**Quantum Cloud 설치 매뉴얼**

0. 시스템 전체 구성 계획

1) 서비스 명 : Quantum Cloud Service ( OpenStack Version : Wallaby )

HW SPEC :

- CPU : AMD EPYC 7282 16-Core Processor 2EA

- RAM : 256GB

- HDD :

-- 500GB SSD 2EA (RAID1) : OS

-- 1TB SSD 2EA : CEPH LOG & DB

-- 4TB SSD 1EA : CEPH SSD

-- 8TB HDD 3EA : CEPH HDD

- NIC : 4port 10GB NIC 1EA

- Network Switch : Netgear L2 Switch Hub 4 VLAN 1EA

2) 노드 구성 : 3 Node AAA HA

* OS : Ubuntu 20.04 LTS
* Partition : / 300GB

/data 146 GB

- qcs01 : controller active, compute active, networking active , storage active

- qcs02 : controller active, compute active, networking active, storage active

- qcs03 : controller active, compute active, networking active, storage active

3) 설치 참고 URL : https://docs.openstack.org/ha-guide/intro-ha.html

4) 사전 준비 : all node

# vi /etc/sysctl.conf

net.ipv4.ip\_nonlocal\_bind = 1 # vip로 listen하기 위해서 필요

net.ipv4.ip\_forward = 1

net.ipv4.conf.default.rp\_filter = 0

net.ipv4.conf.all.rp\_filter = 0

# sysctl –p ( 부팅없이 바로 적용시 필요)

1. HA 관련 구성

1) 3개 노드가 모두 패스워드 없이 root 계정으로 ssh 접속 가능하도록 설정한다.

2) NTP 설정

# root@qcs01:~# qcs apt-get -y install chrony

# 설정 파일 : /etc/chrony/chrony.conf

3) Timezone Setting : cmd timedatectl set-timezone Asia/Seoul

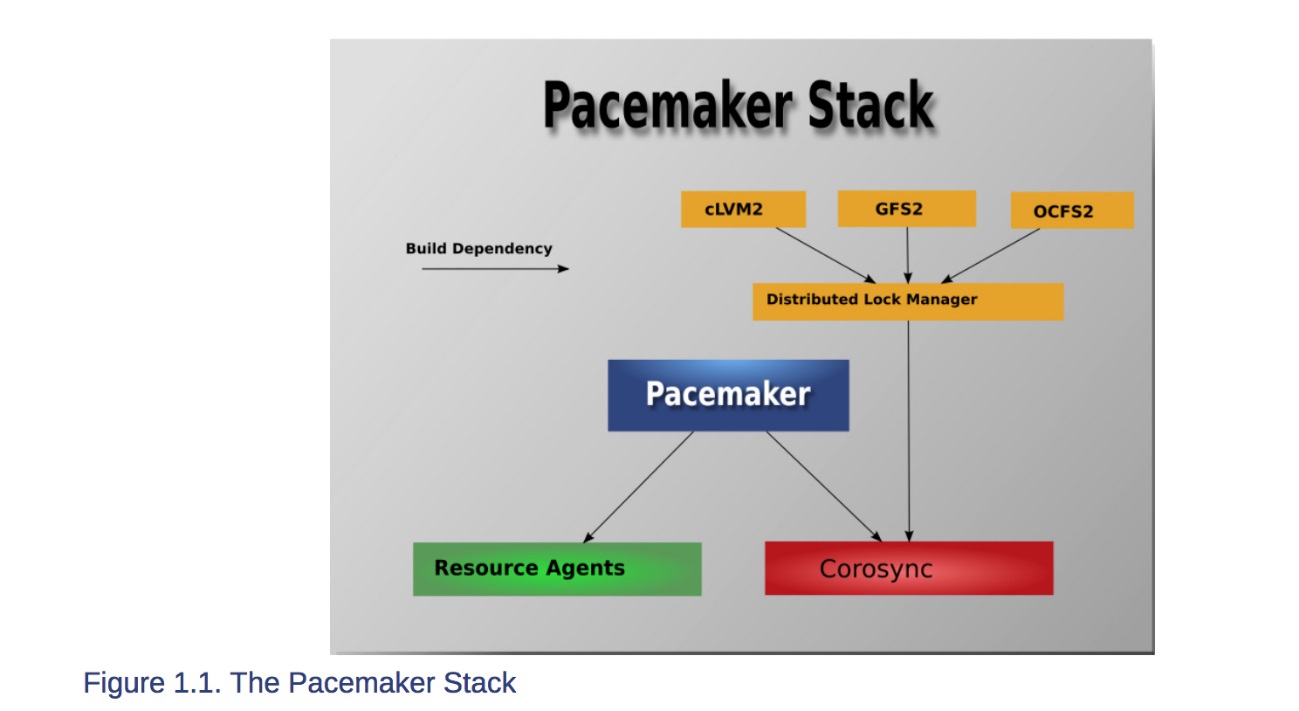
# 설정 참고 URL :

https://joungkyun.gitbooks.io/annyung-3-user-guide/chapter6-chrony.html

<https://docs.openstack.org/ocata/install-guide-ubuntu/environment-ntp-controller.html>

# 설정 확인 방법 : qcs02,qcs03 # chronyc sources –v

3) corosync , pacemaker 구성도



4) corosync, pacemaker 구성 및 CLI 사용법

- # cmd apt-get install -y corosync crmsh pcs

- corosync 설정 :

-- /etc/corosync/corosync.conf 참고

-- # corosync-keygen (/etc/corosync/authkey 를 생성한다.)

