K8s Network 설치

Network 장치명 고정 (모든 장비에 적용)

ostm 계정 생성 (ostm / miruware!)

```
$ sudo adduser ostm

$ sudo usermod -aG sudo ostm
```

NAT 네트워크 설정

커널 모듈 사용하도록 설정

```
$ sudo modprobe iptable_nat

$ sudo vi /etc/sysctl.conf
###################
net.ipv4.ip_forward=1
###################

$ sudo sysctl -p
```

내부 사설 네트워크 장치와 외부 장치에 모든 패킷이 통과하도록 설정, 외부 네트워크 장치에 매스커레이드 허가

```
$ sudo vi /etc/rc.local
###################
#!/bin/bash
iptables -A FORWARD -o <내부 사설 네트워크와 연결되는 장치> -j ACCEPT
iptables -A FORWARD -o <외부 네트워크와 연결되는 장치> -j ACCEPT
```

```
iptables -t nat -A POSTROUTING -o <외부 네트워크 장치> -j MASQUERADE
exit 0
###################
```

rc.local 활성화

```
$ sudo chmod +x /etc/rc.local

$ sudo vi /lib/systemd/system/rc-local.service
##########################
# /lib/systemd/system/rc-local.service 의 마지막에 다음 내용 추가
[Install]
WantedBy=multi-user.target
##########################

$ sudo systemctl enable rc-local.service
$ sudo systemctl start rc-local.service
```

MW OST(K8s) Install

apt 패키지 upgrade & update

```
$ sudo apt update && sudo apt-get upgrade -y
```

기본 패키지 설치

```
$ sudo apt install git vim net-tools ipmitool nfs-server nfs-common pdsh
```

hosts 파일에 노드 정보 추가

공인아이피 수정 필요

```
$ sudo vim /etc/hosts/
------------------------------------------------------------
# OST-HUB
210.115.46.234  ost-hub

# OST-K8s#
115.115.0.100 ost-k8s-master
115.115.0.10 k8s-storage
115.115.0.103 zvrs03 gpu03
115.115.0.104 zvrs04 gpu04
115.115.0.105 zvrs05 gpu05
115.115.0.106 zvrs06 gpu06
115.115.0.107 zvrs07 gpu07
115.115.0.108 zvrs08 gpu08
115.115.0.109 zvrs09 gpu09
115.115.0.110 zvrs10 gpu10
115.115.0.111 zvrs11 gpu11
115.115.0.112 zvrs12 gpu12
```

자동 업데이트 끄기 및 rc.local 활성화 config 파일 복사

```
$ sudo vi /etc/apt/apt.conf.d/10periodic
#########################################################
APT::Periodic::Update-Package-Lists "0";
APT::Periodic::Download-Upgradeable-Packages "0";
APT::Periodic::AutocleanInterval "0";
APT::Periodic::Unattended-Upgrade "0";
#########################################################
```

```
$ sudo vi /etc/apt/apt.conf.d/20auto-upgrades
######################################################
APT::Periodic::Update-Package-Lists "0";
APT::Periodic::Unattended-Upgrade "0";
######################################################
```

swap 끄기

```
$ sudo vim /etc/fstab
"""swapfile 주석 처리"""
#/swapfile                                 none            swap    sw
0       0

$ sudo swapoff /swap.img
```

방화벽 끄기

```
$ sudo systemctl stop ufw.service
$ sudo systemctl disable ufw.service
```

Run Lebel 변경 (multi-user.target) - 절전모드 해제

```
$ sudo systemctl set-default multi-user.target
```

software-properties-common 설치

```
$ sudo apt-get install \
    apt-transport-https \
```

```
    ca-certificates \
    curl \
    gnupg-agent \
    software-properties-common

$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
```

nfs 패키지 설치

```
$ sudo apt update
$ sudo apt-get install nfs-kernel-server nfs-common
```

3. docker 설치

```
$ sudo apt-key fingerprint 0EBFCD88
$ sudo add-apt-repository \
   "deb [arch=amd64] https://download.docker.com/linux/ubuntu \
   $(lsb_release -cs) \
   stable"
$ sudo apt-get update
$ sudo apt-get install docker-ce docker-ce-cli containerd.io
$ sudo apt-mark hold docker-ce docker-ce-cli

# Docker 그룹 추가
$ sudo usermod -aG docker $USER
# 로그아웃 후 재로그인
```

curl -s -L https://nvidia.github.io/nvidia-docker/gpgkey | sudo apt-key add - && \
distribution=$(. *etc*os-release;echo IDVERSION_ID) && \
curl -s -L https://nvidia.github.io/nvidia-docker/$distribution/nvidia-docker.list | sudo tee

