Openstack 구축 (ussuri)

- 외부망
  - master : 100.100.0.30
  - gpu server (Titan black) : 100.100.0.35
  - intel vm server : 100.100.0.36
  - Openstack VIP IP : 100.100.0.31
- SSH Port = 4400, 22
- 계정정보
  - ID : miruware (Sudo User)
  - PW : miruware!

서버 정보

```
# Master (1U CPU server)
eth0 : 100.100.0.30/24
eth1 : x (오픈스택 인스턴스 게이트웨이로 사용 100.100.0.31)
internal : 120.0.0.100/24

# gpu-1 (Titan black server)
eth0 : 100.100.0.35/24
internal : 120.0.0.101/24

# intel vm server
eth0 : 100.100.0.36/24
internal : 120.0.0.102/24
```

==모든 작업은 root 계정으로 진행한다.==

Network 장치명 고정

/etc/default/grub 수정

```
$ vim /etc/default/grub
################################################
# 아래 라인을 수정한다.
GRUB_CMDLINE_LINUX="net.ifnames=0 biosdevname=0"
```

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

$ update-grub
$ reboot
```

/etc/udev/rules.d/70-persistent-net.rules 생성

master

```
$ vim /etc/udev/rules.d/70-persistent-net.rules
########################################################
# ATTR{address}== 부분에 mac address를 입력하고 NAME에 원하는 장치명을 입력한다.
SUBSYSTEM=="net", ACTION=="add", DRIVERS=="?*", ATTR{address}
=="00:15:17:ee:6c:b8", ATTR{dev_id}=="0x0", ATTR{type}=="1", KERNEL=="eth*",
NAME="eth0"
SUBSYSTEM=="net", ACTION=="add", DRIVERS=="?*", ATTR{address}
=="00:15:17:ee:6c:b9", ATTR{dev_id}=="0x0", ATTR{type}=="1", KERNEL=="eth*",
NAME="eth1"
SUBSYSTEM=="net", ACTION=="add", DRIVERS=="?*", ATTR{address}
=="a0:36:9f:8c:6b:64", ATTR{dev_id}=="0x0", ATTR{type}=="1", KERNEL=="eth*",
NAME="eth2"
SUBSYSTEM=="net", ACTION=="add", DRIVERS=="?*", ATTR{address}
=="a0:36:9f:8c:6b:65", ATTR{dev_id}=="0x0", ATTR{type}=="1", KERNEL=="eth*",
NAME="eth3"
SUBSYSTEM=="net", ACTION=="add", DRIVERS=="?*", ATTR{address}
=="a0:36:9f:8c:6b:66", ATTR{dev_id}=="0x0", ATTR{type}=="1", KERNEL=="eth*",
NAME="eth4"
SUBSYSTEM=="net", ACTION=="add", DRIVERS=="?*", ATTR{address}
=="a0:36:9f:8c:6b:67", ATTR{dev_id}=="0x0", ATTR{type}=="1", KERNEL=="eth*",
NAME="internal"
########################################################

$ reboot
```

node01 (titan black)

```
$ vim /etc/udev/rules.d/70-persistent-net.rules
########################################################
SUBSYSTEM=="net", ACTION=="add", DRIVERS=="?*", ATTR{address}
```

```
=="00:1e:67:51:30:dc", ATTR{dev_id}=="0x0", ATTR{type}=="1", KERNEL=="eth*",
NAME="eth0"
SUBSYSTEM=="net", ACTION=="add", DRIVERS=="?*", ATTR{address}
=="00:1e:67:51:30:dd", ATTR{dev_id}=="0x0", ATTR{type}=="1", KERNEL=="eth*",
NAME="eth1"
SUBSYSTEM=="net", ACTION=="add", DRIVERS=="?*", ATTR{address}
=="00:1e:67:51:30:de", ATTR{dev_id}=="0x0", ATTR{type}=="1", KERNEL=="eth*",
NAME="eth2"
SUBSYSTEM=="net", ACTION=="add", DRIVERS=="?*", ATTR{address}
=="00:1e:67:51:30:df", ATTR{dev_id}=="0x0", ATTR{type}=="1", KERNEL=="eth*",
NAME="internal"
#########################################################

$ reboot
```

node02 (intel)