-- /etc/corosync/authkey를 모든 노드에서 공유해야 함. (1번노드에서 만들어 나머지 노드에 scp 한다.)

-- 실행 방법 : # service corosync start / stop /status

-- 설정 확인 방법 : # corosync-cfgtool -s

- pacemake 설정

-- corosync 설정이 끝나면 corosync 와 pacemaker 서비스를 재시작한다.

-- /etc/pacemaker/authkey 를 모든 노드에서 공유해야 함.

-- 참고 URL : chrome-extension://oemmndcbldboiebfnladdacbdfmadadm/http://clusterlabs.org/pacemaker/doc/en-US/Pacemaker/1.1/pdf/Clusters\_from\_Scratch/Pacemaker-1.1-Clusters\_from\_Scratch-en-US.pdf

-- #passwd hacluster => bns2000 으로 변경 후

-- #pcs cluster auth qcs01 qcs02 qcs03 -u hacluster -p qcs2021 --force

-- #pcs cluster setup --force --name QuantumCloud qcs01 qcs02 qcs03

-- #pcs property set pe-warn-series-max=1000 pe-input-series-max=1000 pe-error-series-max=1000 cluster-recheck-interval=5min

--# pcs property set stonith-enabled=false

--# journalctl | grep -i error //서비스 오류 검사

--# crm\_verify -L //설정 오류 검사

-- 대표 IP 설정

# pcs resource create ClusterIP ocf:heartbeat:IPaddr2 \ ip=192.168.10.70 cidr\_netmask=24 op monitor interval=30s

-- pcs CLI

# pcs status or # crm\_mon ( 클러스터 상태 정보 확인 )

# pcs cluster start/stop --all ( 전노드 시작/정지 )

# pcs cluster start/stop qcs01 ( 개별노드 시작/정지 ) => 테스트시 유용

# pcs resource ( 각종 리소스 추가 삭제시 이용 )

# pcs config

# systemctl enable corosync.service

# systemctl enable pacemaker.service

# apt install openstack-resource-agents ( OCF OpenStack Resource Agents )

참고 URL : https://access.redhat.com/documentation/en-us/red\_hat\_enterprise\_linux/6/html-single/configuring\_the\_red\_hat\_high\_availability\_add-on\_with\_pacemaker/index

https://www.digitalocean.com/community/tutorials/how-to-create-a-high-availability-setup-with-corosync-pacemaker-and-floating-ips-on-ubuntu-14-04

4) memcached 설치

# apt install memcached

환경설정 파일 : rcat /etc/memcached.conf

-d

logfile /var/log/memcached.log

-m 1024

-p 11211

-u memcache

-l qcs01,::1

Memcached\_servers = qcs01:11211,qcs02:11211,qcs03:11211

5) DRBD 설치 ( glance 서비스를 위한 공유 파일 서비스 ) <= 불필요

# sudo add-apt-repository ppa:linbit/linbit-drbd9-stack

# sudo apt-get update

# apt-get install drbd-utils python-drbdmanage drbd-dkms

Drbdmanage를 이용해서 drbd cluster 설치

- drbd를 위한 lvm2 volume group 생성

새로운 볼륨을 만들 경우

Qcs1,2,3# vgcreate drbdpool /dev/sdb2(pv)

기존 볼륨을 사용할 경우

vi /etc/drbdmanaged.cfg

drbdctrl-vg = ubuntu-vg

- cluster 설정 (실패하면 lv삭제하고 다시해야하므로 주의 필요)

Qcs1# drbdmanage init 192.168.0.201 #master ip

Qcs1# drbdmanage add-node qcs2 192.168.0.202

Qcs1# drbdmanage add-node qcs3 192.168.0.203

- 공유할 볼륨 생성

# drbdmanage add-resource glance

# drbdmanage add-volume glance 10GB --deploy 3

- 생성된 디스크를 포맷하여 마운트 한다.

#mkfs.xfs /dev/drbd100

#mount /dev/drbd100 /data/glance

=> mount한 host가 primary가 된다. 동시에 두 host 에서 mount불가

(awesome!!)

qcs1# cd /data/glance

qcs1# touch a b c

qcs1# drbdadm status # 상태 확인

glance role:Primary

qcs1# umount /data/glance

qcs2# mount /dev/drbd100 /data/glance

qcs2# cd /data/glance

qcs2# ls

a b c

qcs2# drbdadm status

glance role:Primary

6) haproxy 설정 ( 로드밸런서 ) - OpenStack은 MSA 이므로 haproxy 설정 중요

-- 1.9 version 설치 방법

# cmd "apt-get install -y software-properties-common"

# cmd "add-apt-repository ppa:vbernat/haproxy-2.3"

# cmd "apt update"

# cmd "apt-get install -y haproxy"

-- /etc/haproxy/haproxy.cfg 수정 필요

-- # service haproxy start / stop / status

-- 이중화를 위해 crm resoucre에 등록한다.