*etc*apt*sources.list.d*nvidia-docker.list && \
sudo apt-get update && \
curl -s -L https://nvidia.github.io/nvidia-docker/gpgkey | sudo apt-key add - && \
sudo apt-get install nvidia-docker2 -y && \
sudo pkill -SIGHUP dockerd

5. Kubernetes 설치

```
$ sudo apt update && sudo apt-get upgrade -y
$ curl https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo apt-key add
-

$ sudo su -
$ cat <<EOF > /etc/apt/sources.list.d/kubernetes.list \
deb http://apt.kubernetes.io/ kubernetes-xenial main \
EOF
$ exit

$ sudo apt-get update
$ sudo apt-get install -y kubelet=1.21.9-00 kubeadm=1.21.9-00 kubectl=1.21.9-00
kubernetes-cni
$ sudo apt-mark hold kubelet kubeadm kubectl kubernetes-cni
```

자동완성 설정

```
$ sudo apt-get install bash-completion
$ echo 'source <(kubectl completion bash)' >>~/.bashrc

#$ sudo su -
#$ kubectl completion bash >/etc/bash_completion.d/kubectl

# 로그아웃 후 로그인
```

Master Node

Kubernetes initializing

```
# apiserver ip 는 master 서버의 내부망 설정값에 맞도록 변경
$ sudo kubeadm init

# 토큰 값 기록할 것
$ sudo kubeadm join 100.100.0.70:6443 --token pbtzg9.89mfnsedcmtfsve4 \
  --discovery-token-ca-cert-hash
sha256:6dd5137005e29dd3b5dbeb2ae8eebc756e4aa4ec84c503004099168408d029bf

# kubernetes 인증
$ mkdir -p $HOME/.kube
$ sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
$ sudo chown $(id -u):$(id -g) $HOME/.kube/config
```

만약 token 을 기록하지 않았다면 검색하여 저장할 것

```
# 토큰값 확인
$ kubeadm token list

# hash 값 확인
$ openssl x509 -pubkey -in /etc/kubernetes/pki/ca.crt | openssl rsa -pubin
-outform der 2>/dev/null | openssl dgst -sha256 -hex | sed 's/^.* //'

# kube join 방법 example
kubeadm join {Master IP}:6443 --token {token} \
    --discovery-token-ca-cert-hash sha256:{hash}
```

kubernetes 인증 키 추가

```
$ mkdir -p $HOME/.kube
$ sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
$ sudo chown $(id -u):$(id -g) $HOME/.kube/config
```

weave 서비스 생성

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

dashboard 서비스 생성

```
$ kubectl apply -f https://raw.githubusercontent.com/kubernetes/dashboard/
v2.5.0/aio/deploy/recommended.yaml
```