```
$ vim /etc/udev/rules.d/70-persistent-net.rules
#########################################################
SUBSYSTEM=="net", ACTION=="add", DRIVERS=="?*", ATTR{address}
=="a4:bf:01:05:95:42", ATTR{dev_id}=="0x0", ATTR{type}=="1", KERNEL=="eth*",
NAME="eth0"
SUBSYSTEM=="net", ACTION=="add", DRIVERS=="?*", ATTR{address}
=="a4:bf:01:05:95:43", ATTR{dev_id}=="0x0", ATTR{type}=="1", KERNEL=="eth*",
NAME="internal"
#########################################################

$ reboot
```

netplan 으로 IP Static 설정

```
$ vim /etc/netplan/01-network-manager-all.yaml
############################################
network:
  version: 2
  renderer: networkd
  ethernets:
          eth0:
```

```
                addresses: [100.100.0.30/24]
                gateway4: 100.100.0.1
                nameservers:
                        addresses: [8.8.8.8]
        eth1:
                dhcp4: false
        internal:
                addresses: [120.0.0.100/24]
##############################################
```

apt update && apt upgrade

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

자동 업데이트 끄기

```
$ vim /etc/apt/apt.conf.d/10periodic
################################################
# 아래 줄 마지막 1을 0으로 변경
APT::Periodic::Unattended-Upgrade "0";
################################################
```

방화벽 끄기

```
$ ufw disable
>>> Firewall stopped and disabled on system startup

# 제대로 비활성화 되었는지 확인
$ ufw status
>>> Status: inactive
```

/etc/hosts 파일 수정

```
127.0.0.1 localhost

# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters

# MW-OST
120.0.0.100 master
#GPU NODE
120.0.0.101 gnode01
#CPU NODE
120.0.0.102 cnode01
```

root 계정 활성화 및 SSH root 접속 허용

```
# root 계정 활성화
$ passwd
(password 입력)

# ssh root 접속 허용
$ vim /etc/ssh/sshd_config
################################################
# 내용 수정
PermitRootLogin yes
################################################

$ systemctl restart sshd.service
```

SSH KEY 배포

마스터 서버에서만 실행

```
# ssh-keygen
$ ssh-keygen -t rsa

# root 계정에서 비밀번호 없이 접속 가능하도록 설정
$ ssh-copy-id gnode01
$ ssh-copy-id cnode01
```

기본 패키지 설치

```
$ apt-get install software-properties-common
$ apt-add-repository ppa:ansible/ansible
$ apt-get update
$ apt-get install python-pip-whl python3-pip
$ pip3 install pip -U
```

Docker 설치

```
$ sudo apt-get update &&
sudo apt-get install apt-transport-https ca-certificates curl gnupg-agent
software-properties-common -y &&
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add - &&
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 -y &&
sudo usermod -aG docker $USER
```

오픈스택 구축

- Requirements
  - [x] ~~Master 서버의 etchosts 파일 수정~~
  - [x] ~~방화벽 해제~~
  - [x] ~~자동 업데이트 비활성화~~
  - [x] ~~모든 서버의 root 계정 활성화 및 SSH root 접속 허용 (ssh-keygen & ssh-copy-id)~~
  - [x] ~~모든 서버의 SSH 22번 포트 활성화~~
  - [x] ~~모든 서버의 네트워크 장치명 통일 (Openstack 구축용 장치만 통일해도 됨)~~
  - [x] ~~Master 서버의 Instance 통신용 Ethernet 케이블 장착 여부 (2번째 네트워크 장치 이용)~~

모든 작업은 Master Node 에서 Root 계정으로 실행

Ansible 설치

```
$ apt-add-repository ppa:ansible/ansible
$ apt-get update
$ apt-get install ansible=2.9.6+dfsg-1 libguestfs-tools

# pip3 와 pip 헷갈리지 않도록 주의. pip3로 update 한 pip를 사용하는것.
$ pip install kolla==10.1.0 kolla-ansible==10.1.0 tox gitpython pbr requests
jinja2=3.0.3 oslo_config -U
$ pip install python-openstackclient python-glanceclient python-neutronclient
-U
```

ansible.cfg 파일 수정

```
$ vi /etc/ansible/ansible.cfg
##################################################
# 다 주석처리 되어있으므로 [dafaults], [ssh_connection] 에 내용 추가 후 저장
[defaults]
forks=100
host_key_checking=False

[ssh_connection]
pipelining=True
##################################################
```

kolla-ansible 파일 복사

```
$ mkdir -p ~/kolla-ansible
$ cp /usr/local/share/kolla-ansible/ansible/inventory/* ~/kolla-ansible/

$ mkdir -p /etc/kolla
$ cp -r /usr/local/share/kolla-ansible/etc_examples/kolla/* /etc/kolla/
```

Kolla-ansible Config 파일 수정