2. Rabbit MQ 설치

1) Erlang 설치 => 주의 ( 의존성으로 gnome desktop이 설치됨 )

참고 : https://computingforgeeks.com/how-to-install-latest-erlang-on-ubuntu-18-04-lts/

# wget -O- https://packages.erlang-solutions.com/ubuntu/erlang\_solutions.asc | sudo apt-key add -

# echo "deb https://packages.erlang-solutions.com/ubuntu focal contrib" | sudo tee /etc/apt/sources.list.d/rabbitmq.list

# apt update ; apt -y install erlang

2) Rabbit MQ Server 3.8.16 설치

# wget -O- https://www.rabbitmq.com/rabbitmq-release-signing-key.asc | sudo apt-key add -

# echo "deb https://dl.bintray.com/rabbitmq/debian $(lsb\_release -sc) main" | sudo tee /etc/apt/sources.list.d/rabbitmq.list

# apt update ; apt -y install rabbitmq-server

참고 URL : https://docs.openstack.org/ha-guide/shared-messaging.html#rabbitmq-install

# cmd "rabbitmq-plugins enable rabbitmq\_management" # 통계를 위한 모듈 설치

- 3 node HA 구성

1) .erlang.cookie 공유

root@qcs01# qcs "service rabbitmq-server stop"

root@qcs01:/var/lib/rabbitmq#scp .erlang.cookie qcs02:`pwd`

root@qcs01:/var/lib/rabbitmq#scp .erlang.cookie qcs03:`pwd`

root@qcs01# qcs "service rabbitmq-server start"

2) cluster join : qcs01이 master가 되고 qcs02,qcs03가 qcs1 에 join한다.

root@qcs02 # rabbitmqctl stop\_app

root@qcs02 # rabbitmqctl join\_cluster --ram rabbit@qcs01

root@qcs02 # rabbitmqctl start\_app

=> qcs02는 qcs01에 join할 때 저장모드를 RAM으로 지정한다.

root@qcs03 # rabbitmqctl stop\_app

root@qcs03 # rabbitmqctl join\_cluster rabbit@qcs01

root@qcs03 # rabbitmqctl start\_app

=>qcs03는 qcs1에 join할 때 저장 모드를 디폴트(disc)로 Join한다.

root@qcs01 # rabbitmqctl stop\_app

root@qcs01 # rabbitmqctl change\_cluster\_node\_type ram

root@qcs01 # rabbitmqctl start\_app

=>qcs01은 저장모드를 ram으로 변경한다.

root@qcs01:/var/lib/rabbitmq# rabbitmqctl cluster\_status : 상태 확인

3) HA & Mirroring Queue 설정

# rabbitmqctl set\_policy ha-all '^(?!amq\.).\*' '{"ha-mode": "all"}'

- /etc/rabbitmq/rabbitmq-env.conf 에 노드 이름 추가

NODE\_IP\_ADDRESS=qcs01

- User 추가 :

# rabbitmqctl add\_user rabbit qcs2021

# rabbitmqctl add\_user openstack qcs2021

# rabbitmqctl set\_user\_tags rabbit administrator

- Admin page : https://mq.quantumcns.io , id/pass : rabbit/qcs2021

3. DB 이중화 ( Mariadb 10.3 Galera Cluster )

-- 설치 참고 : https://mariadb.com/kb/en/library/getting-started-with-mariadb-galera-cluster/

# sudo apt-get install -y mariadb-server mariadb-client

-- 환경 설정 : cluster.cnf를 각 노드에 맞게 추가한다.

# root@qcs1:/etc/mysql/conf.d# cat cluster.cnf

[mysqld]

query\_cache\_size=0

binlog\_format=ROW

default\_storage\_engine=innodb

innodb\_autoinc\_lock\_mode=2

query\_cache\_type=0

#bind-address = 192.168.10.83

#Galera Provider Configuration

wsrep\_on=ON

wsrep\_provider=/usr/lib/galera/libgalera\_smm.so

#wsrep\_provider\_options=”gcache.size=32G”

#Galera Cluster Configuration

wsrep\_cluster\_name="QuantumCloud"

wsrep\_cluster\_address="gcomm://192.168.10.83,192.168.10.84,192.168.10.85"

#Galera Synchronization Congifuration

wsrep\_sst\_method=rsync

#wsrep\_sst\_auth=user:pass

#Galera Node Configuration

wsrep\_node\_address="192.168.10.83"

wsrep\_node\_name="qcs01"

character-set-server=utf8

-- db charater set을 모두 utf8mb4 -> utf8로 맞춘다. ( /etc/mysql/mariadb.conf.d/50-client.cnf, 50-mysql-clients.cnf , 50-server.cnf )

-- haproxy로 서비스 해야 하므로 /etc/mysql/mariadb.conf.d/50-server.conf에서 bind-address = < 각 노드 ip > 로 수정한다.

-- 모든 IP에서 접속 가능하도록 mysql 환경 설정 ( 모든 클러스터 노드에 적용한다)

mysql> SELECT Host,User,authentication\_string FROM mysql.user;

mysql> INSERT INTO mysql.user (host,user,authentication\_string,ssl\_cipher, x509\_issuer, x509\_subject) VALUES ('%','root',password('bns2000'),'','','');

mysql> GRANT ALL PRIVILEGES ON \*.\* TO 'root'@'%' IDENTIFIED BY 'bns2000'; mysql> FLUSH PRIVILEGES;

-- Health check 를 위해 haproxy 유저를 추가하고 비밀번호 없이 로그인이 되도록 한다. (모든 노드에 동일하게 적용한다 )

mysql> INSERT INTO mysql.user (host,user,authentication\_string,ssl\_cipher, x509\_issuer, x509\_subject) VALUES ('%','haproxy',password('bns2000'),'','','');

mysql> FLUSH PRIVILEGES;

mysql> SET PASSWORD FOR 'haproxy'@'%' = PASSWORD('');