pod 생성 확인

```
# fail 없는지 확인
$ kubectl get pod -A -o wide
```

Dashboard 설정

kubernetes-dashboard type 변경

```
$ kubectl -n kubernetes-dashboard edit service kubernetes-dashboard
```

```
################################################################################
#######
# and an empty file will abort the edit. If an error occurs while saving this
file will be
# reopened with the relevant failures.
#
apiVersion: v1
kind: Service
metadata:
  annotations:
    kubectl.kubernetes.io/last-applied-configuration: |
      {"apiVersion":"v1","kind":"Service","metadata":{"annotations":
{},"labels":{"k8s-app":"kubernetes-dashboard"},"name":"kubernetes-
dashboard","namespace":"kubernetes-dashboard"},"spec":{"ports":
[{"port":443,"targetPort":8443}],"selector":{"k8s-app":"kubernetes-
dashboard"}}}
  creationTimestamp: 2019-08-02T07:53:31Z
  labels:
    k8s-app: kubernetes-dashboard
  name: kubernetes-dashboard
  namespace: kubernetes-dashboard
  resourceVersion: "743"
  selfLink: /api/v1/namespaces/kubernetes-dashboard/services/kubernetes-
dashboard
  uid: ce676471-4e08-4f46-b80f-08d1fbb088a7
spec:
  clusterIP: 10.105.252.142
  ports:
  - port: 443
    protocol: TCP
    targetPort: 8443
  selector:
    k8s-app: kubernetes-dashboard
  sessionAffinity: None
  type: ClusterIP ###### NodePort 로 변경
status:
  loadBalancer: {}
################################################################################
#######
```

```
# 변경 확인
$ kubectl -n kubernetes-dashboard get service kubernetes-dashboard
kubernetes-dashboard   NodePort   10.96.194.199   <none>        443:30502/TCP
58m


# 확인 후 다시 config를 열어 port 수정
$ kubectl -n kubernetes-dashboard edit service kubernetes-dashboard
################################################################################
#######
apiVersion: v1
kind: Service
metadata:
  creationTimestamp: "2021-04-22T06:58:07Z"
  labels:
    k8s-app: kubernetes-dashboard
  name: kubernetes-dashboard
  namespace: kubernetes-dashboard
  resourceVersion: "2487"
  uid: cc8df335-14a5-444d-9afa-5317384bf46d
spec:
  clusterIP: 10.109.255.157
  clusterIPs:
  - 10.109.255.157
  externalTrafficPolicy: Cluster
  ports:
  - nodePort: 32222  # 포트 수정
    port: 443
    protocol: TCP
    targetPort: 8443
  selector:
    k8s-app: kubernetes-dashboard
  sessionAffinity: None
  type: NodePort
status:
  loadBalancer: {}
################################################################################
```

```
#######
```

설치 완료 후 대쉬보드 접속

인터넷 주소창에 Master 서버IP:Port 입력하여 접속

단, https 로 접속해야 함.

```
https://115.115.0.100:32222
```

필요한 패키지 설치

```
$ sudo apt-get install python3-pip
$ pip3 install pyinstaller

# mwcreate 파일 인코딩
$ pyinstaller -F /home/ostm/k8s/manage/example/mwcreate.py
$ rm /home/ostm/k8s/manage/example/mwcreate.py && rm /home/ostm/k8s/manage/
example/mwcreate.spec
$ rm -rf /home/ostm/k8s/manage/example/build/ && rm -rf /home/ostm/k8s/manage/
example/__pycache__
```

NFS MOUNT 설정

```
$ sudo vim /etc/exports
############################
# <경로>      <ip>(fsid=1,rw,no_root_squash)
/home/         *(fsid=1,rw,no_root_squash)
############################
```

만약 Master node 에 GPU가 있다면

```
$ kubectl taint nodes --all node-role.kubernetes.io/k8s-master-

# 확인
$ kubectl describe node k8s-master | grep Taints

# 원복 방법
kubectl taint nodes k8s-master node-role.kubernetes.io/k8s-master:NoSchedule

# nvidia-docker 설치
$ cd ./install/6-nvidia/nvidia-docker/
$ sudo dpkg -i ./*.deb
```

6. Docker private hub 구축

Docker daemon.json 파일 수정
공인아이피 수정 필요

```
$ sudo vi /etc/docker/daemon.json
#######################################################
# ip는 docker private hub를 서비스하는 서버의 ip를 입력
{
        "insecure-registries" : ["<마스터 공인아이피>:8993","ost-hub:8993"]
}
#######################################################

$ sudo systemctl restart docker


# 모든 노드에 동일하게 적용
```

```
$ scp /etc/docker/daemon.json root@<ip>:/etc/docker/daemon.json
$ pdsh -R ssh -w node[1-3] "systemctl restart docker"
```

인증서 생성

```
$ mkdir -p ~/k8s/manage/cert/docker/
$ openssl req \
  -newkey rsa:4096 -nodes -sha256 -keyout ~/k8s/manage/cert/docker/domain.key \
  -x509 -days 36500 -out ~/k8s/manage/cert/docker/domain.crt
  """
Country Name (2 letter code) [AU]:KR
State or Province Name (full name) [Some-State]:Seoul
Locality Name (eg, city) []:Seoul
Organization Name (eg, company) [Internet Widgits Pty Ltd]:miruware
Organizational Unit Name (eg, section) []:ost-hub
Common Name (e.g. server FQDN or YOUR name) []:ost-hub
Email Address []:
  """

# 확인
$ openssl x509 -in ~/k8s/manage/cert/docker/domain.crt -noout -text
```

