginx 가
```

```
$ sudo docker ps -a | grep -i hi
```

```
d507d8b298c3    nginx    "/docker-
entrypoint..."    26 hours ago    Up 26 hours
k8s_hi_hi_default_6a1464a1-0fea-4ff8-a5c6-426afe281173_0
d8fb1a992247    k8s.gcr.io/pause:3.6    "/pause"
26 hours ago    Up 26 hours
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
```

```
# Container . 가
$ sudo docker exec -it d507d8b298c3 ls
bin  docker-entrypoint.d  home  media  proc  sbin  tmp
boot docker-entrypoint.sh  lib   mnt    root  srv   usr
dev  etc                  lib64 opt    run   sys   var
```

```
$ sudo docker exec -it d507d8b298c3 /bin/bash
root@hi:/#
```

```
# Net 가 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
```

```
nsenter PID (pod namespace가
) 가 .
$ sudo nsenter -t 1244489 -n ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state
UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
```

```
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
2: tunl0@NONE: <NOARP> mtu 1480 qdisc noop state DOWN group
default qlen 1000
    link/ipip 0.0.0.0 brd 0.0.0.0
4: eth0@if12: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1480 qdisc
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
$ kubectl delete hi
```

```
# Pod Delete. replicas 가 .
$ 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-inspect-kubernetes-networking>

[CKA] #1.

: [CKA] #1.

Kubeneretes

```
#                가                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

```
#                ( : Docker), kube-proxy
                iptables                .

## Container / Worker
                ,                netfilter(iptables)

cat <<EOF | sudo tee /etc/modules-load.d/k8s.conf
br_netfilter
EOF

cat <<EOF | sudo tee /etc/sysctl.d/k8s.conf
net.bridge.bridge-nf-call-ip6tables = 1
net.bridge.bridge-nf-call-iptables = 1
EOF
```

```
sudo systemctl --system
```

Container Runtime

```
# CKA POD 가
   가   Docker /
```

```
## Controller / Worker
curl -fsSL https://get.docker.com -o get-docker.sh
sudo sh get-docker.sh
```

```
## Check
sudo docker -v
sudo docker ps -a
```

cgroup

```
# cgroup OS cgroup systemd , docker, kubelet
cgroupfs 가 systemd
```

```
## Controller / Worker
sudo mkdir /etc/docker
cat <<EOF | sudo tee /etc/docker/daemon.json
{
  "exec-opts": ["native.cgroupdriver=systemd"],
  "log-driver": "json-file",
  "log-opts": {
    "max-size": "100m"
  },
  "storage-driver": "overlay2"
}
EOF
```

```
## 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
```

```
# kube /
```

```
## 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.

```
# Controller Node init
```

```
--cri-socket : kubeadm socket
```

가

```
--pod-network-cidr : pod network
```

```
CoreDNS Service
```

```
--apiserver-advertise-address=<ip-address> :
```

```
Controller API
```

```
## Controller.
```

```
IP
```

```
API
```

(Advertise)

```
sudo kubeadm init --ignore-preflight-errors=all --pod-network-cidr=192.168.0.0/16 --apiserver-advertise-address=203.248.23.192
```

```
# init 가 .
```

1) , (regular user) + sudo 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

```
kubectl get nodes
```

NAME	STATUS	ROLES	AGE
VERSION			
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/cluster-administration/addons/>

Pod Network CoreDNS 가

(Pending)

```
kubectl get pods --all-namespaces
```

NAMESPACE	NAME	STATUS	RESTARTS	AGE	READY
kube-system	coredns-64897985d-9sj9j	Pending	0	12m	0/1
kube-system	coredns-64897985d-zfl8q	Pending	0	12m	0/1
kube-system	etcd-user1-controller				1/1

```

Running    0           12m
kube-system kube-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-scheduler-user1-controller 1/1
Running    0           12m

```

Pod Network Plugin Install

, CKA

Callico Plugin

curl

<https://projectcalico.docs.tigera.io/manifests/calico.yaml> -o

kubectl apply -f calico.yaml

kubectl get nodes

Check

, coredns status 가 Running

kubectl get pods --all-namespaces

NAMESPACE	NAME	STATUS	RESTARTS	AGE	READY
kube-system	calico-kube-controllers-56fcbf9d6b-bnxz5	Pending	0	20s	0/1
kube-system	calico-node-khp2h	Init:2/3	0	20s	0/1
kube-system	coredns-64897985d-9sj9j	Pending	0	22m	0/1
kube-system	coredns-64897985d-zfl8q	Pending	0	22m	0/1
kube-system	etcd-user1-controller	Running	0	22m	1/1