-- Master Node 실행 방법 : galera\_new\_cluster (마지막에 셧다운된 서버가 마스터)

식별 방법(mysql종료 되었을때) : cat /var/lib/mysql/grastate.dat | grep safe\_to\_bootstrap => 1

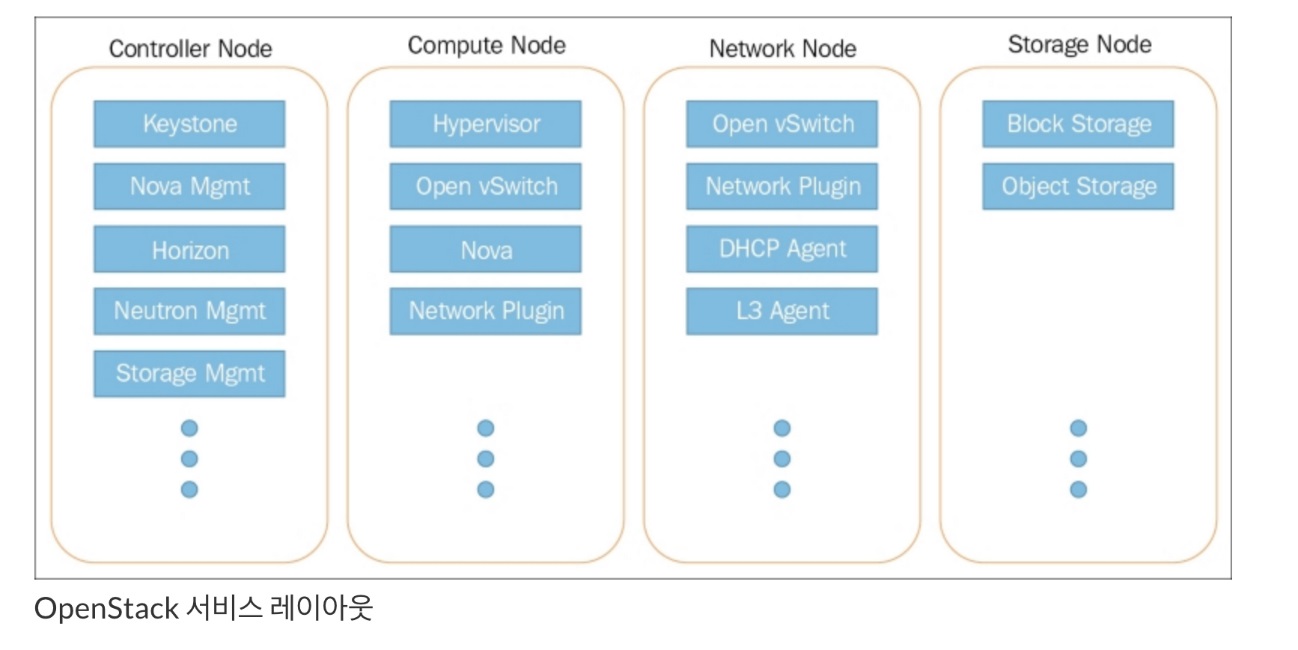
-- Slave Node 실행 방법 : systemctl ( start / stop ) mysql

-- 정상 동작 확인 방법 :

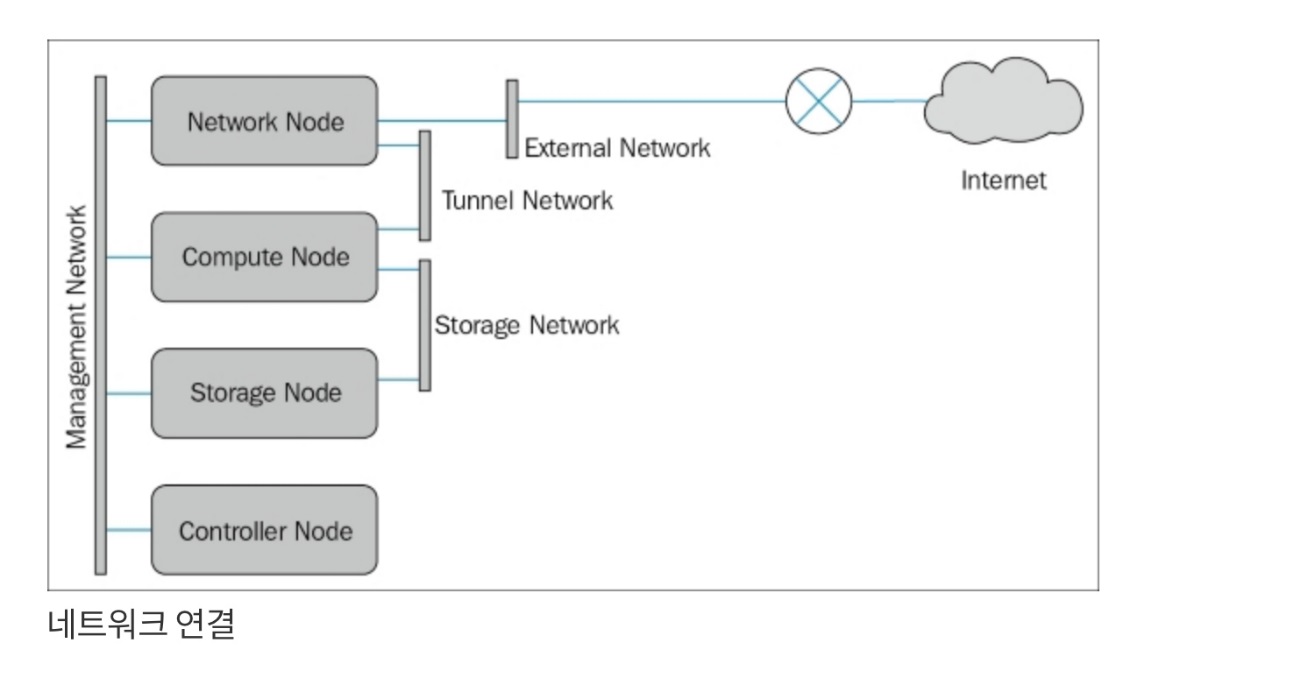
# mysql -u root -e "SHOW STATUS LIKE 'wsrep\_cluster\_size'" = 3 이면 정상