인증서 복사

```
$ sudo mkdir -p /etc/docker/certs.d/ost-hub:8993
$ sudo cp ~/k8s/manage/cert/docker/domain.crt \/etc/docker/certs.d/ost-
hub:8993/ca.crt

# 모든 노드에 인증서 배포
$ pdsh -R ssh -w node[1-3] "mkdir -p /etc/docker/certs.d/ost-hub:8993"
$ scp /etc/docker/certs.d/ost-hub:8993/ca.crt root@<ip>:/etc/docker/certs.d/
```

```
ost-hub:8993/ca.crt
```

Docker registry 서비스 컨테이너 생성

```
$ docker run -d \
 --restart=always \
 --name ost-hub  \
 -v ~/k8s/manage/cert/docker/:/certs \
 -e REGISTRY_HTTP_ADDR=0.0.0.0:8993 \
 -e REGISTRY_HTTP_TLS_CERTIFICATE=/certs/domain.crt \
 -e REGISTRY_HTTP_TLS_KEY=/certs/domain.key \
 -v /home/registry:/var/lib/registry \
 -p 8993:8993 \
 registry:latest
```

Docker image 업로드 테스트

```
$ docker pull nvidia/cuda:11.3.0-base-ubuntu20.04
$ docker tag nvidia/cuda:11.3.0-base-ubuntu20.04 ost-hub:8993/test:20220304
$ docker push ost-hub:8993/test:20220304

# repository에 업로드 되었는지 확인
$ curl --cacert /etc/docker/certs.d/ost-hub\:8993/ca.crt -X GET https://ost-hub:8993/v2/_catalog
{"repositories":["test"]}
```

Kubernetes Pod으로 Docker web service 구축
공인아이피 수정 필요

```
$ kubectl create secret docker-registry miruware --docker-server=https://ost-hub:8993 --docker-username=miruware --docker-password="miruware!" --docker-email="mw@miruware.com"

# 설정값 확인
$ kubectl get secret miruware --output=yaml
$ kubectl get secret miruware --output="jsonpath={.data.\.dockerconfigjson}" | base64 -d

# IP 변경 필요
$ docker run -it -d -p 52222:8080 \
--restart=always \
--name ost-whub \
--link mw-ost \
-e REGISTRY_URL=https://100.100.0.70:8993/v2 \
-e REGISTRY_TRUST_ANY_SSL=true \
-e REGISTRY_NAME=ost-hub:8993 \
-e REGISTRY_BASIC_AUTH="dGVzdDp0ZXN0" \
hyper/docker-registry-web:latest

# 로그 확인
$ docker logs ost-whub
```

Compute Node

Kubernetes 노드 연결

```
# kubeadm join token 입력
$ sudo kubeadm join 100.100.0.70:6443 --token pbtzg9.89mfnsedcmtfsve4 \
```

```
  --discovery-token-ca-cert-hash
sha256:6dd5137005e29dd3b5dbeb2ae8eebc756e4aa4ec84c503004099168408d029bf
```

GPU 서버일 경우 Nvidia Plugin 추가 및 Nvidia Docker 설치

```
$ kubectl create -f https://raw.githubusercontent.com/NVIDIA/k8s-device-plugin/
v0.10.0/nvidia-device-plugin.yml

$ curl -s -L https://nvidia.github.io/nvidia-docker/gpgkey | sudo apt-key add -
&&
distribution=$(. /etc/os-release;echo $ID$VERSION_ID) &&
curl -s -L https://nvidia.github.io/nvidia-docker/$distribution/nvidia-
docker.list | sudo tee /etc/apt/sources.list.d/nvidia-docker.list &&
sudo apt-get update &&
curl -s -L https://nvidia.github.io/nvidia-docker/gpgkey | sudo apt-key add -
&&
sudo apt-get install nvidia-docker2 -y &&
sudo pkill -SIGHUP dockerd
```

Docker daemon runtime 변경

```
$ sudo vim /etc/docker/daemon.json
#################################
# defaults-runtime 줄 추가
{
    "default-runtime": "nvidia",
    "runtimes": {
        "nvidia": {
            "path": "nvidia-container-runtime",
            "runtimeArgs": []
        }
    }
}
#################################
```

Docker private hub 구축

Docker daemon.json 파일 수정
공인아이피 수정 필요

```
$ sudo vi /etc/docker/daemon.json
#####################################################
# ip는 docker private hub를 서비스하는 서버의 ip를 입력
{
        "insecure-registries" : ["<마스터 공인아이피>:8993","ost-hub:8993"]
}
#####################################################

$ sudo systemctl restart docker
```

Docker image pull test

```
$ docker pull ost-hub:8993/test:20220304
```