```
$ vi ~/kolla-ansible/multinode
############################################################
# 파일 내용 수정
[control]
# These hostname must be resolvable from your deployment host
master     # master node 호스트네임

# The above can also be specified as follows:
#control[01:03]     ansible_user=kolla

# The network nodes are where your l3-agent and loadbalancers will run
# This can be the same as a host in the control group
[network]
master       # master node 호스트네임

[compute]
gnode01       # Slave node 호스트네임
cnode01

[monitoring]
master       # master node 호스트네임

# When compute nodes and control nodes use different interfaces,
# you need to comment out "api_interface" and other interfaces from the
globals.yml
# and specify like below:
#compute01 neutron_external_interface=eth0 api_interface=em1
```

```
storage_interface=em1 tunnel_interface=em1


[storage]
master       # master node 호스트네임


[deployment]
master       ansible_connection=local       # master node 호스트네임
############################################################
```

Kolla-ansible 웹 접속 비밀번호 생성

```
$ cd /etc/kolla &&  kolla-genpwd


$ mkdir -p /backup/openstack


$ cp /etc/kolla/passwords.yml /backup/openstack/


$ vi  /etc/kolla/passwords.yml
############################################################
# 검색하여 해당 줄만 수정.
/keystone_admin_password: miruware!
############################################################
```

Openstack 서비스 리스트 config

```
$ vi  /etc/kolla/globals.yml
############################################################
# 검색하여 주석 제거 또는 수정
kolla_base_distro: "ubuntu"
kolla_install_type: "binary"
openstack_release: "ussuri"


 #->오픈스택 서비스 네트워크 대역중 비는것 VIP사용 ## 호로보드에서 사용
kolla_internal_vip_address: "120.0.0.200"


network_interface: "internal"
neutron_external_interface: "eth1"
```

```
neutron_plugin_agent: "openvswitch"



enable_openstack_core: "yes"
enable_glance: "{{ enable_openstack_core | bool }}"
enable_haproxy: "yes"
enable_keystone: "{{ enable_openstack_core | bool }}"
enable_mariadb: "yes"
enable_memcached: "yes"
enable_neutron: "{{ enable_openstack_core | bool }}"
enable_nova: "{{ enable_openstack_core | bool }}"
##############################################################
```

Openstack 배포

```
$ kolla-ansible -i multinode pull

# python3 를 python 으로 사용할 수 있도록 설정
$ update-alternatives --install /usr/bin/python python /usr/bin/python3 100

# ssh 기본 22번 포트로 통신 하므로 만약 변경 했을 시 22번 포트 추가해줘야함
$ kolla-ansible -i ~/kolla-ansible/multinode bootstrap-servers

# Openstack 배포
$ kolla-ansible -i ~/kolla-ansible/multinode deploy
```

배포 완료 후 네트워크 설정 진행

Openstack 네트워크 설정

```
$ kolla-ansible post-deploy
$ . /etc/kolla/admin-openrc.sh
$ cp /etc/kolla/admin-openrc.sh ~/
$ . /usr/local/share/kolla-ansible/init-runonce

$ vi /etc/kolla/haproxy/services.d/horizon.cfg
```

```
############################################################
# 변경하지 말고 추가할것.
bind 100.0.0.30:30000 #->  마스터노트 외부아이피 : 접속할 포트
############################################################