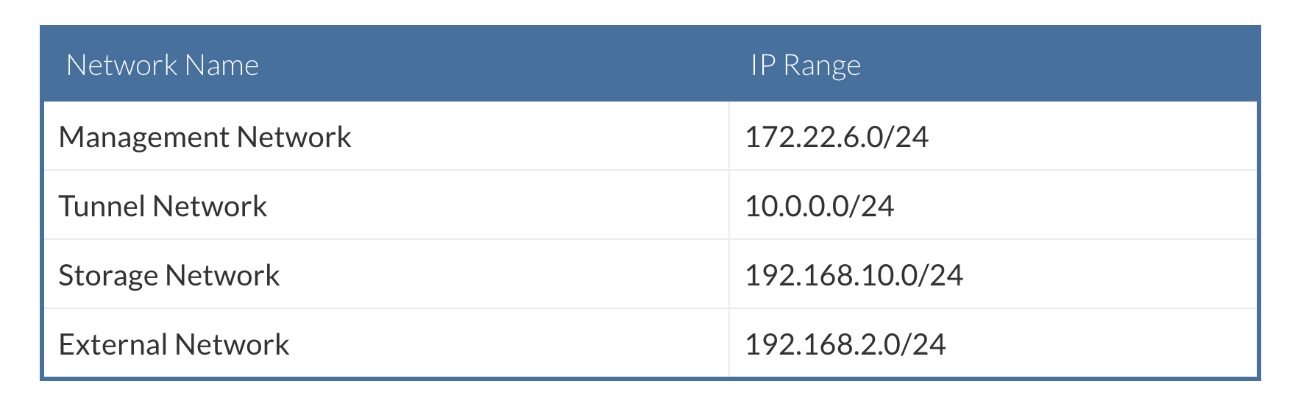
4. Open Stack 관련 설치

1) 시스템 구성 방안 ( 4 개의 서비스 노드 구성 필요 )