Multi NIC 가

INTERNAL-IP

가 K8S NIC IP 가

INTERNAL-IP

INTERNAL-IP Init

```
kubeadm --apiserver-advertise-address IP
```

```
# INTERNAL-IP 가 10.0.2.15 ( Calico Network Default )
```

```
$ kubectl get nodes -o wide
```

NAME	STATUS	ROLES	AGE
VERSION	INTERNAL-IP	EXTERNAL-IP	OS-IMAGE
KERNEL-VERSION	CONTAINER-RUNTIME		
user-controller	Ready	control-plane,master	44h
v1.23.5	10.0.2.15	<none>	Ubuntu 20.04.1 LTS
5.4.0-64-generic	docker://20.10.14		
user-worker	Ready	<none>	44h
v1.23.5	10.0.2.15	<none>	Ubuntu 20.04.1 LTS
5.4.0-64-generic	docker://20.10.14		

```
# Controller.
```

```
cat << EOF | sudo tee /etc/default/kubelet
```

```
KUBELET_EXTRA_ARGS='--node-ip $(hostname -I | cut -d ' ' -f2)'
```

```
EOF
```

```
sudo systemctl daemon-reload
```

```
sudo systemctl restart kubelet
```

```
kubectl cluster-info
```

```
# Worker.
```

```
cat << EOF | sudo tee /etc/default/kubelet
```

```
KUBELET_EXTRA_ARGS='--node-ip $(hostname -I | cut -d ' ' -f2)'
```

```
EOF
```

```
sudo systemctl daemon-reload
```

```
sudo systemctl restart kubelet
```

```
# Check Internal-IP 가 advertise
```

```
$ kubectl get nodes -o wide
```

NAME	STATUS	ROLES	AGE
VERSION	INTERNAL-IP	EXTERNAL-IP	OS-IMAGE
KERNEL-VERSION	CONTAINER-RUNTIME		
user-controller	Ready	control-plane,master	45h
v1.23.5	203.248.23.214	<none>	Ubuntu 20.04.1 LTS
5.4.0-64-generic	docker://20.10.14		
user-worker	Ready	<none>	44h
v1.23.5	203.248.23.215	<none>	Ubuntu 20.04.1 LTS
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
# Controller
sudo scp /etc/kubernetes/admin.conf Worker
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 join 203.248.23.192:6443 --token
wy1lvq.bk2rze7g9lilg2d9 \
--discovery-token-ca-cert-hash
sha256:f7bc17bb974c804821b21427d500cb96615f66c1fd88cb53c023d8b
2c598d3f7

가 ignore 가
sudo kubeadm join 203.248.23.192:6443 --token
wy1lvq.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
kubect1 get nodes
NAME STATUS ROLES AGE
VERSION
user1-controller Ready control-plane,master 33m
v1.23.5
user1-worker Ready <none> 84s
v1.23.5

```

```

kubect1 get pods --all-namespaces
NAMESPACE NAME READY
STATUS RESTARTS AGE
kube-system calico-kube-controllers-56fcbf9d6b-bnxz5 1/1
Running 0 11m
kube-system calico-node-khp2h 1/1
Running 0 11m
kube-system calico-node-skdl 1/1
Running 0 2m3s
kube-system coredns-64897985d-9sj9j 1/1
Running 0 33m
kube-system coredns-64897985d-zfl8q 1/1
Running 0 33m
kube-system etcd-user1-controller 1/1
Running 0 33m
kube-system kube-apiserver-user1-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

```
Pod Network - Calico
Status
(calicoctl) 가 , Kubectl

# Host
$ cd /usr/local/bin
$ sudo curl -L
https://github.com/projectcalico/calico/releases/download/v3.2
2.1/calicoctl-linux-amd64 -o calicoctl
$ sudo chmod +x calicoctl
```

```
# Check
$ calicoctl ipam show --show-blocks
+-----+-----+-----+-----+
-----+
| GROUPING |          CIDR          | IPS TOTAL | IPS IN USE |
IPS FREE  |
+-----+-----+-----+-----+
-----+
| IP Pool  | 192.168.0.0/16        | 65536 | 8 (0%)      |
65528 (100%) |
| Block   | 192.168.136.0/26     | 64 | 3 (5%)      | 61
(95%)      |
| Block   | 192.168.153.192/26   | 64 | 5 (8%)      | 59
(92%)      |
+-----+-----+-----+-----+
-----+
```

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
```