$ docker restart haproxy
$ docker restart horizon
```

인스턴스와 통신하기 위해 route 추가

```
$ route add -net 30.0.0.0/24 gw 100.100.0.31
```

dashboard 접속

```
http://100.100.0.30:30000
```

ID : admin

PW : miruware!

Dashboard 접속 후

(관리 - 네트워크 - 네트워크) 로 이동

external-net 과 internal-net 을 생성한다.

==[+네트워크 생성] 클릭==

external-net

- 네트워크
  - 이름 : external-net
  - 프로젝트 선택 : admin
  - 공급자 네트워크 유형 : flat
  - 물리적인 네트워크 : physnet1 (Deault = physnet1)

공급자 네트워크 유형 및 물리적 네트워크는 *etc*kolla*neutron-server*ml2_conf.ini 파일에서 정의

- [x] ~~관리 상태 활성화~~
- [x] ~~공유~~
- [x] ~~외부 네트워크~~
- [x] ~~서브넷 생성~~

    - 가용구역 힌트 : nova

- 서브넷
    - 서브넷 이름 : external-sub
    - 네트워크 주소 : 100.100.0.0/24 (인터넷 망의 Subnet mask)
    - IP 버전 : IPv4
    - 게이트웨이 IP : 100.100.0.1 (외부망 게이트웨이)
    - [ ] 게이트웨이 비활성
    - [ ]
- 서브넷 세부정보
    - [ ] DHCP 사용
    - Pools 할당 : 100.100.0.31,100.100.0.32 (range)
    - DNS 네임 서버 : 8.8.8.8
    - 호스트 경로 : x

internal-net

- 네트워크
    - 이름 : internal-net
    - 프로젝트 : admin
    - 공급자 네트워크 유형 : VXLAN
    - 구분 ID : 1
    - [x] 관리 상태 활성화
    - [ ] 공유
    - [ ] 외부 네트워크
    - [x] 서브넷 생성
    - 가용구역 힌트 : nova

- 서브넷
    - 서브넷 이름 : internal-subnet
    - 네트워크 주소 : 30.0.0.0/24
    - IP 버전 : IPv4
    - 게이트웨이 IP : 30.0.0.1
    - [ ] 게이트웨이 비활성

- 서브넷 세부정보
    - [x] ~~DHCP 사용~~
    - Pools 할당 : 30.0.0.2,30.0.0.254
    - DNS 네임 서버 : 8.8.8.8
    - 호스트 경로 : x

(관리 - 네트워크 - 네트워크 - 라우터) 로 이동

==[+ 라우터 생성] 클릭==

- 라우터 이름 : external-route
- 프로젝트 : admin

- [x] ~~관리 상태 활성화~~

- 외부 네트워크 : external-net

- [x] ~~SNAT 활성화~~

(프로젝트 - 네트워크 - 라우터 - external-route - 인터페이스) 로 이동

==[+ 인터페이스 추가] 클릭==

- 서브넷 : internal-net: 30.0.0.0/24
- IP 주소 (옵션) : 30.0.0.1

(관리 - 네트워크 - 네트워크 - external-net - 포트) 로 이동

==[+ 포트 생성] 클릭==

network:floatingip

- [x] ~~관리 상태 활성화~~

- 장치 소유자 : network:floatingip
- IP 주소 또는 서브넷을 지정합니다 : 고정 IP 주소
- 고정 IP 주소 : 100.100.0.32
- 바인딩: VNIC 유형 : 표준
  - [ ] 포트 보안
- 바인딩:호스트 : master

(관리 - 시스템 - 디폴트 - Compute 할당량) 으로 이동

[업데이트 기본] 클릭하여 모든 설정값에 0 한개 더 추가

(프로젝트 - Compute - 키페어) 로 이동

기존 생성되어 있는 MyKey 삭제 후 ==[+ 키 페어 생성]== 클릭

- 키 페어 이름 : miruware
- 키 유형 : SSH 키

이미지 생성

(프로젝트 - Compute - 이미지) 로 이동

==[+ 이미지 생성]== 클릭

- 파일 : ==ISO 파일이 아닌 img 파일로 생성 해야함==
- 포맷 : QCOW2
- 가시성 : 공용

Trouble Shooting

Start Docker Error

kolla-ansible -i ~*kolla-ansible*multinode bootstrap-servers 도중 start docker 에러가 나는 경우 *etc*apt*source*list.d 에 다른 OS 버전의 Docker repository가 생성되어 발생함.
Master와 각 노드에 Docker를 직접 설치 후 실행하였더니 해결 되었음.

Docker cgroup error

Docker Cgroup Warnning 이 발생하는 경우 아래 내용을 수정하여 없앨 수 있다.

```
$ sudo vim /usr/lib/systemd/system/docker.service
##################################
# execStart= 라인을 찾아 뒤에 아래 내용 추가
--exec-opt native.cgroupdriver=systemd
##################################

$ sudo systemctl daemon-reload
$ sudo systemctl restart docker
$ sudo docker info | grep -i cgroup
>>> Cgroup Driver: systemd
```

haproxy : Waiting for virtual IP to appear

아래 파일 내용을 수정

```
$ sudo vi /etc/kolla/globals.yml
####################################
keepalived_virtual_router_id: "251"
####################################

$ sudo kolla-genpwd
```

Multinode pull : Could not match supplied host pattern

multinode pull 에서 Could noe match supplied host patter 에러가 발생할 경우 명령어에서 multinode 파일 경로를 절대 경로로 지정한다.

```
$ kolla-ansible -i ~/kolla-ansible/multinode pull
```