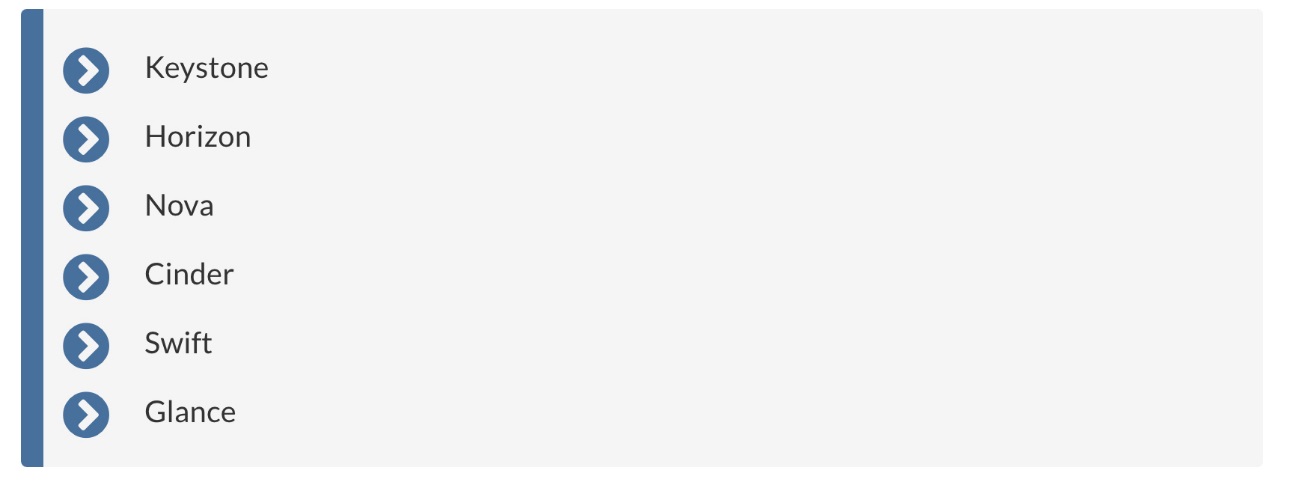


2) 네트웍 구성 방안

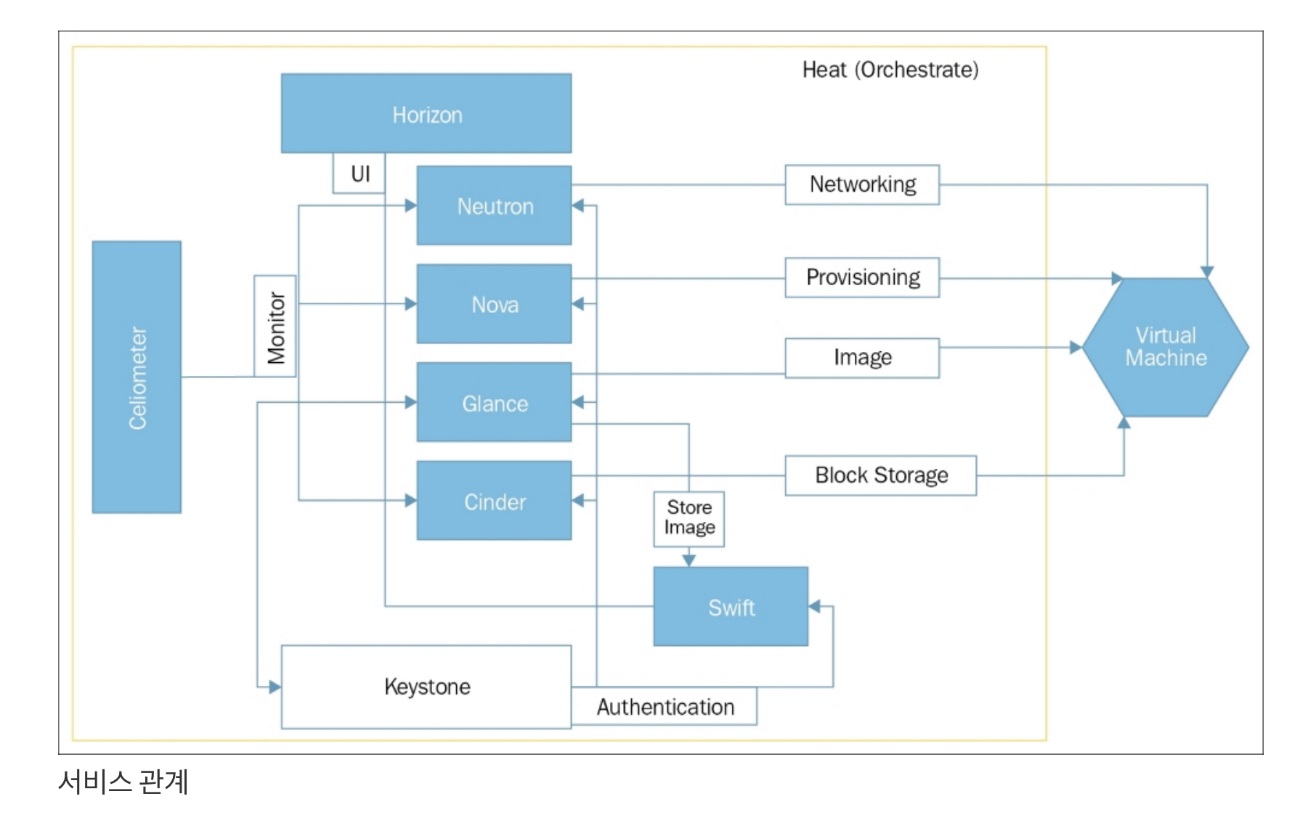




3) 구성해야 할 서비스



4) 전체 서비스 간 구성도



5) 모듈 설치

- 사전 준비

# apt install software-properties-common

# qcs "add-apt-repository cloud-archive:wallaby"

# qcs "apt update && apt dist-upgrade"

# qcs "apt-get install -y python3-openstackclient"

- 설치 순서 : keystone -> glance > nova > neutron -> cinder -> horizon

* Keystone

- Keystone : https://docs.openstack.org/keystone/wallaby/install/keystone-install-ubuntu.html

1. DB 작업

qcs01:~/sbin/keystone# mysql -u root < keystone.sql

### All node # apt -y install keystone

qcs1# su -s /bin/sh -c "keystone-manage db\_sync" keystone

-> 오류날때는 keystone db을 삭제 후 다시 생성한다.

# keystone-manage fernet\_setup --keystone-user keystone --keystone-group keystone

# keystone-manage credential\_setup --keystone-user keystone --keystone-group keystone

# keystone-manage bootstrap --bootstrap-password qcs2021 \

--bootstrap-admin-url http://controller:5000/v3/ \

--bootstrap-internal-url http://controller:5000/v3/ \

--bootstrap-public-url http://controller:5000/v3/ \

--bootstrap-region-id RegionOne

<Create a domain, projects, users, and roles>

$ openstack **domain** create --description "An Example Domain" example

$ openstack **project** create --domain default \

--description "Service Project" service

$ openstack project create --domain default \

--description "Demo Project" demo

$ openstack **user** create --domain default \

--password-prompt demo

$ openstack **role** create user

$ openstack role add --project demo --user demo user

2. apache 설정

대표 IP가 아닌 hostname 으로 listen 한다.

# vi /etc/apache2/sites-available/000-default.conf

# vi /etc/apache2/sites-available/keystone.conf

3. 서비스 정상 동작 확인

# lsof -Pni :5000

=> hostname으로 listen하면 정상

4. 다른 노드에 환경파일 복사 후 재시작

<keystone>

Qcs1:/etc/keystone# scp –rp \* qcs2:`pwd`

Qcs1:/etc/keystone# scp –rp \* qcs3:`pwd`

qcs2,qcs3:/etc/keystone # chown -R keystone:keystone credential-keys/

qcs2,qcs3:/etc/keystone # chown -R keystone:keystone fernet-keys/

<apache>

root@qcs1:/etc/apache2# scp ports.conf qcs2:`pwd`

root@qcs1:/etc/apache2# scp ports.conf qcs3:`pwd`

root@qcs1:/etc/apache2/sites-available# scp \*.conf qcs2:`pwd`

root@qcs1:/etc/apache2/sites-available# scp \*.conf qcs3:`pwd`

root@qcs3:/etc/apache2# vi ports.conf

root@qcs3:/etc/apache2/sites-available# vi 000-default.conf

root@qcs3:/etc/apache2/sites-available# vi keystone.conf

qcs1 -> qcs2 or qcs3

<서비스 재시작>

#service apache2 restart

5. Haproxy 에 등록

#vi /etc/haproxy/haproxy.cfg

listen keystone\_cluster

bind controller:5000

bind controller:5000

balance source

option tcpka

option httpchk

option tcplog

server qcs1 qcs1:5000 check inter 2000 rise 2 fall 5

server qcs2 qcs2:5000 check inter 2000 rise 2 fall 5

server qcs3 qcs3:5000 check inter 2000 rise 2 fall 5

* Neutron

[https://docs.openstack.org/neutron/wallaby/install/](https://docs.openstack.org/neutron/queens/install/)

1. DB 작업

~/sbin/neutron# mysql -u root < neutron.sql

2. neutron 유저 생성 및 권한 부여

# openstack user create --domain default --password-prompt neutron

# openstack role add --project service --user neutron admin

3. neutron service 생성

# openstack service create --name neutron \

--description "OpenStack Networking" network

4. API End point 생성

#openstack endpoint create --region RegionOne \

network public [http://controller:9696](http://deepthinq:9696)

#openstack endpoint create --region RegionOne \

network internal http://controller:9696

#openstack endpoint create --region RegionOne \

network admin http://controller:9696

3. neutron package 설치

# apt –y install neutron-server neutron-plugin-ml2 \

neutron-linuxbridge-agent neutron-dhcp-agent \

neutron-metadata-agent neutron-l3-agent

LB 관리 UI 모듈인 python-neutron-lbaas-dashboard 은 horizon 설치 후에 설치한다.

4. neutron 환경설정 (중요)

1) 환경파일 위치 : /etc/neutron

- neutron.conf, neutron\_lbaas.conf, dhcp\_agent.ini , lbaas\_agent.ini , metadata\_agent.ini , l3\_agent.ini 수정 필요

2) 환경파일 위치 : /etc/neutron/plugins/ml2

- ml2\_conf.ini, linuxbridge\_agent.ini 수정 필요

5. Haproxy 에 API 서비스 등록한다.

listen neutron\_api\_cluster

bind controller:9696

balance source

option tcpka

option httpchk

option tcplog

server qcs1 qcs1:9696 check inter 2000 rise 2 fall 5

server qcs2 qcs2:9696 check inter 2000 rise 2 fall 5

server qcs3 qcs3:9696 check inter 2000 rise 2 fall 5

6. qcs2,qcs3 서버에 deploy 한다.

위의 환경파일을 scp로 복사한 후 서버 명을 바꿔준다.

7. neutron 서비스 재시작

# cd sbin/neutron ; ./neutron\_service.sh restart

* Nova

https://docs.openstack.org/nova/wallaby/install/

1. DB 작업

root@qcs1:~/sbin# mysql -u root < nova.sql

$openstack user create --domain default --password-prompt nova

$openstack role add --project service --user nova admin

$ openstack service create --name nova \

--description "OpenStack Compute" compute

$ openstack endpoint create --region RegionOne \

compute public [http://controller:8774/v2.1](http://deepthinqcloud:8774/v2.1)

$openstack endpoint create --region RegionOne \

compute internal <http://controller:8774/v2.1>

$openstack endpoint create --region RegionOne \

compute admin <http://controller:8774/v2.1>

$ openstack user create --domain default --password-prompt placement

$ openstack role add --project service --user placement admin

$ openstack service create --name placement --description "Placement API" placement

$ openstack endpoint create --region RegionOne placement public [http://controller:8778](http://deepthinqcloud:8778)

$ openstack endpoint create --region RegionOne placement internal <http://controller:8778>

$ openstack endpoint create --region RegionOne placement admin <http://controller:8778>

1. Install Package

Controller Node : # apt install nova-api nova-conductor nova-console nova-consoleauth \

nova-novncproxy nova-scheduler nova-placement-api

Compute Node: # apt install nova-compute

1. 환경설정 : /etc/nova/nova.conf

# mkdir –p /data/nova/instances

# chown nova:nova -R /data/nova

# su -s /bin/sh -c "nova-manage api\_db sync" nova

# su -s /bin/sh -c "nova-manage cell\_v2 map\_cell0" nova

# su -s /bin/sh -c "nova-manage cell\_v2 create\_cell --name=cell1 --verbose" nova

# su -s /bin/sh -c "nova-manage db sync" nova

# su -s /bin/sh -c "nova-manage cell\_v2 list\_cells" nova

1. Compute Node 추가 및 환경설정

# apt install nova-compute

# openstack compute service list --service nova-compute // nova compute node 확인

# su -s /bin/sh -c "nova-manage cell\_v2 discover\_hosts --verbose" nova // nova compute node 찾기

- /etc/apparmor.d/usr.lib.libvirt.virt-aa-helper 수정

usr.lib.libvirt.virt-aa-helper: /data/nova/images/\*\* r,

usr.lib.libvirt.virt-aa-helper: /data/nova/instances/\_base/\*\* r,

usr.lib.libvirt.virt-aa-helper: /data/nova/instances/snapshots/\*\* r,

# service apparmor restart

- /etc/nova/nova.conf

instances\_path = /data/nova/instances

* Glance

https://docs.openstack.org/glance/wallaby/install/install-ubuntu.html

1. DB 작업

root@qcs1:~/sbin/glance# mysql -u root -p < glance.sql

root@qcs1 # cd ; . admin-openrc

root@qcs1:~# openstack user create --domain default --password-prompt glance

root@qcs1:~# openstack user delete --domain default glance (user 삭제시 사용)

root@qcs1:~# openstack user create --domain default --project service --password bns2000 glance

$ openstack role add --project service --user glance admin

$ openstack service create --name glance \

--description "OpenStack Image" image

$ openstack endpoint create --region RegionOne \

image public [http://controller:9292](http://deepthinqcloud:9292)

$ openstack endpoint list

$ openstack endpoint delete <endpoint-id>

root@qcs1:~# openstack endpoint create --region RegionOne image internal <http://controller:9292>

root@qcs1:~# openstack endpoint create --region RegionOne image admin <http://controller:9292>

2. Install Package

# cmd apt-get install -y glance-api

1. 환경설정

/etc/glance/glance-api.conf

/etc/glance/glance-registry.conf

#mkdir –p /data/glance/images

#chown –R glance:glance /data/glance

# su -s /bin/sh -c "glance-manage db\_sync" glance

1. Haproxy 에 서비스 등록한다.

listen glance\_api\_cluster

bind controller:9292

balance source

option tcpka

option httpchk

option tcplog

server qcs1 qcs1:9292 check inter 2000 rise 2 fall 5

server qcs2 qcs2:9292 backup check inter 2000 rise 2 fall 5

server qcs3 qcs3:9292 backup check inter 2000 rise 2 fall 5

listen glance\_registry\_cluster

bind controller:9191

bind controller:9191

balance source

option tcpka

option tcplog

server qcs1 qcs1:9191 check inter 2000 rise 2 fall 5

server qcs2 qcs2:9191 backup check inter 2000 rise 2 fall 5

server qcs3 qcs3:9191 backup check inter 2000 rise 2 fall 5

* Cinder

[https://docs.openstack.org/cinder/wallaby/install/](https://docs.openstack.org/cinder/queens/install/)

1. DB 작업

1)Database 생성

root@qcs1:~/sbin/cinder# mysql -u root -p < cinder.sql

2)cinder user/role/service/endpoint 생성

$ openstack user create --domain default --password-prompt cinder

$ openstack role add --project service --user cinder admin

$ openstack service create --name cinder \

--description "OpenStack Block Storage" volume

$ openstack service create --name cinderv2 \

--description "OpenStack Block Storage" volumev2

$ openstack service create --name cinderv3 \

--description "OpenStack Block Storage" volumev3

$ openstack endpoint create --region RegionOne \

volume public [http://controller:8776/v1/%\(project\_id\)s](http://controller:8776/v1/%25\(project_id\)s)

$ openstack endpoint create --region RegionOne \

volume internal [http://controller:8776/v1/%\(project\_id\)](http://controller:8776/v1/%25\(project_id\))

$ openstack endpoint create --region RegionOne \

volume admin [http://controller:8776/v1/%\(project\_id\)s](http://controller:8776/v1/%25\(project_id\)s)

$ openstack endpoint create --region RegionOne \

volumev2 public [http://controller:8776/v2/%\(project\_id\)s](http://controller:8776/v2/%25\(project_id\)s)

$ openstack endpoint create --region RegionOne \

volumev2 internal [http://controller:8776/v2/%\(project\_id\)](http://controller:8776/v2/%25\(project_id\))

$ openstack endpoint create --region RegionOne \

volumev2 admin [http://controller:8776/v2/%\(project\_id\)s](http://controller:8776/v2/%25\(project_id\)s)

$ openstack endpoint create --region RegionOne \

volumev3 public [http://controller:8776/v3/%\(project\_id\)s](http://controller:8776/v3/%25\(project_id\)s)

$ openstack endpoint create --region RegionOne \

volumev3 internal [http://controller:8776/v3/%\(project\_id\)s](http://controller:8776/v3/%25\(project_id\)s)

$ openstack endpoint create --region RegionOne \

volumev3 admin http://controller:8776/v3/%\(project\_id\)s

3) cinder table 생성

# su -s /bin/sh -c "cinder-manage db sync" cinder

1. Package Install

# apt install lvm2 thin-provisioning-tools

# apt install cinder-volume

# apt install cinder-api

# apt install cinder-scheduler

1. 환경설정

/etc/cinder/cinder.conf

# service cinder-volume restart

- Horizon

URL : https://docs.openstack.org/horizon/wallaby/index.html

# apt install openstack-dashboard

환경파일 : /etc/openstack-dashboard/local\_settings.py

/etc/apache2/conf-available/openstack-dashboard.conf

3. VM 설치 by Ansible & Heat or Teraform – 추후 업데이트 예정

4. CI / CD 설치 ( git / gerrit / Jenkins / ansible / bkubernetes ) – 추후 업데이트 예정

1)Git

2)Gerrit

3)Jenkins

4)Ansible

5)kubernetes

-- 끝 --